Journal Européen des Systèmes Automatisés
Vol. 55, No. 3, 2022, June, pp. 281-298
Journal homepage: http://iieta.org/journals/jesa

Optimal Design and Performance Comparison of a Combined ANFIS-PID with Back Stepping Technique, Using Various Meta-Heuristic Algorithms to Solve Wheeled Mobile Robot Trajectory Tracking Problem

Rafik Euldji[1], Noureddine Batel[1], Redha Rebhi[2], Giulio Lorenzini[3], Nantapat Jarasthitikulchai[4*], Younes Menni[5], Hijaz Ahmad[6], Houari Ameur[5], Weerawat Sudsutad[7]

[1] Laboratory of Advanced Electronic Systems (LSEA), Department of Electrical Engineering, Faculty of Technology, University Yahia Fares, Medea 26000, Algeria
[2] Department of Mechanical Engineering, Faculty of Technology, University of Medea, Medea 26000, Algeria
[3] Department of Engineering and Architecture, University of Parma, Parco Area delle Scienze, 181/A, 43124 Parma, Italy
[4] Department of General Education, Faculty of Science and Health Technology, Navamindradhiraj University, Bangkok 10300 Thailand
[5] Department of Technology, University Centre of Naama (Ctr Univ Naama), P.O. Box 66, Naama 45000, Algeria
[6] Section of Mathematics, International Telematic University Uninettuno, Corso Vittorio Emanuele II, 39, Roma 00186, Italy
[7] Department of Statistics, Faculty of Science, Ramkhamhaeng University, Bangkok 10240, Thailand

Corresponding Author Email: nantapat.j@nmu.ac.th

https://doi.org/10.18280/jesa.550301

Received: 15 January 2022
Accepted: 24 May 2022

Keywords:
wheeled mobile robot (WMR), trajectory tracking, back-stepping, proportional integral derivative controller (PID), adaptive neuro-fuzzy inference system (ANFIS), meta-heuristics

ABSTRACT

The present investigation suggests a novel control technique that merge the advantage of the adaptive Neuro- Fuzzy inference system (ANFIS) with the proportional integral derivative (PID) controller, abbreviated as (ANFIS-PID), for dealing with the dynamics of the of wheeled mobile robot (WMR). A comparative study of various meta-heuristic optimization algorithms is made. The results revealed the highest efficiency of the suggested ANFIS-PID technique compared to seven designed PID controllers, in terms of settling and rise time, overshoot, and the integral square error (ISE) as a cost function. Various cases, study (circular path, diamond path, zigzag path) have highlighted the over performance of mentioned controller. Moreover, this hybrid technique is fused with back-stepping approach for the kinematic control. A lemniscate and a square trajectory were performed to clarify the capability of the mentioned controller.

1. INTRODUCTION

Nowadays, the tracking control issue in the robotics field still challenging and fascinates many researchers. Specifically for the wheeled mobile robot (WMR), since it covers a large civilian and military demands such as agriculture, manufacturing, services, medical, rescue and combat [1-5].

Each application has a special constraint connected to the robot composition (non-holonomic properties of WMRs), Moreover, WMRs are disturbed by task conditions, external load interruption, wheel slips, feedback sensors, which make the design of a precise control technique more difficult [6]. For that purpose, the investigating for new highly precise regulation schemes is required.

In order to overcome this issue, new control methods have been suggested in the last decade. These techniques can be classified into two parts and described by the kinematic or dynamic models [7]. The first group of researchers introduces a basic PID controller to handle the dynamics problem of the robot mobile and concentrating on developing the kinematic controller, an adaptive technique based on neural network using gradient decent for the auto-tuning the parameters for the back-stepping controller is demonstrated in Ref. [8], fuzzy logic controller for the self-tuning of the kinematic controller in Ref. [9], and a radial basis function network has been utilized in Ref. [10].

The major reason behind preferring this kind of controllers is that the dynamic controller's design is more complex compared to the kinematic one, while the dynamics depend on robot internal components such as the moment of inertia, mass, frictions [11]. However, the underestimating of the complete dynamics knowledge produce insufficient control technique, a torque controller is essential in many applications, like high-speed motions and large transportation [11]. For that reason, a sliding mode controller is introduced in references [12-14], an online real parameters identification of the WMR dynamics is suggested by Martins et al. [15], an adaptive back-stepping controller in reference [16], using an adaptive neural network controller for treating unmodeled dynamics in references [17, 18], fuzzy logic controller has been applied in reference [19, 20].

The PID control technique is one of the highly practical and most generally adopted approaches in various industrial applications because of the simplicity structure and good performance [21]. However, optimizing the parameter of the controller is challenging, especially for MIMO systems applications, to overcome this issue, many studies prove the high capability of meta-heuristic algorithms in dealing with this problem. We distingue, Genetic Algorithms (GA) [22], using particle swarm optimizer (PSO) [23], whale optimizer

(WOA) [24], grey wolf optimizer (GWO) [25], using Henry gas solubility optimizer in reference [26], artificial bee colonies (ABC) [27].

Generally, the fixed parameters reduce the control performance [28]. In order to overcome this problem, many researchers focusing on the inventing of new modified PID, like integrating the neural network with traditional PID controller (NN-PID) in references [29-31]. This concept gets the simplicity of a PID controllers and the capacity of understanding, adaptability and treating with nonlinearity from neural networks. Other researchers aiming to use the fuzzy logic controller as a tuning method for adjusting PID gains [32-34], a Reinforcement Learning method, have been utilized in reference [35]. However, the need for new regulation mechanisms growing higher and higher in accordance to complexity of the working environment.

The ANFIS could solve this problem by combining fuzzy logic controller advantage to deal with the uncertainties with the ability of neural networks to learn from plants/processes, this concept has attracted an exceptional attention in various engineering fields [36-40]. In this context, several hybrid designs have been made to enhance the robustness of the ANFIS controlled system, by benefiting from advantage and simplicity of a PID controller, we distingue the following mergers: hybrid ANFIS-PID [41-43], (ANFIS+I) controller optimizing using multidimensional PSO for Quadrotor Position Control [44], (ANFIS-PD+I) controller for robot manipulator [45].

The purpose of this work is to design a new hybrid optimized controller (ANFIS with PID) by using the grey wolf optimizer (GWO). The suggested controller is abbreviated as GWO-ANFIS-PID controller and it combines the advantages of the PID controller with the ANFIS controller to deal with the speed and angle control of the robot mobile. Moreover, a back-stepping kinematic controller is located in external loop for the regulation of the X, Y and theta coordinates. The following points present the main contributions of our work:

Introducing seven new meta-heuristics techniques to optimize the PID controller, based on the objective function under the name integral square error (ISE). Where these techniques are: particle swarm optimizer (PSO), grey wolf optimizer algorithm (GWO), dragonfly algorithm (DA), ant lion optimizer (ALO), grasshopper algorithm (GOA), moth flame optimizer (MFO), and the artificial bee colony algorithm (ABC).

Several tests in time domain have been made (such as rise time (tr), settling time (ts), overshoot (Mp), peak, and peak time (tp)) to give judgments between the selected nature inspired algorithms.

Results in MATLAB Shows the best designed PID candidate to realize the new hybrid controller (ANFIS-PID).

Improving the ANFIS-PID gains by the power full algorithm namely, grey wolf optimizer (GWO). Where several cases study (circular path, diamond path and zigzag path) gives the demonstration of the high robustness for the mentioned controller over the seven designed PID controllers.

Combining the ANFIS-PID controller with a Back-stepping approached in order to guarantee a minimum tracking error.

To prove the need for a kinematic controller, two types of paths are selected, the lemniscates and the squares trajectory to highlight the superiority of the designed technique in handling both smooth and sharp shaped trajectory.

The rest of the paper is presented as follows: The complete mathematical model of the wheeled mobile robot is presented in Section 2. The global control strategy is described in Section 3, where the simulation findings and analyses are summarized in Sections 4 and section 5, the conclusion of the work.

2. MODELING THE WHEELED MOBILE ROBOT

The WMR in Figure 1 has two pairs of DC motors, where R represents the driving wheels radius, φ_L and φ_R are the left and right spinning angles of wheels, respectively. Separated by a distance L, θ is the robot angle. At the distance d from the mind-point A, where (Xa, Ya) is the coordinate of A in the inertial frame (X,Y), and the coordinates of any position in the robot frame are illustrated by (Xr, Yr).

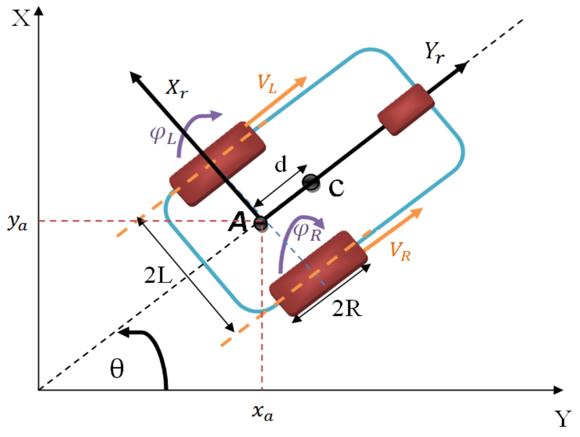


Figure 1. Description of the WMR

There are three basic steps to arrive at the complete mathematical model of the WMR: the kinematic model, dynamic model, and finally the DC actuator modeling, which are expressed as follows:

2.1 Kinematic model

This section concentrates on revealing the relationship between the linear and angular velocities of the mechanical processes without analyzing the forces disturbing the motion [8]. The linear speed of each driving wheel is represented as follows:

$$\begin{cases} v_R = R\dot{\varphi}_R \\ v_L = R\dot{\varphi}_R \end{cases} \tag{1}$$

The linear and angular speeds for the WMR are formulated by Eqns. (2) and (3):

$$v = \frac{v_R + v_L}{2} = R\frac{(\dot{\varphi}_R + \dot{\varphi}_L)}{2} \tag{2}$$

$$\omega = \frac{v_R - v_L}{2L} = R\frac{(\dot{\varphi}_R - \dot{\varphi}_L)}{2L} \tag{3}$$

The kinematic constraint can express by the following equations [34]:

No slip constraint:

$$-\dot{x}_a \sin\theta + \dot{y}_a \cos\theta = 0 \tag{4}$$

Pur rolling constraint:

$$\dot{x}_a \cos\theta + \dot{y}_a \sin\theta + L\dot{\theta} = R\dot{\varphi}_R$$
$$\dot{x}_a \cos\theta + \dot{y}_a \sin\theta - L\dot{\theta} = R\dot{\varphi}_L \tag{5}$$

The three constraint equations are:

$$\Lambda(q)\dot{q} = 0 \tag{6}$$

where:

$$(q) = \begin{bmatrix} -\sin\theta & \cos\theta & 0 & 0 & 0 \\ \cos\theta & \sin\theta & L & -R & 0 \\ \cos\theta & \sin\theta & -L & 0 & -R \end{bmatrix} \tag{7}$$

and

$$\dot{q} = \left[\dot{x}_a \dot{y}_a \dot{\theta} \dot{\varphi}_R \dot{\varphi}_L\right]^T \tag{8}$$

So, the kinematic model obtained is:

$$\begin{bmatrix} \dot{x}_a \\ \dot{y}_a \\ \dot{\theta} \\ \dot{\varphi}_R \\ \dot{\varphi}_L \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \\ \frac{1}{R} & \frac{-L}{R} \\ \frac{1}{R} & \frac{-L}{R} \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} = \frac{1}{2} \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \\ \frac{1}{R} & \frac{-L}{R} \\ \frac{1}{R} & \frac{-L}{R} \end{bmatrix} \begin{bmatrix} \dot{\varphi}_R \\ \dot{\varphi}_L \end{bmatrix} \tag{9}$$

This may be written as:

$$\dot{q} = S(q)_{\eta} \tag{10}$$

where, $\eta = \begin{bmatrix} \dot{\varphi}_R \\ \varphi_L \end{bmatrix}$ is the vector of the angular velocities of two wheels.

2.2 Dynamical model

The dynamic model purpose describes the study of the relationship between many forces and energies influencing on the process. The interpretation of the robot mechanism is expressed in terms of its component parts, such us the inertia, and centre of mass. Where The Lagrangian equation is represented by Ben Jabeur et al. [34, 46]:

$$M(q)\ddot{q} + V(q,\dot{q})\dot{q} + F(\dot{q}) + G(q) + \tau_d = B(q)\tau - \Lambda^T(q)^{\lambda} \tag{11}$$

For simulation purposes and control, Eq. (11) should be transformed into an alternative form, using the kinematic model (10):

$$S^T(q)\Lambda^T(q) = 0 \tag{12}$$

The new matrices are illustrated as following:

$$\begin{cases} \bar{M}(q) = S^T(q)\,M(q)S(q) \\ \bar{V} = S^T(q)M(q)\dot{S}(q) + S^T(q)V(q,\dot{q})S(q) \\ \bar{B} = S^T(q)B(q) \end{cases} \tag{13}$$

The simplified structure of the dynamic equations is given as:

$$\{\bar{M}(q)\dot{\eta} + \bar{V}(q,\dot{q})\eta = \bar{B}(q)\tau \tag{14}$$

where,

$$\bar{M}(q) = \begin{bmatrix} I_w + \frac{R^2}{4L^2}(mL^2 + I) & \frac{R^2}{4L^2}(mL^2 - I) \\ \frac{R^2}{4L^2}(mL^2 - I) & I_w + \frac{R^2}{4L^2}(mL^2 + I) \end{bmatrix} \tag{14a}$$

and

$$\bar{V}(q,\dot{q}) = \begin{bmatrix} 0 & \frac{R^2}{2L}m_c d\dot{\theta} \\ -\frac{R^2}{2L}m_c d\dot{\theta} & 0 \end{bmatrix},\ \bar{B}(q) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{14b}$$

Eq. (14) may be rewritten in a compact form:

$$\begin{cases} \left(m + \frac{2I_w}{R^2}\right)\dot{v} - m_c d\omega^2 = \frac{1}{R}(\tau_R + \tau_L) \\ \left(I + \frac{2L^2}{R^2}I_w\right)\dot{\omega} + m_c d\omega v = \frac{L}{R}(\tau_R - \tau_L) \end{cases} \tag{15}$$

where, $m = m_c + 2m_w$ is the total mass of the robot.

$I = I_c + m_c d^2 + 2m_w L^2 + 2I_m$ is the total equivalent inertia.

2.3 Actuator modeling

The job of the electrical actuator is to drive the mechanical system of a robot. In this work, two pairs of DC motor are used in order to guide the wheels there a determined motion (Figure 2).

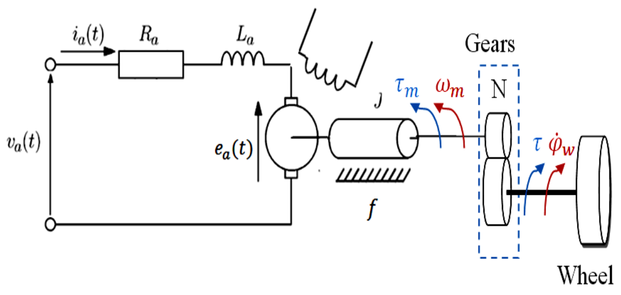


Figure 2. Equivalent electrical scheme of the motor.

The dynamic model of the actuators can be represented as [34]:

$$\begin{cases} v_a = R_a i_a(t) + L_a \frac{di_a(t)}{dt} + e_a(t) \\ e_a(t) = K_b \omega_m(t) \\ \tau_m = j\frac{dw_m(t)}{dt} + f w_m(t) + K_t i_a(t) \\ \tau = N\tau_m \end{cases} \tag{16}$$

where, i_a is the armature current, (R_a, L_a) is the resistance and inductance of the armature winding, respectively, e_a is the back emf, ω_m is the rotor angular speed, τ_m is the motor torque, (K_t, K_b) are the torque constant and back emf constant, respectively, N is the gear ratio, and τ is the output torque applied to the wheel Since the robot motors are mechanically coupled to wheels through the gears. Therefore, each dc motor will have:

For the two motors, the dynamic model is expressed as:

$$\begin{cases} \omega_{mR} = N\dot{\varphi}_{wR} \quad and\ \tau_R = N\tau_{mR} \\ \omega_{mL} = N\dot{\varphi}_{wL} \quad and\ \tau_L = N\tau_{mL} \end{cases} \tag{17}$$

$$\begin{cases} \frac{1}{(R+L_aP)}(e_{ar} - K_bN\dot{\varphi}_{wR}) = i_R \;\; with\; \tau_R = Nk_ti_R \\ \frac{1}{(R+L_aP)}(e_{al} - K_bN\dot{\varphi}_{wL}) = i_L \;\; with\; \tau_L = Nk_ti_L \end{cases} \tag{18}$$

The Physical parameters of the robot are presented in Ref. [34].

3. CONTROLLER DESIGN AND STRATEGY

Figure 3 explains the simplified form of the general control concept based of the WMR dynamics and kinematics. The role of path generator is to supplies the desired variable x, y and θ, a back-stepping act as a kinematic controller, located in the external loop, this controller measures the difference between the recommended signals (taken from the bloc trajectory generator) and real value from the robot, at next realizes its own linear speed and angle which they are passed to the internal loop, where two ANFIS-PID controller deal with the dynamics. Here, the two dynamic controllers give their own angular and linear speeds based on the data selected from the kinematic controller.

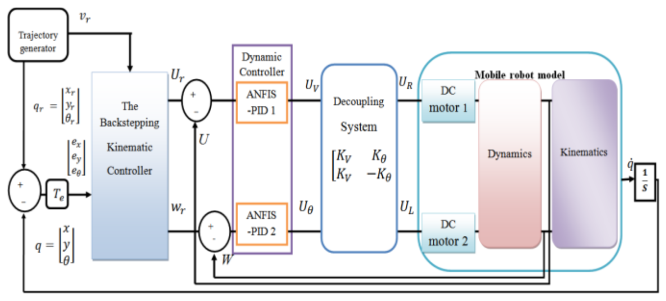

Trajectory generator
v_r
$q_r = \begin{bmatrix} X_r \\ y_r \\ \theta_r \end{bmatrix}$
$\begin{bmatrix} e_x \\ e_y \\ e_\theta \end{bmatrix}$
T_e
$q = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$
The Backstepping Kinematic Controller
U_r
w_r
U
W
Dynamic Controller
ANFIS -PID 1
ANFIS -PID 2
U_v
U_θ
Decoupling System
$\begin{bmatrix} K_V & K_\theta \\ K_V & -K_\theta \end{bmatrix}$
U_R
U_L
Mobile robot model
DC motor 1
DC motor 2
Dynamics
Kinematics
$\dot{q}$
$\frac{1}{s}$

Figure 3. General control scheme of the WMR

3.1 Kinematic controller design

In order to reach at a nearly zero distance error, a kinematic controller is required. The major advantage of the back-stepping approach is its practical and simplicity structure, since it will depend only on the kinematic model. Moreover, it has been classified as a stable tracking control law, where other techniques, such as nonlinear feedbacks and sliding-mode techniques, demand the knowledge of the dynamic model and their hardware realization is difficult to implement. For that reason, this controller makes it adopted in our study [33]. The controller structure is introduced in Figure 3, where the input error and velocity vector (v_c) are:

$$\begin{bmatrix} e_x \\ e_y \\ e_\theta \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_r - & X \\ Y_r - & Y \\ \theta_r - & \theta \end{bmatrix} = T_e e_r \tag{19}$$

$$v_c = \begin{bmatrix} v_r cos(e_\theta) + k_x e_x \\ w_r + k_y v_r e_y + k_\theta v_r sin(e_\theta) \end{bmatrix} \tag{20}$$

where, k_x, k_y and k_θ are tuning parameters.

3.2 The PID dynamic controller

The PID is a feedback controller that is widely utilized in various applications, especially for robotics and automation industry due to its simplicity structure and reliability that usually leads to satisfying results [21, 47] (Figure 4).

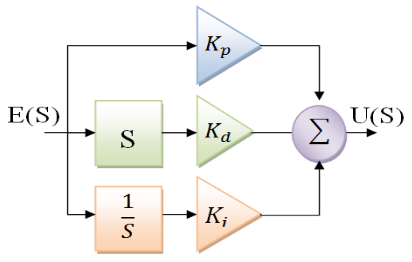


Figure 4. Block diagram of the PID controller

The representation of the transfer function for the PID in the time domain (Eq. (21)) and the transfer function in the Laplace domain (Eq. (22)).

$$u(t) = K_p e(t) + k_i \int_0^t e(t) + K_d \frac{de(t)}{dt} \tag{21}$$

$$C(s) = \frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d S \tag{22}$$

where, K_p, K_i and K_d are proportional, integral and derivative gains constants, respectively. The proportional constant kp role is to regulate the rise time value. where the purpose of the integral action ki is to eliminate steady-state error, the derivative gain kd for reducing the value of overflow and improving a transient response.

The success in designing PID controller rests to the optimal selection of parameters. The Ziegler–Nichols (Z-N) was supported in adjusting the tuning details for the PID. Today, the tuning methodology reaches a new era for soft computing approaches like the meta-heuristics algorithms. For that reason, the next section will focus on describing the tuning strategies.

3.3 Tuning methodology for PID

The block diagram for WMR tuning strategy is highlighted in Figure 5. While the first PID controller obtains the contrast between the desired and real speed from the right DC motor wheel as an input, where the reference velocity is fixed at: Ur =1 m/s.

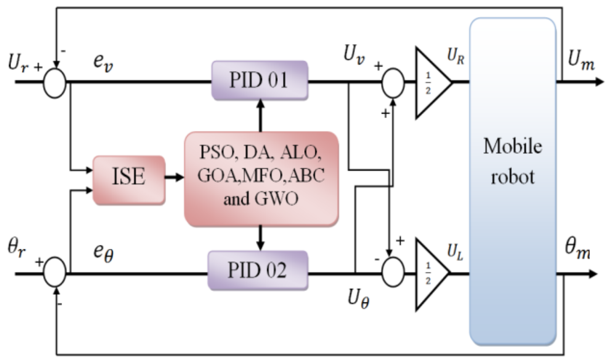


Figure 5. Bloc diagram for tuning methodology

The second PID controller in the left wheel examines the difference between the physical and desired directions angle as input. Here, the reference angle is a constant value: θr=0.785 rad.

The following Eq. (23) represent the relation between the input voltage of the right and left DC actuators, respectively

U_R, U_L with the outputs of the first and second controller U_V, U_θ, respectively.

$$\begin{cases} U_R = \frac{U_V + U_\theta}{2} \\ U_L = \frac{U_V - U_\theta}{2} \end{cases} \tag{23}$$

The cost function used in the study is the integral square error (ISE) which is demonstrated by Eq. (24):

$$ISE = \left(\int_0^\infty [e_v(t)]^2 dt + \int_0^\infty [e_\theta(t)]^2 dt\right) \tag{24}$$

where:

$$\begin{cases} e_v = U_r - U_m \\ e_\theta = \theta_r - \theta_m \end{cases} \tag{25}$$

U_r is the desired velocity, U_m the actual velocity, e_v the velocity error, θ_r the desired orientation, θ_m the measured orientation, and e_θ is the orientation error, The optimization algorithms used to tune the parameters (K_p, K_I, K_d) are described in the next section.

4. META-HEURISTIC ALGORITHMS

A comparative study has been made to demonstrate the capability of the grey wolf optimizer (GWO), compared to particle swarm optimizer (PSO), dragonfly optimizer (DA), Ant Lion optimizer (ALO), grasshopper algorithm (GOA), moth flame optimizer (MFO), and the artificial bee colony algorithm (ABC). Therefore, a brief description of all algorithms is given:

4.1 Particle Swarm Optimizer (PSO)

Each particle in the swarm updates their positions according to the following equations [48, 49]:

$$V_i^{k+1} = w\,V_i^k + C_1 R_1\left(p_i^k + x_i^k\right) + C_2 R_2\left(gbest_i + x_i^k\right) \tag{26}$$

$$X_i^{k+1} = X_i^k + V_i^{k+1} \tag{27}$$

Here, i refer to the particle in the swarm. k the iteration step carried out, w the inertia weight parameter and R_1 and R_2 are random numbers in the range [0, 1]. The coefficients C_1 and C_2 are the optimization parameters, X: position vector, and p_i^k: best position information achieved by the ith particle, $gbest_i$: best position information available in the swarm.

4.2 Dragonfly Algorithm (DA)

This approached was first introduced by Mirjalili [50], and was inspirited by the dynamic and static behaviour of dragonflies in nature explained in the references [50-52]. The life cycle of dragonflies is divided into two phases: the nymph phase and the adult phase [51]. The natural behaviour of each agent in the swarm obligates the attraction to nurturing sources and distract outward enemies [50, 52].

As demonstrated by Reynolds [53], the behaviour of swarm follows the following principles [50, 52, 53]:

4.2.1 Separation

This step aims to avoid individuals' collision with their neighbours that is near to its position, the static swarm defined by the following equation:

$$S_i = \sum_{j=1}^{n} X - X_j \tag{28}$$

Here *X* and X_j are the positions of the current individual and the *j*th neighboring individual, respectively. *n* is the number of neighboring individuals.

4.2.2 Alignment

The velocity matching between dragonflies of the same group is given by:

$$A_i = \frac{\sum_{j=1}^{n} V_j}{n} - X \tag{29}$$

where, v_j is the velocity of neighbouring individual *j*.

4.2.3 Cohesion

The tendency of members towards the centre of the swarm's group is known as the cohesion and it is defined as:

$$C_i = \frac{\sum_{j=1}^{n} X_j}{n} - X \tag{30}$$

4.2.4 Attraction to food

In order to survive, all individuals must move toward the food. As presented by this mathematical formula:

$$F_i = X^+ - X \tag{31}$$

where, X^+ shows the position of the food source.

4.2.5 Distraction from enemy

This equation represents the movement of dragonflies far away from the enemy's sources to survive:

$$E_i = X^- - X \tag{32}$$

where, X^- shows the position of the enemy. The position update of each dragonfly for testing another weight solution and receiving another fitness value is gotten by calculating ΔX and X using Eq. (36) and Eq. (37) [50, 52]:

$$\Delta X_i = (sS_i + aA_i + cC_i + fF_i + eE_i) + w \tag{33}$$

$$X_{t+1} = X_t + \Delta X_{t+1} \tag{34}$$

$$e = 0.1 - 1 * \left(\frac{0.1}{\frac{I}{2}}\right) \tag{35}$$

$$w = 0.9 - i * \left(\frac{0.9 - 0.4}{I}\right) \tag{36}$$

where, s, a, c, f, e, and w are the weights of their correspondent element. w is calculated using Eq. (36) here i is the current iteration and I is the number of iterations, and e is calculated in Eq. (35). s, a, and c are three different random numbers between 0 and 2e, f is a random number between 0 and 2. More details can be found elsewhere [50].

4.3 Ant Lion Optimizer (ALO)

This method proposed by Mirjalili [54] is inspired by the hunting mechanism of antlions in nature. The antlions have about three years average lifespan that it is spent as larvae except for 3-5 weeks of that period is spent in adulthood [55]. The mechanism is explained elsewhere [55, 56]. The mathematical illustration is given as follows [54-56]:

- Random walks of Ants

$$X(t) = [0, cumsum(2r(t_1 - 1), cumsum(2r(t_2) - 1, \cdots, cumsum(2r(t_n) - 1)] \tag{37}$$

where, n is the maximum number of iterations, cumsum calculates the cumulative sum, and t is the step of the random walks. Thus,

$$r(t) = \begin{cases} 1 \ if \ rand > 0.5 \\ 0 \ if \ rand < 0.5 \end{cases} \tag{38}$$

Here (t) is a stochastic function and rand is a random number generated with uniform distribution at interval of [0, 1]. The positions of ants is saved and utilised during optimisation in the matrix:

$$M_{ant} = \begin{bmatrix} A_{1,1} & A_{1,2} & \cdots & \cdots & A_{1,d} \\ A_{2,1} & A_{2,2} & \cdots & \cdots & A_{2,d} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ A_{n,1} & A_{n,2} & \cdots & \cdots & A_{n,d} \end{bmatrix} \tag{39}$$

where, M_{Ant} is the matrix for saving the position of each ant, $A_{i,}$ shows the value of the jth variable of ith ant, $n$is the number of ants, and $d$is the number of variables. Based on the random walk, each ant updates their position with each step of optimisation. The min-max normalisation equation is used to normalise the random walks:

$$X_i^t = \frac{(X_i^t - a_i) \times (d_i - c_i^t)}{(d_i^t - a_i)} + c_i \tag{40}$$

Here ai is the minimum of random walk of ith variable, d_i is the maximum of random walk of ith variable, c_i^t is the minimum of ith variable at tth iteration, and d_i^t is the maximum of ith variable at tth iteration.

- Trapping in Antlion's pit

The random walks of ants are influenced by the Antlion's trap. It is defined as:

$$\begin{aligned} c_i^t &= Antlion_j^t + c^t \\ d_i^t &= Antlion_j^t + d^t \end{aligned} \tag{41}$$

where, c^t represents the minimum of all variables at tth iteration, d^t indicates the vector including the maximum of all variables at tth iteration, c_i^t is the minimum of all variables for ith ant, d_i^t is the maximum of all variables for ith ant, and $Antlion_j^t$ shows the position of the selected jth antlion at tth iteration.

- Building trap

During optimisation the algorithm used a roulette wheel operator to choose antlions according to their fitness. This method increased the chance if catching ants.

- Sliding ants toward antlion

The following equations compute the decrease adaptively to the radius of the random walks hyper sphere of ants:

$$\begin{aligned} c^t &= \frac{c^t}{I} \\ d^t &= \frac{d^t}{I} \end{aligned} \tag{42}$$

where, I is a ratio, c^t is the minimum of all variables at tth iteration, and d^t indicates the vector, including the maximum of all variables at tth iteration.

- Catching Pray and Rebuilding the Pit

In the final hunting stage, antlions update their position to improve the possibility of catching new prey according to the position of the latest hunted ant:

$$Antlion_j^t = Ant_i^t \ if \ f(Ant_i^t) > f\left(Antlion_j^t\right) \tag{43}$$

Here t represents the current iteration, $Antlion_j^t$ is the position of the selected jth antlion at tth iteration, and Ant_i^t represents the position of ith ant at tth iteration.

- Elitism

Through the whole iteration, the best-obtained antlion is called an elite. This elite influences the movement of the rest of the ants during iteration. So, it is considered that every prey walks randomly around a chosen individual by the roulette wheel and the elite altogether represented by the following equation:

$$Ant_i^t = \frac{R_A^t + R_E^t}{2} \tag{44}$$

where, R_A^t is the random walk around the antlion selected by the roulette wheel at tth iteration, $R_E^t$is the random walk, and Ant_i^t represents the position of ith ant at tth iteration.

4.4 Artificial Bee Colony (ABC)

The artificial bee colony algorithm was first introduced by Karaboga and Basturk [57] in 2005, while its behaviour [58] was analysed in 2007. The behaviours of honeybees in finding food sources, splitting the knowledge with the nest, are its main inspiration. The approached classified bees to three types (employed, onlooker, and scout) where each agent plays different roles in the process. More details can be found in reference [52] while the process of the ABC algorithm is the following [52]:

- Initialization

At first, a ratio of the population is randomly sprayed into the solution area. Then, the fitness value, the nectar amounts, is fixed.

- Move the Onlookers

This equation calculates the probability of selecting a food source:

$$P_i = \frac{F(\theta_i)}{\sum_{k=1}^{S} F(\theta_k)} \tag{45}$$

where, θ_i denotes the position of the ith employed bee, S represents the number of employed bees, and P_i is the probability of selecting the ith employed bee. Later, onlooker bees move by the roulette wheel selection. The amount of nectar is then determined:

$$x_{ij}(t+1) = \theta_{ij} + \phi\left(\theta_{ij}(t) - \theta_{kj}(t)\right) \tag{46}$$

Here x_i denotes the position of the ith onlooker bee, t denotes the iteration number, θ_k is the randomly chosen employed bee, j represents the dimension of the solution and $\phi(\cdot)$ produces a series of a random variable in the range $[-1, 1]$.

- Move the Scouts

The given equation shows the move of scouts:

$$\theta_{ij} = \theta_{ijmin} + r \cdot (\theta_{ijmax} - \theta_{ijmin}) \tag{47}$$

where, r is a random number and $r \in [0, 1]$. Next, the best food source obtained so far and the fitness value are updated according to the response, with the respect of termination condition the algorithm outputs the results or repeats the program from step 2.

4.5 Grey Wolf Optimizer (GWO)

The grey wolf algorithm inspired from the social leadership and hunting behaviour of grey wolves in nature [48, 49, 59]. This approached consist of social hierarchy, enriching prey, search for prey, attacking prey, and hunting. Social hierarchy: The alpha (α) wolf is the leader of wolves, while Beta (β) and delta (δ) wolves are the second and third levels in the group, respectively. Those three wolves are considered as the best solution to lead the rest wolves known as omega (ω) wolves toward promising areas in order to find the global solution.

$$d = \left|c \cdot x_p(n) - x(n)\right| \tag{48}$$

$$x(n+1) = x_p(n) - a \cdot d \tag{49}$$

where, x_p: position vector of the prey, n: current iteration, and x: position vector of a grey wolf. The coefficient vectors $a \ and \ c$ are defined as follows:

$$\vec{a}_{(.)} = 2\vec{l} \cdot \vec{r}_1 - \vec{l} \tag{50}$$

$$\vec{c}_{(.)} = 2 \cdot \vec{r}_2 \tag{51}$$

Here, r_1 and r_2 are random numbers in [0, 1]. The following equations illustrate the mechanism of hunting:

$$\begin{cases} \vec{d}_\alpha = |\vec{c}_1 \cdot \vec{x}_\alpha - \vec{x}| \\ \vec{d}_\beta = |\vec{c}_2 \cdot \vec{x}_\beta - \vec{x}| \\ \vec{d}_\delta = |\vec{c}_3 \cdot \vec{x}_\delta - \vec{x}| \end{cases} \tag{52a}$$

$$\begin{cases} \vec{x}_1 = \vec{x}_\alpha - \vec{a}_1 \cdot (\vec{d}_\alpha) \\ \vec{x}_2 = \vec{x}_\beta - \vec{a}_2 \cdot (\vec{d}_\beta) \\ \vec{x}_3 = \vec{x}_\delta - \vec{a}_3 \cdot (\vec{d}_\delta) \end{cases} \tag{52b}$$

$$x(n+1) = \frac{\vec{x}_1 + \vec{x}_2 + \vec{x}_3}{3} \tag{52c}$$

4.6 Grasshoppers Optimization Algorithm (GOA)

The Grasshoppers Optimization Algorithm simulated by [60] is inspired from the natural flocking behaviour of grasshoppers. Those insects are regarded as pests because they damage agricultural corps. The swarming behaviour can be mathematically explained by the following equations [60-62]:

$$X_i = S_i + G_i + A_i \tag{53}$$

where, X_i is the position of the ith grasshopper, S_i is the classical interaction, G_i is the gravity force on the ith grasshopper, and A_i is the wind advection. The classical interaction is given by:

$$S_i = \sum_{j=1}^{N} s(d_{ij})\hat{d}_{ij} \tag{54}$$

$$d_{ij} = |x_i - x_j| \tag{55}$$

$$\hat{d}_{ij} = \frac{x_j - x_i}{d_{ij}} \tag{56}$$

Here, d_{ij} is the distance between the ith and jth grasshoppers, and s is a function for the definition of the strength of social forces defined as follows:

$$s(r) = fe^{-\frac{r}{l}} - e^{-r} \tag{57}$$

where, f is the intensity of attraction, and l is the attractive length scale. The distance between grasshoppers ranges between 0 and 15. Repulsion is observed in the interval [0 2.079]. The grasshoppers enter the comfort zone if they are far from 2.079 units from other grasshoppers. G component is defined as follows:

$$G_i = -g\hat{e}_g \tag{58}$$

Here, g is the gravitational constant and $\hat{e}_g$ is a unity vector towards the center of the earth. Parameter A is given by:

$$A_i = u\hat{e}_w \tag{59}$$

where, u is a constant drift and $\hat{e}_w$ is a unit vector in the direction of the wind. The position is updated to a new position based on the common position of the grass-hoppers, the food source position, and the position of all other grasshoppers:

$$X_i = \sum_{\substack{j=1 \\ j \neq i}}^{N} s(|x_j - x_i|)\frac{x_j - x_i}{d_{ij}} - g\hat{e}_g + u\hat{e}_w \tag{60}$$

Here, N is the number of grasshoppers. Although, this Eq. (65) cannot be straight-forward applied for optimization because grasshoppers do not converge to a specified point. Hence, the following is a corrected equation to update the grasshopper's position:

$$X_i^d = c\left|\sum_{\substack{j=1\\ j\neq 1}}^{N} c\frac{ub_d - lb_d}{2} s(|x_j^d - x_i^d|)\frac{x_j - x_i}{d_{ij}}\right| + \hat{T}_d \quad (61)$$

$$C = (C_{max} - 1)\frac{C_{max} - C_{min}}{Max\ Itr} \quad (62)$$

where, ub_d is the upper bound, lb_d is the lower bound, $\hat{T}_d$ is the value of the D_{th} dimension in the target space (optimal solution found so far), and c is a decreasing coefficient to shrink the comfort zone, repulsion zone, and attraction zone.

4.7 Moth Flame Optimization (MFO) Algorithm

The Moth-flame optimization algorithm was first introduced by Mirjalili [63]. Those fancy insets are quite similar to the butterfly family. The natural behaviour of them in navigation known as transverse orientation is the main motivation for the algorithm. Moths fly using a fixed angle with respect to the moon, which is a very efficient mechanism for long travelling distances in a straight line. The MFO algorithm follows the following steps:

$$MFO = (R, V, T) \quad (63)$$

moths randomly and also the fitness values of them, *V* is a main function that makes the moths move around the search space and *T* is a termination criterion flag. In *V* function, the position of each moth with regard to a flame is updated as per Eq. (64):

$$M_i = S(M_i, G_j) \quad (64)$$

where, *S* is the spiral function, M_i is the i^{th} moth, and G_j is the j^{th} flame, whereas $S(M_i, G_j)$is calculated using the following equations:

$$S(M_i, G_j) = D_i e^{bl}\cos(2\pi l) + G_j \quad (65)$$

Here, *Di* represents distance of i^{th} moth from j^{th} flame, *b* is constant for announcing the shape of logarithmic spiral, and *l* is a random number between [− 1, 1]. *D* is calculated using the following equation.

$$D_i = |G_j - M_i| \quad (66)$$

where, G_j indicate the j^th flame and M_i is the i^{th} moth. The position of moths is updated with respect to n different locations in the search space, which can degrade the best promising solutions exploitation. Consequently, the number of flames adaptively decreases over the course of iterations using the following formula:

$$flame_{no} = round(FN - 1 * \frac{N - 1}{IN}) \quad (67)$$

where, *I* is the current number of iterations, *I N* is the maximum number of iterations and FN is the maximum number of flames. More details can be found elsewhere [63-65].

5. THE ANFIS CONTROLLER DESCRIPTION

Adaptive Network-Based Fuzzy Inference System (ANFIS) is a hybrid soft computing algorithm that combined the neuro-fuzzy technique, namely fuzzy inference system (FIS) with artificial neural network (ANN). ANFIS was firstly introduced by Jang [66] and it is based on the first-order Sugeno fuzzy model [67].The ANFIS approaches integrate the best feature of fuzzy systems and neural networks by exercising learning ability and capability of FIS and ANN, respectively [68]. The ANFIS architecture consists of if-then rules and couples of input-output data of fuzzy. For training, ANFIS uses neural network learning algorithms. As the algorithm used it to fine-tune the membership function (MF) and the associated parameter that methods the desired data sets [68, 69].

For clarifying, a Sugeno fuzzy model with two inputs, x and y, and one output is considered. Normally, the fuzzy rules are reported as follows [66-69]:

Rule I: if $x=A_1$ *and* B_1, then

$$f_1 = p_1x + q_1y + r_1 \quad (68)$$

Rule II: if $x=A_2$ *and* $y=B_2$, then

$$f_2 = p_2x + q_2y + r_2 \quad (69)$$

where, *f* is an output parameter (response), *p*, *q*, & r are linear parameters, and *A&B* are nonlinear parameters. Figure 6 Illustrates the five layers used to create ANFIS structure while the function of each layer is represented as follows:

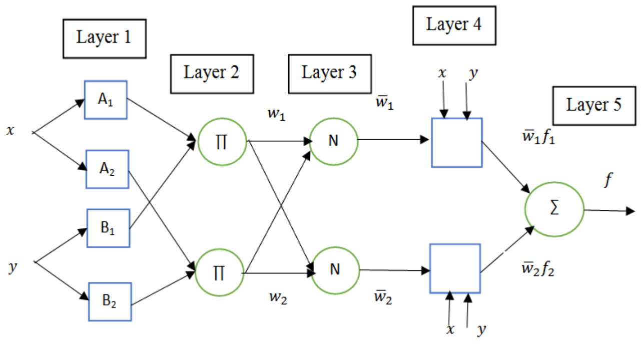


Figure 6. The ANFIS architecture with two input parameters x and y

Layer 1. Input nodes: the nodes of this layer generate membership grades to which they belong to each of the relevant fuzzy sets using MFs.

$$\begin{cases} O_i^1 = \mu_{Ai}(x) \\ O_i^2 = \mu_{Bi}(y) \end{cases} \quad (70)$$

Here, *i*= 1, 2, *x, y* are the crisp inputs to node *i*, and A_i &B_i. The linguistic label (small, large, etc.) associated with the node function. μ_{Ai}, μ_{Bi} respectively. Typically, μ_{Ai} (or μ_{Bi}) is selected as:

$$\mu_{Ai}(x) = \frac{1}{1 + \left[\left(x - {}^{c_i}/_{a_i}\right)^2\right]^{b_i}} \tag{71}$$

or

$$\mu_{Ai}(x) = exp\left\{-\left(\frac{x - c_i}{a_i}\right)^2\right\} \tag{72}$$

where, $\{a_i, b_i, c_i\}$ are the premise parameter set.

Layer 2. Rule nodes: The firing strength of each rule is calculated with mathematical multiplication. For instance,

$$O_i^2 = w_i = \mu_{A_i}(x) \cdot \mu_{B_i}(y) \tag{73}$$

Each node output represents the firing strength of a rule.

Layer 3. Average nodes: the *i*th node calculates the ratio of the *i*th rule firing strength to the summation of the firing strengths of all rules consequently, defined as:

$$O_i^3 = \bar{w}_i = \frac{w_i}{w_1 + w_2} \tag{74}$$

w_i is taken as the normalized firing strength.

Layer 4. Consequent nodes: The node function of the fourth layer calculates the contribution of each *ith* rules toward the overall output, as given:

$$O_i^4 = \bar{w}_i f_i = \bar{w}_i (p_i \;\; O_i^4 = \bar{w}_i f_i \\ = \bar{w}_i (p_i x + q_i y + r_i) \tag{75}$$

where, $\bar{w}_i$ isthe output of the ***layer 3***, and $\{p_i, q_i, r_i\}$ the parameter set.

Layer 5. *Output nodes:* The single-node computes the overall output by adding all the incoming signals from the 4th layer:

$$O_i^5 = overall\ output = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{76}$$

The final output of ANFIS can be given as:

$$\begin{aligned} f_{out} = \bar{w}_1 f_1 + \bar{w}_2 f_2 &= \frac{w_1}{w_1 + w_2} f_1 + \frac{w_2}{w_1 + w_2} f_2 \\ &= (\bar{w}_1 x) p_1 + (\bar{w}_1 y) q_1 + (\bar{w}_1) r_1 \\ &+ (\bar{w}_2 x) p_2 + (\bar{w}_2 y) q_2 + (\bar{w}_2) r_2 \end{aligned} \tag{77}$$

5.1 Designing the ANFIS-PID dynamic controller

The complete control approach that controls the dynamics of the robot mobile based on the ANFIS-PID is divided into three steps:

First step, Figure 7, represent the first step, which indicates the training stage of the ANFIS controller. The ANFIS use their adaptive and learning skills to learn and to predict the best recommended control efforts based on the already data. Which they are obtained from the designed GWO-PID controllers in the previous section.

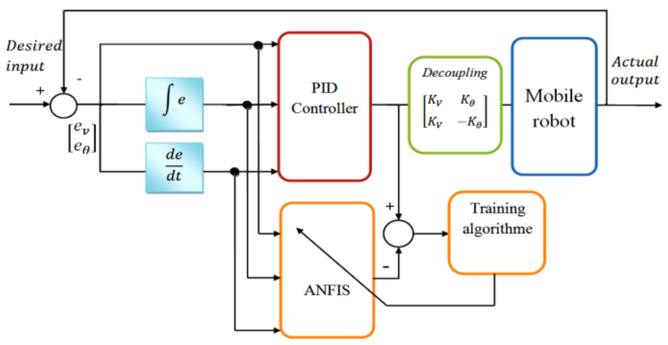
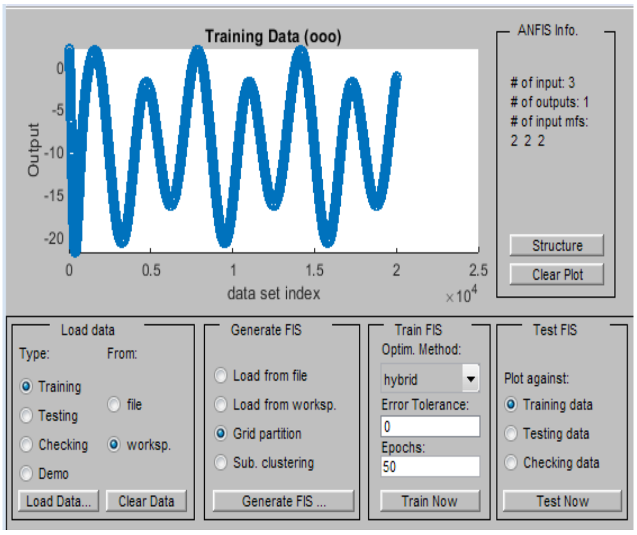


Figure 7. The ANFIS in training phase

The realized data were saved in a MATLAB/SIMULINK file. From the command anfisedit, these data were taken for training. The complete process was illustrated in Figure 8.

Figure 8 presents the internal composition of the ANFIS controller. The method used for ANFIS training is the hybrid training algorithm, with the input nodes (2, 2, 2), gaussmf membership function for the inputs and linear membership function for the output, each having 8 rules (see Figure 8.b), the epoch length was used is 50 iterations for each sample, with, 0.01s as the Simulink sampling time.

At the end of the training the result was saved as a *fis* file with respect to Sugeno-style. The root mean square error (RMSE) was found to be 0.1053 and be 0.094977 06 for ANFIS-PID1 and ANFIS-PID2, respectively.



ANFIS Info.

# of input: 3
# of outputs: 1
# of input mfs:
2 2 2

Structure | Clear Plot

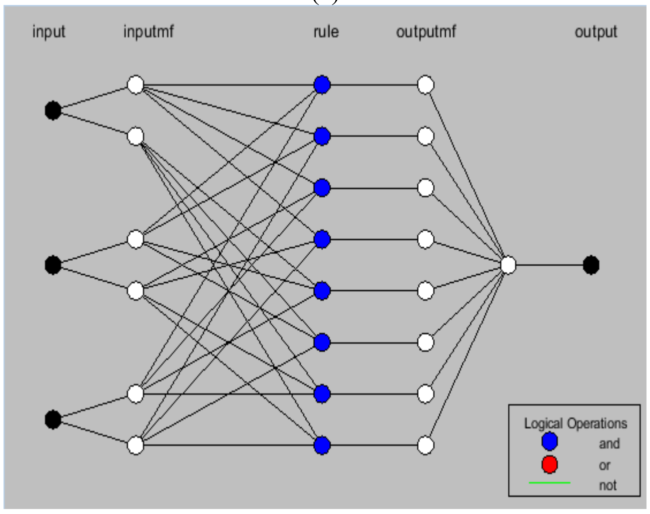
Load data
Type: [✓ Training] Testing Checking Demo
From: file [✓ worksp.]
Load Data... | Clear Data

Generate FIS
Load from file
Load from worksp.
[✓ Grid partition]
Sub. clustering
Generate FIS ...

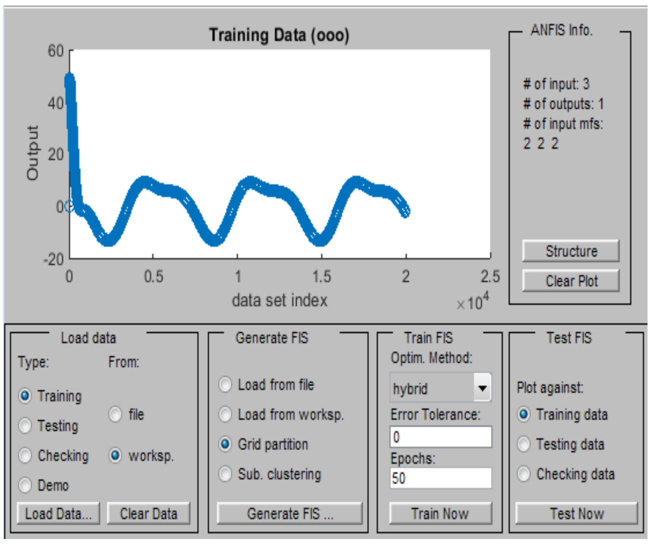
Train FIS
Optim. Method: hybrid
Error Tolerance: 0
Epochs: 50
Train Now

Test FIS
Plot against: [✓ Training data] Testing data Checking data
Test Now

(a)



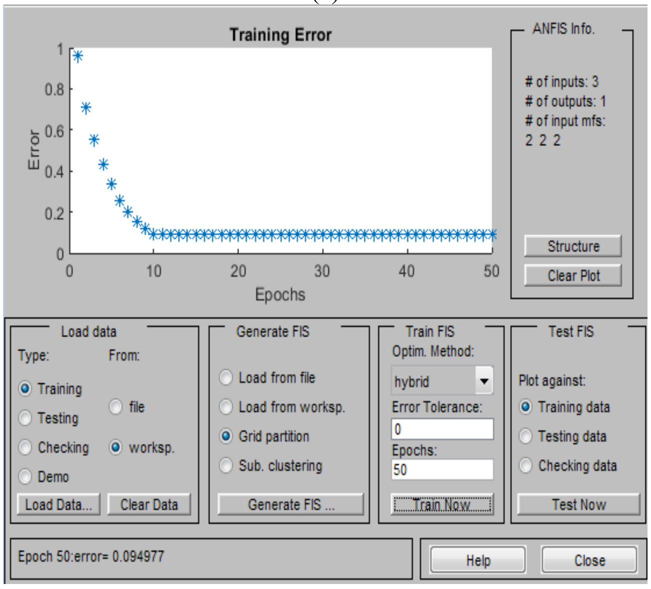
(b)



# of input: 3
# of outputs: 1
# of input mfs:
2 2 2

Type: [✓ Training] Testing Checking Demo
From: file [✓ worksp.]
Load from file / Load from worksp. / [✓ Grid partition] / Sub. clustering
Optim. Method: hybrid
Error Tolerance: 0
Epochs: 50
Plot against: [✓ Training data] Testing data Checking data

(c)



# of inputs: 3
# of outputs: 1
# of input mfs:
2 2 2

Type: [✓ Training] Testing Checking Demo
From: file [✓ worksp.]
Load from file / Load from worksp. / [✓ Grid partition] / Sub. clustering
Optim. Method: hybrid
Error Tolerance: 0
Epochs: 50
Plot against: [✓ Training data] Testing data Checking data

Epoch 50:error= 0.094977

(d)

Figure 8. (a) Data configuration for first ANFIS-PID controller, (b) structure of the ANFIS-PID controller, (c) data configuration for second ANFIS-PID controller, (d) error minimization corresponding to the second controller

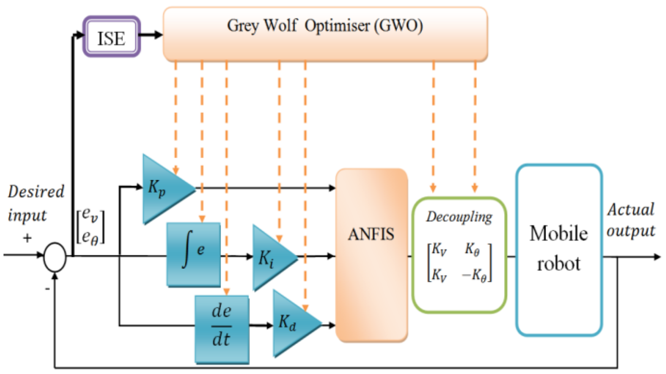

ISE
Grey Wolf Optimiser (GWO)
Desired input
$\begin{bmatrix} e_v \\ e_\theta \end{bmatrix}$
+
-
K_p
$\int e$
K_i
$\frac{de}{dt}$
K_d
ANFIS
Decoupling
$\begin{bmatrix} K_v & K_\theta \\ K_v & -K_\theta \end{bmatrix}$
Mobile robot
Actual output

Figure 9. Adjusting the parameters for the designed ANFIS-PID controller using GWO

Second step, scaling the new parameters using the GWO optimizer.

After the training step is complete and the optimal data are received, the designed ANFIS-PID controller must be developed by improving the parameters, i.e., Kp, KI, Kd, Kv, K_θ as demonstrated in Figure 9. To find the optimal values of the scaling gains, grey wolf method is proposed to guarantee a minimum ISE value.

Third step, the gained control scheme of the dynamic controller is showed in Figure 10. It reveals the hybridize between the ANFIS and PID controller, abridged as ANFIS-PID. Two trained ANFIS-PID controllers are selected for running the right and left DC motor voltage of the WMR, where a decoupling process is applied to separate the two control actions.

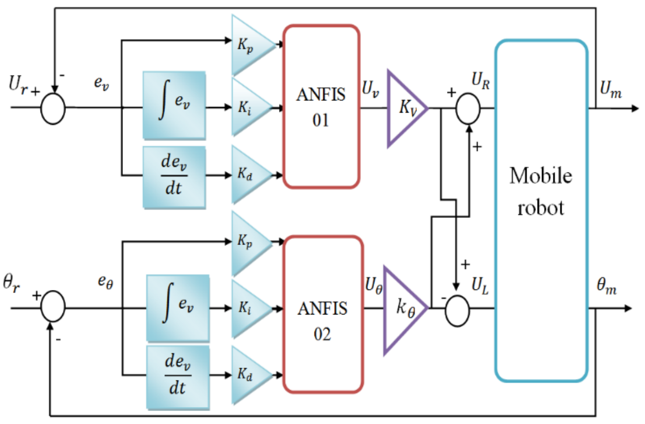


Figure 10. Bloc diagram of the final ANFIS-PID dynamic controller

The first controller takes the difference between the actual and reference speed from the right wheel as an input. The angle controller located in the left wheel studies the difference between the actual and desired orientations as an input.

Eq. (52) describe the relationship between the input voltages of the right and left DC actuators UR and UL, respectively, with the outputs of the first and second controller UV and Uθ, respectively, where the factors of the decoupling system are respectively Kv and K_θ.

$$\begin{cases} U_R = K_V \cdot U_V + K_\theta \cdot U_\theta \\ U_L = K_V \cdot U_V - K_\theta \cdot U_\theta \end{cases} \tag{78}$$

where

$$\begin{cases} e_v = U_r - U_m \\ e_\theta = \theta_r - \theta_m \end{cases} \tag{79}$$

U_r is the desired velocity, U_m the actual velocity, e_v the velocity error, θ_r the desired angle, θ_m the measured orientation, and e_θ is the orientation error.

6. RESULTS AND DISCUSSION

This study selected the use of various criteria and evaluations, a comparative study of the convergence curve performance between all controllers, the transition parameters of linear speed and of the angle for all techniques, also several cases study for the path tracking has been created, in order to confirm the over performance and high robustness of the suggested GWO-ANFIS-PID controller, In comparison to GWO-PID, PSO-PID, ALO-PID, GOA-PID, DA-PID, ABC-PID and MFO-PID.

In order to examine the convergence behavior of the nature inspired algorithms, all simulation experiments were executed more than 30 times for each method on a PC separately, the maximum number iterations were 100, and population size is set to 20.

6.1 Convergence curve of all algorithms

Figure 11 highlights the convergence curves of the seven selected optimizations algorithms for designing the PID controller in comparison to our proposed hybrid controller (GWO-ANFIS-PID).

The result proves the over-performance of the GWO-ANFIS-PID controller. The minimum ISE value attained for all approaches used in the study is shown in Table 1. From Figure 11 and Table 1, it's straightforward that the GWO optimizer gives the lowest ISE value with the faster convergence behavior compared to the other six techniques. Which were made it the best candidate for designing the ANFIS-PID controller. While the proposed GWO-ANFIS-PID controller has the lowest ISE value (0.0889).

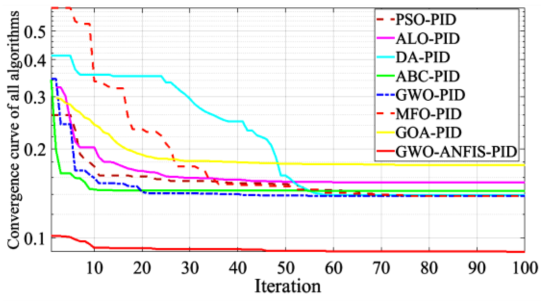


Figure 11. Convergence curve of the seven algorithms compared to GWO-ANFIS-PID

Table 1. ISE values of the five algorithms

algorithms	ISE
PSO-PID	0.138826
DA-PID	0.139292
ALO-PID	0.154018
ABC-PID	0.143813
GOA-PID	0.176242
MFO-PID	0.138489
GWO-PID	0.138510
GWO-ANFIS-PID	0.089690

6.2 Step response performance of speed controller

Figure 12 reveals the comparative analyses of the step responses performance for the linear speed that was designed with the help of the seven different techniques in comparison to the proposed GWO-ANFIS-PID controller.

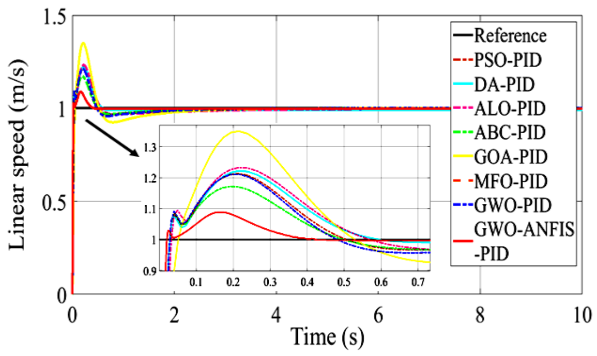


Figure 12. Step response comparison for all velocity controllers

Table 2 shows the parameter details of all meta-heuristic approaches, for the linear speed controller, and Table 3 presents the performance achieved in the time domain.

Table 2. The tuned gain details of linear velocity for all controllers

Methods	Speed controller parameter tuning				
	K_p	K_i	K_d	K_V	K_θ
PSO-PID	149.98	97.40	1.013	0.50	0.50
DA-PID	150	1	1	0.50	0.50
ALO-PID	108.46	52.24	0.72	0.50	0.50
ABC-PID	148.78	142.40	1	0.50	0.50
GOA-PID	81.60	89.97	1	0.50	0.50
MFO-PID	149.88	149.97	1	0.50	0.50
GWO-PID	150	148.94	1.001	0.50	0.50
GWO-ANFIS-PID	452.007	13.59	1.25	1.5	0.9

The best overshoot percentage (Mp %) was found by the ABC-PID, and best settling time (ts) is reached by DA-PID. where the GWO-PID controller advantage is the fast rise time and accepted settling time (ts) value obtained, but a high overshoot, this result were shared with the PSO-PID and MFO-PID.

While the GWO-ANFIS-PID controller enclose all advantages of the previous seven techniques by giving the best settling time and an excellent overshoot value and fastest rise time (tr).

6.3 Step response performance of angle controller

Figure 13 describes the comparative analyses of the angle step responses performance that was designed with the aid of the seven previous mentioned approaches in comparison to the recommended GWO-ANFIS-PID controller.

Table 3. Performance characteristic for velocity controller

Methods	Transition parameters for speed control					
	t_r	Mp%	Undershoot	Peak	t_p	t_s
PSO-PID	0.0190	21.3279	3.3018e-05	1.2133	0.2148	1.5344
DA-PID	0.0191	22.2135	3.3017e-05	1.2221	0.2148	0.5248
ALO-PID	0.0221	23.2889	3.5750e-05	1.2329	0.2148	1.9670
ABC-PID	0.0190	17.1998	3.3097e-05	1.1720	0.1933	1.3128
GOA-PID	0.0332	35.0362	7.8275e-05	1.3504	0.2148	2.0684
MFO-PID	0.0189	21.1015	3.3025e-05	1.2110	0.2148	1.4805
GWO-PID	0.0189	21.1147	3.3017e-05	1.2111	0.2148	1.4849
GWO-ANFIS-PID	0.0125	8.8853	0	1.0889	0.1700	0.2547

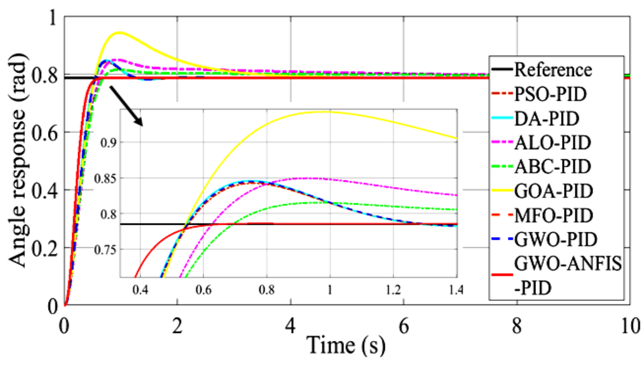


Figure 13. Comparison of step response for the orientation

From Figure 13, Tables 4 and 5, we can say that the GWO-ANFIS-PID gives a zero-overshoot value, with the best rise time (t_r for 10% → 90%) and settling time value (t_s for ±2% tolerance), were the GWO-PID comes at the second place. For that reason, our next examinations are concentrating only on analyzing the two best candidates (GWO-PID and GWO-ANFIS-PID).

Table 4. The tuned main parameters of the three controllers for angle control

Methods	**angle controller parameter tuning**				
	K_p	K_i	K_d	K_V	K_θ
PSO-PID	149.93	1	31.019	0.50	0.50
DA-PID	150	1	30.53	0.50	0.50
ALO-PID	131.90	22.60	33.21	0.50	0.50
ABC-PID	145.24	11.46	37.86	0.50	0.50
GOA-PID	149.95	112.96	41.14	0.50	0.50
MFO-PID	150	1	30.85	0.50	0.50
GWO-PID	150	1.388	30.83	0.50	0.50
GWO-ANFIS-PID	500	1.01	75.31	1.5	0.9

Table 5. Performance characteristics of all angle controllers

Methods	**Transition parameters for angle controller**					
	t_r	Mp%	Undershoot	Peak	t_p	t_s
PSO-PID	0.3539	7.2995	0	0.8423	0.7449	1.1016
DA-PID	0.3499	7.7670	0	0.8460	0.7449	1.1006
ALO-PID	0.4034	8.1941	0	0.8493	0.9259	6.0690
ABC-PID	0.4247	3.8441	0	0.8152	0.9539	3.1976
GOA-PID	0.3606	20.0551	0	0.9424	0.9826	3.3034
MFO-PID	0.3522	7.5138	0	0.8440	0.7449	1.1013
GWO-PID	0.3518	7.5961	0	0.8446	0.7449	1.1057
GWO-ANFIS-PID	0.2834	0.1047	0	0.7858	0.7700	0.4302

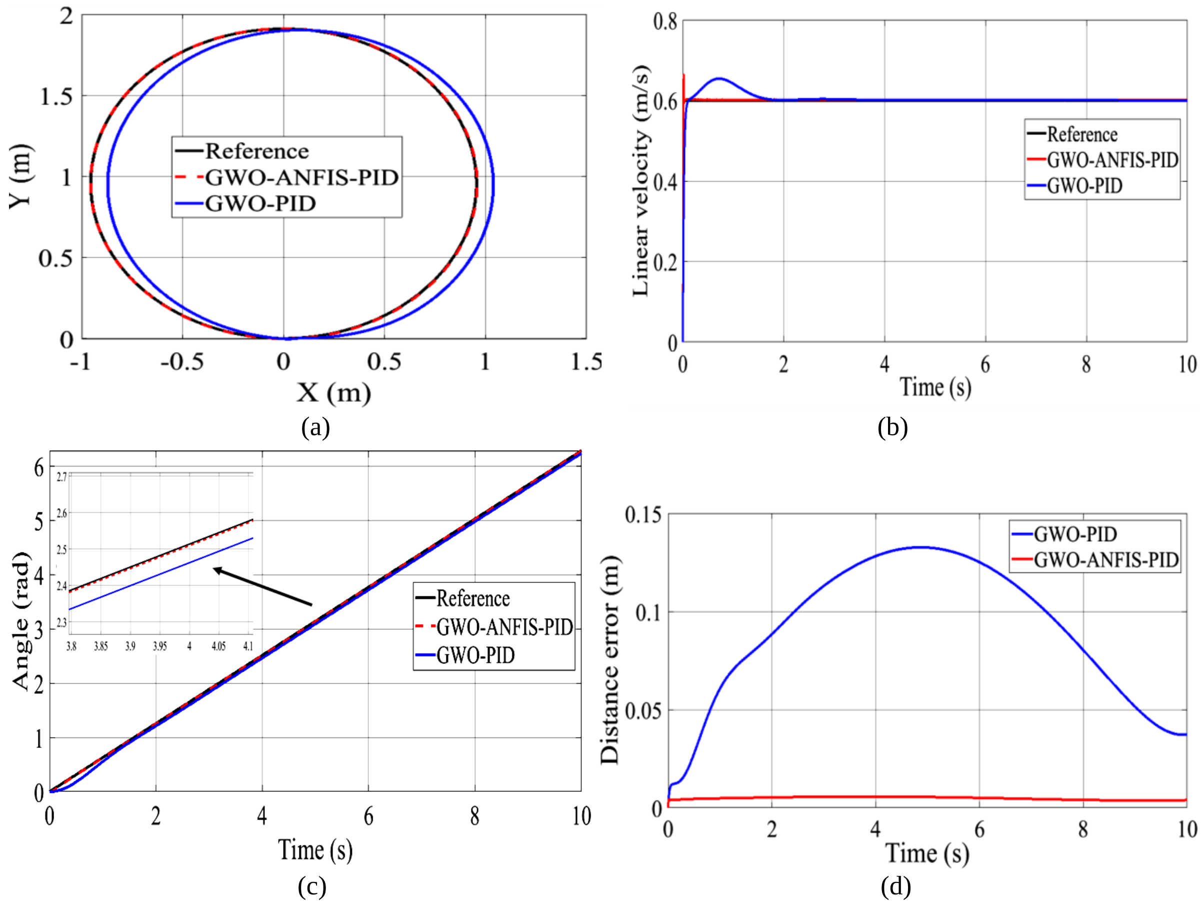


Figure 14. (a) Circle trajectory in XY plane, (b) linear velocity response for all controllers, (c) angle response of all controllers, (d) error of the distance

6.4 Circular path

This case, represent a conventional type of smooth trajectory that have been made to confirm the capacity of the GWO-ANFIS-PID controller in dealing with this kind of paths, in comparison to the GWO-PID (Figure 14).

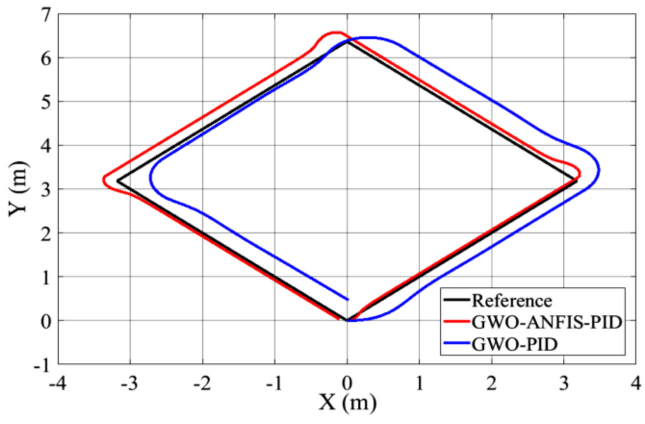
Here, a progressive variation in both linear and angular speed is applied for that purpose, results proves that the GWO-ANFIS-PID controller better handling with smooth trajectory than the GWO-PID controller, due to a minimum tracking error from the suggested controller than the GWO-PID one. Moreover, since the PID controller gives a very high overshoot in the linear velocity response at starting time. Which makes him. unsupported for handling the tracking issue.

6.5 Diamond shape trajectory

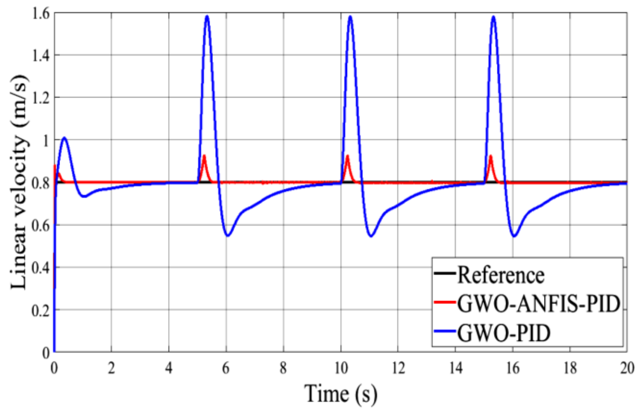
The Diamond shaped trajectory is a clear example of a non-continuous gradient path. The main problem of this type of paths is the sharp and a non-continuous motion, which calls for a serious control technique.

The spontaneous changing in velocity and due to the slow rise time and settling time, makes the GWO-PID controller gives slow actions, that causes high distance error and a very high overshoot value too from this controller, that might generate an unstable and a troubled motion, which could by the time damage the DC actuators, for that reason it's unsuitable for this kind of paths (Figure 15).

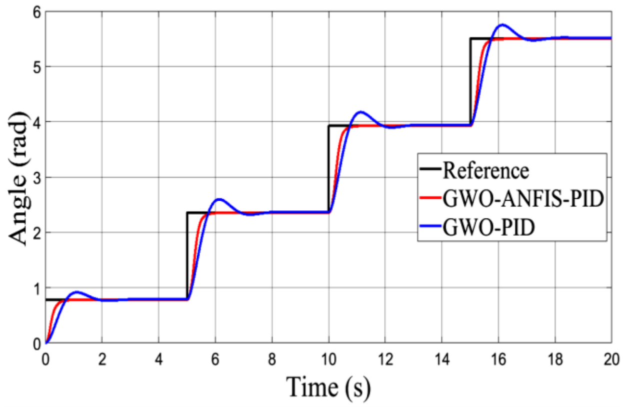
The GWO-ANFIS-PID controller has a tolerable overshoot contrasted to the GWO-PID and rapidly time response, and a very small distance error too, which makes it supported for this type of trajectory.



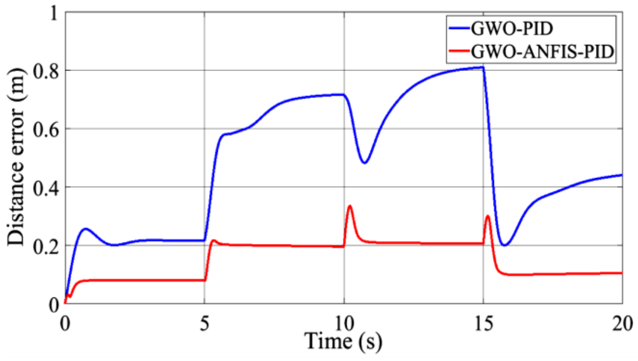
(a)



(b)



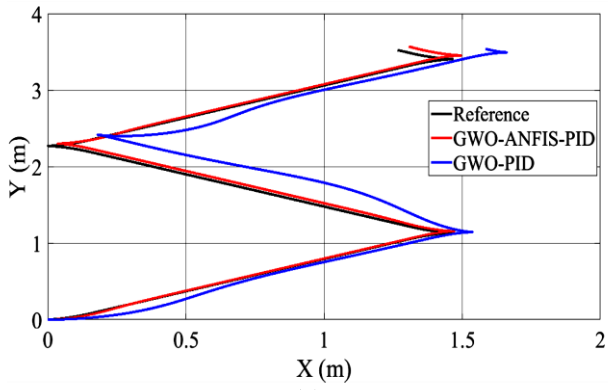
(c)



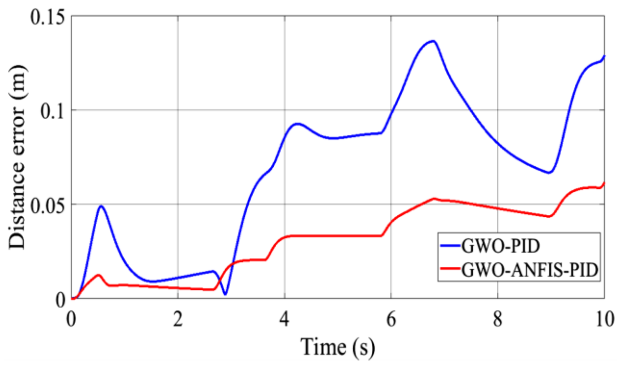
(d)

Figure 15. (a) Diamond shape trajectory in XY plane, (b) linear velocity corresponding to controllers, (c) angle response corresponding to the path for the two controllers, (d) error of the distance

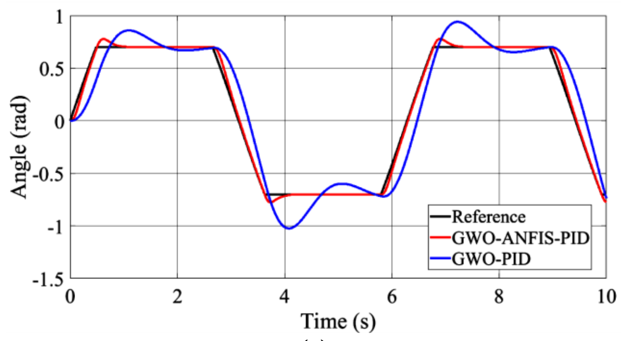
6.6 Zig-zag trajectory



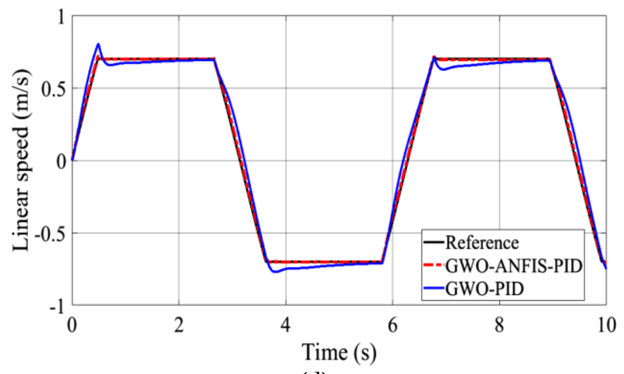
(a)



(b)



(c)



(d)

Figure 16. (a) Zigzag path in XY plane, (b) the tracking error of the controllers, (c) the theta angle response corresponding to the controllers, (d) the linear velocity corresponding to the controllers

Here a trapezoidal signal was applied, for that purpose progressively changing can be seen for both linear speed and angle, the GWO- ANFIS-PID controller gives less overshoot and better time response, contrary to the GWO-PID which gives an important overshoot and a high distance error value (Figure 16).

6.7 Adding the back-stepping controller in the external loop

The designed ANFIS-PID controller role is to guarantee a minimum linear and angular velocity error, which calls for a kinematic controller. Accordingly, a back-stepping controller was suggested to maintain a minimal distance error. The error is less than 0.002 m. To highlight the robustness of the Back-stepping combined with ANFIS-PID controller, a square and a lemniscates trajectory were preferred.

6.7.1 Case of lemniscates

In order to generate this path, the following equation were applied [70]:

$$x_R(n) = 2.5 * cos\left(2 * \pi * \frac{t}{20}\right) \tag{80}$$

$$y_R(n) = -2.5 * sin\left(2 * \pi * \frac{t}{30}\right) \tag{81}$$

$$\theta_R(n) = a\,tan2\left[\frac{\left(\frac{y_R(n) - y_R(n-1)}{t+\epsilon}\right)}{\frac{(x_R(n) - x_R(n-1))}{t+\epsilon}}\right] \tag{82}$$

where the back-stepping parameters are selected as: k_x=10, k_y = 80 and k_θ = 15.

The desired and physical paths of the lemniscates are presented in Figure 17, displayed in blue and red lines, respectively. Figure 18 (a-d) illustrate the errors of x, y, θ, and the tracking trajectory, respectively. The effect of the included kinematic controller in the external loop is indicated in Figure 18 (d), the distance error is less than 0.002 meter. For that reason, the two trajectories have exactly the same form in Figure 17.

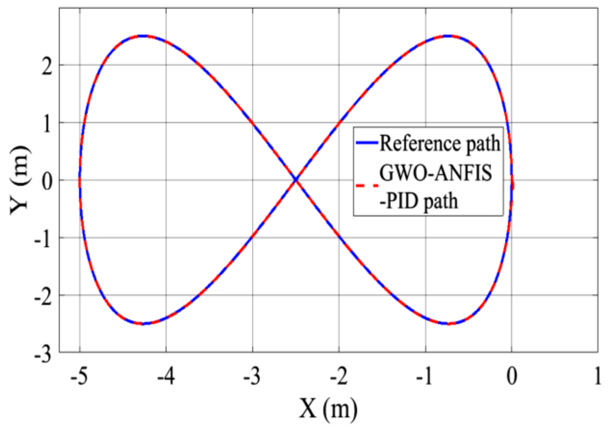
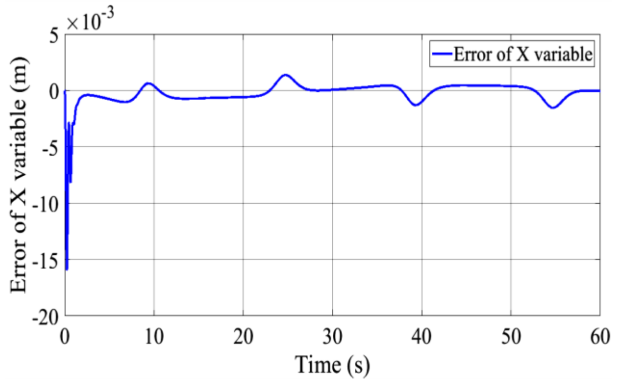
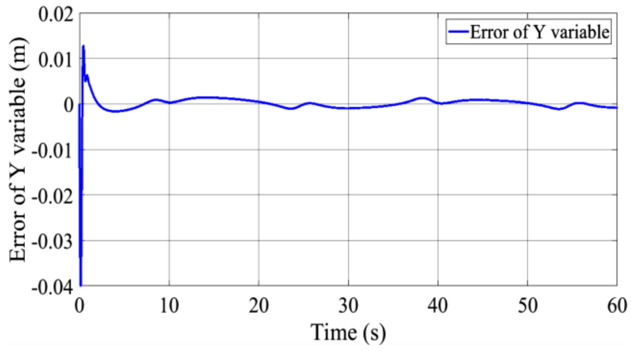


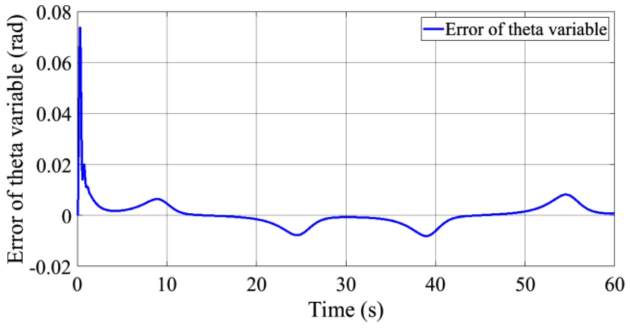
Figure 17. Lemniscate path



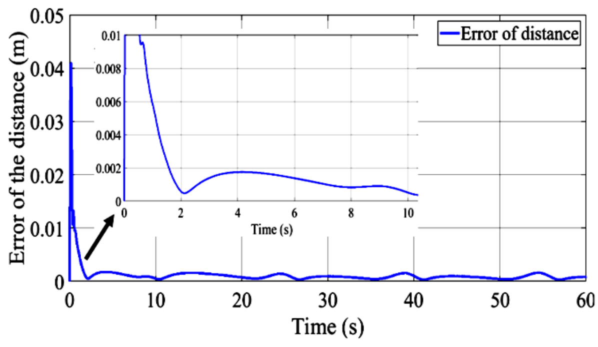
(a)



(b)



(c)



(d)

Figure 18. Error corresponding to (a) x, (b) y, (c) the theta angle, (d) the trajectory tracking

6.7.2 Case of square trajectory

The recommended and actual trajectories of the square path are shown in Figure 19, highlighted in blue and red lines, respectively, Figure 20 (a)-(c) display the errors of x, y, θ, and the tracking distance error, respectively.

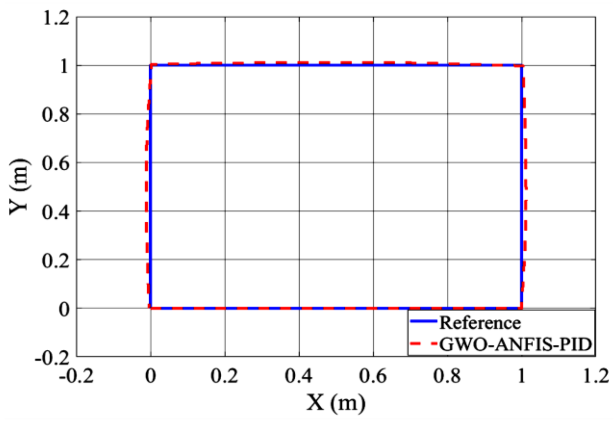
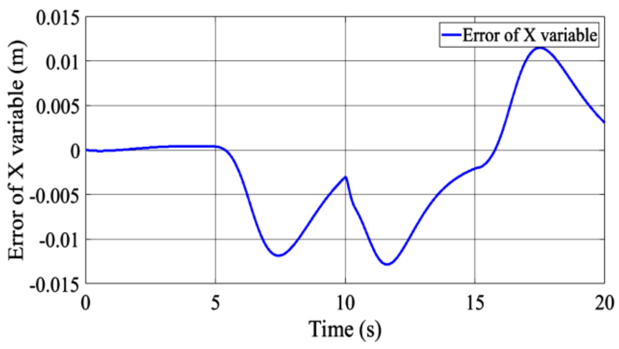
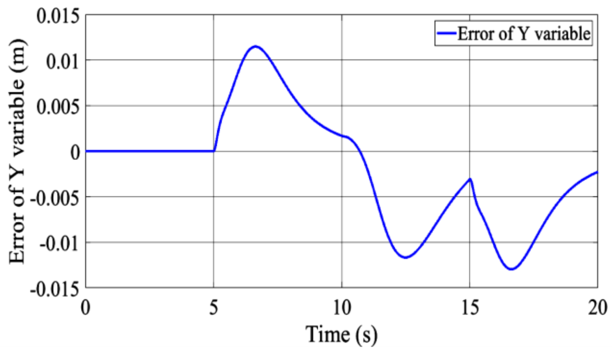


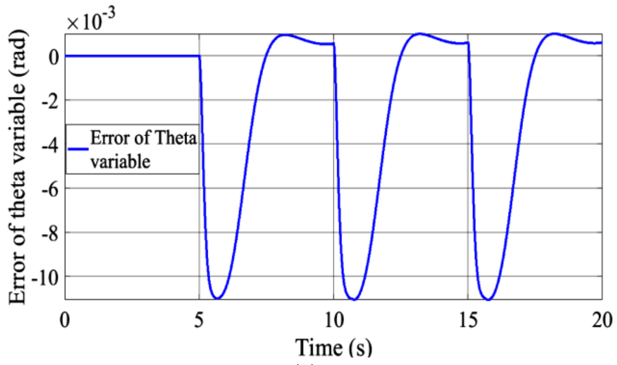
Figure 19. Square trajectory



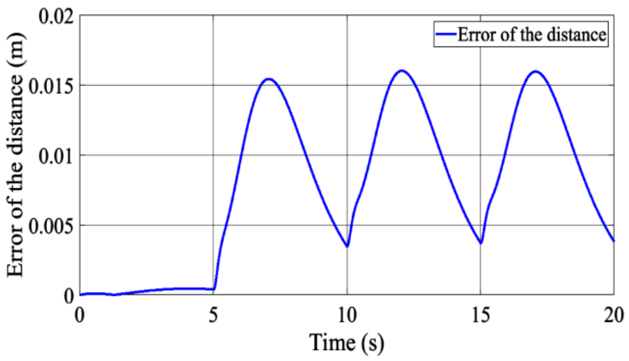
(a)



(b)



(c)



(d)

Figure 20. Error corresponding to (a) x, (b) y, (c) the theta angle, (d) the trajectory tracking

The square trajectory is classified as a sharp-shaped path too, therefore it was chosen for this study in order to justify the capability of the proposed hybrid controller, a minimum distance error reached (about 0.015 meter), and an error angle fewer than 0.01 rad, therefore we can see the nearly superposition of the actual on reference path in Figure 19.

7. CONCLUSIONS

This paper suggested a new combination technique between the PID controller and an adaptive neuro-fuzzy inference system (ANFIS), shortened to ANFIS-PID controller, to deal with the motion regulation of the wheeled mobile robot (WMR). Tuning parameter of the proposed hybrid controller are challenging tasks, a comparative study of various nature inspired methods are introduced in the study, based on the integral square error (ISE). the selected algorithms namely, particle swarm optimizer (PSO), grey wolf optimizer algorithms (GWO), dragonfly optimizer(DA), ant lion optimizer (ALO), grasshopper algorithm (GOA), moth flame optimizer (MFO), and the artificial bee colony algorithm (ABC), simulation in MATLAB environment gives a demonstration and judgment between the ANFIS-PID controller and these seven techniques ,in terms of lowest overshoot value, minimum ISE value, convergence curve, less peak, peak time, and faster rise time and settling time received. Various study cases have been made, demonstrates the ability and stability of the proposed ANFIS-PID controller tuned by GWO algorithm, which has the capacity to generate smooth and suitable control signals for both linear and angular speeds, great performance over the PID controllers.

Moreover, to guarantee a low tracking error, a back-stepping technique was applied at the kinematic control level, where a lemniscate and a square trajectory have already demonstrated the robustness of the mentioned controller in both smooth and sharp, shaped trajectory.

ACKNOWLEDGEMENT

N. Jarasthitikulchai would like to acknowledge financial support by Navamindradhiraj University through the Navamindradhiraj University Research Fund (NURF).

REFERENCES

[1] Poberznik, A., Leino, M., Huhtasalo, J., Jyräkoski, T., Valo, P., Lehtinen, T., Kortelainen, J., Merilampi, S., Virkki, J. (2021). Mobile robots and RFID technology-based smart care environment for minimizing risks related to employee turnover during pandemics. Sustain., 13(22): 12809. https://doi.org/10.3390/su132212809

[2] Saputra, R.P., Rakicevic, N., Kuder, I., Bilsdorfer, J., Gough, A., Dakin, A., de Cocker, E., Rock, S., Harpin, R., Kormushev, P. (2021). Resqbot 2.0: An improved design of a mobile rescue robot with an inflatable neck securing device for safe casualty extraction. Appl. Sci., 11(12): 5414. https://doi.org/10.3390/app11125414

[3] Catalan, J.M., Blanco, A., Bertomeu-Motos, A., Garcia-Perez, J.V., Almonacid, M., Puerto, R., Garcia-Aracil, N. (2021). A modular mobile robotic platform to assist people with different degrees of disability. Appl. Sci., 11(15): 7130. https://doi.org/10.3390/app11157130

[4] Kot, T., Novák, P. (2018). Application of virtual reality in teleoperation of the military mobile robotic system TAROS. Int. J. Adv. Robot. Syst., 15(1): 1-6. https://doi.org/10.1177%2F1729881417751545

[5] Fue, K., Porter, W., Barnes, E., Rains, G. (2020). An extensive review of mobile agricultural robotics for field operations: Focus on cotton harvesting. AgriEngineering, 2(1): 150-174. https://doi.org/10.3390/agriengineering2010010

[6] Cen, H., Singh, B.K. (2021). Nonholonomic wheeled mobile robot trajectory tracking control based on improved sliding mode variable structure. Wireless Communications and Mobile Computing, 2021: 2974839. https://doi.org/10.1155/2021/2974839

[7] Abdulwahhab, O.W., Abbas, N.H. (2018). Design and stability analysis of a fractional order state feedback controller for trajectory tracking of a differential drive robot. Int. J. Control. Autom. Syst., 16: 2790-2800. https://doi.org/10.1007/s12555-017-0234-8

[8] Mohareri, O., Dhaouadi, R., Rad, A.B. (2012). Indirect adaptive tracking control of a nonholonomic mobile robot via neural networks. Neurocomputing, 88: 54-66. https://doi.org/10.1016/j.neucom.2011.06.035

[9] Khai, T.Q., Ryoo, Y.J., Gill, W.R., Im, D.Y. (2020). Design of kinematic controller based on parameter tuning by fuzzy inference system for trajectory tracking of differential-drive mobile robot. Int. J. Fuzzy Syst., 22: 1972-1978. https://doi.org/10.1007/s40815-020-00842-9

[10] Khai, T.Q., Ryoo, Y.J. (2019). Design of adaptive kinematic controller using radial basis function neural network for trajectory tracking control of differential-drive mobile robot. Int. J. Fuzzy Log. Intell. Syst., 19(4): 349-359. http://dx.doi.org/10.5391/IJFIS.2019.19.4.349

[11] Gheisarnejad, M., Khooban, M.H. (2019). Supervised control strategy in trajectory tracking for a wheeled mobile robot. IET Collab. Intell. Manuf., 1(1): 3-9. https://doi.org/10.1049/iet-cim.2018.0007

[12] Nikranjbar, A., Haidari, M., Atai, A.A. (2018). Adaptive sliding mode tracking control of mobile robot in dynamic environment using artificial potential fields. J. Comput. Robot., 11(1): 1-14.

[13] Wu, X., Jin, P., Zou, T., Qi, Z., Xiao, H., Lou, P. (2019). Backstepping trajectory tracking based on fuzzy sliding mode control for differential mobile robots. J. Intell. Robot. Syst. Theory Appl., 96: 109-121. https://doi.org/10.1007/s10846-019-00980-9

[14] Wang, G., Zhou, C., Yu, Y., Liu, X. (2019). Adaptive sliding mode trajectory tracking control for WMR considering skidding and slipping via extended state observer. Energies, 12(17): 3305. https://doi.org/10.3390/en12173305

[15] Martins, F.N., Celeste, W.C., Carelli, R., Sarcinelli-Filho, M., Bastos-Filho, T.F. (2008). An adaptive dynamic controller for autonomous mobile robot trajectory tracking. Control Eng. Pract., 16(11): 1354-1363. https://doi.org/10.1016/j.conengprac.2008.03.004

[16] Binh, N.T., Tung, N.A., Nam, D.P., Quang, N.H. (2019). An adaptive backstepping trajectory tracking control of a tractor trailer wheeled mobile robot. Int. J. Control. Autom. Syst., 17: 465-473. https://doi.org/10.1007/s12555-017-0711-0

[17] Rossomando, F.G., Soria, C., Carelli, R. (2011). Autonomous mobile robots navigation using RBF neural compensator. Control Eng. Pract., 19(3): 215-222. https://doi.org/10.1016/j.conengprac.2010.11.011

[18] Bozek, P., Karavaev, Y.L., Ardentov, A.A., Yefremov, K.S. (2020). Neural network control of a wheeled mobile robot based on optimal trajectories. Int. J. Adv. Robot. Syst., 17(2): 1-10. https://doi.org/10.1177/1729881420916077

[19] Das, T., Kar, I.N. (2006). Design and implementation of an adaptive fuzzy logic-based controller for wheeled mobile robots. IEEE Trans. Control Syst. Technol., 14(3): 501-510. https://doi.org/10.1109/TCST.2006.872536

[20] Štefek, A., Pham, V.T., Krivanek, V., Pham, K.L. (2021). Optimization of fuzzy logic controller used for a differential drive wheeled mobile robot. Appl. Sci., 11(13): 6023. https://doi.org/10.3390/app11136023

[21] Aldair, A.A., Al-Mayyahi, A., Wang, W. (2020). Design of a stable an intelligent controller for a quadruped robot. J. Electr. Eng. Technol., 15: 817-832. https://doi.org/10.1007/s42835-019-00332-5

[22] Serradilla, F., Cañas, N., Naranjo, J.E. (2020). Optimization of the energy consumption of electric motors through metaheuristics and PID controllers. Electron., 9(11): 1-16. https://doi.org/10.3390/electronics9111842

[23] Latha, K., Rajinikanth, V., Surekha, P.M. (2013). PSO-based PID controller design for a class of stable and unstable systems. ISRN Artif. Intell., 2013: 1-11. https://doi.org/10.1155/2013/543607

[24] Mosaad, A.M., Attia, M.A., Abdelaziz, A.Y. (2019). Whale optimization algorithm to tune PID and PIDA controllers on AVR system. Ain Shams Eng. J., 10(4): 755-767. https://doi.org/10.1016/j.asej.2019.07.004

[25] Dutta, P., Nayak, S.K. (2021). Grey wolf optimizer based PID controller for speed control of BLDC motor. J.

Electr. Eng. Technol., 16(6): 955-961. https://doi.org/10.1007/s42835-021-00660-5

[26] Ekinci, S., Hekimoğlu, B., Izci, D. (2021). Opposition based Henry gas solubility optimization as a novel algorithm for PID control of DC motor. Eng. Sci. Technol. an Int. J., 24(2): 331-342. https://doi.org/10.1016/j.jestch.2020.08.011

[27] Bagis, A., Senberber, H. (2017). ABC algorithm based PID controller design for higher order oscillatory systems. Elektron. ir Elektrotechnika, 23(6): 3-9. https://doi.org/10.5755/j01.eie.23.6.19688

[28] Slama, S., Errachdi, A., Benrejeb, M. (2019). Neural adaptive PID and neural indirect adaptive control switch controller for nonlinear MIMO systems. Mathematical Problems in Engineering, 2019: 7340392. https://doi.org/10.1155/2019/7340392

[29] Ye, J. (2008). Adaptive control of nonlinear PID-based analog neural networks for a nonholonomic mobile robot. Neurocomputing, 71(7-9): 1561-1565. https://doi.org/10.1016/j.neucom.2007.04.014

[30] Pei, G., Yu, M., Xu, Y., Ma, C., Lai, H., Chen, F., Lin, H. (2021). An improved PID controller for the compliant constant-force actuator based on bp neural network and smith predictor. Appl. Sci., 11(6): 2685. https://doi.org/10.3390/app11062685

[31] Xu, S.S.D., Huang, H.C., Chiu, T.C., Lin, S.K. (2019). Biologically-inspired learning and adaptation of self-evolving control for networked mobile robots. Appl. Sci., 9(5): 1034. https://doi.org/10.3390/app9051034

[32] Mai, T.A., Dang, T.S., Duong, D.T., Le, V.C., Banerjee, S. (2021). A combined backstepping and adaptive fuzzy PID approach for trajectory tracking of autonomous mobile robots. J. Brazilian Soc. Mech. Sci. Eng., 43: 1-13. https://doi.org/10.1007/s40430-020-02767-8

[33] Tiep, D.K., Lee, K., Im, D.Y., Kwak, B., Ryoo, Y.J. (2018). Design of fuzzy-PID controller for path tracking of mobile robot with differential drive. Int. J. Fuzzy Log. Intell. Syst., 18(3): 220-228. https://doi.org/10.5391/IJFIS.2018.18.3.220

[34] Ben Jabeur, C., Seddik, H. (2021). Design of a PID optimized neural networks and PD fuzzy logic controllers for a two-wheeled mobile robot. Asian J. Control, 23(1): 23-41. https://doi.org/10.1002/asjc.2356

[35] Wang, S., Yin, X., Li, P., Zhang, M., Wang, X. (2019). Trajectory Tracking Control for Mobile Robots Using Reinforcement Learning and PID. Iran. J. Sci. Technol. - Trans. Electr. Eng., 44: 1059-1068. https://doi.org/10.1007/s40998-019-00286-4

[36] Al-Mayyahi, A., Wang, W., Birch, P. (2014). Adaptive neuro-fuzzy technique for autonomous ground vehicle navigation. Robotics, 3(4): 349-370. https://doi.org/10.3390/robotics3040349

[37] Selma, B., Chouraqui, S., Abouaïssa, H. (2020). Optimization of ANFIS controllers using improved ant colony to control an UAV trajectory tracking task. SN Appl. Sci., 2: 1-18. https://doi.org/10.1007/s42452-020-2236-z

[38] Premkumar, K., Manikandan, B.V. (2014). Adaptive neuro-fuzzy inference system based speed controller for brushless DC motor. Neurocomputing, 138: 260-270. https://doi.org/10.1016/j.neucom.2014.01.038

[39] Pang, F., Luo, M., Xu, X., Tan, Z. (2021). Path tracking control of an omni-directional service robot based on model predictive control of adaptive neural-fuzzy inference system. Appl. Sci., 11(2): 1-18. https://doi.org/10.3390/app11020838

[40] Elsisi, M., Tran, M.Q., Mahmoud, K., Lehtonen, M., Darwish, M.M.F. (2021). Robust design of ANFIS-based blade pitch controller for wind energy conversion systems against wind speed fluctuations. IEEE Access, 9: 37894-37904. https://doi.org/10.1109/ACCESS.2021.3063053

[41] Hidayat, Pramonohadi, S., Sarjiya, Suharyanto. (2013). A comparative study of PID, ANFIS and hybrid PID-ANFIS controllers for speed control of Brushless DC Motor drive. Proceeding - 2013 Int. Conf. Comput. Control. Informatics Its Appl. "Recent Challenges Comput. Control Informatics", IC3INA, pp. 117-122. https://doi.org/10.1109/IC3INA.2013.6819159

[42] Pal, D., Bhagat, S.K. (2020). Design and analysis of optimization based integrated ANFIS- PID controller for networked controlled systems (NCSs). Cogent Eng., 7(1): 1-21. https://doi.org/10.1080/23311916.2020.1772944

[43] Guo, Y., Mohamed, M.E.A. (2020). Speed control of direct current motor using ANFIS based hybrid P-I-D configuration controller. IEEE Access, 8: 125638-125647. https://doi.org/10.1109/ACCESS.2020.3007615

[44] Housny, H., Chater, E.A., El Fadil, H. (2020). Multi closed-loop adaptive neuro-fuzzy inference system for quadrotor position control. Adv. Sci. Technol. Eng. Syst., 5(5): 526-535. https://doi.org/10.25046/AJ050565

[45] Chaudhary, H., Panwar, V., Sukavanam, N., Prasad, R. (2014). ANFIS PD+I based hybrid force/ position control of an industrial robot manipulator. Int. J. Mater. Mech. Manuf., 2(2): 107-112. https://doi.org/10.7763/ijmmm.2014.v2.110

[46] Fierro, R., Lewis, F.L. (1995). Control of a nonholonomic mobile robot: backstepping kinematics into dynamics. Proc. IEEE Conf. Decis. Control, 4: 3805-3810. https://doi.org/10.1109/cdc.1995.479190

[47] Demir, B.E., Bayir, R., Duran, F. (2016). Real-time trajectory tracking of an unmanned aerial vehicle using a self-tuning fuzzy proportional integral derivative controller. Int. J. Micro Air Veh., 8: 252-268. https://doi.org/10.1177/1756829316675882

[48] Şenel, F.A., Gökçe, F., Yüksel, A.S., Yiğit, T. (2019). A novel hybrid PSO–GWO algorithm for optimization problems. Eng. Comput., 35: 1359-1373. https://doi.org/10.1007/s00366-018-0668-5

[49] Singh, N., Singh, S.B. (2017). Hybrid algorithm of particle swarm optimization and grey wolf optimizer for improving convergence performance. J. Appl. Math., 2017: 2030489. https://doi.org/10.1155/2017/2030489

[50] Mirjalili, S. (2016). Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Comput. Appl., 27: 1053-1073. https://doi.org/10.1007/s00521-015-1920-1

[51] Reynolds, C.W. (1987). Flocks, herds, and schools: A distributed behavioral model. Proc. 14th Annu. Conf. Comput. Graph. Interact. Tech. SIGGRAPH 1987, 21: 25-34. https://doi.org/10.1145/37401.37406

[52] Yasen, M., Al-Madi, N., Obeid, N. (2018). Optimizing neural networks using dragonfly algorithm for medical prediction. 2018 8th Int. Conf. Comput. Sci. Inf. Technol. CSIT, pp. 71-76. https://doi.org/10.1109/CSIT.2018.8486178

[53] Salam, M.A., Zawbaa, H.M., Emary, E., Ghany, K.K.A., Parv, B. (2016). A hybrid dragonfly algorithm with extreme learning machine for prediction. Proc. 2016 Int. Symp. Innov. Intell. Syst. Appl. INISTA. https://doi.org/10.1109/INISTA.2016.7571839

[54] Mirjalili, S. (2015). The ant lion optimizer. Adv. Eng. Softw., 83: 80-98. https://doi.org/10.1016/j.advengsoft.2015.01.010

[55] Saha, S., Mukherjee, V. (2018). A novel quasi-oppositional chaotic antlion optimizer for global optimization. Appl. Intell., 48: 2628-2660. https://doi.org/10.1007/s10489-017-1097-7

[56] Gupta, E., Saxena, A. (2016). Performance evaluation of antlion optimizer based regulator in automatic generation control of interconnected power system. J. Eng. (United Kingdom). https://doi.org/10.1155/2016/4570617

[57] Karaboga, D., Basturk, B. (2008). On the performance of artificial bee colony (ABC) algorithm. Appl. Soft Comput. J., 8(1): 687-697. https://doi.org/10.1016/j.asoc.2007.05.007

[58] Yu, X., Chen, W., Zhang, X. (2018). An artificial bee colony algorithm for solving constrained optimization problems. Proc. 2018 2nd IEEE Adv. Inf. Manag. Commun. Electron. Autom. Control Conf. IMCEC, pp. 2663-2666. https://doi.org/10.1109/IMCEC.2018.8469371

[59] Mirjalili, S., Mirjalili, S.M., Lewis, A. (2014). Grey wolf optimizer. Adv. Eng. Softw., 69: 46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007

[60] Saremi, S., Mirjalili, S., Lewis, A. (2017). Grasshopper optimisation algorithm: Theory and application. Adv. Eng. Softw., 105: 30-47. https://doi.org/10.1016/j.advengsoft.2017.01.004

[61] Seifi, A., Ehteram, M., Singh, V.P., Mosavi, A. (2020). Modeling and uncertainty analysis of groundwater level using six evolutionary optimization algorithms hybridized with ANFIS, SVM, and ANN. Sustain., 12(10): 4023. https://doi.org/10.3390/SU12104023

[62] Bhuyan, M., Barik, A.K., Das, D.C. (2020). GOA optimised frequency control of solar-thermal/sea-wave/biodiesel generator based interconnected hybrid microgrids with DC link. Int. J. Sustain. Energy, 39(7): 615-633. https://doi.org/10.1080/14786451.2020.1741589

[63] Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. Knowledge-Based Syst., 89: 228-249. https://doi.org/10.1016/j.knosys.2015.07.006

[64] Sayed, G.I., Hassanien, A.E. (2018). A hybrid SA-MFO algorithm for function optimization and engineering design problems. Complex Intell. Syst., 4: 195-212. https://doi.org/10.1007/s40747-018-0066-z

[65] Singh, R.K., Gangwar, S., Singh, D.K., Pathak, V.K. (2019). A novel hybridization of artificial neural network and moth-flame optimization (ANN–MFO) for multi-objective optimization in magnetic abrasive finishing of aluminium 6060. J. Brazilian Soc. Mech. Sci. Eng., 41: 1-19. https://doi.org/10.1007/s40430-019-1778-8

[66] Jang, J.S.R. (1993). ANFIS: Adaptive-network-based fuzzy inference system. IEEE Trans. Syst. Man Cybern., 23(3): 665-685. https://doi.org/10.1109/21.256541

[67] Buragohain, M., Mahanta, C. (2008). A novel approach for ANFIS modelling based on full factorial design. Appl. Soft Comput. J., 8(1): 609-625. https://doi.org/10.1016/j.asoc.2007.03.010

[68] Bektas Ekici, B., Aksoy, U.T. (2011). Prediction of building energy needs in early stage of design by using ANFIS. Expert Syst. Appl., 38(5): 5352-5358. https://doi.org/10.1016/j.eswa.2010.10.021

[69] Boyacioglu, M.A., Avci, D. (2010). An adaptive network-based fuzzy inference system (ANFIS) for the prediction of stock market return: The case of the Istanbul stock exchange. Expert Syst. Appl., 37(12): 7908-7912. https://doi.org/10.1016/j.eswa.2010.04.045

[70] Mohamed, M.J. (2018). Design a fuzzy PID controller for trajectory tracking of mobile robot. Eng. Technol. J., 36(1): 100-110. https://doi.org/10.30684/etj.36.1a.15