International Journal of Computational Intelligence Systems (2024) 17:127
https://doi.org/10.1007/s44196-024-00511-x

RESEARCH ARTICLE

RGG-PSO+: Random Geometric Graphs Based Particle Swarm Optimization Method for UAV Path Planning

Yang Liu[1] · Xiaomin Zhu[1] · Xiao-Yi Zhang[2] · Jiannan Xiao[3] · Xiaohan Yu[1]

Received: 18 January 2024 / Accepted: 25 April 2024

Abstract
Evolutionary algorithms, such as particle swarm optimization (PSO), are widely applied to UAV path planning problems. However, the fixed particle length of PSO, which may not be suitable for the scenario, will compromise the search efficiency. This paper proposes the RGG-PSO+ method, which adapts to scenarios by dynamically adjusting the number of waypoints. Random geometric graphs (RGG) and the divide-and-conquer paradigm are involved in improving the proposed method. Comparative analyses with established heuristic methods demonstrate RGG-PSO+'s superior performance in complex environments, particularly in terms of convergence speed and path length. The implementation of RGG significantly improves the F-Measure, indicating a shift from exploration to exploitation of PSO's iterations, and the implementation of the divide-and-conquer paradigm is evident in the improved mean and variance of normalized path lengths.

Keywords Particle swarm optimization(PSO) · Random geometric graphs(RGG) · UAV · Path planning

1 Introduction

UAV path planning refers to the process of determining a feasible or nearly optimal path for an unmanned aerial vehicle (UAV) to follow from a starting position to a destination. The primary goal of UAV path planning is to ensure that the UAV navigates effectively and safely while avoiding obstacles and meeting specific criteria required by applications [1]. Here, effectiveness refers to the quality of the path, and safety relates to the capability of the path to guide the UAV through obstacles without collision. As for the criteria, these may include kinematic and dynamic constraints, minimizing the execution time of the navigation path, or finding the Pareto solution for multi-objective tasks [2].

Path planning is a highly complex problem. Reif demonstrated that in the piano mover's problem, the complexity of planning a feasible solution has a lower bound of PSPACE-hard [3]. This complexity poses significant challenges in practical applications. Thus, the research work has focused on methods that yield near-optimal solutions, including sampling-based, graph search-based, and evolutionary-algorithm-based methods. Graph search-based methods transform path planning into a graph-theoretic path search problem, using methods such as A* [4, 5], hybrid A* [6], and Jump Point Search [7, 8]. Sampling-based path planning methods integrate the theory of Random Geometric Graphs (RGG) to transform the construction of roadmaps into graph-theoretic path search problems [9]. Supported by the principles of RGG, sampling-based methods possess numerous mathematical properties, including probabilistic completeness and asymptotic optimality [9, 10]. Evolutionary-algorithm-based methods have become increasingly popular in the path planning domain. In particular, these methods are capable of finding high-quality

This work was supported in part by the Fundamental Research Funds for the Central Universities (No. 2023JBMC017), and the National Natural Science Foundation of China (No. 6230070656).

✉ Xiaomin Zhu
xmzhu@bjtu.edu.cn

Yang Liu
yangliu2@bjtu.edu.cn

Xiao-Yi Zhang
xiaoyi@ustb.edu.cn

Jiannan Xiao
xjjnnn@mail.ustc.edu.cn

Xiaohan Yu
23126228@bjtu.edu.cn

[1] Beijing Jiaotong University, Beijing, China

[2] University of Science and Technology Beijing, Beijing, China

[3] University of Science and Technology of China, Hefei, China

Published online: 21 May 2024

satisfactory solutions in complex scenarios, such as genetic algorithm (GE) [11–13], differential evolution (DE) [14, 15], ant colony optimization (ACO) [16], and particle swarm optimization (PSO) [17–20].

Out of these evolutionary methods, PSO stands out due to its widespread use and the development of several variants that enhance its application in UAV path planning. For example, Hoang et al. proposed a phase-angle-encoded and quantum-behaved PSO method [19]. Tang et al. proposed a self-adaptive evolutionary PSO method for UAV Path planning [18]. Phung et al. proposed a spherical vector-based PSO method, validating the proposed method with real UAV operations [21]. Huang et al. proposed cylindrical coordinates and automatically adjusted inertia weight and acceleration coefficients improvements to PSO [17]. These methods can effectively search the configuration space of the UAV to generate quality solutions.

However, the aforementioned PSO-based methods presuppose the dimension of the particle. In other words, all particles are set with a predetermined number of waypoints for their paths. Therefore, how to choose the number of waypoints remains an open question. Specifically, selecting a larger number of waypoints in simple scenarios, i.e., scenarios with fewer obstacles, will lead to the redundancy of the particle dimensions, thereby reducing the search efficiency of the algorithm. Conversely, selecting fewer waypoints in complex scenarios will result in insufficient particle dimensions, thereby reducing the quality of the generated trajectory, and may even lead to the inability to find a feasible path. Therefore, how to dynamically adjust the number of waypoints for particles is a question worth researching.

Furthermore, it is known that PSO has limitations in dealing with local minima occurred in the field. For the UAV path planning problem, the initialization process of PSO randomly generates a few feasible solutions, leading to a lot of computational waste in generating useless particles. As for the evolution process, many illegal solutions are produced, resulting in a smaller number of effective solutions. The exploration ability of PSO inevitably declines due to two factors, making it highly susceptible to getting trapped in local optima. Despite various modifications proposed in previous research, enhancing the solution quality of PSO remains a challenge.

In response to the aforementioned problems, this paper proposes an innovative method for UAV path planning, denoted as the RGG-PSO+, which integrates Random Geometric Graphs (RGG) and divide-and-conquer paradigm. Here, RGG is a type of probabilistic graph model, which can effectively represent the connectivity in the configuration space where paths need to be determined. The proposed method can dynamically adjust the number of waypoints in particles and improve the quality of randomly generated initial particles, thereby enhancing the method's search efficiency. Moreover, the proposed method involves the divide-and-conquer paradigm to reduce the search space of sub-planning problems. This improvement helps to prevent the degradation of PSO's exploration abilities and ensures the full advantage of its exploitation capabilities during the evolution iterations. These advancements contribute to the search quality of RGG-PSO+.

For evaluation, 1000 random scenarios have been generated with increasing levels of complexity, which is measured by the number of obstacles in the scenario. The comparisons between the proposed RGG-PSO+ and other heuristic methods are then conducted on those scenarios to evaluate their performance. The contributions of this study are listed as follows:

- RGG is introduced to generate path candidates, and variable-length particle mechanism is proposed to transform the path candidates into particles.
- Divide-and-conquer paradigm of PSO-based path planning is proposed, which is capable of generating high-quality solutions.

The remainder of the paper is organized as follows. Section 2 introduces the related work of PSO and RGG applied in UAV path planning. Section 3 provides a common preliminary of definitions and assumptions for the literature. Section 4 presents the proposed method, denoted as RGG-PSO+. Sections 5 and 6 provide the experimental settings and results analysis, respectively. Section 7 concludes the paper and discusses the future work.

2 Related Work

We review related work in the area of particle swarm optimization (PSO) and random geometric graphs (RGGs) for the path planning problem.

2.1 Particle Swarm Optimization (PSO)

Particle swarm optimization (PSO) is a computational method that optimizes problems by iteratively enhancing potential solutions based on the predefined quality measure. It operates with a group of candidate solutions, known as particles, navigating them through the search space via mathematical equations related to their position and velocity [22]. Each particle is influenced by its best-known position and the collective best-known positions discovered within the search space. These positions are dynamically updated as particles identify superior solutions, progressively steering the entire swarm closer to the optimal solutions [23]. In the context of path planning, PSO is used to find the most efficient path, considering various constraints and objectives. Specifically, PSO

is widely used for path planning due to its strong searchability, fast convergence, and efficiency [2]. However, it has been found that PSO is prone to getting trapped in local optima and lead to premature convergence while dealing with the complex problems [2, 24].

To avoid possible local optima, the latest PSO methods are improved by incorporating techniques, such as sigmoid-function-based weighting [24], fuzzy-based inertial weighting [20], Bezier-curve-based path smoothness [25], and adaptive parameter strategies [17, 18]. Some works have developed various encoding methods tailored to meet the constraints of cost functions. For example, the phase-angle-encoded and quantum-behaved PSO method [17, 19, 26] and the fitness-scaling adaptive chaotic PSO method [18, 27, 28] are proposed. In addition to the parameter setting, supplementary methods are proposed to enhance the diversity of its particle population. For example, cross-learning strategy [29], spherical vector [21], and orientation angle-based grouping strategy [30] are proposed. These improvements have resulted in better particle quality outcomes with fewer iterations.

2.2 Random Geometric Graphs (RGGs)

Random geometric graphs (RGGs), first conceptualized by Gilbert in 1961, play a pivotal role in enhancing the effectiveness of path planning methods [9]. Essentially, an RGG is a network of points randomly distributed within a specific subspace of $\mathbb{R}^d$, where edges are drawn between points within a predefined threshold $r > 0$, denoted as the connection radius. The connectivity property of random geometric graphs, ensuring a high probability of connections, stands as a necessary and sufficient condition that the connection radius r being in direct proportion to $(\frac{\log n}{n})^{\frac{1}{d}}$, where n denotes the quantity of points dispersed within a unit hypercube $[0, 1]^d$ [31].

Based on the connectivity property, RGGs are integral to the methodologies of lazy motion planning, which primarily concentrates on diminishing the frequency of collision detection. Methods such as Lazy-PRM [32] and Generalized Lazy Search [33] exemplify this by selectively generating an RGG and examining edges only when they are integral to the most direct route towards the intended target. The intricacy of the RGG, in harmony with the operational finesse of the spanning tree, plays a critical role in drastically lowering the occurrence of collision checks.

Furthermore, the essence of RGGs is deeply embedded within sampling-based planning methods, which support the asymptotically optimal property. Sampling-based planning methods can, with a probability approaching 1, asymptotically converge to the optimal solution if one exists, as the number of samples goes to infinity (i.e., they are almost surely asymptotically optimal) [10, 34, 35]. The RGG applied sampling-based methods are particularly effective in environments that are highly complex due to high dimensionality or dynamic changes, such as Fast Marching Tree (FMT*) [36] and Batch-Informed Trees (BIT*) [37].

3 Preliminary

In this part, we introduce fundamental definitions and concepts of the planning problem [38] as the preliminary.

Definition 1 (*Path planning problem*) A planning problem is defined as a triplet $(\mathcal{X}_{free}, x_{init}, x_{goal})$ [10], which is denoted as a **scenario**. Let $\mathcal{X} = [0, x_1] \times [0, x_2]$ be the configuration space, a.k.a. **scene**, where $x_1, x_2 \in \mathbb{R}^+$ and $W \subset \mathbb{R}^2$. Let $\mathcal{X}_{obs}$ be the obstacle region, which is an open set, and denote the obstacle-free space as $\mathcal{X}_{free} = cl(\mathcal{X} \backslash \mathcal{X}_{obs})$, where $cl(\cdot)$ represents the closure of a set. The initial condition x_{init} belongs to $\mathcal{X}_{free}$, as does the goal condition that $x_{goal} \in \mathcal{X}_{free}$.

Definition 2 (*Feasible path and optimal path*) A feasible path $\sigma : [0, 1] \to \mathbb{R}^2$ is a sequence of states, where $\sigma(0) = x_{init}$, $\sigma(1) = x_{goal}$ and $\sigma(r) \in \mathcal{X}_{free}$ for all $r \in [0, 1]$. An optimal path $\sigma^* : [0, 1] \to \mathbb{R}^2$ is the feasible path satisfying $\sigma^* = \arg\min\{c(\sigma) : \sigma \in \Sigma\}$, where Σ is the set of all feasible paths and $c(\cdot) : \Sigma \to \mathbb{R}_{\geq 0}$ is a chosen cost function. Note that any path σ that collides with obstacles is referred to as a non-feasible path.

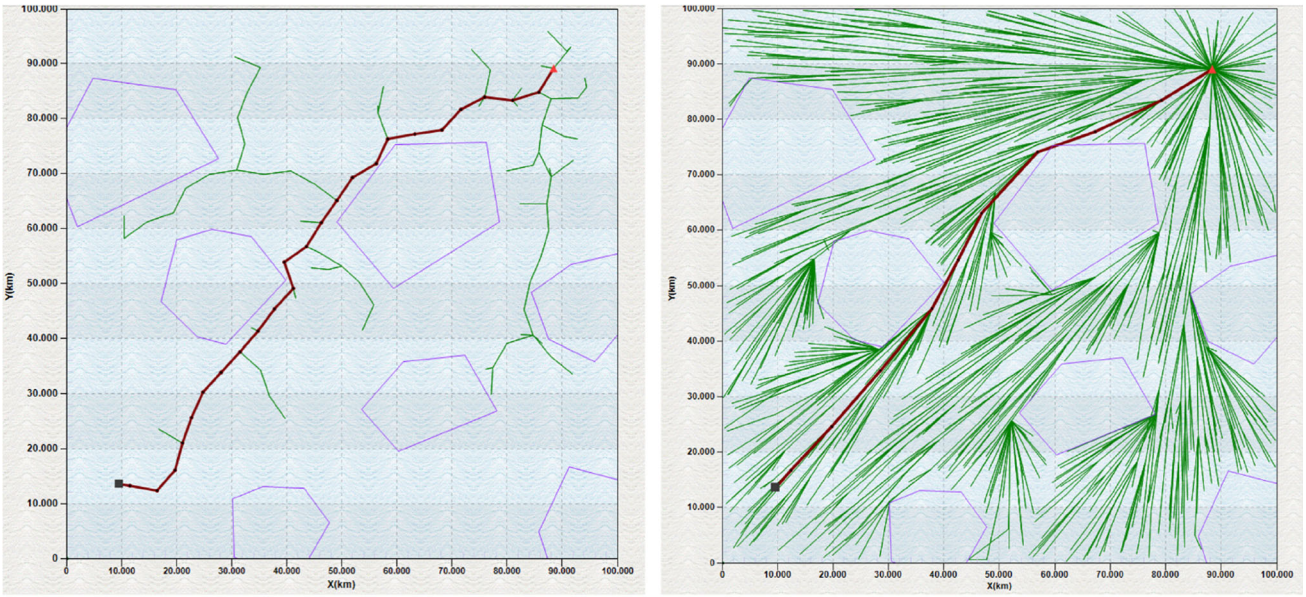
Figure 1 illustrates the UAV path planning results in two randomly generated scenarios. Obstacles $\mathcal{X}_{obs}$ are visually represented by purple circles or polygons. The flight path σ depicts the path from the initial position x_{init} to the goal position x_{goal}, which is illustrated in red. In this paper, the path is consists of a sequence states, denoted as $\sigma = [x_0, x_1, x_2, \ldots, x_n]$. The path length is regarded as the cost function, where $c(\cdot) = \Sigma_{i=0}^{n-1} \sqrt{x_i^2 + x_{i+1}^2}$.

4 Methodology

In this section, we elaborate on our proposed PSO-based path planning algorithm. Section 4.1 presents the methodology of the framework. The detail of each progress is clarified in Sects. 4.2 and 4.3, respectively.

4.1 Framework of Methodology

Our PSO-based path planning method's objective is to ensure timely task completion while securing a path that aligns closely with the optimal solution. In other words, within



(a) RGG filtered by RRT (b) RGG filtered by FMT*

Fig. 1 Demonstrations of the UAV path planning, where red line represents the path, purple circles and polygons represent the obstacles, and the green lines represent the connection edge of RGG filtered by RRT [10] and FMT* [36] methods

a predetermined operational timeframe for the method, the objective is to identify the highest quality solution. To achieve this objective, we propose a methodology comprising two enhanced procedures. These procedures are outlined below.

- **RGG-based variable-length particle.** Path construction utilizes the random geometric graphs approach, then transforms the paths into variable-length particles.
- **Divide-and-conquer paradigm of PSO-based path planning.** The PSO-based method is employed to generate the path more efficiency, involving divide-and-conquer paradigm

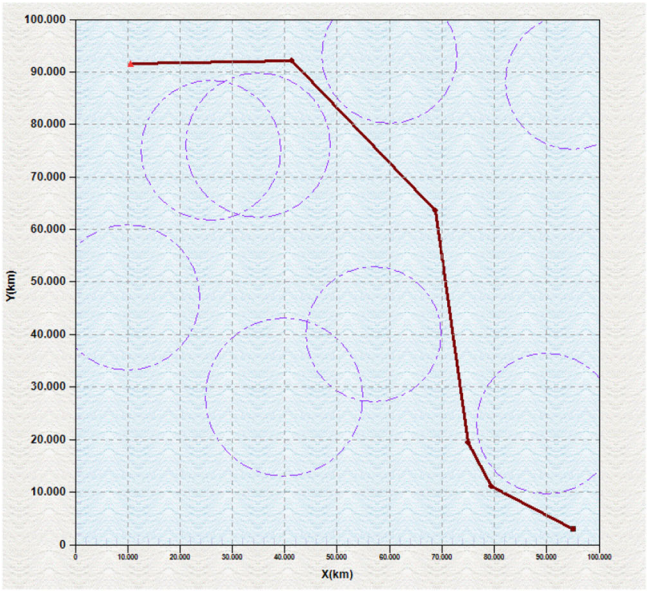
Specifically, procedure 1 uses random geometric graphs to generate path candidates, and then transforms the path candidates into particles by variable-length particle mechanism. This procedure significantly enhances the quality of the initial particles and reduces the dimensionality of the search space, thereby increasing search efficiency. The improvement stems from shifting from list generation to set generation, where randomly generated solutions are not required to satisfy a sequential order. Procedure 2 utilizes the divide-and-conquer paradigm of PSO-based path planning to generate the path. The original planning problem is decomposed into the method of optimizing sub-paths, then the sub-problems are solved separately by PSO algorithm, and finally the solutions of the sub-problems are combined into the solution of the original problem. Through the transformation of the sub-problems, the search space is reduced, which in turn improves the search efficiency.

In the subsequent two subsections, we dive deeper into each methodology procedure. Detailed explanations of procedure 1 and 2 are provided in Sects. 4.2 and 4.3, respectively.

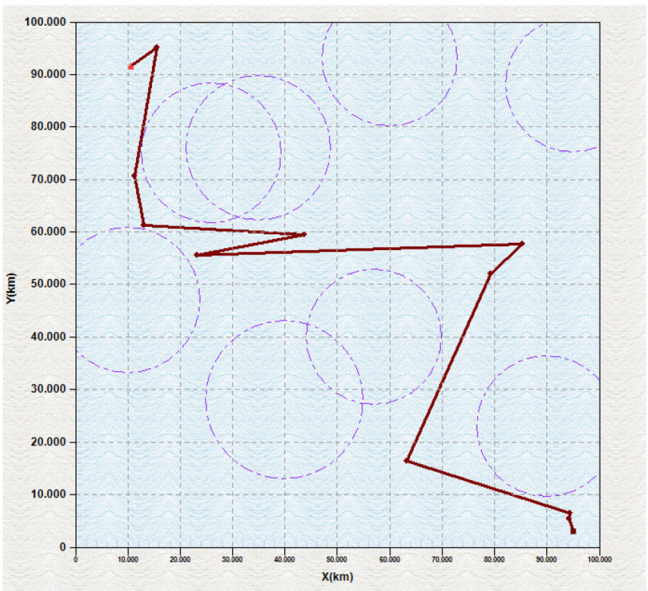
4.2 Construction of Particle

This section of work involves two steps: first, path candidates are obtained by generating random geometric graphs (RGG). Second, these path candidates are transformed into particles, utilizing the variable-length particle representation.

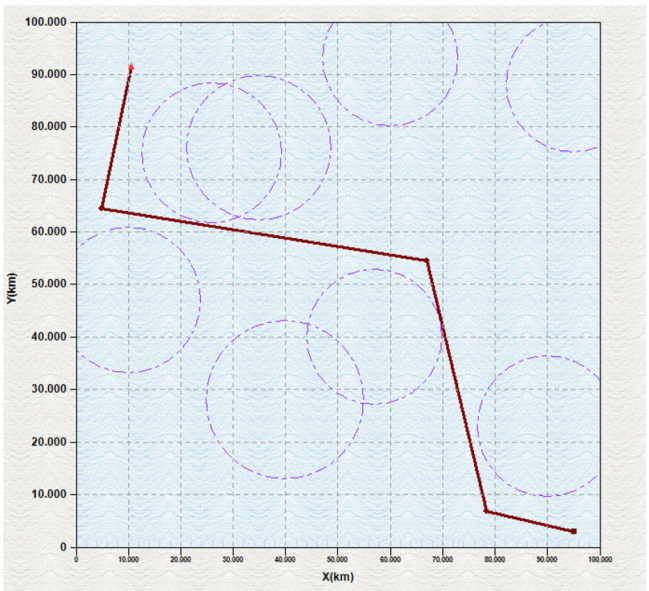
Typically, existing methods assume that particles representing paths are constructed as fixed-length sequences with randomly generated elements. On one hand, within practical planning contexts, accurately determining the optimal number of path's waypoint beforehand is problematic. This issue renders the method of predefining a fixed-length vector for path representation as inflexible and impractical. On the other hand, ensuring path quality through randomly generated elements becomes a significant challenge. Moreover, in certain complex scenarios, the probability of identifying a feasible solution becomes exceedingly low. Specifically, if a larger number of particle dimension(waypoints) are selected in simple scenarios, i.e., scenarios with fewer obstacles, it will lead to waypoints redundancy, thereby reducing the search efficiency of the algorithm. Conversely, if fewer waypoints are selected in complex scenarios, it will lead to insufficient particle dimensions, thereby reducing the quality of the generated



(a) Appropriate Particle Dimension, $N = 4$



(b) High Particle Dimension, $N = 10$



(c) Low Particle Dimension $N = 3$

Fig. 2 Demonstrations of different particle dimensions solving the same scenario, where red line represents the path, purple circles and polygons represent the obstacles

path, and may even result in the inability to find a feasible path. As shown in Fig. 2a displays a high-quality path in this scenario, while Fig. 2b shows a low-quality path generated when the particle dimensions are too long, and Fig. 2c shows a low-quality path generated when the particle dimensions are insufficient.

To address these issues, we argue that directly generating each sequence element is inflexible. This method implicitly assumes that the randomly generated nodes will fit the path's curve precisely, which becomes less possible as the number of nodes increases. Instead of using a sequence, we construct a random geometric graph and generate the path from the graph to ensure path quality. However, a challenge arises in that it is difficult to forecast the number of key nodes in the generated path. This unpredictability leads to variable dimensions in the resulting particles, complicating subsequent optimization processes. Thus, a variable-length

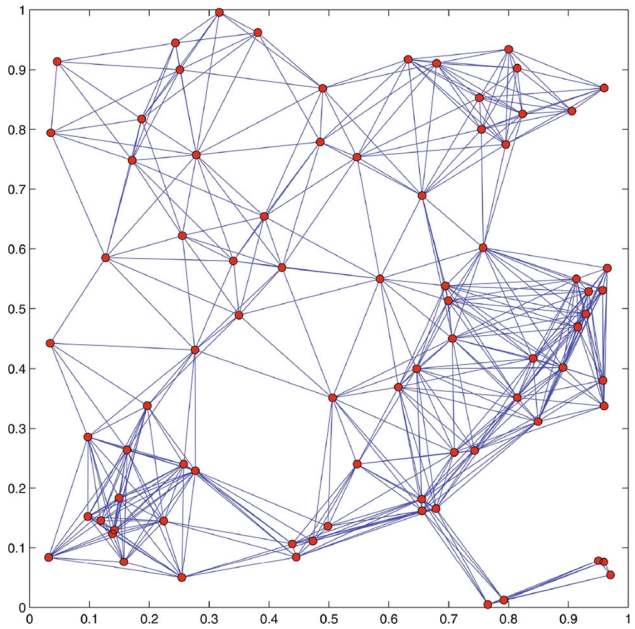


Fig. 3 Demonstrations of random geometric graphs [39]

particle representation is proposed, which converts the path into particles adaptable in dimension.

4.2.1 Generation of Random Geometric Graphs Based Path Candidates

Random geometric graphs (RGG) consist of randomly distributed points within a metric space, where edges connect pairs of points that meet specific criteria, such as distance thresholds. Figure 3 shows a random geometric graph embedded in $X = [0, 1]^2$, with $n = 80$ vertices and radius $\rho = 0.25$ [39].

Specifically, the work in this part includes the following steps:

- Sample N random nodes as candidate waypoints for the path, making sure they avoid obstacle zones. In this context, N denotes the PSO particle vector's length.
- In the RGG framework, form edges in the graph using the randomly generated points combined with the start and end points. An edge here is defined as a direct line between any two nodes.
- Utilize Dijkstra algorithm to generate path candidates from RGG, initializing particles of PSO.

By shifting from creating lists (randomly generated paths) to sets (RGGs), solutions are free from sequential order constraints. Consequently, this enhances the quality of the initial particles, and thus increases the overall efficiency of the search.

4.2.2 Designation of Variable-Length Particle

This section primarily focuses on the construction of PSO particles. Initially, a particle is conceptualized as a set of fixed-length vectors, with each element of the vector representing the coordinates of a point. The starting and ending points are then added to either end of the vector to form a complete path. To reiterate, the vector length, denoted as N, indicates the maximum number of nodes in the path. In this study, the length of the vector corresponds to the maximum number of nodes the path can hold. In cases where the actual path length is insufficient to fill the vector, a method of randomly selecting nodes from the path and replicating subsequent nodes to fill the vector is employed. As Fig. 4 shows, consider a vector length of 10. Suppose the actual path contains only 7 waypoints, denoted as $\sigma = [x_0, x_1, \ldots, x_6]$. In this case, 3 additional waypoints, denoted as x_0, x_2, x_4, are randomly selected from the existing path. Finally, these selected waypoints are replicated and sequentially inserted to fill the vector, as the particle $P = [x_0, x_0, x_1, x_2, x_2, x_3, x_4, x_4, x_5, x_6]$. When interpreting the particle as a path, if a waypoint is identical to its

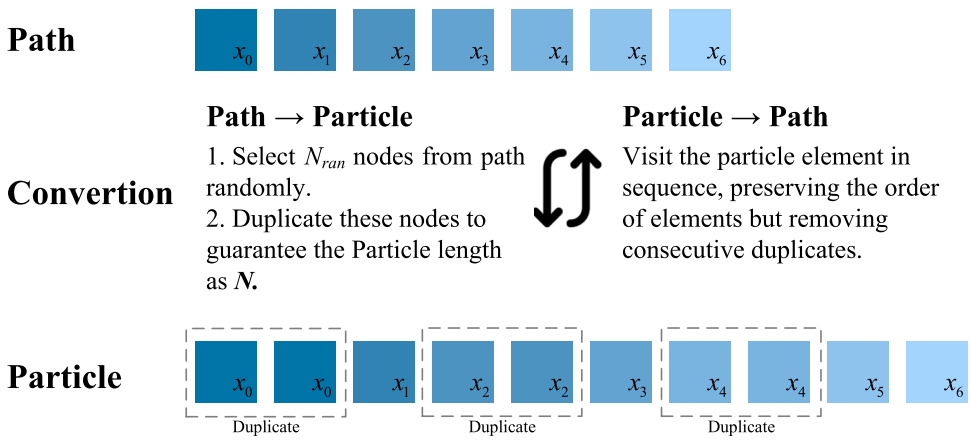


Fig. 4 Conversion between paths and particles

predecessor, it is identified as a filler waypoint and consequently removed from the path.

Specifically, the process of obtaining the particle from the path is described as Algorithm 1:

Algorithm 1: Path to Particle Converter

Input: *path*

Output: *particle*

$N_{dup} \leftarrow \left\lceil \frac{N-\text{len}(path)}{\text{len}(path)} \right\rceil$

$N_{ran} \leftarrow N - \text{len}(path) \times N_{dup}$

Select N_{ran} nodes from *path* randomly, denoted as *Nodes*

foreach $node \in path$ **do**

  if $node \in Nodes$ **then**

    for $i \leftarrow 0$ **to** N_{dup} **do**

      $particle$.append($node$)

  else

    for $i \leftarrow 0$ **to** $N_{dup} - 1$ **do**

      $particle$.append($node$)

As Fig. 4 shows, the transformation procedure of path to particle is illustrated in detail. Algo.1 transforms a **path** into a **particle**. Here, a systematic approach is employed, involving calculated duplication of elements from the original **path**. The pseudo-code outlines the specific logic behind this transformation, detailing how elements from the **path** are selected, duplicated, and assembled into the **particle** vector, thereby achieving the required size with an optimal and efficient methodology.

The designation of Algorithm 1 is instrumental in ensuring that the generated particles uniformly gravitate towards various nodes of both the personal best (pbest) and the global best (gbest) during the iterative process. In PSO, the ability to explore the vicinity of pbest and gbest is essential for finding optimal solutions. Each particle's movement, influenced by both its own experience (pbest) and the collective experience of the swarm (gbest), is crucial in navigating the search space. The proposed algorithm ensures that the exploration is not biased towards any specific region, but rather covers the search space more uniformly. As a result, through successive iterations, the algorithm can more effectively probe different areas, enhancing the likelihood of identifying the most optimal solutions within the search space.

Algorithm 2 illustrates the reverse process, transforming a particle into a path. Since the particle is a vector of fixed length, the path is obtained by removing the duplicate elements from the particle. Specifically, for each element in the particle array (referred to by index i), the algorithm checks if the current element (particle[i]) is equal to the previous element (particle[i-1]). Finally, a path that consists of elements from the particle where consecutive duplicate elements are removed. Essentially, it creates a version of particle with no immediate repetitions.

Algorithm 2: Particle to Path Converter

Input: *particle*

Output: *path*

$path$.append($particle[0]$)

for $i \leftarrow 1$ **to** $len(particle) - 1$ **do**

  if $particle[i] \neq particle[i-1]$ **then**

    $path$.append($particle[i]$)

4.3 Divide-and-Conquer Paradigm of PSO-Based Path Planning

This part first introduces the basic optimization process of PSO for path planning problems. Based on the introduced optimization process, the involvement of the divide-and-conquer paradigm is elucidated.

4.3.1 PSO Process

PSO is a computational method used to optimize a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. First, the definition of PSO is introduced as follows.

- **Particle:** Denoted as P^i, where i is the index of the particle. In this paper, a particle is represented as a vector, which is encoded as a path, as Fig. 4 shown.
- **Position:** The position of each particle in the search space, represented by a vector $\mathbf{x}^i$, where i is the index of the particle. The element of vector $\mathbf{x}^i$ is a waypoint in the path, denoted as x_j^i with the element index j. As the example shown in Sect. 4.2.2, Particle P^i is a vector of fixed-length N, where each element of the vector represents the coordinates of a waypoint, denoted as $P^i = [x_0^i, x_1^i, \ldots, x_N^i]$.
- **Velocity:** The movement velocity of particle P^i, represented by a vector $\mathbf{v}^i$.
- **Personal Best:** The best position that the particle P^i has found in its history, denoted as $\mathbf{p}^{*i}$.
- **Global Best:** The best position found by any particle in the swarm, represented as $\mathbf{g}^*$.

Secondly, Here's an overview of the process, including the update rules, objective function, and termination condition.

- **Update Rules:**

$$\begin{aligned}\mathbf{x}^{i,t+1} &= \mathbf{x}^{i,t} + \mathbf{v}^{(i,t)} \\ \mathbf{v}^{i,t+1} &= w\mathbf{v}^{i,t} + c_1 r_1 \left(\mathbf{p}^{*i} - \mathbf{x}^{i,t}\right) \\ &\quad + c_2 r_2 \left(\mathbf{g}^{*} - \mathbf{x}^{i,t}\right)\end{aligned} \tag{1}$$

  Here, i is the index of the particle, t represents the iteration index, w is the inertia weight, c_1 and c_2 are learning factors, and r_1 and r_2 are random numbers.
- **Objective Function:** A function used to evaluate the goodness of a particle's position, generally expressed as $c(\mathbf{x})$. In this paper, the path distance is used as the objective function, introduced in Sect. 3.
- **Termination Condition:** The condition for ending the algorithm, which could be reaching a maximum number of iterations, finding a satisfactory solution, or other criteria.

4.3.2 Designation of Divide-and-Conquer Paradigm

From the view of divide-and-conquer paradigm, the PSO process can be divided into two parts, divide and conquer.

Divide The divide part is to divide the original path into two sub-paths, and then optimize the two sub-paths, respectively. Based on the previous introduced PSO process in Sect. 4.3, each sub-path is represented as a particle and optimized. The divide process is illustrated in the following pseudo-code.

Algorithm 3: Divide path into two optimized sub-paths

Input: $path$

Output: $path_1, path_2$

1 $idx \leftarrow \lceil \text{len}(path)/2 \rceil$

2 $path_1, path_2 \leftarrow path[0 : idx + 1], path[idx :]$

3 $path_1, path_2 \leftarrow \text{PSO}(path_1), \text{PSO}(path_2)$

4 **return** $path_1, path_2$

Note that, function PSO in line. 3 of the Algorithm 3 is the process introduced in Sect. 4.3, including the RGG-based candidates generation introduced in Sect. 4.2.

Conquer

In this part, the optimization results of the two sub-paths are merged. It is necessary to consider whether the waypoints of the two sub-paths can skip the selected intermediate waypoint to build a better path, and finally stitch the two sub-paths together. Algorithm 4 is the pseudo-code of the conquer part. Note that, $c(path)$ is the objective function, which is the path distance introduced in Sect. 4.3.1.

Algorithm 4: Conquer algorithm for combining paths

Input: $path_1, path_2$

Output: $path$

```
path* ← []
len* ← ∞
for each node1 in path1 do
    for each node2 in path2 do
        if node1 to node2 is feasible then
            path ← path1[0 : node1] + path2[node2 :]
            if c(path) < len* then
                path* ← path
                len* ← c(path)
return path*
```

Algorithm 4 aims to merge two paths, $path_1$ and $path_2$ optimally. It iterates through each node of $path_1$ and $path_2$, constructing new paths by combining segments of these two paths at feasible transition points. For each new path created, it evaluates its cost and records the path with the lowest cost. The algorithm continues until all node combinations have been evaluated. Finally, the merge part returns $path^*$, the path of the lowest cost among the combinations.

5 Experiments Setting

This section reports on our experiments to assess our proposed method's effectiveness. Section 5.1 introduces the Research Questions, and the setup of the experiment follows in Sect. 5.2.

5.1 Research Questions

RQ1 What is the performance of the proposed RGG-PSO+ method compared to PSO, GA and DE?

RQ2 How does the RGG-based variable-length particle compare with the baseline of a randomly generated path?

RQ3 What is the effectiveness of divide-and-conquer Paradigm in the proposed RGG-PSO+ method?

In RQ1, we aim at evaluating the *effectiveness* of RGG-PSO+, which will be answered in Sect. 6.1. We select 1000 stochastic scenarios with various obstacle distributions to evaluate the performance of the proposed RGG-PSO+ and other candidates, including PSO [40], SPSO [21], GA [40]

and DE [41]. For each scenario, 10 times plannings are executed to obtain the statistical data, including the average and standard deviation values. Detailed settings of the scenario, parameters and evaluation metrics are introduced in Sect. 5.2.

In RQ2, we aim at evaluating the effectiveness of RGG-based variable-length particle, which will be answered in Sect. 6.2. The performance of RGG-based variable-length particle is compared with the baseline of pure randomly generated path. Here, F-Measure is used to indicate the first feasible path generated by RGG and pure random. These two metrics are used to indicate how effectiveness of RGG-based variable-length improvement, which are introduced in Sect. 5.2. Detailedly, the forementioned 1000 stochastic scenarios are selected as the testing cases and 10 times plannings are executed for each scenario. The average F-Measure values are collected.

In RQ3, we aim at evaluating the effectiveness of divide-and-conquer Paradigm, which will be answered in Sect. 6.3. The performance of RGG-PSO+ and RGG-PSO are compared. Here, RGG-PSO+ is the proposed method involving RGG-based variable-length particle and divide-and-conquer Paradigm. RGG-PSO is the RGG-based variable-length particle method without divide-and-conquer Paradigm. The forementioned 1000 stochastic scenarios are selected and 10 times plannings are executed for each. Same as RQ1, the average and standard deviation values are collected.

5.2 Experimental Setup

5.2.1 Scenarios

To validate the effectiveness and efficiency of the RGG-based particle and divide-and-conquer paradigm for PSO-based path planning, we perform simulations in 1000 stochastic scenarios with various obstacle distributions. The range of all scenarios is set to 100×100 with 5–15 obstacles randomly distributed in different sizes. For each obstacle setting, we randomly generate 100 test scenarios. For each scenario, we execute the planning process 10 times to obtain the statistical data, including the average and standard deviation values. Worth mentioning, there always exists a feasible path and an optimal one for any scenario. Besides, we avoid invalid scenarios that the optimal path is the segment connecting the start and the goal.

5.2.2 Parameter Settings

Our experiment selects RGG-PSO+, RGG-PSO, PSO, SPSO, GA, and DE, as the testing candidates to compare the performance of the preprocessor. Here, RGG-PSO+ and RGG-PSO are the proposed methods in this paper, and the other 4 methods are the benchmark methods.

For comparisons, all PSO variants are implemented with the same set of parameters: $w = 1$ with the damping rate of 0.98, $\eta_1 = 1.5$ and $\eta_2 = 1.5$. The swarm size is chosen to be 100 particles and the number of iterations is 200. The number of waypoints are selected as $N = 20$, corresponding 21 line segments.[1] As for PSO, SPSO, GA, and DE, the setting of parameters are the same as the original paper [21, 40, 41].

5.2.3 Evaluation Metrics

Normalized Path Length In different scenarios, the distance of each optimal path varies from another, so we normalized the distance in each scenario for comparison. Specifically, we use the min-max normalization to normalize the length of each path, i.e.,

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)}, \tag{2}$$

where $x = (x_1, ..., x_n)$ represents distances of all paths obtained in a scenario from 10 times execution and z_i is the ith normalized distance. For each scenario, we take the distance of the shortest path as $min(x)$, and the longest path as $max(x)$ in normalization.

F-Measure F-Measure, denoted as M_f, indicates, at each setting of scenario, the number of iterations executed before the first feasible particle obtained. In other words, F-Measure indicate the first feasible path, which navigates from the initiation to the destination without any collision. In this paper, we record the average and variance, including the iteration number of the first feasible particle obtained and the normalized path length of the first feasible particle.

6 Analysis

6.1 RQ1

In this section, we compile statistics on the average normalized path lengths for RGG-PSO+, SPSO, PSO, GA, and DE. For each candidates, we use the same 20 waypoints and 100 population as the settings. As for the individual parameters involved in each algorithm, adjustments are made in accordance with the configurations detailed in the respective research papers (see Sect. 5.1). The results are visually represented in a Fig. 5 for comparison, which shows the best result over iterations where the values obtained from each methods. The figure features the number of iterations on the horizontal axis and the relative lengths on the vertical axis.

[1] A total number of node is 22, including 20 waypoints, start and target position. Therefore, there are 21 line segments connected sequentially from the start to the target.

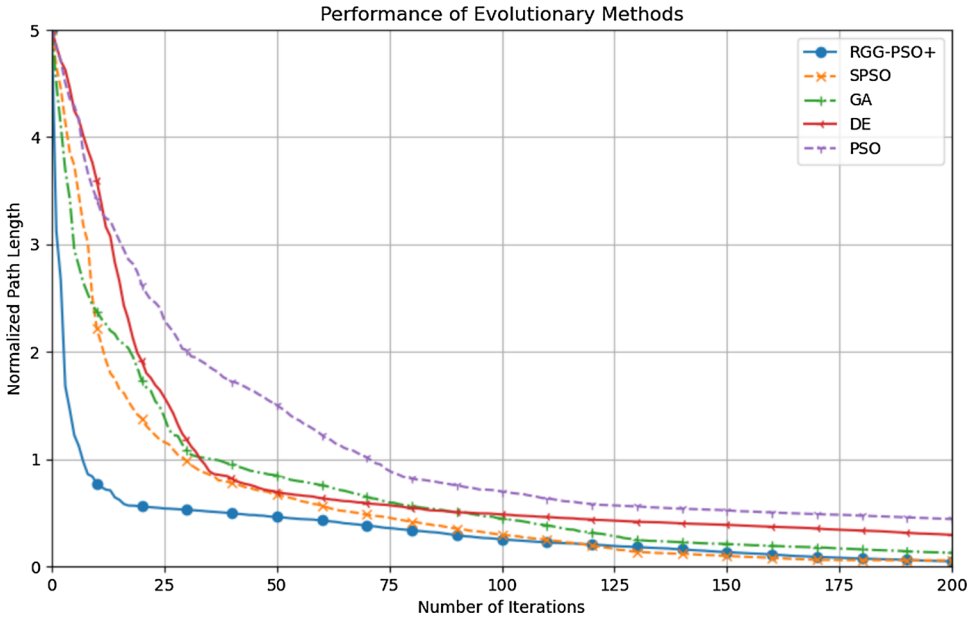


Fig. 5 Best normalized path distance over iterations of RGG-PSO+ and other evolutional methods

Essentially, the path produced by a generated particle is confirmed to be within the [0, 1] range only in the final output results.

Worth mentioning, the calculation of normalized path length involves the use of $min(\cdot)$ and $max(\cdot)$ functions, derived from 10 times execution of the candidate methods' results. In other words, $min(\cdot)$ and $max(\cdot)$ record the final output of the algorithm (the global best particle g^*), not the optimal solution (p^{*i}) for each particle p at each stage i. Although g^{*i} is monotonically decreasing and gradually converges to g^*, there is no guarantee that g^{*i} is smaller than $max(\cdot)$. It is possible that z_i is greater than 1. Besides, a fixed penalty value is assigned to the normalized path length when the iteration number is 0, due to the initialization phase of the methods where the absolute path lengths are infinite. Accordingly, a penalty value of 5 is set for the normalized path length.

Figure 5 reveals that RGG-PSO+ significantly outperforms other methods in terms of convergence speed, which is largely attributed to the RGG method's effective optimization of initial particles (see Sect. 6.2). Furthermore, RGG-PSO+ excels commendable results in path optimization quality. Although SPSO yields a better optimal solution with a higher iteration count compared to RGG-PSO+, integrating RGG and divide-and-conquer strategies with the fundamental PSO method shows promise for further enhancements. However, the challenge lies in integrating these strategies with SPSO's spherical vector-based particles, as the direct application of the RGG method is not feasible. This issue remains an open question.

6.2 RQ2

In this part, we aim to indicate the F-Measure and normalized path length of the first feasible path generated by RGG improvement and baseline PSO, denoted as RGG-PSO and PSO, respectively. Here, the calculation of normalized path length requires $min(\cdot)$ and $max(\cdot)$, which are obtained from 10 times execution of RGG-PSO and PSO. Note that, since we compare the path distance encoded by the particle, which is the candidates of the final result, the normalized length may bigger than 1. In this case, the path of generated particle is longer than the worst result of executions.

For both candidates, we use the same parameter settings, and the length of the particle is set to 20, which indicates the particle have 20 waypoints. From Table 1, we can see that the RGG-PSO improves both the mean and variance of the F-Measure, and the quality of the generated trajectories is significantly improved, as well as the length. Specifically, the mean of F-Measure is improved from 13.984 to 4.139, and the variance is improved from 48.365 to 24.140. As for the quality of the particle, the mean of normalized path length is improved from 3.210 to 0.593, and the variance is improved from 1.555 to 0.074. The improvement indicates that RGG can generate better particles, which obviously speeds up the convergence of the path planning method, and can focus more computation from the exploring process to the exploiting process. Thus, better results can be obtained in the same number of iterations.

Table 1 F-Measure comparison

	F-Measure		Normalized path length	
Method	Mean	Var	Mean	Var
RGG-PSO	4.139	24.140	0.593	0.074
PSO	13.984	48.365	3.210	1.555

Table 2 Divide-and-conquer paradigm comparison

	Normalized path length	
Method	Mean	Var
PSO-RGG+	0.248	0.086
PSO-RGG	0.307	0.089
PSO	0.872	0.781

6.3 RQ3

In this part, we test the divide-and-conquer paradigm to demonstrate its effectiveness in the proposed method. We select RGG-PSO+, RGG-PSO and PSO as test candidates, here RGG-PSO+ is the proposed RGG method with divide-and-conquer paradigm, and RGG-PSO is the proposed RGG method without divide-and-conquer paradigm. The setting of parameters are the same as Sect. 5.2. The same 1000 scenarios are selected and 10 times plannings are executed for each scenario. The number of waypoints encoded by particles is still $N = 20$. Worth mentioning, since RGG-PSO+ involves multiple PSO evolution process, we fix the max iteration number to 200 and compare the result of RGG-PSO+ with the result of RGG-PSO and PSO, which are obtained in the max iteration number. Note that, for RGG-PSO+, we set the iteration number of PSO evaluation for sub-paths to 100.

From Table 2, we can see that the mean and variance of the normalized path length of RGG-PSO+ and RGG-PSO are significantly better than the PSO method. Furthermore, divide and conquer paradigm resulted in a further improvement in the path quality and a slight decrease in the variance of the RGG-PSO+ results. Specifically, the path length mean and variance of RGG-PSO+ are reduced by 0.059 and 0.003, respectively. The experimental result indicates that the strategy is able to improve the path quality with the same number of iterations.

7 Conclusion

In summary, this paper introduces the RGG-PSO+ method for UAV path planning, which integrates random geometric graphs (RGG) with particle swarm optimization (PSO) and incorporates a divide-and-conquer paradigm. RGG-PSO+ dynamically adjusts the number of waypoints in response to the complexity of the scenario, enhancing the method's efficiency and quality of the generated paths. The efficacy of RGG-PSO+ is demonstrated through comparative analysis with other heuristic methods in various complex scenarios. The integration of RGG substantially improves the F-Measure, yielding higher quality initial particles and reflecting a shift in focus from exploration to exploitation. The divide-and-conquer paradigm further elevates the method's effectiveness, as shown by the improved mean and variance in normalized path lengths. In the future, the research plans to explore the integration of RGG with the spherical vector-based particle method (SPSO). The integration combines the strengths of RGG's efficient waypoint adjustment and SPSO's advanced evolution capability, potentially leading to more effective and efficient UAV path planning methods.

Acknowledgements This paper received partial support from the Fundamental Research Funds for the Central Universities (No. 2023JBMC017), and the National Natural Science Foundation of China (No. 6230070656).

Author Contributions Yang Liu initiated the project, wrote the manuscript, and maintained the path planning platform of the experiment; Xiaomin Zhu proposed the main idea and made a revision of the paper; Xiao-Yi Zhang designed the experiment and revised the manuscript; Jiannan Xiao and Xiaohan Yu wrote the code of the experiment program.

Availability of Data and Material The experimental methods employed in this study are publicly available, including PSO [40], SPSO [21], GA [40] and DE [41].

Declarations

Conflict of Interest The authors declare no conflict of interest.

References

1. Liu, Y., Zheng, Z., Qin, F.: Homotopy based optimal configuration space reduction for anytime robotic motion planning. Chin. J. Aeronaut. **34**, 364–379 (2021)
2. Zhao, Y., Zheng, Z., Liu, Y.: Survey on computational-intelligence-based uav path planning. Knowl.-Based Syst. **158**, 54–64 (2018)
3. Reif, J.H.: Complexity of the mover's problem and generalizations. In: 20th Annual Symposium on Foundations of Computer Science (SFCS), pp. 421–427, IEEE Computer Society, (1979)
4. Tang, G., Tang, C., Claramunt, C., Hu, X., Zhou, P.: Geometric a-star algorithm: an improved a-star algorithm for agv path planning in a port environment. IEEE Access **9**, 59196–59210 (2021)
5. Erke, S., Bin, D., Yiming, N., Qi, Z., Liang, X., Dawei, Z.: An improved a-star based path planning algorithm for autonomous land vehicles. Int. J. Adv. Robot. Syst. **17**(5), 1729881420962263 (2020)
6. Dolgov, D., Thrun, S., Montemerlo, M., Diebel, J.: Path planning for autonomous vehicles in unknown semi-structured environments. Int. J. Robot. Syst. **29**(5), 485–501 (2010)
7. Xie, Z., Cheng, L., Li, X., Chen, X.: A directed jump point search with improved preprocess for path planning. In: 2023 IEEE 18th Conference on Industrial Electronics and Applications (ICIEA), pp. 1333–1338, IEEE, (2023)
8. Sturtevant, N.R., Rabin, S.: Canonical orderings on grids. In: International Joint Conference on Artificial Intelligence (IJCAI), pp. 683–689, AAAI, (2016)
9. Solovey, K., Salzman, O., Halperin, D.: New perspective on sampling-based motion planning via random geometric graphs. Int. J. Robot. Syst. **37**(10), 1117–1133 (2018)
10. Karaman, S., Frazzoli, E.: Sampling-based algorithms for optimal motion planning. Int. J. Robot. Res. (IJRR) **30**(7), 846–894 (2011)
11. Pehlivanoglu, Y.V., Pehlivanoglu, P.: An enhanced genetic algorithm for path planning of autonomous uav in target coverage problems. Appl. Soft Comput. **112**, 107796 (2021)
12. Arantes, M.d.S., Arantes, J.d.S., Toledo, C.F.M., Williams, B.C.: A hybrid multi-population genetic algorithm for uav path planning. In: Proceedings of the Genetic and Evolutionary Computation Conference 2016, pp. 853–860, (2016)
13. Zheng, J., Ding, M., Sun, L., Liu, H.: Distributed stochastic algorithm based on enhanced genetic algorithm for path planning of multi-uav cooperative area search. IEEE Trans. Intell. Transport. Syst. **24**(8), 8290–8303 (2023)
14. Zhang, M., Han, Y., Chen, S., Liu, M., He, Z., Pan, N.: A multi-strategy improved differential evolution algorithm for uav 3d trajectory planning in complex mountainous environments. Eng. Appl. Artif. Intell. **125**, 106672 (2023)
15. Zhang, X., Zhang, X., Miao, Y.: Cooperative global path planning for multiple unmanned aerial vehicles based on improved fireworks algorithm using differential evolution operation. Int. J. Aeronaut. Sp. Sci. **24**(5), 1346–1362 (2023)
16. Li, J., Xiong, Y., She, J.: Uav path planning for target coverage task in dynamic environment. IEEE Internet of Things J. **10**(20), 17734–17745 (2023)
17. Huang, C., Zhou, X., Ran, X., Wang, J., Chen, H., Deng, W.: Adaptive cylinder vector particle swarm optimization with differential evolution for UAV path planning. Eng. Appl. Artif. Intell. **121**, 105942 (2023)
18. Tang, B., Xiang, K., Pang, M., Zhanxia, Z.: Multi-robot path planning using an improved self-adaptive particle swarm optimization. Int. J. Adv. Robot. Syst. **17**(5), 1729881420936154 (2020)
19. Hoang, V., Phung, M.D., Dinh, T.H., Ha, Q.P.: Angle-encoded swarm optimization for uav formation path planning. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5239–5244, IEEE, (2018)
20. Maina, R.M., Lang'at, P.K., Kihato, P.K.: Collaborative beamforming in wireless sensor networks using a novel particle swarm optimization algorithm variant. Heliyon **7**(10), e08247 (2021)
21. Phung, M.D., Ha, Q.P.: Safety-enhanced uav path planning with spherical vector-based particle swarm optimization. Appl. Soft Comput. **107**, 107376 (2021)
22. Zhang, Y., Wang, S., Ji, G., et al.: A comprehensive survey on particle swarm optimization algorithm and its applications. Math. Probl. Eng. **2015**, 931256 (2015)
23. Khandelwal, M.K., Sharma, N.: A survey on particle swarm optimization algorithm. In: International Conference on Communication and Computational Technologies, pp. 591–602, Springer, (2023)
24. Liu, W., Wang, Z., Yuan, Y., Zeng, N., Hone, K., Liu, X.: A novel sigmoid-function-based adaptive weighted particle swarm optimizer. IEEE Trans. Cybern. **51**(2), 1085–1093 (2021)
25. Song, B., Wang, Z., Zou, L.: An improved pso algorithm for smooth path planning of mobile robots using continuous high-degree Bezier curve. Appl. Soft Comput. **100**, 106960 (2021)
26. Fu, Y., Ding, M., Zhou, C.: Phase angle-encoded and quantum-behaved particle swarm optimization applied to three-dimensional route planning for uav. IEEE Trans. Syst. Man Cybern. Part A Syst. Humans **42**(2), 511–526 (2012)
27. Fu, Y., Ding, M., Zhou, C., Hu, H.: Route planning for unmanned aerial vehicle (uav) on the sea using hybrid differential evolution and quantum-behaved particle swarm optimization. IEEE Trans. Syst. Man Cybern. Part A Syst. Humans **43**(6), 1451–1465 (2013)
28. Zhang, Y., Wu, L., Wang, S., et al.: Ucav path planning by fitness-scaling adaptive chaotic particle swarm optimization. Math. Probl. Eng. **2013**, 705238 (2013)
29. Liang, B., Zhao, Y., Li, Y.: A hybrid particle swarm optimization with crisscross learning strategy. Eng. Appl. Artif. Intell. **105**, 104418 (2021)
30. Zhong, J., Li, B., Li, S., Yang, F., Li, P., Cui, Y.: Particle swarm optimization with orientation angle-based grouping for practical unmanned surface vehicle path planning. Appl. Ocean Res. **111**, 102658 (2021)
31. Penrose, M.: Random Geometric Graphs. Oxford University Press, 05 (2003)
32. Bohlin, R., Kavraki, L.: Path planning using lazy prm, vol. 1 of Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065), pp. 521–528, IEEE, (2000)
33. Mandalika, A., Choudhury, S., Salzman, O., Srinivasa, S.: Generalized lazy search for robot motion planning: Interleaving search and edge evaluation via event-based toggles, vol. 29 of Proceedings of the International Conference on Automated Planning and Scheduling, pp. 745–753, (2019)
34. Yang, L., Zheng, Z., Fangyun, Q.: Homotopy based optimal configuration space reduction for anytime robotic motion planning. Chin. J. Aeronaut. **34**(1), 364–379 (2021)
35. Gammell, J.D., Strub, M.P.: Asymptotically optimal sampling-based motion planning methods. Annu. Rev. Control Robot. Auton. Syst. **4**, 295–318 (2021)
36. Janson, L., Schmerling, E., Clark, A., Pavone, M.: Fast marching tree: a fast marching sampling-based method for optimal motion planning in many dimensions. Int. J. Robot. Res. (IJRR) **34**(7), 883–921 (2015)
37. Gammell, J.D., Barfoot, T.D., Srinivasa, S.S.: Batch informed trees (BIT*): informed asymptotically optimal anytime search. Int. J. Robot. Res. (IJRR) **39**(5), 543–567 (2020)
38. LaValle, S.M.: Planning Algorithms. Cambridge University Press (2006)
39. Sakaridis, C., Drakopoulos, K., Maragos, P.: Theoretical analysis of active contours on graphs. SIAM J. Imag. Sci. **10**(3), 1475–1510 (2017)

40. Roberge, V., Tarbouchi, M., Labonte, G.: Comparison of parallel genetic algorithm and particle swarm optimization for real-time uav path planning. IEEE Trans. Ind. Inf. **9**(1), 132–141 (2013)
41. Storn, R., Price, K.: Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J. Global Optim. **11**, 341–359 (1997)