DOI: https://doi.org/10.33103/uot.ijccce.21.2.4

An Optimal Path Planning Algorithms for a Mobile Robot

Omar Abdul Razzaq Abdul Wahhab[1], Ahmed Sabah Al-Araji[2]
[1, 2]*Computer Engineering Dept, University of Technology, Baghdad, Iraq*
[1]*ce.19.08@grad.uotechnology.edu.iq*, [2]*60166@uotechnology.edu.iq*

Abstract- *The goal of navigating a mobile robot is to find the optimal path to direct its movement, so path planning is the best solution to find the optimal path. Therefore, the two most important problems of path planning must be solved; the first is that the path must avoid collision with obstacles, and second it must reduce the length of the path to a minimum. This paper will discuss finding the shortest path with the optimum cost function by using the Chaotic Particle Swarm Optimization (CPSO), and A*, compare the results between them and the proposed hybrid algorithm that combines A* and Chaotic Particle Swarm Optimization (ACPSO) algorithms to enhance A* algorithm to find the optimal path and velocities of the wheeled mobile robot. These algorithms are simulated by MATLAB in a fixed obstacles environment to show the effectiveness of the proposed algorithm in terms of minimum number of an evaluation function and the shortest path length as well as to obtain the optimal or near optimal wheel velocities.*

Index Terms*—Mobile robot, Path planning, Chaotic Particle Swarm Optimization (CPSO), A* algorithm, Fixed obstacles.*

I. INTRODUCTION

Path planning can be defined as the task of a mobile robot for finding the optimal or shortest path between two points while avoiding collision with obstacles. Its mechanism should be established by decreasing the number of the bends as well as the rotation amount, which reduces the amount of braking, resulting in the shortest path between the start and the final target point [1], [2]. This is the reason why, the movement of mobile robot management must follow and implement path planning since mobile robots serve many practical purposes in real world applications such as industry, weather forecasting, mining, science, education, entertainment, security, and the military [3], [4], [5].

Recently, many types of wonderful technologies have been invented such as: Daniel et al. in [6] developed an accurate mapping of the trajectory of a moving robot using PSO with radial foundation functions. Its diagram describes the working area of the moving robot. The suboptimal path is obtained with Dijkstra's algorithm and the optimal path is obtained with PSO with radial foundation functions. Only fixed obstacles are considered. It provides a smooth, crash-free trajectory that a moving robot should follow regardless of the obstacle's geometry. Panda and Choudhury [7] proposed genetic algorithm that gives a particular form of solution to the complex motion preparation problems of mobile robots in uncertain dynamic conditions depending on action dynamics. Also, Contreras-Cruz et al. [8] presented the Ant and Bee colony optimizations that used these algorithms in the local search technique and the proposed solution incorporates the artificial bee colony algorithm and the evolutionary programming algorithm to optimize the feasible path discovered by a series of local procedures. These pathways can generate a route for a robot depending on different optimization algorithms inspired by the organism's attitudes, fitness functions, and various constraints in the workspace. Duguleana and Mogan [9] explained the neural network technique with Q-learning to address route planning issues in the generated solution and it addresses the Radial Basis Function Neural Network (RBFNN). Ghosh et al. [10] demonstrated the Wavelet Neural Network (WNN) architecture of smart controllers for mobile robot route planning in an uncertain area and compared it

Received 19/2/2021; Accepted 29/4/2021

 ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

DOI: https://doi.org/10.33103/uot.ijccce.21.2.4

with other methods to show the efficiency of the poroposed method. However, Tang et al. [11] used a Random-Disturbance Self-Adaptive Particle Swarm Optimization (RDSAPSO) to fine-tune the three control parameters in RDSAPSO to dynamically change RDSAPSO's discovery and extraction capability because of PSO convergence was paramount and affects the quality of the route produced. Liu et al. [12] explained Ant Colony Optimization (ACO) that are used in the search process for the globally optimal direction, pheromone diffusion and geometric local optimization are combined. During the ant scanning process, the present path pheromone diffuses in the direction of the possible field power, so ants prefer to aim for a higher fitness subspace, and the search space of the test pattern becomes smaller. Also, firefly algorithm is discussed as swarm intelligence by Hidalgo-Paniagua [13] in order to achieve detailed and effective solutions, the current MO-FA deals with three separate aims. These aims are as follows: protection of the road, length of the path and smoothness of the route (related to the energy consumption).

In this paper, to solve the path planning problem, a hybrid swarm algorithm A* and Chaotic Particle Swarm Optimization (ACPSO) has been proposed and compared to A* and Chaotic Particle Swarm Optimization (CPSO0) algorithms that generate a shorter path in a static environment with a better distance cost function and found the better wheel velocities of the mobile robot that can be applied to path planning in a fixed environment.

This paper is structured as follows: Section 2 describes the kinematic mobile robot model. Section 3 explains the proposed hybrid path planning algorithm. Section 4 shows the numerical results and analysis of the simulation of MATLAB in a static environment, and in Section 5 the conclusions are discussed.

II. THE MODEL OF KINEMATIC MOBILE ROBOT

The platform of the Wheeled Mobility Robot (WMR) is shown in *Fig. 1*, which has two wheels mounted on a parabolic shaft, and two multi-directional wheels are installed in the front and end of the platform. The two castor wheels are carried by the mechanical structure and keep the body stable. Two independent analog Direct Current (DC) motors are actuated as the right and left wheel actuators of the wheeled robot for movement and platform steer. The point O_m is the location of the WMR center mass, the two drive wheels are connected to the center of the axis [14].

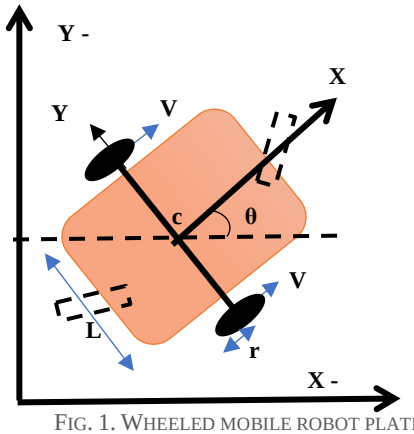


FIG. 1. WHEELED MOBILE ROBOT PLATFORM.

The kinematic equation of the mobile robot platform based on its position in the global coordinate frame $[O_A, X_A, Y_A]$ and the pose surface are x_{Pose} and y_{Pose} are the coordinates of a point O_m . θ_{Pose} is the mobile robot's direction angle measured from the X_A axis and these three

Received 19/2/2021; Accepted 29/4/2021

 ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

DOI: https://doi.org/10.33103/uot.ijccce.21.2.4

generalized coordinates can describe the configuration of the mobile robot. So, the computer simulation equation can be represented as follows [15]:

$$x_{pose}(kT) = x_{pose}((k-1)T)) + \frac{\left((V_R(kT)+V_L\,(kT))\times\cos\left(\theta(kT)_{pose}\right)\right)}{2} \times T \quad (1)$$

$$y_{pose}(kT) = y_{pose}((k-1)T)) + \frac{V_R(kT)+V_L(kT))\times\sin(\theta(kT)_{pose})}{2} \times T \quad (2)$$

$$\theta(kT)_{pose} = \theta((k-1)T)_{pose} + \frac{((V_R(kT)-V_L\,(kT))}{L} \times T \quad (3)$$

Where, *Vr(k)* is denoted as right wheel velocity of the platform. *Vl(k)* is denoted as left wheel velocity of the platform. *L* is denoted as the length between the driving wheels of the platform. *Ts* is denoted as the sampling time of the numerical calculation.

III. PATH PLANNING ALGORITHMS

When we want to move a robot between two nodes, it is useful to use the local map to calculate a global path. Planning the route is an engineering issue because it is defined as constructing an engineering path, without mentioning any specific time law. Many researchers have been working extensively to develop efficient methods to avoid collision with obstacles and obtain a smooth path. A briefl review of path planning algorithms is presented in the section below.

A. A* Path-Planning Algorithm

The A* Algorithm is known as a heuristic search algorithm, finds the optimal path by checking among all possible routes of a solution to problems with the minimal cost. It visits the nodes in the graph from the starting node to the target node. The prescriptive information about the properties of the issue is applied to guide its performance [16]. It is based on two standards algorithms, the first is the Dijkstra's algorithm, and the second is Greedy Best-First-Search's algorithm.

Dijkstra's algorithm is designed to find the shortest path in a graph between two nodes. The algorithm visits the nodes in a graph one by one beginning from the starting point of the object [17]. The Greedy Best-First-Search's algorithm also keeps track of a frontier to locate the target. This algorithm makes use of a heuristic function which determines approximately how far from the goal a particular node is. The Dijkstra's algorithm selects the node nearest to the starting point, while here the node closest to the goal is selected and given higher priority than those nodes which are far away. A* balances between Dijkstra's algorithm by finding the shortest path without fail, g(n) and the Best-First-Search's algorithm by estimating the distance to the target, h(n).

In the main loop, the algorithm repeatedly checks which (n) vertex has the lowest value of f(n) as in the evaluation (4) and (5). *Fig. 2* shows a flowchart of the algorithm [18]. Dijkstra's algorithm consumes time and resources to explore unsafe directions, while Greedy Best-First-Search always fails to find the shortest path to reach a goal. Therefore, Algorithm A* uses both distances from the starting point and approximate distance to the target point to remove the limitations of these traditional algorithms by combining these two algorithms [19].

$$\mathrm{f\,(n)} = \mathrm{g\,(n)} + \mathrm{h\,(n)} \quad (4)$$

$$\mathrm{h\,(n)} = \sqrt{(x_{n+1}-x_n)^2 + (y_{n+1}-y_n)^2} \quad (5)$$

where, n is denoted by the current node, g (n) is denoted by the cost distance function from starting point to the current node n, and h (n) is denoted by the estimation minimum cost distance function from the current node to end point that is calculated by (5).

Received 19/2/2021; Accepted 29/4/2021

 ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

DOI: https://doi.org/10.33103/uot.ijccce.21.2.4

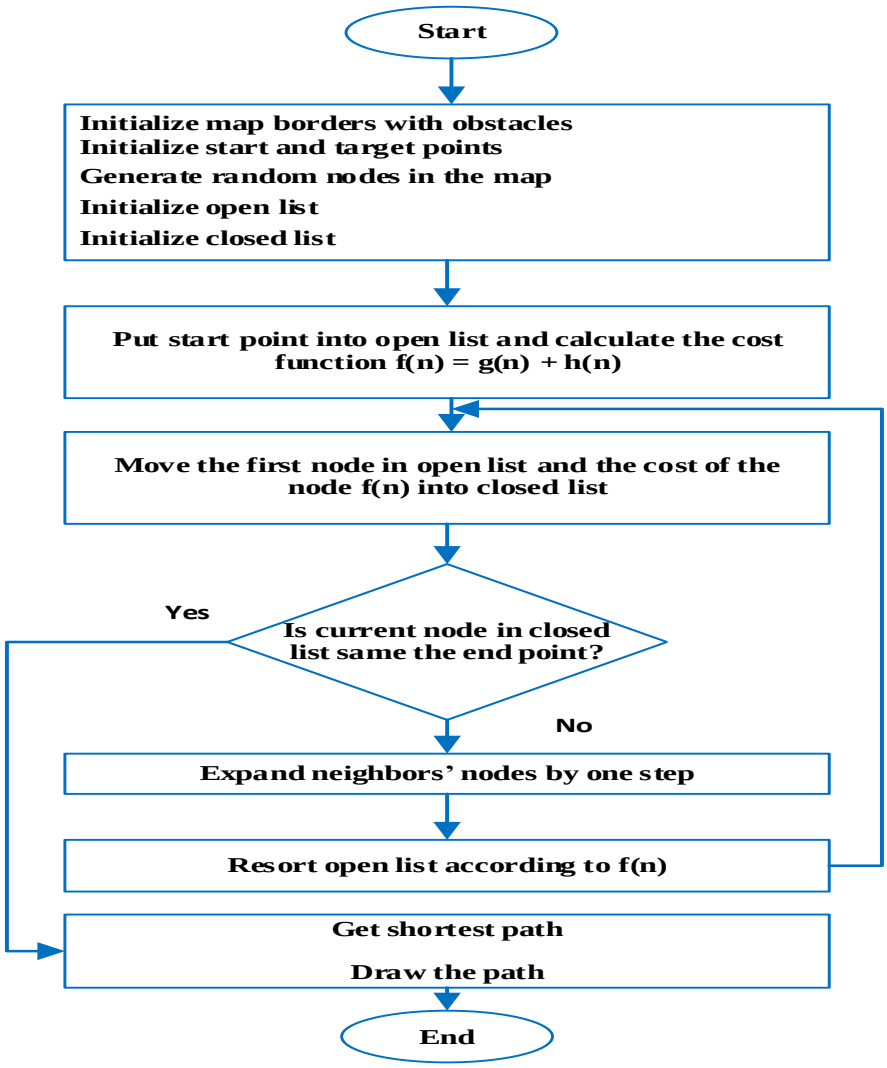


FIG. 2. THE FLOWCHART OF THE A* ALGORITHM [18].

B. Chaotic Particle Swarm Optimization Path-Planning Algorithm

The problem of robot path planning is treated as a minimization problem and is considered on the transformed search space limited by constraints. PSO is an experimental community based on multi-point research technology that simulates the social behavior of a flock of birds, a school of fish, etc. [20]. Research begins with a set of research points called molecules because particles have a memory and they save part of their previous condition. The particles maintain their individuality in all cases, although they share the same point in the belief of space without limitations. The individuality and sociality are two randomly weighted factors that influenced the particle's movement. The definition of individuality is "the tendency to return to the particles best past situation" while sociality is defined as "the tendency to move towards the neighborhood's best previous situation". Each particle is encoded by a location vector (initially randomly chosen) and the position is updated using its velocity (randomly chosen at the beginning) in successive iterations. At each time step, PSO changes the speed of each particle to its optimum positions. Acceleration is measured in random terms, with separate random numbers are generated to accelerate to the best positions. The search by PSO algorithm is subjected to stagnation due to early convergence so the search process should be diversified [21]-[24]. Chaos is introduced into the PSO to induce more randomness in the search for PSO [25]-[26]. A small error in particle position may make a big difference to their behavior for a long time and prevent them from getting trapped in some local optimal solution.

Received 19/2/2021; Accepted 29/4/2021

 ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

DOI: https://doi.org/10.33103/uot.ijccce.21.2.4

After applying chaotic (6), (7) and (8), each particle updates its velocity and position by using the (9) and (10) [27].

$$\beta^{b+1} = \mu \times \beta^b (1 - \beta^b) \qquad 0 \leq \beta^1 \leq 1 \tag{6}$$

$$W = W_{max} - [(W_{max} - W_{min}) * (\frac{iter}{iter_{max}})] \tag{7}$$

$$W_{new} = W \times \beta^{b+1} \tag{8}$$

$$[v(i,j)]_a^{b+1} = \begin{bmatrix} W_{new} \times v(i,j) + c_1 \times rand(\,) \times \left(pbest(i,j) - xy(i,j)\right) \ldots \\ +c_2 \times rand(\,)(gbest(i,j) - xy(i,j)) \end{bmatrix}_a^b \tag{9}$$

$$[x(i,j)]_a^{b+1} = [xy(i,j)]_a^b + [v(i,j)]_a^{b+1} \tag{10}$$

where, a is denoted as the particle number in the total population, b is denoted as the iteration number, and (i, j) is denoted as co-ordinates number in x and y axis, respectively.

Table I shows the parameters of CPSO that will be used in the simulation results and *Fig. 3.* shows the proposed pseudocode of the CPSO algorithm.

TABLE I. THE FINAL CHOICE OF A PARAMETER WAS CONSIDERED TO BE THE OPTIMAL CHOICE

Parameter	Definition with value
β^0	The initial value of deterministic β = 0.3
μ	The control parameter with a real value μ = 4
W	Inertia Weight
W_{min}	Minimum W = 0.3
W_{max}	Maximum W = 0.9
$iter$	Current iteration number
$iter_{max}$	Maximum number of iterations
c_1, c_2	Coefficient of acceleration (1.25, 1.25)
V_i^t	The velocity of particle i^{th} in iteration t^{th}
xy_i^t	The position of particle i^{th} in iteration t^{th}
pbest_i	Best fitness values for particle i^{th}
Gbest	Best fitness values for the whole swarm

Basic CPSO Procedure:

Step 1: Maximum iterations

Step 2: Initialize particle.

Step 3: Each particle**,** checking fitness value, if the fitness value is better than the best fitness value (pbest) then set current value as new pbest

Step 4: Each particle

- Find the particle with the best fitness (gbest) in the particle neighborhood
- Apply Chaotic optimization algorithm eq. (5, 6, and 7)
- According to the velocity equation (8) calculate particle velocity $v(i,j)$
- Apply the new velocity
- According to the position equation (9), update the particle position $x(i,j)$
- Apply the new position

Step 5: Repeat **Step 3** until stop

FIG. 3. THE PROPOSED PSEUDOCODE OF CPSO ALGORITHM.

Received 19/2/2021; Accepted 29/4/2021

 ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

DOI: https://doi.org/10.33103/uot.ijccce.21.2.4

C. Hybrid A* with chaotic particle swarm optimization (ACPSO)

The key to finding the optimal solution (the shortest path) in the A* algorithm is to choose the evaluation function f(n): the value of h(n) is less than the real distance value from n to the target node.

In this case, while there are several search points, which means that the search range is wide and the efficiency is limited, the optimal solution can be found if the approximate distance h(n) is equal to the shortest distance. Therefore, the search will strictly follow the shortest path and the performance of the search is the maximum at this time. The number of search points is limited if the expected value is higher than the real value, which means that the search range is small and the search efficiency is high, but the optimum solution cannot be assured.

In fact, there is a need to create the function closer to the actual shortest path if we want to correctly obtain the optimal path, i.e., there is a need to relate to more heuristic facts, such as the relationship between the selected node and the end point as the weight from the selected node to the end point, etc., but more weaknesses will occur so h(n) should be quantified. This implies that the optimal solution can technically be obtained by the A* algorithm, but its greatest downside is that it takes so much space. For example, by selecting a better evaluation function to minimize the solution space, certain time-for-space approaches can be used to increase efficiency.

A hybrid algorithm is proposed in the development of the search feature of the algorithm A* to find the shortest path to reach the target point in less time using the chaotic particle swarm optimization algorithm by the proposed hybrid flowchart algorithm as shown in in *Fig. 4* comprising of two steps:

- Initialization step: generating random nodes (n) and finding all possible routes (r) from source to destination point, then calculating the cost distance function for each route with A* cost distance function as shows in Equations (11) and (12), a dynamic weight has been proposed and called enhanced factor (Ef) in the heuristic function which can minimize area search, where Ef is a random number less than 1, and then saves the routes cost into matrix called (Rc), after that sorting the matrix (Rc) to find the lowest cost and obtain its index; Finally, the costs of the (Rc) will be saved into the initialized particles and will evaluate the local and global cost distance function for best solution in the CPSO algorithm.
- Iteration step: after getting the minimum global cost distance function, the main iteration starts and enforces all particles to update their velocity and position according to global cost distance function till the end of the iteration.

$$H(n) = Ef(n) \times h(n) \quad (11)$$

$$f(n) = g(n) + H(n) \quad (12)$$

where, H(n) is denoted the enhancement heuristic function. Ef(n) is denoted the enhancement random factor < 1.

Received 19/2/2021; Accepted 29/4/2021

 ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

DOI: https://doi.org/10.33103/uot.ijccce.21.2.4

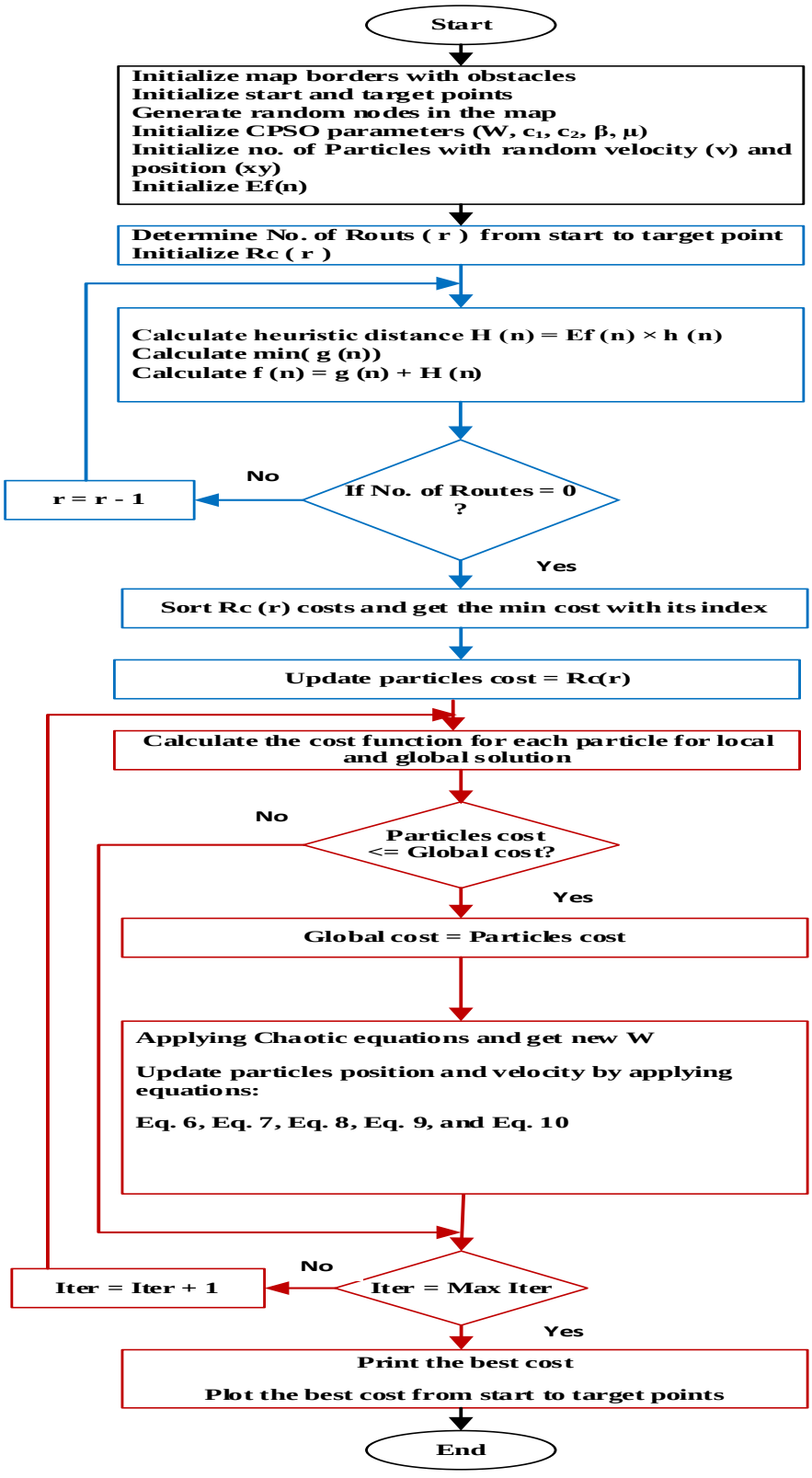


FIG. 4. THE PROPOSED HYBRID ACPSO FLOWCHART.

Received 19/2/2021; Accepted 29/4/2021

ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

DOI: https://doi.org/10.33103/uot.ijccce.21.2.4

IV. NUMERICAL RESULTS AND ANALYSIS

The mobile robot and the obstacle have a particular volume, and the actual obstacles are irregular patterns. If the planned path is too close to obstacles, the mobile robot can easily collide with the obstacles along the planned path. To avoid this, a map of the obstacle is proposed, where the black grids represent the obstacles, and the white grids represent the area where the robot can pass. After the standardization of the obstacle, even if the planned path is closed to the black grids, the robot is still safe to run along the planned path due to the safety distance maintained between the robot and obstacles.

The path should display a smooth curvature to ensure its reliability. The controlled robot is to traverse this workspace 500*500 (cm) as shown in *Fig. 5*.

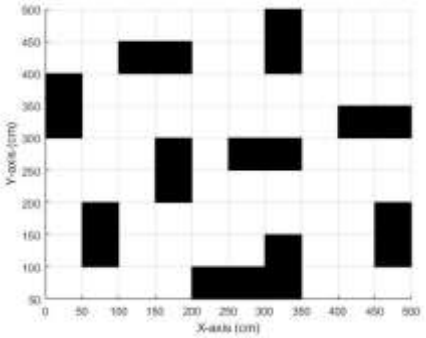
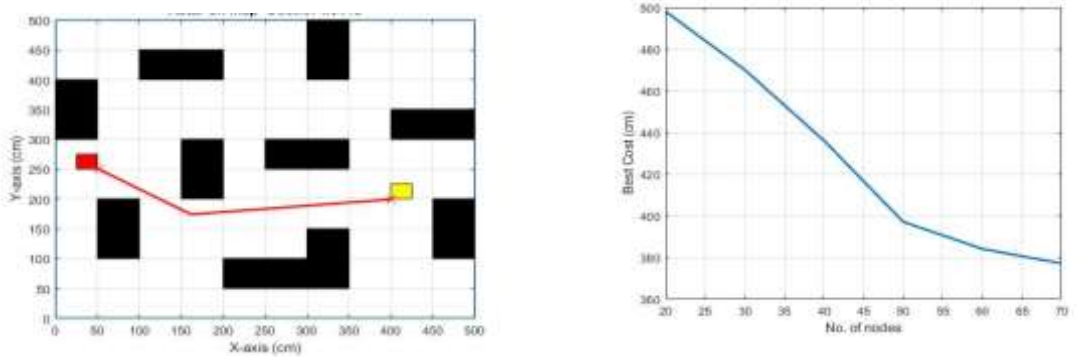


FIG. 5. THE PROPOSED ENVIRONMENT WITH STATIC OBSTACLES.

The environment is populated by static obstacles and full information about the positions of all objects in the workspace is available. The task of finding this collision-free path is the responsibility of a path-generating algorithm, that three algorithms were applied (A*, CPSO, and Hybrid ACPSO) and compared to find the optimum path with the best cost distance function. The program workflow begins with the acquisition of the robot's current position, its destination, and positions of obstacles. The valuation of the world state is based on the collected data, which is earlier preprocessed and transformed. If this condition is met the program checks whether the straight path is not closed due to the obstacles. If the path is open, it becomes the new path. Moreover, two cases of starting and stop point will be used to compare the results of each algorithm.

Case 1:

The initial position 50*250 cm (red point) to the destination point 400*200 cm (yellow point) as shown in *Fig. 6-a* when it applied A* algorithm and obtained the cost distance function is equal to 374 cm as shows in *Fig. 6-b.*



a b

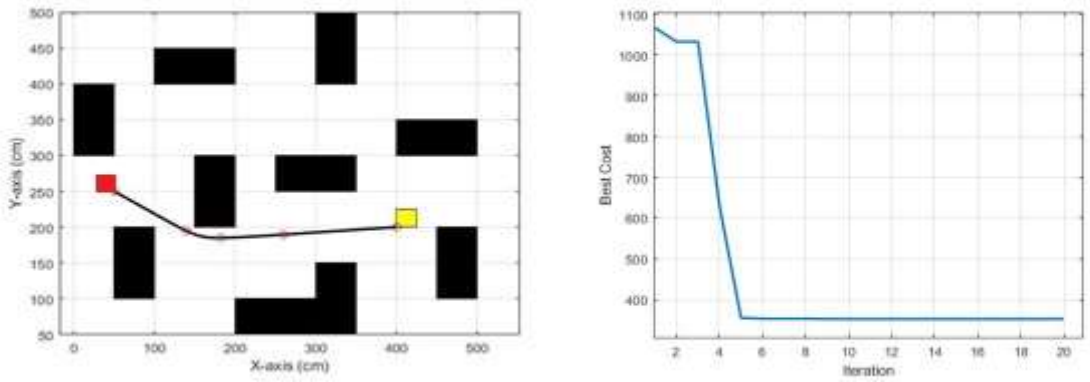
FIG. 6. THE A* PATH PLANNING ALGORITHM CASE1.

Received 19/2/2021; Accepted 29/4/2021

 ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

DOI: https://doi.org/10.33103/uot.ijccce.21.2.4

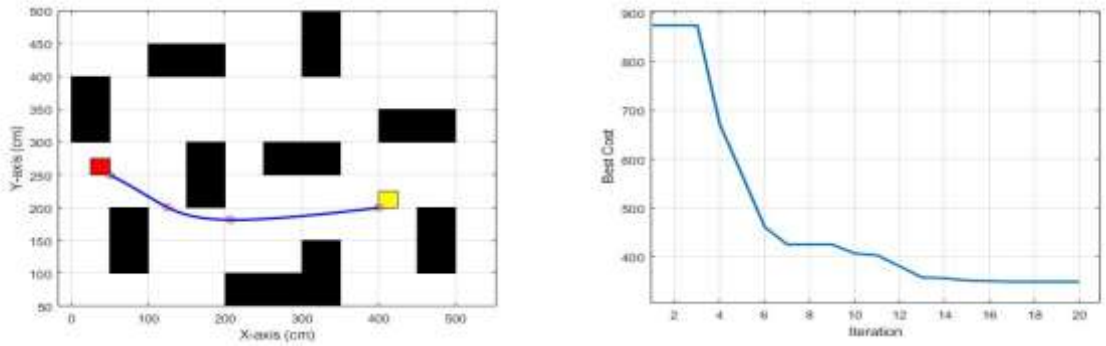
Secondly, applying CPSO algorithm to find the shortest path in any known environments and using the number of iterations is 30 as shown in *Fig. 7-a*. Then the cost distance function based on CPSO algorithm is obtained 367 cm as shown in *Fig. 7-b*.



a b

FIG. 7. THE CPSO PATH PLANNING ALGORITHM CASE 1.

Thirdly, applying the proposed hybrid ACPSO algorithm to find the shortest path in any known environments with the number of iterations is equal to 15 as shown in *Fig. 8-a*. The value of the proposed hybrid ACPSO cost distance function is equal to 357 cm as shown in *Fig. 8-b*.



a b

FIG. 8. THE HYBRID ACPSO PATH PLANNING ALGORITHM CASE1.

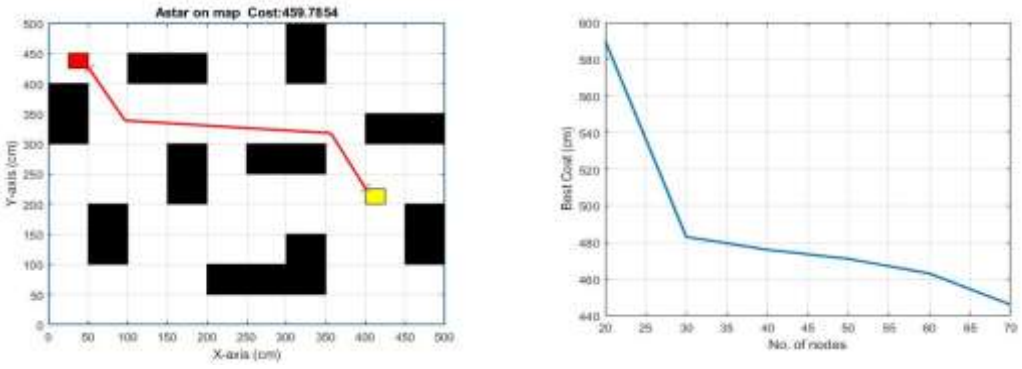
Case 2:

The initial position of the mobile robot at 50*425 cm (red point) to the destination point 400*225 cm (yellow point) as shown in *Fig. 9-a*. Applying A* algorithm and the value of the cost distance function is 459 cm, as shows in *Fig. 9-b*.

Received 19/2/2021; Accepted 29/4/2021

 ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

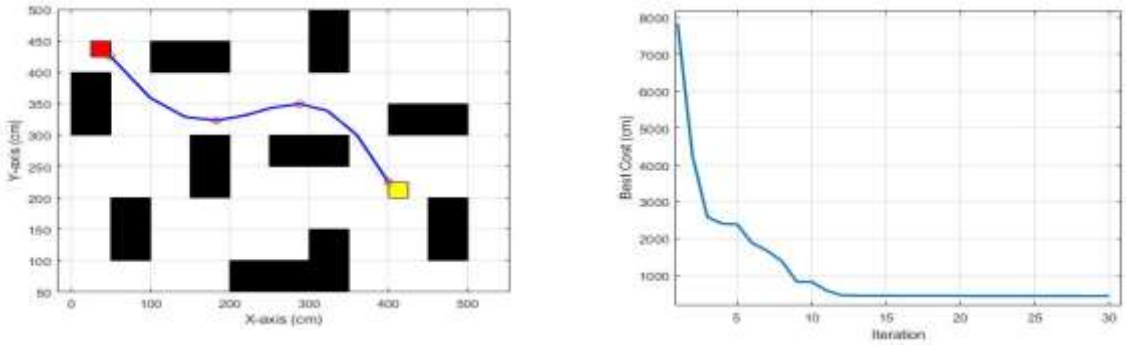
DOI: https://doi.org/10.33103/uot.ijccce.21.2.4



a b

FIG. 9. THE A* ALGORITHM PATH PLANNING CASE 2.

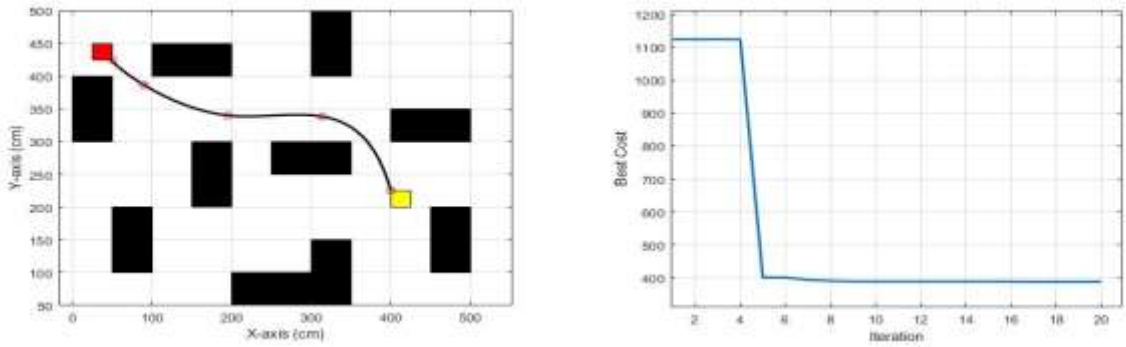
Secondly, *Fig. 10-a* shows the path planning of the mobile robot after applying CPSO algorithm. We found CPSO cost distance function is equal to 443 cm, as shown in *Fig. 10-b.*



a b

FIG. 10. THE CPSO ALGORITHM PATH PLANNING CASE 2.

Finally, applying hybrid ACPSO algorithm to the same environment in case 2, the path planning is generated as shown in *Fig. 11-a* with cost distance function is equal to 387 cm, as shown in *Fig. 11-b.*



a b

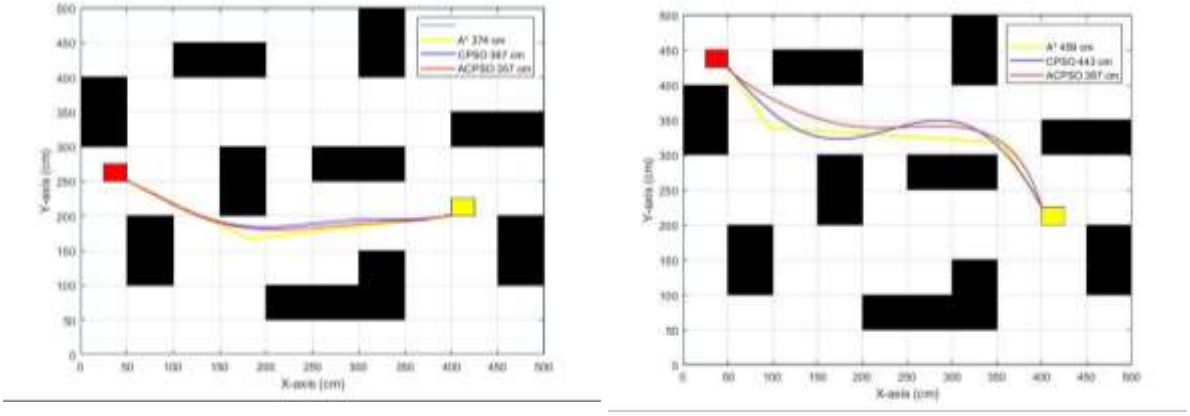
FIG. 11. THE HYBRID ACPSO ALGORITHM PATH PLANNING CASE 2.

Received 19/2/2021; Accepted 29/4/2021

 ISSN (Print) 1811-9212 *ISSN (Online) 2617-3352*

DOI: https://doi.org/10.33103/uot.ijccce.21.2.4

In order to invistigate the effectiveness of the hybrid swarm algorithm, the three algorithms (A*, CPSO and ACPSO) are applied to the environment case 1 and case 2 as shown in the *Fig. 12-a, b* respectivitly. The path planning that is generated from the proposed hybrid algorithm was smooth and the shortest path from strating point to target point when compared with A* and CPSO algorthims.



a b

FIG. 12. ALL ALGORITHMS RESULT PATH PLANNING.

In addition, when comparing between A*, CPSO and ACPSO algorithms as in the Tables II, III and IV respectivtly, in terms of the number of nodes, the number of the iteration, cost distance function and the execution time, the hybrid ACPSO algorithm is much better than A* and CPSO Algorithms for the two cases.

TABLE II: A* ALGORITHM RESULTS.

No. of Case	No. of Nodes	Nodes of the Route	Cost (cm)	Spent time (sec)
Case 1 **Start (50,250)** **End (400,200)**	**20**	4	498	0.956
	30	4	470	1.359
	40	4	436	1.794
	50	3	397	3.951
	60	3	384	5.578
	70	**3**	**374**	7.782
Case 2 **Start (50,425)** **End (400,225)**	**20**	5	590	1.453
	30	4	483	1.879
	40	4	476	3.458
	50	4	471	5.781
	60	4	463	6.673
	70	**4**	**459**	8.256

Received 19/2/2021; Accepted 29/4/2021

 ISSN (Print) 1811-9212 *ISSN (Online) 2617-3352*

DOI: https://doi.org/10.33103/uot.ijccce.21.2.4

TABLE III: CPSO ALGORITHM RESULTS.

No. of Case	Iteration	No. of particles	Cost (cm)	Spent time (sec)
Case 1 Start (50,250) End (400,200)	**5**	20	495	1.502282
	10	20	453	2.608352
	15	20	377	3.775457
	20	20	373	5.070203
	25	20	370	6.272335
	30	20	**367**	7.377480
Case 2 Start (50,425) End (400,225)	**5**	20	2504	1.482984
	10	20	977	2.616973
	15	20	847	3.790400
	20	20	760	4.911652
	25	20	550	6.335959
	30	20	**443**	7.82001

TABLE IV: ACPSO ALGORITHM RESULTS.

No. of Case	Iteration	Nodes of the Route	No. of particles	Cost (cm)	Spent time (sec)
Case 1 Start (50,250) End (400,200)	**5**	3	20	470	2.747106
	10	3	20	406	3.829279
	15	3	20	**357**	4.985177
	20	**3**	20	**357**	-
	25	**3**	20	**357**	-
	30	**3**	20	**357**	-
Case 2 Start (50,425) End (400,225)	**5**	4	20	530	2.364813
	10	4	20	420	3.516424
	15	4	20	389	5.050381
	20	**4**	20	**387**	6.230725
	25	**4**	20	**387**	-
	30	**4**	20	**387**	-

Based on the fitting function, the reference path equation is obtained as in Equation (13) for the case 1 as the optimal path that depends on the hybrid ACPSO algorithm.

$$\mathrm{y(x)} = 2.0857\mathrm{e}-10 \times x^5 - 2.3731\mathrm{e}-07 \times x^4 + 9.5165\mathrm{e}-05 \times x^3 - 0.01435 \times x^2 + 0.19859 \times \mathrm{x} + 266.410 \quad (13)$$

In order to find the reference linear velocity v_r and the reference angular velocity w_r of the platform mobile robot, these equations will bw used as in (14) and (15) [28] depending on the reference path equation.

$$v_r = \sqrt{(\dot{x}_r)^2 + (\dot{y}_r)^2} \quad (14)$$

$$w_r = \frac{\ddot{y}_r \dot{x}_r - \ddot{x}_r \dot{y}_r}{(\dot{x}_r)^2 + (\dot{y}_r)^2} \quad (15)$$

So, the right and left wheels linear velocities V_R , V_L respectively and right and left wheels angular velocities W_R, W_L respectively can be determined as follows [29]:

$$V_R = 0.5(2v_r + Lw_r) \quad (16)$$

$$V_L = 0.5(2v_r - Lw_r) \quad (17)$$

$$W_L = 0.5(2v_r - Lw_r)/r \quad (18)$$

$$W_R = 0.5(2v_r + Lw_r)/r \quad (19)$$

Received 19/2/2021; Accepted 29/4/2021

 ISSN (Print) 1811-9212 ISSN (Online) 2617-3352

DOI: https://doi.org/10.33103/uot.ijccce.21.2.4

Where, r is denoted the radius of the wheel of the mobile robot platform.

Fig. 13-a shows the optimal hybrid ACPSO path for case 1, *Fig. 13-b* shows the reference linear and reference angular velocities. The right and left wheels linear velocities are shown in *Fig. 13-c* that have smooth responses without sharp spikes. *Fig. 13-d* shows the response of the right and left wheels angular velocities.

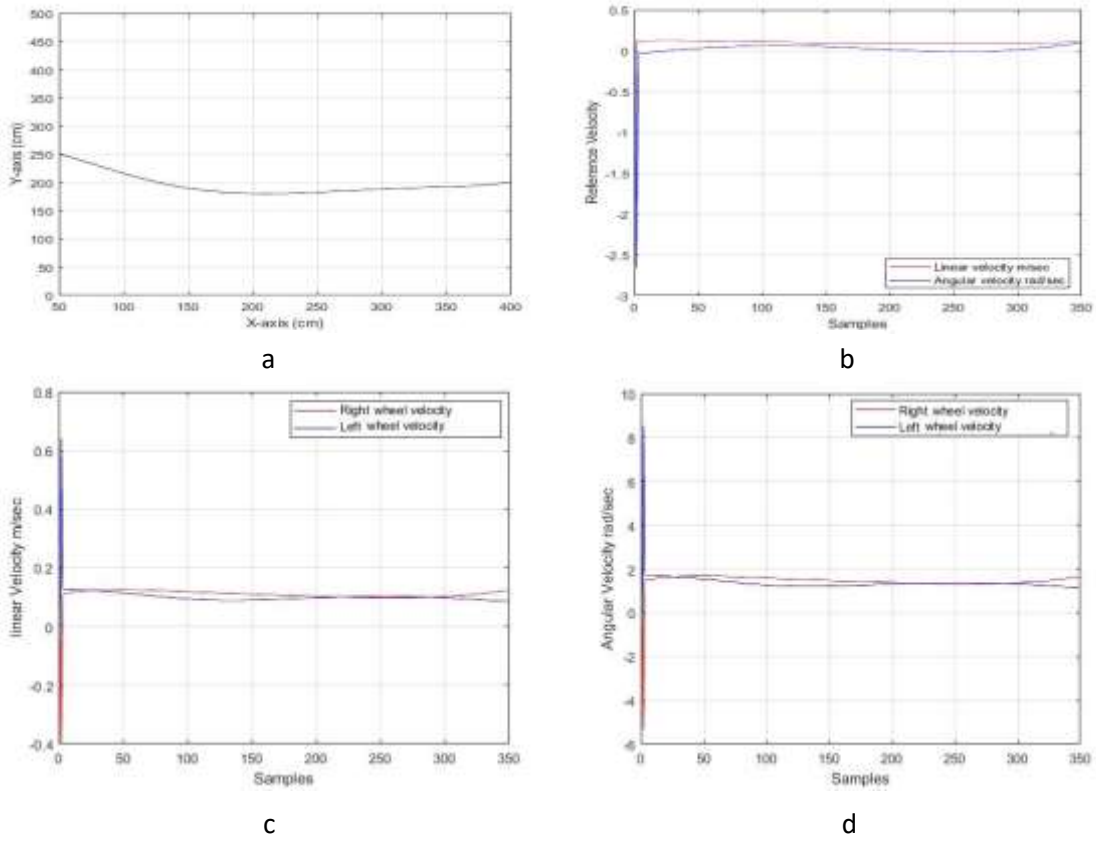


FIG. 13. CASE 1: A) THE OPTIMAL PATH, B) THE REFERENCE LINEAR VELOCITY AND REFERENCE ANGULAR VELOCITY, C) THE RIGHT AND LEFT WHEELS LINEAR VELOCITIES, D) THE RIGHT AND LEFT WHEELS ANGULAR VELOCITIES.

For case 2, the reference path equation is obtained as follows:

$$y(x) = -3.1151e-10 \times x^5 + 2.6975e-07 \times x^4 - 9.2761e-05 \times x^3 + 0.018262 \times x^2 - 2.4557 \times x + 510.79 \quad (20)$$

Fig. 14-a shows the optimal hybrid ACPSO path for case 2, *Fig. 14-b* shows the reference linear and reference angular velocities. The right and left wheels linear velocities are shown in *Fig. 14-c* that have smooth responses without sharp spikes. *Fig. 14-d* shows the response of the right and left wheels angular velocities.

Received 19/2/2021; Accepted 29/4/2021

 ISSN (Print) 1811-9212 *ISSN (Online) 2617-3352*

DOI: https://doi.org/10.33103/uot.ijccce.21.2.4

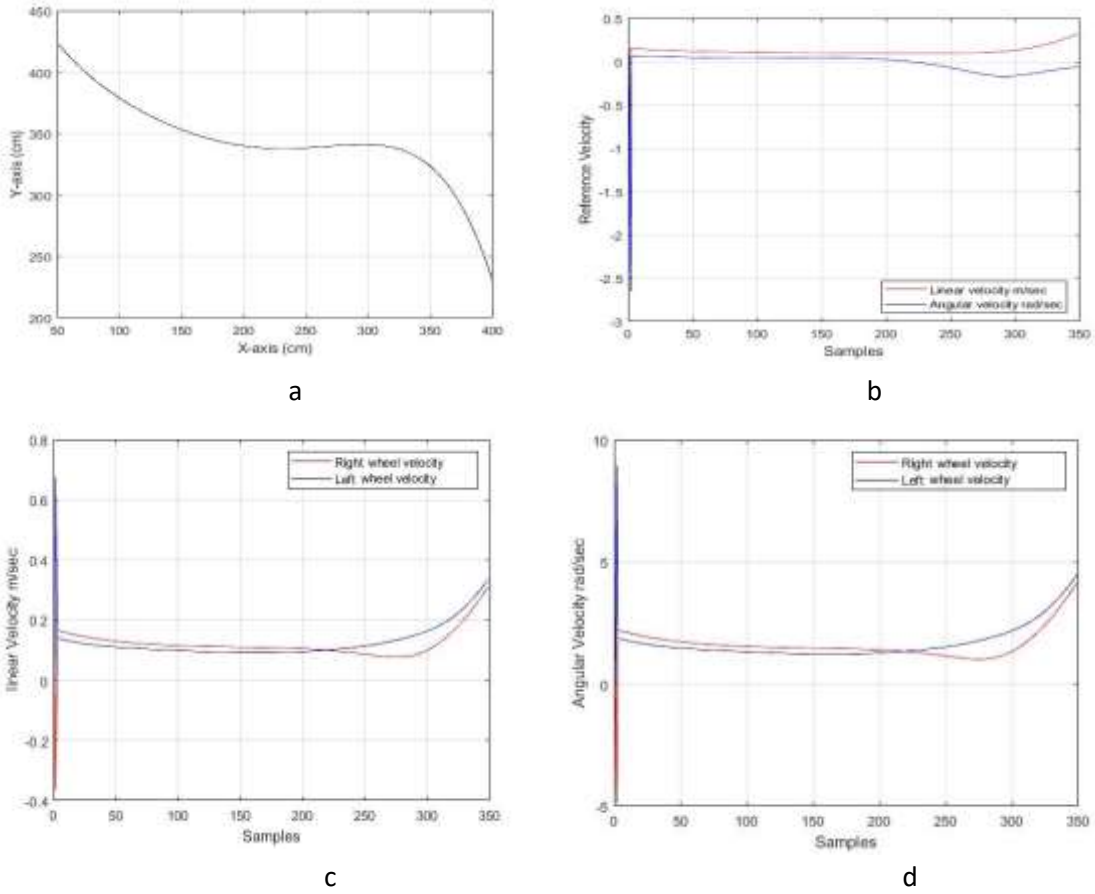


FIG. 14. CASE 2: A) THE OPTIMAL PATH, B) THE REFERENCE LINEAR VELOCITY AND REFERENCE ANGULAR VELOCITY, C) THE RIGHT AND LEFT WHEELS LINEAR VELOCITIES, D) THE RIGHT AND LEFT WHEELS ANGULAR VELOCITIES.

V. CONCLUSIONS

In this paper, a hybrid swarm ACPSO algorithm has been proposed and simulated for path planning of the mobile robot using MATLAB package. The proposed path planning algorithm is based on A* and CPSO algorithms. Therefore, the proposed hybrid swarm algorithm has excellent ability in term of the following:

- Minimizing the number of nodes that are used.
- Reducing the number of the iterations and the evaluation function.
- The value of the cost distance function is less than A* and CPSO Algorithms for the two cases.
- The execution time of the processor unit is reduced.
- Optimal or near optimal smooth path is generated.
- The best response of the reference linear and angular velocities of the mobile robot are obtainned.

REFERENCES

[1] M. S. Alam, M. U. Rafique, and M. U. Khan, "Autonomous Robot Path Planning Using Particle Swarm Optimization in Static and Obstacle Environment," International Journal of Computer Science and Electronics Engineering (IJCSEE) vol 3, No. 3, pp. 253-257, 2015.

[2] K. Kumar and A. K. Dewangan, "Survey Paper on Robotic Path Planning Algorithms," IJSTE - International Journal of Science Technology & Engineering, vol. 2, No. 12, pp.1-4, 2016.

[3] S. K. Malu, and J. Majumdar, "Kinematics, Localization and Control of Differential Drive Mobile Robot," Global Journal of Researches in Engineering: Robotics & Nano-Tech, vol. 14, No.1, 2014.

[4] M. Asif, M. J. Khan, M. Rehan, and M. Safwan," Feedforward and Feedback Kinematics Controller for Wheeled Mobile robot Trajectory Tracking," Journal of Automation and Control Engineering, vol. 3, No.3, pp.178–182, 2015.

Received 19/2/2021; Accepted 29/4/2021

 ISSN (Print) 1811-9212 *ISSN (Online) 2617-3352*

DOI: https://doi.org/10.33103/uot.ijccce.21.2.4

[5] T. Mac, C. Copot, T. T. Duc., and R. D. Keyser, "Heuristic Approaches in Robot Path Planning: A Survey," Robotics and Autonomous Systems, vol. 86, pp. 13–28, 2016.

[6] N. A. Daniel, A. A. Gallegos, C. Lopez-Franco, and A. Y. Alanis, "Smooth Path Planning for a Mobile Robot using Particle Swarm Optimization and Radial Basis Functions, Splines, and Bezier Curves," IEEE Congress on Evolutionary Computation, pp. 175-182, 2014.

[7] R. k. Panda and B.B. Choudhury, "An Effective Path Planning of Mobile Robot using Genetic Algorithm," Proceeding of the IEEE International Conference on Computational Intelligence & Communication Technology, pp. 287-291, 2015.

[8] M. A. Contreras-Cruz, V. Ayala-Ramirez, and U. H. Hernandez-Belmonte, "Mobile Robot Path Planning using Artificial Bee Colony and Evolutionary Programming," Applied Soft Computing, vol. 30, pp. 319-328, 2015.

[9] M. Duguleana and G. Mogan, ''Neural Networks-Based Reinforcement Learning for Mobile Robot's Obstacle Avoidance,'' Expert Systems with Applications, vol. 62, pp. 104–115, Nov. 2016.

[10] S. Ghosh, P. P. Kumar, and D. R. Parhi, ''Performance Comparison of Novel WNN Approach with RBFNN in the Navigation of Autonomous Mobile Robotic Agent,'' Serbian Journal of Electrical Engineering, vol. 13, No. 2, pp. 239–263, 2016.

[11] B. Tang, Z. Zhanxia, and J. Luo, ''A Convergence-Guaranteed Particle Swarm Optimization Method for Mobile Robot Global Path Planning,'' Assembly Automation. vol. 37, No. 1, pp. 114–129, 2017

[12] J. Liu, J. Yang, H. Liu, X. Tian, and M. Gao, ''An Improved Ant Colony Algorithm for Robot Path Planning,'' Soft Computing., vol. 21, No. 19, pp. 5829–5839, 2017.

[13] A. Hidalgo-Paniagua, M. A. Vega-Rodríguez, J. Ferruz, and N. Pavón, ''Solving the Multi-Objective Path Planning Problem in Mobile Robotics with A Firefly-Based Approach,'' Soft Computing, vol. 21, No. 4, pp. 949–964, 2017.

[14] F. Arvin, K.Samsudin and M. A. Nasseri, "Design of Differential-Drive Wheeled Robot Controller with Pulse-Width Modulation," Conference on Innovative Technologies in Intelligent Systems and Industrial Applications, pp. 143-147, 25th & 26th July 2009.

[15] A. Al-Araji, "Design of On-Line Nonlinear Kinematic Trajectory Tracking Controller for Mobile Robot based on Optimal Back-Stepping Technique," Iraqi Journal of Computers, Communication and Control & Systems Engineering, vol. 14, No. 2, pp. 25-36, 2014.

[16] S.M. Persson, I. Sharf, "Sampling-Based A* Algorithm for Robot Path-Planning," International Journal of Robotics Research, vol. 33(13), pp. 1683-1708, 2014.

[17] S. Broumi, A. Bakali, M. Talea, F. Smarandache, and L. Vladareanu, "Applying the Dijkstra Algorithm for Solving Neutrosophic Shortest Path Problem," In International Conference on Advanced Mechatronic Systems (ICAMechS), IEEE, pp. 412-416, November 2016.

[18] F. Duchoň, A. Babinec, M. Kajan, P. Beňo, M. Florek, T. Fico, and L. Jurišica, "Path Planning with Modified A Star Algorithm for A Mobile Robot," Procedia Engineering, vol 96, pp. 59-69, 2014.

[19] Z. Zhang and Z. Zhao, "A Multiple Mobile Robots Path Planning Algorithm Based on A-Star and Dijkstra Algorithm," International Journal of Smart Home, vol 8, No 3, pp. 75-86, 2014.

[20] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," Proceedings of ICNN'95 - International Conference on Neural Networks, pp.1942-1948, 1995.

[21] A. Khatami, S. Mirghasemi, A. Khosravi, C. P. Lim, and S. Nahavandi, "A New PSO-Based Approach to Fire Flame Detection Using K-Medoids Clustering," Expert Systems with Applications, vol. 68, pp. 69–80, 2017.

[22] C.W. Lin, L. Yang, P. Fournier-Viger, T. P. Hong, and M. Voznak, "A Binary PSO Approach to Mine High-Utility Itemset," Soft Computing, vol. 21, pp. 5103–5121, 2017.

[23] Y. Zhou, N. Wang, and W. Xiang, "Clustering Hierarchy Protocol in Wireless Sensor Networks Using an Improved PSO Algorithm," IEEE Access, vol. 5, pp. 2241–2253, 2017.

[24] N. Chouikhi, B. Ammar, N. Rokbani, and A. M. Alimi, "PSO-Based Analysis of Echo State Network Parameters for Time Series Forecasting," Applied Soft Computing, vol. 55, pp. 211–225, 2017.

[25] B. Li and W. S. Jiang, "Optimizing Complex Functions by Chaos Search", Cybernetics and Systems., vol. 29, no. 4, pp. 409–419, 1998.

[26] B. Liu, L.Wang, and Y. H. Jin, "Improved Particle Swarm Optimization Combined With Chaos," Chaos, Solitons Fractals, vol. 25, no. 5, pp. 1261–1271, 2005.

[27] E. A. Jaber, A. S. Al-Araji, and H. A. Dhahad, "Predictive Nonlinear PID Neural Voltage-Tracking Controller Design for Fuel Cell based on Optimization Algorithm," Iraqi Journal of Computers, Communications and Control & Systems Engineering, vol. 19, No. 4, pp. 47-60, 2019.

[28] A. S. Al-Araji and K. E. Dagher, "Cognitive Neural Controller for Mobile Robot," Iraqi Journal of Computers, Communication and Control & Systems Engineering, vol. 15, No. 1, pp. 46-60, 2015.

[29] A. S. Al-Araji, M. F. Abbod, and H. S. Al-Raweshidy, "Design of A Neural Predictive Controller For Nonholonomic Mobile Robot Based On Posture Identifier," Proceedings of the IASTED International Conference Intelligent Systems and Control (ISC 2011), Cambridge, United Kingdom, pp. 198-207, 2011.

Received 19/2/2021; Accepted 29/4/2021

 ISSN (Print) 1811-9212 *ISSN (Online) 2617-3352*