Safe Path Planning of UAV Based on Comprehensive Improved Particle Swarm Optimization Algorithm

Shushu Zhang[1], Yu Xia[1], Weiwei Qi[1], Shanjun Zhang[3], Liucun Zhu[1, 2, 3, *]

[1]*School of Information Engineering, Yangzhou University, Yangzhou225000, China*
[2]*Advanced Science and Technology Research Institute, Beibu Gulf University, Qinzhou535011, China*
[3]*Research Institute for Integrated Science, Kanagawa University, Kanagawa 259-1293, Japan*

Abstract
Path planning is one of the important links in the control process of UAVs. However, the standard particle swarm optimization (PSO) algorithm has the shortcomings of slow convergence speed and easy falling into the "premature" phenomenon in UAV path planning. To solve this problem, this paper proposes an IPSO algorithm based on integrated improved PSO. A fitness function considering constraints including path minimization and smoothness is formulated in the configuration space to describe the path planning problem of UAVs. By introducing chaos initialization, the quality of particle distribution is improved; global search and local search power of particles are balanced with the help of dynamic inertia weights and learning factors. And the Cauchy variational operator is introduced for avoiding local optimal solutions and accelerating the convergence of the algorithm. Simulation results show that the IPSO algorithm is stable with fast convergence and a small path cost while avoiding obstacles.

Keywords
Path planning, PSO, Configuration space, Chaos theory, Self-adaption inertia weight, Cauchy mutation

1. Introduction

With the development of the mobile robot industry, UAVs are widely used in military or civilian fields such as security inspection[1], emergency rescue[2], logistics distribution[3], agricultural irrigation[4], etc. due to their high flexibility, strong maneuverability, and low cost[5]. The applications of UAVs in the above aspects all involve UAV path planning. The so-called path planning [6] is to find an optimal or approximately optimal path from the starting point to the endpoint in an environment containing obstacles, and satisfy certain constraints, such as the shortest path, the least time-consuming and the highest security.

Due to the traditional algorithms such as A* algorithm [7] and artificial potential field method [8], there are problems such as large amount of calculation, large memory occupation, complex process and poor efficiency in solving 3D paths. With the emergence of intelligent optimization algorithms, more and more researches focus on intelligent algorithms and their improved methods, and they are applied to solve the problem of robot path planning[9]. Particle swarm optimization was first proposed by Kennedy and Eberhart [10] in November 1995. The model was derived from the study of bird predation behavior in nature. Abstract each bird as a particle, each particle is a potential solution to the problem[11]. The particle evaluates the position through the fitness function and shares the information of the best position with the local neighboring particles. At the same time, the information is used to update the velocity and position, and the global optimal solution is obtained through iteration. PSO algorithm has the characteristics of simple operation, clear thinking, easy implementation ,and high efficiency, but it is also easy to have shortcomings such as long running time, unsmooth path, and easy to fall into "premature" phenomenon [12].

ICCEIC2022@3rd International Conference on Computer Engineering and Intelligent Control
EMAIL: *Corresponding author's email: lczhu@bbgu.edu.cn (Liucun Zhu)

In response to these problems in PSO, researchers have made many improvements in initialization, particle position and velocity update rules, adjusting parameters and combining them with other optimization strategies. Hu et al. [13] introduced the weight factor to improve the traditional objective function and introduced the crossover operator of the genetic algorithm into the PSO algorithm, which increased the diversity of the population, but the operation effect was unstable. Zhang et al. [14]used the natural selection mechanism of survival of the fittest to obtain a near-optimal solution for UAV trajectory planning, but because it reduces the diversity of the PSO population, it is not conducive to finding the global optimal solution. Dewang et al. [15] used the distance between the robot and the obstacle to construct a new objective function, so that the robot successfully avoided the obstacle, but there were still problems such as slow convergence speed and so on.

Given the above problems, an improved particle swarm algorithm with faster convergence speed and better path cost is proposed in this study. Firstly, the obstacles are inflated and a 3D environment model of UAV flight is established in the configuration space. The adaptation function is constructed to increase the safety of UAV flight and the smoothness of the path by comprehensively weighing the influencing factors such as path length, obstacle collision, altitude change, and path smoothness. Then, chaos initialization is introduced to improve the quality of particle distribution and increase the stability of the algorithm. Dynamic inertia weights and learning factors are set to balance the global search and local search power of particles and improve the convergence speed. The Cauchy variational operator is introduced to improve the diversity of the population and avoid generating local optimal solutions. Finally, our simulation experiments show that the proposed method has a high success rate of planning results, fast convergence speed, small track cost, and the effectiveness and stability of the algorithm have been improved.

2. Path planning problem description

In the path planning of UAV, some influencing factors need to be considered, such as the path length of UAV flight, obstacle threat, flight height and so on. The path planning problem of UAV is expressed through the fitness function, which can calculate the cost of the path, and compare the cost value of different paths to judge the quality of the path[16]. The constraint functions are as follows:

2.1. Constraint description

The planned path needs to be optimal under certain criteria according to different application scenarios and purposes. In this study, we choose to minimize the path length, and let the Euclidean distance of two path nodes be expressed as $l_{ij} = \|P_{ij}P_{i,j+1}\|$, and the cost function of the path length be:

$$F_1(X_i) = \sum_{j=1}^{n-1} \|P_{ij}P_{i,j+1}\| \qquad (1)$$

In UAV path planning, it is also necessary to guide the UAV safely through the obstacles caused by the threats. Let K be the collection of all threats, and each threat is represented by a cylinder [17] with the central coordinates of the projection of the obstacle being C_k and the radius being R_k. In order to make the threat cost of the UAV more accurate, the size of the UAV is considered here, and the diameter size of the UAV is set as D. Introduce the configuration space and expand the range of obstacles based on the size of the drone in the environment map to obtain the collision zone. In this environment, the robot can be used directly as a particle for path planning.

Due to the influence of various external factors and unstable positioning accuracy in the actual flight environment, the UAV still has the probability of collision with obstacles, so these factors are taken into account and the secondary expansion is carried out. The distance S from the collision zone is referred to as the danger zone. S can choose the length according to the environmental situation as shown in Figure 1.

UAV flying in an obstacle area will encounter three situations: If the UAV is outside the danger zone, its cost is 0; when the UAV enters the danger zone, the collision occurs, and its cost is infinite; if the UAV is between the danger zone and the collision zone, its obstacle threat cost can be calculated as:

$$F_2(X_i) = \sum_{j=1}^{n-1} \sum_{k=1}^{K} (S + D + R_k) - d_k + R_k \quad (2)$$

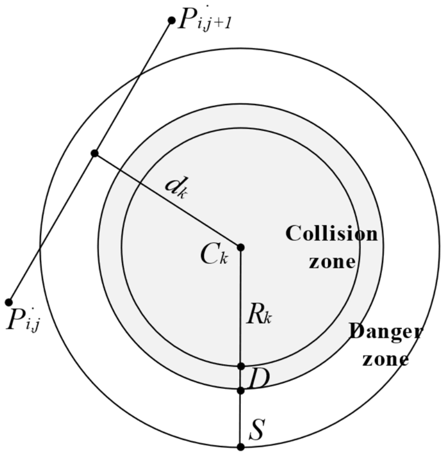

Figure 1 Obstacle cost situation determination.

In order to prevent the UAV from colliding with obstacles, the UAV must increase or decrease its altitude, so it is necessary to constrain the variation of the UAV's flying altitude. The variance of the path height can describe the stability of the flight altitude, and it can be computed as the height change cost as follows：

$$F_3(X_i) = \sqrt{\frac{1}{N}\sum_{d=1}^{D}\left(z_d - \frac{1}{N}\sum_{d=1}^{D} z_d\right)^2} \qquad (3)$$

Where, D represents the total number of track nodes, and z_d is the height value at the *dth* track node.

For UAV, it is not only necessary to shorten the path length as much as possible, but also to reduce the ups and downs of the path in complex environments to make it go to the termination point smoothly. The smoothness of the path is determined by the angle between two imaginary lines connected by two continuous positions of the target point and the UAV in the iterative process. Calculating the smoothing cost requires calculating the turning angle and climbing angle. The turning angle σ_{ij} is the angle between the horizontal plane projections of two consecutive path segments, Path segment $l_{i,j} = P_{ij}P_{i,j+1}$ and $l_{i,j+1} = P_{i,j+1}P_{i,j+2}$ in the horizontal plane projection vector respectively to remember $l_{i,j}^{'} = P_{ij}^{'}P_{i,j+1}^{'}$ and $l_{i,j+1}^{'} = P_{i,j+1}^{'}P_{i,j+2}^{'}$. Then the turning Angle can be computed as:

$$\varphi_{ij} = \arctan\left(\frac{\left\|l_{i,j}^{'} \times l_{i,j+1}^{'}\right\|}{l_{i,j}^{'} \cdot l_{i,j+1}^{'}}\right) \qquad (4)$$

The climb angle is the angle at which the current waypoint is pitched vertically to the next waypoint. If the vertical height difference between two adjacent waypoints is $z_{i,j+1} - z_{i,j}$, then the climbing angle constraint can be computed as:

$$\theta_{ij} = \arctan\left(\frac{z_{i,j+1} - z_{i,j}}{\left\|l_{i,j}^{'}\right\|}\right) \qquad (5)$$

The path smoothing cost associated with turning and climbing angles can be expressed as:

$$F_4(X_i) = a_1 \sum_{j=1}^{n-2} \varphi_{ij} + a_2 \sum_{j=1}^{n-1} \left|\theta_{ij} - \theta_{i,j-1}\right| \quad (6)$$

Here, a_1 and a_2 are the penalty coefficients for the turning angle and the climb angle.

2.2. Overall cost function

In this paper, four constraints are weighed in this paper: path length, obstacle threat, flight altitude, and path smoothing. Define the total fitness function according to equations (1)-(6) as:

$$F(X_i) = \sum_{k=1}^{4} w_k F_k(X_i) \qquad (7)$$

Where w_1, w_2, w_3 and w_4 are constants, which represent the weight values of different costs, and their proportions are related to the tasks performed by the UAV. $F(X_i)$ is the total cost. $F_1(X_i)$ is the path length cost; $F_2(X_i)$ is the obstacle threat cost; $F_3(X_i)$ is the flight altitude cost; $F_4(X_i)$ is the path smoothing cost.

3. Improved PSO path planning

3.1. Standard PSO path planning

For the three-dimensional space path planning problem discussed in this paper, suppose that in a D-dimensional search space, the total number of particles is N, and the position of the ith particle is represented by a d-dimensional vector: $X_i = (x_{i1}, x_{i2}, \cdots, x_{iD})$, The velocity of the ith particle is also a d-dimensional vector, denoted as $V_i = (v_{i1}, v_{i2}, \cdots, v_{iD})$. Particles have the ability to remember, which can save the optimal position $P_{best,i} = (p_{i1}, p_{i2}, \cdots, p_{iD})$ during the ith particle iteration, and also have social learning ability, which can share the optimal position $G_{best} = (g_1, g_2, \cdots, g_D)$ among all particles. Then for the ith particle of the kth generation, the velocity and position of the particle are updated according to equations (8) and (9) as follows:

$$v_{ij}^{k+1} = wv_{ij}^{k} + c_1 r_1 \left(p_{ij}^{k} - x_{ij}^{k}\right) + c_2 r_2 \left(g_j^{k} - x_{ij}^{k}\right) (8)$$

$$x_{ij}^{k+1} = x_{ij}^{k} + v_{ij}^{k+1} \qquad (9)$$

where $i = \{1,2, \cdots, N\}$ is the serial number of the particle,$j = \{1,2, \cdots, D\}$ is the dimension. w is an inertial factor that affects the particle's global search ability and local search ability. c_1 and c_2 are individual learning factors and social learning factors, respectively, which affect the ability of particles to acquire information. r_1 and r_2 are random numbers in the range $[0,1]$ and are used to increase search randomness.

Before each particle flight, determine whether v_{ij}^{k} has crossed the set speed range. If crossed, the velocity boundary value is substituted for the current speed. After flying, determine whether x_{ij}^{k} exceeds the maximum search space. If it is exceeded, the boundary value is also replaced by the current value. Update the $P_{best,i}$ and G_{best} in the particle swarm according to the corresponding change in fitness value, and the update equation is:

$$P_{best,i}^{k+1} = \begin{cases} P_{best,i}^{k}, & F\left(x_{ij}^{k}\right) \le F\left(P_{best,i}^{k}\right), \\ x_{ij}^{k+1}, & F\left(x_{ij}^{k}\right) > F\left(P_{best,i}^{k}\right), \end{cases} \qquad (10)$$

$$G_{best}^{k+1} = \begin{cases} G_{best}^{k}, & F\left(P_{best,i}^{k}\right) \le F\left(G_{best}^{k}\right), \\ P_{best,i}^{k}, & F\left(P_{best,i}^{k}\right) > F\left(G_{best}^{k}\right), \end{cases} \qquad (11)$$

3.2. Comprehensively improved PSO

3.2.1. Chaos initialization

The stability and solution of the PSO algorithm are affected by the initial particle distribution. Chaos initialization can improve the quality of particle distribution, speed up algorithm initialization, and improve the stability of particle swarm optimization. Considering the good uniform distribution of

Logistic chaos[18], Logistic chaos is used to initialize the position of the particle swarm. The basic formula is as follows:

$$x_{n+1} = \mu x_n(1 - x_n) \qquad (12)$$

Where x_n represents the nth chaotic variable; μ is a preset constant. When $\mu = 4$, the system is in a chaotic state at [0,1].

3.2.2. Adaptive parameter adjustment

A larger inertia weight is more conducive to global search, while a smaller inertia weight is more conducive to local search. Moreover, the particle has strong nonlinearity in the search process, and the nonlinear adjustment control parameter is beneficial to adjust the local and global search ability of the particle and improve the convergence speed of the algorithm. Therefore, a nonlinear inertia weight is proposed to dynamically adjust the inertia factor during the search process of the algorithm. As shown in Equation (13):

$$w(k) = w_{max} - (w_{max} - w_{min}) \times \left[\frac{2k}{T} - \left(\frac{k}{T}\right)^2\right] (13)$$

Where k is the current iteration number; w_{max} and w_{min} are the maximum and minimum values of inertia weight, respectively. T is the maximum number of iterations.

The learning factor has a certain influence on the direction of particle search. By adaptive amplitude modulation design of the learning factors, the particle population can accelerate the approach to the optimal solution in the iterative learning process, and at the same time avoid the population falling into the local optimum. As shown in Equations (14) and (15):

$$c_1 = e^{c_{1min} + \frac{k(c_{1max} - c_{1min})}{T}} \qquad (14)$$

$$c_2 = e^{c_{2min} + \frac{k(c_{2max} - c_{2min})}{T}} \qquad (15)$$

When $c2$ is large, it can guide the search direction of the global optimal solution of the particle, which is suitable for the early iteration and helps to improve the convergence speed. When $c1$ is large, the searchability of particles can be strengthened, and it is generally set in the late iteration to prevent the algorithm from falling into the local optimum. c_{1max},c_{1min} is the maximum and minimum values of individual learning factors; c_{2max},c_{2min} is the maximum and minimum values of the social learning factor.

3.2.3. Cauchy mutation

The traditional PSO algorithm lacks the diversity of the particle population, and the Cauchy mutation[19] is introduced into it, which can generate large disturbance near the current individual particle, expand the search space, increase the population diversity, and avoid falling into the local optimal solution.

If $x \in (-\infty, +\infty)$ satisfies the condition given by equation (16), it becomes a Cauchy distribution.

$$f(x; x_0, \gamma) = \frac{1}{\pi\left[1 + \left(\frac{x - x_0}{\gamma}\right)^2\right]} \qquad (16)$$

In the formula: x_0 is the location of the maximum value of the function, γ is the half the width of the scale parameter at half with x_0. When γ and x_0 are 1 and 0, x satisfy the condition of a probability density function, and equation (17) is its cumulative distribution function.

$$F(x) = \frac{1}{\pi}\arctan(x) + \frac{1}{2} \qquad (17)$$

The inverse function can be derived by inverting equation (17), and then the random numbers generated by the uniform distribution can be generated to obey the random numbers of the Cauchy distribution, as shown in equation (18).

$$CM = \tan\left[\pi\left(rand - \frac{1}{2}\right)\right] \qquad (18)$$

Where CM is the Cauchy variational operator and rand is any real number uniformly distributed in the range of (0,1).

The dynamic adjustment of the Cauchy variation step can make the particles generate perturbation at a larger step in the early stage of the algorithm operation to make the particles jump out of the current position; and accelerate the convergence at a smaller step in the later stage. The update rule for each particle in each iteration is as follows:

$$\acute{X_i} = X_i + X_i \cdot CM \cdot \exp[(1-k)\cdot\beta] \qquad (19)$$

Where X_i is the position of particle i, $\acute{X_i}$ is the position of particle i after passing the Corsi variation, β is a constant coefficient to control how fast or slow the variation step changes, and k is the current number of iterations.

3.3. Algorithm specific steps

The pseudo code of the IPSO algorithm is shown below:

Algorithm 1: Pseudocode of the proposed IPSO algorithm

```
1 Get search map;
2 The velocity of the particle is randomly initialized, and the
   position of the particle is initialized with the Logistic map;
3 for $k$ ←1 to $T$ do
4     Calculate the particle fitness value;
5     foreach $i$ ←1 to $N$ do
6         Find $P_{best,i}$;
7         Find $G_{best}$;
8         Update $CM$ to get $X_{i+1}$;
9         Update the inertia weight and the learning factor;
10        Update the position and velocity of the local best
            particle;
11    end
12    Update the position and velocity of the global best
      particle;
13 end
```

4.Result and Discussion

This experiment simulates UAV path planning in MATLAB R2020b. The environment model for the experiment is a 3D terrain environment built from real digital elevation model maps, and obstacle threats are set up in these scenes.

4.1. Parameter setting

In order to evaluate the performance of IPSO algorithm, three typical PSO variants are selected for comparison with IPSO algorithm in this experiment, which is the traditional particle swarm algorithm PSO, particle swarm algorithm with a linear variation of inertia weights (LDWPSO), and CPSO algorithm with the introduction of linear inertia weights and logistic chaos mapping. The specific parameter settings are shown in Table 1. For the purpose of ensuring the fairness of using different algorithms for experimental comparison, the dimensionality D of the test function is set to 20, the initial population number N is set to 500, the maximum number of iterations T is set to 200, and each algorithm is run 50 times independently.

Table 1 Parameter setting of various algorithms

Algorithm	w	c_1	c_2
PSO	1	1.5	1.2
LDWPSO	[0.4,0.9]	1.5	1.2
CPSO	[0.4,0.9]	1.5	1.2
IPSO	[0.4,0.9]	[1.2,1.5]	[1.2,1.5]

4.2. Experimental results

Table 2 shows the data results of IPSO and the other three algorithms, from which it can be seen that IPSO has the shortest algorithm running time of 7.32 seconds, 76% shorter compared to the PSO algorithm, and at the same time, the optimal value, variance and average value of the adaptation of IPSO algorithm is also the smallest, showing better the path optimality and stability of IPSO algorithm.

Figure 2 shows the number of iterations-adaptation function curves of the optimal UAV path for IPSO and the other three algorithms, which shows that the IPSO algorithm has the lowest adaptation value and better convergence accuracy than the other three algorithms.

The top view of the paths of the four algorithms is shown in Figure 3. Combining the results in Table 2 and Figure 2 shows that all four algorithms can generate feasible paths that satisfy the requirements of path length, threat, and smoothness. However, the PSO algorithm tends to fall into the phenomenon of "premature maturity", and there is no way to find a high-quality solution. The CPSO algorithm introduces chaos theory, which makes the algorithm more stable. IPSO algorithm uses Cauchy mutation to increase the diversity of particle population, and can accurately obtain the approximate optimal solution.

Table 2 Algorithm data comparison

Algorithm	Average running time /s	Variance of fitness	Optimal fitness	Average fitness
PSO	30.93	3028.45	6111.40	6367.56
LDWPSO	23.48	1807.60	5859.17	5908.98
CPSO	13.34	926.80	5275.81	5309.27
IPSO	7.32	210.65	5088.47	5103.22

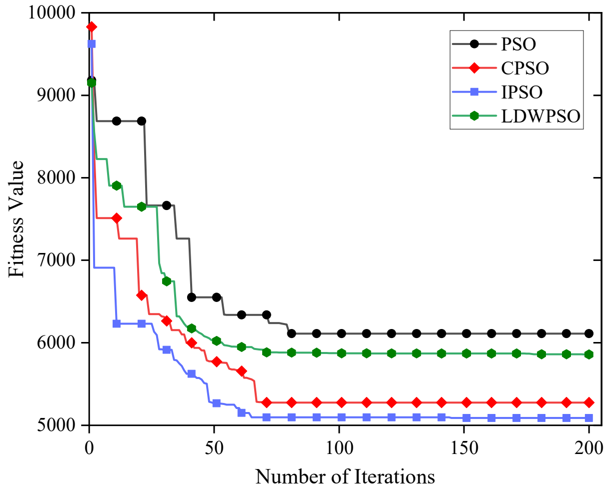

Figure 2 Best fitness values over iterations

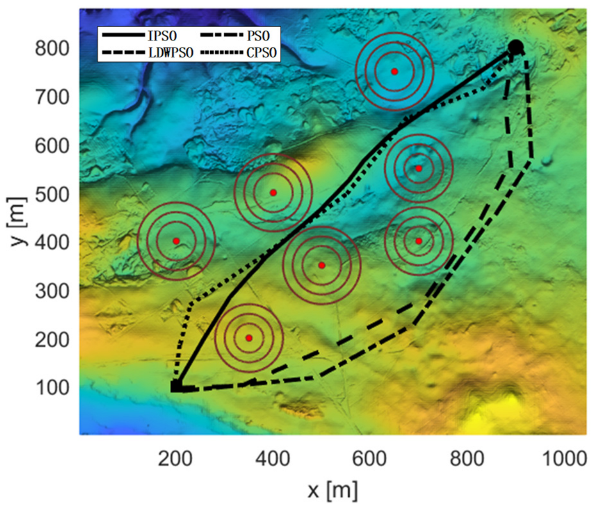

Figure 3 Top view of the paths of the four algorithms

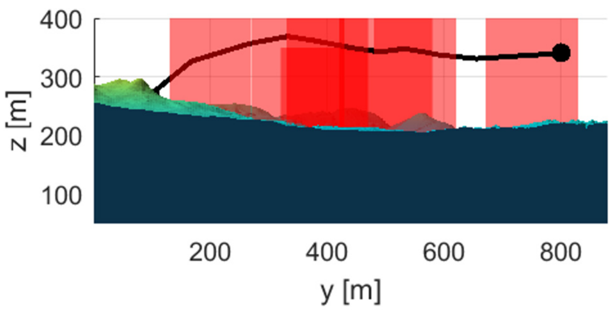

Figure 4 3D path side view of the IPSO algorithm

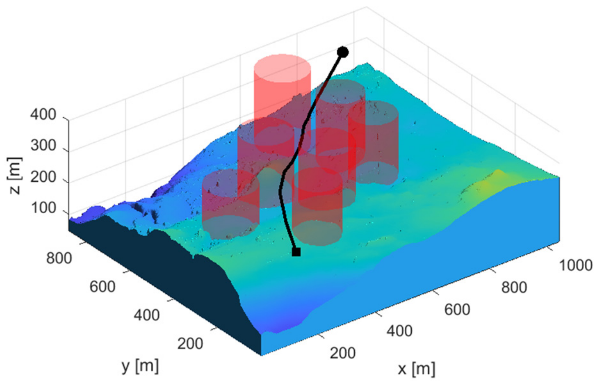

Figure 5 3D path map of the IPSO algorithm

Figure 4 and Figure 5 show the 3D side and map view of the UAV flight path planning using the IPSO algorithm. It can be seen that the path is smooth and collision-free, and the flight height is also appropriate according to the terrain. Compared with the other three PSO variants, the IPSO algorithm can find the optimal flight path and has a suitable altitude during the flight. The IPSO algorithm helps to solve the problems of slow convergence, unstable algorithm, and easy to fall into the "premature" phenomenon.

5. Conclusion

This paper proposes an IPSO algorithm to improve the traditional particle swarm optimization algorithm in path planning, which has the problems of long initialization time, slow convergence speed and easy to fall into local optimal solution. The cost function with various constraints is designed in the configuration space; chaotic initialization is introduced to increase the stability of the algorithm; dynamic inertia weights and learning factors are set to improve the convergence efficiency, and the Cauchy variational operator is used to increase the population diversity. Through experimental simulation, it can be seen that the improved algorithm has the effect of a stable algorithm, fast convergence, and a better path while avoiding obstacles safely. This paper addresses the single UAV path planning problem in the presence of static obstacles, but in the actual environment there may be dynamic obstacles, and the problem of multi-UAV interaction also needs to be considered, so future research will further consider the multi-UAV collaborative planning problem under the influence of dynamic obstacles and other effects, and propose corresponding solutions.

6.Acknowledgements

This research was funded by National Project of Foreign Experts (No.G2022033007L), The Bagui Scholars Program of Guangxi Zhuang Autonomous Region(No.2019A08).

7. References

[1] D. Liu, J. Chen, D. Hu, Z.J.C.i.I. Zhang, Dynamic BIM-augmented UAV safety inspection for water diversion project, Computers in Industry, 108 (2019) 163-177.

[2] S.H. Alsamhi, A.V. Shvetsov, S. Kumar, S.V. Shvetsova, M.A. Alhartomi, A. Hawbani, N.S. Rajput, S. Srivastava, A. Saif, V.O.J.D. Nyangaresi, UAV computing-assisted search and rescue mission framework for disaster and harsh environment mitigation, Drones, 6 (2022) 154.

[3] K. Han, H.J.E. Jung, T. Trends, Trends in logistics delivery services using UAV, Electronics and Telecommunications Trends, 35 (2020) 71-79.

[4] M.E. Karar, F. Alotaibi, A.A. Rasheed, O.J.a.p.a. Reyad, A pilot study of smart agricultural irrigation using unmanned aerial vehicles and IoT-based cloud system, arXiv, (2021).

[5] A. Sharma, S. Shoval, A. Sharma, J.K.J.I.T.R. Pandey, Path planning for multiple targets interception by the swarm of UAVs based on swarm intelligence algorithms: A review, IETE Technical Review, 39 (2022) 675-697.

[6] S. Aggarwal, N.J.C.C. Kumar, Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges, Computer Communications, 149 (2020) 270-299.

[7] Y. Cai, Q. Xi, X. Xing, H. Gui, Q. Liu, Path planning for UAV tracking target based on improved A-star algorithm, 2019 1st International Conference on Industrial Artificial Intelligence (IAI), IEEE, 2019, pp. 1-6.

[8] F. Bounini, D. Gingras, H. Pollart, D. Gruyer, Modified artificial potential field method for online path planning applications, 2017 IEEE Intelligent Vehicles Symposium (IV), IEEE, 2017, pp. 180-185.

[9] M. Jain, V. Saihjpal, N. Singh, S.B.J.A.S. Singh, An Overview of Variants and Advancements of PSO Algorithm, Applied Sciences, 12 (2022) 8392.

[10] J. Kennedy, R. Eberhart, Particle swarm optimization, Proceedings of ICNN'95-international conference on neural networks, IEEE, 1995, pp. 1942-1948.

[11] F. Marini, B.J.C. Walczak, I.L. Systems, Particle swarm optimization (PSO). A tutorial, Chemometrics and Intelligent Laboratory Systems, 149 (2015) 153-165.

[12] M. Clerc, J.J.I.t.o.E.C. Kennedy, The particle swarm-explosion, stability, and convergence in a multidimensional complex space, IEEE transactions on Evolutionary Computation, 6 (2002) 58-73.

[13] X. Hu, Y. Liu, G.J.J.o.S.E. Wang, Electronics, Optimal search for moving targets with sensing capabilities using multiple UAVs, Journal of Systems Engineering and Electronics, 28 (2017) 526-535.

[14] J. ZHANG, Y. LIU, G.J.T. Wang, -.-. Microsystem Tech-nologies, UAV route planning based on PSO algorithm [J], Transducer and Microsystem Technologies, 4 (2017).

[15] H.S. Dewang, P.K. Mohanty, S.J.P.c.s. Kundu, A robust path planning for mobile robot using smart particle swarm optimization, Procedia computer science, 133 (2018) 290-297.

[16] R.K. Mandava, S. Bondada, P.R. Vundavilli, An optimized path planning for the mobile robot using potential field method and PSO algorithm, Soft Computing for Problem Solving, Springer2019, pp. 139-150.
[17] M.D. Phung, Q.P.J.A.S.C. Ha, Safety-enhanced UAV path planning with spherical vector-based particle swarm optimization, Applied Soft Computing, 107 (2021) 107376.
[18] D. Tian, Z.J.S. Shi, e. computation, MPSO: Modified particle swarm optimization and its applications, Swarm and evolutionary computation, 41 (2018) 49-68.