Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 54-10 (2021) 51–58

Real-time Local Path Planning for Intelligent Vehicle combining Tentacle Algorithm and B-spline Curve

Zhuoren Li*, Lu Xiong *,** Dequan Zeng *,****
Zhiqiang Fu*, Bo Leng*,** Fengwu Shan*,**,*****

** School of Automotive Studies, Tongji University, Shanghai, 201804, China (e-mail: zrli_96@163.com; xiong_lu@tongji.edu.cn; zdq1610849@126.com).*

*** Clean Energy Automotive Engineering Center, Tongji University, Shanghai, 201804, China*

**** New Energy Vehicle Corporation, Jiangxi Jiangling Motors Group Nanchang, China, (e-mail: shanfw@126.com)*

Abstract: The obstacle-avoidance problem of intelligent vehicles is one of the challenges that we face in path planning. In order to tackle it, this paper proposes a real-time path planning approach based on tentacle algorithm and B-spline curve. In this approach, firstly, some virtual tentacles are built to represent the precalculated paths of the ego vehicle at a current speed. Secondly, it selects the best tentacle path among them, which is required to provide a safe driving direction and sampling area for generating B-spline paths. When the vehicle drives along the best tentacle path, the B-spline path is generated according to the sampling area. Finally, the designed path is formed by segments of the best tentacle and a B-spline curve. Compared with other sampling-based path sets approaches, using the proposed approach needs shorter reaction time. Simulation and experiment results verify the real-time performance and effectiveness of the algorithm of local path planning.

Copyright © 2021 The Authors. This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0/)

Keywords: Path planning, Intelligent vehicle, B-spline curves, Sampling-based method.

1. INTRODUCTION

Intelligent vehicle is envisaged as a promising technology, which can ease the increasing traffic pressure, ensure road safety, and reduce the consumption of energy. Although it has been developing continuously thanks to the growing advance of the artificial intelligence, information and communication technology, sensor technique, positioning and navigation, etc. (Gonzalez et al. (2016) and Paden et al. (2016)). As one of the core technologies for autonomous vehicle, path planning is in charge of generating a smooth path to pursue driving safety, comfort, and economy.

Dijkstra algorithm is a classical graph search-based planner with its configuration space approximately discretized as cell-grid space, lattices among others in Li et al. (2009). The subsequent improvements of this algorithm contain A-star (A*) in Kammel et al. (2008) and Likhachev et al. (2009), the dynamic A-star (D*)(Stentz(1994) and Ferguson et al. (2006)), the anytime repairing A-star (ARA*) in Daniel et al. (2010) and the anytime dynamic A-star (AD*) in Likhachev et al. (2008). However, the paths obtained from the algorithms above are curvature discontinuous, and it is difficult for them to be directly followed by vehicles. Taking vehicle dynamics, the presence of the static and dynamic obstacles, and traffic rules into account, the two main categories of path planning are mainly used in recent years: optimization and sampling-based approaches (Katrakazas et al. (2015)).

Geometric Curve optimization and Model Predictive Control (MPC) are commonly used optimization-based approaches. In geometric curve optimization, a single curve path(e.g. splines, Bezier curve, clothoid, polynomials and etc.) optimization is performed in Delsart et al. (2009). However, such method is essentially a kind of high-dimensional optimization problem with multiple nonlinear constraints, which requires a number of resources to find the right solution.

At the same time, the Rapidly-exploring Random Trees (RRT) and its derivative algorithms are representative sampling approaches in Jayasree et al. (2017). These sampling-based approaches could produce a valid path by extending nodes in a continuous space until the target position is reached, but these trajectories are discontinuous and not smooth enough, with a relatively long planning horizon. Taking the continuity of path curvature into consideration, the interpolating curves are widely used in the sampling-based approaches. In Li et al.(2019), the path set are generated through sampling the target set via offsetting along the reference path.

The tentacle algorithm is another commonly used methods which is firstly applied in the DARPA Urban Challenge 2007 (Hundelshausen et al. (2008)). The Tentacles approach imitates the behavior of insects that uses their antennae to detect, and then avoid obstacles. For intelligent vehicles, all the tentacles are generated as candidate paths according to their current speed. Due to high-speed calculation, the

2405-8963 Copyright © 2021 The Authors. This is an open access article under the CC BY-NC-ND license.
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2021.10.140

tentacle algorithm is assumed to be a quick-reactive method in Alia et al. (2015). In Hundelshausen et al. (2008), the shape of tentacles is circular, leading to that the tentacle curvature is not well-suit to the current vehicle steering angle. Clothoids and some other shapes were used to build tentacle paths in Akmandor et al. (2020) and Mouhagir et al. (2020), which consider the current steering angle. However, every tentacle path needs to be planned very frequently, which needs to occupy much computing resource all the time. Besides, the vehicle needs to be constantly switched from an old tentacle path to a new one, which is not conducive to its tracking.

In this paper, we propose a real-time approach for path planning of on-road intelligent vehicles to efficiently generate a drivable path. In order to keep the continuity of curvature, cubic B-spline curve is adopted to generating a smooth path. And it is convenient to adjust parametric basic points of the spline curve for controlling the curvature extremum of kinematic constraints on vehicle (Zeng et al. (2019a)). We use the tentacle algorithm to quickly decide the driving direction of the vehicle, and the best tentacle returning to a drivable area. When the vehicle is driving in the safest direction, sampling based on the drivable area to generate a completely safe, smooth and comfortable path by using the B-spline curves. Compared with other sampling-based path set approaches (Zeng et al. (2019b)), the proposed approach shows better real-time performance, which has a lower computational cost and shorter run-time.

2. TENTACLE ALGORITHM

2.1 Environment Information

A 2D occupancy grid map with 501×151 cells is coined with the size of each cell being 0.1 m×0.1 m. Therefore, the range of the vehicle's known surrounding environment in physical space is 50.1 m×15.1 m. The center coordinate of the vehicle is (400,75). The distance between the center of vehicle and the front end of the grid map is 40 m, and the rear end of the grid map is 10.1 m. The intelligent vehicle receives the information of obstacles and the features of the road from LIDAR. The point cloud data from LIDAR is converted to binary values in the occupancy grid map. If the cell value is 1, it means there is an obstacle, or if it is 0, there is no obstacle.

2.2 Tentacle Generation

On the occupancy grid map, a set of vehicle-centered tentacles are generated as possible paths. Tentacles are generated off-line to ensure the calculating speed of the planning system. In this work, the applicable scenario is obstacle avoidance or lane change of structured roads, thus the initial steering angle of the vehicle is small. In order to make the calculation as small as possible and make the reaction time of obstacle avoidance faster, we choose a circle as the shape of the tentacles. They are used for both perception and motion execution. Speed is divided into 20 ranges, each of which is 2 km/h and the velocity for speed set j is computed by:

$$v_i = v_0 + q \times (v_m - v_0) \tag{1}$$

where v_0=1km/h is the speed of the minimum speed set and v_m=40 km/h is the maximum speed. When $v_i \le v \le v_{i+1}$, speed set $j = i$, $v_j = v_i$, and $q=j/20$ is the speed level factor. The vehicle's speed does not change on each tentacle path. Refer to [15], for each speed set, 81 tentacles are generated, which start from the center of gravity of the vehicle. The radius r_k of the kth tentacle in a set is given by:

$$r_k = \begin{cases} p^k R_j, k = 0,...,39 \\ \infty, k = 40 \\ p^{80-k} R_j, k = 41,...,80 \end{cases} \tag{2}$$

where p = 1.2 is the correction factor and the initial radius R_j of speed set j is:

$$R_j = \frac{l}{2(1-q^{0.9})} \tag{3}$$

where l is the basic length of tentacles, and defined as:

$$l = 10 + 60 * q^2 \tag{4}$$

The real length l_k of each tentacle is given by:

$$\begin{cases} l_k = l + 20\sqrt{\dfrac{k}{40}}, k = 0,...,40 \\ l_k = l + 20\sqrt{\dfrac{80-k}{40}}, k = 41,...,80 \end{cases} \tag{5}$$

In this way, the maximum length of the tentacle will not exceed the limit of the occupancy grid map. Besides, when k=40, both r_k and R_j are equal to ∞, which means that the vehicle can only drive straightly. After the generation of tentacles, they are discretized into path points for subsequent work. The generated tentacles are shown in Fig.1.

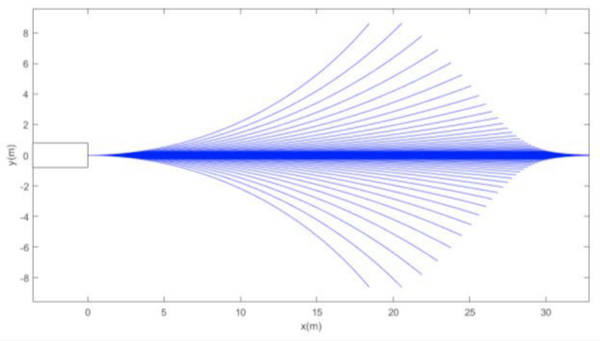

Figure 1. Tentacle Paths for speed set j = 10

2.3 Tentacle Selection

The tentacle algorithm generates and selects the tentacle frequently, which requires much detection and smoothing work. In this work, we only use the selected tentacle to provide the driving direction for obstacle avoidance, and the sampling area for target points of B-spline curve. In order to make the obstacle avoidance reaction as quick as possible, only the safety distance of the tentacles is used as the selection criterion.

The method of collision detection according to the environment map and vehicle configuration is widely used. As is shown in Fig.2, the vehicle configuration is constructed by three circles to detect whether there will be a collision occurring in the grid map.

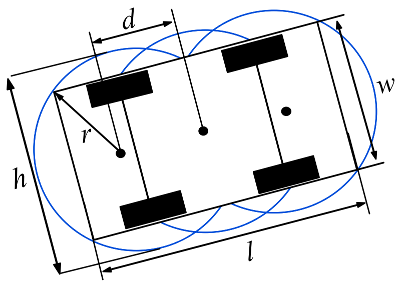

Fig. 2. Vehicle configuration

The size of parameters is designed as:

$$r = \sqrt{\left((l-2d)/2\right)^2 + \left(w/2\right)^2} \tag{6}$$

The value $h = 2r$ is chosen as a safe width and $d = l/3$. In the occupancy grid map, starting from the vehicle, the detection area is generated by expanding with the safe width along the path point on each tentacle. The geometric definition of this area is illustrated in Fig. 3.

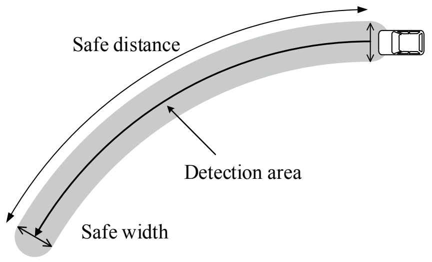

Figure 3. Detection area and safe width.

Once an obstacle is detected on the front reference road, collision detection is performed on the detection area of each tentacle through vehicle configuration, if the grid cell in the vehicle configuration with obstacle (road edge points are also seen as obstacles), the length value of the corresponding tentacle point is recorded as the safe distance l_{k_obs} of this tentacle. The tentacle with the maximum l_{k_obs} is regarded as the best tentacle. In order to reduce the number of collision detections as much as possible, the spatial distance between two adjacent detection points is set to half of the distance from the rear axle to the front of the vehicle. It will be selected for execution and the execution time is t_e second. This time t_e can be set according to the speed set as,

$$t_e = 0.8 + 2 \times \sqrt{q} \tag{7}$$

As the speed increases, the time to travel along the best tentacles will also be appropriately extended to ensure safety. The section of this path is called *executing tentacle*. It is assumed that the speed of vehicle remains unchanged on the executing tentacle path. If the l_{k_obs} of the best tentacle is less than the minimum safe distance, then the vehicle will slow down in the direction of this tentacles and replan at the next moment.

3. B-SPLINE PATH BASED ON THE BEST TENTACLE

After generating and selecting tentacle paths, the vehicle will drive in the direction of the tentacles for t_e seconds. During this period, a real-time path planning approach for on-road intelligent vehicle is used to efficiently generate a drivable path. In order to keep the continuity of curvature, cubic B-spline curve is adopted to generating a smooth path. And it is convenient to adjust parametric basic points of the spline curve for controlling the curvature extremum of kinematic constraints on vehicle.

B-spline curve is a linear combination of the B-spline basis function and control point. For a (n+1) control point $\{P_i\}_{i=0}^{n}$ and parametric nodes $T_{n,k} = \{t_i\}_{i=0}^{n+k}$ $(t_i \le t_{i+1})$, an k degree B-spline curve can be defined as Equation (8) referring to Elbanhawi et al. (2015),

$$P(t) = \sum_{i=0}^{n} P_i N_{i,k}(t), \quad t \in [t_{k-1}, t_{n+1}] \tag{8}$$

where, $N_{i,k}(t)$ is the B-spline basic function, which could be computed using deBox-Cox equations like Equation (9),

$$\begin{cases} N_{i,1}(t) = \begin{cases} 1 & t \in [t_i, t_{i+1}) \\ 0 & t \notin [t_i, t_{i+1}) \end{cases} \\ N_{i,k}(t) = \dfrac{t - t_i}{t_{i+k-1} - t_i} N_{i,k-1}(t) + \dfrac{t_{i+k} - t}{t_{i+k} - t_{i+1}} N_{i+1,k-1}(t), \quad i = 0,1,\dots,n \end{cases} \tag{9}$$

In order to ensure the curvature of the path continuous, at least cubic B-spline curves are selected for the path planning. An example of a cubic B-spline curve with five control points is shown in Fig.4.

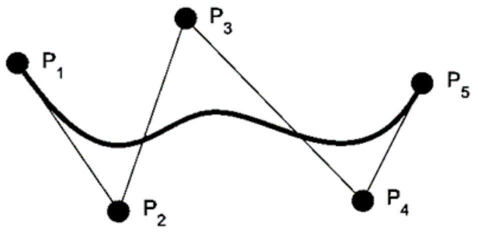

Figure 4. Cubic B-spline curve with five control points

3.1 Control Segment Generation

As shown in Fig. 5, A_1, O and B_1 are the control point of a cubic B-spline curve. A_2 and B_2 are the midpoints of A_1O and OB_1.

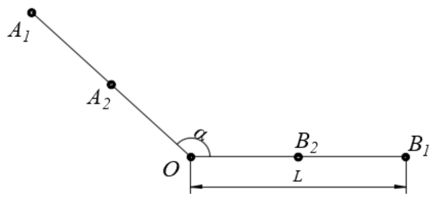

Figure 5. Control segment model

It is assumed that the length of A_1O is equal to the length of OB_1. Then, the control points can be defined as Equation (10) referring to Li et al. (2020),

$$\begin{bmatrix} A_1 \\ A_2 \\ O \\ B_2 \\ B_1 \end{bmatrix} = \begin{bmatrix} L\cos\alpha & L\sin\alpha \\ \frac{L}{2}\cos\alpha & \frac{L}{2}\sin\alpha \\ 0 & 0 \\ \frac{L}{2} & 0 \\ L & 0 \end{bmatrix} \quad (10)$$

If we set the node vector as [0,0,0,0,0.5,1,1,1,1], the expression of the cubic B-spline is respectively described by,

$$\begin{cases} x(t) = L(-t^3 + 3t^2 - 3t + 1 + t^3\cos\alpha) \\ y(t) = Lt^3\sin\alpha \end{cases} \quad (11)$$

and the curvature of the curve is,

$$\kappa = \frac{x'(t)y''(t) - y'(t)x''(t)}{\left(x'(t)^2 + y'(t)^2\right)^{\frac{3}{2}}} \quad (12)$$

According to the equation (11) and (12), the curvature expression can be derived as,

$$\kappa(t) = \frac{2(t\sin\alpha)(1-t)}{3L\left(2t^2(1-\cos\alpha)(t^2-2t+1) + (2t-1)^2\right)^{\frac{3}{2}}} \quad (13)$$

When $\frac{\partial\kappa}{\partial t} = 0$, then the t=0.5 and the curvature has a maximum value. The length L of adjacent vertices and the angle α between them needs to be satisfied as,

$$L \ge L_{\min} = \frac{1}{6}\frac{\sin\alpha}{\kappa_{max}}\left(\frac{1-\cos\alpha}{8}\right)^{-1.5} \quad (14)$$

The length of control segment could be used to form the object function of path planning. By selecting the length and the included angle of the control line segment, a B-spline curve for obstacle avoidance is generated. For the purpose of better integrating with the previous path generated by the tentacle algorithm, we choose 2 control segments to generate subsequent paths. As shown in Fig. 6, X_{start} is the starting point of the executing tentacle path, X_0 is the end point. X_0, X_1, X_2 and X_3 are four control points for B-spline curve, while X_0 is the start point and X_3 is the end.

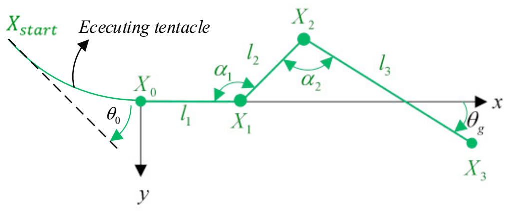

Figure 6. Control path segment

Where $(\alpha_1, \alpha_2) \in [\frac{95}{180}\pi, \frac{175}{180}\pi]$, $\alpha_2 = \alpha_1 - \theta_g$ and θ_g is the heading angle of the target point, $\alpha_1 \in [\frac{95}{180}\pi + \theta_g, \frac{175}{180}\pi]$ and α_1 is the difference between the heading angle of X_0 and X_2. θ_0 is the heading angle of X_0. After completing the obstacle avoidance path, the direction of the vehicle should be parallel to the reference path. Therefore, the relationship between θ_g and θ_0 is,

$$\theta_g = \theta_0 - \beta \quad (15)$$

Where, β is the heading angle of the target point on the reference path. The supplementary constraint equations are,

$$\begin{cases} l_1 + l_2\cos(\pi - \alpha_1) = x_g - l_3\cos\theta_g \\ -l_2\sin(\pi - \alpha_1) = y_g - l_3\sin\theta_g \end{cases} \quad (16)$$

Where, (x_g, y_g) is the coordinate of the target point X_2. So, when α_1 and l_1 are given, the expressions of l_2 and l_3 are,

$$\begin{cases} l_3 = \dfrac{(l_1 - x_g)\sin\alpha_1 + y_g\cos\alpha_1}{\sin\theta_g\cos\alpha_1 - \cos\theta_g\sin\alpha_1} \\ l_2 = \dfrac{l_3\cos\theta_g + l_1 - x_g}{\cos\alpha_1} \end{cases} \quad (18)$$

3.2 B-spline path generation

After generating the segments of the control line, a B-spline path can be generated. We generate a path set and through select the curve with the highest security and the best path curvature.

On the selected tentacle path, starting from the path point corresponding to the target point on the reference road, picking points in the lateral direction of the reference road. And then generating the path set. The way of picking target points is shown in Fig. 7. The blue curve is the reference path, the red curve is the best selected tentacle. G is the reference target point, which is the point on the reference road ahead where the obstacle is detected The sampling distance d of the target points can be changed. $G_n(x_{G_n}, y_{G_n})$ is the coordinate of the target points obtained from the sampling,

$$\begin{cases} x_{G_n} = x_G + nd\sin\theta_g \\ y_{G_n} = y_G + nd\cos\theta_g \end{cases}, n \in [-i, i] \quad (18)$$

Based on the best tentacle path, we only need to sample the target point near in the sampling area. The target points of sampling are greatly reduced and the range of sampling is more accurate. Setting d to 0.2 m and the sampling points to 6, then the coverage range is [-0.8m, 0.8m]. The collision detection method for path set is the same as the tentacle track obstacle detection method in the previous chapter. The path set is selected based on the length, the sum of the squares of the curvature, the sum of the squares of the curvature derivatives, and the distance between the target point and the reference path.

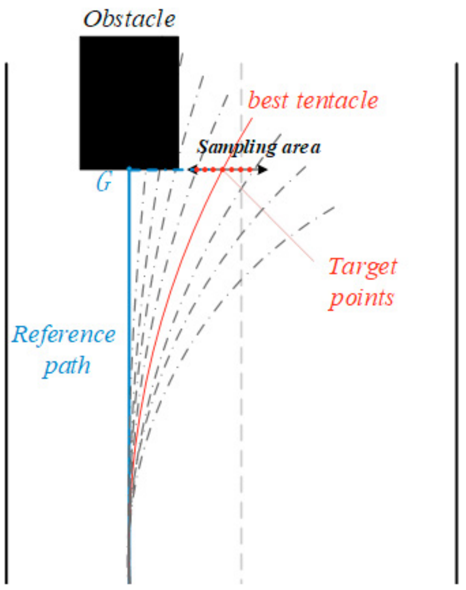

Figure 7. Sampling of target points

The selected B-spline path and the tentacle path generated before are combined to form the designed obstacle avoidance path. At this moment, the vehicle has already begun to execute the previously selected tentacle path. After driving on the tentacles, the vehicle will continue to travel along the B-spline path to complete obstacle avoidance.

4. SIMULATION AND EXPERIMENT RESULTS

4.1 Simulation results

In order to verify the effectiveness of the planning and control algorithms above, the simulation and experiment are designed respectively. Simulations were conducted in the C++ 11/Linux and executed on a Jetson TX2 that runs at HMP Dual Denver 2/2 MB L2 + Quad ARM® A57/2 MB L2. Then, we conducted several experiments based on a modified electric intelligent vehicle platform in a structured road, to verify the practicability of our algorithm.

The simulation is conducted in a successive lane-changing scenario to avoid static obstacles. The length of the vehicle is 3.6 m and the width is 1.6 m. The width of each lane is 3.5 m.

The scenario is on a two-lane road with one obstacle in front of the vehicle. The vehicle generates a set of tentacle trajectories based on the current speed set off-line. In this scenario, the velocity of the vehicle is 10km/h, thus the speed set j is 5. The allowed maximum curvature of our test car is set to 0.25 m^{-1}. When the system detects obstacles within a certain distance and determines that obstacle avoidance is required, obstacle detection and selection are performed on the tentacles. As shown in Fig. 8, the white rectangle is the initial position when the vehicle starts to avoid obstacles, and the black rectangle is the obstacle. (a) is the generated tentacle paths for obstacle detection, (b) is the best tentacle; (c) and (d) demonstrate the process of generating B-spline curve based on the tentacle path. In (c), target points are picked based on the best tentacle to generate a path set. The blue curves are the generated B-spline set and the red curve is the executed tentacle path. (d) is the selected optimal path. Fig.9 demonstrate the curvature of B-spline curve, from which it can be testified that the curve constructed based on B-spline has continuous curvature and limited curvature maximum (absolute value). The curvature has the maximum of 0.02865/m and meet the limitation for motion.

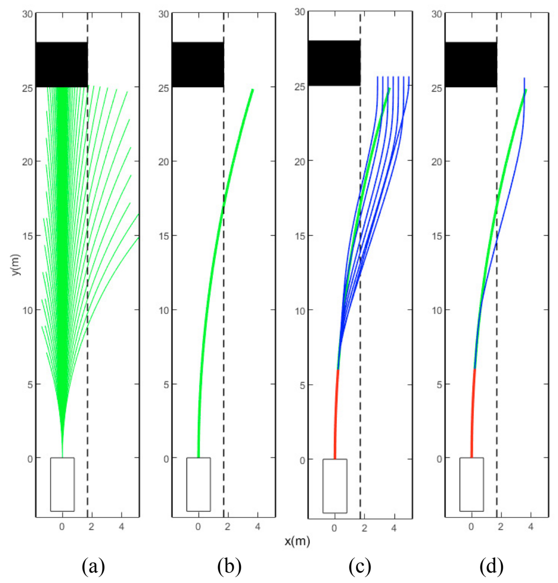

Figure 8. Generated path based on tentacles and B-spline.

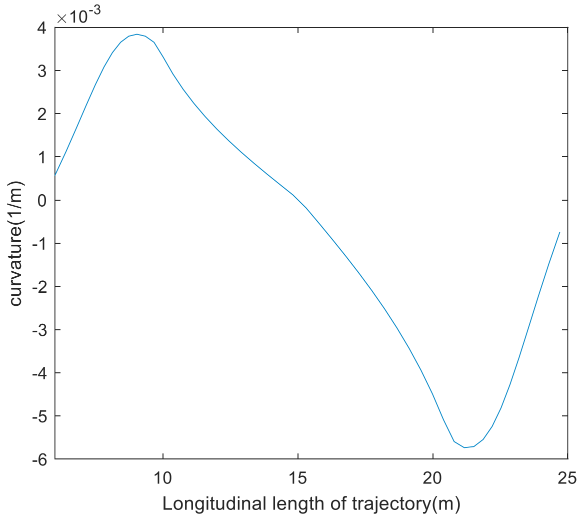

Figure 9. Curvature of B-spline curve.

As shown in Fig.10, the mean reaction time for obstacle avoidance, namely the mean duration for the best tentacle path selection is only 4.8914ms. And as illustrated in Fig.11, the duration has the probability of 15% to exceed 10ms and there is 59% probability to generate path less than 3ms. Shown in Fig.12 and Fig.13 are the calculation time for generating a B-spline path with the sampling area provided by the best tentacle. The mean duration for B-spline path generation is only 9.9033ms. During the execution of the best tentacle path, B-spline curve has sufficient time to be calculated and generated.

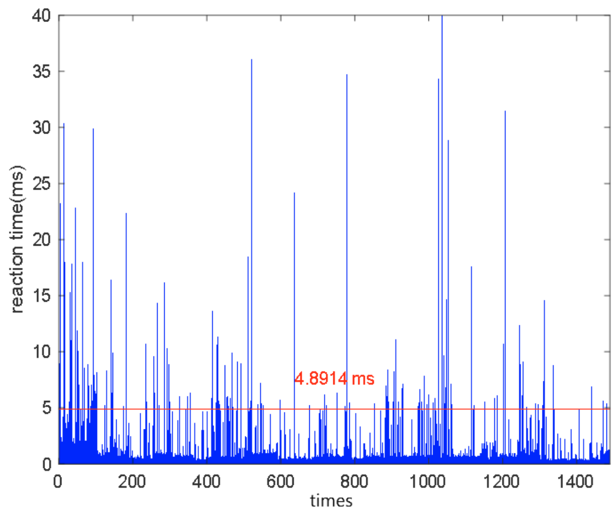

Figure 10. Reaction time for obstacle avoidance.

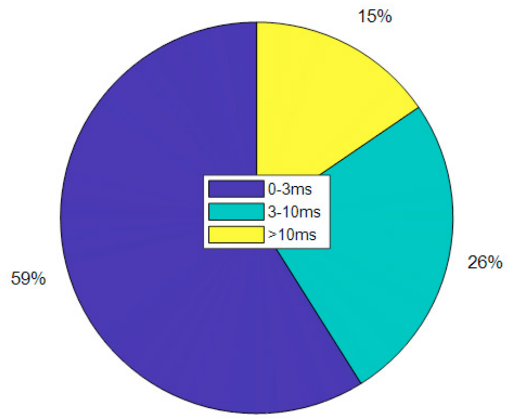

Figure 11. The distribution of response time for obstacle avoidance.

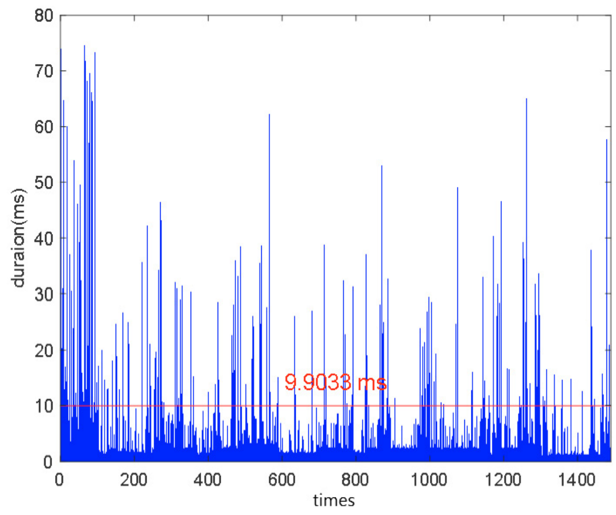

Figure 12. Duration for B-spline path calculation.

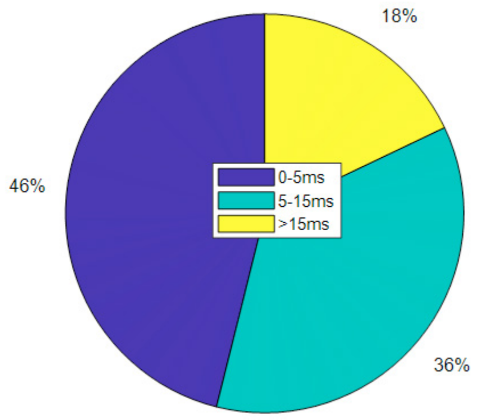

Figure 13. Duration distribution for B-spline path calculation.

Path planning approach in Zeng et al. (2019a) is similar with this work, but for Zeng and his colleagues, B-spline curves are generated by picking a series of target points in the lateral direction of the reference road. Its Response time, namely the generation time for obstacle avoidance path using B-spline curve is shown in Figure 14. The mean duration is 59.1ms, and there are plenty of times over 100ms, even up to 250ms. And as illustrated in Fig.15, the generating duration has the probability of 9% to exceed 100ms. Compared with them, the approach proposed in this paper greatly accelerates the response speed of obstacle avoidance, and its average response time is reduced by about 91.7%. At the same time, the calculation time for generating B-spline trajectories is also greatly reduced, which further reduces the computing resources required by the planning algorithm.

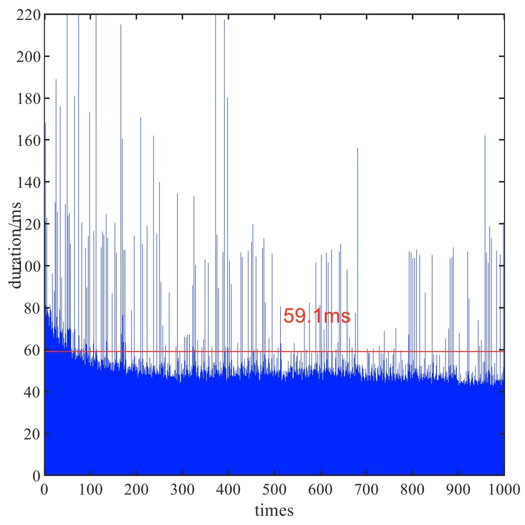

Figure 14. Reaction time for obstacle avoidance in Zeng et al. (2019a)

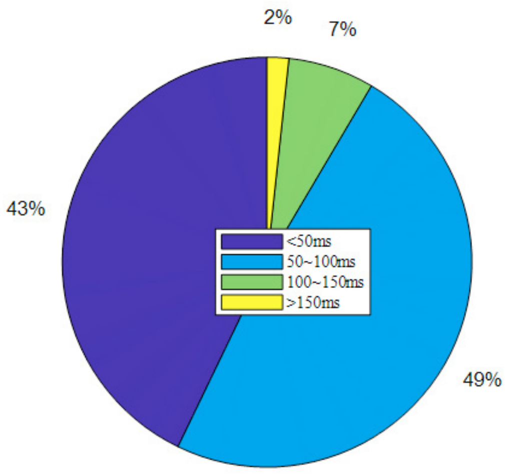

Figure 15. Reaction time distribution for obstacle avoidance in Zeng et al. (2019a).

5.2 Experiment results

To evaluate the developed algorithm, a modified vehicle based on E50 electro car is used as our verification platform as shown in Fig. 16, which was equipped with a centralized drive motor, a steering motor, an electronically controlled hydraulic brake system, an industrial computer and a vehicle control unit. A Pandar 40AC LIDAR, which perceives environment and generates occupancy grid map, is fixed in the roof of vehicle. The exact position of the vehicle is obtained by RTK and IMU.

Fig. 17 and Fig.18 are windows of planner in two obstacle avoidance scenarios. From left to right is 1) environmental point cloud map; 2) reference map: in which the green curves are the tentacle set planned at the current speed, and the blue line is the reference road; 3) the planned paths map: including the reference road, the selected tentacle for sampling target points, red B-spline path set generated based on the best tentacles; 4) tracking path map: including the final designed path for vehicle tracking. The real-time speed is kept about 10km/h in each scenario.

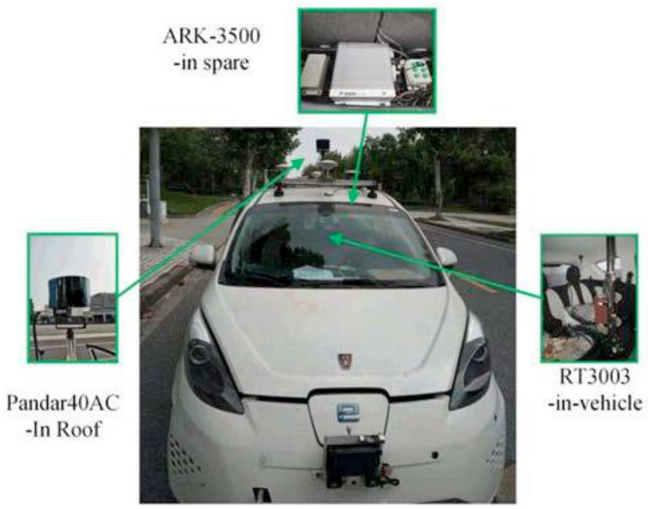

Figure 16. The modified vehicle platform for experiment.

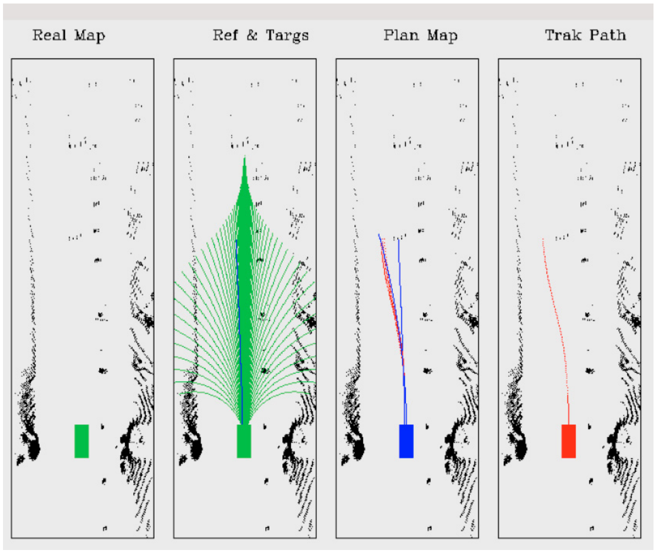

Figure 17. The planner window in scenario 1.

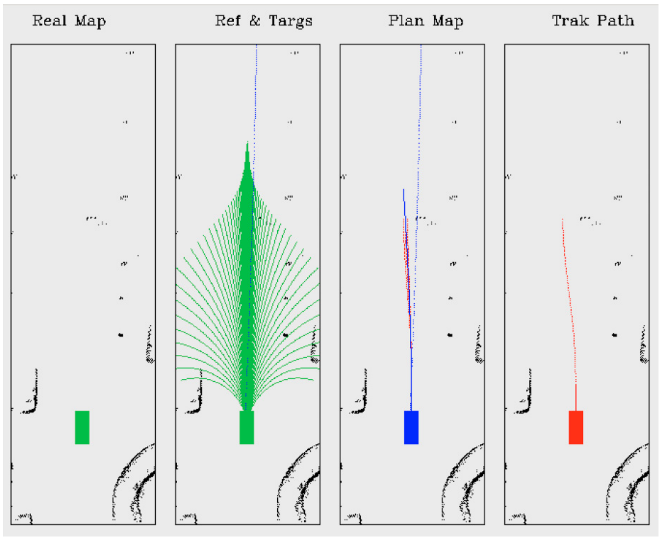

Figure 18. The planner window in scenario 2.

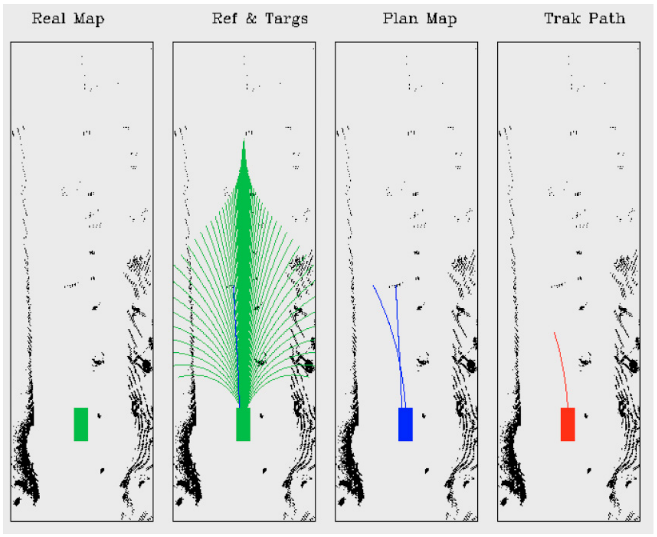

Figure 19. The planner window with close obstacles.

Fig. 19 shows the scenario that the vehicle detects a close obstacle. In this scenario, the distance between the vehicle and the obstacle is not enough to use the B-spline curve to generate a smooth path. Therefore, the vehicle will follow the executing tentacle path and re-plan it at the next planning cycle. Under each experimental situation, the reaction time for obstacles avoidance is within 10ms.

5. CONCLUSIONS

In this work, a real-time local path planning approach for intelligent vehicle is proposed, which combines the tentacle algorithm and B-spline approach. The tentacle algorithm provides a safe driving direction for the vehicle quickly and generates a sampling reference area for B-spline curves. Thus, the target points of the B-spline curve can be sampled more easily and efficiently. During the vehicle driving along the tentacles path, the B-spline path is generated and it does not need to take any reaction time. By combining tentacles with B-sample curve, the generated path is safe and smooth. The approach was tested in multiple obstacle avoidance scenarios on structure road. Simulation and experimental results show that the proposed planning approach has extremely fast obstacle avoidance response. Compared with some sampling-based path set approaches, our approach has better real-time performance. In the future, we plan to consider the impact of the uncertainty of the environmental map on our algorithm.

ACKNOWLEDGEMENTS

This work is supported by the Fundamental Research Funds for the Central Universities (Grant no. 22120190205).

REFERENCES

Akmandor, N. Ü. and T. Padir (2020). A 3D Reactive Navigation Algorithm for Mobile Robots by Using Tentacle-Based Sampling. *IEEE International Conference on Robotic Computing (IRC), pp. 9-16.*

Alia, C. and Gilles, T., Reine, T. and Ali, C. (2015). Local trajectory planning and tracking of autonomous vehicles, using clothoid tentacles method. *2015 IEEE Intelligent Vehicles Symposium (IV), pp. 674-679.*

Daniel, K., Nash, A., Koenig, S. and Feiner, A. (2010). Theta: Any-Angle Path Planning on Grids. *Journal of Artificial Intelligence Research* ***39****: 533-579.*

Delsart, V., Fraichard, T. and Martinez, L. (2009). Real-time trajectory generation for car-like vehicles navigating dynamic environments. *IEEE International Conference on Robotics and Automation, 2009, pp. 3401-3406.*

Elbanhawi, M. and Simic, M. and Jazar, R. N. (2015). Continuous Path Smoothing for Car-Like Robots Using B-Spline Curves. *Journal of Intelligent & Robotic Systems, 80 (S1), pp. 23-56.*

Ferguson, D. and Stentz, A. (2006). Using interpolation to improve path planning: The Field D* algorithm. *Journal of Field Robotics, 23(2), pp. 79-101.*

Gonzalez, D., Perez, J. Milanes, V. and Nashashibi, F. (2016). A Review of Motion Planning Techniques for Automated Vehicles. *IEEE Transactions on Intelligent Transportation Systems, 17(4), pp. 1135-1145.*

Hundelshausen, V. F., Himmelsbach, M., Hecker, F., Mueller, A. and Wuensche, H. (2008). Driving with tentacles: Integral structures for sensing and motion. *Journal of Field Robotics, 25(9), pp. 640-673.*

Jayasree, K. R., Jayasree, P. R., and Vivek, A. (2017). Smoothed RRT techniques for trajectory planning, *2017 International Conference on Technological Advancements in Power and Energy (TAP Energy), pp. 1-8.*

Kammel, S., Ziegler, J., Pitzer, B., Werling, M., Gindele, T., Jagzent, D., Schröder, J., Thuy, M., Goebl, M., Hundelshausen, V. F., Pink, C., Frese, C. and Stiller, C. (2008). Team AnnieWAY's autonomous system for the 2007 DARPA Urban Challenge. *Journal of Field Robotics, 25(9), pp. 615-639.*

Katrakazas, C., Quddus, M. Chen, W. and Deka, L. (2015). Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. *Transportation Research Part C: Emerging Technologies, 60, pp. 416-442.*

Li, Q. and Z. Zeng, Yang, B. and Zhang, T. (2009). Hierarchical route planning based on taxi GPS-trajectories. *17th International Conference on Geoinformatics, pp. 1-5.*

Li, Y., Xiong, L., Zeng, D., Xu, P. and Li, Z. (2019). A Real-Time Obstacle-Avoidance Trajectory Planner for On-Road Autonomous Vehicle. *SAE Technical Paper 2020-01-5018, 2020.*

Likhachev, M., Ferguson, D., Gordon, G., Stenz, A. and Thrun, S. (2008). Anytime search in dynamic graphs. *Artificial Intelligence, 172(14) pp. 1613-1643.*

Likhachev, M. and Ferguson, D. (2009). Planning Long Dynamically Feasible Maneuvers for Autonomous Vehicles. *The International Journal of Robotics Research, 28(8), pp. 933-945.*

Mouhagir, H., Talj, R., Cherfaoui, V., Aioun, F. and Guillemard, F. (2020). Evidential-Based Approach for Trajectory Planning with Tentacles, for Autonomous Vehicles. *IEEE Transactions on Intelligent Transportation Systems, 21(8), pp. 3485-3496.*

Paden, B., Cap, M., Yong, S., Yershov, D. and Frazzoli, E. (2016). A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles. *IEEE Transactions on Intelligent Vehicles, 1(1), pp.33-55.*

Stentz, A. (1994). Optimal and efficient path planning for partially-known environments. *Proceedings of the 1994 IEEE International Conference on Robotics and Automation, 4, pp. 3310-3317.*

Zeng, D., Yu, Z., Xiong, L., Zhao, J., Zhang, P., Li, Z., Fu, Z., Yao, J. and Li, Z. (2019a). A Novel Robust Lane Change Trajectory Planning Method for Autonomous Vehicle. *2019 IEEE Intelligent Vehicles Symposium (IV), pp. 486-493.*

Zeng, D., Yu, Z., Xiong, L., Zhao, J., Zhang, P. and Fu, Z. (2019b). A Steerable Curvature Approach for Efficient Executable Path Planning for on-Road Autonomous Vehicle. *SAE Technical Paper 2019-01-0675, 2019.*