Proceedings of the 4th International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/2026.36919

Multi-robot Path Planning in Dynamic Environments Using the Coati Optimization Algorithm

Hongyu Lu

UWC Changshu China, Changshu, China
hylu25@uwcchina.org

Abstract. Multi-robot path planning in complex dynamic environments is a challenging optimization problem due to the simultaneous requirements of path efficiency, obstacle avoidance, and inter-robot collision prevention. To address these challenges, this study applies the Coati Optimization Algorithm (COA), a bio-inspired swarm intelligence optimization method, to multi-robot path planning. The trajectories of multiple robots are represented by a set of continuous control waypoints, and a comprehensive objective function is formulated by jointly considering path length, static obstacle avoidance, and inter-robot collision penalties. By exploiting the global exploration and local exploitation capabilities of COA, feasible and smooth collision-free trajectories can be generated in environments containing both static and dynamic obstacles. Simulation experiments were conducted in MATLAB R2024a with 12 mobile robots, 9 static obstacles, and 5 dynamic obstacles, and the performance of COA was compared with that of the A* algorithm. The results show that COA produces smoother paths with fewer unnecessary turns and reduces the total travel distance from 670.9 to 617.8355, corresponding to an improvement of approximately 7.9%. These results demonstrate that COA provides an effective approach for coordinated multi-robot path planning in complex dynamic environments and has potential for further application to practical multi-AGV and mobile robot systems.

Keywords: multi-robot system, path planning, Coati Optimization Algorithm, swarm intelligence, dynamic obstacle avoidance

1. Introduction

In contemporary times, with the rapid development of artificial intelligence and automation technology, multi-robot systems (MRS) are increasingly widely used in intelligent warehouse storage and management, industrial manufacturing, disaster relief and rescue, and even in people's daily lives [1]. In multi-robot systems, path planning is one of the cores and most challenging technologies. It requires finding a shortest path from the starting point to the end point for multiple robots in a complex working environment that includes multiple static and dynamic obstacles, thereby saving time and costs [2]. The widespread application of this technology helps to ensure the safe, efficient and orderly operation of robots in industrial environments. Since there are not only obstacles in this hypothetical environment, but also dynamic mutual interference between robots, finding the optimal path becomes the core difficulty of the system.

DOI: 10.54254/2755-2721/2026.36919

Traditional robot path planning methods can be mainly classified into artificial potential field algorithm (APF) [3], A* algorithm [4], fast exploration random tree (RRT) [5], Dijkstra algorithm [6], etc. However, when the number of robots increases or environmental obstacles become more complex and dynamic, traditional methods often face problems such as exponentially increasing computational complexity, easy getting trapped in local minima, or inability to effectively handle multi-robot coordinated collision avoidance.

In order to overcome the limitations of traditional methods, in recent years, metaheuristic optimization algorithms have been widely used in the field of path planning due to their prominent advantages such as not requiring gradient information and strong robustness. For example, Particle Swarm Optimization (PSO) [7], Genetic Algorithm (GA) [8], and Grey Wolf Optimization (GWO) [9] have all achieved significant results in two-dimensional or three-dimensional path optimization. PSO, which simulates the bird flocks foraging based on population; GA, based on evaluation, which preserves the best genetic individuals over generations, mimics the process of biological evolution; GWO, which is derived from the grouped hunting process of wolves, emulates the hierarchy phenomenon observed and hunting mechanism of grey wolves. However, according to the "No Free Lunch" (NFL) [10] theorem, no single algorithm can perfectly solve all optimization problems. When facing high-dimensional, multi-constraint multi-robot dynamic path planning, a single metaheuristic algorithm is often prone to bottlenecks such as loss of population diversity and getting trapped in local optima.

The Coati Optimization Algorithm (COA) is a novel swarm intelligence optimization algorithm based on biological behavior proposed in 2022. It demonstrates excellent optimization ability by simulating the behavior of coatis hunting iguanas and escaping predators. However, when solving multi-robot multi-objective obstacle avoidance problems, the original COA has the defects of premature convergence, decreased population diversity, and inability to directly solve multi-objective trade-off problems.

2. Relevant theories

2.1. Multi-robot path planning problem

Consider a multi-robot system consisting of M mobile robots navigating in a two-dimensional grid workspace bounded by $X_{min}, X_{max}, Y_{min}, Y_{max}$. Let R_i denote the i -th robot ($i = 1,2,\ldots,M$). The continuous path of robot R_i is discretized into N control waypoints, as given in Eq. (1):

$$P_i = \{(x_{i,1}, y_{i,1}), (x_{i,2}, y_{i,2}), \ldots, (x_{i,N}, y_{i,N})\} \tag{1}$$

where $(x_{i,1}, y_{i,1}) = S_i$ is the designated start point and $(x_{i,N}, y_{i,N}) = T_i$ is the target destination of robot R_i . The remaining $N-2$ intermediate waypoints represent decision variables optimized by the algorithm.

The total path length $f_{len}(P_i)$ for robot R_i is computed using the Euclidean distance between consecutive waypoints, as given in Eq. (2):

$$f_{len}(P_i) = \sum_{k=1}^{N-1} \sqrt{(x_{i,k+1} - x_{i,k})^2 + (y_{i,k+1} - y_{i,k})^2} \tag{2}$$

To guarantee physical safety, a static obstacle penalty $f_{obs}(P_i)$ and an inter-robot collision penalty $f_{coll}(P_i, P_j)$ are defined. The static obstacle penalty penalizes waypoints that fall within a

DOI: 10.54254/2755-2721/2026.36919

safety threshold d_{safe} from any static obstacle O_m, as given in Eq. (3):

$$f_{obs}(P_i) = \sum_{k=1}^{N} \sum_{m=1}^{N_{obs}} \max(0, d_{safe} - \text{dist}((x_{i,k}, y_{i,k}), O_m)) \times w_{obs} \tag{3}$$

where w_{obs} is a large penalty weight coefficient. Similarly, to prevent inter-robot collisions at corresponding normalized trajectory time steps k, the inter-robot collision penalty is given by Eq. (4)

$$f_{coll}(P_i, P_j) = \sum_{k=1}^{N} \max(0, D_{min} - \text{dist}((x_{i,k}, y_{i,k}), (x_{j,k}, y_{j,k}))) \times w_{coll} \tag{4}$$

where D_{min} is the minimum allowed clearance distance between any two robots R_i and R_j ($i \neq j$), and w_{coll} is the collision penalty weight.

The comprehensive objective function $F(X)$ for the entire multi-robot team is expressed by Eq. (5) as:

$$F(X) = \sum_{i=1}^{M} f_{len}(P_i) + \sum_{i=1}^{M} f_{obs}(P_i) + \sum_{i=1}^{M} \sum_{j=i+1}^{M} f_{coll}(P_i, P_j) \tag{5}$$

2.2. Coati Optimization Algorithm (COA)

The Coati Optimization Algorithm (COA) is a novel metaheuristic algorithm inspired by the foraging and survival behaviors of coatis in nature, proposed by Dehghani et al. in 2022. Its optimization process mainly consists of two basic stages, as shown in Figure 1.

(1) Phase 1: Attacking and Hunting Iguanas (Exploration)

The coati population performs a global search of the solution space by simulating the process of climbing trees to scare prey and approaching them on the ground. Initially, half of the coatis climb up the trees to get approach to the prey, iguana, which is represented by the mathematical model: assuming the optimal individual position in the current population (i.e., the location of the iguana) is Ig, the formula for updating the individual's position is given by Eq. (6):

$$x'_{i,j} = x_{i,j} + r_1 \cdot (Ig_j - K \cdot x_{i,j}) \tag{6}$$

In the equation, $x_{i,j}$ represents the position of the i th coati in the j th dimension, r_1 is a random number within the range of [0,1], $K \in \{1,2\}$ is a randomly selected discrete inertia weight. The purpose of this stage is to force the population to quickly move towards the currently discovered optimal region.

After the iguana lands on the ground, it is assigned a randomly generated position within the search space. This position serves as a reference for the movement of the coatis on the ground, whose positions are updated according to Eq. (7) and Eq. (8)

$$Ig_j^G = lb_j + r_2 \cdot (ub_j - lb_j) \tag{7}$$

$$x'_{i,j} = \begin{cases} x_{i,j} + r_3 \cdot \left(Ig_j^G - K \cdot x_{i,j}\right), & \text{if } F_{Ig^G} < F(X_i) \\ x_{i,j} + r_3 \cdot \left(x_{i,j} - K \cdot Ig_j^G\right), & \text{otherwise} \end{cases} \tag{8}$$

DOI: 10.54254/2755-2721/2026.36919

For the second half of the population ($i = \lfloor N/2 \rfloor + 1, \ldots, N$), each coati updates its position based on the randomly generated iguana position. If the fitness of the ground iguana is superior to that of the current coati ($F_{Ig^G} < F(X_i)$), the coati moves toward the iguana. Otherwise, it explores the search space by moving away from the iguana, as described in Eq. (8).

After generating the candidate solution, a greedy selection strategy is employed. The updated position is accepted only if it yields a better objective function value; otherwise, the previous position is retained, as expressed in Eq. (9)

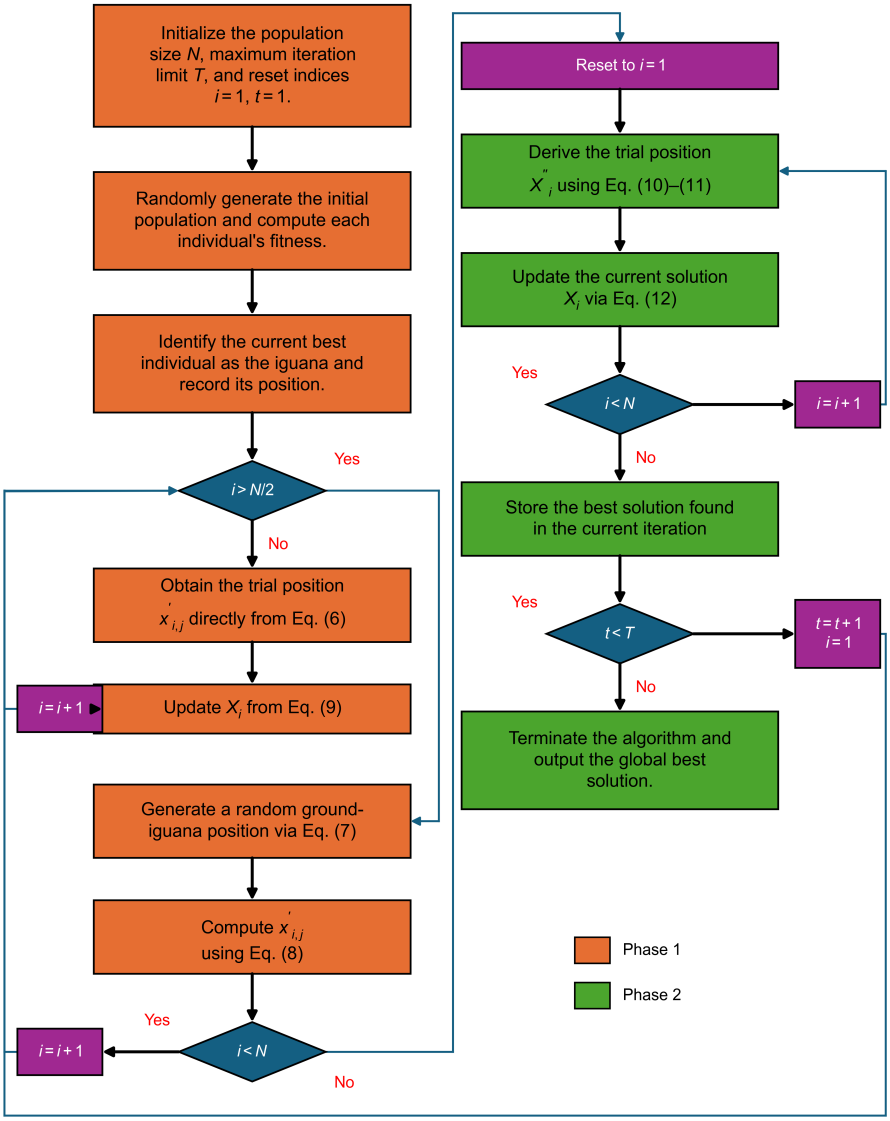

Figure 1. Flowchart of COA

$$X_i = \begin{cases} X_i^{'}, & \text{if } F\left(X_i^{'}\right) < F\left(X_i\right) \\ X_i, & \text{otherwise} \end{cases} \tag{9}$$

Here, $X_i^{'}$ denotes the candidate position of the i -th coati, while $x_{i,j}^{'}$ represents its j -th decision variable. The corresponding objective value is denoted by $F(X_i)$. The variable Ig refers to the current best individual in the population, whereas Ig^G indicates the randomly generated ground

DOI: 10.54254/2755-2721/2026.36919

iguana position. The parameter K is randomly selected from the set $\{1,2\}$, and $\lfloor \cdot \rfloor$ represents the floor operator.

(2) Phase 2: Escaping Predators (Exploitation)

When a predator attacks a coati, it responds by escaping from its current position. In this strategy, the coati moves toward a nearby safe position, resulting in a local change around its current location. This behavior enhances the exploitation capability of the COA by promoting an intensive search within the local neighborhood, as simulated by Eq. (10) and Eq. (11)

$$lb_j^L = \frac{lb_j}{t}, \quad ub_j^L = \frac{ub_j}{t}, \; t = 1, \; 2, \; 3, \ldots, T \tag{10}$$

$$x'_{i,j} = x_{i,j} + (1 - 2r_4) \cdot (lb_j^L + r_5 \cdot (ub_j^L - lb_j^L)) \tag{11}$$

Where t represents current iteration number and T is the maximum iteration number; lb_j and ub_j denotes the lower and upper bound of the j-th decision variable respectively, lb_j^L and ub_j^L are the local search boundaries that shrink dynamically with the number of iterations, r_4, r_5 are both random numbers between $[0,1]$.

The newly generated position is accepted only when it results in an improved objective function value, according to Eq. (12):

$$X_i = \begin{cases} X_i'', & \text{if } F(X_i'') < F(X_i) \\ X_i, & \text{otherwise} \end{cases} \tag{12}$$

3. Experiment results and analysis

3.1. Experimental configuration

This section describes the simulation environment established for the robot path planning problem [11]. All mobile robots are modeled as disks with radius r_R = 1.0 map unit. Dynamic obstacles are circular with radius r_D = 2.0 map units and move at a constant speed v_D = 0.5 map units/step along straight trajectories from their initial positions to their assigned destinations. Static obstacles retain the circular or rectangular geometry shown in Figure 2; the circular static-obstacle radii range from 2.0 to 5.0 map units, whereas rectangular obstacles are checked using their exact boundaries rather than an equivalent radius. Collision tests use an additional safety margin d_{safe} = 1.0 map unit, giving a minimum robot-to-robot center separation D_{min} = 3.0 map units. For COA, the population size is N = 30 and the maximum number of optimization iterations is T = 1000. The obstacle and inter-robot penalty weights are w_{obs} = 1000 and w_{coll} = 2000, respectively. Each COA run retains the best feasible solution at T, while robot-motion simulation terminates when all robots reach their destinations. For the A* baseline, the workspace is discretized at a resolution of 1.0 map unit per cell. An 8-connected neighborhood is used, and the Euclidean distance to the goal is used as the admissible heuristic. Moreover, static obstacles are inflated by $r_R + d_{safe}$ before the graph search. For a dynamic obstacle, its center position is predicted from the constant-velocity model at the candidate node arrival time; a successor is rejected if the robot disk violates the prescribed clearance from the predicted obstacle disk. Thus, the A* baseline accounts for the same dynamic-obstacle trajectories and geometric safety margin used by COA. MATLAB R2024a was used on a 64-bit Windows 11 computer with 32.0 GB RAM and a processor with a 1.70 GHz base frequency.

DOI: 10.54254/2755-2721/2026.36919

3.2. Simulation experiment in a specific scenario

The layout of the simulation scenario is presented in Figure 2. The black trajectories denote the planned paths of the robots, while the blue trajectories indicate the movement paths of the dynamic obstacles. The symbol '×' represents the corresponding destination point of each object. 12 Robots, 9 static obstacles and 5 dynamic obstacles are assigned randomly on the plate, which places considerably significant differences on path smoothness between the algorithms. The number of steps and distance taken by each robot are reported in Table 1 and Table 2, respectively.

Table 1. Number of steps traveled by each robot

	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12
A*	26	79	87	32	54	63	49	28	47	80	46	8
COA	22	76	82	18	49	53	49	24	42	73	43	7

Table 2. Travel distance by each robot

	R1	R2	R3	R4	R5	R6	R7
A*	28.7859	89.4619	96.1867	34.7952	61.505	70.1219	56.0996
COA	24.4902	88.9686	92.9074	21.018	57.6211	59.4752	56.2805
	R8	R9	R10	R11	R12	Total distance	
A*	31.2604	52.634	89.6677	51.5114	8.8703	670.9	
COA	27.6153	48.2274	83.8832	49.3776	7.971	617.8355	

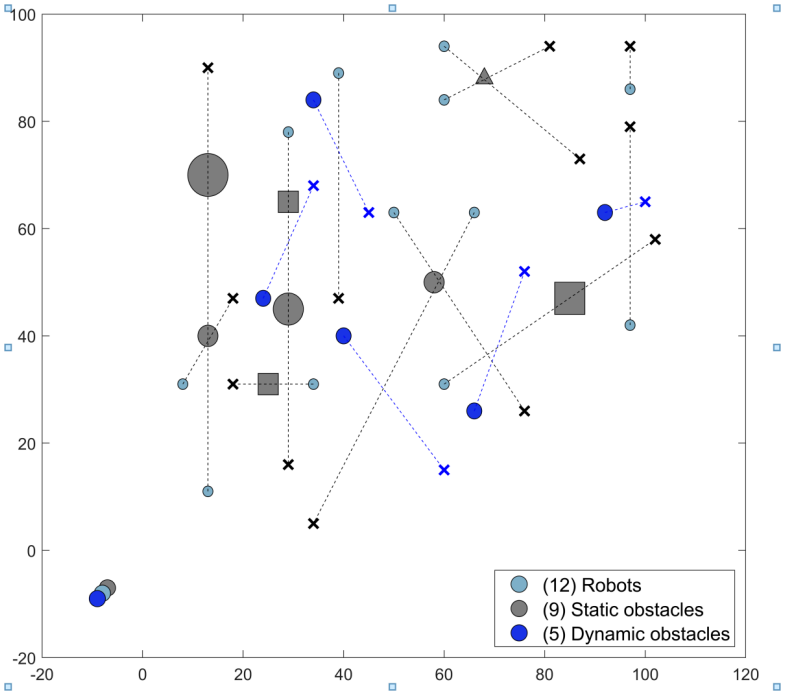

Figure 2. The simulation scenario

DOI: 10.54254/2755-2721/2026.36919

From Figure 3, it can be clearly observed that compared to the A* algorithm, COA presents smoother path with fewer twists and curves respectively. Overall, it displays a higher efficiency and a more straightforward solution to a given complex multi-path planning problem. Figure 4 illustrates the histogram of average travel distances/steps achieved by the different algorithms. The graph demonstrates that COA achieves a substantially lower average travel distance compared to the competing algorithms.

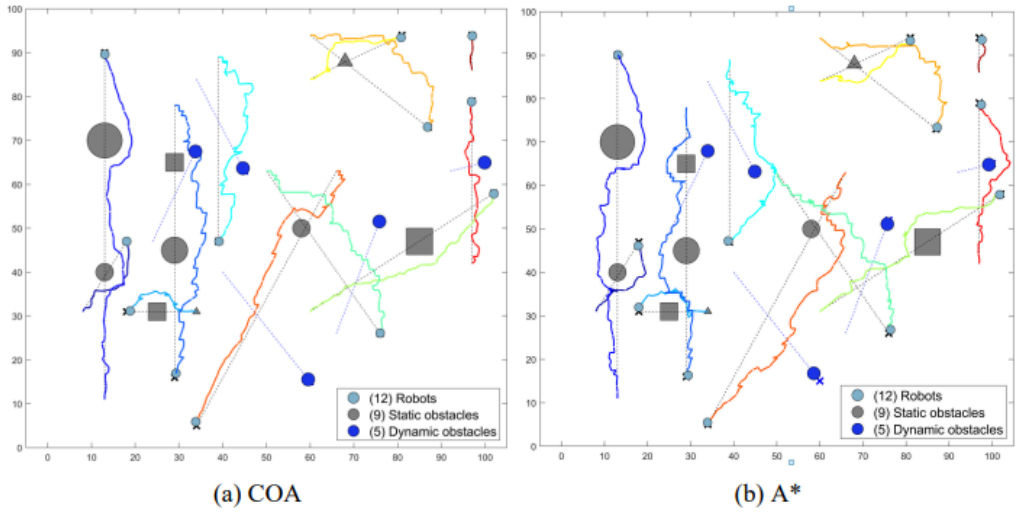

Figure 3. Paths of different algorithms

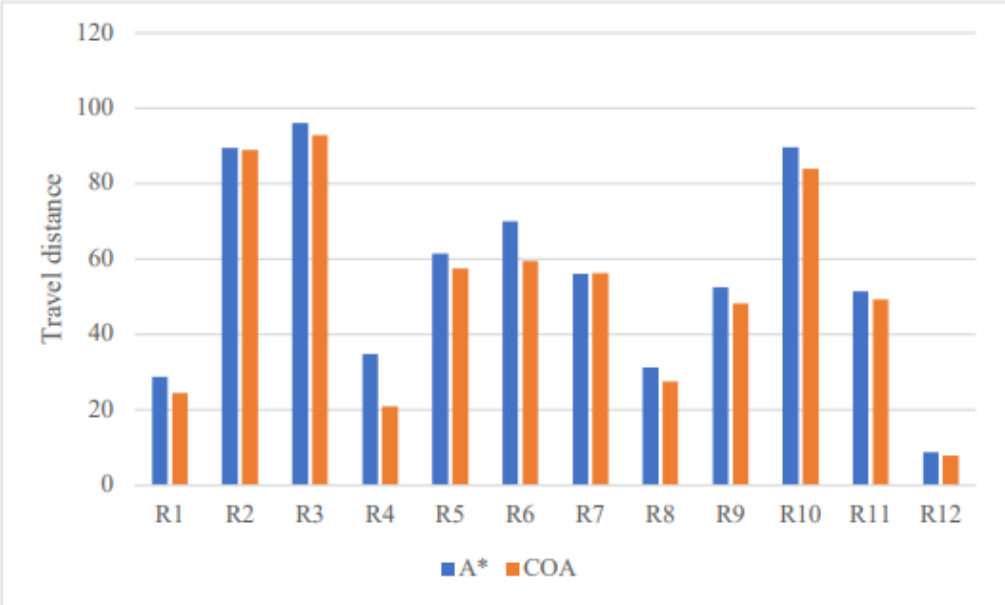

Figure 4. Histogram of travel distance

4. Conclusion

In this study, the Coati Optimization Algorithm (COA) was successfully applied to address the multi-robot path planning problem in complex environments containing static and dynamic obstacles. By modeling the trajectory optimization problem with a comprehensive cost function that

DOI: 10.54254/2755-2721/2026.36919

incorporates path length, obstacle clearance, and inter-robot collision avoidance, the COA framework effectively overcomes the discrete grid limitations and abrupt turning angles inherent in conventional search methods. Experimental comparisons against the traditional A* algorithm demonstrate the clear superiority of the bio-inspired metaheuristic approach. Specifically, the proposed COA-based path planner reduced the average total travel distance across the multi-robot fleet by approximately 7.9%, while simultaneously generating smoother, collision-free continuous trajectories and resolving potential spatial deadlocks.

In line with the No Free Lunch (NFL) theorem [10], no single algorithm can universally optimize all path planning domains; however, metaheuristic search mechanisms like COA offer strong global exploration and local refinement capabilities in continuous search spaces. Future research will focus on the following directions. First, the COA-based path planning framework will be extended to higher-dimensional environments, particularly three-dimensional spaces containing complex static obstacles and high-density dynamic agents. Second, the current cost function will be expanded to support multi-objective optimization, enabling a better balance among path length, energy consumption, execution time, and kinodynamic smoothness constraints. Finally, efforts will be directed toward real-world deployment by transferring the proposed algorithm from simulated environments to physical multi-AGV and mobile robot fleets, thereby validating its real-time performance under practical communication constraints and sensor uncertainties.

References

[1] Rubio, F., Valero, F., & Llopis-Albert, C. (2019). A review of mobile robots: Concepts, methods, theoretical framework, and applications. *International Journal of Advanced Robotic Systems, 16*(2), 1729881419839596. https://doi.org/10.1177/1729881419839596

[2] Zafar, M. N., & Mohanta, J. C. (2018). Methodology for path planning and optimization of mobile robots: A review. *Procedia Computer Science, 133*, 141-152. https://doi.org/10.1016/j.procs.2018.07.018

[3] Hu, L., Wei, C., & Yin, L. (2025). Fuzzy A* quantum multi-stage Q-learning artificial potential field for path planning of mobile robots. *Engineering Applications of Artificial Intelligence, 141*, 109866. https://doi.org/10.1016/j.engappai.2024.109866

[4] Huang, J., Chen, C., Shen, J., Liu, G., & Xu, F. (2025). A self-adaptive neighborhood search A-star algorithm for mobile robots global path planning. *Computers and Electrical Engineering, 123*, 110018. https://doi.org/10.1016/j.compeleceng.2024.110018

[5] Lei, S., Li, T., Gao, X., Xue, P., & Song, G. (2025). Research on improved RRT path planning algorithm based on multi-strategy fusion. *Scientific Reports, 15*(1), 13312. https://doi.org/10.1038/s41598-025-92675-5

[6] Ahmad, J., & Ab Wahab, M. N. (2025). Enhancing the safety and smoothness of path planning through an integration of Dijkstra's algorithm and piecewise cubic Bezier optimization. *Expert Systems with Applications, 289*, 128315. https://doi.org/10.1016/j.eswa.2025.128315

[7] Xu, H., Deng, Q., Zhang, Z., & Lin, S. (2025). A hybrid differential evolution particle swarm optimization algorithm based on dynamic strategies. *Scientific Reports, 15*(1), 4518. https://doi.org/10.1038/s41598-024-82648-5

[8] García-Martínez, C., Rodriguez, F. J., & Lozano, M. (2025). Genetic algorithms. In *Handbook of heuristics* (pp. 627-662). Springer Nature Switzerland. https://doi.org/10.1007/978-3-032-00385-0_28

[9] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69*, 46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007

[10] Wu, J. C., Ye, Q., Deng, D. L., & Yu, L. W. (2025). No-free-lunch theorems for tensor network machine learning models. *Physical Review Letters, 135*(22), 227301. https://doi.org/10.1103/by9g-5bkm

[11] Akay, R., & Yildirim, M. Y. (2023). Multi-strategy and self-adaptive differential sine-cosine algorithm for multi-robot path planning. *Expert Systems with Applications, 232*, 120849. https://doi.org/10.1016/j.eswa.2023.120849