Optimizing Robot Path Planning with the Particle Swarm Optimization Algorithm

Falah Jaber Kshash — Southern Technical University, Iraq
F.J.Albadri@stu.edu.iq

Ahmed Yahia Yaseen — Southern Technical University, Iraq
ahmedyahya@stu.edu.iq

Salam Saadoon Kals — Ministry of Education
salameng2002@yahoo.com

ABSTRACT

The primary objective of the present study is to advance the development of an intelligent and autonomous robotic system that possesses the ability to independently plan its movements across a wide range of situations without the need for human interaction. Path planning is a crucial technological aspect that facilitates the intelligent navigation of mobile robots in environments that are abundant in obstacles. The research utilizes the Particle Swarm Optimization (PSO) method to determine an ideal trajectory for a mobile robot, considering obstacle avoidance. The findings of this study indicate that the technique employed exhibits an important level of effectiveness in successfully navigating obstacles and attaining the most optimal route. The optimization issues were assessed by utilizing a range of static impediments in diverse contexts.

Keywords: Mobile Robotics, Path Planning, Optimization Approach, Particle Swarm Optimization Algorithm

I. Introduction

Mobile robots are entrusted with increasingly complex missions, including planetary exploration, medical support, search and rescue, and reconnaissance, as they become an essential component of human life [1]. Path planning is a fundamental challenge in robotics. Finding a safe route for a robot to travel from an initial location to a goal position is a common challenge [2]. In essence, the primary goal of path planning is to identify a secure and efficient path for a mobile robot to navigate through a hazardous environment, enabling the robot to move from one location to another without encountering obstacles [3], [4]. Depending on the robot's environment, different planning approaches fall into two categories: local path planning and global path planning [5].

Global planning techniques (hierarchical paradigm) generate the mobile robot's path using a known map. These techniques search for a complete path, typically yielding success if such a path exists. They are inherently convergent and highly effective in static situations. Conversely, local planning operates on a reactive paradigm, relying heavily on local sensors instead of possessing a complete map. Due to the unpredictable nature of the environment, local planning instructs robots to detect nearby obstacles and take immediate action based on this information [6]. To avoid unforeseen obstacles, whether static or dynamic, reactive planning must be quick and effective in real-time [7].

In recent decades, numerous studies have been conducted in this field, proposing various algorithms to address and optimize path planning problems. These algorithms include the A* algorithm, the potential field method, the simulated annealing algorithm, the genetic

algorithm (GA), the particle swarm optimization (PSO), the ant colony optimization (ACO), and the gravitational search algorithm (GSA) [1], [8]. However, to find the shortest and smoothest path from an arbitrary starting point to a desired endpoint while ensuring safety through obstacle avoidance, several criteria must be simultaneously satisfied, including minimizing path length and travel time. Consequently, this challenge can be seen as an optimization problem addressable through optimization algorithms [9].

Our study offers two significant contributions. First, it formulates the global path planning problem for mobile robots as a multi-objective optimization problem that encompasses both path length and minimum time as objectives. Second, it proposes a particle swarm optimization approach for optimizing path length in various mobile robot environments, providing four different scenarios.

II. Mobile Robot Path Planning Problem

Artificial intelligence falls under the domain of computer science, focusing on system management aimed at optimizing the system's objective function using optimization techniques. This optimization is carried out once all the necessary constraints have been satisfied. The challenge of robot path planning can be addressed through two main approaches: classical methods and heuristic methods. Both of these approaches can be employed to tackle the robot path planning problem.

However, classical methods often fail to yield optimal paths and are susceptible to getting trapped in local minima. Moreover, when dealing with numerous obstacles or complex scenarios, classical methods may not provide the best solutions. Heuristic techniques, such as Genetic Algorithms (GA) and Particle Swarm Optimization (PSO) [10], [11], have garnered the attention of researchers due to their ability to mitigate the inefficiencies associated with traditional methods.

Heuristic methods offer several advantages, including ease of implementation and rapid development of a viable solution if one exists. These methods are adopted to overcome the limitations inherent in conventional motion planning approaches, including high computational costs and time requirements [12]. Heuristic techniques, in general, exhibit robust capabilities in guiding robots safely through their environment while avoiding collisions with obstacles [13].

A. Mathematical Modeling of the Mobile Robot Environment

In this study, MATLAB 2020 was utilized to create models of the environments. Within a two-dimensional workspace, a mobile robot is simplified as a point mass to facilitate modeling. To prevent collisions, all obstacles are expanded by the mobile robot's inscribed radius. In this conceptualization, obstacles are represented as circles of various sizes. Inside these circles, mobile robots face obstruction and a potential collision risk, while outside, they remain unaffected.

The path planning process encompasses three fundamental components:

- The proposed environments are standardized to a size of 200*200 units.
- Three scenarios are considered, each featuring varying numbers of obstacles: five, four, and three obstacles.
- Circular shapes of different dimensions are employed to depict the obstacles.

The shortest distance is determined by minimizing the distance function $f(x, y)$, also known as the Euclidean distance equation.

$$f(x,y) = \sqrt{\left(x_{j+1} - x_j\right)^2 + \left(y_{j+1} - y_j\right)^2} \quad (1)$$

where: x_{j+1}, y_{j+1} represent the next position, x_j, y_j represent the current position.

B. Obstacle Detection and Avoidance

In robotics, route planning often focuses on developing algorithms that enable efficient movement with optimal alternative paths. The environment where the robot operates is divided into two categories: empty space and space occupied by obstacles. The starting point and the target location are located within the empty space. In this document, the following assumptions are necessary to detect environmental obstacles:

1. The program's model analyzes the robot's environment by converting it into binary codes.
2. Binary coding distinguishes between obstacles and free space, with binary 1 representing occupied space and binary 0 representing unoccupied space.

Hence, the mobile robot identifies paths between any two points within the obstacle-laden environment through obstacle detection and avoidance. This involves considering various movement directions, such as right, upper right, lower right, up, down, lower left, upper left, and left, to navigate from one location to another.

C. Particle Swarm Optimization Algorithm (PSO)

The fundamental principle underlying the PSO method is the exchange of information among individuals within a group. This exchange aims to transform the collective movement of the group from disorder to order within the solution space, ultimately seeking to obtain the optimal solution for the problem at hand [12]. The solution to each optimization problem involves updating the "particle's" speed and position according to equations (2) and (3).

In equation (2), the first term signifies that the particle's next movement is influenced by the magnitude and direction of its previous velocity. The second term represents that the particle's subsequent action is derived from its individual experience. The third term indicates that the particle's next move is influenced by the collective experience, particularly the best experience gained from its companions [14]. In essence, the particle's next step is determined by both its individual experience and the collective knowledge gathered from its fellow particles.

$$V_{id}^{k+1} = \omega V_{id}^{k} + c_1 r_1 \cdot X_{id}^{k}\big) + c_2 r_2 \big(P_{qd}^{k} - X_{id}^{k}\big) \quad (2)$$

$$X_{id}^{k+1} = X_{id}^{k} + V_{id}^{k+1} \quad (3)$$

Where:

V_{id}^{k}: The dth component of the flight velocity vector of particle i in the kth iteration.

X_{id}^{k}: The dth element of the position vector of particle i in the kth iteration.

c_1 and c_2: Learning factors used to adjust the learning towards the maximum step size.

r_1 and r_2: Two random functions with values ranging from 0 to 1, introducing randomness to the search process.

ω: The inertia weight used to fine-tune the search capability within the solution space [14], [1].

PSO offers several advantages over traditional approaches, including simplicity, ease of implementation, robustness, insensitivity to population size, rapid convergence rates, effectiveness, efficiency, superior search quality, and the ability to find optimal global solutions for both linear and nonlinear optimization problems [15].

In the context of path planning, after each iteration, particle positions and velocities are updated using Equations (2) and (3). Subsequently, an evaluation is conducted to determine if the particle positions lie inside obstacles. If they do, the swarm is discarded, and further updates are performed. If not, the global best ($\boldsymbol{G}_{\mathbf{best}}$) and particle best ($\boldsymbol{P}_{\mathbf{best}}$) values are computed for the particles. Finally, a check is made to ascertain whether the termination condition has been met. If it has, the process terminates; otherwise, it proceeds to forecast the fitness function. Therefore, the fitness function plays a crucial role in this context.

The PSO algorithm[16]:

1. Initialize the parameters for the Particle Swarm Optimization (PSO) algorithm.
2. For the maximum number of iterations to find the optimal solution:
   a. For each particle in the swarm: Calculate the fitness value based on the objective function.
   b. End of iteration.
3. For each particle in the swarm:
   Calculate the velocity and apply velocity constraints to groups of particles.
4. End of iteration.
5. Update the fitness value of each particle if it is better than the previous value.

These steps outline the PSO algorithm, which involves initializing parameters, evaluating fitness values, updating particle velocities, and

continually improving fitness values during each iteration.

The flowchart below can be utilized to implement PSO [17]:

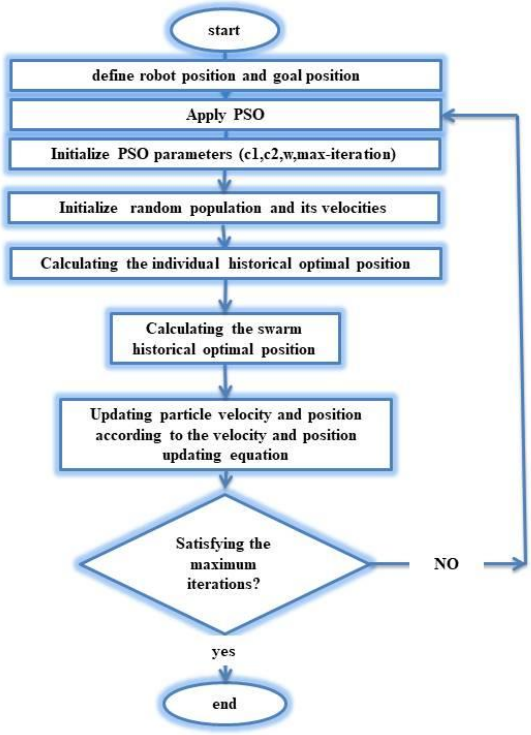

Figure 1. PSO flowchart

III. SIMULATION RESULTS

The Particle Swarm Optimization (PSO) algorithm proposed for mobile robot path planning was implemented using MATLAB R2020a. Experiments were conducted on a platform equipped with an Intel Core i7-3770K CPU running at 4GHz, 16GB of RAM, and the Windows 10 Professional operating system. The objective is for the mobile robot to navigate from its starting point (S) to its destination (T).

The proposed PSO algorithm utilized specific parameters, namely (c1=c2=2, w=1), which were determined empirically through preliminary testing. To identify feasible paths using the local search procedure, 20 random points were generated. The population size for the PSO algorithm was set to 20, and the maximum number of iterations was capped at 20.

Testing was performed on a 200x200 grid search space containing 3, 4, and 5 obstacles. The proposed method was executed 10 times for each test environment, generating different paths in each run. This demonstrates the robustness of the PSO algorithm.

The starting and goal points, as well as the obstacle positions, influenced the path length. Obstacles were represented as green circles. Two cases were tested in each environment:

The first case set the starting and goal point coordinates as (30,20) and (170,190), respectively.

The second case set the starting and target point coordinates as (20, 50) and (50, 200), respectively.

The size and position of each obstacle remained constant during the path length estimation. The first environment featured three obstacles, the second had four obstacles, and the last environment contained five obstacles.

Table 1: Size and Position of the Obstacles

Obstacle Number	X-Axis Coordinate	Y-Axis Coordinate	Circle Radius
Obstacle1	30	150	20
Obstacle2	50	60	30

Obstacle3	100	100	25
Obstacle4	160	25	25
Obstacle5	150	150	30

This table provides details about the obstacles in the environment, including their identification number, X and Y-axis coordinates, and the radius of each obstacle's circular shape. In the first test environment, which contains three static obstacles, we have depicted the obtained path using our proposed method in Figures 2 and 3. The computed time required to reach the destination and the corresponding path length for each case can be found in Table 2. We conducted a series of tests to determine the shortest path in each scenario. Additionally, we have included a graph illustrating the path length versus iterations to showcase the best results achieved in each situation. The graph also illustrates the robot's optimal path at each iteration.

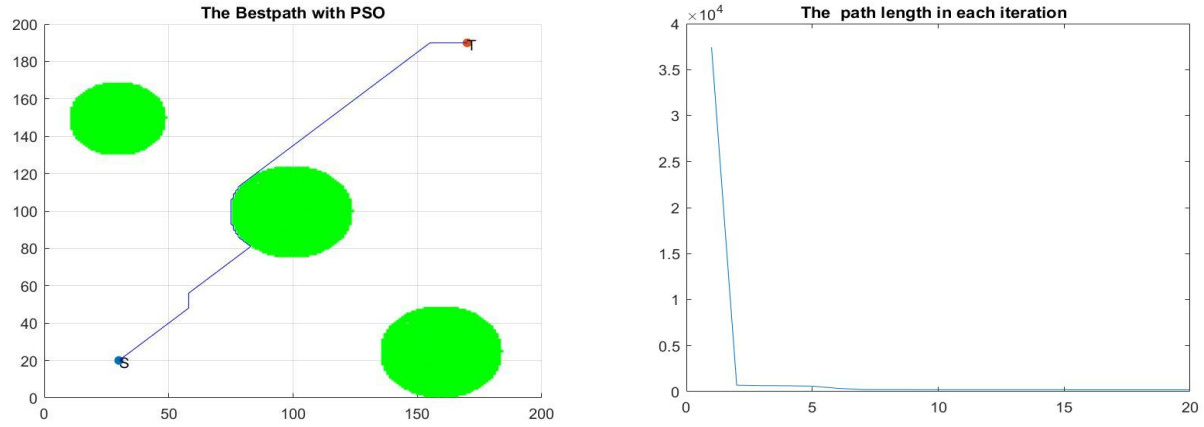

Figure 2. Case1: Robot Path Planning with Three Obstacles

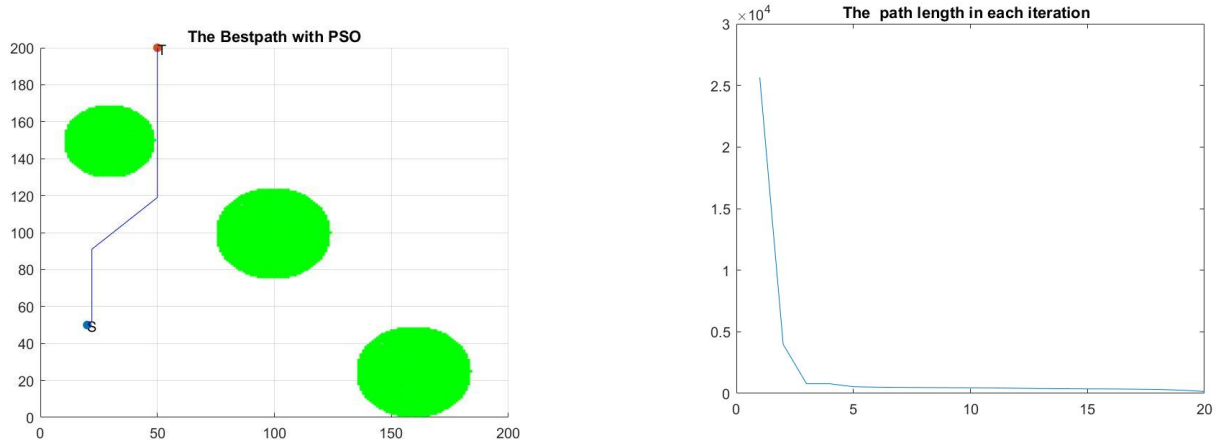

Figure 3. Case2: Robot Path Planning with Three Obstacles

In the second test scenario, there are four obstacles, but the conditions are otherwise similar to the first scenario. The goal is to find a path that avoids these obstacles and reaches the desired destination. The routes generated using our proposed PSO approach are depicted in

Figures 4 and 5. The calculated path length and time required to reach the destination for each example are presented in Table 2. We conducted a series of experiments to identify the shortest path in each scenario. The graph below displays the robot's shortest path length in this environment, along with a graphical representation of path length versus iterations, illustrating the best outcomes achieved in each case. The graph also highlights the robot's optimal path at each iteration.

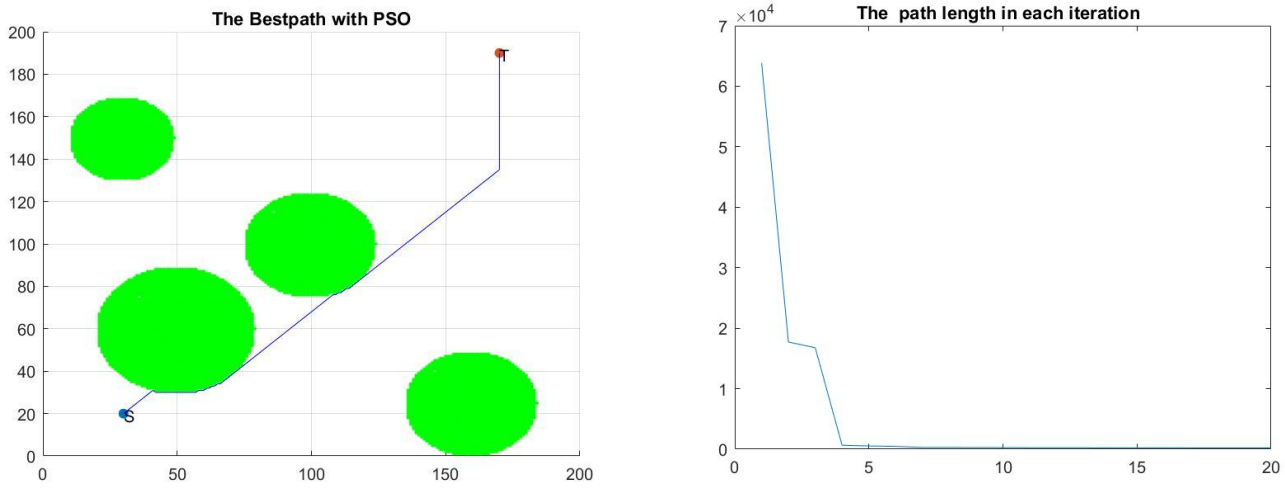

Figure 4. Case1: Robot Path Planning with Four Obstacles

This figure illustrates the robot's path in the first test case within Test Environment 2. The robot successfully navigates a path through an environment containing four static obstacles using the proposed PSO-based path planning method. It's evident that the robot has successfully reached its destination, as indicated by the blue path marked on the PSO-generated route. The effectiveness of the PSO algorithm is confirmed by the minimal time required and the path length taken by the robot to reach its destination. The proposed algorithm has proven to be highly effective in identifying the optimal path while avoiding obstacles.

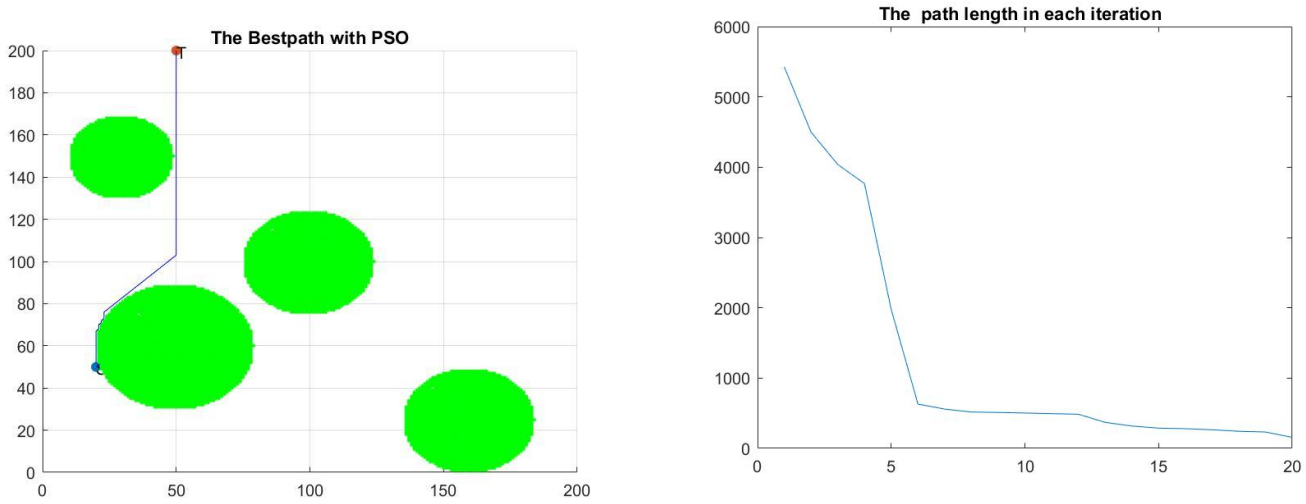

Figure 5.Case2: Robot Path Planning with Four Obstacles

This figure illustrates the robot's path in the second test case within Test Environment 2. The robot successfully navigates a path through an environment containing four static obstacles

using the proposed PSO-based path planning method.

In the third test environment, the complexity increases as two additional obstacles are introduced compared to the previous test examples. The path followed by the mobile robot can be visualized in Figures 6 and 7. The time taken to reach the destination and the corresponding path length for each scenario are provided in Table 2. To identify the shortest path in each situation, a series of tests were conducted. The graph below displays the robot's shortest path length in this environment and illustrates the path length versus iterations, showcasing the best outcomes achieved in each condition. The graph also depicts the robot's optimal path at each iteration.

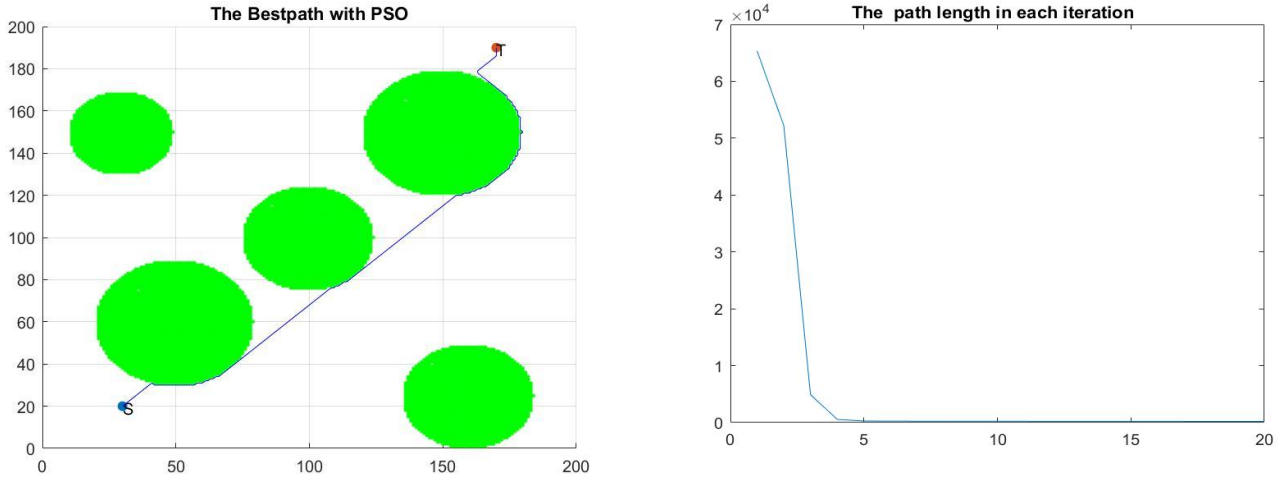

Figure 6.Case1: Robot Path Planning with Five Obstacles

This figure illustrates the path taken by the robot in the first test case within Test Environment 3. The robot successfully navigates a path through an environment containing five static obstacles using the proposed PSO-based path planning method.

Table 2: Path Length and Execution Time for Each Case

Environment 1 with 3 Obstacles			Environment 2 with 4 Obstacles			Environment 3 with 5 Obstacles		
Case Number	Path Length (Pixel)	Execution Time (Sec)	Case Number	Path Length (Pixel)	Execution Time (Sec)	Case Number	Path Length (Pixel)	Execution Time (Sec)
Case1	187	26.9	Case1	197	29.7	Case1	204	32.7
Case2	152	22.9	Case2	152	27.4	Case2	152	28.5

This table provides path length (in pixels) and execution time (in seconds) for different cases in three different environments, each with a varying number of obstacles.

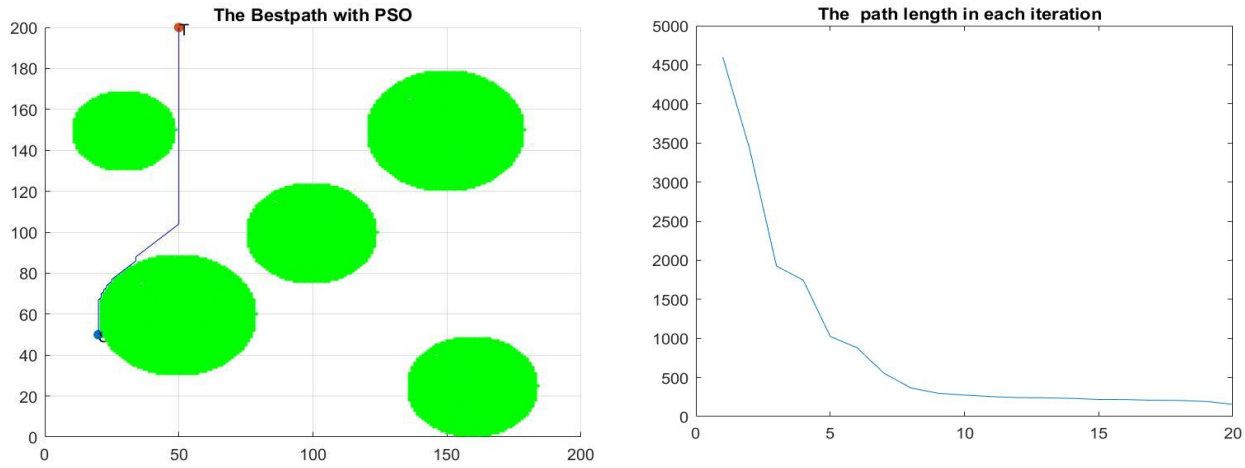

Figure 7. Case2: Robot Path Planning with Five Obstacles

IV. CONCLUSION

The primary objective of this research was to reduce the time it takes for a mobile robot to travel from its source point to its destination point by optimizing the path length. The PSO (Particle Swarm Optimization) method was employed to establish the optimal trajectory path for a mobile robot navigating through a static environment. Based on the results of the simulation study, it can be concluded that the PSO soft computing technique offers a simpler and more straightforward implementation compared to other contemporary approaches. Furthermore, it efficiently identifies the shortest path in a significantly reduced amount of time.

The primary goal of this study was to identify the shortest, safest, and collision-free path between two locations. The data extracted from the study indicate that environments containing three obstacles tend to result in a reduced path length and a quicker arrival time for the robot at its goal, when compared to environments with four or five obstacles.

REFERENCES

[1] G. Li and W. Chou, "Path planning for mobile robot using self-adaptive learning particle swarm optimization", *Science China Information Sciences*, vol. 61, no. 5, pp. 1–18, 2018.

[2] A. Koubaa *et al.*, "Introduction to mobile robot path planning," *Stud. Comput. Intell.*, vol. 772, no., pp. 3–12, November 2018.

[3] M. Malathi, K. Ramar, and C. Paramasivam, "Optimal Path Planning for Mobile Robots Using Particle Swarm Optimization and Dijkstra Algorithm with Performance Comparison," *Middle-East J. Sci. Res.*, vol. 24, no. S1, pp. 312–320, 2016.

[4] Z. Elmi and M. Ö. Efe, "Online path planning of mobile robot using grasshopper algorithm in a dynamic and unknown environment," *J. Exp. Theor. Artif. Intell.*, vol. 33, no. 3, pp. 467–485, 2021.

[5] N. A. K. Zghair and A. S. Al-Araji, "A one decade survey of autonomous mobile robot systems," *Int. J. Electr. Comput. Eng.*, vol. 11, no. 6, pp. 4891–4906,Dec. 2021.

[6] B. K. Patle *et al.*, "A review: On path planning strategies for navigation of mobile robot," *Def. Technol.*, vol. 15, no. 4, pp. 582–606, 2019.

[7] A. A. Ravankar, A. Ravankar, T. Emaru, and Y. Kobayashi, "HPPRM: Hybrid Potential Based Probabilistic Roadmap Algorithm for Improved Dynamic Path Planning of Mobile Robots," *IEEE Access*, vol. 8, pp. 221743–221766, Dec.2020.

[8] P. D. Pandey, Anish, Shalini Pandey, "Mobile Robot Navigation and Obstacle Avoidance Techniques: A Review," *Int. Robot. Autom. J.*, vol. 2, no. 3, pp. 96–105,

May 2017.

[9] S. Hosseininejad and C. Dadkhah, "Mobile robot path planning in dynamic environment based on cuckoo optimization algorithm", International Journal of Advanced Robotic Systems, vol. 16, no. 2, pp. 1–13, 2019.

[10] E. Masehian and D. Sedighizadeh, "Classic and Heuristic Approaches in Robot Motion Planning – A Chronological Review", International Journal of Mechanical, Industrial Science and Engineering , Vol.1, No:5, 2007.

[11] T. T. Mac, C. Copot, D. T. Tran, et al., "Heuristic approaches in robot path planning: A survey", Robitcs and Autonomous Systems, vol. 86, pp. 13–28, 2016.

[12] M. S. Alam and M. U. Rafique, "Mobile robot path planning in environments cluttered with non-convex obstacles using particle swarm optimization", 2015 International Conference on Control ,Automation and Robotics (ICCAR) IEEE, 2015, vol. 3, no. 3, pp. 32–36, 2015.

[13] M. Samadi and M. F. Othman, "Global path planning for autonomous mobile robot using genetic algorithm", 2013 International Conference on Signal-Image Technolgy and Internet-Based Systems IEEE, pp. 726–730, 2013.

[14] X. Li *et al*, "An Improved Method of Particle Swarm Optimization for Path Planning of Mobile Robot," *J. Control Sci. Eng.*, vol. 2020,pp.1-12,May 2020.

[15] D. Agarwal and P. S. Bharti, "MATLAB Simulation of Path Planning and Obstacle Avoidance Problem in Mobile Robot using SA, PSO and FA," *2020 IEEE Int. Conf. Innov. Technol. INOCON 2020*, pp. 1–6, 2020.

[16] H. S. Dewang *et al*, "A Robust Path Planning for Mobile Robot Using Smart Particle Swarm Optimization," *Procedia Comput. Sci.*, vol. 133, pp. 290–297,Jan. 2018.

[17] D. Wang, D. Tan, and L. Liu, "Particle swarm optimization algorithm: an overview", Soft Computing, vol. 22, no. 2, pp. 387–408, 2018.