processes

MDPI

Article

Research on Path Planning and Tracking Control of Autonomous Vehicles Based on Improved RRT* and PSO-LQR

Yong Zhang, Feng Gao and Fengkui Zhao *

College of Automobile and Traffic Engineering, Nanjing Forestry University, Nanjing 210037, China; zy.js@163.com (Y.Z.); gaofeng@njfu.edu.cn (F.G.)
* Correspondence: zfk@njfu.edu.cn; Tel.: +86-159-5185-3958

Abstract: Path planning and tracking control are essential parts of autonomous vehicle research. Regarding path planning, the Rapid Exploration Random Tree Star (RRT*) algorithm has attracted much attention due to its completeness. However, the algorithm still suffers from slow convergence and high randomness. Regarding path tracking, the Linear Quadratic Regulator (LQR) algorithm is widely used in various control applications due to its efficient stability and ease of implementation. However, the relatively empirical selection of its weight matrix can affect the control effect. This study suggests a path planning and tracking control framework for autonomous vehicles based on an upgraded RRT* and Particle Swarm Optimization Linear Quadratic Regulator (PSO-LQR) to address the abovementioned issues. Firstly, according to the driving characteristics of autonomous vehicles, a variable sampling area is used to limit the generation of random sampling points, significantly reducing the number of iterations. At the same time, an improved Artificial Potential Field (APF) method was introduced into the RRT* algorithm, which improved the convergence speed of the algorithm. Utilizing path pruning based on the maximum steering angle constraint of the vehicle and the cubic B-spline algorithm to achieve path optimization, a continuous curvature path that conforms to the precise tracking of the vehicle was obtained. In addition, optimizing the weight matrix of LQR using POS improved path-tracking accuracy. Finally, this article's improved RRT* algorithm was simulated and compared with the RRT*, target bias RRT*, and P-RRT*. At the same time, on the Simulink–Carsim joint simulation platform, the PSO-LQR is used to track the planned path at different vehicle speeds. The results show that the improved RRT* algorithm optimizes the path search speed by 34.40% and the iteration number by 33.97%, respectively, and the generated paths are curvature continuous. The tracking accuracy of the PSO-LQR was improved by about 59% compared to LQR, and its stability was higher. The position error and heading error were controlled within 0.06 m and 0.05 rad, respectively, verifying the effectiveness and feasibility of the proposed path planning and tracking control framework.

Keywords: autonomous vehicle; path planning; tracking control; RRT*; linear quadratic regulator; particle swarm optimization

Citation: Zhang, Y.; Gao, F.; Zhao, F. Research on Path Planning and Tracking Control of Autonomous Vehicles Based on Improved RRT* and PSO-LQR. *Processes* **2023**, *11*, 1841. https://doi.org/10.3390/pr11061841

Academic Editors: Raul D. S. G. Campilho, Tao Zhao, Xiangpeng Xie and Songyi Dian

Received: 10 May 2023
Revised: 1 June 2023
Accepted: 16 June 2023
Published: 19 June 2023

1. Introduction

In recent years, the rapid development of artificial intelligence has driven the technological progress of self-driving vehicles and intelligent transportation systems. Autonomous vehicles are one of the core components of intelligent transportation systems with increasing safety and reliability requirements [1–4]. Path planning and tracking control are two core issues in the autonomous driving system. The main goal of path planning is to efficiently find a collision-free and feasible path from the starting point to the target point in the workspace, while path tracking control solves the problem of how to control the vehicle to move along the pre-planned path [5,6]. The planned path's quality directly affects the vehicle's driving performance during autonomous driving. In contrast, the ability to accurately and stably track the planned path directly affects the vehicle's driving

safety. Therefore, path planning and tracking performance are significant for autonomous driving vehicles.

Scholars have proposed many practical algorithms and methods to address the challenges and difficulties in path planning for autonomous vehicles. Path planning algorithms can be divided into five types based on their principles: graph search methods, sampling methods, intelligent bio-inspired methods, numerical optimization methods, and artificial potential field methods [7,8]. Among them, the Rapidly-Exploring Random Tree (RRT), a sampling-based path planning method, is widely used in autonomous vehicle path planning due to its advantages, such as probability completeness, efficiency, scalability, and robustness [9]. However, the traditional RRT algorithm has limitations regarding long path search time and slow algorithm convergence speed. To address the limitations of the traditional RRT algorithm, scholars have proposed many improved RRT methods. A path planning technique based on Goal-Biased RRT [10] has been developed to increase the planning efficiency while also increasing the search efficiency of the algorithm by adding a probability element to the random sampling to create a non-random sampling extension mode. Additionally, some scholars have proposed the RRT-Connect algorithm [11], which generates random trees separately from the starting and target nodes, reducing the search space and improving search speed. To address the shortcomings of the RRT algorithm in ensuring asymptotic optimality, the RRT* algorithm changes the search mode by reselecting the parent node and rewiring, thus generating paths with the best or approximate best length [12].

In order to improve the convergence performance and feasibility of the RRT* algorithm, scholars have proposed the Informed-RRT* [13], which uses heuristic methods to reduce the planning problem to a subset of the original domain, thus improving the algorithm's search speed. However, its performance is sensitive to the initial solution. Other researchers have attempted to introduce additional algorithmic features to improve RRT and enhance the efficiency of path planning. Qureshi et al. [14] proposed P-RRT*, which uses potential artificial fields to obtain new nodes and reduce the time required to expand to the goal. Chen et al. [15] proposed a VPF-RRT* algorithm for path planning, which introduces virtual potential fields into the RRT* algorithm to adjust the positions of path nodes and solve the problem of excessive randomness in the RRT algorithm. Fan et al. [16] proposed a trajectory planning method based on the target-biased bidirectional APF-RRT* algorithm, which integrates the Artificial Potential Field (APF) method with the Bi-RRT* algorithm that has a target-bias strategy, significantly reducing redundant points. Ayawli et al. [17] proposed an improved heuristic RRT-A* algorithm, in which the A-Star (A*) cost function is introduced into the RRT algorithm to optimize performance. Meanwhile, several metric functions are used as heuristic information functions to measure the performance of different metric functions, optimizing the planning path and reducing computational costs. However, although these algorithms have made significant improvements compared to the traditional RRT, they have yet to consider the steering constraints of the wheels, which means that they have not been directly applied to path planning for autonomous vehicles. Therefore, Ghosh et al. [18] proposed a kinematic constraint Bi-RRT (KB-RRT) algorithm, which limits the number of generated nodes without affecting accuracy and combines kinematic constraints to generate smooth trajectories. Peng et al. [19] proposed a path planning algorithm based on a kinematic constraint RRT integrated with the trajectory parameter space (TP space), introducing a new method to select candidate nodes for tree expansion and integrating RRT with TP space to meet kinematic constraints and improve performance through narrow passages. Liao et al. [20] proposed the Stack-RRT*, which can be combined with smoothing schemes based on different parameter curves to generate feasible paths with different continuity. These algorithms have significantly improved planning efficiency, reduced computational cost, and met kinematic constraints. However, further research and improvements are still needed to achieve more efficient, precise, and safe path planning for autonomous vehicles.

In addition to research on path planning techniques, tracking and control technology are also essential in autonomous vehicles. Generally, there are two types of autonomous vehicle path-tracking control methods: geometric-based methods and model-based methods. Geometric-based methods include the pure pursuit [21] and Stanley methods [22], which are widely used for handling vehicle or robot path tracking problems due to their simplicity and good performance. However, geometric methods are only suitable for vehicles that ignore their dynamic characteristics. In contrast, the model-based path-tracking method is more suitable for autonomous vehicles in real driving scenarios. Model-based tracking methods can be divided into methods based on the kinematics model and methods based on the dynamic model. The control method based on a dynamic model improves the safety and reliability of vehicles under complex working conditions [23]. There are many control methods based on dynamic models, such as Proportional–Integral–Differential (PID) control [24], Fuzzy Logic control [25], Sliding Mode control [26], Model Predictive control [27], and Linear Quadratic Regulator (LQR) control [28]. LQR controllers are widely used in autonomous vehicle path tracking due to their high-precision tracking performance, algorithmic simplicity, applicability to nonlinear systems, and ability to consider dynamic constraints. However, while LQR methods can stabilize the system, they do not consider disturbance terms and can lead to system errors. Therefore, Kapania et al. [29] proposed a feedback + feedforward steering controller to maintain vehicle stability under extreme maneuvering conditions while minimizing the path deviation. However, at high speeds, the steady-state path deviation significantly increases. Xu et al. [30] added path curvature feedforward control to LQR feedback control to reduce the steady-state error of the controller. Yang et al. [31] proposed a feedforward + predictive LQR Lateral control method, which suits intelligent vehicles' lateral tracking control problem under complex working conditions. Although research based on the LQR control methods has been widely applied to autonomous vehicles, problems, such as input instability in solving LQR, can affect the controller's performance. Therefore, many researchers have improved control effects by adding optimization algorithms to optimize LQR. Lu et al. [32] proposed an adaptive LQR Path-Tracking controller, which uses a genetic algorithm to optimize the parameters of LQR and applies a fitness function that considers tracking accuracy and vehicle stability. The results show the effectiveness of the controller in improving tracking accuracy and vehicle stability. Wang et al. [33] proposed a lateral path control strategy based on an improved LQR algorithm, which uses fuzzy control to adjust the weight coefficients of LQR in real time according to the vehicle state, improving tracking accuracy, steering stability, and computational efficiency. The improvements made to the LQR algorithm in these studies have shown promising results, but there is still room for further improvement in control accuracy and stability.

The literature review above provides an overview of various research achievements on path planning and tracking control strategies for autonomous vehicles, with numerous studies conducted by scholars in this field. However, the performance of path planning and tracking control for autonomous driving still needs to be improved, and a comprehensive framework is needed to cope with different driving scenarios. This paper proposes a novel framework that combines path planning and tracking control. The framework utilizes an improved RRT* algorithm for path planning, considering the actual operating environment of the vehicle and whether the planned path can meet the requirements for vehicle tracking. Additionally, a PSO-LQR controller is designed for path-tracking testing to form a closed loop, and the feasibility of the proposed planning approach is verified.

The main contributions of this paper:

(1) For the driving intention of autonomous driving and the urban road scenario in which it is located, this study changes the sampling range of random points from the original complete state space sampling to adaptive Gaussian sampling around the nearest node, which reduces the invalid execution in the sampling process, decreases the number of algorithm iterations, and shortens the computation time.

(2) This study introduces an improved artificial potential field method into the RRT* algorithm. For the nearest nodes, the target point and random sampling point are set to have different attractive forces on them, and the concept of road boundary repulsion is introduced to make the obstacles and road boundary repel the nearest nodes. The combined direction of gravitational and repulsive forces is used as the extension direction of the new node to control the growth of the random tree toward the target point, reduce randomness, and speed up the path search. This study also uses path pruning based on the maximum steering angle constraint of the vehicle and the thrice B-spline algorithm to optimize the sampled generated paths to obtain paths that match the actual tracking of the vehicle.

(3) In this study, a PSO-LQR controller with feedforward control is designed to eliminate the external disturbances caused by the target path. Meanwhile, an objective function considering both tracking accuracy and vehicle stability is established to maintain vehicle stability and achieve higher tracking accuracy. The performance of the PSO-LQR controller is verified in this study by conducting simulation experiments on the LQR controller before and after optimization. Moreover, tracking experiments are conducted for planned paths at different speeds using the PSO-LQR controller to verify the feasibility of the improved RRT* algorithm for path planning and the effectiveness of the tracking control performance.

The organization of this paper is as follows: Section 2 introduces the improved RRT* algorithm, Section 3 introduces the PSO-LQR path tracking method, and Section 4 conducts experiments and analysis. Finally, in Section 5, this paper summarizes its contributions and proposes future research directions.

2. Path Planning Algorithm

2.1. Vehicle Kinematic Model

In vehicle path planning, it is necessary to consider the kinematic characteristics of the vehicle itself to ensure that the planned path meets the kinematic constraint conditions of the vehicle. This ensures that the planned path can meet the vehicle's steering motion characteristics, thereby ensuring the stability and safety of the driving process. The vehicle kinematic model established in this paper is shown in Figure 1.

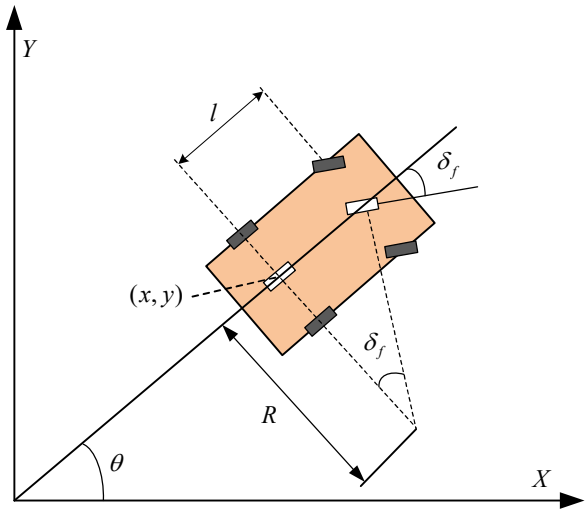

Figure 1. Vehicle kinematic model.

In the state space shown in Figure 1, the state of the vehicle can be represented by the variable $q = (x, y, \theta, \delta_f)$, where (x, y) represents the coordinate position of the rear axle of the vehicle, θ represents the angle between the longitudinal axis of the vehicle and the X-axis, and δ_f represents the steering angle of the front wheel. It is necessary to satisfy $|\delta_f| \leq \delta_{f\max}$ and l as the wheelbase, and R as the turning radius.

Under the constraint of the front wheel steering angle, the vehicle cannot reach any position in the state space. Assuming that the vehicle moves at a constant speed v parallel to the ground, the vehicle can be simplified as a two-degree-of-freedom bicycle model represented by Equation (1).

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = v \cdot \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \\ \frac{\tan \delta_f}{l} \end{bmatrix}. \tag{1}$$

The relationship between the front wheel steering angle, wheelbase, turning radius, and curvature can be obtained.

$$\rho_{\max} = \frac{1}{R_{\min}} = \frac{\tan \delta_{f\max}}{l} \tag{2}$$

where $R_{\min}$ represents the minimum turning radius, $\rho_{\max}$ represents the maximum curvature, and $\delta_{f\max}$ represents the maximum steering angle of the front wheels.

Based on the kinematic model and the analysis of the maximum front wheel turning angle, it is known that the curvature of the planned path should satisfy the constraint of the maximum front wheel turning angle. However, the RRT* algorithm does not consider the constraints of the vehicle kinematic model, and the planned path does not fit the kinematic characteristics of the vehicle. Therefore, the path planning of the vehicle can be constrained based on the maximum front wheel turning angle and path curvature to meet the actual driving path requirements of the autonomous vehicle.

2.2. RRT* Algorithm

Some researchers have proposed the RRT* algorithm. Before introducing the RRT* algorithm, this paper first presents the RRT algorithm proposed by other researchers as a basis. The RRT algorithm uses a random sampling method to expand the tree from the initial point to the target point. During each iteration, the RRT algorithm randomly selects a point q_{rand} in the state space. If the point falls within a non-obstacle interval, it will traverse the random expansion tree to find the nearest node $q_{nearest}$ to the random point. If the line between q_{rand} and $q_{nearest}$ does not collide with obstacles and the distance between them is less than the expansion step size, q_{rand} is added to the random tree as a new node q_{new}. If the distance between q_{rand} and $q_{nearest}$ is greater than the expansion step size, a point on the line between q_{rand} and $q_{nearest}$, at a distance of the expansion step size from $q_{nearest}$, is intercepted as q_{new} and added to the random expansion tree. The nearest node to q_{new} is then considered the parent node of q_{new}. Repeat this iteration process until a feasible path is found from the initial point to the target point.

In contrast, the RRT* algorithm introduces two improvements: reselecting parent nodes and rewiring. Reselecting the parent nodes means that after the RRT algorithm finds a new node, the nearest node to the new node among all nodes within a predefined radius r is selected as the parent node of the new node. Rewiring refers to reorganizing all nodes in the tree near the newly selected parent node after selecting a new parent node, with the principle of minimizing the cost of all nodes to the starting point. These two processes complement each other, with reselecting the parent nodes minimizing the cost of the newly generated node's path as much as possible, and rewiring reducing redundant paths in the random tree after generating a new node, thereby reducing path costs. These improvements significantly reduce the path cost of the RRT algorithm.

To improve the search efficiency of the RRT* algorithm, researchers have proposed a goal-biased RRT* algorithm, which, given a target-biased probability factor p_{target}, and a random probability value p is obtained between 0 and 1. When $p > p_{target}$, a random state named q_{rand} is obtained in the search space, otherwise q_{rand} is equal to q_{goal}. This algorithm maintains the characteristics of the original algorithm and accelerates the convergence speed to the target node.

2.3. Improved RRT Algorithm*

The flow chart of the improved RRT* algorithm in this paper is shown in Figure 2. Firstly, the algorithm is initialized, and relevant parameters are set. Then the variable sampling area method is used to restrict the generation of random sampling points. After generating the random sampling point q_{rand}, find its nearest node $q_{nearest}$, use the improved APF method to calculate the generation direction of the new node, and finally generate a new node q_{new} in that direction. If there is no collision with the obstacle, the new node will be added to the random tree and determine whether it reaches the target point. If there is a collision or the target point is not reached, the initialization is returned to continue the cycle until the target point is reached. Then the final curvature continuous path is obtained by the path optimization method. The new improvements of the algorithm are explained in detail below.

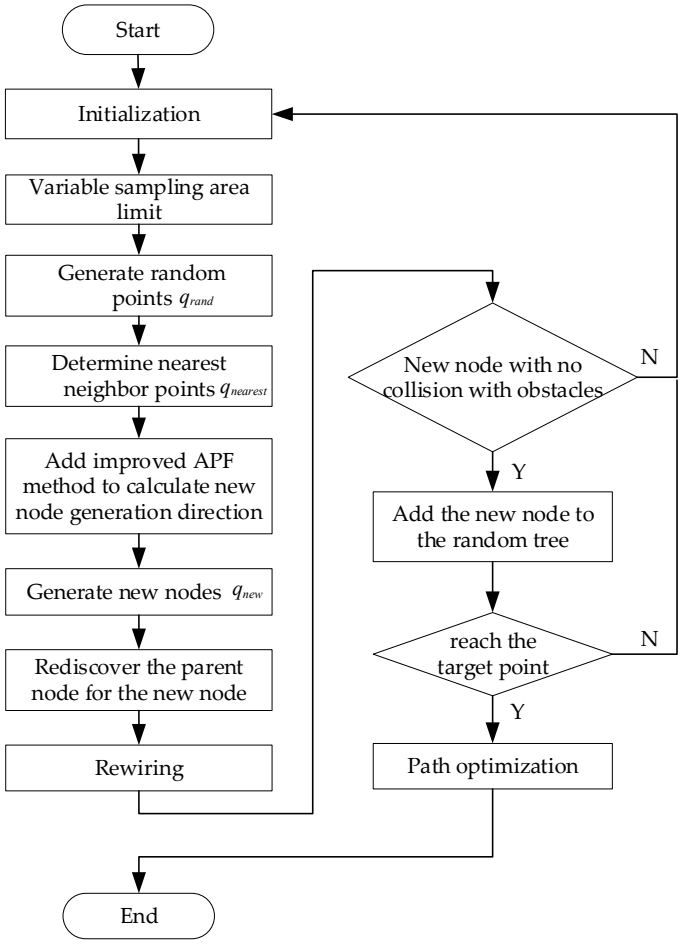

Figure 2. Improved RRT* algorithm flow chart.

2.3.1. Variable Sampling Area

In traditional RRT* and Goal-Bias RRT* algorithms, the random sampling of nodes is based on the entire state space. However, in structured road scenarios where autonomous vehicles operate, the vehicle's movement is constrained by traffic rules. Therefore, random sampling based on the entire state space will produce many invalid plans. The variable sampling area method adaptively samples within a range around the current nearest node using Gaussian sampling. This method dynamically adjusts the sampling interval according to the characteristics of the current state space, which helps improve the efficiency and accuracy of the algorithm.

As shown in the coordinate system in Figure 3, vehicles need to move in the direction of the lane markings while avoiding obstacles represented by the black box. Depending

on the driving intent of the vehicle, the RRT* algorithm should focus more on sampling in front of the vehicle during random sampling. Therefore, the Gaussian sampling will be more consistent with actual driving behavior. Using the state space information around the current nearest node, the sampling area of the new random node is limited to an adaptive fan-shaped range along the *X*-axis direction of the nearest node, as shown in Equation (3):

$$\Phi = \{x, y | x = x_o + r\cos\eta, y = y_o + r\sin\eta\} \tag{3}$$

$$\begin{cases} r = \sigma_r r_{rand} + r_o \\ \eta = \sigma_\eta \eta_{rand} + \eta_o \end{cases}, \quad \begin{cases} r_o = \varepsilon\rho(q, q_{obs}) \\ \eta_o = \frac{\pi}{2} - \arctan\left(\frac{\rho_x(q, q_{goal})}{\rho_y(q, q_{goal})}\right) \end{cases} \tag{4}$$

where Φ is the dynamic sampling area, $q(x_o, y_o)$ is the nearest node on the random tree during the sampling process, (x, y) is the coordinate of the new sampling point, (r, η) is the Gaussian parameter obtained from Equation (4), (r_o, η_o) is the mean of the Gaussian distribution relative to (x_o, y_o), (σ_r, σ_η) is the standard deviation of the Gaussian distribution in the radial and circumferential directions, (r_{rand}, η_{rand}) is a random variable that follows a standard Gaussian distribution. $\rho(q, q_{obs})$ represents the Euclidean distance between the nearest node and the center of the obstacle, ε is a constant, $\rho_x(q, q_{goal})$ and $\rho_y(q, q_{goal})$ represent the Euclidean distances between the nearest node and the target point in the X and Y directions, respectively.

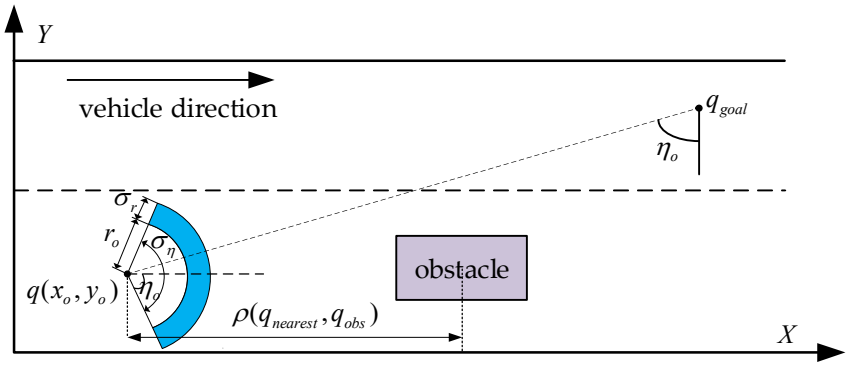

Figure 3. Schematic diagram of the variable sampling interval.

2.3.2. Improved APF-RRT* Algorithm

Based on the RRT* algorithm, adding an improved artificial potential field can improve the direction of new node generation and speed up the search efficiency of the random tree. The traditional APF algorithm is first introduced to pave the way for introducing the improved strategy. $q(x_o, y_o)$ is the nearest node in the random tree during the sampling process, which generates an attractive field function $U_{att}(q)$ according to Equation (5) and a repulsive field function $U_{rep}(q)$ according to Equation (6). The total potential field $U_{total}(q)$ is the sum of these two fields, as shown in Equation (7):

$$U_{att}(q) = \frac{1}{2} k_{ag} \rho^2(q, q_{goal}) \tag{5}$$

$$U_{rep}(q) = \begin{cases} \frac{1}{2} k_r \left(\frac{1}{\rho(q, q_{obs})} - \frac{1}{\rho_0}\right) & \rho(q, q_{obs}) \le \rho_0 \\ 0 & \rho(q, q_{obs}) \ge \rho_0 \end{cases} \tag{6}$$

$$U_{total}(q) = U_{rep}(q) + U_{att}(q) \tag{7}$$

where k_{ag} is the attraction gain factor of the target point to the nearest node is taken as 1.5 in this paper, k_r is the repulsion gain factor of the obstacle is taken as 2 in this paper, ρ_0 is the maximum distance at which the repulsion force acts, and $\rho(q, q_{goal})$ represents the Euclidean distance between the nearest node and the target point. The magnitudes of the

attraction and repulsion forces are the negative gradients of the attraction field function and the repulsion field function, respectively. Therefore, the attraction function is defined as shown in Equation (8). Similarly, the repulsion function is shown in Equation (9).

$$\mathrm{F_{att}}(q) = k_{ag}\rho(q, q_{goal}) \tag{8}$$

$$\mathrm{F_{rep}}(q) = \begin{cases} k_r\left(\frac{1}{\rho(q,q_{obs})} - \frac{1}{\rho_0}\right) & \rho(q, q_{obs}) \le \rho_0 \\ 0 & \rho(q, q_{obs}) \ge \rho_0 \end{cases} \tag{9}$$

To increase the exploratory nature of the algorithm and enable better exploration of the unknown regions, this paper proposes an improvement to the attraction field as follows. Firstly, the attraction field generated by the target point and the randomly sampled point are defined as $\mathrm{U_{attg}}(q)$ and $\mathrm{U_{attr}}(q)$, respectively, and their sum is the new attraction field function $\mathrm{U_{att}^{'}}(q)$, as shown in Equation (10). This produces a new attraction function $\mathrm{F_{att}^{'}}(q)$, as shown in Equation (11).

$$\mathrm{U_{att}^{'}}(q) = \mathrm{U_{attg}}(q) + \mathrm{U_{attr}}(q) = \frac{1}{2}(k_{ag}\rho^2(q, q_{goal}) + k_{ar}\rho^2(q, q_{rand})) \tag{10}$$

$$\mathrm{F_{att}^{'}}(q) = \mathrm{F_{attg}}(q) + \mathrm{F_{attr}}(q) = k_{ag}\rho(q, q_{goal}) + k_{ar}\rho(q, q_{rand}) \tag{11}$$

where k_{ar} is the gain coefficient for the attraction field produced by the sampled point towards the nearest node is taken as 1.5 in this paper, $\rho(q, q_{rand})$ is the Euclidean distance between the nearest node and the sampled point, $\mathrm{F_{attg}}(q)$ and $\mathrm{F_{attr}}(q)$ are the attractions generated by the target point, and the random sample towards the nearest node, respectively.

In the primary APF method, when the car reaches the target point, the attractive force becomes zero while the repulsive force remains non-zero, leading to the problem of the elusive target. Moreover, when the resultant direction of all the repulsive forces from obstacles is the same as the direction of the attractive force, and the car has not yet reached the target point, it is possible to get stuck in a local optimum where the repulsive and attractive forces are equal. To solve these problems, the traditional repulsive field of the local optimal improvement method proposed by other researchers is improved in this paper by introducing a modulation factor $\rho^n(q, q_{goal})$ in the repulsive field function, which produces a new repulsive field function $\mathrm{U_{rep}^{'}}(q)$ as shown in Equation (12), which ensures that the repulsive and attractive forces only reduce to zero simultaneously when the car reaches the target point, thus solving the problems of local optima and elusive targets.

$$\mathrm{U_{rep}^{'}}(q) = \begin{cases} \frac{1}{2}k_r\left(\frac{1}{\rho(q,q_{obs})} - \frac{1}{\rho_0}\right)^2\rho^n(q, q_{goal}) & \rho(q, q_{obs}) \le \rho_0 \\ 0 & \rho(q, q_{obs}) > \rho_0 \end{cases} \tag{12}$$

$$\begin{aligned} \mathrm{F_{rep}^{'}}(q) = {} & k_r\left(\frac{1}{\rho(q,q_{obs})} - \frac{1}{\rho_0}\right)\frac{\rho^n(q,q_{goal})}{\rho^2(q,q_{obs})}\nabla\rho(q, q_{obs}) \\ & -\frac{n}{2}k_r\left(\frac{1}{\rho(q,q_{obs})} - \frac{1}{\rho_0}\right)^2\rho^{n-1}(q, q_{goal})\nabla\rho(q, q_{goal}) \end{aligned} \tag{13}$$

where $\mathrm{F_{rep}^{'}}(q)$ is the new obstacle repulsion function, $\nabla\rho(q, q_{obs})$ and $\nabla\rho(q, q_{goal})$ are the unit vectors of the nearest node and obstacle and the direction of the target point, respectively, n is an arbitrary constant, and $n = 2$ is taken here after many tests.

However, vehicles are different from robots when they travel on roads. They are not only constrained by obstacles but also by the road itself. According to the driving experience, the boundary area of the road is the most dangerous, followed by the centerline area, and the lane centerline area is the least dangerous. The force exerted on the vehicle by the road boundary is shown in Figure 4.

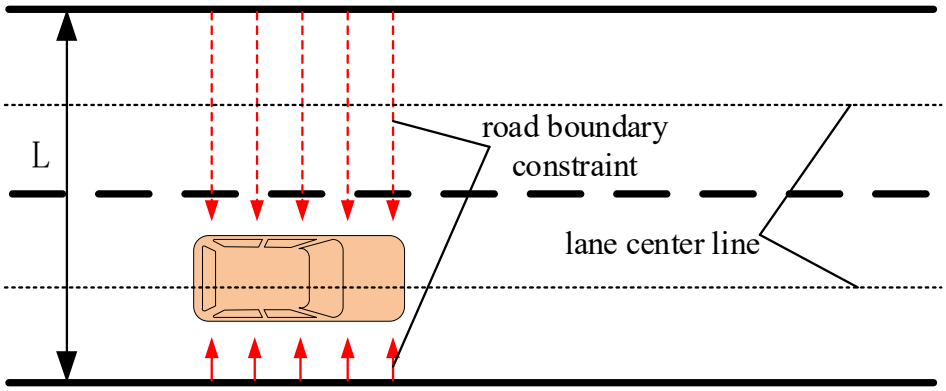

Figure 4. Schematic diagram of road boundary constraints.

According to the distribution of road danger levels mentioned above, the potential field function of the road boundary needs to be considered in segments. When the vehicle is in the area between the centerlines of two lanes, it is in a relatively safe zone. The influence of the potential field changes slowly with position, so a function with a relatively gentle change trend is used. In other areas where the danger level is higher, the influence of the potential field changes quickly with position, so a function with a more extensive change trend is used. Taking a dual carriageway as an example, considering the above factors, the potential field function of the road boundary established is shown as Equation (14), and the boundary potential field generated by the road boundary produces the road boundary repulsive force shown as Equation (15), pushing the vehicle away from illegal areas on the road boundary and ensuring the safety of the path.

$$\mathrm{U}_{\mathrm{road}}(q) = \begin{cases} k_{\mathrm{road1}}\left(\mathrm{e}^{-|y_o - y_l|} - 1\right), y_o \le \frac{L}{4} \\ k_{\mathrm{road2}} \sin\left(\frac{2(y_o - y_l)}{L}\right), \frac{L}{4} < y_o < \frac{3L}{4} \\ k_{\mathrm{road3}}\left(\mathrm{e}^{|y_o - y_r|} - 1\right), y_o \ge \frac{3L}{4} \end{cases} \tag{14}$$

$$\mathrm{F}_{\mathrm{road}}(q) = \begin{cases} k_{\mathrm{road1}}\mathrm{e}^{-|y_o - y_l|}, y_o \le \frac{L}{4} \\ k_{\mathrm{road2}} \cos\left(\frac{2(y_o - y_l)}{L}\right), \frac{L}{4} < y_o < \frac{3L}{4} \\ k_{\mathrm{road1}}\mathrm{e}^{|y_o - y_r|}, y_o \ge \frac{3L}{4} \end{cases} \tag{15}$$

where k_{road1} and k_{road2} are the potential energy gain coefficients of the road boundary; y_l and y_r is the horizontal position of the centerline of the inner and outer lanes; L is the width of the road.

Based on the Goal Biased RRT*, an improved APF method is combined, where both the target point and random sampling point exert attraction on the nearest node, and obstacles exert a repulsive force on it. At the same time, considering the road environment where the autonomous driving vehicle is located, the road boundary repulsive force is introduced. The force diagram of the nearest node $q(x_o, y_o)$ is shown in Figure 5, and the direction of the resultant force F'_{total} is drawn using the parallelogram rule, as shown in Equation (16). Its direction serves as the extension direction for new nodes, controlling the growth of the random tree towards the target point to reduce randomness and accelerate path search speed.

$$\mathrm{F}'_{\mathrm{total}}(q) = \mathrm{F}'_{\mathrm{att}}(q) + \mathrm{F}'_{\mathrm{rep}}(q) + \mathrm{F}_{\mathrm{road}}(q) \tag{16}$$

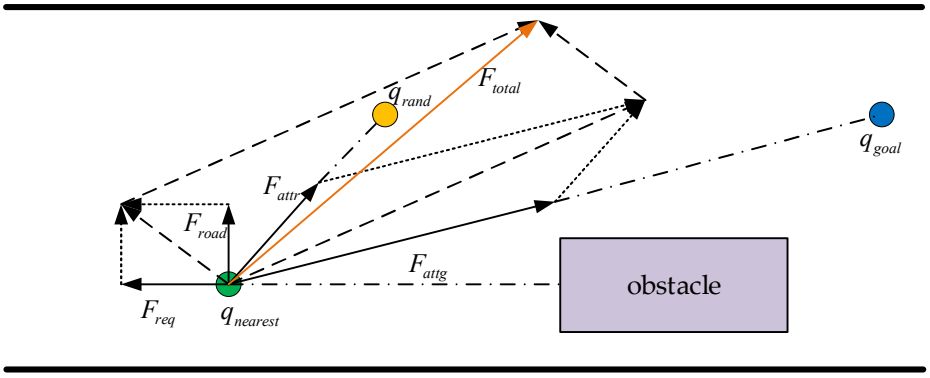

Figure 5. Force analysis diagram of the nearest node.

Then the resultant force F'_{total} is decomposed into F_x and F_y, corresponding to the components of the x-axis and y-axis, respectively, then in the case of the expansion step λ, the position of the new node is $q_{new}(x_{new}, y_{new})$, as shown in Equation (17):

$$\begin{cases} x_{new} = x_o + \lambda \frac{F_x}{F_x + F_y} \\ y_{new} = y_o + \lambda \frac{F_y}{F_x + F_y} \end{cases} \tag{17}$$

The path planning for autonomous vehicles in road scenarios was achieved by fusing the RRT* algorithm and the improved APF method. The algorithm introduces the concepts of attraction field and repulsion field to control the growth direction of the RRT* algorithm towards the target point as much as possible in the tree expansion direction. After defining the attraction field and repulsion field functions, their resultant force direction can be used as the expansion direction of new nodes for node expansion. In this way, the tree growth towards the target point can be controlled during the path planning process to reduce randomness and improve path search speed.

2.3.3. Path Optimization Algorithms

Due to the random sampling nature of the RRT* algorithm during path searching, if the path is directly generated from the goal point back to the starting point, the resulting path will inevitably be more tortuous. It may not meet the maximum curvature constraints required for vehicle steering, as well as resulting in redundant nodes, leading to an unnecessary increase in path cost. Therefore, this paper proposes a path-pruning method based on the maximum steering angle constraint of the vehicle. As shown in Figure 6, starting from the goal node q_{goal}, it is connected to the starting node q_{start}, and the intersection between the line segment and obstacles is checked. If there is no intersection, the nodes between the two path nodes are pruned, and the line connecting them is considered as the optimal path. If there is an intersection, node q_2 is connected to the goal node q_{goal}, and the intersection check is repeated while also checking whether the angle formed by the two adjacent path segments generated by nodes q_{start}, q_2, and q_{goal} satisfies the maximum steering angle constraint of the vehicle. If it does not satisfy one of the judgment criteria, it will continue to connect to q_3 and q_{goal}, and repeat the above judgment. If both judgment criteria are met, it will be considered as the next path point of q_{start}, and the above operation is repeated until the final path that does not intersect with obstacles and meets the maximum steering angle constraint of the vehicle is obtained. This ensures that the path curvature does not exceed the maximum curvature when using B-spline curves for smooth paths and outputs the path. The path after pruning optimization is shown as a solid line in Figure 6.

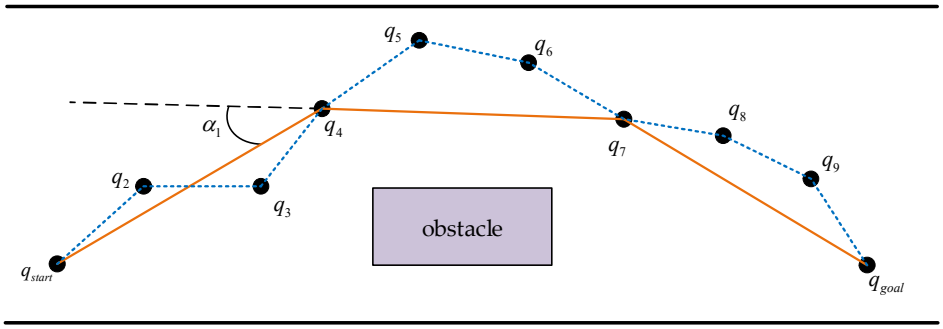

Figure 6. Path clipping based on vehicle maximum turning angle constraint.

After the above path clipping, the path still has the problem of discontinuous curvature. Therefore, a cubic B-spline curve is used to smooth the path and make the curvature of the planned path continuous. If there are $n+1$ control points in total, the K-th order B-spline curve is defined as:

$$\mathrm{P}(t) = [p_0, p_1, \ldots, p_n] \begin{bmatrix} \mathrm{B}_{0,K}(t) \\ \mathrm{B}_{1,K}(t) \\ \cdots \\ \mathrm{B}_{n,K}(t) \end{bmatrix} = \sum_{i=0}^{n} P_i B_{i,K}(t) \tag{18}$$

where $\mathrm{P}(t)$ are control points; $\mathrm{B}_{i,K}(t)$ is the basis function of cubic B-splines, recursively obtained according to the DeBoor-Cox formula:

$$\begin{cases} \mathrm{B}_{i,0}(t) \begin{cases} 1 \;\; t_i \le t \le t_{i+1} & K = 1 \\ 0 \;\; Otherwise \end{cases} \\ \mathrm{B}_{i,3}(t) = \frac{(t-t_i)B_{i,2}(t)}{t_{i+3}-t_i} + \frac{(t_{i+4}-t)B_{i+1,2}(t)}{t_{i+4}-t_{i+1}} \qquad K \ge 2 \end{cases} \tag{19}$$

In this study, three uniform B-splines were selected to smooth the planned path, and the repeatability of the nodes at both ends was set to 3. Then the basis function can be expressed as:

$$\begin{cases} \mathrm{B}_{0,3}(t) = \frac{1}{6}(1-t)^3 \\ \mathrm{B}_{1,3}(t) = \frac{1}{6}(3t^3 - 6t^2 + 4) \\ \mathrm{B}_{2,3}(t) = \frac{1}{6}(-3t^3 + 3t^2 + +3t + 1) \\ \mathrm{B}_{3,3}(t) = \frac{1}{6}t^3 \end{cases} \tag{20}$$

Here, the vector interval of parameter nodes is set to [0, 1]. By substituting Equation (20) into Equation (18), the expression of the third-order quasi-uniform B-spline curve can be obtained:

$$\mathrm{P(t)} = \mathrm{P}_0 B_{0,3}(t) + \mathrm{P}_1 B_{1,3}(t) + \mathrm{P}_2 B_{2,3}(t) + \mathrm{P}_3 B_{3,3}(t), \; t \in [0,1]. \tag{21}$$

3. Path Tracking Controller

3.1. Establishment of Path Tracking Model

3.1.1. Vehicle Dynamic Model

The path tracking control of autonomous driving vehicles mainly studies the lateral dynamic characteristics of the vehicle, involving the vehicle's lateral and yawing movements. To simplify the calculations, it is assumed that the vehicle's coaxial wheels have the same lateral stiffness and turning angle, and thus, the coaxial wheels can be merged. This simplifies the vehicle's dynamic model to a two-degree-of-freedom lateral dynamic model [34,35], as shown in Figure 7.

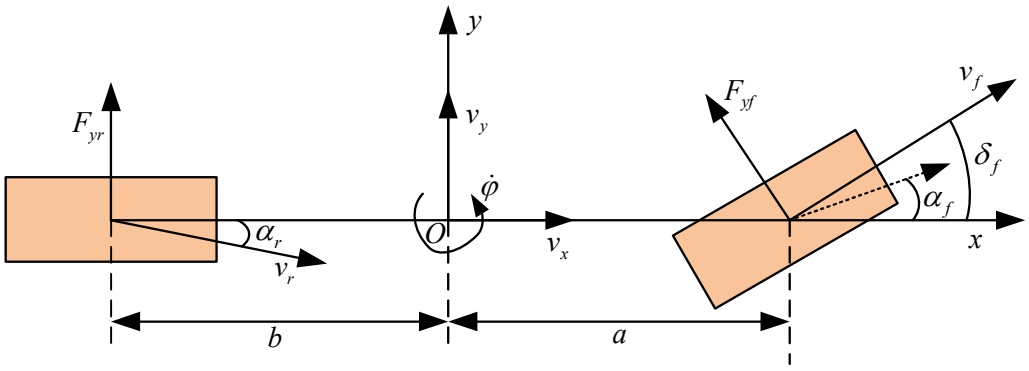

Figure 7. Vehicle dynamics model.

Assuming the vehicle is driving at a constant speed and under the condition where the front wheel steering angle δ_f and the front and rear wheel slip angles α_f and α_r are small, lateral load transfer can be ignored. According to the magic formula tire model [36], the vehicle's dynamic Equation can be expressed as:

$$\begin{cases} ma_y = F_{yf} + F_{yr} = C_f\alpha_f + C_r\alpha_r = C_f\left(\frac{\dot{\varphi}a+v_y}{v_x} - \delta_f\right) + C_r\left(\frac{v_y-\dot{\varphi}b}{v_x}\right) \\ I_z\ddot{\varphi} = aF_{yf} - bF_{yr} = aC_f\alpha_f - bC_r\alpha_r = aC_f\left(\frac{\dot{\varphi}a+v_y}{v_x} - \delta_f\right) - bC_r\left(\frac{v_y-\dot{\varphi}b}{v_x}\right) \end{cases} \tag{22}$$

where m is the mass of the entire vehicle; F_{yf} and F_{yr} are the lateral forces of the front and rear wheels, respectively; I_z is the rotational inertia of the vehicle around the Z-axis; $\dot{\varphi}$ is the vehicle's yaw rate; a and b are the distances from the center of mass of the vehicle to the front and rear axles, respectively, C_f and C_r are the lateral stiffness of the front and rear wheels, respectively; v_x is the longitudinal speed of the vehicle; v_y is the lateral velocity of the vehicle.

Define $\dot{y} = v_y$ to get the vehicle dynamics model:

$$\begin{pmatrix} \ddot{y} \\ \ddot{\varphi} \end{pmatrix} = \begin{pmatrix} \frac{C_f+C_r}{mv_x} & \frac{aC_f-bC_r}{mv_x} - v_x \\ \frac{aC_f-bC_r}{I_zv_x} & \frac{a^2C_f+b^2C_r}{I_zv_x} \end{pmatrix} \begin{pmatrix} \dot{y} \\ \dot{\varphi} \end{pmatrix} + \begin{pmatrix} -\frac{C_f}{m} \\ -\frac{aC_f}{I_z} \end{pmatrix} \delta_f \tag{23}$$

3.1.2. Path Tracking Error Model

When an autonomous vehicle is tracking a reference path, it will mainly experience lateral error and heading angle error. As shown in Figure 8, the shortest distance between the vehicle's center of mass and the reference path projection point is defined as the lateral error e_d. The difference between the vehicle's actual heading angle θ and the reference heading angle θ_r is defined as the heading angle error e_θ. For convenience of calculation, it is assumed that the vehicle's center of mass has a lateral deviation angle $\beta = 0$, and the vehicle's heading angle error is $e_\varphi = \varphi - \theta_r$. In actual control, the controller is required to eliminate these two errors in real time to enable the vehicle to track the planned path in real time. Based on the lateral error and heading angle error, the first-order derivatives of lateral error $\dot{e}_d$ and heading angle error $\dot{e}_\varphi$ can be calculated.

$$\dot{e}_d = v_x e_\varphi + v_y \tag{24}$$

$$\dot{e}_\varphi = \dot{\varphi} - \dot{\varphi}_r \tag{25}$$

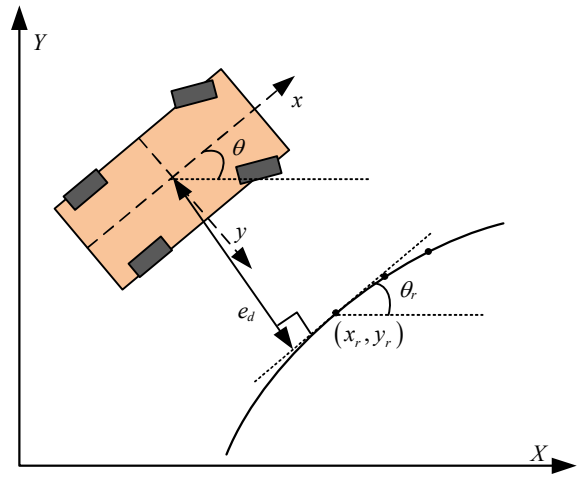

Figure 8. Path tracking model.

Combining Equation (23), the second-order derivative of the lateral error $\ddot{e}_d$ and the second-order derivative of the heading angle error $\ddot{e}_\varphi$ can be obtained.

$$\begin{aligned}\ddot{e}_d &= \left(\frac{C_f+C_r}{mv_x}\right)\dot{e}_d + \left(-\frac{C_f+C_r}{m}\right)e_\varphi + \left(\frac{aC_f-bC_r}{mv_x}\right)\dot{e}_\varphi \\ &+\left(\frac{aC_f-bC_r}{mv_x} - v_x\right)\dot{\theta}_r + \left(-\frac{C_f}{m}\right)\delta_f\end{aligned} \tag{26}$$

$$\begin{aligned}\ddot{e}_\varphi &= \left(\frac{aC_f-bC_r}{I_z v_x}\right)\dot{e}_d + \left(-\frac{aC_f-bC_r}{I_z}\right)e_\varphi + \left(\frac{a^2C_f+b^2C_r}{I_z v_x}\right)\dot{e}_\varphi \\ &+\left(\frac{a^2C_f+b^2C_r}{I_z v_x}\right)\dot{\theta}_r + \left(-\frac{aC_f}{I_z}\right)\delta_f\end{aligned} \tag{27}$$

Further transformations on the above formula result in the following state-space Equations for the lateral and heading errors in the steering process of autonomous driving vehicles:

$$\begin{aligned}\begin{pmatrix}\dot{e}_d\\ \ddot{e}_d\\ \dot{e}_\varphi\\ \ddot{e}_\varphi\end{pmatrix} &= \begin{pmatrix}0 & 1 & 0 & 0\\ 0 & \frac{C_f+C_r}{mv_x} & -\frac{C_f+C_r}{m} & \frac{aC_f-bC_r}{mv_x}\\ 0 & 0 & 0 & 1\\ 0 & \frac{aC_f-bC_r}{I_z v_x} & -\frac{aC_f-bC_r}{I_z} & \frac{a^2C_f+b^2C_r}{I_z v_x}\end{pmatrix}\begin{pmatrix}e_d\\ \dot{e}_d\\ e_p\\ \dot{e}_\varphi\end{pmatrix} \\ &+\begin{pmatrix}0\\ -\frac{C_f}{m}\\ 0\\ -\frac{aC_f}{I_z}\end{pmatrix}\delta_f + \begin{pmatrix}0\\ \frac{aC_f-bC_r}{mv_x} - v_x\\ 0\\ \frac{a^2C_f+b^2C_r}{I_z v_x}\end{pmatrix}\dot{\theta}_r\end{aligned} \tag{28}$$

Then the state space Equation about the tracking error of vehicle dynamics can be obtained:

$$\dot{\mathbf{X}} = \mathbf{AX} + \mathbf{BU} + \mathbf{C}\dot{\theta}_r. \tag{29}$$

In practical applications, we frequently work with discrete data, and the path generated by the path planning algorithm described earlier consists of a series of reference points rather than a continuous path. Hence, we opt for discrete LQR control. To design a discrete LQR controller, we discretize Equation (31) by ignoring the impact of the $C\dot{\theta}_r$ term, resulting in the state space Equation for discrete vehicle tracking errors.

$$\mathbf{X}_{k+1} = \overline{\mathbf{A}}\mathbf{X}_k + \overline{\mathbf{B}}\mathbf{U}_k \tag{30}$$

where: $\overline{\mathbf{A}} = \left(\mathbf{I} - \frac{\mathbf{A}dt}{2}\right)^{-1}\left(\mathbf{I} + \frac{\mathbf{A}dt}{2}\right)$; $\overline{\mathbf{B}} = \mathbf{B}dt$.

3.2. Design of LQR Path Tracking Controller Based on Particle Swarm Optimization

The overall structure of the path-tracking controller is shown in Figure 9. Firstly, based on the path tracking error model, an LQR controller is designed as the central part of the path-tracking controller so that the vehicle can travel along the reference path; on this basis, the particle swarm optimization algorithm is used to optimize the weight matrices Q and R of the LQR controller to improve its performance, and a feedforward control method is used as the steering angle compensation to eliminate the steady-state error of the system and further optimize the path tracking effect.

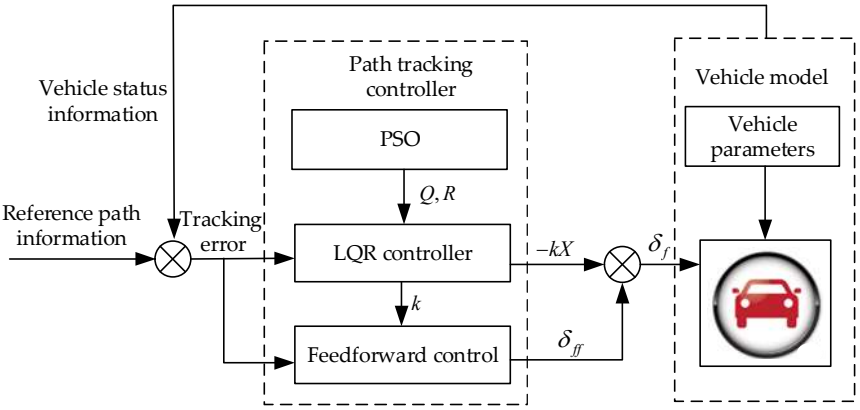

Figure 9. Path tracking controller framework.

3.2.1. LQR Controller Design

LQR is an optimization problem that minimizes the performance objective function under given constraints and obtains the optimal feedback control by solving the corresponding algebraic Riccati Equation [37]. Its advantage in autonomous driving path tracking control is that when the system state deviates from the steady state due to obstacles or unexpected events, it can ensure that the vehicle tracking control system approaches the ideal path without causing too much computational workload. The LQR control problem of the vehicle lateral path tracking controller can be expressed as:

$$\min \mathrm{J} = \sum_{k=0}^{\infty} \left(\mathbf{X}_k^T \mathbf{Q} \mathbf{X}_k + \mathbf{U}_k^T \mathbf{R} \mathbf{U}_k \right) \tag{31}$$

$$\mathbf{Q} = \mathrm{diag}[q_1, q_2, q_3, q_4] \tag{32}$$

$$\mathbf{R} = [q_5] \tag{33}$$

where **X** denotes the system's state variable, while **U** represents its control variable. The weighted matrix **Q** indicates the level of emphasis placed on the corresponding control target in terms of state error representation. The weight matrix of the control variable is denoted by **R**. The weight coefficients for lateral error, the lateral error rate of change, heading error, and the heading error rate of change are, respectively, represented by q_1, q_2, q_3, q_4. The weight coefficient of the front wheel steering angle is given by q_5. These weight coefficients reflect the relative importance of the variables. Larger values correspond to faster convergence of the related state variable towards the target value.

Then, the Lagrange multiplier method is used to construct the constrained optimization problem for solving extreme values. The cost function under constraint conditions is:

$$\mathrm{J} = \sum_{k=0}^{n-1} \left[\mathbf{X}_k^T \mathbf{Q} \mathbf{X}_k + \mathbf{U}_k^T \mathbf{R} \mathbf{U}_k + \boldsymbol{\lambda}_{k+1}^T \left(\overline{\mathbf{A}} \mathbf{X}_k + \overline{\mathbf{B}} \mathbf{U}_\mathbf{k} \right) - \boldsymbol{\lambda}_{k+1}^T \mathbf{X}_{k+1} \right] + \mathbf{X}_n^T \mathbf{Q} \mathbf{X}_n. \tag{34}$$

The constructed Hamiltonian function $\mathbf{H}_k = \mathbf{X}_k^T \mathbf{Q} \mathbf{X}_k + \mathbf{U}_k^T \mathbf{R} \mathbf{U}_k + \boldsymbol{\lambda}_{k+1}^T \left(\overline{\mathbf{A}} \mathbf{X}_k + \overline{\mathbf{B}} \mathbf{U}_k \right)$, Equation (34) can be simplified as:

$$\mathrm{J} = \sum_{k=0}^{n-1} (\mathbf{H}_k - \boldsymbol{\lambda}_k^T \mathbf{X}_k) + \mathbf{X}_n^T \mathbf{Q} \mathbf{X}_n - \boldsymbol{\lambda}_n^T \mathbf{X}_n. \tag{35}$$

The control quantity of LQR obtained by solving the extreme value of Equation (35) is:

$$\mathbf{U}_k = -(\mathbf{R} + \overline{\mathbf{B}}^T \mathbf{P} \overline{\mathbf{B}})^{-1} \overline{\mathbf{B}}^T \mathbf{P} \overline{\mathbf{A}} \mathbf{X}_k \tag{36}$$

where $\mathbf{P}$ is obtained by calculating Riccati Equation $\mathbf{P}_{k-1} = \mathbf{Q} + \overline{\mathbf{A}}^T \mathbf{P}_k (\mathbf{I} + \overline{\mathbf{B}} \mathbf{R}^{-1} \overline{\mathbf{B}}^T \mathbf{P}_k)^{-1} \overline{\mathbf{A}}$.

Equation (36) can be simplified as:

$$\mathbf{U}_k = -\mathbf{k} \mathbf{X}_k \tag{37}$$

where $\mathbf{k} = [k_1, k_2, k_3, k_4]$ is the result of the LQR controller and also the feedback coefficient of the front wheel angle.

3.2.2. Feedforward Control

By bringing Equation (37) into Equation (29), it can be obtained that:

$$\dot{\mathbf{X}} = (\mathbf{A} - \mathbf{B}\mathbf{k}) \mathbf{X} + \mathbf{C} \dot{\theta}_r \tag{38}$$

At this point, no matter what value $\mathbf{k}$ takes, $\dot{\mathbf{X}}$ cannot be zero. If only LQR feedback control is used, there will be a constant steady-state error; Therefore, we remove the influence of the $\mathbf{C}\dot{\theta}_r$ term by introducing a feedforward control variable δ_{ff}, and the system control variable after adding feedforward control is:

$$\mathbf{U} = -\mathbf{k} \mathbf{X} + \delta_{ff} \tag{39}$$

When $\dot{\mathbf{X}} = 0$, the expression for the state variable of the system without steady-state error is:

$$\mathbf{X} = -(\mathbf{A} - \mathbf{B}\mathbf{k})^{-1} \mathbf{A} \left(\mathbf{B} \delta_{ff} + \mathbf{C} \dot{\theta}_r \right) \tag{40}$$

From Equation (40), it can be concluded that in order to achieve the optimal control effect, it is necessary to calculate the appropriate δ_{ff}, so that the steady-state error of the system is 0. Based on Equation (28), the steady-state error Equation of the system is calculated as follows:

$$\begin{bmatrix} e_d \\ \dot{e}_d \\ e_\varphi \\ \dot{e}_\varphi \end{bmatrix} = \begin{bmatrix} \frac{1}{k_1} \left\{ \delta_{ff} - \frac{\dot{\theta}_r}{v_x} \left[a + b - b k_3 - \frac{m v_x^2}{a+b} \left(\frac{b}{C_f} + \frac{a}{C_r} k_3 - \frac{a}{C_r} \right) \right] \right. \\ 0 \\ -\frac{\dot{\theta}_r}{v_x} \left(b + \frac{a}{a+b} \frac{m v_x^2}{C_r} \right) \\ 0 \end{bmatrix} \tag{41}$$

According to Equation (41), it can be seen that:

$$e_\varphi = -\frac{\dot{\theta}_r}{v_x} \left(b + \frac{a}{a+b} \frac{m v_x^2}{C_r} \right) = -\beta. \tag{42}$$

In the previous text, for the convenience of calculation, assuming that the lateral deviation angle of the vehicle's center of mass is $\beta = 0$, the heading angle error of the vehicle is $e_\varphi = \varphi - \theta_r$ but $e_\theta = \varphi + \beta - \theta_r = 0$, so the heading error can be directly eliminated. If you want the lateral error $e_d = 0$, the feedforward control quantity δ_{ff} is:

$$\delta_{ff} = \rho \left[a + b - b k_3 - \frac{m v_x^2}{a+b} \left(\frac{b}{C_f} + \frac{a}{C_r} k_3 - \frac{a}{C_r} \right) \right] \tag{43}$$

where $\rho = \frac{\dot{\theta}_r}{v_x}$ is the path curvature.

3.3. Optimization of LQR Controller Based on Particle Swarm Optimization Algorithm

PSO (particle swarm optimization) is widely used in parameter optimization problems because of its simplicity and fast convergence [38–40]. In the PSO algorithm, the optimization problem is transformed into a search problem for the optimal solution, which is easy to implement and has global search ability. The PSO algorithm finds the optimal LQR weight matrix by conducting a global search in the search space, further improving the control system's performance. Figure 10 shows the algorithm flowchart of applying PSO to optimize the LQR weight matrix.

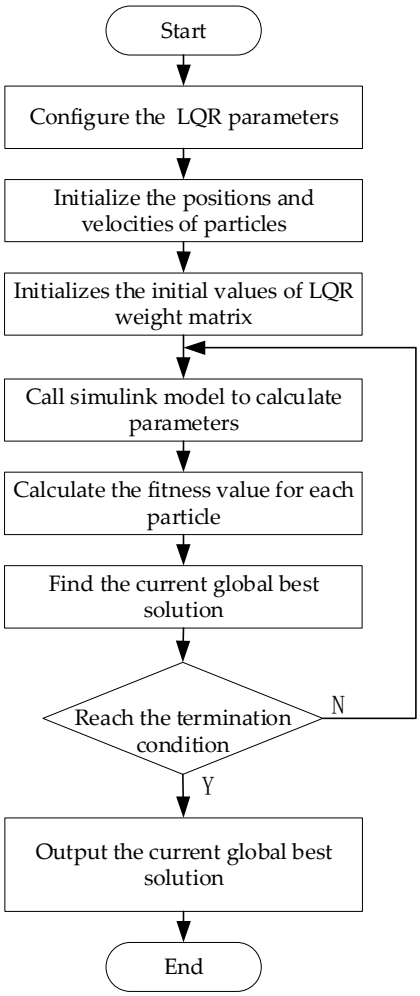

Figure 10. Flow chart of particle swarm optimization LQR algorithm.

Firstly, the parameters required for the LQR algorithm need to be configured, followed by the initialization of the position and velocity of particles, including the initial value of the LQR weight matrix, which should be randomly generated within a feasible range. Then, the control quantity of LQR is calculated using the generated weight matrix Q and R, and the Simulink model is called to calculate relevant parameters. The fitness function of each particle is obtained by calculating the fitness function based on the parameters obtained from running the model. The fitness function is a vital component of the particle swarm algorithm. Taking into account the characteristics of autonomous driving vehicles and driving scenarios and considering the relationship between the frequency domain and time domain, this study designed a fitness function F based on the sum of the integral time multiplied by the absolute error (ITAE) of lateral error e_d, heading error e_φ, roll angle

velocity $\dot{\varphi}$, and lateral acceleration a_y, to simultaneously consider path tracking accuracy and vehicle stability.

$$\min F = \int_0^T t|e_d(t)|\mathrm{d}t + \int_0^T t|e_\varphi(t)|\mathrm{d}t + \int_0^T t|\dot{\varphi}(t)|\mathrm{d}t + \int_0^T t|a_y(t)|\mathrm{d}t \tag{44}$$

Then, for each particle, find its best solution: the position and weight matrix with the smallest fitness value. Find the current global best solution, which is the position and weight matrix with the smallest fitness value among all particles, and update the velocity and position of each particle based on Equation (45):

$$\begin{cases} V_i(t+1) = \omega V(t)_i + c_1 \mathrm{r}_1 (p_{\text{best\_i}} - X_i(t)) + c_2 \mathrm{r}_2 (g_{\text{best}} - X_i(t)) \\ X_i(t+1) = X_i(t) + V_i(t+1) \end{cases} \tag{45}$$

where $V_i(t)$ and $X_i(t)$ are the velocity and position of the i-th particle in the t-th iteration, $p_{\text{best\_i}}$ is the personal optimal solution of the i-th particle, g_{best} is the current global optimal solution, ω is the inertia factor, c_1 and c_2 are learning factors, r_1 and r_2 are random numbers between [0, 1].

Loop until the specified termination criterion is reached, and output the current global optimal solution and the optimal value of the LQR Weight Matrix, as shown in Table 1.

Table 1. Optimal values of LQR weight matrix.

Longitudinal Velocity (m/s)	Q	R
10	diag [300 0.01 0.01 4.49]	6.02
15	diag [270.71 0.01 0.01 119.35]	4.91
20	diag [1.23 0.01 99.47 62.88]	1.39

4. Simulation Analysis

4.1. Simulation Analysis of Path Planning

Simulations were performed using Matlab 2020(b) (sourced from MathWorks, a company located in Natick, MA, United States) on a Windows 10 computer with an Intel Core i7-12700F CPU and 32 G RAM. This paper compares the simulation results of four algorithms, RRT*, Goal-Biased RRT*, P-RRT*, and Improved-RRT*, for three Map scenarios. The Improved-RRT* algorithm exhibits fast convergence, short planning time, and high security of the planned path. The simulated Maps are 2D environments, with Map 1 and 3 having dimensions of 7 m × 100 m and Map 2 having dimensions of 7 m × 120 m, representing three different scenarios. As shown in Figure 11, in the simulation, the vehicle starts at the starting point in rose red and ends at the endpoint in blue. The obstacle vehicles and road boundaries are represented in dark green, the blue line indicates the sampling process, and the red line indicates the final generated path. Each method was tried 30 times in each Map situation to remove the influence of randomness, and the average result was calculated. The results include the average path length, average path search time, average number of iterations, and average memory consumption, which are recorded and presented in Tables 2–4, respectively.

Table 2. Simulation data under Map 1.

Algorithm	RRT*	Goal-Biased-RRT*	P-RRT*	Improved-RRT*
Average path length (m)	102.4621	100.9605	100.8112	100.7826
Average running time (s)	14.7644	7.0604	1.8454	1.3079
Average iterations	480.8	310.4	219.3	146.4
Average memory consumption (MB)	44.3	33.1	26.6	22.4

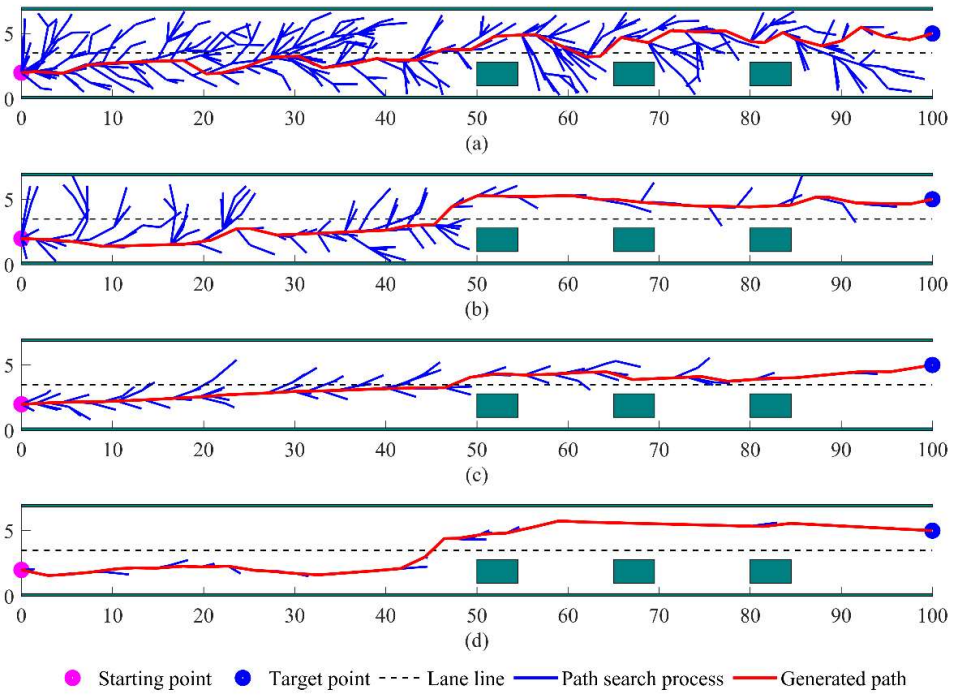

Figure 11. Comparison of path search results in the Map 1 scenario. (**a**) RRT* path search results; (**b**) Goal Biased RRT* path search results; (**c**) P-RRT* path search results; (**d**) Improved RRT* path search results.

Table 3. Simulation data under Map 2.

Algorithm	RRT*	Goal-Biased-RRT*	P-RRT*	Improved-RRT*
Average path length (m)	122.3938	121.5995	121.2425	121.2192
Average running time (s)	16.8818	11.9701	2.8668	1.0641
Average iterations	634.4	401.8	334.1	218.2
Average memory consumption (MB)	49.7	35.3	32.1	26.2

Table 4. Simulation data under Map 3.

Algorithm	RRT*	Goal-Biased-RRT*	P-RRT*	Improved-RRT*
Average path length (m)	102.8306	101.2469	100.7349	100.6376
Average running time (s)	12.4730	8.0140	2.1727	1.4296
Average iterations	518.5	349.7	282.5	177.1
Average memory consumption (MB)	47.4	34.1	29.3	24.5

Map 1 depicts a two-lane changing scenario where the vehicle must avoid three obstacles on the right lane and switch to the left lane. As shown in Figure 11a, the original RRT* algorithm employs full-state space random sampling, generating unnecessary sampling processes and resulting in a longer and more tortuous path. Figure 11b displays the simulation results of the Goal-Biased-RRT* algorithm, which significantly reduces the sampling process and improves the search efficiency by incorporating a target bias strategy. Figure 11c shows the simulation results of the P-RRT* algorithm, which enhances search speed and simplifies redundant nodes by adding attractive field constraints. However, it still generates redundant nodes around obstacles due to the lack of obstacle exclusion effect. In contrast, Figure 11d illustrates the Improved-RRT* algorithm, which restricts the variable sampling area and introduces an improved APF algorithm to enhance search efficiency. Although the path length of the Improved-RRT* algorithm does not show a

significant advantage compared to the previous algorithms, the distance between the path and obstacles has a certain safety margin, thus improving the path safety.

By analyzing the data in Table 2, the path length of the proposed algorithm is 1.67% shorter than that of the original RRT* algorithm. Compared with the P-RRT* algorithm, the average running time and iteration times are reduced by 29.13% and 33.24%, respectively, and reduces memory consumption by about 15.8%. The results demonstrate that the proposed algorithm can significantly reduce the number of random points, improve the convergence speed, and ensure algorithm stability.

Map 2 depicts a passing scenario in a double-lane road, where the vehicle needs to pass a single obstacle on the right lane and also avoid two obstacles on the left lane, requiring two lane changes to complete the passing maneuver. The simulation results are shown in Figure 12. It can be seen that for such a complex environment, the advantages of the proposed algorithm are more evident. Although the reduction in path length compared to other methods is relatively small, the paths generated by other methods are not good and too close to the obstacle cars, with a low safety margin. As shown in Figure 12d, the improved algorithm proposed in this paper plans a smoother path in this scenario, with a greater distance from the obstacle cars and a higher level of safety. It also reduces the large number of redundant nodes, shortening the search time. Analyzing the data in Table 3, the new algorithm reduces the average path length by 0.96% compared to the original RRT* algorithm. Compared with the P-RRT* algorithm, the average running time is reduced by 39.67%, the average number of iterations is reduced by 34.69%, and the algorithm memory consumption is reduced by 18.4%. Therefore, the proposed algorithm can significantly reduce unnecessary sampling processes and iterations, demonstrating its stability.

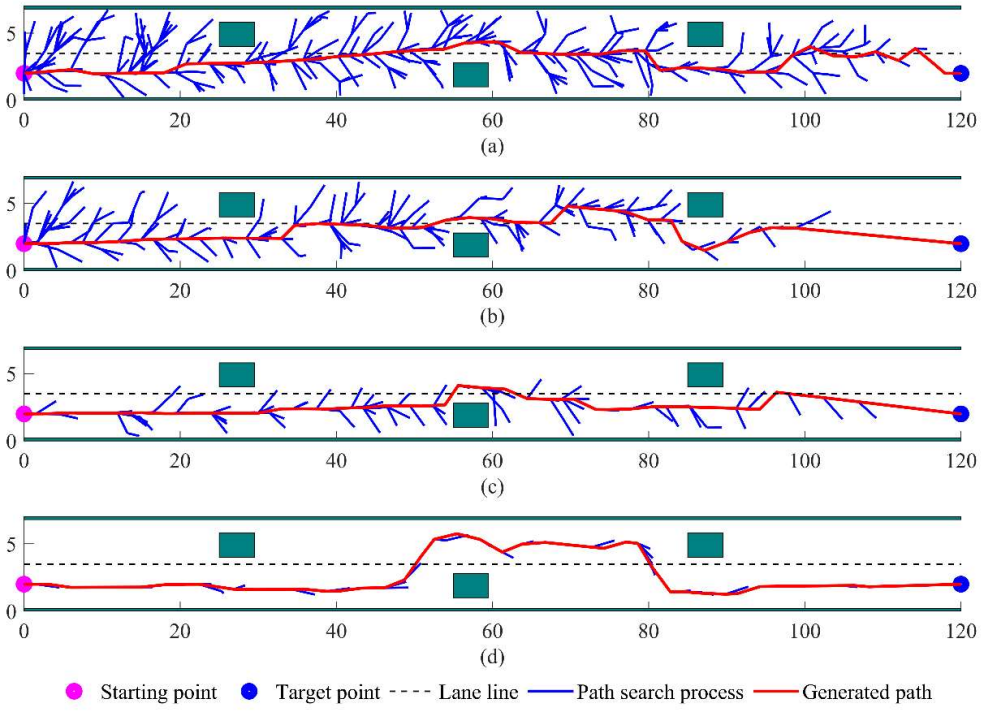

Figure 12. Comparison of path search results in the Map 2 scenario. (**a**) RRT* path search results; (**b**) Goal Biased RRT* path search results; (**c**) P-RRT* path search results; (**d**) Improved RRT* path search results.

The Map 3 scenario is mainly used to verify the effectiveness of the improved algorithm in this paper under different obstacle sizes and shapes, and the simulation results are shown in Figure 13. On the same road as Map 1, obstacles of different sizes and shapes appear, and the proposed improved algorithm is still practical for such a complex environment. The improved algorithm in this paper can still plan the path with a high safety factor while

the redundant nodes are significantly reduced, and the path search speed is accelerated. According to the data analysis in Table 4, the average path length planned by the improved algorithm is 2.14% less than that of the RRT* algorithm. Compared with the P-RRT* algorithm, the average running time of the improved algorithm is reduced by 34.56%, the average number of iterations is reduced by 37.31%, and the memory consumption is reduced by 16.4%. The proposed algorithm still performs well in the face of obstacles of different sizes and shapes.

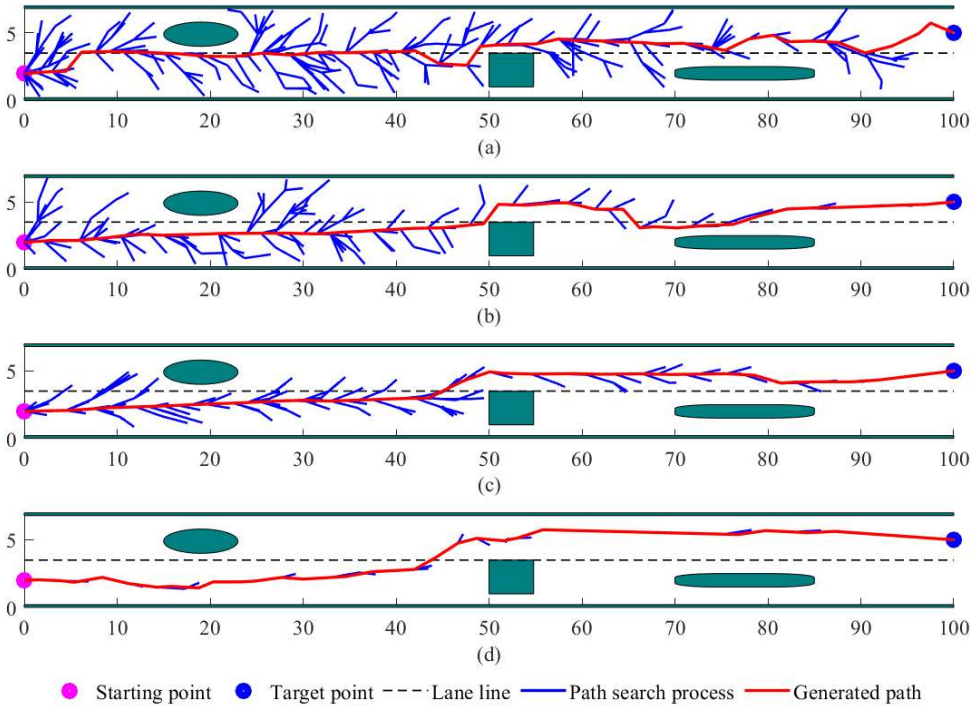

Figure 13. Comparison of path search results in the Map 3 scenario. (**a**) RRT* path search results; (**b**) Goal Biased RRT* path search results; (**c**) P-RRT* path search results; (**d**) Improved RRT* path search results.

In the simulations of the three Map scenarios shown in Figures 11–13, although the improved RRT* algorithm generates the base path, there are problems such as too many path points, too long path length, and too large path curvature, which do not meet the driving conditions of autonomous vehicles and are not the required optimal paths. Therefore, in this study, the path generated by the improved algorithm was optimized using path pruning based on the maximum steering angle of the vehicle and the third-order B-spline algorithm. The optimization results are shown in Figure 14. The red path represents the initially generated path, while the green path represents the trimmed path, and it can be seen that the redundant points in the path have been reduced. The black path represents the optimized final, whose curvature is shown in Figure 15. The path curvature after path optimization in all three Map scenarios is continuous and can meet the vehicle driving requirements.

4.2. Simulation Analysis of Path Tracking Control

The effectiveness and adaptability of the proposed path planning algorithm and path tracking control strategy were verified through joint simulation experiments using CarSim and MATLAB/Simulink. The main parameters of the vehicle are shown in Table 5. To validate the feasibility of the planned path and the effectiveness of the tracking control strategy, the planned paths in two different Map scenarios were tested for path tracking. As the driving environment was a normal urban road with well-dried asphalt surfaces, the road adhesion coefficient was set to 0.8 in this study.

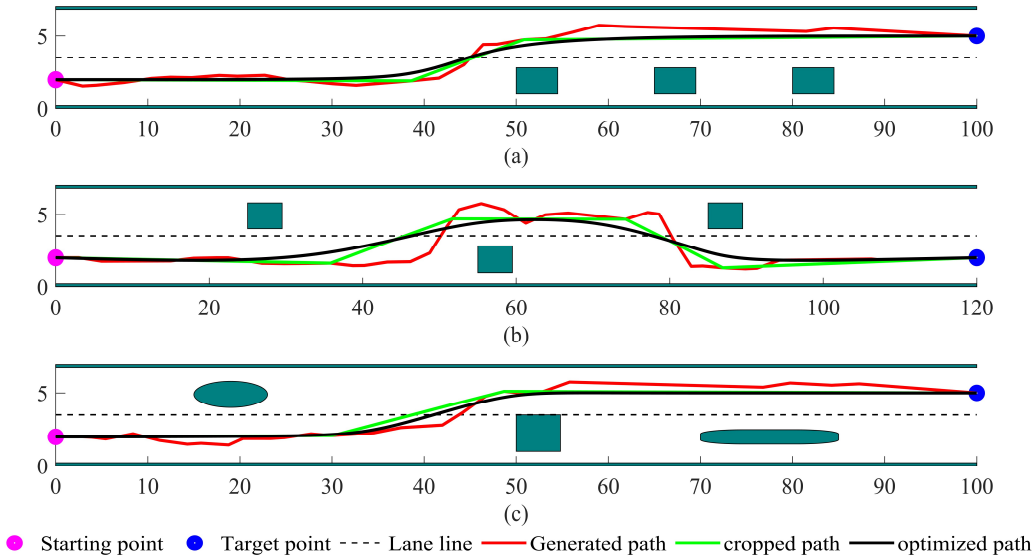

Figure 14. Path optimization in the two Map scenarios. (**a**) The path optimization process under Map 1; (**b**) the path optimization process under Map 2; (**c**) the path optimization process under Map 3.

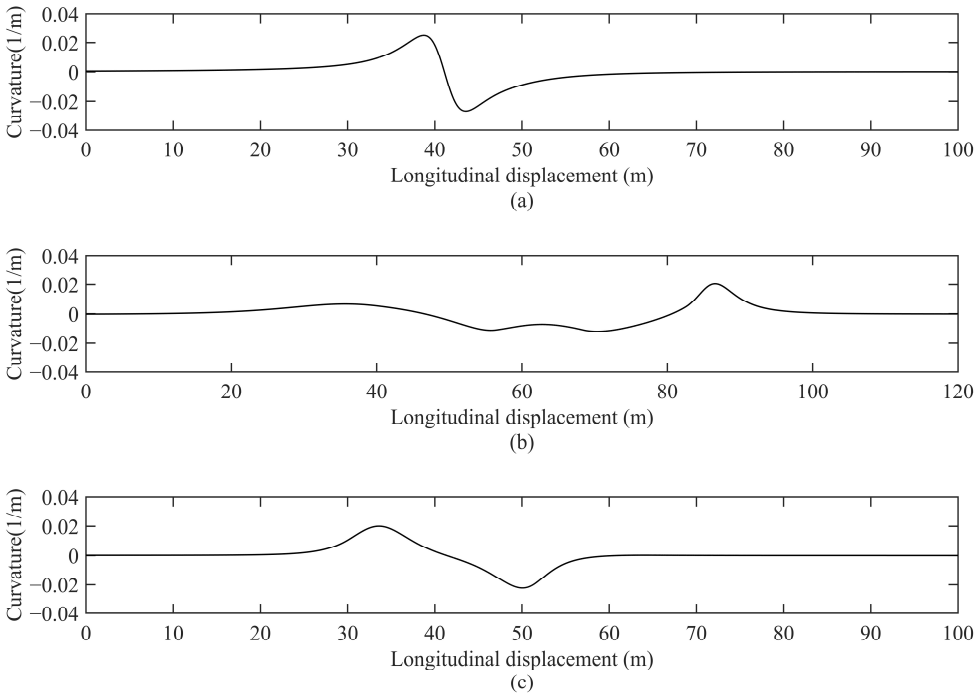

Figure 15. The curvature of optimized paths in the two Map scenarios. (**a**) Optimized path curvature under Map 1; (**b**) optimized path curvature under Map 2; (**c**) optimized path curvature under Map 3.

The performance of the PSO-optimized LQR path tracking controller was first verified by using the two paths planned in this paper. Simulation tests were conducted at speeds of 10 m/s and 20 m/s for both LQR and PSO-optimized LQR controllers, and representative parameters were selected for comparison. The results are shown in Figures 16 and 17.

Figure 16 describes the tracking performance of the LQR controller and the PSO-optimized LQR controller under the path of Map 1 at speeds of 10 m/s and 20 m/s. According to parts (a) and (c), the maximum lateral tracking error of the LQR controller can be controlled to within 0.13 m at different speeds, and the PSO-LQR is reduced by about 54% compared to it. Parts (b) and (d) provide the analysis of the overall vehicle stability.

When the longitudinal speed is 10 m/s, there is not much difference in the stability of LQR before and after optimization. However, when the speed is increased to 20 m/s, the overall vehicle stability under PSO-optimized LQR control is significantly better.

Table 5. Main parameters of the vehicles.

Parameters/Units	Value
Vehicle mass/kg	1412
Distance from the center of mass to the front axis/mm	1015
Distance from the center of mass to the rear axis/mm	1895
Moment of inertia/kg·m^2	1536.7
Front-wheel cornering stiffness/N/rad	−148,970
Rear wheel cornering stiffness/N/rad	−82,204
Wheelbase of the front axle/mm	1675
Height of the center of mass/mm	540
Effective radius of wheel/mm	325

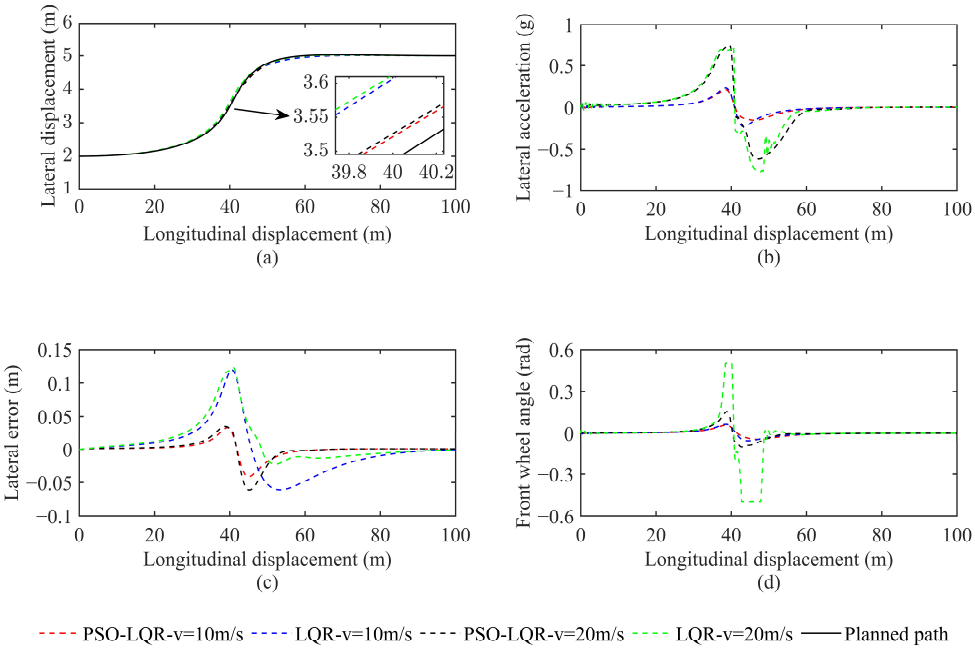

Figure 16. Comparison of controller tracking effects under path 1. (**a**) Horizontal position tracking comparison; (**b**) comparison of lateral acceleration; (**c**) comparison of lateral errors; (**d**) compare the front wheel angles.

Figure 17 describes the tracking performance of the LQR controller and PSO-optimized LQR controller under two speeds, 10 m/s, and 20 m/s, on the path of Map 2. Perfect driving in the double-lane scenario is challenging, and both control algorithms have some errors in turns with large curvatures. However, the PSO-optimized LQR controller has a more minor lateral error, and the vehicle can quickly correct the current state when the error is large. By comparing parts (a) and (c), the maximum lateral tracking error of the LQR controller at different speeds in both paths can be controlled within 0.14 m. For the PSO-optimized LQR controller, it is reduced by about 64% to 0.05 m. This indicates that the improved LQR controller can more effectively reduce errors and keep the vehicle in a safe state when the tracking error is large. Similarly, at a speed of 20 m/s, parts (b) and (d) show that the whole vehicle stability under the PSO-optimized LQR controller is higher.

To verify the feasibility of the planned paths and to investigate the accuracy and stability of the improved LQR controller in terms of speed, two planned paths were tracked and tested at three different vehicle speeds. The simulation results are shown in Figures 18 and 19.

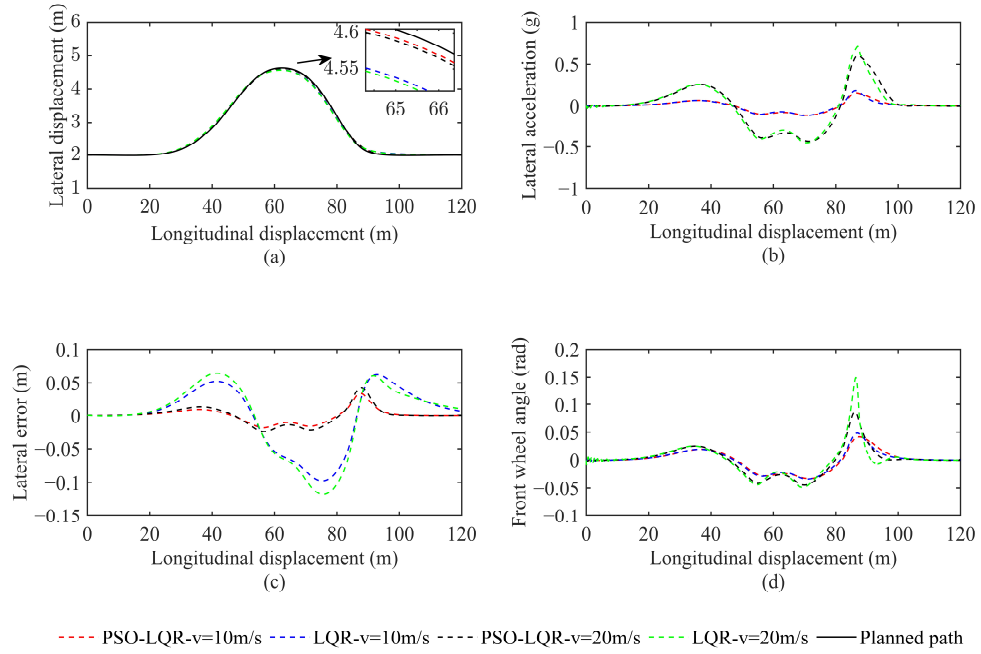

Figure 17. Comparison of controller tracking effects under path 2. (**a**) Horizontal position tracking comparison; (**b**) comparison of lateral acceleration; (**c**) comparison of lateral errors; (**d**) compare the front wheel angles.

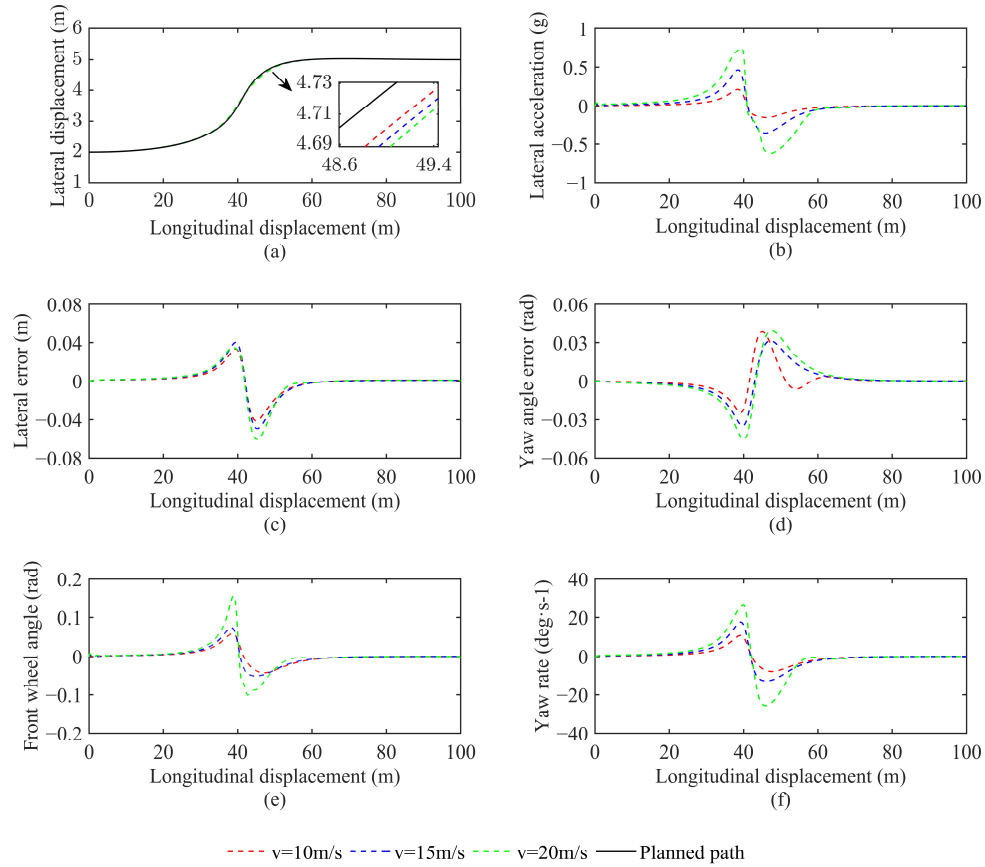

Figure 18. Comparison of path tracking effects on Map 1 under three different speeds. (**a**) Horizontal position tracking comparison; (**b**) comparison of lateral acceleration; (**c**) comparison of lateral errors; (**d**) comparison of heading errors; (**e**) compare the front wheel angles; (**f**) comparison of yaw rate.

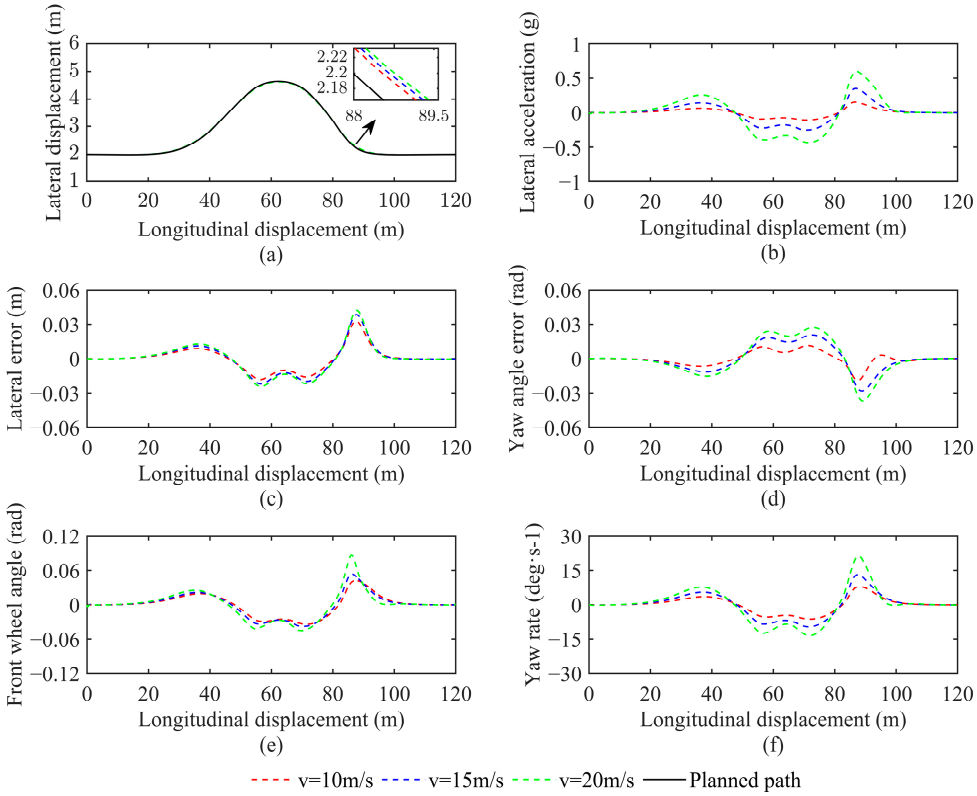

Figure 19. Comparison of path tracking effects on Map 2 under three different speeds. (**a**) Horizontal position tracking comparison; (**b**) comparison of lateral acceleration; (**c**) comparison of lateral errors; (**d**) comparison of heading errors; (**e**) compare the front wheel angles; (**f**) comparison of yaw rate.

A vehicle speed of 20 m/s was added to the first two simulated speeds to better validate the tracking performance of the PSO-LQR tracking controller. The simulation results are presented in Figures 18 and 19. Figures 18a and 19a demonstrate that the vehicle can effectively follow the planned path at all speeds, indicating the robustness of the controller. Figures 18b,c and 19c show that the lateral position tracking error increases with speed; however, the lateral position errors at different paths and speeds can be controlled within 0.06 m, which is within the acceptable range and performs well. Similarly, Figures 18d and 19d indicate that the yaw tracking error can be controlled within 0.05 rad at different speeds, and the differences are insignificant, suggesting that the vehicle's tracking process is relatively stable. Figures 18b,e,f, and 19b,e,f, describe that as the speed increases, the changes in lateral acceleration, front-wheel steering angle, and yaw rate become greater. Still, the front-wheel angle and lateral acceleration at different speeds do not have any step changes, and the yaw rate can be controlled within an acceptable range, indicating that the tracking process is relatively stable. Overall, the PSO-optimized LQR controller's robustness, accuracy, and stability are acceptable at speeds that are not too fast. The planned paths meet the tracking requirements of autonomous vehicles and achieve good tracking performance at different speeds.

5. Conclusions

To address the slow convergence and randomness of the RRT* algorithm and the relatively empirical selection of the weight matrix of the LQR algorithm. This paper proposes a framework to improve the RRT* and PSO-LQR algorithms for path planning and tracking control of self-driving vehicles under ordinary urban road conditions. The framework employs a variable sampling area to limit the generation of random sampling points. Moreover, an improved artificial potential field method is integrated into the RRT* algorithm to improve its convergence speed. The path optimization is then performed using path prun-

ing based on the vehicle's maximum steering angle constraint and the three-time B-spline algorithm to obtain curvature-continuous paths. The simulation results of the average three Map scenarios show that the improved RRT* algorithm optimizes 34.45%, 35.08%, and 16.87% of the path search time, iteration number, and memory consumption, respectively, compared with P-RRT*, demonstrating the significant advantages of the improved algorithm. In addition, the final path optimized by the path optimization algorithm is smooth and curvature continuous, which satisfies the path tracking requirements. In the path-tracking stage, the PSO algorithm optimizes the LQR weight matrix and adds feedforward control compensation to improve the path-tracking accuracy. According to the simulation results under two different paths, the path tracking accuracy of the PSO-LQR path tracking controller is improved by 59% compared with the conventional LQR, and its robustness is also significantly improved. In addition, by comparing the tracking effects under different vehicle speeds, the path tracking errors can all be controlled within 0.06 m, which verifies the effectiveness of the proposed path planning and trajectory tracking framework.

However, further research is required to investigate and address the deployment of the proposed path planning algorithm and tracking control strategy for accurate vehicle testing. In future work, the integration of machine learning and intelligent optimization algorithms into path planning and path tracking control research will be emphasized to develop faster and safer path planners and more precise and efficient path tracking controllers.

Author Contributions: Conceptualization, Y.Z. and F.G.; methodology, Y.Z. and F.G.; software, F.G. and F.Z.; validation, F.Z. and F.G.; formal analysis, Y.Z. and F.G.; investigation, F.Z.; resources, F.G.; data curation, F.G.; funding acquisition, Y.Z. and F.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Industrial Proactive and Key Technology Program of Jiangsu Province (Grant number BE2022053-2), the Modern Agriculture-Key and General Program of Jiangsu Province (Grant number BE2021339), the Philosophy and Social Science Program of the Higher Education Institutions of Jiangsu Province (Grant number 2021SJA0151), and the Science and Technology Innovation Foundation for Young Scientists of Nanjing Forestry University (Grant number CX2019018).

Data Availability Statement: Not applicable.

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

References

1. Lv, Z.; Shang, W. Impacts of Intelligent Transportation Systems on Energy Conservation and Emission Reduction of Transport Systems: A Comprehensive Review. *Green Technol. Sustain.* **2023**, *1*, 100002. [CrossRef]
2. Yang, H.; Zheng, C.; Zhao, Y.; Wu, Z. Integrating the Intelligent Driver Model With the Action Point Paradigm to Enhance the Performance of Autonomous Driving. *IEEE Access* **2020**, *8*, 106284–106295. [CrossRef]
3. Khaled Ahmed, S.; Mohammed Ali, R.; Maha Lashin, M.; Fayroz Sherif, F. Designing a New Fast Solution to Control Isolation Rooms in Hospitals Depending on Artificial Intelligence Decision. *Biomed. Signal Process. Control* **2023**, *79*, 104100. [CrossRef]
4. Talavera, E.; Díaz-Álvarez, A.; Naranjo, J.E.; Olaverri-Monreal, C. Autonomous Vehicles Technological Trends. *Electronics* **2021**, *10*, 1207. [CrossRef]
5. Sun, Y.; Ren, D.; Lian, S.; Fu, S.; Teng, X.; Fan, M. Robust Path Planner for Autonomous Vehicles on Roads with Large Curvature. *IEEE Robot. Autom. Lett.* **2022**, *7*, 2503–2510. [CrossRef]
6. bt Mohd Shamsuddin, P.N.F.; bin Mansor, M.A. Motion Control Algorithm for Path Following and Trajectory Tracking for Unmanned Surface Vehicle: A Review Paper. In Proceedings of the 2018 3rd International Conference on Control, Robotics and Cybernetics (CRC), Penang, Malaysia, 26–28 September 2018; pp. 73–77.
7. González, D.; Pérez, J.; Milanés, V.; Nashashibi, F. A Review of Motion Planning Techniques for Automated Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 1135–1145. [CrossRef]
8. Bresciani, M.; Ruscio, F.; Tani, S.; Peralta, G.; Timperi, A.; Guerrero-Font, E.; Bonin-Font, F.; Caiti, A.; Costanzi, R. Path Planning for Underwater Information Gathering Based on Genetic Algorithms and Data Stochastic Models. *J. Mar. Sci. Eng.* **2021**, *9*, 1183. [CrossRef]
9. Zhao, P.; Chang, Y.; Wu, W.; Luo, H.; Zhou, Z.; Qiao, Y.; Li, Y.; Zhao, C.; Huang, Z.; Liu, B.; et al. Dynamic RRT: Fast Feasible Path Planning in Randomly Distributed Obstacle Environments. *J. Intell. Robot. Syst.* **2023**, *107*, 48. [CrossRef]

10. Jin, X.; Yan, Z.; Yang, H.; Wang, Q.; Yin, G. A Goal-Biased RRT Path Planning Approach for Autonomous Ground Vehicle. In Proceedings of the 2020 4th CAA International Conference on Vehicular Control and Intelligence (CVCI), Hangzhou, China, 18–20 December 2020; pp. 743–746.
11. Zhu, Y.; Tang, Y.; Zhang, Y.; Huang, Y. Path Planning of Manipulator Based on Improved RRT-Connect Algorithm. In Proceedings of the 2021 2nd International Conference on Big Data & Artificial Intelligence & Software Engineering (ICBASE), Zhuhai, China, 24–26 September 2021; pp. 44–47.
12. Dai, J.; Zhang, Y.; Deng, H. Novel Potential Guided Bidirectional RRT* with Direct Connection Strategy for Path Planning of Redundant Robot Manipulators in Joint Space. *IEEE Trans. Ind. Electron.* **2023**, 1–10. [CrossRef]
13. Gammell, J.D.; Srinivasa, S.S.; Barfoot, T.D. 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, Chicago, IL, USA, 14–18 September 2014; pp. 2997–3004.
14. Qureshi, A.H.; Ayaz, Y. Potential Functions Based Sampling Heuristic for Optimal Path Planning. *Auton. Robot.* **2016**, *40*, 1079–1093. [CrossRef]
15. Chen, Z.; Yu, J.; Zhao, Z.; Wang, X.; Chen, Y. A Path-Planning Method Considering Environmental Disturbance Based on VPF-RRT*. *Drones* **2023**, *7*, 145. [CrossRef]
16. Fan, J.; Chen, X.; Liang, X. UAV Trajectory Planning Based on Bi-Directional APF-RRT* Algorithm with Goal-Biased. *Expert Syst. Appl.* **2023**, *213*, 119137. [CrossRef]
17. Ayawli, B.B.K.; Mei, X.; Shen, M.; Appiah, A.Y.; Kyeremeh, F. Optimized RRT-A* Path Planning Method for Mobile Robots in Partially Known Environment. *Inf. Technol. Control* **2019**, *48*, 179–194. [CrossRef]
18. Ghosh, D.; Nandakumar, G.; Narayanan, K.; Honkote, V.; Sharma, S. Kinematic Constraints Based Bi-Directional RRT (KB-RRT) with Parameterized Trajectories for Robot Path Planning in Cluttered Environment. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 8627–8633.
19. Peng, J.; Chen, Y.; Duan, Y.; Zhang, Y.; Ji, J.; Zhang, Y. Towards an Online RRT-Based Path Planning Algorithm for Ackermann-Steering Vehicles. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; pp. 7407–7413.
20. Liao, B.; Hua, Y.; Wan, F.; Zhu, S.; Zong, Y.; Qing, X. Stack-RRT*: A Random Tree Expansion Algorithm for Smooth Path Planning. *Int. J. Control Autom. Syst.* **2023**, *21*, 993–1004. [CrossRef]
21. Li, H.; Luo, J.; Yan, S.; Zhu, M.; Hu, Q.; Liu, Z. Research on Parking Control of Bus Based on Improved Pure Pursuit Algorithms. In Proceedings of the 2019 18th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES), Wuhan, China, 8–10 November 2019; pp. 21–26.
22. Wang, L.; Zhai, Z.; Zhu, Z.; Mao, E. Path Tracking Control of an Autonomous Tractor Using Improved Stanley Controller Optimized with Multiple-Population Genetic Algorithm. *Actuators* **2022**, *11*, 22. [CrossRef]
23. Yao, J.; Ge, Z. Path-Tracking Control Strategy of Unmanned Vehicle Based on DDPG Algorithm. *Sensors* **2022**, *22*, 7881. [CrossRef]
24. Rupp, A.; Stolz, M. Survey on Control Schemes for Automated Driving on Highways. In *Automated Driving: Safer and More Efficient Future Driving*; Watzenig, D., Horn, M., Eds.; Springer International Publishing: Cham, Switzerland, 2017; pp. 43–69, ISBN 978-3-319-31895-0.
25. Hu, C.; Chen, Y.; Wang, J. Fuzzy Observer-Based Transitional Path-Tracking Control for Autonomous Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 3078–3088. [CrossRef]
26. Wu, Y.; Wang, L.; Zhang, J.; Li, F. Path Following Control of Autonomous Ground Vehicle Based on Nonsingular Terminal Sliding Mode and Active Disturbance Rejection Control. *IEEE Trans. Veh. Technol.* **2019**, *68*, 6379–6390. [CrossRef]
27. Tian, J.; Yang, M. Research on Trajectory Tracking and Body Attitude Control of Autonomous Ground Vehicle Based on Differential Steering. *PLoS ONE* **2023**, *18*, e02732552023. [CrossRef]
28. Tian, J.; Zeng, Q.; Wang, P.; Wang, X. Active Steering Control Based on Preview Theory for Articulated Heavy Vehicles. *PLoS ONE* **2021**, *16*, e02520982021. [CrossRef]
29. Kapania, N.R.; Gerdes, J.C. Design of a Feedback-Feedforward Steering Controller for Accurate Path Tracking and Stability at the Limits of Handling. *Veh. Syst. Dyn.* **2015**, *53*, 1687–1704. [CrossRef]
30. Xu, S.; Peng, H. Design, Analysis, and Experiments of Preview Path Tracking Control for Autonomous Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2020**, *21*, 48–58. [CrossRef]
31. Yang, T.; Bai, Z.; Li, Z.; Feng, N.; Chen, L. Intelligent Vehicle Lateral Control Method Based on Feedforward + Predictive LQR Algorithm. *Actuators* **2021**, *10*, 228. [CrossRef]
32. Lu, A.; Lu, Z.; Li, R.; Tian, G. Adaptive LQR Path Tracking Control for 4WS Electric Vehicles Based on Genetic Algorithm. In Proceedings of the 2022 6th CAA International Conference on Vehicular Control and Intelligence (CVCI), Nanjing, China, 28–30 October 2022; pp. 1–6.
33. Wang, Z.; Sun, K.; Ma, S.; Sun, L.; Gao, W.; Dong, Z. Improved Linear Quadratic Regulator Lateral Path Tracking Approach Based on a Real-Time Updated Algorithm with Fuzzy Control and Cosine Similarity for Autonomous Vehicles. *Electronics* **2022**, *11*, 3703. [CrossRef]
34. Zhang, J.; Wang, H.; Zheng, J.; Cao, Z.; Man, Z.; Yu, M.; Chen, L. Adaptive Sliding Mode-Based Lateral Stability Control of Steer-by-Wire Vehicles With Experimental Validations. *IEEE Trans. Veh. Technol.* **2020**, *69*, 9589–9600. [CrossRef]

35. Li, B.L.; Zeng, L. Fractional Calculus Control of Road Vehicle Lateral Stability after a Tire Blowout. *Mechanika* **2021**, *27*, 475–482. [CrossRef]
36. Pacejka, H.B.; Besselink, I.J.M. Magic Formula Tyre Model with Transient Properties. *Veh. Syst. Dyn.* **1997**, *27*, 234–249. [CrossRef]
37. Hou, Y.; Xu, X. High-Speed Lateral Stability and Trajectory Tracking Performance for a Tractor-Semitrailer with Active Trailer Steering. *PLoS ONE* **2022**, *17*, e02773582022. [CrossRef]
38. Li, Y.; Ma, Z.; Zheng, M.; Li, D.; Lu, Z.; Xu, B. Performance Analysis and Optimization of a High-Temperature PEMFC Vehicle Based on Particle Swarm Optimization Algorithm. *Membranes* **2021**, *11*, 691. [CrossRef]
39. Xu, X.; Lin, P. Parameter Identification of Sound Absorption Model of Porous Materials Based on Modified Particle Swarm Optimization Algorithm. *PLoS ONE* **2021**, *16*, e02509502021. [CrossRef]
40. Cheng, Z.; Lu, Z. Regression-Based Correction and I-PSO-Based Optimization of HMCVT's Speed Regulating Characteristics for Agricultural Machinery. *Agriculture* **2022**, *12*, 580. [CrossRef]