Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 4977041, 13 pages
https://doi.org/10.1155/2021/4977041

Research Article

Research on Global Dynamic Path Planning Method Based on Improved A^* Algorithm

Chuanhu Niu,[1] **Aijuan Li**,[1,2] **Xin Huang**,[3] **Wei Li**,[1] **and Chuanyan Xu**[1]

[1]*School of Automotive Engineering, Shandong Jiaotong University, Jinan 250357, China*
[2]*Energy and Power Engineering College, Nanjing University of Aeronautics & Astronautics, Nanjing 210016, China*
[3]*School of Information Science and Electrical Engineering, Shandong Jiaotong University, Jinan 250357, China*

Correspondence should be addressed to Aijuan Li; 65826990@qq.com and Xin Huang; huangxin@sdjtu.edu.cn

Received 17 May 2021; Accepted 26 July 2021; Published 4 August 2021

Academic Editor: Xianjian Jin

Aiming at the optimal path and planning efficiency of global path planning for intelligent driving, this paper proposes a global dynamic path planning method based on improved A^* algorithm. First, this method improves the heuristic function of the traditional A^* algorithm to improve the efficiency of global path planning. Second, this method uses a path optimization strategy to make the global path smoother. Third, this method is combined with the dynamic window method to improve the real-time performance of the dynamic obstacle avoidance of the intelligent vehicle. Finally, the global dynamic path planning method of the proposed improved A^* algorithm is verified through simulation experiments and real vehicle tests. In the simulation analysis, compared with the modified A^* algorithm and the traditional A^* algorithm, the method in this paper shortens the path distance by 2.5%~3.0%, increases the efficiency by 10.3%~13.6% and generates a smoother path. In the actual vehicle test, the vehicle can avoid dynamic obstacles in real time. Therefore, the method proposed in this paper can be applied on the intelligent vehicle platform. The path planning efficiency is high, and the dynamic obstacle avoidance is good in real time.

1. Introduction

As one of the development directions of future automobiles, intelligent driving is receiving more and more attention [1]. In particular, path planning is an important part of intelligent driving. Path planning is an obstacle-free path from the starting point to the target point that the intelligent vehicle plans out based on environmental information [2]. Especially in the dynamic environment, in order to ensure the real-time obstacle avoidance and the efficiency of path planning, it is necessary to improve the path planning algorithm.

In recent years, the most representative and common path planning algorithms in the field are mainly divided into neural network algorithm [3], artificial potential field algorithm [4], rapidly expanding random tree algorithm [5], ant colony algorithm [6], and A^* algorithm [7, 8]. In particular, the A^* algorithm is a heuristic search algorithm; because of its strong global search ability, high search efficiency, and shortest path, it is widely used. Ziang Zhang et al. [9] proposed an improved hybrid path planning method for a spherical mobile robot based on a pendulum, which improves the efficiency of path search, but it is aimed at a spherical mobile robot. Bijun Tang et al. [10] proposed an algorithm that uses an artificial potential field method to optimize the path of the hybrid A^* algorithm. The generated path not only is smooth but also maintains a safe distance from obstacles. However, the real-time obstacle avoidance is not good in a dynamic environment. Oleiwi et al. [11] proposed a path planning method for multiobjective mobile robots based on the ant colony algorithm and genetic algorithm which can efficiently select the optimal path for multiobjectives in a static environment, but it is not suitable for dynamic environments. Jikai Wang et al. [12] proposed a global path planning framework based on hybrid mapping, which improved the efficiency of path planning in complex environments, but it cannot guarantee the optimal path. Xiaoru Song et al. [13] proposed a dynamic global path planning method suitable for mobile robots, which can plan a smooth path in a dynamic environment, but the efficiency of path planning still needs to be improved. The A^* algorithm based on a grid map is suitable for global path planning. This algorithm has the advantages such as a simple structure and small calculation amount [14]. However, the path planned by the traditional A^* algorithm has many folding points, which is not conducive to the driving of the intelligent vehicle. Moreover, if the space of the environment increases, A^* algorithm needs large storage space and it has low efficiency and poor real-time performance. The dynamic window method has good obstacle avoidance ability in a dynamic environment, but it is not suitable for global path planning [15].

Aiming at the optimal path and planning efficiency of global path planning for intelligent driving, this paper proposes a global dynamic path planning method based on improved A^* algorithm and dynamic window method. The improved path planning method has many advantages. First, the heuristic function of the traditional A^* algorithm is improved to make the A^* path search more biased. The time of the path planning is reduced, and the efficiency is increased. Then, the optimization strategy is used to reduce the redundant turning points and nodes of the path planning. The distance of the path is optimized, and the smoothness of the path is improved. Finally, the A^* algorithm global search capability is combined with the dynamic window method local planning capability so that the intelligent vehicle can perform global dynamic path planning, and the real-time performance of dynamic obstacle avoidance is good. The remainder of the paper is organized as follows: Section 2 discusses the improved A^* algorithm; Section 3 discusses the dynamic window method; Section 4 discusses the simulation and experimental results; Section 5 discusses the real vehicle test; and Section 6 discusses the conclusions of this research.

2. The Improved $A*$ Algorithm

The traditional A^* algorithm is a heuristic search algorithm, which constantly expands the nodes and calculates the value of each node. Finally, we can find a path with the least value. The use of heuristic function can greatly improve the search efficiency. The formula of traditional A^* algorithm is shown as follows:

$$f(n) = g(n) + h(n), \tag{1}$$

where $f(n)$ is the estimated value from the initial node to the target node, $g(n)$ is the actual value from the initial node to the node of state n, and $h(n)$ is the estimated value from the node of state n to the target node.

The selection of $h(n)$ directly affects the performance of the algorithm. Only when the value of $h(n)$ is closer to the actual cost value from the node of state n to the target node, the optimal path can be guaranteed and the efficiency of the algorithm can be improved. Therefore, the following improvements have been made.

2.1. Improved Heuristic Function h (n). Assume that $d(n)$ is the actual value from the node of state n to the target node. When the heuristic function $h(n)$ is selected differently, the following three conditions will occur:

(1) When $h(n) > d(n)$, the search range of A^* algorithm is small and the extended nodes are relatively few. Therefore, the algorithm has high efficiency, but the result is not the optimal path.

(2) When $h(n) < d(n)$, A^* algorithm has a large search range and a relatively large number of extended nodes. Therefore, the algorithm has low efficiency, but the result can get the optimal path.

(3) When $h(n) = d(n)$, this is the most ideal choice, so the A^* algorithm will search strictly according to the shortest path. Therefore, the A^* algorithm has the highest search efficiency.

In the traditional A^* algorithm, the heuristic function $h(n)$ usually used Euclidean distance [16] $h_1(n)$, Manhattan distance [17] $h_2(n)$, or Chebyshev distance $h_3(n)$:

$$h_1(n) = D * \sqrt{\left(M_x - G_x\right)^2 + \left(M_y - G_y\right)^2}, \tag{2}$$

$$h_2(n) = D * \left(\left|M_x - G_x\right| + \left|M_y - G_y\right|\right), \tag{3}$$

$$h_3(n) = D * \max\left(\left|M_x - G_x\right|, \left|M_y - G_y\right|\right), \tag{4}$$

where (M_x, M_y) represents the coordinate value of the current node, (G_x, G_y) represents the coordinate value of the target node, and D is the actual cost value of the intelligent vehicle moving one grid.

In order to make the heuristic function $h(n)$ closer to the actual value $d(n)$, a new heuristic function is designed using Manhattan distance and Chebyshev distance. The heuristic function is then dynamically weighted:

$$\begin{aligned} h(n) &= D * \Big(\left|M_x - G_x\right| + \left|M_y - G_y\right| - 2 \\ &\quad * \min\left(\left|M_x - G_x\right|, \left|M_y - G_y\right|\right)\Big) \\ &\quad + \sqrt{2} * \min\left(\left|M_x - G_x\right|, \left|M_y - G_y\right|\right), \\ f(n) &= g(n) + (1 + k(n))h(n), \\ k(n) &= \begin{cases} 1 - \dfrac{m(n)}{R}, & \dfrac{m(n)}{R} > K, \\ 1 - K, & \dfrac{m(n)}{R} \le K, \end{cases} \end{aligned} \tag{5}$$

where $m(n)$ is the depth of search, R is the expected path length, and K is constant. Other parameters are the same as formulas (2)–(4).

2.2. Path Optimization Strategy. The traditional A^* algorithm expands nodes based on the grid, which contains more turning points and redundant nodes [18]. They are not good for driving intelligent vehicles. In order to solve these problems, this paper proposes a path optimization strategy.

(1) Find redundant nodes and remove them. Assume that the path planned by the A^* algorithm is $\{M_k \mid k = 1, 2, 3, \ldots, n\}$. First, start from the second node M_2 of the path plan and judge whether the child node M_3 of M_2 and its parent node M_1 are in the same straight line. If it is on the same straight line, M_2 is a redundant node. Delete and update the path list. Then, check whether the child node of the next node and its parent node are on the same line. Delete redundant nodes and update the path list. Finally, all nodes are traversed to get a set of points including the starting point, turning point, and target point.

(2) Look for redundant turning points and delete them. Assume that the path planned by the A^* algorithm is $\{M_k \mid k = 1, 2, 3, \ldots, n\}$. After the optimization of strategy (1), except the starting point M_1 and the target point M_n, the other nodes are turning points. First, connect the node M_1M_3. If the straight line M_1M_3 does not pass through the obstacle and the distance from the nearest obstacle is greater than the set safe distance, M_2 is the redundant turning point and M_2 is deleted. Then connect M_1M_k ($k = 4, 5, 6, \ldots, n$) and repeat the above inspection steps. If M_1M_k passes through an obstacle or the distance from the nearest obstacle is less than the safe distance, the node M_{k-2} is deleted and the node M_1M_{k-1} is connected. Update the path list and connect node M_2M_k ($k = 4, 5, 6, \ldots, n$). Finally, repeat the above checking steps until all the nodes are traversed.

3. Dynamic Window Method

The dynamic window method can make an intelligent vehicle have a good obstacle avoidance ability in a dynamic environment. The dynamic window method is mainly used to sample multiple groups of velocities in the velocity space (linear velocity v and angular velocity w) and simulate the trajectory of intelligent vehicle in the next time interval. After obtaining multiple sets of trajectories, the multiple sets of trajectories are evaluated according to the evaluation function [19] and the intelligent vehicle will select the speed corresponding to the optimal trajectory for the next step of driving [20].

3.1. The Vehicle Kinematics Model. According to the dynamic window method, it can continuously simulate the trajectory of the intelligent vehicle in a period of time. Therefore, it is necessary to know the kinematics model of the intelligent vehicle [21, 22]. The trajectory is represented by $(\dot{x}_t, \dot{y}_t)$. The kinematic model is shown in Figure 1.

Using the fundamental law of dynamics, we can get the dynamic formula:

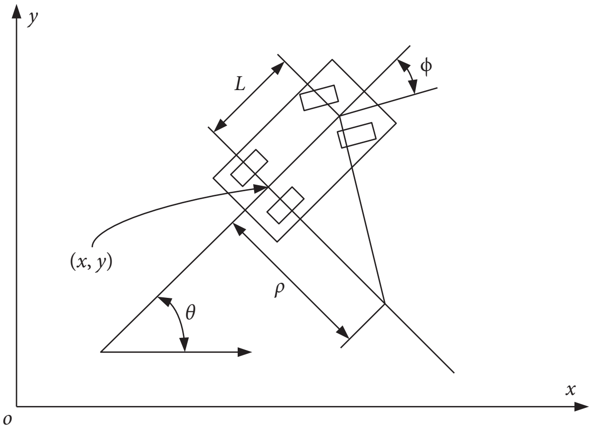


Figure 1: The vehicle kinematics model.

$$\begin{cases} \dot{x}_t = v * \cos(\theta_t), \\ \dot{y}_t = v * \sin(\theta_t), \\ \dot{\theta}_t = v * \dfrac{\tan \phi_t}{L}, \end{cases} \tag{6}$$

where $(\dot{x}_t, \dot{y}_t)$ is the projection speed of the intelligent vehicle on the coordinate axis at time t, v is the speed of the intelligent vehicle, θ_t is the attitude angle at time t, $\dot{\theta}_t$ is the angular velocity of attitude at time t, L is the wheelbase of the intelligent vehicle, ϕ_t is the front wheel angle at time t, and ρ is the turning radius.

In practical application, considering the omnidirectional motion of the intelligent vehicle and the transformation of the world coordinate system, the new kinematics formula is obtained:

$$\begin{aligned} x_t &= x_{t-1} + v_x\Delta t \cos(\theta_t) - v_y\Delta t \sin(\theta_t), \\ y_t &= y_{t-1} + v_y\Delta t \sin(\theta_t) + v_y\Delta t \cos(\theta_t), \\ \theta_t &= \theta_{t-1} + \omega\Delta t. \end{aligned} \tag{7}$$

3.2. Design of Speed Sampling. After establishing the kinematics model of the intelligent vehicle, the trajectory can be calculated according to its speed (linear velocity v and angular velocity w). However, there are infinite groups of speed in the speed space. In order to control the speed sampling better, the speed group must be limited in a certain control range according to the limitations of the intelligent vehicle and the environment space.

(1) The linear speed of the intelligent vehicle and its angular speed limit range formula are shown as follows:

$$V_t = \{v_t \in [v_{\min}, v_{\max}], w_t = [w_{\min}, w_{\max}]\}. \tag{8}$$

(2) In practical application, the motor must go through a certain time interval to make the intelligent vehicle reach the corresponding speed and the speed is within a dynamic range under the influence of the motor torque. Therefore, the formula is shown as follows:

$$V_{t1} = \left\{ (v_t, \omega_t) \,\middle|\, \begin{array}{l} v_t \in [v_d - \dot{v}_e \Delta t, v_d + \dot{v}_c \Delta t] \cap \\ \omega_t \in [\omega_d - \dot{\omega}_e \Delta t, \omega_d + \dot{\omega}_c \Delta t] \end{array} \right\}, \qquad (9)$$

where v_d and ω_d are the current front linear speed and angular speed of the intelligent vehicle, $\dot{v}_c$ and $\dot{v}_e$ are the maximum acceleration and deceleration of linear velocity, respectively, and $\dot{\omega}_c$ and $\dot{\omega}_e$ are the maximum acceleration and deceleration of angular velocity, respectively.

(3) During the operation of the intelligent vehicle, when an obstacle is detected within a safe distance, the intelligent vehicle needs to slow down or even stop. Therefore, it is necessary to further limit the velocity (linear velocity and angular velocity):

$$V_{t2} = \left\{ (v_t, \omega_t) | v_t \leq \sqrt{2 \cdot \text{dist}(v_t, \omega_t) \cdot \dot{v}_e} \cap \omega_t \leq \sqrt{2 \cdot \text{distance}(v_t, \omega_t) \cdot \dot{\omega}_e} \right\}, \qquad (10)$$

where $\text{dist}(v_t, \omega_t)$ is the distance between the current position of the intelligent vehicle and the nearest obstacle.

3.3. Design of Dynamic Window Evaluation Function. According to the dynamic window method, we need an appropriate evaluation function to select the optimal trajectory from the final planned multiple trajectories. The priority criterion of the evaluation function is to make the intelligent vehicle avoid obstacles and move toward the target with the shortest track. The formulas are shown as follows:

$$\begin{aligned} G(v, w) &= \sigma(\alpha \cdot s\_\text{head}(v, w) + \beta \cdot s\_\text{dist}(v, w) + \gamma \cdot s\_\text{velocity}(v, w)), \\ s\_\text{head}(v_t, \omega_t) &= \frac{\text{head}(v_t, \omega_t)}{\sum_{i=1}^{n} \text{head}(v_t, \omega_t)}, \\ s\_\text{dist}(v_t, \omega_t) &= \frac{\text{dist}(v_t, \omega_t)}{\sum_{i=1}^{n} \text{dist}(v_t, \omega_t)}, \\ s\_\text{velocity}(v_t, \omega_t) &= \frac{\text{velocity}(v_t, \omega_t)}{\sum_{i=1}^{n} \text{velocity}(v_t, \omega_t)}, \end{aligned} \qquad (11)$$

where $\text{head}(v_t, \omega_t)$ is the sampling speed of the robot at time t, $\text{dist}(v_t, \omega_t)$ is the distance between the robot's trajectory and the nearest obstacle at time t, and $\text{velocity}(v_t, \omega_t)$ is the velocity of the robot at time t.

3.4. Flowchart of the Algorithm. The flowchart of the algorithm is shown in Figure 2.

Step 1: the map is initialized and the improved A^* algorithm is used for global path planning

Step 2: strategy optimization of the planned path

Step 3: the kinematics model is established, and the velocity group is sampled

Step 4: according to the planned global path and the multiple trajectories simulated by the speed at the next moment, the optimal trajectory is selected by using the evaluation function

Step 5: establishing the optimal path

4. Simulation Experiment and Analysis

In order to verify the effectiveness of the fusion algorithm designed in this paper, MATLAB 2019b is used for simulation experiments to build a grid map scene (20 m × 20 m, grid spacing 1 m) and simultaneously place seven static obstacles of different shapes and sizes and two dynamic obstacles. In the grid map built by the simulation experiment, the starting point coordinates are (1.5 m, 1.5 m) and the target point coordinates are (19.5 m, 19.5 m).

4.1. Simulation Experiment of Improved Heuristic Function. The traditional A^* algorithm has many redundant nodes and large search range, which reduces the efficiency of the algorithm. In this paper, the A^* algorithm is improved to reduce the scope of search and improve the efficiency of the

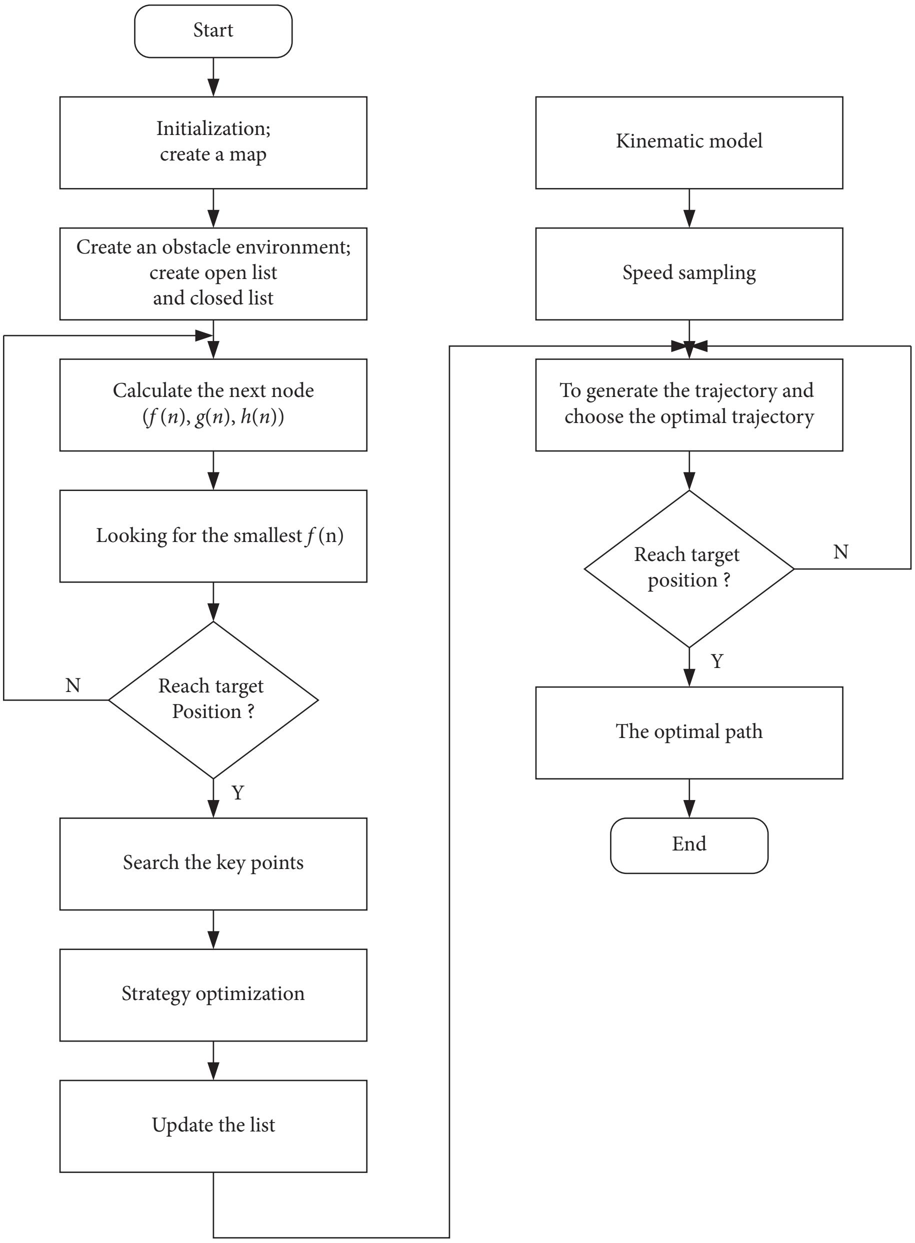

Start
Initialization; create a map
Create an obstacle environment; create open list and closed list
Calculate the next node $(f(n), g(n), h(n))$
Looking for the smallest f(n)
Reach target Position ?
N
Y
Search the key points
Strategy optimization
Update the list
Kinematic model
Speed sampling
To generate the trajectory and choose the optimal trajectory
Reach target position ?
N
Y
The optimal path
End

Figure 2: Flowchart of the algorithm.

algorithm. Black is the initial position, green is the target position, red is the optimal path, and yellow is the search area except the optimal path. The experimental results are shown in Figure 3.

The experimental image of the improved A^* algorithm used in this paper is shown in Figure 3(a). The search area is $128\,\mathrm{m}^2$, the path length is 28.38 m, and the time is 0.040 s.

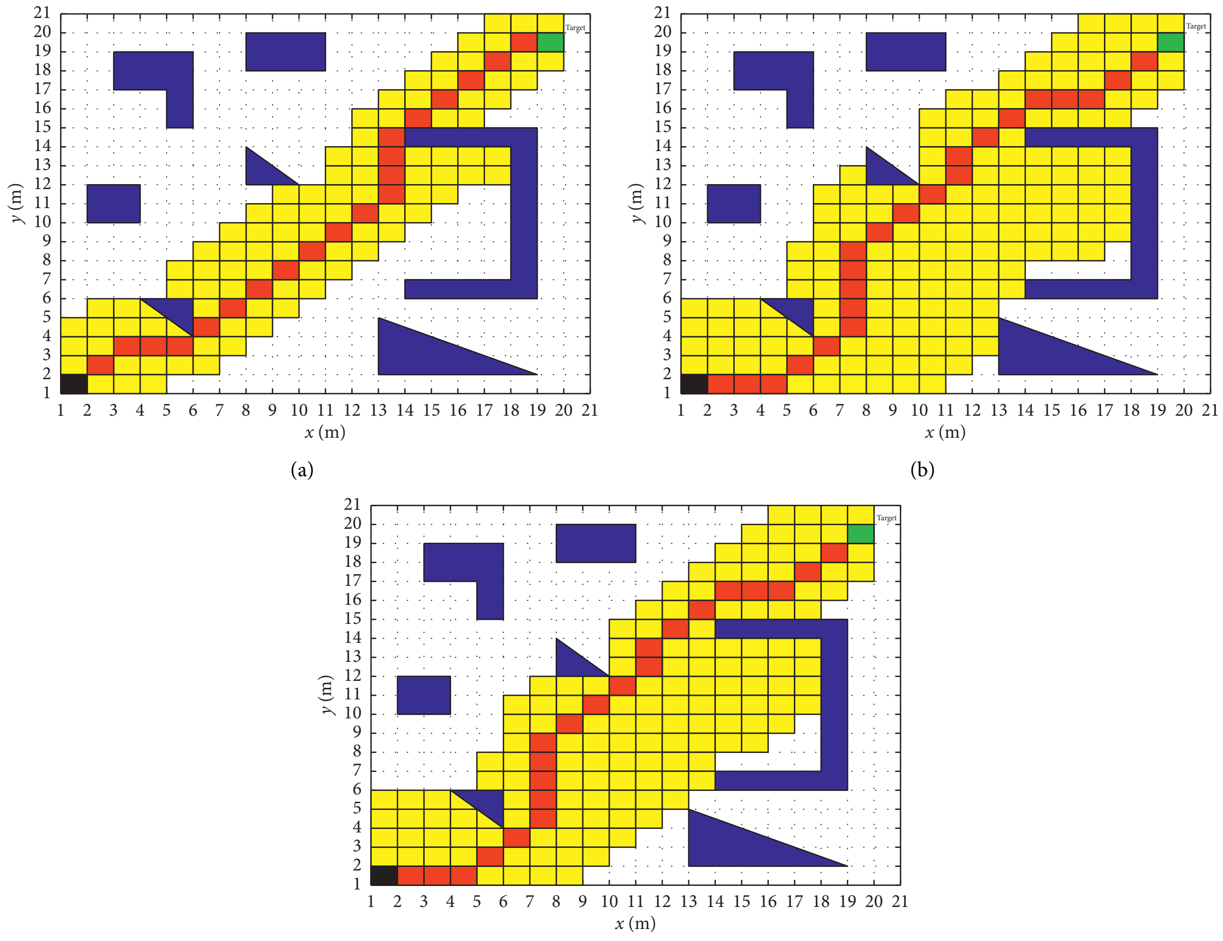
The experimental image of traditional A^* algorithm is shown in Figure 3(b). The search area is $180\,\mathrm{m}^2$, the path length is 28.38 m, and the time is 0.050 s.

The experimental image of the improved A^* algorithm using Manhattan distance and Euclidean distance is shown in Figure 3(c). To distinguish, it is named the modified A^* algorithm. The search area is $166\,\mathrm{m}^2$, the path length is 28.38 m, and the time is 0.045 s.

The detailed data are shown in Table 1.

From the above data, compared with the traditional A^* algorithm, the improved A^* algorithm in this paper can reduce the search area by 28.9% and increase the efficiency by 20.0%. Compared with the modified A^* algorithm, the improved A^* algorithm in this paper can reduce the search area by 22.9% and increase the efficiency by 11.1%.

4.2. Static Global Path Planning. In a static simulation environment, the simulation experiments results based on different algorithms will be compared in this section. The simulation experiments in this paper are in the same environment, the maximum speed and acceleration of the intelligent vehicle are the same, and the red line is the actual trajectory of the intelligent vehicle.



(a)

(b)

(c)

Figure 3: Path planning results of various algorithms. (a) The improved A^* algorithm. (b) Traditional A^* algorithm. (c) Modified A^* algorithm.

Table 1: Statistics 1.

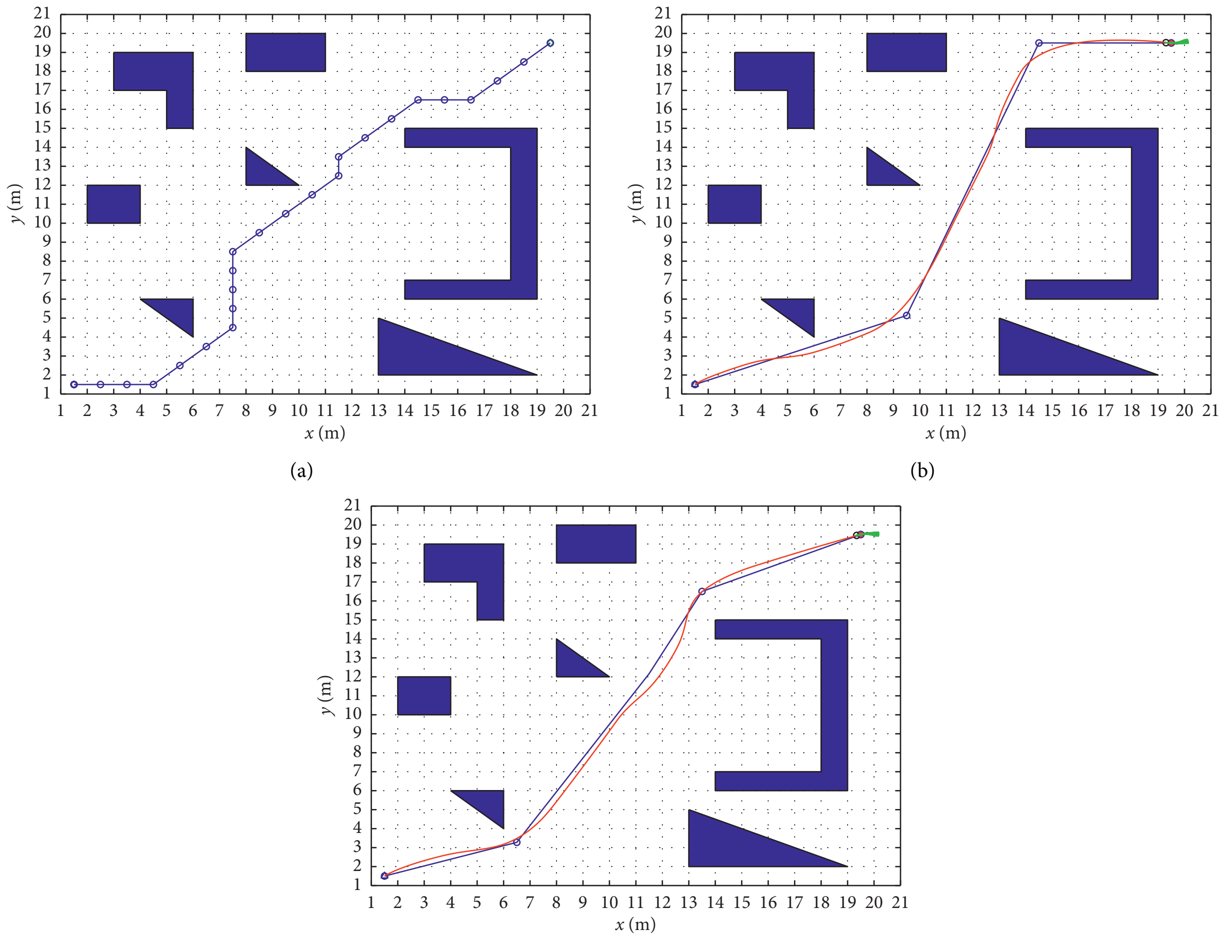
Name	The improved A^* algorithm	Traditional A^* algorithm	Modified A^* algorithm
Search area (m^2)	128	180	166
Path length (m)	28.38	28.38	28.38
Average time (s)	0.040	0.050	0.045

Working condition 1: the experimental image of the traditional A^* algorithm is shown in Figure 4(a). Each circle in the figure represents a node, and various polygons are obstacle environments. According to the results of simulation experiments, this algorithm has many redundant nodes and turning points, the number of optimal path nodes is 24, the number of turning points is 7, and a total of 180 nodes are expanded. The search path length is 28.38 m, and the time is 0.050 s.

Working condition 2: the experimental image of the modified A^* algorithm is shown in the blue line in Figure 4(b). Each circle in the figure represents a node, and various polygons are obstacle environments. According to the results of simulation experiments, the number of optimal path nodes in this algorithm is 4, the number of turning points is 2, and a total of 166 nodes are expanded. The length of the search path is 27.50 m, and the time is 0.045 s.

Working condition 3: the experimental images of the modified A^* algorithm and the dynamic window method are shown in the red line in Figure 4(b). Each circle in the figure represents a node, and various polygons are obstacle environments. According to the results of the simulation experiment, the trajectory planned by this algorithm is smooth, the optimal path length is 28.56 m, and the time is 51.31 s.

Working condition 4: the experimental image of the improved A^* algorithm in this paper is shown in the blue line in Figure 4(c). Each circle in the figure represents a node, and various polygons are obstacle environments. According to the results of the simulation experiment, the number of optimal path nodes of this



(a)

(b)

(c)

Figure 4: Path planning results of three algorithms. (a) Traditional A^* algorithm. (b) Modified A^* algorithm. (c) Improved A^* algorithm.

algorithm is 4, the number of turning points is 2, and a total of 128 nodes are expanded. The optimal path length of this algorithm is 26.99 m, and the time is 0.04 s.

Working condition 5: the experimental images of the improved A^* algorithm and dynamic window method in this paper are shown in the red line in Figure 4(c). Each circle in the figure represents a node, and various polygons are obstacle environments. According to the results of the simulation experiment, the path planned by this algorithm is smooth, the optimal path length is 27.13 m, and the time is 46.69 s.

Detailed statistics are shown in Tables 2 and 3.

4.3. Dynamic Global Path Planning. In the dynamic simulation environment, this section will compare simulation experiments based on different algorithms. The simulation experiment in this paper is in the same environment, the maximum speed and acceleration of the intelligent vehicle are the same, and the red line is the actual trajectory of the intelligent vehicle.

Working condition 1: the experimental images of the traditional A^* algorithm and dynamic window method are shown in Figure 5. The figure shows the dynamic obstacle avoidance situation at different moments. Various polygons are the obstacle environment. According to the results of simulation experiments, the optimal path length of this algorithm is 29.63 m and the total time is 54.77 s.

Working condition 2: the experimental images of the modified A^* algorithm and dynamic window method are shown in Figure 6. The figure shows the dynamic obstacle avoidance situation at different moments. Various polygons are the obstacle environment. According to the results of simulation experiments, the optimal path length of this algorithm is 29.48 m and the total time is 52.75 s.

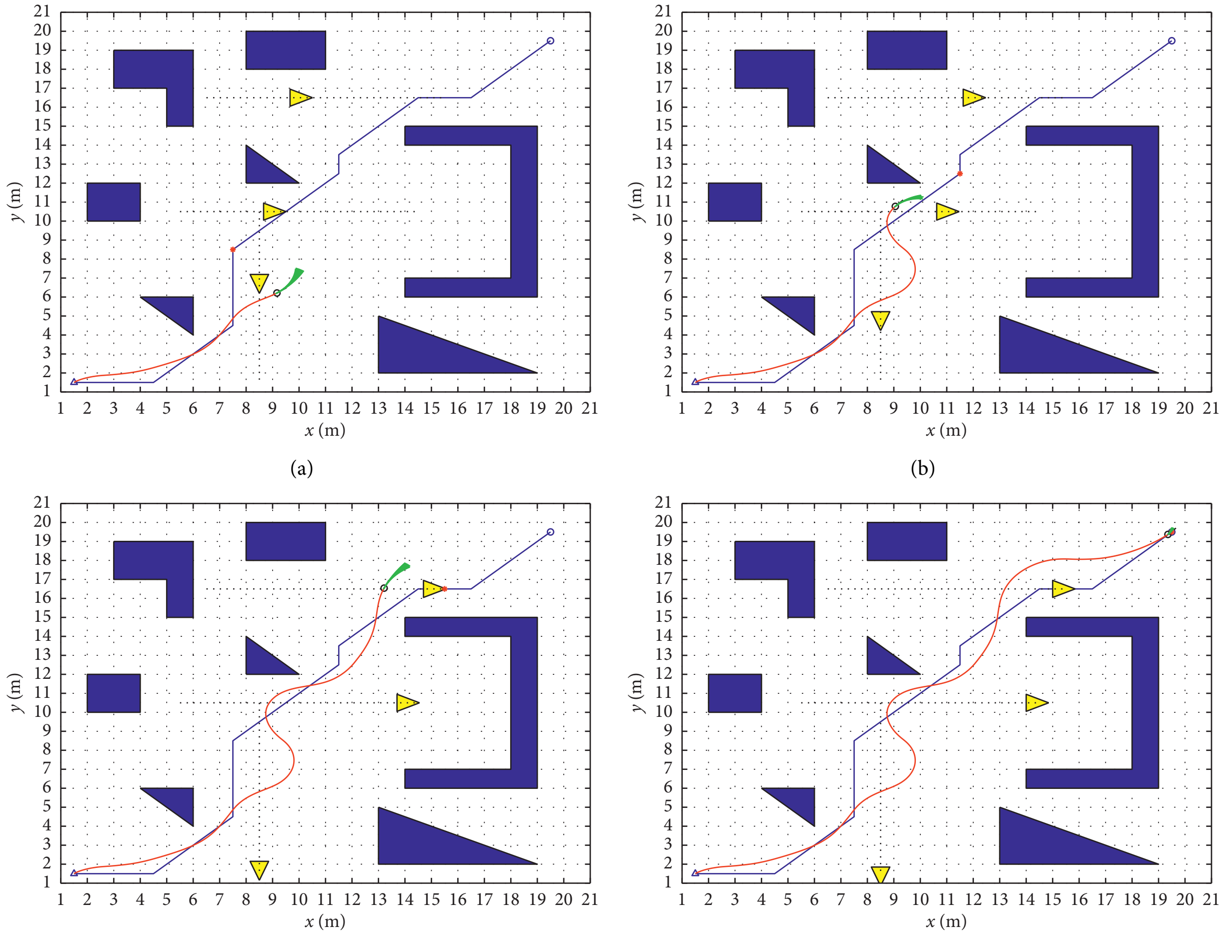
Working condition 3: the experimental images using the improved A^* algorithm and dynamic window method in this article are shown in Figure 7. The figure shows the dynamic obstacle avoidance situation at different moments. Various polygons are the obstacle environment. According to the results of simulation

Table 2: Simulation data statistics 2.

Name	Number of nodes (piece)	Number of break points (piece)	Search node (piece)	Path length (m)	Average time (s)
Working condition 1 (static)	24	7	180	28.38	0.050
Working condition 2 (static)	4	2	166	27.50	0.045
Working condition 4 (static)	4	2	128	26.99	0.040

Table 3: Simulation data statistics 3.

Name	Working condition 3 (static)	Working condition 5 (static)	Working condition 1 (dynamic)	Working condition 2 (dynamic)	Working condition 3 (dynamic)
Path length (m)	28.56	27.13	29.63	29.48	28.74
Average time (s)	51.31	46.69	54.77	52.75	47.30



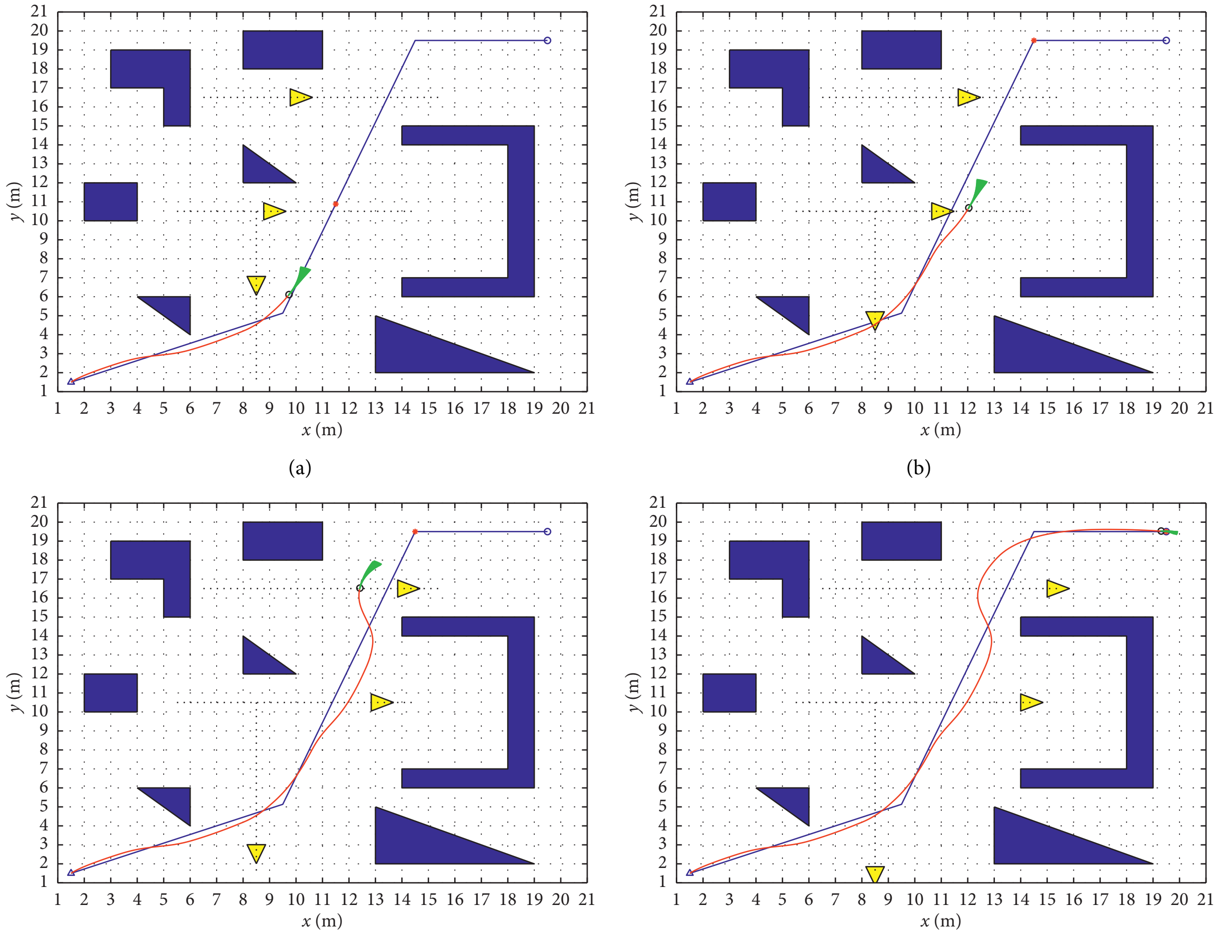
(a)

(b)

(c)

(d)

Figure 5: Trajectory of the traditional A^* algorithm. (a) Traditional A^* algorithm 1. (b) Traditional A^* algorithm 2. (c) Traditional A^* algorithm 3. (d) Traditional A^* algorithm 4.



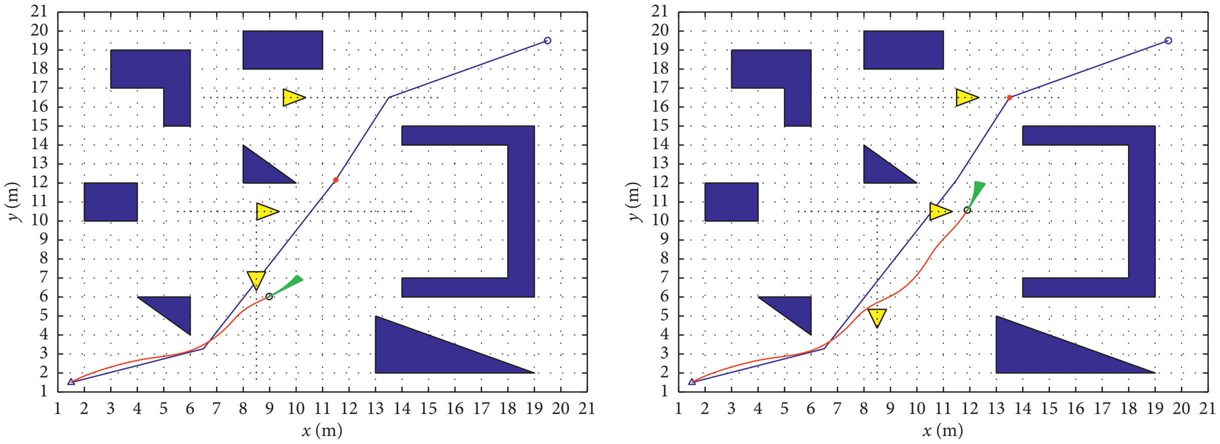
(a)

(b)

(c)

(d)

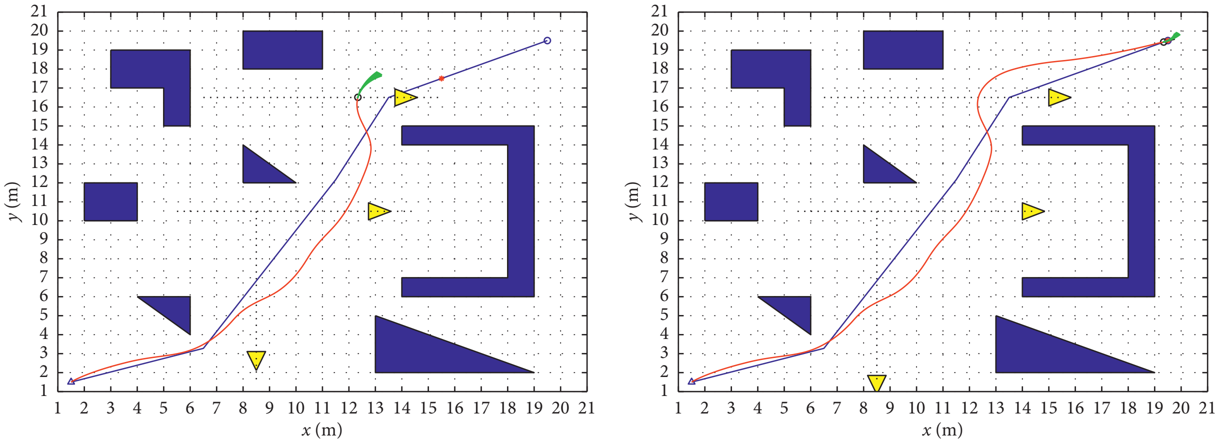
Figure 6: Trajectory of the modified A^* algorithm. (a) Modified A^* algorithm 1. (b) Modified A^* algorithm 2. (c) Modified A^* algorithm 3. (d) Modified A^* algorithm 4.



(a)

(b)

Figure 7: Continued.



(c) (d)

Figure 7: Trajectory of the improved A^* algorithm. (a) Improved A^* algorithm 1. (b) Improved A^* algorithm 2. (c) Improved A^* algorithm 3. (d) Improved A^* algorithm 4.

experiments, the optimal path length of this algorithm is 28.74 m and the total time is 47.30 s.

Detailed statistics are shown in Table 3.

4.4. Simulation Experiment Results and Analysis. In the static obstacle environment, compared with the modified A^* algorithm and the dynamic window algorithm, the improved A^* algorithm and the dynamic window algorithm in this paper reduce the path distance by 5.0% and the time by 9.0%.

In the dynamic obstacle environment, compared with the modified A^* algorithm and dynamic window algorithm, the improved A^* algorithm and dynamic window algorithm in this paper reduce the path distance by 2.5% and the time by 10.3%. Compared with the traditional A^* algorithm and dynamic window algorithm, the path distance of the algorithm proposed in this paper is reduced by 3.0% and the time is reduced by 13.6%

Therefore, the algorithm proposed in this paper is more efficient. The planned path is shorter and smoother, which is conducive to the driving of intelligent vehicle.

5. Real Vehicle Test

This paper uses an unmanned dual-head driving test vehicle based on the Linux system to verify the improved A^* path planning algorithm. The platform supports complete independent development, equipped with 16-line laser radar, millimeter wave radar, binocular vision module, GPS positioning module, and other equipment, with high-precision positioning, automatic navigation, and tracking functions. The actual vehicle is shown in Figure 8. The static obstacle is a cardboard box, and the dynamic obstacle is a tester. To ensure safety, the test vehicle is equipped with a driver responsible for emergency situations, and the maximum speed is set to 30 km/h.

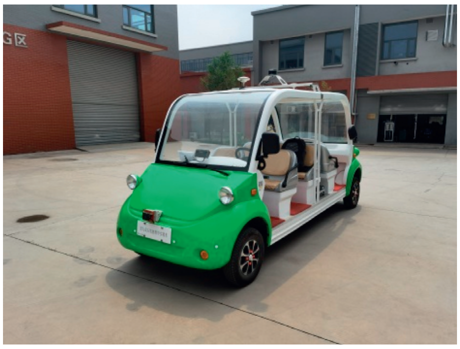


Figure 8: Intelligent driving platform.

Figure 9 is an image displayed by binocular vision, which shows an obstacle environment. Figure 10 is a lidar image, including the location of obstacles. Figure 11 shows the static obstacle avoidance trajectory. The red trajectory is the trajectory of global path planning and local path planning, the yellow is the obstacle, and the blue is the actual trajectory of the experimental vehicle. The static obstacles are cartons with a large width, so the obstacle avoidance range is large. Figures 12 and 13 are dynamic obstacle avoidance trajectories. The red trajectory is the trajectory of global path planning and local path planning, the yellow is the obstacle, the blue is the actual trajectory of the experimental vehicle, and the dynamic obstacle is the experimental personnel. Figures 14–17 are the obstacle avoidance pictures taken during the real vehicle test at high speed.

The real vehicle test shows that the vehicle can avoid dynamic obstacles in real time. The trajectory is smooth. Therefore, the algorithm proposed in this paper can be applied to the practical application of intelligent electric vehicle platform and has practical value of engineering application.

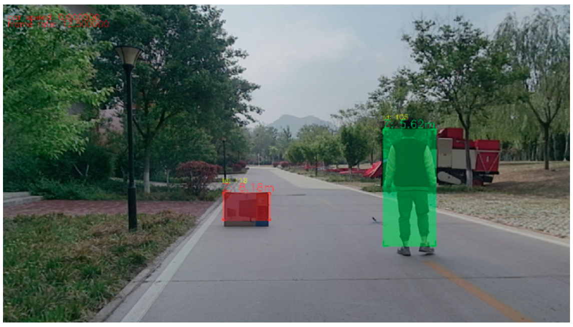


Figure 9: Binocular vision image.

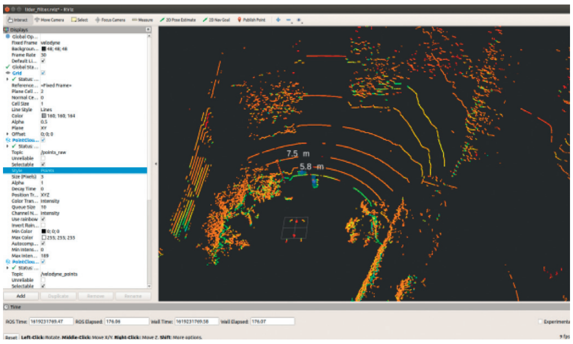
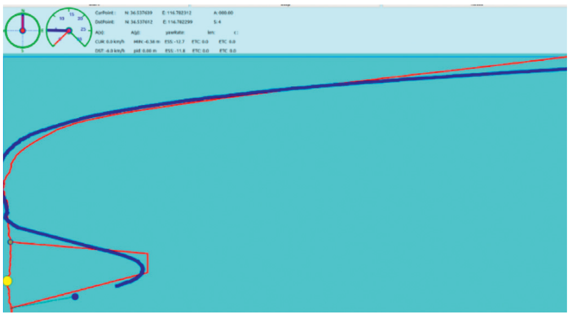
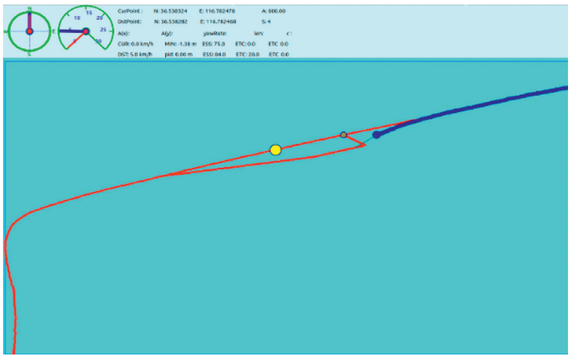


Figure 10: Lidar image.



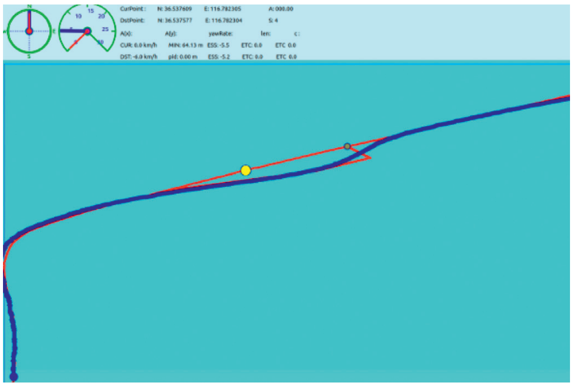
```
CurPoint :   N: 36.537639   E: 116.782312         A: 000.00
DstPoint:    N: 36.537612   E: 116.782299         S: 4
A(x):             A(y):          yawRate:         len:        c:
CUR: 0.0 km/h     MIN: -0.38 m   ESS: -12.7   ETC: 0.0    ETC 0.0
DST: -6.0 km/h    pid: 0.00 m    ESS: -11.8   ETC: 0.0    ETC 0.0
```

Figure 11: Static obstacle avoidance trajectory diagram.



```
CurPoint :   N: 36.538324   E: 116.782478         A: 000.00
DstPoint:    N: 36.538282   E: 116.782468         S: 4
A(x):             A(y):          yawRate:         len:        c:
CUR: 0.0 km/h     MIN: -1.38 m   ESS: 75.0    ETC: 0.0    ETC 0.0
DST: 5.0 km/h     pid: 0.00 m    ESS: 84.0    ETC: 28.0   ETC 0.0
```

Figure 12: Dynamic obstacle avoidance trajectory 1.



```
CurPoint :   N: 36.537609   E: 116.782305         A: 000.00
DstPoint:    N: 36.537577   E: 116.782304         S: 4
A(x):             A(y):          yawRate:         len:        c:
CUR: 0.0 km/h     MIN: 64.13 m   ESS: -5.5    ETC: 0.0    ETC 0.0
DST: -6.0 km/h    pid: 0.00 m    ESS: -5.2    ETC: 0.0    ETC 0.0
```

Figure 13: Dynamic obstacle avoidance trajectory 2.

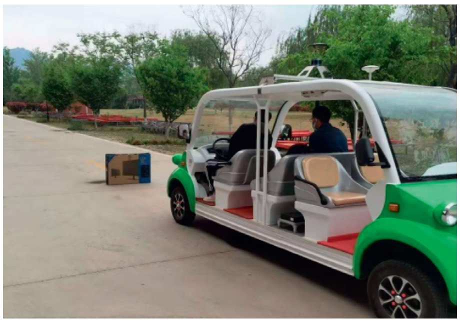


Figure 14: Static obstacle avoidance diagram.

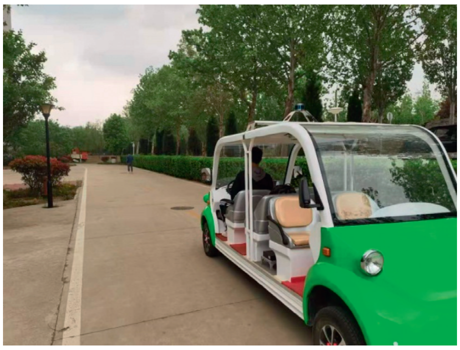


Figure 15: Dynamic obstacle avoidance 1.

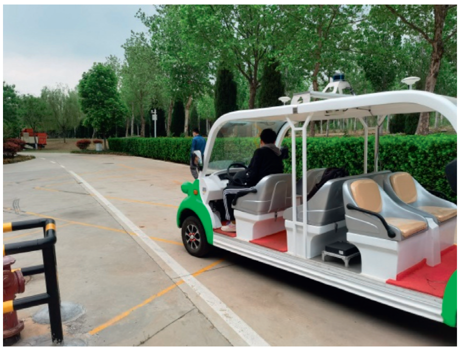


Figure 16: Dynamic obstacle avoidance 2.

Figure 17: Dynamic obstacle avoidance 3.

6. Conclusions

In this paper, a global dynamic path planning method based on the improved A^* algorithm is proposed to deal with the optimal path and planning efficiency of global path planning for intelligent driving. This method has high efficiency and smoother path planning, and the real-time performance of dynamic obstacle avoidance is good. The specific contents of this article are summarized as follows:

(1) The heuristic function $h(n)$ of traditional A^* algorithm is improved, and the dynamic weighting method is used to make $h(n)$ closer to the actual distance $d(n)$. This method reduces the search area in the A^* path search, reduces the planning time, and improves the efficiency of the algorithm.

(2) Use optimization strategies to optimize the optimal path, reduce redundant nodes and turning points of the optimal path, and make the path smoother, which is conducive to the driving of intelligent vehicle. The improved A^* algorithm is combined with the dynamic window method for dynamic obstacle avoidance. This not only ensures that the improved A^* algorithm can efficiently plans the optimal path but also improves the local optimal problem of dynamic window method so that the intelligent vehicle has global dynamic path planning capabilities.

(3) In the simulation analysis, the method in this paper shortens the path distance by 2.5%~3.0%, increases the efficiency by 10.3%~13.6%, and generates a smoother path. Through the actual vehicle test, the results show that the algorithm proposed in this paper has good real-time performance and good stability for dynamic obstacle avoidance. The improved A^* method can be applied in practice on the intelligent electric vehicle platform, and it has a practical value in engineering application.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Authors' Contributions

Chuanhu Niu was responsible for methodology and prepared the original draft; Aijuan Li and Chuanyan Xu were responsible for software and validated the data; Xin Huang and Wei Li reviewed and edited the manuscript. All authors have read and agreed to the published version of the manuscript.

Acknowledgments

This project was supported by the National Natural Science Foundation of China (Grant nos. 51505258, 61601265, and 51405272), Natural Science Foundation of Shandong Province, China (Grant nos. ZR2015EL019 and ZR2020ME126), The Youth Science and Technology Plan Project of Colleges and Universities in Shandong Province (Grant no. 2019KJB019), Open Project of State Key Laboratory of Mechanical Behavior and System Safety of Traffic Engineering Structures, China (Grant no. 1903), and Open Project of Hebei Traffic Safety and Control Key Laboratory, China (Grant no. JTKY2019002).

References

[1] X. Jin, Z. Yan, G. Yin et al., "An adaptive motion planning technique for on-road autonomous driving," *IEEE Access*, vol. 14, pp. 2169–3536, 2020.

[2] B. Pradhan, A. Nandi, N. B. Hui, D. S. Roy, and J. J. P. C. Rodrigues, "A novel hybrid neural network-based multirobot path planning with motion coordination," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 1319–1327, 2020.

[3] M. Chen and D. Zhu, "Optimal time-consuming path planning for autonomous underwater vehicles based on a dynamic neural network model in ocean current environments," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 14401–14412, 2020.

[4] S.-O. Park, M. C. Lee, and J. Kim, "Trajectory planning with collision avoidance for redundant robots using jacobian and artificial potential field-based real-time inverse kinematics," *International Journal of Control, Automation and Systems*, vol. 18, no. 8, pp. 2095–2107, 2020.

[5] B. Hu, Z. Cao, and M. Zhou, "An Efficient rrt-based framework for planning short and smooth wheeled robot motion under kinodynamic constraints," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 4, pp. 3292–3302, 2021.

[6] C. Wu, S. Zhou, and L. Xiao, "Dynamic path planning based on improved ant colony algorithm in traffic congestion," *IEEE Access*, vol. 8, pp. 180773–180783, 2020.

[7] C. Liu, Q. Mao, X. Chu et al., "An improved A-Star algorithm considering water current, traffic separation and berthing for vessel path planning," *Applied Sciences*, vol. 9, no. 6, pp. 1–17, 2019.

[8] S. Erke, D. Bin, N. Yiming et al., "An improved A-star based path planning algorithm for autonomous land vehicles,"

International Journal of Advanced Robotic Systems, vol. 17, no. 5, pp. 1–13, 2020.

[9] Z. Zhang, Y. Wan, Y. Wang et al., "Improved hybrid A * path planning method for spherical mobile robot based on pendulum," *International Journal of Advanced Robotic Systems*, vol. 18, no. 1, pp. 1–14, 2021.

[10] B. Tang, K. Hirota, K. Hirota, X. Wu, Y. Dai, and Z. Jia, "Path planning based on improved hybrid A * algorithm," *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol. 25, no. 1, pp. 64–72, 2021.

[11] B. K. Oleiwi, H. Roth, and B. I. Kazem, "A hybrid approach based on ACO and GA for multi objective mobile robot path planning," *Applied Mechanics and Materials*, vol. 527, pp. 203–212, 2014.

[12] J. Wang and Z. Chen, "A novel hybrid map based global path planning method," in *Proceedings of the 2018 3rd Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)*, pp. 66–70, Singapore, July 2018.

[13] X. Song, S. Gao, C. B. Chen, K. Cao, and J. Huang, "A new hybrid method in global dynamic path planning of mobile robot," *International Journal of Computers Communications & Control*, vol. 13, no. 6, pp. 1032–1046, 2018.

[14] A. V. Le, V. Prabakaran, V. Sivanantham, and R. E. Mohan, "Modified A-star algorithm for efficient coverage path planning in tetris inspired self-reconfigurable robot with integrated laser sensor," *Sensors*, vol. 18, no. 8, pp. 1–27, 2018.

[15] T. J. Chen, Y. Sun, W. Dai, W. Tao, and S. Liu, "On the shortest and conflict-free path planning of Multi-AGV system based on dijkstra algorithm and the dynamic time-window method," *Advanced Materials Research*, vol. 645, pp. 267–271, 2013.

[16] C. Juan, E. Leal, E. F. P. González et al., "The exact euclidean distance transform: a new algorithm for universal path planning," *International Journal of Advanced Robotic Systems*, vol. 10, no. 6, pp. 1–10, 2013.

[17] W. Liu and W. Zhang, "A quantum protocol for secure manhattan distance computation," *IEEE Access*, vol. 8, pp. 16456–16461, 2020.

[18] H. Ren, S. Chen, L. Yang, and Y. Zhao, "Optimal path planning and speed control integration strategy for UGVs in static and dynamic environments," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 10, pp. 10619–10629, 2020.

[19] L. Chang, L. Shan, C. Jiang, and Y. Dai, "Reinforcement based mobile robot path planning with improved dynamic window approach in unknown environment," *Autonomous Robots*, vol. 45, no. 1, pp. 51–76, 2020.

[20] X. Jin, J. Yang, Y. Li, B. Zhu, and G. Yin, "Online estimation of inertial parameter for lightweight electric vehicle using dual unscented kalman filter approach," *IET Intelligent Transport Systems*, vol. 14, no. 5, 2020.

[21] L. Jiang, H. Huang, and Z. Ding, "Path planning for intelligent robots based on deep Q-learning with experience replay and heuristic knowledge," *IEEE/CAA Journal of Automatica Sinica*, vol. 7, no. 4, pp. 1179–1189, 2020.

[22] X. Jin, J. Wang, S. Sun, S. Li, J. Yang, and Z. Yan, "Design of constrained robust controller for active suspension of in-wheel-drive electric vehicles," *Mathematics*, vol. 9, no. 3, pp. 2227–7390, 2021, https://www.mdpi.com/.