Neural Controller for Nonholonomic Mobile Robot System Based on Position and Orientation Predictor

Abeer Fadhil Shimal[1]

Received on: 30/9/2010
Accepted on :9/3/2011

Abstract

This paper proposes a neural controller to guide a nonholonomic mobile robot during trajectory tracking. The structure of the controller used consists of two models that describe the kinematical mobile robot system. These models are modified Elman neural networks (MENN) and feed forward multi-layer perceptron (MLP). The modified Elman neural networks model is trained with two stages; off-line and on-line, in order to guarantee that the outputs of the model accurately represent the actual outputs of the mobile robot system. The neural model, after being trained, acts as the position and orientation predictor. The feed forward multi-layer perceptron neural networks controller is trained on-line to find the inverse kinematical model, which controls the outputs of the mobile robot system. The general back propagation algorithm is used to learn the feed forward kinematics neural controller and the predictor. The results obtained from the conducted simulation show the effectiveness of the proposed neural control algorithm. This is demonstrated by the minimized tracking error and the smoothness of the control signal obtained.

Keywords: Nonholonomic Mobile Robots, Neural Networks Controller, Trajectory Tracking.

الخلاصة

هذا البحث يقترح مسيطر عصبي لتوجيه الإنسان الآلي اللاشمولي خلال تتابع المسار. أن هيكلية المسيطر العصبي المستخدم يتألف من نموذجين يصفان النظام الحركي للإنسان الآلي النقال, وهذان النموذجان للشبكة العصبية هما (Modified Elamn Neural Networks) و (Multi-Layer Perceptron).
أن النموذج للشبكة العصبية (MENN) يتم تدريبها بمرحلتين Off-line و On-line و ذلك لضمان دقة نتائج الإخراج للنموذج مع الإخراج الفعلي لمنظومة الإنسان الآلي المتحرك. يعمل نموذج الشبكة العصبية بعد تدريبه كمتنبيء للموقع و الاتجاه.يتم تدريب المسيطر العصبي ذو التغذية الأمامية والمتعدد الطبقات (MLP) online لإيجاد النموذج المعكوس الحركي الذي يسيطر على مخرجات الإنسان الآلي.
تم استخدام خوارزمية الانتشار العكسي لتعليم الشبكة العصبية التنبؤي والمسيطر العصبي الأمامي الحركي .
ان نتائج المحاكاة تبيّن فعالية خوارزمية السيطرة العصبية المقترحة في التقليل من خطأ التتابع و نعومة أشارة. التحكم.

[1] Control & Systems Eng. Dept./ University of Technology, Baghdad, Iraq.

1. Introduction

Mobile robots have been used in many applications such as moving material between work stations. They can also be found in many areas such as industrial, medical environmental and even domestic machines. Research on mobile robots has mounted and attracted so much attention in recent years, since they are increasingly used in a wide range of applications [1, 2]. Several controllers were proposed for mobile robots with nonholonomic constraints. The traditional control methods for path tracking the mobile robot use linear feedback control or non-linear feedback control and the artificial intelligent controller that carried out by using neural networks or fuzzy inference [3]. There are other techniques of the controllers such as predictive control technique that is a very important area of research and in the field of mobile robotics predictive approaches to path tracking also seem to be very promising because the reference trajectory is known beforehand. In [4] model predictive trajectory tracking control applied to a mobile robot and used linearise tracking error dynamics to predict future system behaviour and a control law is derived from a quadratic cost function penalizing the system trucking error and the control effort.

A model predictive control algorithm developed for stabilizing a team of nonholonomic mobile robots navigating information within an obstacle-populated environment in order to avoid collisions and accomplish mission objectives is presented in [5]. In [6] a switched control algorithm to stabilize a car-like mobile robot which possesses the velocity level nonholonomic constraint and the control approach rests on splitting the system into several second-order subsystems and then stabilizing the system sequentially using finite-time controllers.

An adaptive trajectory-tracking controller based on the robot dynamics, and its stability property is proven using the Lyapunov theory is proposed in [7].

A trajectory tracking control for a nonholonomic mobile robot by the integration of a kinematics controller and neural dynamic controller based on the sliding mode theory is presented in [8].

In [9] an adaptive controller of nonlinear PID based analogue neural networks is developed for the velocity and orientation tracking control of a nonholonomic mobile robot. In [10] a variable structure control algorithm is proposed to study the trajectory tracking control based on the kinematics model of a 2-wheel differentially driven mobile robot by using of the back stepping method and virtual feedback parameter with the sigmoid function. The trajectory tracking controllers are designed by pole-assignment approach for mobile robot model is presented in [11]. The model of the mobile robot obtained from the combination of kinematical and robust H_∞ dynamical tracking controller used to design the kinematical tracking controller by applying the Lyapunov stability theorem is proposed in [12].

The design of a dynamic Petri recurrent fuzzy neural network (DPRFNN) structure is applied to the path-trajectory control of a nonholonomic mobile robot for verifying its validity and convergence of the path-tracking errors based on a discrete-type Lyapunov function is presented in [13]. In [14] two novel dual adaptive neural control schemes are proposed for dynamic control of nonholonomic mobile robots. The first scheme is based on Gaussian radial basis function (GaRBF) ANNs and the second on sigmoidal multilayer perceptron (MLP) ANNs, where ANNs are employed for real-time approximation of the robot's nonlinear dynamic functions assumed to be unknown.

A novel linear interpolation based methodology to design control algorithms for the trajectory tracking of mobile robotic systems is presented in [15] assuming that the evolution of the system can be approximated by a linear interpolation in each sampling time and knowing the desired state. A value for control action needed to force the system to go from its current state to a desired one can be obtained.

The remainder of the paper is organized as follows. In section two there is a description of the kinematics model of the nonholonomic mobile robot. Section three the proposed neural controller is derived. The simulation results for the neural control obtained are presented in section four, and the conclusion is given in section five.

2. The Kinematics Model of Mobile Robot

The nonholonomic mobile robot shown in figure (1) consists of a cart with two driving wheels mounted on the same axis and an omni-directional castor in the front of the cart, which carries the mechanical structure and keeps the platform more stable.

Two independent servo DC motors are the actuators of left and right wheels for motion and orientation. The two wheels have the same radius denoted by r and L as the distance between the two wheels. The center of mass of the mobile robot is located at point c which is the center of wheel axle.

The pose of the mobile robot in the global coordinate frame $[O, X, Y]$ and the pose vector in the surface is defined as:

$$q = (x, y, \theta)^T \qquad (1)$$

Where x and y are the coordinates of point c, and θ is the robotic orientation angle measured from X-axis and these three generalized coordinates can describe the configuration of the mobile robot.

The mobile robot's motion can be determined by the two wheel velocities, the velocity of the left wheel V_L, the velocity of the right wheel V_R, the linear velocity V_l and the angular velocity V_w. The linear and angular velocities can be described in terms of the left and right velocities as follows [16]:

$$V_l(t) = \frac{V_R(t) + V_L(t)}{2} \qquad (2)$$

$$V_w(t) = \frac{V_R(t) - V_L(t)}{L} \qquad (3)$$

Then the mobile robot kinematics can be described by:

$$\begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} 0.5\cos\theta(t) & 0.5\cos\theta(t) \\ 0.5\sin\theta(t) & 0.5\sin\theta(t) \\ 1/L & -1/L \end{bmatrix} \begin{bmatrix} V_R(t) \\ V_L(t) \end{bmatrix} \qquad (4)$$

It is assumed that the mobile robot wheels are ideally installed in such a way that they have ideal rolling without skidding. Therefore, the mobile robot can be steered to any position in a free workspace [17]. However, the wheels of the mobile robot cannot move sideways. Therefore, the freedom of the motion is limited because no lateral slippage is allowed and the wheels must not slide orthogonally to the wheel plane and the velocity of the point c of the mobile robot must be in the direction of the axis of symmetry, the x-axis which is referred to as the nonholonomic constraint [18], as shown in Equation (5):

$$-\dot{x}(t)\sin\theta(t) + \dot{y}(t)\cos\theta(t) = 0 \qquad (5)$$

To verify the controllability of the nonlinear MIMO system in equation (4), the accessibility rank condition is globally satisfied, and this implies controllability.

From equation (4), the mobile robot kinematics can be described by:

$$[\dot{q}] = [f]V_R(t) + [g]V_L(t) \tag{6}$$

Where f and g can be defined by two vectors with components as:

$$f = \begin{bmatrix} 0.5\cos\theta(t) \\ 0.5\sin\theta(t) \\ 1/L \end{bmatrix} \quad \text{and} \quad g = \begin{bmatrix} 0.5\cos\theta(t) \\ 0.5\sin\theta(t) \\ -1/L \end{bmatrix} \tag{7}$$

Using the Jacobi-Lie-Bracket of f and g, $[f,g]$ is found [19].

$$[f,g] = \begin{bmatrix} [f,g]^1 \\ [f,g]^2 \\ [f,g]^3 \end{bmatrix} = \begin{bmatrix} \frac{-1}{L}\sin\theta(t) \\ \frac{1}{L}\cos\theta(t) \\ 0 \end{bmatrix} \tag{8}$$

$$\text{rank}\{f,g,[f,g]\} = \text{rank}\begin{bmatrix} 0.5\cos\theta(t) & 0.5\cos\theta(t) & \frac{-1}{L}\sin\theta(t) \\ 0.5\sin\theta(t) & 0.5\sin\theta(t) & \frac{1}{L}\cos\theta(t) \\ 1/L & -1/L & 0 \end{bmatrix} \tag{9}$$

The determinant of the matrix in equation (9) is equal to $(1/L^2) \neq 0$, and then the full rank of the matrix is equal to 3. Therefore, the system in equation (4) is controllable.

3. Neural Control Methodology

The control of a nonlinear MIMO mobile robot system is considered in this section. The approach used to control the mobile robot depends on the information available about the system and the control objectives. The information of the unknown nonlinear system can be known by the input-output data only and the system is considered as (modified Elman recurrent neural networks model). The first step in the procedure of building the control structure is the identification of the kinematical mobile robot from the input-output data, and then a feed forward kinematical neural networks controller is used because the inverse of the kinematical mobile robot depends on feed forward multi-layer perceptron neural networks.

The integrated control structure that consists of the inverse of the kinematical system, the position and orientation neural networks predictor brings together the advantages of the inverse method with the robustness of feedback in order to achieve good tracking of the reference trajectory and to use minimum control effort.

The proposed structure of the neural controller can be given in the form of the block diagram shown in figure (2). The neural controller applied to the mobile robot system consists of:

1- Position and Orientation Neural Networks Predictor.

2- Feed forward Kinematics Neural Networks Controller.

In the following sections, each part of the proposed controller will be explained in detail.

3.1 Position and Orientation Neural Networks Predictor

This section focuses on nonlinear MIMO system identification of kinematics mobile robot (position and orientation) using the modified Elman recurrent neural network model to construct the position and orientation neural networks predictor as shown in figure (3) with the nodes of input layer, context layer, hidden layer and output layer. The network uses two models configuration series-parallel and parallel identification structure, which is trained using dynamic back-propagation algorithm.

The structure shown in figure 3 is based on the following equations:

$$h(k) = F\{VH\overline{G}(k), VC\overline{h^o}(k), bias\overline{Vb}\} \tag{10}$$

$$O(k) = (Wh(k), bias\overline{Wb})$$

(11)

Where *VH*, *VC* and *W* are weight matrices, $\overline{Vb}$ and $\overline{Wb}$ are weight vectors and *F* is a non-linear vector function. The multi-layered modified Elman neural network is shown in Figure (3). It is composed of many interconnected processing units called neurons or nodes. The network weights are denoted as follows:

VH : Weight matrix of the hidden layers.
VC : Weight matrix of the context layers.
$\overline{Vb}$: Weight vector of the hidden layers.
W : Weight matrix of the output layer.
$\overline{Wb}$: Weight vector of the output layer.
L : Denotes linear node.
H : Denotes nonlinear node with sigmoidal function

In order to improve the network memory ability, self-connections (α fixed value) are introduced into the context units of the network in order to give these units a certain amount of inertia [20 & 21]. The introduction of self-connection in the context units increases the possibility of the Elman network to model high-order systems.

The output of the context unit in the modified Elman network is given by:

$$h_c^o(k) = \alpha h_c^o(k-1) + \beta h_c(k-1) \qquad (12)$$

Where $h_c^o(k)$ and $h_c(k)$ are the output of the context unit and the hidden unit respectively. α is the feedback gain of the self-connections and β is the connection weight from the hidden units (c^{th}) to the context units (c^{th}) at the context layer. The value of α and β are selected randomly between (0 and 1).

To explain these calculations, consider the general j^{th} neuron in the hidden layer and the inputs to this neuron consisting of an (i) – dimensional vector (i is the number of the input nodes). Each of the inputs has a weight *VH* and *VC* associated with it.

$\overline{Vb}$ is the weight vector for the bias input that is set equal to -1 to prevent the neurons quiescent. The first calculation within the neuron consists of calculating the weighted sum net_j of the inputs as [20]:

$$net_j = \sum_{i=1}^{nh} VH_{ji} \times G_i + \sum_{c=1}^{C} VC_{jc} \times h_c^o + bias \times Vb_j \qquad (13)$$

Where $j = c \,.\, nh = C$ Which is the number of the hidden nodes and context nodes, and $\overline{G}$ is the input vector

The output of the predictor is the modelling pose vector in the surface and is defined as:

$q_m = (x_m, y_m, \theta_m)^T$, where x_m and y_m are the modelling coordinates and θ_m is the modelling orientation angle.

A learning algorithm is used to adjust the weights of dynamical recurrent neural network. Dynamic back propagation algorithm is used to train the Elman network. The sum of the square of the differences between the desired outputs $q = (x, y, \theta)^T$ and neural network predictor outputs $q_m = (x_m, y_m, \theta_m)^T$ is given by equation (14).

$$E = \frac{1}{2}\sum_{i=1}^{np}((x - x_m)^2 + (y - y_m)^2 + (\theta - \theta_m)^2) \qquad (14)$$

Where *np* is the number of patterns.

The connection matrix between hidden layer and output layer is W_{kj}

$$\Delta W_{kj}(k+1) = -\eta \frac{\partial E}{\partial W_{kj}} \qquad (15)$$

$$\frac{\partial E}{\partial W_{kj}} = \frac{\partial E}{\partial q_m(k+1)} \frac{\partial q_m(k+1)}{\partial o_k} \frac{\partial o_k}{\partial net_k} \frac{\partial net_k}{\partial W_{kj}} \qquad (16)$$

$$\Delta W_{kj}(k+1) = \eta \times h_j \times e_k \qquad (17)$$

$$W_{kj}(k+1) = W_{kj}(k) + \Delta W_{kj}(k+1)$$

(18)

The connection matrix between input layer and hidden layer is VH_{ji}

$$\Delta VH_{ji}(k+1) = -\eta \frac{\partial E}{\partial VH_{ji}} \quad (19)$$

$$\frac{\partial E}{\partial VH_{ji}} = \frac{\partial E}{\partial q_m(k+1)} \frac{\partial q_m(k+1)}{\partial o_k} \frac{\partial o_k}{\partial net_k} \frac{\partial net_k}{\partial h_j} \frac{\partial h_j}{\partial net_j} \frac{\partial net_j}{\partial VH_{ji}} \quad (20)$$

$$\Delta VH_{ji}(k+1) = \eta \times f(net_j)' \times G_i \sum_{k=1}^{K} e_k W_{kj} \quad (21)$$

$$VH_{ji}(k+1) = VH_{ji}(k) + \Delta VH_{ji}(k+1) \quad (22)$$

The connection matrix between context layer and hidden layer is VC_{ji}

$$\Delta VC_{jc}(k+1) = -\eta \frac{\partial E}{\partial VC_{jc}} \quad (23)$$

$$\frac{\partial E}{\partial VC_{jc}} = \frac{\partial E}{\partial q_m(k+1)} \frac{\partial q_m(k+1)}{\partial o_k} \frac{\partial o_k}{\partial net_k} \bullet \frac{\partial net_k}{\partial h_j} \frac{\partial h_j}{\partial net_c} \frac{\partial net_c}{\partial VC_{jc}} \quad (24)$$

$$\Delta VC_{jc}(k+1) = \eta \times f(net_j)' \times h_c^o \sum_{k=1}^{K} e_k W_{kj} \quad (25)$$

$$VC_{jc}(k+1) = VC_{jc}(k) + \Delta VC_{jc}(k+1) \quad (26)$$

3.2 Feed forward Kinematics Neural Controller

The feed forward kinematics neural controller is of prime importance in the structure of the proposed controller because of its necessity to track error in the transients and minimize steady-state error to zero. This means that the functions of the (FFKNC) $uff_1(k)$ and $uff_2(k)$ are to use the outputs of the mobile robot as the velocity of the left wheel V_L and the velocity of the right wheel V_R. Hence the (FFKNC) is supposed to learn the inverse kinematics of the mobile robot system on-line to calculate mobile robot inputs drive to keep the robot on a desired trajectory if there are any disturbances or initial state errors.

To achieve this, a multi-layer perceptron model is used as shown in figure (4). The system is composed of many interconnected processing units called neurons or nodes. The network notations are as follows:

$Vcont$: Weight matrix of the hidden layers.

$\overline{Vbc}$: Weight vector of the hidden layers.

$Wcont$: Weight matrix of the output layer.

$\overline{Wbc}$: Weight vector of the output layer.

To explain these calculations, consider the general a^{th} neuron in the hidden layer shown in figure (4). The inputs to this neuron consist of a (n) – dimensional vector and (n is the number of the input nodes). Each of the inputs has a weight $Vcont$ associated with it. The first calculation within the neuron consists of calculating the weighted sum $netc_a$ of the inputs as [22]:

$$netc_a = \sum_{a=1}^{nhc} Vcont_{an} \times Z_n + bias \times Vbc_a \quad (27)$$

Where nhc is the number of the hidden nodes.

Next, the output of the neuron h_a is calculated as the continuous sigmoid function of the $netc_a$ as:

$$hc_a = H(netc_a) \quad (28)$$

$$H(netc_a) = \frac{2}{1+e^{-netc_a}} - 1 \quad (29)$$

Once the outputs of the hidden layer have been calculated, they are passed to the output layer.

In the output layer, two linear neurons are used to calculate the weighted sum (*netco*) of its inputs (the output of the hidden layer as in equation (30).

$$netco_b = \sum_{a=1}^{nhc} Wcont_{ba} \times hc_a + bias \times Wbc_b \tag{30}$$

Where $Wcont_{ba}$ are the weights between the hidden neuron hc_a and the output neurons. Then, passes the sum ($netco_b$) through a linear activation function of slope 1 (another slope can be used to scale the output) as:

$$Oc_b = L(netco_b) \tag{31}$$

These outputs of the feed forward kinematics neural networks controller are $uff_1(k)$ and $uff_2(k)$.

The training of the inverse kinematics is done on-line. It is dependent on the position and orientation neural network predictor which is used to find the mobile robot Jacobian through the neural predictor model. This approach is currently considered one of the better approaches that can be followed to overcome the lack of initial knowledge. The learning algorithm is utilised to adjust the weights of the feed forward kinematics neural networks controller. The dynamic back propagation algorithm is used to realize the training of multi-layer perceptron neural network, the sum of the square of the differences between the desired posture $q_r = (x_r, y_r, \theta_r)^T$ and neural network posture $q_m = (x_m, y_m, \theta_m)^T$ is given by:

$$Ec = \frac{1}{2}\sum_{i=1}^{npc}((x_r - x_m)^2 + (y_r - y_m)^2 + (\theta_r - \theta_m)^2) \tag{32}$$

Where *npc* is number of patterns.

The connection matrix between the hidden layer and the output layer is $Wcont_{ba}$ and is defined as follows:

$$\Delta Wcont_{ba}(k+1) = -\eta \frac{\partial Ec}{\partial Wcont_{ba}} \tag{33}$$

$$\frac{\partial Ec}{\partial Wcont_{ba}} = \frac{\partial Ec}{\partial q_m(k+1)} \frac{\partial q_m(k+1)}{\partial uff_b(k)} \bullet \frac{\partial uff_b(k)}{\partial oc_b} \frac{\partial oc_b}{\partial netc_b} \frac{\partial netc_b}{\partial Wcont_{ba}} \tag{34}$$

$$\frac{\partial Ec}{\partial Wcont_{ba}} = \frac{\partial Ec}{\partial q_m(k+1)} \frac{\partial q_m(k+1)}{\partial uff_b(k)} \frac{\partial uff_b(k)}{\partial oc_b} f'(netc_b)hc_a \tag{35}$$

$$\frac{\partial Ec}{\partial q_m(k+1)} = \frac{\partial \frac{1}{2}\sum((x_r - x_m)^2 + (y_r - y_m)^2 + (\theta_r - \theta_m)^2)}{\partial q_m(k+1)} \tag{36}$$

This is done in the local coordinates with respect to the body of the mobile robot, which are the same outputs of the position and the orientation neural networks predictor, the configuration error can be represented by using this transformation matrix:

$$\begin{bmatrix} ex_m \\ ey_m \\ e\theta_m \end{bmatrix} = \begin{bmatrix} \cos\theta_m & \sin\theta_m & 0 \\ -\sin\theta_m & \cos\theta_m & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x_m \\ y_r - y_m \\ \theta_r - \theta_m \end{bmatrix} \tag{37}$$

Where x_r, y_r and θ_r are the reference position and orientation of the mobile robot respectively.

$$Jacobian = \frac{\partial q_m(k+1)}{\partial uff_b(k)} \tag{38}$$

Where the outputs of the predictor are $q_m = (x_m, y_m, \theta_m)^T$.

$$\frac{\partial q_m(k+1)}{\partial uff_b(k)} = \frac{\partial q_m(k+1)}{\partial o_k(k)} \frac{\partial o_k(k)}{\partial net_k} \frac{\partial net_k}{\partial h_j} \frac{\partial h_j}{\partial net_j} \frac{\partial net_j}{\partial uff_b(k)} \tag{39}$$

The linear activation function in the outputs layers are going to be presented as follows:

$$\frac{\partial q_m(k+1)}{\partial uff_b(k)} = \frac{\partial net_k}{\partial h_j} \frac{\partial h_j}{\partial net_j} \frac{\partial net_j}{\partial uff_b(k)} \tag{40}$$

And the nonlinear activation function in the hidden layers is:

$$\frac{\partial q_m(k+1)}{\partial uff_b(k)} = \sum_{j=1}^{nh} f(net_j)' VH_{jb} \sum_{k=1}^{K} W_{kj} \tag{41}$$

Then substituting equations (41 and 36) into equation (35) will result in $\Delta Wcont_{ba}(k+1)$ as follows:

$$\Delta Wcont_{ba}(k+1) = \eta hc_a \times \sum_{j=1}^{nh} f(net_j)' VH_{jb} ((ex_m(k+1)W_{1j}) + (ey_m(k+1)W_{2j}) + (e\theta_m(k+1)W_{3j})) \tag{42}$$

$$Wcont_{ba}(k+1) = Wcont_{ba}(k) + \Delta Wcont_{ba}(k+1) \tag{43}$$

The connection matrix between input layer and hidden layer is designated $Vcont_{an}$

$$\Delta Vcont_{an}(k+1) = -\eta \frac{\partial Ec}{\partial Vcont_{an}} \tag{44}$$

$$\frac{\partial Ec}{\partial Vcont_{an}} = \frac{\partial Ec}{\partial uff_b(k)} \times \frac{\partial uff_b(k)}{\partial oc_b} \times \frac{\partial oc_b}{\partial netc_b} \times \frac{\partial netc_b}{\partial hc_a} \times \frac{\partial hc_a}{\partial netc_a} \times \frac{\partial netc_a}{\partial Vcont_{an}} \tag{45}$$

$$\frac{\partial Ec}{\partial Vcont_{an}} = \frac{\partial Ec}{\partial q_m(k+1)} \frac{\partial q_m(k+1)}{\partial uff_b(k)} \frac{\partial uff_b(k)}{\partial oc_b} \times \frac{\partial oc_b}{\partial netc_b} \times \frac{\partial netc_b}{\partial hc_a} \times \frac{\partial hc_a}{\partial netc_a} \times \frac{\partial netc_a}{\partial Vcont_{an}} \tag{46}$$

$$\frac{\partial Ec}{\partial Vcont_{an}} = \frac{\partial Ec}{\partial q_m(k+1)} \frac{\partial q_m(k+1)}{\partial uff_b(k)} \bullet \sum_{b=1}^{B} Wcont_{ba} \times f(netc_a)' \times Z_n \tag{47}$$

Substituting equations (41 and 36) into equation (47) will give $\Delta Vcont_{an}(k+1)$ in the following form:

$$\Delta Vcont_{an}(k+1) = \eta Z_n f(netc_a)' \sum_{b=1}^{B} Wcont_{ba} \sum_{j=1}^{nh} f(net_j)' \sum_{i=1}^{I} VH_{ji} ((ex_m(k+1)W_{1j}) + (ey_m(k+1)W_{2j}) + (e\theta_m(k+1)W_{3j})) \tag{48}$$

The outputs of the feed forward kinematics neural networks controller are equal to two. Figure (3) shows that $uff1(k)$ are exciting nodes VH_{j1} and $uff2(k)$ are exciting nodes VH_{j2} then (B=I=2).

$$Vcont_{an}(k+1) = Vcont_{an}(k) + \Delta Vcont_{an}(k+1) \tag{49}$$

After the neural network has learned the inverse kinematics, $uff_1(k)$ and $uff_2(k)$ are considered as the control action required to keep the output of the mobile robot at the reference trajectory, hence the velocity of the left wheel $V_L(k)$ and the velocity of the right wheel $V_R(k)$ will be known.

4. Simulation Results

The proposed neural controller is verified using computer simulation. The simulation program was written using C++ language. The simulation is carried out by tracking a desired position (x, y) and orientation angle (θ) with a lemniscates trajectory in the tracking control of the nonholonomic mobile robot for which the kinematics model described in section 2 is used. The model parameter values of the nonholonomic wheeled mobile robot are taken from the reference model [23] and have the following parameters: mass=9 kg, I=5 kgm^2, L=0.306 m and r=0.052 m.

The proposed controller is implemented as shown in Figure (2). The first stage of operation is to set the (position and orientation) neural network predictor. This task is carried out using identification technique based on series-parallel and parallel configuration with modified Elman recurrent neural networks model. The identification scheme of the nonlinear MIMO mobile robot system is needed for input-output training data pattern to provide enough information about the kinematics of the mobile robot model to be modelled. This can be achieved by injecting a sufficiently rich input signal to excite all process modes of interest while also ensuring that the training patterns adequately cover the specified operating region. A hybrid excitation signal has been used for the mobile robot model. As shown in figures (5a and b) the input signals (velocity of the right wheel V_R and velocity of the left wheel V_L) and (linear velocity V_l and angular velocity V_W respectively) consists of a random amplitude PRBS signal with small amplitude of higher frequency PRBS signal superimposed. The training set is generated by feeding a series of PRBS signals, with a sampling time equal to 0.1 second to the mobile robot model, and measure its corresponding outputs (position x, y and orientation θ). By using back propagation learning algorithm with the structure of the modified Elman recurrent neural network which is given by 5-6-6-3; then the node number of input layer, hidden layer, context layer and output layer will be 5, 6, 6, and 3 respectively as shown in figure (3).

A training set of 125 patterns has been used with learning rate equal to 0.1. After 1900 epochs, the predictor outputs of the neural network (position x, y and orientation θ) are approximated to the actual outputs of the mobile robots model trajectory as shown in figures (6a, b and c), and the objective cost function MSE became less than 0.0005 as shown in figure (7).

In order to guarantee the similarity of outputs of the neural network predictor with the actual outputs of the mobile robot model trajectory, it should be used in parallel configuration. At 3357 the same training set patterns has been achieved with an MSE figure of less than 7.3×10^{-6}. The neural network predictor (position and orientation) outputs and the mobile robot model trajectory are shown in figure (8).

The second stage of operation of the proposed controller is the feed forward kinematics neural network controller. It uses a multi-layer perceptron neural network 8-11-2 as shown in figure (4).

The desired lemniscates trajectory can be described by these equations:

$$x_r(t) = 0.75 + 0.75\times Sin(\frac{2\pi t}{10}) \quad (50)$$

$$y_r(t) = Sin(\frac{4\pi t}{10}) \quad (51)$$

$$\theta_r(t) = a\tan 2(\frac{(y_r(t) - y_r(t-1))/t}{(x_r(t) - x_r(t-1))/t}) \quad (52)$$

This trajectory has been learned (on-line) by the feed forward kinematics neural controller using a back propagation algorithm, and after 5329 epochs, the objective cost function MSE became less than 0.0031, which is required to find the suitable control action.

For simulation purposes, the desired trajectory is taken from equations (50 & 51) and the desired orientation angle is taken from equation (52). The mobile robot model starts from the initial position and orientation $q(0) = [0.75, -0.1, \pi/2]$ for initial conditions.

The demonstration mobile robot trajectory tracking obtained by the proposed neural controller is shown in figures (9a and b). From these figures, it can be noted that excellent position and orientation tracking performance has been obtained.

The simulation results demonstrate the effectiveness of the proposed neural controller and show that it has the ability to generate small smooth values of the control input velocities (right wheel and left wheel) without sharp spikes. The actions that are described in figures (10a and b) shows that smaller power is required to drive the DC motors of the mobile robot model. In addition to that, the convergence of the (position and orientation) trajectory error for the mobile robot model motion is very evident in figures (11-a, b, c).

5. Conclusion

The neural trajectory tracking control methodology for nonholonomic mobile robot is presented in this paper. The proposed controller consisted of two parts: position and orientation neural network predictor and feed forward kinematics neural network controller. The aim of the proposed control scheme is to minimize the tracking errors as well as the control effort. It uses two models of neural networks in the structure of the controller, multi-layer perceptron (MLP) and modified Elman neural network (MENN), which are trained off-line and on-line by using a back propagation algorithm with two configurations series-parallel and parallel to guarantee that the model outputs of the neural network match those of the mobile robot model outputs. From observing the simulation results, the proposed neural controller model has the capability to generate smooth and suitable velocity commands (V_R and V_L) without sharp spikes. This behaviour led to smoother drive of the mobile robot. The proposed controller has demonstrated the capability to track the desired trajectory and minimize the tracking error to zero approximation. Therefore, the proposed neural control methodology can be considered to be capable of effectively eradicating the tracking errors for the nonholonomic mobile robot model.

References

[1] K.Z. Karam and A. Albagul "Dynamic Modelling and Control Issues for a Three Wheeled Vehicle". Proc. 5th Int. Con. on Control Automation, Robotics and Vision, Singapore, 1998.

[2] G.Walsh, D. Tilbury, S. sastry, et al, "Stabilization of trajectories for systems with nonholonomic constraints", IEEE Tran. Automatic Control, Vol. 39, No. 1, (1994), pp. 216-222.

[3] F. Mnif and F. Touati "An Adaptive Control Scheme for Nonholonomic Mobile Robot with Parametric Uncertainty". International Journal of Advanced Robotic Systems, Vol. 2, No. 1 (2005), pp 59-63.

[4] G. Klancar and I. Skrjanc "Tracking Error Model-Based Predictive Control for Mobile Robots in Real Time". Robotics and Autonomous System, Vol. 55, (2007), pp. 460-469.

[5] F. Xie and R.Fierro "Stabilization of Nonholonomic Robot Formations: A First-State Contractive Model Predictive Control Approach". Journal of Computing and Information Technology- Cit 17, (2009), pp. 37-50.

[6] V. Sankaranarayanan and Arun D. Mahindrakar "Switched Control of a Nonholonomic Mobile Robot". Commun Nonlinear Sci Numer Simulat. Vol. 14, (2009), pp. 2319-2327.

[7] F.N. Martins, W.C. Celesta, R. Carelli, M. Sarcinelli–Filho, and T.F. Bastos–Filho "An Adaptive Dynamic Controller for Autonomous Mobile Robot Trajectory Tracking". Control Engineering Practice. Vol. 16, (2008), pp. 1354-1363.

[8] N.A. Martins, D.W. Bertol, E.R. De Pieri, E.B. Castelan and M.M. Dias "Neural Dynamic Control of a Nonholonomic Mobile Robot Incorporating the Actuator Dynamics". CIMCA 2008, IAWTIC 2008, and ISA 2008, IEEE Computer Society (2008).

[9] J. Ye "Adaptive Control of Nonlinear PID-Based Analogue Neural Network for a Nonholonomic Mobile Robot". Neurocompting. Vol. 71, (2008), pp. 1561-1565.

[10] H-X Zhang, G-J Dai and H Zeng "A Trajectory Tracking Control Method for Nonholonomic Mobile Robot". Proceedings of the 2007 International Conference on Wavelet Analysis and Pattern Recognition Beijing, China, 2-4 Nov. 2007.

[11] S. Sun "Designing Approach on Trajectory Tracking Control of Mobile Robot" Robotics and Computer Integrated Manufacturing. Vol. 21, (2005), pp. 81-85.

[12] C-K Hwang, B-S Chen and Y-T Chang "Combination of Kinematical and Robust Dynamical Controllers for Mobile Robotics Tracking Control: (I) Optimal H_∞ Control". IEEE International Conference on Control Applications, Vol. 4, (2004), pp. 1205-1210.

[13] R-J Wai and C-M Liu "Design of Dynamic Petri recurrent Fuzzy Neural Network and Its Application to Path-Tracking Control of Nonholonomic Mobile Robot". IEEE Transactions on Industrial Electronics, Vol. 56, No. 7, (2009), pp. 2667-2683.

[14] M.K. Bugeja, S.G. Fabri and L. Camilleri "Dual Adaptive Dynamic Control of Mobile Robots Using Neural Networks". IEEE Transactions on Systems, Man, and Cybernetics-Part B: CYBERNETICS, Vol. 39, No. 1, (2009), pp.129-141.

[15] G. Scaglia, A. Rosales, L. Quintero, V. Mut and R. Agarwal "A Linear Interpolation Based Controller Design for Trajectory Tracking of Mobile Robots". Control Engineering Practice Vol. 18, (2010), pp. 318-329.

[16] S. Han, B. Choi and J-M Lee "A Precise Curved Motion Planning for a Differential Driving Mobile Robot" Mechatronics Vol. 18, (2008) pp. 486-494.

[17] S. X. Yang, H. Yang, Q. Max and H. Meng "Neural Dynamics Based Full-State Tracking Control of a Mobile Robot". IEEE International Conference on Robotics & Automation, New Orleans, LA, April (2004).

[18] R. Siegwart and I. R. Nourbakhah "Introduction to Autonomous Mobile Robots", MIT Press (2004).

[19] A. M. Bloch "Nonholonomic Mechanics and Control", Springer-Verlag, New York (2003).

[20] D. T. Pham and L. Xing "Neural Networks for Identification, Prediction and Control", Springer, (1995).

[21] J. Xu, M. Zhang and J. Zhang "Kinematic Model Identification of Autonomous Mobile Robot Using Dynamical Recurrent Neural Networks". IEEE International Conference on Mechatronics & Automation, Niagara Falls, Canada, July (2005).

[22] J. M. Zurada, Introduction to Artificial Neural Systems. Jaico Publishing House, Pws Pub Co. (1992).

[23] T. Das, I.N. Kar and S. Chaudhury "Simple Neuron-Based Adaptive Controller for a Nonholonomic Mobile Robot Including Actuator Dynamics" Neurocomputing Vol. 69, (2006), pp. 2140-2151.

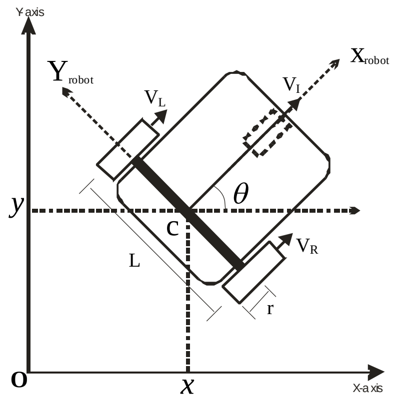

Figure 1: Schematic of the nonholonomic mobile robot.

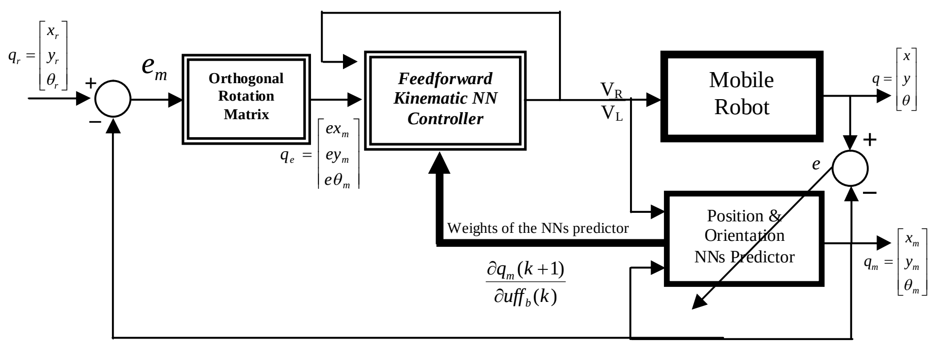

Figure 2: The proposed structure of the neural controller for the mobile robot system

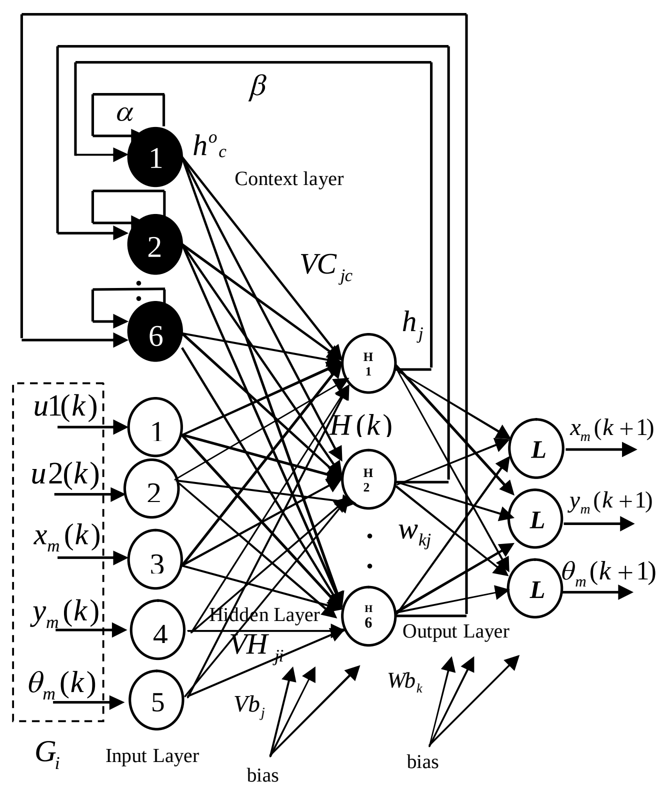

Figure (3): The Position and Orientation Neural Networks Predictor

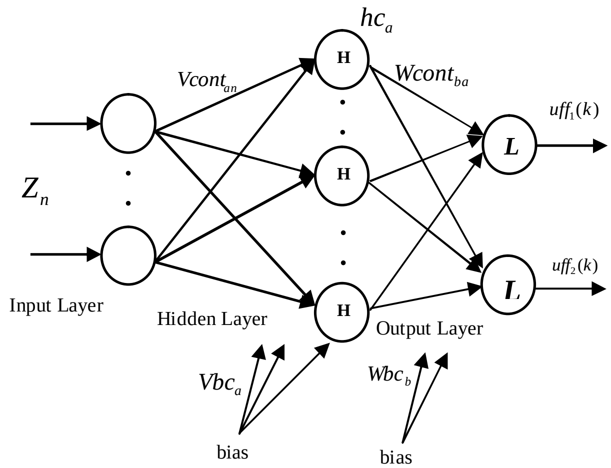

Figure 4: The Multi-Layer Perceptron Neural Networks.

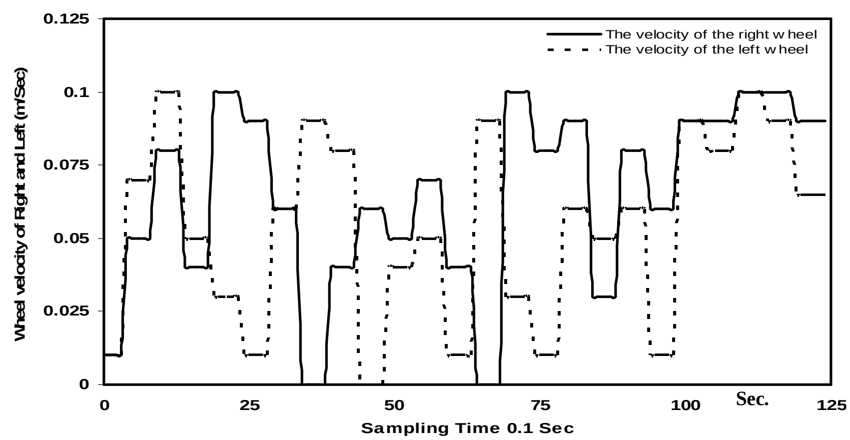

Figure (5-a): The PRBS input signals used to excite the mobile robot

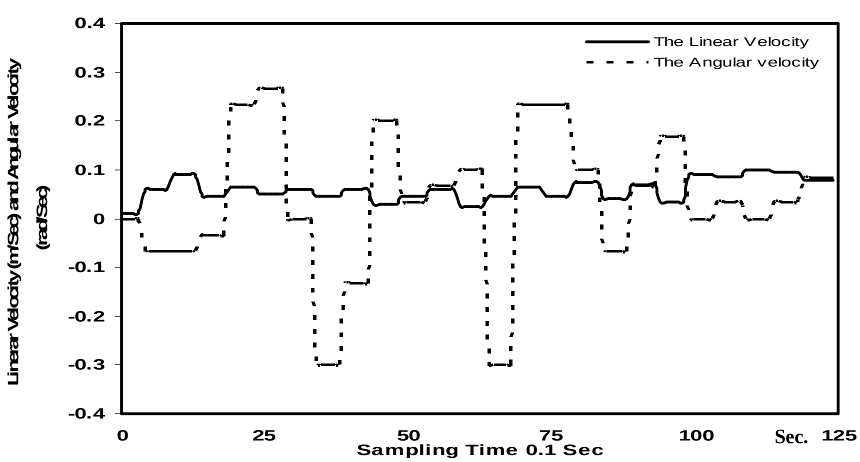

Figure (5-b): The linear and angular velocity inputs to the mobile robot

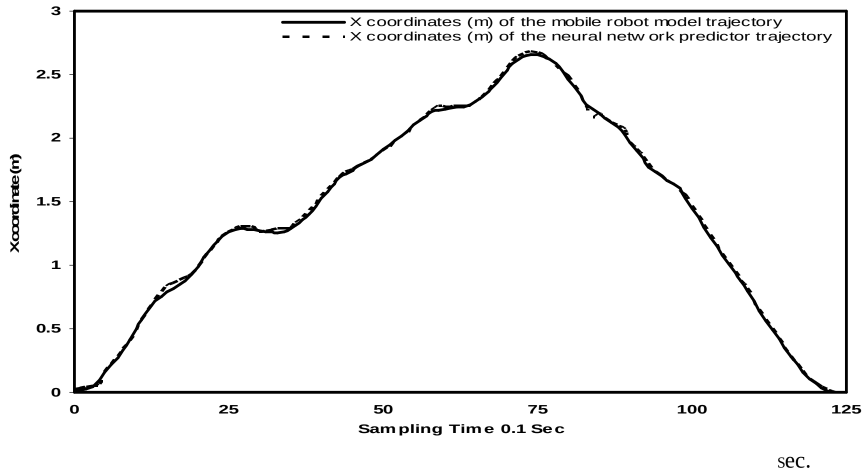

Figure (6-a): The response of the neural network predictor with the actual mobile robot model output in the X-coordinate.

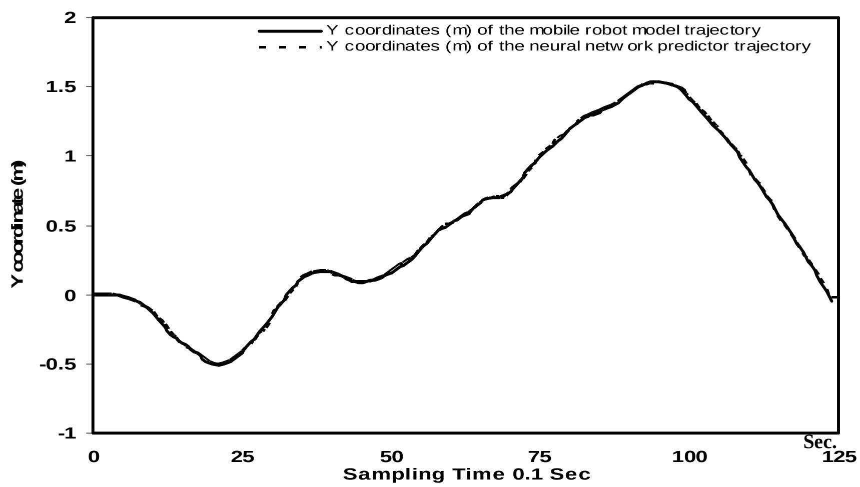

Figure (6-b): The response of the neural network predictor with the actual mobile robot model output in the Y-coordinat.

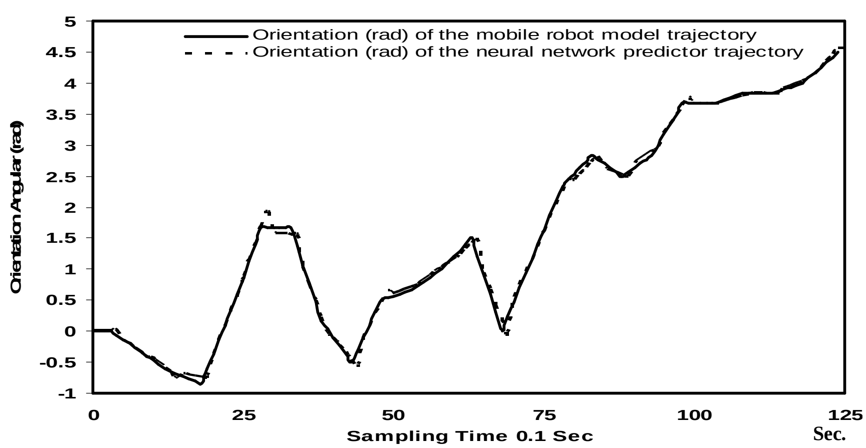

Figure (6-c): The response of the neural network predictor with the actual mobile robot model output in the θ orientation.

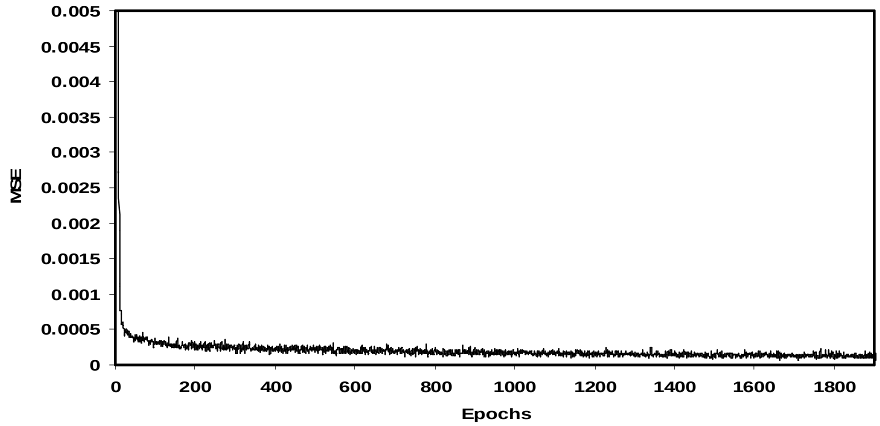

Figure (7): The objective cost function MSE

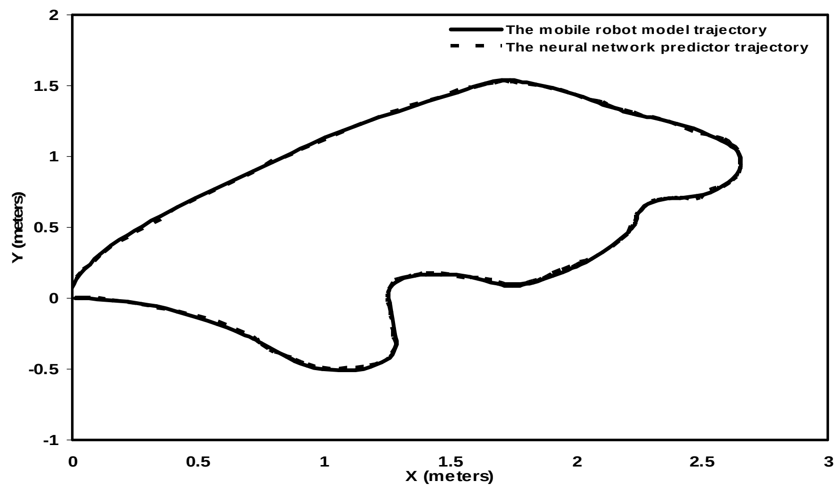

Figure (8): The response of the modified Elman neural network predictor with the actual mobile robot model outputs for the training patterns.

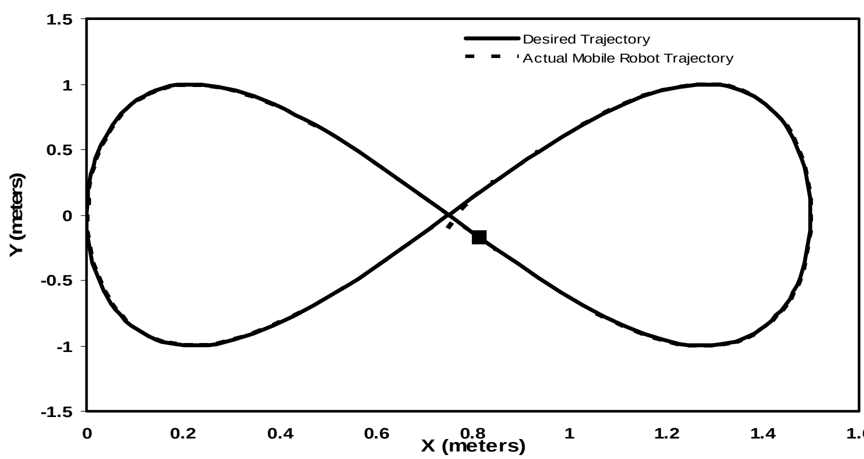

Figure (9-a): Mobile Robot actual position and desired trajectory.

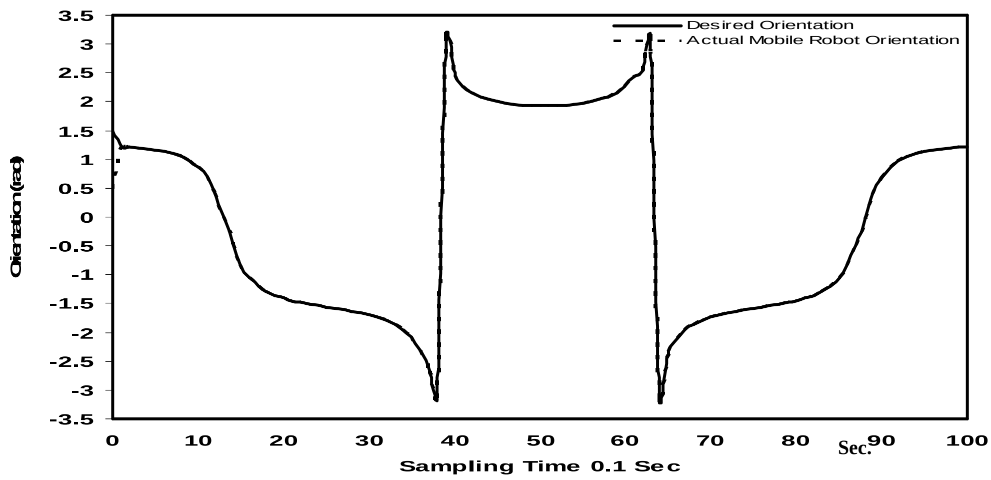

Figure (9-b): Mobile Robot actual orientation and desired trajectory.

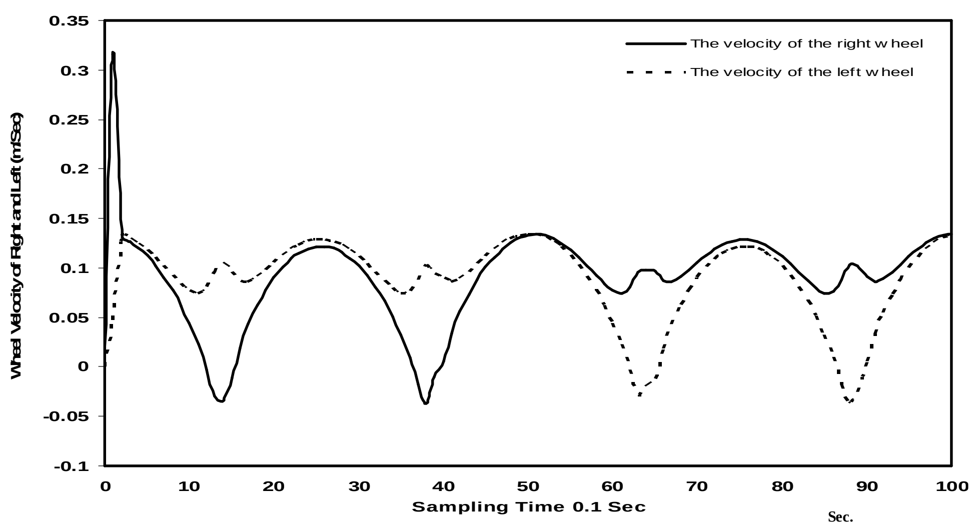

Figure (10-a): The velocity of the right and left wheel action for the neural controller.

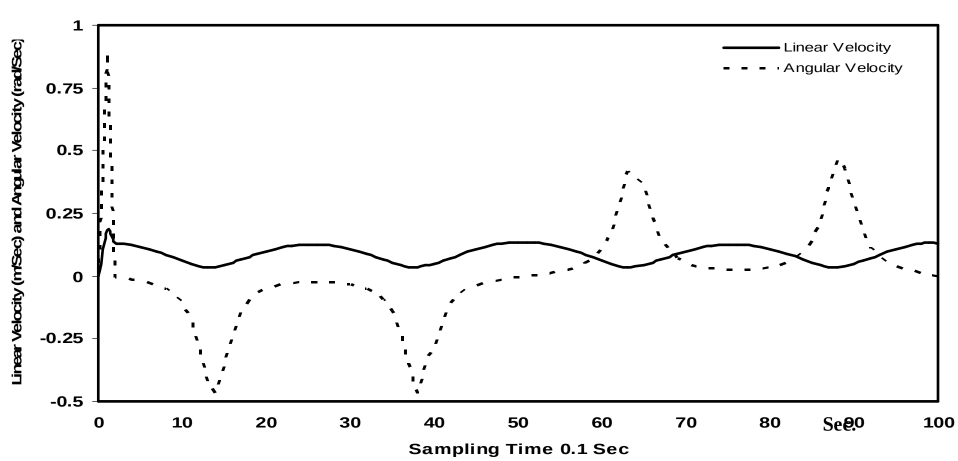

Figure (10-b): The linear velocity and angular velocity of the mobile robot.

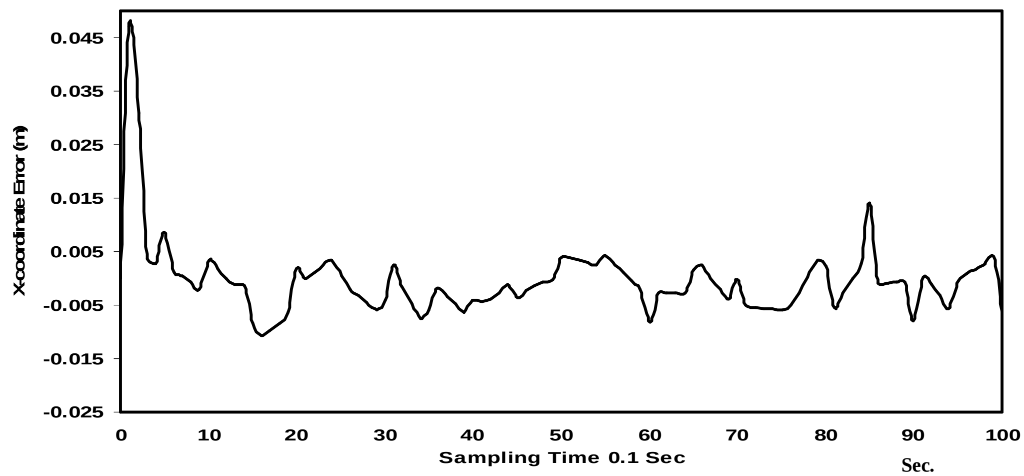

Figure (11-a): X-coordinate error.

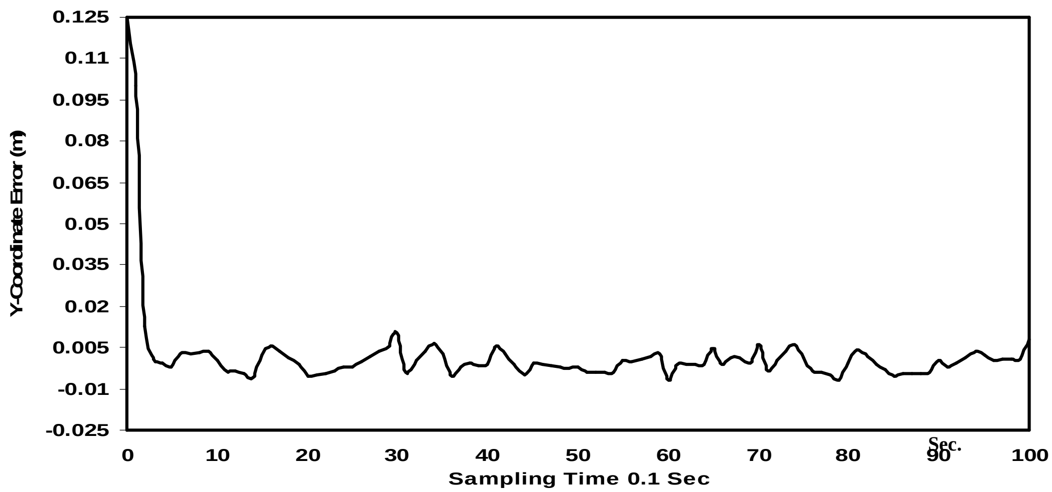

Figure (11-b): Y-coordinate error

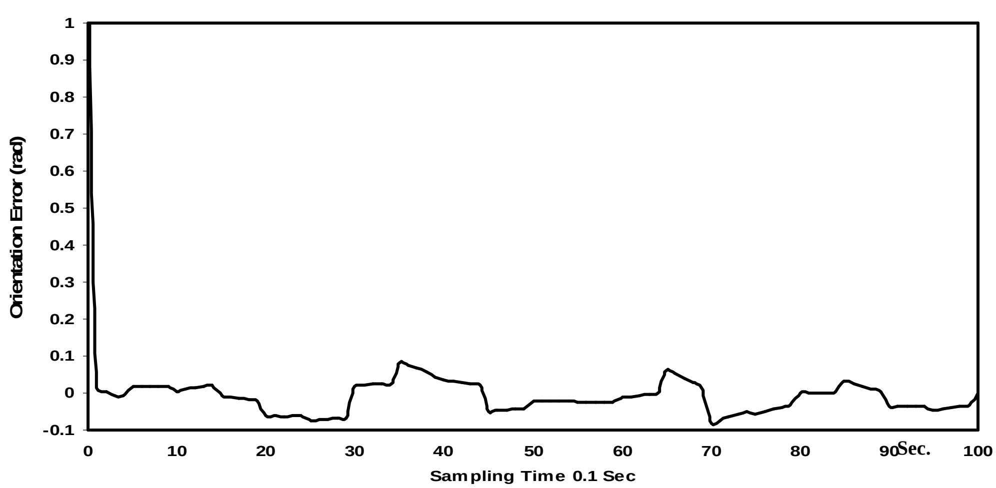

Figure (11-c): Orientation error.