Int. J. Nonlinear Anal. Appl. 10 (2019) No. 1, 153-166
ISSN: 2008-6822 (electronic)
https://dx.doi.org/10.22075/ijnaa.2019.4060

Adaptive fuzzy sliding mode and indirect radial-basis-function neural network controller for trajectory tracking control of a car-like robot

Masoud Shirzadeh[a], Abdollah Amirkhani[b,*], Mohammad H. Shojaeefard[b], Hamid Behroozi[c]

[a]*Department of Electrical Engineering, Amirkabir University of Technology, Tehran 15875-4413*
[b]*Department of Automotive Engineering, Iran University of Science and Technology, Tehran 16846-13114, Iran*
[c]*Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran*

(Communicated by Mohammad Bagher Ghaemi)

Abstract

The ever-growing use of various vehicles for transportation, on the one hand, and the statistics of soaring road accidents resulting from human error, on the other hand, reminds us of the necessity to conduct more extensive research on the design, manufacturing and control of driver-less intelligent vehicles. For the automatic control of an autonomous vehicle, we need its dynamic model, which, due to the existing uncertainties, the un-modeled dynamics and the performed simplifications, is impossible to determine exactly. Add to this, the external disturbances that exist on the movement path. In this paper, two adaptive controllers have been proposed for tracking the trajectory of a car-like robot. The first controller includes an indirect radial-basis-function neural network whose parameters are updated online via gradient descent. The second controller is adaptively updated online by means of fuzzy logic. The proposed controller includes a nonlinear robust section that uses the sliding mode method and a fuzzy logic section that updates some of the nonlinear control parameters online. The fuzzy logic system has been designed to deal with the chattering problem in the controller of car-like robot. In both controllers, the parameters have been determined by means of genetic algorithm. The obtained results indicate that even with the consideration of un-ideal effects such as uncertainties and external disturbances, the proposed controller has been able to perform successfully.

Keywords: dynamic model, trajectory tracking, car-like robot, sliding mode, fuzzy logic system
2010 MSC: 26D15, 26D10

*Corresponding author
Email address: amirkhani@iust.ac.ir (Abdollah Amirkhani[b,*])

Received: May 2019 *Revised:* October 2019

1. Introduction

Vehicles as means of transportation are considered a necessity in today's world. However, the use of vehicles has resulted in detrimental outcomes, as well. Road accidents involving automotive have taken the lives of 1.3 million individuals around the world in 2015 and about 1.4 million in 2016. According to the World Health Organization, fatal driving accidents rank as the 10th cause of death in the world [1]. Driver error is the cause of 72% of car accidents [2]. With the rapid advancement of artificial intelligence and car technology, autonomous vehicles are expected to take the pressure and stress of driving off the drivers and thus improve their workload and safety. Here, we have focused on the control system of an autonomous vehicle, and tried to present a controller for a basic model of such a car. Many of the previous works have introduced tracking controllers for driver-less automotive. Researchers initially designed motion controllers for these cars based on their kinematic model [3] and then started designing tracking controllers using the dynamic model. A simple fuzzy proportional–integral–derivative (PID) controller for the kinematic model of a car-like robot has been presented in [4]. Some researchers have designed kinematic model-based controllers while considering the skidding and slipping of car tires [5]. Using the virtual vehicle method, several articles have presented controllers for car-like robots based on their dynamic models [6]. The trajectory tracking of autonomous vehicles using the predictive control method has been discussed in [7]. In this study, the front wheel angle has been used in the control scheme. A path-following scheme for an autonomous four-wheeler using the combination of sliding mode and feedback control, along with the control of yaw moment, has been presented in [8] for the direct control of the car wheel. Numerous research works have been conducted on the lateral motion control of autonomous vehicles at the Stanford University dynamic design lab [9]. It is always challenging to determine the exact kinematic and dynamic models, and the uncertainties in these models cannot be avoided. In this regard, many researchers have presented adaptive and robust controllers for dealing with the issue of path tracking in wheeled mobile robots [10, 11]. Due to the existence of parametric and non-parametric uncertainties in system models and also the existence of non-holonomic constraints, the designing of output feedback controllers is a challenging task. Besides, the separation principle cannot be easily applied in this case. Both the trajectory-tracking and path-following problems have their own particular complexities when it comes to the kinematic model of car-like moving robots, which happens to be non-holonomic. The uncertainties associated with parametric modeling and unknown external disturbances constitute a significant concern in the development of advanced controllers for uncrewed vehicles. The unknown external disturbances may be caused by changing driving conditions. An adaptive neural controller for controlling the movement of an autonomous vehicle has been presented in this paper. The target of this work is to design an indirect adaptive output feedback controller by using an artificial neural network (NN) and considering model uncertainties and external disturbances. The network used in this indirect controller is a type of NN with radial basis functions (RBFs).

In this paper, two controllers have been designed for tracking the trajectory of a car-like robot. The first controller includes a Proportional-Integral-Derivative (PID) controller, a kinematic controller, and an indirect adaptive neural controller (IANC) that uses a radial-basis-function neural network (RBFNN). In the proposed method, neural network (NN) is updated online by means of gradient descent, and also the constant gains and the initial values of NN parameters are determined by appropriately employing a genetic algorithm (GA) and minimizing a defined cost function. The second controller, which is an adaptive fuzzy sliding mode nonlinear controller, is adaptively updated online by means of fuzzy logic. The sliding nonlinear control scheme has been considered for dealing with the existing uncertainties and external disturbances. To reduce the effect of chattering, the sign

function gains are adaptively modified by applying fuzzy logic. In addition to using the fuzzy logic system (FLS), the other constant parameters in the designed nonlinear controller are determined to employ GA. This paper has been organized as follows: The dynamic model of the car-like robot has been given in Section 2. IANC, along with its kinematic control and stability analysis, have been presented in Section 3. The adaptive sliding mode nonlinear controller, the stability analysis, and the FLS have been presented in Section 4. Simulation results and the conclusion have been presented in Section 5 and Section 6, respectively.

2. The dynamic model of the car-like robot

Consider the car-like robot in Figure 1. The configuration of variables is presented as $\mathbf{q} := (x_b, y_b, \theta_b, \varphi_b)^T$; where (x_b, y_b) are the Cartesian coordinates of the midpoint between two rear wheels, θ_b is the angle of car body relative to the x-axis, and φ_b is the control wheel angle. Assuming that the rear and front tires do not skid on the ground, the system constraints are written as

$$A(\mathbf{q})\dot{\mathbf{q}} = \begin{pmatrix} -\sin(\theta_b+\phi_b) & \cos(\theta_b+\phi_b) & 0 & 0 \\ -\sin(\theta_b) & \cos(\theta_b) & -l_b & 0 \end{pmatrix} \dot{q} = 0 \tag{2.1}$$

where l_b is the distance between the axles of front and rear wheels. The kinematic model of the car-like robot is obtained as

$$\dot{q} = \underbrace{\begin{pmatrix} \cos(\theta_b) & \sin(\theta_b) & \frac{\tan(\varphi_b)}{lb} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}^T}_{\mathbf{S^T}} \underbrace{\begin{pmatrix} v_b \\ \omega_b \end{pmatrix}}_{\mathbf{V}} \tag{2.2}$$

ω_b denotes the angular velocity of the front wheels and v_b is the car velocity. According to the Lagrange method, the dynamic equation of the system is derived as (see [12]):

$$\begin{gathered} \underbrace{\begin{pmatrix} m & 0 & I_c\sin(\theta_b) & 0 \\ 0 & m & -I_c\cos(\theta_b) & 0 \\ I_c\sin(\theta_b) & -I_c\cos(\theta_b) & I_b & I_w \\ 0 & 0 & I_w & I_w \end{pmatrix}}_{\mathbf{M}} \ddot{q} + \\ \underbrace{\begin{pmatrix} 0 & 0 & -I_c\dot{\theta}_b\cos(\theta_b) & 0 \\ 0 & 0 & -I_c\dot{\theta}_b\sin(\theta_b) & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}}_{\mathbf{V}} \dot{q} = \underbrace{\begin{pmatrix} T_x \\ T_y \\ T_\theta \\ T_\varphi \end{pmatrix}}_{\mathbf{T}} + \tau_{\mathbf{d}} + \mathbf{A}^T \underbrace{\begin{pmatrix} \lambda_1 \\ \lambda_2 \end{pmatrix}}_{\mathbf{\Lambda}} \end{gathered} \tag{2.3}$$

where $m := M_b + 4m_w$, $I_w := 2I_{w_{zz}}$, $I_c := ((l_b - l_c)\, M_b + 2l_b m_w)$ and $I_b := M_b(l_b - l_c)^2 + 4W^2 m_w + 2{l_b}^2 m_w + I_{b_{zz}} + 4I_{w_{zz}}$. $\mathbf{T}$ is the vector of forces and moments applied to the system in the generalized coordinates $\mathbf{q}$, and it is related to the vector of moments applied to the robot as follows:

$$\underbrace{\begin{pmatrix} T_x \\ T_y \\ T_\theta \\ T_\varphi \end{pmatrix}}_{\mathbf{T}} = \underbrace{\begin{pmatrix} \cos(\theta_b) & 0 \\ \sin(\theta_b) & 0 \\ l_b\cos(\varphi_b)\sin(\varphi_b) & 0 \\ 0 & 1 \end{pmatrix}}_{\mathbf{B}} \underbrace{\begin{pmatrix} \tau_1 \\ \tau_2 \end{pmatrix}}_{\tau} \tag{2.4}$$

Using the pseudo-inverse method, the inverse of Eq. (2.4) is obtained as

$$\underbrace{\begin{pmatrix}\tau_1\\ \tau_2\end{pmatrix}}_{\tau} = \underbrace{\begin{pmatrix}-\frac{\cos(\theta_b)}{b} & \frac{-\sin(\theta_b)}{b} & \frac{-l_b\cos(\varphi_b)\sin(\varphi_b)}{b} & 0\\ 0 & 0 & 0 & 1\end{pmatrix}}_{\mathbf{B}^{\dagger}} \underbrace{\begin{pmatrix}T_x\\ T_y\\ T_\theta\\ T_\varphi\end{pmatrix}}_{\mathbf{T}}, b := l_b^2\left(\cos^4(\varphi_b)-\cos^2(\varphi_b)\right)-1 \tag{2.5}$$

In Eq. (2.3), $\mathbf{\Lambda}$ is the Lagrange coefficients, τ_d is the vector of disturbing forces, τ_1 is the moment applied to the rear wheels, is the moment applied to the control wheel, is the wheel radius, is the wheel mass, $I_{w_{zz}}$ is the wheel moment of inertia about the z-axis passing through the wheel centroid, M_b is the car body mass, and $I_{b_{zz}}$ is the moment of inertia of car body about the z-axis passing through the car body's center of mass.

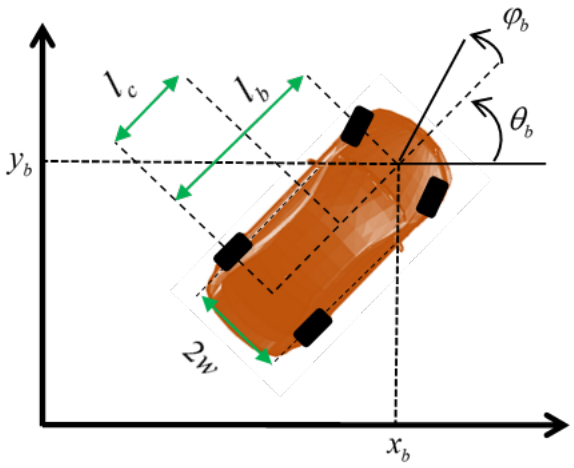

Figure 1: The car-like robot along with the relevant coordinate frames.

The term $\mathbf{S^T A^T}= \mathbf{0}$ can be used to eliminate $\mathbf{\Lambda}$ and $\mathbf{A}$ from the vehicle's dynamic equation [8]. The reduced form of the car's equation is expressed as follows:

$$\begin{aligned}&\mathbf{M_x}\dot{V}+\mathbf{V_x}V=\mathbf{B_x}\tau+\tau_{\mathbf{x}}, \mathbf{M_x}:=\mathbf{S^T M S}=[m_{ij}], \mathbf{V_x}:=\mathbf{S^T}(\mathbf{M\dot{S}}+\mathbf{VS})=[v_{ij}],\\ &\tau_{\mathbf{x}}:=\mathbf{S^T}\tau_{\mathbf{d}}, \mathbf{B_x}:=\mathbf{S^T B}=[b_{ij}], i=1,2, j=1,2\end{aligned} \tag{2.6}$$

In order to obtain the Lagrange coefficients, Eq. (2.3) and the time-derivative of Eq. (2.1) are simultaneously solved, as follows:

$$\begin{bmatrix}\tau\\ \mathbf{\Lambda}\end{bmatrix} = \begin{bmatrix}\mathbf{AM^{-1}B} & \mathbf{AM^{-1}A^T}\end{bmatrix}^T \begin{pmatrix}\begin{bmatrix}\mathbf{AM^{-1}B} & \mathbf{AM^{-1}A^T}\end{bmatrix} * \\ \begin{bmatrix}\mathbf{AM^{-1}B} & \mathbf{AM^{-1}A^T}\end{bmatrix}^T\end{pmatrix}^{-1} \left(-\mathbf{\dot{A}}\dot{q}+\mathbf{AM^{-1}V}\dot{q}\right) \tag{2.7}$$

After deriving the robot's motion equations, now it is time to design a suitable controller that can help the car-like robot track a desired path despite the existing uncertainties. In the first step, an indirect adaptive controller is designed for the robot. Then, by employing fuzzy sliding mode and fuzzy logic, a controller is devised for tracking the trajectory of the car-like robot.

3. Indirect RBF neural network controller by means of genetic algorithm

In this section, by using an RBFNN, an indirect adaptive controller is designed for tracking the trajectory of the car-like robot. The general structure of this controller is presented in Figure 2. A car model designed in the MapleSim has also been displayed in this figure [13]. The controller structure includes the kinematic controller section, the PID controller section, the IANC section, the section for determining system Jacobean, and the GA section. The reference path is considered as $\mathbf{q_r} := (x_r, y_r, \theta_r, \varphi_r)^T$. The goal for the controller is to bring error $e_q := \mathbf{q_v} - \mathbf{q}$ down to zero. First, the kinematic controller is designed. Suppose the variables of a virtual vehicle are in the form of $\mathbf{q_v} := (x_v, y_v, \theta_v, \varphi_v)^T$. By designing a kinematic controller based on the Lyapunov method, we intend to eventually obtain the values of v_v and ω_v, which are the linear and the angular velocities of the virtual vehicle, respectively. To this end, we implement the procedure presented in [14] for a wheeled mobile robot. First, the generalized error in the local coordinate system is obtained as follows [14]:

$$\varepsilon = (\varepsilon_1, \varepsilon_2, \varepsilon_3), \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \end{pmatrix} = \begin{pmatrix} \cos(\theta_v + \varphi_v) & \sin(\theta_v + \varphi_v) \\ -\sin(\theta_v + \varphi_v) & \cos(\theta_v + \varphi_v) \end{pmatrix} \begin{pmatrix} e_x \\ e_y \end{pmatrix}, \varepsilon_3 = \theta_r + \varphi_r - (\theta_v + \varphi_v) \quad (3.1)$$

The nonholonomic constraint for the car-like robot is expressed as $\dot{x}_b \sin(\theta_b) - \dot{y}_b \cos(\theta_b)$. By considering this constraint, the kinematic model for error ε is obtained.

$$\begin{aligned} \dot{\varepsilon}_1 = \varepsilon_2(\dot{\theta}_v + \dot{\varphi}_v) + v_r \cos(\varepsilon_3) - v, \dot{\varepsilon}_2 = -\varepsilon_1(\dot{\theta}_v + \dot{\varphi}_v) + v_r \sin(\varepsilon_3), \\ \dot{\varepsilon}_3 = \dot{\theta}_r + \omega_r - \dot{\theta}_v - \omega_v = \frac{v_r}{l_l}\sin(\varphi_r) - \frac{v}{l_l}\sin(\varphi) + \omega_r - \omega_v \end{aligned} \quad (3.2)$$

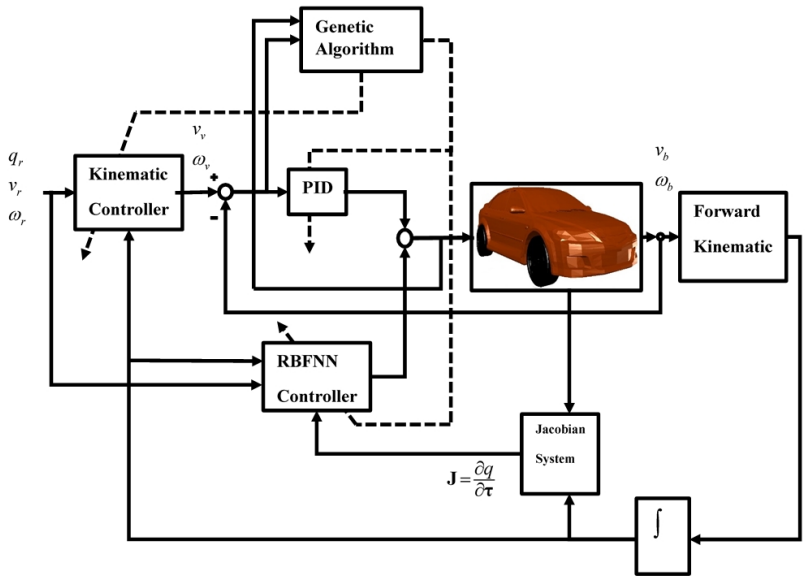

Figure 2: The general structure of the RBFNN controller for the car-like robot.

To design our controller for system (3.2), a Lyapunov function [14] and [15] is selected as follows:

$$L_1 = \frac{k_1}{2}(\varepsilon_1^2 + \varepsilon_2^2) + 2\left(\sin\left(\frac{\varepsilon_3}{2}\right)\right)^2 \quad (3.3)$$

By taking the derivative of (3.3) and performing a series of mathematical operations, (v_v, ω_v) are obtained.

$$\omega_v = k_1 v_r \varepsilon_2 + k_3 \sin(\varepsilon_3) + \omega_r + \frac{v_r}{l_l}\sin(\varphi_r) - \frac{v}{l_l}\sin(\varphi), v_v = v_r \cos(\varepsilon_3) + k_2 \varepsilon_1 \tag{3.4}$$

Following the kinematic control of the vehicle, a controller is designed to get $(v_b, \omega_b) \rightarrow (v_v, \omega_v)$. The main advantage of the kinematic controller is its simplicity; because it only relies on the kinematic model. The designed controller has three constant gains with positive values. In this paper, GA has been employed to determine these gains.The structure of the adaptive neural controller has been considered such that its parameters are updated online in order to minimize the following cost function:

$$F_1 = \frac{1}{2}\mathbf{E^T E}, \mathbf{E} = (e_1, e_2)^T, e_1 := v_b - v_v, e_2 := \omega_b - \omega_v \tag{3.5}$$

The network used in this scheme is a type of artificial NN with RBFs. In this paper, the NN undergoes a data-to-data form of training, and this process enables the control of the vehicle [16]. In the considered network, $\mathbf{h} = (h_1, h_2, ..., h_{N_2})^T$ denotes the vector of radial functions in the hidden layer [16]; and it is defined as:

$$h_j = \exp\left(-\frac{\|\mathbf{x} - \mathbf{c}_j\|^2}{2b_j^2}\right), j = 1, ..., N_2 \tag{3.6}$$

where $\mathbf{x} = (x_1, x_2, ..., x_{N_1})^T$ is the RBFNN's input vector, $\mathbf{c}_j = (c_{1j}, c_{2j}, ..., c_{N_3})^T$ are the centers of Gaussian functions, $b_j, j = 1, ..., N_2$ are the widths of Gaussian functions, $\mathbf{w} = [w_{ij}], i = 1, ..., N_2, j = 1, ..., N_3$ are the weights of RBFNN's output layer, and $\mathbf{y} = \mathbf{w}^T\mathbf{h} = (y_1, ..., y_{N_3})^T$ is the RBFNN's output vector. The descending gradient method is employed to train the NN online. By using the descending gradient approach, the RBFNN updating laws are obtained as

$$\begin{aligned}
&\Delta w_{jk}(t) = -\eta\frac{\partial F_1}{\partial w_{jk}} = \eta\alpha_k h_j, \Delta b_j(t) = -\eta\frac{\partial F_1}{\partial b_j} = \eta(\mathbf{w}\alpha)_j h_j b_j^{-3}\|\mathbf{x} - \mathbf{c}_j\|^2, \alpha = sign(\mathbf{J})\mathbf{E} \\
&\Delta c_{ij}(t) = -\eta\frac{\partial F_1}{\partial c_{ij}} = \eta(\mathbf{w}\alpha)_j h_j (x_i - c_{ij}) b_j^{-2}, i = 1, ..., N_1, j = 1, ..., N_2, k = 1, ..., N_3 \\
&w_{jk}(t + \Delta t) = w_{jk}(t) + \Delta w_{jk}(t) \\
&b_j(t + \Delta t) = b_j(t) + \Delta b_j(t) \\
&c_{ij}(t + \Delta t) = c_{ij}(t) + \Delta c_{ij}(t)
\end{aligned} \tag{3.7}$$

where $0 < \eta < 1$ is the updating rate of the NN parameters. $\mathbf{J} = \partial q/\partial\tau$ is the system's Jacobean, which is expressed as the sensitivity of system output to input [16]. The system's Jacobean is obtained from the reduced equations of the system, as follows [17].

$$\begin{aligned}
&\dot{V} = \mathbf{M}_x^{-1}\mathbf{B_x}\tau - \mathbf{M}_x^{-1}\mathbf{V_x}V \rightarrow V = \int_0^t \mathbf{M}_x^{-1}\mathbf{B_x}\tau dt' - \int_0^t \mathbf{M}_x^{-1}\mathbf{V_x}V dt' \rightarrow \\
&\frac{\partial V}{\partial \tau} = \int_0^t \mathbf{M}_x^{-1}\mathbf{B_x} dt' = \begin{bmatrix} \frac{\partial v_b}{\partial \tau_1} & \frac{\partial v_b}{\partial \tau_2} \\ \frac{\partial \omega_b}{\partial \tau_1} & \frac{\partial \omega_b}{\partial \tau_2} \end{bmatrix}, \mathbf{J} = \int \mathbf{S}\frac{\partial V}{\partial \tau} dt'
\end{aligned} \tag{3.8}$$

Only the sign of Jacobean $sign(\mathbf{J})$ has been used in the equation for updating the NN parameters. The kinematic controller has a gain of $\mathbf{K} = (k_1, k_3, k_4)$, and the coefficients of the PID controller are: $\mathbf{K_p} = diag([k_{p1}, k_{p2}])$, $\mathbf{K_i} = diag([k_{i1}, k_{i2}])$ and $\mathbf{K_d} = diag([k_{d1}, k_{d2}])$. The NN also has parameters such as $\mathbf{w}$, $\mathbf{c}$, $\mathbf{b} = (b_1, ..., b_{N_3})^T$ and η, whose values should be selected properly

GA can be an appropriate method for determining the values of the mentioned gains. GA can also be used for path programming in mobile robots [19]. In this paper, GA is employed to obtain the abovementioned constant values as well as the initial values of NN parameters. For this purpose, first, a cost function is defined as

$$F_2 = \frac{1}{N}\beta_1 \sum E^T E + \frac{1}{N}\beta_2 \sum (q_r - q)^T (q_r - q) + \frac{1}{N}\sum (\beta_3 |\tau_1| + \beta_4 |\tau_2|), N := \frac{FT}{\Delta t} \tag{3.9}$$

In the above equation, $\beta = (\beta_1, \beta_2, \beta_3, \beta_4)$ are the weights considered for each term of the cost function, is the simulation time, and is the sampling time step. The steps taken for the implementation of GA are as follows: Step 1: Generating and evaluating random populations. In this paper, every generated population is evaluated immediately; and by "evaluation", we mean the calculation of cost function (Eq. (3.9)) for each part of the population. Step 2: Selecting the parents and crossing them over to produce the offspring population. Step 3: Selecting the members of the offspring population and creating the mutated population. Step 4: Combining the original population with their offspring and the mutated population and establishing a new original population. The newly created population will form the current generation. Step 5: If the algorithm termination conditions are not met, go back to Step 2. Step 6: End of algorithm. Several conditions can be considered as the algorithm termination conditions: 1) Reaching an acceptable predefined limit for solution (e.g., the tracking error falling below a specified value). 2) Elapse of a specific time duration or reaching a definite number of iterations; if the time spent for executing the iterations is greater than a specified duration or the number of algorithm iterations is higher than a set value, stop executing the algorithm. 3) Elapse of a specific time duration or reaching a definite number of iterations without observing a significant improvement in the results.

In this paper, the second condition has been used for terminating the algorithm. The procedures for combining the three mentioned populations (Step 4) are as follows: if N_{pop} is the number of initial population members, after producing three main populations, the offspring and the mutations of these three populations are considered together and are arranged (in ascending order) according to the cost function (Eq. (3.9)). Then, N_{pop} members are selected from the beginning of this arranged population and considered as the current generation. In this paper, the parents for producing the offspring are selected randomly.

4. Fuzzy nonlinear adaptive sliding mode controller

In this section, by employing the sliding mode and fuzzy logic techniques, a fuzzy sliding mode nonlinear controller will be designed. The structure of this controller, which has been displayed in Fig. 3, includes the sliding mode controller (SMC) section, the FLS and the GA section.

The controller error and the sliding mode manifolds are defined as

$$\mathbf{e_q} := \mathbf{q_r} - \mathbf{q} = (e_x, e_y, e_\theta, e_\varphi)^T = (x_r - x, y_r - y, \theta_r - \theta, \varphi_r - \varphi)^T \tag{4.1}$$

$$\mathbf{S_M} := \mathbf{C}\mathbf{e_q} + \dot{\mathbf{e}}_\mathbf{q} \tag{4.2}$$

where $\mathbf{C} := diag(c_1, c_2, c_3, c_4)$ represents a definite positive diagonal matrix and $\mathbf{S_M} := (s_1, s_2, s_3, s_4)^T$ are the sliding mode manifolds. The SMC for the car-like robot has been designed as

$$\mathbf{T} = \mathbf{M}\left(\ddot{q}_r + \mathbf{C}\dot{\mathbf{e}}_\mathbf{q} + \mathbf{K}\mathbf{S_M} + \eta sign(\mathbf{S_M})\right) - \mathbf{A}^\mathbf{T}\mathbf{\Lambda} + \mathbf{V}\dot{q} \tag{4.3}$$

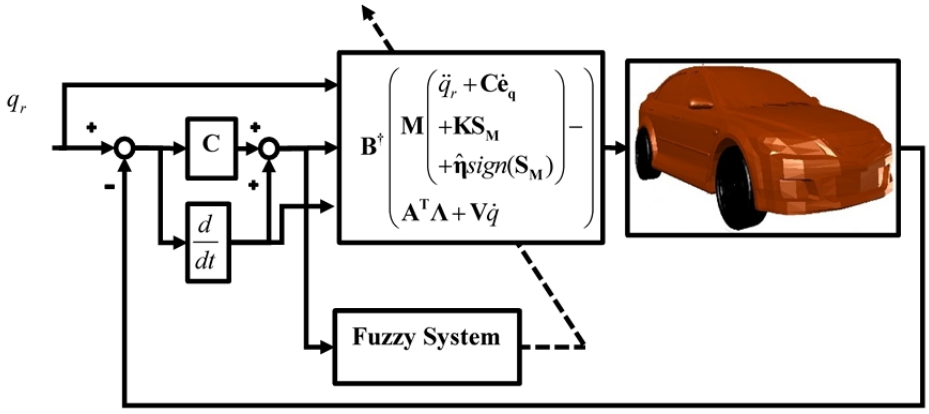

Figure 3: The structure of the fuzzy adaptive nonlinear controller designed for the car-like robot.

The time derivatives of sliding mode manifolds $\dot{\mathbf{S}}_\mathbf{M}$ have been given in Eq. (4.4), and the reaching rule has been presented in Eq. (4.5).

$$\dot{\mathbf{S}}_\mathbf{M} = \mathbf{C}\dot{\mathbf{e}}_\mathbf{q} + \ddot{\mathbf{e}}_\mathbf{q} = \mathbf{C}\dot{\mathbf{e}}_\mathbf{q} + \ddot{q}_r - \ddot{q} = \mathbf{C}\dot{\mathbf{e}}_\mathbf{q} + \ddot{q}_r - \mathbf{M}^{-\mathbf{1}}\left(\mathbf{T} + \tau_\mathbf{d} + \mathbf{A}^{\mathbf{T}}\boldsymbol{\Lambda} - \mathbf{V}\dot{q}\right) \tag{4.4}$$

$$\dot{\mathbf{S}}_\mathbf{M} = -\eta sign(\mathbf{S}_\mathbf{M}) - \mathbf{K}\mathbf{S}_\mathbf{M} \tag{4.5}$$

$\eta := diag(\eta_1, \eta_2, \eta_3, \eta_4)$ and $\mathbf{K} := diag(k_1, k_2, k_3, k_4)$ in Eqs. (4.3) and (4.5) represent definite positive diagonal matrices.

4.1. Stability analysis

To confirm the stability of the system and to ensure that error converges to zero, the Lyapunov function is considered as follows:

$$L_2 = \frac{1}{2}\mathbf{S}_\mathbf{M}^\mathbf{T}\mathbf{S}_\mathbf{M} \tag{4.6}$$

By taking the derivative of Eq. (4.6) and substituting Eqs. (4.3)-(4.5) in it, we will have

$$\dot{L} = \mathbf{S}_\mathbf{M}^\mathbf{T}\dot{\mathbf{S}}_\mathbf{M} = \mathbf{S}_\mathbf{M}^\mathbf{T}\left(\mathbf{C}\dot{\mathbf{e}}_\mathbf{q} + \ddot{q}_r - \mathbf{M}^{-\mathbf{1}}\left(\mathbf{T} + \tau_\mathbf{d} + \mathbf{A}^{\mathbf{T}}\boldsymbol{\Lambda} - \mathbf{V}\dot{q}\right)\right) = \mathbf{S}_\mathbf{M}^\mathbf{T}\left(-\mathbf{K}\mathbf{S}_\mathbf{M} - \eta sign(\mathbf{S}_\mathbf{M}) - \mathbf{M}^{-\mathbf{1}}\tau_\mathbf{d}\right) \tag{4.7}$$

By assuming $\|\mathbf{M}^{-\mathbf{1}}\tau_\mathbf{d}\| \leq \boldsymbol{\Gamma}_\mathbf{1}$ and $\|\eta\| = \boldsymbol{\Gamma}_\mathbf{1} + \|\eta_1\|$ (where $\boldsymbol{\Gamma}_\mathbf{1}$ has a positive value), Eq. (4.7) turns into Eq. (4.8).

$$\begin{aligned}\dot{L} = \mathbf{S}_\mathbf{M}^\mathbf{T}\left(-\mathbf{K}\mathbf{S}_\mathbf{M} - \eta sign(\mathbf{S}_\mathbf{M}) - \mathbf{M}^{-\mathbf{1}}\tau_\mathbf{d}\right) \leq \\ -\|\mathbf{K}\|\|\mathbf{S}_\mathbf{M}\|^2 - \left(\boldsymbol{\Gamma}_\mathbf{1} + \|\eta_1\|\right)\|\mathbf{S}_\mathbf{M}\| - \mathbf{S}_\mathbf{M}^\mathbf{T}\mathbf{M}^{-\mathbf{1}}\tau_\mathbf{d} \leq -\|\mathbf{K}\|\|\mathbf{S}_\mathbf{M}\|^2 - \|\eta_1\|\|\mathbf{S}_\mathbf{M}\|\end{aligned} \tag{4.8}$$

$\dot{L}$ is written as Eq. (4.8), and both sides of the equation are integrated.

$$\dot{L} = -\|\mathbf{K}\|\|\mathbf{S}_\mathbf{M}\|^2 - \|\eta_1\|\|\mathbf{S}_\mathbf{M}\| \tag{4.9}$$

$$\int_0^t \dot{L}dt' = \int_0^t \left(-\|\mathbf{K}\|\,\|\mathbf{S_M}\|^2 - \|\eta_1\|\,\|\mathbf{S_M}\|\right)dt' \to L(0) - L(t) = \int_0^t \left(-\|\mathbf{K}\|\,\|\mathbf{S_M}\|^2 - \|\eta_1\|\,\|\mathbf{S_M}\|\right)dt'$$
$$\to L(0) = L(t) + \int_0^t \left(\|\mathbf{K}\|\,\|\mathbf{S_M}\|^2 + \|\eta_1\|\,\|\mathbf{S_M}\|\right)dt' \to \geq \int_0^t \left(\|\mathbf{K}\|\,\|\mathbf{S_M}\|^2 + \|\eta_1\|\,\|\mathbf{S_M}\|\right)dt' \tag{4.10}$$

$$\lim_{t\to\infty}\left(\int_0^t \left(\|\mathbf{K}\|\,\|\mathbf{S_M}\|^2 + \|\eta_1\|\,\|\mathbf{S_M}\|\right)dt'\right) \leq L(0) < \infty \tag{4.11}$$

According to Barbalat's Lemma [18], when $t \to \infty$, then $\|\mathbf{K}\|\,\|\mathbf{S_M}\|^2 + \|\eta_1\|\,\|\mathbf{S_M}\| \to 0$; i.e. $\lim_{t\to\infty} \mathbf{S_M} = 0$; in other words, the surface of sliding mode $\mathbf{S_M}$ is asymptotically stable. Thus, the closed-loop errors converge to zero, and the stability of the closed-loop system is verified. In practice, for the prevention of chattering and for reducing the run time of the program (making the controller faster), function $\tanh(\sigma_i s_i)$ is used instead of function $sign(s_i)$; in this way, a uniformly ultimately bounded stability is guaranteed.

4.2. Designing the fuzzy system

The fuzzy method utilized to reduce the fluctuations. In a SMC, the intensity of fluctuations depends on the gain of the sign function [20]; so by utilize the fuzzy system (FS) adaptively, the control gain of the sign function is tuned according to the sliding surface (SS) and, thus, the amount of chattering is reduced [16]. Sliding occurs when $s_i\dot{s}_i < 0, i = 1, ..., 4$; if this condition is satisfied, when system states reach the SS, they will be on the SS[20]. If system states are to reach the SS, the selection of $\eta := diag(\eta_1, \eta_2, \eta_3, \eta_4)$ should eliminate the effects of uncertainties. There are two rules that ensure the establishment of sliding condition [20]. Rule 1: if $s_i\dot{s}_i > 0$, η_i should be increased. Rule 2: if $s_i\dot{s}_i < 0$ then η_i should be decreased. From the two stated rules, the relationship between $s_i\dot{s}_i$ and $\dot{\eta}_i, i = 1, ..., 4$ can be expressed as a FS. $s_i\dot{s}_i$ and $\dot{\eta}_i$ have been respectively chosen as the input and the output of the FLS. For generating the gain of the sign function, the designed FS comprises 5 fuzzy rules as R_f^j : $IF\ s_i\dot{s}_i\ is\ F_j\ Then\ \Delta\eta_i\ is\ G_j, (j = 1, ..., 5)$; where $\tilde{F} = \{NB, NM, ZO, PM, PB\}$ denotes the fuzzy set related to the input and $\tilde{G} = \{NB, NM, ZO, PM, PB\}$ is the fuzzy set related to the output. In fuzzy sets $\tilde{F}$ and $\tilde{G}$, "NB" denotes "negative big", "NM" represents "negative medium", "ZO" indicates zero, "PM" means "positive medium" and "PB" is "positive big".

The output of the FS shows the changes of sign function gain; therefore, by integrating the FS output, the estimated gain for the sign function is obtained according to Eq. (4.12) [20].

$$\hat{\eta}_i = g_i \int_0^t \dot{\eta}_i dt' \tag{4.12}$$

where, $g_i, i = 1, ..., 4$ have constant values. By employing the FS presented, control gain $\hat{\eta}_i, i = 1, ..., 4$ varies adaptively and in a programmed way by means of fuzzy rules, and according to the changes of the SS and its time derivative. Hence, the adaptive, fuzzy, integrated, sliding mode control can be expressed as

$$\mathbf{T} = \mathbf{M}\left(\ddot{q}_r + \mathbf{C}\dot{\mathbf{e}}_\mathbf{q} + \mathbf{K}\mathbf{S_M} + \hat{\eta}sign(\mathbf{S_M})\right) - \mathbf{A^T}\mathbf{\Lambda} + \mathbf{V}\dot{q} \tag{4.13}$$

where $\hat{\eta} := diag(\hat{\eta}_1, \hat{\eta}_2, \hat{\eta}_3, \hat{\eta}_4)$ indicates the time-variant switching control gain. For the inputs of fuzzy systems to fall between -1 and 1, gains of $f_i, i = 1, ..., 4$ are considered at the inputs. Like the IANC, GA is used here to determine the constant parameter values ($\mathbf{K}$, $\mathbf{C}$, g_i and f_i) for the

nonlinear fuzzy sliding mode adaptive controller. The selected cost function is Eq. (3.9), and the work procedure is similar to that explained in Section 3 for the IANC. After designing the controller, the simulation results will be presented in the next section.

5. Results

The following parameters and values have been used in the simulation: $M_b = 2117$ kg, $m_w = 28$ kg, $I_{w_{zz}} = 0.78$ kg.m^2, $l_l = 2.787$ m, $l_c = 1.487$ m, $I_{b_{zz}} = 1925$ kg.m^2, $W = 0.7775$ m, $\beta = (\beta_1, \beta_2, \beta_3, \beta_4) = (1, 1, 0.1, 0.1)$, $\mathbf{K} = (40.71, 33.3, 67.78, 49.44)$, $\mathbf{C} = (5.13, 85.59, 84.21, 74.84)$, $\Delta t = 0.01$ s and $r_w = 0.4$ m. The inputs of RBFNN are: $\mathbf{x} = (q, q_r, \dot{q}, \dot{q}_r)^T$, and there are 10 neurons in the middle layer. The upper and lower bounds considered for control signal saturation are as follows: $\max(\tau_1) = 25, \min(\tau_1) = -25, \max(\tau_2) = 200, \min(\tau_2) = -200$.

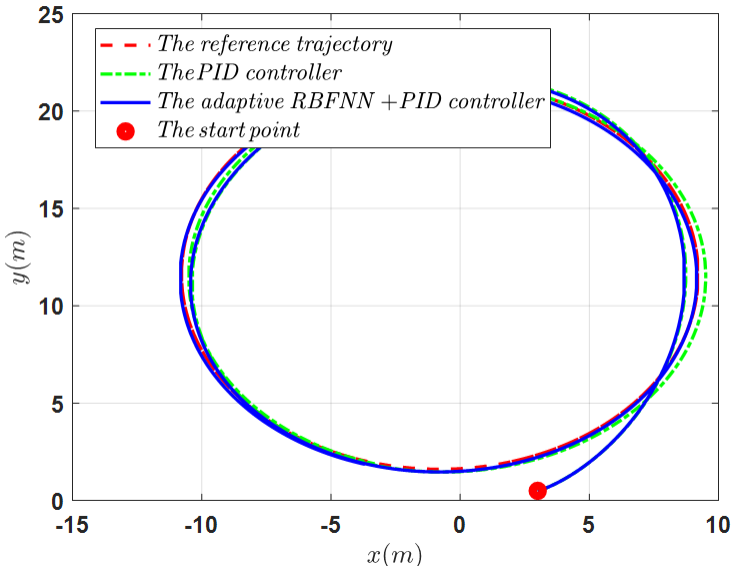

Figure 4: The trajectory of the car-like robot achieved by simulation under ideal conditions and by the IANC.

To evaluate the effects of disturbance and uncertainty, an external disturbance in the form of $\tau_d = (\gamma_1 \sin(\mu_1 t) \exp(-b_1 |t - t_1|), \gamma_2 \sin(\mu_2 t) \exp(-b_2 |t - t_2|))^T$ and with numerical values of $(b_1, b_2, t_1, t_2, \gamma_1, \gamma_2, \mu_1, \mu_2) = (0.01, 0.03, 50s, 120s, 100, 40, 0.3, 0.5)$ is applied to the system. The mean square error (MSE) is defined as

$$MSE := \frac{1}{FT} \sum (q_r - q)^T (q_r - q) \tag{5.1}$$

The results obtained for the IANC have been presented in Figures 4 through 8. The simulation time is $ST = 600s$. Figure 4 illustrates the car trajectory on a 2-D plane along with the reference path. According to this figure, the designed controller has been able to successfully control the car in following the given reference path. In this figure, a comparison has been made between the results of the controller designed in this paper and the results of a PID controller (whose coefficients have been determined by means of GA). Figure 4 demonstrates the superiority of the controller designed in this paper over the PID controller based on GA. Figure 5 shows the trajectory of the car-like robot obtained by incorporating the effects of uncertainty and external disturbance.

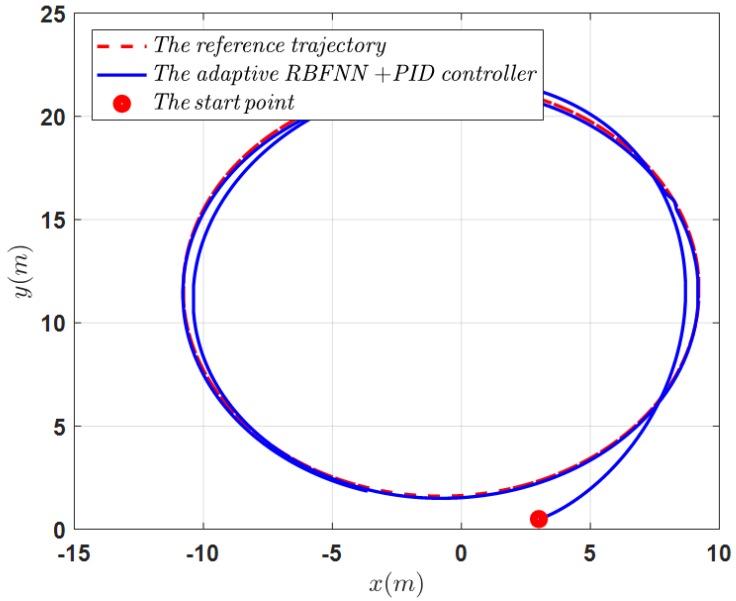

Figure 5: The trajectory of the car-like robot achieved by using the IANC and by considering the effects of uncertainty, external disturbance and control signal saturation.

To examine the robustness of the designed controller against control signal saturation, the effect of the saturation of control signals has also been considered in this simulation. Figures 5 and 6 reveal that the designed controller has been robust against the applied disturbance and has performed successfully. To explore the effect of car mass fluctuation on the performance of the designed controller, the mass of the vehicle has been increased by 20% at time t = 220s and reduced by 30% at time t = 400s.

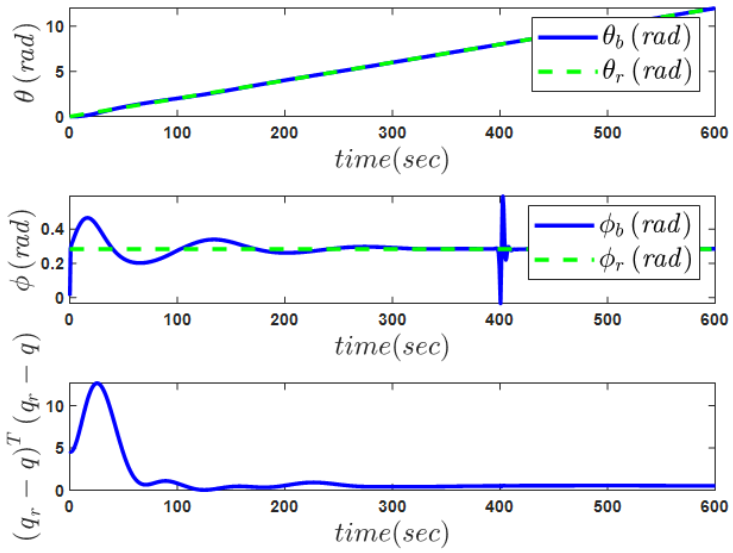

Figure 6: The diagrams of car angle and error norm versus time obtained by considering the effects of uncertainty, external disturbance and control signal saturation.

The graph of error norm versus time has been illustrated in Figure 6. In the presence of external disturbance, the designed adaptive neural controller successfully adapts itself online to the new conditions. This can be verified by the reduction of error norm and, consequently, the MSE values

shown in the simulation results of Figures 4, 5 and 6.

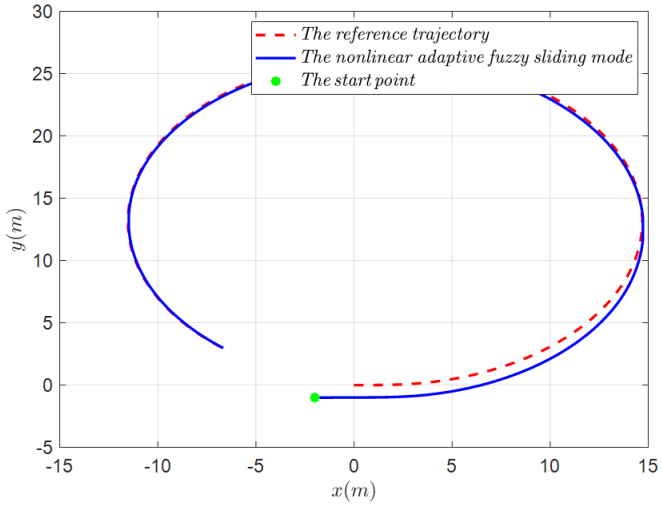

Figure 7: The trajectory of the car-like robot achieved by using the nonlinear fuzzy adaptive SMC and by considering the effects of uncertainty, external disturbance and control signal saturation.

The results obtained from the nonlinear fuzzy adaptive SMC for the car-like robot have been presented in Figure 7 and 8. In Figure 7, which shows the trajectory of the car-like robot achieved by considering the effects of uncertainty and external disturbance, the robustness of the designed controller against these disturbing factors has been well demonstrated. In addition to disturbance, the effect of uncertainty has been incorporated by changing the robot body mass (increasing the body mass by 30%at time t = 60s and reducing it by 20% at time t = 130s). Figure 7 and 8 indicate that the designed nonlinear fuzzy adaptive SMC has been quite effective in dealing with the external disturbances.

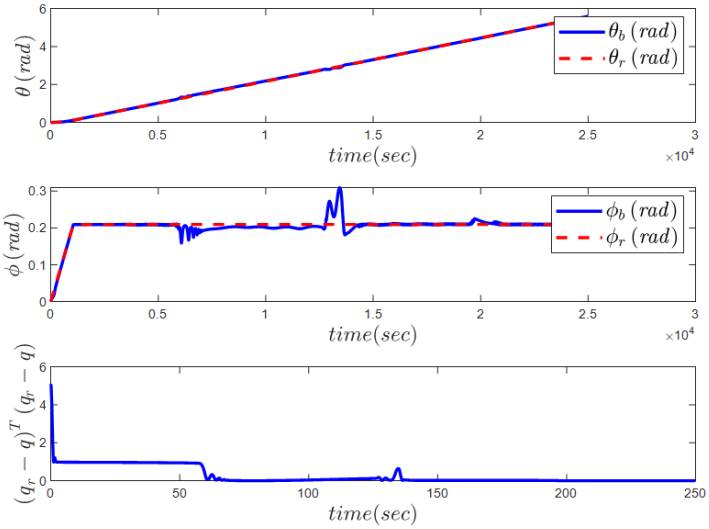

Figure 8: The diagrams of vehicle angle and error norm versus time obtained by considering the effects of uncertainty, external disturbance and control signal saturation and by using the nonlinear fuzzy adaptive SMC for the car-like robot.

The graph of error norm versus time has been illustrated in Figure 8. When exposed to external disturbances, the designed nonlinear fuzzy adaptive SMC is able to adapt itself online to the existing

conditions quite successfully. This can be confirmed by the reduction of error norm and MSE values shown in the simulation results of Figures 7 and 8. Besides the satisfactory performance of the nonlinear fuzzy adaptive SMC (according to Figures 7 and 8), by comparing Figure 6 (the IANC) with Figure 8 (the nonlinear fuzzy adaptive SMC), it can be realized that in the presence of external disturbances and with the fluctuations of car-like robot mass, the nonlinear fuzzy adaptive SMC performs better than the IANC. As Figures 7 and 8 indicate, despite the presence of disturbance and uncertainty, the car-like robot has succeeded in tracking the considered path.

6. Conclusion

The necessity of using intelligent and autonomous vehicles and also the difficulty of determining the exact model of a car-like robot as well as the existence of unknown external disturbances prompted us to employ the adaptive method to deal with these problematic factors related to the automatic control of such vehicles. In this paper, two adaptive methods were devised for the control of a car-like robot. The first approach is based on the neural network (NN) with radial basis functions, which acts as an indirect adaptive neural controller (IANC). The second approach is a nonlinear adaptive fuzzy sliding mode control technique. First, a simple model was presented for the car-like robot. Then a controller was designed for the trajectory tracking control of the robot. The first controller comprises a kinematic controller with constant gains; which has been designed by defining a Lyapunov function for it. This controller also includes an IANC whose parameters are updated online. The constant parameters of the controller and also the initial values of the NN used in this controller were determined by means of genetic algorithm (GA). The nonlinear fuzzy adaptive sliding mode controller (SMC) comprises a sliding mode section, whose stability has been evaluated by defining a Lyapunov function. It also includes a fuzzy section which has been designed, for the purpose of reducing the effect of chattering, by adaptively tuning the sign function gains. The constant control parameters of this controller were also determined by means of GA. Finally, simulations were performed by considering factors such as external disturbance and model uncertainty. In the Results section, the Proportional-Integral-Derivative controller based on GA was compared with the designed IANC; which showed the superiority of the latter to the former. In addition, the IANC (the first controller) was compared with the nonlinear fuzzy adaptive SMC (the second controller). The findings indicated the better performance of the nonlinear fuzzy adaptive SMC. The simulation results at the end of this work have been presented by considering the effects of model uncertainty and external disturbance. These results reveal that the designed methods (both the first and the second controllers) can effectively deal with disturbing factors and successfully control a car-like robot in autonomously tracking a desired path.

References

[1] L.D. Burns, *A vision of our transport future*, Nature **497** (2013) 181–182.
[2] *If autonomous vehicles rule the world: from horseless to driverless*, Economist 2015.
[3] S.A. Cohen and D. Hopkins, *Autonomous vehicles and the future of urban tourism*, Ann. Tourism Res. **74** (2019) 33–42.
[4] T. Litman, *Autonomous vehicle implementation predictions*, Victoria Transport Policy Institute, 2015.
[5] C. Gkartzonikas and K. Gkritza, *What have we learned? A review of stated preference and choice studies on autonomous vehicles*, Transport. Res. Part C: Emerg. Technol. **98** (2019) 323–337.
[6] World Health Organization (2015), "The top 10 causes of death," 2016. [Online]. Available: http://www.who.int/en/news-room/factsheets/detail/the-top-10-causes-of-death. [Accessed November 2018].
[7] P. Thomas, A. Morris, R. Talbot, and H. Fagerlind, *Identifying the causes of road crashes in Europe*, Ann. Adv. Automotive Med. **57** (2013) 13–22.

[8] R. Fierro and F.L. Lewis, *Control of a nonholonomic mobile robot: backstepping kinematics into dynamics*, 34th IEEE Conf. Decision Control, 1995, pp. 3805–3810.
[9] J. Funke, M. Brown, S.M. Erlien, and J.C. Gerdes, *Collision avoidance and stabilization for autonomous vehicles in emergency scenarios*, IEEE Trans. Control Syst. Technol. **25** (2017) 1204–1216.
[10] B.S. Park, S.J. Yoo, J.B. Park, and Y.H. Choi, *A simple adaptive control approach for trajectory tracking of electrically driven nonholonomic mobile robots*, IEEE Trans. Control Syst, 18 (2010) 1199-1206.
[11] D. Chwa, *Tracking control of differential-drive wheeled mobile robots using a backstepping-like feedback linearization*, IEEE Trans. Syst. Ma Cyber **40** (2010) 1285–1295.
[12] R. Murray, Z. Li, and S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, Boca Raton, 1994.
[13] Generic 4-Wheeled Vehicle with Independent Suspension, Maplesoft, [Online].Available:https://www.maplesoft.com/products/maplesim/mod elgallery/detail.aspx?id=60.
[14] F. Pourboghrat and M.P. Karlsson, *Adaptive control of dynamic mobile robots with nonholonomic constraints*, Comput. Electri. Engin. **28** (2002) 241–253.
[15] G. Klancar, A. Zdesar, S. Blazic, and I. Skrjanc, *Wheeled Mobile Robotics: From Fundamentals Towards Autonomous Systems*, Elsevier Science, 2017.
[16] J. Liu, *Radial Basis Function (RBF) Neural Network Control for Mechanical Systems: Design, Analysis and Matlab Simulation*, Springer, 2013.
[17] O. Mohareri, R. Dhaouadi, and A.B. Rad, *Indirect adaptive tracking control of a nonholonomic mobile robot via neural networks*, Neurocomputing **88** (2012) 54–66.
[18] J.-J.E. Slotine, and W. Li, *Applied Nonlinear Control*, Prentice-Hall, Englewood Cliffs, NJ., 1991.
[19] N. Noguchi and H. Terao, *Path planning of an agricultural mobile robot by neural network and genetic algorithm*, Comput. Electron. Agricul. **18** (1997) 187–204.
[20] J. Liu and X. Wang, *Fuzzy Sliding Mode Control*, Advanced sliding mode control for mechanical systems, Springer, Berlin, 2011, pp. 233–279.