electronics

MDPI

Article

A UAV Path Planning Method in Three-Dimensional Space Based on a Hybrid Gray Wolf Optimization Algorithm

Jianxin Feng *, Chuanlin Sun, Jianhao Zhang, Yue Du, Zhiguo Liu and Yuanming Ding

Communication and Network Key Laboratory, Dalian University, Dalian 116622, China; sunchualin24@gmail.com (C.S.); zjh20231202@163.com (J.Z.); duyue0217@gmail.com (Y.D.); liuzhiguo_dldx@163.com (Z.L.); dingyuanming@dlu.edu.cn (Y.D.)
* Correspondence: fengjianxin863@163.com

Abstract: Path planning, which is needed to obtain collision-free optimal paths in complex environments, is one key step within unmanned aerial vehicle (UAV) systems with various applications, such as agricultural production, target tracking, and environmental monitoring. A new hybrid gray wolf optimization algorithm—SSGWO—is proposed to plan paths for UAVs under three-dimensional agricultural environments in this paper. A nonlinear convergence factor based on trigonometric functions is used to balance local search and global search. A new relative-distance fitness adaptation strategy is created to increase the convergence speed of the SSGWO. Integrating the simulated annealing (SA) algorithm, an alternative position update strategy based on SA is proposed to improve the search process with diverse capabilities. Finally, a B-spline curve is introduced into a smooth path to ensure the path's feasibility. The simulation results show that the SSGWO algorithm has better convergence accuracy and stability, and can obtain higher-quality paths in a three-dimensional environment, compared with GWO, MGWO, IGWO, and SOGWO.

Keywords: unmanned aerial vehicles; path planning; gray wolf optimizer; simulated annealing; B-spline curve

check for updates

Citation: Feng, J.; Sun, C.; Zhang, J.; Du, Y.; Liu, Z.; Ding, Y. A UAV Path Planning Method in Three-Dimensional Space Based on a Hybrid Gray Wolf Optimization Algorithm. *Electronics* **2024**, *13*, 68. https://doi.org/10.3390/electronics13010068

Received: 20 October 2023
Revised: 15 December 2023
Accepted: 18 December 2023
Published: 22 December 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Unmanned aerial vehicles (UAVs) can execute flight tasks autonomously without direct human control [1]. In applications such as disaster reconnaissance, target tracking, communications relays, aerial photography, agriculture, environmental monitoring, and mine exploration, UAVs play indispensable roles. The degree of agriculture informatization seriously affects the development of human beings. Unmanned aerial vehicles (UAVs) are becoming increasingly important in agricultural practices with numerous innovations and improvements [2]. Equipped with high-resolution cameras and multi-spectral sensors, UAVs can monitor farmland in a quick and accurate way. With GPS technology and intelligent control systems, UAVs have great significance in achieving the goals of precision agriculture, such as sowing, fertilizing, and fixed-point spraying.

UAV path planning in agricultural production has become a concentrated research area. However, in the complex environments of agricultural areas, UAVs will encounter diverse threats and obstacles when navigating, which bring risks to flight missions. Therefore, the implementation of effective path planning becomes crucial. This not only enhances the obstacle-avoidance capabilities of UAVs, but also serves to minimize flight costs to meet the practical requirements of agricultural tasks, considering multiple factors such as flight efficiency, obstacle avoidance, safety, and traffic laws [3]. The problem of path planning is inherently intricate, so it needs to be solved with a combination of mathematical modeling and computer algorithms [4]. With increasing mission complexity in dynamic, changing environments, UAV path planning must employ real-time, high-performance, and diversified approaches to ensure the safe and efficient completion of UAV flight

missions [5]. Several algorithms have been proposed, including a graph search approach, the A* algorithm [6], particle swarm optimization (PSO) [7], a genetic algorithm (GA) [8], a heuristic algorithm, and a reinforcement learning model [9]. However, these algorithms still have some limitations in path planning. For instance, graph search methods may encounter efficiency challenges with large search spaces or multiple goals, given their need to traverse an extensive array of potential paths. The A* algorithm is affected by the search space. When the search space is very large, the computational complexity of the algorithm will increase sharply, resulting in a long operation time [10]. In actual flight, the diverse constraints and performance characteristics of UAVs will result in substantial cost disparities among different path segments. This intricacy restricts the applicability of dynamic programming to actual path planning, leading to inconsistencies between the planning results and the actual situation [11]. At the same time, these algorithms may fall in local optima, so that the planning results are not the globally optimal paths.

Heuristic algorithms are types of computing methods used to solve complex optimization problems. Without relying on domain-specific knowledge, heuristic algorithms operate based on general heuristic principles to search the solution space for a given problem. Metaheuristic algorithms are usually more general and work similarly to heuristic behaviors in nature: for example, simulated annealing, genetic evolution, ant colony searching, etc. Typically organized as a cohort of individuals, these algorithms iteratively refine themselves to discover optimal or near-optimal solutions. Yu et al. [12] proposed an innovative, adaptive, and selective variation-constrained differential evolution algorithm for UAV path planning for disasters. Combining adaptive and evolutionary strategies, this algorithm enhances adaptability to complex environments and multiple constraints. Phung et al. [13] proposed a spherical vector-based SPSO algorithm tailored to path planning in complex environments. This algorithm contributed to subsequent path planning research, incorporates the unique characteristics of spherical vectors, and aims to better adapt to multi-constraint and multi-variable problems. Huang et al. [14] introduced an improved quantum particle swarm optimization (QPSO) algorithm to accelerate and improve the efficiency and accuracy of UAV path planning. This algorithm incorporates the idea of quantum computing and searches for the best path by simulating the behavior of particles so as to find the best path more quickly. Chen et al. [15] proposed a flower pollination algorithm based on neighborhood global learning. This provided a new idea for improving path planning, but it was not verified in a three-dimensional environment. Mirjalili et al. [16] proposed a gray wolf optimization (GWO) algorithm inspired by the gray wolf group hunting mechanism. Compared with some other optimization algorithms, GWO has fewer parameters that need to be adjusted. The algorithm is relatively simple and easy to understand and implement. Although the GWO algorithm performs well in some situations, its adaptability is limited in complex environments and it can easily fall into local optimal solutions. These seriously affect the quality of path planning. In order to enhance GWO for tackling UAV path planning challenges effectively, researchers usually introduce simple adaptive mechanisms or create new combination algorithms to enhance the optimization capabilities of the algorithm. Up until now, the stability of algorithms based on GWO providing quick and accurate methods in complex environments has not been achieved yet. Considering complex agricultural environments, SSGWO—an improved algorithm based on the GWO algorithm—is proposed in this paper to solve the problem of UAV path planning in a three-dimensional agricultural environment. The main contributions of this paper are as follows:

1. A new hybrid algorithm, SSGWO, is used to solve the problem of UAV path planning in a three-dimensional agricultural environment.
2. An improved nonlinear convergence factor based on trigonometric functions is presented to balance global search capability and local search capabilities with better accuracy.
3. A relative-distance fitness adaptation strategy is proposed to improve the solution gradually during the search process and accelerate convergence to the global optimization solution with adaptation capability.

4. An alternative position-update strategy based on SA is proposed to improve the search process with diversity capability.
5. Multiple benchmark functions were used to verify the performance of the SSGWO algorithm, and the convergence of the algorithm was analyzed.
6. The SSGWO algorithm is used to solve the UAV path planning problem in agricultural environments. After smoothing the path with a B-spline curve, our UAV path planning method with SSGWO achieved better experimental results compared with other algorithms.

This paper is organized as follows. Section 2 presents a GWO-related review and the traditional GWO algorithm. Section 3 introduces the SSGWO algorithm in detail. Section 4 introduces flight constraints and path smoothing. Section 5 verifies the effectiveness of the SSGWO algorithm and the UAV path planning method with SSGWO. Section 6 discusses the experimental results. Section 7 summarizes the conclusions of this paper.

2. Related Work

GWO is a heuristic algorithm proposed based on the hunting behavior of gray wolf groups in nature, which is used to solve complex optimization problems. It discovers the optimal solution by simulating the social interaction of gray wolves. It has fewer parameters and stronger global search ability than other swarm intelligence algorithms [17]. GWO can intelligently adjust the behavior of wolves to adapt to the needs of diverse problems and environments through self-adjusting search strategies. This flexibility makes it perform well in solving diverse optimization problems and improves the convergence speed of the algorithm [18]. GWO has attracted widespread attention from scholars and has been used in many fields, including job shop scheduling [19], parameter extraction [20], feature selection [21], disease classification prediction [22], engineering design [23], path planning [24], etc. While theoretical analysis and the industrial applications of GWO have achieved fruitful results, some shortcomings still exist, including the low accuracy and slow convergence speed of unimodal functions and the local optima of multimodal functions, which hinder the further development of GWO. In practical applications, the optimization objectives of UAV path planning problems are complex and diverse. In recent years, researchers have explored various variants of GWO. When complex environmental factors and constraints are involved, the GWO algorithm easily falls into the local optimal solution resulting in low search accuracy [25]. Yang Zhang [26] proposed MGWO, which introduced an exponential regular convergence factor strategy, an adaptive update strategy, and a dynamic weighting strategy to improve GWO's search capabilities. Experimental results prove that its convergence speed and the algorithm's search ability are improved effectively. Nadimi-Shahraki et al. [27] proposed IGWO, an algorithm based on the dimensional learning hunting (DLH) search strategy, to enhance the information exchange between wolves and wolves in their neighborhoods. It can balance their local and global searches and preserve wolf diversity. Dhargupta et al. [28] applied Spearman's coefficient to determine opposition learning for operations on wolves. A GWO algorithm based on reverse learning was proposed, named SOGWO, and its performance was greatly improved. Research in the literature [29] combined the communication mechanism and horizontal comparison strategies to propose a constrained optimization algorithm based on the gray wolf approach and applied it to the pathfinding and collision avoidance of UAVs in a three-dimensional environment. The authors of [17] divided the search space into multiple dimensionality-reduction subspaces and proposed a parallel co-evolutionary GWO algorithm to overcome the local optima problem caused by the increase in dimensions in the search space.

The GWO algorithm is a swarm intelligence (SI) algorithm and has received extensive attention in recent years [16]. The structure of the traditional GWO is shown in Figure 1. The traditional GWO divides wolves into four ranks based on their social status and responsibilities: α, β, δ, and ω wolf. The α wolf is first-rank, and primarily makes decisions in the hunt. The α wolf dominates all other wolves and has the best information about the location of its prey. The β wolf is second-rank, and helps the α wolf to make decisions.

The β wolf follows the α wolf and can command other lower-rank wolves. The δ wolf is third-rank. The rest are ω wolves, belonging to the fourth rank, which is the lowest rank. The predation process has two steps. The first step is to surround the prey. The second step is to attack the prey. The model for surrounding the prey is shown by Formulas (1) and (2):

$$\vec{D} = \left| \vec{C} \times \vec{X}_{\mathrm{p}}(t) - \vec{X}(t) \right| \tag{1}$$

$$\vec{X}(t+1) = \vec{X}_{\mathrm{p}}(t) - \vec{A} \times \vec{D} \tag{2}$$

where $\vec{D}$ is the distance between the gray wolf and the prey, $\vec{X}_{\mathrm{p}}(t)$ is the current position vector of the prey, $\vec{X}(t)$ is the current position vector of the gray wolf, and $\vec{X}(t+1)$ is the updated position of the current wolf. $\vec{A}$ and $\vec{C}$ are coefficient vectors indicating disturbances, and can be obtained as follows:

$$\vec{A} = 2\vec{a} \times \vec{r_1} - \vec{a} \tag{3}$$

$$\vec{C} = 2 \times \vec{r_2} \tag{4}$$

$$\vec{a} = 2 - \frac{2t}{I_{\max}} \tag{5}$$

where $\vec{r_1}$ and $\vec{r_2}$ are random numbers ranging from 0 to 1, t is the current iteration number, $I_{\max}$ is the max iterations, and $\vec{a}$ is a convergence factor decreasing linearly from 2 to 0 during the iteration process. When $\vec{a}$ decreases from 2 to 0 over several iterations, the gray wolf completes the hunt by attacking the prey and stopping the action. $\vec{a}$ controls the balance between mining and exploration. The size of the $\vec{A}$ value indirectly affects the optimization performance of the GWO algorithm. $\vec{C}$ contributes to random behavior in the optimization process and helps the algorithm to explore and avoid local optima.

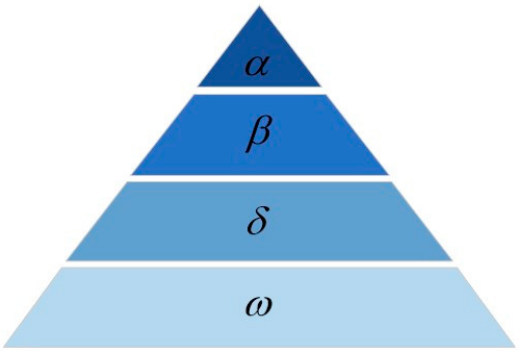


Figure 1. Grey wolf hierarchies.

The GWO algorithm solves the target problem by simulating the hunting behavior of wolves. The α, β, and δ wolves have more information about the prey and dominate the hunting behavior of different ranks. Other search wolves update their positions relative to them. The model of the attack process is as follows:

$$\vec{D}_{\alpha} = \left| \vec{C_1} \times \vec{X}_{\alpha} - \vec{X}(t) \right| \tag{6}$$

$$\vec{D}_{\beta} = \left| \vec{C_2} \times \vec{X}_{\beta} - \vec{X}(t) \right| \tag{7}$$

$$\vec{D}_\delta = \left| \vec{C}_3 \times \vec{X}_\delta - \vec{X}(t) \right| \tag{8}$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 \times \vec{D}_\alpha \tag{9}$$

$$\vec{X}_2 = \vec{X}_\beta - \vec{A}_2 \times \vec{D}_\beta \tag{10}$$

$$\vec{X}_3 = \vec{X}_\delta - \vec{A}_3 \times \vec{D}_\delta \tag{11}$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{12}$$

where $\vec{D}_\alpha$, $\vec{D}_\beta$, and $\vec{D}_\delta$ are the distances from wolf ω to the α, β, and δ wolves respectively. $\vec{X}_\alpha$, $\vec{X}_\beta$, and $\vec{X}_\delta$ are the positions of the α, β, and δ wolves respectively. $\vec{C}_1$, $\vec{C}_2$, $\vec{C}_3$, $\vec{A}_1$, $\vec{A}_2$, and $\vec{A}_3$ are coefficient vectors. $\vec{X}_1$, $\vec{X}_2$, and $\vec{X}_3$ are the final positions of the α, β, and δ wolves, and $\vec{X}(t+1)$ is the final position of the prey.

3. SSGWO Algorithm

3.1. Nonlinear Convergence Factor

Coordinating global search capability and local search capabilities helps GWO to search for optimal solutions. The role of the convergence factor $\vec{a}$ is to balance the global search capability and local search capabilities of the algorithm [26]. During the iteration process, the convergence factor $\vec{a}$ in the traditional GWO algorithm decreases linearly from 2 to 0 with the number of iterations. However, in a practical search process, the initial stage is required to converge slowly to expand the search space, and the later stage is required to accelerate convergence to improve algorithm efficiency dynamically.

Therefore, $\vec{a}$, a new nonlinear convergence factor based on trigonometric functions, is proposed, and its expression is shown in Formula (13):

$$\vec{a} = \begin{cases} \sin\left(\frac{\pi * \left(t + \frac{t_{\max}}{2}\right)}{t_{\max}} + \frac{\pi}{2}\right) + 2 & 0 \le t < \frac{t_{\max}}{2} \\ \sin\left(\frac{\pi * \left(t - \frac{t_{\max}}{2}\right)}{t_{\max}} + \frac{\pi}{2}\right) & \frac{t_{\max}}{2} \le t \le t_{\max} \end{cases} \tag{13}$$

where t is the current iteration number and $t_{\max}$ is the max iterations. Figure 2 shows the iteration image for the nonlinear convergence factor.

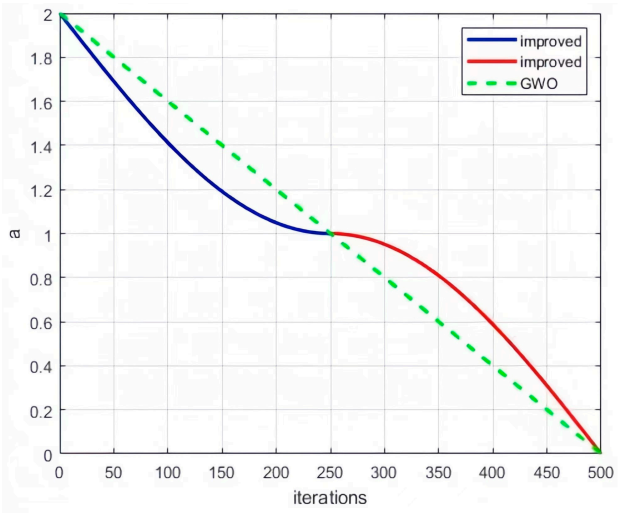


Figure 2. Convergence factor $\vec{a}$ curve.

As shown in Figure 2, the original convergence factor $\vec{a}$ decreases linearly during the iteration process. The improved convergence factor $\vec{a}$ changes slightly in the early stage of iteration and remains high for a long time. Thus, $\vec{A}$ with a high value for a long time improves global search capability. In the later stage of iteration, the status of $\vec{a}$ and $\vec{A}$ is exactly the opposite, and local search capabilities are improved. The balance between global search and local search can be obtained with better accuracy.

3.2. Fitness Value

In the process of solving the path planning problem with the GWO algorithm, evaluating the gray wolves individually to select α, β, δ, and ω wolves according to fitness value is necessary [16]. α, β, and δ are the three best wolves. The α wolf uses a search with larger strides in order to explore a wider area of the problem space. The β and δ wolves use a moderate search strategy to optimize the solution's quality further. The other ω wolves, led by the three best wolves, take smaller steps to avoid premature convergence to the local optimal solution.

In the iterative process, the fitness value proposed in this paper is proportional to adaptability. The closer the relative distance, the greater the fitness value, and the better the adaptability. This fitness adaptation strategy is shown as follows:

$$d_i^{\alpha,\beta,\delta} = \vec{X}_i(t) - \vec{X}_{\alpha,\beta,\delta} \tag{14}$$

$$d_{ave}^{\alpha,\beta,\delta} = \frac{\sum_{i=1}^{n} d_i^{\alpha,\beta,\delta}}{n} \tag{15}$$

$$f\left(\vec{X}_{\alpha,\beta,\delta}\right) = \frac{d_{ave}^{\alpha,\beta,\delta}}{d_i^{\alpha,\beta,\delta}} \tag{16}$$

where $\vec{X}_i(t)$ is the coordinates of the current wolf, $\vec{X}_{\alpha,\beta,\delta}$ represents the coordinates of the α, β, and δ wolves, respectively, n is the number of wolves, and $f\left(\vec{X}_{\alpha,\beta,\delta}\right)$ represents the fitness values of the α, β, and δ wolves, respectively. $d_i^{\alpha,\beta,\delta}$ is the distance between the current wolf and the α, β, and δ wolves, respectively. $d_{ave}^{\alpha,\beta,\delta}$ is the average distance between all gray wolf individuals and the α, β, and δ wolves, respectively. The ratio of $d_{ave}^{\alpha,\beta,\delta}$ to $d_i^{\alpha,\beta,\delta}$ is used to represent the distance of the current relative position of the individual, that is, the fitness value. With stride adjustments in the search process, the GWO algorithm can maintain a certain degree of adaptation capability and accelerate convergence to the global optimal solution.

3.3. The Alternative Position Update Strategy Based on SA

In the search process, the α wolf is the leader of all other wolves. The β wolf is the leader of the δ and ω wolves [27]. The δ wolf is the leader of the ω wolves. The position update is as follows:

$$\vec{X}_1 = \vec{X}_\alpha - \vec{a} \times b \times (\vec{X}_{r_1} - \vec{X}_{r_2}) \tag{17}$$

$$\vec{X}_2 = \vec{X}_\beta - \vec{a} \times b \times (\vec{X}_\alpha - \vec{X}_\beta) \tag{18}$$

$$\vec{X}_3 = \vec{X}_\delta - \vec{a} \times b \times (\vec{X}_\alpha - \vec{X}_\delta) \tag{19}$$

where $\vec{X}_1$, $\vec{X}_2$, and $\vec{X}_3$ are the final positions of the α, β, and δ wolves; $\vec{X}_\alpha$, $\vec{X}_\beta$, and $\vec{X}_\delta$ are the current positions of the α, β, and δ wolves; $\vec{X}_{r1}$ and $\vec{X}_{r2}$ are the current positions of the ω wolf individuals, respectively; $\vec{a}$ is the nonlinear coefficient factor, which decreases from

2 to 0; r_1 and r_2 are different integers in the range of 1 to the number of ω wolves; and b is a random number in [0, 2].

The higher-ranked wolves, which have more information, have greater weights. The weights for the α, β, and δ wolves, w_α, w_β, and w_δ, are calculated based on the fitness value as follows:

$$w_a = \frac{f\left(\vec{X}_\alpha\right)}{\left(f\left(\vec{X}_\alpha\right) + f\left(\vec{X}_\beta\right) + f\left(\vec{X}_\delta\right)\right)} \tag{20}$$

$$w_\beta = \frac{f\left(\vec{X}_\beta\right)}{\left(f\left(\vec{X}_\alpha\right) + f\left(\vec{X}_\beta\right) + f\left(\vec{X}_\delta\right)\right)} \tag{21}$$

$$w_\delta = \frac{f\left(\vec{X}_\delta\right)}{\left(f\left(\vec{X}_\alpha\right) + f\left(\vec{X}_\beta\right) + f\left(\vec{X}_\delta\right)\right)} \tag{22}$$

$$\vec{X'}(t+1) = w_\alpha \times \vec{X}_1 + w_\beta \times \vec{X}_2 + w_\delta \times \vec{X}_3 \tag{23}$$

where $\vec{X}_\alpha$, $\vec{X}_\beta$, and $\vec{X}_\delta$ are the positions of the α, β, and δ wolves. $f\left(\vec{X}_\alpha\right)$, $f\left(\vec{X}_\beta\right)$, and $f\left(\vec{X}_\delta\right)$ are their fitness values, respectively. $\vec{X}_1$, $\vec{X}_2$, and $\vec{X}_3$ are the final positions of the α, β, and δ wolves. $\vec{X'}(t+1)$ is the final position of the prey.

The SA algorithm, inspired by physical phenomena during metal quenching, is a global optimization algorithm used to find a solution close to the global optimal solution in the search space [30]. In order to prevent the GWO algorithm from easily falling into local optimal solutions, the SA algorithm is introduced to get the probability P of being promoted to an α wolf with fitness. The probability P enhances the search space of the α wolf and creates an alternative position update strategy representing the diversity capability. P is obtained as follows:

$$P = e^{\frac{-(f(\vec{X}_\alpha)_{t+1} - f(\vec{X}_\alpha)_t)}{T_t}} \tag{24}$$

$$T_{t+1} = \eta \times T_t \tag{25}$$

where $f\left(\vec{X}_\alpha\right)_t$ represents the fitness value of the α wolf in the current iteration and $f\left(\vec{X}_\alpha\right)_{t+1}$ is the fitness value of the α wolf after cooling in the current iteration, namely the fitness value of the α wolf in the next iteration. T_t represents the temperature of the current iteration and T_{t+1} is the temperature after cooling, namely the temperature of the α wolf in the next iteration. η is the cooling coefficient. The global search ability is improved with this strategy.

3.4. Pseudo Code of SSGWO

SSGWO can jump out of the local optimal solution, improving the ability to search for the global optimal solution, balance global and local search capabilities, and accelerate the convergence speed effectively. Algorithm 1 is the pseudo code of the SSGWO algorithm.

Algorithm 1 Pseudo code of SSGWO

Initialize the wolf population $\vec{X}_i (i = 1, 2, \ldots, N)$, $\vec{A}$, and $\vec{C}$
Calculate the fitness of all search agents
$\vec{X}_\alpha$ = the first-rank search agent
$\vec{X}_\beta$ = the second-rank search agent
$\vec{X}_\delta$ = the third-rank search agent
While (t < Max number of iterations) do
For each search agent
While $(P > rand(0,1))$ do
Calculate the weights of the three best wolves
Get P using Equation (24)
Update the position of the agent by Equation (23)
End while
End for
Update $\vec{A}$, $\vec{C}$ and $\vec{a}$ according to Equations (3), (4) and (13)
Calculate the fitness of all search agents
Update $\vec{X}_\alpha$, $\vec{X}_\beta$, and $\vec{X}_\delta$
$t = t + 1$
End while

The time complexity and space complexity of SSGWO primarily depend on the iteration number and the operations performed in each search agent. The time complexity of initializing the population is O (N) and the time complexity of computing fitness is O (N). The time complexity of each iteration depends on the operations in each search agent. The total iteration time complexity is O (T × N). The space complexity mainly depends on the array storing the search agent positions; each search agent has a location, so its space complexity is O (N). T is the number of iterations. N is the number of search agents.

4. UAV Flight Constraints and Path Smoothing

4.1. Constraint Modeling

In the practical UAV flight process, some performance constraints should be considered to establish a more efficient path. In the optimization process, flight distance, flight angle, and obstacles are considered with cost functions as follows.

1. Path distance cost $f_1(d)$

In the flight mission of a UAV, the limitation of energy and battery capacity is a key factor, which directly affects the flight time and flight range of the UAV [13]. Therefore, the flight distance of the UAV is limited, and the shortest path that satisfies the constraints should be selected. As shown in Figure 3, if there are M intermediate nodes in a path, the path is divided into $M + 1$ segments. The current node coordinate is (x_i, y_i, z_i). The path distance cost function is shown in Equation (26):

$$f_1(d) = \sum_{i=1}^{M+1} d_i \times \mu_d \tag{26}$$

$$d_i = \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2 + (z_i - z_{i+1})^2} \tag{27}$$

where d_i is the distance between adjacent nodes and μ_d is the penalty coefficient of the path distance.

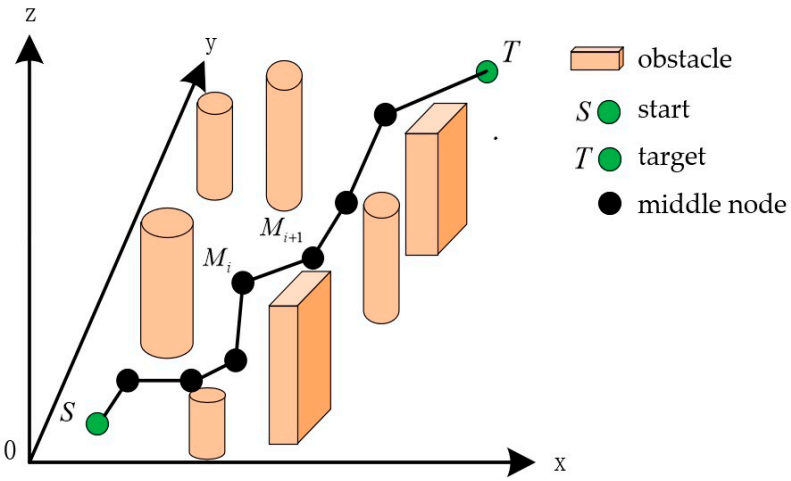


Figure 3. Path distance diagram.

2. Angle change cost $f_2(a)$

UAV path planning must fully consider angle constraints. By penalizing changes in yaw angle, the path planning method is encouraged to generate smooth, coherent paths [13]. When the UAV is flying, the changes in yaw angle and pitch angle should be as small as possible. The cost function is as follows:

$$f_2(a) = f_2^{yaw}(a) + f_2^{pit}(a) \tag{28}$$

$$f_2^{yaw}(a) = \begin{cases} \sum_{i=1}^{M} \mu_{yaw} \times e^{(\theta_i - \theta_{i-1})} & \theta_i \neq \theta_{i-1} \\ 0 & \theta_i = \theta_{i-1} \end{cases} \tag{29}$$

$$f_2^{pit}(a) = \begin{cases} \sum_{i=1}^{M} \mu_{pit} \times e^{(\phi_i - \phi_{\max})} & \phi_i > \phi_{\max} \\ 0 & \phi_i < \phi_{\max} \end{cases} \tag{30}$$

where $f_2^{yaw}(a)$ and $f_2^{pit}(a)$ are the penalty values for the changes in yaw angle and pitch angle, respectively, M is the number of intermediate nodes in each path; and μ_{yaw} and μ_{pit} are the penalty coefficients for the changes in yaw angle and pitch angle, respectively. θ_i and θ_{i-1} are the current yaw angle and the previous yaw angle, respectively. ϕ_i represents the pitch angle of the current node, and $\phi_{\max}$ represents the maximum pitch angle. This restriction is to ensure the maneuverability and safety of the UAV without excessive attitude changes during flight. Thus, the UAV can maintain good motion status and stability in the path planning process with quality and efficiency.

3. Obstacle cost $f_3(obs)$

In complex environments, obstacles are the key factor directly affecting safety and the planning of the UAV [29]. In order to ensure that the UAV can avoid obstacles and maintain a safe flight, the obstacle cost function is defined by the safety constraint associated with obstacles, as shown in Formula (31):

$$f_3(obs) = \begin{cases} \mu_{obs} \times \frac{1}{d_{obs}} & d_{obs} < d_{safe} \\ 0 & d_{obs} > d_{safe} \end{cases} \tag{31}$$

$$d_{obs} = \sum_{i=1}^{M} \sum_{j=1}^{S} \sqrt{\left(x_i - x_{obs_j}\right)^2 + \left(y_i - y_{osb_j}\right)^2 + \left(z_i - z_{obs_j}\right)^2} - (R_{uav} + R_{obs}) \tag{32}$$

where d_{obs} is the distance between the current node and an obstacle, M is the number of nodes, S is the number of obstacles, (x_i, y_i, z_i) represents the coordinates of the current node, $(x_{obs_j}, y_{osb_j}, z_{obs_j})$ are the coordinates of the current obstacle, R_{uav} is the radius of the UAV, R_{obs} is the radius of the obstacle, d_{safe} is the safety distance, and μ_{obs} is the obstacle penalty coefficient.

4. Path evaluation cost F

Taking into account the path distance, yaw and pitch angle, and the safety constraint associated with obstacles, the path cost function used to evaluate path quality is shown in Formula (33):

$$F = \rho_1 \times f_1(d) + \rho_2 \times f_2(a) + \rho_3 \times f_3(obs) \tag{33}$$

$$\rho_1 + \rho_2 + \rho_3 = 1 \tag{34}$$

where $f_1(d)$ is the path distance cost function, $f_2(a)$ is the yaw angle and pitch angle cost function, and $f_3(obs)$ is the obstacle cost function. ρ_1, ρ_2, and ρ_3 are the corresponding weight factors, respectively.

4.2. Path Smoothing

In order to ensure the feasibility and smoothness of the path, the B-spline curve strategy is introduced as shown in Figure 4 [31]. This curve is an interpolation curve, usually used to fit data or generate smooth curves, which can quickly calculate local curve segments without recalculating the entire curve. B-spline converts a given control point into a set of parameters and uses these parameters to calculate points on the curve. The B-spline curve is defined as shown in Formula (35):

$$P(u) = \sum_{i=0}^{N} p_i \times N_{i,k}(u) \tag{35}$$

where p_i is the current control point. $N_{i,k}(u)$ is the current k-degree basis function for B-spline, as defined in Formulas (36) and (37):

$$N_{i,k}(u) = \begin{cases} 1 & u_i \le u \le u_{i+1} \\ 0 & other\ wise \end{cases} \tag{36}$$

$$N_{i,k}(u) = \frac{u - u_i}{u_{i+k} - u_i} N_{i,k-1}(u) + \frac{u_{i+k+1} - u}{u_{i+k+1} - u_{i+1}} N_{i+1,k-1}(u) \tag{37}$$

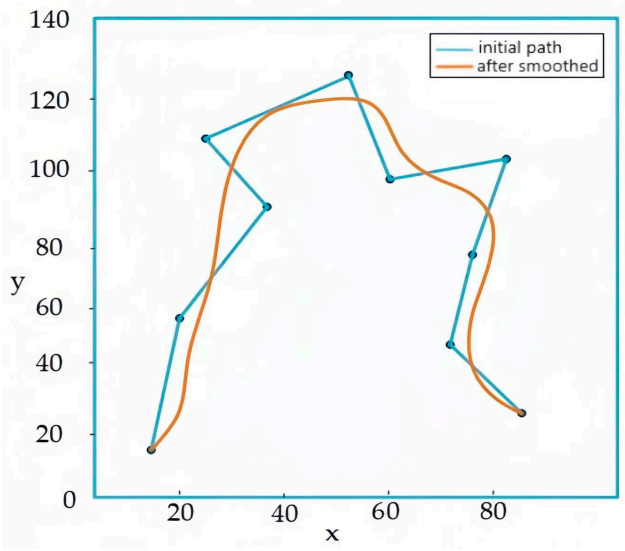


Figure 4. Path diagram smoothed by B-spline.

The basic function is determined by a non-decreasing parameter sequence, and u_i is the current parameter node.

5. Experiment Results

5.1. SSGWO Algorithm Performance

The experiment used the 64-bit Windows 10 operating system for testing, and the experimental software used MATLAB R2020b to perform algorithmic operations. In order to verify the performance of the improved SSGWO algorithm, nine benchmark functions were selected for testing. Table 1 gives the detailed descriptions of nine functions, including function expression, number of iterations, range of values, and theoretical optimal values. In order to fully prove the feasibility and accuracy of the SSGWO algorithm and at the same time ensure the relative fairness of the algorithm's operation results, this paper unifies the algorithm test method and the test data in the simulation experiment. The population number was set to 30, the maximum number of iterations was 500, and all experiments were run independently 50 times to prevent the influence of randomness on the algorithm's operation results. GWO [16], MGWO [26], IGWO [27], SOGWO [28], and the improved algorithm SSGWO from this paper were selected for better optimization results. The simulation experiment compared and recorded the optimal value, average value, and standard deviation of the experimental results. Table 2 records a comparison of the average values and standard deviations of the results of 50 independent runs of the algorithm. The smaller the average value, the higher the optimization accuracy of the algorithm; the smaller the standard deviation, the more stable the algorithm.

Table 1. Benchmark functions.

Function	Iterations	Ranges	Optimal Value
$f_1(x) = \sum_{i=1}^{n} x_i^2$	500	$[-100, 100]$	0
$f_2(x) = \sum_{i=1}^{n} x_i + \prod_{i=1}^{n} \lvert x_i \rvert$	500	$[-10, 10]$	0
$f_3(x) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$	500	$[-30, 30]$	0
$f_4(x) = \max\lvert x_i \rvert, 1 \le i \le n$	500	$[-100, 100]$	0
$f_5(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2$	500	$[-100, 100]$	0
$f_6(x) = \sum_{i=1}^{n} i x_i^4 + random[0,1)$	500	$[-1.28, 1.28]$	0
$f_7(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	500	$[-5.12, 5.12]$	0
$f_8(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{30}\sum_{i=1}^{n} \cos(2\pi x_i) + 20 + e\right)$	500	$[-32, 32]$	0
$f_9(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	500	$[-600, 600]$	0

Table 2. Results of benchmark functions.

Function	Algorithm	Average Value	Standard Deviation	Optimal Value
f1	GWO	3.2532×10^{-26}	1.3267×10^{-26}	4.4327×10^{-30}
	MGWO	3.5394×10^{-196}	2.8467×10^{-198}	0
	IGWO	2.8327×10^{-28}	3.2597×10^{-29}	1.3879×10^{-32}
	SOGWO	3.3227×10^{-76}	2.79548×10^{-78}	1.3261×10^{-80}
	SSGWO	0	2.6741×10^{-199}	1.1349×10^{-199}

Table 2. *Cont.*

Function	Algorithm	Average Value	Standard Deviation	Optimal Value
f2	GWO	3.2671×10^{-13}	3.2481×10^{-14}	2.2738×10^{-15}
	MGWO	1.5371×10^{-77}	2.6137×10^{-78}	1.3367×10^{-80}
	IGWO	3.1679×10^{-19}	3.2467×10^{-20}	3.2497×10^{-23}
	SOGWO	1.8342×10^{-65}	2.8643×10^{-68}	2.1973×10^{-70}
	SSGWO	1.1463×10^{-85}	2.7239×10^{-86}	1.2371×10^{-87}
f3	GWO	2.8237×10^{01}	2.2537×10^{-01}	2.6192×10^{-01}
	MGWO	2.7136×10^{01}	6.2276×10^{-01}	4.7431×10^{-01}
	IGWO	2.4837×10^{01}	2.6927×10^{-01}	4.4293×10^{-01}
	SOGWO	2.5197×10^{01}	2.4977×10^{-01}	2.3291×10^{-01}
	SSGWO	2.0637×10^{01}	4.1876×10^{-02}	2.4037×10^{-02}
f4	GWO	2.6217×10^{-06}	3.5127×10^{-06}	6.1837×10^{-07}
	MGWO	5.2739×10^{-90}	3.2267×10^{-90}	1.6239×10^{-91}
	IGWO	3.3392×10^{-46}	1.9643×10^{-45}	1.7234×10^{-48}
	SOGWO	3.5731×10^{-96}	4.3797×10^{-98}	2.7944×10^{-101}
	SSGWO	0	0	0
f5	GWO	2.7945×10^{-07}	2.8153×10^{-06}	4.1687×10^{-08}
	MGWO	3.3754×10^{-160}	4.7687×10^{-161}	0
	IGWO	2.2738×10^{-04}	3.7327×10^{-04}	1.7822×10^{-06}
	SOGWO	1.7322×10^{-106}	2.7528×10^{-107}	4.2975×10^{-109}
	SSGWO	2.7254×10^{-178}	3.3784×10^{-180}	4.8374×10^{-181}
f6	GWO	1.4687×10^{-03}	1.3573×10^{-03}	2.6874×10^{-04}
	MGWO	1.7225×10^{-04}	1.2539×10^{-04}	7.1783×10^{-05}
	IGWO	1.2769×10^{-04}	1.2677×10^{-04}	2.5723×10^{-05}
	SOGWO	3.6728×10^{-04}	3.7392×10^{-05}	1.5973×10^{-05}
	SSGWO	2.9271×10^{-05}	3.2795×10^{-05}	5.3764×10^{-06}
f7	GWO	2.1733×10^{-09}	4.2734×10^{-08}	0
	MGWO	0	0	0
	IGWO	4.3651×10^{-12}	5.7532×10^{-13}	0
	SOGWO	1.8327×10^{-45}	4.3687×10^{-46}	0
	SSGWO	0	0	0
f8	GWO	1.2764×10^{-13}	1.4791×10^{-14}	2.5794×10^{-14}
	MGWO	3.9763×10^{-15}	5.7219×10^{-15}	2.4564×10^{-16}
	IGWO	2.3715×10^{-14}	3.7912×10^{-15}	1.3741×10^{-14}
	SOGWO	3.7941×10^{-18}	2.7646×10^{-18}	0
	SSGWO	0	0	0
f9	GWO	3.7911×10^{-04}	2.3257×10^{-03}	0
	MGWO	0	0	0
	IGWO	7.7497×10^{-04}	5.4259×10^{-04}	2.7491×10^{-05}
	SOGWO	0	0	0
	SSGWO	0	0	0

As shown in Table 2, for the benchmark function f1, the averages and standard deviations for the SSGWO algorithm are better than for the other four algorithms, and the theoretical optimal value is close to 0. Taken together, the optimization effect of the SSGWO algorithm is better than of the other four algorithms; for function f2, the average and standard deviation of the improved algorithm are 1.1463×10^{-85} and 2.7239×10^{-86}, respectively. The optimization accuracy and stability of this algorithm are significantly higher than of GWO and IGWO. On average, it is nearly 10 orders of magnitude better than other algorithms in terms of numerical accuracy; for functions f3 and f4, the IGWO algorithm has better search capabilities, but lacks the ability to converge to globally better values, because it does not incorporate an appropriate perturbation strategy. The SSGWO algorithm achieved the best optimization results, and the average value, standard deviation,

and optimal value of the algorithm all reached the theoretical optimal value 0. For function f5, the solution accuracy between algorithms shows a small range of differences, with an average difference of only one order of magnitude. Although the optimal value of SSGWO is slightly lower than the accuracy of the MGWO algorithm, the mean and standard deviation are smaller than that of each compared algorithm. This shows that SSGWO has stronger algorithmic optimization capabilities in its test functions; for test function f6, the three indicators calculated by SSGWO are all better than those of the other algorithms, so the optimization accuracy and stability are outstanding; for functions f7, f8, and f9, SSGWO not only achieves the best average value, but its standard deviation is also the smallest, indicating that SSGWO has high robustness.

As shown in Table 2, for the average values among the nine benchmark functions, the average results of the SSGWO algorithm are better than those of the other algorithms, and the average values for functions f1, f4, f7, f8, and f9 all obtained the theoretical optimal value 0; for standard deviation, MGWO and SSGWO achieved the optimal value 0 for functions f7 and f9, SOGWO and SSGWO achieved the optimal value 0 for function f9, and the standard deviation results for the SSGWO algorithm were better than for the others on the benchmark test function.

Compared with the standard GWO algorithm, the SSGWO algorithm achieved better optimization results on nine functions. For function f3, the GWO and SSGWO algorithms achieved similar results; for function f6, the standard GWO algorithm achieved better results. But the mean and standard deviation for SSGWO were better than those for standard GWO.

Compared with the MGWO algorithm, SSGWO obtained better optimization results on the f1 function, and MGWO obtained better optimal values. For functions f2, f3, f5, and f6, all three indicators for SSGWO were slightly better than for MGWO; for the functions f4, f7, f8, and f9, both the mean and standard deviation of SSGWO obtained the theoretical optimal value 0.

Compared with the IGWO algorithm, for function f6, SSGWO and IGWO obtained similar average values and standard deviations; for function f3, SSGWO and IGWO obtained similar average values. For the other functions, the mean and standard deviation of SSGWO were significantly better than for IGWO.

Compared with SOGWO, for function f9, the average, standard deviation, and optimal value for the two algorithms all obtained the theoretical optimal value 0; for other functions, SOGWO achieved better optimization results, but SSGWO was better than SOGWO in that most of the test functions had a higher solution accuracy optimization effect.

The above comparison results indicate that SSGWO exhibits higher solution accuracy and stability compared with the standard GWO, MGWO, IGWO, and SOGWO algorithms across most test functions.

5.2. Path Planning

The main goal of path planning is to find optimal and collision-free trajectories. To assess the effectiveness of the SSGWO algorithm in addressing path planning challenges, the algorithm underwent comparisons with relevant algorithms through simulated examples. MATLAB R2020b software was employed to conduct the simulations.

5.2.1. Path Planning Based on Eight Obstacle Threat Areas

In order to verify the performance of the SSGWO algorithm, this study simulated the flight environment for UAVs in field vegetation areas. The SSGWO algorithm was applied to UAV path planning for verification analysis, and the path was smoothed through the B-spline curve. The size of the flight space was $800 \times 800 \times 20$, and the starting point and end point were [0, 0, 0] and [800, 800, 15], respectively. The cylindrical obstacles and obstacle information are shown in Table 3.

Table 3. Information 1 on the obstacles.

Obstacle Center	Obstacle Radius/m	Obstacle Height/m
(250, 200, 9)	40	18
(600, 700, 9)	30	18
(130, 120, 8)	30	16
(300, 280, 8.5)	40	17
(350, 600, 9.5)	40	19
(480, 400, 9)	60	18
(700, 650, 8.5)	40	17
(720, 760, 9.5)	50	19

The GWO, MGWO, IGWO, SOGWO, and SSGWO algorithms were used for path planning and compared. The maximum number of iterations was 500, the initial number of wolves was 30, the simulation experiment was run 50 times, and the number of path nodes M was set to 30. In order to display the path effect more clearly, the three perspectives with the best results are shown in Figures 5–7, with five paths in each figure, corresponding to the five different algorithms for UAV path planning under the same environmental constraints, showing a comparison of the path planning trajectories of the five algorithms in the three-dimensional environment. Figure 8 shows the path top view, and Table 4 records the flight records of the UAV, where pitch represents the average pitch angle of the M nodes on the path, yaw represents the average yaw angle difference between two adjacent nodes among the M nodes on the path, and dis-to-obs represents the average distance between the current node and the nearest obstacle among M nodes on the path.

Table 4. Navigation data 1.

Algorithm	Minimum Distance/m	Average Distance/m	Number of Collisions	Pitch/°	Yaw/°	Dis-to-Obs/m
GWO	1376	1462	5	41	57	102
MGWO	1268	1379	3	35	51	81
IGWO	1337	1394	2	37	53	95
SOGWO	1233	1297	2	34	48	65
SSGWO	1132	1203	0	29	41	57

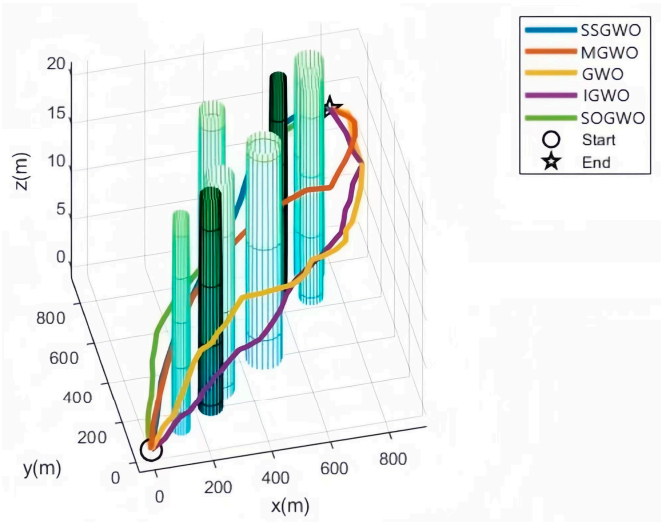


Figure 5. Perspective 1.

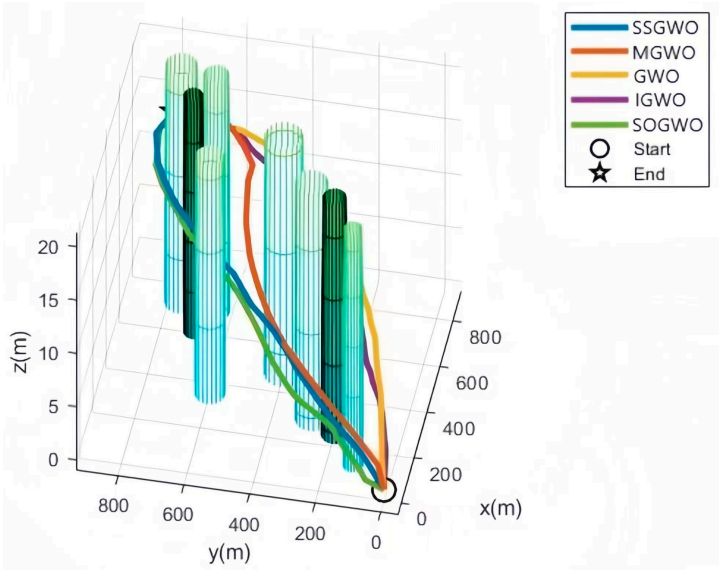


Figure 6. Perspective 2.

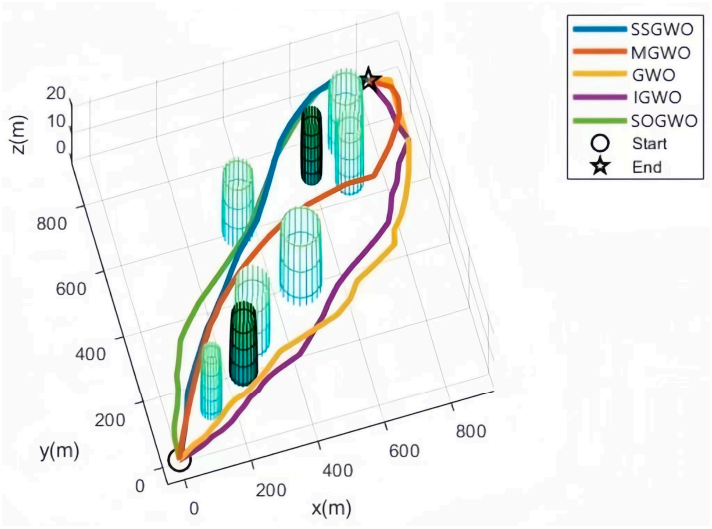


Figure 7. Perspective 3.

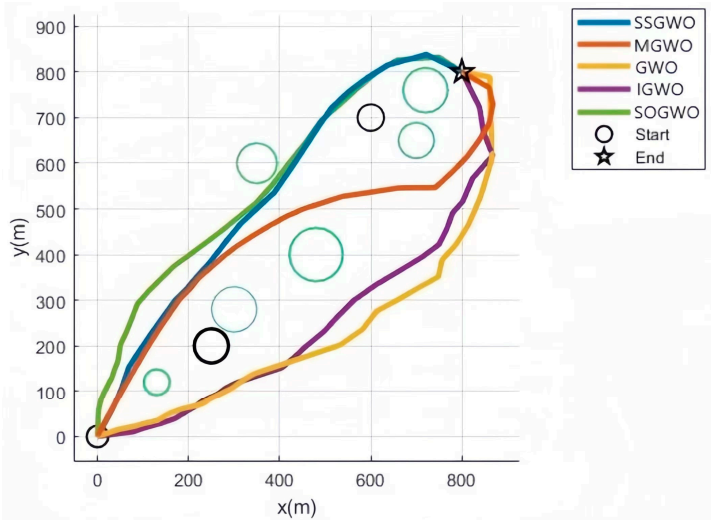


Figure 8. Top view 1.

As shown in Figures 5–8, these five algorithms can effectively find feasible solutions. This fact fully proves that these algorithms have extremely high practical value for practical applications. Compared with traditional mathematical methods, these metaheuristic algorithms show more outstanding practicability in problem solving. They can effectively overcome diversity and uncertainty in complex problems. In terms of optimal results, GWO and the four GWO variants showed different performance. In terms of performance, the GWO algorithm showed relatively weak performance, mainly because it is highly dependent on the first three wolves in the optimization process, which often causes the algorithm to quickly fall into a local optimal solution. The path planned by GWO is the farthest from the obstacle, but the path is too long and is less helpful for practical applications. SSGWO has higher convergence accuracy and stability than the other algorithms. The path found by the SSWGO algorithm is better than for the other four algorithms. Although it is the closest to the obstacle, it ensures the shortest path and also meets the safe distance requirement. The minimum path distance and the average path distance are both optimal values, and in the 50 optimization processes the number of collisions was 0. The path length for SOGWO is shorter, but the planned path has too many inflection points, which affects navigation efficiency and is not conducive to the applications of UAVs.

Through comparative experiments, the SSWGO algorithm can plan feasible and safe optimal paths for UAVs to perform agricultural tasks. Its application in agricultural production can help improve production efficiency and reduce costs. By improving the limitations of the GWO algorithm, a safe and reliable path can be planned stably. This path can not only satisfy various constraints under the premise of optimal cost, but also make full use of the performance of the algorithm to ensure the effectiveness and feasibility of path planning. This improved method has wide applicability for solving practical problems and provides a more reliable and efficient solution for the application of UAV path planning.

5.2.2. Path Planning Based on Twelve Obstacle Threat Areas

In order to further prove the effectiveness of SSGWO, the number of obstacles was increased and tested in a simulation task. As the complexity of the drone flight environment increases, multi-dimensional testing is needed to analyze the performance advantages and disadvantages of various algorithms. By simulating complex environments, the performance of each algorithm can be more comprehensively evaluated and the trajectory planning path maps solved by each algorithm can be generated to test the scalability of the SSGWO algorithm for agricultural UAV path planning under the constraints of 12 obstacle threat areas. The trajectory results are shown in Figures 9 and 10. The obstacle information is shown in Table 5, and Table 6 records the UAV flight records.

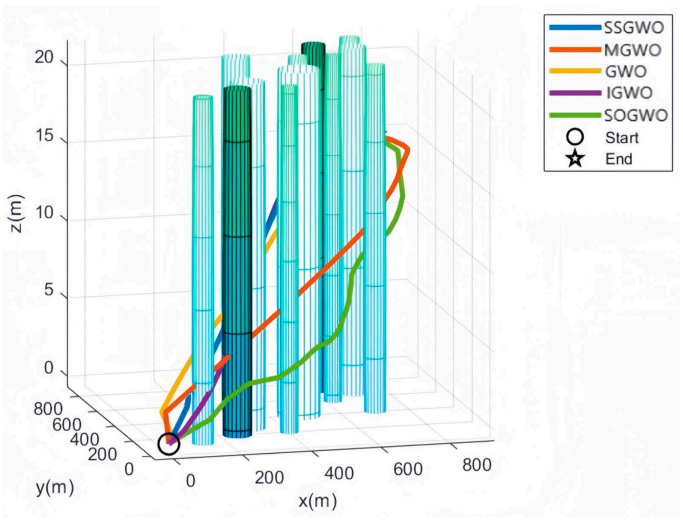


Figure 9. Path planning based on 12 obstacles.

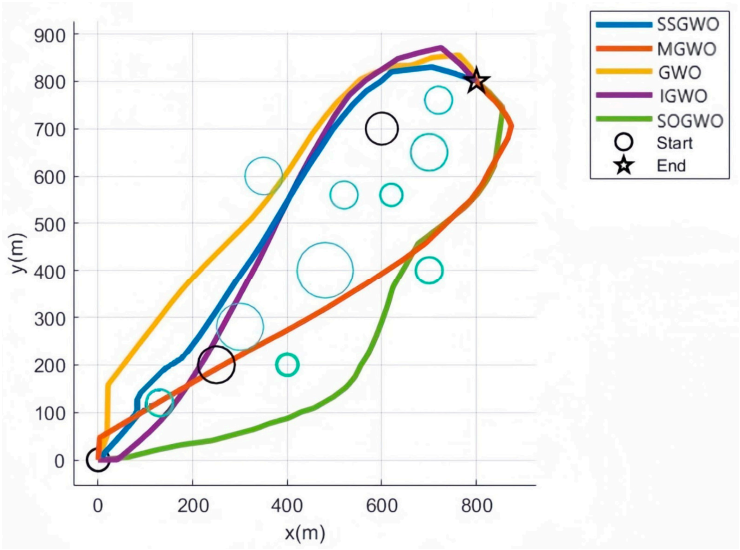


Figure 10. Top view 2.

Table 5. Information 2 on the obstacles.

Obstacle Center	Obstacle Radius/m	Obstacle Height/m
(250, 200, 9)	40	18
(600, 700, 8)	35	16
(130, 120, 8)	30	16
(300, 280, 8.5)	50	17
(350, 600, 9.5)	40	19
(480, 400, 9)	60	18
(700, 650, 8.5)	40	17
(720, 760, 8.5)	30	17
(620, 650, 7.5)	25	15
(520, 560, 9.5)	30	19
(400, 200, 8)	25	16
(700, 400, 8.5)	30	17

Table 6. Navigation data 2.

Algorithm	Minimum Distance/m	Average Distance/m	Number of Collisions	Pitch/°	Yaw/°	Dis-to-Obs/m
GWO	1278	1351	8	44	59	93
MGWO	1281	1362	5	37	56	87
IGWO	1226	1376	4	39	55	76
SOGWO	1321	1347	3	33	51	94
SSGWO	1167	1203	0	26	40	68

In complex environments with multiple constraints, the path planned by the SSGWO algorithm not only appears to be the shortest, but also maintains the highest level of smoothness. By observing the five algorithms in Figures 9 and 10, it can be clearly seen that, as the complexity of the environment increases, only the paths planned by the two algorithms SOGWO and SSGWO show greater feasibility. In 50 experiments, the number of collisions with the SSGWO algorithm was zero, which means that its planned path is not only the shortest, but also performs best in terms of smoothness. The SSGWO algorithm shows excellent performance under complex environmental models and successfully generates safe and feasible flight paths. This demonstrates SSGWO's strong adaptability to handle diverse threats and constraints, and proves the feasibility of SSGWO in complex environments.

5.2.3. Path Planning in Agricultural Environments

In order to further verify the adaptability of the SSGWO algorithm in agriculture, a terrain with obstacles in a three-dimensional agricultural environment was simulated. The simulation results are shown in Figures 11 and 12, the obstacle information is shown in Table 7, and the flight records are shown in Table 8.

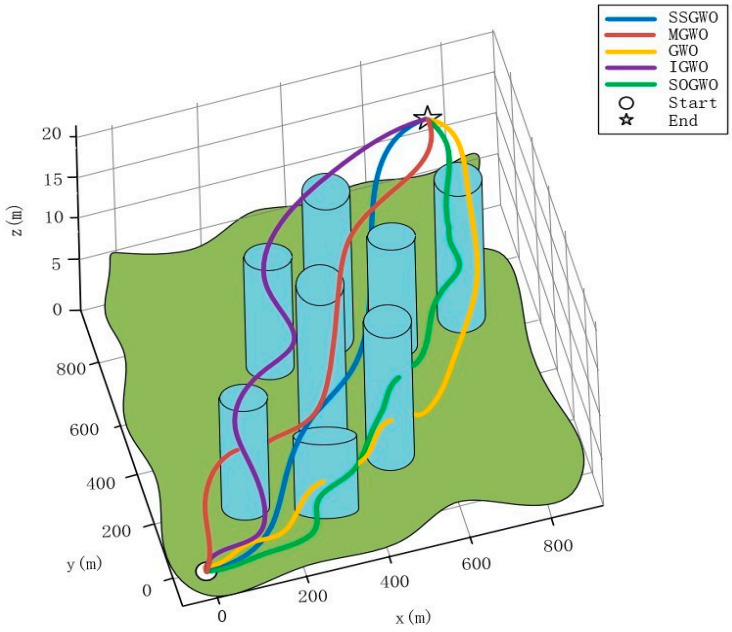


Figure 11. Path planning in an agricultural environment.

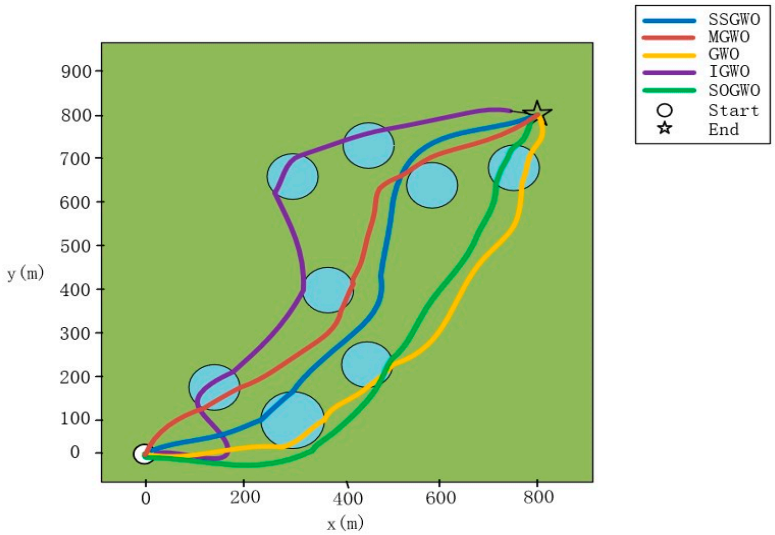


Figure 12. Top view 3.

From Figures 11 and 12, among the five algorithms selected in this paper, only SSGWO and IGWO generated safe paths. However, the path planned by IGWO has some problems, including that the pitch angle is too large, the path is too long, and the path is not smooth enough. Table 8 shows that, in 50 simulation experiments, SSGWO performed well, with only one collision, and the planned path was the shortest. It also satisfied the angle change constraints and obstacle constraints. This proves that the path planned by the SSGWO algorithm is still the shortest and smoothest under the influence of various threats and constraints. This provides a reliable solution for path planning for UAVs in agricultural applications.

Table 7. Information 3 on the obstacles.

Obstacle Center	Obstacle Radius/m	Obstacle Height/m
(150, 180, 7)	45	13
(300, 100, 6)	60	11
(380, 400, 8)	45	15
(480, 210, 8)	45	14
(320, 650, 7.5)	45	13.5
(430, 700, 8)	45	14
(560, 630, 7)	45	13
(750, 660, 8)	45	14

Table 8. Navigation data 3.

Algorithm	Minimum Distance/m	Average Distance/m	Number of Collisions	Pitch/°	Yaw/°	Dis-to-Obs/m
GWO	1263	1358	18	53	64	92
MGWO	1256	1289	14	45	54	84
IGWO	1292	1384	9	57	49	97
SOGWO	1225	1276	12	39	49	61
SSGWO	1129	1178	1	26	39	51

6. Discussion

Through simulation tests and comparative experiments, the SSGWO algorithm has good performance in terms of convergence and stability and can create feasible and safe optimal trajectory paths for UAVs when performing tasks. The advantages and disadvantages of the work done in this paper and in other literature are recorded in Table 9.

Table 9. Comparison of advantages and disadvantages.

Algorithm	Advantages	Disadvantages
GWO	It is easy to implement and performs well on some optimization problems.	The convergence speed is relatively slow. When dealing with some problems with complex local structures, the local search ability is relatively weak and it is easy to fall into the local optimal solution.
MGWO	Prevents the algorithm from falling into the local optimal solution prematurely.	Increased complexity can make it more difficult to understand and use the algorithm.
IGWO	A balance is achieved between local and global search, helping to better explore the problem space.	The choice of initial solution will have an impact on the performance of the algorithm.
SOGWO	Improves global search performance and prevents the algorithm from falling into local optimality.	It is sensitive to the nature of the problem and is not suitable for all types of optimization problems.
SSGWO	Performs well in global search and improves convergence through the SA algorithm.	The adaptability to specific problems is weak or parameters may need to be adjusted for some problems.

The MGWO proposed in [26] introduces an exponential regular convergence factor strategy, an adaptive update strategy, and a dynamic weighting strategy to enhance the search performance of GWO, but there are still some potential shortcomings, and the introduction of new strategies and factors may increase the complexity of the algorithm, including the difficulty of implementing and tuning these parameters. This may make it more difficult to understand and use the algorithm.

The IGWO algorithm proposed in [27] shows excellent search capabilities, but it has some shortcomings in its ability to converge to the global optimal value. This is due to the lack of appropriate perturbation strategies. This algorithm shows good flexibility and extensive exploration capabilities during the search process, but it is relatively weak in quickly converging to the global optimal solution.

SOGWO, proposed in [28], applies Spearman's coefficient to determine wolf pack operations. Spearman's coefficient is a non-parametric statistical method that does not depend on the specific distribution of data. It is insensitive to the distribution shape of the data and is suitable for various types of data. The Spearman coefficient is highly sensitive to sorting information, which may result in an inability to accurately reflect small changes in operations. Therefore, the planned path will have more turning points.

7. Conclusions

Aiming at the path planning problem for agricultural UAVs, a new hybrid algorithm named SSGWO was proposed. In this paper, a better convergence factor was proposed; the position update equation was improved; and, combined with a suitable fitness function, the iterative process of the global optimal solution of the GWO algorithm was combined with the SA algorithm. This improves the diversity of the population, avoids falling into the local optimal solution, and enhances the search capability of the SSGWO algorithm. The SSGWO algorithm was effectively applied to solve the three-dimensional UAV path planning problem by converting it into an optimization problem. The optimal path is determined by minimizing the cost function, and, finally, the B-spline curve is used to smooth the path to improve path quality and feasibility. UAV path planning faces multiple challenges, such as energy consumption, flight distance, and safety in real-life applications. This paper considers the cost of yaw angle changes and pitch angles during the path planning process, which effectively prevents the UAV from turning too sharply during flight and improves the stability of overall path planning. The path length cost is considered to make the UAV flight path shorter and reduce energy consumption. In addition, we also considered drone and obstacle costs to ensure that the drone can avoid obstacles and follow the predetermined path. Experimental results show that the SSGWO algorithm has better performance than the GWO, MGWO, IGWO, and SOGWO algorithms. This method has achieved remarkable results in path planning and can better meet flight distance restrictions, safety constraints, and steering angle requirements. The effectiveness and significance of the SSGWO algorithm for agricultural UAV path planning problems were proved.

Although SSGWO has demonstrated notable achievements in algorithmic performance and path planning, this research also has some limitations. There are special environments in agricultural production, such as different types of farmland, climate changes in different seasons, etc., which require higher real-time performance from the algorithm. In future work, we will try to use multi-UAV systems to solve more complex agricultural tasks. Research should explore collaborative operations between UAVs, realize information sharing and collaborative operations between multiple UAVs, and improve the efficiency of agricultural production.

Author Contributions: Conceptualization, J.F. and C.S.; methodology, J.F. and C.S.; software, C.S.; validation, J.F., C.S., J.Z. and Y.D. (Yue Du); formal analysis, J.F.; investigation, J.F., C.S. and Y.D. (Yue Du); resources, C.S.; data curation, J.Z.; writing—original draft preparation, C.S.; writing—review and editing, J.F.; visualization, J.F., C.S., Z.L. and Y.D. (Yuanming Ding); supervision, J.F.; project administration, C.S.; funding acquisition, J.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported in part by the interdisciplinary project of Dalian University (DLUXK-2023-ZD-001).

Data Availability Statement: The data are not publicly available due to privacy restrictions.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

Parameters	Definition
UAV	Unmanned aerial vehicle
PSO	Particle swarm optimization
GA	Genetic algorithm
QPSO	Quantum particle swarm optimization
GWO	Gray wolf optimization
MGWO	Modified grey wolf optimization algorithm for global optimization problems
IGWO	Improved grey wolf optimizer for solving engineering problems
DLH	Dimensional learning hunting
SOGWO	Selective-opposition-based grey wolf optimization
SI	Swarm intelligence
SSGWO	A UAV path-planning method in three-dimensional space based on the hybrid gray wolf optimization algorithm
SA	Simulated annealing
B-spline	Basis spline

References

1. Lin, H.X.; Xiang, D.; Ou, Y.J. Review of Path Planning Algorithms for Mobile Robots. *Comput. Eng. Appl.* **2021**, *57*, 38–48.
2. Rahman, M.F.F.; Fan, S.; Zhang, Y.; Chen, L. A Comparative Study on Application of Unmanned Aerial Vehicle Systems in Agriculture. *Agriculture* **2021**, *11*, 22. [CrossRef]
3. Bassetto, M.; Quarta, A.A.; Mengali, G. Generalized sail trajectory approximation with applications to MagSails. *Aerosp. Sci. Technol.* **2021**, *118*, 106991. [CrossRef]
4. Azzabi, A.; Regaieg, M.; Adouane, L.; Nasri, O. Hybrid and multi-controller architecture for autonomous system: Application to the navigation of a mobile robot. In Proceedings of the 2014 11th International Conference on Informatics in Control, Automation and Robotics (ICINCO), Vienna, Austria, 2–4 September 2014; pp. 491–497.
5. Feng, J.; Zhang, J.; Zhang, G.; Xie, S.; Ding, Y.; Liu, Z. UAV Dynamic Path Planning Based on Obstacle Position Prediction in an Unknown Environment. *IEEE Access* **2021**, *9*, 154679–154691. [CrossRef]
6. Chen, J.; Tan, C.; Mo, R.; Wang, Z.; Wu, J.; Zhao, C. Path planning of mobile robot with A* algorithm based on the artificial potential field. *Comput. Sci.* **2021**, *48*, 327–333.
7. Hu, Z.; Chunyi, F.; Yuan, L. Improved particle swarm optimization algorithm for mobile robot path planning. *Comput. Appl. Res.* **2021**, *38*, 3089–3092.
8. Wang, H.; Zhao, X.; Yuan, X. Robot Path Planning Based on Improved Adaptive Genetic Algorithm. *Electron. Optics Control* **2022**, *29*, 72–76.
9. You, S.; Diao, M.; Gao, L. Deep reinforcement learning for tar-get searching in cognitive electronic warfare. *IEEE Access* **2019**, *7*, 37432–37447. [CrossRef]
10. Azar, A.; Vaidyanathan, S. *Computational Intelligence Applications in Modeling and Control*; Springer: Berlin/Heidelberg, Germany, 2015.
11. Sun, C.; Ni, W.; Wang, X. Joint Computation Offloading and Trajectory Planning for UAV-Assisted Edge Computing. *IEEE Trans. Wirel. Commun.* **2021**, *20*, 5343–5358. [CrossRef]
12. Yu, X.; Li, C.; Zhou, J.F. A constrained differential evolution algorithm to solve UAV path planning in disaster scenarios. *Knowl.-Based Syst.* **2020**, *204*, 106209. [CrossRef]
13. Phung, M.D.; Ha, Q.P. Safety-enhanced UAV path planning with spherical vector-based particle swarm optimization. *Appl. Soft Comput.* **2021**, *107*, 107376. [CrossRef]
14. Huang, C.; Fei, J.; Deng, W. A Novel Route Planning Method of Fixed-Wing Unmanned Aerial Vehicle Based on Improved QPSO. *IEEE Access* **2020**, *8*, 65071–65084. [CrossRef]
15. Chen, Y.; Pi, D.; Xu, Y. Neighborhood global learning based flower pollination algorithm and its application to unmanned aerial vehicle path planning. *Expert Syst. Appl.* **2020**, *170*, 114505. [CrossRef]
16. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
17. Jarray, R.; Al-Dhaifallah, M.; Rezk, H.; Bouallègue, S. Parallel cooperative coevolutionary grey wolf optimizer for path planning problem of unmanned aerial vehicles. *Sensors* **2022**, *22*, 1826. [CrossRef] [PubMed]
18. Peng, T.; Zhou, B. Hybrid bi-objective gray wolf optimization algorithm for a truck scheduling problem in the automotive industry. *Appl. Soft Comput.* **2019**, *81*, 105513. [CrossRef]
19. Rao, R.V.; Savsani, V.J.; Vakharia, D.P. Teaching learning-based optimization: A novel method for constrained mechanical design optimization problems. *Comput. Aided Des.* **2011**, *43*, 305–311. [CrossRef]
20. Long, W.; Cai, S.; Jiao, J.; Xu, M.; Wu, T. A new hybrid algorithm based on grey wolf optimizer and cuckoo search for parameter extraction of solar photovoltaic models. *Energy Conv. Manag.* **2020**, *203*, 112243. [CrossRef]

21. Zhang, C.; Wang, W.; Pan, Y. Enhancing electronic nose performance by feature selection using an improved grey wolf optimization based algorithm. *Sensors* **2020**, *19*, 4065. [CrossRef]
22. Goel, T.; Murugan, R.; Mirjalili, S.; Chakrabartty, D.K. OptCoNet: An optimized convolutional neural network for an automatic diagnosis of COVID-19. *Appl. Intell.* **2021**, *51*, 1351–1366. [CrossRef]
23. Gupta, S.; Deep, K. A memory-based Grey Wolf Optimizer for global optimization tasks. *Appl. Soft Comput.* **2020**, *93*, 106367. [CrossRef]
24. Shao, S.; Peng, Y.; He, C.; Du, Y. Efficient path planning for UAV formation via comprehensively improved particle swarm optimization. *ISA Trans.* **2020**, *97*, 415–430. [CrossRef] [PubMed]
25. Zhang, W.; Zhang, S.; Wu, F.; Wang, Y. Path planning of UAV based on improved adaptive grey wolf optimization algorithm. *IEEE Access* **2021**, *9*, 89400–89411. [CrossRef]
26. Zhang, Y. Modified grey wolf optimization algorithm for global optimization problems. *J. Univ. Shanghai Sci. Techol.* **2021**, *43*, 73–82.
27. Nadimi-Shahraki, M.H.; Taghian, S.; Mirjalili, S. An improved grey wolf optimizer for solving engineering problems. *Expert Syst. Appl.* **2021**, *166*, 113917. [CrossRef]
28. Dhargupta, S.; Ghosh, M.; Mirjalili, S.; Sarkar, R. Selective Opposition based Grey Wolf Optimization. *Expert Syst. Appl.* **2020**, *151*, 113389. [CrossRef]
29. Jiang, W.; Lyu, Y.; Li, Y.; Guo, Y.; Zhang, W. UAV path planning and collision avoidance in 3D environments based on POMPD and improved grey wolf optimizer. *Aerosp. Sci. Technol.* **2022**, *121*, 107314. [CrossRef]
30. Zhao, X. Simulated Annealing Algorithm with Adaptive Neighborhood. *Appl. Soft Comput.* **2011**, *11*, 1827–1836.
31. Cao, H.; Zoldy, M. Implementing B-Spline Path Planning Method Based on Roundabout Geometry Elements. *IEEE Access* **2022**, *10*, 81434–81446. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.