Three Dimensional Path Planning for UAV Based on Chaotic Gravitational Search Algorithm

Keming Jiao[1], Jie Chen[1], Bin Xin[2, 3,*], Li Li[1, 4], Yifan Zheng[1], Zhixin Zhao[1]

[1]School of Electronics and Information Engineering, Tongji University, Shanghai 201804, China
[2]School of Automation, Beijing Institute of Technology, Beijing 100081, China
[3]State Key Laboratory of Intelligent Control and Decision of Complex Systems, Beijing 100081, China
[4]Shanghai Institute of Intelligent Science and Technology, Tongji University, Shanghai 201804, China
E-mail: brucebin@bit.edu.cn

Abstract: Path planning is a critical problem for an unmanned aerial vehicle (UAV), which assists UAV to find a desirable path under some constraints. A chaotic gravitational search algorithm (CGSA) is proposed for UAV path planning in three dimensional (3D) environment. Firstly, the cost function considers the path length, turning angle, climbing angle, and maximum and minimum flight heights. Secondly, the chaotic sequence of the sine map is used to determine the size of the *Kbest* set storing the best agents in gravitational search algorithm (GSA), which improves the balance of exploration and exploitation. Finally, the simulation results demonstrate that CGSA is more effective and better than the moth flame optimization algorithm (MFO) and GSA in 3D path planning.

Keywords: Unmanned Aerial Vehicle, Path Planning, Gravitational Search Algorithm, Chaos

1. INTRODUCTION

With the rapid development of automation technology, the performance of the unmanned aerial vehicle (UAV) has been greatly improved, and UAV shows significant potential in civil and military applications in recent years, especially in dull, dirty, dangerous, and deep missions [1]. In the process of executing tasks, the autonomy level of UAV relies on its path planning and control method. UAV path planning is to search for the optimal or satisfactory path from a starting point to its destination according to some performance indicators, which is a key aspect of UAV intelligence [2] [3].

In the last several decades, many scholars have made indepth research on path planning, and numerous algorithms about path planning have been proposed, which can be divided into classical algorithms and heuristic algorithms [4].

The classical algorithms mainly include the cell decomposition, potential field, and sampling based method, etc. The cell decomposition method divides the search space into lots of small units called the cells, and provides a barrier free sequence from starting point to the goal. Wu and Xu *et al*. put forward the bidirectional adaptive A star algorithm, which replaces the multidirection with the directional search strategy in A star algorithm, and employs the adaptive step and weight strategy to increase the exploration speed [5]. Potential field method assigns the gravitation and repulsion to the goal and obstacle, respectively, pulling UAV toward the goal and keeping it away from the obstacle. To solve the local optimal problem, Li applied the chaos theory to change the coefficients of gravitation and repulsion [6]. Sampling based method detects the barriers by sampling, and then constructs an effective path. For solving the obstacle avoidance in dynamic environment, Chen and Mantagh proposed the fuzzy kinodynamic rapidly exploring random tree (RRT), which utilizes the RRT to generate the path and employs the fuzzy logic to avoid the obstacles [7].

The methods based on artificial neural network (ANN), fuzzy logic, and evolutionary computation methods belong to heuristic algorithms. ANN is an artificial brain model, having the ability of learning and reasoning. One of the challenges for online path planning is environmental uncertainty, for overcoming this difficulty, Sung and Choi *et al.* applied different path data sets to train a neural network as a path planner and make a decision online [8]. Unlike ANN, fuzzy logic mainly uses expert knowledge and experience to make a decision. Adhikari and Kim *et al.* turned the path planning problem into a multiobjective unconstrained optimization problem, and adopted fuzzy logic to optimize the parameters of differential evolution, minimizing the fuel, threat cost, and path length [9]. Evolutionary algorithms are inspired by biological behaviors, such as ant colony optimization (ACO), particle swarm optimization (PSO), and genetic algorithm (GA), which are often applied in path planning. For the problem of covering target points in minimum time, Li and Xiong *et al*. proposed an ACO variant with the greedy strategy to find the optimal number of UAVs and plan the paths with the minimum time [10].

Some simple three dimensional (3D) path planning problems can be transformed into 2D path, and there have been many researches on 2D path planning problems. In fact, some important information, such as the kinematic constraints, will be ignored in the transformation process. Taking into account the completeness of the problem assumptions, a 3D path planning method based on chaotic gravitational search

algorithm (CGSA) is proposed in this paper, and compared with the existing works, the path length, turning angle, and climbing angle, and maximum and minimum flight heights for UVA are considered.

The remainder of this paper is organized as follows: Section 2 introduces the related works, including the environment model, GSA, and chaotic sine map. Section 3 presents the proposed algorithm CGSA for path planning in detail. The simulations and results are shown in Section 4. Finally, the conclusions are elaborated in the last section.

2. RELATED WORKS

This section provides a basic description of the environment model, Gravitational Search Algorithm (GSA), and chaotic sine map.

2.1. Modeling the environment

It is assumed that the UAV plans to fly from start point S to terminal point T for a mission, and the spatial coordinate is established, just like in Fig. 1, in which the coordinates of S and T are (x_1, y_1, z_1) and (x_n, y_n, z_n), respectively. The range $[x_1, x_n]$ of the *x*-axis is divided into $n-1$ equal parts and the vertical planes $(\beta_1, \beta_2, \dots, \beta_n)$ of the *x*-axis are obtained according to the corresponding points. One discrete point is taken in each vertical plane β_i, and the collection of points $P = \{S, (x_2, y_2, z_2), (x_3, y_3, z_3), \dots, (x_{n-1}, y_{n-1}, z_{n-1}), T\}$ is got. The flight path can be acquired by connecting these points in turn. So the path planning problem is transformed into optimizing these coordinate sequences to minimize the objective function. Due to that the dimension of variables can affect the efficiency and complexity of evolutionary computation, the optimization variable P can be simplified as $P = (y_2, z_2, y_3, z_3, \dots, y_{n-1}, z_{n-1})$, where the number of variables is reduced by $n-2$.

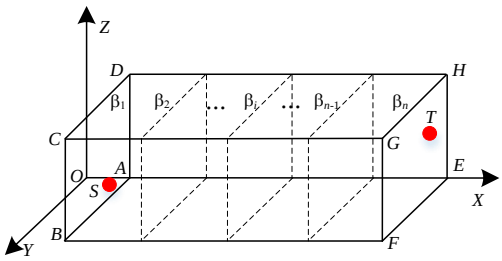

Fig. 1 The 3D environment model

2.2. Gravitational search algorithm

The GSA is a population-based optimization algorithm and inspired by the law of gravity and mass interaction, which is proposed by Rashedi *et al.* in 2009 [11] [12]. In GSA, every solution is regarded as an agent, whose performance is evaluated by its mass. The agent $X_i(t)$ can interact with other agents through mass and move at speed $V_i(t)$ in *n*-dimensional search space.

Supposing there are N agents, the position and velocity of the *i*th agent are defined as follows:

$$X_i(t) = \left(x_i^1(t), x_i^2(t), \dots, x_i^d(t), \dots, x_i^n(t)\right), i = 1, 2, \dots, N \quad (1)$$

$$V_i(t) = \left(v_i^1(t), v_i^2(t), \dots, v_i^d(t), \dots, v_i^n(t)\right), i = 1, 2, \dots, N \quad (2)$$

The mass $M_i(t)$ of the agent $X_i(t)$ is related to fitness, which is described as:

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \quad (3)$$

$$m_i(t) = \frac{fit_i(t) - fit_{worst}(t)}{fit_{best}(t) - fit_{worst}(t)} \quad (4)$$

where, for the iteration *t*, $fit_i(t)$ is the fitness value of the agent $X_i(t)$, $fit_{best}(t)$ and $fit_{worst}(t)$ are the best fitness value and worst fitness value in the current population, respectively.

According to the law of gravity, these agents attract other agents by gravitational force, causing the agents to move towards the agent with the larger mass. The gravitational force between the agent *i* and *j* is defined in the following equation:

$$F_{ij}^d(t) = G(t)\frac{M_i(t) \cdot M_j(t)}{R_{ij}(t) + \varepsilon}\left(x_j^d(t) - x_i^d(t)\right) \quad (5)$$

where $M_i(t)$ and $M_j(t)$ denote the mass of the agent *i* and *j*, respectively. $R_{ij}(t)$ is the distance between the agent *i* and *j*, ε denotes a small constant to prevent the denominator from being 0. $x_j^d(t)$ and $x_i^d(t)$ represent the position of the agent *i* and *j* in the *d*th dimension, respectively. $G(t)$ is the gravitational constant in the *t*th iteration, which can be calculated using (6)

$$G(t) = G_0 e^{-\alpha \frac{t}{T}} \quad (6)$$

where G_0 denotes the initial value for the gravitational constant, α represents the coefficient of decrease, and T stands for the maximum number of iteration.

To increase the randomness of the algorithm, the total force that acts on agent *i* in *d*th dimension can be calculated using (7)

$$F_i^d(t) = \sum_{j \in Kbest, j \neq i} rand_j F_{ij}^d(t) \quad (7)$$

where *Kbest* denotes the set that stores first *K* best agents, and *K* decreases linearly over iterations. $rand_j$ is a rand number, generated in the interval [0, 1].

The acceleration of agent *i* in the *d*th dimension can be obtained by (8) according to the law of motion.

$$a_i^d(t) = \frac{F_i^d(t)}{M_i(t)} \quad (8)$$

Finally, the velocity and position of the agent *i* can be updated using (9) and (10).

$$v_i^d(t+1) = rand_i \cdot v_i^d(t) + a_i^d(t) \quad (9)$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \quad (10)$$

where v_i^d and x_i^d stand for the velocity and position of the agent *i* in the *d*th dimension respectively.

2.3. Sine map

The sine map is one of the famous chaotic maps [13], which can be defined by (11).

$$x_{n+1} = F(r, x_n) = r \cdot \sin(\pi \cdot x_n) \quad (11)$$

where r is a control parameter greater than 0, and x_n denotes the output chaotic sequence. The bifurcation diagram of a Sine map with $r \in (0, 4]$ is shown in Fig. 2.

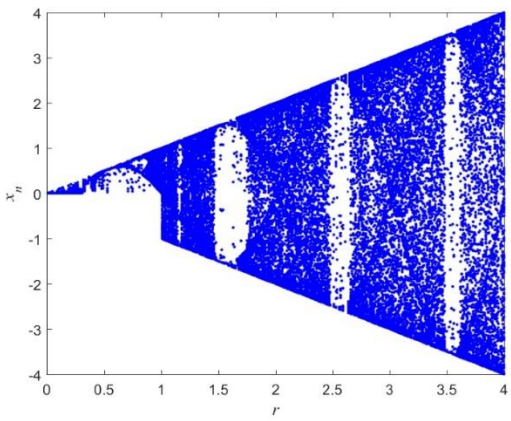

Fig. 2 Bifurcation diagram of the Sine map, $r \in (0,4]$

3. PROPOSED METHOD

In this section, the overall cost function is defined, then the proposed path planning algorithm is described.

3.1. Cost function

Considering the energy and time consumption, it is very known that the search of the optimal path is usually to find the path with the smallest length [14]. Therefore, the length of a path is regarded as the cost function, which is defined as follows.

$$F_1 = \sum_{i=1}^{n-1} F_1^i \tag{12}$$

$$F_1^i = \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2 + (z_i - z_{i+1})^2} \tag{13}$$

where F_1^i is the Euclidean distance between node *i* and the next node in the path.

The turning and climbing angles are two important constraints for a UAV, affecting the smoothness and feasibility of the path. Just as shown in Fig. 3, P_{i-1}, P_i, and P_{i+1} are three adjacent points in the path, whose projection points on the plane *Oxy* are P'_{i-1}, P'_i, and P'_{i+1}, respectively. P''_{i+1} is the projection of point P_i on line $P_{i+1}P'_{i+1}$. The turning angle γ_i is the angle between path segments $\overrightarrow{P'_{i-1}P'_i}$ and $\overrightarrow{P'_iP'_{i+1}}$, which can be calculated using (15). The turning cost function can be computed by (17).

$$\gamma_i = \arccos\left(\frac{\overrightarrow{P'_{i-1}P'_i}\cdot\overrightarrow{P'_iP'_{i+1}}}{\left\|\overrightarrow{P'_{i-1}P'_i}\right\|\cdot\left\|\overrightarrow{P'_iP'_{i+1}}\right\|}\right) \tag{14}$$

$$\gamma_i = \arccos\left(\frac{(x_i - x_{i-1}, y_i - y_{i-1})\cdot(x_{i+1} - x_i, y_{i+1} - y_i)^T}{\|x_i - x_{i-1}, y_i - y_{i-1}\|\cdot\|x_{i+1} - x_i, y_{i+1} - y_i\|}\right) \tag{15}$$

$$F_2^i = \begin{cases} 0, & if\ \gamma_i \le \gamma_{max} \\ k_t \cdot \gamma_i, & otherwise \end{cases} \tag{16}$$

$$F_2 = \sum_{i=2}^{n-1} F_2^i \tag{17}$$

where $\|\cdot\|$ denotes the L2 norm operation, and k_t and γ_{max} are the coefficient and threshold of turning angle, respectively.

The climbing angle θ_i is the angle between path segment $\overrightarrow{P_iP_{i+1}}$ and its projection on the horizontal plane, which is described by (21). The climbing cost function is given by (24).

$$\theta_i = arctan\left(\frac{|z_{i+1} - z_i|}{\left\|P'_iP''_{i+1}\right\|}\right) \tag{18}$$

$$\theta_i = arctan\left(\frac{|z_{i+1} - z_i|}{\|(x_i - x_{i+1}, y_i - y_{i+1})\|}\right) \tag{19}$$

$$F_3^i = \begin{cases} 0, & if\ \theta_i \le \theta_{max} \\ k_c \cdot \theta_i, & otherwise \end{cases} \tag{20}$$

$$F_3 = \sum_{i=1}^{n-1} F_3^i \tag{21}$$

where k_c and θ_{max} are the coefficient and threshold of climbing angle, respectively.

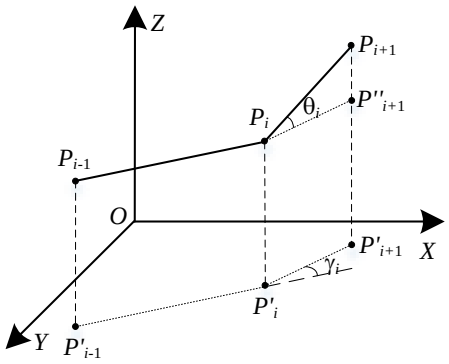

Fig. 3 UAV performance constraints

Considering geographic and UAV performance constraints, the feasible flight height of UAV is also critical for path planning. Let the minimum and maximum heights be h_{min} and h_{max}, respectively. The height cost function of the path is computed as:

$$F_4^i = \begin{cases} 0, & if\ h_{min} \le z_i \le h_{max} \\ k_h \cdot \max(z_i - h_{max},\ h_{min} - z_i), & otherwise \end{cases} \tag{22}$$

$$F_4 = \sum_{i=1}^{n} F_4^i \tag{23}$$

where k_h is the coefficients of flight height.

Taking into the length, environment, and UAV performance constraints for the path *P*, the overall cost function can be defined as:

$$F(P) = \sum_{t=1}^{4} F_t(P) \tag{24}$$

3.2. CGSA for UAV path planning

Owing to randomness, ergodicity, and sensitivity to the initial value, the chaotic system has the advantage of global optimization. As described in Section II.B, *K* is also the number of forces exerting on an agent, which has an important impact on exploration and exploitation. To better balance the exploration and exploitation, chaotic GSA is proposed, where *Kbest* function has the chaotic behavior of the sine map, just as shown in (26).

$$x(t) = r \cdot \sin(\pi \cdot x(t-1)) \tag{25}$$

$$K(t) = f + |x(t)| \cdot \left(\frac{T-t}{T}\right) \cdot (N - f) \tag{26}$$

where *f* stands for the percent of agents that apply force to others, *N* is the population size.

According to the above analysis, the flow chart of our proposed CGSA for path planning is exhibited in Fig. 4, and the implementation steps can be described as follows:

Step 1: Initialize algorithm parameters, the population size *N*, the gravitational constant G_0 and α, initialize the positions of all agents randomly and set the maximum number of iterations *T*;

Step 2: Evaluate the overall cost function or fitness of each agent in the population using (24);

Step 3: Calculate the mass of each agent by (3);

Step 4: Update the gravitational constant G and K_{best} using (6) and (26), respectively and compute the total force of each agent according to (7);
Step 5: Compute the acceleration of each agent by (8), and update the velocity and position of agents as per (9) and (10), respectively;
Step 6: Check whether the iteration termination condition is met. If yes, go to **Step 7**, else set $i = i + 1$ and return to **Step 2**;
Step 7: Stop algorithm iteration, and output the best solution and fitness.

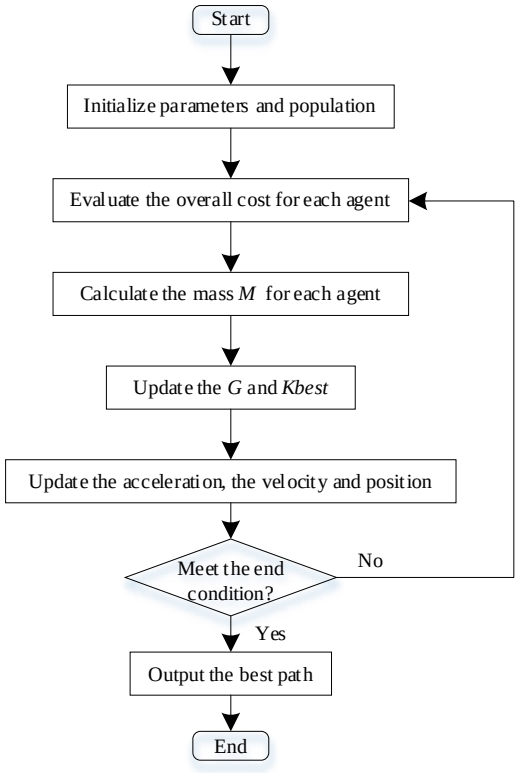

Fig. 4 The flow chart of CGSA algorithm

4. SIMULATION EXPERIMENTS

4.1. Parameter setting

To verify the effectiveness of the proposed algorithm, the proposed algorithm CGSA is compared with moth flame optimization algorithm (MFO) [15] [16] and GSA [11] in two simulation environments. To be fair, all simulation experiments are run on the computer with an Intel Core i7 2.6GHz and Windows 10; the population size N is set as 50, the maximum iteration number T is 3000 for three algorithms; G_0 and α are set to 100 and 20 for CGSA and GSA, respectively, which are the same as that in [11]. The constant b is 1 in MFO, just as in [16]. The coefficients k_t, k_c, and k_h are set to 20, 20, and 30. The thresholds γ_{max}, θ_{max}, and h_{max} of turning angle, climbing angle, and maximum flight height are are equal to 60, 45, and 200, and the minimum flight height h_{min} is 5m higher than the ground level. The value of r for chaos coefficient is set to 1. The three algorithms are excuted 30 times in both environments, and the path results in the two environments are analyzed by randomly selecting one time from all the results, the statistical results of the three algorithms are put at the end.

4.2. Simple environment

Assuming that there is only one obstacle in this environment, which is called a simple environment. The starting point and terminal point are set to (1, 5, 10) and (201, 190, 70), respectively. Fig. 5 shows the paths optimized by three methods, and the corresponding convergence curves are drawn in Fig. 6. From Fig. 5 and Fig. 6, it is obvious that the path obtained by CGSA is better than that got by MFO and GSA. In terms of the convergence curves, the CGSA algorithm converges faster than the other algorithms and gets the lowest overall cost value. So in this simple environment, CGSA performs better than MFO and GSA. In fact, we also find that F_2, F_3, and F_4 in the overall cost function are all 0 for the last obtained paths, which shows that the three methods can finally find the path satisfying the constraints and the length of the path is the main factor affecting their optimality.

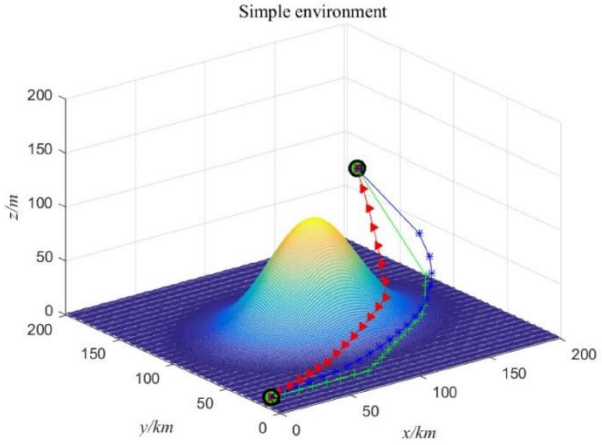

Fig. 5 Path planning results of three algorithms in simple environment; the green, blue, and red path are obtained by MFO, GSA, and CGSA, respectively.

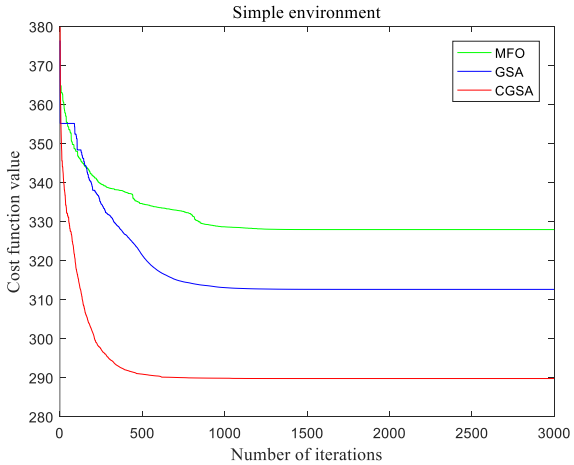

Fig. 6 Convergence curves of three algorithms in simple environment

4.3. Complex environment

Supposing that there are multiple obstacles in the environment, such as mountains and buildings, which is viewed as a complex environment. The settings of

starting point and terminal point are the same as that in simple environment. The paths and convergence curves obtained are potted in Fig. 7 and Fig. 8, respectively. The path got by CGSA is shorter and smoother than that obtained by MFO and GSA. At the same time, the convergence speed of CGSA is the fastest and its cost value is the smallest. Therefore, the performance of CGSA is the best in this environment. Just like that in simple environment, the path length becomes the major contributor affecting the optimality for the last obtained path.

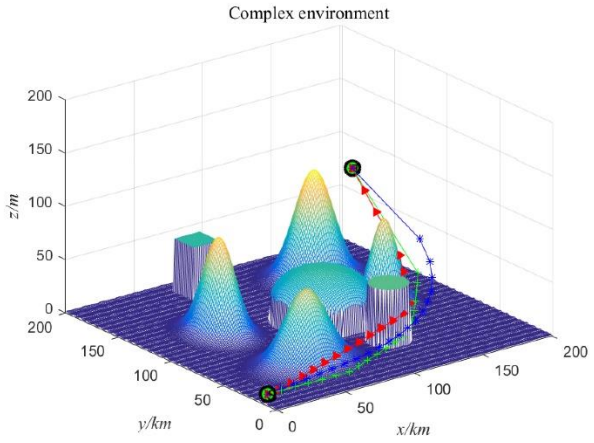

Fig. 7 Path planning results of three algorithms in complex environment; the green, blue, and red path are obtained by MFO, GSA, and CGSA, respectively.

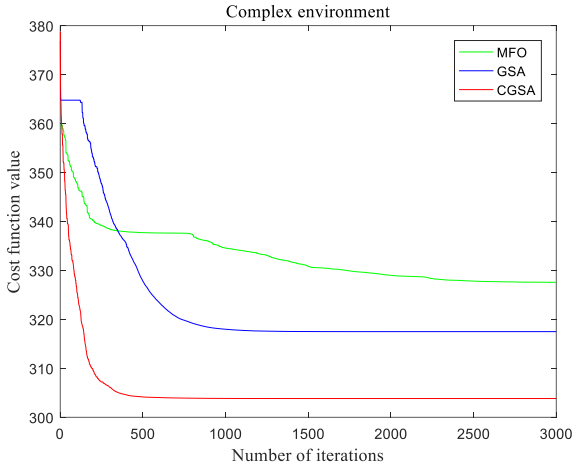

Fig. 8 Convergence curves of three algorithms in complex environment

After the three algorithms are run 30 times in two environments, the mean and standard deviation (Std) of overall cost values are shown in Table 1. It is obvious that the mean and Std of CGSA are much smaller than the results of MFO and GSA. Therefore, the proposed CGSA has better effectiveness and robustness than MFO and GSA for path planning in these two environments.

Table 1 Comparison of simulation results of three algorithms

Environment	MFO		GSA		CGSA	
	Mean	Std	Mean	Std	Mean	Std
Simple environment	316.9510	13.1586	313.6128	1.8159	287.8801	1.3560
Complex environment	325.2160	10.4573	314.7781	1.8197	304.2916	1.1928

5. CONCLUSIONS

This paper presents a CGSA algorithm for UAV path planning in 3D environment. In CGSA, the sine map is used to determine the number of agents acting on each other in GSA, which can better balance exploration and exploitation. Then, the UAV can obtain the optimal path by connecting the selected nodes while considering the path length, turning angle, climbing angle, and flight height. The simulation results clearly show that the CGSA has faster convergence and stronger stability in UAV path planning than MFO and GSA algorithm. Our future works will focus on the path planning of multiple UAVs considering the security constraints of UAVs, and use the benchmark to test CGSA.

Acknowledgment

This work was supported in part by the National Outstanding Youth Talents Support Program 61822304, in part by the National Key R&D Program of China (2018YFB1308000, 2018YFE0105000, 2018YFB1305304), in part by the Basic Science Center Programs of NSFC under Grant 62088101, in part by the Projects of Major International (Regional) Joint Research Program of NSFC under Grant 61720106011, in part by Consulting Research Project of the Chinese Academy of Engineering (2019-XZ-7), in part by Shanghai Municipal Science and Technology Major Project under grant 2021SHZDZX0100, in part by the Shanghai Municipal Commission of Science and Technology (1951113210, 19511132101), in part by Beijing Advanced Innovation Center for Intelligent Robots and Systems.

REFERENCES:

[1] D. Wang, H. Lv, and J. Wu, “In-flight initial alignment for small UAV mems-based navigation via adaptive unscented kalman filtering approach,” Aerospace Science and Technology, vol. 61, pp. 73–84, 2017.

[2] D. Zhang, Y. Xu, and X. Yao, “An improved path planning algorithm for unmanned aerial vehicle based on RRT-connect,” in *2018 37th Chinese Control Conference (CCC)*. IEEE, 2018, pp. 4854–4858.

[3] C. Xiong, B. Xin, M. Guo, Y. Ding, and H. Zhang, “Multi-UAV 3D path planning in simultaneous attack,” in *2020 IEEE 16th*

International Conference on Control & Automation (ICCA). IEEE, 2020, pp. 500–505.

[4] T. T. Mac, C. Copot, D. T. Tran, and R. De Keyser, "Heuristic approaches in robot path planning: A survey," *Robotics and Autonomous Systems*, vol. 86, pp. 13–28, 2016.

[5] X. Wu, L. Xu, R. Zhen, and X. Wu, "Bi-directional adaptive A* algorithm toward optimal path planning for large-scale uav under multi-constraints," *IEEE Access*, vol. 8, pp. 85431–85440, 2020.

[6] W. Li, "An improved artificial potential field method based on chaos theory for UAV route planning," in *2019 34rd Youth Academic Annual Conference of Chinese Association of Automation (YAC)*. IEEE, 2019, pp. 47–51.

[7] L. Chen, I. Mantegh, T. He, and W. Xie, "Fuzzy kinodynamic RRT: a dynamic path planning and obstacle avoidance method," in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2020, pp. 188–195.

[8] I. Sung, B. Choi, and P. Nielsen, "On the training of a neural network for online path planning with offline path planning algorithms," *International Journal of Information Management*, vol. 57, pp. 102142, 2021.

[9] D. Adhikari, E. Kim, and H. Reza, "A fuzzy adaptive differential evolution for multi-objective 3D UAV path optimization," in *2017 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2017, pp. 2258–2265.

[10] J. Li, Y. Xiong, and J. She, "An improved ant colony optimization for path planning with multiple UAVs," in *2021 IEEE International Conference on Mechatronics (ICM)*. IEEE, 2021, pp. 1–5.

[11] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "GSA: a gravitational search algorithm," *Information sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.

[12] K. Jiao and Z. Pan, "A novel method for image segmentation based on simplified pulse coupled neural network and gbest led gravitational search algorithm," *IEEE Access*, vol. 7, pp. 21310–21330, 2019.

[13] A. Belazi and A. A. Abd El-Latif, "A simple yet efficient s-box method based on chaotic sine map," *Optik*, vol. 130, pp. 1438–1444, 2017.

[14] H. Zhang, B. Xin, L. Dou, J. Chen, and K. Hirota, "A review of cooperative path planning of an unmanned aerial vehicle group," *Frontiers of Information Technology & Electronic Engineering*, vol. 21, no. 12, pp. 1671–1694, 2020.

[15] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *Knowledge-based Systems*, vol. 89, pp. 228–249, 2015.

[16] M. Shehab, L. Abualigah, H. Al Hamad, H. Alabool, M. Alshinwan, and A. M. Khasawneh, "Moth–flame optimization algorithm: variants and applications," *Neural Computing and Applications*, vol. 32, no. 14, pp. 9859–9884, 2020.