

Article

Hybrid A-Star Path Planning Method Based on Directional Guidance and Segment-Based Heuristic Weighting

Sanqiang Zhang ^{1,2,*} , Qize Guo ^{1,2}, Wei Liu ^{1,2}, Hongbin Kang ^{1,2}, Dianjian Wu ^{1,2} and Hongyu Zhou ³

¹ Agricultural Machinery Engineering Research and Design Institute, Hubei University of Technology, Wuhan 430068, China; gqz_article@163.com (Q.G.); liuwei_article@163.com (W.L.); xiangxiedongshaywh@163.com (H.K.); hbut_wdj@163.com (D.W.)

² Hubei Provincial Engineering Research Center for Intelligent Agricultural Machinery Equipment, Wuhan 430068, China

³ School of Industrial Design, Hubei University of Technology, Wuhan 430068, China; hongyu@hbut.edu.cn

* Correspondence: zsqa_article@hbut.edu.cn

Abstract

To improve the planning efficiency of the Hybrid A-Star algorithm in facility-based agriculture environments, we propose a novel method that integrates directional guidance and segment-based heuristic weighting. Jump point search (JPS) is first employed to generate a reference path, from which key jump points and their heading angles are extracted to guide node expansion. The path is then segmented into straight and turning segments, each assigned dynamic heuristic weights to form a segment-wise heuristic sequence. Simulation results show that, while maintaining comparable path lengths, the proposed method significantly improves computational efficiency, reduces node expansion, and enhances overall path smoothness. These results highlight the proposed method's effectiveness for path planning in autonomous mobile platforms operating within agricultural facilities.

Keywords: path planning; Hybrid A-Star algorithm; facility agriculture; directional guidance; dynamic weighting



Academic Editor: Pedro Couto

Received: 7 October 2025

Revised: 11 November 2025

Accepted: 18 November 2025

Published: 19 November 2025

Citation: Zhang, S.; Guo, Q.; Liu, W.; Kang, H.; Wu, D.; Zhou, H. Hybrid A-Star Path Planning Method Based on Directional Guidance and Segment-Based Heuristic Weighting. *Appl. Sci.* **2025**, *15*, 12298. <https://doi.org/10.3390/app152212298>

Copyright: © 2025 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

Under the dual pressures of continuous global population growth and tight arable land resources, the smart agricultural system, with the core goals of precision, efficiency, and sustainability, is developing rapidly. Aiming to increase agricultural output and resource utilization efficiency, reduce labor intensity, and ensure food safety, smart agriculture uses technologies such as the Internet of Things (IoT), big data, artificial intelligence, and robotics to monitor and make decisions on the environment, soil, crop growth conditions, and production processes in real time [1–3]. In facility-based agriculture such as greenhouse farms, key indicators such as temperature, humidity, light and water, and fertilizer can be precisely monitored and managed. The automation and intelligence of agricultural equipment is another important aspect to ensure the efficient and stable production process.

Against this backdrop, the automatic navigation technology for agricultural machinery is a key element of smart agriculture, with path planning constituting the core of such systems [4,5]. In facility-based environments such as greenhouses, the working areas are mostly characterized by multiple rows of symmetry, narrow and long corridors, and dense obstacles. When performing tasks, intelligent transportation vehicles need to pass through complex U-shaped areas, and this poses significant challenges to the robustness and safety of the path planning. To ensure the crops are not damaged and to enhance the delivery

efficiency, the planned path not only needs to comply with the kinematic constraints of the vehicle but also needs to generate a feasible trajectory with low oscillation and high smoothness and maintain an adequate safety distance from the corridor walls and obstacles.

Conventional path planning algorithms primarily encompass graph search-based methods (e.g., A-star and Dijkstra), sampling-based approaches (e.g., RRT), optimization-based techniques (e.g., gradient descent), and curve interpolation methods (e.g., Dubins and Bézier curves) [6]. Among these, the A-star algorithm is commonly used for global path planning in static maps due to its stability and heuristic characteristics [7,8]. A-star algorithm uses a heuristic function to search for the shortest path on a grid map. However, in situations with dense obstacles, it is prone to generating a large number of redundant nodes and resulting in decreased computational efficiency [9]. In a real agricultural environment, the A-star algorithm faces challenges such as excessive node expansion, abrupt curvature changes, and inability to accommodate dynamic constraints [10,11].

To address these issues, the Hybrid A-star algorithm was developed. This method combines the heuristic search framework of A-star with the trajectory generation mechanism in continuous space. It enhances the feasibility and smoothness of the path by integrating kinematically feasible extension strategies (such as Reeds-Shepp or Dubins paths) [12,13]. The Hybrid A-star algorithm has demonstrated excellent performance in automatic parking [14–16], complex structural environments, and intelligent agricultural scenarios [17–19].

To further enhance the path planning performance of the Hybrid A-star algorithm in complex constraint environments, researchers have proposed various improvement strategies. From the standpoint of path continuity, some researchers apply Bézier curves or cubic splines to smooth the discrete paths generated by the Hybrid A-star algorithm, which enhance the stability of path tracking control [20]. Some studies have incorporated G2 continuity constraints during the path generation stage to ensure the smoothness of curvature changes, thereby improving path quality and compatibility with vehicle control systems [21]. To address the challenges of excessive node expansions in the Hybrid A-star algorithm when dealing with high-resolution maps, Cui et al. [22] proposed an improved method based on geometric curves and direction guidance. During the planning stage, multiple controllable arcs are implemented, and the path expansion direction is dynamically guided by the local geometric structure, effectively reducing the expansion of invalid nodes and improving the search efficiency. To further reduce the search space, some studies have incorporated the Jump Point Search (JPS) algorithm into the initial guidance stage of the ant colony optimization. As reported in the literature [23], initializing pheromone distribution with jump points expedites early iterations by directing the algorithm toward more promising solution regions. The above studies all demonstrate that the Hybrid A-star algorithm has excellent scalability and integration capabilities and is suitable for path planning tasks in facility agriculture that involve multiple types of obstacles and highly coupled kinematic constraints.

In recent years, research on path planning for facility-based agricultural such as greenhouses and orchards has exhibited significant trends of integration, multi-level structure, and high real-time performance. In orchard operation, in order to meet the actual requirements of the grass-cutting robot for operation in the dense planting rows, Li et al. [24] proposed a path generation strategy based on safety corridors and secondary planning. Their approach divides the environment into high-security zones and drivable areas and then performs trajectory optimization. It effectively avoids obstacles while maintaining high controllability. For articulated tractor path optimization, some scholars have utilized a multi-objective function that integrates genetic algorithms with a multi-segment structure encoding method to comprehensively optimize the path length, the number of turns, and

the alignment accuracy. It demonstrates that multi-dimensional performance indicators are gradually gaining attention in agricultural path planning [25]. Overall, agricultural path planning is progressing towards structural adaptation and model integration, emphasizing the deployability and stability of algorithms in actual operation systems.

However, in large-scale greenhouse or multi-narrow corridor environments, the conventional Hybrid A-star algorithm often exhibits insufficiencies in node expansion owing to unnecessary exploration into dead zones. The adopted fixed heuristic poses challenges in balancing search efficiency with path quality. To overcome these limitations, this study proposes a novel Hybrid A-star algorithm path planning method based on directional guidance and segmented heuristic weighting, which takes into account both search efficiency and path feasibility. The proposed approach introduces two principal enhancements: directional node expansion and adaptive heuristic weighting. First, the JPS algorithm is employed to extract key jump points from the planned path and utilize them as the direction guiding points for the Hybrid A-star algorithm. The guiding nodes expand successively in the direction of the key jump points to avoid the ineffective node expansion. Secondly, by calculating the difference in reference heading angles between adjacent key jump points and combining with axis alignment detection, determine whether the current path segment is a straight segment or a turning segment. Higher heuristic weights are allocated to straight segments to accelerate the search process along linear corridors, whereas lower weights are retained for turning segments to preserve trajectory smoothness and planning quality.

2. Proposed Path Planning Framework and Methodology

2.1. Overview of the Proposed Path Planning Framework

In structured agriculture environments such as greenhouses, U-shaped obstacles are frequently encountered. When applying the Hybrid A-star algorithm for path planning in these settings, the search process often extends into dead zones, resulting in a large number of invalid node expansions and decreased computational efficiency. As shown in Figure 1a, a typical tomato greenhouse is depicted. For path planning purposes, the structural layout of the greenhouse is abstracted into a 2D grid map. Figure 1b shows the corresponding path generated by the Hybrid A-star algorithm. In the grid representation, black cells indicate obstacle regions, white cells represent navigable workspace, and the blue box underscores an area where the algorithm expands into a dead zone, leading to inefficient and unnecessary node exploration.

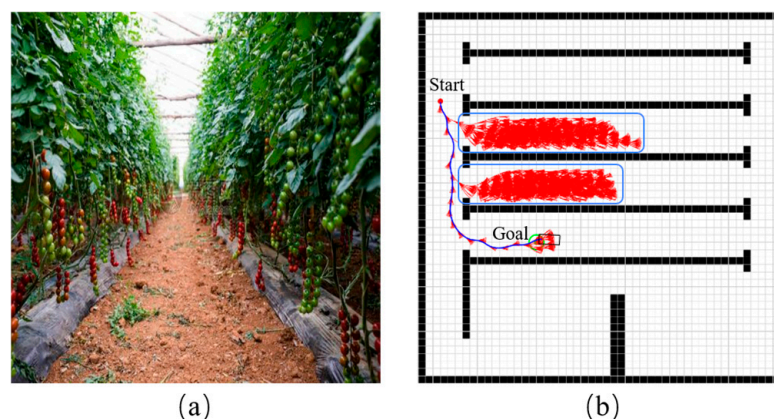


Figure 1. (a) A typical tomato greenhouse and (b) redundant expansion of the Hybrid A-star. The red dot is the starting point, the green dot is the target point, and the green circle indicates the parking area.

A preliminary path is quickly generated using the JPS algorithm between the start and goal positions, from which a sequence of key jump points is extracted. These jump points act as directional references for the subsequent Hybrid A-star planning process, effectively reducing unnecessary exploration into dead zones. For each jump point, a reference heading angle is calculated based on the coordinates of two adjacent jump points. By analyzing heading angle differences and axis alignment between successive segments, the path is classified into either turning or straight segments. Each segment is then assigned a dynamic heuristic coefficient, resulting in a segment-wise heuristic coefficient list. This allows accelerated node expansion in straight-line regions and enhanced path smoothness in turning regions. When the expanding node enters a predefined tolerance region around the next jump point, the algorithm switches to the corresponding heuristic coefficient for that segment. The total cost function integrates the travel cost, steering continuity penalty, and directional deviation penalty, thereby promoting efficient forward exploration while constraining node expansion in non-optimal directions. The overall algorithm workflow is shown in Figure 2.

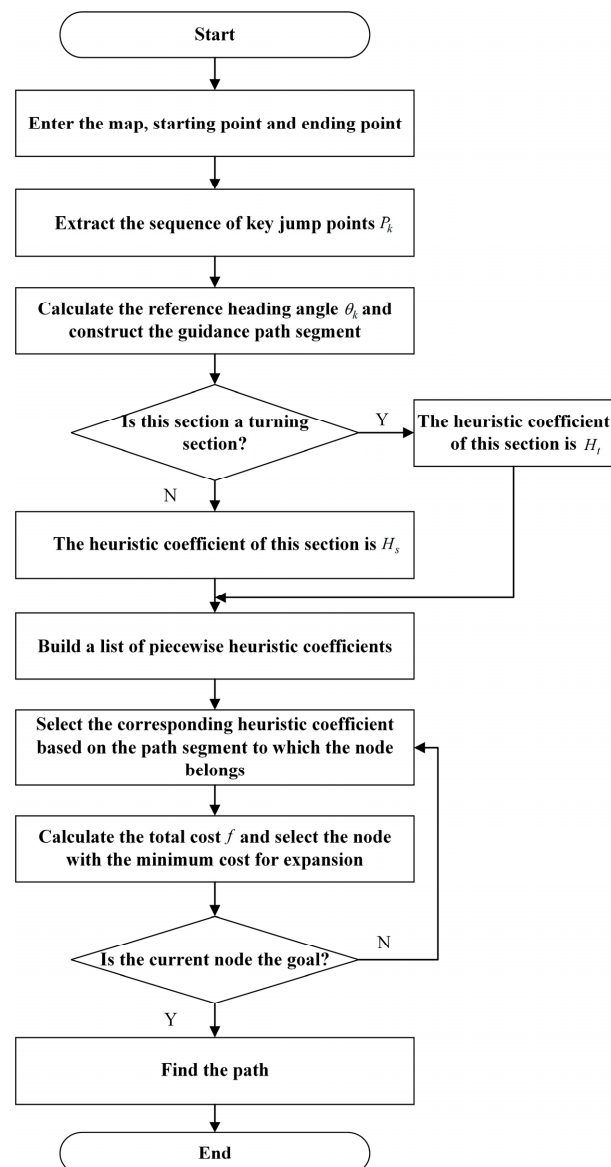


Figure 2. Algorithm workflow.

2.2. Vehicle Kinematic Model and Node Expansion Strategy

The improved Hybrid A-star algorithm proposed in this study is formulated based on an Ackermann-steering vehicle model. The kinematic structure of the Ackermann model is presented in Figure 3. In the diagram, (P_x, P_y) represent the coordinates of the rear axle center, θ denotes the vehicle's orientation (heading angle), and φ represents the front-wheel steering angle. In addition to these motion parameters, the figure also presents key geometric parameters of the vehicle, including L (vehicle length), L_f (front overhang), L_m (wheelbase), L_r (rear overhang), and W (vehicle width). Based on the Ackermann kinematic constraints and the vehicle's geometric configuration, the minimum turning radius R_{min} can be calculated as:

$$R_{min} = \frac{L_m}{\tan \varphi_{max}} \quad (1)$$

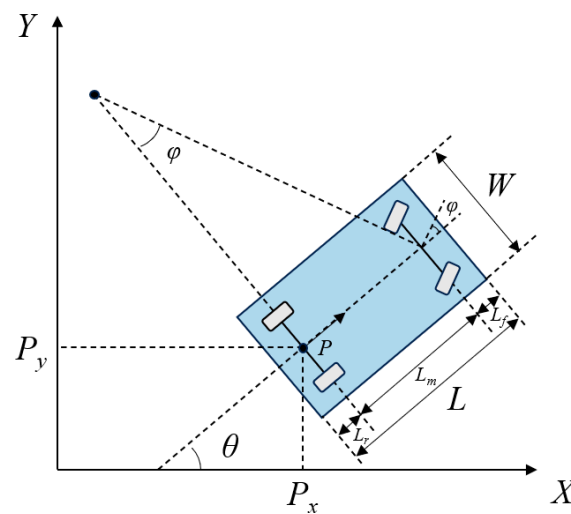


Figure 3. Kinematic structure of the Ackermann steering model.

The Hybrid A-star algorithm enhances the conventional A-star algorithm by incorporating the vehicle's kinematic constraints, thereby allowing the generation of paths that are physically feasible and trackable by real vehicles. Unlike the A-star algorithm, which expands nodes strictly based on adjacent grid centers, the Hybrid A-star algorithm utilizes a set of discrete motion primitives defined by step length, turning radius, and front wheel steering angles, as shown in Figure 4. The red line segments correspond to expansion trajectories, red dots mark the expanded nodes, and the blue solid line shows the generated path. This approach allows successor nodes to occupy arbitrary positions within the grid, rather than being confined to grid cell centers as in the standard A-star algorithm. As a result, the Hybrid A-star algorithm more accurately models continuous motion and supports the planning of kinematically feasible trajectories.

The state of the vehicle is represented by the position and heading angle of the rear-axle center, denoted as $s = (P_x, P_y, q)$. Assuming each control input consists of a linear displacement D and a front-wheel steering angle φ , the relationship between the current node and its successor node can be described using the following kinematic equations:

$$\begin{cases} P_{x(i+1)} = P_{x(i)} + D \cos(\theta_i) \\ P_{y(i+1)} = P_{y(i)} + D \sin(\theta_i) \\ \theta_{i+1} = \theta_i + \frac{D}{L} \tan \varphi \end{cases} \quad (2)$$

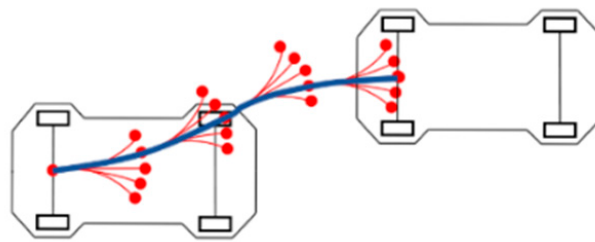


Figure 4. Node expansion process in the Hybrid A-star algorithm. The blue solid line represents the planned path, red dots indicate the expanded nodes, and red lines denote the expansion trajectories between successive nodes.

2.3. Jump Point-Based Global Guidance Mechanism

The JPS algorithm is an optimized variant of the conventional A-star algorithm. In the standard A-star approach, each node expansion involves evaluating potential movements in all eight directions around the current node, making the computation process relatively complex and time-consuming. JPS enhances efficiency by eliminating unnecessary intermediate nodes and retaining only a set of representative key nodes, referred to as jump points, to guide the search. These jump points allow the algorithm to reduce redundancy and accelerate the overall path planning process.

Based on the distribution of obstacles in the map, the neighboring nodes around a given node can be categorized into two types: natural neighbors and forced neighbors. For example, in the absence of surrounding obstacles, consider a scenario where the parent node expands horizontally from node 4 to node x . If a neighboring node n satisfies the condition that the path passing through x is shorter than any alternative path that avoids x , then n is identified as a natural neighbor. This relationship is illustrated by the gray cells in Figure 5a.

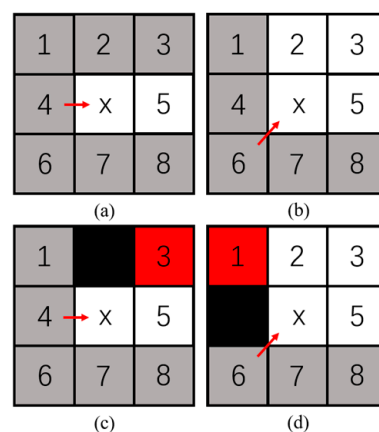


Figure 5. Illustration of forced neighbors. (a,b) respectively illustrate the cases of horizontal expansion and diagonal expansion of nodes in an obstacle-free environment. (c,d) respectively illustrate the cases of horizontal expansion and diagonal expansion of nodes in the presence of obstacles. The black grids represent obstacles, the gray grids represent natural neighbors, the red grids represent forced neighbors, and the white grids represent passable areas.

When the expansion direction is either horizontal or vertical, the criteria for identifying natural neighbors can be formally defined as follows:

$$L(P(x), \dots, n) \geq L(P(x), x, n) \quad (3)$$

where $L(\cdot)$ represents path length; $\langle P(x), \dots, n \rangle$ denotes a path from $P(x)$ to n that does not pass through x ; $\langle P(x), x, n \rangle$ denotes the path $P(x) \rightarrow x \rightarrow n$; $P(x)$ is the parent

node of x . When expanding diagonally from parent node 6 to node x , a neighboring node n is considered a natural neighbor if the path that passes through node x is longer than the alternative path that bypasses x , as illustrated in Figure 5b. This condition typically occurs during diagonal expansions under specific geometric constraints.

In an obstacle-free environment, when the expansion direction is diagonal, the criteria for identifying natural neighbors can be defined as follows:

$$L(P(x), \dots, n) > L(P(x), x, n) \quad (4)$$

When obstacles are present around the current node, a neighboring node n is identified as a forced neighbor if the cost of the path passing through node x from the parent node is lower than the cost of bypassing x to reach n . In other words, the presence of an obstacle forces the path to include node x to maintain optimality. Such nodes are highlighted in red in Figure 5c,d, which illustrate typical scenarios where forced neighbors arise owing to the presence of adjacent obstacles.

A neighboring node n is considered a forced neighbor if it satisfies both of the following conditions:

- (1) Node n is not a natural neighbor;
- (2) The path passing through node x from the parent node has a lower total cost than any alternative path reaching n that avoids x .

A node x is identified as a jump point if it satisfies at least one of the following conditions:

- (1) Node x is the start or end point;
- (2) Node x has at least one forced neighbor;
- (3) When expanding diagonally, both the horizontal and vertical neighbors of x satisfy either condition 1 or 2;

The path is initially planned on the grid map using the JPS algorithm. Based on the jump point selection criteria, key jump points are extracted from the resulting path and serve as directional guidance references for the Hybrid A-star algorithm. The selection of key jump points is illustrated in Figure 6.

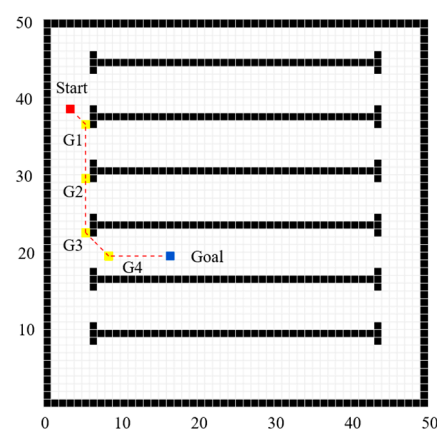


Figure 6. Selection of key jump points. The red grids represent the starting points, the blue grids represent the ending points, the red dotted lines indicate the planned paths, and the yellow grids (G1, G2 . . .) represent the key jump points.

2.4. Design of Tolerance Region Around Key Jump Points

To ensure collision-free path planning and enable node expansion into regions guided by key jump points, a tolerance region is defined around each jump point, as illustrated in

Figure 7. This tolerance area also enhances the continuity between different path segments that are assigned different heuristic weights.

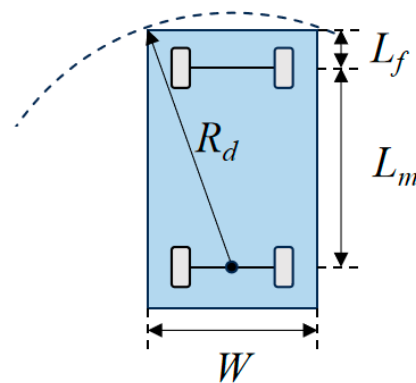


Figure 7. Radius of tolerance zone.

The tolerance region is defined as a circular area centered at each jump point, with its radius determined based on the vehicle's geometric dimensions. Let R_d represent the diagonal distance from the rear-axle center to the frontmost point of the vehicle. Given L_f (front overhang), L_m (wheelbase), and W (vehicle width), the radius R_d of the positional tolerance region is expressed as:

$$R_d = \sqrt{(L_m + L_f)^2 + \left(\frac{W}{2}\right)^2} \quad (5)$$

This tolerance ensures that any node entering the vicinity of a jump point can be considered to have successfully reached it, thereby facilitating smooth heuristic transitions and stable trajectory continuity across path segments.

2.5. Segment-Based Heuristic Weighting Strategy

In the traditional Hybrid A-star algorithm, the heuristic weight H_{cost} is typically held constant throughout the planning process. However, maintaining a constant heuristic coefficient makes it challenging to strike an optimal balance between search efficiency in straight-line segments and path smoothness in turning segments. To overcome this limitation, a segment-based heuristic weighting mechanism is proposed. By calculating the heading angle difference between two adjacent key jump points and determining their axis alignment, each path segment is categorized as either a straight or a turning segment. Distinct heuristic weights are then assigned based on this classification: higher weights are used in straight segments to accelerate the search, whereas lower weights are used in turning segments to preserve path quality.

Because the key jump points extracted from the JPS algorithm lack heading information, a reference heading angle θ_k is computed for each key jump point (excluding the start and goal points), based on the coordinates of its adjacent jump points. The reference heading angle θ_k for the k jump point is calculated as:

$$\theta_k = \text{atan2}(y_{k+1} - y_k, x_{k+1} - x_k) \quad (6)$$

where $P_{k+1}(x_{k+1}, y_{k+1})$ and $P_k(x_k, y_k)$ are the coordinates of two adjacent jump points, respectively.

To classify path segments and assign appropriate heuristic weights, both the heading angle difference and axial alignment are evaluated for all pairs of adjacent key jump points,

including the start and goal points. A path segment between two consecutive jump points P_{k+1} and P_k is classified as a turning segment if either of the following conditions is met:

- (1) The absolute difference between their reference heading angles exceeds a predefined threshold θ_s ;
- (2) The two points are not aligned along the same axis (i.e., they are not directionally collinear).

Otherwise, the segment is classified as a straight segment. The heuristic weight H_i for each segment is assigned accordingly:

$$H_i = \begin{cases} H_t, \Delta\theta > \theta_s \wedge (x_i \neq x_{i+1} \wedge y_i \neq y_{i+1}) \\ H_s = \min\left\{\left(1 + \frac{\theta_s}{\Delta\theta}\right)H_t, H_{s-max}\right\}, \Delta\theta \leq \theta_s \end{cases} \quad (7)$$

$$\Delta\theta = |\theta_{k+1} - \theta_k| \quad (8)$$

These parameters were empirically tuned through multiple preliminary simulations to balance path smoothness and computational efficiency, and the same values were consistently used across all experiments.

- θ_s represents the threshold for heading angle difference used to distinguish between straight and turning segments.
- H_s represents the heuristic weight for straight segments.
- H_t represents the heuristic weight for turning segments.
- H_{s-max} represents the upper bound for the heuristic weight applied to straight segments.
- θ_k represents the reference heading angle corresponding to jump point P_k .
- $\Delta\theta$ represents the difference in the reference heading angles between two adjacent key jump points P_k .

2.6. Cost Function

In the conventional Hybrid A-star algorithm, the total cost function $f(n)$ comprises two parts: the accumulated cost $g(n)$ and the heuristic cost $h(n)$. The total cost is expressed as:

$$f(n) = g(n) + h(n) \quad (9)$$

The heuristic cost $h(n)$ estimates the distance from the current node to the goal. It is typically computed using two different methods, and the maximum of the two is selected as the final heuristic value:

- The Reeds–Shepp path length h_{R-S} , which accounts for vehicle kinematic constraints but does not consider obstacles;
- The A-star algorithm path length h_A , which considers static obstacles but ignores the vehicle's nonholonomic constraints.

Thus, the conventional heuristic function is defined as:

$$h(n) = \max(h_{R-S}, h_A) \quad (10)$$

However, in structured environments such as greenhouses, where wide open spaces are absent, a heuristic focused solely on obstacle avoidance often delivers superior performance. Therefore, in this study, we discard the h_{R-S} term and adopt only the obstacle-aware heuristic. Additionally, a segment-based heuristic weighting mechanism is applied according to the path segment classification introduced earlier. The final heuristic function is expressed as:

$$h(n) = H_i h_A \quad (11)$$

To better capture the frequency and intensity of steering adjustments during vehicle path tracking, we refine the accumulated cost function $g(n)$ by incorporating both the cumulative steering angle change and a directional guidance penalty. The enhanced accumulated cost is expressed as:

$$\begin{cases} g(n_1) = D \\ g(n_2) = \sum_{i=0}^{n-1} |\varphi_{i+1} - \varphi_i| \\ g(n_3) = \sum_{i=0}^{n-1} |\theta_k - \theta_i| \\ g(n_i) = g(n) + \alpha g(n_1) + \beta g(n_2) + \gamma g(n_3) \end{cases} \quad (12)$$

These parameters were empirically tuned through multiple preliminary simulations to balance path smoothness and computational efficiency, and the same values were consistently used across all experiments.

- $g(n_1)$ denotes the set expansion step length;
- $g(n_2)$ denotes the total change in front wheel steering angles across all steps;
- $g(n_3)$ denotes the cumulative heading deviation from the reference heading angle of the corresponding key jump point P_k ;
- φ_i denotes the steering control input at step i ;
- θ_i is the actual heading angle of the vehicle at step i ;
- θ_k is the reference heading angle corresponding to key jump point P_k ;
- $g(n_i)$ denotes the accumulated cost at the child node;
- $g(n)$ denotes the accumulated cost at the parent node;
- α , β , and γ are weighting coefficients controlling the contributions of path length, steering effort, and directional penalty, respectively.

2.7. Simulation Setup

The proposed Hybrid A-star algorithm was implemented and evaluated using MATLAB R2020a. All simulations were performed on a laptop computer with the following specifications: AMD R7-7840H CPU, 16 GB RAM, and an NVIDIA GeForce RTX 4060 (8 GB GPU). The test environment consisted of a 50 m $\times$ 50 m structured map populated with several elongated U-shaped obstacles, with a grid resolution of 1 m. To evaluate the effect of the segment threshold on the proposed path planning algorithm, a single-factor analysis was conducted. The influence of different threshold settings on planned path length, sum of steering angle changes, computational time, and expanded node count was examined.

Experiments were performed using the Dy-DG-Hybrid A-Star algorithm under Map Setup 2. For each segment threshold configuration, five repeated simulations were carried out, and the average values were used for analysis, as summarized in Table 1. The results demonstrate that as the segment threshold increases, the planned path length, sum of steering angle changes, and expanded node count remain nearly constant. However, the computational time exhibits a slight nonlinear trend, first decreasing and then increasing marginally. Based on the results in Table 1, when the segment threshold is set between 10° and 15°, the computational time reaches its minimum, indicating that this range achieves a balanced trade-off between search efficiency and computational stability.

Table 1. Single-Factor Analysis of the Segment Threshold Parameter.

Segment Threshold (°)	Planned Path Length (m)	Sum of Steering Angle Changes (Rad)	Computational Time (s)	Expanded Node Count
5	30.3	5.28	3.70	177
10	30.3	5.28	2.86	177
15	30.3	5.28	2.87	177
20	30.3	5.28	2.99	177

To assess the effectiveness of the proposed method, three path planning algorithms were compared: Standard Hybrid A-star [26]; Direction-Guided Hybrid A-star (DG-Hybrid A-star), which incorporates directional guidance based on JPS key points; and Direction-Guided and Dynamically Weighted Hybrid A-star (Dy-DG-Hybrid A-star), which integrates both directional guidance and segment-based heuristic weighting. The experimental parameters of the path planning platform are shown in Table 2.

Table 2. Experimental Parameters of the Path Planning Platform.

Length of vehicle (L)	2.8 m
Width of vehicle (W)	1.3 m
Wheelbase (L_m)	1.9 m
Front overhang (L_f)	0.4 m
Rear overhang (L_r)	0.5 m
Minimum turning radius	2.78 m
Segment threshold (θ_s)	10°
Tolerance radius (R_d)	2.4 m

3. Results and Discussion

Each algorithm was evaluated under two simulation configurations. Setup 1 and setup 2 used the same map and identical starting positions but differed in their goal locations. The path length was computed by discretizing the final path obtained through backtracking, using a fixed interval of 0.1 m, and summing the Euclidean distances between consecutive points.

Table 3 presents the planned path lengths, cumulative steering angle changes, computational times, and the number of expanded nodes for each algorithm under the two simulation setups.

Table 3. Comparison results.

Indexes	Setup	Hybrid A-Star	DG-Hybrid A-Star	Dy-DG-Hybrid A-Star
Planned path length (m)	Setup 1	15.9	15.9	15.9
	Setup 2	30.3	30.3	30.3
Sum of steering angle changes (rad)	Setup 1	6.12	4.20	4.20
	Setup 2	8.04	5.76	5.28
Computational time (s)	Setup 1	1.80	1.21	1.12
	Setup 2	182.98	2.85	2.26
Expanded node count	Setup 1	119	109	103
	Setup 2	2324	206	177

The planned paths generated by the JPS algorithm for setup 1 and setup 2 are illustrated in Figure 8. Figure 8a presents the JPS planning result for setup 1, and Figure 8b presents the corresponding planning result for setup 2.

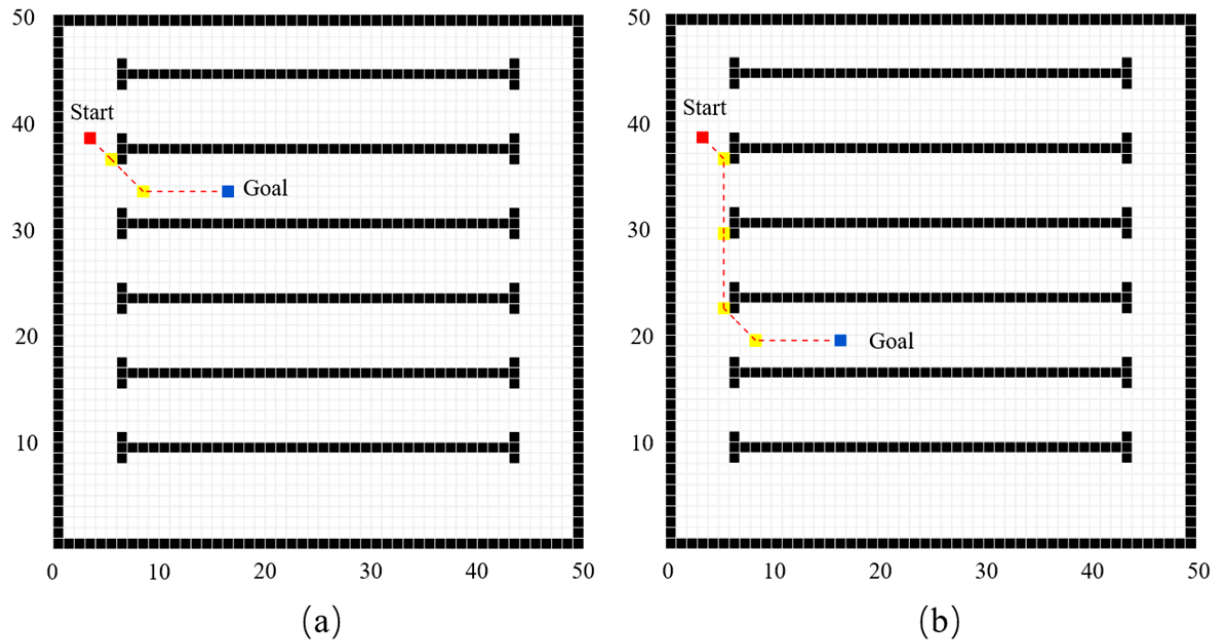


Figure 8. Path planning results of the JPS algorithm under (a) setup 1 and (b) setup 2. The red grids represent the starting points, the blue grids represent the ending points, the red dotted lines indicate the planned paths, and the yellow grids represent the key jump points.

In setup 1, the planned path lengths of the Hybrid A-star algorithm, DG-Hybrid A-star algorithm, and Dy-DG-Hybrid A-star algorithm were all approximately 15.9 m, indicating that the path lengths remained essentially consistent across all three methods. The cumulative steering angles for the Hybrid A-star, DG-Hybrid A-star, and Dy-DG-Hybrid A-star algorithms were 6.12 radians, 4.20 radians, and 4.20 radians, respectively. Compared to the standard Hybrid A-star, both the DG-Hybrid A-star and the Dy-DG-Hybrid A-star algorithms reduced the cumulative steering angle by 31.37%. The computational times for the Hybrid A-star, DG-Hybrid A-star, and Dy-DG-Hybrid A-star algorithms are 1.80 s, 1.21 s, and 1.12 s, respectively. Compared to the Hybrid A-star algorithm, the computational times of the DG-Hybrid A-star and Dy-DG-Hybrid A-star algorithms are reduced by 32.78% and 37.78%, respectively. The number of expanded nodes for the Hybrid A-star, DG-Hybrid A-star, and Dy-DG-Hybrid A-star algorithms was 119, 109, and 103, respectively. Compared to the Hybrid A-star, the number of expanded nodes for the DG-Hybrid A-star and Dy-DG-Hybrid A-star algorithms was reduced by 8.4% and 13.45%, respectively.

The planned paths under setup 1 for the three algorithms are shown in Figure 9. Figure 9a–c present the results of the standard Hybrid A-star, DG-Hybrid A-star, and Dy-DG-Hybrid A-star algorithms, respectively. In Figure 9, the red dots are the starting points, the green dots are the goal points, the blue solid lines represent the planned paths, and the yellow dots represent the key jump points. The green circles in Figure 9a represent the parking area, while the red circles in Figure 9b,c represent the tolerance radius. The parking criteria of all algorithms are consistent.

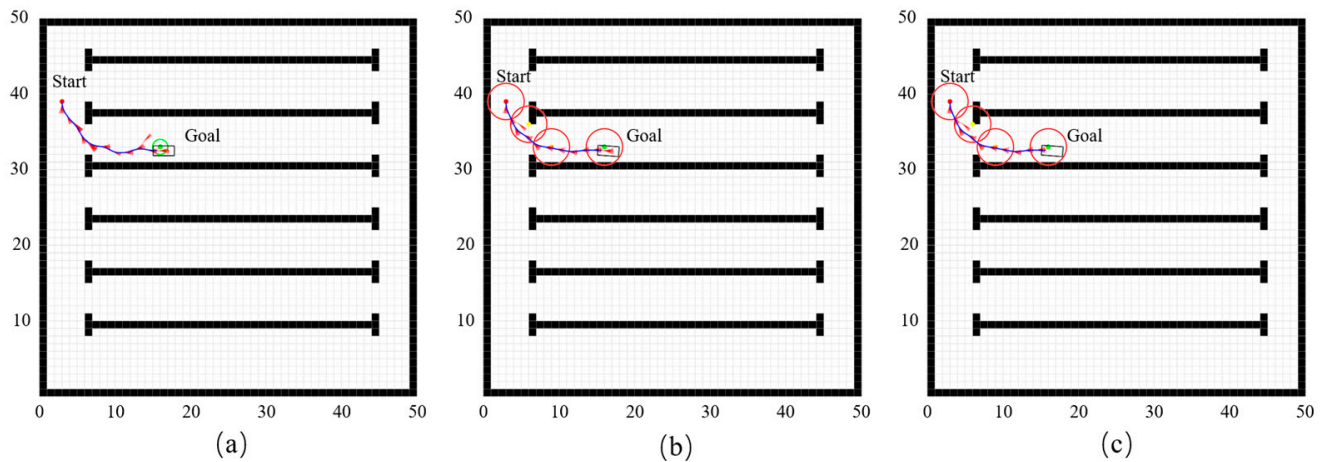


Figure 9. Comparison results for the (a) Hybrid A-star algorithm, (b) DG-Hybrid A-star algorithm, and (c) Dy-DG-Hybrid A-star algorithm under setup 1.

To further assess the practical performance of the algorithms under varying conditions, simulations were also conducted using setup 2. In setup 2, the planned path lengths for the Hybrid A-star, DG-Hybrid A-star, and Dy-DG-Hybrid A-star algorithms are all 30.3 m, remaining largely unchanged; The cumulative steering angles for the Hybrid A-star, DG-Hybrid A-star, and Dy-DG-Hybrid A-star algorithms are 8.04 radians, 5.76 radians, and 5.28 radians, respectively. Compared to the Hybrid A-star algorithm, the cumulative steering angles of the DG-Hybrid A-star and Dy-DG-Hybrid A-star algorithms are reduced by 28.36% and 34.33%, respectively. The computational times for the Hybrid A-star, DG-Hybrid A-star, and Dy-DG-Hybrid A-star algorithms are 182.98 s, 2.85 s, and 2.26 s, respectively. Compared to the Hybrid A-star algorithm, the computational times for the DG-Hybrid A-star and Dy-DG-Hybrid A-star algorithms are reduced by 98.44% and 98.76%, respectively. The number of expanded nodes for the Hybrid A-star, DG-Hybrid A-star, and Dy-DG-Hybrid A-star algorithms was 2324, 206, and 177, respectively. Compared to the Hybrid A-star algorithm, the number of expanded nodes for the DG-Hybrid A-star and Dy-DG-Hybrid A-star algorithms was reduced by 91.14% and 92.38%, respectively.

The planned paths under setup 2 for the three algorithms are presented in Figure 10. Figure 10a–c present the paths generated by the standard Hybrid A-star, DG-Hybrid A-star, and Dy-DG-Hybrid A-star algorithms, respectively. The visual elements in Figure 10 follow the same definitions as in Figure 9.

By comparing the path length, cumulative steering angle, computation time, and number of expanded nodes under setup 1 and setup 2, several key conclusions can be drawn. Compared to the original Hybrid A-star algorithm, the proposed DG-Hybrid A-star algorithm significantly reduces redundant node expansions, particularly around U-shaped obstacles, while maintaining a consistent path length. Specifically, DG-Hybrid A-star reduces the cumulative steering angle by 31.37%, computation time by 32.78%, and the number of expanded nodes by 8.4%. In addition, the Dy-DG-Hybrid A-star algorithm, which incorporates segmented heuristic weighting, offers further improvements—reducing computation time by 5% and node expansion by 5.04% compared to the DG-Hybrid A-star.

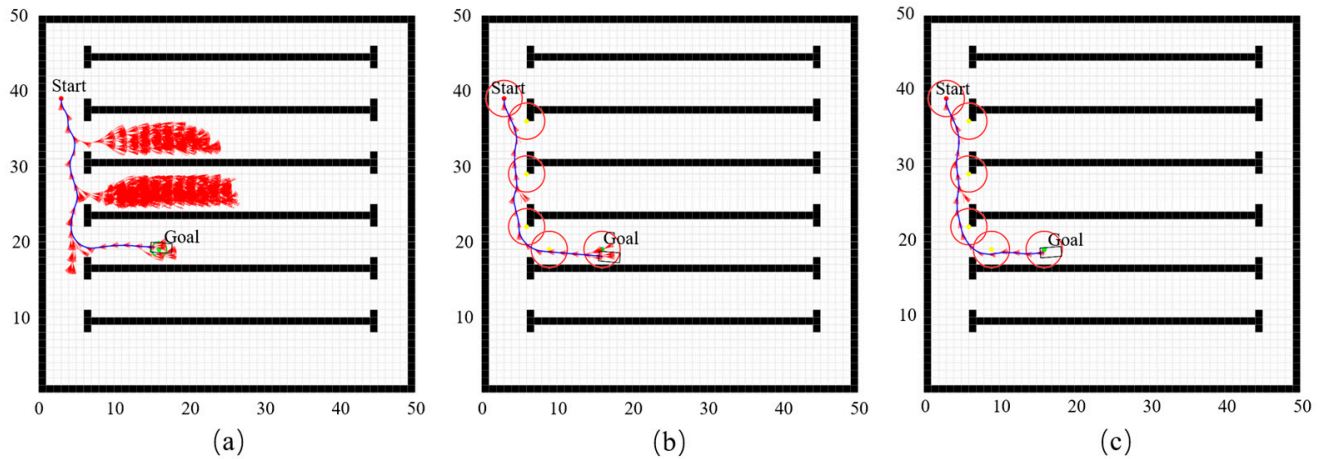


Figure 10. Comparison results for the (a) Hybrid A-star algorithm, (b) DG-Hybrid A-star algorithm, and (c) Dy-DG-Hybrid A-star algorithm under setup 2.

To more effectively illustrate the comparative performance of key evaluation metrics across the algorithms, both the absolute values and relative improvement rates are illustrated in Figures 11 and 12, respectively. In Figure 11, Dg-H-A represents the DG-Hybrid A-star algorithm, and Dy-Dg-H-A represents the Dy-DG-Hybrid A-star algorithms. In Figure 12a, Dg-H-A improvement represents the improvement rates of the DG-Hybrid A-Star algorithm compared to the Hybrid A-Star algorithm. In Figure 12a, the Dy-Dg-H-A improvement represents the improvement rates of the Dy-DG-Hybrid A-Star algorithm compared to the Hybrid A-Star algorithm. Notably, in Figure 12, as all algorithms produce identical path lengths, only the improvements in the remaining three metrics are evaluated for comparison. The Dy-DG-Hybrid A-star algorithm consistently achieves better performance than the other two methods in terms of steering smoothness, planning efficiency, and search efficiency. In the case of long-distance path planning under setup 2, the DG-Hybrid A-star and Dy-DG-Hybrid A-star algorithms reduce computation time by 98.44% and 98.76%, and the number of expanded nodes by 91.14% and 92.38%, respectively, highlighting the enhanced scalability of the proposed approach.

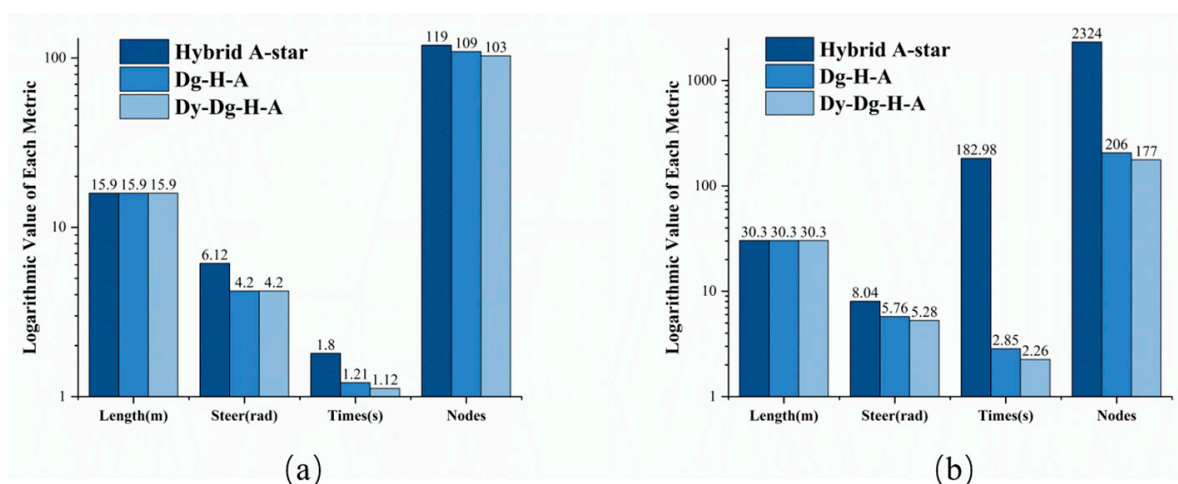


Figure 11. Comparison of absolute values of indexes under (a) setup 1 and (b) setup 2.

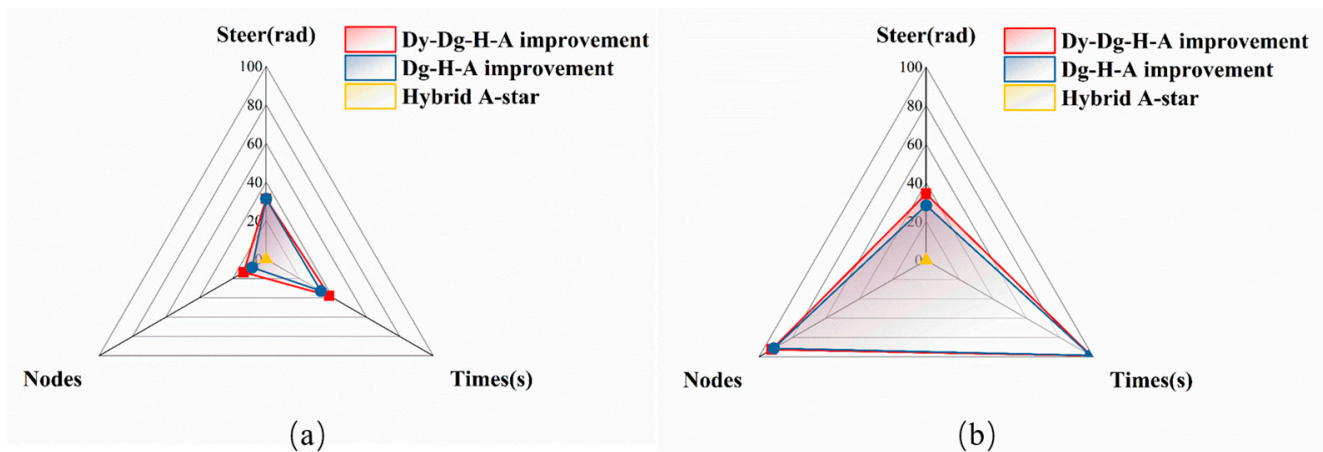


Figure 12. Improvement rates under (a) setup 1 and (b) setup 2.

To further investigate the performance of the proposed algorithm in complex environments and verify its feasibility and stability under realistic conditions, additional obstacles were introduced into the original map to create a more challenging scenario. For this environment, the path planning results of the Standard Hybrid A-Star algorithm, JPS algorithm, DG-Hybrid A-Star algorithm, and Dy-DG-Hybrid A-Star algorithm are summarized in Table 4.

Table 4. Path Planning Performance Comparison in Complex Environments.

Indexes	Hybrid A-Star	DG-Hybrid A-Star	Dy-DG-Hybrid A-Star
Planned path length (m)	30.7	30.7	30.7
Sum of steering angle changes (rad)	8.02	6.36	6.48
Computational time (s)	182.37	6.81	6.53
Expanded node count	1727	272	270

As shown in the results, the DG-Hybrid A-Star algorithm significantly reduces computational time, cumulative steering angle change, and the number of expanded nodes compared with the standard Hybrid A-Star algorithm. Furthermore, the Dy-DG-Hybrid A-Star algorithm, building upon DG-Hybrid A-Star, achieves further reductions in computation time and node expansion. However, a slight increase in the cumulative steering angle change is observed.

This increase is likely due to the frequent switching of dynamic heuristic coefficients during turning-intensive segments, which may cause some intermediate node expansions to be skipped, resulting in a larger overall steering angle accumulation.

To evaluate the path planning performance of different algorithms in complex environments, four methods were tested under the same obstacle configuration: the Standard Hybrid A-Star algorithm Figure 13a, the DG-Hybrid A-Star algorithm Figure 13b, and the Dy-DG-Hybrid A-Star algorithm Figure 13c.

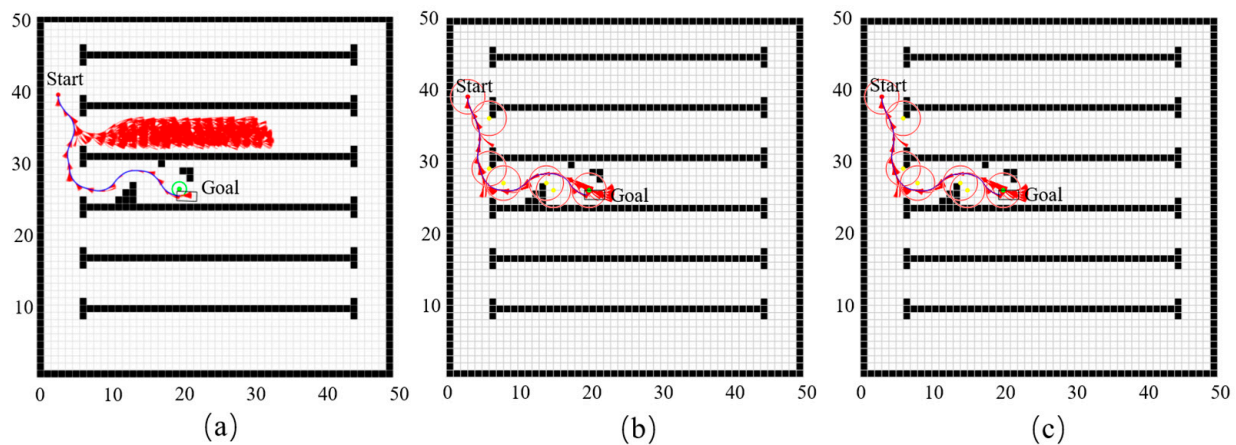


Figure 13. Comparison of path planning results under complex environments: (a) Standard Hybrid A-Star algorithm; (b) DG-Hybrid A-Star algorithm; (c) Dy-DG-Hybrid A-Star algorithm.

4. Conclusions

To enhance the planning efficiency of the Hybrid A-star algorithm in structured agricultural environments, such as greenhouses with elongated U-shaped obstacles, this study introduced an enhanced Hybrid A-star path planning method that combines directional guidance and a segmented heuristic weighting strategy. This method mitigates the problem of redundant node expansion near elongated obstacle regions. The proposed method first uses the JPS algorithm to extract key waypoints, which serve as directional references for guiding node expansion in the Hybrid A-star algorithm. Simulation results show that the DG-Hybrid A-star algorithm, which combines directional guidance, significantly reduces the number of expanded nodes and enhances both path smoothness and planning efficiency compared to the standard Hybrid A-star.

To further improve path planning efficiency and smoothness, a segmented heuristic weighting mechanism was introduced, yielding the Dy-DG-Hybrid A-star algorithm. This mechanism dynamically adjusts the heuristic weight based on path segment type, accelerating search along straight paths while preserving curvature continuity in turning segments. Comparative experiments across various scenarios demonstrate that the Dy-DG-Hybrid A-star algorithm consistently achieves lower cumulative steering angles, fewer expanded nodes, and faster computation times, validating its superior efficiency and robustness in structured agricultural settings.

The proposed Dy-DG-Hybrid A-star algorithm integrates directional guidance with a segment-based dynamic heuristic strategy, which significantly reduces redundant node expansion and computation time while maintaining high path smoothness. However, the algorithm's performance still depends on the quality of the JPS-generated reference path and is primarily applicable to static or semi-structured environments. In future work, we plan to extend the algorithm to handle dynamic obstacles and incorporate motion prediction models, thereby improving its robustness and adaptability in real-world agricultural scenarios.

Author Contributions: Conceptualization, S.Z. and Q.G.; methodology, S.Z. and Q.G.; software, Q.G.; validation, Q.G. and W.L.; formal analysis, Q.G. and W.L.; resources, S.Z.; writing—original draft preparation, Q.G.; writing—review and editing, S.Z., H.K., D.W. and H.Z.; visualization, Q.G. and W.L.; supervision, S.Z., H.K., D.W. and H.Z.; funding acquisition, S.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key Research and Development Program of China, via grant number 2018YFD0301300.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article; further inquiries can be directed to the corresponding author. Due to privacy restrictions, the data cannot be publicly provided.

Acknowledgments: We would like to express our gratitude to the Agricultural Machinery Engineering Research and Design Institute of Hubei University of Technology for providing administrative and technical support during the research. We also thank all co-authors for their assistance throughout the preparation of this manuscript.

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

References

1. Friha, O.; Ferrag, M.A.; Shu, L.; Maglaras, L.; Wang, X. Internet of Things for the Future of Smart Agriculture: A Comprehensive Survey of Emerging Technologies. *IEEE-CAA J. Autom. Sin.* **2021**, *8*, 718–752. [\[CrossRef\]](#)
2. Soussi, A.; Zero, E.; Sacile, R.; Trincherio, D.; Fossa, M. Smart Sensors and Smart Data for Precision Agriculture: A Review. *Sensors* **2024**, *24*, 2647. [\[CrossRef\]](#)
3. Yang, X.; Shu, L.; Chen, J.; Ferrag, M.A.; Wu, J.; Nurellari, E.; Huang, K. A Survey on Smart Agriculture: Development Modes, Technologies, and Security and Privacy Challenges. *IEEE-CAA J. Autom. Sin.* **2021**, *8*, 273–302. [\[CrossRef\]](#)
4. Hossain, M.S.; Rahman, M.; Rahman, A.; Kabir, M.M.; Mridha, M.F.; Huang, J.; Shin, J. Automatic Navigation and Self-Driving Technology in Agricultural Machinery: A State-of-the-Art Systematic Review. *IEEE Access* **2025**, *13*, 94370–94401. [\[CrossRef\]](#)
5. Yao, Z.; Zhao, C.; Zhang, T. Agricultural machinery automatic navigation technology. *iScience* **2024**, *27*, 108714. [\[CrossRef\]](#) [\[PubMed\]](#)
6. Reda, M.; Onsy, A.; Haikal, A.Y.; Ghanbari, A. Path planning algorithms in the autonomous driving system: A comprehensive review. *Robot. Auton. Syst.* **2024**, *174*, 104630. [\[CrossRef\]](#)
7. Guo, T.; Sun, Y.; Liu, Y.; Liu, L.; Lu, J. An Automated Guided Vehicle Path Planning Algorithm Based on Improved A* and Dynamic Window Approach Fusion. *Appl. Sci.* **2023**, *13*, 10326. [\[CrossRef\]](#)
8. Tian, Y.; Zhang, J.; Wang, Q.; Liu, S.; Guo, Z.; Zhang, H. Application of Hybrid Algorithm Based on Ant Colony Optimization and Sparrow Search in UAV Path Planning. *Int. J. Comput. Intell. Syst.* **2024**, *17*, 286. [\[CrossRef\]](#)
9. Li, Y.; Jin, R.; Xu, X.; Qian, Y.; Wang, H.; Xu, S.; Wang, Z. A Mobile Robot Path Planning Algorithm Based on Improved A* Algorithm and Dynamic Window Approach. *IEEE Access* **2022**, *10*, 57736–57747. [\[CrossRef\]](#)
10. Ghadiri Nejad, M.; Li, P.; Hao, L.; Zhao, Y.; Lu, J. Robot obstacle avoidance optimization by A* and DWA fusion algorithm. *PLoS ONE* **2024**, *19*, e0302026. [\[CrossRef\]](#)
11. Kumar, A.; Tiwari, S.; Majumder, A. A*-VG algorithm: A hybrid algorithm for the path planning of inspection robots. *J. Braz. Soc. Mech. Sci. Eng.* **2023**, *45*, 386. [\[CrossRef\]](#)
12. Liu, J.; Liu, C.; Han, M.; Wan, Z.; Li, T.; Jia, X. Path planning algorithm for articulated loader based on bidirectional Dubins curve. *Meas. Sci. Technol.* **2025**, *36*, 026309. [\[CrossRef\]](#)
13. Xue, T.; Cao, Y.; Zhao, Y.; Ai, J.; Dong, Y. Hybrid A*-Based Valley Path Planning Algorithm for Aircraft. *Aerospace* **2024**, *11*, 516. [\[CrossRef\]](#)
14. Lian, J.; Ren, W.; Yang, D.; Li, L.; Yu, F. Trajectory Planning for Autonomous Valet Parking in Narrow Environments with Enhanced Hybrid A* Search and Nonlinear Optimization. *IEEE Trans. Intell. Veh.* **2023**, *8*, 3723–3734. [\[CrossRef\]](#)
15. Meng, T.; Yang, T.; Huang, J.; Jin, W.; Zhang, W.; Jia, Y.; Wan, K.; Xiao, G.; Yang, D.; Zhong, Z. Improved Hybrid A-Star Algorithm for Path Planning in Autonomous Parking System Based on Multi-Stage Dynamic Optimization. *Int. J. Automot. Technol.* **2023**, *24*, 459–468. [\[CrossRef\]](#)
16. Qin, Z.; Chen, X.; Hu, M.; Chen, L.; Fan, J. A novel path planning methodology for automated valet parking based on directional graph search and geometry curve. *Robot. Auton. Syst.* **2020**, *132*, 103606. [\[CrossRef\]](#)
17. Guo, Y.; Yao, D.; Li, B.; He, Z.; Gao, H.; Li, L. Trajectory Planning for an Autonomous Vehicle in Spatially Constrained Environments. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 18326–18336. [\[CrossRef\]](#)
18. Yu, X.; Jiang, N.; Wang, X.; Li, M. A hybrid algorithm based on grey wolf optimizer and differential evolution for UAV path planning. *Expert Syst. Appl.* **2023**, *215*, 119327. [\[CrossRef\]](#)
19. Zhang, S.; Jian, Z.; Deng, X.; Chen, S.; Nan, Z.; Zheng, N. Hierarchical Motion Planning for Autonomous Driving in Large-Scale Complex Scenarios. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 13291–13305. [\[CrossRef\]](#)
20. Wu, Y.; Li, X.; Gao, J.; Yang, X. Research on Automatic Vertical Parking Path-Planning Algorithms for Narrow Parking Spaces. *Electronics* **2023**, *12*, 4203. [\[CrossRef\]](#)

21. Yoon, S.; Jung, C.; Shim, D.H. Shape-Aware and G2 Continuous Path Planning Based on Bidirectional Hybrid A* for Car-Like Vehicles. *J. Intell. Robot. Syst.* **2021**, *103*, 39. [[CrossRef](#)]
22. Cui, G.; Yin, Y.; Xu, Q.; Song, C.; Li, G.; Li, S. Efficient Path Planning for Automated Valet Parking: Integrating Hybrid A* Search with Geometric Curves. *Int. J. Automot. Technol.* **2024**, *26*, 243–253. [[CrossRef](#)]
23. Chen, T.; Chen, S.; Zhang, K.; Qiu, G.; Li, Q.; Chen, X. A jump point search improved ant colony hybrid optimization algorithm for path planning of mobile robot. *Int. J. Adv. Robot. Syst.* **2022**, *19*. [[CrossRef](#)]
24. Li, J.; Li, H.; Zeng, Y.; Jiang, R.; Mai, C.; Ma, Z.; Cai, J.; Xiao, B. Research on the local path planning of an orchard mower based on safe corridor and quadratic programming. *Front. Plant Sci.* **2024**, *15*, 1403385. [[CrossRef](#)]
25. Xu, N.; Li, Z.; Guo, N.; Wang, T.; Li, A.; Song, Y. Global Path Planning for Articulated Steering Tractor Based on Multi-Objective Hybrid Algorithm. *Sensors* **2024**, *24*, 4832. [[CrossRef](#)]
26. Dolgov, D. Path Planning for Autonomous Vehicles in Unknown Semi-structured Environments. *Int. J. Robot. Res.* **2010**, *29*, 485–501. [[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.