International Journal of Electrical and Computer Engineering (IJECE)
Vol. 13, No. 2, April 2023, pp. 1449~1462
ISSN: 2088-8708, DOI: 10.11591/ijece.v13i2.pp1449-1462

Path tracking control of differential drive mobile robot based on chaotic-billiards optimization algorithm

Reham H. Mohammed[1], Mohamed E. Dessouki[2,3], Basem E. Elnaghi[4]
[1]Electrical Engineering Department, Faculty of Engineering, Suez Canal University, Ismailia, Egypt
[2]Electrical Engineering Department, Faculty of Engineering, King Abdulaziz University, Rabigh, Saudi Arabia
[3]Electrical Engineering Department, Faculty of Engineering, Port Said University, Port Said, Egypt
[4]Electrical power and machines Department, Faculty of Engineering, Suez Canal University, Ismailia, Egypt

Article Info

Article history:

Received May 30, 2022
Revised Sep 25, 2022
Accepted Oct 20, 2022

Keywords:

Ant colony optimization algorithm
Chaotic-billiards optimizer algorithm
Differential drive mobile robots

ABSTRACT

Mobile robots are typically depending only on robot kinematics control. However, when high-speed motions and highly loaded transfer are considered, it is necessary to analyze dynamics of the robot to limit tracking error. The goal of this paper is to present a new algorithm, chaotic-billiards optimizer (C-BO) to optimize internal controller parameters of a differential-drive mobile robot (DDMR)-based dynamic model. The C-BO algorithm is notable for its ease of implementation, minimal number of design parameters, high convergence speed, and low computing burden. In addition, a comparison between the performance of C-BO and ant colony optimization (ACO) to determine the optimum controller coefficient that provides superior performance and convergence of the path tracking. The ISE criterion is selected as a fitness function in a simulation-based optimization strategy. For the point of accuracy, the velocity-based dynamic compensation controller was successfully integrated with the motion controller proposed in this study for the robot's kinematics. Control structure of the model was tested using MATLAB/Simulink. The results demonstrate that the suggested C-BO, with steady state error performance of 0.6 percent compared to ACO's 0.8 percent, is the optimum alternative for parameter optimizing the controller for precise path tracking. Also, it offers advantages of quick response, high tracking precision, and outstanding anti-interference capability.

Corresponding Author:

Mohamed E. Dessouki
Department of Electrical Engineering, Faculty of Engineering, King Abdulaziz University
Rabigh 21911, Saudi Arabia
Email: dessouky_m@yahoo.com

1. INTRODUCTION

Wheeled mobile robots (WMRs) have become more prevalent for jobs that are too risky or time-consuming for humans. It incorporates numerous technologies such as actuators, electronic components, sensors, controller structure, communication, and data processing [1]. The controller plays important role in the design and enhancement of mobile robots, because the working capacity of a mobile robot relies on the execution of the controller. Mobile robot's stabilization and tracking control is a significant challenge. A study of the non-holonomic WMR kinematic error model has shown that a control scheme with a combination of feed-forward and a feedback control law can be used for path tracking control [2]. Furthermore, owing to the presence of uneven ground, effective path-tracking control techniques with disturbance rejection ability are vital and essential for WMR to accomplish mission objectives. Because a mobile robot is required to move at a specific time and location, tracking control is essential [3].

Recently, numerous controllers have been proposed for controlling WMR. Some of the controllers were developed based on the kinematics of mobile robots. In any case, when executing tasks that demand high-speed motion and/or heavy load transfer, it is necessary to take into account the robot dynamics in addition to its kinematics. As a result, some research has been conducted on the structure of controllers that depend on the robot dynamics. For example, Ben and Seddik [4], presented a robust adaptive controller with disturbances and non-modeled dynamics reliant on neural networks and fuzzy logic-proportional derivative (PD) controllers and then generated torque control for mobile robots. Das and Kar [5] presented adaptive controller based on fuzzy logic, where fuzzy logic evaluates the system uncertainty and the system parameters are tuned online. The actuator dynamics were included in the dynamic model, and the generated commands were the voltages. In [6], [7], various forms of trajectory-tracking controllers primarily depend on robot dynamics were developed. Signals of control by some of the dynamic controllers shown in literature, as in a significant number of the aforementioned studies, are voltages or torques for robot motors, although business robots usually receive velocity commands. For such a unique situation, De La Cruz and Carelli [8] presented an improvement in a dynamic model that utilized both angular and linear velocities as inputs. Khai *et al.* [9] presented a fuzzy inference system for tuning the parameters of the kinematic controller of differential-drive mobile robot (DDMR) without considering a dynamic model. Ammar and Azar [10] proposed a fractional-order PID (FOPID) for path tracking control of mobile robots. Azizi *et al.* [11] designed a nonlinear model predictive control (NMPC) strategy based on the derived mathematical model for motion planning and control of an omnidirectional mobile robot in dynamic environments. Chen *et al.* [12] proposed a knowledge-based neural fuzzy controller (KNFC) for mobile robot navigation control where the parameter of KNFC adjusted by knowledge-based cultural multi-strategy differential evolution (KCMDE). Li *et al.* [13] proposed fuzzy-torque approximation-enhanced sliding mode control for lateral stability of mobile robot. Xie *et al.* [14] presented coupled fractional-order sliding mode control for trajectory tracking control of a four-wheeled steerable mobile robot with the ability of collision avoidance.

One of the limitations of previous dynamic controllers is the inability to effectively choose an appropriate arrangement of controller gains to improve robot performance. Most importantly, it is critical to characterize the meaning of "good performance", and how it can be processed for the determination of the optimal setting of controller factors. The utilization of evolutionary algorithms for solving optimization challenges, such as controller gain settings, has been extensively investigated [15]. The question of choosing the gains of the controller is presented in [16], where the authors proposed evolutionary optimization techniques for adjusting PI controller parameters. Following a similar idea, Sathiya and Chinnadurai [17] proposed an evolutionary algorithm-based multi-objective effective mobile robot trajectory tracking. Sathiya's study has proven that system efficiency can be enhanced using evolutionary optimization strategies.

According to the no-free lunch idea, the meta-heuristic optimization approaches able to solve certain problems but not others. At this time, the use of a revolutionary meta-heuristic algorithm-depend on ideal control strategy to obtain accurate and satisfied results. This shows that the main motive for utilizing the chaotic-billiards optimizer (C-BO) algorithm for tuning the gain parameter of the mobile robot's model controller. The billiards optimizer algorithm (BOA) is a revolutionary meta-heuristic optimization algorithm influenced by the famous games of billiard. In 2020, Kaveh *et al.* [18] presented it. Each strategy is represented in the BOA by a multi-dimensional billiard game, with the pockets representing the best solutions obtained. If the ball collides with the other balls, the other ball positions in the strategy of the optimization are obtained using vector algebra and conserve principles. The BOA is utilized to successfully resolve seven limited engineering standard issues and twenty-three mathematical operations [18]. The C-BO method incorporates chaotic logistic maps with the BOA to improve the algorithm's total output. In meta-heuristic strategies, initialization procedure starts at random based on random choice of the population and initial values, and this cannot cause a significant starting technique in the optimization strategy. As a result, selecting the correct initial conditions can help them perform better overall. Chaotic logistic maps are being used to reorder the agents in order to improve the initialization process. As a result, the authors add a chaotic element to the BOA. The C-BO algorithm differs from other algorithms in that it requires fewer parameters to design, is simple to create, has a low computation complexity, has a fast speed convergence, and can tackle a variety of optimization techniques in various engineering fields.

This paper presents a proposed C-BO approach for locating the appropriate controller coefficients for perfect path-tracking control with superior performance. At a rapid speed convergence, the C-BO algorithm tune the coefficients used in the controller. The ISE criterion is used as a fitness function in a simulation-based optimization strategy. Ant colony optimization is a search technique based on ant foraging behavior. It has been utilized to solve numerous types of optimization problems such as vehicle routing, as reported by Dorigo and Gambardella [19]. The ant colony optimization (ACO) algorithm has the ability to solve difficult combinatorial optimization challenges. The utilization of ACO has contributed to numerous investigations of robot path planning [20]–[22].

The paper is organized as follows: section 2 presents the configuration, kinematic, and dynamic model of the mobile robot with differential drive, as well as the development details of the kinematic and dynamic controllers; section 3 present the optimization techniques C-BO and ACO and the tuning process based on them; section 4 provides simulation results as well as a comparison of system performance for the optimization techniques used. Finally, section 5 presents the conclusions of the work carried are highlighted. The dynamic model control of DDMR based C-BO algorithm is a new contribution as it has not been described in robotic systems.

2. METHOD

Mobile robots have become more prevalent for jobs that are too risky or time-consuming for humans. The controller strategy plays important role in the design and enhancement of mobile robots. The configuration of a mobile robot depending on control strategy with a differential drive is presented in this study.

2.1. Differential drive mobile robot model

The dynamic model is written as a function of torque which is similar to a traditional dynamic equation. Figure 1 presents the variables of interest the DDMR. Where *G* is center of the mass, and ω and *u* are the angular and linear velocities, respectively. Ψ is the orientation of the robot; *h* is the point of interest with coordinates *x* and *y* in the XY plane; *a* is the separation between the point of interest and the point in the middle of the virtual axle that links the traction wheels (point B); *d* is the distance from the points of contact of the traction wheels to the floor; and *b* is the distance from two points G and B. A complete analysis of this mathematical model is presented in [23]. As illustrated in the literature, the entire mathematical model of the robot is written as a kinematic and dynamic model. The kinematic model can be described by (1).

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\varphi}\end{bmatrix}=\begin{bmatrix}cos\varphi & -asin\varphi\\ sin\varphi & acos\varphi\\ 0 & 1\end{bmatrix}\begin{bmatrix}u\\ \omega\end{bmatrix}+\begin{bmatrix}\delta x\\ \delta y\\ 0\end{bmatrix} \quad (1)$$

The dynamic model can be expressed by (2):

$$\begin{bmatrix}\dot{u}\\ \dot{\omega}\end{bmatrix}=\begin{bmatrix}\frac{\theta_3}{\theta_1}\omega^2-\frac{\theta_4}{\theta_1}u\\ -\frac{\theta_5}{\theta_2}u\omega-\frac{\theta_6}{\theta_2}\omega\end{bmatrix}+\begin{bmatrix}\frac{1}{\theta_1} & 0\\ 0 & \frac{1}{\theta_2}\end{bmatrix}\begin{bmatrix}u_{ref}\\ \omega_{ref}\end{bmatrix}+\begin{bmatrix}\delta_u\\ \delta_\omega\end{bmatrix} \quad (2)$$

where $\theta = [\theta_1\ \theta_2\ \theta_3\ \theta_4\ \theta_5\ \theta_6]^T$ is vector of the model parameters (identified) and δx, δy, δu, and $\delta\omega$ are parameter uncertainty related to mobile robot. The parameters vector of θ are functions of the robot's physical features, such as moment of inertia I_z at *G*, mass *m*, and resistance of motors *Ra*, electromotive motor constant *kb*, friction coefficient *Be*, the torque constant ka, the inertia moment of each group *Ie* rotor reduction gear-wheel, the wheels radius *r*, and the distances d and b as shown in Figure 1.

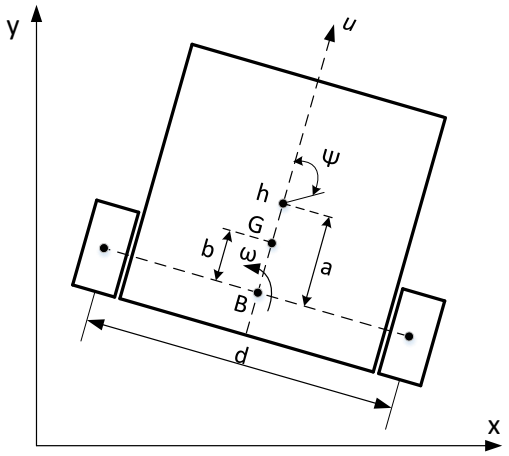

Figure 1. Mobile robot model

The robot servo motors are considered to have PD controllers to control the motor velocity, with proportional gains $kPT > 0$ and $kPR > 0$, and gains derivative $kDT \geq 0$ and $kDR \geq 0$. It is also considered that the both driven wheels' motors have the same characteristics, and negligible inductances. It should be noticed that $\theta i > 0$ for $i = 1,2,4,6$. The parameters $\theta 3$ and $\theta 5$ can be negative and null if the center of mass G is located directly in the middle of the imaginary axis connecting the traction wheels (point B), i.e. $b = 0$. However, in this study, it is assumed that $b \neq 0$. The equations for the parameters θi were firstly presented in [8], and are:

$$\theta_1 = \left[\frac{R_a}{K_a}(mR_t r + 2I_e) + 2rk_{DT}\right]\frac{1}{2rk_{PT}},$$
$$\theta_2 = \frac{\left[\frac{R_a}{k_a}\left(I_e d^2 + 2R_t r\left(I_z + mb^2\right)\right) + 2rdk_{DR}\right]}{(2rdk_{PR})},$$
$$\theta_3 = \frac{R_a}{k_a}\frac{mbR_t}{2k_{PT}}, \quad \theta_4 = \frac{R_a}{k_a}\left(\frac{k_a k_b}{R_a} + B_e\right)\frac{1}{rk_{PT}} + 1, \quad \theta_5 = \frac{R_a}{k_a}\frac{mbR_t}{dk_{PR}} \quad and$$
$$\theta_6 = \frac{R_a}{K_a}\left(\frac{k_a k_b}{R_a} + B_e\right)\frac{d}{2rk_{PR}} + 1, \tag{3}$$

2.2. Control strategy

The controller was built such that it was dependent on the robot model. The controller was divided into two parts. The first part relies on inverse kinematics, and the second part relies on robot dynamics. Figure 2 depicts the control scheme.

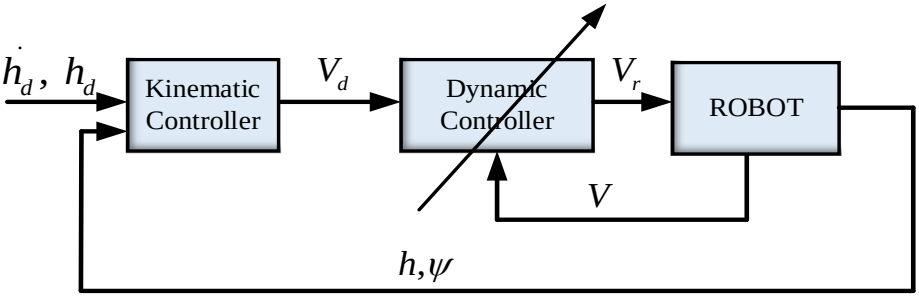

Figure 2. The structure of the proposed control system

As illustrated in Figure 2, the controller kinematic accepts the desired values of position $h_d = [x_d\ y_d]^T$ and velocity $\dot{h}_d$. The controller kinematic then calculates the intended robot velocities $V_d = [u_d\ \omega_d]^T$ depending on such values as well as the real position $h = [x\ y]^T$ and orientation of the robot Ψ. The dynamic controller receives the required velocities as well as the actual robot velocity $V = [u\ \omega]^T$. The controller uses this data, coupled with estimations of the robot parameters, to generate the actual velocity command $V_r = \left[u_{ref}\ \omega_{ref}\right]^T$ which are provided to the robot's inner controller as references.

2.3. Kinematic controller

The kinematic model formalism was introduced in detail in [24]. The kinematic controller depends on the robot kinematic model which is given by (1). The controller kinematic law used in this case is:

$$\begin{bmatrix} u_d \\ \omega_d \end{bmatrix} = \begin{bmatrix} cos\varphi & sin\varphi \\ -\frac{1}{a}sin\varphi & \frac{1}{a}cos\varphi \end{bmatrix} + \begin{bmatrix} \dot{x_d} + l_x tanh(\frac{k_x}{l_x}\tilde{x}) \\ \dot{y_d} + l_y tanh(\frac{k_y}{l_y}\tilde{y}) \end{bmatrix} \tag{4}$$

where $a > 0$; $K_x > 0$, and $K_y > 0$ are controller gains. Also $h_e = h_d - h$ is errors position vector; and l_x and l_y are the saturation constants. The $tanh$ terms are incorporated to restrict the values of the ideal velocities V_d to maintain saturation of the actuators of the robot when the errors of the position are too large [25].

2.4. Dynamic controller

To simulate and design the motion control algorithms of a mobile robot, information about the dynamic model must be known. The proposed model includes the dynamics of the robot actuators that is not

case for the model that uses torques as inputs [23], [24]. Expression (2) can be expressed in a simple format by rearranging the terms as (5):

$$\Delta + H\dot{V}' + C(V')V' + F(V')V' = V_r \tag{5}$$

where $V' = [iU\ \omega]^T$ is the vector of modified velocities given by $V' = \begin{bmatrix} i & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} u \\ \omega \end{bmatrix}$. The value of i=1 rad²/s. The *H*, *C(v')* and *F(v')* matrices, as well as Δ vector are as below.

$$H = \begin{bmatrix} \frac{\theta_1}{i} & 0 \\ 0 & \theta_2 \end{bmatrix}, F(V') = \begin{bmatrix} \frac{\theta_4}{i} & 0 \\ 0 & \theta_6 + (\frac{\theta_5}{i} - \theta_3)iU \end{bmatrix}, C(V') = \begin{bmatrix} 0 & -\theta_3\omega \\ \theta_3\omega & 0 \end{bmatrix}, \Delta = \begin{bmatrix} -\theta_1 & 0 \\ 0 & -\theta_2 \end{bmatrix}\begin{bmatrix} \delta_u \\ \delta_\omega \end{bmatrix}$$

As shown in Figure 2, the dynamic controller receives the references of the kinematic controller for angular and linear velocities V_d and produces a new set of angular and linear velocity commands V_r for the robot. The modified desired vector of velocities V_d' is defined as $V_d' = \begin{bmatrix} i & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} u_d \\ \omega_d \end{bmatrix}$. The velocity error vector is given by $\tilde{V}' = V_d' - V'$. Expression (5) is expressed in its linear parameterization form to design the dynamic controller. The new expression for the reference velocity vector is as in (6). The uncertainty vector is ignored in this case. The suggested control law for parametric uncertainty is as in (7):

$$V_r = G'\theta = \begin{bmatrix} \dot{u} & 0 & -\omega^2 & u & 0 & 0 \\ 0 & \dot{\omega} & 0 & 0 & u\omega & \omega \end{bmatrix}\theta \tag{6}$$

$$V_r = \hat{H}\left(\dot{V}_d' + T(\tilde{V}')\right) + \hat{C}V_d' + \hat{F}V_d' \tag{7}$$

where $\hat{H}, \hat{C}$, and $\hat{F}$ are estimations of *H*, *C*, and *F*, respectively, $T(\tilde{V}') = \begin{bmatrix} l_u & 0 \\ 0 & l_\omega \end{bmatrix}\begin{bmatrix} \tanh(\frac{k_u}{l_u}I\tilde{U}) \\ \tanh(\frac{k_\omega}{l_\omega}\tilde{\omega}) \end{bmatrix}$, $k_u > 0$ and $k_\omega > 0$ are gain constants, $l_u \in \mathbb{R}$ and $l_\omega \in \mathbb{R}$ are constants of saturation, and $\tilde{\omega} = \omega_d - \omega$, $\tilde{u} = u_d - u$ are the current velocity errors. $T(\tilde{V}')$ gives saturation to make sure that the robot's commands are always within the physical boundaries set by the robot, assuming that V_D' and $\dot{V}_D'$ are bounded to specific values. When the dynamic variables are not accurately described or changed, an updated control law is applied. The control law is stated in its linear parameterized form to develop the updated law.

$$V_r = G\hat{\theta} = \begin{bmatrix} \delta 1 & 0 & -\omega_d\omega & u_d & 0 & 0 \\ 0 & \delta_2 & (iu_d\omega - iu\omega_d) & 0 & u\omega_d & \omega_d \end{bmatrix}\hat{\theta} \tag{8}$$

where $\delta_1 = \dot{u}_d + l_u \tanh\left(\frac{k_u}{l_u}\tilde{u}\right), \delta_2 = \dot{\omega}_d + l_\omega \tanh(\frac{k_\omega}{l_\omega}\tilde{\omega})$. The vector of parametric errors is defined $\tilde{\theta} = \hat{\theta} - \theta$ where $\hat{\theta}$ is the parameter estimation vector, (7) can be written as (9):

$$V_r = G\theta + G\tilde{\theta} = H\sigma + CV_d' + FV_d' + G\tilde{\theta} \tag{9}$$

where $\sigma = \dot{V}_d' + T(\tilde{V}')$.

Considering some of the parameters of the entire controller, such as $l_x, l_y, l_u\ and\ l_\omega$ must be calculated based on the robot's physical limits. But others, like k_x, k_y, k_u and k_ω can be selected by the user. "How to choose the right set of gains of the robot controller to operate well?" is an essential question that arises. Collection of gains to minimize a cost function is one approach to solving this problem (like energy consumption or tracking error). The C-BO will be discussed in the next subsection, which is utilized to choose the gains based on the functions we provide.

3. THE PROPOSED METHOD

The meta-heuristic optimization approaches able to solve certain problems but not others. At this time, the use of a revolutionary meta-heuristic algorithm-depend on ideal control strategy to obtain accurate and satisfied results. An overview of the BOA and the proposed C-BO are explained in this section.

3.1. The BOA technique

Different meta-heuristic algorithms have been constructed for dealing with complicated structure of current industrial systems and find optimum solutions. The BOA optimization method inspired by the popular game of billiards [18], [26]. Billiards is a game in which balls are struck with a cue and then passed around for a table. A six-pocket table is used for billiards, with one pocket in each corner and another in the long sides. To shatter the balls, the player moves a cue ball towards them. Then he must reposition the balls in better placements. A multi-dimensional billiard ball, consisting of a number of decision factors, is represented a possible solution in the BOA. The balls are considered as populations, and each dimension containing as a design variable. The procedure starts with a generation random of balls, with some of the better balls being selected for pockets. Ordinary and cue balls are the two types of balls. A target ball is hit by the cue ball, which pushes it into a corner. After balls collision, the rules of the Kinematic and the collision are formed, condition and moving directions of colliding balls are provided. The following is an example of the BOA optimization process.

a) Initialization: the balls' agents in space are originally distributed as (10):

$$B_{n,m}^{0} = Var_{m}^{min} + rand_{[0,1]}(Var_{m}^{max} - Var_{m}^{min}) \quad \text{n=1,2,3...,2N; m=1,2,3...., M} \tag{10}$$

where, $B_{n,m}^{0}$ signifies the variable's initial record for n[th] ball. Var_{m}^{max} and Var_{m}^{min} set the upper and lower boundaries for the m^{th} variable. $rand_{[0,1]}$ denote a random value that distributes evenly in [0, 1], *M* and *2N* denote variables numbers and populations.

b) Evaluation: fitness function is measured using the ball and pocket positions.

c) The determination of pockets: The pocket includes two roles in this algorithm; i) a ball objective that provides this BOA's exploit capability and ii) memory, which remembers the first top solutions found. This memory's purpose is to improve BOA's behavior without increasing the computing cost. The optimal balls found locations are updated using this memory in each iteration

d) Balls Grouping: In this step, balls are sorted according to their accuracy. Regular balls and cue balls are the two types of balls. The first half of these balls are regular balls (i.e., n=1, 2, …, N), and the second sets cue balls (i.e., n=N+1, …, 2N). Each cue ball in an outstanding group corresponds to the same rank [18].

e) Assigning the pockets to balls: A roulette-wheel selection strategy is used to assign a destination pocket to each ordinary ball. More features are promised in the pockets with a smaller fitness value. The following is the likelihood of picking a pocket:

$$P_K = \frac{e^{-\beta f k}}{\sum_k e^{-\beta f k}} \; ; k = 1,2,3 \ldots . k \tag{11}$$

where, β represents the pocket's fitness value and indicates a selection pressure>zero and f_k indicates the fitness of k^{th} pocket. The target balls are struck by the cue balls and travel into the pockets.

f) Ball position updating: The updated locations of ordinary balls are recorded after the collision. Ordinary balls' new placements are as (12) and (13),

$$PR = \frac{iter}{iter_{max}} \tag{12}$$

$$B_{n,m}^{new} = rand_{[-ER,ER]}(1 - PR)\left(B_{n,m}^{old} - P_{k,m}^{n}\right) + P_{k,m}^{n} \text{ , n=1,2,3.....N} \tag{13}$$

where $\boldsymbol{B_{n,m}^{new}}$ and $\boldsymbol{B_{n,m}^{old}}$ signify new and old value of the m^{th} parameter from the ordinary ball. The $\boldsymbol{P_{k,m}^{n}}$ sgnify the m^{th} variable of the k^{th} pocket. The accuracy rate is clarified by $\boldsymbol{PR}$. $\boldsymbol{rand}_{[-ER,ER]}$ Indicates a uniformly distributed random number in the range$[-\boldsymbol{ER}, \boldsymbol{ER}]$. $\boldsymbol{ER}$ signify error rate, $\boldsymbol{iter}$ and $\boldsymbol{iter_{max}}$ represent the current and max iterative numbers, respectively. The placements of cue balls after the collision are determined by their velocities, which are calculated as (14),

$$\overrightarrow{v_n'} = sqrt(2a\overrightarrow{B_n^{old}B_n^{new}})\widehat{B_n^{old}B_n^{new}} \tag{14}$$

where $\overrightarrow{\boldsymbol{v_n'}}$ is ordinary ball velocity; $\overrightarrow{\boldsymbol{B_n^{old}B_n^{new}}}$ is the ball movement vector and $\widehat{\boldsymbol{B_n^{old}B_n^{new}}}$ is the movement unit vector of n[th] ball ordinary after collision. The letter a stand for acceleration rate, and it equals one. Cue ball speeds are calculated as (15) and (16).

$$\overrightarrow{v_{n+N}} = \frac{\overrightarrow{v_n'}}{B_n^{old}\widehat{B_n^{new}}.B_{n+N}^{old}\widehat{+N}B_n^{old}} B_{n+N}^{old}\widehat{B_n^{old}}; \quad n = 1,2,3 \ldots . N \tag{15}$$

$$\overrightarrow{v_{n+N}'} = \omega(1 - \frac{iter}{iter_{max}})(\overrightarrow{v_{n+N}} - \overrightarrow{v_n'}) \tag{16}$$

where $\overrightarrow{\boldsymbol{v_{n+N}'}}$ and $\overrightarrow{\boldsymbol{v_{n+N}}}$ signify the n[th] cue ball velocities after and before the collision. $\boldsymbol{B_n^{old}}$ Indicate location of n^{th} cue ball before billiard stick. $\boldsymbol{\omega}$ Identifies a user-defined parameter [0,1] for controlling the cue ball movement. The (17) are the updated cue ball position:

$$\overrightarrow{B_{n+N}^{new}} = \frac{\overrightarrow{v_{n+N}'}}{2a}\overrightarrow{v_{n+N}'} + \overrightarrow{B_n^{old}}, n = 1,2,3 \ldots . N \tag{17}$$

g) Testing the boundary conditions' limits: As the balls' locations are updated, they may fall out of the table, causing the final balls' positions to be placed outside of the specified range. As a result, the ball size should be adjusted.
h) Testing the termination conditions: The procedure will be completed once specific criteria, such as the number of iterations, have been met. It will be continued if not.

3.2. The C-BO algorithm

Because the novel proposed C-BO algorithm has a straightforward formulation that makes it simple to apply, we chose it over ACO. Selection of ACO as the performance comparison because ACO had better results in earlier optimization efforts. The initialization process in the BOA-based meta-heuristic technique is initiated randomly because of the population sample selection as in (10), and therefore cannot achieve accurate beginning procedure. Because meta-heuristic algorithms are overly sensitive to initialization states and good beginning states will improve the performance of these algorithms. The main importance of combining chaotic with meta-heuristic algorithms and analyzing their influence on performances in [27], [28]. Because of their higher computing efficiency, chaotic logistic maps are the best option [27]. Furthermore, logistic maps have a higher probability of obtaining values between 0 and 1, allowing for faster local searches. The following is a formula for this chaotic map:

$$y_1 = rank$$
$$y_{i+1} = 4.y_i(1 - y_i), i = 1,2, \ldots . N \tag{18}$$

This part is replacing the initialization of BOA in steps a) to h) presented in (10) to (17). where rank is a vector of random integers in the range of [0, 1]. The C-BO algorithm is built by substituting the random $rand_i$ with the chaotic mapping vector (18). The traditional BOA with a chaotic character can be strengthened and improved in this way.

3.2.1. Tuning and simulation process-based C-BO algorithm

In this section, C-BO is used to optimize parameters k_x, k_y, k_u and k_ω of the tracking controller described in (4) and (7). Figure 3 depicts the ISE fitness function convergence of the C-BO, where shows a lower change after 300 iterative. The C-BO is implemented with 40 agents and 300 iterative. Constraints of the ISE are the gain factors (kx , ky , ku and kω), $0.01 \leq$ kx, ky, k_u and ky ≤ 15. This optimization technique repeats itself until the ISE's lowest value is found. To ensure that C-BO will work for the majority of runs since it is a stochastic optimization, the average optimized fitness function (ISE) and corresponding standard deviation for thirty independent runs are generated, displayed, and compared as illustrated in the Table 1. It is important to emphasize that the C-BO's low standard deviations demonstrate its stability. In 11 seconds, the C-BO arrived at best solution with the smaller value of the ISE. The C-BO algorithm, in particular, has a faster convergence rate than the ACO. The four gains after tuning for the controller ($k_x = 9.90$, $k_y = 4{,}367$, $k_u = 9.77$ and $k_\omega = 3{,}767$) change this error signal to offer the system control input. The system is thus forced to provide an output that is as close to the intended position as possible by the control input.

3.3. The ant colony optimization algorithm

Ant colony optimization is a graph-based evolutionary metaheuristic technique that has been used to address a variety of difficult optimization issues. The fundamental goal of ACO is to determine the cheapest path in a graph. Artificial ants explore this graph in the quest for better routes. Each ant has a rather simple behavior; thus, it will often find only law-quality paths on its own. In the colony, ant cooperative leads to better

pathways [29]. Figure 4 present the flowchart of ACO for optimize the parameter of the tracking controller. The optimum controller parameter calculation realized depending on the cost functions equation [30]:

$$ISE(e) = \sum_{i=1}^{n} {e_i}^2$$

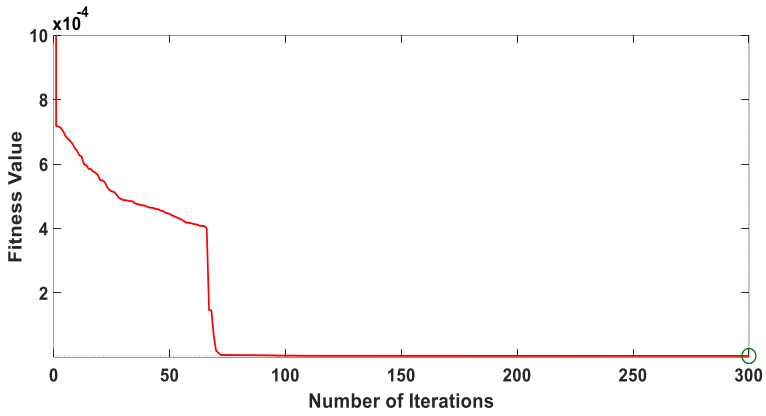

Figure 3. Cost convergence over iterations

Table 1. Comparison of the statistical results for C-BO and ACO algorithms

Technique	Ave.	Std. dev.
Tuning using ACO	0.0914	0.0334
Tuning using C-BO	0.038	0.0031

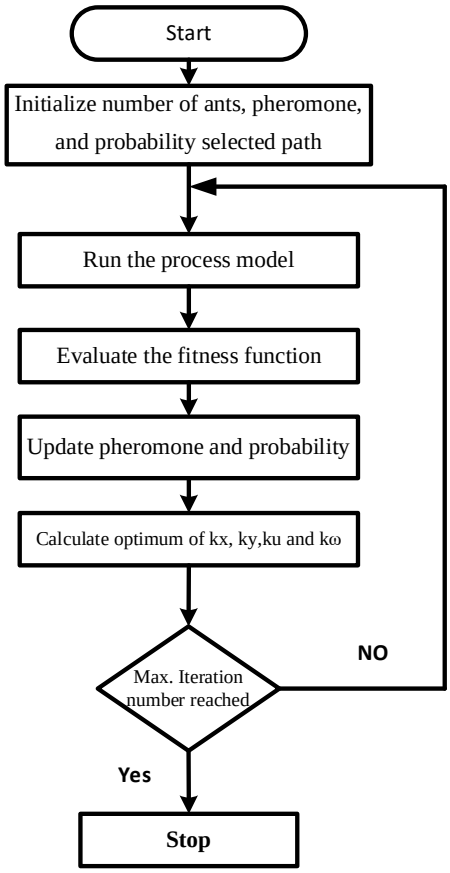

Figure 4. ACO flowchart

3.3.1. Tuning and simulation process based ACO

In this section, ACO is used to optimize the parameters of the tracking controller described in the controller section. It uses an ISE to ensure a perfect control performance under nominal operating conditions. One main point to be noted from this ACO specification is that some of the significant variables in the ACO algorithm are map size dependent, as shown in Table 2. This enables the entire calculation to function in simulation environments of varying sizes and designs, without having to re-program any part of the algorithm. The four gains after tuning for the controller ($k_x = 4{,}659, k_y = 4{,}271, k_u = 5{,}342$, and $k_\omega = 2{,}932$) at that point alter this error signal to provide the control input for the system.

Table 2. ACO specifications

Specifications	values
No. of ants	1000
Number of iterations	1000
Alpha	0.8
Beta	0.2
evaporation rate	0.7

4. RESULTS AND DISCUSSION

MATLAB 2020a/Simulink was used to investigate and implement the mobile robot controller approaches. A comparison of the simulation results for the two scenarios is presented to show the efficiency of the suggested C-BO algorithm, as well as the significance of the dynamic model. In case 1, the kinematic model only of the robot was considered without looking to dynamic model. The robot was directly controlled and dependent on the C-BO and ACO optimization techniques. For case 2, the overall dynamic model was considered, including the speed and acceleration limitations dependent on the C-BO and ACO. The controller was expected to follow the path of an 8-shaped trajectory. In the simulation cases, the robot started at position $(0.2, 0.0)$ m with 0° orientation, and followed an 8-shaped trajectory starting at $(0.0, 0.0)$ m. The path of the robot is represented by a sequence of intended positions and velocities, both of which differ in time. Noise in the position values and velocities that were fed back to the controllers were included. The simulation repeats with similar conditions for optimization purposes. The simulation started with no parameter refreshing, and this began only at $t = 100$ s. It remained active until the end of the simulation period. The accompanying parameters that were utilized in the simulation cases are: fixed sample time of 0.1 s and controller gains k_x, k_x, k_u, and k_w; saturation constants l_x, l_x, l_u, l_w; adaptation gains.

$\gamma = \text{diag}(1.7, 1.1, 0.5, 0.3, 0.01, 0.5)$; and sigma modification
$\Gamma = \text{diag}(0.0005, 0.001, 0.001, 0.00006, 0.001, 0.001)$.

4.1. Case 1: Robot with the kinematic model only

Figures 5 and 6 show the robot's path and the development of the distance error as case 1 of the simulation. The distance error is defined as the difference between the targets position *hd* and real position *h* at any given time. Figures 7 and 8 individually represent the advancement of the ideal and real X and Y positions with and without tuning the controller parameter during case 1 of the simulation. It is worth noting that a difference in performance when compared to the case without the optimization technique should be observed. As in the simulation without the optimization technique, the distance error did not reduce to zero. Instead, it oscillates around 0.1 m, causing the robot's path to be distorted. The distance error was reduced by roughly 0.01 after the ACO was used to tune the controller gain value. While the distance error was also reduced by 0.02 after applying C-BO to tune the controller gain value.

4.2. Case 2: Robot model with perfect dynamic compensation

In this case, the dynamic model of the pioneer 3-DX with LASER was considered, with limitations placed on its speed and acceleration. The proposed tuning strategy has the drawback of being reliant on dynamic model knowledge. In this case, ignoring the effects of the dynamic model was not significant. the noise was applied to the position and velocity values that were supplied back to the controllers to make the simulation more realistic. The principal objective of this study is to see the successful of the C-BO and ACO in discovering the ideal coefficients that will result in improved performance and better convergence. Table 3 indicates the optimum controllers 'parameters. The offline learning approach was utilized in both C-BO and ACO algorithms. After utilizing the ACO, the objective function value obtained was 3,490 with the parameter values ($k_x = 4{,}659$, $k_y = 4{,}271$, $k_u = 5{,}342$ and $k_\omega = 2{,}932$). Conversely, after utilizing C-BO, the results are ($k_x = 9.90$, $k_y = 4{,}367$, $k_u = 9.77$, and $k_\omega = 3{,}767$).

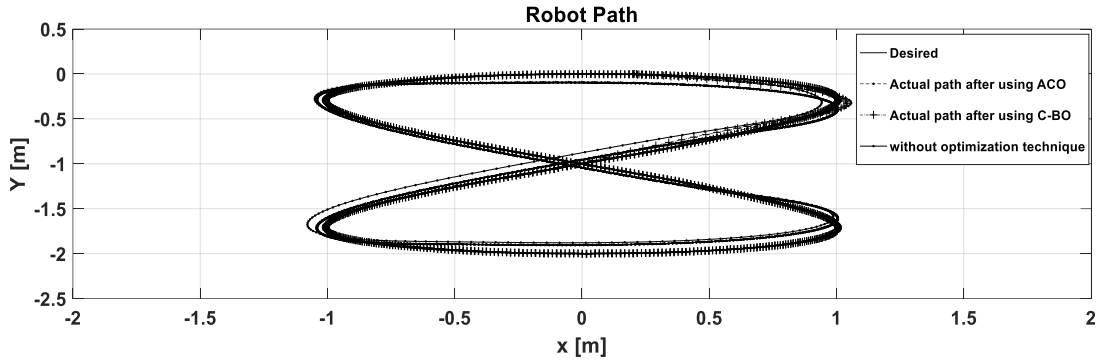

Figure 5. Robot path with kinematic controller only after using C-BO and ACO optimization technique

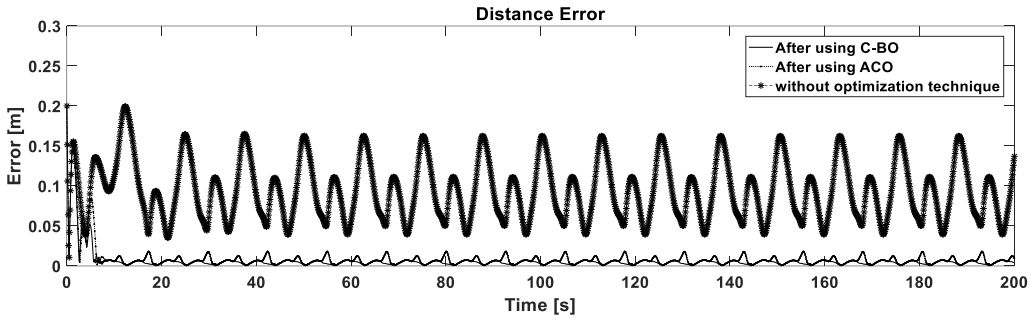

Figure 6. Distance error with kinematic controller only after using C-BO and ACO optimization technique

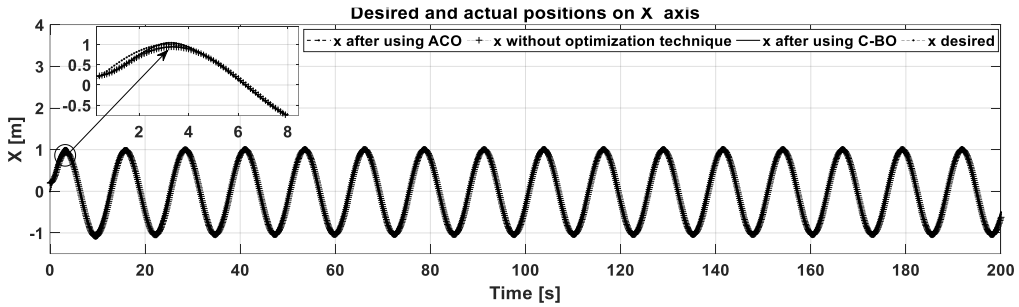

Figure 7. Desired and actual position on *x* axis with kinematic controller only after using the C-BO and ACO optimization techniques

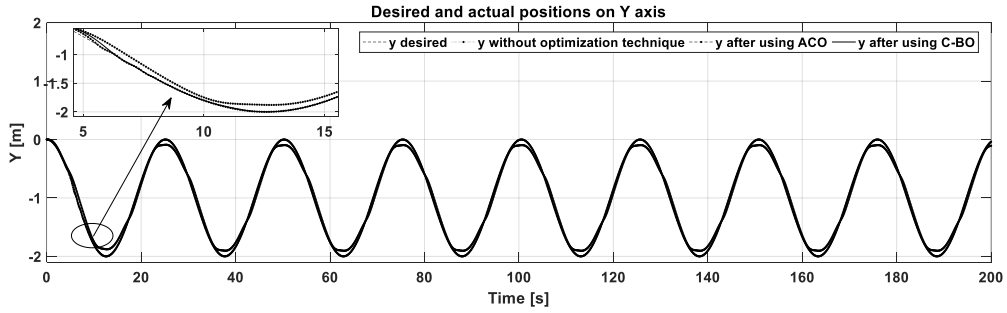

Figure 8. Desired and actual position on *Y* axis with kinematic controller only after using the C-BO and ACO optimization technique

Figures 9 and 10 show the robot path before and after upgrading the parameters, as well as the evolution of the distance error during the simulation. It is worth noting that without optimization, the distance error fluctuates nearly about 0.2 m till the upgrading of the parameter is enabled at $t = 100$ s. This results in reduction in the error to a value lower than 0.05 m at $t = 200$ s. After tuning the controller gain, which uses ACO, the distance error vibrates at 0.02 m till the parameter adjusted at $t = 100$ s is activated, the error drops to less than 0.008 m at $t = 200$ s. Similarly, after tuning the controller gain using C-BO with the distance error vibration around 0.02 m till the parameter activation updated at $t = 100$ s, the distance error vibrated around 0.02 m till the parameter activation updated at $t = 100$ s. At $t = 200$ s, the error is minimized to a value less than 0.006 m. Figures 11 and 12 show the ideal and actual positions of X and Y, respectively, after using the C-BO and ACO optimization techniques. From the figures, it can be observed that the robot is always behind the ideal location when no optimization approaches are used.

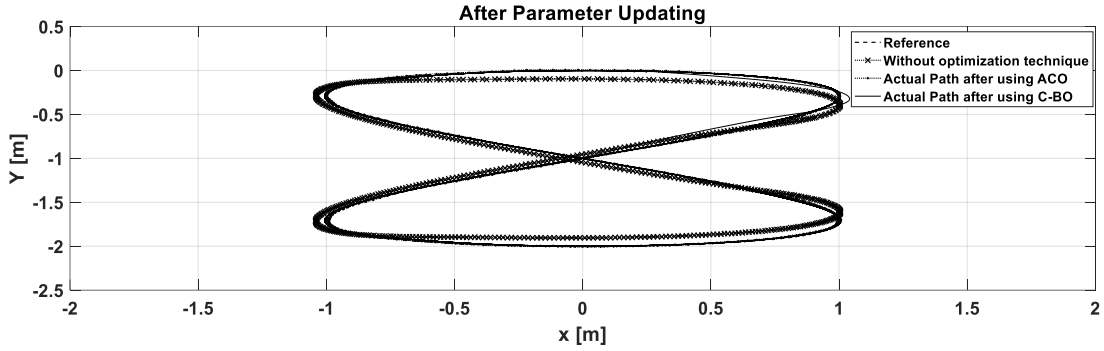

Figure 9. Robot path with a perfect dynamic compensation after using C-BO and ACO optimization techniques

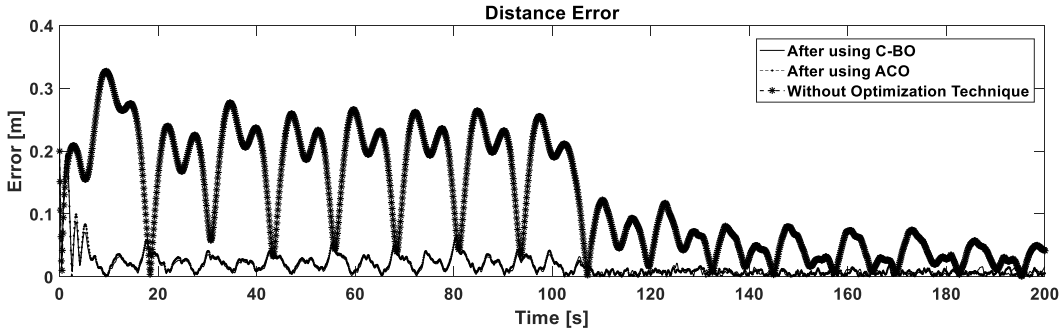

Figure 10. Distance error with a perfect dynamic compensation after using C-BO and ACO optimization techniques

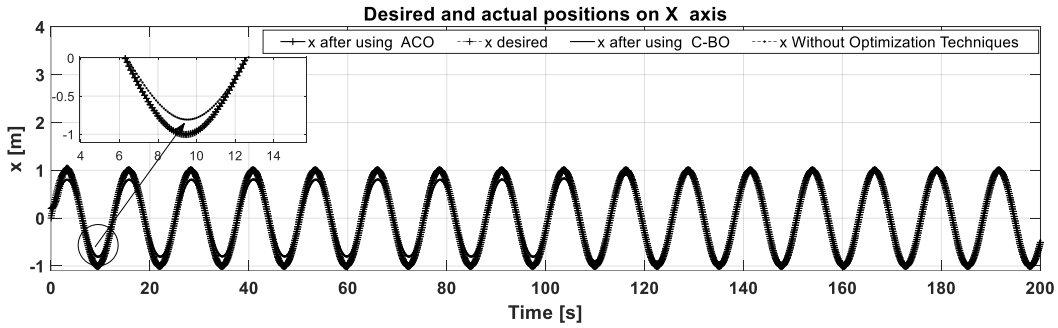

Figure 11. Desired and actual position on *x* axis with perfect dynamic compensation after using C-BO and ACO optimization techniques

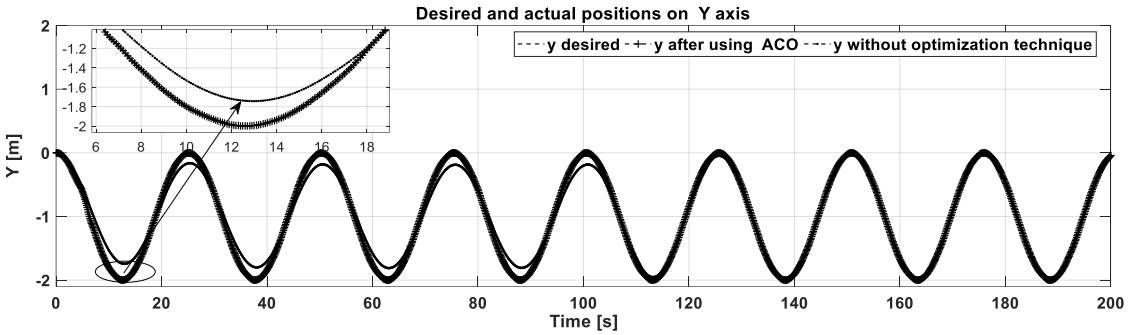

Figure 12. Desired and actual position on *Y* axis with perfect dynamic compensation after using C-BO and ACO optimization technique

Table 3. The comparison results of C-BO and ACO controllers after perfect dynamic compensation with impact of disturbances

Technique	Controller gains	ISE
Without an optimization technique	$kx = 0.1, ky = 0.1,$ $ku = 4, kw = 4$	8.6
Tuning using ACO	$kx = 4.659, ky = 4.271$ $ku = 5.342, k\omega = 2.932$	12.4
Tuning using C-BO	$kx = 9.90, ky = 4.367,$ $ku = 9.77, k\omega = 3.767$	15.3

From the results for the control gains presented in Table 3, it can be seen that the minimum error (ISE, IAE) value is accomplished by considering the set of gains after utilizing the C-BO or ACO. The simulation results show that the C-BO technique exhibits a better performance and convergence rate than the ACO. It has the benefits of quick response, tracking accuracy, good anti-interference, and high stability hence, the C-BO technique offers the best decision for path-tracking control of mobile robots with differential drive. Also, from the result presented tuning utilizing C-BO has the least error margin (i.e., a steady-state error for the robot's path is 0.006, $ISE = 0.1172$) as compared to ACO (which gave a steady-state error of 0.008 for the robot's path, $ISE = 0.1325$).

5. CONCLUSION

Motion control for a differential-drive mobile robot-based dynamic model compensation controller was presented in this study. Compared to models that are based only on the robot's kinematics, this approach produces more accurate simulation results allowing for a more precise assessment of robot behavior. The control parameters can be fine-tuned using either the proposed C-BO algorithm or the ACO methodology. We used the novel proposed C-BO algorithm because it has a simple formulation that makes it easy to implement. MATLAB/Simulink was used to test the control design of the mobile robot using a differential drive. The simulation results demonstrate that the C-BO technique outperforms the ACO technique in terms of the performance and convergence rate. A quick response, outstanding anti-interference, and tracking precision are all advantages. Consequently, the C-BO method is the most effective method for determining the tuning parameters that offer the optimal path tracking control for mobile robots with differential drives. The applicability, effectiveness, and superiority of the C-BO to identify the optimal solution have been demonstrated by comparisons of the results of C-BO optimization with the ACO approaches.

REFERENCES

[1] H. R. Beom and H. S. Cho, "A sensor-based navigation for a mobile robot using fuzzy logic and reinforcement learning," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, no. 3, pp. 464–477, Mar. 1995, doi: 10.1109/21.364859.

[2] H. Yang, M. Guo, Y. Xia, and L. Cheng, "Trajectory tracking for wheeled mobile robots via model predictive control with softening constraints," *IET Control Theory & Applications*, vol. 12, no. 2, pp. 206–214, Jan. 2018, doi: 10.1049/iet-cta.2017.0395.

[3] H. Yang, S. Wang, Z. Zuo, and P. Li, "Trajectory tracking for a wheeled mobile robot with an omnidirectional wheel on uneven ground," *IET Control Theory & Applications*, vol. 14, no. 7, pp. 921–929, Apr. 2020, doi: 10.1049/iet-cta.2019.1074.

[4] C. Ben Jabeur and H. Seddik, "Design of a PID optimized neural networks and PD fuzzy logic controllers for a two-wheeled mobile robot," *Asian Journal of Control*, vol. 23, no. 1, pp. 23–41, Jan. 2021, doi: 10.1002/asjc.2356.

[5] T. Das and I. N. Kar, "Design and implementation of an adaptive fuzzy logic-based controller for wheeled mobile robots," *IEEE*

Transactions on Control Systems Technology, vol. 14, no. 3, pp. 501–510, May 2006, doi: 10.1109/TCST.2006.872536.
[6] M. S. Kim, J. H. Shin, and J. J. Lee, "Design of a robust adaptive controller for a mobile robot," in *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000) (Cat. No.00CH37113)*, 2000, vol. 3, pp. 1816–1821, doi: 10.1109/IROS.2000.895235.
[7] C. Z. Resende, F. Espinosa, I. Bravo, M. Sarcinelli-Filho, and T. F. Bastos-Filho, "A trajectory tracking controller with dynamic gains for mobile robots," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2011, pp. 3746–3751, doi: 10.1109/IROS.2011.6094540.
[8] C. De La Cruz and R. Carelli, "Dynamic modeling and centralized formation control of mobile robots," in *IECON 2006 - 32nd Annual Conference on IEEE Industrial Electronics*, Nov. 2006, pp. 3880–3885, doi: 10.1109/IECON.2006.347299.
[9] T. Q. Khai, Y.-J. Ryoo, W.-R. Gill, and D.-Y. Im, "Design of kinematic controller based on parameter tuning by fuzzy inference system for trajectory tracking of differential-drive mobile robot," *International Journal of Fuzzy Systems*, vol. 22, no. 6, pp. 1972–1978, Sep. 2020, doi: 10.1007/s40815-020-00842-9.
[10] H. H. Ammar and A. T. Azar, "Robust path tracking of mobile robot using fractional order PID controller," in *International Conference on Advanced Machine Learning Technologies and Applications*, 2020, pp. 370–381.
[11] M. R. Azizi, A. Rastegarpanah, and R. Stolkin, "Motion planning and control of an omnidirectional mobile robot in dynamic environments," *Robotics*, vol. 10, no. 1, Mar. 2021, doi: 10.3390/robotics10010048.
[12] C.-H. Chen, C.-J. Lin, S.-Y. Jeng, H.-Y. Lin, and C.-Y. Yu, "Using ultrasonic sensors and a knowledge-based neural fuzzy controller for mobile robot navigation control," *Electronics*, vol. 10, no. 4, Feb. 2021, doi: 10.3390/electronics10040466.
[13] J. Li, J. Wang, H. Peng, Y. Hu, and H. Su, "Fuzzy-torque approximation-enhanced sliding mode control for lateral stability of mobile robot," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 4, pp. 2491–2500, Apr. 2022, doi: 10.1109/TSMC.2021.3050616.
[14] Y. Xie *et al.*, "Coupled fractional-order sliding mode control and obstacle avoidance of a four-wheeled steerable mobile robot," *ISA Transactions*, vol. 108, pp. 282–294, Feb. 2021, doi: 10.1016/j.isatra.2020.08.025.
[15] P. J. Fleming and R. C. Purshouse, "Evolutionary algorithms in control systems engineering: a survey," *Control Engineering Practice*, vol. 10, no. 11, pp. 1223–1241, Nov. 2002, doi: 10.1016/S0967-0661(02)00081-3.
[16] R.-C. David, R.-E. Precup, S. Preitl, J. K. Tar, and J. Fodor, "Three evolutionary optimization algorithms in PI controller tuning," in *Applied Computational Intelligence in Engineering and Information Technology*, 2012, pp. 95–106.
[17] V. Sathiya and M. Chinnadurai, "Evolutionary algorithms-based multi-objective optimal mobile robot trajectory planning," *Robotica*, vol. 37, no. 8, pp. 1363–1382, Aug. 2019, doi: 10.1017/S026357471800156X.
[18] A. Kaveh, M. Khanzadi, and M. R. Moghaddam, "Billiards-inspired optimization algorithm; a new meta-heuristic method," *Structures*, vol. 27, pp. 1722–1739, Oct. 2020, doi: 10.1016/j.istruc.2020.07.058.
[19] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, Apr. 1997, doi: 10.1109/4235.585892.
[20] S.-H. Chia, K.-L. Su, Jr-H. Guo, and C.-Y. Chung, "Ant colony system based mobile robot path planning," in *2010 Fourth International Conference on Genetic and Evolutionary Computing*, Dec. 2010, pp. 210–213, doi: 10.1109/ICGEC.2010.59.
[21] T. Wang, L. Zhao, Y. Jia, and J. Wang, "Robot path planning based on improved ant colony algorithm," in *2018 WRC Symposium on Advanced Robotics and Automation (WRC SARA)*, Aug. 2018, pp. 70–76, doi: 10.1109/WRC-SARA.2018.8584217.
[22] Q. Luo, H. Wang, Y. Zheng, and J. He, "Research on path planning of mobile robot based on improved ant colony algorithm," *Neural Computing and Applications*, vol. 32, no. 6, pp. 1555–1566, Mar. 2020, doi: 10.1007/s00521-019-04172-2.
[23] C. De La Cruz and R. Carelli, "Dynamic model-based formation control and obstacle avoidance of multi-robot systems," *Robotica*, vol. 26, no. 3, pp. 345–356, May 2008, doi: 10.1017/S0263574707004092.
[24] F. N. Martins, M. Sarcinelli-Filho, and R. Carelli, "A velocity-based dynamic model and its properties for differential drive mobile robots," *Journal of Intelligent & Robotic Systems*, vol. 85, no. 2, pp. 277–292, Feb. 2017, doi: 10.1007/s10846-016-0381-9.
[25] F. N. Martins, W. C. Celeste, R. Carelli, M. Sarcinelli-Filho, and T. F. Bastos-Filho, "An adaptive dynamic controller for autonomous mobile robot trajectory tracking," *Control Engineering Practice*, vol. 16, no. 11, pp. 1354–1363, Nov. 2008, doi: 10.1016/j.conengprac.2008.03.004.
[26] J.-F. Landry and J.-P. Dussault, "AI optimization of a billiard player," *Journal of Intelligent and Robotic Systems*, vol. 50, no. 4, pp. 399–417, Nov. 2007, doi: 10.1007/s10846-007-9172-7.
[27] Z. S. Ma, "Chaotic populations in genetic algorithms," *Applied Soft Computing*, vol. 12, no. 8, pp. 2409–2424, Aug. 2012, doi: 10.1016/j.asoc.2012.03.001.
[28] L. dos Santos Coelho and P. Alotto, "Multiobjective electromagnetic optimization based on a nondominated sorting genetic approach with a chaotic crossover operator," *IEEE Transactions on Magnetics*, vol. 44, no. 6, pp. 1078–1081, Jun. 2008, doi: 10.1109/TMAG.2007.916027.
[29] H. A. Varol and Z. Bingul, "A new PID tuning technique using ant algorithm," in *Proceedings of the 2004 American Control Conference*, 2004, pp. 2154–2159 vol.3, doi: 10.23919/ACC.2004.1383780.
[30] M. Ünal, H. Erdal, and V. Topuz, "Trajectory tracking performance comparison between genetic algorithm and ant colony optimization for PID controller tuning on pressure process," *Computer Applications in Engineering Education*, vol. 20, no. 3, pp. 518–528, Sep. 2012, doi: 10.1002/cae.20420.

BIOGRAPHIES OF AUTHORS

Reham H. Mohammed was born on March 21, 1983. She received the B.Sc., M.Sc. degrees from Faculty of Engineering, Mansoura University in 2005 and 2010, respectively and PH.D. degree from Faculty of Engineering, Suez Canal University in 2016. In 2005, she joined the Mansoura University, Faculty of Engineer, in Egypt, as assistant Research. In 2011, she worked as assistant lecturer in Suez Canal University, Faculty of Engineer, in Egypt. Currently, she worked as lecturer at Suez Canal University, Faculty of Engineer, in Egypt. She has more than 16 scientific research papers published in prestigious international journals. She can be contacted at email: reham.elenani@eng.suez.edu.eg.

Mohamed E. Dessouki was born in Bohiera, Egypt in 1976. He received his B.Sc., M.Sc. and Ph.D. degrees, from Faculty of Engineering, Suez Canal University in 1999, 2004 and 2010 respectively, all in Electrical Engineering. He joined the Department of Electrical Engineering, Faculty of Engineering, Suez Canal University as a demonstrator in 2000. Then, he became an assistant Lecturer from 2004 to 2010 and has been a Lecturer since 2010. He joined the Department of Electrical Engineering, King Abdulaziz University, Rabigh. His main professional interests include AC and DC drives, direct torque and field-oriented control techniques, and DSP control and power electronics applications. He can be contacted at email: dessouky_m@yahoo.com.

Basem E. Elnaghi received the B.Sc., M.Sc. and Ph.D. degrees in electrical power engineering in 2002 and 2009, and 2015 respectively, from Suez Canal University, Port Said, Egypt. Since 2015, he has been Associate professor with the Electrical Engineering Department, Faculty of Engineering, Suez Canal University. He joined the Department of Faculty of Engineering, Suez Canal University, Rabigh. His main professional interests include AC and DC drives, direct torque and field-oriented control techniques, and DSP control, control of electrical machines and wind energy conversion system, and power electronics applications. He can be contacted at email: Basem_elhady@eng.suez.edu.eg.