Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 5910503, 14 pages
https://doi.org/10.1155/2020/5910503

Research Article

Research on Intelligent Vehicle Path Planning Based on Rapidly-Exploring Random Tree

Yangyang Shi,[1] **Qiongqiong Li**,[1] **Shengqiang Bu**,[1] **Jiafu Yang**,[1] **and Linfeng Zhu**[1,2]

[1]*Nanjing Forestry University, Nanjing 210037, China*
[2]*North Information Control Institute Group Co., Ltd., Nanjing 211106, China*

Correspondence should be addressed to Jiafu Yang; jfyang@njfu.edu.cn

Received 19 December 2019; Revised 11 February 2020; Accepted 29 February 2020; Published 24 March 2020

Academic Editor: Mijia Yang

Aiming at the problems of large randomness, slow convergence speed, and deviation of Rapidly-Exploring Random Tree algorithm, a new node is generated by a cyclic alternating iteration search method and a bidirectional random tree search simultaneously. A vehicle steering model is established to increase the vehicle turning angle constraint. The Rapidly-Exploring Random Tree algorithm is improved and optimized. The problems of large randomness, slow convergence speed, and deviation of the Rapidly-Exploring Random Tree algorithm are solved. Node optimization is performed on the generated path, redundant nodes are removed, the length of the path is shortened, and the feasibility of the path is improved. The *B*-spline curve is used to insert the local end point, and the path is smoothed to make the generated path more in line with the driving conditions of the vehicle. The feasibility of the improved algorithm is verified in different scenarios. MATLAB/CarSim is used for joint simulation. Based on the vehicle model, virtual simulation is carried out to track the planned path, which verifies the correctness of the algorithm.

1. Introduction

In recent years, with the rapid development of smart vehicles, the advantages of smart vehicles themselves have become increasingly prominent. For example, smart vehicles can reduce driving pressure, improve the driving safety of smart vehicles, avoid traffic congestion, and reduce environmental pollution [1, 2]. Path planning is the core of intelligent driving technology and multirobot collaboration technology [3–6]. The ability to plan for unknown roads is an important criterion for measuring smart cars. Vehicle path planning refers to planning a path that does not collide with obstacles according to certain standards when the starting position, ending position of vehicles, and the distribution of obstacles in the environment are known. In recent years, scholars have done a lot of research on path-planning algorithms, and new path-planning algorithms are constantly emerging and developing. The most representative and common path-planning algorithms in the field of path planning are mainly divided into map-based path-planning algorithms, bionics-based path-planning algorithms, and sampling-based path-planning algorithms [7–9].

Sampling-based search algorithms include probability map algorithms and Rapidly-Exploring Random Tree algorithms. The Rapidly-Exploring Random Tree algorithm is a path-planning algorithm proposed by LaValle [10, 11]. Its advantages include the following four aspects: first, it does not need to model the planning space and is a random sampling algorithm; second, it considers the objective constraints of unmanned vehicles; third, it is suitable to solve the path-planning problem under dynamic and multi-obstacle conditions; and fourth, it can be applied to the path-planning problem under the high-dimensional environment. Therefore, it has been widely used [12]. The basic Rapidly-Exploring Random Tree algorithm also has the following disadvantages in path planning: first, the path is randomly generated, the path is biased; second, the random tree is nonoriented in the search process; and third, the convergence speed is slow, and the search efficiency is low [13, 14].

Aiming at the shortcomings of the basic Rapidly-Exploring Random Tree algorithm, researchers at home and abroad have carried out a lot of improvements. Typical improvements include the RRT-Connect algorithm, asymptotically-optimal Rapidly-Exploring Random Tree (RRT*), asymptotically-optimal bidirectional Rapidly-Exploring Random Tree (B-RRT*), and intelligent bidirectional RRT* (IB-RRT) [15–19]. RRT-Connect algorithm was proposed by Kuffner and LaValle in 2000 [15]. The algorithm improves the speed of path finding by generating two random trees in parallel. The Rapidly-Exploring Random Tree algorithm, which is biased to search and bidirectional expansion, improves the convergence speed and search efficiency but does not overcome the randomness when random trees generate nodes [20, 21]. The RRT ∗ algorithm was proposed by Adiyatov and Varol [22]. In 2010, the proposed algorithm solves the nonoptimal problem of the path generated by the RRT algorithm, but the efficiency of path generation is greatly reduced due to the increase in the amount of calculation during exploration. Jordan borrowed the idea of the RRT-Connect algorithm. In 2013, a two-way extended RRT* algorithm (B-RRT*) was proposed. At the same time, the connection function of the RRT-Connect algorithm was improved to ensure that the two trees connected by the algorithm could generate an optimal path. Qureshi proposed the IB-RRT * algorithm in 2015 [19] and introduced a smart sample insertion function in the B-RRT * algorithm to improve the speed of the algorithm's convergence to the optimal path. Wang Daowei proposed the concept of dynamic step size in 2016. The rapid expansion of the dynamic step size random tree algorithm improves the uncertainty of the algorithm and improves the obstacle avoidance ability, but it is impossible to determine the step size for different obstacles [23]. Song Xiaolin proposed to introduce heuristic function into the Rapidly-Exploring Random Tree algorithm in 2017, which makes the search tree more oriented in the search process, but the algorithm is easy to fall into a dead cycle in path planning [24].

This paper proposes an improved intelligent vehicle path-planning algorithm based on a Rapidly-Exploring Random Tree, which uses a cyclic alternating iterative search method to generate new nodes. The bidirectional random tree expands simultaneously. The turning angle constraint of the vehicle is increased, the generated nodes are optimized, and the path is smoothened. The shortcomings of the Rapidly-Exploring Random Tree algorithm in path planning are improved.

2. Vehicle Steering Model

In the process of intelligent vehicle driving, ensure not only the driving safety of the vehicle but also ensure the ride comfort of the passengers on the vehicle. Therefore, when the intelligent vehicle is planning the route, it is necessary to ensure that the vehicle avoids all obstacles and the smoothness of the planned route.

According to Newton's second law and the geometric relationship of steering, the equation of intelligent vehicle's steady-state steering can be derived. In order to facilitate the analysis of the steering state of the intelligent vehicle, a two-wheel model, as shown in Figure 1, is used.

In Figure 1, δ is the wheel angle of the front wheels. When the front wheels turn, the body generates centrifugal force. The lateral reaction forces on the front and rear wheels are F_{y1} and F_{y2}, and the corresponding sideslip angles α_f and α_r are generated. Assuming that the center of mass and instant center of rotation of the vehicle are O and P, respectively, the distance R between the two points is the turning radius. r is the yaw angular velocity, and then the velocity at the center of mass is $v_c = rR$. β is the sideslip angle at the center of mass, that is, the angle between the direction of travel of the vehicle at the center of mass and the X axis.

It is assumed that the vehicle moves only in a plane direction, and there is no vertical direction and roll and pitch movement around the X and Y axes. The speed of the car is a fixed value, and the effects of air resistance and tangential force on the vehicle are ignored. The previous wheel angle was used as the only input, ignoring the effect of the steering system on the vehicle and ignoring the changes in tire characteristics and the effect of returning torque due to load changes on the left wheel. The component of v_c on the X axis is

$$u = v_c \cos\beta, \tag{1}$$

where u is the component of v_c on the X axis, v_c is the velocity at the center of mass, and β is the sideslip angle at the center of mass.

Vehicle's β during the turn is small, and approximately $\cos\beta \approx 1$ is obtained. Therefore,

$$u = v_c = rR, \tag{2}$$

where u is the component of v_c on the X axis, v_c is the velocity at the center of mass, R is the turning radius, and r is the yaw angular velocity.

The component of v_c on the Y axis is

$$v = v_c \sin\beta, \tag{3}$$

where v is the component of $v_c$on the Y axis, v_c is the velocity at the center of mass, and β is the sideslip angle at the center of mass.

From this, the acceleration a_c of the center of mass is

$$a_c = \left(\dot{v} + \frac{u^2}{R}\right) = \left(\dot{v} + \frac{{v_c}^2}{R}\right) = (\dot{v} + v_c r), \tag{4}$$

where a_c is the acceleration of the center of mass, v is the component of v_c on the Y axis, u is the component of v_c on the X axis, R is the turning radius, r is the yaw angular velocity, and v_c is the velocity at the center of mass.

From the force and moment balance equations, the differential motion equation can be derived as

$$F_{y1} + F_{y2} = ma_c = m(\dot{v} + v_c r), \tag{5}$$

$$l_f F_{y1} - l_r F_{y2} = I_z \dot{r}, \tag{6}$$

where F_{y1} and F_{y2} are the lateral reaction forces on the front and rear wheels, m is the mass of the entire vehicle, I_z is the

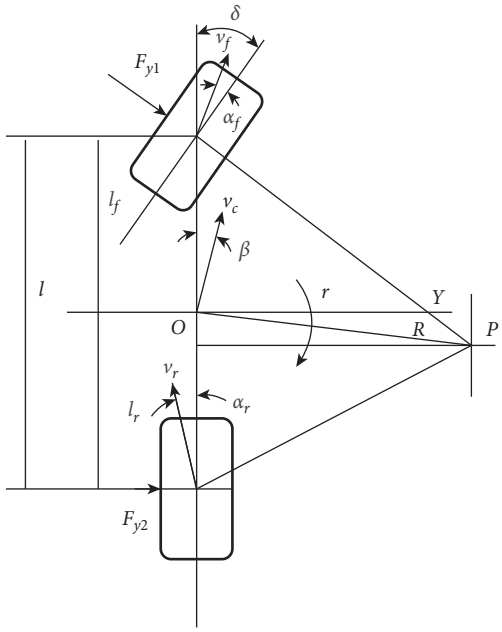


Figure 1: Two-wheel steering model.

moment of inertia of the vehicle body about Z axis, v_c is the velocity at the center of mass, v is the component of v_c on the Y axis, r is the yaw angular velocity, l_f is the distance from the front wheel to the center of mass, and l_r is the distance from the rear wheel to the center of mass.

$$\begin{aligned} F_{y1} &= C_f \alpha_f, \\ F_{y2} &= C_r \alpha_r, \end{aligned} \tag{7}$$

where F_{y1} and F_{y2} are the lateral reaction forces on the front and rear wheels, C_f and C_r are the front and rear tire sideslip stiffness, and α_f and α_r are the front and rear wheel sideslip angles.

This can be obtained from the geometric relationship

$$\begin{aligned} \alpha_f &= l_f \frac{r}{v_c} + \beta - \delta, \\ \alpha_r &= \beta - l_r \frac{r}{v_c}, \end{aligned} \tag{8}$$

where α_f and α_r are the front and rear wheel sideslip angles, l_f is the distance from the front wheel to the center of mass, l_r is the distance from the rear wheel to the center of mass, r is the yaw angular velocity, v_c is the velocity at the center of mass, β is the sideslip angle at the center of mass, and δ is the wheel angle of the front wheels.

Substituting into formula (5) and formula (6), we can get

$$C_f\left(l_f \frac{r}{v_c} + \beta - \delta\right) + C_r\left(\beta - \frac{l_r}{v_c}\right) = m\left(\dot{v} + v_c r\right), \tag{9}$$

$$l_f C_f\left(l_f \frac{r}{v_c} + \beta - \delta\right) + l_r C_r\left(\beta - \frac{l_r}{v_c}\right) = I_z \dot{r}, \tag{10}$$

where C_f and C_r are the front and rear tire sideslip stiffness, l_f is the distance from the front wheel to the center of mass, l_r is the distance from the rear wheel to the center of mass, r is the yaw angular velocity, v_c is the velocity at the center of mass, β is the sideslip angle at the center of mass, δ is the wheel angle of the front wheel, m is the mass of the entire vehicle, and I_z is the moment of inertia of the vehicle body about Z axis.

Get $\beta \approx \sin\beta = v/v_c$ from formula (3), substitute it into formulas (9) and (10), and get

$$m\left(\dot{v} + v_c r\right) = C_f \delta - \frac{v}{v_c}\left(C_f + C_r\right) - \frac{r}{v_c}\left(l_f C_f - l_r C_r\right), \tag{11}$$

$$I_z \dot{r} = l_f C_f \delta - \frac{v}{v_c}\left(l_f C_f - l_r C_r\right) - \frac{r}{v_c}\left(l_f^2 C_f + l_r^2 C_r\right), \tag{12}$$

where m is the mass of the entire vehicle, v_c is the velocity at the center of mass, r is the yaw angular velocity, C_f and C_r are the front and rear tire sideslip stiffness, δ is the wheel angle of the front wheel, v is the component of v_c on the Y axis, l_f is the distance from the front wheel to the center of mass, l_r is the distance from the rear wheel to the center of mass, and I_z is the moment of inertia of the vehicle body about Z axis.

When the front wheel is a step angle input, its response is a constant velocity circular motion, and the yaw angular velocity r is constant, $\dot{r} = 0$ and $\dot{v} = 0$, which can be obtained by substituting into formulas (11) and (12):

$$m v_c r = C_f \delta - \frac{v}{v_c}\left(C_f + C_r\right) - \frac{r}{v_c}\left(l_f C_f - l_r C_r\right), \tag{13}$$

$$0 = l_f C_f \delta - \frac{v}{v_c}\left(l_f C_f - l_r C_r\right) - \frac{r}{v_c}\left(l_f^2 C_f + l_r^2 C_r\right), \tag{14}$$

where m is the mass of the entire vehicle, v_c is the velocity at the center of mass, r is the yaw angular velocity, C_f and C_r are the front and rear tire sideslip stiffness, δ is the wheel angle of the front wheel, v is the component of v_c on the Y axis, l_f is the distance from the front wheel to the center of mass, and l_r is the distance from the rear wheel to the center of mass.

By synchronizing and eliminating the above two types, we can get

$$\frac{r}{\delta} = \frac{v_c/l}{1 + K v_c^2}, \tag{15}$$

where $K = m/I^2\left((l_f/C_f) - (l_r/C_r)\right)$ is called the stability factor and r/δ is called the steady-state yaw rate gain, also called sensitivity.

From formula (15), we get

$$\delta = \frac{r\left(1 + K v_c^2\right)}{v_c/l}. \tag{16}$$

The stability factor is an important index that affects the steering stability performance. Its value is divided into three cases (neutral steering, insufficient steering, and excessive steering) to discuss.

(1) *Neutral Steering*. At this time, $K = 0$. This situation is equivalent to a state of equilibrium where the lateral

acceleration at the vehicle's center of mass is equal to the sideslip angles produced by the front and rear wheels. The yaw rate gain has a linear relationship with the vehicle speed. This steering characteristic is called neutral turning.

(2) *Insufficient Steering.* At this time, $K > 0$, as can be seen from formula (15), the yaw angle speed gain is smaller than that during neutral steering, r/δ is no longer linearly related to the vehicle speed, and $(r/\delta) - v$ is a negative second derivative and exists in a certain speed range The maximum value curve, the stable steering characteristic at this time, is called understeer. The amount of understeer increases as the value of K increases, and the yaw rate gain decreases. In the case of understeer, the degree of sideslip of the front wheels caused by the lateral acceleration at the center of mass will be greater than that of the rear wheels. In order to keep the radius unchanged, the front wheel rotation angle needs to be increased.

(3) *Oversteering.* At this time, $K < 0$, the denominator in formula (15) is less than 1, the yaw rate gain is greater than the neutral steering, and the second derivative of the gain curve is greater than zero. This steering characteristic is called oversteering.

In neutral steering, when the turning radius is constant, the steering angle does not change when the speed changes. The steering angle depends only on the turning radius and wheelbase. When the wheelbase is constant, the larger the turning radius, the smaller the steering angle of the front wheels of the vehicle, the better the stability of the vehicle, and the higher the safety factor.

According to the above analysis, the steering angle of the front wheels of the vehicle is mainly affected by the turning radius. Regardless of vehicle speed, a vehicle with a larger turning radius will have a smaller front wheel side deflection angle, the better the vehicle's maneuverability and stability, and the safety factor will be greatly improved. In order to ensure the smooth and smooth steering, when the RRT is planning the path, the new node needs to meet certain angular constraints to make the generated path closer to the vehicle's motion requirements.

3. Improved Rapidly-Exploring Random Tree Algorithm

3.1. Basic Rapidly-Exploring Random Tree Algorithm. The expansion diagram of the basic RRT algorithm is shown in Figure 2. The basic RRT algorithm takes the initial point x_{init} as the root node of the random tree and selects the random sampling point x_{rand} by searching for the free space. Choose a node x_{near} closest to the random sampling point x_{rand} on the known random tree and connect node x_{near} and node x_{rand}. A new node x_{new} from x_{near} is generated with a certain step size ρ. If there is no collision with the obstacle during the expansion from x_{near} to x_{new}, this new node x_{new} is added to the random tree to generate a random tree. When a child node in a random tree contains a target point x_{goal}, a path from the initial point x_{init} to the target point x_{goal} can be generated in the random tree. Conversely, if a collision occurs, then we discard the expansion.

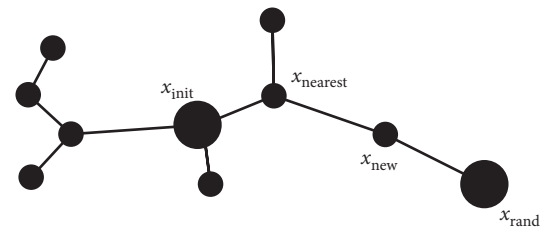


Figure 2: Extended diagram of the Rapidly-Exploring Random Tree algorithm.

3.2. Cyclic Alternating Iterative Searching. The Rapidly-Exploring Random Tree algorithm is based on sampling, which has strong randomness. Aiming at the shortcomings of the basic Rapidly-Exploring Random Tree algorithm such as low search efficiency, nonoriented search process, and slow convergence speed, the basic Rapidly-Exploring Random Tree algorithm is optimized. First, a random sampling point x_{rand} is generated in a random manner as the growth target point of the random tree child node, and then the target point x_{goal} is used as the growth target point of the random tree child node. The two search methods alternately and iteratively search until a feasible path is found or a set threshold of the number of iterations is reached, the search is exited, and the search path fails.

3.3. Bidirectional RRT Algorithm. The bidirectional RRT algorithm defines two random trees in the free space, which select the starting point and the ending point as the random root node, respectively, expanding in the opposite direction. The expansion is ended until the two trees meet. That is to say, the search path is found.

When the random tree with the starting point as the root node searches for the free space to establish the random tree, the random tree with the ending point as the root node is also established. The two random trees generate a new node by turns and detect whether the Euclidean distance between the new node and the other random tree node is less than the set threshold. When the distance between two nodes is less than the set threshold, the two nodes are connected, that is, the two random trees are merged into one random tree to generate a path.

3.4. Increase Angular Constraint. It can be known from the vehicle's steering model that the stability factor is an important index affecting the steering stability performance. The value of K is divided into three cases: neutral steering ($K = 0$), insufficient steering ($K > 0$), and excessive steering ($K < 0$). Ignoring other influencing factors of the vehicle, based on neutral steering, it can be obtained from formula (16) that

$$\delta = \frac{l}{R}. \tag{17}$$

According to the above analysis, the steering angle of the front wheels of the vehicle is affected by the turning radius. In order to ensure the smooth and smooth steering and the driving safety of the vehicle when using the Rapidly-Exploring Random Tree algorithm for path planning, the new node needs to meet certain angular constraints to make the generated path closer to the vehicle's motion requirements and meet the vehicle's kinematics model.

Assume that the angular constraint of the Rapidly-Exploring Random Tree in generating new nodes x_{new} is φ, as shown in Figure 3; when the angle φ_1 between x_{new}, $x_{nearest}$, and x_{init} is less than the constraint value φ, the generated new node is discarded. When φ_2 is greater than the constraint value of φ, the new node is retained and the new node is added to the random tree.

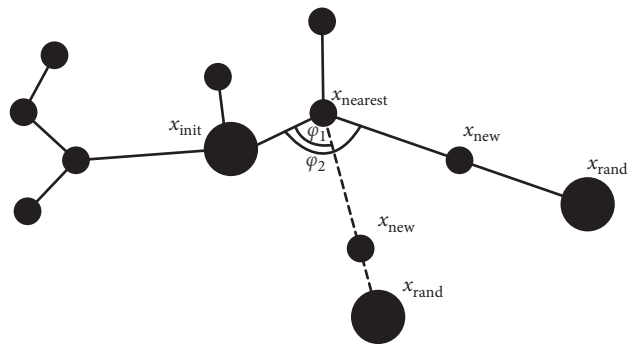


Figure 3: Expansion schema of Rapidly-Exploring Random Tree with increased angle constraints.

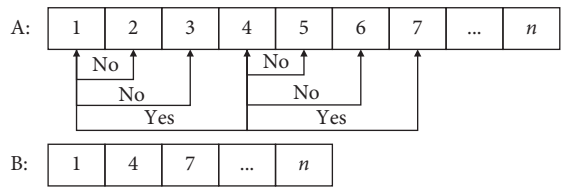


Figure 4: Node optimization.

4. Node Optimization

The Rapidly-Exploring Random Tree algorithm expands in the map with a certain step size, the generated path nodes are too many, and the path has many turning angles, which cannot meet the actual driving conditions of the vehicle. Therefore, the feasible points of the generated path are optimized, redundant nodes are deleted, and the generated path is optimized.

As shown in Figure 4, array A stores the feasible points of the accessible path obtained through the Rapidly-Exploring Random Tree algorithm search. 1 represents the initial point x_{init}, n represents the target point x_{goal}, and $2 \sim (n-1)$ represents the feasible point from the starting point to the target point.

The idea of optimizing nodes is as follows:

(1) Add the initial points x_{init} to the new array B.

(2) Determine whether there is an obstacle between node n and node $n+1$. If not, skip node $n+1$ and determine whether there is an obstacle between node n and the next node $n+2$. If there is an obstacle, add this node $n+2$ to the array B, search backward with $n+2$ as the new node, and so on.

(3) When the target point x_{goal} is found, the search is ended and the target point is added to the array B.

Array B is the optimized node set.

5. Path Smoothing

In this paper, the B-spline curve is selected to smooth the path generated by Rapidly-Exploring Random Tree planning.

The expression of the B-spline curve is

$$P_{i,n}(t) = \sum_{k=0}^{n} P_{i+k} \cdot F_{k,n}(t), \tag{18}$$

$$F_{k,n}(t) = \frac{1}{n!} \sum_{j=0}^{n-k} (-1)^j \cdot C_{n+1}^{j} \cdot (t+n-k-j)^n. \tag{19}$$

In the formula, $0 \le t \le 1$, $i = 0, 1, 2, \ldots, m$, and $k = 0, 1, 2, \ldots, n$.

B-spline curves are defined in sections. If $m+n+1$ vertices $P_i (i = 0, 1, 2, \ldots, m+n)$ are given, then a parametric curve of $m+1$ order n times can be defined.

The B-spline curve has the advantages of geometric invariance, convex hull, convexity, and variation reduction, and the number of control points has little correlation with the order of the function. Therefore, it has a good effect on smoothing the path.

For the feasible path generated by the Rapidly-Exploring Random Tree search, vertices are generated when each feasible node is connected. In order to avoid collision between the path processed by the B-spline curve and the obstacle, local endpoints are generated at both ends of the vertices. As shown in Figure 5, A_{n-1}, A_n, and A_{n+1} are feasible points of the path, and local endpoints B_{n1} and B_{n2} are generated at both ends of fold point A_2, respectively.

Due to the different angles of the vertices, the local endpoints B_{n1} and B_{n2} are selected in an adaptive manner, and the linear function $y(x)$ is determined by points A_{n-1} and A_n:

$$y(x) = A \cdot x + B, \tag{20}$$

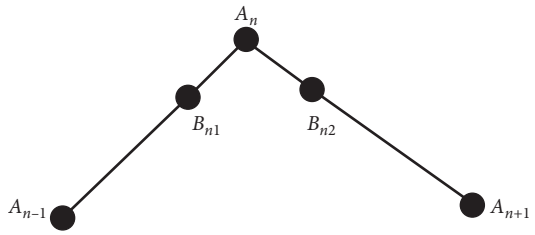


Figure 5: Insert local endpoints.

where A and B are constants.

The abscissa of B_{n1} satisfies

$$\left|x\left(B_{n1}\right) - x\left(A_n\right)\right| = \frac{1}{m}\left|x\left(A_n\right) - x\left(A_{n-1}\right)\right|, \tag{21}$$

where m is the adaptive coefficient.

Substituting the abscissa $x(B_{n1})$ of B_{n1} into equation (19) to obtain 2, calculate the coordinates of point B_{n1}, and then calculate the coordinates of point B_{n2}. The local endpoints B_{n1} and B_{n2} are added to the feasible points to smooth the path.

6. Simulation Experiment Analysis

Whether the improved Rapidly-Exploring Random Tree algorithm meets the requirements and whether the intelligent vehicle can accurately reach the set target position and complete the path planning need to be verified and analyzed by software. Therefore, MATLAB software is used to build a simulation experiment platform to verify the correctness of the improved Rapidly-Exploring Random Tree algorithm. At the same time, it is compared and verified with the basic Rapidly-Exploring Random Tree algorithm. The simulation map is a 500*500 two-dimensional grid space. The simulation experiment conditions are shown in Table 1.

6.1. Increased Angular Constraint Simulation Experiment Results. In a two-dimensional grid space with a simulation map of 500*500, a point (250, 250) is used as the root node of the rapid expansion random tree to generate a random tree around the two-dimensional map. The schematic diagram of random tree expansion without increasing angle constraint is shown in Figure 6, and the schematic diagram of random tree expansion with increasing angle constraint is shown in Figure 7.

Analyzing, Figure 6, the Rapidly-Exploring Random Tree algorithm without increasing the angle constraint, the size of the angle between the feasible points generated cannot be determined, and too small an angle cannot satisfy the vehicle's kinematic model. Analyzing, Figure 7, an angular constraint Rapidly-Exploring Random Tree is added, the angle between feasible points is smoother during the expansion process. Increasing the angle between feasible points meets the constraint requirements, and the generated path is more in line with the actual needs of the vehicle.

6.2. Simulation Results of the Rapidly-Exploring Random Tree Algorithm. In a two-dimensional grid space with a simulation map of 500*500, point (10, 10) is the starting point and (490, 490) is the target point. Black squares represent obstacles in the two-dimensional map and are impassable.

Table 1: Computer parameters.

Name	Model
Processor	Inter (R) core (TM)
Hard disk	1T
RAM	16.00 G
Operating system	Windows 10

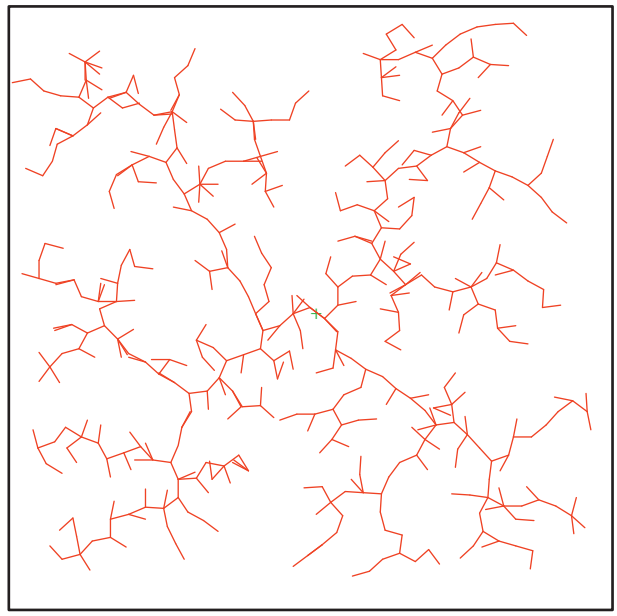


Figure 6: Extended diagram of the Rapidly-Exploring Random Tree.

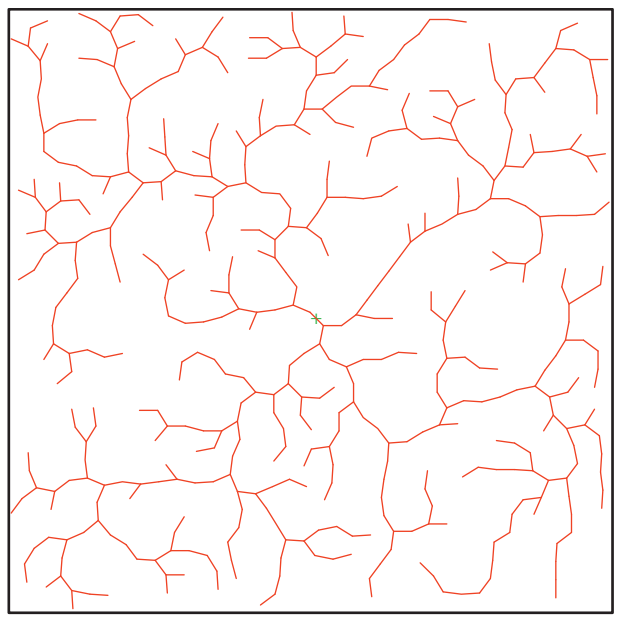


Figure 7: Expansion diagram of the random tree with increased angle constraints.

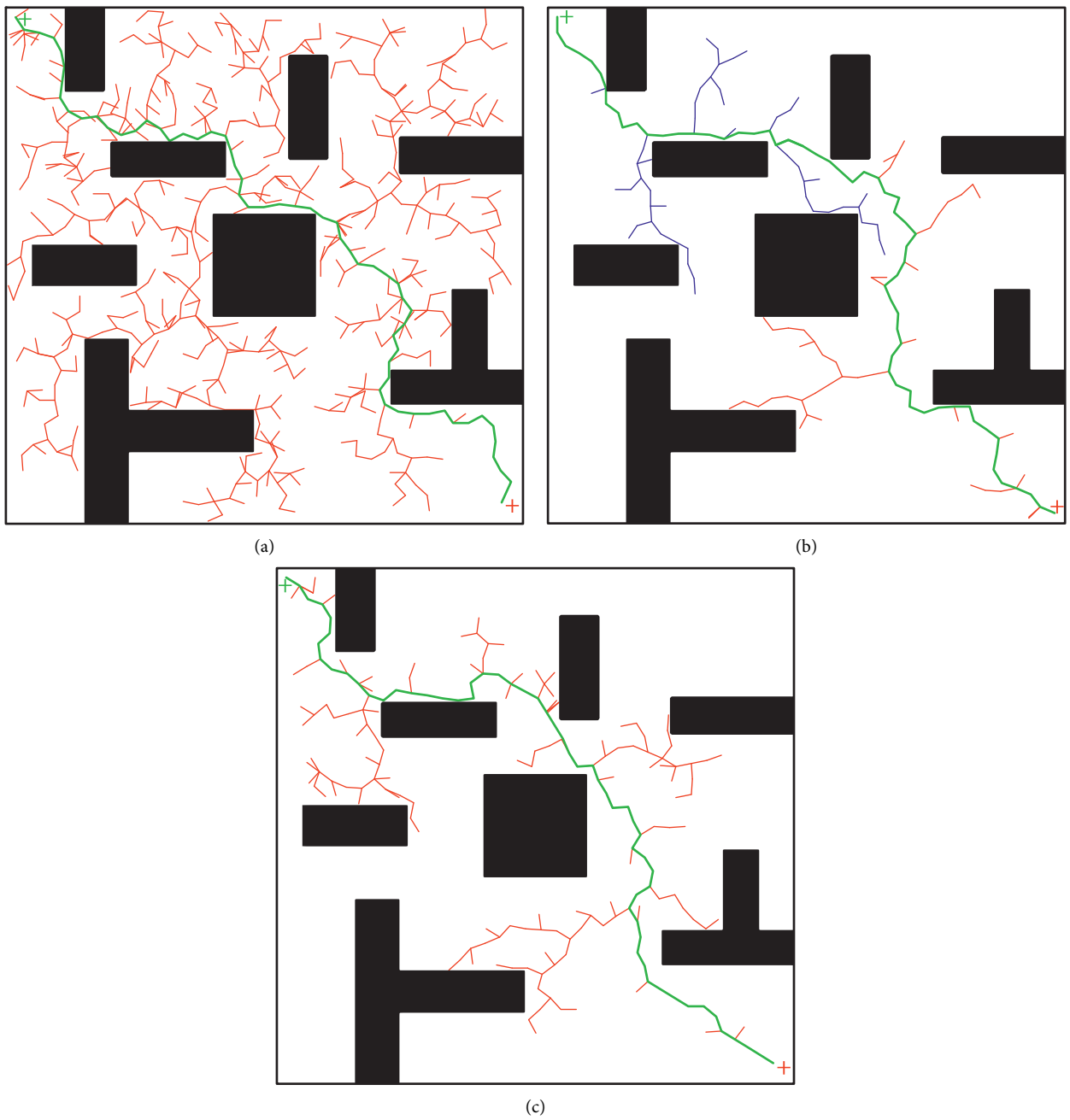


Figure 8: Experimental results of the Rapidly-Exploring Random Tree algorithm, (a) basic algorithm, (b) bidirectional algorithm, and (c) heuristic algorithm.

The simulation results of the basic Rapidly-Exploring Random Tree algorithm are shown in Figure 8(a). The simulation results of the bidirectional Rapidly-Exploring Random Tree algorithm are shown in Figure 8(b). The simulation results of the heuristic Rapidly-Exploring Random Tree algorithm are shown in Figure 8(c).

As shown in Figure 8, red and blue indicate a generated random tree, and green indicates the Rapidly-Exploring Random Tree algorithm to search a feasible path generated in a two-dimensional space. As shown in Figure 8(a), the basic Rapidly-Exploring Random Tree algorithm is non-oriented in the search process, randomly generates new nodes, has large randomness, and generates a large number of nodes. As shown in Figures 8(b) and 8(c), although the bidirectional Rapidly-Exploring Random Tree algorithm and the heuristic Rapidly-Exploring Random Tree algorithm improve the search efficiency of the algorithm, the generated path is of poor quality, has bias, and cannot meet the driving state of the vehicle.

6.3. Simulation Results of the Improved Rapidly-Exploring Random Tree Algorithm. In a two-dimensional grid space with a simulation map of 500*500, point (10, 10) is the

starting point and (490, 490) is the target point. Black squares represent obstacles in the two-dimensional map and are impassable. The simulation results of the improved Rapidly-Exploring Random Tree algorithm are shown in Figure 9.

As shown in Figure 9, red and blue represent the bidirectional random tree generated with the starting point (10, 10) and the target point (490, 490) as the root node, respectively, and green is the feasible path generated by the improved Rapidly-Exploring Random Tree algorithm to search the two-dimensional space. Analyzing Figures 8 and 9, the improved Rapidly-Exploring Random Tree algorithm uses a cyclic alternating iterative search method, which greatly improves the randomness of the basic Rapidly-Exploring Random Tree algorithm. The number of nodes generated is significantly reduced, and the number of iterations of the algorithm is reduced.

Node optimization is performed on the feasible path generated in Figure 9, and the node optimization simulation experiment results are shown in Figure 10.

As shown in Figure 10, blue is the feasible path in the two-dimensional space after node optimization. In Figure 9, nodes are not optimized. Although a feasible path is generated, there are many feasible points, the path length is large, and the curvature of the path cannot meet the driving conditions of the vehicle, which is not the optimal path. In Figure 10, node optimization is added to remove redundant nodes, greatly reducing the number of feasible path nodes, reducing the length of the path, improving the feasibility of the path, and making the generated path more consistent with the actual running track of the vehicle.

The smoothness processing is performed on the path generated in Figure 10, and the result of path smoothing simulation is shown in Figure 11.

As shown in Figure 11, red is a feasible path in a two-dimensional space after path smoothing. Analyzing Figures 10 and 11, inserting local endpoints at both ends of the vertices makes the path vertices smoother, improves the accuracy of the Rapidly-Exploring Random Tree algorithm, greatly improves the quality of the path, and is more in line with the actual conditions of vehicle driving.

The improved algorithm, the basic Rapidly-Exploring Random Tree algorithm, the bidirectional Rapidly-Exploring Random Tree algorithm, and the heuristic Rapidly-Exploring Random Tree algorithm are simulated 50 times, respectively, under the premise of the same step size, and the average running time, average path length, and average number of iterations of the simulation experiment are calculated as shown in Figures 12–14 and Table 2.

Analysis of Figures 12–14 and Table 2 shows that compared with the basic Rapidly-Exploring Random Tree algorithm, the bidirectional Rapidly-Exploring Random Tree algorithm, and the heuristic Rapidly-Exploring Random Tree algorithm, the algorithm in this paper has shorter running time, shorter path length, and fewer iterations. The randomness of the Rapidly-Exploring Random Tree algorithm is reduced, the number of iterations of the algorithm is reduced, and the convergence speed of the algorithm is accelerated. All new nodes grow towards the target point, avoiding global search, improving the real-time performance of unmanned vehicle movement, and easier to obtain the optimal path, improving the bias of the basic Rapidly-Exploring Random Tree algorithm.

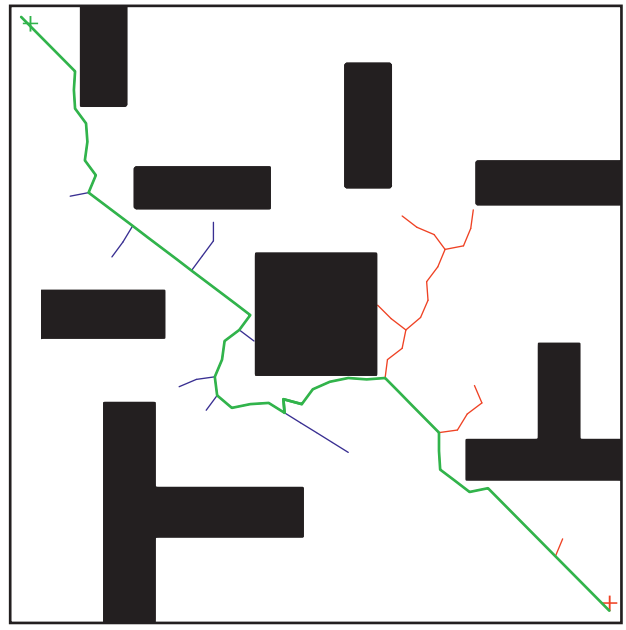


Figure 9: Experimental results of the improved Rapidly-Exploring Random Tree algorithm.

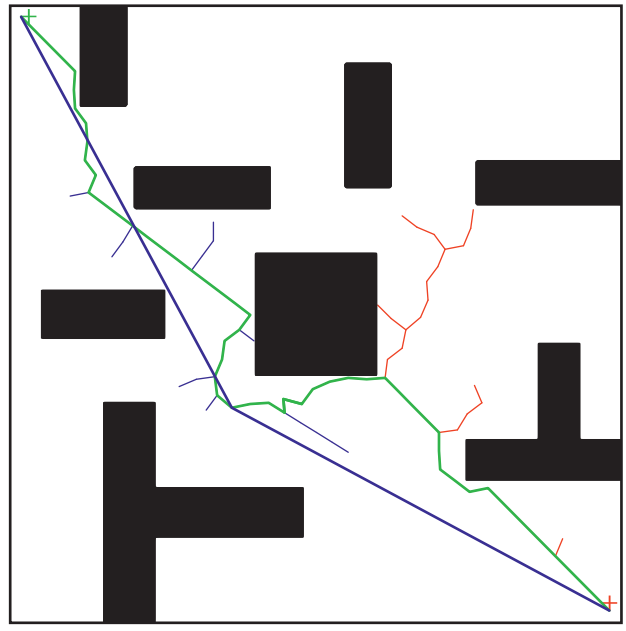


Figure 10: Experimental results of node optimization.

6.4. Analysis of Simulation Results in Different Environments. In order to further verify the advantages and disadvantages of the improved algorithm, simulation analysis is carried out in different scenarios.

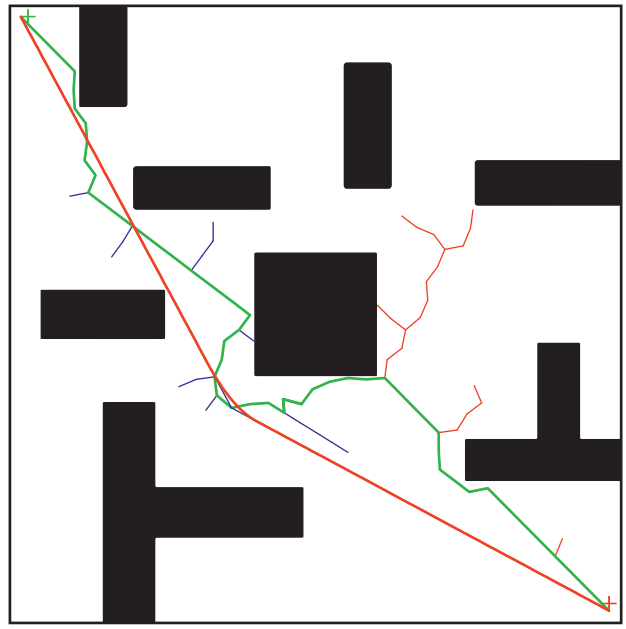


Figure 11: Experimental results of path smoothing.

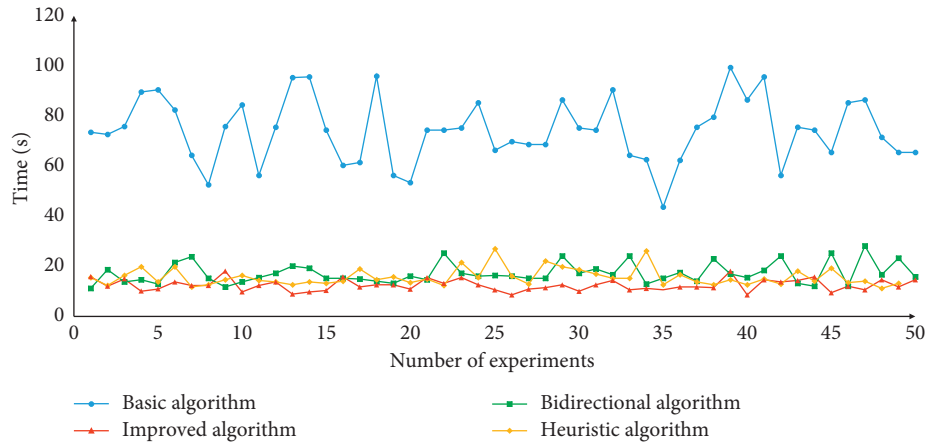


Figure 12: Time comparison.

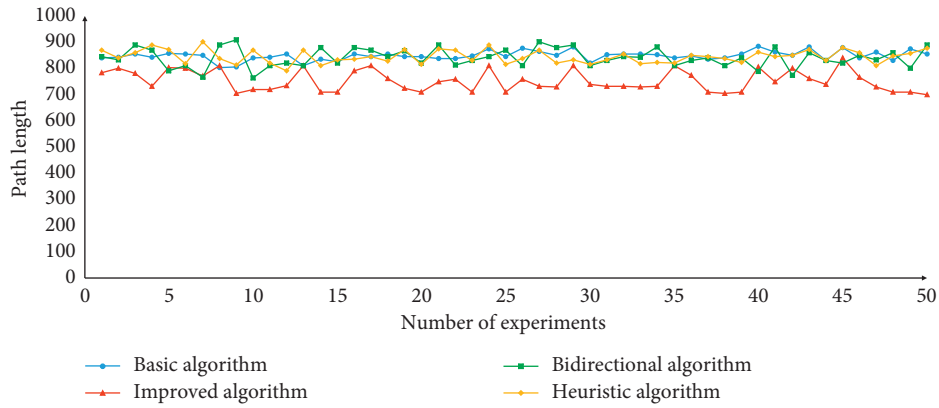


Figure 13: Path length comparison.

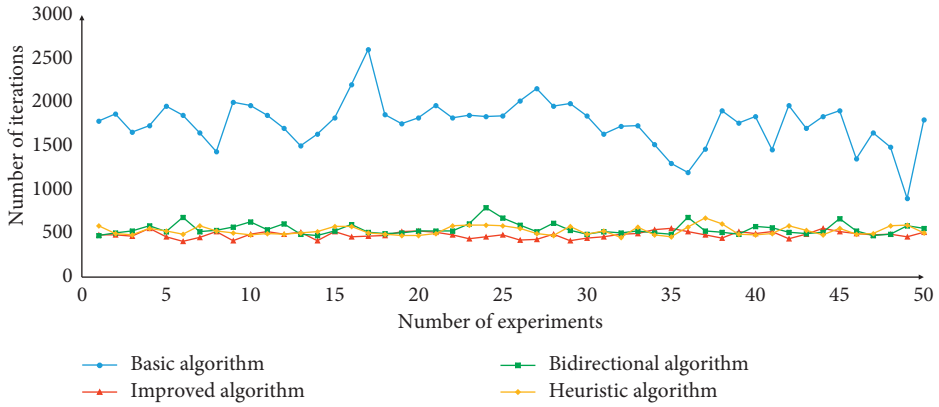
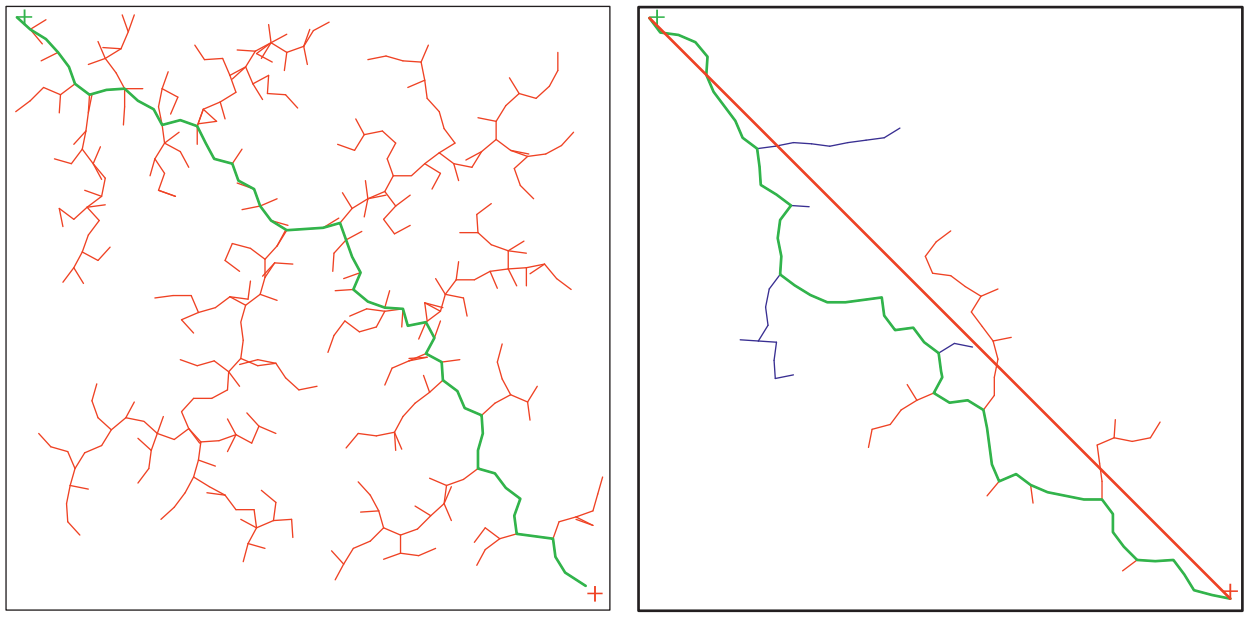


Figure 14: Iteration number comparison.

Table 2: Data comparison.

Algorithm	Average running time (s)	Average path length	Average iterations
Improved algorithm	12.41	753.28	487
Basic algorithm	74.04	848.94	1762
Bidirectional algorithm	17.052	841.30	553
Heuristic algorithm	15.52	845.31	533



(a) (b)

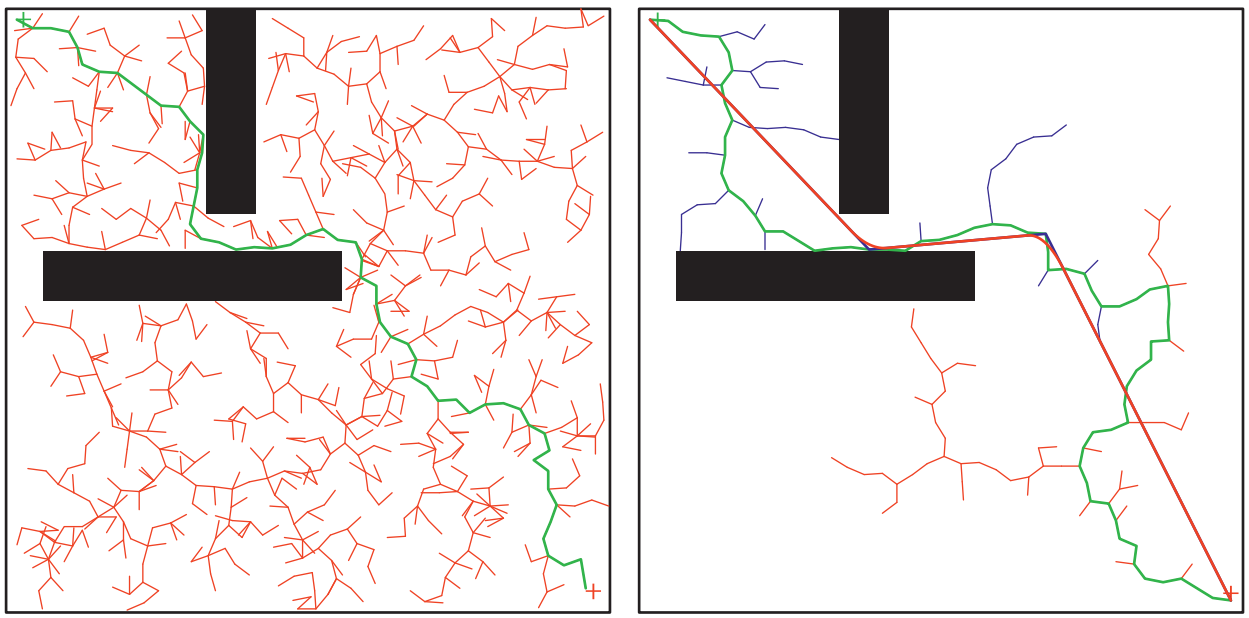
Figure 15: Scene 1: (a) original algorithm; (b) improved algorithm.

6.4.1. Scene 1. No obstacles in the environment: when there is no obstacle in the environment, the simulation results of the basic Rapidly-Exploring Random Tree algorithm and the improved algorithm are shown in Figure 15.

6.4.2. Scene 2. Narrow channels in the environment: when there are narrow channels in the environment, the simulation results of the basic Rapidly-Exploring Random Tree algorithm and the improved algorithm are shown in Figure 16.

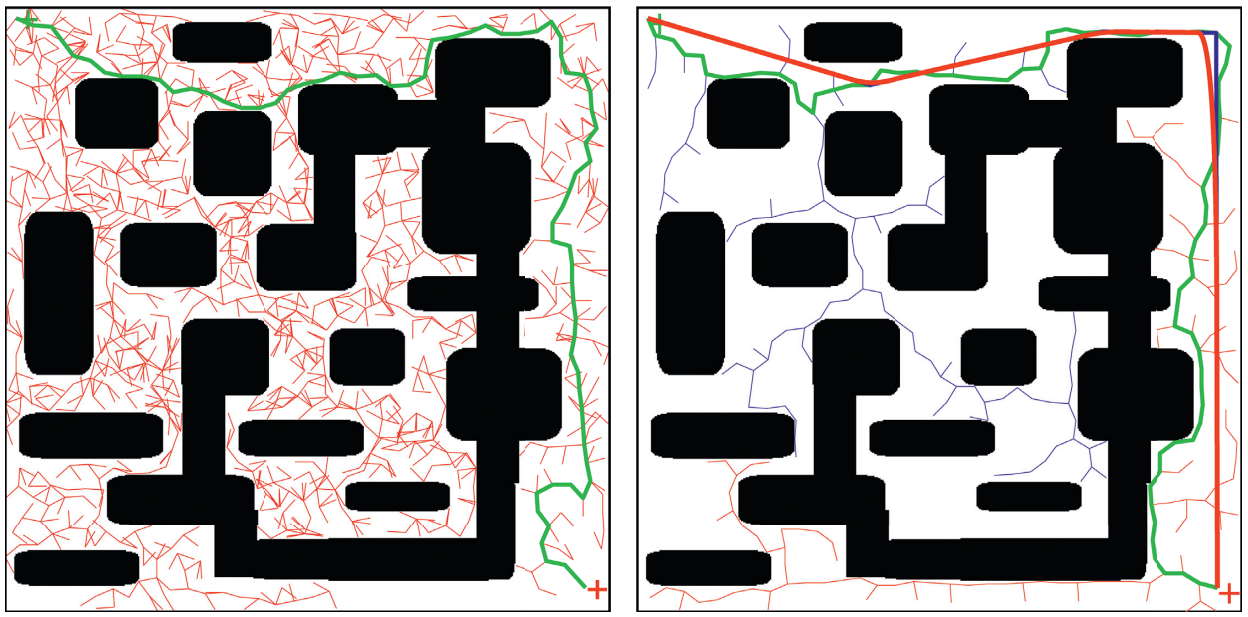
6.4.3. Scene 3. Complex obstacles in the environment: when there are complex obstacles in the environment, the simulation results of the basic Rapidly-Exploring Random Tree algorithm and the improved algorithm are shown in Figure 17.

Analysis of Figures 15–17 shows that, in different scenarios, based on the vehicle turning model, the improved algorithm addresses the problem of too many feasible nodes in the initial path and cannot meet the vehicle's driving conditions. The redundant nodes in the feasible nodes are removed, which greatly shortens the length of the path.



(a) (b)

Figure 16: Scene 2: (a) original algorithm; (b) improved algorithm.



(a) (b)

Figure 17: Scene 3: (a) original algorithm; (b) improved algorithm.

Finally, the vertices in the path are smoothed to make the path vertices smoother, the accuracy of the algorithm is improved, and the quality of the path is greatly improved.

6.5. Path following Simulation. In order to verify the feasibility of the path, a car model was built using CarSim simulation software, and the steering controller based on the pure tracking model was used to track the path planning results.

In the CarSim environment, the vehicle model parameters shown in Figure 18 are established, and some important parameters are shown in Figure 18.

In the Matlab/CarSim environment, a steering controller model based on a pure tracking model is built, and the software interface between CarSim and Matlab is used to

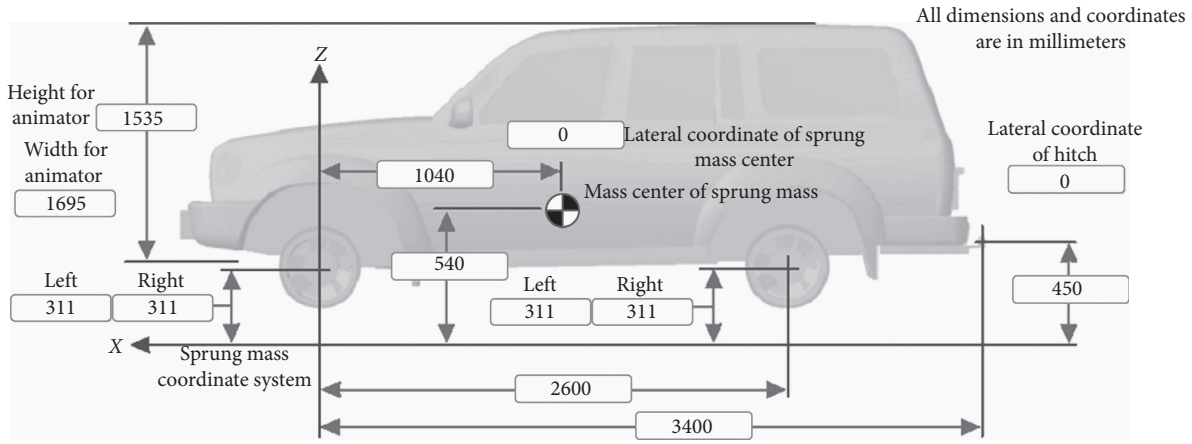


Figure 18: Vehicle model parameters.

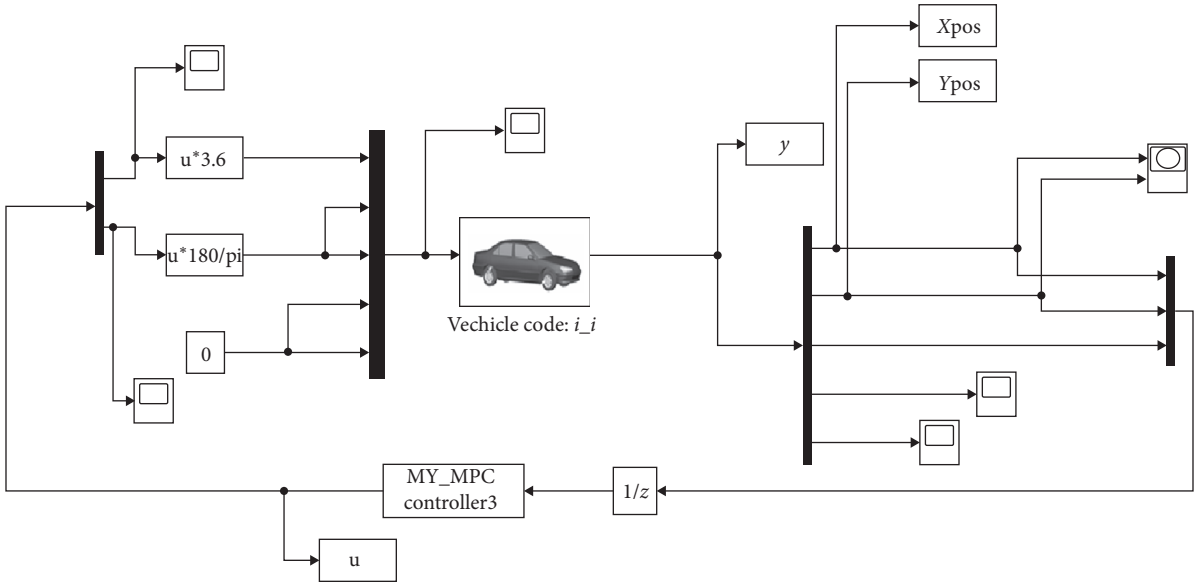
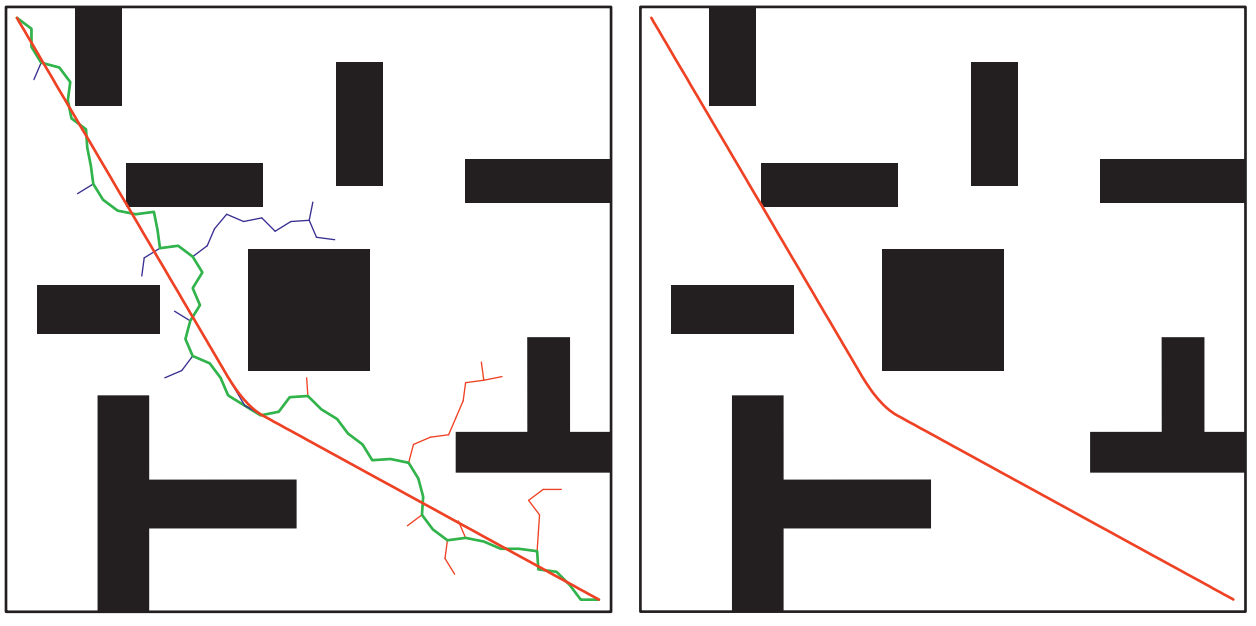


Figure 19: Joint simulation model.



(a) (b)

Figure 20: Extraction of the path, (a) original path, and (b) extracted path.

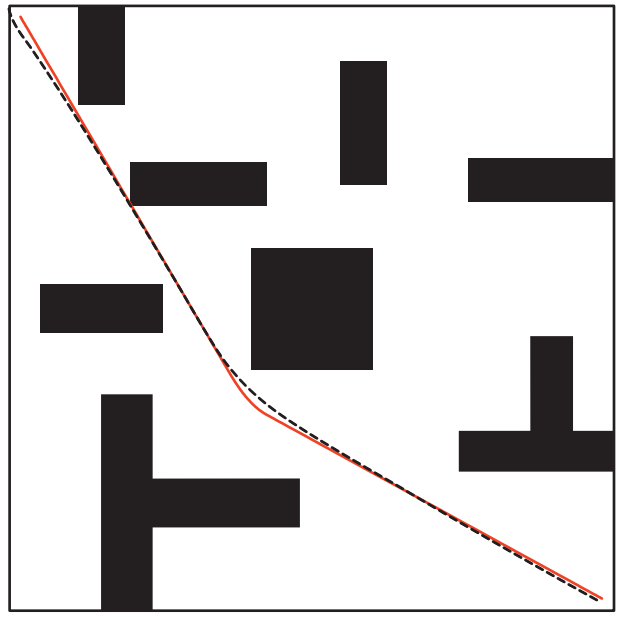


Figure 21: Path tracking results.

integrate the vehicle model and the steering controller model, as shown in Figure 19.

In order to be able to put the planned path obtained by the improved algorithm into the joint simulation model, it is necessary to output the data of the path planning result and output the planned path in the form of two-dimensional coordinates. The output result is shown in Figure 20.

The path output by the improved algorithm is extracted and input to the controller simulation model, and the controller model is allowed to track the path and obtain the corresponding tracking path. The tracking result is shown in Figure 21.

As can be seen from Figure 21, the red path is the planned path, and the black dotted line is the tracking path. The path planned by the improved algorithm differs slightly from the tracked path, and the curve curvature changes more smoothly, so the path generated by the improved algorithm is more in line with the driving characteristics of the actual vehicle. For intelligent vehicles, the actual steering controller of the entire vehicle and the steering controller set by CarSim use the same type of controller, and then the optimized path will also be more in line with the constraints of the vehicle. So, for vehicles, the optimized path is easier to track, and for passengers, riding is more comfortable.

7. Concluding Remarks

In this paper, a new node is generated by cyclic iterative search, and a two-way random tree is used to search at the same time. The basic Rapidly-Exploring Random Tree algorithm is improved and optimized by adding the turning angle constraint of the vehicle. The problems of large randomness, slow convergence speed, and deviation of the basic Rapidly-Exploring Random Tree algorithm are solved. The node optimization of the generated path shortens the length of the path, and the smoothness of the path is processed to make the generated path more in line with the driving conditions of the vehicle. However, whether it is a Rapidly-Exploring Random Tree algorithm, particle swarm algorithm, or an improved Rapidly-Exploring Random Tree algorithm, they are model-driven and have certain limitations; further research is needed to combine data-driven path planning and obstacle avoidance for intelligent vehicles and ultimately make the technology of smart vehicles more mature.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] L. F. Zhu, J. F. Yang Ji, Y. Y. Shi, and P. P. Fang, "Research progress on lateral control strategy of unmanned vehicles," *World Sci-Tech R & D*, vol. 40, no. 5, pp. 506–518, 2018.

[2] I. Noreen, A. Khan, H. Ryu, N. L. Doh, and Z. Habib, "Optimal path planning in cluttered environment using RRT-AB," *Intelligent Service Robotics*, vol. 11, no. 1, pp. 41–52, 2018.

[3] Z. R. Zhi, S. H. Jiang, W. W. Yuan, and C. H. Shi, "Robot path-planning based on deep Q-learning," *Measurement & Control Technology*, vol. 38, no. 7, pp. 24–28, 2019.

[4] P. P. Fang, J. F. Yang, Y. Y. Shi, and L. Y. Yu, "Gradient descent method and improved artificial potential field method for obstacle avoidance of unmanned vehicle," *Manufacturing Automation*, vol. 40, no. 11, pp. 81–84, 2018.

[5] H. Fazlollahtabar and S. Hassanli, "Hybrid cost and time path planning for multiple autonomous guided vehicles," *Applied Intelligence*, vol. 48, no. 2, pp. 482–498, 2017.

[6] P. Wang, Q. Y. Wang, M. S. Wan, and N. Chen, "A fractional derivative-based lateral preview driver model for autonomous automobile path tracking," *Mathematical Problems in Engineering*, vol. 2018, Article ID 7320413, 9 pages, 2018.

[7] A. H. Hasan and A. M. Mosa, "Multi-robot path planning based on max–min ant colony optimization and D algorithms in a dynamic environment," in *Proceedings of the 2018 International Conference on Advanced Science and Engineering (ICOASE)*, pp. 110–115, Duhok, Iraq, October 2018.

[8] Y. Rasekhipour, A. Khajepour, S.-K. Chen, and B. Litkouhi, "A potential field-based model predictive path-planning controller for autonomous road vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1255–1267, 2017.

[9] J. Ballesteros, C. Urdiales, A. B. M. Velasco, and G. Ramos-Jimenez, "A biomimetical dynamic window approach to navigation for collaborative control," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 6, pp. 1123–1133, 2017.

[10] S. M. Lavalle, "Rapidly-exploring random trees:a new tool for path-planning," Technical Report 98-11, Computer Science Department, Iowa State University, Ames, USA, 1998.

[11] S. M. Lavalle and J. J. Kuffner, "Rapidly-exploring random trees: progress and prospects," in *Algorithmic and Computational Robotics New Direction*, B. Donald, Ed., pp. 293–308, CRC Press, Boca Raton, FL, USA, 2000.

[12] H. F. Qiao, G. Z. Pan, and Q. Yin, "Research and simulation of real— time path-planning for mobile robot," *Computer Simulation*, vol. 32, no. 1, pp. 406–410, 2015.

[13] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed RRT: optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2997–3004, Chicago, IL, USA, September 2014.

[14] A. A. Doshi, A. J. Postula, A. Fletcher, and S. P. N. Singh, "Development of micro-UAV with integrated motion planning for open-cut mining surveillance," *Microprocessors and Microsystems*, vol. 39, no. 8, pp. 829–835, 2015.

[15] J. J. Kuffner and S. M. LaValle, "RRT-connect: an efficient approach to single-query path planning," in *Proceedings of the 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings*, vol. 2, pp. 995–1001, San Francisco, CA, USA, April 2000.

[16] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[17] Z. Tahir, A. H. Qureshi, Y. Ayaz, and R. Nawaz, "Potentially guided bidirectionalized RRT for fast optimal path planning in cluttered environments," *Robotics and Autonomous Systems*, vol. 108, pp. 13–27, 2018.

[18] M. Jordan and A. Perez, "Optimal bidirectional rapidly-exploring random trees," Computer Science and Artificial Intelligence Laboratory Technical Report, IEEE, Piscataway, NJ, USA, 2013.

[19] A. H. Qureshi and Y. Ayaz, "Intelligent bidirectional rapidly-exploring random trees for optimal motion planning in complex cluttered environments," *Robotics and Autonomous Systems*, vol. 68, pp. 1–11, 2015.

[20] K. Shi, J. Denny, and N. M. Amato, "Spark PRM: using RRTs within PRMs to efficiently explore narrow passages," in *Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4659–4666, Hong Kong, China, June 2014.

[21] J. Denny, M. Morales, S. Rodriguez, and N. M. Amato, "Adapting RRT growth for heterogeneous environments," in *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1772–1778, Tokyo, Japan, November 2013.

[22] O. Adiyatov and H. A. Varol, "A novel RRT-based algorithm for motion planning in Dynamic environments," in *Proceedings of the 2017 IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 1416–1421, Takamatsu, Japan, August 2017.

[23] D. W. Wang, M. F. Zhu, and H. Liu, "Rapidly—exploring random tree algorithm based on dynamic step," *Computer Technology and Development*, vol. 26, no. 3, pp. 105–107, 2016.

[24] X. L. Song, N. Zhou, Z. Y. Huang, and H. T. Cao, "An improved RRT algorithm of local path-planning for vehicle collision avoidance," *Journal of Hunan University (Natural Sciences)*, vol. 44, no. 4, pp. 30–37, 2017.