International Journal of Engineering Approaches

Araştırma Makalesi / Research Article

A composite objective function specifically tuned for multi-robot path planning

Çoklu robot yol planlaması için özel olarak ayarlanmış bileşik bir amaç fonksiyonu

Bilal Özak[1]*, Mustafa Gül[2]

[1] Amasya University, Department of Mechanical Engineering, Amasya, Turkey

[2] Turkish Aerospace Industries, Ankara, Turkey

* *Sorumlu Yazar / Corresponding Author:* bilal.ozak@amasya.edu.tr

Makale Bilgileri / Article Info

Keywords
Swarm robotics
Path planning
Collision avoidance
Meta-heuristic algorithms

Anahtar Kelimeler
Sürü robotlar
Yol planlama
Çarpışmadan kaçınma
Meta-sezgisel algoritmalar

Makale tarihçesi / Article history
Geliş / Received: 16.09.2025
Düzeltme / Revised: 08.11.2025
Kabul / Accepted: 24.11.2025

Abstract

In swarm robotic systems, the problem of multi-robot path planning (MRPP), especially in obstacle-filled environments, presents significant challenges in terms of coordinated navigation. This work proposes a new fitness function for online MRPP in environments containing dynamic obstacles. The proposed method optimizes the collision-free paths of 20 robots using metaheuristic algorithms such as Particle Swarm Optimization (PSO), Artificial Bee Colony Optimization (ABC), Ant Colony Optimization (ACO), Genetic Algorithm (GA), and Differential Evolution (DE). The fitness function balances conflicting objectives such as proximity to the target, obstacle, and collision avoidance with other robots. Simulations have confirmed that 20 robots divided into two groups safely reach their destinations in a 100x100 unit environment containing dynamic and static obstacles. Simulations showed that the ABC algorithm achieved the best average path length (2667.01 units) in static environments, while PSO provided the fastest computation time (13.15 s). In dynamic environments, ABC again outperformed others in path length (2790.13 units), and PSO remained the fastest (17.30 s). The contributions of the work are a new fitness function, a path planning framework that improves the efficiency of metaheuristic algorithms, and demonstration of the success of the method in dynamic environments.

Öz

Sürü robot sistemlerinde, özellikle engellerle dolu ortamlarda çoklu robot yol planlama (ÇRYP) problemi, koordineli navigasyon açısından önemli zorluklar ortaya koymaktadır. Bu çalışma, dinamik engeller içeren ortamlarda çevrimiçi ÇRYP için yeni bir uygunluk fonksiyonu önermektedir. Önerilen yöntem, Parçacık Sürü Optimizasyonu (PSO), Yapay Arı Kolonisi Optimizasyonu (ABC), Karınca Kolonisi Optimizasyonu (ACO), Genetik Algoritma (GA) ve Diferansiyel Evrim (DE) gibi metasezgisel algoritmalar kullanarak 20 robotun çarpışmadan kaçınan yollarını optimize etmektedir. Uygunluk fonksiyonu, hedefe yakınlık, engel ve diğer robotlarla çarpışmadan kaçınma gibi çatışan hedefleri dengelemektedir. Simülasyonlar, iki gruba ayrılan 20 robotun, dinamik ve statik engeller içeren 100x100 birimlik bir ortamda hedeflerine güvenli bir şekilde ulaştığını doğrulamıştır. Simülasyonlar, ABC algoritmasının statik ortamlarda en iyi ortalama yol uzunluğunu (2667,01 birim) elde ettiğini, PSO'nun ise en hızlı hesaplama süresini (13,15 sn) sağladığını göstermiştir. Dinamik ortamlarda ise ABC, yol uzunluğu açısından (2790,13 birim) yine diğerlerinden daha iyi performans göstermiş ve PSO en hızlı algoritma olmaya devam etmiştir (17,30 sn). Çalışmanın katkıları, yeni bir uygunluk fonksiyonu, metasezgisel algoritmaların verimliliğini artıran bir yol planlama çerçevesi ve yöntemin dinamik ortamlarda başarısının gösterilmesidir.

1. Introduction

In recent years, robotic technologies have gained an important place in both daily life and industrial automation [1]. Compared to a single robot, multi-robot systems offer increased durability and reliability, with the ability to solve more complex problems through collaboration and interaction within the team [2]. These advantages have made multi-robot systems common in areas such as smart warehousing [3] and oilfield inspection [4]. Multi-robot path planning (MRPP) is a core component of these systems and aims to find collision-free paths from robots' initial positions to their targets [5].

Path planning algorithms are generally classified as classical, graph-based and heuristic approaches [6]. Classical methods (e.g., artificial potential field, roadmap), while initially effective, are time-consuming in complex or dynamic environments and may fall into the local optima trap [7]. Graph-based methods (Dijkstra, A*) are successful on regular maps, but the computational burden increases in complex scenarios [8]. These limitations have increased the importance of heuristic approaches, especially meta-heuristic algorithms. Meta-heuristic algorithms provide flexible and efficient solutions inspired by nature, overcome the problem of local optima, and are widely used in path planning problems [9].

Optimization problems are common in science, engineering, and industry, and often involve finding the best solution among a large number of possible solutions. While traditional optimization techniques are effective for well-defined problems, they often fall short when faced with complex, nonlinear, or high-dimensional challenges. To address such challenges, metaheuristic algorithms inspired by natural processes have emerged as powerful tools. Some of the most prominent among these are Particle Swarm Optimization (PSO), Artificial Bee Colony Optimization (ABC), Ant Colony Optimization (ACO), Genetic Algorithm (GA), and Differential Evolution (DE) [10-14]. Each of these iteratively improves candidate solutions by drawing inspiration from the collective behavior of bird flocks, the foraging strategies of bees and ants, the principles of natural selection, or the mathematical elegance of vector differences.

In this study, a new fitness function is proposed for the online MRPP problem with dynamic obstacles. The proposed method optimizes the collision-free paths of robots using meta-heuristic algorithms (PSO, ABC, ACO, GA, DE). The fitness function balances the path length and safety by taking into account the distance to the target, the safe distance from obstacles and other robots. The effectiveness of the proposed method is verified by simulations involving 20 robots. The contributions of the study are as follows:

- A new fitness function for online MRPP.
- A path planning approach that improves the efficiency of metaheuristic algorithms.
- Demonstration of the method's success with simulations in dynamic and complex environments.

In the rest of the article, the proposed method, environment description, simulation results and conclusion sections are presented.

2. Multi-robot path planning method

In this study, a method based on metaheuristic algorithms is proposed for the problem of online MRPP in an environment with dynamic and static obstacles. The proposed method aims to ensure that 20 robots are divided into two groups (10 robots in each group) reach their targets from their initial positions with collision-free paths. The method allows the flexible use of different metaheuristic algorithms (Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Ant Colony Optimization (ACO), Genetic Algorithm (GA) and Differential Evolution (DE)) and optimizes the safety and efficiency of paths with proposed fitness function.

Algorithm 1 summarizes the proposed method using metaheuristic algorithms for MRPP. The algorithm starts by taking inputs such as the number of robots, start and target points, environment map, robot size, safety distance, and search radius. First, the parameters and robot positions are initialized. Then, a loop runs until all robots reach their destinations: for each active robot, the next position is calculated using metaheuristic algorithms such as PSO, ABC, ACO, GA, or FA; the robot positions are updated. The process ends when all robots reach their destinations. This method provides a flexible framework for generating collision-free and efficient paths in environments with dynamic and static obstacles.

Algorithm 1 Multi-robot path planning with metaheuristic algorithms

```
Input Number of robots, Start points, Goal points, Map, Robot size, safety distance, Search radius
Initialize Parameters and Robots
repeat
    if All robots reached goal, then
        break repeat
    end if
    For each active robot
        Calculate next position with Metaheuristics
    End for
    update robot positions
until all the robots reach the goal positions
```

2.1. The objective of MRPP

In the Metaheuristic based path planning problem, a fitness function is defined to evaluate the quality of each individual (potential solution). This function quantitatively measures how desirable a position P = (x, y) is. The goal is to maximize fitness, which is typically formulated as the inverse of a cost function. The fitness function used in this work has a composite structure that combines multiple, sometimes conflicting, objectives into a single scalar value: proximity to the target, obstacle avoidance, and collision avoidance with other robots.

There are three main cost components that form the basis of the fitness function:

1. Target Oriented Cost (F_1)
2. Obstacle Violation Penalty (F_2)
3. Robot Interaction Penalty (F_3)

The Target Oriented Cost component measures the separation between the evaluated position $P_i^e = (x_i^e, y_i^e)$

and the robot's assigned target position $P_n^t = (x_n^t, y_n^t)$ with the most basic and common metric, the Euclidean distance.

$$F_1 = \sqrt{(x_i^e - x_n^t)^2 + (y_i^e - y_n^t)^2} \quad (1)$$

Minimizing this term guides the robot towards its goal. The Obstacle Violation Penalty term is designed to prevent the robot from coming dangerously close to or colliding with static or dynamic obstacles. Dynamic and static obstacles are modeled as circular regions with centers $C_k^o = (x_k^o, y_k^o)$ and radius r_k^o, with index k. The distance between the center of each obstacle and the robot's position point is calculated by the following Eq 2.

$$F_2 = \min_{k=1,2,\ldots,N^o} \left(\sqrt{(x_i^e - x_k^o)^2 + (y_i^e - y_k^o)^2} - r_k - d_{min}\right) \quad (2)$$

d_{min} is the minimum safety distance to be maintained between robots and obstacles, and r_k is the radius of the kth obstacle and N^o represents number of obstacles.

$$F_3 = \begin{cases} \infty & if\ N^R = 1 \\ \min_{j=1,2,\ldots,N^R} \left(\sqrt{(x_i^e - x_j^r)^2 + (y_i^e - y_j^r)^2}\right) & other \end{cases} \quad (3)$$

While N^R represents the number of robots, (x_j^r, y_j^r) represents the positions of other robots and with this function we calculate the distance of the active robot to the nearest robot.

The fitness function is defined as:

$$F = \frac{1}{F_1 + w_1 max(0, -F_2) + w_2 \max(0, d_{min} - F_3)} \quad (4)$$

The overall workflow for each time step is:

Step 1: For each active robot (for those that have not yet reached their destination):

- Candidate locations are generated using the Metaheuristic Algorithm.
- The fitness of each candidate is evaluated.
- The candidate locations are updated iteratively according to the Metaheuristics.
- The location with the highest fitness that satisfies the collision constraints is selected.

Step 2: All robot locations are updated simultaneously according to the planned movements.

Step 3: Convergence to the destination is checked and robots that have reached their destination are disabled.

Step 4: These steps are repeated until all robots have reached their destination or until a maximum time limit is reached.

3. Environment description

This study addresses the problem of coordinated path planning of multiple mobile robots in a two-dimensional environment with both static and dynamic obstacles. The study area is defined as a Cartesian plane of [100] x [100] unit dimensions. In this area, there are 20 holonomic, 2-unit diameter circular footprint mobile robots. The robots are divided into two groups with different initial and target configurations. As seen in Figure 1 and 2, the static environment contains 6 static, circular obstacles, each with different centers and defined with different radius and the dynamic environment contains 6 static, circular obstacles and 4 dynamic circular obstacles. Obstacles are modeled as circular shapes to simplify collision detection calculations and to ensure consistent safety distance measurements across all directions. This is a common approach in multi-robot path planning literature to reduce computational complexity. Dynamic obstacles move linearly at constant speed and direction during the simulation. The motion parameters, given in Table 1, are predefined and remain unchanged across all algorithm tests to ensure consistency.

Table 1. Dynamic obstacle properties

Obstacle ID	Initial Position (x, y)	Radius (units)	Velocity (units/step)	Direction (degrees)	Motion Type
DO1	(10, 85)	6	0.7	0	Linear
DO2	(40, 70)	7	0.7	270	Linear
DO3	(45, 10)	5	0.6	135	Linear
DO4	(85, 50)	4	0.4	135	Linear

The locations and sizes of the obstacles are known at the beginning of the simulation. The robots are divided into two main groups. The first group consists of 10 robots, and they start from a 10-unit square area centered on the coordinates (5,5) at random positions. The common target of this group is the point (95,95). The second group consists of the remaining 10 robots, and they start from a 10-square-meter area centered on coordinates (5,95) at random positions. The target of this group is the point (95,5). The values of the weight factors w_1 and w_2 in the fitness function are given in Table 2 and the values were determined by empirical parameter tuning. Simulation environment parameters are presented in detail in Table 2.

The main objective is for each robot to find a safe path from its initial position to its assigned target position without colliding with the defined obstacles and other robots. A robot reaching its target is determined when the Euclidean distance to the target point is less than 4 units. The path planning process requires the robots to always maintain a safety distance of at least 2 units from both static obstacles and other robots. This problem is formulated as a sequential decentralized optimization problem where each robot makes decisions locally but considers the planned movements of the other robots.

Table 2. Simulation environment parameters

Symbol / Parameter Name	Value
Path Planning Parameters	
Total number of robots	20
Minimum safety distance	2 unit
Euclidean distance tolerance for target	5 unit
Map Dimensions	100 x 100 units
Search radius	3 unit
Objective Function Weights	
w_1 (Obstacle weight)	10
w_2 (Robot weight)	5

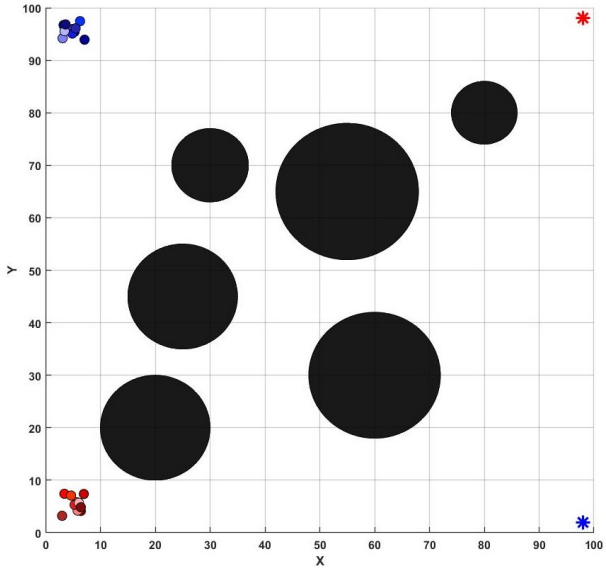

Figure. 1. Simulation environment 1 – Static obstacles

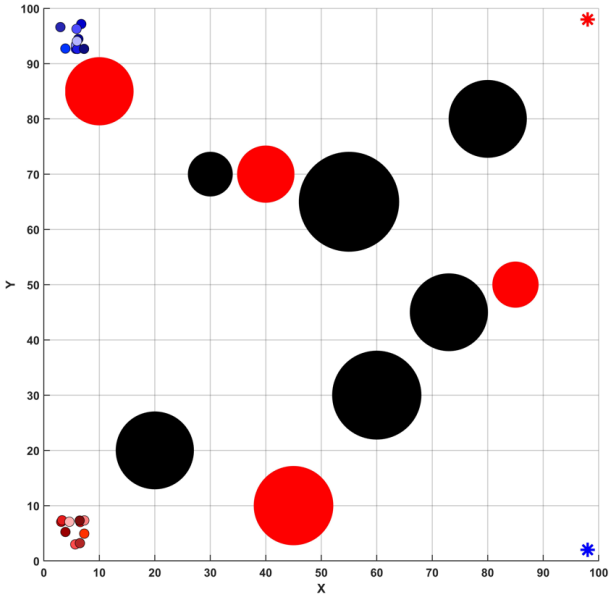

Figure. 2. Simulation environment 2 - Dynamic obstacles

4. Simulation Results

The proposed multi-robot path planning (MRPP) method was implemented in MATLAB R2019a to evaluate the performance of five metaheuristic algorithms—Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC), Ant Colony Optimization (ACO), Genetic Algorithm (GA), and Differential Evolution (DE)—in a dynamic environment with 6 static and 4 dynamic circular obstacles. Each algorithm was run 30 times independently. The simulation environment, detailed in Table 1, consisted of a 100x100 unit 2D Cartesian plane with 20 robots divided into two groups: Group 1 (10 robots) starting at a 10-unit square centered at (5,5) with a target at (95,95), and Group 2 (10 robots) starting at (5,95) with a target at (95,5). A minimum safety distance of 2 units and a target-reaching tolerance of 5 units were enforced. The algorithms were configured as shown in Table 3, with consistent population sizes (30) and maximum iterations (30) to ensure fair comparison. The simulations are conducted on the 64-bit operating system, AMD Ryzen 5 3600 6-Core Processor 3.59 GHz CPU and 16 GB memory.

Table 3. Parameters of algorithms

Algorithm	Parameter Name	Value
	Population Size	30
	Maximum Iterations	30
PSO	Inertia Weight (w)	0.729
	Cognitive Coefficient (c1)	1.49445
	Social Coefficient (c2)	1.49445
ABC	Limit	15
ACO	Archive Size	10
	Deposition Rate	0.1
	Evaporation Rate	0.85
GA	Crossover Rate	0.8
	Mutation Rate	0.1
	Mutation Strength	0.5
	Tournament Size	3
	Elitism Count	2
DE	Crossover probability	0.9
	Mutation factor	0.8

Table 4 summarizes the mean, standard deviation and best values of the algorithms in terms of path length and computational time in the static environment. In terms of path length, ABC algorithm exhibited the best average performance (2667.0061 units), while ACO algorithm produced the shortest path (2642.5750 units) in a single run. This shows that both algorithms are effective in optimizing collision-free paths. GA showed consistent performance with the lowest standard deviation (11.3331), while DE exhibited the weakest performance with the highest variability (16.8209) and the longest average path (2679.2178 units). PSO showed a moderate performance with an average path length of 2670.1083 units.

In terms of computational time, PSO achieved the lowest average time (13.1506 seconds) and the lowest standard deviation (0.1951), demonstrating high suitability for real-time applications. ABC was the second fastest algorithm with 14.0750 seconds, while GA exhibited the slowest performance (15.8023 seconds) but provided a stable computation with low standard deviation (0.2219). ACO and DE showed moderate computation times of 15.4019 and 13.9082 seconds, respectively.

Table 5 summarizes the performance of the algorithms in the dynamic environment. In terms of path length, ABC again achieved the best average performance (2790.1288

units), while PSO produced the shortest path (2765.5787 units) in a single run. GA exhibited the most consistent performance with the lowest standard deviation (11.8096), while DE performed poorly with the highest variability (18.5330) and longest average path (2823.2305 units). ACO exhibited a moderate performance with an average path length of 2817.1049 units.

In terms of computation time, PSO was again the fastest algorithm with an average of 17.3003 seconds and a standard deviation of 0.2648. ABC was the second fastest algorithm with 18.0388 seconds and a stable performance with a low standard deviation (0.2043). While ACO was the slowest algorithm with 20.7636 seconds, GA and DE provided moderate computation times with 19.7864 and 19.0721 seconds, respectively. GA stood out in terms of computational stability with low standard deviation (0.1970).

Table 4. Algorithms based on path length and computation time in the static environment

	Path Length (unit)			Computation Time (s)		
	Avarage	**Std**	**Best**	**Avarage**	**Std**	**Best**
PSO	2670.1083	12.3058	2642.6281	**13.1506**	**0.1951**	**12.7910**
ABC	**2667.0061**	**11.3331**	2648.4628	14.0750	0.2727	13.6256
ACO	2693.5924	18.5665	**2642.5750**	15.4019	0.2457	15.0164
GA	2677.4905	12.2238	2656.8456	15.8023	0.2219	15.2939
DE	2679.2178	16.8209	2643.9480	13.9082	0.3566	13.2117

Table 5. Algorithms based on path length and computation time in the dynamic environment

	Path Length (unit)			Computation Time (s)		
	Avarage	**Std**	**Best**	**Avarage**	**Std**	**Best**
PSO	2792.2862	11.9768	**2765.5787**	**17.3003**	**0.2648**	**16.9674**
ABC	**2790.1288**	14.4460	2765.9616	18.0388	0.2043	17.7654
ACO	2817.1049	14.5962	2792.9376	20.7636	0.5555	20.1897
GA	2797.9575	**11.8096**	2774.1813	19.7864	0.1970	19.4095
DE	2823.2305	18.5330	2787.1480	19.0721	0.5090	18.2630

In terms of computation time, ACO achieved the lowest mean (13.4794 s) and standard deviation (0.1569), demonstrating high suitability for real-time applications. PSO was the second fastest algorithm with a mean of 17.3003 s, while GA exhibited the slowest performance (19.7864 s) but offered a stable computation with low standard deviation (0.1970). ABC and DE showed moderate computation times of 18.0388 s and 19.0721 s, respectively; ABC was more stable with lower standard deviation (0.2043).

The simulation results are visualized with dynamic animations shown in Figure 3 and Figure 4. Figure 3 shows the robot trajectories of the PSO algorithm in a static environment, while Figure 4 shows the trajectories of the ABC algorithm in a dynamic environment. The trajectories are drawn in red tones for Group 1 and in blue tones for Group 2, with static obstacles shown in black and dynamic obstacles in red. These visualizations confirm that all algorithms guide the robots to their targets without collisions.

5. Conclusion

This study proposed a new fitness function for online multi-robot trajectory planning (MRPP) in a dynamic environment and optimized the collision-free paths of 20 robots using metaheuristic algorithms such as PSO, ABC, ACO, GA and DE. The contributions of the work are a composite fitness function that integrates target proximity, obstacle avoidance, and inter-robot collision penalties, a flexible framework that improves the efficiency of metaheuristic algorithms, and validation of the method's success in dynamic environments. The method is tested in MATLAB simulations with dynamic visualizations that verify the collision-free navigation of the robots.

These results have important implications for swarm robotics applications such as smart warehousing and autonomous control, where efficient and safe navigation is critical. However, the study is limited to a 2D environment and assumes that obstacle positions are known, which may not fully reflect real-world complexities. Furthermore, the computational cost of algorithms such as GA and DE may limit real-time applications in large-scale systems.

Future work can explore extending the method to 3D environments, integrating hybrid metaheuristic algorithms such as PSO-ACO combinations, and incorporating real-time sensor data for dynamic obstacle detection. Experimental validation on physical robots and scalability tests with larger robot fleets will increase the practical applicability of the method. This work provides a solid foundation for advancing MRPP in complex and dynamic environments and contributes to the field of autonomous multi-robot systems.

In general, ABC and PSO stand out as the most effective algorithms in terms of path length optimization in both static and dynamic environments. PSO offers a suitable option for real-time applications, especially by showing superior performance in terms of computation time. While GA showed consistent performance in terms of path length and computation time, ACO and DE gave more variable results. These results show that the proposed fitness function increases the efficiency of metaheuristic algorithms and provides collision-free navigation in dynamic environments.

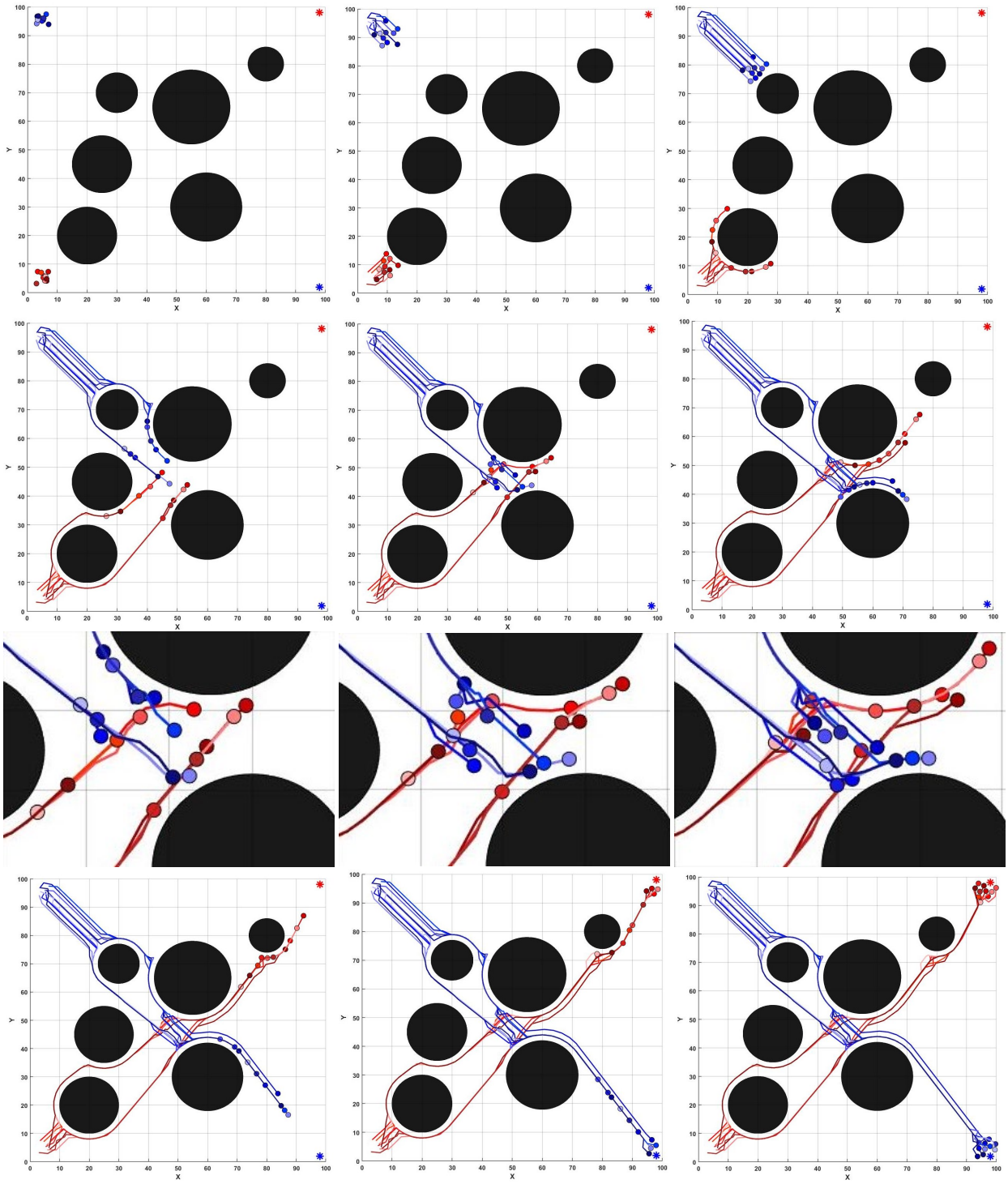

Figure. 3. MRPP process for twenty robots via PSO algorithm in the static environment.

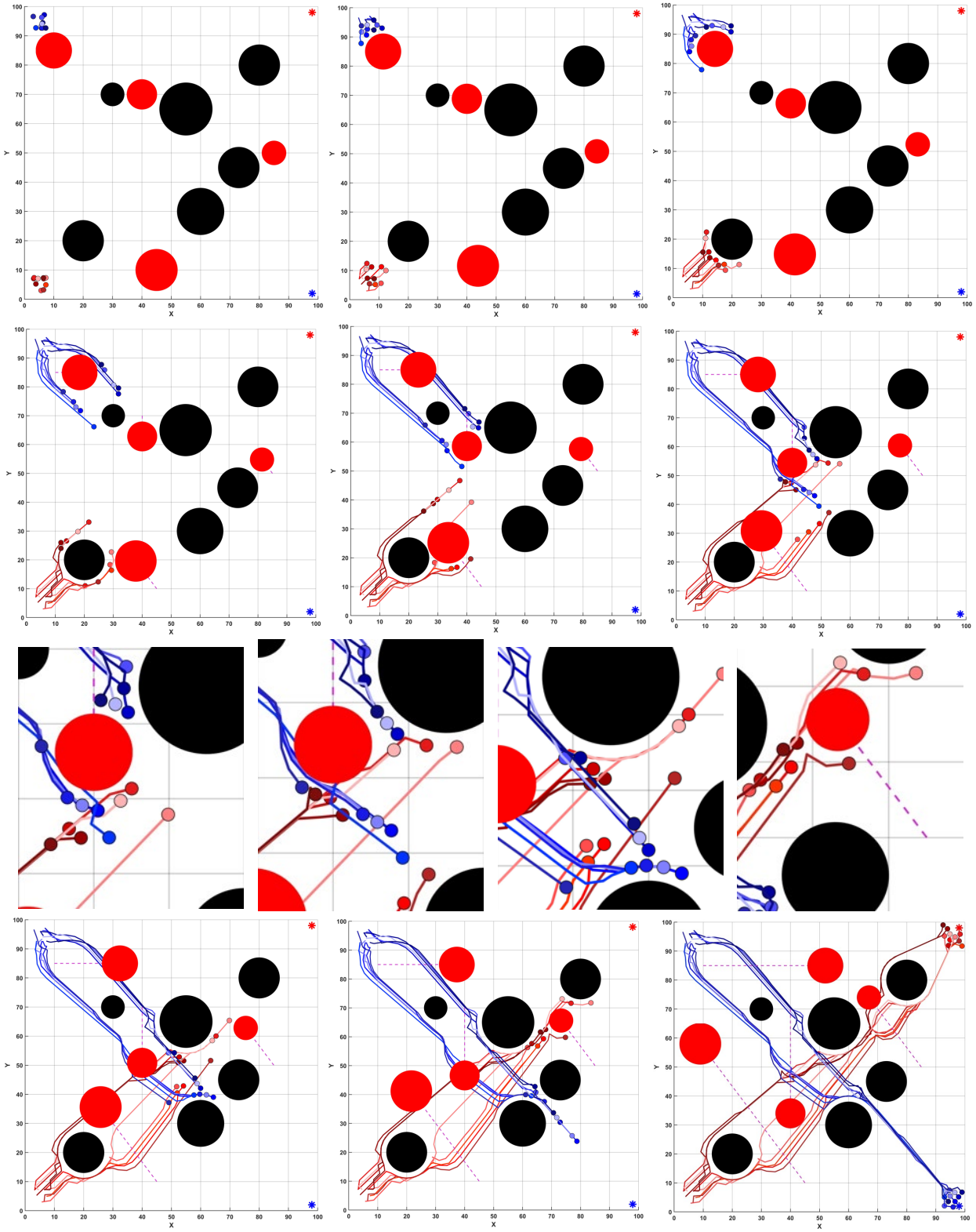

Figure. 4. MRPP process for twenty robots via ABC algorithm in the dynamic environment.

Conflict of Interest Statement: The author declares that there is no financial or personal conflict of interest related to this study. The study was conducted in an impartial manner and was not associated with any commercial organization, funder or third party.

Funding Information: This study was not supported by any funding.

Author Contributions: The authors confirm their responsibilities for the design of the study, data collection, analysis and interpretation of the results, and preparation of the manuscript.

Data Availability Statement: The data used in this study were generated through simulations performed in MATLAB R2019a. The simulation environment and parameters are described in detail in the "Environment Description" and "Simulation Results" sections of this article. Simulation codes are available upon request.

References

[1]. Goel, R. and Gupta, P. (2020) *Robotics and industry 4.0, A roadmap to industry 4.0: Smart production*, Sharp Business and Sustainable Development, 157-169.

[2]. Gielis, J., Shankar, A., and Prorok, A. (2022) *A critical review of communications in multi-robot systems*, Current Robotics Reports, 3(4): 213-225.

[3]. Bolu, A. and Korçak, Ö. (2021) *Adaptive task planning for multi-robot smart warehouse*, IEEE Access, 9: 27346-27358.

[4]. Yu, L., Yang, E., Ren, P., Luo, C., Dobie, G., Gu, D., and Yan, X. (2019) *Inspection robots in oil and gas industry: a review of current solutions and future trends. 2019* 25th International Conference on Automation and Computing (ICAC), Lancaster, UK, pp. 1-6.

[5]. Nazarahari, M., Khanmirza, E., and Doostie, S. (2019) *Multi-objective multi-robot path planning in continuous environment using an enhanced genetic algorithm*, Expert Systems with Applications, 115: 106-120.

[6]. Ugwoke, K.C., Nnanna, N.A., and Abdullahi, S.E.Y. (2025) *Simulation-based review of classical, heuristic, and metaheuristic path planning algorithms*, Scientific Reports, 15(1): 12643.

[7]. Tamizi, M.G., Yaghoubi, M., and Najjaran, H. (2023) *A review of recent trend in motion planning of industrial robots*, International Journal of Intelligent Robotics and Applications, 7(2): 253-274.

[8]. Chen, R. and Gotsman, C. (2021) *Efficient fastest-path computations for road maps,* Computational Visual Media, 7: 267-281.

[9]. Rahman, M.A., Sokkalingam, R., Othman, M., Biswas, K., Abdullah, L., and Abdul Kadir, E. (2021) *Nature-inspired metaheuristic techniques for combinatorial optimization problems: Overview and recent advances,* Mathematics, 9(20): 2633.

[10]. Kennedy, J. and Eberhart, R. (1995) *Particle swarm optimization*. Proceedings of ICNN'95-International Conference on Neural Networks, Perth, WA, Australia, Vol. 4, pp. 1942-1948.

[11]. Karaboga, D. and Basturk, B. (2007) *A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm*, Journal of Global Optimization, 39: 459-471.

[12]. Dorigo, M., Birattari, M., and Stutzle, T. (2007) *Ant colony optimization*, IEEE Computational Intelligence Magazine, 1(4): 28-39.

[13]. Holland, J.H. (1992) *Genetic algorithms*, Scientific American, 267(1): 66-73.

[14]. Price, K.V., Storn, R.M., and Lampinen, J.A. (2005) *Differential evolution: a practical approach to global optimization*, The Differential Evolution Algorithm, 37-134.