Cognitive Neural Controller for Mobile Robot

Asst. Prof. Dr. Ahmed S. Al-Araji[1] and M. Sc. Khulood E. Dagher[2]

[1] Control and Systems Eng. Dept., University of Technology, Baghdad
[2] Computer Science Dept. Science College, Baghdad University, Baghdad
Baghdad – Iraq
E-mail: dr.ahmed_alaraji@uotechnology.edu.iq

Received: 26/5 /2014

Accepted: 21/1 /2015

Abstract – This paper proposes a cognitive neural controller to guide a nonholonomic mobile robot during continuous and non-continuous trajectory tracking and to navigate through static obstacles with collision-free and minimum tracking error. The structure of the controller consists of two layers; the first layer is a neural network topology that controls the mobile robot actuators in order to track a desired path based on back-stepping technique and posture identifier. The second layer of the controller is cognitive layer that collects information from the environment and plans the optimal path. In addition to this, it detects if there is any obstacle in the path so it can be avoided by re-planning the trajectory using particle swarm optimization (PSO) technique. The stability and convergence of control system are proved by using the Lyapunov criterion. Simulation results and experimental work show the effectiveness of the proposed cognitive neural control algorithm; this is demonstrated by minimizing tracking error and obtaining the smooth torque control signal, especially when the robot navigates through static obstacles with collision-free and the external disturbances applied.

.

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

1. Introduction

Over the last decade, the design and engineering of mobile robot systems acting autonomously in complex, dynamic and uncertain environments has remained a challenge. Such systems have to be able to perform multiple tasks, and therefore must integrate a variety of knowledge-intensive information processes on different levels of abstractions guaranteeing real-time execution, robustness, adaptability and scalability [1]. Cognitive control methodologies have been proven to be a source of inspiration and guidance to overcome current limitations in the controller for more complex and adaptive systems, and these methodologies have been utilizing mobile robot systems as demonstrators, serving as an important proof of the concept for cognitive models [2].

In recent years, several studies have been published for solving mobile robot control problems which can be classified into three categories: The first category is the position estimation control approach for navigation problems of the mobile robot on interactive motion planning in dynamics environments and obstacle motion estimation [3]. Since the working environment for mobile robots is unstructured and may change with time, the robot must use its on-board sensors to cope with dynamic environment changes while for proper motion planning such as environment configuration prediction and obstacle avoidance motion estimation it uses sensory information [4]. The second category for navigation problems of the mobile robot is path planning. The path is generated based on a prior map of the environment and used certain optimization artificial algorithms based on minimum time, distance and energy performance index for avoiding both static and moving obstacles as presented in [5]. The third category for the navigation problems of mobile robot is designing and implementing the motion control that mobile robot must execute the desired path accurately and minimize the tracking error.

Many control algorithms are proposed in the path-tracking control problems, such as Lyapunov-based nonlinear controllers [6], adaptive nonlinear neural PID controller [7], model-based predictive controllers [8], neural-fuzzy controllers [9], neural networks [10], the back-stepping method and feedback controller [11], etc.

The basic prediction problems in the path-tracking for mobile robot are still there waiting to be addressed; therefore, the fundamental essences of the motivation for this work are [10], [12] – [17]:

- To track the desired path with minimum tracking error.
- To generate optimal control action for mobile robot.
- To overcome un-modeled disturbances.
- To save the battery energy of the robot system.

The contributions of this paper obviously are summarized by the following points.

- Detecting the obstacle in the path and planning the optimum path for collision-free path between a starting and target location.
- Overcoming the challenge in modelling and identifying the position and orientation of the mobile robot for steps-ahead prediction.
- Developing the analytical derive for the cognitive neural control law with high computational

accuracy which based on Lyapunov criterion stability, back-stepping method and posture identifier. This drive is used to obtain the best torque control action quickly also it leads to minimize the tracking error of the mobile robot.

- Investigating the cognitive control methodology robustness and adaptation performance through adding undesirable boundary disturbances.
- Verifying experimentally the capability of the proposed controller in tracking different types of trajectories with continuous gradients (lemniscates) or non-continuous gradients (square) with obstacle by using real Boe-Bot mobile robot model.

Simulation results and experimental work show that the proposed cognitive controller is robust and effective in terms of minimum tracking error and in generating an optimal velocity control action quickly and re-planning the desired path to avoid the obstacle despite the presence of bounded external disturbances.

The remainder of the paper is organised as follows. Section two is a description of the kinematics and dynamics model of the nonholonomic wheeled mobile robot. In section three, the proposed cognitive neural controller is derived. Simulation results and experimental work of the proposed controller are presented in section four and the conclusions are drawn in section five.

2. Nonholonomic Mobile Robot Model

The schematic of the nonholonomic mobile robot shown in Fig. 1 consists of a cart with two driving wheels mounted on the same axis and an omni-directional castor in the front of cart. The castor carries the mechanical structure and keeps the platform more stable [7], [18] and [19]. Two independent analogous DC motors are the actuators of left and right wheels for motion and orientation.

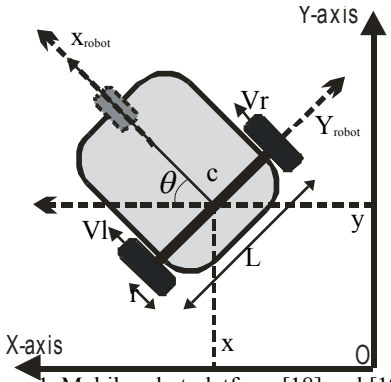

Figure 1 Mobile robot platform [18] and [19]

The two wheels have the same radius denoted by r, and L is the distance between the two wheels. The centre of mass of the mobile robot is located at point c, centre of axis of wheels. The pose of mobile robot in the global coordinate frame $[O,X,Y]$ and the pose vector in the surface is defined as:

$$q=(x,y,\theta)^T \tag{1}$$

x and y are coordinates of point c and θ is the robotic orientation angle measured with respect to the X-axis. These three generalized coordinates can describe the configuration of the mobile robot with kinematics constraints that they have ideal rolling without skidding [7], [18] and [19]. Therefore, the kinematics of the robot can be described as:

$$\dot{q}=\begin{bmatrix}\dot{x}(t)\\ \dot{y}(t)\\ \dot{\theta}(t)\end{bmatrix}=\begin{bmatrix}\cos\theta(t) & 0\\ \sin\theta(t) & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}V_l(t)\\ V_w(t)\end{bmatrix}=S(q)\begin{bmatrix}V_l(t)\\ V_w(t)\end{bmatrix} \tag{2}$$

where $S(q)$ is defining a full rank matrix and V_l and V_w, the linear and angular velocities. The dynamic model can be described by the following form of

dynamic equations based on Euler Lagrange formulation [18] and [19] as follows:

$$\begin{bmatrix} M & 0 & 0 \\ 0 & M & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\theta} \end{bmatrix} + \tau d = \frac{1}{r} \begin{bmatrix} \cos\theta & \cos\theta \\ \sin\theta & \sin\theta \\ \frac{L}{2} & \frac{-L}{2} \end{bmatrix} \begin{bmatrix} \tau_L \\ \tau_R \end{bmatrix} + \begin{bmatrix} -\sin\theta \\ \cos\theta \\ 0 \end{bmatrix} \lambda \quad (3)$$

where

τ_L and τ_R are the torques of left and right motors respectively.

M and I present the mass and inertia of the mobile robot respectively.

λ is the vector of constraint forces.

τ_d denotes bounded unknown disturbances including unstructured and unmodelled dynamics.

3. Cognitive Neural Controller

The approach used to detect the static obstacle in the desired path of the mobile robot is based on neural network model and to re-plan optimal smoothness desired path for mobile robot by using artificial intelligent technique in order to avoid the static obstacle with minimum distance and to track the desired trajectory by using an adaptive neural control methodology. The proposed controller can be given in the form of the block diagram shown in Fig. 2 [19]. It consists of:

a) Neural Network Topology Layer.

b) Cognitive Layer.

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

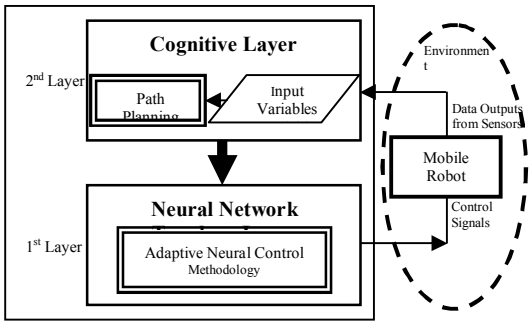

Figure 2 The proposed structure of the cognitive neural controller for the mobile robot system [19]

3.1. Neural Network Topology Layer

Neural network topology layer is the execution layer because it controls the mechanical actuators in order to follow the mobile robot's desired trajectory (fed from the cognitive layer). The general structure of this layer is an adaptive neural controller and can be given in the form of block diagram, as shown in Fig. 3 [19]. It consists of:

a) Posture Identifier.

b) Inverse-Dynamics Neural Feedback Controller.

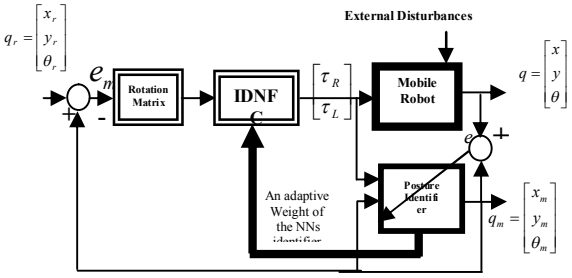

Figure 3 The proposed structure of the adaptive neural controller [19]

3.1.1. Posture Identifier

The modified Elman recurrent neural network model is applied to construct the position and orientation neural network identifier, as shown in Fig. 4 [19]. The nodes of input, context, hidden and output layers are highlighted.

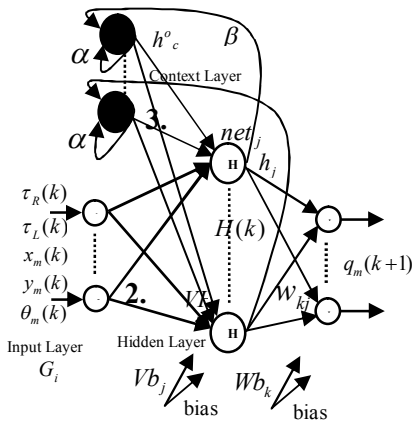

Figure 4 Elman neural networks acts as the posture identifier [18] and [19].

The structure shown in Fig. 4 is based on the following equations [20]:

$$h(k) = F\{VH\,\overline{G}(k), VC\,\overline{h^o}(k), bias\,\overline{Vb}\} \quad (4)$$

$$O(k) = (Wh(k), bias\,\overline{Wb}) \quad (5)$$

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 outputs of the identifier are the modelling pose vector in the surface and are 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. Dynamic back propagation algorithm is used to train the Elman network based on mean square error as in (6):

$$J = \frac{1}{np}\sum_{i=1}^{np}((x - x_m)^2 + (y - y_m)^2 + (\theta - \theta_m)^2) \quad (6)$$

where np is the number of patterns.

3.1.2. Inverse-Dynamics Neural Feedback Controller

The inverse-dynamic neural feedback controller (IDNFC) is essential to stabilize the tracking error of the mobile robot system when the trajectory of the robot drifts from the desired trajectory during transient state. The feedback controller consists of the nonlinear feedback acceleration control equation based on back-stepping technique and posture neural network identifier with optimization algorithm. The structure of the proposed nonlinear inverse-dynamic neural feedback controller can be shown in Fig. 5 [19].

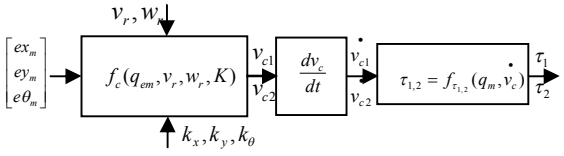

Figure 5 The nonlinear inverse-dynamic feedback neural controller structure [19].

The reference linear velocity and the reference angular velocity are given by (7) and (8) respectively [20].

$$v_r = \sqrt{(\dot{x}_r)^2 + (\dot{y}_r)^2} \quad (7)$$

$$w_r = \frac{\ddot{y}_r\dot{x}_r - \ddot{x}_r\dot{y}_r}{(\dot{x}_r)^2 + (\dot{y}_r)^2} \quad (8)$$

where

$q_r = (x_r, y_r, \theta_r)^T$ the desired pose vector.

$\dot{x}_r$ is the $\Delta x_r / T_s$.

$\dot{y}_r$ is the $\Delta y_r / T_s$.

$\ddot{x}_r$ is the $\Delta \dot{x}_r / T_s$.

$\ddot{y}_r$ is the $\Delta \dot{y}_r / T_s$.

T_s is the sampling time.

The control law for the smooth velocity control action can be expressed as equation (9) for $v_r > 0$ and $w_r > 0$ in order to make the system is asymptotically stable [19].

$$v_c = f_c(q_{em}, v_r, w_r, K) \quad (9)$$

where

q_{em} is the pose error vector $(ex_m, ey_m, e\theta_m)^T$.

K is nonlinear control gains.

The orthogonal rotation matrix as in (10) used to map the error of global coordinate system onto the error of local coordinate system and can be deduced from geometrical relationship of Fig. 6.

$$\begin{bmatrix} ex_m \\ ey_m \\ e\theta_m \end{bmatrix} = \begin{bmatrix} \cos\theta_m & \sin\theta_m & 0 \\ -\sin\theta_m & \cos\theta_m & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x_m \\ y_r - y_m \\ \theta_r - \theta_m \end{bmatrix} \quad (10)$$

where

ex_m and ey_m are the modelling coordinates errors and $e\theta_m$ is the modelling orientation angle error.

After taking the time derivative of the configuration error as follows:

$$\begin{bmatrix} e\dot{x}_m \\ e\dot{y}_m \\ e\dot{\theta}_m \end{bmatrix} = \begin{bmatrix} v_w ey_m - v_i + v_r \cos e\theta_m \\ -v_w ex_m + v_r \sin e\theta_m \\ w_r - v_w \end{bmatrix} \quad (11)$$

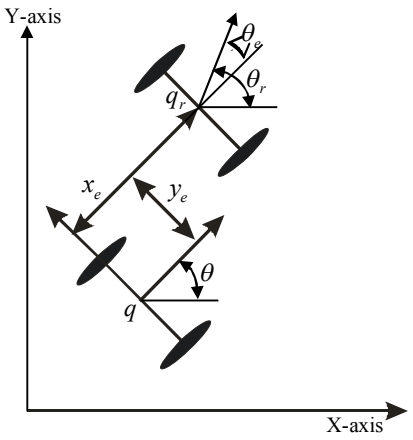

Figure 6 Path-tracking error of mobile

The control objective, a differentiable, time-varying controller will be proposed based on back-stepping method and posture identifier and proved by using the Lyapunov criterion as follows:

$$v_c = \begin{bmatrix} v_i \\ v_w \end{bmatrix} = \begin{bmatrix} v_r \cos e\theta_m + k_x ex_m \\ w_r + k_y v_r ey_m + k_\theta v_r \sin e\theta_m \end{bmatrix} \quad (12)$$

where $k_x, k_y, k_\theta > 0$ are design control gain parameters.

The proposed nonlinear feedback acceleration control input is the time derivative of v_c as follows:

$$\dot{v}_c = \begin{bmatrix} k_x & 0 & -vr \sin e\theta_m \\ 0 & k_y v_r & k_\theta v_r \cos e\theta_m \end{bmatrix} \begin{bmatrix} e\dot{x}_m \\ e\dot{y}_m \\ e\dot{\theta}_m \end{bmatrix} \quad (13)$$

The Lyapunov based nonlinear are the simplest but also successful methods in kinematics stabilization. A constructive Lyapunov functions criterion is considered based on as follows [23]:

$$V = \frac{1}{2}(ex_m^2 + ey_m^2) + \frac{1}{k_y}(1 - \cos e\theta_m) \quad (14)$$

Time derivative of (14) becomes:

$$\dot{V} = e\dot{x}_m ex_m + e\dot{y}_m ey_m + \frac{1}{k_y}\dot{\theta}_m \sin\theta_m \quad (15)$$

Then

$$\dot{V} = ((w_r + v_r(k_y ey_m + k_\theta \sin e\theta_m))ey_m - k_x ex_m)ex_m \quad (16)$$
$$+(-(w_r + k_y v_r ey_m + k_\theta v_r \sin e\theta_m)ex_m + v_r \sin e\theta_m)ey_m$$
$$+\frac{1}{k_y}\sin e\theta_m(-v_r(k_y ey_m + k_\theta \sin e\theta_m))$$

$$\dot{V} = -k_x ex_m^2 - v_r \frac{k_\theta}{k_y}\sin^2 e\theta_m \le 0 \quad (17)$$

Clearly, $V \ge 0$. If $q_{em} = 0$, $V = 0$. If $q_{em} \ne 0, V > 0$. and $\dot{V} \le 0$. If $q_{em} = 0$, $\dot{V} = 0$. If $q_{em} \ne 0, \dot{V} < 0$. then, V becomes a Lyapunov function.

So the closed loop system is globally asymptotically stable with three weighting parameters of error variables $(k_x, k_y, k_\theta) > 0$.

The controller gains (k_x, k_y, k_θ) are determined by two stages as follows:

$$\begin{aligned} k_x &= (k_x^* + \Delta k_x) > 0 \\ k_y &= (k_y^* + \Delta k_y) > 0 \\ k_\theta &= (k_\theta^* + \Delta k_\theta) > 0 \end{aligned} \quad (18)$$

where $(k_x^*, k_y^*, k_\theta^*)$ are determined by comparing the actual and the desired characteristic polynomial equations. While $(\Delta k_x, \Delta k_y, \Delta k_\theta)$ are determined by using the gradient- descent delta rule method in order to adjust the parameters of the nonlinear feedback acceleration controller. The desired characteristic polynomial takes the following form:

$$(Z + z_1)(Z + z_2)(Z + z_3) = 0 \quad (19)$$

where $z_1 = -e^{-2\xi w_n T_s}$ and $z_{2,3} = -e^{-\xi w_n T_s \pm j w_n \sqrt{1-\xi^2} T_s}$

The desired damping coefficient $\xi \in (0,1)$ and the characteristic frequency $w_n >> |w_{rMax}|$ are selected.

where w_{rMax} is the maximum allowed mobile robot angular velocity.

Substituting (12) in (11) then linearization the derivative state vector error and comparing coefficients at the same power

of Z in equation (19) the $(k_x^*, k_y^*, k_\theta^*)$ gains are obtained as follows:

$$k_x^* = k_\theta^* = \frac{(z_1 + z_2 + z_3 + 3)}{T_s(1 + v_r)} \tag{20}$$

$$k_y^* = \frac{z_1 z_2 + z_1 z_3 + z_2 z_3 - 3 - T_s^2 w_r^2 + 2(z_1 + z_2 + z_3 + 3) - T_s v_r \left(\frac{z_1 + z_2 + z_3 + 3}{1 + v_r}\right)^2}{T_s^2 v_r^2} \tag{21}$$

The control parameters (k_x, k_y, k_θ) of nonlinear feedback acceleration controllers are adjusted by using the gradient- descent delta rule method in order to find the suitable control gains.

$$\Delta(k_\gamma)(k+1) = -\eta \frac{\partial J}{\partial k_\gamma(k)} \tag{22}$$

where γ is x, y, θ for each time.

$$k_x(k+1) = k_x^*(k) + \Delta k_x(k) \tag{23}$$

$$k_y(k+1) = k_y^*(k) + \Delta k_y(k) \tag{24}$$

$$k_\theta(k+1) = k_\theta^*(k) + \Delta k_\theta(k) \tag{25}$$

k here indicates that calculations are performed at the k^{th} sample.

3.2. Cognitive Layer

Cognitive layer is the planning layer, which collects all the information from the environment by using sensors such as IR, 2D laser scanner, ultrasound and camera [19]. The cognitive layer can also re-plan the desired path if any static obstacle in the trajectory is detected, in order to avoid the mobile robot colliding with entities in the environment, ensuring that the trajectory tracking of the mobile robot allows collision-free navigation [19].

To apply the cognition path planning for the mobile robot, it needs a model of the obstacle in the cognitive layer to determine the obstacle's dimensions in order to avoid the accident between the mobile robot cart and the obstacle. It also needs an AI method to re-plan the path with minimum distance to avoid the obstacle and reach the desired path [19].

3.2.1. Obstacle Neural Network Model

The neural network can be described by the obstacle model in the path, as shown in Fig. 7 [2], [19], [20] and this structure is based on the following equations:

$$IH_m = xw_m x_i + yw_m y_i + \phi_{\text{Im}} \tag{26}$$

$$OH_m = f(IH_m) \tag{27}$$

$$C_o = f\left(\sum_{m=1}^{M} OH_m - \phi_o\right) \tag{28}$$

$$f(r) = \frac{1}{1 + e^{-r/p}} \tag{29}$$

where x_i and y_i are the coordinate of i^{th} points of the desired path; xw_m and yw_m are the network weights for x_i and y_i respectively; and ϕ_{Im} is the bias, which is equal to the free element in (26) expressing the shape of the obstacle.

IH_m is the weighted input of the m^{th} neuron of the middle layer and the neural activation function is $f(.)$ and p is a parameter that controls the step of curve shape.

m is the number of the neurons in the middle layer and it is equal to the number of vertices of the obstacle.

OH_m is the output of the m^{th} neuron of the middle layer.

It uses a repulsive penalty function (RPF) in the output neuron and ϕ_o is a bias which is equal to the number of the vertices of the obstacle decreased by 0.5.

C_o is the output of the obstacle neural network model of each point in the workspace is equal to 0 or 1. If the output is equal to 1, then the coordinate (x_i, y_i) is in the obstacle region, otherwise the point is not in it.

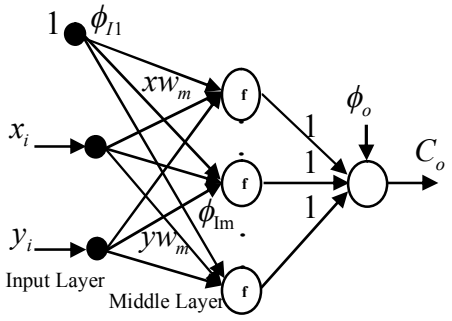

Figure 7 The obstacle neural network model

To build an optimal and robust path planning algorithm for maneuvering the mobile robot to avoid the obstacle in the environment while minimizing costs such as time, energy and distance, the following algorithm is used [19]:

- Determine the localization of the obstacle centre point (ob_x,ob_y) with respect to the reference point (x_0,y_0) and determine the dimensions of the obstacle as length L_{ob} and width W_{ob}. To calculate these main points, neural network obstacle model is used (as shown in Figure 9) with calculation of the vertices points, as a condition for minimum or maximum distance between mobile robot centre point and obstacle centre point using (26), (27), (28) and (29).
- The detection of the start point is undertaken by applying (x_i,y_i) for each coordinate point of the desired trajectory in the neural network obstacle model and finding the output of the model; if the model output C_o=1, the start point in the algorithm is detected as (x_{i-1},y_{i-1}), which means that the mobile robot is approaching the obstacle body. After finding the start point, (x_j,y_j) is applied for the coordinate point of the desired trajectory starting from start point in the neural network obstacle model, and finding the output of the model if the model output is changed from C_o=1 to C_o=0, the end point in the algorithm is detected as (x_j,y_j), that means the mobile robot is away from the obstacle body and it returns to the desired trajectory and the number of points between start and end points is denoted as φ.
- After determining the start and end points and the optimal side for re-planning the path, AI technique (e.g. particle swarm optimization (PSO)) is applied to plan the optimal smoothness path without overshooting between the start and end points with minimum distance.

3.3.2. Particle Swarm Optimization Technique

PSO is a kind of algorithm to search for the best solution by simulating to find the optimal path and to avoid static or dynamic obstacles [2]. PSO algorithms use a population of individuals (called particles) whose positions represent the potential solutions for the studied problem, with velocities are randomly initialized in the search space. The aim of the algorithm is to determine the points (x_i,y_i) (i=1, 2, 3…φ) that constitute the optimal smoothness path from the starting point to the end point. In order to reduce the length of the point's string, the point's x_i is determined by using the x-axes of the start and end points. Therefore, y_i becomes the search space for each via-point of the mobile robot trajectory and the via-point candidates are specified by one-dimensional data. The conventional evolutionary equations of particle swarm optimization are as follows [19], [20]:

$$V_{i,d}^{k+1} = V_{i,d}^{k} + c_1 r_1 (pbest_{i,d}^{k} - y_{i,d}^{k}) + c_2 r_2 (gbest_{d}^{k} - y_{i,d}^{k}) \quad (30)$$

$$y_{i,d}^{k+1} = y_{i,d}^{k} + V_{i,d}^{k+1} \quad (31)$$

$i = 1,2,3,.....pop$, $d = 1,2,3,.....\varphi$

where:

pop is the number of particles;

$V_{i,d}^{k}$ is the velocity of the i^{th} particle at k iteration;

$y_{i,d}^{k}$ is the position of the i^{th} particle at k iteration;
c_1 and c_2 are the acceleration constants with positive values equal to 2;
r_1 and r_2 are random numbers between 0 and 1;
$pbest_i$ is best previous weight of i^{th} particle;
$gbest_d$ is best particle among all the particle in the population.
The particles are evaluated using a fitness function to see how close they are to the optimal solution and the stability of the PSO algorithm. Two evaluation functions must be integrated into a fitness function, the collision avoidance and the shortest distance. Collision avoidance is essential to path planning and makes the mobile robot travel in the workspace safely. Collision avoidance can be described in two main points [2]:

1- The via-point y_i should not be in the obstacle region.

$$CA_{Fit1}=\begin{cases}1 & if \quad C_o=0\\ 0 & others\end{cases} \tag{32}$$

2- The section y_iy_{i+1} should not intersect obstacle region.

$$CA_{Fit2}=\begin{cases}0 & y_iy_{i+1}\cap obstacle\\ 1 & others\end{cases} \tag{33}$$

The fitness function of the collision avoidance can be given by (34) [19]:

$$CA_{Fit}=\begin{cases}1 & if \quad CA_{Fit1}\times CA_{Fit2}=1\\ 0 & othrs\end{cases} \tag{34}$$

The minimum distance is the second fitness function which makes the mobile robot travel in the workspace with minimum travelling time and travel distance and can be expressed as follows [2]:

$$MD_{Fit}=\sum_{j=1}^{\varphi-1}\sqrt{(x_{j+1}-x_j)^2+(y_{j+1}-y_j)^2} \tag{35}$$

The final fitness function is constructed as shown in (36) [2]:

$$Fit=MD_{Fit}/CA_{Fit} \tag{36}$$

When the final fitness function reaches the minimum value, the global optimal smoothness path is found. To investigate whether the new desired trajectory is optimal travelling time for the mobile robot, the desired linear velocity of the mobile robot while tracking the optimal path in order to avoid the static obstacle should not exceed the V_{Imax} and can be calculated using (37) [2]:

$$V_I=\frac{MD_{Fit}}{T}\rightarrow V_I<V_{Imax} \tag{37}$$

T must be calculated, which is the travelling time of the tracking between the start and end points, using equation (38) based the sampling time T_s as follows [2]:

$$T=\varphi\times T_s \tag{38}$$

where
φ is the maximum number of points.

4. Simulation Results and Experimental Work

The proposed controller is verified by means of computer simulation using MATLAB program. The simulation is carried out by tracking a desired position (x, y) and orientation angle (θ) with a lemniscates and square trajectories in the tracking control of the robot [2]. The parameter values of the robot model are taken from [7], [18] - [20]: M=0.65kg, I=0.36kgm^2, L=0.105 m and r=0.033 m. The first stage of operation is to set the posture identifier. Modified Elman recurrent neural networks model (MERNN) is used to oppose the conventional neural networks in the nonlinear mobile robot modelling for the following reasons [2]:

- To increase the speed of learning and to minimize the numbers of nodes in hidden layer because it has the context units that is used only to

memorize the previous activations of the hidden units.

- 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.
- To increase the order of the neural model for matching with actual model through self-connection in the context units for the Elman network.
- To reduce the output oscillation and to minimize the error between the actual output and neural network output.

This task is performed using modified Elman recurrent neural networks model and used PRBS signals as training set of 125 patterns with a learning rate of 0.1and sampling time of 0.5 second [2]. After 3244 epochs with a mean square error less than 5.7×10^{-6} the identifier outputs of the neural network, position x, y and orientation θ, are approximated to the actual outputs of the model trajectory, as shown in Fig. 8 [2].

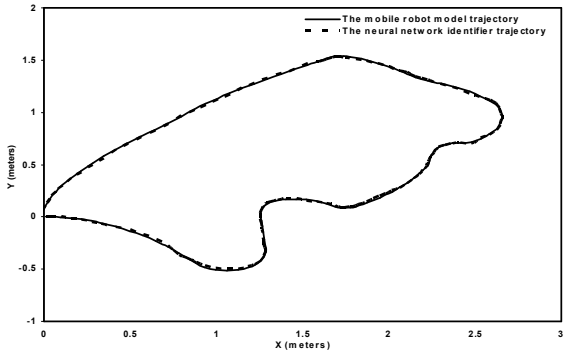

Figure 8. The response of posture identifier with the actual mobile robot model outputs for the training patterns.

Case I:

The desired lemniscates trajectory which has explicitly continuous gradients, this trajectory can be described by the following equations [2]:

$$x_r(t) = 0.75 + 0.75\times\sin(\frac{2\pi t}{50}) \quad , \quad y_r(t) = \sin(\frac{4\pi t}{50}) \tag{39}$$

$$\theta_r(t) = 2\tan^{-1}(\frac{\Delta y_r(t)}{\sqrt{(\Delta x_r(t))^2 - (\Delta y_r(t))^2} + \Delta x_r(t)}) \tag{40}$$

The robot model starts from the initial posture $q(0) = [0.75, -0.25, \pi/2]$ as its initial conditions.

A disturbance term $\overline{\tau d} = [0.01\sin(2t) \quad 0.01\sin(2t)]^T$ [2], [18] - [20] is added to the robot system as unmodelled kinematics and dynamics disturbances in order to prove the adaptation and robustness ability of the proposed controller [2].

The robot trajectory tracking obtained by the proposed cognitive neural controller is shown in Fig. 9.

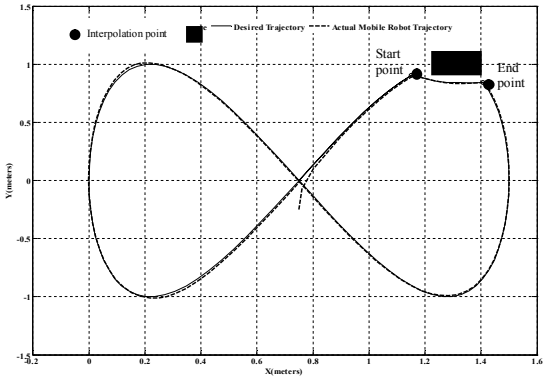

Figure (9). Actual trajectory of mobile robot and desired lemniscates trajectory with obstacle

This figure demonstrates excellent position and orientation tracking performance in terms of tracking the desired path as well as re-planning the path to avoid the static obstacle and the

simulation results demonstrated the effectiveness of the proposed controller by showing its ability to generate small smooth values of the control input torques for right and left wheels without sharp spikes, as shown in Fig. 10.

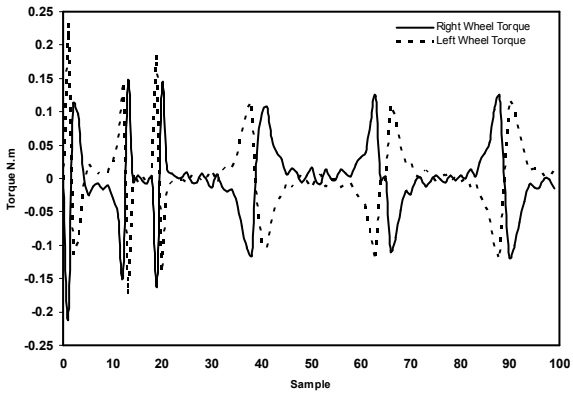

Figure 10. The torque of the right and left wheel action.

Case II:

Simulation is also carried out for desired square trajectory which has explicitly non-continuous gradient for verification the capability of the proposed controller performance [2]. Fig. 11 shows that the mobile robot tracks the square desired trajectory quite accurately and free-collision.

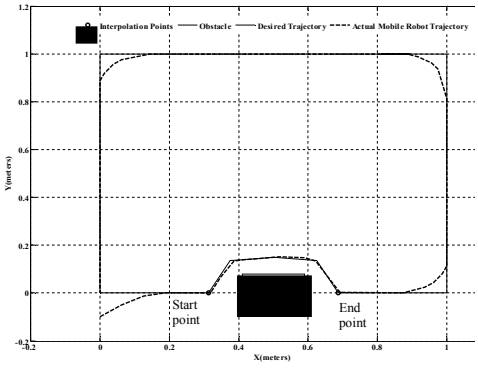

Figure 11 Actual trajectory of mobile robot and desired square trajectory with obstacle avoidance

Figure 12 shows the behaviour of the control action torques for right and left wheels is smooth values with small sharp spikes when the desired orientation angle changes suddenly at each corner.

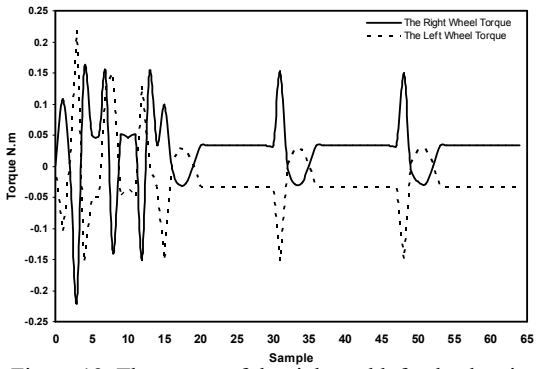

Figure 12. The torque of the right and left wheel action.

Case III:

In order to validate the applicability of the proposed control methodology, experiments have executed by using mobile robot from PARALLAX Inc. The lab experiments have been conducted using a Boe-Bot robotics type nonholonomic wheeled mobile robot (V3), as shown in Fig. 13 [2].

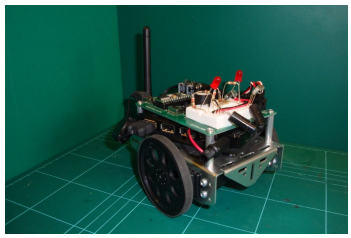

Figure 13. Boe-Bot mobile robot for the experiments [19].

The wheeled mobile robot is equipped with BASIC Stamp 2 programmable (BS2) microcontroller type (PIC16C57c) consisting of EEPROM 2KByte, a

decoding logic unit, infrared sensors, PWM generator for differential control of the robot [2], [7], [9], [18] - [20]. Velocities commands sent by the computer are coded messages which are recognized by microcontroller. Based on received characters, the microcontroller creates control actions for servo motors.

The output voltages of the two IR sensors are converted to coded messages by microcontroller and sent to the perso computer in order to calculate the tracki error of the mobile robot during motion. It is modified the data transmitti between the Boe-Bot robot and ma computer from wire to wirele communication by using wireless US Hub and adapter that has radio speed to 480Mbps and forty times faster th wireless Internet (802.11b) protocol [[19] and [20].

In the experiments, the best control data action of the simulations was the f step-ahead action of the cont methodology. These control data transmitted to the Boe-Bot mobile rol model, which admits right wheel veloc and left wheel velocity as input referei signals by using wireless USB h communication after has been conver the data format from MATLAB file simulations to BASIC Stamp Edi Software version 2.5 format as a look table [2].

The velocities of the simulation results for right and left wheels have downloaded to the memory of the Boe-Bot mobile robot as commands which have smooth values without sharp spikes and can be shown in Fig. 14a.

The mean of the mobile robot linear velocity is equal to 0.135 m/sec, and maximum angular velocity is equal to ± 0.65 rad/sec, as shown in Fig. 14b.

The initial pose for the Boe-Bot mobile robot starts at position 0.75 and -0.25 meter and orientation 1.57 radian, and should follow desired lemniscates trajectory, as show in Fig. 15. The desired trajectory starts at position 0.75 and 0.

After 50 second, the mobile robot has finished the tracking of the desired path and the tracking was reasonably accurate.

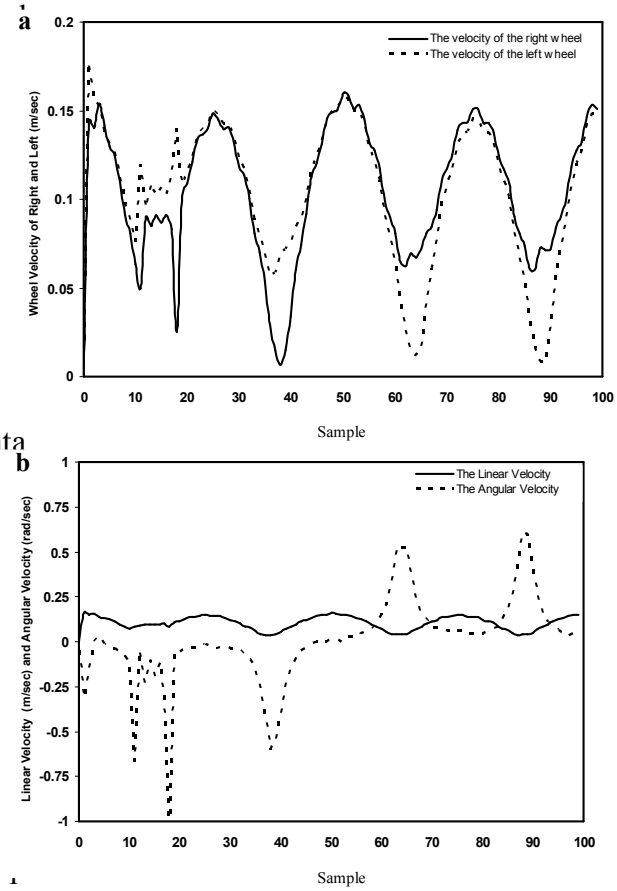

Figure 14 Velocity action: (a) the right and left wheel velocity; and (b) the linear and angular velocity

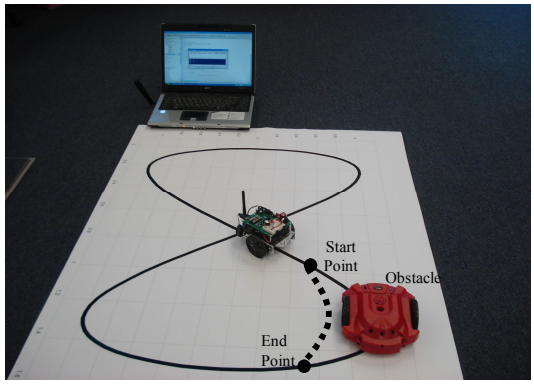

Figure 15 Real set-up experiment of Boe-Bot robot for lemniscates trajectory tracking with obstacle avoidance [19]

For the desired square trajectory, the Boe-Bot mobile robot starts at initial position 0 and -0.1 meter and initial orientation zero radian, and should follow desired path, as show in Fig. 16.

The desired square trajectory starts at position 0 and 0. After 32.5 second, the mobile robot has finished the tracking of the desired trajectory with good performance tracking.

The best control data action of the simulations has transmitted and downloaded to the memory of Boe-Bot mobile robot model, which admits right wheel velocity and left wheel velocity as control signals, as shown in Fig. 17a where the velocities of the right wheel and the left wheel were smooth values without sharp spikes and the mean of linear velocity of the mobile robot is equal to 0.125 m/sec, and the maximum peak of the angular velocity is equal to 0.625 rad/sec can be shown in Fig. 17b.

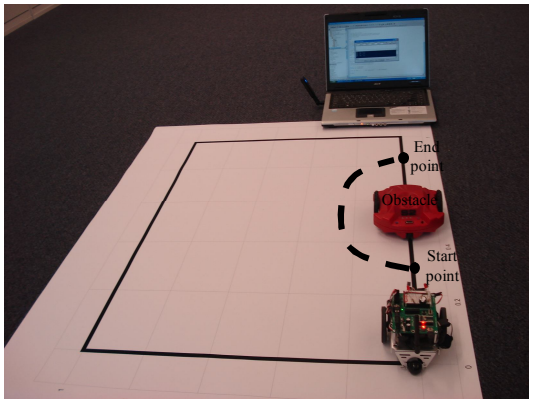

Fig. 16 Real set-up experiment of Boe-Bot robot for square trajectory tracking with obstacle avoidance [19]

The mean-square error for each component of the state error $MSE(ex,ey,e\theta)$ for simulation results and experimental work are calculated, as shown in Table 1 by using (41).

$$J1=\frac{1}{po}\sum_{i=1}^{po}((x_r-x_m)^2+(y_r-y_m)^2+(\theta_r-\theta_m)^2) \quad (41)$$

where *po* is the number of the desired trajectory points.

Table 1: The MSE for simulation results and experimental work.

Cognitive Control Methodology	Simulation Results	Experimental Work
$MSE(ex,ey,e\theta)$ for Lemniscates path	(0.023, 0.027, 0.035)	(0.024, 0.029, 0.0399)
$MSE(ex,ey,e\theta)$ for Square path	(0.039, 0.055, 0.03)	(0.062, 0.09, 0.0339)

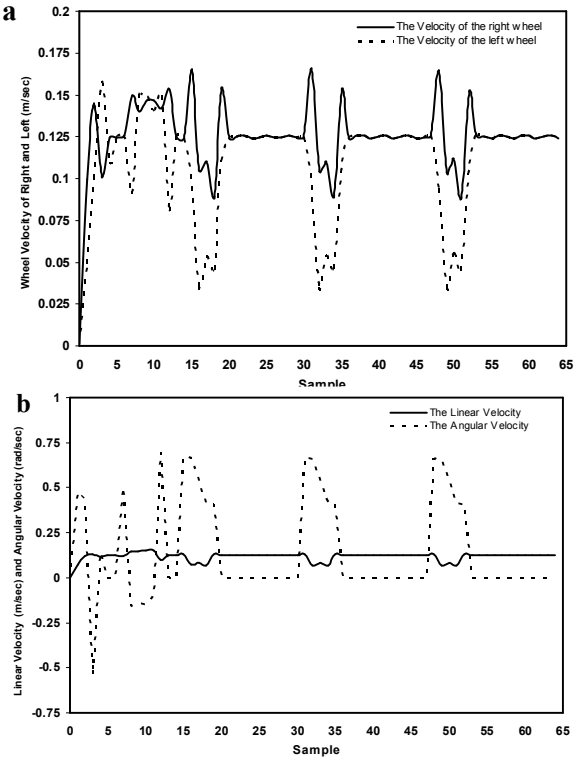

Figure 17. Velocity action: (a) the right and left wheel velocity; and (b) the linear and angular velocity.

The percentage of the mean square error between simulation results and experimental work can be shown in Table 2.

Table 2: The percentage of MSE between simulation results and experimental work

(MSE) for Desired Path	Lemniscates path	Square path
(MSE of X-coordinate) 100%	4.16%	37.09%
(MSE of Y-coordinate) 100%	6.89%	38.88%
(MSE of Orientation) 100%	10.77%	6.48%

The difference between simulation results and experimental work caused the residual errors in the experimental results due to the inherent friction present in the real system especially during tracking the non-continuous gradient path and modelling errors due to the difficulty estimating or measuring the geometric, kinematics or inertial parameters or from the lack of a complete knowledge of the components of the system.
In addition to that, calibration and alignment of the IR sensors for reading X-Y coordinate of the mobile robot trajectory cause some of error readings which were not presented in the simulation.

5. Conclusions

The Matlab simulation results and the experimental work of the proposed cognitive neural trajectory tracking control methodology for non-holonomic wheeled mobile robot which consists of two layers: neural network topology layer and cognitive layer illustrate evidently that the controller has been capable of:

- Identifying and modelling the system through using modified Elman recurrent neural network.
- Tracking continuous and non-continuous desired trajectories with collision-free navigation.
- Detecting the static obstacle and avoided it by re-planning desired trajectory based on PSO technique with minimum distance.
- Minimizing the tracking errors as well as overcoming the unmodelled kinematic disturbance.
- Generating smooth and suitable torque commands, τ_R and τ_L .

References

[1] M. Hulse and M. Hild, Informatics for cognitive robots. Advanced Engineering Information,Vol. 24, 2010, pp. 2-3.

[2] A. Al-Araji, Design of a cognitive neural predictive controller for mobile robot, PhD thesis (2012), Brunel University, United Kingdom.

[3] A.A. Razavian and J. Sun, Cognitive based adaptive path planning algorithm for autonomous robotic vehicles. Proceedings of the IEEE Southeast Con, 2005, pp. 153-160.

[4] W. Yaonan, Y. Yimin, Y. Xiaofang, Z. Yuanli, Y. Feng and T. Lei, Autonomous mobile robot navigation system designed in dynamic environment based on transferable belief model. Measurement, Vol. 44, 2011, pp. 1389-1405.

[5] C-Y. Tsai and K-T. Song, Visual tracking control of a wheeled mobile robot with system model and velocity quantization robustness. IEEE Transactions on Control Systems Technology, Vol. 17, No. 3, 2009, pp. 520-527.

[6] C. Samson, Time-varying feedback stabilization of car like wheeled mobile robot, International Journal of Robotics Research, vol. 12, no. 1, 1993, pp. 55–64.

[7] A. Al-Araji, M. Abbod and H. Al-Raweshidy, Design of an adaptive nonlinear PID controller for nonholonomic mobile robot based on posture identifier, Processding of the IEEE International Conference on Control System, Computing and Engineering, 2011, pp. 337-342.

[8] G. Klancar, I. Skrjanc, Tracking-error model-based predictive control for mobile robots in real time, Robotics and Autonomous Systems, vol. 55, no. 6, 2007, pp. 460–469.

[9] K-H. Su, Y-Y. Chen and S-F. Su, Design of neural-fuzzy-based controller for two autonomously driven wheeled robot. Neurocomputing. Vol. 73, 2010, pp. 2478-2488.

[10] S. Aydin, I. Kilic and H. Temeltas, Using Linde Buzo Gray Clustering Neural Networks

for Solving the Motion Equations of a Mobile Robot. Arabian Journal for Science and Engineering, vol. 36, no. 5, 2011, pp. 795-807.

[11] E.-H. Guechi, J. Lauber, M. Dambrine, G. Klancar, S. Blazic, Control design for non-holonomic wheeled mobile robots with delayed outputs, Journal of Intelligent & Robotic Systems, vol. 60, no. 3, 2010, pp. 395–414.

[12] X. Yin, C. Yang, W. Zhou and D. Xiong, Energy-efficient Tracking Control for Wheeled Mobile Robots Based on Bio-inspired Neurodynamic. Journal of Computational Information Systems, Vol. 10, No. 6, 2014, pp. 2533-2541.

[13] H. Wei, B. Wang, Y. Wang, Z. Shao and K. C. Chan, Staying-alive Path Planning with Energy Optimization for Mobile Robot. Expert Systems with Applications, Vol. 39, 2012, pp. 3559-3571.

[14] A. S. Matveev, C. Wang and A. V. Savkin, Real-time Navigation of Mobile Robots in Problems of Border Patrolling and Avoiding Collisions with Moving and Deforming Obstacle. Robotics and Autonomous Systems, Vol. 60, 2012, pp. 769-788.

[15] X. Linlin, T. Changxing, Y. Xuedong, K. Yunhan and F. Yan, 2-Dimensional SVFC Application in Path Planning of Mobile Robots. Energy Procedia, Vol. 17, 2012, pp. 1563-1569.

[16] N. Ghita and M. Kloetzer, Trajectory Planning for a Car-like Robot by Environment Abstraction. Robotics and Autonomous Systems, Vol. 60, 2012, pp. 609-619.

[17] S. Blazic, A novel trajectory-tracking control law for wheeled mobile robots. Robotics and Autonomous Systems, vol. 59 ,2011, pp. 1001–1007.

[18] A. Al-Araji, M. Abbod and H. Al-Raweshidy, Applying Posture Identifier in Designing an Adaptive Nonlinear Predictive Controller for Nonholonomic Mobile Robot. Neurocomputing, vol. 99, 2013, pp. 543-554.

[19] K. Dagher and A. Al-Araji, Design of a Nonlinear PID Neural Trajectory Tracking Controller for Mobile Robot based on Optimization Algorithm. Eng. & Tech. Journal. University of Technology. 32, (4), (2014), pp. 973-986.

[20] A. Al-Araji, Design of On-Line Nonlinear Kinematic Trajectory Tracking Controller for Mobile Robot based on Optimal Back-Stepping Technique. Iraqi Journal of Computers, Communication and Control & Systems Engineering. University of Technology. Vol. 14, No. 2, (2014), pp. 25-36.