ATLANTIS PRESS

International Journal of Computational Intelligence Systems
Vol. **13(1)**, 2020, *pp.* 291–299
DOI: https://doi.org/10.2991/ijcis.d.200310.005; ISSN: 1875-6891; eISSN: 1875-6883
https://www.atlantis-press.com/journals/ijcis/

International Journal of Computational Intelligence Systems

Multi-Objective Particle Swarm Optimization Algorithm for the Minimum Constraint Removal Problem

Bo Xu[1,*], Feng Zhou[1], Antonio Marcel Gates[2]

[1] *School of Information Science, Guangdong University of Finance and Economics, Guangzhou, 510320, China*
[2] *Hawaii Pacific University, Honolulu, Hawaii, 96813, USA*

ARTICLE INFO

Article History

Received 27 Apr 2019
Accepted 03 Mar 2020

Keywords

Minimum constraint removal
Minimum constraint set
Path planning
Multi-objective optimization
Multi-objective particle swarm optimization algorithm

ABSTRACT

This paper proposes a multi-objective approach for the minimum constraint removal (MCR) A problem. First, a multi-objective model for MCR path planning is constructed. This model takes into account factors such as the minimum constraint set, the route length, and the cost. A multi-objective particle swarm optimization (MOPSO) algorithm is then designed based on the fitness function of the multi-objective MCR problem, and an iteration formula based on the personal best (pbest) and global best (gbest) of the algorithm is constructed to update the particle velocity and position. Finally, compared with ant colony optimization (ACO) A and the crow search algorithm (CSA) A, the experimental results show that the MOPSO-based path planning algorithm can find a shorter path that traverses fewer obstacle areas and can thus perform MCR path planning more effectively.

1. INTRODUCTION

Path planning is an important step for operating robots, robotic arms, and manipulators to accomplish predefined tasks; therefore, it is critical in robotic technologies [1]. The primary goal of path planning is to apply an analysis standard to find the shortest collision-free path for a robot from the initial state (including the position and posture) to the target state in an environment that contains obstacles [2]. Selecting the optimal path from the start position to the target position is the key issue in the early stage of robot path planning. To address this issue, most studies have focused on avoiding obstacles and have used global or local optimization to find the shortest obstacle-free path [3].

Not all robotic operational environments are obstacle free [4]. However, not all obstacles must be avoided. For example, in indoor navigation, the obstacle may be a closed door that a robot can open. In autonomous outdoor navigation, the obstacle may be an area of rough but not insurmountable terrain that a robot can pass through. However, in these situations, traversing the obstacle may be costly and risky. These two practical problems have sparked novel ideas in robot path planning research. In 2012, Hauser proposed a new path planning problem, which is called the minimum constraint removal (MCR) problem [5]. The goal is to find the minimum number of geometrical constraints that should be removed to connect the start state to the target state via a free path. This requires the removal of the minimum number of constraints (obstacles) to create an obstacle-free path in the environment. MCR path planning takes into account the possibility that the robot may not be able to find an obstacle-free path and attempts to find a path for the robot with minimal constraints. In this context, constraints are obstacles or areas of obstacles along the route. The significance of MCR path planning is that even if the robot must travel across constraints along a path, an effort should be made to ensure that the path traverses the minimum number of constraints. Once such a path has been found, whether by clearing the obstacles in the path or crossing areas of obstacles, higher efficiency will be achieved [6].

The MCR problem has recently become a popular research topic in the area of path planning. The MCR problem was initially proposed in 2012 by Hauser [5], who employed a method of reduction from the minimum set cover to prove that the discrete MCR problem is non-deterministic-polynomial-hard (NP-hard) A. He provided numerous new discrete theoretical MCR results and employed two search algorithms that have excellent practical performance, the exact search algorithm and the greedy search algorithm, to solve the discrete MCR problem. Hauser also demonstrated three use cases: the generation of a failure reason that is understandable by humans, motion planning in an uncertain environment, and the selection of a movable obstacle in a complicated environment [7]. The exact search algorithm and the greedy search algorithm take the minimum-obstacle constraint set as the target and pursue the optimal solution instead of satisfying a solution that meets the requirements at that time.

[*] *Corresponding author. Email: xubo807127940@163.com*

Erickson and LaValle used the reduction of the maximum satisfiability problem for the binary Horn clause to prove that the discrete MCR problem is NP-hard [8], but the Horn clause satisfiability method had very strict restrictions on the diagrams and obstacles obtained by transformation. Cohen focused on robotic arm path planning, constructed a discrete MCR model, and demonstrated that the discrete MCR problem is also NP-hard; he then employed a breadth-first algorithm that was based on graph-cut theory to find the path with minimum constraints to improve the robotic arm's operational efficiency [9], but this algorithm also aimed for the minimum-obstacle constraint set. Krontiris and Shome also leveraged graph theory to solve the MCR problem in robot path planning; the start and target positions were set to the graph's source point and sink point, respectively, and the maximum flow and minimum cut methods were employed to complete the MCR path planning [10], but the effect of their method was not very satisfactory. Sharon employed a multi-agent system intelligent algorithm to solve the multi-robot MCR path search problem, which took into account conflicts between individual robots and used a coordinated multi-agent mechanism to determine the path with minimum constraints [11]. However, their method relied on the learning ability of agents. Gorbenko and Popov investigated the discrete MCR problem in motion planning that could be used for path planning with an explanation for failure and an efficient approach to solving the problem. In particular, they investigated an explicit reduction of a decision version of the problem to a satisfiability problem [12], but the process for proving was very complicated. Krontiris and Bekris used the graph method to solve the rearrangement problem; set the starting position and the target position as the source point and the junction point of the graph, respectively; and proposed an approximate but significantly accelerated monotonic rearrangement method. This method defined the dependency graph between objects and allowed each object to reach its target by finding the MCR path [13]. Krontiris and Bekris also proposed a bounded length method to find a path of a certain length that could balance between minimum constraints, cost calculation, and route length [14,15], but they considered only cost and path length in the method. Bandyapadhyay proposed improved approximation bounds that gave an $O\left(\sqrt{n}\right)$ approximation for the MCR problem and proposed some inapproximability results for the geometric constraint removal problem [16]. Unfortunately, they did not conduct effective experimental testing.

Our research team believes that once discrete MCR problems have been confirmed to be NP-hard, the primary focus should shift to the use of various intelligent heuristic algorithms to solve these problems. Therefore, a highly effective approximate solution method should be developed. In our previous study, an ant colony optimization (ACO) algorithm was introduced to solve the MCR problem in motion planning; the algorithm achieved the desired goal because the solution quality and convergence speed were superior to those of the accurate search and greedy algorithms. For details, please refer to our previous research findings [17,18]. In the past, we worked from only a single target.

Current research on solving the MCR problem has made substantial progress; however, the main idea is to convert continuous MCR problems into discrete MCR problems and then use various methods to solve the discrete problems. To date, the researchers have used the greedy search algorithm, the exact search algorithm, and ACO to solve the MCR problem. However, these methods fail to achieve satisfactory results due to inherent defects and isolate the complex relationships among objects, robots, and obstacles in the process of solving MCR problems. For example, factors such as route length and cost are seldom addressed. However, in complex environments, the MCR problem is dynamic and multi-objective in nature; factors such as the minimum constraint set, route length, and cost should be considered. Therefore, this paper proposes a multi-objective approach for the MCR path planning problem.

2. PRELIMINARIES

2.1. Mathematical Model for MCR Path Planning

To facilitate this research, the MCR problem in robot path planning is described first. A typical MCR path planning scenario is shown in Figure 1.

In Figure 1, the robot starts at the start point in the lower-left corner and ultimately reaches the goal point in the top-right corner. This scenario contains seven obstacle areas, which are denoted as obstacles A, B, C, D, E, F, and G. The goal of MCR path planning is to plan a path from the start point to the goal point that traverses the minimum number of obstacles.

In Figure 1, the black dots represent nodes along each path where the robot can stay. All the nodes constitute a node network (Figure 2). In Figure 2, some of the nodes are labeled with obstacle names; for example, the label "B, G" means that the node is covered simultaneously by obstacles B and G. Some of the nodes are not labeled with obstacle names because they are not covered simultaneously by obstacles.

If the robot follows a path that contains node B, G, it must pass at least two obstacle areas, B and G. In this way, the MCR path planning problem is transformed into a problem to find a path that passes a series of nodes and is covered by the minimum number of obstacles in the node network (Figure 2).

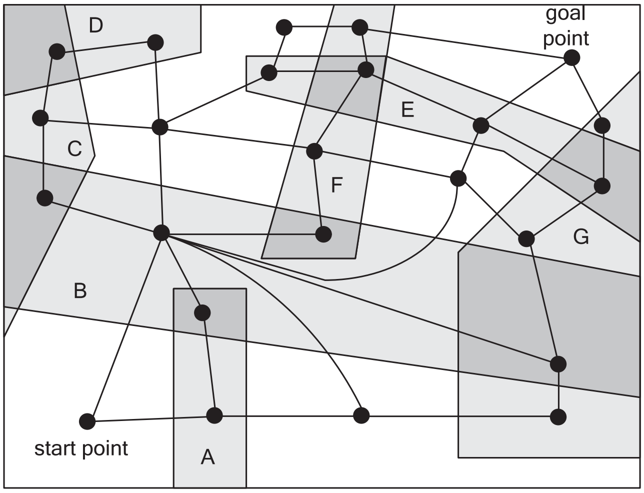

Figure 1 A typical scenario of MCR path planning.

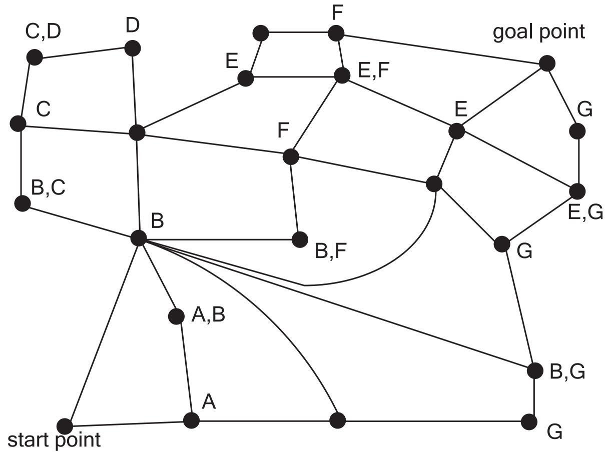

Figure 2 Node network for MCR path planning.

Next, the MCR path planning problem is described via a mathematical model. The mathematical model for **the continuous MCR problem** is defined as follows:

Input: It is assumed that all the nodes in the node network constitute a set, which is denoted as P^*. In this node set, p_s represents the node at the start position, and p_g represents the node at the target position; thus, $p_s, p_g \in P^*$. Assume that the entire space of the MCR path planning contains n obstacles $O_1, \ldots, O_n$ that constitute an obstacle set O^*. Then, $O_1, \ldots, O_n \subseteq O^*$.

Definition 1: A cover function $C(p) : P^* \rightarrow 2^{\{1,\ldots,n\}}$ is created, which is different from the obstacle set $p : C(p) = \left\{i|p \in O_i\right\}$. Next, a subset of obstacle set $\overline{O}$ is created. Thus, $\overline{O} \subseteq O^*$. Next, a cover function $C\left(\overline{O}\right)$ is created for this subset. Therefore, the cover function for this obstacle subset and $C(p)$ form a union, and $C(\overline{O}) = \left\{i|\overline{O} \cap O_i \neq \Phi\right\}$.

Definition 2: For a label subset $S \subseteq \{1, \ldots, n\}$, if there is a continuous path $R(s) : [0, 1] \rightarrow O^*$, then the start point p_s is said to have reachability to the goal point p_g. This path starts at point p_s, ends at point p_g, and satisfies $C(R(s)) \subseteq S, s \in [0, 1]$.

Output: The MCR set $S*$ is a path from q_s to q_g that traverses the minimum number of obstacles.

The discrete MCR problem is elaborated further via graph theory. The node set and the path that connects these nodes constitute a graph, which is represented as $G = (P^*, E)$. P denotes the node set, and E denotes the path that connects these nodes.

In addition, this graph is associated with the cover function $C(p)$, where $p \in P^*$. In $C(p)$, each node p is represented as a label set of covering obstacles. Hence, this graph is also associated with $\overline{O} \subseteq \{1, 2, \cdots, n\}$. $R(\overline{O}) : [0, 1] \rightarrow O^*$ is the final output result when the planning for this graph is complete.

The discrete MCR problem is defined as follows:

Input: The input includes the graph $G = (P^*, E)$, cover function $C[p], p \in P^*$, and start and goal vertexes $p_s, p_g \in P^*$. In $C[p]$, each vertex p is marked as a subset of the covering obstacles' label set $\{1, \ldots, n\}$, which means that vertex p is covered by these obstacles.

Output: S^* is a minimum subset of the label set $\{1, \ldots, n\}$ that satisfies the following condition: for any route R from p_s to p_g in graph G, every node p in the route satisfies $C[p] \subseteq S^*$.

Furthermore, if there is a label subset S that can satisfy the reachability from the start point p_s to the goal point p_g and the route contains the minimum number of obstacle labels, then there is a minimum-obstacle path from the start point p_s to the goal point p_g. There may be multiple minimum paths (because only a minimum of obstacles, and not the shortest distance, is mandated), and identifying the optimum path requires additional constraints.

2.2. Multi-Objective Optimization

A 3-tuple (X, C, F) is defined to describe a multi-objective optimization problem [19], where X is an n-dimensional decision space, $x = (x_1, x_2 \ldots x_n) \in X$, C is a constraint set that the decision variables should satisfy, and F is a vector that contains $m \geq 2$ objective functions, $F = \left(f_1(x), f_2(x), \ldots f_m(x)\right)$. The goal of multi-objective optimization is to minimize these objective functions, as described in (1).

$$\begin{cases} \min\ F = \left[f_1(x), f_2(x), \cdots, f_m(x)\right] \\ c_i(x) \leq 0,\ \ i = 1, 2, \cdots I \end{cases} \tag{1}$$

where $c_i(x)$ is the i-th constraint condition.

Feasible solution—**definition 1:** x is a feasible solution if and only if $x \in X$, and all the constraint conditions in C are satisfied.

Feasible solution set—**definition 2:** a set of all feasible solutions x is called a feasible solution set X_f:

$$X_f = \left\{\bar{x} \in X \mid e\left(\bar{x}\right) \leq 0\right\} \tag{2}$$

Pareto dominated—**definition 3:** for decision vectors $\vec{a}, \vec{b}$,

1. $\bar{a} \prec \bar{b}$ ($\bar{a}$ dominates $\bar{b}$), if and only if $\forall i \in \{1, 2, \ldots, K\}, f_i(\bar{a}) \leq f_i(\bar{b})$ and $\exists j \in \{1, 2, \ldots, K\}, f_i(\bar{a}) < f_i(\bar{b})$
2. $\bar{a} \leq \bar{b}$ ($\bar{a}$ weakly dominates $\bar{b}$), if and only if $\forall i \in \{1, 2, \ldots, K\}$, $f_i(\bar{a}) \leq f_i(\bar{b})$
3. $\bar{a} \sim \bar{b}$ ($\bar{a}$ is not different from $\bar{b}$), if and only if $\bar{a} \nprec \bar{b} \wedge \bar{b} \nprec \bar{a}$

Pareto optimal solution—**definition 4:** if a feasible solution x^* satisfies condition $\neg\exists x \in X_f$ such that $x \succ x^*$, then x^* is called a Pareto non-dominated solution (or optimal solution).

Pareto optimal solution set—**definition 5:** the Pareto optimal solution set is the set of all Pareto optimal solutions.

The goal of multi-objective optimization is to achieve a uniform distribution of the solution set and a fast approximation to the actual Pareto frontier. Based on this goal, researchers have proposed numerous effective multi-objective evolutionary algorithms. The research on evolutionary multi-objective optimization algorithms can be divided into three phases. In phase 1, Pareto dominance was employed to design a simple fitness function; typical algorithms include the multi-objective genetic algorithm (MOGA) A by Fonseca and Fleming, the niched Pareto genetic algorithm

(NPGA) A by Horn and Nafpliotis, and the non-dominated sorting genetic algorithm (NSGA) A by Srinivas and Deb [20]. In phase 2, an elitist group was employed to maintain the current Pareto optimal solution's distribution properties; typical algorithms include the Pareto archived evolutionary strategy (PAES) A by Knowles *et al.*, the strength Pareto evolutionary algorithm (SPEA) A and SPEA2 by Zitzler and Thiele *et al.* [21], and the improved NSGA A (NSGA-II), which is based on an elitist strategy, by Deb and Pratap *et al.* [22]. In phase 3, hybrid algorithms were developed from combinations of the multi-objective evolutionary algorithm and other emerging intelligent algorithms, such as the immune multi-objective optimization algorithm (IMOA) A [23] and multi-objective particle swarm optimization (MOPSO) A [24].

3. MULTI-OBJECTIVE MODEL FOR MCR PATH PLANNING

In the real world, the MCR problem has multiple objectives, and the goals are complex, dynamic, and influenced by many factors. The robot will face a complex and dynamic MCR path. Based on these characteristics, the real-world multiple-objective MCR problem is abstracted into a mathematical model to create a multi-objective model for MCR path planning.

Because several influencing factors of the MCR path planning problem are correlated and affect each other, they should be taken into account as a whole. Further, the actual conditions should be satisfied. The following three factors are taken into account: the number of minimum obstacles in the constraint set C_k, the route length $R(L)$, and the total cost to remove all obstacles in the minimum constraint set C_k.

Assume that an obstacle in the real environment is denoted as $O_i(1 \le i \le n)$. The robot R scans the obstacles in the environment and determines the obstacle's observed parameter vector $K = \{k_1, k_2, ..k_i..k_n\}$, where k_i indicates whether the robot can detect an obstacle in a particular environment; and k_i is either 0 or 1, where 1 means that O_i is detectable by robot R, and 0 means that O_i is undetectable by robot R. The observed parameter vector K represents the cognition levels of the obstacles in the environment, and $|K|$ is the number of observable obstacles.

Assume that any robot R has an r-dimensional capability vector B, and B quantitatively describes a particular capability of robot R, where $B_i =< b_i^1, b_i^2, \dots b_i^r >$, $b^j \ge 0$, and $\left(1 \le j \le r\right)$. For an intelligent robot, b^j represents the intelligent robot's capabilities, such as computing, opening, and moving. The number of observed obstacles is $n = |K|$; each obstacle $O_i\,(1 \le i \le n)$ has an r-dimensional capability requirement vector $B_{O_i} =< b_{O_i}^1, b_{O_i}^2, \dots b_{O_i}^r >$.

Definition 1 (Necessary condition): The minimum constraint set C is defined as a non-empty subset of N that is composed of $O_i\,(1 \le i \le n)$. The minimum constraint set C_k that contains k obstacles has the capability requirement vector $B_{C_k} = \sum_{i=1}^{k} B_{O_i}$, $(1 \le i \le k)$, where B_{C_k} is the sum of all obstacle capability requirement vectors in the minimum constraint set. The necessary condition that the minimum constraint set C_k can be removed is $B_{C_k} \le B$ $(1 \le k \le n)$.

Definition 2: The total cost to remove the obstacles in the minimum constraint set C_k is calculated using the formula

$$Cost\,(C_k) = B_{C_k} = \sum_{i=1}^{k} B_{O_i}, (1 \le i \le k) \tag{3}$$

Definition 3: The route length calculation is as follows:

$$R(L) = \sum_{i,j=1}^{k} e_{ij} \tag{4}$$

where e_{ij} denotes an edge that connects i and j.

The multi-objective model for the MCR path planning problem is described as follows. When the precondition of the minimum constraint set C_k removal is satisfied, the goals are to find the path with the shortest length that traverses the minimum number of obstacles and to keep the total cost of obstacle removal in the minimum constraint set C_k at the minimum level. The MCR multi-objective model is defined as

$$\Theta(Cost) = \min\left\{\sum_{R_{s\to g}} B(o_i)\right\}$$

$$\Theta(R) = \min\left\{\sum_{R_{s\to g}} R(e_i)\right\}$$

$$\Theta(N) = \min\left\{\sum_{R_{s\to g}} N(p_i)\right\}$$

$$\text{The constraint condition is } B_{C_k} \le B\,(1 \le k \le n) \tag{5}$$

where $B(o_i)$ denotes the cost for the robot to remove obstacles; the goal is to find the path from the start point p_s to the goal point p_g for which the removal or traversal cost is the minimum. $R(e_i)$ denotes the route length; the goal is to find the shortest path from the start point p_s to the goal point p_g. $N(p_i)$ denotes the number of obstacles that the robot encounters; the goal is to find the path from the start point p_s to the goal point p_g that traverses the minimum number of obstacles.

4. MCR PATH PLANNING BASED ON MULTI-OBJECTIVE PARTICLE SWARM OPTIMIZATION ALGORITHM

In the conventional robot path planning problem, several intelligent optimization algorithms are employed for path search and optimization. After the MCR problem was developed, intelligent algorithms, such as the greedy algorithm and the ACO algorithm, were applied to MCR path planning. The particle swarm algorithm is also a typical intelligent optimization algorithm. This paper applies the particle swarm optimization (PSO) A algorithm to multi-objective MCR path planning and designs a MOPSO algorithm according to the characteristics of the MCR multi-objective path planning problem.

When the PSO algorithm is used to solve a multi-objective optimization problem, the key is to select the personal best (pbest) and global best (gbest) of the PSO algorithm. Three general goals of multi-objective optimization should be satisfied [25]: to obtain the maximum number of Pareto solutions, to ensure that the obtained Pareto solution set is as close to the actual Pareto frontier as possible, and to maximize the Pareto solution's distribution range. It is fairly easy to determine the individual leader in a single objective problem because there is only one gbest at each moment. However, in multi-objective optimization, because the non-dominated solution is not unique, each particle should identify a gbest in a leader set to update its own position; the swarm should take into account not only the optimization level of the "solution" but also the distribution of the Pareto frontier. The diversity of the swarm should also be maintained to avoid local optima.

Assuming that a particle swarm contains m particles, the velocity and position of particle i at moment t are described by

$$v_i^t = \left\{v_{i1}^t, v_{i2}^t, \cdots v_{id}^t\right\} \tag{6}$$

$$p_i^t = \left\{p_{i1}^t, p_{i2}^t, \cdots p_{id}^t\right\} \tag{7}$$

where v_i^t denotes the velocity of particle i at moment t, d is the number of components, and p_i^t denotes the position of particle i at moment t.

In this paper, path planning involves more than the position. Therefore, a fitness function is used for the evaluation. The state of p_i^t should be checked to determine if it has reached its pbest position and the gbest position, which ensures that all particles are optimal. The two best positions are described by the following two formulae:

$$_l p_i^t = \left\{_l p_{i1}^t, _l p_{i2}^t, \mathrm{L} _l p_{id}^t\right\} \tag{8}$$

$$_g p_i^t = \left\{_g p_{i1}^t, _g p_{i2}^t, \mathrm{L} _g p_{id}^t\right\} \tag{9}$$

where $_l p_i^t$ denotes the pbest position of particle i, and $_g p_i^t$ denotes the gbest position of particle i. When time steps from moment t to moment $(t+1)$, the velocity of particle i is updated according to the following formula:

$$v_{id}^{t+1} = \alpha v_{id}^t + \beta\left(_l p_{id}^t - p_{id}^t\right) + \gamma\left(_g p_{id}^t - p_{id}^t\right) \tag{10}$$

where α is the inertia factor, which represents the inertia of a particle that is traveling at its original velocity, and β and γ are learning factors, which represent the particle's learning and adjustment capabilities according to the pbest and gbest positions, respectively.

$$p_{id}^{t+1} = p_{id}^t + v_{id}^{t+1} \tag{11}$$

For the multi-objective MCR path planning problem, particles are randomly generated at the start point p_s to travel to the goal point p_g, and the fitness function is iterated and updated continuously using formula (5).

Based on this analysis and the aforementioned idea of a MOPSO-based solution to the MCR problem, a simple and highly effective multi-objective particle swarm algorithm is proposed to address the MCR problem. The procedure of the algorithm is as follows (Figure 3):

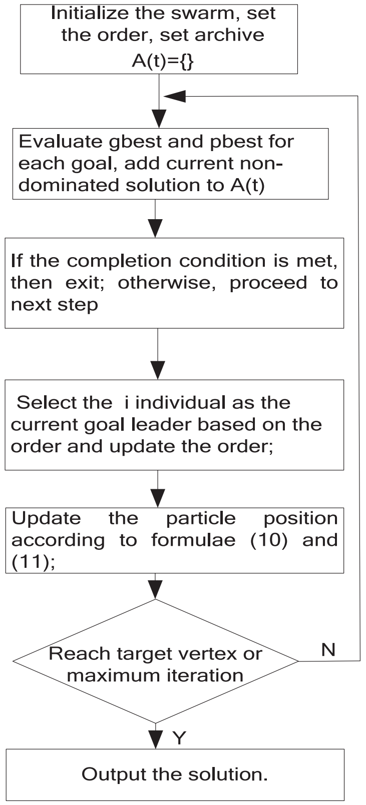
Initialize the swarm, set the order, set archive A(t)={}
Evaluate gbest and pbest for each goal, add current non-dominated solution to A(t)
If the completion condition is met, then exit; otherwise, proceed to next step
Select the i individual as the current goal leader based on the order and update the order;
Update the particle position according to formulae (10) and (11);
Reach target vertex or maximum iteration
N
Y
Output the solution.

Figure 3 The procedure of the MCR path planning algorithm.

First, the algorithm randomly generates an initialized swarm and an empty archive and sets an order. The gbest and historical pbest for each objective function are then evaluated, the non-dominated solution for the current generation of individuals is archived, and the individuals that are dominated by the current solution are deleted from the original archive. If the storage exceeds the threshold, then some elements are deleted based on certain rules. In step 4, the goal leader is identified based on the order; the particle position is then updated based on the goal leader, and the order is modified. This process is repeated until the constraint conditions are satisfied.

5. EXPERIMENT RESULT AND ANALYSIS

To verify the effectiveness of the proposed MOPSO method, the following experiment is performed.

The test scenario is assumed to be a 10*10 grid area. Therefore, the entire test scenario contains 100 grid sub-blocks; the obstacle area is randomly generated and contains at least one grid. The four vertexes of a grid are nodes, which are positions where the robot can stay. Figure 4 shows the entire test scenario.

In this test, the maximum iteration of the MOPSO will not exceed 1000. In addition, the ACO algorithm in reference [18] and the crow search algorithm (CSA) in reference [26] are compared with the proposed algorithm. Two sets of test data with the following detailed settings are configured:

1. V = 25, E = 80, S = 10, 20;
2. V = 441, E = 1680, S = 10, 20.

where V denotes a vertex, E denotes an edge, and S denotes the number of obstacles.

The three algorithms are executed separately, and the results are shown in Tables 1 and 2 and Figures 5 and 6. In the results, K denotes the number of removable obstacles, Cost denotes the cost to remove obstacles, R(L) denotes the route length, and Cover denotes the obstacles to be removed. The following formula is defined for the test:

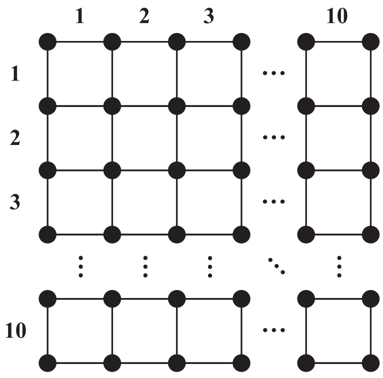

Figure 4 Diagram of the test scenario.

Table 1 The best path comparison of the two algorithms in test data 1.

		Found Solution	R(L)	Cost	K [Cover]
V = 25	ACO	True	24	13	2 [‘O2’, ‘O3’]
E = 80	CSA	True	24	13	2 [‘O2’, ‘O3’]
S = 10	MOPSO	True	24	13	2 [‘O2’, ‘O3’]
V = 25	ACO	True	38	256	5 [‘O2’, ‘O3’, ‘O8’, ‘O9’,‘10’]
E = 80	CSA	True	28	156	4 [‘O2’, ‘O3’, ‘O8’, ‘O9’]
S = 20	MOPSO	True	28	156	4 [‘O2’, ‘O3’, ‘O8’, ‘O9’]

Table 2 The best path comparison of the two algorithms in test data 2.

		Found Solution	R(L)	Cost	K [Cover]
V = 441	ACO	True	413	42	3 [‘O1’, ‘O4’, ‘O5’]
E = 1680	CSA	True	413	42	3 [‘O1’, ‘O4’, ‘O5’]
S = 10	MOPSO	True	413	42	3 [‘O1’, ‘O4’, ‘O5’]
V = 441	ACO	True	441	223	5 [‘O1’, ‘O10’, ‘O4’, ‘O5’, ‘O9’]
E = 1680	CSA	True	441	223	5 [‘O1’, ‘O10’, ‘O4’, ‘O5’, ‘O9’]
S = 20	MOPSO	True	421	207	5 [‘O1’, ‘O10’, ‘O5’, ‘O9’]

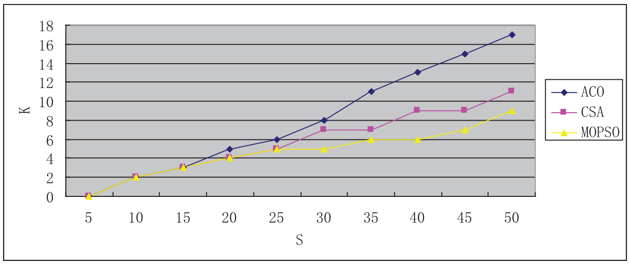

Figure 5 Comparison of the number of removable obstacles (V = 25, E = 80).

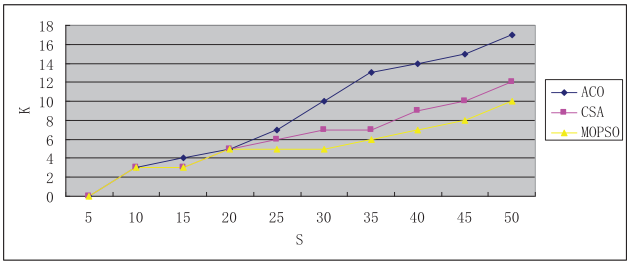

Figure 6 Comparison of the number of removable obstacles (V = 441, E = 1680).

$$\text{Cost} = B_{C_k} = \sum_{i=1}^{k} B_{O_i} = \sum_{i=1}^{k} \left(O_i{}^{*}i\right),\ (1 \le i \le k)$$

$$R(L) = \sum_{R_{s\to g}} e_{ij} = \sum_{R_{s\to g}} \left(i\ {}^{*}j\right) \tag{12}$$

SPSS 19.0 statistical analysis software was used to process the data. We ran 50 experiments; the Wilcoxon test was used to compare the counting data. P < 0.05 was considered statistically significant. The statistical results show the following. In test data 1, compared with the MOPSO algorithm, the ACO and CSA algorithms had longer path lengths, higher costs, and more obstacles, but there was no significant difference between the MOPSO and CSA algorithms (P > 0.05) or between the MOPSO and ACO algorithms (P > 0.05). In test data 2, compared with the MOPSO algorithm, the ACO and CSA algorithms had longer path lengths, higher costs, and more obstacles, and there was a significant difference between the MOPSO and CSA algorithms (P < 0.05) and between the MOPSO and ACO algorithms (P < 0.05).

In Figures 5 and 6, S denotes the number of obstacles, and K denotes the number of removable obstacles. Tables 1 and 2 show that path planning using the MOPSO algorithm proposed in this paper can find a shorter optimal path at a lower cost than using the ACO and CSA algorithms. Moreover, with an increase in the number of obstacles, the advantage of path planning using the MOPSO algorithm increases.

Figures 5 and 6 also show that with an increase in the number of obstacles in the scenario, all the optimal paths that were identified by the three algorithms contain obstacles. The paths that were identified by the path planning algorithms that used the ACO and CSA algorithms contain significantly more obstacles, and this trend is magnified as the number of obstacles increases. A comparison of the results in Tables 1 and 2 and Figures 5 and 6 shows that because of the superiority of the MCR planning result, the proposed algorithm is superior.

Figures 7 and 8 show the Pareto solution set distribution of the MOPSO algorithm.

Figures 7 and 8 show that the MOPSO algorithm has a wide search space and a large number of well-distributed solutions. The Pareto solution set space distributions in Figures 7 and 8 show relationships between three goal vectors: the shortest route length $R(L)$, the number of obstacles K, and the total cost to remove obstacles in the minimum constraint set C_k. Because the three goals are mutually exclusive, all the goals can rarely be optimized under normal conditions. Therefore, the Pareto solution set must be selected according

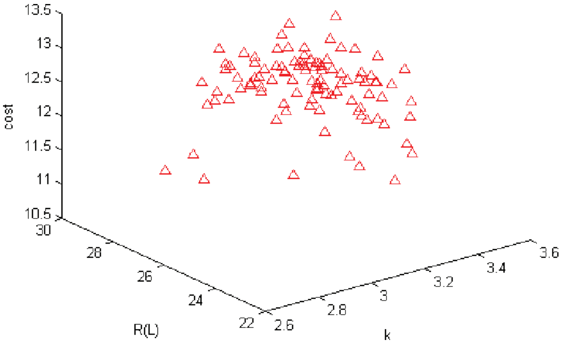

Figure 7 | Pareto solution set distribution of the MOPSO algorithm (V = 441, E = 1680, S = 10, 500 iterations).

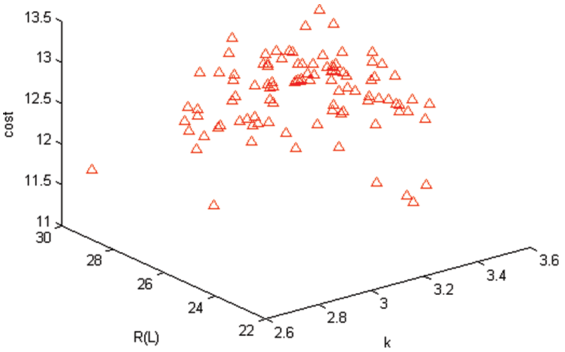

Figure 8 | Pareto solution set distribution of the MOPSO algorithm (V = 441, E = 1680, S = 10, 1000 iterations).

to the actual requirements. When the system sets a minimum number of obstacles K as the primary goal, a combination is selected from the plans with fewer numbers of obstacles. When the shortest route length $R(L)$ is set as the primary goal, a combination is selected from the plans with shorter route lengths. When the minimum total cost to remove the obstacles in the minimum constraint set C_k is set as the primary goal, a combination is selected from the plans with lower costs. When the system has no specific goal, a plan with balanced results for the three goals is selected.

6. CONCLUSIONS

This paper used the MOPSO algorithm for multi-objective MCR path planning. First, a multi-objective model for MCR path planning was constructed. A MOPSO algorithm was then designed based on the fitness function of the multi-objective MCR problem, and an iteration formula based on pbest and gbest was constructed to update the particle velocity and position. Finally, a comparison with the ACO and CSA algorithms showed that in scenarios with an identical number of obstacles, the MOPSO path planning algorithm can obtain a shorter best path that traverses fewer obstacle areas, which is better for solving the MCR path planning problem, and provide a global optimization plan for MCR path planning that satisfies multiple design goals. A single execution of the algorithm obtains multiple Pareto optimal solutions, which provides a reference and can be applied to actual cases.

The MCR problem in complex environments is a new MCR research problem. The contradictions between the targets and the dynamic changes between the obstacles and the robot are research challenges for the MCR problem in complex environments.

CONFLICT OF INTEREST

Authors have no conflict of interest to declare.

AUTHORS' CONTRIBUTIONS

Bo Xu and Feng Zhou conceived and designed the study, performed the experiments and wrote the paper. Antonio Marcel Gates reviewed and edited the manuscript. All authors read and approved the manuscript.

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China (11901113), Natural Science Foundation of Guangdong Province (2018A030307032), and the Guangzhou Science and Technology Plan Project, China (201904010225). The authors thank all the reviewers and editors for their valuable comments and work.

REFERENCES

[1] J. Canny, The Complexity of Robot Motion Planning, MIT Press, Cambridge, MA, USA, 1988. https://dl.acm.org/doi/book/10.5555/49142

[2] L. Zhang, Y. Kim, D. Manocha, A simple path non-existence algorithm using c-obstacle query, in: S. Akella, N.M. Amato, W.H. Huang, B. Mishra (Eds.), Algorithmic Foundation of Robotics VII, Springer Tracts in Advanced Robotics, Springer, Berlin, Heidelberg, 2008, pp. 269–284.

[3] M. Gobelbecker, T. Keller, P. Eyerich, *et al.*, Coming up with good excuses: what to do when no plan can be found, in Proceedings of the Twentieth International Conference on Automated Planning and Scheduling, Toronto, Ontario, Canada, 2010, pp. 81–88. https://aaai.org/ocs/index.php/ICAPS/ICAPS10/paper/view/1453

[4] Z. McCarthy, T. Bretl, S. Hutchinson, Proving path non-existence using sampling and alpha shapes, in Proceedings of IEEE International Conference on Robotics and Automation, Saint Paul, MN, USA, 2012, pp. 2563–2569.

[5] K. Hauser, The minimum constraint removal problem with three robotics applications, in Proceedings of Workshop on the Algorithmic Foundations of Robotics, Massachusetts, USA, 2012.

[6] K. Hauser, On responsiveness, safety, and completeness in real-time motion planning, Auton. Robots. 32 (2012), 35–48.

[7] K. Hauser, The minimum constraint removal problem with three robotics applications, Int. J. Robot. Res. 33 (2014), 5–17

[8] L.H. Erickson, S.M. LaValle, A simple, but NP-hard, motion planning problem, in Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence (AAAI-13), Bellevue, Washington, USA, 2013, pp. 1388–1393. https://www.aaai.org/ocs/index.php/AAAI/AAAI13/paper/view/6280

[9] J.B. Cohen, S. Chitta, M. Likhachev, Single and dual-arm motion planning with heuristic search, Int. J. Robot. Res. 33 (2014), 305–320.

[10] A. Krontiris, R. Shome, A. Dobson, *et al.*, Rearranging similar objects with a manipulator using pebble graphs, in IEEE-RAS International Conference on Humanoid Robots, Madrid, Spain, 2014, pp. 332–340.

[11] G. Sharon, R. Stern, A. Felener, *et al.*, The conflict-based search algorithm for multi-agent path finding, Artif. Intell. J. 219 (2015), 40–66.

[12] A. Gorbenko, V. Popov, The discrete minimum constraint removal motion planning problem, in AIP Conference Proceedings, AIP Publishing LLC, Jatinangor, Indonesia, 2015, pp. 113–126.

[13] A. Krontiris, K.E. Bekris, Efficiently solving general rearrangement tasks: a fast extension primitive for an incremental sampling-based planner, in Proceedings of IEEE International Conference on Robotics and Automation (ICRA), IEEE, Stockholm, Sweden, 2016, pp. 3924–3931.

[14] A. Krontiris, K.E. Bekris, Computational tradeoffs of search methods for minimum constraint removal paths, in AAAI Publications, Eighth Annual Symposium on Combinatorial Search, Ein Gedi, the Dead Sea, Israel, 2015. https://aaai.org/ocs/index.php/SOCS/SOCS15/paper/view/11286

[15] A. Krontiris, K.E. Bekris, Trade-off in the computation of minimum constraint removal paths for manipulation planning, Adv. Robot. 12 (2017), 1–12.

[16] B. Sayan, K. Neeraj, S. Subhash, *et al.*, Improved approximation bounds for the minimum constraint removal problem, in: E. Blais, K. Jansen, J.D.P. Rolim, D. Steurer (Eds.), Approximation, Randomization, and Combinatorial Optimization, Algorithms and Techniques, vol. 2, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2018, pp. 1–19.

[17] B. Xu, H. Min, Solving Minimum Constraint Removal (MCR) problem using a social-force-model- based ant colony algorithm, Appl. Soft Comput. 43 (2016), 553–560.

[18] B. Xu, Study on Minimum Constraint Problem and Its Algorithm Based on Mobile Obstacle Environment, South China University of Technology, Guangzhou, China, 2017. http://new.oversea.cnki.net/KCMS/detail/detail.aspx?dbcode=CDFD&dbname=CDFDLAST2018&filename=1017734339.nh&v=MDQ5MTZXN3pNVkYyNkdiUzdHdExQcHBFYlBJUjhlWDFMdXhZUzdEaDFUM3FUcldNMUZyQ1VSN3FmWk9kckZDamc=

[19] X.F. Zou, Y. Chen, M.Z. Liu, *et al.*, A new evolutionary algorithm for solving many-objective optimization problems, IEEE Trans. Syst. Man Cybern. Part B. 38 (2008), 1402–1412.

[20] N. Srinivas, K. Deb, Multi-objective optimization using nondominated sorting in genetic algorithms, Evol. Comput. 2 (1994), 221–248.

[21] E. Zitzler, K. Deb, L. Thiele, Comparison of multi-objective evolutionary algorithms: empirical results, Evol. Comput. 8 (2000), 173–195.

[22] K. Deb, S. Agrawal, A. Pratap, *et al.*, A fast and elitist nondominated sorting genetic algorithm for multi-objective optimization: NSGA-II, IEEE Trans. Evol. Comput. 6 (2002), 182–197.

[23] D.D. Yang, L.C. Jiao, M.G. Gong, *et al.*, Clone selection algorithm to solve preference multi-objective optimization, J. Softw. 21 (2010), 14–33.

[24] V. Kumar, S. Minz, Multi-objective particle swarm optimization: an introduction, Smart Comput. Rev. 4 (2014).

[25] D. Cai, Y. Wang, Y. Miao, A new multi-objective particle swarm optimization algorithm based on decomposition, Inf. Sci. 325 (2015), 541–557.

[26] A. Askarzadeh, A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm, Comput. Struct. 169 (2016), 1–12.

Appendix A

Minimum constraint removal	MCR
Ant colony optimization	ACO
Crow search algorithm	CSA
Particle swarm algorithm	PSO
Non-deterministic-polynomial-hard	NP-hard
Multi-objective genetic algorithm	MOGA
Niched Pareto genetic algorithm	NPGA
Non-dominated sorting genetic algorithm	NSGA
Pareto archived evolutionary strategy	PAES
Strength Pareto evolutionary algorithm	SPEA
Immune multi-objective optimization algorithm	IMOA
Multi-objective particle swarm algorithm	MOPSO