electronics

MDPI

Article

A Hierarchical Trajectory Planning Algorithm for Automated Guided Vehicles in Construction Sites

Yu Bai [1], Pengpeng Li [2], Zhipeng Cui [3], Peng Yang [3] and Weihua Li [1,4,5,*]

[1] School of Automotive Engineering, Harbin Institute of Technology (Weihai), Weihai 264201, China; 22s130342@stu.hit.edu.cn
[2] School of Astronautics, Harbin Institute of Technology, Harbin 150001, China; 22b918141@stu.hit.edu.cn
[3] China Construction Institute of Advanced Technology, Wuhan 430073, China; 18846774001@163.com (Z.C.); 13122175818@163.com (P.Y.)
[4] State Key Laboratory of Robotics and System, Harbin Institute of Technology, Harbin 150001, China
[5] Yangtze River Delta HIT Robot Technology Research Institute, Wuhu 241060, China
* Correspondence: liweihua@hit.edu.cn

Abstract: Herein, to address the challenges faced by Automatic Guided Vehicles (AGVs) in construction site environments, including heavy vehicle loads, extensive road search areas, and randomly distributed obstacles, this paper presents a hierarchical trajectory planning algorithm that combines coarse planning and precise planning. In the first-level coarse planning, lateral and longitudinal sampling is performed based on road environment constraints. A multi-criteria cost function is designed, taking into account factors such as deviation from the road centerline, shortest path cost, and obstacle collision safety cost. An efficient dynamic programming algorithm is used to obtain the optimal path. Considering nonholonomic constraints of vehicles, eliminating inflection points using improved B-Spline path fitting, and a quadratic programming algorithm is proposed to enhance path smoothness, completing the coarse planning algorithm. In the second-level precise planning, the coarse planning path is used as a reference line, and small-range sampling is conducted based on AGV motion constraints, including lateral displacement and longitudinal velocity. Lateral and longitudinal polynomials are constructed. To address the impact of randomly appearing obstacles on vehicle stability and safety, an evaluation function is designed, considering factors such as jerk and acceleration. The optimal trajectory is determined through collision detection, ensuring both safe obstacle avoidance and AGV smoothness. Experimental results demonstrate the effectiveness of this method in solving the path planning challenges faced by AGVs in construction site environments characterized by heavy vehicle loads, extensive road search areas, and randomly distributed obstacles.

Keywords: hierarchical planning; quadratic programming; constraint sampling; trajectory planning

check for updates

Citation: Bai, Y.; Li, P.; Cui, Z.; Yang, P.; Li, W. A Hierarchical Trajectory Planning Algorithm for Automated Guided Vehicles in Construction Sites. *Electronics* **2024**, *13*, 1080. https://doi.org/10.3390/electronics13061080

Academic Editor: Giuseppe Prencipe

Received: 29 January 2024
Revised: 29 February 2024
Accepted: 5 March 2024
Published: 14 March 2024

Copyright: © 2024 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

Automatic Guided Vehicles (AGVs) are autonomous navigation devices driven by planning algorithms [1–4]. In recent years, in the construction site environment, AGVs have demonstrated their autonomy and have been used to replace manual labor for flexible material transportation tasks, resulting in cost savings and improved work efficiency [5–7]. However, the challenges of heavy vehicle loads, extensive road search areas, and randomly appearing obstacles in construction sites have presented new hurdles for AGV applications in this context, prompting researchers to focus more on path planning techniques tailored to construction site environments [8,9].

Path planning technology is a method that plans an optimal path based on given target point positions and road conditions [10–13]. By reviewing the research on path planning algorithms in new scenarios of road in recent years, two main research approaches have been summarized. Firstly, combining two or more path planning algorithms, leveraging the strengths of each algorithm, to design path planning algorithms tailored to specific

scenarios [14–16]. For instance, in reference [17], a global path planning algorithm based on an improved A* algorithm and a local path planning algorithm based on an improved artificial potential field algorithm were proposed, achieving theoretical validation for precise obstacle avoidance and rapid navigation of indoor mobile robots. Secondly, by improving a known algorithm and designing heuristic functions, cost functions, etc., tailored to specific scenarios to achieve effective planning results [18–21]. For example, in reference [22], a Target-Optimized JPS algorithm was introduced, which constructs search direction priorities based on target point direction vectors and also improves the cost function to obtain shorter paths, reducing computation time.

Dynamic programming techniques based on the greedy approach are well-suited for path planning on large, wide roads [23–25]. For example, in reference [26], a solution was proposed that uses dynamic programming to inspire continuous state spaces and obtain globally optimal path planning. In reference [27], a static obstacle avoidance algorithm based on a grid map was introduced, generating feasible paths by considering the weight of each obstacle and then optimizing these feasible paths through dynamic programming to find several shortest paths. However, these dynamic programming solutions solely focus on finding the shortest path and do not consider the high load conditions of AGVs, resulting in paths without speed information. On the other hand, Frenet sampling planning decouples the lateral and longitudinal aspects of planning and includes speed information in the planned trajectory [28–30]. As shown in reference [31], it uses the global planning path as a reference line and decouples the local trajectory in the Frenet coordinate system both horizontally and vertically. By considering the designed endpoint sampling rules and taking speed planning into account, it finds the optimal trajectory. However, this method has a higher sampling cost in wide road scenarios.

After analyzing the aforementioned works, it was observed that there are currently very few path planning algorithms specifically designed for construction site environments. Existing algorithms struggle to balance high-speed calculation and extensive sampling range. Therefore, this paper proposes a hierarchical trajectory planning method that integrates road environment constraint sampling (coarse planning) and AGV motion constraint sampling (precise planning). This approach leverages the respective strengths of coarse planning and precise planning and tailored to the specific environment of construction sites, designs cost functions.

The rest of this paper is organized as follows. Section 2 introduces the coarse planning method based on road environment constraint sampling. Section 3 discusses the precise planning algorithm based on motion constraint sampling, and its integration with the coarse planning algorithm to form a hierarchical trajectory planning algorithm. Section 4 conducts experimental tests and data analysis to verify the practical effectiveness and feasibility of the algorithm. Section 5 provides an overall summary.

2. Coarse Planning Algorithm

In the construction site environment, the extensive road search area requires the collection of a large amount of data for both lateral and longitudinal sampling, creating a significant computational burden. To address the challenge of enabling AGVs to quickly compute an accurate, safe, and efficient planning path, this section introduces a dynamic programming algorithm based on the greedy approach. By considering safety, shortest path, and other metrics, a multi-objective evaluation function is designed to efficiently obtain the optimal path. Nonholonomic vehicle constraints are also taken into account, with interpolation at the starting and ending points and the use of the B-Spline algorithm to fit the optimal path and eliminate sharp turns. Given the high load of AGVs, which requires road smoothness, the fitted path is further improved through quadratic programming.

2.1. Lateral and Longitudinal Sampling under Road Environment Constraints

The sampling for coarse planning is based on the S-L coordinate system of the road, with road environment constraints taken into account. It considers road boundaries, the

size and positions of obstacles, and the road centerline for sampling. Laterally, uniformly distributed sampling points from the road's centerline towards both sides at a fixed interval of Δl until reaching the road boundary L_a. Longitudinally, it uniformly distributes sampling points along the road centerline with a fixed Δs, setting a fixed sampling length Sa. Sampling points are filtered based on the size and position of obstacles, using the obstacle's center point as the origin and extending outward by dobs to remove sampling points near the obstacle, improving search efficiency. In scenarios with multiple obstacles, this sampling method significantly reduces computational complexity. Additionally, before planning, if it is determined that there are no available sampling points in the s-direction due to obstacles, it can quickly identify an impassable path and initiate a stopping operation, thus enhancing the safety of the AGV. The specific sampling process is illustrated in the following figure (Figure 1). The blue points represent the sampling points, and the red point represents the target points.

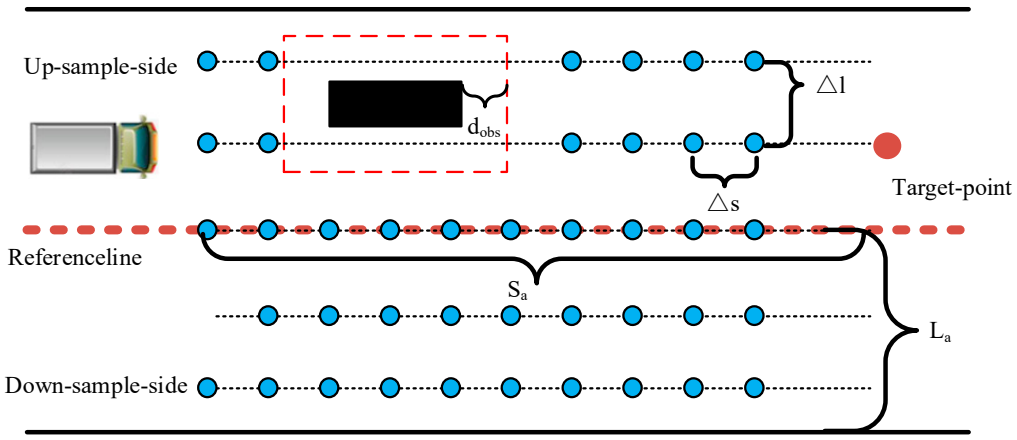


Figure 1. Sampling based on road environment constraints.

2.2. Selecting the Optimal Path

Traditional dynamic programming algorithms focus solely on finding the shortest path [32]. However, in the construction site environment, we besides considering path length, it is essential to take into account the influence of obstacles in the construction site and keep the path as close to the road centerline as possible. Then we can accurately assess the cost between two nodes. This paper proposes a multi-objective evaluation function to evaluate the cost between nodes. In addition to the cost of the shortest path, we introduce costs related to deviation from the reference line and obstacle collision safety. These three aspects are considered to provide a diversified evaluation of a path. The implementation method is as follows.

Linear interpolation is performed between two points on adjacent s-layers to obtain several interpolated points. The cost of the shortest path is determined by calculating the cumulative distance between all interpolated points on the s-layers, providing the length cost of the planned path. The cost of deviation from the reference line is characterized by the cumulative sum of l^2 values of all interpolated points, taking advantage of the properties of the Frenet coordinate system. For obstacle safety cost, a segmented calculation approach is employed. If the distance between the planning point and the obstacle is greater than 2 m, it is considered sufficiently safe, and no cost is calculated. If the distance is less than 0.5 m, it is considered impassable, resulting in an infinite cost. If the distance falls within the range of 0.5 m to 2 m, the cost is represented by the sum of the squared distances between the interpolated points and the obstacle.

Given the coordinates of interpolated points between two points as $(s_1, l_1), (s_2, l_2) \ldots (s_n, l_n)$ and the coordinates of obstacles as $(s_{obs1}, l_{obs1}), (s_{obs2}, l_{obs2}) \ldots (s_{obsn}, l_{obsn})$, the cost equation for the path between these two points is calculated as follows:

$$\begin{cases} cost_1 = \sum_{i=1}^{n-1} \sqrt{(s_{i+1} - s_i)^2 + (l_{i+1} - l_i)^2} \\ cost_2 = \sum_{i=1}^{n} c_i {l_i}^2 \\ cost_3 = \begin{cases} 0 & (|dis_{obsi}| > 2) \\ \sum_{i=1}^{n} c_i dis_{obsi}^2 & (2 > |dis_{obsi}| > 0.5) \\ \infty & (0.5 > |dis_{obsi}|) \end{cases} \\ c_i = (d)^{s_i - s_n}, 0 < d < 1 \\ dis_{obsi} = \sqrt{(s_{obsi} - s_i)^2 + (l_{obs} - l_i)^2} \\ cost = k_1 cost_1 + k_2 cost_2 + k_3 cost_3 \end{cases}, \quad (1)$$

where $cost_1$ represents the cost associated with the shortest path, $cost_2$ represents the cost associated with deviation from the reference line, $cost_3$ represents the cost associated with ensuring safety against collisions with obstacles, and dis_{obsi} represents the distance of the ith point from the nearest obstacle. c_i is an increasing function, and we are more concerned about the final operating state of the vehicle. Therefore, the proportion of trajectory points at the beginning of the planning is not as high as near the endpoint.

As shown in the Figure 2a, each sampling point represents a node, and nodes with the same s-coordinate are grouped into the same layer. Starting from the last layer, we traverse all the points sampled in the l-direction for the same s layer, recording the minimum cost value for each point to reach the endpoint after visiting all points in the next layer in the s-direction. The cost equation is as follows:

$$cost_{min} = min\{cost_{p_1} + c_1, cost_{p_2} + c_2, \ldots\ldots cost_{p_i} + c_i\}, \quad (2)$$

where $cost_{pi}$ represents the cost from the current node to the next-level target point, and c_i represents the minimum cost from the next-level node to the endpoint.

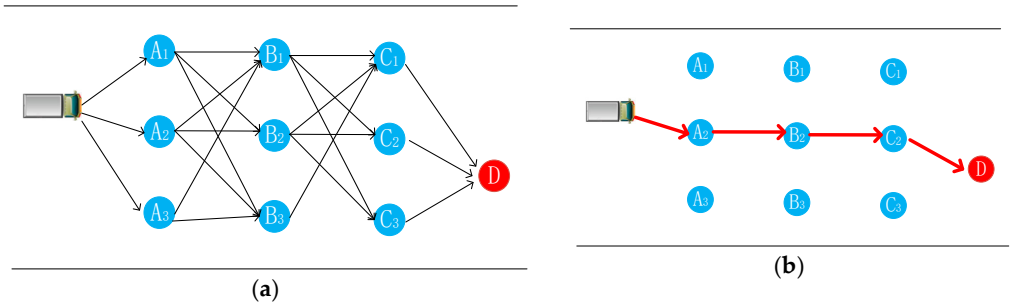


Figure 2. Optimal path selection based on greedy algorithm principles. (**a**) Path Searching Based on Greedy Algorithm Principles. (**b**) Optimal Path.

It continues until reaching the first layer in the s-direction, finding the optimal path with the minimum cost for the vehicle to reach point D, as shown in Figure 2b.

2.3. Curve Fitting and Optimization

As shown in Figure 3a below, in order to ensure planning efficiency, the chosen values of Δs and Δl in dynamic programming cannot be excessively small. This results in relatively large spacing between the output path points generated by the algorithm. Consequently, the planned path often exhibits significant inflection points, lacking the desired smoothness. Thus, the resultant path is typically composed of line segments.

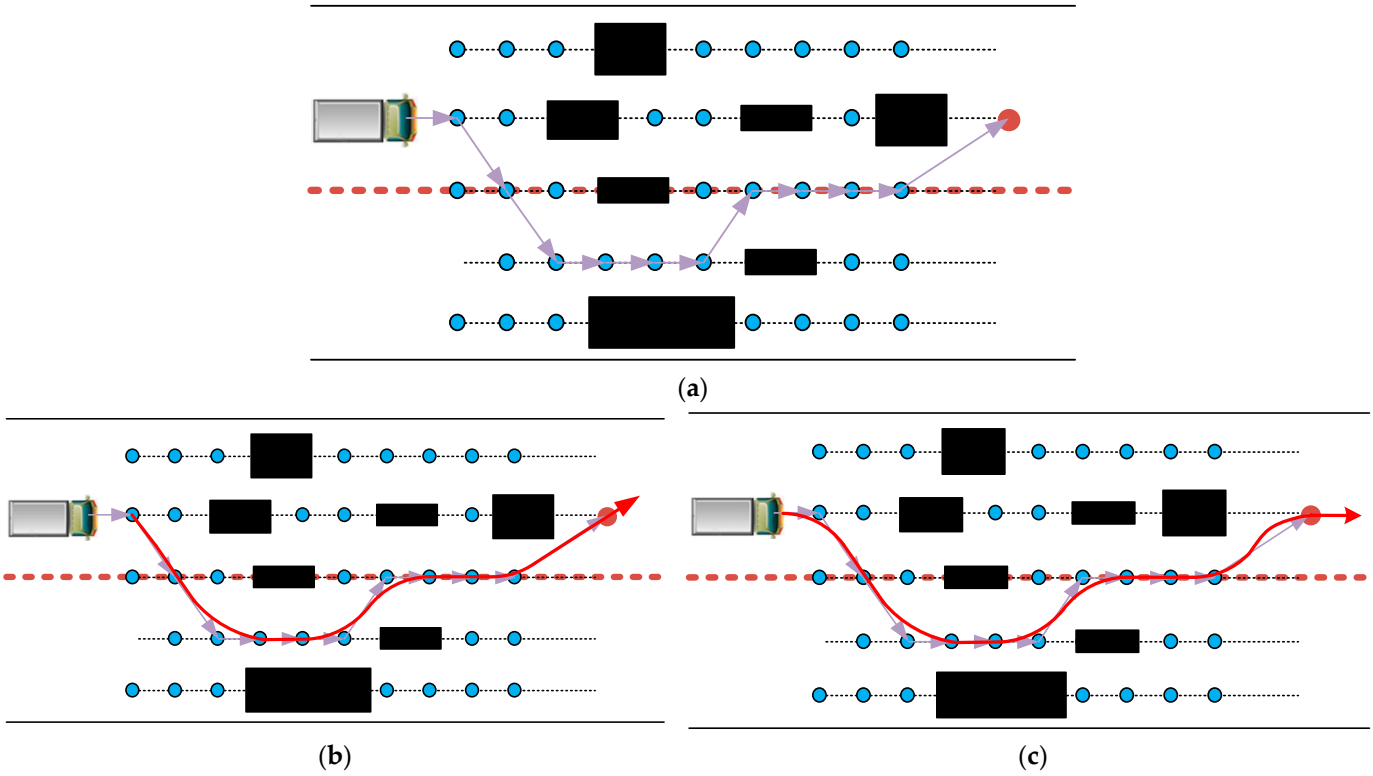


Figure 3. Comparison of fitting effects. (**a**) Unfitted. (**b**) Traditional Fitting. (**c**) B-Spline Fitting after Interpolation.

We certainly find it imperative to engage in curve fitting. However, if we exclusively apply curve fitting solely to the sampled points, we overlook the non-holonomic constraints posed by the initial and final velocity directions of the AGV. Consequently, while the fitted curve adeptly mitigates the issue of inflection points, it falls short of ensuring that the vehicle's velocity direction aligns with the planned trajectory direction upon reaching the endpoint. Alternatively, at the starting point, the vehicle may encounter challenges in tracking the planned curve due to velocity direction issues, as depicted in Figure 3b below.

Utilizing the fitting characteristics of B-Splines, this paper proposes a method that, while adhering to the traditional properties of B-Spline fitting, interpolates the coarse planning path to ensure that the tangent direction at the start point of the planned path aligns with the initial velocity direction of the vehicle, and the tangent direction at the end point matches the terminal velocity direction of the vehicle, as shown in the following Figure 3c.

As depicted in Figure 4, the blue points are selected based on the multi-objective evaluation function, the yellow points are interpolation points, and the red points is the endpoints. Since the starting point of the B-Spline fitting curve lies on the line connecting the first and second sampled points and is tangent to that line, to ensure that the tangent direction at the starting point of the B-Spline fitted curve aligns with the vehicle's velocity direction, a new virtual starting sample point, denoted as 1.1, is inserted after sample point 1. The insertion is made at a distance of d_a. Similarly, to ensure that the tangent direction at the endpoint of the fitted B-Spline curve corresponds to the vehicle's final velocity direction, a new virtual endpoint sample point, denoted as N.1, is inserted before the last sample point. This insertion is made at a distance of D_a along the extension of the vehicle's final velocity direction. The B-Spline fitted curve's endpoint lies on the line connecting the last sample point and the second-to-last sample point and is tangential to that line.

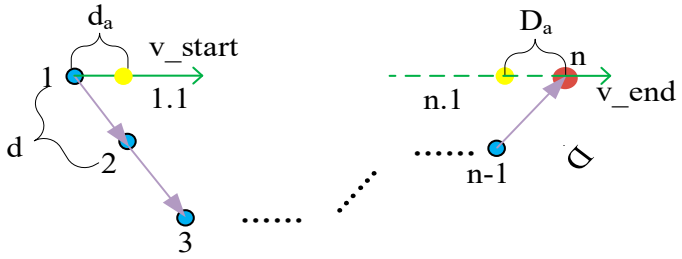


Figure 4. Improved B-Spline interpolation.

The equation for calculating D_a and d_a are as follows:

$$\begin{cases} D_a = D/n_1 \\ d_a = d/n_2 \end{cases}, \tag{3}$$

where d represents the distance between sample points 1 and 2, D represents the distance between sample points n − 1 and n. n_1 and n_2 are interpolation weighting coefficients.

To ensure the quality of the fitted curve while considering the velocity direction, it is important to choose appropriate values for n_1 and n_2. Taking the interpolation position of n.1 as an example, as illustrated in Figure 5, different values of n_2 can result in significant differences in the fitting performance. Factors such as the alignment of the fitted curve with the original path points, the length of the fitted curve, and its smoothness vary with different values of n_2. Therefore, it is necessary to determine the optimal values for n_1 and n_2 to ensure the best fitting results.

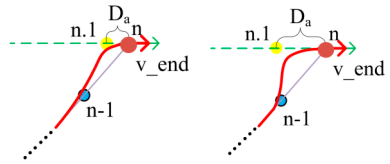


Figure 5. The fitting effect of different n_2.

As Section 2.1 involved the filtering of sampling points close to obstacles based on road environment sampling, and Section 2.2 considered obstacle distance as one of the cost functions for optimal path planning, there is no need to consider the obstacle cost function in the small-scale optimization during curve fitting. In contrast, our primary focus is on the smoothness, shortest path, and alignment with sampling points of the planned path. Therefore, considering the original coordinates as P_{k_ref} and the optimized coordinates as P_k we can improve Equation (2) as follows:

$$\begin{cases} P_k = \begin{bmatrix} x_k \\ y_k \end{bmatrix} \\ cost_a = v_1 \sum_{k=1}^{n-2} \left\| 2P_k - P_{k-1} - P_{k+1} \right\|_2^2 \\ cost_b = v_2 \sum_{k=0}^{n-2} \left\| P_{k+1} - P_k \right\|_2^2 \\ cost_c = v_3 \sum_{k=0}^{n-1} \left\| P_k - P_{k_ref} \right\|_2^2 \end{cases}, \tag{4}$$

where $cost_a$ represents the smoothness cost, $cost_b$ represents the length cost, and $cost_c$ represents the deviation cost relative to the original points. v_1, v_2, and v_3 are cost coefficients.

The value ranges for n_1 and n_2 are defined as follows:

$$\begin{cases} N_1 = [1, 2, 3 \dots n_z] \\ N_2 = [1, 2, 3 \dots n_z] \end{cases}, \tag{5}$$

By traversing N_1 and N_2 at different values in Equation (5), the cost calculated according to Equation (4) will be obtained. The optimal values for n_1 and n_2 will be selected based on the minimum cost value. After obtaining the best n_1 and n_2, the entire path from the coarse planning phase will undergo quadratic programming using the same parameter criteria as defined in Equation (4).

The variable set P representing the entire path is defined as follows:

$$P = [x_1 \quad y_1 \quad \dots \quad x_n \quad y_n], \tag{6}$$

Based on Equations (4) and (6), the expressions for the smoothness cost W_1, length cost W_2, and similarity cost W_3 can be derived as follows:

$$\begin{cases} A_1^T = \begin{bmatrix} 1 & 0 & & & & & & \\ 0 & 1 & 1 & 0 & & & & \\ -2 & 0 & 0 & 1 & 1 & 0 & & \\ 0 & -2 & -2 & 0 & 0 & 1 & & \\ 1 & 0 & 0 & -2 & -2 & 0 & & \\ 0 & 1 & 1 & 0 & 0 & -2 & & \\ & & 0 & 1 & 1 & 0 & & \\ & & & & 0 & 1 & & \\ & & & & & & \mathrm{O} & \\ & & & & & & & \mathrm{O} \end{bmatrix}, A_2^T = \begin{bmatrix} 1 & 0 & & & & & \\ 0 & 1 & & & & & \\ -1 & 0 & 1 & 0 & & & \\ 0 & -1 & 0 & 1 & & & \\ & & -1 & 0 & & & \\ & & 0 & 1 & & & \\ & & & & \mathrm{O} & & \\ & & & & & \mathrm{O} & \end{bmatrix}, \\ R = \begin{bmatrix} -2x_{1\mathrm{r}} \\ -2y_{1\mathrm{r}} \\ \vdots \\ -2x_{nr} \\ -2y_{nr} \end{bmatrix}, A_3^T = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & \mathrm{O} & \\ & & & & 1 \end{bmatrix}, \begin{cases} W_1 = w_1 P^T A_1^T A_1 P \\ W_2 = w_2 P^T A_2^T \mathrm{A}_2 P \\ W_3 = w_3 P^T A_3^T A_{\ 3} \mathrm{P} + \mathrm{R^T P} \end{cases} \end{cases} \tag{7}$$

With the above derivations, the cost function can be modified as follows:

$$cost = P^T(W_1 \mathrm{A}_1^T \mathrm{A}_1 + W_2 \mathrm{A}_2^T \mathrm{A}_2 + W_3 \mathrm{A}_3^T \mathrm{A}_3) P + W_3 H^T P, \tag{8}$$

The quadratic programming equation is as follows:

$$x^T(\frac{H}{2})x + f^T x, \tag{9}$$

Combining Equations (7)–(9), we can obtain:

$$\begin{cases} H = 2(w_1 A_1^T A_1 + w_2 A_2^T A_2 + w_3 A_3^T A_3) \\ f^T = w_3 H^T \end{cases}, \tag{10}$$

To prevent excessive impact on the B-spline fitting due to the quadratic programming, interval tolerances are introduced. The tolerance set B is defined as follows:

$$\begin{cases} \left| P_k - P_{k-ref} \right| < B \\ B = [b_1, b_2 \dots b_{n1} \dots b_{n2} \dots b_n] \\ b_1, b_2 \dots b_{n1} = b_{n2} \dots b_n = b_{\min} \\ b_{n1+1} \dots b_{n2-1} = b_{\max} \end{cases}, \tag{11}$$

This is performed to prevent the quadratic programming from adversely affecting the fitting of the improved B-spline. With this, Equations (6)–(11) are incorporated into the quadratic programming solver for iterative solution, yielding the optimal coarse planning path.

3. Precise Planning Algorithm

The coarse planning phase is used to obtain a high-quality planning path with relatively low computational resources. However, the planned path does not include velocity information. In construction environments, AGVs often carry heavy loads, and planning without velocity information can be highly dangerous under high-load conditions. Additionally, the B-Spline fitting and quadratic optimization used in coarse planning make minor adjustments to the path points obtained from dynamic planning. Therefore, collision safety detection is still necessary. In this section, we propose a precise planning algorithm based on the results of coarse planning. This algorithm uses the coarse planning results as a reference line and samples in the longitudinal direction based on velocity and in the lateral direction based on lateral displacement. It considers additional factors like jerk to ensure the smoothness of AGV motion under velocity planning. Moreover, it includes collision detection to achieve a safe and high-quality trajectory planning solution. Implementing the precise algorithm on the basis of coarse planning significantly reduces the sampling count, thereby addressing the drawback of high computational load associated with precise planning.

3.1. Construction of Trajectory Polynomials under Motion Constraint Sampling

In the Frenet coordinate system, the planning problem involves a total of three dimensions, namely (s, d, t). This high-dimensional optimization problem can be decoupled into two independent optimization problems in the lateral (d, t) and longitudinal (s, t) directions, which are sampled and processed separately.

In accordance with the hardware specifications, environmental constraints, safety considerations, and other factors related to the AGV, there are boundary constraints on the lateral displacement (d), velocity (v), acceleration (a), jerk (j), and planned curvature (k). The lateral and longitudinal sampling must adhere to the following motion constraints.

$$\begin{cases} d = \left\{ d \mid \begin{array}{c} d \in [d_{last} - v_{\max}\Delta t, d_{last} + v_{\max}\Delta t] \\ d \in [-d_{\max}, d_{\max}] \end{array} \right\} \\ v = \left\{ v \mid \begin{array}{c} v \in [v_{last} - a_{\max}\Delta t, v_{last} + a_{\max}\Delta t] \\ v \in [-v_{\max}, v_{\max}] \end{array} \right\} \\ a = \left\{ a \mid \begin{array}{c} a \in [a_{last} - j_{\max}\Delta t, a_{last} + j_{\max}\Delta t] \\ a \in [-a_{\max}, a_{\max}] \end{array} \right\} \\ j \in [-j_{max}, j_{max}] \\ k \in [-k_{max}, k_{max}] \\ \Delta t \in [T_{min,} T_{max}] \end{cases}, \tag{12}$$

Lateral sampling involves the vehicle starting from the reference line and sampling on both sides of the vehicle at intervals of Δd while adhering to the motion constraints defined in Equation (12). This process continues until sampling reaches the road boundary. After completing this lateral sampling, the next step is to sample in the time domain with intervals of Δt. When the sampling point D_1, initial time t_0, and planning time t_1 are determined, a fifth-degree polynomial is constructed for the relationship between time (t) and lateral displacement (d) as follows:

$$d(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5, \tag{13}$$

The fifth-degree polynomial for lateral sampling can be solved given the known initial position $D_0 = \left[d_0, \dot{d}_0, \ddot{d}_0\right]$, final position $D_1 = \left[d_1, \dot{d}_1, \ddot{d}_1\right]$ and time instants t_0 and t_1.

Under the conditions specified in Equation (12), the longitudinal (s) sampling is slightly different. Instead of using Δs for longitudinal sampling, it uses a given target velocity v_t. Setting Δv with v_t as the reference and sampling n_v velocities on both sides to obtain v can be expressed as follows:

$$v = v_t + n \triangle v, n \in [-n_v, n_v], \tag{14}$$

Subsequently, time t is sampled at intervals of Δt. Upon completing the sampling of both velocity and time, it is tantamount to having accomplished the sampling of s. The advantage of this approach is that, since the sampled values represent velocities, it facilitates the planning of velocity profiles. Consequently, a quartic polynomial concerning velocity and time is constructed as follows.

$$v(t) = b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4, \tag{15}$$

Analogous to lateral sampling planning, this quartic polynomial can be resolved, thereby concluding the solution process for the current planning iteration.

Upon the resolution of both lateral and velocity sampling polynomials, an assortment of trajectories is generated through the permutation and combination of each lateral sampling and longitudinal velocity sampling under the same planning timeframe. The series of trajectory clusters generated by sampling are shown in the following figure (Figure 6). The dark blue line represents the centerline of the road, the purple line represents the sampling trajectory, and the green points represent the trajectory points selected for planning.

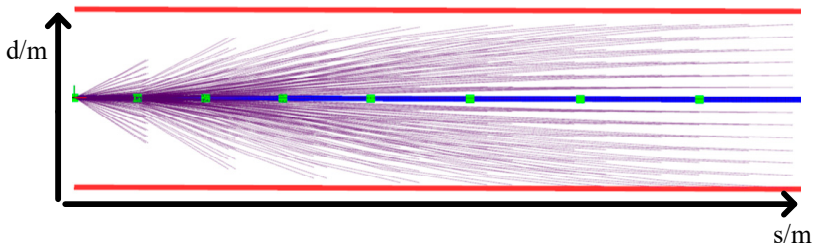


Figure 6. Trajectory cluster.

3.2. Selection of the Optimal Trajectory

The evaluation function for precise planning is set up from three aspects, driving stability, conformity with the reference line, and convergence effect of the desired speed. Paths with potential collision risks are excluded through collision detection, thus obtaining the Optimal Trajectory. Driving stability is characterized by the absolute values of the longitudinal and lateral jerk throughout the vehicle's planned trajectory, ensuring that high-load AGVs can smoothly operate while avoiding obstacles at construction sites or during acceleration and deceleration. The degree of conformity to the desired path is represented by the absolute value of the d-coordinate in the Frenet frame of the planned path points, aiming to align the AGV as closely as possible with the desired path tracking. The convergence effect on the desired speed is represented by the squared difference between the planned speed and the desired speed, ensuring that the vehicle can converge to the desired speed relatively quickly. The specific calculation equation is as follows:

$$\begin{cases} cost_j = \sum_{i=0}^{n} \left(\left| \dddot{d}_i \right| + \left| \dddot{s}_i \right| \right) \\ cost_d = \sum_{i=0}^{n} d_i \\ cost_s = \left(v_{target} - v_{end} \right)^2 \end{cases}, \tag{16}$$

where $cost_j$ represents the cost of stability, $cost_d$ represents the cost of conformity with the desired path, and $cost_s$ represents the cost of conformity with the desired speed.

The calculation equation is as follows (c_1, c_2, and c_3 as cost coefficients).

$$cost = c_1 cost_j + c_2 cost_d + c_3 cost_s, \tag{17}$$

During actual testing, it was observed that when facing obstacles at construction sites, the vehicle might frequently encounter situations where it is positioned between two trajectories with similar costs. This leads to frequent replanning of the vehicle's trajectory, resulting in variability in the trajectory planning and causing instability in the vehicle's movement, as illustrated in the following Figure 7. Due to similar costs, the AGV may frequently switch optimal trajectories among the following three trajectories.

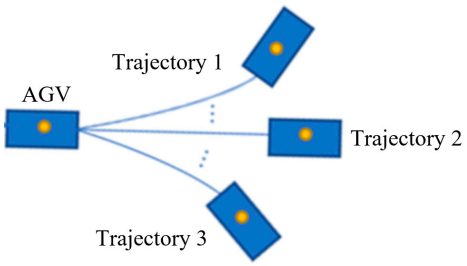


Figure 7. Frequent alteration of trajectory planning.

To address this phenomenon, a new cost function is introduced as follows:

$$cost_l = c_4 \sum_{i=0}^{n} (d_{1_i} - d_{1_last_i})^2, \tag{18}$$

where $cost_l$ represents the similarity between the starting point of the current planned path and that of the previous frame's planned path.

Under the premise that other paths do not significantly differ from the previous frame's planned path, to ensure vehicular stability, a preference is given to planning results similar to the previous frame's plan. This approach effectively circumvents the issue of repetitive planning for AGVs under critical cost function scenarios. Consequently, the comprehensive evaluation equation for precise planning is as follows.

$$cost = c_1 cost_j + c_2 cost_d + c_3 cost_s + c_4 cost_l, \tag{19}$$

Although the coarse planning algorithm considers obstacle distances and sets a safe distance, to prevent minor adjustments in the planned trajectory due to curve fitting in Section 2.3 and precise planning in Section 3.1, collision detection is still necessary to ensure the safety of the trajectory. All trajectories are sorted from lowest to highest cost according to Equation (18). The obstacle collision detection starts with the trajectory having the lowest cost, and the detection process is illustrated in the following figure (Figure 8). The black rectangle represents obstacles, and the red circle represents the minimum circumscribed circle of obstacles.

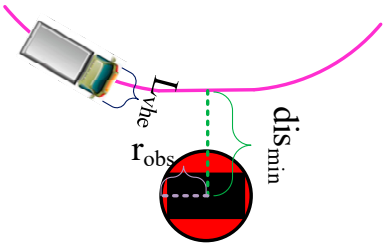


Figure 8. Collision detection.

By iterating through all points of the trajectory output from the precise planning process, the trajectory point closest to the obstacle is identified, and the minimum distance

dis_{save} is calculated. Based on the dimensions of the obstacle obtained from the perception module, the minimum circumscribed circle radius r_{obs} of the obstacle is determined. L_{veh} represents the width of the vehicle's front. A trajectory is deemed safe if it satisfies the following condition; otherwise, the next trajectory is examined. The safety determination condition is as follows:

$$dis_{min} > \frac{L_{veh}}{2} + r_{obs} + dis_{save}, \tag{20}$$

If the trajectory is found to be free of collisions with obstacles, it is selected and outputted. Otherwise, the next trajectory is examined until a suitable one is identified. If no trajectory meets the safety requirements, the vehicle will stop and trigger an alarm.

3.3. The Hierarchical Trajectory Planning Algorithm Based on Coarse and Precise Planning

Section 2 describes a coarse planning algorithm based on a greedy approach. Although it is computationally simple and efficient, ensuring smooth path processing, this algorithm does not take into account velocity planning. In construction site environments, where obstacles may appear randomly, the absence of velocity planning is considerably unsafe. High-load AGVs must consider the impact of changes in velocity.

The precise planning algorithm described in this chapter, while incorporating velocity planning, still requires a substantial amount of sampling calculations for wide roads and larger environments at construction sites. It necessitates frequent construction and solving of quintic and quartic polynomials. Moreover, the decoupled lateral and longitudinal planning, coupled through time, demands permutation and combination, resulting in a significant computational load.

This paper proposes a hierarchical trajectory planning algorithm for construction sites, and the specific implementation process is as follows:

(1) According to Section 2.1, under road environmental constraints, starting from the centerline of the road, a smaller Δs and Δl for sampling, remove unnecessary sampling points based on obstacles to improve sampling efficiency.
(2) According to Section 2.2, consider the cost of path length, obstacle collision safety, and deviation from the reference line to calculate the cost function between nodes in adjacent columns. Then, based on greedy thinking, find the optimal path.
(3) According to Section 2.3, the optimal path point is fitted and optimized. Using the improved B-Spline for fitting, eliminate inflection points while considering the non-holonomic constraints of the VGA at the starting and ending points. A quadratic programming equation is constructed considering smoothness, shortest path, and similarity with the original path. Then, the coarse planning has been completed as shown in Figure 9.
(4) According to Section 3.1, horizontal and vertical sampling are carried out based on the motion constraints of AGV, and lateral fifth and horizontal fourth degree polynomials are constructed separately. The reference line of the second layer planning algorithm no longer uses the road centerline as the reference line but uses the result of coarse planning as the reference line. A smaller sampling width L is used to perform small-scale horizontal sampling on the results of coarse planning. Combine horizontal and lateral trajectories at the same sampling time to form trajectory bundles.
(5) According to Section 3.2, from driving stability, conformity with the reference line, the rate of convergence from the expected speed and the degree of conformity with the previous trajectory is used to calculate the cost for each trajectory. Starting from the trajectory with the lowest cost, collision detection is performed sequentially. Output is the trajectory with the lowest cost and satisfying collision detection. Then, the Precise planning algorithm has been completed as shown in Figure 10 (The blue line represents the sampling boundary, the red line represents the result of coase planning as a reference line).

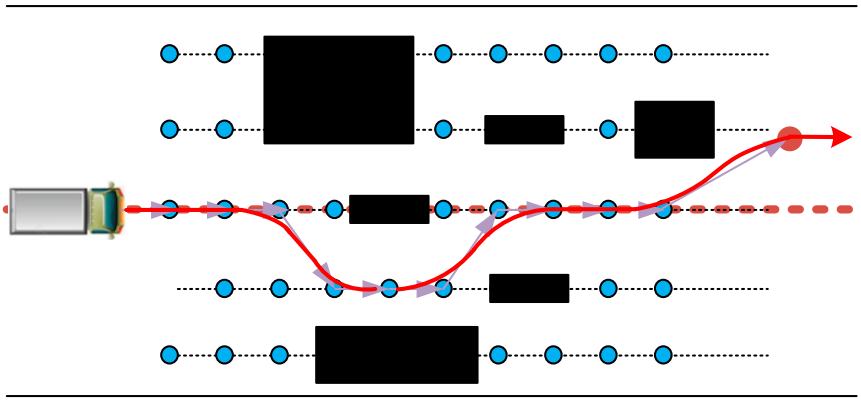


Figure 9. Coarse planning.

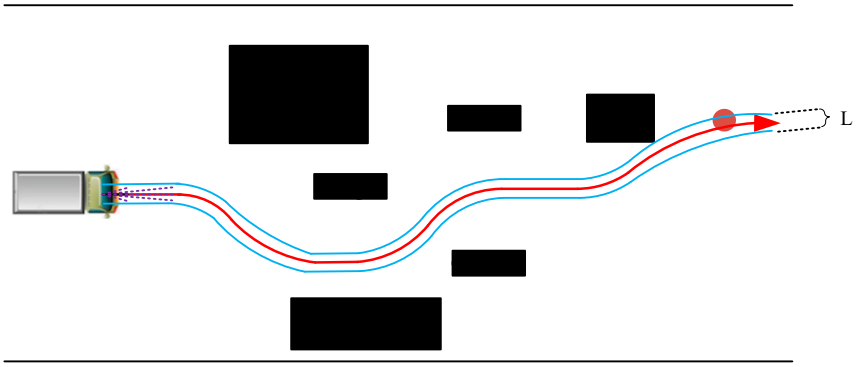


Figure 10. Precise planning.

The hierarchical planning algorithm process is depicted in the Figure 11.

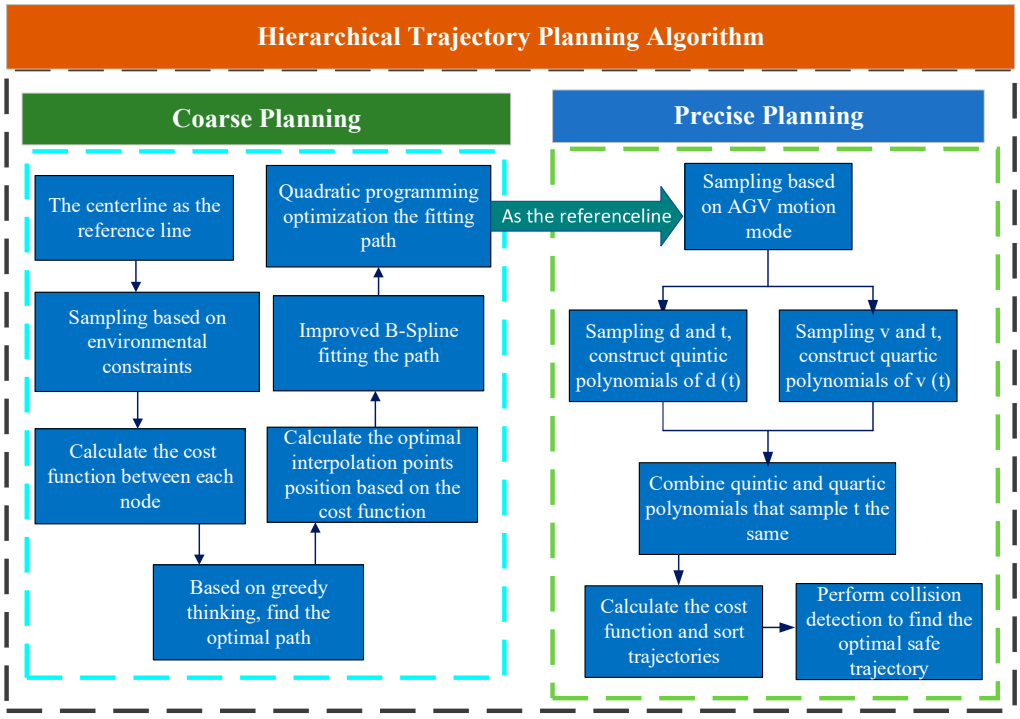


Figure 11. Algorithm flowchart.

In terms of operational efficiency, coarse planning utilizes the idea of greedy algorithms, reducing the time complexity from $O(n^2)$ in traditional dynamic programming algorithms to $O(n)$. Precise planning relies on the rough solutions of coarse planning, with only a number of three to five horizontal samples. Therefore, the time complexity of precise planning is reduced from $O(n^2)$ to $O(n)$, resulting in a time complexity of $O(n)$ for the entire hierarchical planning algorithm. It can meet the demand for algorithm efficiency for the wide road area in the construction site environment.

In terms of safety, the sampling points near obstacles are filtered in Section 2.1. As shown in Equation (1) of Section 2.2, the cost function of coarse planning requires the planned trajectory to be far away from obstacles and set a minimum distance of 0.5 m. The collision detection conducted in Section 3.2 prevents potential safety hazards due to the curve fitting optimization in Section 2.3, ensuring the safety of the planned path.

In terms of trajectory smoothness, the improved B-spline fitting takes into account the non-holonomic constraints of AGV, eliminates trajectory inflection points and smooths and optimizes the path through quadratic programming. And precise planning ultimately obtains a smooth trajectory based on indicators such as lateral and longitudinal jerk.

4. Experimental Validation

In this section, experiments were conducted using an AGV built by Weihai Tiante Intelligent Technology Co., Ltd. (Weihai, China) as the experimental platform, as shown in Figure 12. The exact specifications of the AGV are presented in Table 1. Consequently, the velocity planning range should be within 0 to 1.5 (m/s), while the maximum curvature allowed is specified as 1 m^{-1}.



Figure 12. AGV.

Table 1. The exact specifications of the AGV.

Params	Value
AGV Dimensions	2000 × 1000 × 950 mm
AGV Mass	≤800 kg
Max Load	1000 kg
Wheel Specifications	165/55R13
Max Speed	1.5 m/s
Min Turning Radius	1000 mm
Max Climbing Slope	20°
Braking Distance	<20 mm

The experiments were carried out in the construction site environment at Guangzhou Baiyun Airport, as shown in the following figure. The operating road within this environment constitutes a temporary or semi-permanent structure. The width of the road is typically ranging from 6 to 10 m or wider. In the experiment, the boxes were used to simulate the obstacles encountered by AGV, as shown in the following figure (Figure 13).



Figure 13. Experimental environment and obstacles.

Experiments involved variations in vehicle positions, target point locations, initial velocities, target velocities, and other parameters. These experiments were designed to test and validate the effectiveness of the hierarchical planning proposed in this paper. The stability and reliability of the algorithm was also verified. Representative experimental results were selected for detailed analysis and illustration.

4.1. Experimental System

As shown in Figures 14–17. The industrial computer used in this experiment is the NVIDIA AGX Orin 64G. The depth camera (Stereolabs Technologies Inc., San Francisco, CA, USA; Beijing, China) was employed as the obstacle perception sensor, while a laser radar (DJI, Shenzhen, China) and INS570D inertial navigation components (Guangdong Daoyuan Electronic Technology Co., Guangzhou, China) served as localization sensors. These sensors were chosen for experimental testing of the planning algorithm in this paper, under the condition that the perception algorithms meet all the requirements of the experiment.



Figure 14. JETSON AGX ORIN.



Figure 15. Livox-MID-70.



Figure 16. ZED.



Figure 17. INS570D.

Detailed information about industrial computers and sensors are shown in the tables below (Tables 2–5).

Table 2. JETSON AGX ORIN information.

Ai	275TOPS
GPU	2048-core NVIDIA Ampere architecture GPU with 64 Tensor Cores
CPU	12-core Arm⊙ Cortex⊙. A78AE v8.2 64-bit CPU 3MBL2+6MBL3
OS	ubuntu 20.04

Table 3. Livox-MID-70 information.

Wavelength	905 nm
Blind Area	0~0.05 m
FOV	70.4°
Angle Error	<0.1°

Table 4. ZED information.

Resolution Ratio	4416 × 1242(2k)@15fps
Depth Range	0.2–20 m
Data Rate	400 Hz
Postural Drift	0.35%; 0.005°/m

Table 5. NS570D information.

Accuracy	Position Drift (1 km or 2 min) 0.20%
	Heading Drift (1 min) 0.15°
Position (RMS)	1.5 m, 2 cm + 1 ppm (RTK)
Speed (RMS)	0.03 m/s

As shown in Figure 18, the INS570D module, the Livox-MID-70 and the ZED, respectively, communicate with the industrial computer through the RS485 serial port, the Ether Net, and the USB 3.0. The industrial computer runs the perception and planning algorithm, sends the motion commands to the AGV through the Controller Area Network (CAN).

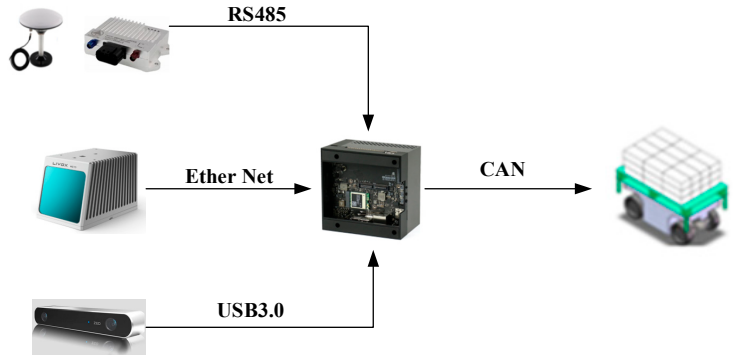


Figure 18. Device communication.

As shown in Figure 19 there are four ZED cameras installed around the AGV, a LiDAR is installed before and after the AGV, and two are installed on each side. The industrial computer and INSS570D are integrated inside the AGV, and the INS570D antenna is placed diagonally in front and behind the AGV.

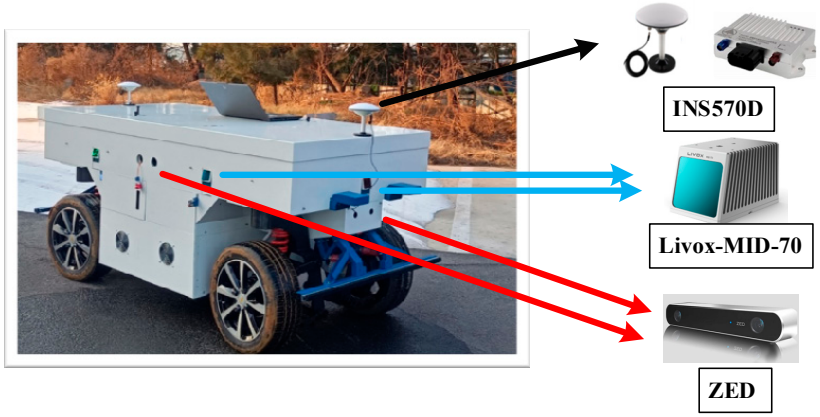


Figure 19. Device installation.

The sensors used in the perception system in this paper (including ZED, LiDAR, inertial navigation, etc.) were pre calibrated using their official calibration methods and procedures before the experiment. Please refer to the official websites of each sensor for details. During the experiment, a large number of tests have shown that the accuracy of obstacle recognition and positioning can reach the centimeter level.

4.2. AGV Tracking Control for Experiments

The kinematic model of AGV is shown in the Figure 20. AGV is a four-wheel steering model, which regards all movements of AGV to be considered as circular movements around point O′ on the plane of the AGV body. Linear motion and spin are its special limiting motion states. When all components of the AGV, including the wheels, move in a circular motion around a certain point, the angular velocity ω is the same, and the linear velocity is perpendicular to the center of the circle. We set the wheelbase of the AGV to L_{agv} and the wheel-base to H_{agv}, The speeds of the four drive motors are V_{LF}, V_{RF}, V_{LB}, V_{RB}. The angle of the four steering motors are δ_{LF}, δ_{RF}, δ_{LB}, δ_{RB}. The angle between AGV speed and V_x is δ. The turning radii of the four wheels are R_{LF}, R_{RF}, R_{LB}, R_{RB}. The turning radius of AGV is R.

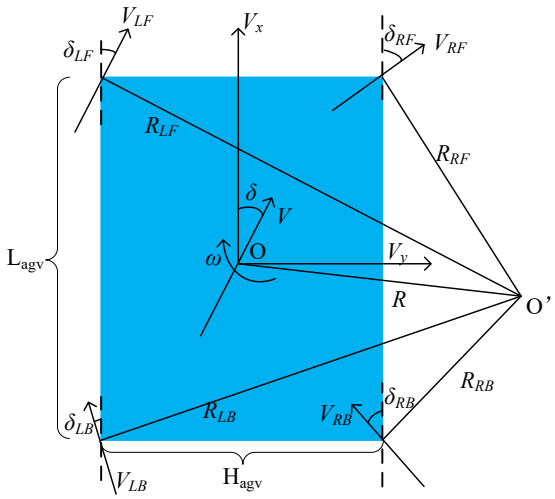


Figure 20. AGV kinematic model.

The V, R, δ can be obtained according to the following equation after obtaining the V_x, V_y and ω.

$$\begin{cases} V = \sqrt{V_x + V_y} \\ \delta = tan(\frac{V_y}{V_x}) \\ R = \frac{V}{\omega} \end{cases} \tag{21}$$

Then, the R_{LF}, R_{RF}, R_{LB}, R_{RB} can be calculated according to the following equation.

$$\begin{cases} R_{LF} = \sqrt{[\frac{L_{agv}}{2} + Rsin(\delta)]^2 + [\frac{H_{agv}}{2} + Rcos(\delta)]^2} \\ R_{LB} = \sqrt{[\frac{L_{agv}}{2} - Rsin(\delta)]^2 + [\frac{H_{agv}}{2} + Rcos(\delta)]^2} \\ R_{RB} = \sqrt{[-\frac{H_{agv}}{2} + Rcos(\delta)]^2 + [\frac{L_{agv}}{2} - Rsin(\delta)]^2} \\ R_{RF} = \sqrt{[-\frac{H_{agv}}{2} + Rcos(\delta)]^2 + [\frac{L_{agv}}{2} + Rsin(\delta)]^2} \end{cases} \tag{22}$$

Then, V_{LF},V_{RF}, V_{LB}, V_{RB} can be obtained according to R_{LF},R_{RF}, R_{LB}, R_{RB}.

$$\begin{cases} V_{LF} = R_{LF}\omega \\ V_{LB} = R_{LB}\omega \\ V_{RF} = R_{RF}\omega \\ V_{RB} = R_{RB}\omega \end{cases} \tag{23}$$

According to the following equation, δ_{LF},δ_{RF}, δ_{LB}, δ_{RB} can be obtained

$$\begin{cases} \delta_{LF} = \frac{\pi}{2} - arctan(\frac{Rcos(\delta)+\frac{H_{agv}}{2}}{Rsin(\delta)+\frac{L_{agv}}{2}}) \\ \delta_{LB} = \frac{\pi}{2} - arctan(\frac{Rcos(\delta)+\frac{H_{agv}}{2}}{-Rsin(\delta)+\frac{L_{agv}}{2}}) \\ \delta_{RF} = \frac{\pi}{2} - arctan(\frac{Rcos(\delta)-\frac{H_{agv}}{2}}{Rsin(\delta)+\frac{L_{agv}}{2}}) \\ \delta_{RB} = \frac{\pi}{2} - arctan(\frac{Rcos(\delta)-\frac{H_{agv}}{2}}{-Rsin(\delta)+\frac{L_{agv}}{2}}) \end{cases} \tag{24}$$

At this point, we have achieved the kinematic solution of the AGV, and based on V_x, and ω, we can obtain V_{LF},V_{RF}, V_{LB}, V_{RB} and δ_{LF},δ_{RF}, δ_{LB}, δ_{RB}.

After obtaining the optimal trajectory in Section 3.3, the trajectory is composed of a set of points with speed information. By traversing the trajectory points with positioning information, we find the closest trajectory point to AGV, set the preview distance to be Len_{ahead}. We find the target point. The angle deviation between AGV and target point is δ_{err}.

$$\delta_{err} = arctan(\frac{y_{tar} - y_{AGV}}{x_{tar} - x_{AGV}}) - \delta_{AGV}, \tag{25}$$

where x_{tar}, y_{tar} are the coordinates of the target point. x_{AGV}, y_{AGV} are the coordinates of AGV, δ_{AGV} is the heading of AGV.

Using PD control to calculate the ω.

$$\omega = k_{p1}\delta_{err} + k_{d1}(\delta_{err} - \delta_{err_last}), \tag{26}$$

where k_{p1}, k_{d1} are the PID parameters, and δ_{err_last} is the previous angle error.

The target point is equipped with speed information $speed_{target}$, the current speed ($speed_{now}$) of AGV can be obtained through the perceived information, the difference between the two results in a speed deviation is the $speed_{err}$, then the speed control is using an incremental PID algorithm as shown in the following equation:

$$v_x = speed_{now} + k_{p2}speed_{err} + k_{i2}\int_0^t speed_{err}dt + k_{d2}(speed_{err} - speed_{err_last}), \tag{27}$$

where k_{p2}, k_{i2}, k_{d2} are the PID parameters, and $speed_{err_last}$ is the previous speed error.

According to the Equations (26) and (27), we calculate the v_x and ω, respectively, to control AGV.

The parameters used in the control are shown in the table below (Table 6).

Table 6. Control parameters.

Params	Len_{ahead}	k_{p1}	k_{d1}	k_{p2}	k_{i2}	k_{i2}	H_{agv}	L_{agv}
Value	3 m	0.6	0.1	0.8	0.01	0.1	900 mm	1600 mm

4.3. Verification of Selection of Optimal Path in Coarse Planning

In the real road environment at the construction site, sampling was performed based on constraints such as road boundaries, obstacles, and reference lines. Over 30 sets of parallel experiments were conducted with different start and end points to validate the performance of the multi-objective evaluation function in coarse planning. Finally, three representative sets of experiments were selected for presentation. Test 1 is the starting point located on the centerline of the road, and the endpoint is also located on the centerline of the road. Test 2 is the starting point located away from the centerline of the road, and the endpoint is also located away from the centerline of the road. Test 3 starts at the center of the road and ends far away from the center of the road and near obstacles. By setting different conditions, we fully verify the reliability of the planning algorithm. The parameter values during the experiments are listed in Table 7.

Table 7. Sampling and multi-objective evaluation function parameters.

Params	Δs	Δl	L_a	S_a	d_{obs}	n	k_1	k_2	k_3	d
Value	1 m	1.5 m	4 m	12 m	0.5 m	10	10	10	1	0.5

The experimental results are presented in the following figures (Figures 21–23). Blue arrows indicate the current pose of the vehicle, brown rectangles represent obstacles, green squares represent obstacle safety distances, blue dots represent sampling points, red dots

represent the output points from coarse planning, and red arrows indicate the vehicle's target endpoint.

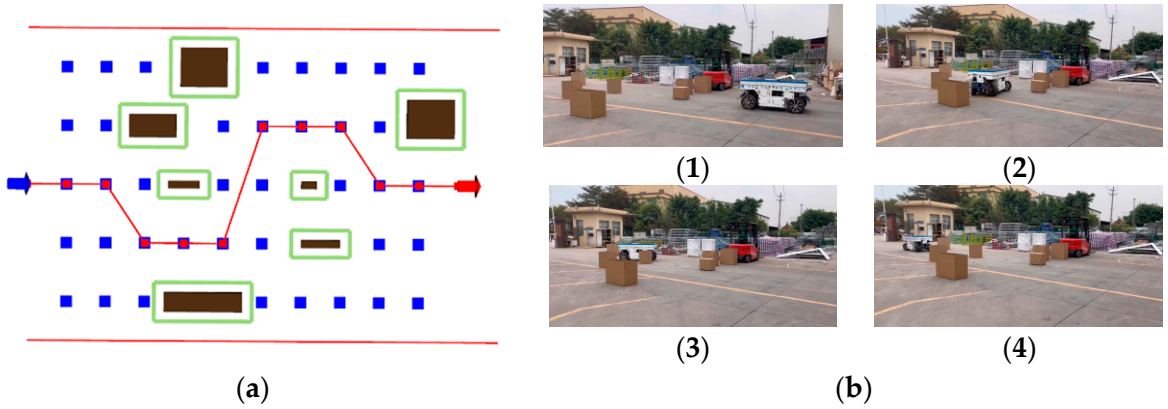


Figure 21. Effectiveness of Coarse Planning in test 1. (**a**) Display on the host computer in test 1. (**b**) (1)–(4) are the testing process at the corresponding real site in test 1.

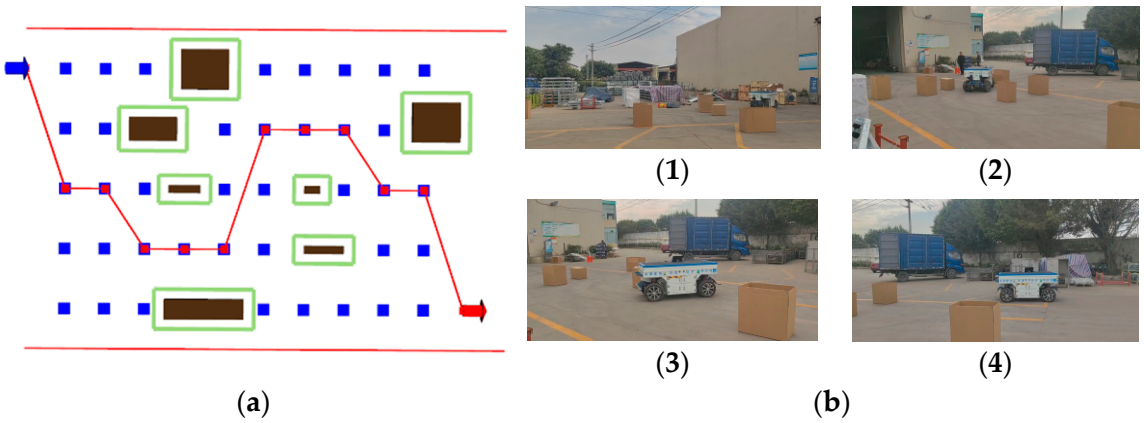


Figure 22. Effectiveness of coarse planning in test 2. (**a**) Display on the host computer in test 2. (**b**) (1)–(4) are the testing process at the corresponding real site in test 2.

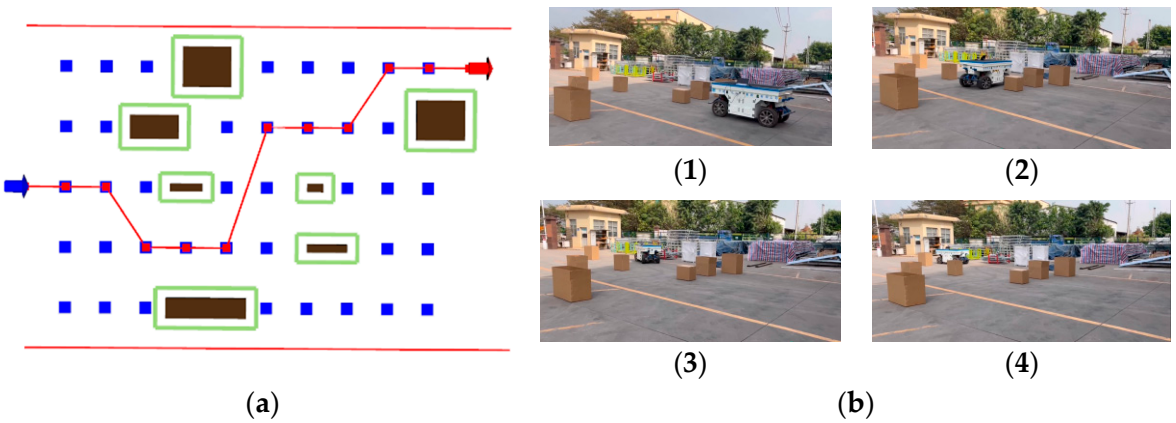


Figure 23. Effectiveness of coarse planning in test 3. (**a**) Display on the host computer in test 3. (**b**) (1)–(4) are the testing process at the corresponding real site in test 3.

Based on the experimental results, we can obtain the following conclusions.

(1) Sampling based on environmental constraints efficiently and succinctly achieved waypoint sampling. In the experiment, a total of 50 points were sampled to cover a width

of 8 m and extended longitudinally by 12 m. Eleven invalid sampling points were filtered out through obstacle filtering (denoted by "×" in the table, as in Table 8, (S4, L1)), resulting in computational workload savings.

Table 8. Evaluation Results of est 1.

Cost	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
L_1	334.3	374.9	∞	×	×	131.9	96.5	64.7	98.8	∞
L_2	314.6	286.2	×	×	149.5	114.6	80.3	45.7	35.2	×
L_3	303.7	273.1	∞	×	152.2	194.8	×	∞	23.4	11.8
L_4	313.7	269.5	236.7	200.6	163.6	283.5	×	×	32.9	34.3
L_5	333.2	287.2	254.7	×	×	288.7	187.4	86.3	52.3	96.3

(2) The selection of the optimal path is based on a greedy approach, where the optimal path is selected in a reverse manner. For example, in Table 9, under the S2 column, L3 is not the point with the minimum cost, but in conjunction with the choice made by the global path planning, the optimal path includes (S2, L3). Therefore, this point is chosen as the optimal sampling point in that longitudinal section.

Table 9. Evaluation Results of Test 2.

Cost	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
L_1	362.3	402.9	∞	×	×	159.9	124.4	92.7	114.5	∞
L_2	342.6	314.2	×	×	177.5	142.6	108.2	73.6	63.2	×
L_3	331.7	301.0	∞	×	180.2	222.7	×	∞	51.3	39.8
L_4	341.7	297.4	264.7	228.6	191.6	311.4	×	×	60.9	50.1
L_5	361.2	315.2	282.7	×	×	316.7	215.4	114.2	80.2	98.4

(3) Although some sampling points are not within the range of obstacle filtering, due to linear interpolation points connecting adjacent columns of sampling points, if there is always an interpolation point within the obstacle filtering range during the process of reaching the endpoint for all selected paths, the cost of that point is considered infinite, as shown in Table 10 (S6, L4).

Table 10. Evaluation results of test 3.

Cost	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
L_1	585.6	626.7	∞	×	×	383.3	347.8	302.1	202.9	101.7
L_2	565.9	537.6	×	×	400.8	365.9	331.6	297.0	∞	×
L_3	555.0	524.4	∞	×	403.6	446.1	×	∞	∞	∞
L_4	565.0	520.8	488.1	451.9	414.9	∞	×	×	∞	∞
L_5	584.6	538.5	506.1	×	×	∞	169.3	∞	∞	∞

(4) On the basis of safely avoiding all obstacles and considering the evaluation function set, the final planned path tends to be close to the centerline of the road and ensures the shortest path. The planning results are excellent and align with expectations.

4.4. Curve Fitting and Optimization Experiments

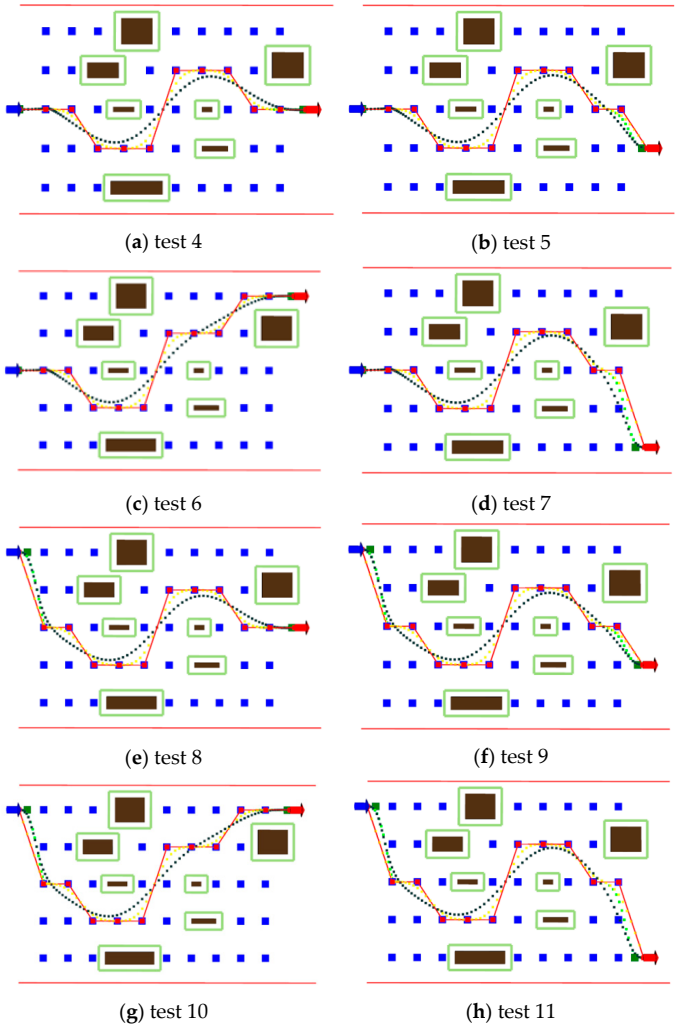
Building upon the foundation of Section 4.2, the coarse planning algorithm-generated path points were improved using B-spline fitting as described in Section 2.3, followed by optimization using quadratic programming. The parameters used are as shown in the table below (Table 11).

Table 11. Parameters for Curve Fitting and Optimization.

Params	n_z	v_1	v_2	v_3	w_1	w_2	w_3	b_{n1}	b_{n2}	b_{max}	b_{min}
values	9	500	1	10	400	1	10	7	$b_n - 7$	0.5	0.01

Out of the more than 30 sets of experiments conducted, 8 representative sets of experiments were selected. The starting point of tests 4–7 are at the centerline of the road. Their endpoints are on the road centerline, slightly off the road centerline, significantly off the road centerline and close to obstacles, and significantly off the road centerline and away from obstacles. The starting points of tests 8–11 are all far from the centerline of the road, and the endpoint setting rules are the same as tests 4–7. By setting eight different operating conditions, the fitting and optimization effects of curves under different rough solutions were fully tested. The visualization of the experimental results is shown in the following figure (where the red line represents the discrete path point's output by coarse planning, the yellow line represents the traditional B-spline fitting curve, the green line represents the improved B-spline fitting curve, and the black line represents the final path after quadratic programming optimization).

Taking Figure 24h as an example, as shown in Figure 25, importing the fitting curves into Origin, we can perform a specific analysis of the four types of paths: original path, traditional B-spline, improved B-spline, and the path after quadratic programming optimization. Additionally, we can calculate the cost values n_1 and n_2 to analyze the algorithm's effectiveness.



(**a**) test 4 (**b**) test 5 (**c**) test 6 (**d**) test 7 (**e**) test 8 (**f**) test 9 (**g**) test 10 (**h**) test 11

Figure 24. Curve fitting and optimization.

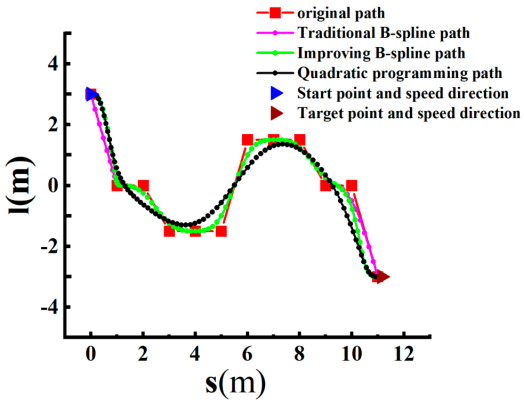


Figure 25. Test 11 displaying the analysis in Origin.

As shown in Figure 26①, B-spline fitting eliminates the sharp corners in the original path caused by the sparse sampling points. At these corners, a smooth curve is used to ensure the smoothness of AGV motion. As depicted in Figure 26②, at both the starting and ending points, the vehicle's velocity direction is horizontal to the right. However, traditional B-spline fitting paths do not consider velocity direction; instead, they purely fit based on the coordinates of path points. This can lead to difficulties in smoothly tracking the planned path at the starting point and pose inaccuracies at the ending point.

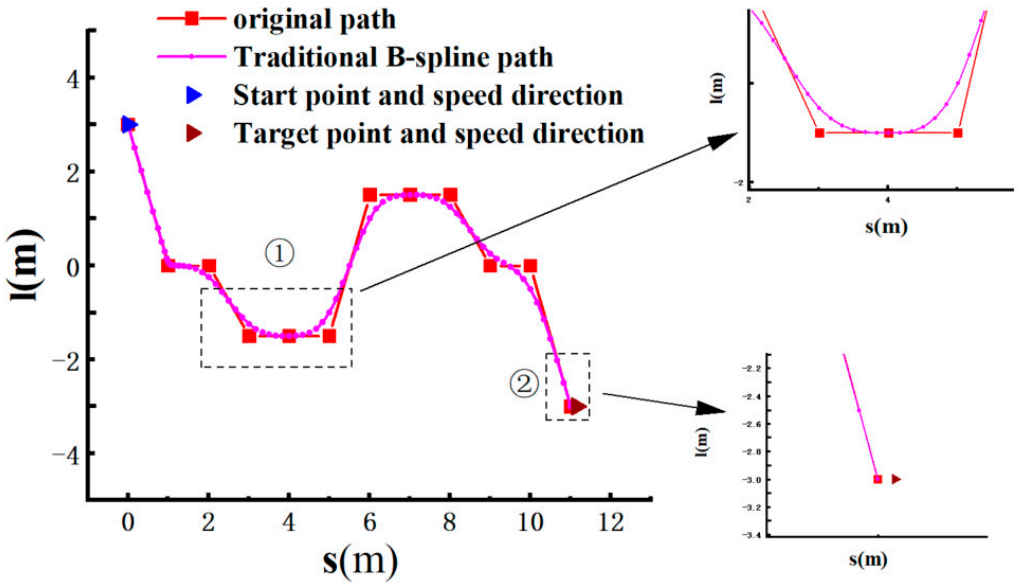


Figure 26. Comparison before and after B-spline fitting.

As shown in Figure 27②, the improved B-spline fitting, while eliminating sharp corners, also considers the velocity direction at the starting and ending points. The green curve ultimately ensures that the vehicle reaches the endpoint with a horizontal rightward velocity. The cost values n_1 and n_2 are calculated as shown in Figure 28, and the best interpolation results are achieved when $n_1 = 6$ and $n_2 = 7$. As depicted in Figure 27①, although the improved B-spline eliminates most of the sharp corners, in areas with steeper corners, the fitted path may still not be smooth enough, and there is room for overall path improvement.

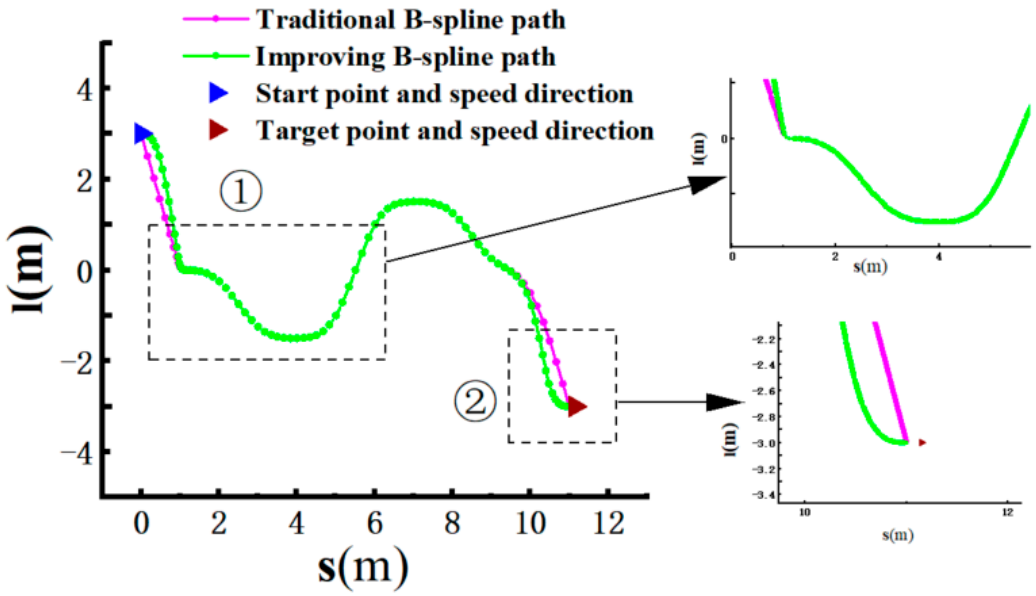


Figure 27. Traditional B-spline and improved B-spline.

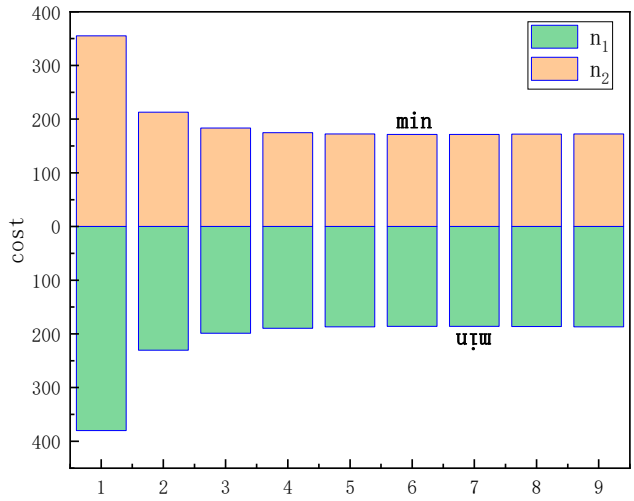


Figure 28. Cost of n_1, n_2.

As shown in Figure 29②, under the improved B-spline fitting path with a relatively large value for k1, the quadratic programming prioritizes smoothness. Due to the setting of segmental tolerance, it ensures that the smooth optimization does not disrupt the planned path's velocity direction at the starting and ending points, as depicted in Figure 29① and ③.

After a series of processing steps, the path output by quadratic programming has been fitted to create a high-quality path that aligns with the original path, ensuring smoothness, shortest-path characteristics, and proximity to the reference line. This path serves as the final result of coarse planning and is sent to precise planning. The remaining seven sets of experiments in Figure 24 all align with the conclusions drawn in the analysis above.

4.5. Validation of Lateral Planning Effectiveness in Precise Planning

As shown in Figure 30, the planning results from Section 4.4 are used as the input reference lines for precise planning, taking test6 as an example. Sampling planning is performed under the motion constraints defined in Section 3.1 to validate the planning effectiveness of the lateral planning in precise planning. The parameters used are as listed in Table 12.

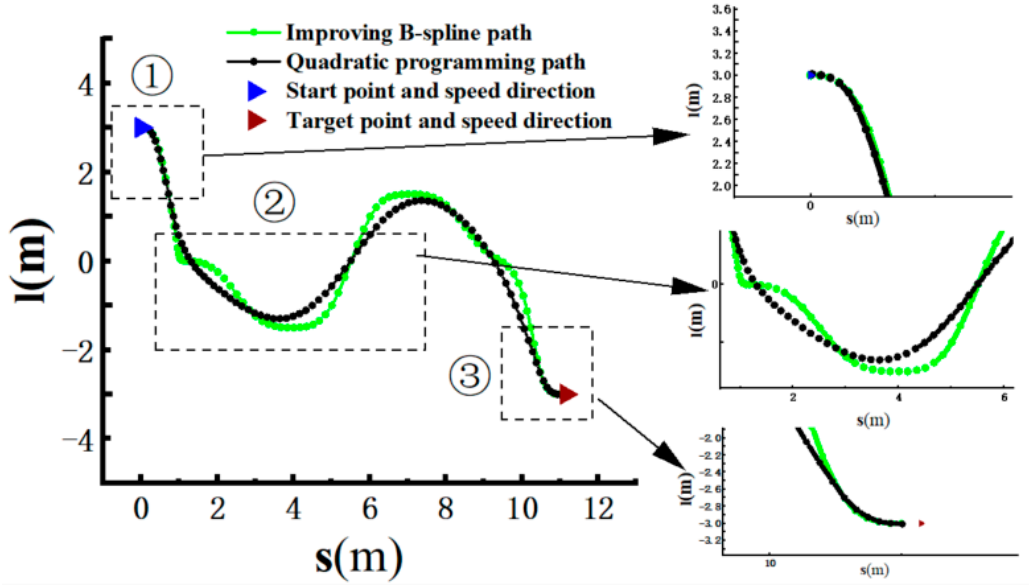
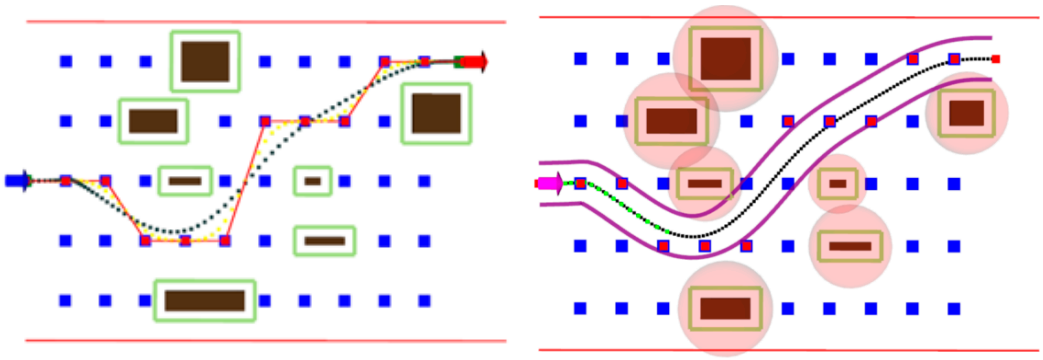


Figure 29. Effectiveness of quadratic programming.



(**a**) Coarse planning (**b**) Precise planning

Figure 30. Coarse planning vs. precise planning comparison.

Table 12. Precise planning parameters.

Params	v_{max}	a_{max}	j_{max}	k_{max}	T_{max}	T_{min}	Δd	Δv
Value	1.5 m/s	0.2 m/s^2	0.25 m/s^3	1	8.0 s	3.0 s	0.1 m	0.3 m/s
Params	n_v	c_1	c_3	c_4	dis_{save}	L_{veh}	L	
Value	5	0.5	100	80	0.4 m	0.5 m	1 m	

The experimental process, as described in Section 3.2, involves calculating the optimal path using the evaluation function and then eliminating dangerous trajectories through collision detection. The vehicle is controlled to track the planned trajectory using a pure tracking algorithm. The visualization of the experimental process is shown in the following figure (Figures 31 and 32) (where the pink arrow represents the AGV, and the light red circle represents the danger zone of obstacles).

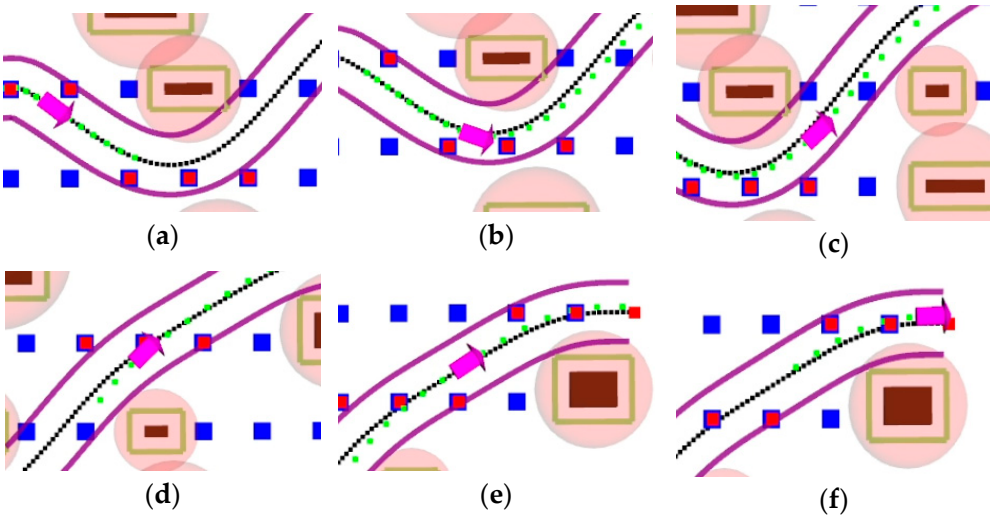


Figure 31. Precise planning lateral planning in simulation. (**a**)–(**f**) are the precise planning experiment simulation process.

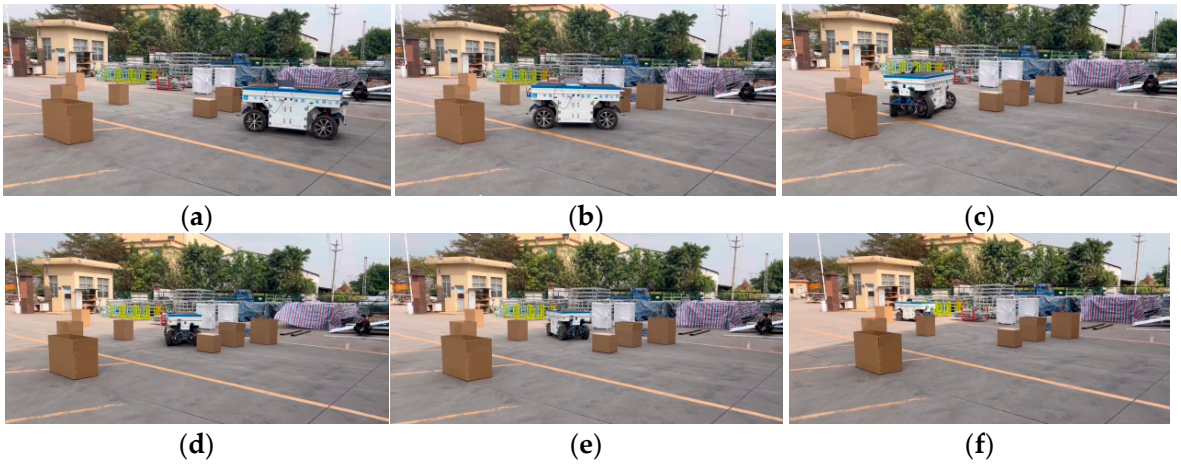


Figure 32. Precise planning lateral planning in reality. (**a**)–(**f**) are the precise planning of the actual testing process.

Compare the reference line output by coarse planning with the planned path in precise planning as shown in Figure 33a, the planned road curvature in Figure 33b, and analyze the changes in trajectory acceleration and jerk during lateral planning as depicted in Figure 33c.

Based on the experimental results, we can obtain the following conclusions.

(1) As shown in Figure 30 and Table 12, compared to coarse planning, the lateral sampling range has been reduced from $[-4, 4]$ to $[-0.5, 0.5]$, with lateral sampling points reduced from 80 to 10 under the condition of $\Delta d = 0.1$. This significant reduction in the sampling range greatly improves algorithm speed while ensuring efficient, accurate, and safe lateral sampling based on the coarse planning reference line. This greatly enhances both sampling efficiency and quality.

(2) As depicted in Figure 31, although the reference line output by coarse planning is already capable of avoiding all obstacles, precise planning, due to curve fitting, optimization, and the inclusion of a safety margin (dis_{save}), still performs local obstacle avoidance, as shown in Figure 33a for cases ① and ②. This corresponds to (b), (e), and (f) in Figure 31. As shown in Figure 33b, the lateral deviation is between $[-0.17, 0.1]$.

(3) As demonstrated in Figure 33b, the planned road curvature is within the range $[-0.6, 0.87]$, and as shown in Figure 33c, lateral acceleration ranges from $[-0.05, 0.016]$, while jerk ranges from $[-0.06, 0.012]$. These values all satisfy the constraints specified in Table 12.

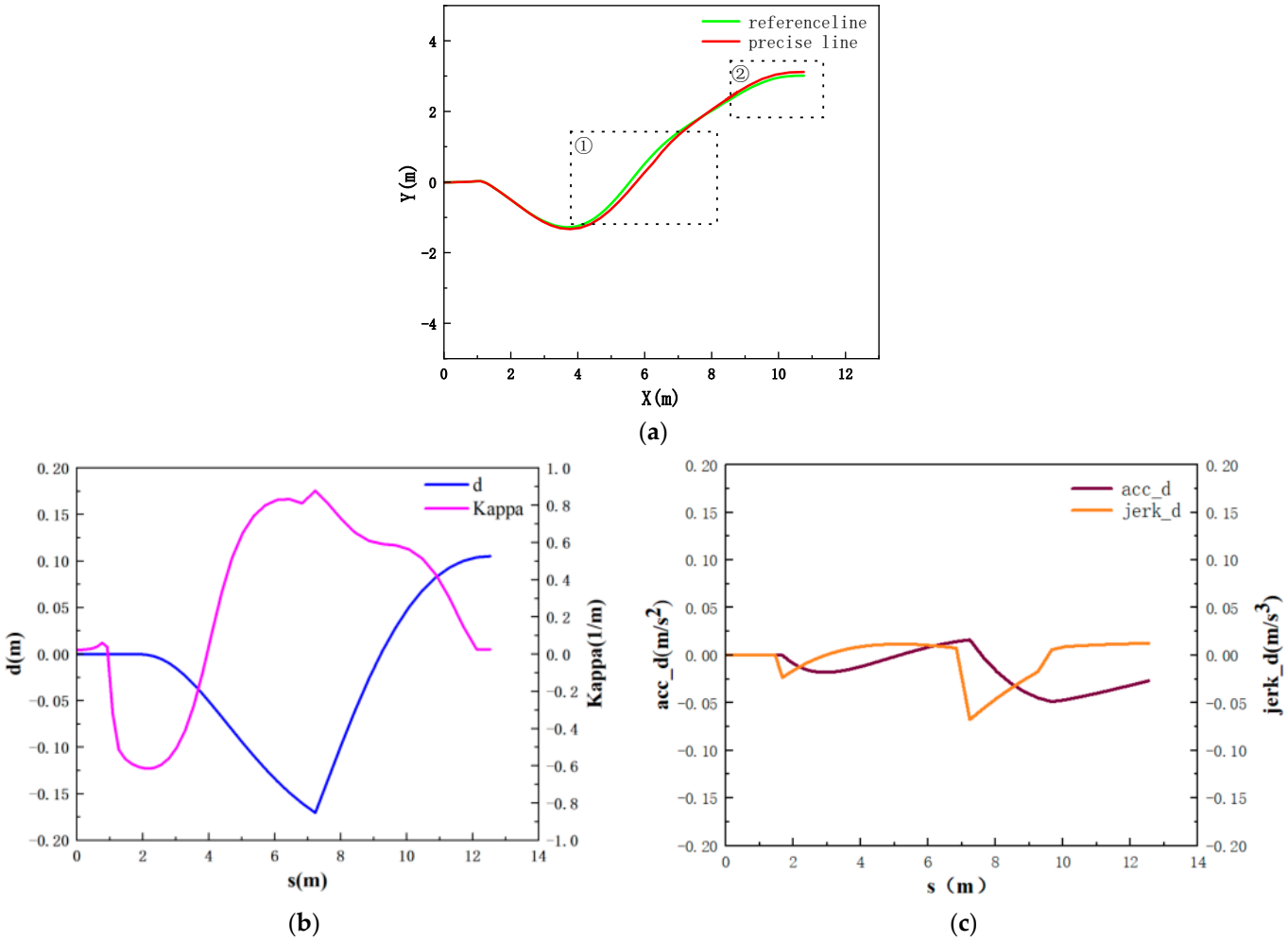


Figure 33. Analysis of precise planning lateral planning. (**a**) Comparison between reference line and precise-planning trajectory. (**b**) Lateral acceleration and jerk. (**c**) Lateral deviation and planned road curvature.

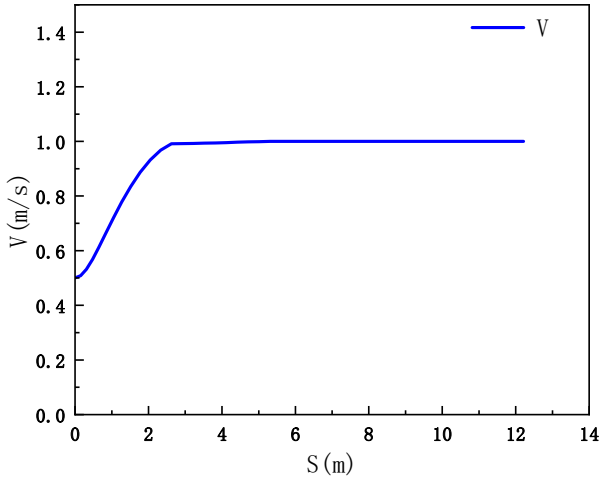
4.6. Validation of Longitudinal Planning in Precise Planning

In the experiments conducted in Section 4.4, the effectiveness of longitudinal planning in precise planning is validated by setting different initial speeds and target speeds. Four sets of speed planning experiments are conducted to evaluate its performance in small-scale acceleration and deceleration as well as large-scale acceleration and deceleration. The results of speed planning are shown in the following figure.

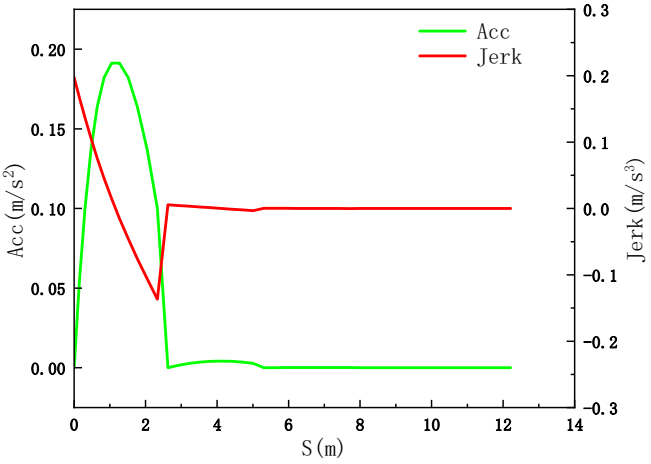
The experimental statistics are presented in the following table.

The experimental results indicate the following:

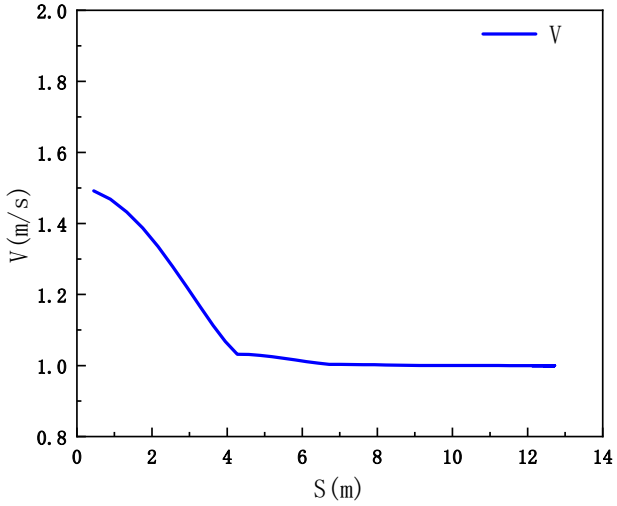
(1) As shown in Figure 34a–d, for small-scale acceleration and deceleration, precise planning can achieve the target speed within a single planning cycle. However, for large-scale acceleration and deceleration (0 m/s→1.5 m/s and 1.5 m/s→0.5 m/s), as observed in Figure 34f,h, it takes three and two planning cycles, respectively. This is because for significant speed changes, precise planning needs to consider factors like acceleration and jerk costs, so it cannot achieve the target speed within a single planning cycle.



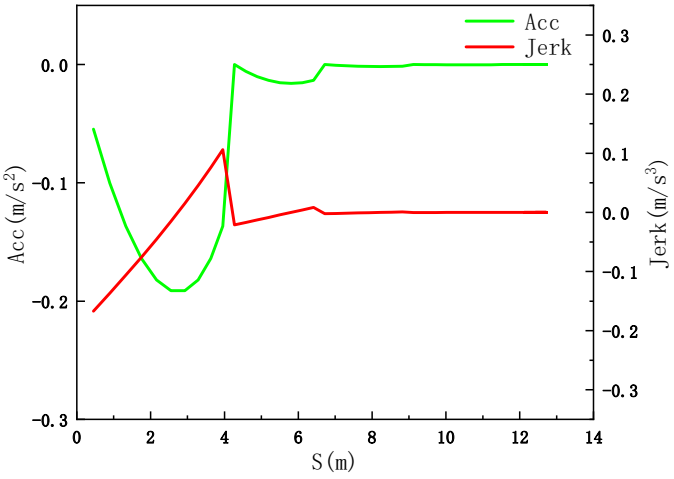
(**a**) The speed planning from 0.5 m/s to 1 m/s



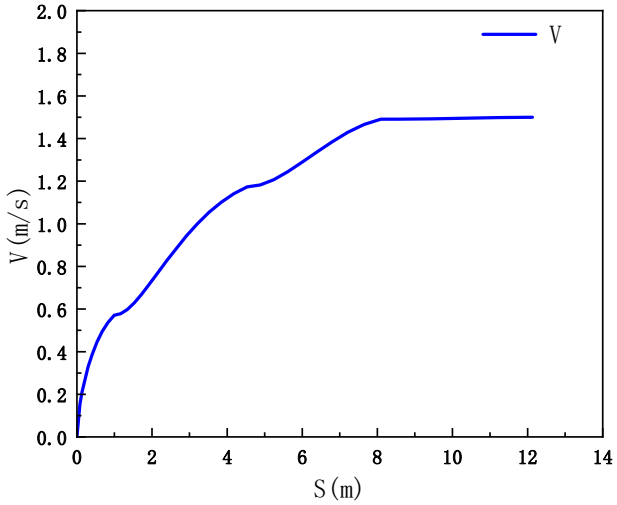
(**b**) The Acc and jerk charts from 0.5 m/s to 1 m/s



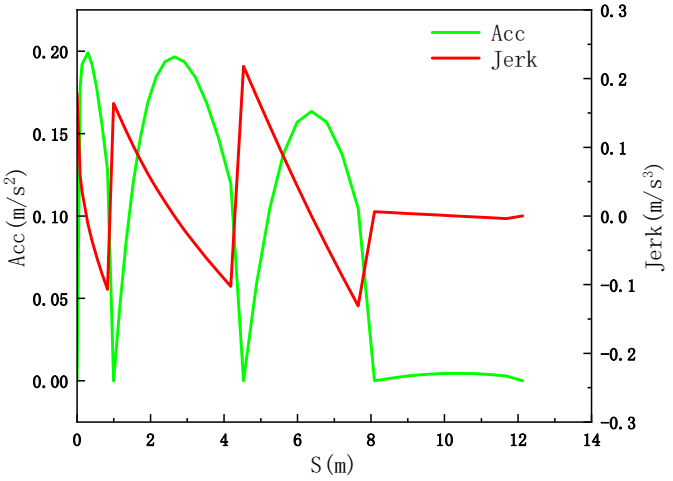
(**c**) The speed planning from 1.5 m/s to 1 m/s



(**d**) The Acc and jerk charts from 1.5 m/s to 1 m/s

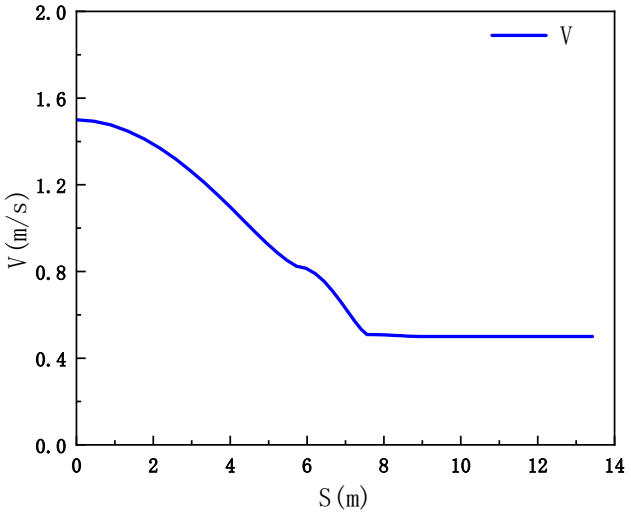


(**e**) The speed planning from 0.5 m/s to 1.5 m/s

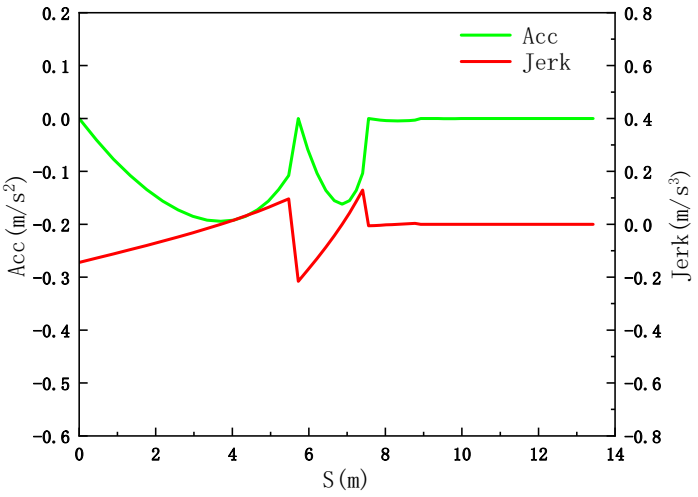


(**f**) The Acc and jerk charts from 0 m/s to 1.5 m/s

Figure 34. *Cont.*



(**g**) The speed planning from 1.5 m/s to 0.5 m/s



(**h**) The Acc and jerk charts from 1.5 m/s to 0.5 m/s

Figure 34. Speed planning statistics chart.

(2) According to Table 13, in all four sets of experiments, the vehicles eventually reach or get close to the target speed. This indicates accurate speed planning, and the acceleration and jerk values meet the requirements in Table 12, demonstrating the effectiveness of sampling constraints based on the motion model.

Table 13. Speed planning statistics.

(m/s)	Final Velocity	Max Acc	Max Jerk	Min Jerk
0.5→1	1.00	0.1911	0.1972	−0.1365
1.5→1	1.00	−0.1912	0.1062	−0.1668
0→1.5	1.49	0.1991	0.2180	−0.1308
1.5→0.5	0.50	−0.1944	0.1296	−0.2160

(3) The maximum acceleration for all four sets of experiments reaches 0.19 m/s^2, and the acceleration changes gradually under the constraints of jerk. This reflects that speed planning doesn't blindly pursue reaching the desired speed but considers a balance between speed, longitudinal acceleration, and jerk values. This ensures the AGV's stability while achieving the desired speed as quickly as possible.

4.7. Comparative Experiments

To validate the proposed method in this paper, it is compared with the traditional Frenet sampling planning algorithm [28–30], and the experimental process is shown in the Figures 35 and 36.

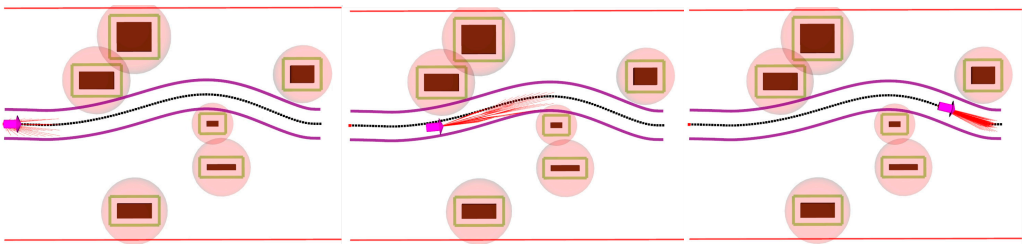


Figure 35. Hierarchical trajectory planning algorithm.

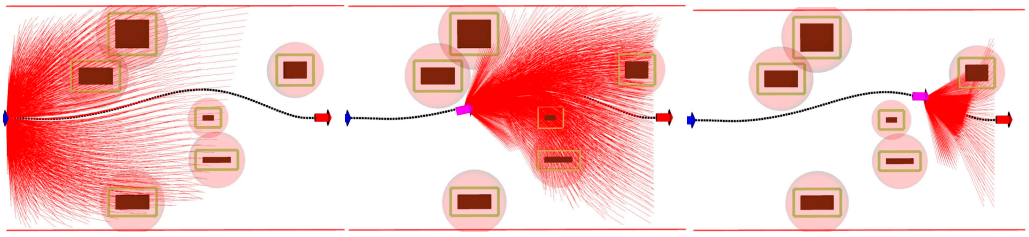


Figure 36. Frenet sampling planning algorithm.

As described in Section 3.3, the hierarchical trajectory planning algorithm reduces the time complexity of traditional Frenet sampling planning algorithms from $O(n^2)$ to $O(n)$. In the experiment, the hierarchical trajectory planning algorithm samples 80 sets of trajectories each time, while the traditional Frenet sampling planning algorithm samples 1156 sets of trajectories each time, Under the results of coarse planning the precise planning algorithm is applied, and the number of samples is significantly reduced. This provides an effective solution for dealing with the retrieval area of wide roads in the construction site environment.

At the same time, this paper uses the DWA planning algorithm [33] to compare the hierarchical trajectory planning algorithm. The experimental process is shown in Figures 37 and 38.

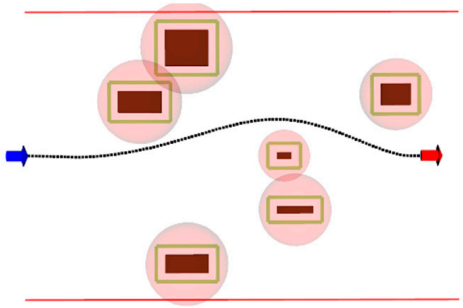


Figure 37. Hierarchical trajectory planning algorithm compare with DWA.

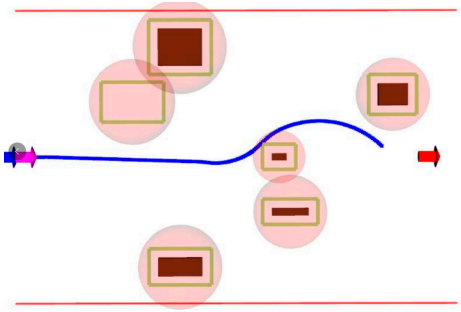


Figure 38. DWA planning algorithm.

It can be observed that the DWA planning algorithm is a path planning algorithm that does not consider road boundaries and is used in unknown spaces. The planning results have uncertainty. The hierarchical trajectory planning algorithm can obtain a rough solution path through coarse planning, and the rough solution path is optimized through fitting, which is significantly better than the DWA planning algorithm in terms of smoothness. During the experiment, the DWA planning algorithm may also encounter the problem of local optimal traps, as shown in Figure 39. The hierarchical trajectory planning algorithm uses greedy thinking to find the optimal path from the endpoint in coarse planning, ensuring that the algorithm does not fall into local optimal traps.

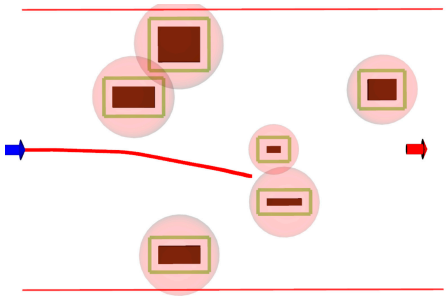


Figure 39. DWA of local optimal trap.

Based on the experiments conducted in Sections 4.3–4.7. This paper presents a comparison table of the hierarchical trajectory planning algorithm, dynamic programming algorithm, Frenet sampling algorithm, and DWA algorithm as follows: According to Table 14, although dynamic programming algorithms have high efficiency and no local traps, it cannot smooth paths and does not have speed planning. The disadvantage of Frenet sampling planning is that it frequently solves solutions of fifth and fourth polynomials over a wide range of horizontal and lateral directions, resulting in low running efficiency. DWA planning may fall into local traps, and the RRT algorithm does not have the ability to optimize speed planning and curve smoothing. Through comparison with various algorithms, it was found that the hierarchical trajectory planning algorithm is a trajectory planning algorithm with high operational efficiency, no local traps, smooth trajectory optimization, and has the capable of speed planning. In the table, × represents that the corresponding function has defects, while √ represents that it has the corresponding function.

Table 14. Algorithm comparison.

	Efficiency	Local Trap	Smooth Optimization	Speed Planning
Hierarchical Trajectory	high	×	√	√
Dynamic Programming [32]	high	×	×	×
Frenet Sampling [28–30]	low	√	√	√
DWA [33]	high	√	×	×
RRT [34]	high	×	×	×

5. Conclusions

This paper constructs a hierarchical trajectory planning algorithm through the design of two algorithms: coarse planning based on road environment sampling and precise planning based on AGV motion sampling. The hierarchical trajectory planning algorithm ensures efficient algorithm operation while also considering the smoothness of the planned trajectory. It ensures that high-capacity AGVs can output accurate, safe, and smooth planned paths efficiently and stably on wide retrieval roads in a construction site environment.

In the final tests and experiments, we conducted tests based on AGV real vehicles in a real construction site environment. From the actual test results, the dynamic programming algorithm for coarse planning can quickly select the optimal path. The interpolated B-spline fitting method, considering the nonholonomic constraints of the vehicle, ensures the rationality of the path while eliminating the planning turning points. Secondary optimization further ensures the smoothness of the path. Fine planning, based on coarse planning, reduces the lateral sampling range from 8 m to 1 m. In the speed planning process, various acceleration and deceleration scenarios of AGV were tested. The experimental results show that the planned trajectory meets collision detection requirements while achieving the desired speed rapidly and considering trajectory smoothness. In all test experiments, the maximum acceleration was 0.1944 m/s^2, and the maximum jerk was 0.1972 m/s^3, indicating good speed planning performance. This implementation achieves high-quality and efficient trajectory planning.

This research effectively addresses challenges faced by AGVs in construction site environments, such as heavy loads, wide road sampling ranges, and random obstacle distributions. It provides new insights into solving path planning problems in construction site environments. The proposed hierarchical trajectory planning algorithm can be used for single AGV working conditions. It can be further improved while considering the cooperation of multiple vehicles. In the future, we are planning to integrate the leader follower control, game theory and other algorithms for planning and control under multi vehicle working conditions to meet the real requirements of building sites.

Author Contributions: Conceptualization, Y.B., P.L. and W.L.; Methodology, Y.B. and W.L.; Software, Y.B.; Validation, Y.B.; Formal analysis, W.L.; Resources, Z.C. and P.Y.; Writing—original draft, Y.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under Grant 52175007, the China Postdoctoral Science Foundation under Grant 2018M630348/2019T120259, and the Yangtze River Delta HIT robot technology research institute under Grant HIT-CXY-CMP2-ADTIL-21-01.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Ge, X.; Li, L.; Chen, H. Research on Online Scheduling Method for Flexible Assembly Workshop of Multi-AGV System Based on Assembly Island Mode. In Proceedings of the 2021 IEEE 7th International Conference on Cloud Computing and Intelligent Systems (CCIS), Xi'an, China, 7–8 November 2021; pp. 371–375. [CrossRef]
2. Lian, Y.; Zhang, L.; Xie, W.; Wang, K. An Improved Heuristic Path Planning Algorithm for Minimizing Energy Consumption in Distributed Multi-AGV Systems. In Proceedings of the 2020 International Symposium on Autonomous Systems (ISAS), Guangzhou, China, 6–8 December 2020; pp. 70–75. [CrossRef]
3. Yang, J.; Yu, F.; Wu, E.; Yang, Y. Energy saving of automated terminals considering AGV speed variation. In Proceedings of the 2022 International Symposium on Sensing and Instrumentation in 5G and IoT Era (ISSI), Shanghai, China, 17–18 November 2022; pp. 68–73. [CrossRef]
4. Macias-Sola, J.; Uttendorf, S.; Blech, J.O. A Ground Texture-based Mapping and Localization Method for AGVs. In Proceedings of the 2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN), Lloret de Mar, Spain, 29 November–2 December 2021; pp. 1–6. [CrossRef]
5. Zhu, H.; Liu, J. Improved Harmony Search Algorithm and Application in Optimal Scheduling of Hospital Transport AGV. In Proceedings of the 2021 IEEE International Conference on Electrical Engineering and Mechatronics Technology (ICEEMT), Qingdao, China, 2–4 July 2021; pp. 739–743. [CrossRef]
6. Wang, H.; Chen, T.; Wang, S.; Lu, B. Semantic Segmentation Algorithms for Ground AGV and UAV Medical Transport Scenes. In Proceedings of the 2022 16th ICME International Conference on Complex Medical Engineering (CME), Zhongshan, China, 4–6 November 2022; pp. 252–255. [CrossRef]
7. Jodejko-Pietruczuk, A.; Werbińska-Wojciechowska, S. Availability assessment for a multi-AGV system based on simulation modeling approach. In Proceedings of the 2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), Mauritius, Mauritius, 7–8 October 2021; pp. 1–6. [CrossRef]
8. Spišáková, M.; Mésáros, P.; Kaleja, P.; Mandičák, T. Virtual Reality as a Tool for Increasing Safety of Construction Sites. In Proceedings of the 2020 18th International Conference on Emerging eLearning Technologies and Applications (ICETA), Košice, Slovenia, 12–13 November 2020; pp. 1–6. [CrossRef]
9. Wu, H.-T.; Yu, W.-D.; Gao, R.-J.; Wang, K.-C.; Liu, K.-C. Measuring the Effectiveness of VR Technique for Safety Training of Hazardous Construction Site Scenarios. In Proceedings of the 2020 IEEE 2nd International Conference on Architecture, Construction, Environment and Hydraulics (ICACEH), Hsinchu, Taiwan, 25–27 December 2020; pp. 36–39. [CrossRef]
10. Wang, C.; Mao, J. Summary of AGV Path Planning. In Proceedings of the 2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE), Xiamen, China, 18–20 October 2019; pp. 332–335. [CrossRef]
11. Yuan, H.; Zhang, G.; Li, Y.; Liu, K.; Yu, J. Research and implementation of intelligent vehicle path planning based on four-layer neural network. In Proceedings of the 2019 Chinese Automation Congress (CAC), Hangzhou, China, 22–24 November 2019; pp. 578–582. [CrossRef]
12. Zu, W.; Fan, G.; Gao, Y.; Ma, Y.; Zhang, H.; Zeng, H. Multi-UAVs Cooperative Path Planning Method based on Improved RRT Algorithm. In Proceedings of the 2018 IEEE International Conference on Mechatronics and Automation (ICMA), Changchun, China, 5–8 August 2018; pp. 1563–1567. [CrossRef]

13. Liu, Q.; Cheng, X.; Hui, J.; Ming, R. Research on Robot Path Planning Method Based on Tangent Intersection Method. In Proceedings of the 2020 International Conference on Information Science, Parallel and Distributed Systems (ISPDS), Xi'an, China, 14–16 August 2020; pp. 272–276. [CrossRef]
14. Shi, K.; Wu, P.; Liu, M. Research on path planning method of forging handling robot based on combined strategy. In Proceedings of the 2021 IEEE International Conference on Power Electronics, Computer Applications (ICPECA), Shenyang, China, 22–24 January 2021; pp. 292–295. [CrossRef]
15. Zhu, J.; Zhao, S.; Zhao, R. Path Planning for Autonomous Underwater Vehicle Based on Artificial Potential Field and Modified RRT. In Proceedings of the 2021 International Conference on Computer, Control and Robotics (ICCCR), Shanghai, China, 8–10 January 2021; pp. 21–25. [CrossRef]
16. Jin, Y. Research on Path Planning of Airport VIP Service Robot Based on A* Algorithm and Artificial Potential Field Method. In Proceedings of the 2021 5th Asian Conference on Artificial Intelligence Technology (ACAIT), Haikou, China, 29–31 October 2021; pp. 485–489. [CrossRef]
17. Yao, J. Path Planning Algorithm of Indoor Mobile Robot Based on ROS System. In Proceedings of the 2023 IEEE International Conference on Image Processing and Computer Applications (ICIPCA), Changchun, China, 28–30 June 2023; pp. 523–529. [CrossRef]
18. Ma, B.; Wei, C.; Huang, Q.; Hu, J. APF-RRT*: An Efficient Sampling-Based Path Planning Method with the Guidance of Artificial Potential Field. In Proceedings of the 2023 9th International Conference on Mechatronics and Robotics Engineering (ICMRE), Shenzhen, China, 10–12 February 2023; pp. 207–213. [CrossRef]
19. Bi, Q.; Pian, J.; Wu, F.; Dai, Y. Research on Robot Path Planning Based on Improved Genetic Algorithm. In Proceedings of the 2022 China Automation Congress (CAC), Xiamen, China, 25–27 November 2022; pp. 242–246. [CrossRef]
20. Chen, Z.; Gao, Q.; Wang, X.; Liu, X. Local path planning of intelligent vehicle based on improved artificial potential field. In Proceedings of the 2020 IEEE 3rd International Conference on Electronic Information and Communication Technology (ICEICT), Shenzhen, China, 13–15 November 2020; pp. 110–116. [CrossRef]
21. Li, J.; Zhang, J. Global path planning of unmanned boat based on improved ant colony algorithm. In Proceedings of the 2021 4th International Conference on Electron Device and Mechanical Engineering (ICEDME), Guangzhou, China, 19–21 March 2021; pp. 176–179. [CrossRef]
22. Fan, W.; Li, Y.; Yang, J.; Wang, B. Global path planning method for mobile vehicles based on Goal-Optimized JPS algorithm. In Proceedings of the 2023 26th International Conference on Electrical Machines and Systems (ICEMS), Zhuhai, China, 5–8 November 2023; pp. 2498–2501. [CrossRef]
23. Peng, Q.; Guo, H.; Hu, C. Graph Embedding and Approximate Dynamic Programming for the Reliable Shortest Path Problem. In Proceedings of the 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), Indianapolis, IN, USA, 19–22 September 2021; pp. 653–659. [CrossRef]
24. Barros, R.M.; Lage, G.G.; De Andrade Lira Rabêlo, R. Graph Shortest Path and Dynamic Programming Applied to Sequencing ORD Control Adjustments. In Proceedings of the 2023 IEEE Belgrade PowerTech, Belgrade, Serbia, 25–29 June 2023; pp. 1–6. [CrossRef]
25. Jiang, S.; Zhang, Y.; Liu, R.; Jafari, M.; Kharbeche, M. Data-Driven Optimization for Dynamic Shortest Path Problem Considering Traffic Safety. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 18237–18252. [CrossRef]
26. Ren, J.; Huang, X. Dynamic Programming Inspired Global Optimal Path Planning for Mobile Robots. In Proceedings of the 2021 IEEE 4th International Conference on Information Systems and Computer Aided Education (ICISCAE), Dalian, China, 24–26 September 2021; pp. 461–465. [CrossRef]
27. Gamayanti, N.; Kadir, R.E.A.; Alkaff, A. Global Path Planning for USV Waypoint Guidance System Using Dynamic Programming. In Proceedings of the 2020 International Seminar on Intelligent Technology and Its Applications (ISITIA), Surabaya, Indonesia, 22–23 July 2020; pp. 248–253. [CrossRef]
28. Kusuma, Z.D.S.; Indriawati, K.; Widjiantoro, B.L.; Hija, A.I.; Nurhadi, H. Optimal Trajectory Planning Generation for Autonomous Vehicle Using Frenet Reference Path. In Proceedings of the 2022 5th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), Yogyakarta, Indonesia, 8–9 December 2022; pp. 480–484. [CrossRef]
29. Zhang, D.; Zhao, J.; Jiao, X. Trajectory planning of autonomous vehicle based on Frenét frame system. In Proceedings of the 2021 China Automation Congress (CAC), Beijing, China, 22–24 October 2021; pp. 7973–7977. [CrossRef]
30. Teng, Q.; Wang, X.; Han, B.; Liu, J.; Tang, M. Research on the trajectory planning method of inland waterway vessels based on Frenet coordinate system. In Proceedings of the 2023 7th International Conference on Transportation Information and Safety (ICTIS), Xi'an, China, 4–6 August 2023; pp. 644–650. [CrossRef]
31. Wang, X.; Yu, H.; Liu, J. Research on Local Path Planning Algorithm Based on Frenet Coordinate System. In Proceedings of the 2022 4th International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI), Shanghai, China, 28–30 October 2022; pp. 249–252. [CrossRef]
32. Lin, X.; Cao, N.; Lin, Y. Optimal control for a class of nonlinear systems with state delay based on Adaptive Dynamic Programming with ε-error bound. In Proceedings of the 2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL), Singapore, 16–19 April 2013; pp. 177–182. [CrossRef]

33. Zhang, Y.; Xiao, Z.; Yuan, X.; Li, S.; Liang, S. Obstacle Avoidance of Two-Wheeled Mobile Robot based on DWA Algorithm. In Proceedings of the 2019 Chinese Automation Congress (CAC), Hangzhou, China, 22–24 November 2019; pp. 5701–5706. [CrossRef]
34. Yang, H.; Li, H.; Liu, K.; Yu, W.; Li, X. Research on path planning based on improved RRT-Connect algorithm. In Proceedings of the 2021 33rd Chinese Control and Decision Conference (CCDC), Kunming, China, 22–24 May 2021; pp. 5707–5712. [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.