Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

An Improved Local Dynamic Path Planning Algorithm for Autonomous Driving

Huakang Chen[1], Pengju Chen[2], Haiying Liu[1,✉], *Member, IEEE,* Jason Gu[3], *Member, IEEE,*
Lixia Deng[1], Juanting Zhou[1], Huiyuan Zhou[1]
[1]*School of Electrical Engineering and Automation, Qilu University of Technology, Jinan, Shandong Province, 250353, China*
[2]*Xiamen ABB Switchgear Co., LTD, Xiamen, Fujian Province, 361101, China*
[3]*Electrical and Computer Engineering, Dalhousie University, Nova Scotia, Halifax, Canada*
1175609983@ qq. com

***Abstract*— In this paper, an improved autonomous driving local dynamic path planning algorithm is proposed. Based on the predefined road center points, a set of path control points is constructed, and a one-dimensional cubic equation is used to fit the path and construct a center line. A new curved coordinate system is provided using the center line, and the path candidates are generated by arc length and lateral offset. The overall path is selected in consideration of the total cost of path safety and comfort. The results showed that under different scenarios, the proposed local path planning algorithm can plan an optimal path that does not collide with static obstacles, and can ensure the comfort of autonomous driving vehicles and the real-time path planning.**

***Index Terms*— Autonomous driving, real-time path planning, obstacle avoidance, minimum cost function.**

I. Introduction

The rise of intelligent transportation systems has brought about tremendous changes in the development of the automotive industry. The core of the development of intelligent transportation systems is autonomous vehicles. Autonomous driving research includes environmental awareness, navigation positioning, path planning, and decision control. Path planning is a bridge for autonomous driving vehicle information perception and intelligent control, and is the basis for autonomous driving.

The action planning of autonomously driving a car can be regarded as a special scene of ordinary robot motion planning. The vehicle trajectory is attached to a two-dimensional plane and is used to sense the surrounding environment of the vehicle through various sensors. According to the obtained road, vehicle position and obstacle information, the steering and speed of the vehicle are controlled, and finally the safe and reliable autonomous driving function is realized. The physical model of its trajectory is simpler than the trajectory of the normal robot pose.

*This work is supported by NSFC #61601256, International cooperation project #QLUTGJHZ2018019, Shandong Provincial Natural Science Foundation #ZR2018QF005 and Key R&D plan of Shandong Province #2019GGX104079.

The traditional path planning is mature, the algorithm principle is simple and easy to implement, and it is widely used. The path planning algorithm based on graph search and the path planning algorithm based on sampling are two kinds of traditional path planning algorithms which are well applied in autonomous driving vehicles.

The path planning algorithm based on graph search is represented by A* algorithm[1]. The A* algorithm is improved on the Dijkstra algorithm. While the heuristic search improves the efficiency of the algorithm, the evaluation function can guarantee to find an optimal path. The basic formula is as follows: $f(n) = g(n) + h(n)$. Where: $f(n)$ represents the estimated cost of the initial state to the target state; $g(n)$ represents the actual cost from the starting point to the arbitrary vertex n; $h(n)$ represents the estimated cost of any vertex n to the target vertex. Because of its high efficiency, A* algorithm is widely used in path finding and graph traversal, which can effectively solve the shortest path problem. However, it is not suitable for high-speed driving due to the problems such as easy to fall into a dead cycle, many planning path break points and poor planning effect in dynamic environment[2].

The sampling-based path planning algorithm, represented by the rapidly exploring random tree (RRT), plays an important role in the field of robot planning. The RRT algorithm has probability completeness. As the number of sampling points increases, the search tree will eventually fill the free space of the entire state space and find a feasible solution. The algorithm does not require precise connection between states, and is suitable for solving the path planning problem of autonomous vehicles in complex environments. The RRT algorithm performs uniform random sampling on the state space. This blind random extended sampling is accompanied by a large number of collision detection, which seriously affects the efficiency of the algorithm [3].

In this paper, an improved autonomous driving local dynamic path planning algorithm is proposed for autonomous driving path planning. The algorithm selects the optimal path from a limited data set and plans the speed at which the

978-1-7281-6321-5/19//$31.00 ©2019 IEEE

vehicle is traveling. This paper assumes that a precise path point is omitted to omit the global path plan. Based on the Frenet coordinate system, a one-dimensional cubic equation is used to fit the path to construct a center line. The centerline provides a new curvilinear coordinate system that generates path candidates by arc length and lateral offset. Design a new security cost function to choose the optimal path.

The rest of this paper is structured as follows. The basic curve coordinate system for path planning is described in Section II. An improved local dynamic path planning algorithm is described in Section III. The simulation results are described in Section IV. Finally, conclusions are drawn in Section V.

II. Basic coordinate system

As shown in Fig. 1(b), based on the Frenet coordinate system, we use the center line of the road as the reference line and take the vehicle itself as the origin. The coordinate axes are perpendicular to each other and are divided into longitudinal s and transverse d. Compared to the Cartesian coordinate system, the Frenet coordinate system significantly simplifies the problem, because on the road, we can always find the centerline of the road [4]. The position can be described simply by using the longitudinal distance (distance along the road direction) and the lateral distance (distance from the reference line). Similarly, the calculation of the speeds s' and d' in both directions is relatively simple.

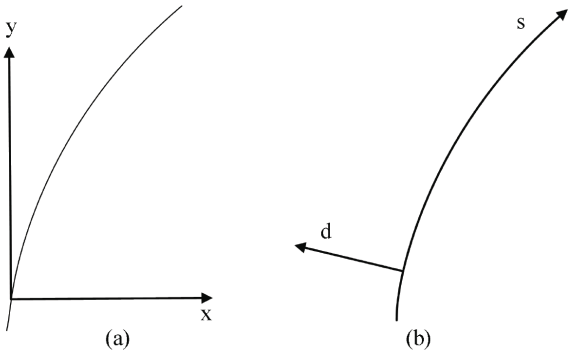

Fig. 1. (a) Cartesian coordinate system, (b) Frenet coordinate system.

In this paper, the global path planning result is used to intercept the center line on the global path in real time as the reference line of the current autonomous driving vehicle (the center line is parallel to the lane line), and the center line is the ideal trajectory of the current time. Considering the continuity of the second derivative of the curve, fitting the centerline using a one-dimensional cubic equation can meet the requirements [5,6]. The centerline equation is described as (1).

$$\begin{cases} x_0(s) = a_x s^3 + b_x s^3 + c_x s^3 + d_x \\ y_0(s) = a_y s^3 + b_y s^3 + c_y s^3 + d_y \end{cases} \tag{1}$$

Calculate the derivative of the cubic spline curve in (1).

$$\begin{cases} x_0' = \frac{dx_0}{ds} = cos\theta_0 \\ y_0' = \frac{dy_0}{ds} = sin\theta_0 \\ x_0'' = \frac{d^2x_0}{ds^2} \\ y_0'' = \frac{d^2y_0}{ds^2} \end{cases} \tag{2}$$

The coordinates, heading and curvature of any point on the center line are represented by (3) and (4).

$$\theta_0 = arctan\frac{dy_0}{dx_0} \tag{3}$$

$$\kappa_0 = \frac{x_0' y_0'' - x_0'' y_0'}{\sqrt{(x_0' + y_0')^3}} \tag{4}$$

In order to use the heading and curvature of the centerline in the curvilinear coordinate system during vehicle travel, the relative position of the vehicle to the centerline needs to be calculated. As shown in Fig.2.(b), the vehicle position information is mapped from the Cartesian coordinate system to the curvilinear coordinate system, and the closest point d_0 of the center line to the vehicle and the shortest distance d are determined.

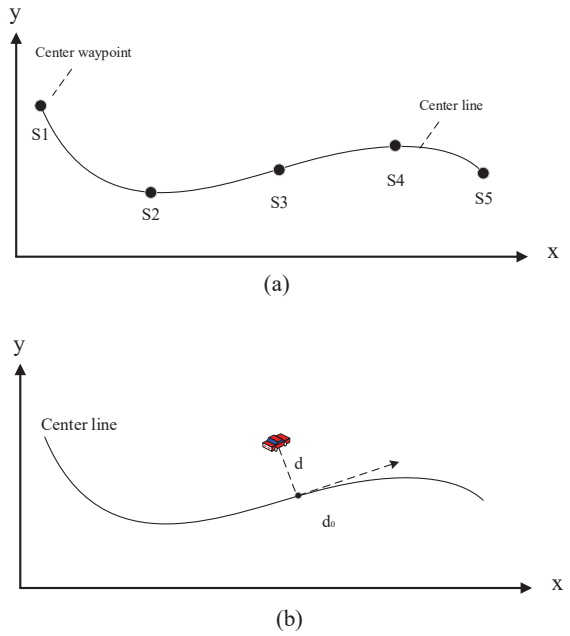

Fig. 2. (a) Center waypoints and center line segments, (b) Localization on the center line.

III. Improved Local Dynamic Path Planning Algorithm

A. Path Candidate Generation

In order to generate path candidates, the range of values of the target configuration is constrained by definition (d_{min}, d_{max}), and the sampling density is limited by Δd,

thereby obtaining a finite set of alternative trajectories. The candidate path covers the entire road with different lateral offsets, while the position information of the obstacle can be marked with a lateral offset in the curvilinear coordinate system. A function describing the lateral offset d with respect to the arc length s is designed.

$$d(s)=\begin{cases}\alpha_0\Delta s^3+\alpha_1\Delta s^2+\alpha_2\Delta s+d_{start}, s\in[s_{start},s_{end}]\\ d_{end}\quad others\end{cases} \tag{5}$$

where$\Delta s=s-s_{start}$.

The initial position information of the vehicle can obtain an initial arc length s_{start} and a lateral offset d_{start}, and the relative angle $\Delta\theta_{start}$ represents the difference between the vehicle heading angle and the tangential angle of the current position center line. The initial configuration can be described as (6). If the lateral offset d_{end} of the target configuration is set, the coefficients α_0, α_1 and α_2 in (5) can be calculated. Thereby obtaining a limited set of candidate trajectories.

$$\begin{cases}d(s_{start})=d_{start}\\ d(s_{end})=d_{end}\\ d'(s_{start})=tan\Delta\theta_{start}\\ d'(s_{end})=0\end{cases} \tag{6}$$

B. Coordinate Transformation

In order to pass the path planning result to the maneuver system, the coordinate points in the curve coordinate system are converted into coordinate points in the Cartesian coordinate system [7]. The attitude vector $(x, y, \theta, \kappa, v)$ of the vehicle is given, where (x, y) represents the position of the vehicle in a two-dimensional plane, θ represents the orientation of the vehicle, κ represents the curvature (rate of change toward θ), and v represents the speed of the vehicle (tangential speed at any point of the track). The scalar size of these attitude variables of the vehicle satisfies the following relationship:

$$x'=vcos\theta,\quad y'=vsin\theta,\quad \theta'=v\kappa \tag{7}$$

For the conversion between the Cartesian coordinate system and the curvilinear coordinate system, the curvature k of the path can be calculated as (8).

$$\kappa=\frac{A}{B}\left[\kappa_0+\frac{(1-d\kappa_0)(d'')+\kappa_0(d')^2}{B^2}\right] \tag{8}$$

In (8), A and B are defined as :

$$A=sgn(1-d\kappa_0),\quad B=\sqrt{(d')^2+(1-d\kappa_0)^2} \tag{9}$$

To ensure that the generated path is feasible, the relationship between the lateral offset d and the radius of curvature of the centerline is calculated. When $d=1/\kappa_0$, $A=0$, path candidate curvature $\kappa=0$, this path passes the centerline curvature center; When $d>1/\kappa_0$, $A=-1$, the path candidate curvature $\kappa<0$, and the curvature k is opposite to the direction of the generated path, the candidate path will be deleted; when $d<1/\kappa_0$, $A=1$, the path candidate curvature $\kappa>0$, the path is feasible.

In order to ensure that the planned candidate path satisfies the minimum turning radius constraint of the autonomous vehicle, the path is required to be satisfied (10).

$$|\kappa_0|\le\kappa_{max} \tag{10}$$

Where κ_{max} is the maximum turning rate of autonomously driven vehicles.

The arc length s of the base coordinate system (i.e., the driving distance on the global path) becomes the horizontal axis of the curvilinear coordinate system, and the lateral offset d becomes the vertical axis. The systematic relationship of the trajectory with respect to the vehicle attitude is given by the following partial differential equation:

$$\frac{dx}{ds}=Bcos\theta,\quad \frac{dy}{ds}=Bsin\theta,\quad \frac{d\theta}{ds}=B\kappa \tag{11}$$

C. Path Security Cost Function

Static safety cost functions need to take into account static obstacles, lane lines, and road edge problems. The key to local path planning is that the vehicle cannot collide with obstacles, so how to indicate the degree of danger of obstacles is especially critical. The closer the obstacle is to the trajectory, the more likely the unmanned vehicle collides with the obstacle. The calculation model of the collision numerical value established in this paper is (12).

$$f_{obs}(r)=kv(\frac{1}{\rho(d,d_c)}-\frac{1}{d_c})\quad d_c\le d\le d_n \tag{12}$$

Where $\rho(d,d_c)=\sqrt{(d-d_c)^2}$, d_n is the safety distance threshold, and d_c is the collision distance threshold. When $d>d_n$, it is considered that the obstacle is not dangerous to the current trajectory; when $d<d_c$, it is considered that the driverless car must collide with the obstacle; $\rho(d,d_c)$ is the difference in distance between the distance from the obstacle to the trajectory s and the collision distance threshold. When evaluating the degree of danger of obstacles, the value is $dn=1.5Lw$, $dc=0.5Lw$, and Lw is the single-lane vehicle width.

For multi-lane roads, when calculating the obstacle hazard level, the roadway and adjacent lane road traffic participants have a greater influence on the normal driving of the vehicle, and the obstacle coordinate position is quickly determined by the horizontal axis coordinate value d in the curvilinear coordinate system. During normal driving, the vehicle is encouraged to continue driving in the current lane, minimizing lane changes. Unless an accident occurs, the vehicle cannot travel in the opposite lane. Therefore, we set a higher collision coefficient for the vehicle passing through the other lane ($k=1.5$); we set a lower collision coefficient for the vehicle that changed lanes ($k=1.2$); We set the collision coefficient ($k=1$) for the vehicle traveling in the current

lane. kv is the influence factor for calculating the degree of danger of obstacles [8]. The higher the speed, the greater the danger of the same obstacle to the vehicle.

D. Comfort Cost Function

The comfort cost function and the constraints on vehicle handling are closely related to the comfort of the passenger [9]. In general, too much path curvature can cause discomfort for passengers. Therefore, in terms of ride comfort, the amount of path curvature should be optimized, and the cost function of path smoothness is described as (13).

$$f_{smoth}(i) = (\omega_1 + \omega_2) \int \kappa_i(s)^2 ds \tag{13}$$

Where

$$\omega_1 = k_1 \Delta\beta, \quad \omega_2 = k_2 v \tag{14}$$

In (14): $\Delta\beta$ is the angle between the direction of the trajectory s at the origin and the heading of the car. When $\Delta\beta$ is large, the car needs to quickly adjust its direction. Therefore, the larger $\Delta\beta$ is, the more it affects $f_{smoth}(i)$; when the speed is higher, the car has higher requirements on the smoothness of the trajectory, and the directional change frequency of the trajectory should be lower. In order to improve the influence of trajectory smoothness on the comfort cost function, ω_2 increases rapidly with the increase of the vehicle speed, so that $f_{smoth}(i)$ increases when the vehicle speed is high.

In the process of partial path planning of autonomously driving vehicles, if the difference between the current generated path and the previous path is too large, the steering wheel angle will be changed too much and severe jitter will occur. In order to avoid this situation, the trajectory planned at the time before and after the time should ensure that the slope of the origin of the local coordinate system does not change much. The path consistency cost function is described as (15).

$$f_{change}(i) = \Delta\beta_t - \Delta\beta_{t-1} \tag{15}$$

Where: $\Delta\beta_{t-1}$ is the slope of the trajectory s at the origin of $t-1$; $\Delta\beta_t$ is the slope of the trajectory s at the origin of time t.

Equation (16) represents the degree of deviation of evaluating a trajectory to a reference trajectory. The larger the f_{dev} indicates the farther the trajectory deviates from the centerline. We hope that the autonomous vehicle can continue to travel along the centerline after completing the obstacle avoidance, and the target state should not be too far from the center of the road [10].

$$f_{dev}(i) = d_i^2 \tag{16}$$

The comfort cost function f_{com} is defined as the sum of smoothness, consistency, and degree of deviation, which is described as (17).

$$f_{com}(i) = a f_{smoth}(i) + b f_{change}(i) + c f_{dev}(i) \tag{17}$$

Where a, b, c are the coefficients of the three loss functions.

E. Path Selection Total Cost Function

The total cost function shown in (18) is defined as the weighted sum of the two cost functions.

$$f_{total}(i) = \omega_{obs} f_{obs}(i) + \omega_{com} f_{com}(i) \tag{18}$$

Where ω_{obs} and ω_{com} are the weight of two cost functions, the ratio of these two weights is determined by the driving style, the ratio determines the optimization of which aspect of our cost function. In the simulation of this paper, ω_{obs} and ω_{com} are set to 0.6 and 0.4 respectively. The optimal path can be obtained by finding the minimum of all candidate trajectory cost functions.

IV. Simulation results

In order to verify the local dynamic path planning algorithm proposed in this paper, the center line and obstacles to be tracked are first generated, and the road center line is described by cubic splines. In the simulation, the red line is the road center line, the blue point is the obstacle, the green point is the optimized trajectory, and the blue triangle is the planning vehicle.

Fig.3 shows the results of the algorithm in the straight road test. Fig.3.(a) shows the road centerline and static obstacles, and Fig.3.(b) shows the vehicle turning right to avoid the first obstacle, then left to avoid the second obstacle. Fig.3.(c) and (d) show that the vehicle returns to the centerline after continuously avoiding the obstacle and continues to travel to the end point.

Fig.4 shows the results of the algorithm in the "S" shaped road test. Fig.4.(a) shows the road center line and the static obstacle, and Fig.4.(c) shows the vehicle turning left to avoid the obstacle, and the vehicle chooses to turn left during the obstacle avoidance because of the degree of deviation. Fig.4.(d) shows the vehicle reaching the end point after turning right to avoid obstacles.

In this paper, A* algorithm and Dijkstra algorithm are used as references to evaluate the cost function of straight and curve scenes. As shown in Table I, the algorithm proposed in this paper has a better minimum cost function when integrating various factors.

The experimental hardware platform of this paper is Intel Core i5-4200H CPU (2.80 GHz) with 4 GB of memory. It can be seen from Fig.3 and Fig.4 that although the number of obstacles is large, the local path planning algorithm proposed in this paper can generate a smooth and safe route for the vehicle to successfully bypass the obstacle. Table II shows the operation time of each stage. The total time of one step is 54ms, which can meet the real-time requirements of the system.

TABLE I

MINIMUM COST FUNCTION CALCULATIONS

Road Type \ Algorithm	A*	Dijkstra	Improve Algorithm
Straight	2634	2502	2284
"S" shaped road	12740	11054	10213

TABLE II

ALGORITHM RUNNING TIME

Stage	Time(ms)
Generate reference path	4
Path generation	46
Path selection	4
Total	54

V. CONCLUSIONS

This paper presents an improved autonomous driving local dynamic path planning algorithm. The algorithm considers the effect of speed on trajectory smoothness and designs a new cost function. The simulation results show that the proposed algorithm can not only meet the real-time requirements, but also ensure the safety and smoothness of driving. The focus of this paper is on the motion planning of autonomously driving vehicles, considering only the avoidance and motion planning of static obstacles. For the complex urban road environment, the local path planning algorithm needs to consider more elements. How to plan a safe path that can be driven will become the research focus. Optimize the calculation of obstacle hazard levels and dynamic obstacle predictions to be strengthened.

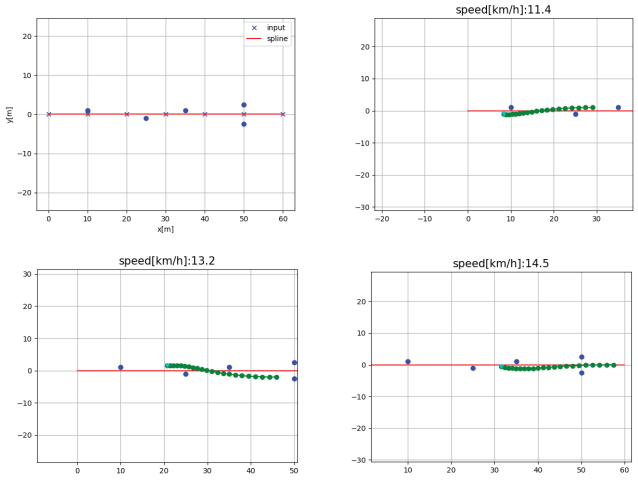

Fig. 3. Straight road

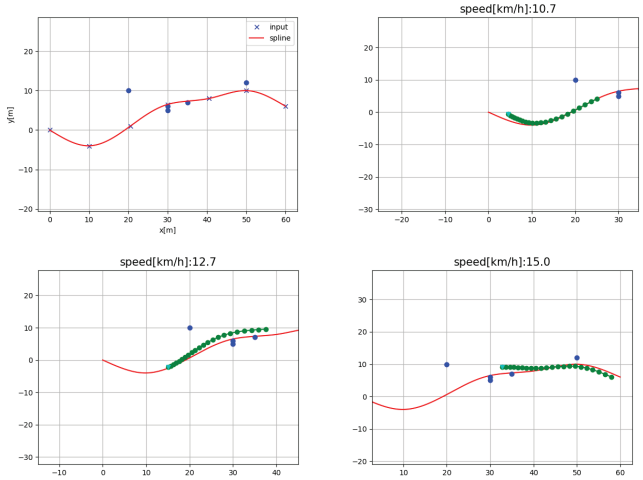

Fig. 4. "S" shaped road

REFERENCES

[1] Z. Liang, M. Lan, and Z. Chen, "Optimization of A* Algorithm in Finding Shortest-Path," Computer Systems and Applications, vol. 27, no. 7, pp. 255-259, 2018.

[2] X. Zhao and G. HU, "Application of Smoothing ARA* Algorithm in Intelligent Vehicles Path Planning," Mechanical Science and Technology for Aerospace Engineering, vol. 36, no. 8, pp. 1272-1275, 2017.

[3] L. Jaillet, J. Cortes, and T. Simeon, "Sampling-Based Path Planning on Configuration-Space Costmaps," IEEE Transactions on Robotics, vol. 26, no. 4, pp. 635-646, 2010.

[4] X. Hu, L. Chen, B. Tang, D. Cao, and H. He, "Dynamic path planning for autonomous driving on various roads with avoidance of static and moving obstacles," Mechanical Systems and Signal Processing, vol. 100, pp. 482-500, 2018.

[5] X. Wu, B. Guo, J. Wang, "Mobile robot path planning algorithm based on particle swarm optimization of cubic splines," Robot, vol. 31, no. 6, pp. 556-560, 2009.

[6] Y. Zhang, Y. Du, F. Pan and Y. Wei, "Intelligent vehicle path tracking algorithm based on cubic B-spline curve fitting," Journal of Computer Applications, vol. 38, no. 6, pp. 1562-1567, 2018.

[7] K. Chu, M. Lee, M. Sunwoo, "Local path planning for Off-Road autonomous driving with avoidance of static obstacles," IEEE transactions on intelligent transportation systems, vol. 13, no. 4, pp. 1599-1616, 2012.

[8] J. Ji, Z. Tang, M. Wu, et al, "Optimization of A* Algorithm in Finding Shortest-Path," China Journal of Highway and Transport, vol. 31, no. 4, pp. 172-179, 2018.

[9] B. Song, Z. Wang, L. Zhao, et al, "A new approach to smooth global path planning of mobile robots with kinematic constraints," International Journal of Machine Learning and Cybernetics, vol. 10, no. 1, pp. 107-119, 2017.

[10] H. Chen, C. Shen, H. Guo and J. Liu, "Moving Horizon Path Planning for Intelligent Vehicle Considering Dynamic Obstacle Avoidance," China Journal of Highway and Transport, vol. 32, no. 01, pp. 162-172, 2019.

[11] C. Simon, F. Philippe, Y. Zhao, et al, "Multilayer path planning control for the simulation of manipulation tasks: involving semantics and topology," Robotics and Computer integrates Manufacturing, vol. 57, pp. 17-58, 2019.

[12] C. Maxime, C. Sebastien, B. Thomas, et al, "Automatic multi-axis path planning for thinwall tubing through robotized wire deposition," Procedia CIRP, vol. 79, pp. 89-94, 2019.

[13] H. Liu, J. Gu, M. Meng, and W. Lu, "Fast Weighted Total Variation

Regularization Algorithm for Blur Identification and Image Restoration, IEEE Access, vol. 4, no. 1, pp. 1-10, 2016.
[14] L. Deng, X. Ma, J. Gu and Y. Li, “Artificial Immune Network-based Multi-Robot Formation Path Planning with Obstacle Avoidance, International Journal of Robotics and Automation, pp. 1-10, 2016.
[15] Z. Huang, Y. Yu, J. Gu, H. Liu, “An Efficient Method for Traffic Sign Recognition based on Extreme Learning Machine, IEEE Transactions on Cybernetics, vol. 47, no. 4, pp. 920-933, 2017.