

## Article

# The Improved A\* Algorithm for Quadrotor UAVs under Forest Obstacle Avoidance Path Planning

Jiale Li <sup>1</sup>, Feng Kang <sup>1</sup> , Chongchong Chen <sup>1</sup>, Siyuan Tong <sup>1</sup>, Yalan Jia <sup>2</sup>, Chenxi Zhang <sup>1</sup> and Yaxiong Wang <sup>1,\*</sup> <sup>1</sup> Key Laboratory of State Forestry and Grassland Administration on Forestry Equipment and Automation, School of Technology, Beijing Forestry University, Beijing 100083, China<sup>2</sup> UAV Center Amy Aviation Institute, Beijing 101116, China

\* Correspondence: yaxiongwang87@bjfu.edu.cn

**Abstract:** In order to improve the obstacle avoidance and endurance capability of quadrotor UAVs performing tasks such as forest inspection and rescue search, this paper proposes improvements to address the problems of too many traversed nodes, too many redundant corners, too-large turning angles and unsmooth generated paths in the traditional A\* algorithm in path planning. The traditional A\* algorithm is improved by using a segmented evaluation function with dynamic heuristic and weighting processing, a steering cost heuristic function based on heading angle deviation control, a redundant turning points removal strategy, and a quasi-uniform cubic b-spline. Through the test comparison of different complexity map scenarios, it is found that the improved A\* algorithm reduces the number of traversed nodes by 64.87% on average, the total turning angle by 54.53% on average, the path search time by 49.64% on average, and the path length by 12.52% on average compared to the traditional A\* algorithm, and there is no obvious turning point in the path. The real-world applicability of the improved A\* algorithm is confirmed by comparing the effect of different algorithms on obstacle avoidance in a map of a real plantation forest environment.

**Keywords:** UAV; improved A\* algorithm; path planning; remove redundant turning points; path smoothing



**Citation:** Li, J.; Kang, F.; Chen, C.; Tong, S.; Jia, Y.; Zhang, C.; Wang, Y. The Improved A\* Algorithm for Quadrotor UAVs under Forest Obstacle Avoidance Path Planning. *Appl. Sci.* **2023**, *13*, 4290. <https://doi.org/10.3390/app13074290>

Academic Editors: Yosoon Choi and Dimitris Mourtzis

Received: 11 December 2022

Revised: 5 March 2023

Accepted: 23 March 2023

Published: 28 March 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) have been extensively utilized in forestry applications such as forest inspection, rescue, and monitoring due to their high flexibility and real-time inspection capabilities [1,2]. However, limited payload capacity and hardware technology pose challenges to the endurance of UAVs [3]. With the expansion of plantation forest areas, it has become crucial to enhance the autonomous flight and endurance capabilities of UAVs to efficiently perform continuous inspection and monitoring tasks.

Due to the level of industrial development, most researchers have optimized the path planning algorithm to achieve reasonable task path planning according to the actual application needs of UAVs, thereby reducing energy consumption and enhancing obstacle avoidance and safety [4]. Key performance indicators, such as the shortest path length, path search time, number of turning points, total turning angle, and path smoothness, are used to evaluate energy consumption [5]. Obstacle avoidance path planning techniques have been extensively studied in two-dimensional environments [6,7] to plan safe and collision-free paths in working environments with obstacles. These techniques have gradually been applied to automatic manipulators [8], ground robots [9], and other related fields. The rapidly-exploring random tree (RRT) [10] algorithm, based on uniform random sampling, is among the global path planning algorithms that have probabilistic completeness. However, its expansion efficiency is low, and it tends to fall into optimal local values. To address these limitations, a number of variants have been derived, such as RRT-connect [11], Q-RRT\* [12], Informed RRT\* [13], BIT\* [14], etc. However, these algorithms do not consider path

smoothness, resulting in a large number of turning points that increase energy consumption. To address this issue, many scholars have attempted to use intelligent algorithms to solve related problems.

Yang et al. [15] proposed an improved RRT algorithm that introduces ant colony optimization (ACO) in the process of expanding the random tree to make the planning path asymptotically optimal. Li et al. [16] introduced the metropolitan criterion into the ACO node screening mechanism to solve the multi-UAV trajectory planning problem in static environments. Zhou et al. [17] proposed a 3D spatial path planning algorithm that imitates basic mechanisms of plant growth, including phototropism, negative geotropism, and branching to achieve fast planning speed, few path nodes, and low-latency, real-time planning. Zhang et al. [18,19] proposed the use of a priori maps to pre-plan alternative paths offline and select one of them for execution by online navigation. They later proposed a heuristic navigation method without a priori maps [20] that use an environment-based probabilistic representation to maximize the likelihood of successful navigation to a target. These algorithms improve the efficiency of path search and achieve asymptotic optimality of paths to some extent, but path smoothing is not well covered.

Several researchers have applied neural network techniques to the field of path planning. Among them, Fan et al. [21] proposed an efficient and locally connected Hopfield neural network (HNN) that dynamically tracks the numerical potential field of the HNN using a hill-climbing method to find collision-free paths without any unexpected local attraction points. Gao et al. [22] proposed a novel competitive self-organizing neural network algorithm with new adversarial-based learning to generate better neurons in the initial stage. They followed it up with a secondary competitive layer on top of the hidden layer, thus improving the accuracy and convergence speed of the algorithm. However, path planning often needs to deal with complex and changing environments, and the neural network algorithm has poor generalization ability, making it less effective in such scenarios.

Among heuristic path planning algorithms, ABC [23] algorithm, GA [24,25] algorithm, SA [26] algorithm, Grey Wolf [27,28] algorithm, and PSO [29] algorithm are commonly used. However, the ABC algorithm and Grey Wolf algorithm have the problem of convergence speed being too fast or premature convergence, resulting in relatively low accuracy. The GA algorithm, SA algorithm, and PSO algorithm have the problem of slow convergence speed and being easily affected by parameters, which leads to their low efficiency. The Dijkstra algorithm [30], proposed by EW Dijkstra, solves the problem of finding the shortest distance between two points through a breadth-first search. However, with increasing grid density, the computational volume rapidly expands, making it difficult to meet real-time performance requirements. Local path planning algorithms, such as the artificial potential field (APF) algorithm, have been utilized in numerous scenarios, including mobile robots [31,32] and autonomous vehicles [33]. However, the traditional APF algorithm is susceptible to falling into local minima, which may prevent it from reaching the target point, and local oscillations. To address these issues, Liu and colleagues [34] replaced the repulsive potential field function in the APF algorithm with a Gaussian function. Nevertheless, the path planning of Unmanned Aerial Vehicles (UAVs) may still become trapped in local minima when the obstacle is situated on the connecting line between the current position and the target position. Moreover, the D\*Lite [35] algorithm, which is suitable for dynamic unknown environments and allows variable starting points but fixed target points, can result in non-smooth paths that are prone to colliding with obstacles, and the search process may also fail to converge, leading to the algorithm becoming stuck in a dead loop.

The A\* algorithm based on grid map search, originally proposed by Hart et al. [36], combines the strengths of mathematical and heuristic searches, using a heuristic function to guide the search direction, thus reducing search volume and improving efficiency. Several researchers have contributed to the improvement of the A\* algorithm in various ways. For example, Ju et al. [37] developed novel distance formulas for the A\* algorithm to obtain the shortest path under certain conditions, while Cao et al. [38] assigned a fixed weight to the heuristic function to reduce the number of node searches and shorten the search

time. Gochev proposed the anytime-tree-restoring-weighted A\* (ATRA\*) algorithm [39], which uses a heuristic approach to focus on the search scope and reuses previous search information to further reduce the search time. Huang et al. [40] improved the A\* algorithm's heuristic function by dynamically adjusting weights and optimizing the search and turning points to reduce data processing time and grid search. However, the four-directional search range reduces the possibility of obtaining the shortest path, and the path generated may not be smooth due to the large turning angles.

To address these limitations, Xin et al. [41] extended the search neighborhood of the A\* algorithm to an infinite number of neighbors, resulting in shorter path lengths with a smaller number of turning points, but at the expense of significantly increased search time. Chen et al. [42] improved the algorithm with directed search to reduce the number of traversed nodes by eliminating the path of symmetric search. Kong et al. [43] introduced the bidirectional search mechanism, improved the traditional calculation method of the evaluation function, and selected the appropriate weight coefficient for the evaluation function to reduce redundant nodes in the search process and improve computational efficiency. Lin et al. [44] optimized the path by analyzing the influence of the parent node of the current node on the heuristic function and using optimal weights to propose an improved A\* algorithm to improve computational efficiency, albeit at the expense of the optimal path. Islam et al. [45] proposed a dynamic multi-heuristic A\* (DMHA\*) algorithm to solve the problem of wasting a lot of search time around local minima, while Fu et al. [46] proposed an improved A\* algorithm that judges the presence of obstacles between the current path point and the target point to shorten the path. However, the computational cost of Fu's method is high, and it cannot smooth the turning path. Zhou et al. [47] adopted a hybrid-state A\* kinodynamic path searching method to find a safe and minimum-time initial trajectory in the discretized control space and to improve the smoothness and clearance of the trajectory by a B-spline optimization. Wu et al. [48] introduced a rewiring process to remove redundant points in the path, which in turn re-corrected the path and optimized the path length, but there were still invalid turning points. Muñoz et al. [49] assigned weights to each pair of vertices in the path generated by the A\* algorithm to develop a pruning strategy, which removed invalid waypoints from the path and achieved path smoothing, creating a better navigation experience for the user. Liu et al. [50] designed a dynamic fusion pathfinding algorithm (DFPA) based on Delaunay triangulation and improved A\* algorithm to implement path planning for mobile robots.

In summary, the A\* algorithm, which incorporates heuristic intelligent search, exhibits good path optimality and search efficiency and is more suitable for static map scenarios. However, due to the special characteristics of plantation forests (such as the unstructured and complex inter-forest environment and the high density of tree trunks and branches obstruction), the path planned by A\* algorithms may increase the UAV's flight energy consumption and affect flight safety. In addition, current improvements to the A\* algorithm often sacrifice the performance of other indicators to achieve the desired indicator performance in specific scenarios. To address these challenges, this paper proposes an improved algorithm that enhances the autonomous flying capability of UAVs in forest environments and reduces flight energy consumption by developing cost evaluation functions, heuristic functions, turning point removal strategies, and smooth optimization.

In addition, the innovation and main contributions of this paper are as follows:

1. This paper proposes a segmented cost evaluation function with parameterized weight factors ( $a$  and  $b$ ) assigned to actual cost and heuristic cost in each stage, where the weight factor of the heuristic cost function is adaptively and dynamically adjusted. The optimal parameter combinations under different environmental conditions are obtained to enhance the global search capability for the fast and accurate acquisition of optimal paths.
2. The algorithm incorporates turn cost with UAV heading angle constraints into the heuristic function to enhance heuristic search, reducing unnecessary turning points

and sharp turns, making the planning path smoother, and guiding the UAV towards the target direction.

3. The paper proposes a strategy for removing redundant turning points by reselecting path nodes on path segments composed of turning points, avoiding the process of turning point removal from falling into local optima while achieving a quadratic programming process on the path, allowing for a more significant reduction of turning points in the shortest path.

The rest of the paper is organized as follows. Section 2 presents a comprehensive account of the proposed enhancements to the traditional A\* algorithm, along with the experimental design details. The comparative simulation results of various algorithms are illustrated in Section 3. The experimental results are discussed in Section 4. Finally, the conclusions are drawn in Section 5.

## 2. Materials and Methods

In this section, a segmented cost evaluation function is proposed to optimize the global search. The steering cost, constrained by the heading angle, is introduced into the heuristic function to enhance the heuristic capabilities. The cost functions are endowed with parametrical weight factors (the weight factor of the heuristic function is set as an adaptive dynamic adjustment), and the optimal parameter combination is explored to improve the global search ability. In the operation of removing redundant turning points and smoothing the path, a new turning point removal strategy and path smoothing optimization scheme are proposed.

The performance simulation experiments were designed to prove the improved A\* algorithm's superiority over the traditional A\* algorithm. According to previous work [51], comparison experiments based on a 3D point cloud map of a plantation forest were carried out using data gathered by a mobile measurement platform constructed with a Velodyne VLP-16 LiDAR system and an inertial measurement unit (IMU) to test the various algorithms in this article in order to confirm the effectiveness of the upgraded A\* algorithm's path planning functionality.

### 2.1. Traditional A\* Algorithm

The A\* algorithm is a grid-based optimal path search algorithm that can be combined with the Dijkstra algorithm and Best-First-Search algorithm (BFS) [52]. Its core idea is to plan a collision-free minimum movement cost path from the starting position to the target position on a given map. It calculates the priority of each node through the cost function represented by Equation (1):

$$f(n) = g(n) + h(n) \quad (1)$$

where  $f(n)$  is the total cost estimate from the initial state via state  $n$  to the target state, and the higher the value, the lower priority;  $g(n)$  is the actual cost from the initial state to state  $n$ ; and  $h(n)$  is the estimated cost of the optimal path from state  $n$  to the target state, also known as the heuristic cost of A\* algorithm.

### 2.2. Improved A\* Algorithm

The heuristic search of the traditional A\* algorithm is relatively simplistic, and the weight relationship between cost functions is not taken into account, which leads to a higher number of nodes being searched and reduced efficiency. Additionally, to ensure the shortest path, the algorithm has more turning points, resulting in a path that is not smooth [40]. In this paper, to meet the requirements of high efficiency, low energy consumption, smoothness, and continuity for forestry inspection UAVs, an improved A\* obstacle avoidance algorithm is proposed. The algorithm improves the traditional A\* algorithm in the following four steps.

### 2.2.1. Segmented Evaluation Function with Dynamic Heuristics and Weighted Processing

In the A\* algorithm, both the cost function and the heuristic function are typically composed of distances. However, the value of the heuristic function can significantly impact the algorithm's performance. If the estimated value of  $h(n)$  is much smaller than  $g(n)$ , then the value of  $f(n)$  depends on  $g(n)$ , and the A\* algorithm behaves similarly to the Dijkstra algorithm, which conducts an extensive search with a slower convergence speed. Conversely, if the estimated value of  $h(n)$  is much larger than  $g(n)$ , then the A\* algorithm evolves into the Best-First-Search algorithm (BFS), which can quickly find a path, but the path may not be optimal for a UAV. Therefore, the A\* algorithm can balance the rate of convergence and path quality by assigning weights to  $g(n)$  and  $h(n)$  and adjusting them dynamically.

Additionally, assuming that the actual cost of the shortest distance from the current point to the target point is  $d(n)$ , three situations may arise. When  $h(n) < d(n)$ , the algorithm may need to search more nodes, leading to lower efficiency, but the shortest path can be found. When  $h(n) > d(n)$ , the number of search nodes decreases, and the search efficiency increases, but the shortest path cannot be guaranteed. When  $h(n) = d(n)$ , the shortest path can be searched with high efficiency. Based on these considerations, different weights are assigned to  $g(n)$  and  $h(n)$ , respectively, with the weight coefficient of  $h(n)$  set as the dynamic response. If the current node is far away from the target position, the weight coefficient of  $h(n)$  is larger. As the UAV approaches the target point, the weight coefficient of  $h(n)$  gradually decreases. Therefore, a preliminary evaluation function can be obtained.

$$f(n) = ag(n) + \left(b + \frac{d}{L}\right)h(n) \quad (2)$$

where  $a$  and  $b$  are the weight factors,  $d$  is the distance from the current node to the target position, and  $L$  is the distance between the starting position and the target position.

To balance search speed and path quality, this study adopts a segmented evaluation function that assigns different weights to the cost function and heuristic function in different stages. This approach enhances heuristics to speed up search without compromising path quality. The evaluation function of the algorithm is calculated as follows:

$$f(n) = \begin{cases} ag(n) + \left(b + \frac{d}{L}\right)h(n), & h(n) \geq \frac{L}{3} \\ g(n) + \left(\frac{d}{L}\right)h(n), & h(n) < \frac{L}{3} \end{cases} \quad (3)$$

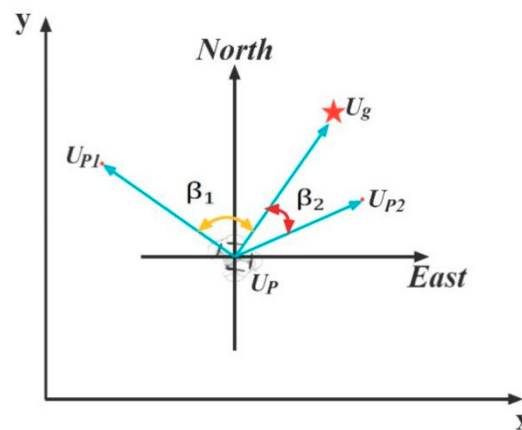
According to the above discussion on the weight relationship between  $g(n)$  and  $h(n)$ , the factors of  $a$  and  $b$  in the second segment of the evaluation function to 1 and 0, respectively, in order to maintain the accuracy of the later path search. To determine the appropriate values of weight factors  $a$  and  $b$  for  $g(n)$  and  $h(n)$  in the first segment of the evaluation function, the experiments were conducted using a randomly distributed grid map of size  $30 \times 30$  with an obstacle proportion of 0.4. The experimental results are presented in Section 3.1.

### 2.2.2. Heuristic Function for Adding Steering Cost

The heuristic function utilized in the conventional A\* algorithm employs a single distance approach for node selection, which can result in an increased number of turning points and subsequently impact the UAV's flight continuity and smoothness. To address this issue, a corner constraint is introduced during each node selection to decrease the number of points with large angles. Additionally, excess corners are corrected to avoid local optima. Therefore, this paper introduces the difference between the current heading angle of UAVs and the target direction angle and establishes the heuristic function containing steering cost under different angle differences.

While the theoretical shortest distance between the current node and the target point is a straight line, obstacles will exist during the actual flight. As such, the line linking the

current node to the next node will form a particular angle  $\beta \in [0, \pi]$  with the line linking the current node to the target point. The degree of path fitting is enhanced as  $\beta$  decreases. In the global coordinate system, the vector  $\overrightarrow{U_P U_g}$  denotes the direction from the current position to the target position, while vector  $\overrightarrow{U_P U_{pi}}$  ( $i = 1, 2, 3, 4 \dots$ ) denotes the possible heading of the UAVs toward the next location. Figure 1 provides a visual representation of this concept, and the red star represents the target point.



**Figure 1.** The vector representation from the current point to the travel point and the target point.

The angle  $\beta$  can be calculated using the formula for finding the angle between two vectors in the global coordinate system.

$$\beta_i = \cos^{-1} \frac{\overrightarrow{U_P U_g} \cdot \overrightarrow{U_P U_{pi}}}{\left| \overrightarrow{U_P U_g} \right| \left| \overrightarrow{U_P U_{pi}} \right|}, i = 1, 2, 3, 4 \dots \quad (4)$$

where  $\overrightarrow{U_P U_g} = (x_g - x_p, y_g - y_p)$ ;  $\overrightarrow{U_P U_{pi}} = (x_{pi} - x_p, y_{pi} - y_p)$ , and  $x_g, y_g, x_p, y_p, x_{pi}, y_{pi}$  are the coordinate values of the target point  $U_g$ , the current point  $U_p$ , and the next possible node  $U_{pi}$  in the global coordinate system.

Equation (5) shows the improved heuristic function of steering cost.

$$h(\beta) = w \times \beta \times h_d \quad (5)$$

where  $h(\beta)$  is the heuristic function of steering cost,  $h_d$  is the initial distance-based heuristic function, and  $w$  is the weight coefficient, whose value is taken to adjust the strength of the heuristic information and facilitate the calculation and selection of the optimal path, mainly determined by the grid size and the steering cost function.

The equation of the improved heuristic function is obtained as follows:

$$h(n) = (1 + k \times \beta) \times h_d, \beta \in [0, \pi] \quad (6)$$

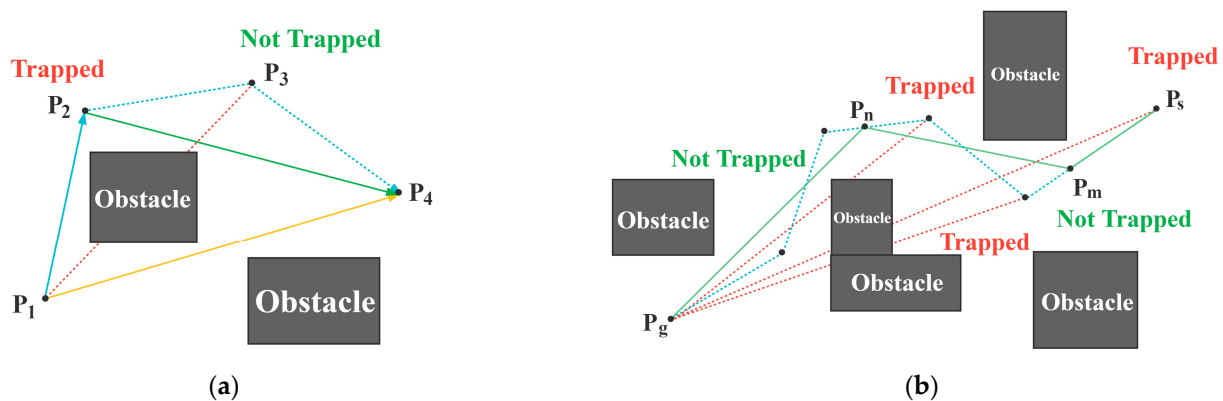
Based on the difference between the current heading direction of the UAV and the target position, the steering cost proportion is adjusted accordingly to minimize each steering angle and approach the target direction, thus ensuring global or relatively optimal path planning.

### 2.2.3. Removing Redundant Turning Points

As previously discussed, the traditional A\* algorithm tends to generate paths with multiple turning points in order to achieve the shortest path. However, the excessive

number of turns during UAV forest inspection not only leads to unstable flight but also consumes more energy.

Currently, most redundant turning point removal strategies adopt the approach illustrated in Figure 2a, where  $P_1$ ,  $P_2$ ,  $P_3$ , and  $P_4$  represent the initial turning points of the generated path. According to the rule of redundant turning point removal, node  $P_1$  should be connected to turning point  $P_3$ . After collision detection, if the path segment formed passes through an obstacle, turning point  $P_2$  is retained as the initial node of the new path segment and connected to node  $P_4$ . As the path segment formed by  $P_2P_4$  does not collide with any obstacles, the redundant turning point  $P_3$  can be removed. Finally, a new path  $P_1 \rightarrow P_2 \rightarrow P_4$  is obtained. However, the actual optimal path would be a direct connection between nodes  $P_1$  and  $P_4$ .



**Figure 2.** (a) Schematic diagram of local optimum; and (b) strategies for removing redundant turning points.

To address this local optimum, as shown in Figure 2b, the target node  $P_g$  is connected directly to the initial node  $P_s$ . If the newly formed path intersects with an obstacle, the target node is connected to the initial node's sub-node (not necessarily a turning point in the path), and collision detection is performed until an obstacle-free path segment connecting  $P_g$  to  $P_n$  (a node in the middle) is obtained. The nodes, including turning points between  $P_g$  and  $P_n$  are then removed, and the node  $P_n$  is used as the starting point of the next path segment to repeat the above operation until it is connected to  $P_m$  (a node in the path close to the initial node, but not necessarily a turning point). Then the node  $P_m$  is used as the new starting point of the next path segment to repeat the above steps until it is connected to the initial node  $P_s$ . This brief introduction outlines the process of removing turning points. In actual applications, it may require multiple iterations of steps to obtain an optimized path with redundant turning points removed.

The above comparison indicates that the rewiring strategy proposed in this paper allows for a broader selection of nodes when rewiring the path, which is not limited to turning points, resulting in a shorter path with fewer turns. Moreover, our pruning strategy for removing turning points is forward-backward, which enables efficient removal with minimal collision detection and effectively avoids local optima.

#### 2.2.4. Smoothing Based on Quasi-Uniform Cubic B-Spline Curves

After removing the turning points, the smoothness of the path will be improved, and the number of turning points, total turn angle, and path length will be reduced, but the path is not yet better smooth due to the presence of some necessary corners.

B-spline is a special manifestation of the spline curve, which can flexibly smooth the path by taking advantage of its merits, such as convex hull, geometric invariance, local support, and non-negativity [47] to better adapt to various actual trajectory requirements.

To ensure that the optimized path encompasses both the starting and target points while maintaining continuous smoothness, this paper utilizes quasi-uniform cubic B-spline

curves to smooth the path. However, in practical applications, excessive smoothness may lead to collisions with obstacles. To mitigate this issue, the search range of control points is constrained, ensuring that the path after quasi-uniform cubic B-spline curve processing remains close to the path after removing redundant turning points. To achieve this, the paper sets the search radius of the control points accordingly.

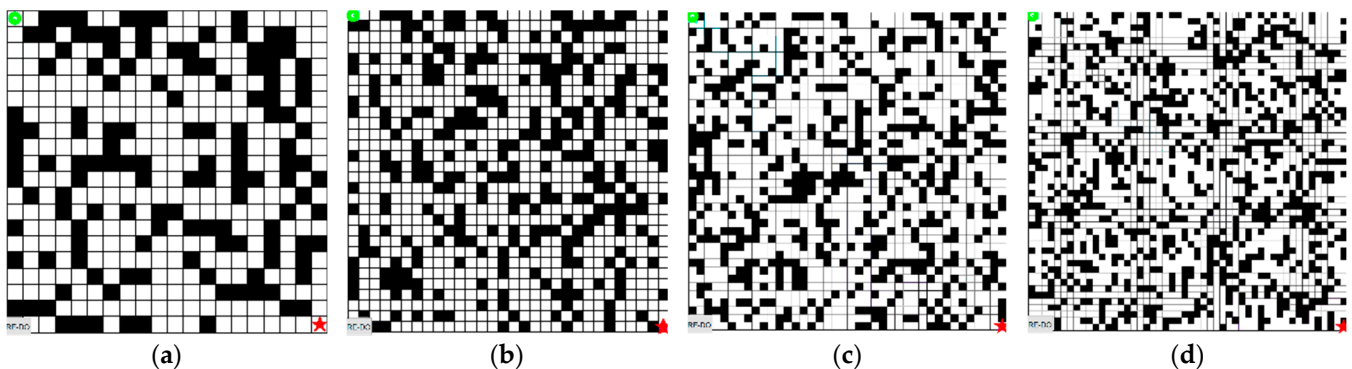
$$R \leq 0.6r \quad (7)$$

where  $R$  is the search radius of the control points and  $r$  is the grid grain size.

### 2.3. Improved A\* Algorithm Performance Simulation

As pointed out above, the enhanced A\* obstacle avoidance algorithm represents an improvement over the conventional A\* algorithm. This is achieved through the use of a segmented cost function that takes into account weighted processing, as well as the incorporation of a dynamic heuristic function that considers steering cost, turning points optimization, and smoothing optimization.

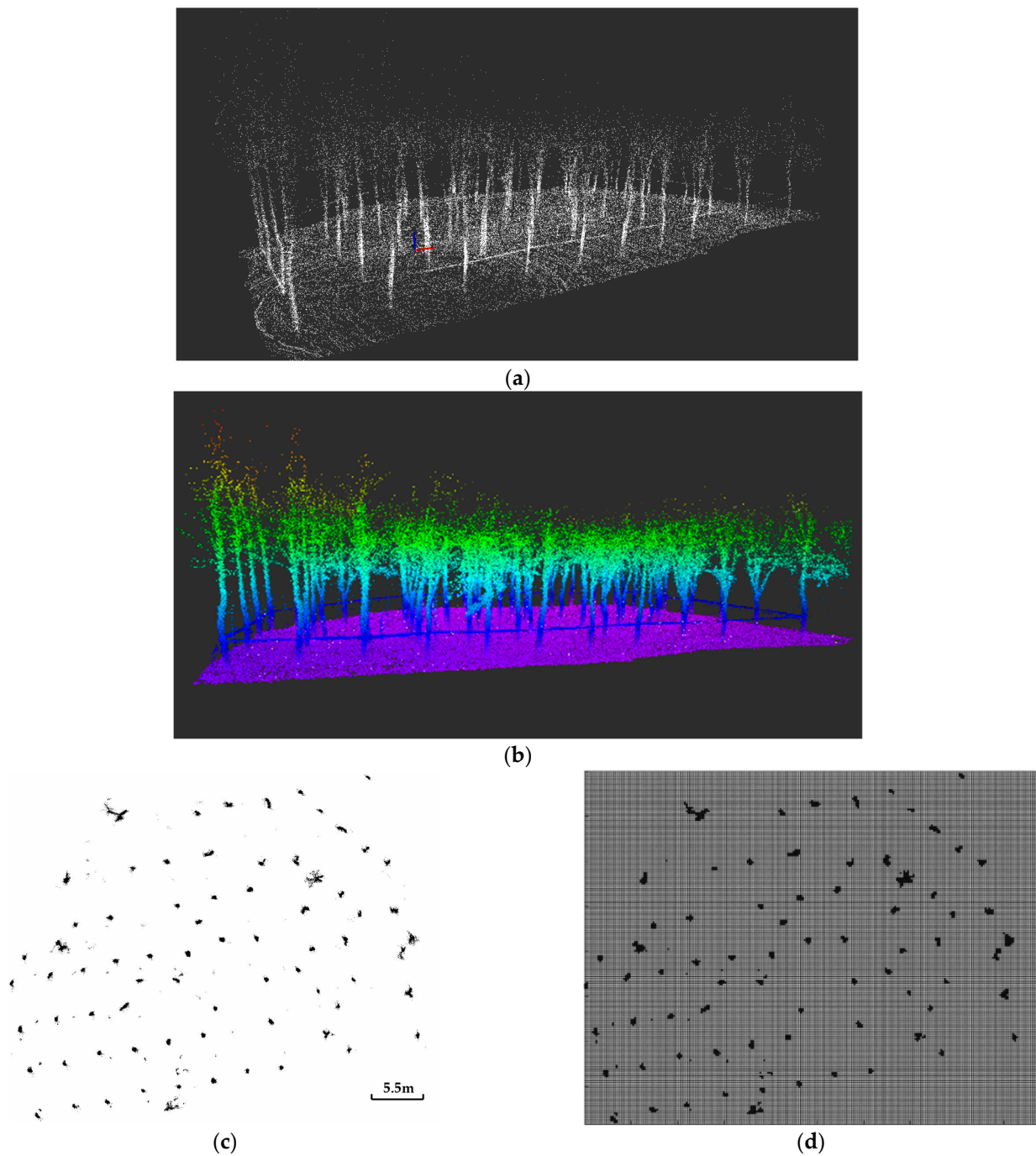
To evaluate the superiority of the improved A\* algorithm, the simulation experiments were conducted using MATLAB R2022a on an Intel (R) Core (TM) i5-7500 CPU @ 3.40 GHz computer. The four simulation environments were created with a fixed obstacle proportion of 40% but a varying number of grids and obstacle distributions. The scene layout and the locations of the starting and target points are depicted in Figure 3. And the green circle represents the starting point and the red star represents the target point. The paper compared the obstacle avoidance performance of the improved A\* algorithm against the traditional A\* algorithm using various evaluation metrics, including data processing time, number of traversal nodes, number of turning points, total turning angle, and path length.



**Figure 3.** Obstacle environment maps with different numbers of grids: (a) grid size  $20 \times 20$ , start point coordinates (1, 20), end point coordinates (20, 1); (b) grid size  $30 \times 30$ , start point coordinates (1, 30), end point coordinates (30, 1); (c) grid size  $40 \times 40$ , start point coordinates (1, 40), end point coordinates (40, 1); and (d) grid size  $50 \times 50$ , start point coordinates (1, 50), end point coordinates (50, 1).

### 2.4. Comparison Experiments Based on 3D Point Cloud Map of Plantation Forest

The present study conducted a comparative experiment between the improved A\* algorithm and the traditional A\* algorithm in various obstacle environments to validate the superiority of the former. However, this experiment alone is insufficient since the actual environment encountered by UAVs during flight within a forest may differ from the map created in a random setting. To address this limitation, this study utilized a real 3D point cloud map of a plantation forest to compare the path planning effectiveness of different algorithms, including the traditional A\* algorithm, the improved A\* algorithm proposed in the literature [38], the RRT algorithm, and the APF algorithm. Figure 4 illustrates the environment map information and processing flow of the plantation forest map.



**Figure 4.** Acquisition and processing process of actual plantation forest map: (a) three-dimensional point cloud map of the plantation forest; (b) Octomap of the plantation forest; (c) grid map of the two-dimensional projection surface of the plantation forest; and (d) two-dimensional projection map of the plantation forest after re-rasterization.

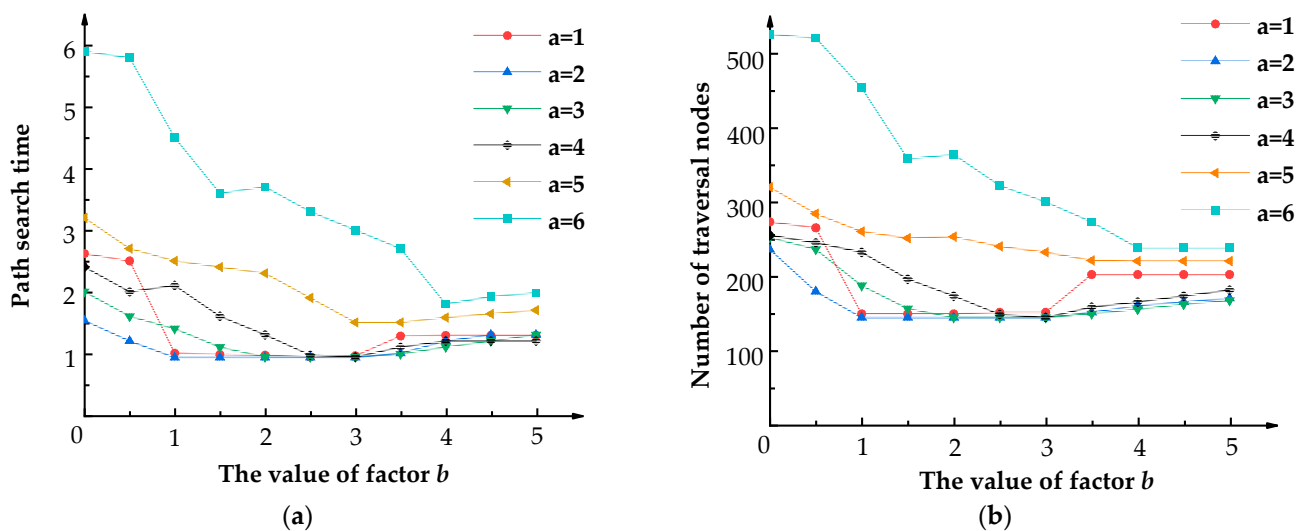
As shown in Figure 4a, the point cloud data was obtained from a handheld 3D scanning device of the plantation forest located in Bajia Park, Haidian District, Beijing. The test area was selected using CloudCompare software, and the PassThrough filtering method and RadiusOutlierRemoval were applied to remove the background noise and outliers, respectively [51]. Since the point cloud data was relatively sparse, voxelization was performed to transform the point clouds into an Octomap, which is a three-dimensional map creation tool based on octree, and the plantation forest's 3D reconstruction was completed in ROS [53], as shown in Figure 4b. Subsequently, the two-dimensional grid

map was obtained by projecting the plane of the plantation forest where the UAV flight height was located, as shown in Figure 4c. This map was discretized into a large number of pixel grids composed of black and white grids, with black representing obstacles and white representing passable areas. The pixel size of the map is  $974 \times 785$ . In computer science, a binary matrix is usually used to represent this grid map, with the position of each matrix element representing the grid position, the element value of 1 indicating that the position is occupied by obstacles, and 0 representing the unoccupied area. Due to the large number of grids in the obtained two-dimensional map, directly importing it into MATLAB 2022a for use would consume a considerable amount of storage, and path search would require significant computational time. Therefore, on the premise of preserving the location, shape, and size of obstacles in the original map, simple image processing was performed to obtain a real map with fewer grids. As illustrated in Figure 4d, the black outline of the tree trunk intercepted in this plane is considered an obstacle. The two-dimensional map of the plantation forest after re-rasterization was composed of  $246 \times 200$  grids, based on the image pixel ratio of the two-dimensional grid map. By setting different start and end coordinates, the performance comparison results of the five algorithms in different complex scenes were obtained.

### 3. Results

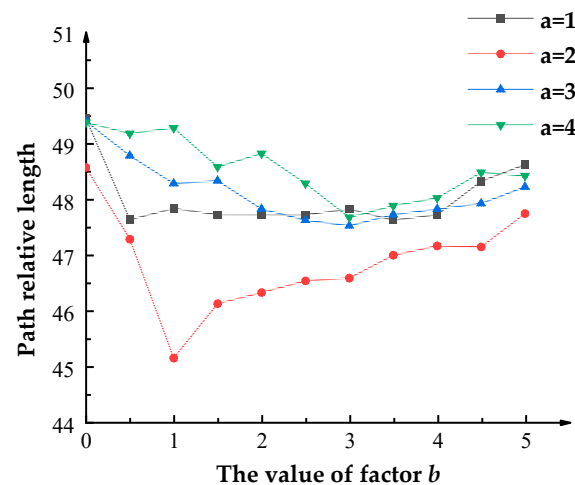
#### 3.1. Segmented Evaluation Function Parameter Determination Results

Experimental designs were conducted to assign different values to parameters  $a$  and  $b$  in order to obtain the relevant index results. Figure 5 presents the number of nodes traversed and the search time of the algorithm when the weight factors  $a$  and  $b$  take various values. It can be observed that when factor  $a$  was set to five or six, the number of nodes traversed and the path search time of the algorithm was found to be the highest. This is primarily due to the fact that, at this stage, the algorithm is more inclined toward the Dijkstra algorithm, resulting in weaker heuristic information.



**Figure 5.** Results of evaluation indexes with different parameters: (a) path search time with different parameters; and (b) number of traversal nodes with different parameters.

The optimal parameter combination obtained in the initial experiment was adopted for the second test, where different obstacle maps were utilized, and the starting and ending positions were varied. The relative lengths of the resulting paths are presented in Figure 6.



**Figure 6.** Relative length of paths with different parameters.

The results depicted in Figure 6 indicate that when factor  $a$  is set to two, the path length obtained is the shortest among all tested values, with an even shorter path length observed when  $b$  is in the range of [1, 3]. Additionally, Table 1 shows the corresponding path search time and the number of traversed nodes within this interval.

**Table 1.** Path search time and number of traversed nodes corresponding to different  $b$  values.

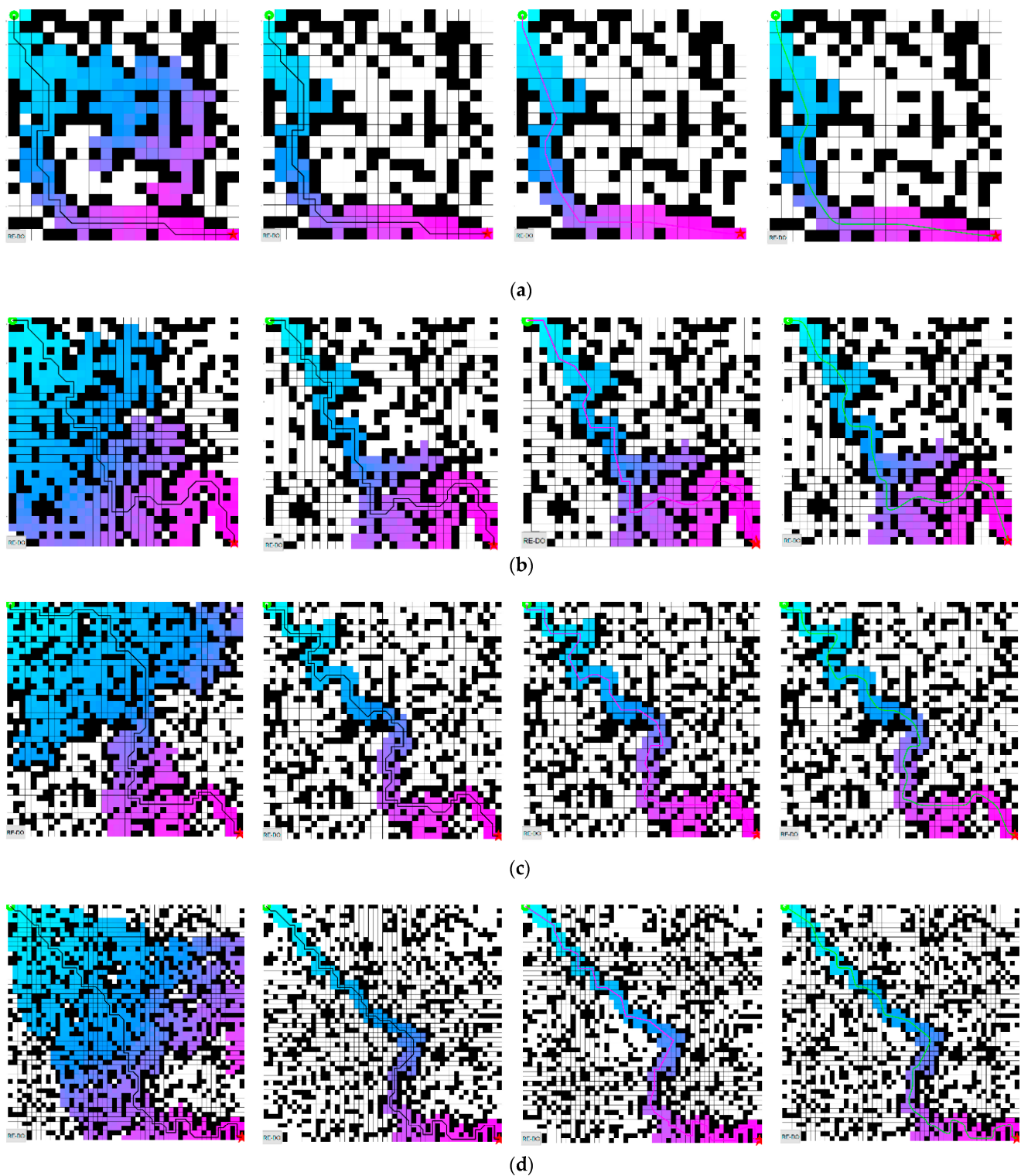
| Parameter $b$ | Path Search Time(s) | Number of Nodes Traversed |
|---------------|---------------------|---------------------------|
| 1             | 0.714               | 144                       |
| 1.5           | 0.709               | 144                       |
| 2             | 0.713               | 142                       |
| 2.5           | 0.732               | 141                       |
| 3             | 0.775               | 141                       |

Based on the results shown in Table 1, the differences in path search time and the number of traversed nodes are found to be relatively small when factor  $a$  takes the value of two, and factor  $b$  falls in the range of [1, 3]. Figure 6 illustrates that when  $a$  is set to two and  $b$  is one, the shortest paths are obtained in different obstacle maps, and the optimal results are achieved when compared. Therefore, this study ultimately determines the values of factors  $a$  and  $b$  to be two and one, respectively, to obtain the final evaluation function:

$$f(n) = \begin{cases} 2g(n) + \left(1 + \frac{d}{L}\right)h(n), & h(n) \geq \frac{L}{3} \\ g(n) + \left(\frac{d}{L}\right)h(n), & h(n) < \frac{L}{3} \end{cases} \quad (8)$$

### 3.2. Performance Simulation Experiment Results

This study employed Matlab to build four simulation environments to test the performance of the improved A\* algorithm proposed in Section 2.2. The evaluation metrics included data processing time, number of traversal nodes, number of turning points, total turning angle, and path length. In Section 3.1, the weight parameter values in the evaluation function of the improved A\* algorithm were determined. Based on the simulation experiments, the path planning trajectories of the traditional A\* algorithm and the improved A\* algorithm in different environments were obtained and are shown in Figure 7.



**Figure 7.** Path planning trajectories for different environmental maps and optimization steps. (a) Environments 1; (b) Environments 2; (c) Environments 3; (d) Environments 4. Note: In the above four groups of graphs, from left to right, each group shows the path planning effect of traditional A\* algorithm, add Optimization 1, add Optimization 1 and Optimization 2, add Optimization 1 and Optimization 2 and Optimization 3. And the green circle represents the starting point and the red star represents the target point.

It can be seen from Figure 7 that both the traditional A\* algorithm and the optimized A\* algorithm can achieve effective obstacle avoidance. Nevertheless, the traditional A\* algorithm requires more grid traversals during pathfinding, resulting in longer pathfinding times. After optimization 1, the algorithm became more targeted in its path search, resulting in a substantial reduction in the number of grid searches and data processing time. Optimization 2, built on the foundation of optimization 1, eliminated all redundant turning points, significantly reducing the total number of turning points and total turning angle while still maintaining the necessary turning points for obstacle avoidance. Subsequently, optimization 3 was added to smooth the path using quasi-uniform cubic B-spline curves based on optimizations 1 and 2. The broken lines at the turning points were replaced by smooth curves, resulting in a path with no distinct turning points. The total turning angle was further reduced, and the corresponding path length was also shortened.

The performance evaluation index results of the algorithms for different environments and optimization steps are shown in Table 2.

**Table 2.** Performance of the algorithms with different environments and optimization steps.

| Environment  | Optimization Steps and Methods | Data Processing Time (s) | Number of Nodes Traversed | Number of Turning Points | Path Length | Total Turning Angle |
|--------------|--------------------------------|--------------------------|---------------------------|--------------------------|-------------|---------------------|
| 20 × 20 Grid | Traditional A*                 | 1.52                     | 161                       | 14                       | 36.49       | 810°                |
|              | Optimization 1                 | 0.49                     | 75                        | 14                       | 36.49       | 810°                |
|              | Optimization 1 + 2             | 0.53                     | 75                        | 8                        | 31.62       | 190.50°             |
|              | Optimization 1 + 2 + 3         | 0.54                     | 75                        | 0                        | 30.58       | 187.84°             |
| 30 × 30 Grid | Traditional A*                 | 3.20                     | 456                       | 35                       | 62.38       | 2070°               |
|              | Optimization 1                 | 1.06                     | 195                       | 35                       | 62.38       | 2070°               |
|              | Optimization 1 + 2             | 1.42                     | 195                       | 20                       | 55.55       | 975.95°             |
|              | Optimization 1 + 2 + 3         | 1.43                     | 195                       | 0                        | 54.01       | 895.78°             |
| 40 × 40 Grid | Traditional A*                 | 4.77                     | 719                       | 33                       | 84.38       | 2250°               |
|              | Optimization 1                 | 1.01                     | 218                       | 44                       | 86.48       | 2870°               |
|              | Optimization 1 + 2             | 2.97                     | 218                       | 29                       | 79.18       | 1475.90°            |
|              | Optimization 1 + 2 + 3         | 2.99                     | 218                       | 0                        | 75.13       | 1424.28°            |
| 50 × 50 Grid | Traditional A*                 | 7.77                     | 1166                      | 48                       | 100.70      | 2925°               |
|              | Optimization 1                 | 1.43                     | 243                       | 50                       | 102.18      | 2925°               |
|              | Optimization 1 + 2             | 4.53                     | 243                       | 33                       | 97.25       | 1755.03°            |
|              | Optimization 1 + 2 + 3         | 4.55                     | 243                       | 0                        | 91.14       | 1525.22°            |

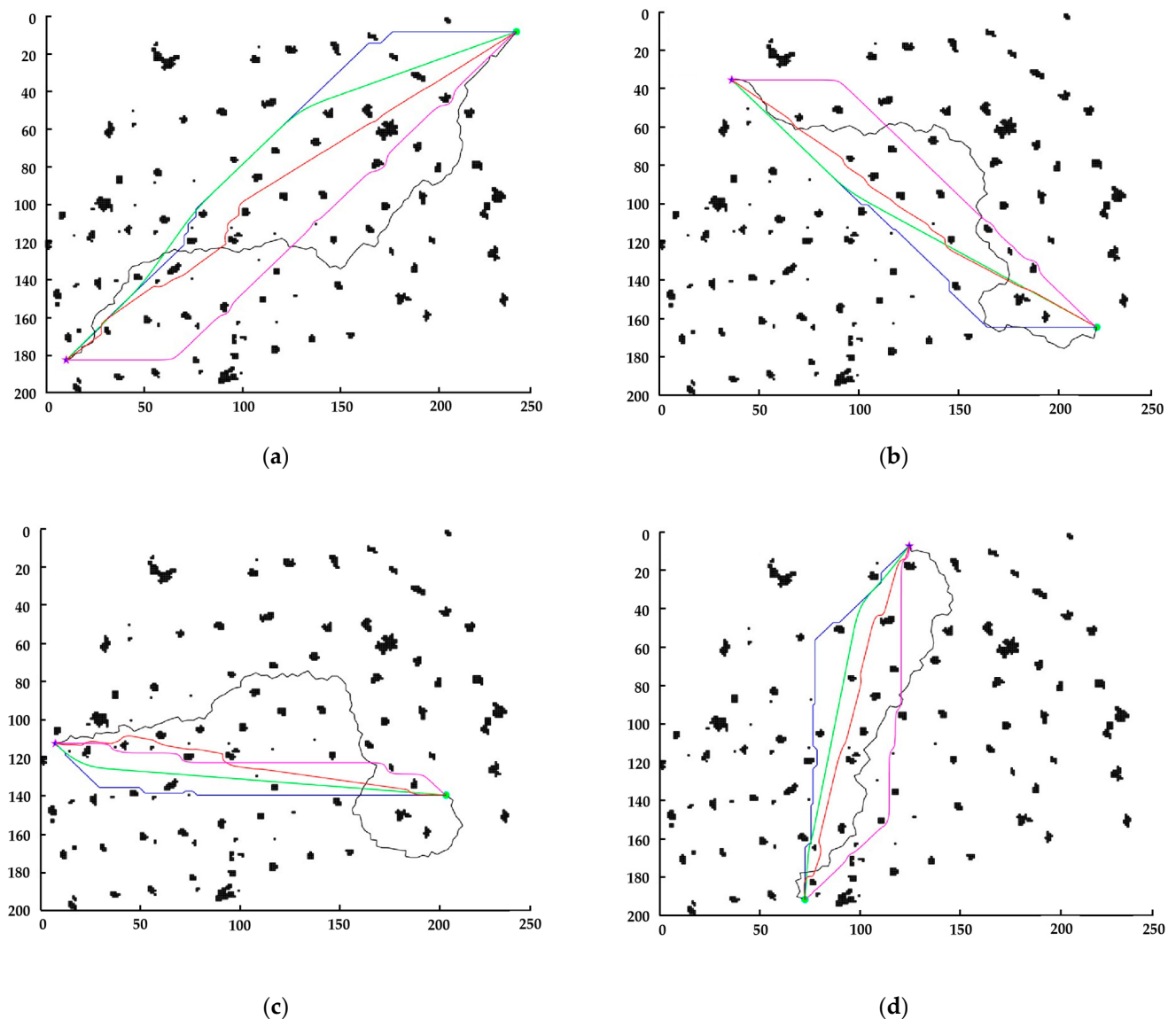
Note: Where optimization 1 is the segmented evaluation function with dynamic heuristics and weighted processing, Optimization 2 is the redundant turning point removal strategy, and optimization 3 is the smoothing optimization of quasi-uniform cubic B-sample curves.

Further analysis of the data presented in Table 2 reveals that optimization 1 results in a decrease of 64.87% in the average number of traversed nodes and a reduction of 73.77% in the data processing time. On the basis of optimization 1, optimization 2 further reduces the number of newly generated path turns by an average of 32.27%, path length by an average of 8.48%, total turn angle by an average of 50.83%, and data processing time by an average of 50.05%. By adding optimization 3 to the paths generated by optimization 1 and optimization 2, the final performance of the improved A\* algorithm is achieved, which demonstrates an average reduction of 64.87% in the number of traversed nodes, an average reduction of 49.64% in data processing time, and a shorter and smoother path with no significant turning point. The average reduction in path length and total turning angle is 12.52% and 54.53%, respectively.

The simulation results show that the improved A\* algorithm proposed in this paper significantly reduces the data processing time, the number of traversed nodes, the number of turning points, and the total turning angle of obstacle avoidance, resulting in a shorter and smoother path, which is better suited for efficient, smooth and continuous flights of UAVs in plantation forests.

### 3.3. Comparison Experiments Results Based on 3D Point Cloud Map of Plantation Forest

To validate the effectiveness of the improved A\* algorithm in practical plantation forest environments, it was compared with the traditional A\* algorithm, an improved A\* algorithm proposed in the literature [38], the RRT algorithm, and the APF algorithm. The evaluation criteria were based on the path search time, path length, and total turning angle of the generated path. Figure 8 depicts the trajectories planned by the four algorithms under different starting and ending points, where the blue five-pointed star indicates the starting point, and the green circle represents the target position. Moreover, the coordinates presented in the figure are the grid coordinates of the map after re-rasterization, and each grid was set to  $1 \times 1$  in size. The results of evaluation metrics corresponding to different algorithms are shown in Table 3.



**Figure 8.** Path planning trajectory with different starting and target point positions. (a) start point coordinates (10, 182), end point coordinates (240, 8); (b) start point coordinates (38, 35), end point coordinates (220, 162); (c) start point coordinates (8, 112), end point coordinates (206, 138); (d) start point coordinates (72, 194), end point coordinates (125, 8); Note: Where blue path trajectory represents traditional A\* algorithm, green represents the improved A\* algorithm in this paper, purple represents the improved A\* algorithm in literature [38], red represents the APF algorithm, and black represents the RRT algorithm.

**Table 3.** Algorithm performance for different starting points and target point positions.

| Environment | Algorithm            | Path Search Time (s) | Path Length | Total Turning Angle |
|-------------|----------------------|----------------------|-------------|---------------------|
| 1           | Traditional A*       | 4.36                 | 312.10      | 495.00°             |
|             | RRT                  | 47.92                | 370.82      | 3816.09°            |
|             | APF                  | 78.64                | 299.26      | 1025.95°            |
|             | A* in reference [39] | 4.09                 | 304.32      | 482.46°             |
|             | Improved A*          | 3.49                 | 294.50      | 44.61°              |
| 2           | Traditional A*       | 2.24                 | 239.36      | 315.00°             |
|             | RRT                  | 52.20                | 339         | 3636.93°            |
|             | APF                  | 56.19                | 228.88      | 397.07°             |
|             | A* in reference [39] | 1.98                 | 237.43      | 157.93°             |
|             | Improved A*          | 1.35                 | 225.64      | 15.40°              |
| 3           | Traditional A*       | 0.48                 | 210.60      | 405.00°             |
|             | RRT                  | 35.80                | 360.00      | 4571.42°            |
|             | APF                  | 47.89                | 204.23      | 635.60°             |
|             | A* in reference [39] | 0.42                 | 206.32      | 508.93°             |
|             | Improved A*          | 0.31                 | 200.53      | 41.85°              |
| 4           | Traditional A*       | 3.23                 | 211.54      | 765°                |
|             | RRT                  | 24.92                | 258.00      | 3181.67°            |
|             | APF                  | 46.60                | 197.87      | 601.88°             |
|             | A* in reference [39] | 2.21                 | 204.99      | 589.46°             |
|             | Improved A*          | 1.97                 | 194.95      | 54.13°              |

The results in Table 3 indicate that the traditional A\* algorithm generates a path with a short path length and spends less time in the pathfinding process due to its inherent heuristic capabilities.

However, it generates more turning points and has a larger total turning angle, and the trajectory is often close to the edge of obstacles, which increases the risk of collision. Due to the lack of heuristic guidance, the RRT algorithm based on random sampling generates paths with a large number of nodes, which appear more tortuous, resulting in more time consumption during the search process, and the obtained path length and total turning angle are the largest. The APF algorithm generates paths with shorter lengths but takes the longest time in the pathfinding process. The improved A\* algorithm proposed in the literature [38] reduces the path search time and path length to some extent and can generally achieve a reduction in the total turning angle. However, the trajectory is sometimes close to the edge of obstacles. In contrast, the improved A\* algorithm proposed in this paper has the shortest path search time and generates paths with no significant turning points, the shortest length, and the smallest total turning angle. Importantly, it avoids the situation where the algorithm searches for a path close to the edge of an obstacle to obtain the shortest path. Specifically, compared with the traditional A\* algorithm, the improved A\* algorithm reduces the path search time by an average of 33.53%, the path length by an average of 6.00%, and the total turning angle by an average of 92.17%. Compared with the improved A\* algorithm proposed in the literature [38], the improved A\* algorithm reduces the path search time by an average of 20.89%, the path length by an average of 3.98%, and the total turning angle by an average of 90.90%. Compared with the RRT algorithm, the improved A\* algorithm reduces the path search time by an average of 95.34%, the path length by an average of 30.69%, and the total turning angle by an average of 98.95%. Compared with the APF algorithm, the improved A\* algorithm reduces the path search time by an average of 97.07%, the path length by an average of 1.58%, and the total turning angle by an average of 94.05%.

#### 4. Discussion

In order to enhance the safety of UAV autonomous obstacle avoidance and reduce unnecessary flight energy consumption in the plantation forest. Combined with the op-

erating environment and flight requirements of UAVs, the traditional A\* algorithm has been improved through the segmented cost function of weighted processing with the dynamic heuristic function of steering cost, turning points optimization, and smoothing optimization.

Through the parameter determination experiment, the optimal combination of weight coefficients for the cost function was obtained as the basis for subsequent experiments. The performance simulation experiment results show that, compared with the traditional A\* algorithm with a unified weight, the average number of traversed nodes is reduced by 64.87%, and the data processing time is reduced by 73.77%, indicating that the combination of weight coefficients can effectively reduce the data processing time and the number of node searches, while achieving a better balance between the rate of convergence and the path quality. Furthermore, by introducing the strategy of removing turning points, the redundant nodes (including turning points) in the path were eliminated, further reducing the path length and total turning angle while improving the path's smoothness. Subsequently, the path with redundant turning points removed was smoothed using quasi-uniform cubic B-spline curves, with the fitting degree of the curves controlled to ensure the flight safety of UAVs by maintaining a distance range between the trajectory and obstacles. After smoothing, the final path did not exhibit obvious turning points, and the path length and total turning angle were further reduced. On average, the shortest path length is reduced by 12.52%, the total turning angle is reduced by 54.53%, and the total time spent is reduced by 49.64%, meeting the requirements of low energy consumption, high efficiency, and high safety for UAV flight in plantation forests.

The comparison experiments of various algorithms conducted in the plantation forest map revealed that, among the selected algorithms, the RRT algorithm yielded the longest path length and largest total turning angles and consumed a considerable amount of time, thus exhibiting lower overall efficiency. The APF algorithm, the improved A\* algorithm proposed in the literature [38], and the improved A\* algorithm proposed in this paper yielded shorter paths and generated smoother trajectories in comparison to the traditional A\* algorithm. However, the APF algorithm and the improved A\* algorithm proposed in the literature [38] exhibited collision risks as the path trajectory was closer to the edge of the obstacle. Moreover, the algorithm proposed in the literature [38] did not consider the effect of changing weights on path search, thus leading to a longer path and larger total turns. The APF algorithm took more search time, and its lack of global perception led to an increased probability of falling into a local optimum. In contrast, the improved A\* algorithm in this paper reduced the path length and search time while ensuring the safety of obstacle avoidance, significantly reducing the total turning angle, improving the smoothness and continuity of UAV flight, reducing energy consumption, and proving to be more suitable for the operating environment of plantation forests.

To summarize, this study aims to enhance the obstacle avoidance and endurance capability of UAVs in unstructured plantation forests. It may provide new insights for improving the safety and efficiency of UAV autonomous flight, which can broaden the range of applications for UAVs. In the future, we plan to optimize the obstacle avoidance algorithm further by combining it with other intelligent algorithms to plan optimal paths for UAVs in dynamic environments and extend its application to other scenarios.

## 5. Conclusions

This study presented an improved A\* obstacle avoidance algorithm for UAVs to avoid obstacles in unstructured plantation forests. Improved the traditional A\* obstacle avoidance algorithm through segmented cost function of weighted processing with the dynamic heuristic function of steering cost, turning points optimization, and smoothing optimization. In performance experiments, the improved algorithm outperforms the traditional A\* algorithm with a reduction of 64.87% in traversed nodes on average, 49.64% in search time on average, and 12.52% in path length on average. Additionally, the improved algorithm exhibits no obvious turning point and a 54.53% reduction in total turning angle

on average. In comparison with the RRT algorithm and APF algorithm, the improved algorithm significantly reduces path search time and total turning angle by at least 90%. Moreover, the path length is reduced by 30.69% on average compared with the RRT algorithm and 1.58% on average compared with the APF algorithm. Compared with the traditional A\* algorithm, the improved algorithm reduces the path search time by 33.53% on average, the path length by 6.00% on average, and the total turning angle by 92.17% on average. Compared with the improved A\* algorithm proposed in the literature [38], the path search time is reduced by 20.89% on average, the path length is reduced by 3.98% on average, and the total turning angle is reduced by 90.90% on average. Based on these results, the algorithm presented in this article can achieve a better balance between path search efficiency and trajectory quality while improving obstacle avoidance safety of UAVs and reducing energy consumption, providing a new solution for the autonomous flight of UAVs in plantation forest environments.

**Author Contributions:** Conceptualization, F.K., Y.W. and J.L.; methodology, C.C., S.T., C.Z. and J.L.; investigation, F.K. and Y.W.; resources, F.K., Y.J. and Y.W.; data curation, J.L.; writing—original draft preparation, J.L.; writing—review and editing, C.C., S.T. and J.L.; supervision, F.K. and Y.W.; funding acquisition, F.K. and Y.W. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Fundamental Research Funds for the Central Universities (No. BLX201820).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The raw data required to reproduce these findings cannot be shared at this time as the data also forms part of an ongoing study.

**Acknowledgments:** Thanks to the Key Lab of State Forestry and Grassland Administration on Forestry Equipment and Automation for providing the equipment for algorithm validation and data analysis.

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

## References

1. Yang, L.; Meng, F. Application analysis of multi-rotor UAV in forest fire prevention and control. *For. Sci. Technol. Inf.* **2020**, *52*, 105–107.
2. Wang, Z.; Zhou, Y.; Li, X.; Wu, Z.; Cen, B.; Zeng, Y. Application analysis of UAV in forest fire prevention. *For. Environ. Sci.* **2016**, *32*, 31–35.
3. Chiaraviglio, L.; D'Andreagiovanni, F.; Liu, W.; Gutierrez, J.; Blefari-Melazzi, N.; Choo, K.K.; Alouini, M.S. Multi-Area Throughput and Energy Optimization of UAV-aided Cellular Networks Powered by Solar Panels and Grid. *IEEE Trans. Mob. Comput.* **2021**, *20*, 2427–2444. [\[CrossRef\]](#)
4. Di Franco, C.; Buttazzo, G. Energy-aware coverage path planning of UAVs. In Proceedings of the International Conference on Autonomous Robot Systems and Competitions, Vila Real, Portugal, 8–10 April 2015; pp. 111–117. [\[CrossRef\]](#)
5. Ji, X.; Meng, X.; Wang, A.; Hua, Q.; Wang, F.; Chen, R.; Zhang, J.; Fang, D. E2PP: An Energy-Efficient Path Planning Method for UAV-Assisted Data Collection. *Secur. Commun. Netw.* **2020**, *2020*, 8850505:1–8850505:13. [\[CrossRef\]](#)
6. Kuwata, Y. *Real-Time Trajectory Design for Unmanned Aerial Vehicles Using Receding Horizon Control*; Institute of Technology: Cambridge, MA, USA, 2003.
7. Ye, W.; Ma, D.W.; Fan, H.D. Algorithm for low altitude penetration aircraft path planning with improved ant colony algorithm. *Chin. J. Aeronaut.* **2005**, *18*, 304–309. [\[CrossRef\]](#)
8. Chu, X.; Hu, Q.; Zhang, J. Path planning and collision avoidance for a multi-arm space maneuverable robot. *IEEE Trans. Aerosp. Electron. Syst.* **2018**, *54*, 217–232. [\[CrossRef\]](#)
9. Xi, F.; Zeng, X.; Ji, S.; Chen, G.; Cai, C. Path planning of mobile robot based on Improved-RRT algorithm. *Comput. Sci.* **2019**, *46*, 247–253. [\[CrossRef\]](#)
10. LaValle, S.; Kuffner, J. Rapidly-exploring random trees: Progress and prospects. *Algorithmic Comput. Robot. New Dir.* **2001**, *5*, 293–308. [\[CrossRef\]](#)

11. LaValle, S.; Kuffner, J. RRT-connect: An efficient approach to single-query path planning. In Proceedings of the 2000 ICRA. Millennium Conference, San Francisco, CA, USA, 24–28 April 2000; IEEE International Conference on Robotics and Automation, Symposia Proceedings (Cat. No. 00CH37065). pp. 995–1001. [\[CrossRef\]](#)
12. Jeong, I.; Lee, S.; Kim, J. Quick-RRT\*: Triangular inequality-based implementation of RRT\* with improved initial solution and convergence rate. *Expert Syst. Appl.* **2019**, *123*, 82–90. [\[CrossRef\]](#)
13. Gammell, J.; Srinivasa, S.; Barfoot, T. Informed RRT\*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Chicago, IL, USA, 14–18 September 2014. [\[CrossRef\]](#)
14. Gammell, J.; Srinivasa, S.; Barfoot, T. Batch informed trees (BIT\*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 26–30 May 2015. [\[CrossRef\]](#)
15. Yang, F.; Fang, X.; Gao, F.; Zhou, X.; Li, H.; Jin, H.; Song, Y. Obstacle Avoidance Path Planning for UAV Based on Improved RRT Algorithm. *Discret. Dyn. Nat. Soc.* **2022**, *2022*, 4544499. [\[CrossRef\]](#)
16. Li, B.; Qi, X.; Yu, B.; Liu, L. Trajectory Planning for UAV Based on Improved ACO Algorithm. *IEEE Access* **2020**, *8*, 2995–3006. [\[CrossRef\]](#)
17. Zhou, Y.; Su, Y.; Xie, A.; Kong, L. A newly bio-inspired path planning algorithm for autonomous obstacle avoidance of UAV. *Chin. J. Aeronaut.* **2021**, *34*, 199–209. [\[CrossRef\]](#)
18. Zhang, J.; Chadha, R.G.; Velivela, V.; Singh, S. P-CAP: Precomputed alternative paths to enable aggressive aerial maneuvers in cluttered environments. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018. [\[CrossRef\]](#)
19. Zhang, J.; Chadha, R.G.; Velivela, V.; Singh, S. P-CAL: Pre-computed alternative lanes for aggressive aerial collision avoidance. In Proceedings of the 12th International Conference on Field and Service Robotics (FSR), Tokyo, Japan, 29–31 August 2019.
20. Zhang, J.; Hu, C.; Chadha, R.G.; Singh, S. Maximum Likelihood Path Planning for Fast Aerial Maneuvers and Collision Avoidance. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macao, China, 3–8 November 2019; pp. 2805–2812. [\[CrossRef\]](#)
21. Fan, C.; Chen, W.; Xi, Y. Hopfield neural networks for path planning in dynamic and unknown environments. *Control Theory Appl.* **2004**, *21*, 345–350.
22. Gao, M.; Wei, P.; Liu, Y. Competitive Self-Organizing Neural Network Based UAV Path Planning. In Proceedings of the IEEE 6th International Conference on Computer and Communications (ICCC), Chengdu, China, 11–14 December 2020; pp. 2376–2381. [\[CrossRef\]](#)
23. Wei, B.; Yang, R.; Shu, S.; Wan, Y.; Miao, J. Path planning of mobile robots based on ion motion-artificial bee colony algorithm. *J. Comput. Appl.* **2021**, *41*, 379–383.
24. Pan, Y.; Yang, Y.; Li, W. A Deep Learning Trained by Genetic Algorithm to Improve the Efficiency of Path Planning for Data Collection with Multi-UAV. *IEEE Access* **2021**, *9*, 7994–8005. [\[CrossRef\]](#)
25. Das, M.; Roy, A.; Maity, S.; Kar, S.; Sengupta, S. Solving fuzzy dynamic ship routing and scheduling problem through new genetic algorithm. *Decis. Mak. Appl. Manag. Eng.* **2022**, *5*, 329–361. [\[CrossRef\]](#)
26. Chen, X.; Chen, J.; Du, C.; Xu, Y. Region Coverage Path Planning of Multiple Disconnected Convex Polygons Based on Simulated Annealing Algorithm. In Proceedings of the IEEE 4th International Conference on Computer and Communication Engineering Technology (CCET), Beijing, China, 13–15 August 2021; pp. 238–242. [\[CrossRef\]](#)
27. 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. [\[CrossRef\]](#)
28. Negi, G.; Kumar, A.; Pant, S.; Ram, M. Optimization of Complex System Reliability using Hybrid Grey Wolf Optimizer. *Decis. Mak. Appl. Manag. Eng.* **2021**, *4*, 241–256. [\[CrossRef\]](#)
29. Ganguly, S. Multi-objective distributed generation penetration planning with load model using particle swarm optimization. *Decis. Mak. Appl. Manag. Eng.* **2020**, *3*, 30–42. [\[CrossRef\]](#)
30. Dijkstra, E. A note on two problems in connexion with graphs. *Numer. Math.* **1959**, *1*, 269–271. [\[CrossRef\]](#)
31. Kovacs, B.; Szayer, G.; Tajti, F.; Burdelis, M.; Korondi, P. A novel potential field method for path planning of mobile robots by adapting animal motion attributes. *Robot. Auton. Syst.* **2016**, *82*, 24–34. [\[CrossRef\]](#)
32. Orozco-Rosas, U.; Montiel, O.; Sepulveda, R. Mobile robot path planning using membrane evolutionary artificial potential field. *Appl. Soft Comput.* **2019**, *77*, 236–251. [\[CrossRef\]](#)
33. Huang, Z.; Chu, D.; Wu, C.; He, Y. Path planning and cooperative control for automated vehicle platoon using hybrid automata. *IEEE Trans. Intell. Transp. Syst.* **2019**, *20*, 959–974. [\[CrossRef\]](#)
34. Liu, J.; Xu, C.; Wu, Z.; Chen, Y. Intelligent rebar layout in rc building frames using artificial potential field. *Autom. Constr.* **2020**, *114*, 103172. [\[CrossRef\]](#)
35. Huang, L.; Zhou, F. Path planning of moving robot based on path optimization of D\*Lite algorithm. *Control Decis.* **2020**, *35*, 877–884. [\[CrossRef\]](#)
36. Hart, P.; Nilsson, N.; Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.* **1972**, *4*, 28–29. [\[CrossRef\]](#)

37. Ju, C.; Luo, Q.; Yan, X. Path Planning Using an Improved A-star Algorithm. In Proceedings of the 11th International Conference on Prognostics and System Health Management (PHM), Jinan, China, 23–25 October 2020; pp. 23–26. [\[CrossRef\]](#)
38. Cao, R.; Zhang, Z.; Li, S.; Zhang, M.; Li, H.; Li, M. Multi-machine Cooperation Global Path Planning Based on A-star Algorithm and Bezier Curve. *Trans. Chin. Soc. Agric. Mach.* **2021**, *52*, 548–554. [\[CrossRef\]](#)
39. Gochev, K.; Safonova, A.; Likhachev, M. Anytime tree-restoring weighted A\* graph search. In Proceedings of the Seventh Annual Symposium on Combinatorial Search, Prague, Czech Republic, 15–17 August 2014; pp. 80–88.
40. Huang, X.; Dong, X.; Ma, J.; Liu, K.; Ahmed, S.; Lin, J.; Qiu, B. The Improved A\* Obstacle Avoidance Algorithm for the Plant Protection UAV with Millimeter Wave Radar and Monocular Camera Data Fusion. *Remote Sens.* **2021**, *13*, 3364. [\[CrossRef\]](#)
41. Xin, Y.; Liang, H.; Du, M. An improved A\* algorithm for searching infinite neighborhoods. *Robot* **2014**, *36*, 627–633. [\[CrossRef\]](#)
42. Chen, J.; Cui, Y.; Liu, X. Path planning method of mobile robot based on improved A\* algorithm. *Appl. Res. Comput.* **2020**, *37*, 118–119.
43. Kong, J.; Zhang, P.; Liu, X. Research on Improved A\* Algorithm of Bidirectional Search Mechanism. *Comput. Eng. Appl.* **2021**, *57*, 231–237. [\[CrossRef\]](#)
44. Lin, M.; Yuan, K.; Shi, C.; Wang, Y. Path planning of mobile robot based on improved A\* algorithm. In Proceedings of the 29th Chinese Control and Decision Conference (CCDC), Chongqing, China, 28–30 May 2017; pp. 3570–3576. [\[CrossRef\]](#)
45. Islam, F.; Narayanan, V.; Likhachev, M. Dynamic multi-heuristic A\*. In Proceedings of the IEEE International Conference on Robotics and Automation, Seattle, WA, USA, 26–30 May 2015; pp. 2376–2382, 2152–4092. [\[CrossRef\]](#)
46. Fu, B.; Chen, L.; Zhou, Y. An improved A\* algorithm for the industrial robot path planning with high success rate and short length. *Robot. Auton. Syst.* **2018**, *106*, 26–37. [\[CrossRef\]](#)
47. Zhou, B.; Gao, F.; Wang, L.; Liu, C.; Shen, S. Robust and efficient quadrotor trajectory generation for fast autonomous flight. *IEEE Robot. Autom. Lett.* **2019**, *4*, 3529–3536. [\[CrossRef\]](#)
48. Wu, X.; Xu, L.; Zhen, R.; Wu, X. Bi-Directional Adaptive A\* Algorithm Toward Optimal Path Planning for Large-Scale UAV Under Multi-Constraints. *IEEE Access* **2020**, *8*, 85431–85440. [\[CrossRef\]](#)
49. Muñoz, J.; Li, B.; Rong, X.; Xiao, J.; Tian, Y.; Ardit, A. An assistive indoor navigation system for the visually impaired in multi-floor environments. In Proceedings of the IEEE 7th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER), Honolulu, HI, USA, 31 July–4 August 2017; pp. 7–12. [\[CrossRef\]](#)
50. Liu, Z.; Liu, H.; Lu, Z. A dynamic fusion pathfinding algorithm using delaunay triangulation and improved A-star for mobile robots. *IEEE Access* **2021**, *9*, 20602–20621. [\[CrossRef\]](#)
51. Zhou, S.; Kang, F.; Li, W.; Kan, J.; Zheng, Y.; He, G. Extracting Diameter at Breast Height with a Handheld Mobile LiDAR System in an Outdoor Environment. *Sensors* **2019**, *19*, 3212. [\[CrossRef\]](#)
52. Shen, K.; You, Z.; Liu, Y.; Huang, T. Mobile robot planning based on improved a\* algorithm. *Appl. Res. Comput.* **2022**, *41*, 75–79. [\[CrossRef\]](#)
53. Wang, H.; Liu, Y. A low-cost autonomous navigation system for a quadrotor in complex outdoor environments. *Int. J. Adv. Robot. Syst.* **2020**, *17*, 1729881420905150. [\[CrossRef\]](#)

**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.