Journal of Robotics and Control (JRC)
Volume 5, Issue 1, 2024
ISSN: 2715-5072, DOI: 10.18196/jrc.v5i1.20489

Path Planning and Trajectory Tracking Control for Two-Wheel Mobile Robot

Ibrahim A. Hassan [1], Issa A. Abed [2*], Walid A. Al-Hussaibi [3]
[1, 2, 3] BETC, Southern Technical University, Basra, Iraq
Email: [1] brah81athab@gmail.com, [2] issaahmedabed@stu.edu.iq, [3] alhussaibi@stu.edu.iq
*Corresponding Author

***Abstract*—The mobile robot is a system that can work in various environments. This means that the robot must be able to navigate without delay and avoid any obstacles placed within the boundaries of its movement. Designing mobile robots that can be intelligently managed and operate autonomously when traveling from one place to another requires at least two steps. To start with, path planning is required to prevent motion collisions. Tracking the robot's trajectory is a crucial second task. The primary goal of this study is to find the quickest and safest path between the two positions. In this work, we investigated the path planning of a mobile robot with dynamic, and dynamic obstacles with moving goal environments using RRT, BiRRT, and HA* algorithms. These algorithms are easy, computationally inexpensive, and simple to use. They have been chosen for numerous real-time path-planning applications. The DDMR's kinematic model has been utilized in this paper to control path tracking, and a PID controller has been proposed to reduce tracking deviations between the robot's actual route and the reference trajectory. This work introduced the PSO, FPA, CSA, SSA, BWOA, and proposed HBPO optimization techniques for obtaining PID parameters (k_p,k_i,k_d) for improved mobile robot trajectory tracking. The simulation results have been examined using three trajectory shapes: step, circular, and infinite. The simulation findings reveal that HA* outperforms the other algorithms by generating collision-free pathways that are smoother and shorter than their RRT and BiRRT equivalents. On the other hand, the proposed HBPO outperforms the other methods. The HBPO method converges quicker than the other proposed algorithms.**

Keywords—Mobile Robot; Path Planning; Trajectory Tracking; PID Controller.

I. INTRODUCTION

Robotics is the study of the conception, design, development, control, and application of autonomous systems. It is a rapidly growing field with a wide range of applications. With the development of technology, robotics has expanded from the lab into almost every field of science, engineering, industry, manufacturing, transportation, and agriculture[1], [2], [3], [4], [5]. Self-contained robots like drones, weaponry, and Unmanned Aerial Vehicles (UAVs) are utilized in military applications and everyday life. Even at home, robots are capable of doing almost any activity, including food service, room cleaning, and door control [6], [7]. Autonomous Mobile Robots (AMR) are one of the technologies that are now growing, because of their relevance and uses in today's society. The mobile robot is a system that can work in various environments. This means that the robot must be able to navigate without delay and avoid any obstacles placed within the boundaries of its movement [8], [9], [10], [11]. Therefore, to design mobile robots that can be intelligently managed and operate autonomously when traveling from one site to another, at least two steps are required. In the beginning, path planning is critical in order to avoid motion collisions. A critical second task is robot trajectory tracking [12]. The AMR's task of path planning is crucial. In order to go safely from the start configuration to the goal configuration, it is desired to locate a collision-free motion in an environment that is full of obstacles. Mobile robots are being used in more environments, both static and dynamic. Normally, there are many feasible paths for a robot to take to reach the target from the start location. The best feasible path is chosen based on criteria like the shortest distance, smoothness of the path, minimum energy consumption, or the most frequently used criteria, the shortest distance with the minimal possible time [13], [14], [15], [16]. Trajectory tracking is an essential feature of mobile robots. Mobile robots' primary purpose is to move along a specified path and decide where to go, and that information is taken by a reference path or leader robot. In order to make a robot follow a trajectory, the process of trajectory tracking involves calculating the robot's speed and steering settings at each instant of time. The positional coordinates of a particular route are represented by a group of points called a trajectory [12], In a trajectory-tracking issue, a mobile robot should follow a time-relationship graph. Geometric parameters define the desired path to trace in the path-tracking issue [17].

A. Aims and Contributions

Based on the foregoing, the main aims of this paper can be described as follows:

1. Designing various environments for a mobile robot with dynamic, and dynamic obstacles with a moving target, Then, use the RRT, BiRRT, and HA* algorithms to find the shortest route from the robot's starting point to its intended destination.
2. Investigate trajectory tracking for an autonomous nonholonomic two wheeled mobile robot with a reference trajectory and use the proportional integral derivative (PID) to regulate the robot's velocities such that tracking errors are minimized between the actual and reference trajectories.
3. In this paper, we propose optimization algorithms such as BWOA, SSA, CSA, FPA, PSO, and HBPO to obtain PID parameters (k_p, k_i, k_d) for enhancing mobile robot trajectory tracking.

This paper's main contributions are as follows:

1. This paper is concerned with simulating a mobile robot that can travel from the start point to the goal point using artificial intelligent algorithms.
2. Comparison of the HA* and BiRRT versus RRT algorithms in dynamic, and dynamic obstacles with moving target environments for path planning mobile robots.
3. Many proposed optimization algorithms, such as CSA, and BWOA, have been employed for the first time in this study to tune the PID controller for improving mobile robot trajectory tracking.

To propose a new technique for tuning PID controllers that combines the BWOA and PSO algorithms (HBPO) in order to improve mobile robot trajectory tracking.

B. Paper Organization and Notations

This paper is organized as follows: Section II presents related work on path planning and trajectory tracking. Section III introduces details of the modeling of two-wheel DDMR. Section IV presents robot path planning. This section has four subsections: subsections A, B, and C explain the RRT, BiRRT, and HA* algorithms, respectively, while subsection D introduces the simulation and results of path planning using these algorithms. Trajectory tracking control and optimization are discussed in Section V. This section has four subsections: Subsection A proposes wheeled mobile robot trajectory tracking based on a PID controller. Subsection B presents intelligent PID and optimization algorithms. Subsection C introduces the Hybrid Black Widow Optimization Algorithm and Particle Swarm Optimization Algorithm (HBPO), and Subsection D focuses on simulation and the results of control trajectory tracking. Subsection E discusses the results. Finally, the conclusion and future work are presented in Section VI.

II. RELATED WORKS

Many papers have focused on path planning. Finding a path from the beginning (current) state to the goal is widely described in other literature. Many path-planning strategies have been presented and tested in a variety of settings with both static and moving obstacles. Trajectory tracking is a fundamental task for mobile robots that has recently received much attention in many publications. The control of mobile robots is a vital part of engineering research due to the development of several intelligent and classical control systems. The purpose of trajectory tracking control is to develop control rules for mobile robot velocity that minimize tracking errors between the robot's real route and the reference trajectory. For many years, PID controller research has received considerable attention. This is the most well-known method of controlling the navigation of mobile robots.

Fuzzy logic and filter smoothing based on the data from the laser scan sensor, path planning, and autonomous obstacle avoidance based on this algorithm are emphasized because they can automatically find the best path according to the size and position of the gaps between the obstacles in the dynamic environment proposed in [18]. Two nature-inspired meta-heuristic algorithms, the cuckoo-search and bat algorithms, address mobile robot path planning in an unknown or partially known environment [19]. The use of a genetic algorithm (GA) for robot path planning in a static environment has been discussed, along with a review of the subject of path planning optimization, in [20]. By enhancing the artificial potential field (APF) approach, the robot can travel the shortest distance possible while avoiding collisions with stationary obstacles [21]. In [22], researchers created a new path-planning method that uses reinforcement learning and integrated environment representation to run a mobile robot with nonholonomic constraints in unpredictable dynamic settings. In [23], the particle swarm optimization (PSO) algorithm is combined with the potential field method (PFM) for static obstacles and the PFM for dynamic obstacles to provide an effective path-planning solution for the robot. A new hybrid path-planning approach that combines the A* algorithm with an adaptive window approach to conduct global path planning, real-time tracking, and obstacle avoidance has been suggested. In [24], for mobile robots in large-scale dynamic environments. In [25], described a new method based on the Bug algorithm to overcome run-time barriers. A new replanning method in [26], based on an extended rapidly exploring random tree. A new dynamic replanning approach allows the robot to correctly plan a route in complex environments. To improve the fundamental fast-exploring random tree strategy, [27] used a target bias search approach and a novel metric function that took both distance and angle into account. Curve-smoothing has been used in [28] to develop a goal-biased bidirectional Rapidly Exploring Random Trees (RRTs) technique. The Hybrid A-Star search engine and visibility diagram planning have been used in a novel and effective way to discover the shortest non-holonomic path in hybrid environments for valet parking in [29]. The path planning strategy proposed in [30] that combines Bi-RRT with an enhanced potential field to increase search probability and efficiency. In [31], Fast-RRT, a new RRT-based path-finding method that quickly locates a path that is close to ideal, has been used. The Fast-RRT algorithm comprises two modules: Improved RRT and Fast-Optimal. In [32], developed a trajectory planning method for mobile robots with maneuver limitations. The robot's velocity affects the maneuver limits. Virtual barriers that close off the robot-inaccessible sectors while it goes at a certain speed operate as restrictions. Any path planning algorithm is compatible with the suggested approach. In [33], proposed a modified version of the HA* technique used to navigate spherical mobile robots. Using a pendulum, the new method allows for limited and partial lateral motion. In [34], suggested an autonomous parking trajectory planning method in an unstructured environment with narrow passages. In [35], conducted research on the hybrid A-star algorithm for nonholonomic robots, focusing on two phases: the forward search phase and the analytic expansion phase. The forward search phase takes into account the robot's kinematics to plan its continuous motion on discrete grid maps. Meanwhile, the analytic expansion phase employs the Reeds-Shepp (RS) curve to enhance the algorithm's accuracy and speed.

The control of mobile robots using their trajectories has attracted the attention of numerous researchers. Trajectory tracking is an essential feature of mobile robots. The primary job of mobile robots is to move along a predetermined course

and decide where to go next, with that information being taken from a reference path. Tracking control is crucial because the robot must move along the desired trajectory. Several controllers have been developed recently to operate mobile robots. The PID controller is one of many control strategies that are frequently employed in the robot field. The trajectory of an autonomous nonholonomic wheeled mobile robot using a virtual reference trajectory investigated in [36], the follower robot's velocity is controlled using a PID controller enhanced by a genetic algorithm (GA) to improve control accuracy and convergence speed. In [37], an intelligent fuzzy controller and genetic algorithm have been used to perform target tracking, obstacle avoidance, and time and path optimization. Using the Particle Swarm Optimization (PSO) technique, [38] created an optimal fuzzy PD + I controller for trajectory control of a mobile robot. PSO has been utilized to modify the controller's gains and membership functions to reduce the Integral Square Error (ISE) index, which captures linear and angular velocity errors. The goal had been to develop the best controller and improve results. In [39], A fuzzy logic-based navigation control system has been proposed for a 2-wheel mobile robot. It uses a kinematic model and a fuzzy logic controller designed in Simulink to safely navigate the environment and reach the destination. A novel trajectory tracking control system for mobile robots has been developed, combining reinforcement learning and PID control. In [40], looked into the wheeled mobile robot's navigation approach in a known area with static obstacles. The technique is based on an assessment of the visibility situation, which determines whether there is a point where the intended path and obstacles cross. In [41], presented a design for the kinematic control structure of the wheeled Mobile Robot (WMR) route planning and path following. The system utilizes Q-learning and PID techniques to track the desired trajectory of the robot [42]. A variable-parameter PID controller has been proposed by [43] for a differential-drive mobile robot to follow a NURBS trajectory with time-varying velocity. The controller minimizes kinematic error by linearizing the robot's kinematic model and selecting appropriate coefficients through modeling and experimentation. In [44], A novel hybrid technique has been presented, integrating a neural network-based kinematic controller and a model reference adaptive control. The proposed controller achieves superior tracking accuracy and fast convergence compared to PID, kinematic, and adaptive dynamic controllers. Performance analysis indices confirm its effectiveness, even in the presence of parameter uncertainties and slip disturbances.

III. MODELING OF TWO-WHEEL DDMR

The robot features two parallel, conventional wheels (one on each side) that are operated by two independent actuators. It is also assumed that each wheel is perpendicular to the ground and that the wheels' contact with the ground is non-slipping and pure rolling [44]. This paper considers a 2-wheeled, non-holonomic mobile robot. Fig. 1 depicts the robot's design.

The quantities in Fig. 1 are described [39], [45], as: V : Linear velocity (ms^{-1}), V_r: is the linear velocity of the right wheel (ms^{-1}), V_l: is the linear velocity of the left wheel (ms^{-1}), ω : Angular velocity (rads^{-1}), r: is the radius of the right and the left wheels (m), D: is the distance between the right and the left wheels (m).

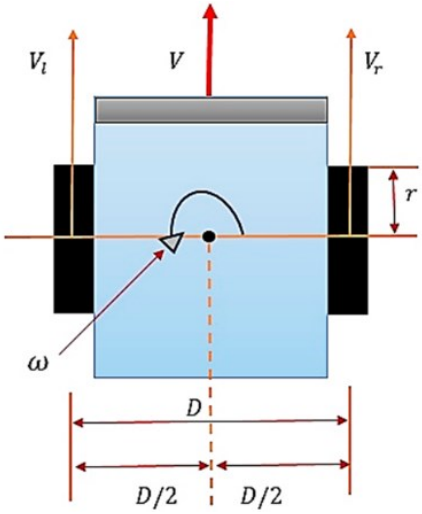


Fig. 1. Two-wheel mobile robot

A. Kinematics Model of the Two-Wheel DDMR

Kinematic modeling investigates the mathematics of motion without considering the influencing factors and instead focuses on the geometric relationships that control the system [46]. Fig. 2 shows the position of the differential drive mobile robot along the global coordinate axes of {O, X, Y}[47].

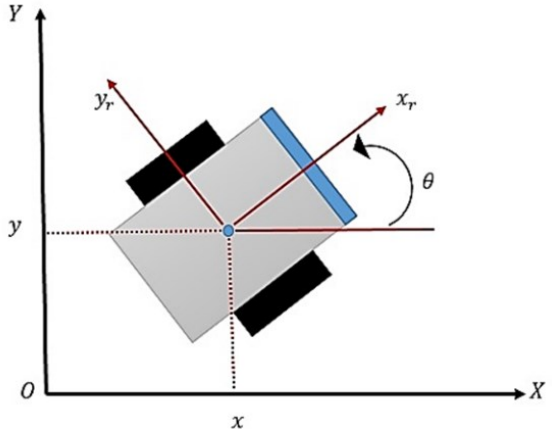


Fig. 2. Geometrical structure of DDMR

The differential drive robot's kinematic equations in the world frame are as follows given the specified constraints [37], [39], [46], [47] .

Equation (1) represents the motion of a two-wheel DDMR.

$$\left.\begin{aligned} V_r &= r\,\omega_r \\ V_l &= r\,\omega_l \end{aligned}\right\} \tag{1}$$

where: $\boldsymbol{\omega_r}$**:** Angular velocity of the right driving wheel (rads^{-1}), $\boldsymbol{\omega_l}$**:** Angular velocity of the left driving wheel (rads^{-1}).

The robot's nonholonomic constraint equation is as follows:

$$\dot{y}\cos(\theta) - \dot{x}\sin(\theta) = 0 \tag{2}$$

The robot's dynamic function is defined as (3).

$$\left.\begin{aligned} x^{\cdot} &= V\cos(\theta) \\ y^{\cdot} &= V\sin(\theta) \\ \theta^{\cdot} &= \omega \end{aligned}\right\} \quad (3)$$

The previous equations can be written in matrix form, and the differential drive mobile robot's simplified kinematic model, which is used to design the robot, is represented by equation (4):

$$\begin{bmatrix} x^{\cdot} \\ y^{\cdot} \\ \theta^{\cdot} \end{bmatrix} = \begin{bmatrix} cos\theta & 0 \\ sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V \\ \omega \end{bmatrix} \quad (4)$$

By transforming these velocities components into rotational velocities (ω_r, ω_l), the above model can be improved to:

$$\begin{bmatrix} V \\ \omega \end{bmatrix} = \begin{bmatrix} r/2 & r/2 \\ r/D & -r/D \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} \quad (5)$$

Equation (5) is substituted into Equation (4) to obtain a more detailed kinematic model of WMR as (6).

$$\begin{bmatrix} x^{\cdot} \\ y^{\cdot} \\ \theta^{\cdot} \end{bmatrix} = \begin{bmatrix} \frac{r}{2}\cos\theta & \frac{r}{2}\cos\theta \\ \frac{r}{2}\sin\theta & \frac{r}{2}\sin\theta \\ \frac{r}{D} & -\frac{r}{D} \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} \quad (6)$$

Equation (6) can be written in terms of V_r and V_l as (7).

$$\begin{bmatrix} x^{\cdot} \\ y^{\cdot} \\ \theta^{\cdot} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\cos\theta & \frac{1}{2}\cos\theta \\ \frac{1}{2}\sin\theta & \frac{1}{2}\sin\theta \\ \frac{1}{D} & -\frac{1}{D} \end{bmatrix} \begin{bmatrix} V_r \\ V_l \end{bmatrix} \quad (7)$$

The robot's velocity is the sum of its two speeds, V_r and V_l. Also, the relationship between the robot's angular speed, V_r, and V_l, and the separation between the two wheels is as (8).

$$\left.\begin{aligned} V &= \frac{1}{2}\,(V_r + V_l) \\ \theta^{\cdot} &= \frac{1}{D}\,(V_r - V_l) \end{aligned}\right\} \quad (8)$$

Substituting (1) into (8), we get an equation that explains the relationship between each wheel's angular velocity and the robot's linear and angular velocity in (9).

$$\left.\begin{aligned} V &= \frac{r}{2}\,(\omega_r + \omega_l) \\ \theta^{\cdot} &= \frac{r}{D}\,(\omega_r - \omega_l) \end{aligned}\right\} \quad (9)$$

These two equations can be solved simultaneously to yield the required differential wheel velocities as (10).

$$\left.\begin{aligned} V_r &= \frac{2V + \omega D}{2} \\ V_l &= \frac{2V - \omega D}{2} \end{aligned}\right\} \quad (10)$$

IV. ROBOT PATH PLANNING

Path planning in autonomous mobile robot applications is challenging and depends on the environmental model, robot type, and specific application. In a 2D binary occupancy map, the robot's workspace is defined by Cartesian coordinates (x, y). The start point represents the robot's initial position, and the goal point is the desired destination. To find a collision-free path, we used MATLAB 2021a and implemented RRT, BiRRT, and Hybrid A* algorithms. These algorithms efficiently navigate the robot from start to goal while considering free space, static obstacles, dynamic obstacles, and a moving target.

A. RRT Algorithm

The fundamental RRT is a practical random sampling-based path planning technique that provides numerous advantages, such as fast speed and solid real-time performance. The RRT scheme involves iteratively growing a randomized data structure tree by adding new nodes while providing control input until it reaches the desired configuration. Rapid search space expansion combined with path modifications results in collision-free paths. The implemented algorithm can be used to resolve the uncertainty problem in state space, where the tree has grown alongside the sampled state [48], [49], [50].

B. Bidirectional RRT Algorithm

The BiRRT uses a bidirectional search tree to enhance the efficiency of node expansion. The basic RRT's tree expands in space by connecting random locations in emptiness, but it cannot select a new node when it meets a barrier [51]. In comparison to the original RRT, the BiRRT generates another parallel tree in the intended growth area. Each iteration follows the same initial procedures as the RRT, which involve sampling random spots first and then expanding [52].

C. Hybrid A Algorithm*

HA* has been the subject of more research over the past decade as a result of its exceptional performance in planning for motion applications, particularly in uncharted regions [53]. For non-holonomic self-driving vehicles, the HA* is among the most effective path planners. The algorithm's hybrid term employs an occupancy grid map with which the surrounding is depicted in a discrete manner to plan the vehicle's travel in a continuous domain. The HA* search algorithm enhances the standard A* search method for non-holonomic robot implementation. The key to accomplishing this is to use the vehicle's kinematics to forecast the vehicle's motion based on velocity, gear, and driving angle. This strategy assists the planner in selecting the appropriate successor node that a nonholonomic robot can follow [29].

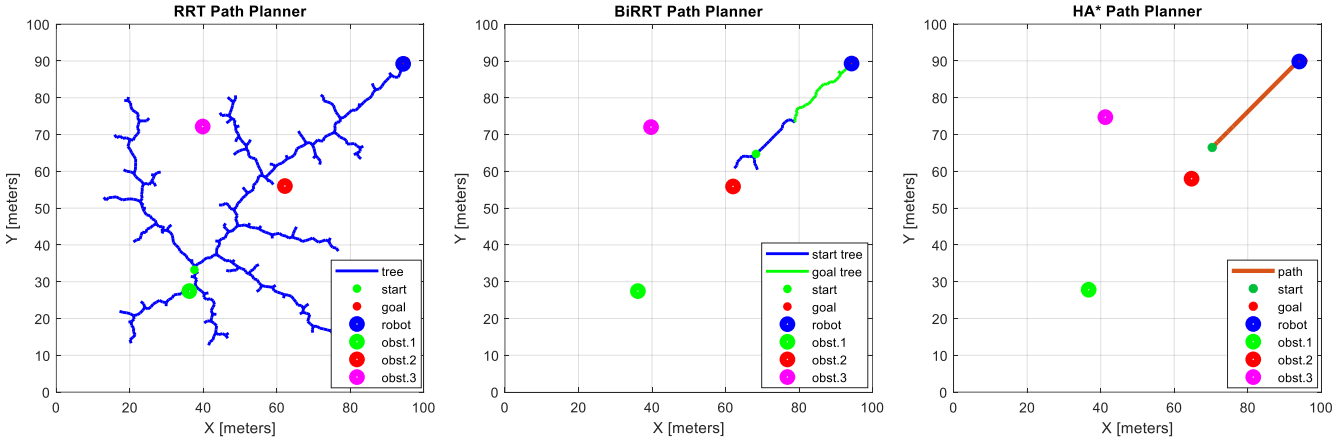
D. Simulation and Results of Path Planning

This section compares the performance of the BiRRT and HA* methods to the reference RRT method for mobile robot path planning using MATLAB simulation. The experiments have been carried out in two environment modules: dynamic obstacles, and a moving target in a dynamic environment. Specifications of the utilized computer are: The operating system is Windows 10, which is a 64-bit operating system, and the processor is an Intel(R) Core(TM) i7-1165G7U CPU @ 2.80 GHz. The results have been presented as follow:

1) Dynamic Environment

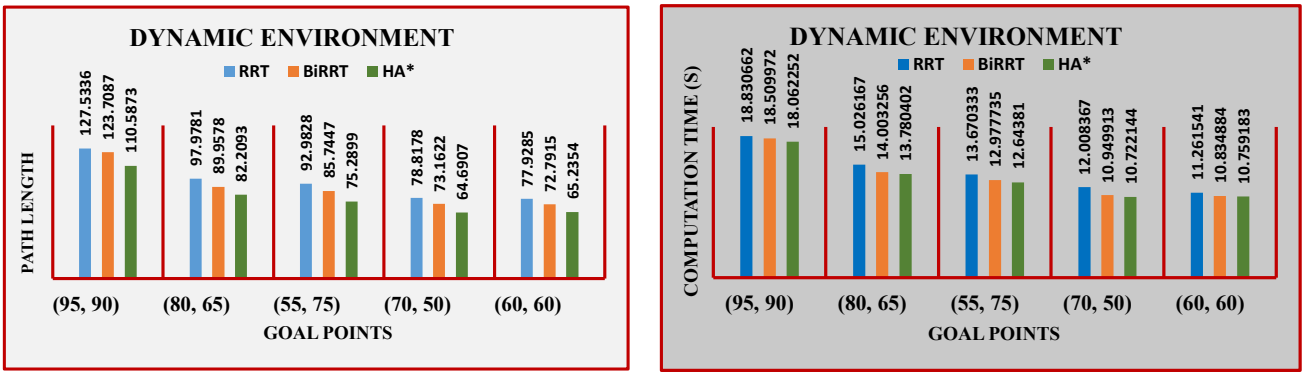
This scenario is considered an environment whose dimensions are (100, 100) with three dynamic obstacle shapes; the initial positions of these obstacles are (40, 30) moving toward (10, 10), (80, 70) moving toward (10, 15), and

(50, 90) moving toward (10, 20). (20, 10) is the starting point, and various goal points are used. The maximum connection distance is 1, and the validation distance is 0.01. Fig. 3 depicts the planned paths for example goal points in this environment using the RRT algorithm (a), the BiRRT technique (b), and the HA* technique (c). In this example, the goal point is (95, 90). Fig. 4 shows (a) the length of the path in meters and (b) the time required reaching the destination in seconds. Table I provides a summary of the results obtained. The example of a goal point is explained as Fig. 3 and Fig. 4.



(a) (b) (c)

Fig. 3. Path planning in a dynamic environment with (95, 90) goal point using: (a) RRT, (b) BiRRT, (c) HA*



(a) (b)

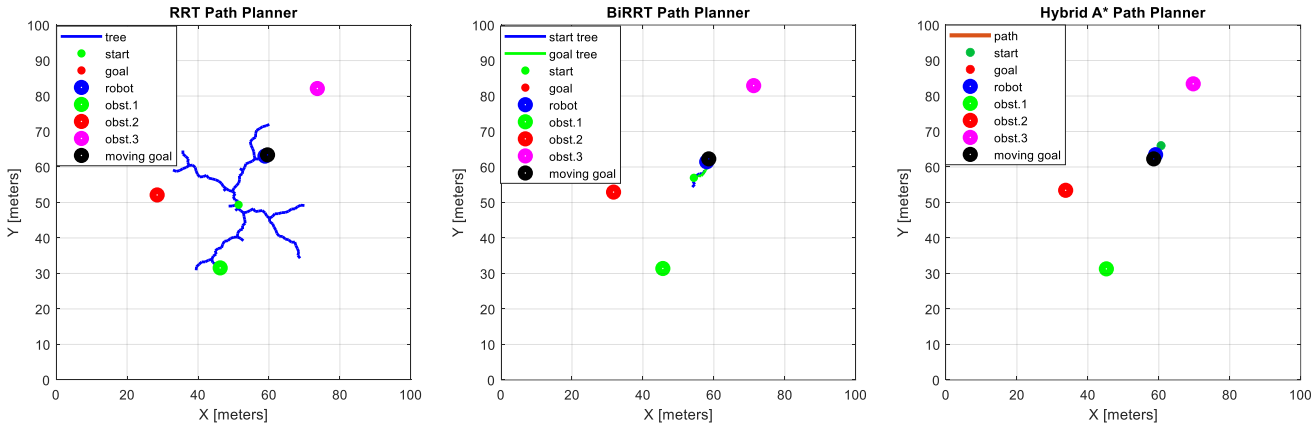
Fig. 4. Results of dynamic environment scenario: (a) Path length, (b) Computation time

TABLE I. SUMMARY RESULTS OF DYNAMIC ENVIRONMENT SCENARIO

No. Goal Point	Goal Point	Algorithm	Path Length (m)	Computation time(s)
1	(95, 90)	RRT	127.5336	18.830662
		BiRRT	123.7087	18.509972
		HA*	110.5873	18.062252
2	(80, 65)	RRT	97.9781	15.026167
		BiRRT	89.9578	14.003256
		HA*	82.2093	13.780402
3	(55, 75)	RRT	92.9828	13.670333
		BiRRT	85.7447	12.977735
		HA*	75.2899	12.696381
4	(70, 50)	RRT	78.8178	12.008367
		BiRRT	73.1622	10.949913
		HA*	64.6907	10.722144
5	(60, 60)	RRT	77.9285	11.261541
		BiRRT	72.7915	10.834884
		HA*	65.2354	10.759183

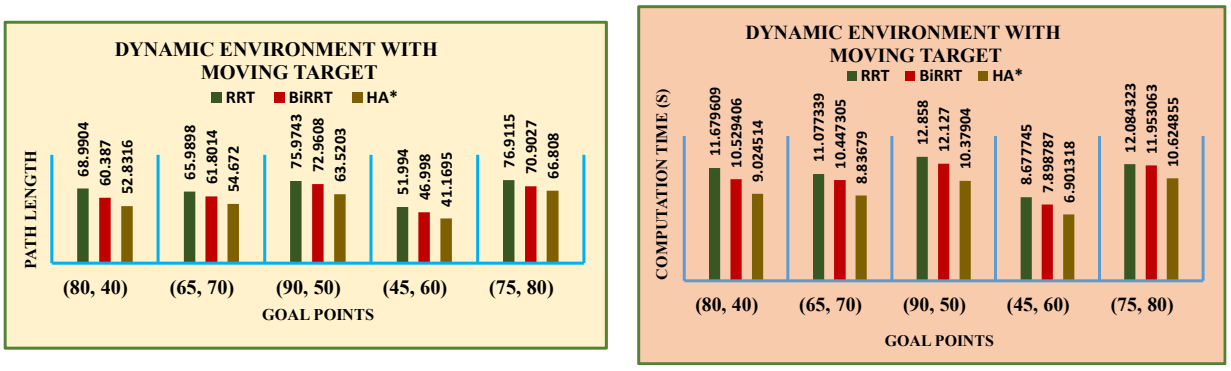
2) Dynamic Environment with Moving Target

This scenario is considered an environment whose dimensions are (100, 100) with three dynamic obstacles and a moving target. The initial positions of these obstacles are (40, 30) moving toward (80, 40), (60, 60) moving toward (20, 50), and (50, 90) moving toward (80, 80). The obstacles are moving in different directions. (20, 20) is the starting point, and various goal points are used. The maximum connection distance is 1, and the validation distance is 1. Fig. 5 depict the planned paths for example of goal points in this environment using the RRT algorithm (a), the BiRRT technique (b), and the HA* technique (c). In this example, the initial position of the goal point is (75, 80) and moving toward (10, 10). Fig. 6 shows (a) the length of the path in meters and (b) the time required to reach the destination in seconds. Table II provides a summary of the results obtained. The example of a goal point is explained as Fig. 5 and Fig. 6.



(a) (b) (c)

Fig. 5. Path planning in dynamic environment with (75, 80) moving target using: (a) RRT, (b) BiRRT, (c) HA*



(a) (b)

Fig. 6. Results of dynamic environment with moving target scenario: (a) Path length, (b) Computation time

TABLE II. SUMMARY RESULTS OF DYNAMIC ENVIRONMENT WITH MOVING TARGET SCENARIO

No. Goal Point	Goal Point	Algorithm	Path Length (m)	Computation time(s)
1	(80, 40)	RRT	68.9904	11.67961
		BiRRT	60.3870	10.52941
		HA*	52.8316	9.024514
2	(65, 70)	RRT	65.9898	11.07734
		BiRRT	61.8014	10.44731
		HA*	54.6720	8.83679
3	(90, 50)	RRT	75.9743	12.85878
		BiRRT	72.9608	12.12719
		HA*	63.5203	10.37904
4	(45, 60)	RRT	51.9940	8.677745
		BiRRT	46.9980	7.898787
		HA*	41.1695	6.901318
5	(75, 80)	RRT	76.9115	12.08432
		BiRRT	70.9027	11.95306
		HA*	66.8080	10.62486

V. TRAJECTORY TRACKING CONTROL AND OPTIMIZATION

The problem of trajectory tracking is described as follows: It is supposed that the robot does have the position: $p_r = [x_r \quad y_r \quad \theta_r]^T$, and the target does have the position: $p_t = [x_t \quad y_t \quad \theta_t]^T$. The goal is to identify control laws for robots' linear and angular velocities (v, w), such as [48] :

$$\lim_{t\to\infty}|x_r(t) - x_t(t)| = 0 \quad \text{and} \quad \lim_{t\to\infty}|y_r(t) - y_t(t)| = 0 \text{ and } \lim_{t\to\infty}|\theta_r(t) - \theta_t(t)| = 0$$

The following equation can be used to express the distance between the robot's current location and its target position [37]:

$$d = \sqrt{\left(X_{target} - X_{robot}\right)^2 + (Y_{target} - Y_{robot})^2} \tag{11}$$

The desired angle of the trajectory θ is computed as follows:

$$\theta = \tan^{-1}\frac{Y_{target} - Y_{robot}}{X_{target} - Y_{robot}} \tag{12}$$

Where: X_{target} and Y_{target} are the desired target or the new position, X_{robot} and Y_{robot} are the current position of the robot or the old position in the x-y plane.

A. Some Common Proposed Wheeled Mobile Robot Trajectory Tracking Based on PID Controller

The PID controller is used to track the trajectory of the autonomous wheeled mobile robot. The kinematics model in Equation (4) describes the robot's position (x, y, θ) and its linear and angular velocities (V, ω). In Fig. 7, the robot travels in the $X - Y$ plane. The reference trajectory is represented by $R_r(x_r, y_r)$, and the robot's coordinates are $R_f(x_f, y_f)$ on the $X - Y$ plane. The tracking error vector,

denoted by $\hat{X}$, represents the difference between the follower robot and the reference trajectory [36].

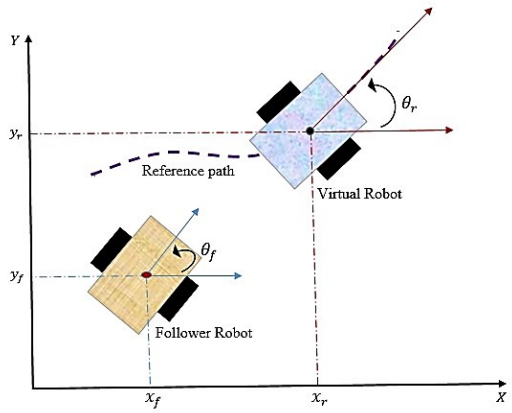


Fig. 7. Mobile robot trajectory tracking [36]

$$\hat{X} = \begin{bmatrix} x_r - x_f \\ y_r - y_f \\ \theta_r - \theta_f \end{bmatrix} = \begin{bmatrix} \hat{X}_x \\ \hat{X}_y \\ \hat{X}_\theta \end{bmatrix} \quad (13)$$

Equation (11) has been used to express the distance between the robot's current location and reference position. The following formula has been used to determine the tracking error [36]:

$$e_d = \sqrt{(\hat{X}_x)^2 + (\hat{X}_y)^2} - d_o \quad (14)$$

Where d_o is a small positive constant that is utilized to ensure that there is minimal separation between the follower robot and the reference trajectory. The error e_d is reduced to zero using the standard PID controller to modify the linear velocity of the follower robot V_f as follows [36]:

$$V_f = k_p e_d + k_i \int e_d dt + k_d \frac{d}{dt} e_d \quad (15)$$

Equation (12) has been used to express angle of the reference trajectory. The angle error between the follower robot and the reference can be expressed as:

$$e_\theta = (\theta_r - \theta_f) = \hat{X}_\theta \quad (16)$$

The error e_θ is reduced to zero using the standard PID controller to modify the angular velocity of the follower robot ω_f as follows:

$$\omega_f = k_p e_\theta + k_i \int e_\theta dt + k_d \frac{d}{dt} e_\theta \quad (17)$$

Fig. 8 depicts the overall block structure of a mobile robot that includes a trajectory-tracking control system.

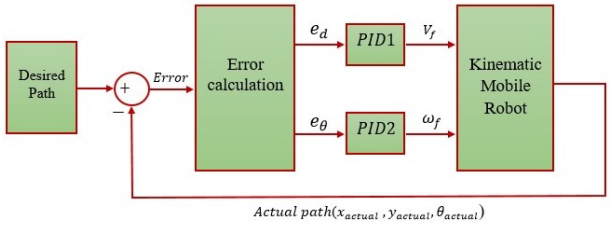


Fig. 8. A mobile robot's overall block diagram

B. Intelligent PID and Optimization Algorithms

Due to characteristics like their durability, simplicity, transparency, reliability, and high efficiency, PID controllers are especially well-liked and accepted for control in process industries. The optimization of PID control settings has been and continues to be an active field of research. The basic objectives of PID control settings are to minimize steady-state overshoot and shorten settling time. Several strategies have been employed to tune the PID. Approaches for adjusting PID parameters are classified as either classical, such as Ziegler-Nichols, Cohen-Coon, and the Gain and Phase Method, or meta-heuristics, such as the PSO, ACO, ABC, BA, and Cuckoo optimization algorithm, etc. Approaches based on meta-heuristics solve problems by offering almost optimal answers in a reasonable amount of time. Meta-heuristics have increased in popularity in recent years as a result of their effectiveness and efficiency in resolving enormous and complex problems. Each meta-heuristic algorithm uses a unique combination of local and global search randomization and frequently produces varied result [54].

In this paper, hybrid BWOA-PSO (HBPO) new optimization algorithms have been presented to obtain PID parameters (k_p, k_i, k_d) for enhancing mobile robot trajectory tracking and compared results with Black widow Optimization algorithm (BWOA), Salp Swarm algorithm (SSA), Crow Search Algorithm (CSA), Flower Pollination algorithm (FPA), and Particle Swarm Optimization (PSO).

C. Hybrid Black Widow Optimization Algorithm and Particle Swarm Optimization Algorithm (HBPO)

The combination of two optimization algorithms yields a better solution to a variety of optimization problems and results in faster convergence. In this thesis, the BWOA and PSO are combined to enhance performance. The BWOA-PSO approach has been developed to combine the benefits of the black widow optimization algorithm and particle swarm optimization.

The following details make the PSO algorithm a popular optimization algorithm: Only three parameters (inertia weight, cognitive ratio, and social ratio) control PSO, making it easy to implement and code. PSO is also adaptable enough to combine with other optimization algorithms [55]. In PSO, a swarm of particles searches for the best solution in a D-dimensional search space. $V_i = [v_{i1}, v_{i2}, \ldots\ldots, v_{iD}]$ is the current velocity vector of each particle i. In addition, $X_i = [x_{i1}, x_{i2}, \ldots\ldots, x_{iD}]$ is a current position vector. D is the number of dimensions. V_i and X_i are first initialized at random in the PSO process. Then, in each iteration, the best position obtained by particle i $Pbest_i = [Pbest_{i1}, Pbest_{i2}, \ldots\ldots, Pbest_{iD}]$, and the best position found by the particles in the swarm $= [Gbest_1, Gbest_2, \ldots\ldots, Gbest_D]$, are calculated. Instruct particle i to update its position and velocity according to (4.9) and (4.10) [55]:

$$[v_{id}(t+1) = w\, v_{id}(t) + c_1 r_1 (Pbest_{id}(t) - x_{id}(t)) + c_2 r_2 (Gbest_d(t) - x_{id}(t))] \quad (18)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (19)$$

By updating the low pheromone rate search agent rather than the entire search agent before the next iteration, the BWOA enhances fitness quality and encourages a better balance between the exploitation and exploration processes. The BWOA Features are easy to use, increase convergence speed, significantly reduce trapping of local optima, have acceptable accuracy, and reduce complexity [56]. The BWOA has some drawbacks in that it cannot ensure that it will find the best solution because it is a meta-heuristic algorithm.The BWOA mathematical modeling is described as follows [57]:

Movement: The spider made linear and spiral movements inside the web, as shown in the following Equation:

$$\vec{x}_i(t+1) = \begin{cases} \vec{x}_*(t) - m\,\vec{x}_{r1}(t), & if\ rand \le 0.3 \\ \vec{x}_*(t) - \cos(2\pi\beta)\vec{x}_i(t), & in\ other\ case \end{cases} \quad (20)$$

Where: $\vec{x}_i(t+1)$: indicates the search agent's new position and the spider's movement. $\vec{x}_*(t)$: The best search agent obtained in the previous iteration. m: is a float number produced at random between [0.4, 0.9]. r_1: is the random integer number generated in the interval from 1 to the size of the maximum of search agents. $\vec{x}_{r1}(t)$: is the r_1 search agent that has been chosen. with $i \neq r_1$. β: A random float number computed in the range [- 1.0, 1.0]. $\vec{x}_i(t)$: The current search agent.

Pheromones: play a key role in spider mating behavior. The following equation describes the pheromone rate value of black widow spiders [57]:

$$pheromone(i) = \frac{fitness_{max} - fitness(i)}{fitness_{max} - fitness_{min}} \quad (21)$$

Where: $fitness_{max}$ & $fitness_{min}$: The current generation's worst and best fitness values, respectively. $fitness(i)$: represents the i^{th} search agent's current fitness value.

In equation (21), the pheromone vector contains the normalized fitness in the interval [0, 1]. For low pheromone rate values of 0.3 or below, the search agent update in the following equation [57]:

$$\vec{x}_i(t) = \vec{x}_*(t) + \frac{1}{2}\left[\vec{x}_{r1}(t) - (-1)^{\sigma} * \vec{x}_{r2}(t)\right] \quad (22)$$

Where: $\vec{x}_i(t)$: is the updated search agent with a low pheromone rate. $r1$ & $r2$: are the random integer numbers generated in the interval from 1 to the size of the maximum of search agents. With $r1 \neq r2$. $\vec{x}_{r1}(t)$ & $\vec{x}_{r2}(t)$ **:** are the selected r1 and r2 search agents, respectively. $\vec{x}_*(t)$: The best search agent obtained in the previous iteration. σ: is a randomly generated binary number, $\in [0, 1]$.

The adjustment of PID controllers refers to the process of resolving an optimization problem through the sequential use of BWOA and PSO, which is called HBPO. Initially, BWOA performs optimization and the resulting optimized K_p, K_i, and K_d values are then used as the initial input for PSO. Fig. 9 depicts the Pseudo-Code of the HBPO algorithm.

Algorithm HBPO Pseudo-Code for a Minimization Problem

Start BWOA
Create the initial population
while *iter* < *itermax* **do**
Initialization random of parameters m and β, where $0.4 \le m \le 0.9$ and $-1.0 \le \beta \le 1.0$
  if random < 0.3 **then**
    $\vec{x}_i(t+1) = \vec{x}_*(t) - m\,\vec{x}_{r1}(t)$
  else
    $\vec{x}_i(t+1) = \vec{x}_*(t) - cos(2\pi\beta)\vec{x}_i(t)$
  end if
Calculate the pheromone for each search agent using an equation (21)
Update search agents with low pheromone values using equation (22)
Determine the $\vec{x}_i(t+1)$ fitness value of the new search agents.
  if $\vec{x}_i(t+1) < \vec{x}_*(t)$ **then**
    $\vec{x}_*(t) = \vec{x}_i(t+1)$
  end if
Iteration = iteration + 1
end while
Display $\vec{x}_*(t)$, the best optimal solution
end BWOA
Start PSO
Initialization taken from the output of BWOA results
Define the swarm size S and the number of dimensions D
for each particle $i \in [1...S]$
Randomly generate X_i and V_i, and evaluate the fitness of X_i denoting it as $f(X_i)$
Set $Pbest_i = X_i$ and $f(Pbest_i) = f(X_i)$
end for
Set $Gbest = Pbest1$ and $f(Gbest) = f(Pbest1)$
for each particle $i \in [1...S]$
if $f(Pbest_i) < f(Gbest)$ **then**
$f(Gbest) = f(Pbest_i)$
end if
end for
while $t <$ maximum number of iterations
for each particle $i \in [1...S]$
Evaluate its velocity $v_{id}(t+1)$ using Equation (18)
Update the position $x_{id}(t+1)$ of the particle using Equation (19)
if $f(x_i(t+1)) < f(Pbest_i)$ **then**
$Pbest_i = x_i(t+1)$
$f(Pbest_i) = f(x_i(t+1))$
end if
if $f(Pbest_i) < f(Gbest)$ **then**
$Gbest = Pbest_i$
$f(Gbest) = f(Pbest_i)$
end if
end for
$t = t + 1$
end while
Display the best optimal solution

Fig. 9. Pseudo-Code of the HBPO algorithm

D. Simulation and Results of Control Trajectory Tracking

The results of tests performed to evaluate the effectiveness of the controllers and optimization algorithms mentioned in this paper. Fig. 10 depicts the block diagram of the proposed PID controllers for DDMR path tracking with an optimization algorithm to increase the performance of these controllers.

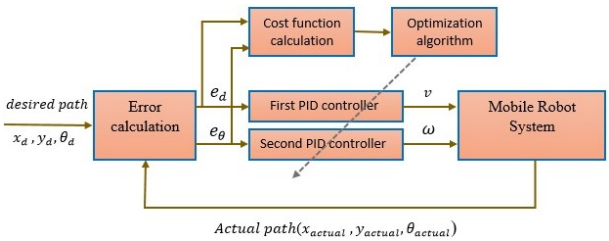


Fig. 10. A block diagram of the PID controller auto-tuning using optimization algorithms

The simulation experiment had been carried out on the following computer specifications: The operating system is Windows 10, which is a 64-bit operating system, and the processor is an Intel(R) Core(TM) i7-1165G7U CPU @ 2.80 GHz. Physical parameters taken from [58] include the wheel radius of 0.033m and the distance between two wheels of 0.105m. The performance of controllers is determined using the ISE criterion, which is given [59]:

$$ISE = \int_0^\infty [e(t)]^2 \, dt \tag{23}$$

The integral square of the error method, as shown in Equation (5.2), has been used to simulate the objective function used in the proposed optimization algorithms [59].

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

The proposed algorithms are used to determine the best values for the two PID controller parameters. The swarm population size of 30 and the maximum number of iterations of 100 have been taken from [40]. The parameters of the proposed algorithms are as follows: In PSO and HBPO, c_1, c_2 are 2, and w is 0.4. In SSA, c_3 is rand(). In CSA, AP is 0.8 and fl is 4. In FPA, c is 0.5 and probability (P) is 0.8. In BWOA and HBPO, P is rand().

To reduce the tracking error between the desired and actual path Several experiments have been carried out to solve this problem using the proposed algorithms-based PID controllers. The simulation results tested with three trajectory shapes, Step, Circular, Infinity, are as follows:

1) Step Trajectory

Fig. 11 depicts the results of the testing of the mobile robot for following the step path utilizing the suggested algorithm-based PID controllers.

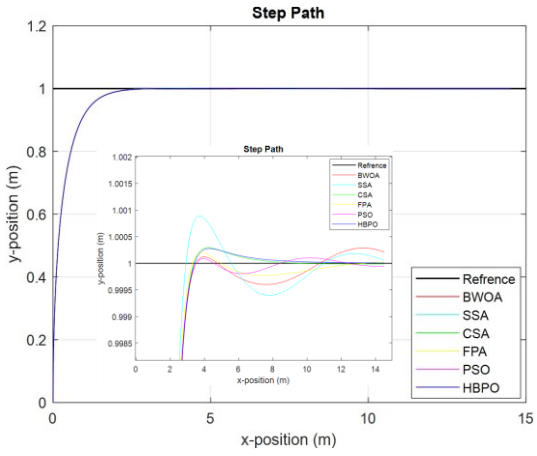


Fig. 11. Step Trajectory

The BWOA, SSA, CSA, FPA, PSO, and HBPO algorithms optimize the PID controllers' parameters shows in Table III. Fig. 12 illustrates the convergence curve of the objective functions for these algorithms, with HBPO showing faster convergence. Fig. 13 and Fig. 14 display the step response and theta response for the trajectory using the HBPO algorithm, respectively. Table IV shows the specifications of the optimized two PID controller's response in terms of steady-state error (SSE), overshoot (OS), and rise time, in addition to the cost function and computation time.

TABLE III. THE TUNED PARAMETERS OF PID CONTROLLERS USING THE PROPOSED ALGORITHMS

Parameters / Algorithm	K_{p1}	K_{i1}	K_{d1}	K_{p2}	K_{i2}	K_{d2}
BWOA	100	100	0	45.6146	0	100
SSA	42.9632	50.3425	20.0289	62.7554	1.8723	81.7839
CSA	11.9689	50.9254	0	39.0076	0.8895	86.9655
FPA	55.3151	88.8281	10.0719	66.0847	1.4761	88.9655
PSO	94.3493	60.3425	20.4682	57.2069	0	100
HBPO	73.0033	24.0381	0	45.6146	0	100

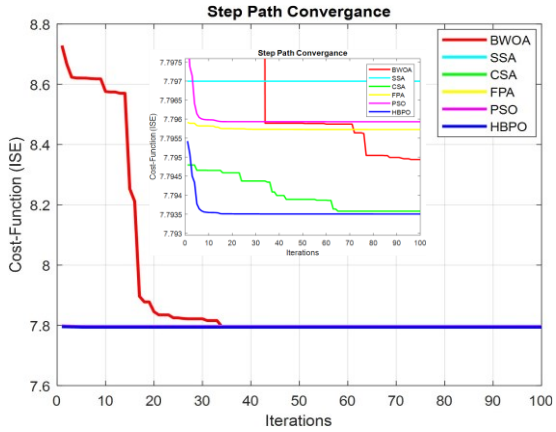


Fig. 12. Convergence curve of step trajectory

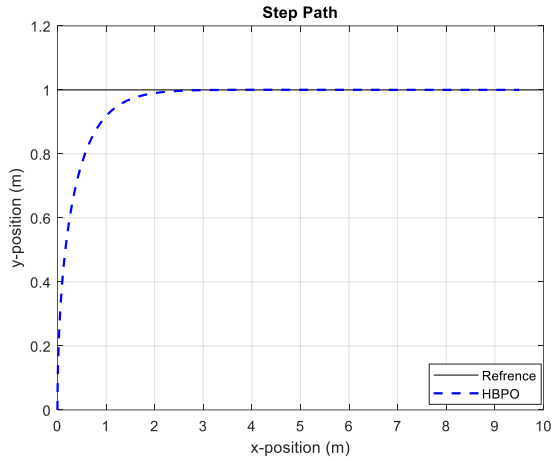


Fig. 13. The step trajectory response

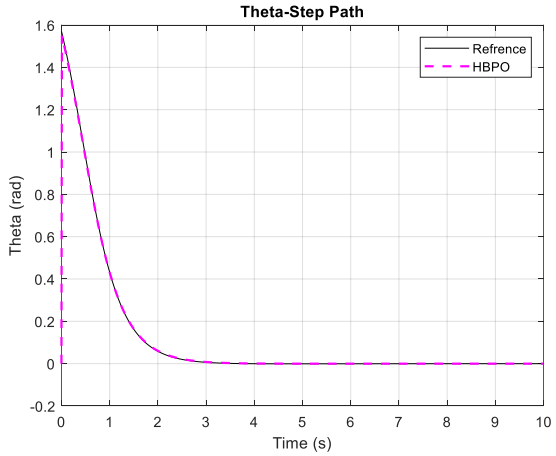
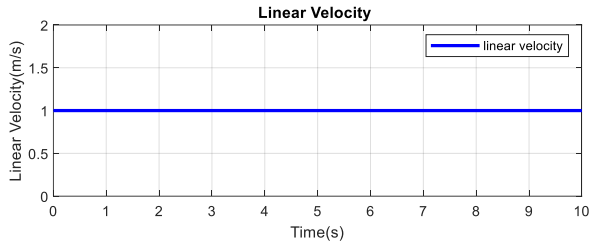


Fig. 14. Theta response for the step trajectory

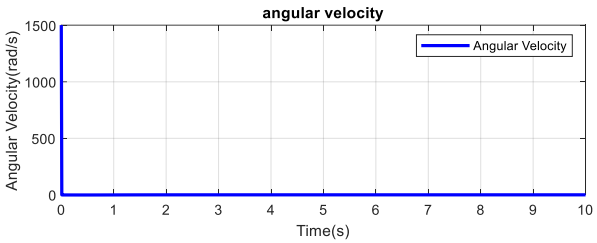
TABLE IV. COST FUNCTIONS, COMPUTATION TIME, AND THE SPECIFICATIONS OF THE OPTIMIZED TWO PID CONTROLLER'S RESPONSE

Algorithm	Cost function (ISE)	Steady State Error	Os%	Rise Time	Computation time (s)
BWOA	7.794938	2.140e-04	0.012	1.2726	207.5861311
SSA	7.796996	5.716e-05	0.089	1.2724	205.8857722
CSA	7.793578	1.204e-05	0.03	1.2696	191.3467333
FPA	7.795929	2.115e-05	0.02	1.2674	188.1626445
PSO	7.795727	5.340e-05	9e-3	1.2681	187.2791222
HBPO	7.793504	6.223e-06	0.028	1.2701	188.7617778

Because the HBPO algorithm converges faster, the two PID controller parameters computed by it have been used to examine the linear and angular velocities, right and left wheel velocities, position error, and theta error. In Fig. 15 (a), the mean linear velocity of the mobile robot is 1m/s, while the mean angular velocity is 1.3453 rad/sec in Fig. 15 (b). Fig. 16 (a) and Fig. 16 (b) demonstrate smooth velocity values for the right and left wheels without spikes. The position and theta trajectory errors for the mobile robot motion are depicted in Fig. 17 (a) and Fig. 17 (b), respectively. The mean tracking error is 0.0163m for position and 0.0028 rad for theta.

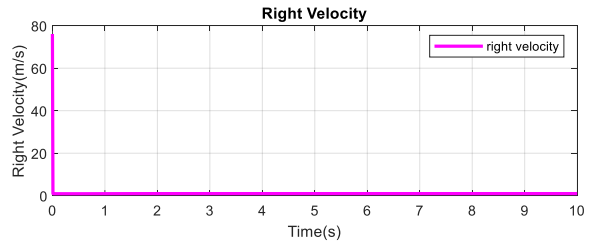


(a) Linear Velocity

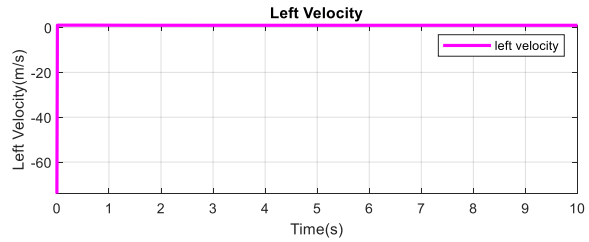


(b) Angular Velocity

Fig. 15. Linear and Angular Velocities

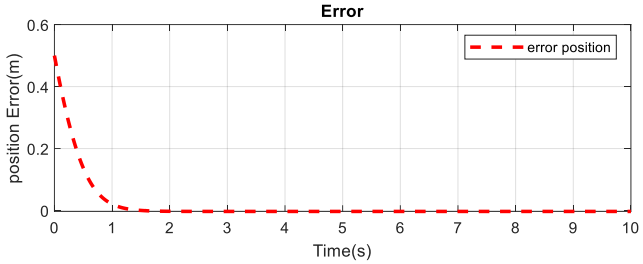


(a) Right Velocity

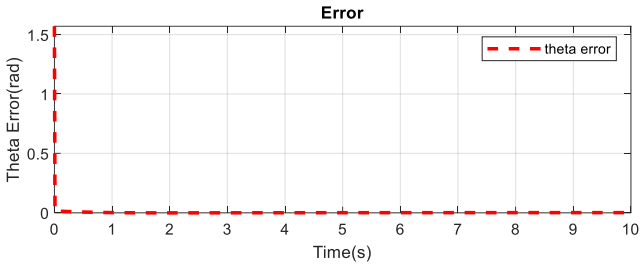


(b) Left Velocity

Fig. 16. Right and Left Velocities



(a) Position Error



(b) Theta Error

Fig. 17. Position and Theta Error

2) Circular Trajectory

The following equations can be used to describe the 2-WMR's required circle path [60] :

$$\left.\begin{aligned} x_{ref} &= 1+\cos\left(\frac{t}{10}\right) \\ y_{ref} &= \sin\left(\frac{t}{10}\right) \end{aligned}\right\} \quad (25)$$

Where: x_{ref} and y_{ref} represent trajectory reference.

The results of the testing of the mobile robot for following the circular path are depicted as follows: Table V shows the mean and RMS error values for each algorithm, as well as the cost function and computation time. Table VI shows the PID controllers' parameters. Graphically, the convergence curve is shown in Fig. 18. The circular and theta responses for the trajectory using the HBPO algorithm are shown in Fig. 19 and Fig. 20, respectively.

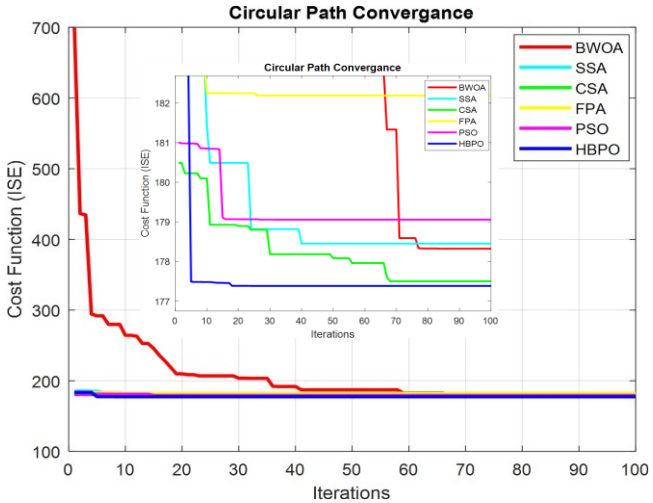


Fig. 18. Convergence curve of circular trajectory

TABLE V. THE MEAN, RMS, THE COST FUNCTION, AND COMPUTATION TIME

Algorithm	Cost function (ISE)	Mean absolute theta error	RMS theta error	Mean position error	RMS position error	Computation time(s)
BWOA	178.3209477	0.010344	0.0392	0.077	0.42023	210.6478667
SSA	178.4514863	9.755e-03	0.0783	0.070955	0.420155	237.7954556
CSA	177.5024909	5.644e-03	0.01828	0.07851	0.420955	203.5981778
FPA	182.1830604	0.014711	0.203044	0.09808	0.441666	200.9670556
PSO	179.054176	0.013466	0.10554	0.06951	0.420788	225.2624778
HBPO	177.3844145	4.8e-03	0.018166	0.06788	0.421388	213.9983889

TABLE VI. THE TUNED PARAMETERS OF PID CONTROLLERS

Parameters / Algorithm	K_{p1}	K_{i1}	K_{d1}	K_{p2}	K_{i2}	K_{d2}
BWOA	4.9523	3.5044	0.5044	7.6460	4.9261	0
SSA	0	15.3075	57.6537	74.3207	79.0809	78.2815
CSA	0	12.0816	3.6663	71.8918	3.4893	57.4392
FPA	98.4396	24.9320	49.1363	97.8517	93.0542	93.4386
PSO	3.1701	11.0525	95.5090	52.6500	9.5805	52.8107
HBPO	17.1237	58.1017	71.6045	28.4260	0	82.3290

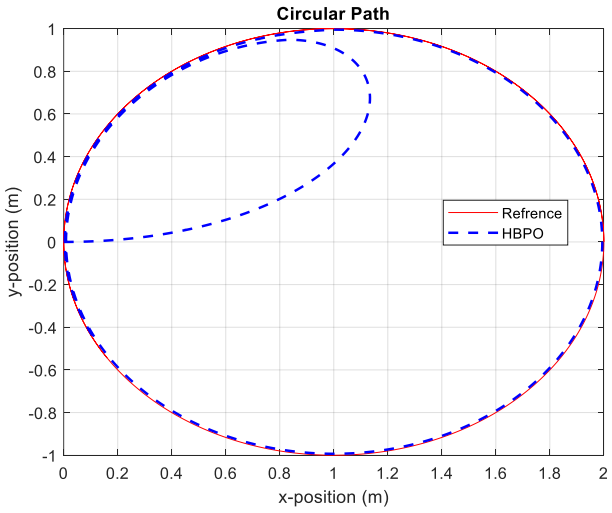


Fig. 19. Circular Trajectory

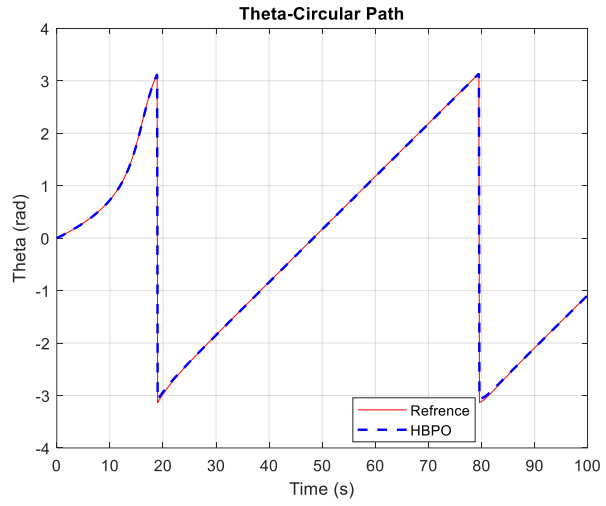
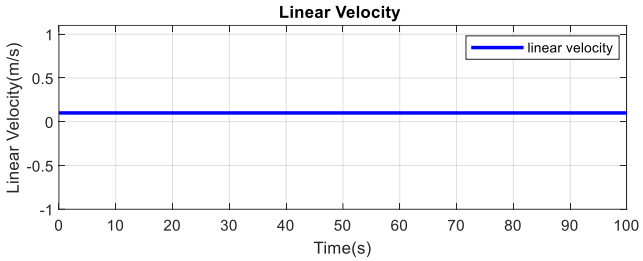
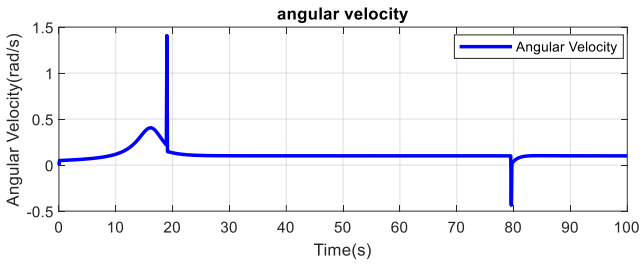


Fig. 20. Theta response

Because the HBPO algorithm converges faster, the two PID controller parameters computed by it have been used to examine the linear and angular velocities, right and left wheel velocities, position error, and theta error. In Fig. 21 (a), the mean linear velocity of the mobile robot is 0.1000 m/s, while the mean angular velocity is 0.1141 rad/sec in Fig. 21 (b). Fig. 22 (a) and Fig. 22 (b) demonstrate velocity values for the right and left wheels with spikes. The position and theta trajectory errors for the mobile robot motion are depicted in Fig. 23 (a) and Fig. 23 (b), respectively. The mean tracking error is 0.0698m for position and mean tracking absolute error is 0.0061rad for theta.

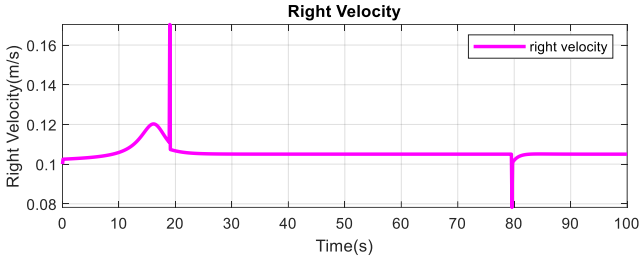


(a) Linear Velocity

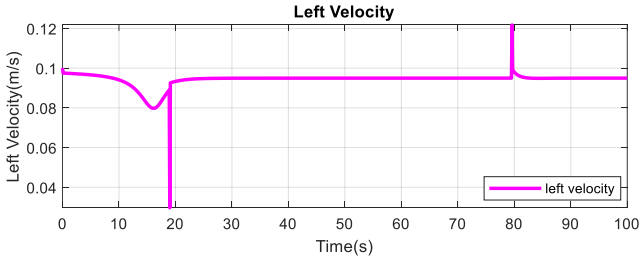


(b) Angular Velocity

Fig. 21. Linear and Angular Velocities

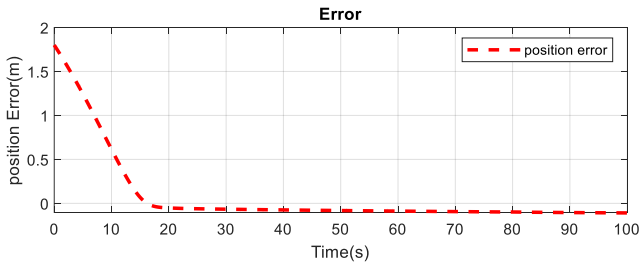


(a) Right Velocity

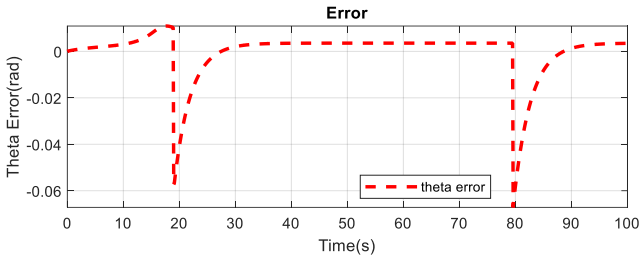


(b) Left Velocity

Fig. 22. Right and Left Velocities



(a) Position Error



(b) Theta Error

Fig. 23. Position and Theta Error

3) Infinity Trajectory

The following equations can be used to describe the 2-WMR's required infinity path [60]:

$$\left.\begin{aligned} x_{ref} &= 1.75 + 1.75 * \sin(pi * \frac{t}{25}) \\ y_{ref} &= \sin(2 * pi * \frac{t}{25}) \end{aligned}\right\} \quad (26)$$

Where: x_{ref} and y_{ref} represent trajectory reference.

The results of the testing of the mobile robot for following the infinity path are depicted as follows: Table VII shows the mean and RMS error values for each algorithm, as well as the cost function and computation time. Table VIII shows the PID controllers' parameters. Graphically, the convergence curve is shown in Fig. 24. The infinity and theta responses for the trajectory using the HBPO algorithm are shown in Fig. 25 and Fig. 26, respectively.

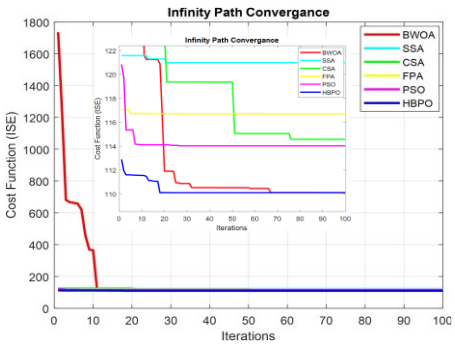


Fig. 24. Convergence curve of infinity trajectory

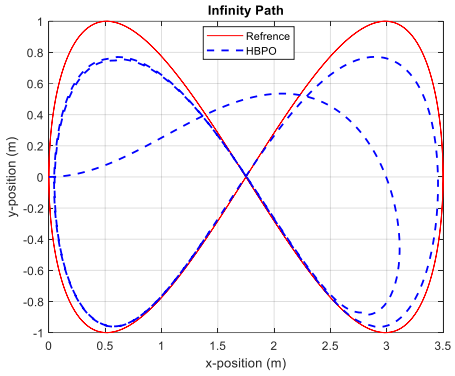


Fig. 25. Infinity Trajectory

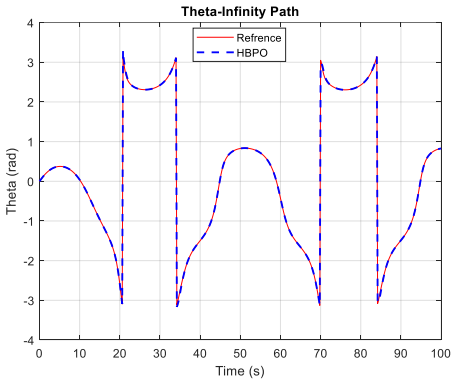


Fig. 26. Theta response for the Infinity trajectory

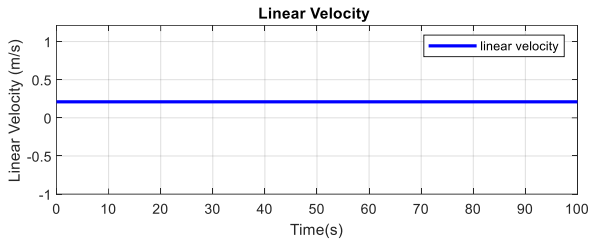
TABLE VII. THE MEAN, RMS, THE COST FUNCTION, AND COMPUTATION TIME

Algorithm	Cost function (ISE)	Mean absolute theta error	RMS theta error	Mean position error	RMS position error	Computation time(s)
BWOA	110.1105399	5.844e-03	0.016455	0.10622	0.48164	229.3048556
SSA	120.9838025	0.014055	0.035766	0.115211	0.49222	218.3599
CSA	114.5678863	0.02006	0.034533	0.129911	0.48838	224.4454778
FPA	116.7042237	0.01357	0.041133	0.105355	0.48193	204.961556
PSO	114.0221917	5.677e-03	0.021477	0.1074	0.481588	215.9737222
HBPO	110.1002824	7.955e-03	0.01984	0.105344	0.47707	219.2750444

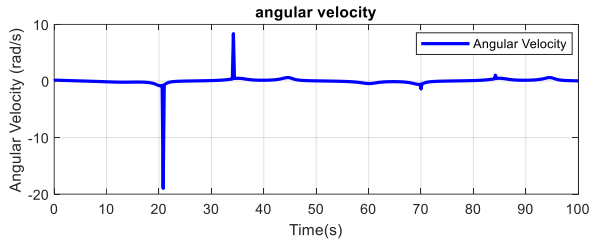
TABLE VIII. THE TUNED PARAMETERS OF PID CONTROLLERS

parameters / Algorithm	K_{p1}	K_{i1}	K_{d1}	K_{p2}	K_{i2}	K_{d2}
BWOA	29.9948	82.8134	92.1403	22.7539	9.0083	92.1403
SSA	16.9525	10.1654	5.9273	7.2740	12.0810	17.6589
CSA	11.6308	5.2432	71.1011	69.7079	13.3668	2.4071
FPA	42.2155	6.0082	58.6787	17.1339	3.0096	85.0459
PSO	76.5364	13.5711	97.9813	67.6214	87.9217	5.7824
HBPO	3.1741	85.5424	69.1278	100	63.8467	0.5572

Because the HBPO algorithm converges faster, the two PID controller parameters computed by it have been used to examine the linear and angular velocities, right and left wheel velocities, position error, and theta error. In Fig. 27 (a), the mean linear velocity of the mobile robot is 0.2100 m/s, while the mean angular velocity is -0.0130 rad/sec in Fig. 27 (b). Fig. 28 (a) and Fig. 28 (b) demonstrate velocity values for the right and left wheels with spikes. The position and theta trajectory errors for the mobile robot motion are depicted in Fig. 29 (a) and Fig. 29 (b), respectively. The mean tracking error is 0.1054 m for position and mean tracking absolute error is 0.0032 rad for theta.

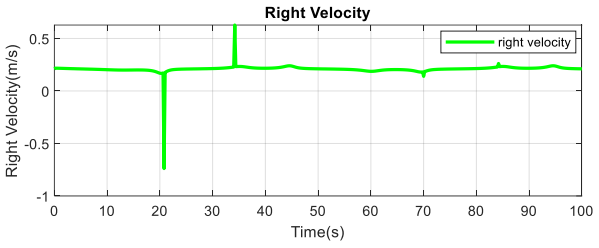


(a) Linear Velocity

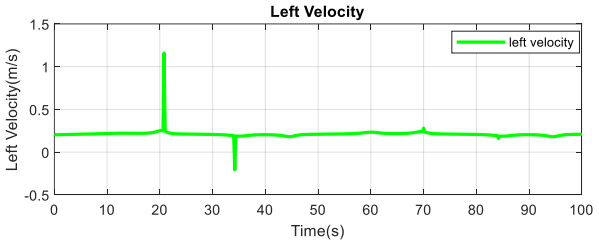


(b) Angular Velocity

Fig. 27. Linear and Angular Velocities

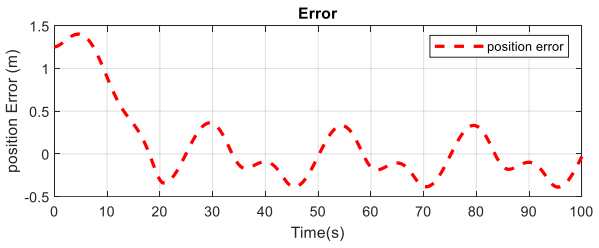


(a) Right Velocity

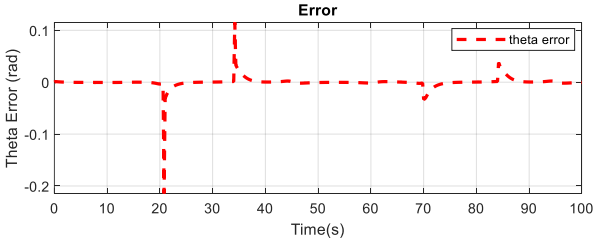


(b) Left Velocity

Fig. 28. Right and Left Velocities



(a) Position Error



(b) Theta Error

Fig. 29. Position and Theta Error

VI. DISCUSSION OF THE RESULTS

The following are a number of major findings that have been drawn from the results of the tests:

A. Robot Path Planning

Taking into account the physical constraints of the moving item by HA*, such as turning and acceleration rates. To recreate the steering angles for a specific car, a tree is grown from the nodes; therefore, the simulation results demonstrate that HA* outperforms the other algorithms by creating a collision-free path that is smoother and shorter than the paths of RRT and BiRRT. At percentages, the results showed that HA* outperformed RRT and BiRRT by reducing the path length by 16.25% and 10.63%, respectively, and the running time by 6.82% and 1.94%, respectively. Moreover, in a dynamic environment with a moving target, HA* demonstrated even better performance by reducing the path length by 17.91% and 10.87% compared to RRT and BiRRT, respectively, and the running time by 18.82% and 13.57%, respectively.

B. Robot Trajectory Tracking

In trajectory tracking, the results of the simulation show that HBPO performs better than the other proposed algorithms. The results demonstrated that the HBPO had been more effective and able to reduce steady-state error at percentages in step trajectory to 97.09%, 8.3176%, 48.3140%, 70.5768%, and 88.3464% compared to the BWOA, SSA, CSA, FPA, and PSO, respectively. The computation times to 9.0685%, 8.3176%, and 1.351% compared to the BWOA, SSA, and CSA, respectively. In the circular trajectory, the results in percentages of RMS theta error are 53.66%, 76.7995%, 0.6236%, 91.0532%, and 82.7876% compared to the BWOA, SSA, CSA, FPA, and PSO, respectively; the percentages of RMS position error are 4.5913% compared to the FPA; and computation time is 10.0074% and 5.0004% compared to the SSA and PSO, respectively. In the infinity trajectory, the results in percentages of RMS theta error are 44.5283%, 42.5477%, 51.7662%, and 7.6221% compared to the SSA, CSA, FPA, and PSO, respectively. The percentages of RMS position error are 0.9488%, 3.0779%, 2.3158%, 1.0084%, and 0.9381% compared to the BWOA, SSA, CSA, FPA, and PSO, respectively. The computation times are 1.3737% and 2.335% compared to the BWOA and CSA, respectively.

VII. CONCLUSION AND FUTURE WORKS

One of the technologies that is currently being developed is autonomous mobile robots (AMR). Due to their importance and applications in society today. A system that can operate in different environments is the mobile robot. As a result, the robot must be able to travel quickly through its environment and avoid any obstacles that are placed in its path. Therefore, at least two steps must be taken in the design of mobile robots in order for them to be intelligently controlled and work independently when they move from one location to another. To prevent motion collisions at first, path planning is essential. Tracking the robot's trajectory is a crucial second task. In this paper, we investigate the path planning of a mobile robot with dynamic, and dynamic obstacles with a moving goal Using the RRT, BiRRT, and HA* algorithms.

By producing collision-free paths that are smoother and shorter than their RRT and BiRRT comparable paths, HA* outperforms the other algorithms. To reduce tracking deviations between the robot's actual route and the reference trajectory, the DDMR's kinematic model has been utilized to control the path-tracking and PID controller. In this work, HBPO outperforms the other approaches in almost every aspect of simulation results.

In future work, the proposed work will be evaluated for its effectiveness in various scenarios and along various paths using simulated investigations on the prototype, a real mobile robot. Designing and implementing a dynamic model of DDMR with the same proposed PID and optimization algorithms that were used in this study and comparing it with a kinematic model.

REFERENCES

[1] M. A. K. Niloy *et al.*, "Critical Design and Control Issues of Indoor Autonomous Mobile Robots: A Review," *IEEE Access*, vol. 9, pp. 35338–35370, 2021, doi: 10.1109/ACCESS.2021.3062557.

[2] A. M. Alshorman, O. Alshorman, M. Irfan, A. Glowacz, F. Muhammad, and W. Caesarendra, "Fuzzy-based fault-tolerant control for omnidirectional mobile robot," *Machines*, vol. 8, no. 3, pp. 2–20, Sep. 2020, doi: 10.3390/MACHINES8030055.

[3] H. Y. Zhang, W. M. Lin, and A. X. Chen, "Path planning for the mobile robot: A review," *Symmetry (Basel).*, vol. 10, no. 10, pp. 2–17, 2018, doi: 10.3390/sym10100450.

[4] F. Rubio, F. Valero, and C. Llopis-Albert, "A review of mobile robots: Concepts, methods, theoretical framework, and applications," *Int. J. Adv. Robot. Syst.*, vol. 16, no. 2, pp. 1–22, 2019, doi: 10.1177/1729881419839596.

[5] G. Ren, T. Lin, Y. Ying, G. Chowdhary, and K. C. Ting, "Agricultural robotics research applicable to poultry production: A review," *Comput. Electron. Agric.*, vol. 169, p. 105216, 2020, doi: 10.1016/j.compag.2020.105216.

[6] D. Patil, M. Ansari, D. Tendulkar, R. Bhatlekar, V. N. Pawar, and S. Aswale, "A Survey on Autonomous Military Service Robot," *Int. Conf. Emerg. Trends Inf. Technol. Eng. ic-ETITE 2020*, pp. 1–7, 2020, doi: 10.1109/ic-ETITE47903.2020.78.

[7] S. G. Tzafestas, "Mobile Robot Control and Navigation: A Global Overview," *Journal of Intelligent & Robotic Systems*, vol. 91, pp. 35-58, 2018.

[8] M. B. Alatise and G. P. Hancke, "A Review on Challenges of Autonomous Mobile Robot and Sensor Fusion Methods," *IEEE Access*, vol. 8, pp. 39830–39846, 2020, doi: 10.1109/ACCESS.2020.2975643.

[9] S. Pudaruth, U. G. Singh, Institute of Electrical and Electronics Engineers. South African Section, and Institute of Electrical and Electronics Engineers, *2020 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD) : conference proceedings : 6-7 August 2020, Durban, KwaZulu Natal, South Africa*.

[10] M. N. A. Wahab, S. Nefti-Meziani, and A. Atyabi, "A comparative review on mobile robot path planning: Classical or meta-heuristic methods?," *Annu. Rev. Control*, vol. 50, pp. 233–252, 2020, doi: 10.1016/j.arcontrol.2020.10.001.

[11] M. T. Saeed and J. Pan, "An Intelligent Hybrid Control to Enhance Applicability of Mobile Robots in Cluttered Environments," *IEEE Access*, vol. 9, pp. 50151–50162, 2021, doi: 10.1109/ACCESS.2021.3068988.

[12] S. Campbell, N. O'Mahony, A. Carvalho, L. Krpalkova, D. Riordan, and J. Walsh, "Path planning techniques for mobile robots a review," in *2020 6th International Conference on Mechatronics and Robotics Engineering (ICMRE)*, pp. 12-16, 2020.

[13] M. N. Zafar and J. C. Mohanta, "Methodology for Path Planning and Optimization of Mobile Robots: A Review," in *Procedia Computer Science*, vol. 133, pp. 141–152, 2018.

[14] T. T. Mac, C. Copot, D. T. Tran, and R. De Keyser, "Heuristic approaches in robot path planning: A survey," *Robotics and Autonomous Systems*, vol. 86, pp. 13-28, 2016, doi: 10.1016/j.robot.2016.08.001.

[15] B. Wang, Z. Liu, Q. Li, and A. Prorok, "Mobile Robot Path Planning in Dynamic Environments Through Globally Guided Reinforcement Learning," in *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6932-6939, Oct. 2020, doi: 10.1109/LRA.2020.3026638.

[16] R. Abiyev, D. Ibrahim, and B. Erin, "Navigation of mobile robots in the presence of obstacles," *Adv. Eng. Softw.*, vol. 41, no. 10–11, pp. 1179–1186, 2010, doi: 10.1016/j.advengsoft.2010.08.001.

[17] D. B. H. Abid, N. Y. Allagui, and N. Derbel, "Navigation and trajectory tracking of mobile robot based on kinematic PI controller," in *2017 18th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA)*, pp. 252-256, 2017.

[18] Y. Yan and Y. Li, "Mobile robot autonomous path planning based on fuzzy logic and filter smoothing in dynamic environment," in *2016 12th World congress on intelligent control and automation (WCICA)*, pp. 1479-1484, 2016.

[19] M. Saraswathi, G. B. Murali, and B. B. V. L. Deepak, "Optimal Path Planning of Mobile Robot Using Hybrid Cuckoo Search-Bat Algorithm," *Procedia Comput. Sci.*, vol. 133, pp. 510–517, 2018, doi: 10.1016/j.procs.2018.07.064.

[20] S. Choueiry, M. Owayjan, H. Diab, and R. Achkar, "Mobile robot path planning using genetic algorithm in a static environment," in *2019 Fourth International Conference on Advances in Computational Tools for Engineering Applications (ACTEA)*, pp. 1-6, 2019.

[21] S. M. H. Rostami, A. K. Sangaiah, J. Wang, and X. Liu, "Obstacle avoidance of mobile robots using modified artificial potential field algorithm," *Eurasip J. Wirel. Commun. Netw.*, vol. 2019, no. 1, 2019, doi: 10.1186/s13638-019-1396-2.

[22] J. Zhang, "Path planning for a mobile robot in unknown dynamic environments using integrated environment representation and reinforcement learning," *2019 Aust. New Zeal. Control Conf. ANZCC 2019*, pp. 258–263, 2019, doi: 10.1109/ANZCC47194.2019.8945595.

[23] R. K. Mandava, S. Bondada, and P. R. Vundavilli, "An optimized path planning for the mobile robot using potential field method and PSO algorithm," in *Soft Computing for Problem Solving: SocProS 2017*, vol. 2, pp. 139-150, 2019, doi: 10.1007/978-981-13-1595-4.

[24] X. Zhong, J. Tian, H. Hu, and X. Peng, "Hybrid Path Planning Based on Safe A* Algorithm and Adaptive Window Approach for Mobile Robot in Large-Scale Dynamic Environment," *J. Intell. Robot. Syst. Theory Appl.*, vol. 99, no. 1, pp. 65–77, 2020, doi: 10.1007/s10846-019-01112-z.

[25] S. K. Das, K. Roy, T. Pandey, A. Kumar, A. K. Dutta, and S. K. Debnath, "Modified Critical Point - A Bug Algorithm for Path Planning and Obstacle Avoiding of Mobile Robot," *Proc. 2020 IEEE Int. Conf. Commun. Signal Process. ICCSP 2020*, pp. 351–356, 2020, doi: 10.1109/ICCSP48568.2020.9182347.

[26] D. Connell and H. Manh La, "Extended rapidly exploring random tree–based dynamic path planning and replanning for mobile robots," *Int. J. Adv. Robot. Syst.*, vol. 15, no. 3, pp. 1–15, 2018, doi: 10.1177/1729881418773874.

[27] P. Zhang, C. Xiong, W. Li, X. Du, and C. Zhao, "Path planning for mobile robot based on modified rapidly exploring random tree method and neural network," *Int. J. Adv. Robot. Syst.*, vol. 15, no. 3, pp. 1–8, 2018, doi: 10.1177/1729881418784221.

[28] H. Liu, X. Zhang, J. Wen, R. Wang, and X. Chen, "Goal-biased Bidirectional RRT based on Curve-smoothing," *IFAC-PapersOnLine*, vol. 52, no. 24, pp. 255–260, 2019, doi: 10.1016/j.ifacol.2019.12.417.

[29] S. Sedighi, D. Van Nguyen, and K. D. Kuhnert, "Guided Hybrid A-star Path Planning Algorithm for Valet Parking Applications," *2019 5th Int. Conf. Control. Autom. Robot. ICCAR 2019*, pp. 570–575, 2019, doi: 10.1109/ICCAR.2019.8813752.

[30] X. Tang and F. Chen, "Robot Path Planning Algorithm based on Bi-RRT and Potential Field," *2020 IEEE Int. Conf. Mechatronics Autom. ICMA 2020*, pp. 1251–1256, 2020, doi: 10.1109/ICMA49215.2020.9233539.

[31] Z. Wu, Z. Meng, W. Zhao, and Z. Wu, "Fast-RRT: A RRT-based optimal path finding method," *Appl. Sci.*, vol. 11, no. 24, 2021, doi: 10.3390/app112411777.

[32] M. Medvedev, V. Pshikhopov, B. Gurenko, and N. Hamdan, "Path planning method for mobile robot with maneuver restrictions," in *2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, pp. 1-7, 2021, doi: 10.1109/ICECCME52200.2021.9591090.

[33] Z. Zhang, Y. Wan, Y. Wang, X. Guan, W. Ren, and G. Li, "Improved hybrid A* path planning method for spherical mobile robot based on pendulum," *Int. J. Adv. Robot. Syst.*, vol. 18, no. 1, pp. 1–14, 2021, doi: 10.1177/1729881421992958.

[34] W. Sheng, B. Li, and X. Zhong, "Autonomous Parking Trajectory Planning with Tiny Passages: A Combination of Multistage Hybrid A-Star Algorithm and Numerical Optimal Control," *IEEE Access*, vol. 9, pp. 102801–102810, 2021, doi: 10.1109/ACCESS.2021.3098676.

[35] C. Van Dang, H. Ahn, D. S. Lee, and S. C. Lee, "Improved Analytic Expansions in Hybrid A-Star Path Planning for Non-Holonomic Robots," *Appl. Sci.*, vol. 12, no. 12, pp. 1–11, 2022, doi: 10.3390/app12125999.

[36] A. Alouache and Q. Wu, "Genetic algorithms for trajectory tracking of mobile robot based on PID controller," *Proc. - 2018 IEEE 14th Int. Conf. Intell. Comput. Commun. Process. ICCP 2018*, pp. 237–241, 2018, doi: 10.1109/ICCP.2018.8516587.

[37] N. P. Varma, A. Vivek, and V. R. Pandi, "Target tracking, path planning and obstacle avoidance by intelligent robot," in *2017 International Conference on Technological Advancements in Power and Energy (TAP Energy)*, pp. 1-6, 2017.

[38] J. Campos, S. Jaramillo, L. Morales, O. Camacho, D. Chávez, and D. Pozo, "PSO tuning for fuzzy PD+ I controller applied to a mobile robot trajectory control," in *2018 International Conference on Information Systems and Computer Science (INCISCOS)*, pp. 62-68, 2018.

[39] L. A. Yekinni and A. Dan-isa, "Fuzzy Logic Control of Goal-Seeking 2-Wheel Differential Mobile Robot Using Unicycle Approach," *2019 IEEE Int. Conf. Autom. Control Intell. Syst.*, pp. 300–304, 2019.

[40] I. Hassani, I. Maalej, and C. Rekik, "Control points searching algorithm for autonomous mobile robot navigation with obstacle avoidance," *19th Int. Conf. Sci. Tech. Autom. Control Comput. Eng. STA 2019*, no. 1, pp. 158–163, 2019, doi: 10.1109/STA.2019.8717220.

[41] M. Dirik, A. F. Kocamaz, and O. Castillo, "Global Path Planning and Path-Following for Wheeled Mobile Robot Using a Novel Control Structure Based on a Vision Sensor," *Int. J. Fuzzy Syst.*, vol. 22, no. 6, pp. 1880–1891, 2020, doi: 10.1007/s40815-020-00888-9.

[42] S. Wang, X. Yin, P. Li, M. Zhang, and X. Wang, "Trajectory Tracking Control for Mobile Robots Using Reinforcement Learning and PID," *Iran. J. Sci. Technol. - Trans. Electr. Eng.*, vol. 44, no. 3, pp. 1059–1068, 2020, doi: 10.1007/s40998-019-00286-4.

[43] N. H. Thai, T. T. K. Ly, H. Thien, and L. Q. Dzung, "Trajectory Tracking Control for Differential-Drive Mobile Robot by a Variable Parameter PID Controller," *Int. J. Mech. Eng. Robot. Res.*, vol. 11, no. 8, pp. 614–621, 2022, doi: 10.18178/ijmerr.11.8.614-621.

[44] N. Hassan and A. Saleem, "Neural Network-Based Adaptive Controller for Trajectory Tracking of Wheeled Mobile Robots," *IEEE Access*, vol. 10, pp. 13582–13597, 2022, doi: 10.1109/ACCESS.2022.3146970.

[45] A. Stefek, T. van Pham, V. Krivanek, and K. L. Pham, "Energy comparison of controllers used for a differential drive wheeled mobile robot," *IEEE Access*, vol. 8, pp. 170915–170927, 2020, doi: 10.1109/ACCESS.2020.3023345.

[46] N. Leena and K. K. Saju, "Modelling and Trajectory Tracking of Wheeled Mobile Robots," *Procedia Technol.*, vol. 24, pp. 538–545, 2016, doi: 10.1016/j.protcy.2016.05.094.

[47] M. J. Rabbani and A. Y. Memon, "Trajectory Tracking and Stabilization of Nonholonomic Wheeled Mobile Robot Using Recursive Integral Backstepping Control," *electronics*, vol. 10, pp. 1–22, 2021.

[48] O. Sharma, N. C. Sahoo, and N. B. Puhan, "A Survey on Smooth Path Generation Techniques for Nonholonomic Autonomous Vehicle Systems," *IECON Proc. (Industrial Electron. Conf.*, vol. 2019-Octob, pp. 5167–5172, 2019, doi: 10.1109/IECON.2019.8926946.

[49] J. Chen, Y. Zhao, and X. Xu, "Improved RRT-Connect Based Path Planning Algorithm for Mobile Robots," *IEEE Access*, vol. 9, pp. 145988–145999, 2021, doi: 10.1109/ACCESS.2021.3123622.

[50] S. Ramasubramanian and S. A. Muthukumaraswamy, "On the Enhancement of Firefighting Robots using Path-Planning Algorithms," *SN Comput. Sci.*, vol. 2, no. 3, pp. 1–11, 2021, doi: 10.1007/s42979-021-00578-9.

[51] F. Gul and W. Rahiman, "An Integrated approach for Path Planning for Mobile Robot Using Bi-RRT," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 697, no. 1, 2019, doi: 10.1088/1757-899X/697/1/012022.

[52] Y. Gao, T. Hu, Y. Wang, and Y. Zhang, "Research on the Path Planning Algorithm of Mobile Robot," *Proc. - 2021 13th Int. Conf. Meas. Technol. Mechatronics Autom. ICMTMA 2021*, pp. 447–450, 2021, doi: 10.1109/ICMTMA52658.2021.00102.

[53] S. Luu, "songl/Masterthesis: Comparing the motion planning methods Hybrid A* and RRT for autonomous off-road driving of bicycle vehicles," 2021, [Online]. Available: https://github.uio.no/songl/Masterthesis

[54] S. B. Joseph, E. G. Dada, A. Abidemi, D. O. Oyewola, and B. M. Khammas, "Metaheuristic algorithms for PID controller parameters tuning: review, approaches and open problems," *Heliyon*, vol. 8, no. 5, p. e09399, 2022, doi: 10.1016/j.heliyon.2022.e09399.

[55] T. M. Shami, A. A. El-Saleh, M. Alswaitti, Q. Al-Tashi, M. A. Summakieh, and S. Mirjalili, "Particle Swarm Optimization: A Comprehensive Survey," *IEEE Access*, vol. 10, pp. 10031–10061, 2022, doi: 10.1109/ACCESS.2022.3142859.

[56] M. Madhiarasan, D. T. Cotfas, and P. A. Cotfas, "Black Widow Optimization Algorithm Used to Extract the Parameters of Photovoltaic Cells and Panels," *Mathematics*, vol. 11, no. 4, 2023, doi: 10.3390/math11040967.

[57] A. F. Peña-Delgado *et al*., "A novel bio-inspired algorithm applied to selective harmonic elimination in a three-phase eleven-level inverter," *Mathematical Problems in Engineering*, pp. 1–10, 2020.

[58] M. Mohamed and M. Hamza, "Design PID Neural Network Controller for Trajectory Tracking of Differential Drive Mobile Robot Based on PSO," *Eng. Technol. J.*, vol. 37, no. 12A, pp. 574–583, 2019, doi: 10.30684/etj.37.12a.12.

[59] A. Al-Mayyahi, W. Wang, and P. Birch, "Path tracking of autonomous ground vehicle based on fractional order PID controller optimized by PSO," *SAMI 2015 - IEEE 13th Int. Symp. Appl. Mach. Intell. Informatics, Proc.*, pp. 109–114, 2015, doi: 10.1109/SAMI.2015.7061857.

[60] G. A. Ibraheem, "Motion Control of An Autonomous Mobile Robot using Modified Particle Swarm Optimization Based Fractional Order PID Controller," vol. 34, no. 13, pp. 2406–2419, 2016.