



# An intelligent optimize path planner for efficient mobile robot path planning in a complex terrain

Sunil Kumar<sup>1</sup> · Afzal Sikander<sup>1</sup>

Received: 14 March 2022 / Accepted: 19 May 2022 / Published online: 18 June 2022  
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2022

## Abstract

Path planning is an essential necessity for the proper functioning of mobile robot in a complex terrain. Conventional approaches face different challenges such as balancing exploration and exploitation ability, premature convergence, weak searching ability, and longer path length. To overcome these flaws, an Intelligent Modified Particle Swarm Optimization approach with a different strategy is proposed. Firstly, a velocity regularized strategy based on regularized coefficients has been applied to balance the exploration and exploitation ability. Secondly, a neighborhood search strategy based on reward value and utilization probability has been employed, enriching search behaviors and avoiding premature convergence. Finally, a path smoothness principle based on hypocycloid curves has been used to smooth the sharp turns. The comparative analysis conducted in four different terrains with different complexity. Different performance indices are being measured to validate the effectiveness of the proposed approach. The outcome acquired from different terrains indicates that the proposed approach outperforms the GA-PSO, Advance PSO, FACO, and other conventional approaches with a maximum improvement (%) of 17.59% in path length and 76.66% in convergence rate.

## 1 Introduction and related work

Mobile robots are utilized in a broad range of applications including the industrial sector, nuclear power plants, space research, underwater investigation, domestic purpose, medical support etc., therefore regarded as smart operating tools (He et al. 2019; Patle et al. 2019; Liao et al. 2021). For a robot to operate effectively, path planning is one of the essential control parameters. Path planning may be considered as an optimization issue whose aim is to identify a path from the initial location to the destination through numerous waypoints in a cluttered environment without colliding with the obstacles (Dolgov et al. 2008). It is divided into two categories such as global path planning and local path planning. In global path planning, the mobile robot must have previous knowledge of the environment such as obstacle location, and target location. On the other hand,

local path planning does not need prior knowledge of the surroundings. Global path planning methodology works in an entirely known environment. Whereas, local path planning operates in a partially known or unknown environment. Each of the aforementioned scenarios require distinct path planning approaches.

Path planning research emerged in the late 1960s, and different approaches have been introduced including dynamic programming (DP) (Cesarone and Eman 1989), A\* approach (Loong et al. 2011), Dijkstra approach (DA) (Wang et al. 2011), roadmap approaches (Clark 2005), and potential field (Xu and Park 2020). The main disadvantages of these approaches are smoothness, local minima issue and high computational costs. Researchers have always been looking for alternative strategies and more effective methods to solve these issues. Various meta-heuristic approaches have been used in previous studies to solve these challenges. These approaches include particle swarm optimization (PSO) (Huang and Tsai 2011), genetic algorithm (GA) (Arora et al. 2014), cuckoo search approach (CS) (Mohanty and Parhi 2016), sparrow search algorithm (Zhang et al. 2021), bacterial foraging optimization (BFO) (Hossain and Ferdous 2015), artificial bee colony (ABC) (Kumar and Sikander 2022), and ant colony optimization (ACO) (Yen and Cheng 2018). In contrast to other

✉ Sunil Kumar  
sunnybodla@gmail.com

Afzal Sikander  
afzals@nitj.ac.in

<sup>1</sup> Department of Instrumentation and Control Engineering, Dr B R Ambedkar NIT Jalandhar, Jalandhar, Punjab 144011, India

approaches, each approach has its own pros and limitations. These algorithms have several advantages such as the capacity to do a global search, need for fewer tuning parameters, the ability to solve unconstrained problems in a short amount of time, and so on. In recent years, some new intelligence path planning algorithms have performed excellently such as DDM (Diversified-path Database driven Multi-robot Path planning) algorithm (Han and Yu 2020), MOD-*RRT\** (Multiobjective dynamic rapidly exploring tree) (Qi et al. 2021) and Self-Adaptive Harmony Search Algorithm (Quan et al. 2021). Han and Yu (2020) introduced a unique DDM algorithm with two alternative strategies, the first of which is to move a collection of robots from their beginning vertices to target vertices as fast as possible, and the second of which is to demand frequent re-planning to account for target configuration changes. This algorithm is not suitable for path planning in narrow passages due to its concise mechanism. Also, Qi et al. (2021) proposed a MOD-*RRT\** algorithm that selects the best node among several nodes by considering the path length and path smoothness. This algorithm suffers from the limitation of longer path length due to the random sampling approach. Similarly, Quan et al. (2021) developed an improved version of the harmony search algorithm to improve the neighbours, self-adaptive and probability disturbance updating strategies. Although this algorithm has better search accuracy, its disadvantages are randomness and instability.

However, many prior literature appreciated the use of Particle swarm optimization (PSO) to solve the path planning issues. A lot of research has been done on PSO due to its concise mechanism and few control parameters. Still, there are some challenging issues such as easily trapped in local minima and lacking guarantee in optimal solutions. To tackle the issues outlined above, numerous updated variants of PSO have been developed to produce a feasible path. A mutation-based technique has been utilized by Gong et al. (2011) to repair the incorrect paths by using the self-adaptive mutation operation. Zhang et al. (2013) developed a multi-objective PSO that depends on a random sampling of particles for mobile robot path planning in the test area with unpredictable danger sources. Also, Deepak et al. (2014) presented a fitness function-based approach that mainly depends on the distance of each particle to the target location. Another recent study on this topic is presented by Li and Chou (2018) and Zhang et al. (2020). Li and Chou (2018) introduced a self-adaptive learning mechanism using different learning strategies and a new bound violation handling scheme to improve the feasibility of the acquired paths. Similarly, Zhang et al. (2020) developed an improved localized PSO that uses the extended Gaussian distribution, inertia weights, and interpolation-based path smoothness principle to enhance the search ability.

The aforementioned researchers conducted a number of experiments to prove the effectiveness of the suggested approach, although there are still some shortcomings:

- Most studies employ a random sampling strategy. The random sampling strategy does not ensure to provide the global optimal solution every time (Clark 2005; Zhang et al. 2013; Deepak et al. 2014). This will increase the path length.
- In the case of high dimensional and complex terrains, the existing literature faces various learning ability issues such as slow convergence rate, weak adaptability and weak exploration ability (Huang and Tsai 2011; Deepak et al. 2014; Yen and Cheng 2018).
- Past study uses the inertia weight factor strategy. Which increases the computational load. Therefore, the processing time of the acquired path increases (Li and Chou 2018; Zhang et al. 2020).
- Most algorithms do not vary the number of particles or samples or populations while calculating the performance indices such as path length and processing time, as variation in samples plays an important role to authenticate the effectiveness of the proposed approach (Huang and Tsai 2011; Deepak et al. 2014; Mohanty and Parhi 2016; Yen and Cheng 2018).

To address these limitations, establishing an intelligent optimum path planning approach in complex terrain would be effective and beneficial, which is the primary goal of this research study. Therefore, this research study presents an intelligent modified particle swarm optimization (IMPSO) approach for resolving mobile robot path planning issues in a complex terrain. The innovations and key contributions of this research work are outlined as follows.

- A velocity regularized strategy based on regularized coefficients has been applied to balance the exploration and exploitation ability.
- A neighborhood search strategy based on reward value and utilization probability has been employed, enriching search behaviors and avoiding premature convergence.
- A path smoothness principle based on hypocycloid curves has been used to smooth the sharp turns.
- The proposed approach is validated and analyzed against the GA-PSO (Huang and Tsai 2011), Advance PSO (Deepak et al. 2014), FACO (Yen and Cheng 2018), and other conventional approaches in terms of different performance indices.
- The comparative studies are conducted in four different small and large dimension terrain with different complexity.
- The performance and effectiveness of the proposed IMPSO are validated using simulation outcomes.

The rest of the research work is outlined in the following ways. The problem formulation and performance indices are presented in Sect. 2. The proposed methodologies for mobile robot path planning are introduced in Sect. 3. In Sect. 4, a set of simulation findings are presented that demonstrate the proposed technique’s efficacy compared to prior research. Finally, the conclusion of the research study is presented in Sect. 5.

## 2 Problem formulation and performance indices

A model should firstly be constructed in order to develop an efficient trajectory path for a mobile robot. Figure 1 shows a two-dimensional corridor like terrains (terrain #1, terrain #2, terrain #3, and terrain #4) with different dimensions. These terrains comprise all entities such as

obstacles, mobile robot, number of samples or population size, initial and target locations with different shapes and sizes. The mobile robot is represented as  $R$  (with radius  $r_{ROB}$ ) having its coordinate  $C(x_0, y_0)$ .

A green circular object signifies the initial location, which is denoted by  $S(x_0, y_0)$ . Whereas, a red circular object signifies the target location denoted by  $T(x_t, y_t)$ . The mobile robot at the initial location uses a number of samples to find the next best current location. Within the sensor detection range, generate a map that joins the initial and target location. To execute the task of predicting the behaviour and location of obstacles, the sensor range of the mobile robot is divided into four circles having radii  $\mu_1, \mu_2, \mu_3,$  and  $\mu_4$  as presented in Fig. 2. The range of sensors depends upon its types. There are various types of sensors used in mobile robots such as infrared, ultrasonic, tactile, GPS, etc. The goal of path planning is to discover an effective path without colliding with any obstacle’s

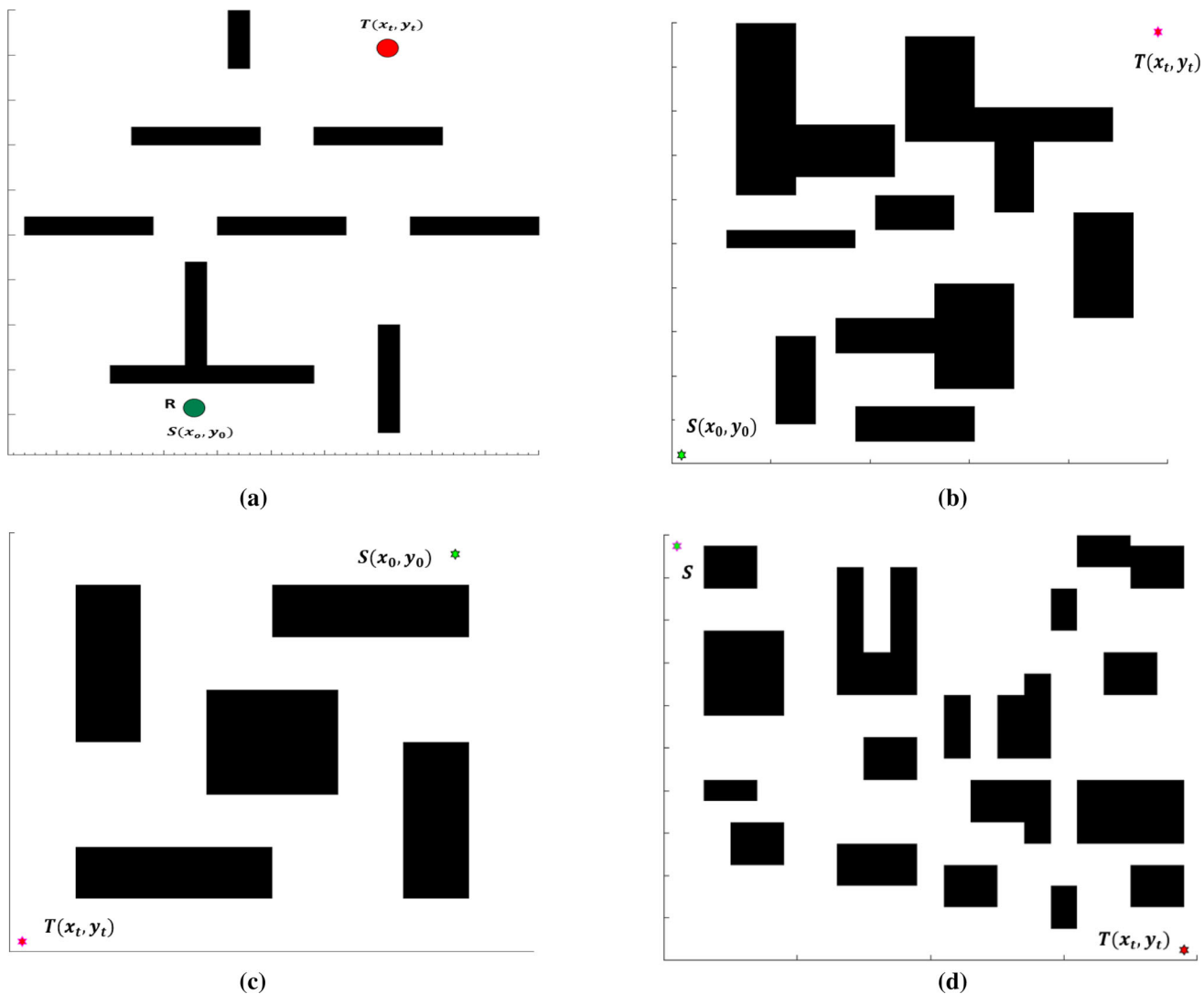


Fig. 1 Model of different terrains a terrain #1; b terrain #2; c terrain #3; d terrain #4

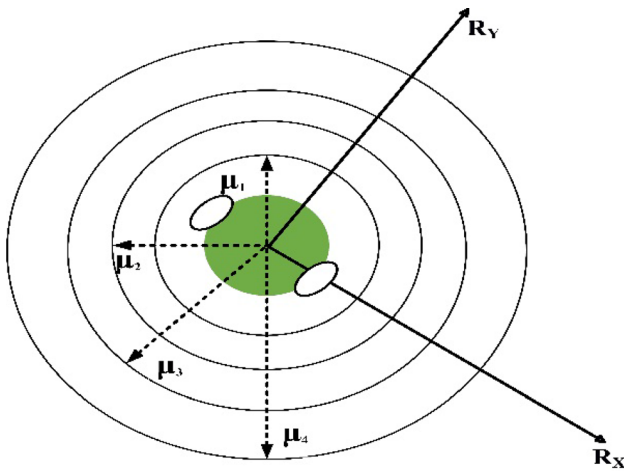


Fig. 2 Sensor range with regards to obstacles location

regions. If the robot encounters any obstacles in its path, it must move some angles, either left or right, without colliding to reach the desired target location. In order to solve the path planning problem, certain assumptions are taken in this research study.

**Assumption 1.** The obstacles are depicted as rectangular shapes with static in nature.

**Assumption 2.** The test scenario is partially known or unknown for the robot.

**Assumption 3.** Robot have some sensor range and it can move in any direction due to its omnidirectional nature.

### 2.1 Performance indices

Path planning tries to construct the best possible collision-free path while taking into consideration specified performance indices. This study focuses on path length, processing time and smoothness.

(a) Path length

The objective is to acquire the shortest path as much as possible. Suppose, the initial and target locations are  $N_{F0}$  and  $N_{F(N+1)}$ , then the path length can be calculated as:

$$P(F) = \sum_{i=0}^N N_{F(i+1)} - N_{F(i)} \tag{1}$$

where  $N_{F(i+1)} - N_{F(i)}$  denotes the Euclidean distance between  $N_{F(i)}$  and  $N_{F(i+1)}$ .

(b) Smoothness

The smoothness is computed by summing the robot's turning angles along the predefined path. The formula below can be used to calculate the smoothness.

$$S(F) = \sum_{i=1}^N \gamma_i = \sum_{i=1}^N \arccos \left( \frac{(N_{F(i)} - N_{F(i-1)}) \cdot (N_{F(i+1)} - N_{F(i)})}{|N_{F(i)} - N_{F(i-1)}| \times |N_{F(i+1)} - N_{F(i)}| \times 180} \right) \tag{2}$$

where  $\gamma_i$  represents the value of the acquired path's  $i$ th turning angle (measured in radians ranging from  $0$  to  $\pi$ ).  $(F_i - F_{i-1}) \cdot (F_{i+1} - F_i)$  are the inner product of two vectors.

(c) Improvement (%)

$$Improvement(\%) = \left( \frac{X_{Conventional} - X_{Proposed}}{X_{Conventional}} \right) \times 100 \tag{3}$$

where X can be any performance indices.

## 3 Proposed Methodology

This section outlines the complete path generation process in two stages. The first stage describes the stranded Particle Swarm Optimization (PSO), while the second stage introduces the proposed approach (Intelligent Modified Particle Swarm optimization). Three modules are included in the proposed IMPSO. A velocity regularized strategy is used in the first module, whereas a neighborhood search strategy is utilized in the second module. Lastly, a path smoothness principle is applied in the third module. Figure 3 represents the proposed methodology.

### 3.1 Standard PSO

James Kennedy and Russell Eberhart introduced Particle Swarm Optimization in 1995. PSO is a population-based stochastic optimization approach that is inspired by the social behavior of flocks of birds. Initially, all of the birds travel at random velocities and positions, but after a period, depending on their own flying experience and that of the other birds, all of the birds begin to follow the bird closest to the food. All birds or particles in PSO have two attributes i.e. position and velocity. These particles are introduced into the search space of a problem or function. The fitness values associated to the fitness function are calculated for each particle, and two best fitness values are obtained; the first is the best fitness value a particle has attained so far known as 'pbest'. The second is the best fitness value attained so far by the whole swarm, which is referred to as 'gbest.' In the d-dimensional search space, the  $i$ th particle's velocity and position can be represented as  $V_i = [v_{i,1}, v_{i,2}, \dots, v_{i,d}]$  and  $X_i = [x_{i,1}, x_{i,2}, \dots, x_{i,d}]$ , respectively. Each particle has its own best position (pbest),  $Pb_i = [p_{i,1}, p_{i,2}, \dots, p_{i,d}]$  which corresponds to the personal best objective value attained so far at time t and the best one among  $Pb_i$  in a group is designated as the global best

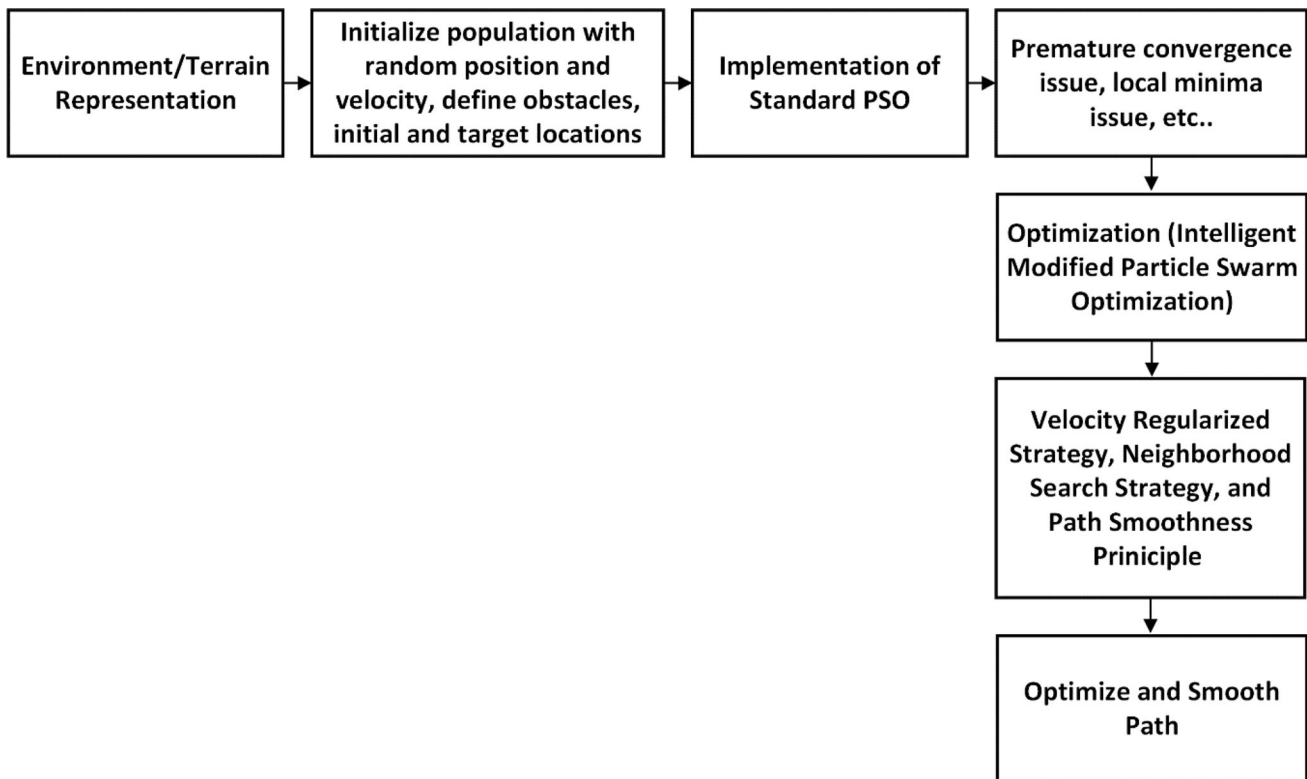


Fig. 3 Proposed Methodology

particle (gbest) denoted by  $P_g$ . In each iteration  $t$ , the velocity and position of each particle is computed as follows

$$v_{i,j}(t + 1) = wv_{i,j}(t) + \eta_1 r_1 (Pb_i - x_{i,j}(t)) + \eta_2 r_2 (P_g - x_{i,j}(t)), j = 1, 2, \dots, d \tag{4}$$

$$x_{i,j}(t + 1) = x_{i,j}(t) + v_{i,j}(t + 1), \tag{5}$$

where  $w$  signifies the inertia weight factor,  $\eta_1$  and  $\eta_2$  represent the personal and global learning coefficients,  $r_1$  and  $r_2$  are two random numbers that lies in the range of 0 to 1. To restrict excessive roaming of particles beyond the search space, the value of each component in  $V_i$  can be clamped to the range  $[-v_{max}, v_{max}]$ . The following are the main procedures of standard PSO.

Step 1. Generate a population of particles with random velocities and positions, each having  $d$  variables.

Step 2. Compute the objective values of all particles; set pbest and its objective value equal to the particle’s current location and objective value, and set gbest and its objective value equal to the location and objective value of the best starting particle.

Step 3. Update the velocity and location of each particle by using Eqs. (4) and (5).

Step 4. Compute each particle’s objective values.

Step 5. Compare the current objective value of each particle to the pbest objective value. If the current value is

superior, then replace pbest and its objective value with the current position and objective value.

Step 6. Find the best particle in the current swarm that has the best objective value. If the current best particle’s objective value is higher than gbest, replace gbest and its objective value with the current best particle’s position and objective value.

Step 7. If a stopping requirement is fulfilled, generally satisfying the fitness value; otherwise, return to Step 3.

The standard PSO has the advantage of having parameters that are easy to adjust and implement. However, some drawbacks still exist in standard PSO, such as improper balance between exploration and exploitation ability, poor searching behaviour, premature convergence, and being easily stuck into local minima.

### 3.2 Intelligent Modified Particle Swarm Optimization (IMPSO)

To address the drawbacks of standard PSO, a new approach is introduced, the intelligent modified particle swarm optimization. The IMPSO mainly depends on the velocity regularized strategy, neighborhood search strategy, and path smoothness principle. The velocity regularized strategy is based on regularized coefficients and is used to balance the exploration and exploitation ability. A



neighborhood search strategy based on reward value and utilization probability has been employed enriching search behaviors and avoiding premature convergence. A path smoothness principle based on hypocycloid curves has been used to smooth the sharp turns.

### 3.2.1 Velocity regularized strategy (VRS)

Any optimization technique necessitates a trade-off between exploration and exploitation to determine its efficacy. These contradictory features are ideally balanced by an efficient optimization strategy. In standard PSO, velocities of the particles can readily climb to unacceptable levels and sometimes exceed the bounds of the search region within a few iterations. Therefore, a velocity regularized strategy is applied by introducing regularized coefficients to govern the velocities of the particles as well as to keep inside the boundary constraints. The coefficient governs particle movement and leads it toward convergence. The modified particle velocities can be expressed in the following way:

$$v_{\max(i,j)} = w_{RF}v_{i,j}(t) + C_1r_1(Pb_i - x_{\max(i,j)}(t)) + C_2r_2(P_g - x_{\max(i,j)}(t)), \quad (6)$$

$$v_{\min(i,j)} = w_{RF}v_{i,j}(t) + C_1r_1(Pb_i - x_{\min(i,j)}(t)) + C_2r_2(P_g - x_{\min(i,j)}(t)), \quad (7)$$

$$v_{i,j}(t+1) = \begin{cases} v_{i,j}(t+1), & \text{if } v_{i,j}(t+1) < v_{\max(i,j)} \\ v_{\max(i,j)}, & \text{otherwise} \end{cases} \quad (8)$$

where  $x_{\min(i,j)}$  and  $x_{\max(i,j)}$  are the minimum and maximum positions of the particles. Here higher value of  $v_{\max(i,j)}$  encourages exploration whereas lower value encourages exploitation. Also,  $w_{RF} = w * RF$ , and  $C_1, C_2$  signifies the cognitive and social components that impact exploration and exploitation ability as well as the finding of an optimal point in the search area. The  $C_1$  and  $C_2$  can be computed as follows:

$$C_1 = RF * \eta_1 \quad (9)$$

$$C_2 = RF * \eta_2 \quad (10)$$

where

$$RF = \frac{2}{\eta - 2 + \sqrt{\eta^2 - 4\eta}} \quad (11)$$

$$\eta = \eta_1 + \eta_2 \quad (12)$$

where  $RF$  and  $\eta$  denotes a regularized factor and co-efficient, respectively. Now the regularized particles can help to converge the process towards the optimal solutions and maintain a proper balance between the exploration and exploitation ability.

### 3.2.2 Neighborhood search strategy

In IMPSO, neighborhood selection is necessary for enriching search behaviors and avoiding premature convergence. In this study, a neighborhood search strategy is used based on the reward value and utilization probability of each particle. The following equation can be used to calculate the reward value for each particle.

$$\beta = \frac{|pf - cf|}{N(N-1)} \quad (13)$$

where  $N$  signifies the total number of particles in the search space,  $pf$  is the function value of each particle whose coordinate is represented as  $(x_i, y_i)$  and  $cf$  is the function value of target particle whose coordinate is denoted as  $(x_t, y_t)$ . After the reward value is obtained, the utilization probability ( $Prob_{uti}$ ) of each particle is calculated with the following formula.

$$Prob_{uti,i} = \frac{\beta_i}{\sum_{i=1}^N \beta_i} \quad (14)$$

where  $\beta_i$  is the  $i$ th particle's reward value, and  $N$  represents the total number of populations or particles. A fit neighborhood will be chosen based on the particle with the highest reward value and utilizing probability. Initially,  $N$  particles are scattered in the search area as shown in Fig. 4a. After applying the proposed strategy, the new neighborhood is  $N_{F1}, N_{F2}, N_{F3}, N_{F4}$ , and  $N_{F5}$ , as depicted in Fig. 4b.

### 3.2.3 Path smoothness principle

Although the attained path avoids obstacles, still there are too many sharp turns that result in longer path length. To avoid these sharp turns, a path smoothness principle is applied based on hypocycloid curves (Weisstein 2016). The basic steps are outlined as follows.

Step 1. It can be seen from Fig. 5a that a robot traversing from initial (S) to target location (T) encounters sharp turns at points  $N_{F1}, N_{F2}, N_{F3}, N_{F4}$ , and  $N_5$ . The complete path from S to T is made up of various line segments labelled  $SN_{F1}, N_{F1}N_{F2}, N_{F2}N_{F3}, N_{F3}N_{F4}, N_{F4}N_{F5}$ , and  $N_{F5}T$ . Each of the six-line segments is represented as

$$y = m_i x + c_i, i \in \{1, 2, \dots, 13\} \quad (15)$$

where  $m_i$  and  $c_i$  denotes the slopes and the intercepts of the thirteen lines.

Step 2. With respect to line segment  $SN_{F1}$  and  $N_{F1}N_{F2}$ , let  $(x_0, y_0), (x_1, y_1)$ , and  $(x_2, y_2)$  be the coordinates of points S,  $N_{F1}$ , and  $N_{F2}$ . In Fig. 5a, the midpoints of line segments  $SN_{F1}$  and  $N_{F1}N_{F2}$  are  $G(\frac{x_0+x_1}{2}, \frac{y_0+y_1}{2})$  and

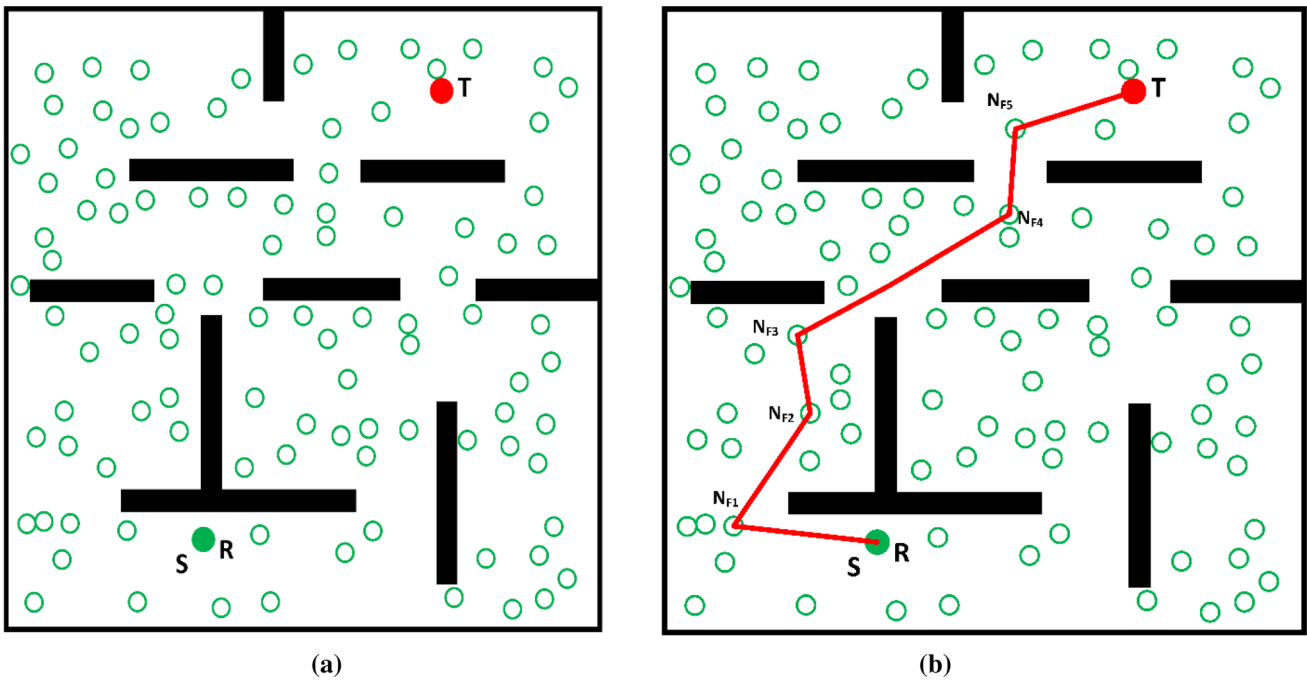


Fig. 4 a Model of the search area, b path attained using proposed neighborhood strategy

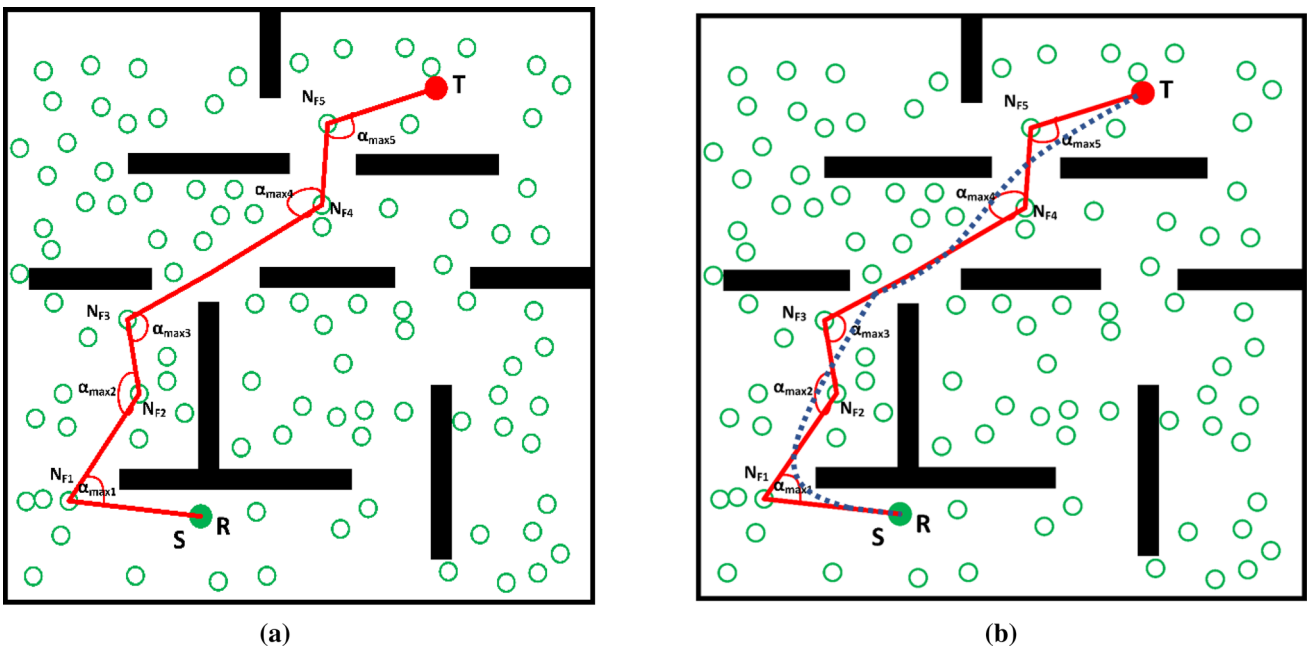


Fig. 5 a Initial path. b Path after smoothness principle (blue dotted line)

$H(\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2})$ , respectively. The slope of line  $\overline{SN_{F1}}(m_1)$  is  $\frac{y_1-y_0}{x_1-x_0}$  and slope of line  $\overline{N_{F1}N_{F2}}(m_2)$  is  $\frac{y_2-y_1}{x_2-x_1}$ . Therefore, the equation of line ( $L_1$ ) is:

$$y - \frac{y_0 + y_1}{2} = -\frac{1}{m_1} \left( x - \frac{x_0 + x_1}{2} \right) \tag{16}$$

For  $L_2$ , it is

$$y - \frac{y_1 + y_2}{2} = -\frac{1}{m_2} \left( x - \frac{x_1 + x_2}{2} \right) \tag{17}$$

Form Eqs. (15) and (16), a first point of turn is determined.

Step 3. Similarly, all point of turn is calculated corresponding to each line segment. Once the point of turn is calculated, the next step is to smooth the path.

Step 4. After calculating the first point of turn corresponding to line segments  $SN_{F1}$  and  $N_{F1}N_{F2}$ , the coordinates  $(x_0, y_0)$ ,  $(x_1, y_1)$ , and  $(x_2, y_2)$  are considered as diffused points. From the diffused points  $S(x_0, y_0)$ ,  $S_{F1}(x_1, y_1)$ , and  $S_{F2}(x_2, y_2)$ , the angle between the line segments  $\alpha_{max}$  can be calculated by using the cosine rule such as

$$\alpha_{max} = \cos^{-1} \left( \frac{a^2 + c^2 - b^2}{2ac} \right) \quad (18)$$

where

$$a = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

$$b = \sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2}$$

$$c = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}$$

For each point of turn, the turns are smoothed by varying the  $\alpha$  from 0 to  $\frac{\alpha_{max}\pi}{180}$  by using the following equations

$$x(\alpha) = (R_L - r_s) \cos(\alpha) + r_s \cdot \cos\left(\frac{R_L - r_s}{r_s} \alpha\right) \quad (19)$$

$$y(\alpha) = (R_L - r_s) \sin(\alpha) + r_s \cdot \sin\left(\frac{R_L - r_s}{r_s} \alpha\right) \quad (20)$$

where  $R_L = 4 \times r_s$ . Finally, the smooth path (blue dotted line) has been attained as shown in Fig. 5b. Figure 6 presents the flow chart of path planning using the proposed IMPSO, whereas the pseudo-code of the proposed IMPSO is provided in Algorithm 1.

## 4 Performance evolution

In this part, the outcomes attained by employing the proposed IMPSO are compared by GA-PSO (Huang and Tsai 2011), Advance PSO (Deepak et al. 2014), FACO (Yen

---

### Algorithm 1. Proposed IMPSO

---

**Input:** Set parameters such as population or particle size  $N$ , random particle's velocity and position, robot radius, sensor range, obstacles, initial and target locations

**Output:** Optimize and Smooth path

1: Initialize the particle with some random velocity and position

2: **For**  $i = 1$  to  $N$

3: **For**  $k = 1$  to  $itr_{max}$  do

4: Calculate fitness ( $fit_i$ ) of each particle

5: if  $fit(x_i, y_i) > fit(pbest)$

6: then  $fit(x_i, y_i) = pbest$  and set best of pbest as gbest

7: **End if**

8: If termination condition is not satisfied

9: **then** update each particle's velocity based on velocity regularized strategy by using Eq. (8)

10: **End if**

11: Implement the neighborhood search strategy based on reward value ( $\beta$ ) and utilization probability  $Prob_{uti}$  of each particle by using Eq. (13) and Eq. (14)

12: Get the new current location of the particles

13: if the new current location of particle  $i$  is better than its  $X_{p_b}^K$

14: then update  $X_{p_b}^K$

15: **End if**

16: if the new current location of particle  $i$  is better than its  $X_{p_g}^K$

17: **then** update  $X_{p_g}^K$

18: **End if**

19: Implement the path smoothness principle based on hypocycloid curves for path optimization

20: **End For**

21: **End For**

22: Smooth and optimize path

---



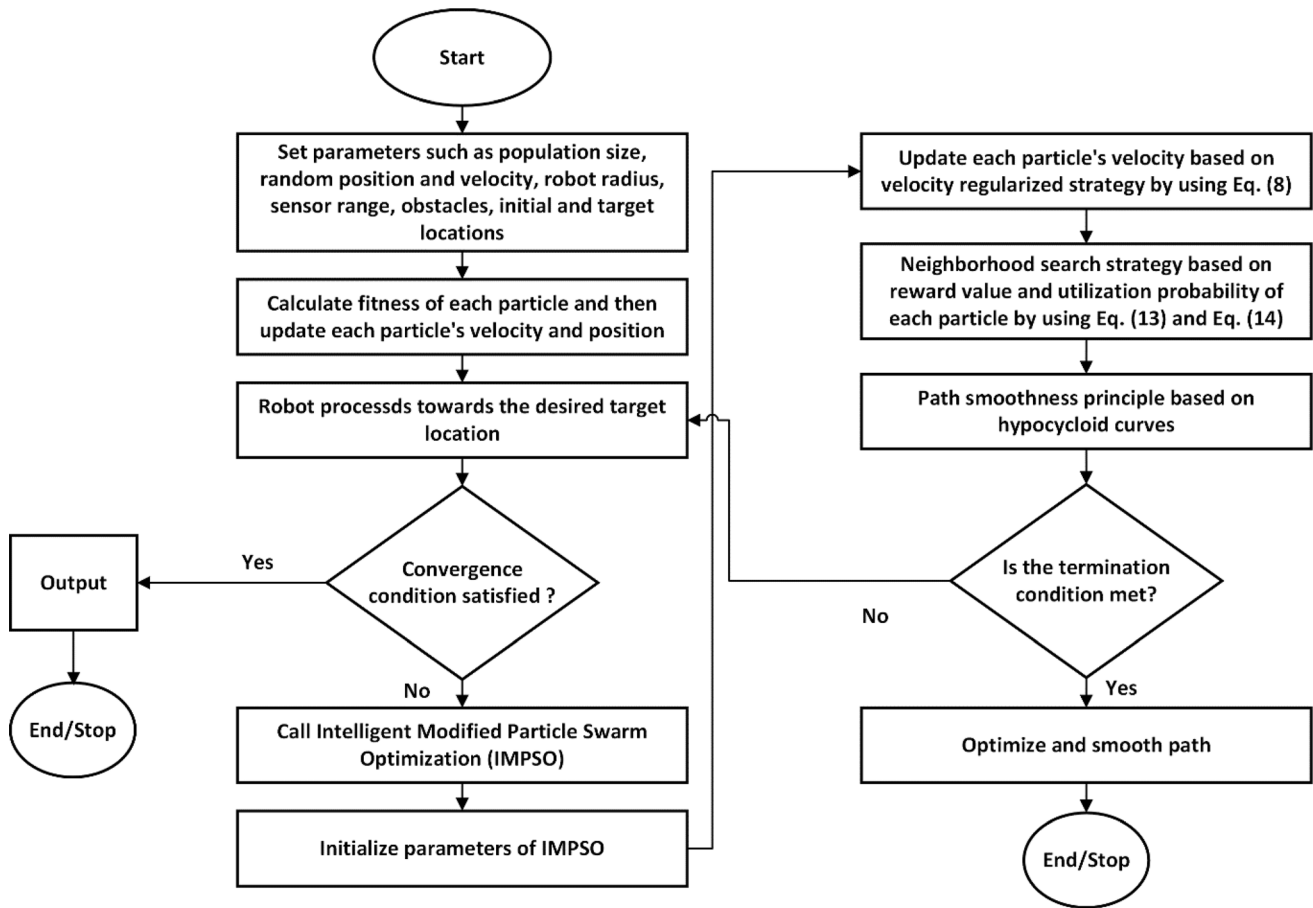


Fig. 6 Flowchart of path planning using the proposed IMPSO

and Cheng 2018), and other conventional approaches in terms of different performance metrics. In order to prove the superiority and efficacy of the proposed IMPSO, four distinct terrains with different complexity have been taken into account namely, terrain #1, terrain #2, terrain #3, and terrain #4. The simulation tests of all the techniques were carried out in a corridor-like complex terrains with a size of  $100 \times 100$ ,  $50 \times 50$ ,  $400 \times 400$ , and  $20 \times 20$  of the literature. The simulation has been carried out using the MATLAB (MATLAB 2020a) programming language with Intel® core i5 4.2 GH processor, hard disk of 500 GB and 6 GB of RAM memory. There is a variation in the number of samples or populations in the search space, such as  $[N = 40, 80, 120, 160]$  for terrain #1,  $[N = 50]$  for terrain #2 and  $[N = 15]$  for terrain #3, and  $[N = 15]$  for terrain #4. For better performance, the simulation tests were run 50 times with a maximum of 100 iterations. In each terrain, the robot has different initial and target locations. The following are the performance comparison in different terrains:

### 4.1 Performance comparison in terrain #1

Initially, the mobile robot is standing at the initial location (34, 10) and is required to reach the target location (70, 90). The parameters to be considered are stated in Table 1. As shown in Figs. 7, 8, 9, 10, all the path planning approaches efficiently attained the target without any collision with obstacles. The statistical findings in terms of path length, processing time, and smoothness are graphically presented in Fig. 11 for the different number of samples. As depicted in Fig. 11a, the proposed IMPSO gives the smallest path (112.59, 111.14, 107.50, 106.74) and RRT gives the longest path (237.52, 255.17, 264.34, 285.10) for each value of  $N$ . This signifies that the proposed IMPSO easily attain the target location as compared to other approaches. In terms of processing time, the proposed IMPSO surpasses the other approaches, but takes slightly more time (14.07, 17.82, 22.68, 23.07) as compared to PRM (2.20, 3.53, 5.75, 6.01) as displayed in Fig. 11b. In terms of smoothness, the proposed IMPSO provides smoother path (0.8391, 0.8835, 0.8527, 0.7739) as compared to other techniques as shown in Fig. 11c. The reason is that the proposed IMPSO have less sum of turning angles than that of other approaches.

**Table 1** Parameters

---

Test scenario size = $100 \times 100$
Initial location, $S(x_0, y_0) = (34, 10)$
Target location, $T(x_t, y_t) = (70, 90)$
Sensor range = $0.35 \times 0.35$
Hindrance width = $4m$
Hindrance height = $24m$
Robot radius, $r_{ROB} = 0.25$
Personal learning coefficient, $\eta_1 = 1.5$
Global learning coefficient, $\eta_2 = 1.5$
Probablility value, $P_a = \{0.2, 0.2, 0.4, 0.2\}$
Number of samples, $N = \{40, 80, 120, 160\}$
Target = Known

---

This implies that the proposed IMPSO consumes more energy. In addition, the Table 2 clearly illustrate that the proposed IMPSO is more stable as compared to other approaches under the different value of  $N$ .

To sum up, it has been clearly analysed from Fig. 11 and Table 2 that the performance parameters of each algorithm change significantly from the initial sample size to final sample size. In case of dynamic programming, A-star, dijkstra, PRM, PSO, and IMPSO, there is reduction in path length as the number of sample increases. Whereas in case of RRT, ABC, and ABC-EP, there is increment in path length as the number of sample increases. The optimal path length acquired by the proposed IMPSO is 106.74 when the sample size is 160. As the number of sample increases, the computational load also increases, which in turn increases the processing time. The minimum processing time taken by the modified IMPSO is 14.07 s when the sample size is 40. In case of smoothness, the proposed IMPSO gives the best smoothness value that is 0.7739 when the number of samples is 160. Overall, the proposed IMPSO performed more efficiently in all path planning performance indices for each number of samples  $N$ .

#### 4.2 Performance comparison in terrain #2 (50 × 50)

For terrain #2, the mobile robot is at the initial location (1, 1) and is necessary to reach the target (49, 49). The parameters used in GA-PSO (Huang and Tsai 2011) are: global learning coefficient ( $\eta_1$ ) = 1.5, personal learning coefficient ( $\eta_2$ ) = 1.5, inertia weight ( $w$ ) = 0.8, crossover probability = 0.7, mutation probability = 0.1, and iterations = 100. The path acquired by the GA-PSO is shown in Fig. 9 in Ref. (Huang and Tsai 2011), whereas the path acquired by the proposed IMPSO is depicted in Fig. 12a.

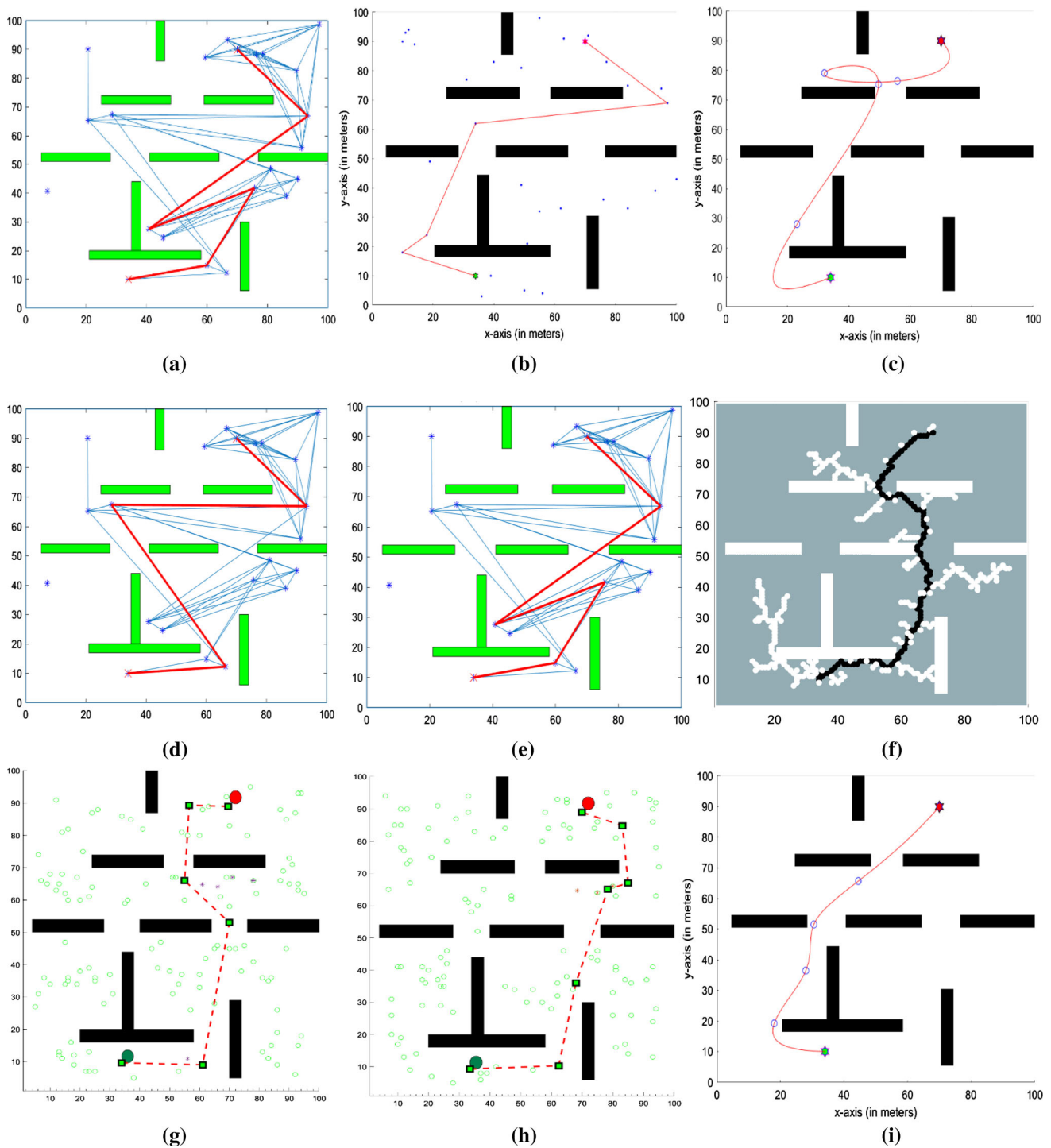
The proposed IMPSO delivers the optimal path length that is 72.18, whereas the optimal path length in GA-PSO is 76.94. It can be observed from Fig. 12b that the proposed IMPSO takes 21 iterations to find the global optimum solutions as compared to GA-PSO, that takes 90 iterations as shown in Fig. 10 in Ref. (Huang and Tsai 2011). This indicates that the proposed IMPSO is more efficient and faster convergence rate to reach the optimal path than GA-PSO. In addition, Table 3 illustrates that the proposed IMPSO is more stable as compared to GA-PSO (Huang and Tsai 2011).

#### 4.3 Performance comparison in terrain #3 (400 × 400)

For terrain #3, the mobile robot is at the initial location (340, 380) and is necessary to reach the target (10, 10). The parameters used in Advance PSO (Deepak et al. 2014) are: global learning coefficient ( $\eta_1$ ) = 2, personal learning coefficient ( $\eta_2$ ) = 2, inertia weight ( $w$ ) = 1.3, population size = 80, and iterations = 100. The path acquired by the Advance PSO is shown in Fig. 13b in Ref. (Deepak et al. 2014), whereas the path acquired by the proposed IMPSO is depicted in Fig. 13a. The proposed IMPSO delivers the optimal path length that is 543.71, whereas the optimal path length in Advance PSO is 659.8. The proposed PSO reduces the path length by 17.59% compared to Advance PSO (Deepak et al. 2014). This signifies that the proposed IMPSO can easily find the optimal path. It can be observed from Fig. 13b that the proposed IMPSO takes 17 iterations to find the global optimum solutions, whereas the algorithm in Ref. (Deepak et al. 2014) does not calculate the convergence rate. In addition, Table 4 demonstrate that the proposed IMPSO surpasses the Advance PSO (Deepak et al. 2014).

#### 4.4 Performance comparison in terrain #4 (20 × 20)

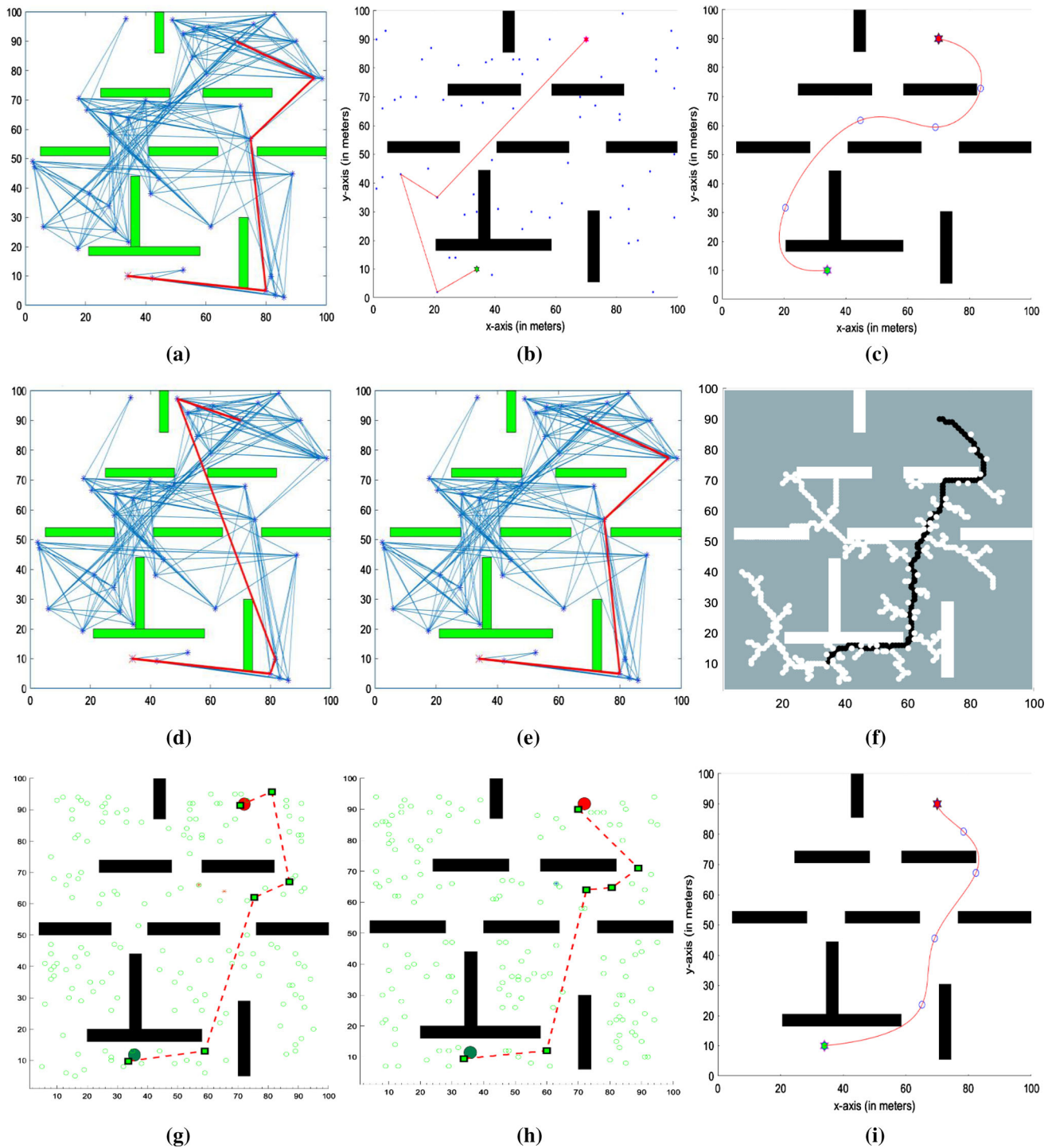
For terrain #4, the mobile robot is at the initial location (340, 380) and is necessary to reach the target (10, 10). The parameters used in FACO (Yen and Cheng 2018) are: pheromone factor ( $\alpha$ ) = 1, heuristic factor ( $\beta$ ) = 18, pheromone evaporation factor ( $\rho$ ) = 0.4, pheromone intensity coefficient ( $Q$ ) = 1, and population size = 50, and iterations = 100. The path acquired by the FACO is shown in Fig. 14a in Ref. (Yen and Cheng 2018), whereas the path acquired by the proposed IMPSO is depicted in Fig. 14a. The proposed IMPSO delivers the optimal path length that is 27.9647, whereas the optimal path length in FACO is 29.3848. The proposed IMPSO reduce the path length by 4.83% as compared to FACO. It can be observed from the Fig. 14b that the proposed IMPSO takes 19 iterations to



**Fig. 7** Path found by different path planning approaches when  $N = 40$ . **a** A\* algorithm; **b** PRM method; **c** PSO approach; **d** Dynamic programming; **e** Dijkstra algorithm; **f** RRT algorithm; **g** ABC algorithm; **h** ABC-EP algorithm; **i** Proposed IMPSO

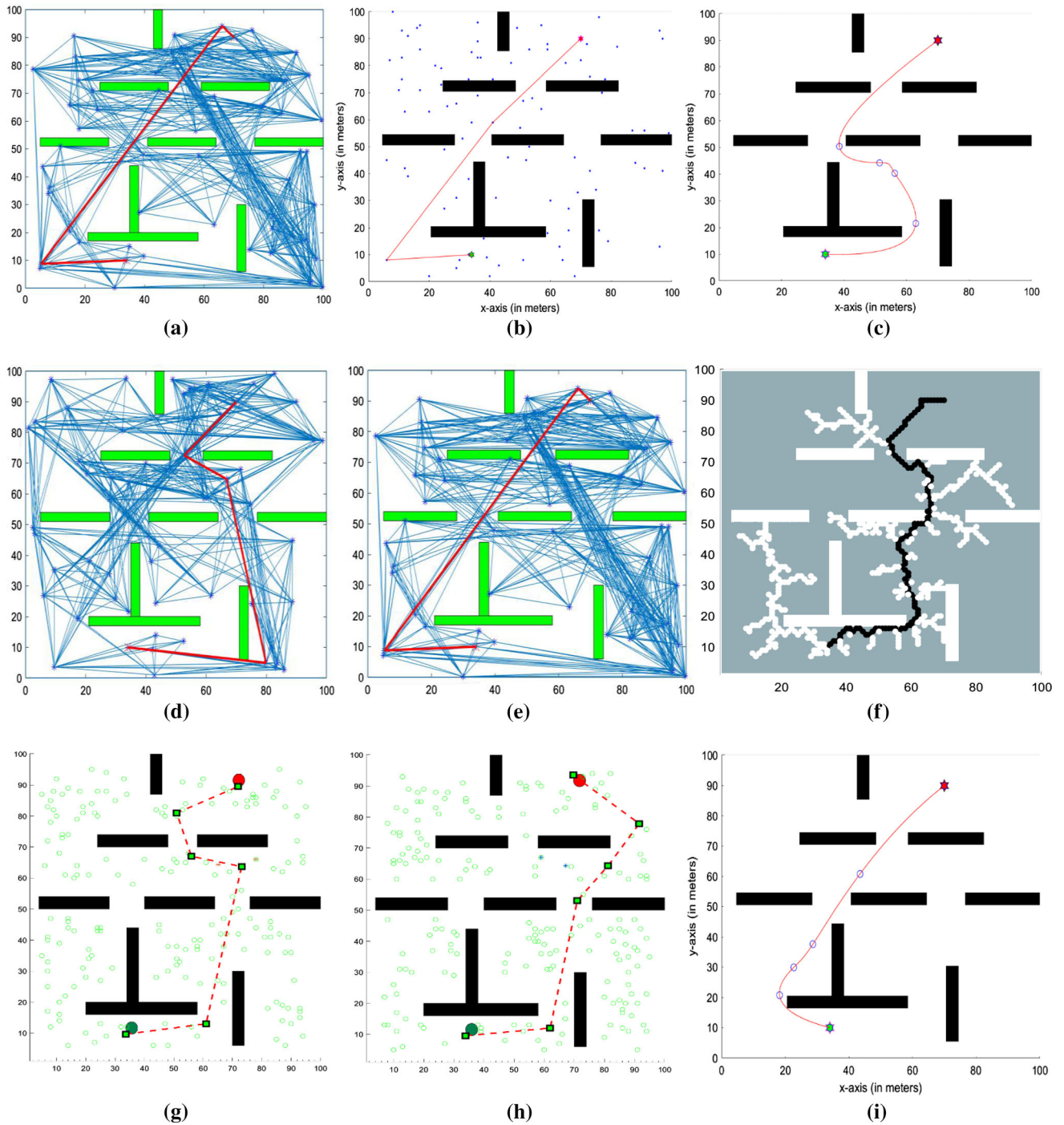
find the global optimum solutions as compared to FACO that takes 23 iterations as shown in Fig. 14b in Ref. (Yen and Cheng 2018). The convergence speed of proposed IMPSO increases by 17.39% than that of FACO. This indicates that the proposed IMPSO is more efficient and

faster convergence rate to reach the optimal path than FACO. In addition, Table 5 illustrates that the proposed IMPSO surpasses the FACO (Yen and Cheng 2018) in terms of path length and convergence rate.

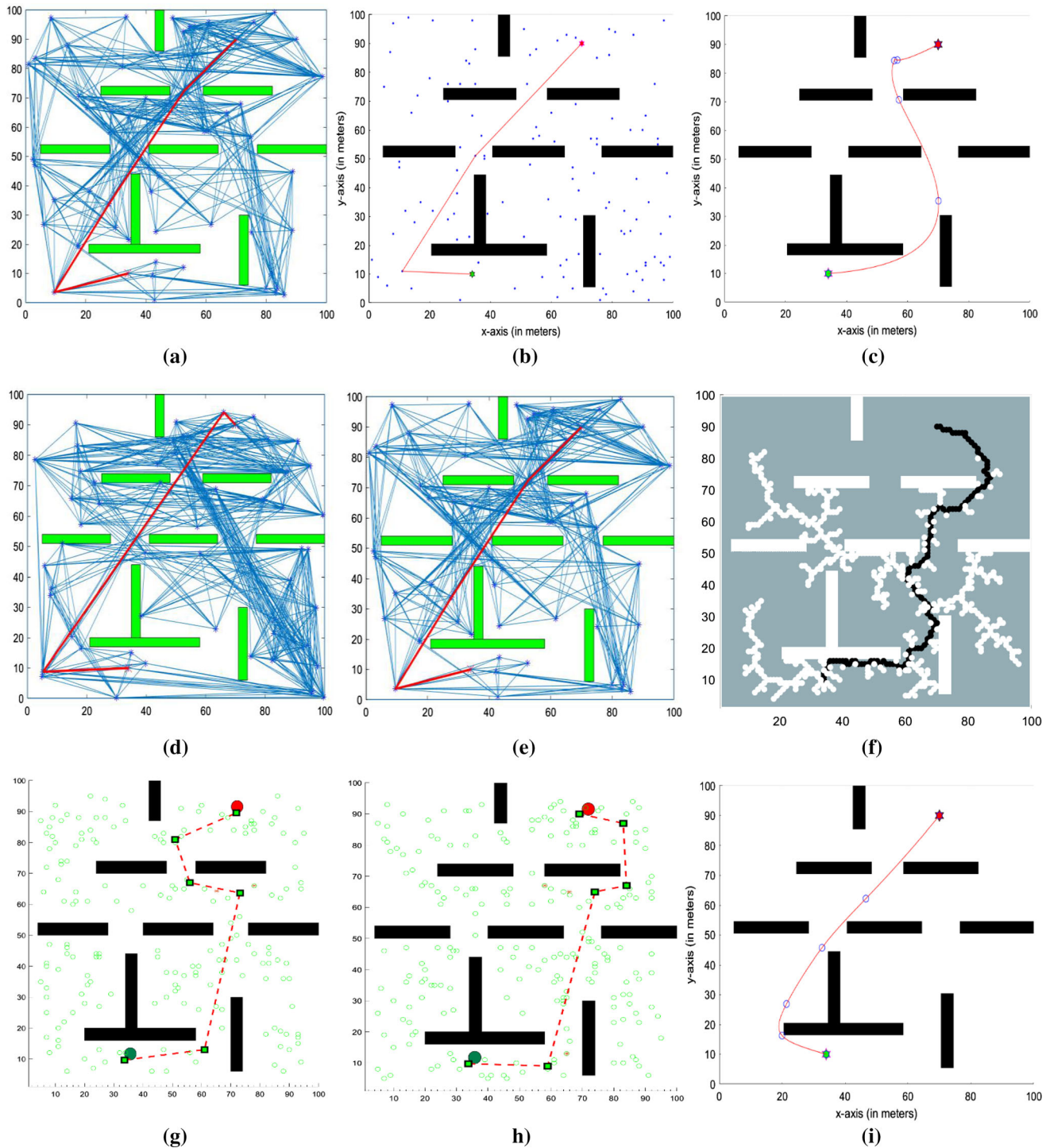


**Fig. 8** Path found by different path planning approaches when  $N = 80$ . **a** A\* algorithm; **b** PRM method; **c** PSO approach; **d** Dynamic programming; **e** Dijkstra algorithm; **f** RRT algorithm; **g** ABC algorithm; **h** ABC-EP algorithm; **i** Proposed IMPSO



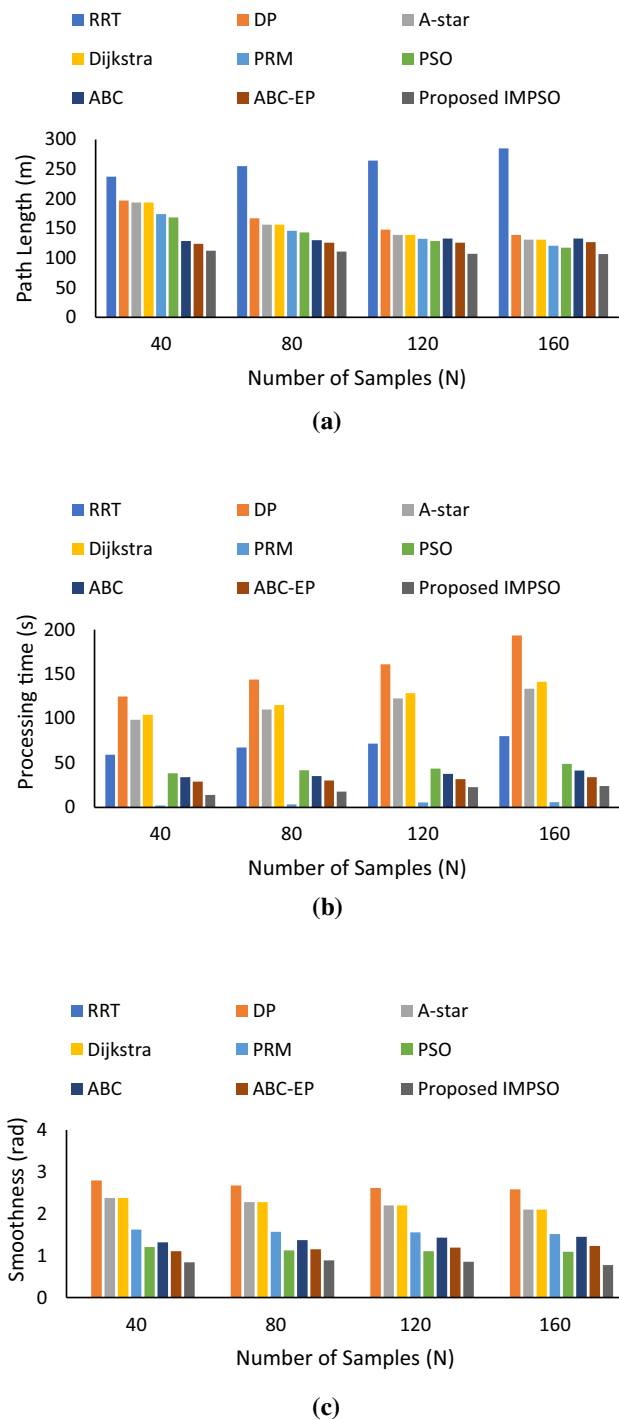


**Fig. 9** Path found by different path planning approaches when  $N = 120$ . **a**  $A^*$  algorithm; **b** PRM method; **c** PSO approach; **d** Dynamic programming; **e** Dijkstra algorithm; **f** RRT algorithm; **g** ABC algorithm; **h** ABC-EP algorithm; **i** Proposed IMPSO



**Fig. 10** Path found by different path planning approaches when  $N = 160$ . **a** A\* algorithm; **b** PRM method; **c** PSO approach; **d** Dynamic programming; **e** Dijkstra algorithm; **f** RRT algorithm; **g** ABC algorithm; **h** ABC-EP algorithm; **i** Proposed IMPSO





**Fig. 11** Comparative analysis of mobile robot path planning using all approaches for different value of *N*. **a** Path Length; **b** Processing time; **c** Smoothness

## 5 Conclusion

This research study presents an Intelligent Modified Particle Swarm Optimization (IMPSO) approach with a different strategy to address the path planning issues in a partially known or unknown complex terrain. Firstly, a velocity regularized strategy based on regularized coefficients has been applied to balance the exploration and exploitation ability. Secondly, a neighborhood search strategy based on reward value and utilization probability has been employed enriching search behaviors and avoiding premature convergence. Finally, a path smoothness principle based on hypocycloid curves has been used to smooth the sharp turns. A number of comparison tests have been carried out in a simulation scenario to validate the efficiency and performance of the proposed IMPSO as compared to GA-PSO, Advance PSO, FACO, and other conventional approaches. Moreover, the following results have been achieved:

- The proposed IMPSO outperforms the conventional approaches in terms of path length and smoothness.
- In case of processing time, the proposed IMPSO surpasses dynamic programming, A\*, dijkstra, PSO, ABC, and ABC-EP, but performs slightly less than PRM.
- When compared to GA-PSO, Advance PSO, and FACO, the proposed IMPSO reduces path length by 6.18%, 17.59% and 4.83%, respectively.
- In regards to convergence rate, the proposed IMPSO provides a better convergence rate than GA- PSO and FACO with improvements of 76.66%, 17.39%, respectively.

Form the performance comparison in different terrain, it is clear that the proposed IMPSO perform better in small dimension terrain as well as large dimension terrain. Following this proposed framework, future work will focus on moving obstacles and moving targets in clutter environments.

**Acknowledgements** This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

## Declarations

**Table 2** Performance analysis in terms of different performance indices

N	Methods	Path length (m)	Processing time (s)	Smoothness (radian)
		Mean	Mean	Mean
40	RRT	237.52	59.44	NaN
	Dynamic programming	197.09	124.73	2.7949
	A-star	193.74	98.65	2.3746
	Dijkstra	193.74	104.30	2.3746
	PRM	174.12	<b>2.20</b>	1.6244
	PSO	168.82	38.41	1.2027
	ABC	128.81	34.11	1.3182
	ABC-EP	124.05	29.07	1.1031
	Proposed IMPSO	<b>112.59</b>	14.07	<b>0.8391</b>
80	RRT	255.17	67.31	NaN
	Dynamic programming	166.96	143.81	2.6718
	A-star	156.63	110.11	2.2743
	Dijkstra	156.63	115.02	2.2743
	PRM	146.07	<b>3.53</b>	1.5702
	PSO	143.48	41.97	1.1232
	ABC	130.18	35.19	1.3702
	ABC-EP	125.82	30.20	1.1488
	Proposed IMPSO	<b>111.14</b>	17.82	<b>0.8835</b>
120	RRT	264.34	71.85	NaN
	Dynamic programming	148.04	160.90	2.6182
	A-star	139.19	122.59	2.1963
	Dijkstra	139.19	128.67	2.1963
	PRM	132.38	<b>5.75</b>	1.5562
	PSO	128.64	43.66	1.1022
	ABC	132.91	37.90	1.4290
	ABC-EP	125.99	31.78	<b>1.1921</b>
	Proposed IMPSO	<b>107.50</b>	22.68	<b>0.8527</b>
160	RRT	285.10	80.21	NaN
	Dynamic programming	139.19	193.36	2.5790
	A-star	131.23	133.61	2.1005
	Dijkstra	131.23	141.25	2.1005
	PRM	121.07	<b>6.01</b>	1.5151
	PSO	117.73	48.98	1.0918
	ABC	132.91	41.52	1.4521
	ABC-EP	127.09	34.16	1.2305
	Proposed IMPSO	<b>106.74</b>	23.97	<b>0.7739</b>

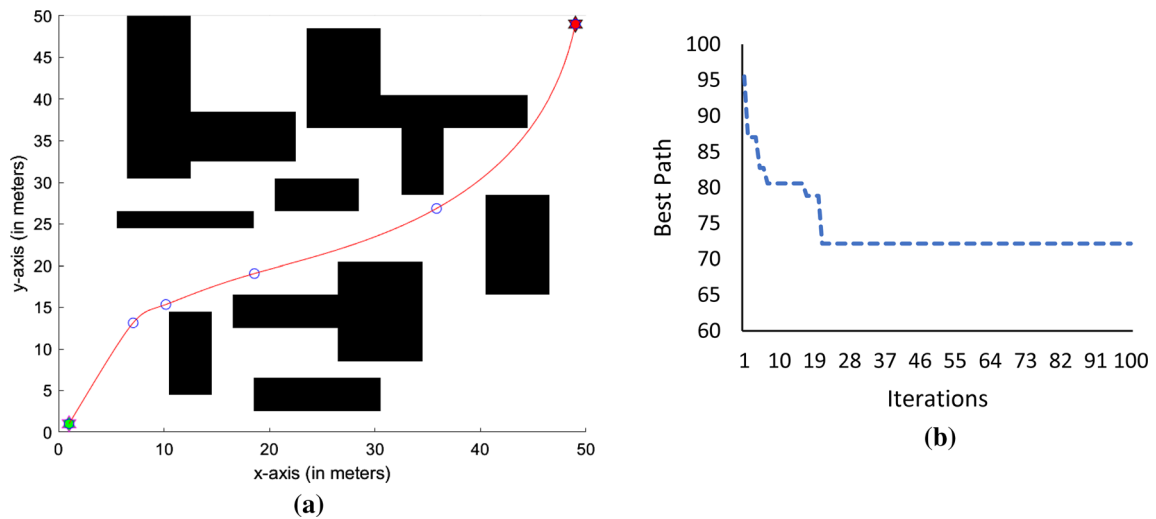


Fig. 12 a Path acquired by proposed IMPSO; b Convergence rate

Table 3 Comparison of proposed IMPSO with respect to GA-PSO (Huang and Tsai 2011)

Algorithm	Stop criteria	Path length	Convergence iterations times
GA-PSO (Huang and Tsai 2011)	Target reached	76.94	90
Proposed IMPSO	Target reached	72.18	21

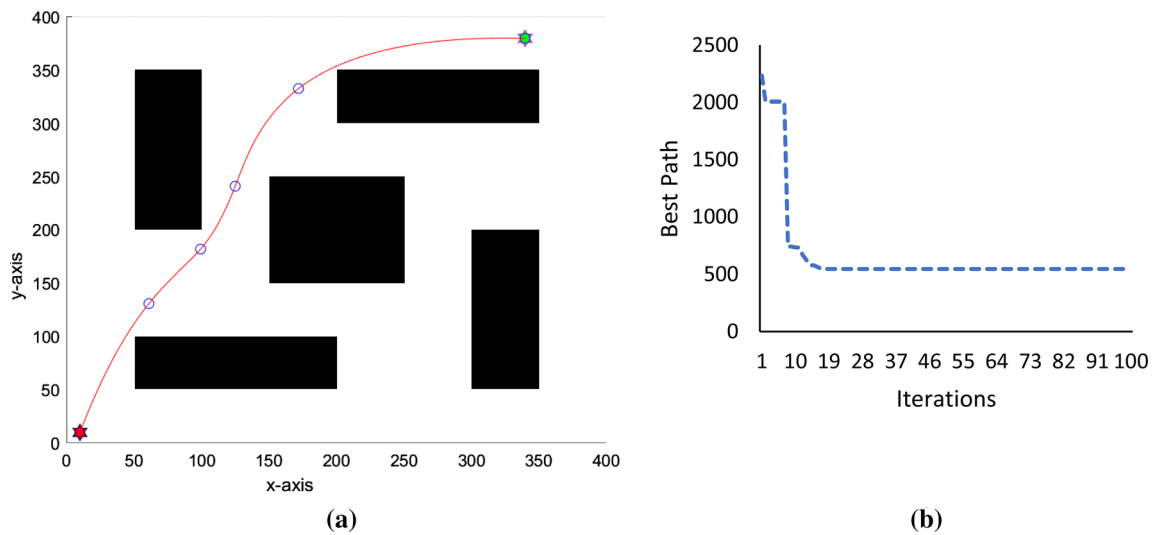
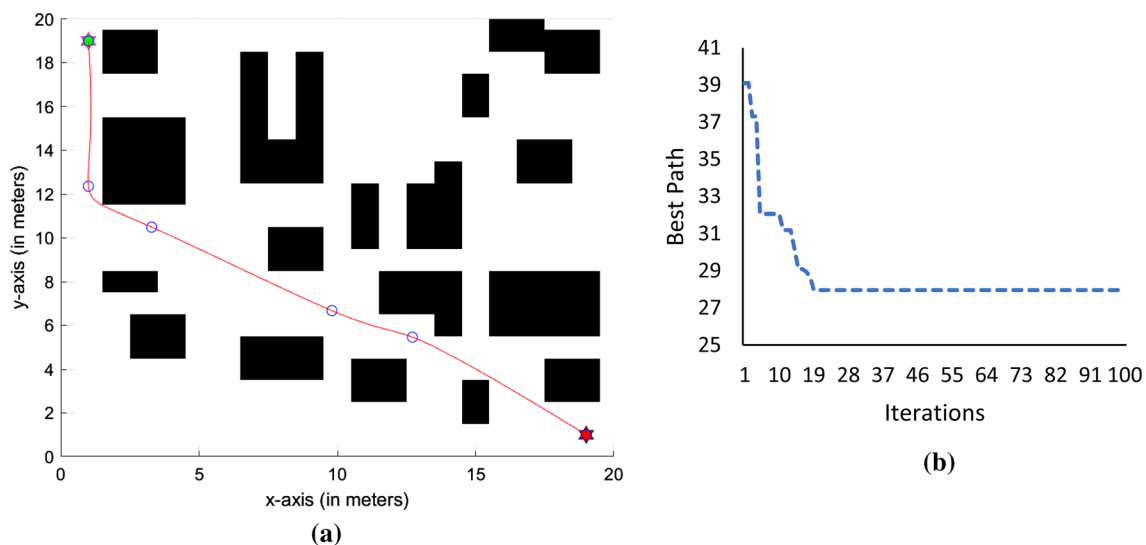


Fig. 13 a Path acquired by proposed IMPSO; b Convergence rate

Table 4 Comparison of proposed IMPSO with respect to Advance PSO (Deepak et al. 2014)

Algorithm	Population size	Stop criteria	Path length	Runs
Advance PSO (Deepak et al. 2014)	80	Target reached	659.8	20
Proposed IMPSO	80	Target reached	543.71	20



**Fig. 14** **a** Path acquired by proposed IMPSO; **b** Convergence rate

**Table 5** Comparison of proposed IMPSO with respect to FACO (Yen and Cheng 2018)

Algorithm	Population size	Stop criteria	Path length	Convergence iteration times
FACO (Yen and Cheng 2018)	50	Target reached	29.3848	23
Proposed IMPSO	50	Target reached	27.9647	19

**Conflict of interest** The authors declare that they have no competing interests.

## References

- Arora T, Gigras Y, Arora V (2014) Robotic path planning using genetic algorithm in dynamic environment. *Int J Comput Appl* 89:8–12. <https://doi.org/10.5120/15674-4422>
- Cesarone J, Eman KF (1989) Mobile robot routing with dynamic programming. *J Manuf Syst* 8:257–266. [https://doi.org/10.1016/0278-6125\(89\)90003-4](https://doi.org/10.1016/0278-6125(89)90003-4)
- Clark CM (2005) Probabilistic Road Map sampling strategies for multi-robot motion planning. *Rob Auton Syst* 53:244–264. <https://doi.org/10.1016/j.robot.2005.09.002>
- Deepak BBVL, Parhi DR, Raju BMVA (2014) Advance particle swarm optimization-based navigational controller for mobile robot. *Arab J Sci Eng* 39:6477–6487. <https://doi.org/10.1007/s13369-014-1154-z>
- Dolgov D, Thrun S, Montemerlo M, Diebel J (2008) Practical search techniques in path planning for autonomous driving. *Ann Arbor* 1001:32–37
- Gong DW, Zhang JH, Zhang Y (2011) Multi-objective particle swarm optimization for robot path planning in environment with danger sources. *J Comput* 6:1554–1561. <https://doi.org/10.4304/jcp.6.8.1554-1561>
- Han SD, Yu J (2020) DDM: fast near-optimal multi-robot path planning using diversified-path and optimal sub-problem solution database heuristics. *IEEE Robot Autom Lett* 5:1350–1357. <https://doi.org/10.1109/LRA.2020.2967326>
- He Y, Zhu L, Sun G, Dong M (2019) Study on formation control system for underwater spherical multi-robot. *Microsyst Technol* 25:1455–1466. <https://doi.org/10.1007/s00542-018-4173-y>
- Hossain MA, Ferdous I (2015) Autonomous robot path planning in dynamic environment using a new optimization technique inspired by bacterial foraging technique. *Rob Auton Syst* 64:137–141. <https://doi.org/10.1016/j.robot.2014.07.002>
- Huang HC, Tsai CC (2011) Global path planning for autonomous robot navigation using hybrid metaheuristic GA-PSO algorithm. In: *SICE annual conference*, Tokyo, pp 1338–1343
- Kumar S, Sikander A (2022) Optimum mobile robot path planning using improved artificial bee colony algorithm and evolutionary programming. *Arab J Sci Eng* 47:3519–3539. <https://doi.org/10.1007/s13369-021-06326-8>
- Li G, Chou W (2018) Path planning for mobile robot using self-adaptive learning particle swarm optimization. *Sci China Inf Sci* 61:1–18. <https://doi.org/10.1007/s11432-016-9115-2>
- Liao TI, Chen SS, Lien CC et al (2021) Development of a high-endurance cleaning robot with scanning-based path planning and path correction. *Microsyst Technol* 27:1061–1074. <https://doi.org/10.1007/s00542-018-4048-2>
- Loong WY, Long LZ, Hun LC (2011) A star path following mobile robot. In: *2011 4th International conference on mechatronics (ICOM)*, Kuala Lumpur, pp 1–7. <https://doi.org/10.1109/ICOM.2011.5937169>
- Mohanty PK, Parhi DR (2016) Optimal path planning for a mobile robot using cuckoo search algorithm. *J Exp Theor Artif Intell* 28:35–52. <https://doi.org/10.1080/0952813X.2014.971442>
- Patle BK, Ganesh Babu L, Pandey A et al (2019) A review: on path planning strategies for navigation of mobile robot. *Def Technol* 15:582–606. <https://doi.org/10.1016/j.dt.2019.04.011>

- Qi J, Yang H, Sun H (2021) MOD-RRT\*: a sampling-based algorithm for robot path planning in dynamic environment. *IEEE Trans Ind Electron* 68:7244–7251. <https://doi.org/10.1109/TIE.2020.2998740>
- Quan Y, Ouyang H, Zhang C et al (2021) Mobile robot dynamic path planning based on self-adaptive harmony search algorithm and morphin algorithm. *IEEE Access* 9:102758–102769. <https://doi.org/10.1109/ACCESS.2021.3098706>
- Wang H, Yu Y, Yuan Q (2011) Application of Dijkstra algorithm in robot path-planning. In: 2011 2nd International conference on mechanic automation and control engineering, Hohhot, pp 1067–1069. <https://doi.org/10.1109/MACE.2011.5987118>
- Weisstein EW (2016) Hypocycloid from mathworld-a wolfram web resource. <https://mathworld.wolfram.com/Hypocycloid.html>
- Xu J, Park KS (2020) A real-time path planning algorithm for cable-driven parallel robots in dynamic environment based on artificial potential guided RRT. *Microsyst Technol* 26:3533–3546. <https://doi.org/10.1007/s00542-020-04948-w>
- Yen CT, Cheng MF (2018) A study of fuzzy control with ant colony algorithm used in mobile robot for shortest path planning and obstacle avoidance. *Microsyst Technol* 24:125–135. <https://doi.org/10.1007/s00542-016-3192-9>
- Zhang Y, Gong DW, Zhang JH (2013) Robot path planning in uncertain environment using multi-objective particle swarm optimization. *Neurocomputing* 103:172–185. <https://doi.org/10.1016/j.neucom.2012.09.019>
- Zhang L, Zhang Y, Li Y (2020) Mobile robot path planning based on improved localized particle swarm optimization. *IEEE Sens J* 21:6962–6972. <https://doi.org/10.1109/JSEN.2020.3039275>
- Zhang Z, He R, Yang K (2021) A bioinspired path planning approach for mobile robots based on improved sparrow search algorithm. *Adv Manuf* 10:114–130. <https://doi.org/10.1007/s40436-021-00366-x>

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.