Received: June 6, 2022. Revised: June 27, 2022.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Intelligent Hybrid Path Planning Algorithms for Autonomous Mobile Robots

Noor Abdul Khaleq Zghair[1*] **Ahmed Sabah Al-Araji**[1]

[1]*Computer Engineering Department, University of Technology- Iraq, Baghdad, Iraq*
* Corresponding author's Email: Noor.A.Zghair@uotechnology.edu.iq

Abstract: The primary goal of a mobile robot is to reach the desired goal by traversing an optimized path defined according to some criteria such as time, distance, and safety of the robot from any obstacles that may be in its way. Therefore, the backbone of autonomous mobile robots (AMRs) is path planning and avoiding obstacles. Many algorithms for path planning and obstacle avoidance have been presented by many researchers and each of these algorithms has several benefits and drawbacks. This paper focuses on comparing the performance of several metaheuristic algorithms that result in a more efficient, smoother, and shorter path for the mobile robot to reach the target in a complex environment. These algorithms include particle swarm optimization (PSO), chaotic particle swarm optimization (CPSO), modified chaotic particle swarm optimization (MCPSO), and firefly algorithm (FA). On the other hand, the paper proposes a hybrid algorithm by combining the FA and the MCPSO, namely the (HFAMCPSO). To demonstrate the effectiveness of the proposed algorithm in terms of the optimum cost function and obtaining the shortest path length, the optimal solution is compared to those of other path planning algorithms. Moreover, inverse dynamic and kinematic modeling are utilized to obtain the best torque and the best velocity actions for the wheels of the autonomous mobile robot. The proposed hybrid (FAMCPSO) algorithm provides enhancement on the path length equals (43.3%) and (25.5%) compared to the radial cell decomposition (RCD) and the A* algorithm, respectively. Moreover, the enhancement on the path length equals (2.3%) and (22.7%) compared to the firefly algorithm (FA) and the genetic algorithm (GA), respectively. All methods are simulated in an environment with static obstacles using the 2018b MATLAB package.

Keywords: Autonomous mobile robot, Obstacle avoidance, Path planning, Firefly algorithm, Chaotic particle swarm optimization.

1. Introduction

Path planning or path finding problems are well-known in mobile robots, and they play a crucial role in autonomous mobile robot navigation. In this regard, navigation, which is defined as the process or activity of planning and directing a route or path, is a task that an autonomous robot must perform successfully to safely move from one location to another without being lost or colliding with other objects. Hence, path planning is one of the most important techniques for mobile robot autonomy. In particular, path planning's major goal is to create an ideal and viable path for a mobile robot to follow in order to get to the target places as rapidly as possible, utilizing optimal trajectories [1-3].

Recently, many algorithms have been developed to tackle the problems of mobile robot path planning [4] including A* algorithms [5], D* algorithms [6], fuzzy logic (FL) [7], genetic algorithms (GAs) [8], particle swarm optimization (PSO) [9], ant colony algorithms (ACOs) [10], artificial potential fields [11], probabilistic road map (PRM), bug algorithms, and others [4]. Each of these algorithms has advantages and limitations in different environments. In particular, these algorithms attempt to find the shortest path length and the total time consumed while avoiding any collisions with obstacles [2, 12]. For instance, the authors in [13] suggested designing the smooth path of mobile robots by developing a new technique that integrates parametric cubic Bezier curve (PCBC) and particle swarm optimization with adaptive delayed velocity (PSO-ADV) algorithms. This path can achieve an equal curvature at the segment joints, allowing it to establish a continuous curvature across the smooth path. However, the drawbacks are that the degree of the Bezier curve depends on the number of control points. In addition, the Bezier curve lacks local control and changing the position of one control point affects the entire curve. In [14], a route planning method and a control approach were suggested for the mobile robot system using a hybrid swarm optimization algorithm and the convolutional neural network trajectory tracking (CNNTT) for controller design. This hybrid algorithm (the chaotic particle swarm optimization (CPSO) algorithm and the A* algorithm) determined the shortest route with the best cost function. Nevertheless, the problem was not solved, the A* algorithm has simpler calculations and can solve problems quickly, but it is unable to recalculate if a problem occurs along the path.

To ensure the best path and to enhance the final path, the researchers in [15] created an intelligent hybrid optimization approach called quarter orbits particle swarm optimization (QOPSO). This approach is a combination of two algorithms: the quarter orbits (QO) algorithm and the particle swarm optimization (PSO) algorithm. The quarter orbits algorithm, on the other hand, can discover a collision-free path, but there are no assurances that it will find the best path since it moves the mobile robot from one orbit to another, which consumes more power and results in an unsmooth path.

For the cluttered and the corridor environments, the authors in [16] proposed a new approach to cell decomposition, named the radial cell decomposition (RCD) algorithm, which may create shorter pathways with a slightly faster processing time compared to the vertical cell decomposition (VCD) algorithm. Based on a collection of arches drawn from the center point, the RCD algorithm splits the environment into a set of free cells. However, the drawback of these algorithms is that they use a grid or vertical-line cells with a long distance between cells based on the obstacles in the environment.

The authors demonstrated how to find the shortest feasible (collision-free) path using an approach based on the firefly algorithm (FA) [17, 18]. They also compared their approach to two well-known swarm optimization algorithms: the genetic algorithm (GA) and the particle swarm algorithm with inertia weight (PSO-w). Because the PSO-w and the GA are likely to be locked in local optima, the test demonstrates that the FA outperforms the PSO-w and the GA in terms of success rate within the acceptable length. On the other hand, the increment rate of the FA and the GA is so low, that the average length of the optimum path found using the PSO-w is less than that found using the FA and the GA. While in another work [19], the authors proposed a hybrid firefly algorithm (HFA) by combining both the FA and the differential evolution (DE). The hybridization effectively increased the diversity of solutions and helped to avoid the stagnation problem. A modified version of the FA in a 3D sphere environment was studied in [20]. In [21], the defects of the traditional GA, such as the slow convergence speed and the tendency to fall into local optimum, were improved using a multi-objective genetic algorithm. This algorithm improved the initial path and generated a multi-objective fitness function based on three indicators, including path length, path security, and path energy consumption, to further ensure the quality of the planned path. The authors of [11, 22] created a hybridized algorithm that includes a fast marching method hybridized with regression search (FMMHRS) methodology and a hybrid method combining the particle swarm optimization (PSO) algorithm with the potential field method (APF) in both static and dynamic obstacles' environments.

These algorithms can generate collision-free pathways from the beginning to the end. However, the disadvantage of these methods is that the optimal path provided by these algorithms is still somewhat long. The fast marching method (FMM) and FMMHRS algorithms, on the other hand, attempt to create a straight line between intermediate locations and the destination point before attempting to establish the position and grant specific permissions around obstacles, resulting in collision-free robot navigation. The APF approach, which used the attractive potential field function to select the optimal path, and the PSO algorithm were utilized to optimize the created path to overcome the limitation of becoming trapped at local minima.

Therefore, in this work, we propose a solution to solve the problem of mobile robot path planning by proposing a hybrid algorithm, namely the HFAMCPSO, and comparing the results with those of the original intelligent algorithms (the CPSO and the FA) and those of other researchers who use different path planning algorithms in a static environment. The hybrid algorithms generate a shorter path and an improved distance cost function that may be used in a static environment.

The rest of this paper is organized as follows: section (2) describes the wheeled mobile robot model. Section (3) explains the original intelligent algorithms and the suggested hybrid path planning algorithm. While section (4) demonstrates the numerical results and analysis of the MATLAB simulation in a static environment, and finally the conclusions of the paper are discussed in section (5).

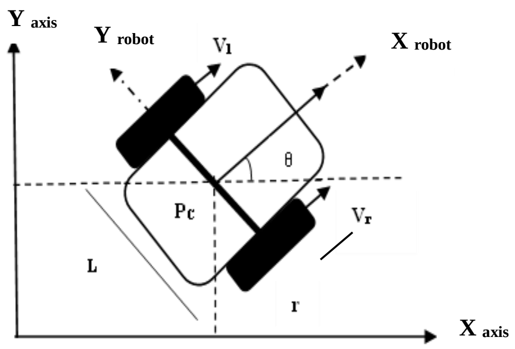


Figure. 1 Nonholonomic mobile robot model [23]

2. Wheeled mobile robot modelling

In general, the wheeled mobile robot platform is shown in Fig. 1. This platform consists of left and right wheels that are placed on a parabolic shaft, and two multi-directional wheels installed in the front or back for stabilization. The nonlinear dynamic response of the robot motion and the steering are determined by the two independent actuators of analog direct current (DC) motors that provide suitable torques to the right and the left wheels of the mobile robot [23].

The global reference frame is [X_{axis}, Y_{axis}], while the position vector of the local reference frame of the mobile robot is defined as given in Eq. (1):

$$Q = [x, y, \theta]^T \tag{1}$$

where (x, y) specifies the position coordinates at the midpoint p_c that denotes the place where the right and the left wheels meet in the middle, and it is the center mass of the wheeled mobile robot. r denotes the radius of the left and the right wheels, while θ acts as the orientation of the motion based on pure rolling and non-slipping non-holonomic constraints for the mobile robot, as given in Eq. (2) [23]:

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

where $\dot{x}$ is the velocity in the X_{axis} and $\dot{y}$ is the velocity in the Y_{axis}. As a result, the three kinematic equations for nonholonomic wheels mobile robots can be represented as in Eqs. (3), (4), and (5) [24, 25]:

$$x(t) = 0.5 \times [v_r(t) + v_l(t)] \times \cos\theta(t) \times T_S + x(t-1) \tag{3}$$

$$y(t) = 0.5 \times [v_r(t) + v_l(t)] \times \sin\theta(t) \times T_S + y(t-1) \tag{4}$$

$$\theta(t) = \frac{1}{L} \times [v_l(t) - v_r(t)] \times T_S + \theta(t-1) \tag{5}$$

where $v_r(t)$ and $v_l(t)$ are the right and the left wheels' velocities of the platform, respectively. The distance between the driving wheels of the platform is taken as L and the sampling time of the numerical calculation is denoted by Ts.

3. Path planning intelligent algorithms

If we want to move a mobile robot to reach the desired destination, the first challenge that will encounter the work is finding the ideal or the closed-to-optimal desired way by avoiding the obstacles to reach the destination with acceptable accuracy. Hence, it is critical to have the ability to avoid obstacles.

The robot must be reliable to accomplish its job without risking itself or others, with the requirement of keeping the path as short as possible. Thus, we will need the most basic environmental data, as well as modeling the building's layout to determine the position of the goal point. In addition, we will need to determine the minimum distance from the starting position to the goal position using the Euclidean plane that can be achieved by the following distance cost function:

$$Dist_{fun}\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \tag{6}$$

where $Dist_{\text{fun}}$ is the distance between two points, x_i and y_i are x and y coordinates of the current waypoints, and x_{i-1} and y_{i-1} are x and y coordinates of feasible waypoints in the next iteration.
Several algorithms have been presented in the past few years to devise an optimal path and avoid collisions with obstacles [4, 14]. In the section below, we explain several intelligent algorithms, including the modified chaotic particle swarm optimization (MCPSO) algorithm, the firefly algorithm (FA), and the proposed hybrid algorithm.

3.1 Modified chaotic particle swarm optimization algorithm (MCPSO)

In general, the PSO is a calculation technology like other evolutionary algorithms. It is based on a multi-point research technology that mimics the social behavior of animals through individual interaction and competition.

In this algorithm, the search is initialized with a population of candidate solutions that are called particles. These particles have a memory and can save a portion of their prior state. The particle's mobility is governed by two randomly weighted factors: individuality and sociality.

The definition of individuality is "the tendency to return to the particle's best past situation", while sociality is defined as "the tendency to move towards the neighborhood's best previous situation".

In the PSO algorithm, each particle has its own position (which represents a point in the search space) and velocity (which represents the ratio of position change) in successive iterations [14, 26, 27].

The PSO algorithm has some drawbacks such as the tendency to fall into local extreme values and they cannot obtain the global best solution [25, 27]. To overcome the limitations of the PSO, several research works have been conducted. One of these research works considered the chaotic searching by changing the inertia weight with the iterative process, which effectively improves the global search ability of the algorithm, and it is included into the PSO to induce more randomness in the search, increase global searching capability, and prevent a slide into the premature convergence to local minima. The algorithm combined the PSO and the chaotic map technique and it was called the chaotic particle swarm optimization (CPSO) algorithm, which effectively blends the chaotic searching behavior with population based evolutionary searching abilities [14, 25, 26, 28]. Furthermore, the parameter *w* is the inertia weight factor, which is utilized to improve the search stability. The velocities of the particles are decreased using *w* to allow them to converge more accurately and efficiently.

A greater value of *w* is utilized to promote global swarm exploration, whereas a smaller value of *w* is recommended to promote local exploration.
A typical linearly decreasing inertia weight technique may be employed to create a balance between local and global exploration. The chaotic model is adopted in Eqs. (7), (8) and (9).

$$Z^{\text{iter}+1} = \mu \times Z^{\text{iter}}(1 - Z^{o}) \qquad (7)$$

where μ is the control parameter and when μ= 4 the system enters into a chaotic state [15].

$$W = W_{MX} - \frac{(W_{MX} - W_{MN})}{T_{\max}} \times \text{iter} \qquad (8)$$

$$W_{\text{new}} = W \times Z^{\text{iter}+1} \qquad (9)$$

where Z_0 is the initial value of deterministic; W_{MN} and W_{MX} are the minimum and the maximum inertia weights, respectively, $T_{\max}$ is the maximum iterations' number, and iter is the present iteration.
On the other hand, in the PSO algorithm, the acceleration coefficient (C) is an essential parameter. Whether or not it has an appropriate value is strongly tied to the algorithm's optimal optimization [28]. The parameters C_1 and C_2 are the learning factors, which represent the weight of each particle to the statistical acceleration in the item of the extreme position. Based on this fact, in this article, we proposed to improve the optimal performance of the PSO algorithm with the chaotic searching by dynamically changing the acceleration coefficient with the iterative process and we obtained better optimal performance compared with the chaotic state in the inertia weight after applying the chaotic equations adopted in this article, as shown in Eqs. (10), (11), and (12):

$$C = C_{MX} - \frac{(C_{MX} - C_{MN})}{T_{\max}} \times \text{iter} \qquad (10)$$

$$C_{1\text{new}} = C \times Z^{\text{iter}+1} \qquad (11)$$

$$C_{2\text{new}} = C_{1\text{new}} \qquad (12)$$

where C_{MN} and C_{MX} are the minimum and the maximum acceleration values, respectively.
The new update equations for velocity and position are described in Eqs. (13) and (14) below:

$$[V(i,j)]_p^{\text{iter}+1} = \begin{bmatrix} W \times V(i,j) + C_{1\text{new}} \times r_1 \\ \times \left(P_{\text{best}}(i,j) - xy(i,j)\right) \\ +C_{2\text{new}} \times r_2 \times (G_{\text{best}}(i,j) - xy(i,j)) \end{bmatrix}_p^{\text{iter}} \qquad (13)$$

$$[xy(i,j)]_p^{\text{iter}+1} = [xy(i,j)]_p^{\text{iter}} + [V(i,j)]_p^{\text{iter}+1} \qquad (14)$$

Where P_{best} is the best fitness values for particle pth; G_{best} is the best fitness values for the whole swarm; the proposed values of r_1 and r_2 are 0.9, $[xy(i,j)]_p^{\text{iter}}$ and $[V(i,j)]_p^{\text{iter}}$ of the p particle represent the position and velocity at the iterth iteration, respectively, and (i,j) denotes the coordinates' values in X_{axis} and Y_{axis} [14, 15].

Fig. 2 represents the pseudo code of the proposed MCPSO algorithm.

3.2 Firefly algorithm (FA)

Nature-inspired optimization algorithms have recently gained popularity. The firefly algorithm (FA) is a kind of stochastic, nature-inspired, metaheuristic algorithm that tries to simulate the attraction behavior of fireflies and lighting patterns [29]. Because of qualities such as high error tolerance, automated segmentation of the population into subgroups, and non-sensitivity to first values, this

```
Step 1: The maximum number of iterations $T_{max}$.
Step 2: Initialize particle p.
Step3: For each particle p, calculate the fitness function based on Eq. (6), and check if the fitness value is better than the previous better fitness value $P_{best}$, then set the current value as the new $P_{best}$
Step 4: for each p:
    - Find p with the best fitness in particle neighbourhood $G_{best}$.
    - Apply the modified CPSO algorithm using Eqs. (7, 9, 10, 11 and 12).
    - Depending on the velocity of Eq. (13), calculate particle velocity V(i,j).
    - Depending on the position of Eq. (14), update the particle position xy(i,j).
    - Apply the new position.
Step 5: Repeat Step 3 until reaching $T_{max}$, and find the best global path for the mobile robot.
```

Figure. 2 The pseudo code of the proposed MCPSO algorithm

```
Step 1: Initialize the algorithm parameters.
Step 2: Generate an initial population of the fireflies.
Step3: For each firefly, calculate the fitness function value. Eq. (6) is the respective maximum fluorescence fireflies' brightness.
Step 4: Calculate the distance ŕ between $x_i$ and $x_j$ using Eq. (17).
Step 5: Calculate the attractive β that varies with distance ŕ by Eq. (16).
Step 6: Evaluate new solutions and update light intensity via Eq. (18).
Step 7: Repeat Step 3 until reaching $T_{max.}$ and find the best global path for the mobile robot.
```

Figure. 3 The pseudocode of the FA algorithm

method is extensively used for solving optimization and engineering problems, and it can produce good results [30]. For simplicity, this algorithm is based on the following three characteristics: Fireflies are unisex so that any individual will be attracted to others regardless of their gender; their attractiveness is proportionate to their brightness, so when there are two lighting fireflies, the less brilliant one will migrate towards the brighter one; and the brightness of a firefly is associated or determined by the objective function [31, 32]. The standard FA consists of two important characteristics; the first one is the formulation of light intensity change $\hat{I}$ that will be computed using Eq. (15), while the second one is the attractiveness β, which is calculated using Eq. (16) [31, 33].

$$\hat{I} = \hat{I}_o \times e^{-\gamma \acute{r}^2} \tag{15}$$

$$\beta = \beta_o \times e^{-\gamma \acute{r}^2} \tag{16}$$

where β_o is the attractiveness at ŕ = 0, $\hat{I}_o$ denotes the original light intensity, γ is the fixed light absorption coefficient, and ŕ is the distance between two fireflies. The light intensity varies with the distance ŕ between two fireflies. The Cartesian distance between any two fireflies at (x_i, y_i) and (x_j, y_j), respectively, is expressed as shown in Eq. (17) [17, 18]:

$$\acute{r}(i,j) = \sqrt{(x_i - x_j)^2 - (y_i - y_j)^2} \tag{17}$$

Thus, the movement of a firefly is attracted to another brighter one, and the new position of the firefly is given by the following update formula shown in Eq. (18) [20, 30, 31, 34]:

$$xy_{i+1}^{iter} = xy_i^{iter} + \beta\left(xy_i^{iter} - xy_j^{iter}\right) + \alpha \times £ \tag{18}$$

where α represents the randomization parameter [0 to 1], £ represents a vector of random variables (rand – 0.5), and xy_i^{iter} is the ith coordinate element of a firefly in the iterth iteration. In addition, the first part of Eq. (18) represents the firefly's current position, and the second portion represents attraction. Fig. 3 represents the pseudo code of the FA algorithm.

3.3 The proposed hybrid optimization algorithm

By merging the strengths of the two algorithms, the hybridization between the firefly and the modified chaotic particle swarm optimization (HFAMPSO) was proposed. Due to the velocity parameter's fast convergence, the CPSO method is quite successful and conducts fast searching. Due to oscillations in local searches, the algorithm occasionally fails to get the best results. The FA, on the other hand, lacks the ability to maintain a personal best position and lacks a velocity characteristic. As a result, regardless of their past best placements, they will move.

Consequently, fireflies can choose the most appropriate solution based on the local search space conditions. On the other hand, to improve the firefly algorithm's convergence and prevent it from falling into the local minimum, MCPSO features are merged with the FA algorithm to create a hybrid optimization method called the (HFAMCPSO). Based on this, the suggested hybrid combination between the FA and the PSO will find a better solution for firefly's local search capabilities and the PSO algorithm's global search capabilities [30, 32].

The ideas of personal best and global best are introduced into the FA in the proposed algorithm. All the procedures in the FA remain the same, except for the firefly movement, which has been modified to include the concepts of personal and global best [35]. As a result, the FA algorithm's adjusted position vector can be represented as shown in Eqs. (19) and (20):

$$\mathrm{D_{pxy}} = \sqrt{\left(P_{best(i,iter)} - xy_{(i,iter)}\ \right)^2} \qquad (19)$$

$$\mathrm{D_{gxy}} = \sqrt{\left(\mathrm{G_{best(i,iter)}} - \mathrm{xy_{(i,iter)}}\ \right)^2} \qquad (20)$$

where $\mathrm{D_{pxy}}$ is the distance between the best local fitness values for the ith particle's position in the iterth iteration, and $\mathrm{D_{gxy}}$ is the distance between the best global fitness values for all particles and the ith particle's position in the iterth iteration.

The new position's $x_i^{(iter+1)}$ and $y_i^{(iter+1)}$ values of the particles are calculated according to Eq. (21) denoting coordinates' values in x and y axes, respectively.

$$\mathrm{xy_i}^{\ iter+1} = \mathrm{W} \times \mathrm{xy_i}^{\ iter} + \mathrm{C_{1new}} \times e^{-D^2_{pxy}} (\mathrm{P_{best,i}} - \mathrm{xy_i}^{\ iter}) + \mathrm{C_{2new}} \times e^{-D^2_{gxy}} (\mathrm{G_{best,i}} - \mathrm{xy_i}^{iter)} + \alpha \times £ \qquad (21)$$

The flowchart of the proposed hybrid algorithm is illustrated in Fig. 4. While Table 1 shows all the parameters that will be used in the simulation results.

4. Numerical results and analysis

Path planning is an essential technology for tackling the autonomous navigation of mobile robots, and the main goal is to plan a collision-free optimal path from the present position to the destination. Therefore, the efficiency of the suggested hybrid algorithms was investigated in this paper using the MATLAB software (2018b) in an environment with fixed obstacles with a limited size of a map forming [600×800] cm for both x and y directions, as shown in Fig. 5.

In Fig. 5, the black grids represent obstacles and the white grids represent the area where the robot can move. A route planning algorithm is responsible for finding this collision-free path, and several

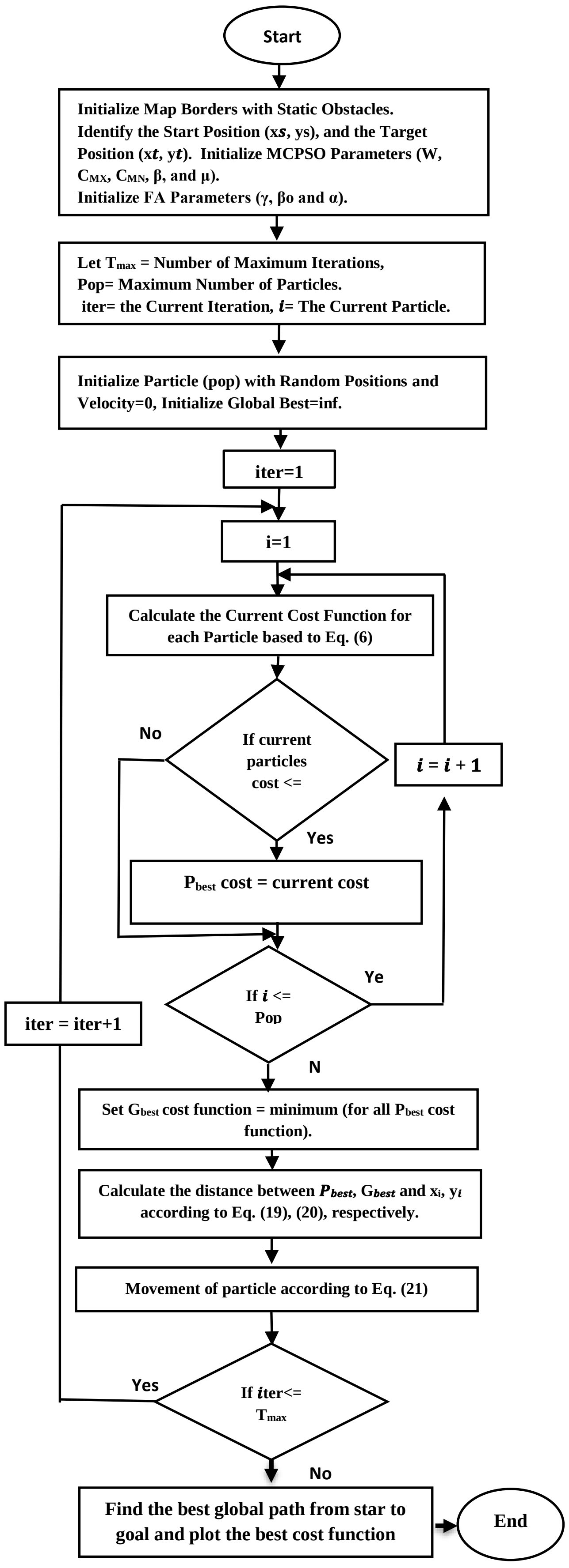

Start
Initialize Map Borders with Static Obstacles.
Identify the Start Position (xs, ys), and the Target Position (xt, yt). Initialize MCPSO Parameters (W, CMX, CMN, β, and μ).
Initialize FA Parameters (γ, βo and α).
Let Tmax = Number of Maximum Iterations,
Pop= Maximum Number of Particles.
iter= the Current Iteration, i= The Current Particle.
Initialize Particle (pop) with Random Positions and Velocity=0, Initialize Global Best=inf.
iter=1
i=1
Calculate the Current Cost Function for each Particle based to Eq. (6)
No
If current particles cost <=
i = i + 1
Yes
Pbest cost = current cost
Ye
If i <= Pop
iter = iter+1
N
Set Gbest cost function = minimum (for all Pbest cost function).
Calculate the distance between Pbest, Gbest and xi, yi according to Eq. (19), (20), respectively.
Movement of particle according to Eq. (21)
Yes
If iter<= Tmax
No
Find the best global path from star to goal and plot the best cost function
End

Figure. 4 The flowchart of the HFAMCPSO algorithm

Table 1. The definitions of all parameters

Parameter	Definition
v_r and v_l	The right and the left wheels' velocities
L	The distance between driving wheels
T_s	The sampling time
Z_0	The initial value of deterministic
μ	The control parameter
W_{MN}	Minimum inertia weight
W_{MX}	Maximum inertia weight
T_{max}	The maximum iterations' number
iter	The present iteration.
C_{MN}	Minimum acceleration values
C_{MX}	Maximum acceleration values
P_{best}	Best fitness values for particle pth
G_{best}	Best fitness values for the whole swarm
r_1 and r_2	Random numbers with a uniform distribution of the range [0, 1].
β_o	The attractiveness at ŕ = 0
I_o	The original light intensity
γ	Fixed light absorption coefficient
ŕ	The distance between two fireflies
α	A randomization parameter [0 to 1]
£	A vector of random variables (rand – 0.5)

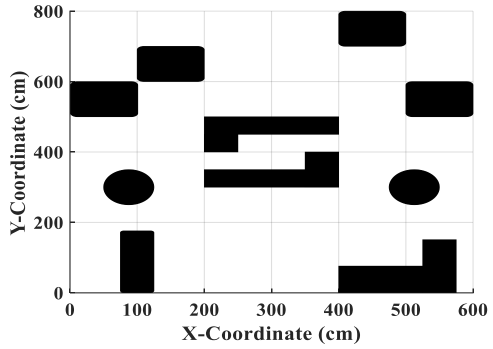


Figure. 5 The proposed environment with static obstacles

techniques including (PSO, CPSO, MCPSO, FA, and the hybrid algorithms) are utilized and compared to find the shortest distance, taking into consideration that a safe distance must be maintained between the robot and the obstacles. The acquisition of the robot's current location, destination, and obstacle positions is the first step in the program coding. In addition, all the cases were executed several times with various iterations' numbers ranging from 50 to 100 and particles' number ranging from 25 to 50. The computer hardware specifications include Intel Core i7-10750H with 16.0 GB of RAM, and CPU of 2.60GHz. The proposed values of the parameters in each algorithm are presented in Table 2.

Table 2. The best values of the parameters of the MCPSO and the FA algorithms

Type of Algorithm	Parameter	Value
MCPSO	Z_0	0.3
	μ	4
	W_{MN}	0.4
	W_{MX}	0.7
	C_{MN}	1
	C_{MX}	2
FA	β_o	2
	γ	1
	α	0.5

Case A

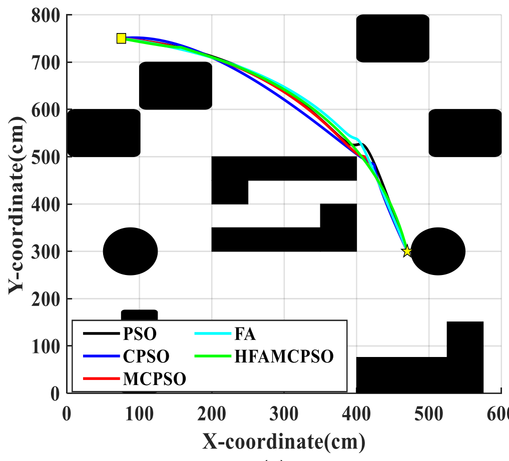
In this case, the initial position of the mobile robot is [75, 750] cm (yellow square), while the target position is [470, 300] cm (yellow star). When applying the algorithms several times with a maximum iterations number equals 100 iterations, the best distance based on the hybrid FAMCPSO algorithm (green path) is 737.399 cm at iteration 50, while the best distance when applying the MCPSO algorithm (red path) is 738.507 cm at iteration 58. The minimum distance based on the CPSO algorithm (blue path) is 739.168 cm at iteration 60, the minimum distance based on PSO algorithm (black path) is 750.834 cm at iteration 73, and finally, the minimum distance based on the FA algorithm (cyan path) is 743.555 cm at iteration 65. All the best cost functions and the best paths of the different algorithms are shown in Fig. 6 (a) and (b).

As a result, when compared to the original algorithms, the path generated by the proposed hybrid algorithm was smooth, and it was the shortest path from the starting point to the goal. This result was achieved based on merging the strengths of the two algorithms; the faster convergence ability of the PSO algorithm, while in the FA, the most appropriate solution is chosen based on the local search space conditions. All the results for the comparison of case A are summarized in Table 3.

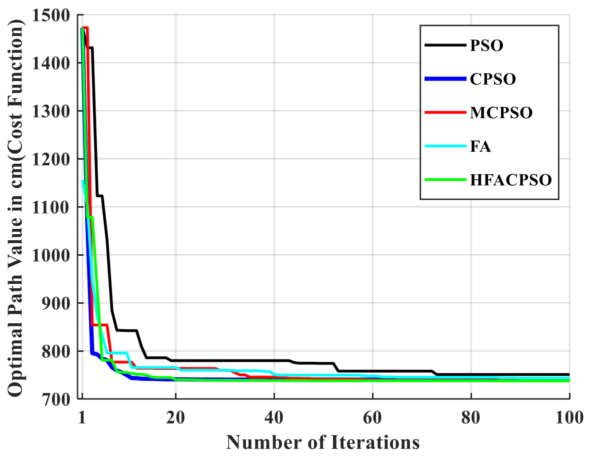
The equation of the reference path for the optimal path of the hybrid FAMCPSO is represented in Eq. (22):

$$y_r\,(x_r) = -4.8791 \times 10^{-13}\, x_r^6 + 5.6773 \times 10^{-10}\, x_r^5 - 2.4954 \times 10^{-7}\, x_r^4 + 4.886 \times 10^{-5}\, x_r^3 - 0.0053179 x_r^2 + 0.14714 x_r + 753.15 \quad (22)$$

Received: June 6, 2022. Revised: June 27, 2022.



(a)



(b)

Figure. 6 The result for case A: (a) the shortest path and (b) the best distance cost function

Table 3. Comparison of the shortest paths

Algorithm Types	Best Path Length	Best No. of Iterations
PSO	750.834	73
CPSO	739.168	60
MCPSO	738.507	58
FA	743.555	65
HFAMCPSO	737.399	50

To generate the best torque actions of the right and the left wheels, and to follow the reference path equations, the inverse dynamic model of the nonlinear wheeled mobile robot is used based on Eqs. (23) to (30) [15].

$$V_{Lin}(t) = \sqrt{(\dot{x}_r(t))^2 + (\dot{y}_r(t))^2} \tag{23}$$

$$W_{ang}(t) = \frac{\ddot{y}_r(t) \times \dot{x}_r(t) - \ddot{x}_r(t) \times \dot{y}_r(t)}{{x_r}^2 + {y_r}^2} \tag{24}$$

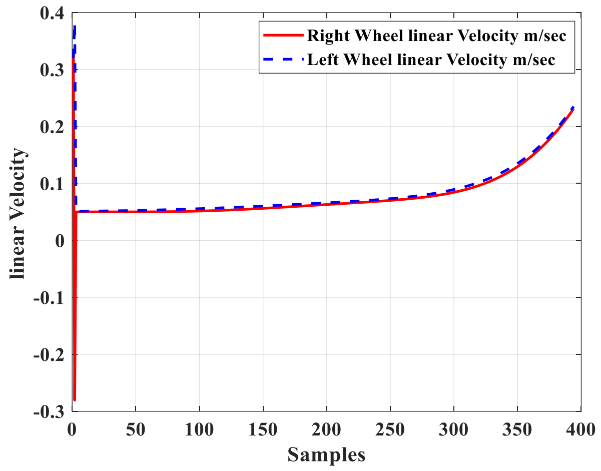


Figure. 7 The linear velocities of the right and the left wheels

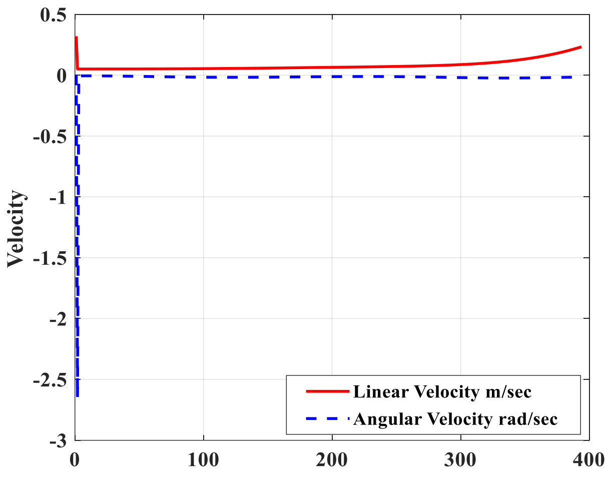


Figure. 8 The linear and the angular velocities of the platform

$$Acc_{Lin}(t) = \frac{V_{Lin}(t) - V_{Lin}(t-1)}{T_s} \tag{25}$$

$$Acc_{ang}(t) = \frac{W_{ang}(t) - W_{ang}(t-1)}{T_s} \tag{26}$$

$$F(t) = Acc_{Lin}(t) \times M \tag{27}$$

$$\tau_a(t) = Acc_{ang}(t) \times I \tag{28}$$

$$\tau_R(t) = \frac{r \times F(t) + (2 \times \frac{r}{L}) \times \tau_a(t)}{2} \tag{29}$$

$$\tau_L(t) = \frac{r \times F(t) - (2 \times \frac{r}{L}) \times \tau_a(t)}{2} \tag{30}$$

where $\tau_L(t)$ is the torque on the left wheel, $\tau_R(t)$ is the torque on the right wheel, M=1.130 kg is the mobile robot mass, I=0.36 kg. m^2 is the inertia of the

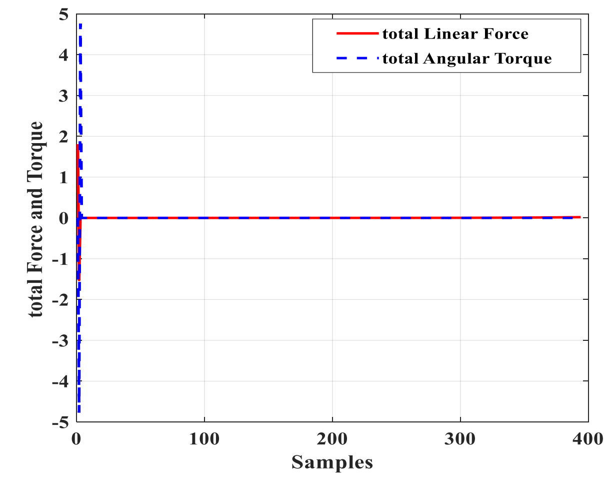


Figure. 9 The linear force and the angular torque of the platform

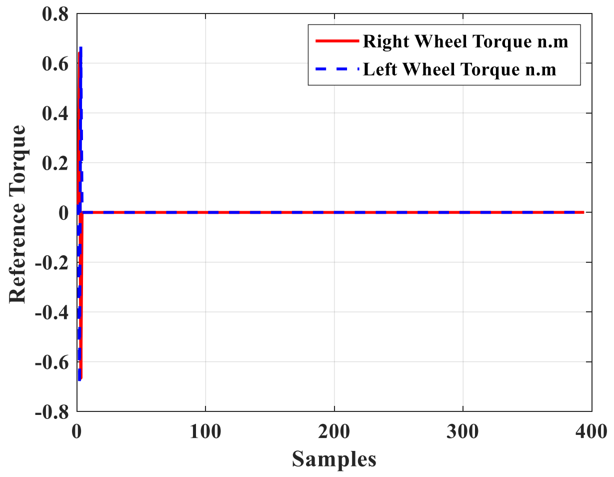
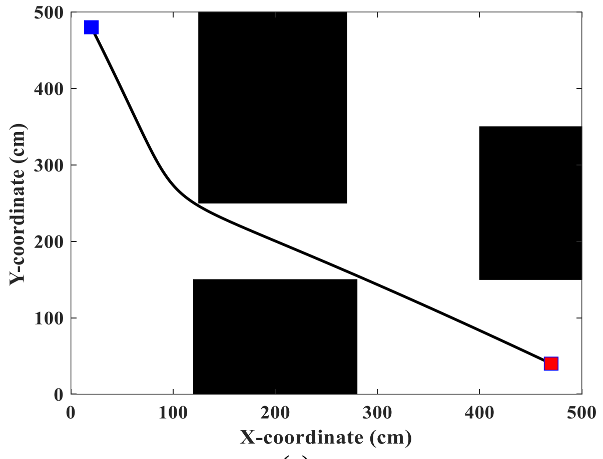


Figure. 10 The right and the left wheels torque control actions

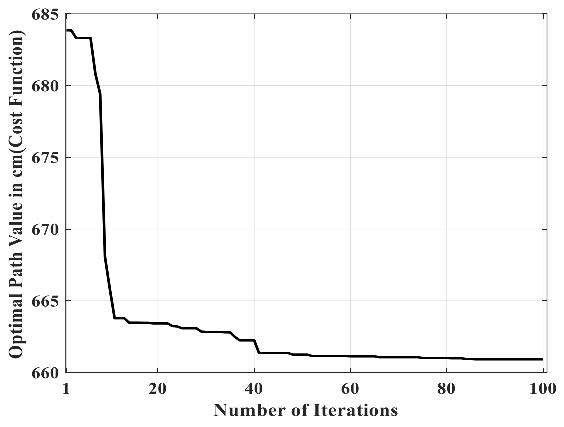
mobile robot, $F(\mathrm{t})$ denotes the linear total force, $\tau_a(\mathrm{t})$ represents the total angular torque, $\mathrm{Acc}_{\mathrm{Lin}}(\mathrm{t})$ is the linear acceleration, $\mathrm{Acc}_{\mathrm{ang}}(\mathrm{t})$ is the angular acceleration, $\mathrm{V}_{\mathrm{Lin}}$ is the linear velocity, $\mathrm{W}_{\mathrm{ang}}$ is the angular velocity, and T_s is the sampling time.

The parameters of the mobile robot platform are used with the following values: r = 0.035m and L= 0.25m with a sampling time equals 0.2 sec.

Based on Eqs. (23) to (30), Fig. 7 represents the smooth velocity of the right and the left wheels of the mobile robot, while Fig. 8 represents the linear and the angular velocities of the platform to follow the desired path. Fig. 9 shows the linear force and the angular torque of the mobile robot platform. Finally, Fig. 10 represents the right and the left wheels torque actions that were generated from the inverse dynamic model.



(a)



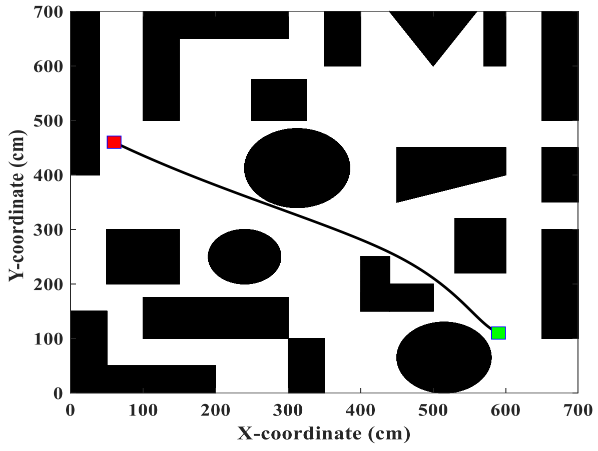
(b)

Figure. 11 Results of the hybrid FAMCPSO based on map [11]: (a) the shortest path and (b) the best distance cost function

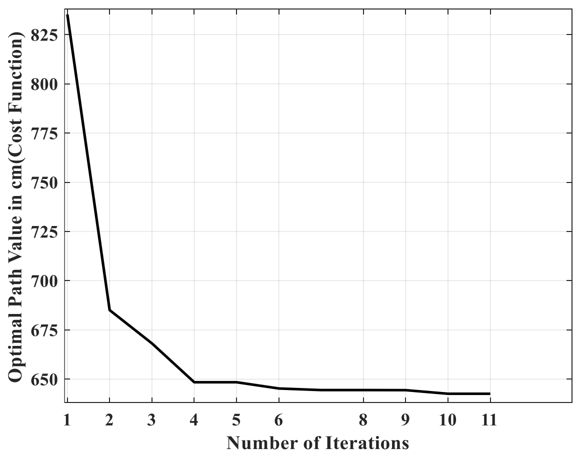
A comparative study was conducted with a previous research work that used different path planning algorithms with varied static environments (simple, cluttered and corridors) containing diverse obstacle forms to verify that the proposed hybrid method, namely the (FAMCPSO), delivers the shortest way. Firstly, the hybrid FAMCPSO was tested using a simple environment with a workspace of [500×500] cm, that was used in [11, 15, and 22] using the artificial potential field (APF) method combined with the particle swarm optimization (PSO) with a three-point smoothing method, the hybrid quarter orbits particle swarm optimization (QOPSO) algorithm, the fast-marching method (FMM), and the fast marching method hybridized with the regression search (FMMHRS) methodology. When the same environment was used with the hybrid FAMCPSO algorithm, the results of the simulation process produce a path length of 660.918 cm, as shown in Fig. 11 (a) and (b). The simulation results indicate that the hybrid FAMCPSO algorithm

Table 4. Comparison of the shortest path with the literature [11, 15, 22]

Algorithm Types	Best Path Length
APF+PSO [11]	819. 87 cm
APF+ PSO+3-point [11]	753.26 cm
The hybrid QOPSO algorithm [15]	663.856 cm
FMM [22]	676.08 cm
FMMHRS [22]	664.26 cm
The Proposed hybrid FAMCPSO	660.918 cm



(a)

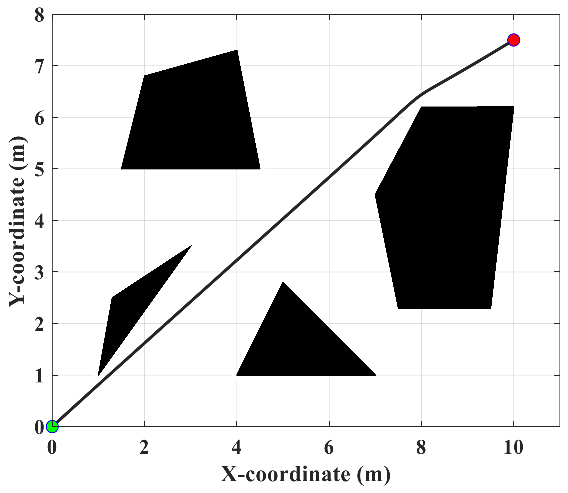


(b)

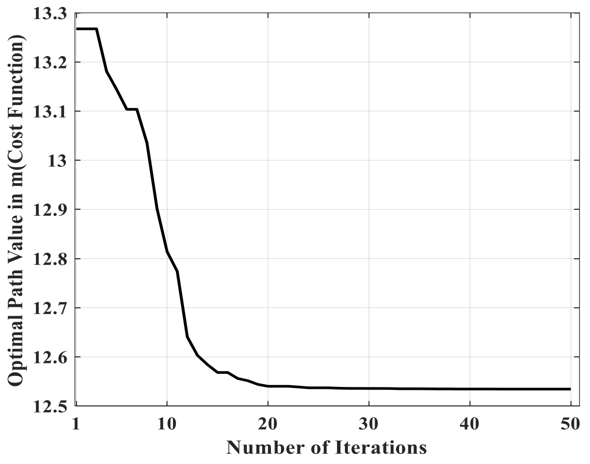
Figure. 12 Results of the hybrid FAMCPSO based on map [15]: (a) the shortest path and (b) the best distance cost function

Table 5. Comparison of the shortest path with the literature [15]

Algorithm Types	Best Path Length	No. of Iterations
QO [15]	659.420 cm	-
PSO [15]	757.1048 cm	20
hybrid QOPSO [15]	657.1271 cm	12
The Proposed hybrid	642.5958 cm	11



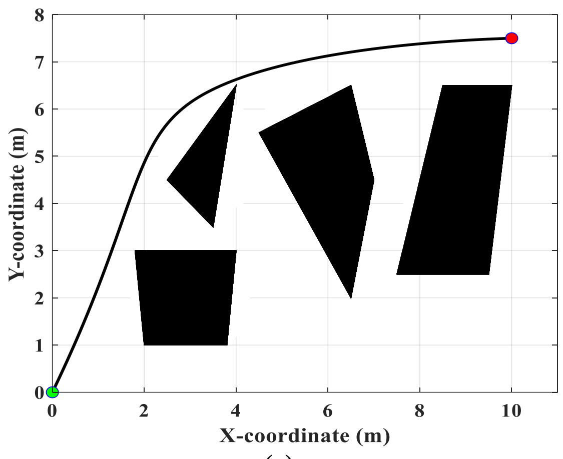
(a)



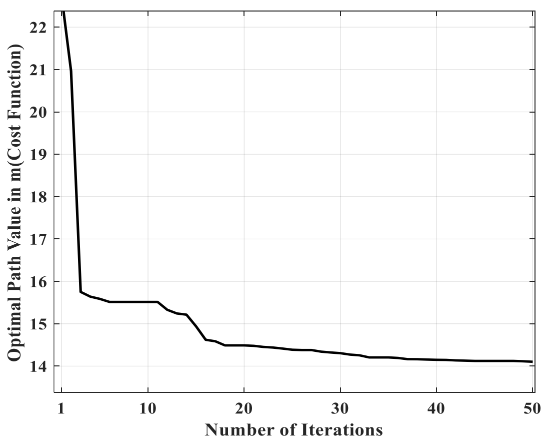
(b)

Figure. 13 Results of the hybrid FAMCPSO based on map [16]: (a) the shortest path and (b) the best distance cost function

can successfully generate the shortest path compared with the APF approach [11] that used the attractive potential field function to select the optimal path. The PSO algorithm was utilized to optimize the created path to overcome the limitation of becoming trapped at local minima. Therefore, the best path generated by these algorithms is still a long one. Moreover, it generates the shortest path compared to the QOPSO algorithm [15]. The QO [15] algorithm can discover a collision-free path, but there are no assurances that it will find the best path since it moves the mobile robot from one orbit to another, which consumes more power and results in an unsmooth path. The FMM and FMMHRS algorithms [22] attempt to create a straight line between intermediate locations and the destination point before attempting to establish the position and grant specific permissions around obstacles, resulting in collision-free robot navigation, but the optimal path provided by these

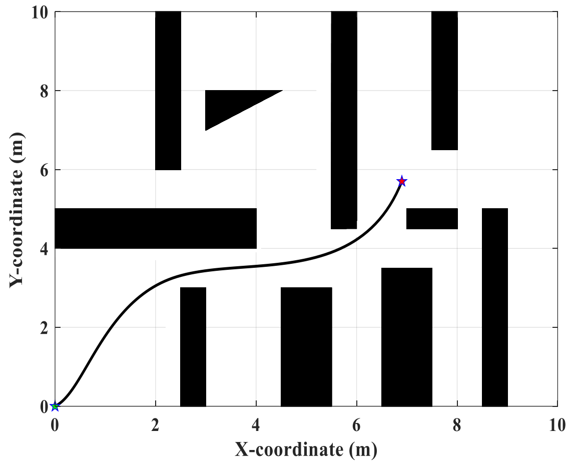


(a)

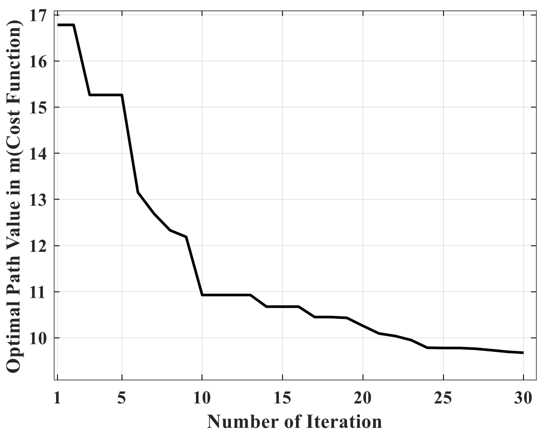


(b)

Figure. 14 Results of the hybrid FAMCPSO based on map [16]: (a) the shortest path and (b) the best distance cost function



(a)



(b)

Figure. 15 Results of the hybrid FAMCPSO based on map [16]: (a) the shortest path and (b) the best distance cost function

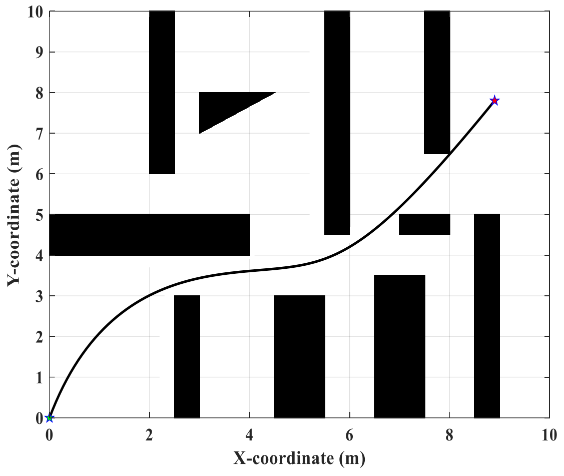
algorithms is still somewhat long.

The results of the comparison study are demonstrated in Table 4.

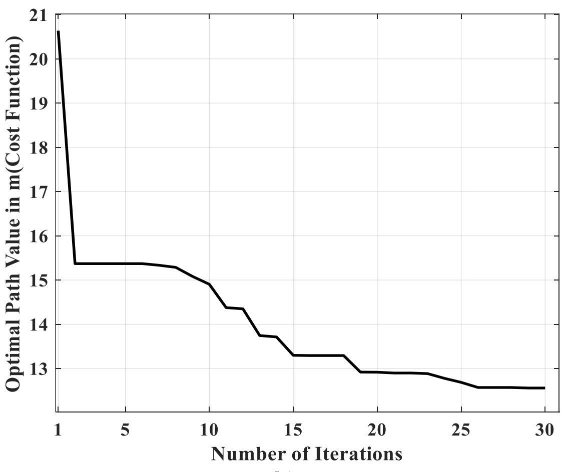
In addition, the hybrid method was compared with the hybrid Quarter Orbits Particle Swarm Optimization (QOPSO) algorithm in the same congested environment utilized in [15] with a workspace of [700×700] cm. The simulation process using the hybrid FAMCPSO algorithm shown in Fig. 12 (a) and (b) produces a path length equals 642.59 cm with an iteration number equals 11.

Table 5 shows the outcomes of the comparison study.

Thirdly, the proposed hybrid FAMCPSO was compared with the vertical cell decomposition (VCD) algorithm and the radial cell decomposition (RCD) algorithm that were suggested in [16], using different static environments including two cluttered environments with a workspace of [11×8] m, where the starting position is at (0, 0) and the end position is at (10, 7.5). Moreover, the comparison was also conducted in the corridor environment with a workspace of [10×10] m, where the starting position is at (0, 0) and the goal is at (6.9, 5.7) and (8.9, 7.8). The results of the simulation process with the first cluttered environment using the hybrid FAMCPSO algorithm are shown in Fig. 13 (a) and (b), which produces a path of length equals 12.534 m. While the results of the simulation process with the second environment using the proposed hybrid algorithm are shown in Fig. 14 (a) and (b), producing a path of length equals 14.103m. The simulation result with a corridor environment produced a path length of 9.677m and 12.558m to two different goals, as shown in Fig. 15 (a) and (b) and Fig. 16 (a) and (b). The simulation results in all the cases indicate that the hybrid algorithm can successfully generate the shortest path compared to both the VCD and the RCD in all environments [16] because these algorithms use



(a)

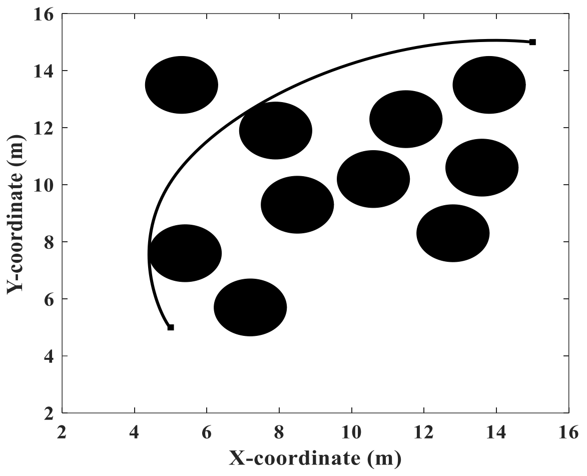


(b)

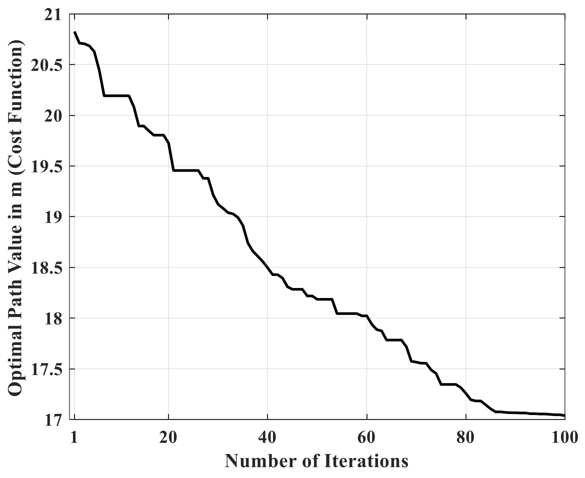
Figure. 16 Results of the hybrid FAMCPSO based on map [16]: (a) the shortest path and (b) the best distance cost function

Table 6. Comparison of the shortest path with the literature [16]

Environment no.	Algorithm Types	Best Path Length
Cluttered Environment 1 [16]	VCD [16]	17.88 m
	RCD [16]	16.66 m
	The Proposed hybrid	12.534 m
Cluttered Environment 2 [16]	VCD [16]	19.55 m
	RCD [16]	15.88 m
	The Proposed hybrid	14.103 m
Corridor environment 1 [16]	A* algorithm [16]	12.62 m
	RCD algorithm [16]	17.44 m
	The Proposed hybrid	9.677 m
Corridor environment 2 [16]	A* algorithm [16]	16.87 m
	RCD algorithm [16]	25.36 m
	The Proposed hybrid	12.558 m



(a)



(b)

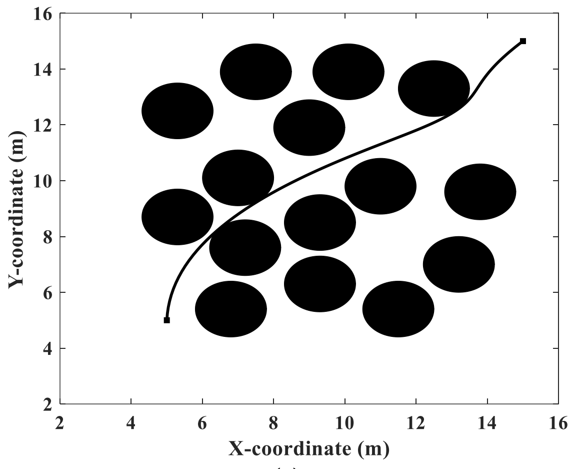
Figure. 17 Results of the hybrid FAMCPSO based on map [17]: (a) the shortest path and (b) the best distance cost function

a grid or vertical-line cells with a long distance between the cells based on the obstacles in the environment.

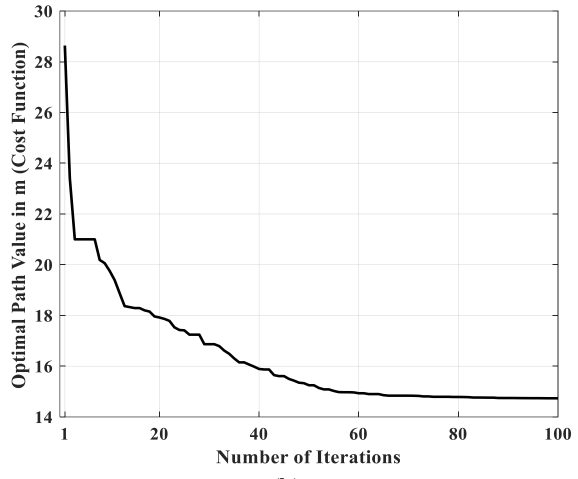
Table 6 The result of the comparison between the algorithms.

Finally, the hybrid FAMCPSO was compared with the GA, the PSO-W, and the FA that was developed in [17], using an environment with a workspace of [16×16] m, where the starting and ending positions are at (5,5) and (15,15), respectively.

The results of the simulation process using the hybrid FAMCPSO algorithm are shown in Fig. 17 (a) and (b) and Fig 18 (a) and (b), producing a path of length equals 17.03m and 14.73 m for map (a) and map (b), respectively. The results indicate that the hybrid FAMCPSO algorithm can successfully generate the shortest path compared to that of the GA, the PSO-W, and the FA [17], because the PSO-w and the GA are likely to be locked in local optima. The



(a)



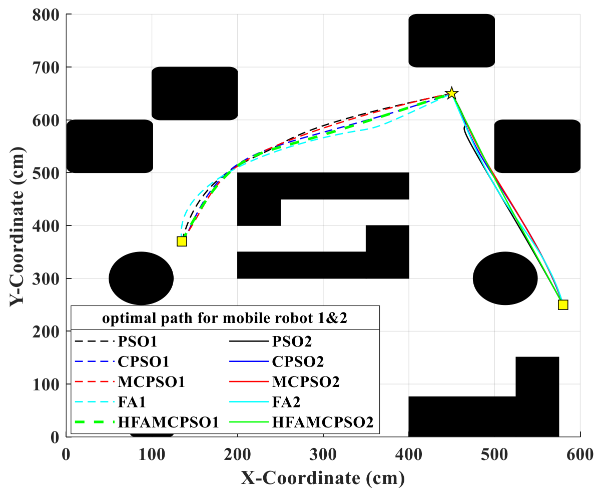
(b)

Figure. 18 Results of the hybrid FAMCPSO based on map [17]: (a) the shortest path and (b) the best distance cost function

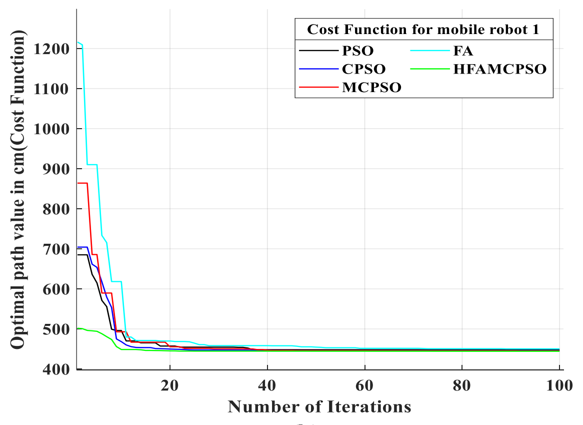
Table 7. Comparison of the shortest path with the literature [17]

Environment no.	Algorithm Types	Best Path Length
Map (a) [17]	GA [17]	17.3
	PSO-W [17]	17.13
	FA [17]	17.44
	The Proposed hybrid	17.03
Map (b) [17]	GA [17]	19.07
	PSO-W [17]	17.85
	FA [17]	17.96
	The Proposed hybrid	14.73

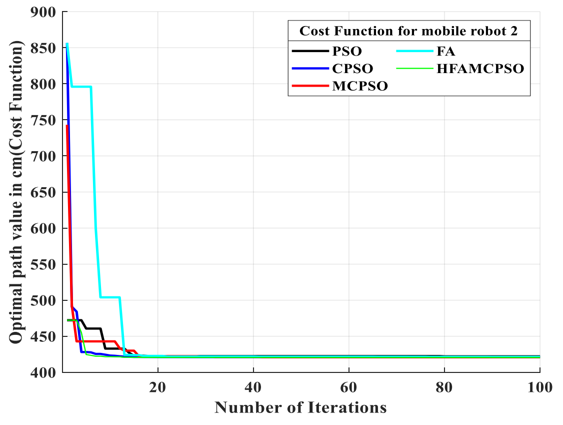
test also demonstrates that the FA beats the PSO-w and the GA in terms of the success rate within the acceptable length. On the other hand, the increment rate of the FA and the GA are so low, that the average



(a)



(b)



(c)

Figure. 19 The result for case B: (a) the shortest path for the two mobile robots, (b) the best distance cost function for mobile robot 1 and (c) the best distance cost function for mobile robot 2

length of the optimum path found using the PSO-w is less than that found using the FA and the GA. Table 7 shows the results of the comparison study.

Received: June 6, 2022. Revised: June 27, 2022.

Table 8. Comparison of the shortest paths

Mobile no.	Algorithm Types	Best Path Length	No. of Iterations
Mobile robot 1	PSO	447.961 cm	40
	CPSO	445.944 cm	37
	MCPSO	445.119 cm	29
	FA	450.245 cm	60
	FAMCPSO	444.037 cm	20
Mobile robot 2	PSO	422.167 cm	20
	CPSO	421.137cm	18
	MCPSO	420.944 cm	16
	FA	421.447 cm	23
	FAMCPSO	420.617 cm	13

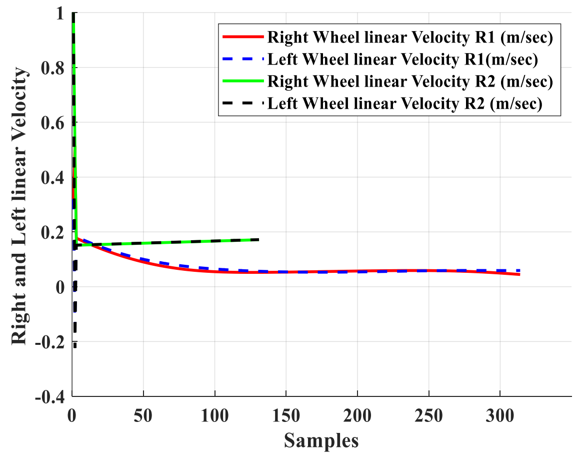


Figure. 20 The right and the left wheels' linear velocities for the two mobile robots

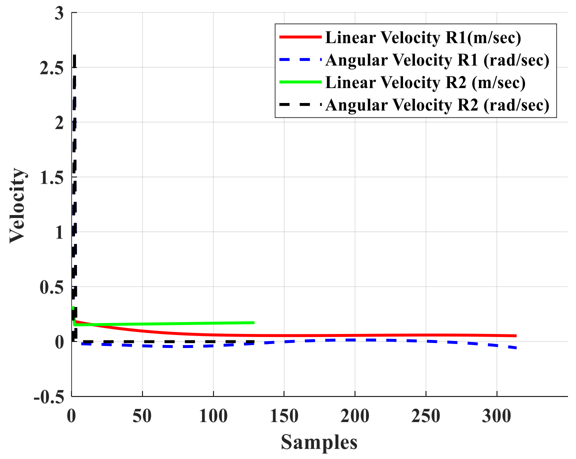


Figure. 21 The linear and the angular velocities for the two mobile robots

Case B

In this case, we tested the proposed algorithms with two mobile robots with different initial points at (135,370) cm and (580,250) cm (yellow square) and one goal point at (450,650) cm (yellow star). All cost functions and paths for case B are summarized in Table 8 and Fig. 19 (a), (b) and (c).

After executing the program several times, the result for the first mobile robot is obtained as the minimum distance based on the hybrid FAMCPSO algorithm, which is equal to 444.0371 cm at iteration 20, while the distance for the MCPSO algorithm is obtained as 445.1197 cm at iteration 29. The distance for the CPSO algorithm is found as 445.9442 cm at iteration 37, the distance for the PSO algorithm is equal to 447.9613 cm at iteration 40, and finally, the minimum distance for the FA is equal to 450.245 cm at iteration 60.

On the other hand, the result for the second robot is obtained as the minimum distance based on the proposed hybrid FAMCPSO algorithms, which is equal to 420.6174 cm at iteration 13, while the distance that was generated from the MCPSO algorithm is equal to 420.944 cm at iteration 16 and the distance obtained by the CPSO algorithm is equal to 421.1377cm at iteration 18.

The distance obtained from the PSO algorithm is 422.1673 cm at iteration 20. Finally, the minimum distance for the FA was 421.4473 cm at iteration 23. As a result, the distance generated from the two mobile robots using the suggested hybrid algorithm was smooth and the shortest way from the initial position to the target position when we compared with the original algorithms.

The equations of the reference path for the optimal path of the hybrid FAMCPSO for mobile robots 1 and 2 are represented in Eqs. (31) and (32), respectively.

$$Y_r(x_r) = -1.113 \times 10^{-7}\, x_r^4 + 0.00015035 \times x_r^3 - 0.074488 \times x_r^2 + 16.517 \times x_r - 837.45 \quad (31)$$

$$Y_r(x_r) = 0.0016411 \times x_r^2 - 4.7627 \times x_r + 2461.7 \quad (32)$$

Finally, utilizing Eqs. (23) to (30), Fig. 20 represents the smooth velocity response of the right and the left wheels for the two mobile robots. While Fig. 21 demonstrates the linear and the angular velocities of platform 1 and platform 2. Fig. 22 shows the linear force and the angular torque of the mobile robot platform for the two robots. Finally, Fig. 23 represents the right and the left wheels' torque control actions for mobile robot 1 and mobile robot 2.

Received: June 6, 2022. Revised: June 27, 2022.

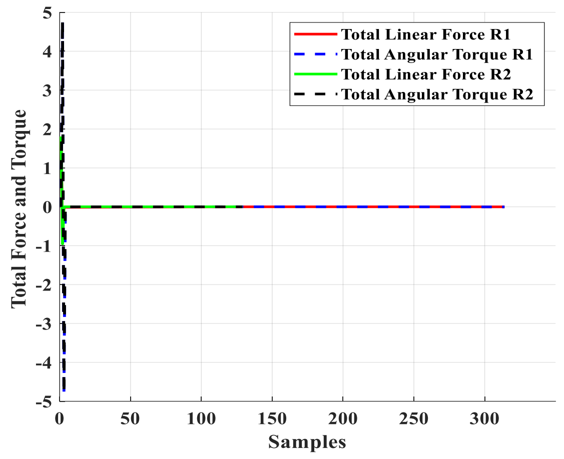


Figure. 22 The linear force and the angular torque for the two mobile robots

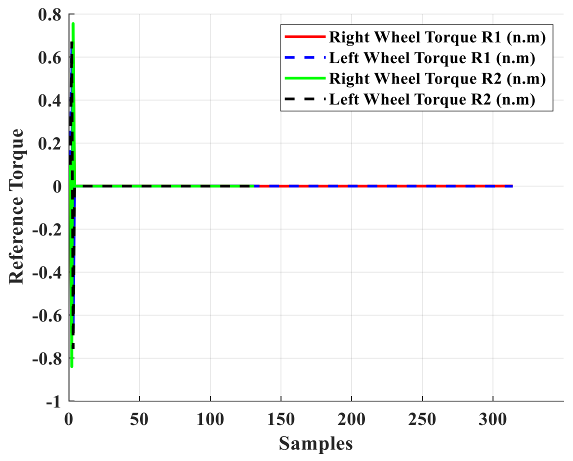


Figure. 23 The right and left wheels' torques for the two mobile robots

5. Conclusions

The mobile robot path-planning algorithm is a crucial part of the robotics area that focuses on finding the smoothest and the shortest path of the mobile robot in the global environment and this algorithm has been proposed in this paper. A meta-heuristic algorithm is proposed for solving the path planning problem for the mobile robot, and it is used to find the best way in the environment with obstacles. This algorithm combined the MCPSO with the FA algorithms. From the simulation results of MATLAB, we can find that the suggested hybrid algorithm can find the most optimum path for the mobile robot compared to those of the original algorithms under the same conditions of obstacles in the environment.
On the other hand, the hybrid algorithm was compared with the artificial potential field (APF) method combined with the particle swarm optimization (PSO) with a three-point smoothing method [11] and the fast-marching method hybridized with regression search (FMMHRS) methodology [22]. The comparison results showed that the hybrid algorithm provides enhancement on the path length of 12.25% compared to the APF+PSO+3-point algorithm and 0.5% compared to the FMMHRS algorithm.

On the other hand, when comparing the hybrid algorithm with the quarter orbits particle swarm optimization (QOPS) algorithm [15], the hybrid algorithm provides an enhancement on the path length of 2.2% compared to the QOPS algorithm.

In addition, the hybrid algorithm was compared with the radial cell decomposition (RCD) [16], the vertical cell decomposition (VCD) [16], and the A* algorithm [16] using four different (cluttered and corridor) environments. The comparison results of the first cluttered environment showed that the hybrid algorithm provides enhancement on the path length of 24.7%, up to 29.8% compared to the RCD and VCD algorithms, respectively. Moreover, the comparison results of the second cluttered environment showed that the hybrid algorithm provides enhancement on the path length equals 11.1% and 27.8% compared to the RCD and VCD algorithms, respectively. While the first corridor environment comparison results showed that the hybrid algorithm improves the path length by 44.5% and 23.3% when compared to the RCD and A* algorithms, respectively. Finally, the other corridor environment enhancement on the path length equals 43.3% and 25.5% compared to the RCD and the A* algorithm, respectively.

Finally, the hybrid algorithm provides a smooth path with the shortest distance compared to those of the genetic algorithm (GA) [17], the particle swarm with inertia weight (PSO-w) algorithm [17], and the firefly algorithm (FA) [17] in two different maps. In the first map, when comparing the hybrid algorithm with the GA algorithm, the hybrid algorithm provided an enhancement on the path length equals to 1.5% compared to the GA algorithm, while the proposed hybrid algorithm enhancement on the path length equals 0.5% compared to the PSO-w algorithm and 2.3% compared to the FA algorithm. While in the second map, when comparing the hybrid algorithm with the GA algorithm, the hybrid algorithm provided an enhancement on the path length equals 22.7% compared to the GA algorithm, while the proposed hybrid algorithm enhancement on the path length equals 17.4% compared to the PSO-w algorithm and 17.9% compared to the FA algorithm.

A comparison of the mobile robot path length with other research works revealed that the proposed

Received: June 6, 2022. Revised: June 27, 2022.

hybrid method provided the shortest path with collision avoidance.
We recommend that the suggested hybrid algorithm be adjusted to work in a dynamic environment in the future.

Conflicts of interest

There are no conflicts of interest declared by the authors.

Author contributions

Noor Abdul Khaleq Zghair, and Ahmed Sabah Al-Araji enhanced and developed an intelligent path-planning algorithm for mobile robot model. Noor Abdul Khaleq Zghair described the proposed algorithm for path planning. Ahmed Sabah Al-Araji explained the dynamic mobile model. The two authors discussed the proposed simulation results of this work.

References

[1] N. Buniyamin, W. A. J. Wan, N. Sariff, and Z. Mohamad, "A Simple Local Path Planning Algorithm for Autonomous Mobile Robots", *International Journal of Systems Applications, Engineering & Development*, Vol. 5, No. 2, pp. 151-159, 2011.

[2] B. Song, Z. Wang, and L. Zou, "An Improved PSO Algorithm for Smooth Path Planning of Mobile Robots Using Continuous High-Degree Bezier Curve", *Applied Soft Computing*, Vol. 100, pp. 1-11, 2021.

[3] N. A. Zghair, and A. S. A. Araji, "A One Decade Survey of Autonomous Mobile Robot Systems", *International Journal of Electrical and Computer Engineering (IJECE)*, Vol. 11, No. 6, pp. 4891-4906, 2021.

[4] B. Mahadevaiah and S. M., "An Overview of Path Planning and Obstacle Avoidance Algorithms in Mobile Robots", *International Journal of Engineering Research and Technology*, Vol. 8, No. 12, 2019.

[5] P. Sudhakara and V. Ganapathy, "Trajectory Planning of a Mobile Robot Using Enhanced A-Star Algorithm", *Indian Journal of Science and Technology*, Vol. 9, No. 41, pp. 478- 482, 2016.

[6] C. Saranya, M. Unnikrishnan, S. A. Ali, D. S. Sheela, and V. R. Lalithambika, "Terrain Based D∗ Algorithm for Path Planning", *IFAC-Papers Online*, Vol. 49, No. 1, pp. 178–182, 2016.

[7] B. K. Patle, A. Jha, A. Pandey, N. Gudadhe and S. K. Kashyap, "The Optimized Path for A Mobile Robot Using Fuzzy Decision Function", In: *Proc. of Conf. On Materials Processing and Characterization*, Vol. 18, No. 7, pp. 3575-3581, 2019.

[8] M. Li, C. Wang, Z. Chen, X. Lu, M. Wu, and P. Hou, "Path Planning of Mobile Robot Based on Genetic Algorithm and Gene Rearrangement", *Chinese Automation Congress (CAC)*, pp. 6999-7004, 2017.

[9] P. I. Adamu, H. I. Okagbue, and P. E. Oguntunde, "Fast and Optimal Path Planning Algorithm (FAOPPA) for A Mobile Robot", *Wireless Personal Communications*, Vol. 106, No. 2, pp. 577-592, 2019.

[10] F. H. Ajeil, K. I. Ibraheem, A. T. Azar, A. J. Humaid, "Grid-Based Mobile Robot Path Planning Using Aging-Based Ant Colony Optimization Algorithm in Static and Dynamic Environments", *Sensors*, Vol. 20, No. 7, pp. 1-26, 2020.

[11] R. K. Mandava, S. Bondada, and P. R. Vundavilli, "An Optimized Path Planning for the Mobile Robot using Potential Field Method and PSO algorithm", *Soft Computing for Problem Solving, Advances in Intelligent Systems and Computing*, Vol. 817, pp. 139-150, 2017.

[12] M. Fuad, T. Agustinah, and D. Purwanto, "Modified Headed Social Force Model based on Hybrid Velocity Obstacles for Mobile Robot to Avoid Disturbed Groups of Pedestrians", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 3, pp. 222-241, 2021, doi: 10.22266/ijies2021.0630.20.

[13] L. Xu, B. Song, and M. Cao, "A New Approach to Optimal Smooth Path Planning of Mobile Robots with Continuous-Curvature Constraint", *Systems Science & Control Engineering*, Vol. 9, No. 1, pp. 138–149, 2021.

[14] O. A. Wahhab and A. A. Araji, "Path Planning and Control Strategy Design for Mobile Robot Based on Hybrid Swarm Optimization Algorithm", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 3, pp. 565-579, 2021, doi: 10.22266/ijies2021.0630.48.

[15] Z. E. Kanoon, A. A. Araji, and M. A. Salam, "Enhancement of Cell Decomposition Path-Planning Algorithm for Autonomous Mobile Robot Based on an Intelligent Hybrid Optimization Method", *International Journal of Intelligent Engineering and Systems*, Vol. 15, No. 3, pp. 161-175, 2022, doi: 10.22266/ijies2022.0630.14.

[16] O. A. Salama, M. E. Eltaib, H. A. Mohamed, and O. Salah, "RCD: Radial Cell Decomposition

Algorithm for Mobile Robot Path Planning", *IEEE Access*, Vol. 9, pp. 149982-149992, 2021.

[17] B. Li, L. Liu, Q. Zhang, D. Lv, Y. Zhang, J. Zhang, and X. Shi, "Path Planning Based on Firefly Algorithm and Bezier Curve", In: *Proc. of International Conf. on Information and Automation (ICIA)*, pp. 630-633, 2014.

[18] X. S. Yang, "Firefly Algorithms for Multimodal Optimization", In: *Proc. of International Symposium on Stochastic Algorithms. Springer*, Vol. 5792, pp. 169-178, 2009.

[19] L. Zhang1, L. Liu, X. S. Yang, and Y. Dai, "A Novel Hybrid Firefly Algorithm for Global Optimization", *PLoS ONE*, Vol. 11, No. 9, pp. 1-17, 2016.

[20] A. K. A. Hassan and D. J. Fadhil, "Mobile Robot Path Planning Method Using Firefly Algorithm For 3D Sphere Dynamic & Partially Known Environment", *Journal of University of Babylon for Pure and Applied Sciences*, Vol. 26, No. 7, pp. 309-320, 2018.

[21] K. Li, Q. Hu, and J. Liu, "Path Planning of Mobile Robot Based on Improved Multi Objective Genetic Algorithm", *Wireless Communications and Mobile Computing*, Vol. 2021, pp. 1-12, 2021.

[22] R. K. Mandava, K. Mrudul, and P. R. Vundavilli, "Dynamic Motion Planning Algorithm for a Biped Robot Using Fast Marching Method Hybridized with Regression Search", *Acta Polytechnica Hungarica*, Vol. 16, No. 1, pp. 189-208, 2019.

[23] A. S. A. Araji, A. K. Ahmed, and M. K. Hamzah, "Development of a Path Planning Algorithms and Controller Design for Mobile Robot", In: *Proc. of Scientific Conf. On Electrical Engineering*, pp. 72–77, 2018.

[24] A. S. A. Araji, M. F. Abbod, and H. S. A. Raweshidy, "Design of a Neural Predictive Controller for Nonholonomic Mobile Robot based on Posture Identifier", In: *Proc. of the Lasted International Conf. on Intelligent Systems and Control*, pp. 198-207, 2011.

[25] A. S. A. Araji, K. E. Dagher, and B. A. Ibraheem, "An Intelligent Cognitive System Design for Mobile Robot based on Optimization Algorithm", In: *Proc. of the Scientific Conf. On Electrical Engineering*, pp. 84-89, 2018.

[26] R. Hosseinpourfard and M. M. Javidi, "Chaotic PSO using the Lorenz System: An Efficient Approach for Optimizing Nonlinear Problems", *Çankaya University Journal of Science and Engineering*, Vol. 12, No. 1, pp. 040-059, 2015.

[27] J. Lian, W. Yu, K. Xiao, and W. Liu, "Cubic Spline Interpolation-Based Robot Path Planning Using a Chaotic Adaptive Particle Swarm Optimization Algorithm", *Mathematical Problems in Engineering*, Vol. 2020, pp. 1-20, 2020.

[28] W. Yang, X. Zhou, and Y. Luo, "Simultaneously Optimizing Inertia Weight and Acceleration Coefficients Via Introducing New Functions into PSO Algorithm", *Journal of Physics: Conference Series*, Vol. 1754, No. 1, pp. 012195, 2021.

[29] B. Abhishek, S. Ranjit, T. Shankar, G. Eappen, P. Sivasankar, and A. Rajesh, "Hybrid PSO-HSA and PSO-GA algorithm for 3D path planning in autonomous UAVs", *SN Applied Sciences*, Vol. 2, No. 11, pp. 1-16, 2020.

[30] M. M. Jawad and E. A. Hadi, "A Comparative Study of Various Intelligent Algorithms-Based Path Planning for Mobile Robots", *Journal of Engineering*, Vol. 25, No. 6, pp. 83-100, 2019.

[31] I. Fister, I. Fister Jr., X. S. Yang, and J. Brest, "A Comprehensive Review of firefly Algorithms", *Swarm and Evolutionary Computation*, Vol. 13, pp. 34-46, 2013.

[32] I. B. Aydilek, I. H. Karaçizmeli, M. E. Tenekeci, S. KAYA, and A. Gümüşçü, "Using Chaos Enhanced Hybrid Firefly Particle Swarm Optimization Algorithm for Solving Continuous Optimization Problems", *Sādhanā*, Vol. 46, No. 65, 2021.

[33] W. Pei, G. Huayu, Z. Zheqi, and L. Meibo, "A Novel Hybrid Firefly Algorithm for Global Optimization", In: *Proc. of International Conf. on Computer and Communication Systems (ICCCS), IEEE*, pp. 164-168, 2019.

[34] V. Rajarajan and J. S. C. M. Raj, "Mppt Based on Modified Firefly Algorithm", *Journal of Selected Areas in Microelectronics*, Vol. 8, No. 2, pp. 94-100, 2016.

[35] K. E. Dagher and M. N. Abdullah, "Airborne Computer System Based Collision-Free Flight Path Finding Strategy Design for Drone Model", *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 6, pp. 234-248, 2021, doi: 10.22266/ijies2021.1231.22.