Adaptive sampling-based motion planning with a non-conservatively defensive strategy for autonomous driving

Zhaoting Li [*] **Wei Zhan** [**] **Liting Sun** [**] **Ching-Yao Chan** [**] **Masayoshi Tomizuka** [**]

[*] *Harbin Institute of Technology, Harbin, China, 150001.*
[**] *University of California, Berkeley, CA, 94706 USA (e-mail: wzhan@berkeley.edu)*

Abstract: Sampling-based motion planning methods are widely adopted in autonomous driving. Typically, sampling can be decoupled into two layers: a path sampling layer and a speed profile sampling layer. For the path sampling layer, traditional methods tend to sample with a uniform distribution over the whole feasible space, which might cause either computational inefficiency or poor performance if the sampling resolution is not set appropriately. To solve this problem, we propose an adaptive path sampling approach that samples from a time-varying distribution depending on the dynamic environment and potential costs of the ego vehicle. Such sampling strategy is then combined with a non-conservatively defensive strategy in the speed sampling layer to generate a set of safe but not overcautious trajectories. The proposed motion planning framework is tested both in simulation and a real autonomous vehicle in a roundabout scenario. The results demonstrate that it can efficiently generate non-conservative but defensive trajectories to safely drive the vehicles in dynamic environments full of uncertainties.

Keywords: Autonomous vehicles, Motion planning, Adaptive Sampling, Non-conservatively defensive strategy

1. INTRODUCTION

In the past three decades, both academia and industry have invested tremendous efforts in autonomous driving. Motion planning for autonomous driving plays an important role in generating vehicles trajectories that reach a goal region from a start point while avoiding obstacles and satisfying many requirements including time-efficiency and user comfort. Typically, motion planning includes two components: path generation and trajectory generation (i.e., the speed), both of which are fundamental modules for safe autonomous driving.

For path generation, many traditional sampling-based motion planning methods in robotics have been applied into autonomous vehicles, including rapidly-random exploring tree (RRT), Probabilistic RoadMap (PRM), etc., according to LaValle (2006). Lindemann and LaValle (2005) has pointed out that randomization has great success in finding a feasible path even in a NP-hard problems. The application of randomization in autonomous driving, however, might not be safe enough since it may result in poorly-connected graphs with bad reproducibility, according to Claussmann et al. (2019). Although they are in general computationally efficient to find a feasible path, these random sampling-based methods tend to fail to find the optimal path in most of the scenarios. On the other hand, deterministic sampling approaches enumerates all possibilities over the feasible space with a pre-defined resolution. Although their reproducibility is good, they suffer from the curse of dimension. There is a trade-off between time efficiency of the algorithm and optimality of the solutions, as addressed in Gu et al. (2013). Higher sampling resolution will help to find better solutions, but it will in the meanwhile decrease the time efficiency, leading to dangerous manuveaurs for autonomous vehicles. One direction to speed up the runtime of sampling algorithms while preserving the quality of the solution is to change from uniform resolution to non-uniform ones to improve the sampling efficiency. Ichter et al. (2018) proposed to use a non-uniform resolution in RRT-based on a learned distribution from human demonstrations. Kumar and Chakravorty (2012) introduced an adaptive-resolution-based sampling strategy in PRM by explicitly utilizing the information encoded in the connectivity graph.

Path planning is mainly handling the routing and static obstacles in the environment. Most of the collision avoidance with dynamic obstacles, as well as increasing comfort and time-efficiency of the trajectory are tackled by the trajectory planning layer since it directly generates the speed and acceleration of the vehicle. Dealing with dynamic obstacles are hard, since the behaviors of other road participants are full of uncertainties. The trajectory planner needs to consider all possibilities and generates trajectories that are defensive enough to protect the autonomous vehicle, but not too conservative, i.e., sacrificing too much efficiency and comfort of the vehicle/passengers (Zhan et al. (2016)). However, traditional sampling approaches in the trajectory planning layer cannot generate such non-conservatively defensive strategies due to their uniform sampling along the planning horizon. Most of them tend

Manuscript 4261 submitted to 21st IFAC World Congress, 2020.
Received November 20, 2019.

to be over cautious. For instance, the autonomous vehicles make too many unnecessary stops when entering roundabouts, over-reacting to the others' actions (Broggi et al. (2014)).

In this paper, we propose novel sampling approaches in both the path layer and the trajectory layer to address the aforementioned two problems. For path layer, we also use non-uniform resolution. Instead of learning it from human demonstrations which requires additional data, we propose to adaptively adjust the resolution based on the surrounding environment and the objectives of the ego vehicle. For trajectory layer, we introduce the decision-making module with uncertainties (as in Zhan et al. (2016)) into the speed sampling strategy to generate non-conservatively defensive trajectories.

The remainder of the paper is organized as follows. Section 2 presents the overall framework of the motion planning strategy, followed by the details for the proposed path planner by adaptive sampling method in Section 3. Section 4 shows the trajectory planning layer with the non-conservatively defensive strategy (NCDS). Section 5 presents the experiment results and Section 6 concludes the paper.

2. OVERVIEW OF THE FRAMEWORK

The overall planning scheme is shown in Fig. 1.

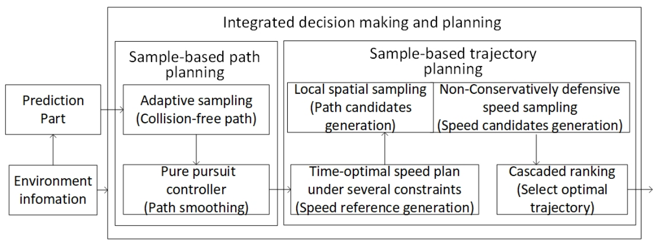
Integrated decision making and planning
Sample-based path planning
Sample-based trajectory planning
Prediction Part
Environment infomation
Adaptive sampling (Collision-free path)
Pure pursuit controller (Path smoothing)
Local spatial sampling (Path candidates generation)
Non-Conservatively defensive speed sampling (Speed candidates generation)
Time-optimal speed plan under several constraints (Speed reference generation)
Cascaded ranking (Select optimal trajectory)

Fig. 1. An overview of the motion planning framework

As shown in Fig. 1, the path generation part performs an adaptive sampling method whose sampling distribution is adaptively changed according to the environment, particularly for the static obstacles. After generating a graph with many sampling nodes, dynamic programming is used to find the optimal collision-free path. After that, the pure pursuit controller is used to track this piecewise-linear path to generate a smooth and collision-free path. Based on the path generated by the first part, the trajectory generation part will generate the optimal speed profiles, considering the comfort, efficiency and safety in the presence of dynamic obstacles. This is achieved by two steps. First, the speed reference is generated by time-optimal speed plan that satisfies several constraints. Then, a local speed sampling is conducted to avoid dynamic obstacles. In the second step, NCDS is utilized. Once all samples from both layers are obtained, a cascaded ranking method is applied to select the best trajectory candidate for the autonomous vehicle.

3. PATH PLANNER WITH ADAPTIVE SAMPLING

By sampling many points whose distribution is uniform on the road or along a reference, we can generate a graph consisting of nodes with different costs. Then we can use dynamic programming to find the nodes sequences with minimum costs.

3.1 Graph generation

Roads are always bounded by two edges, and a centerline or reference can be obtained via perception module or map information. Therefore, this enables us to sample many nodes along the centerline whose positions have regular patterns. A uniform sampling pattern can be found in Gu et al. (2013), which is shown in the first picture of Fig.2. The quality of path generated by this uniform pattern is highly limited with the resolution. However, higher resolution tends to result in more runtime during dynamic programming. In our adaptive sampling method, this limit is solved by sampling important areas with more layers and sampling each layers guided by cost distribution.

We define the centerline as the s axis of the coordinate, the middle point of our car as the origin and the vertical line of the centerline as the d axis, respectively. The graph of road is constructed by many ordered layers and one layer is composed of many nodes. The sampling patterns of nodes in a graph are determined by two distributions: the sampling distribution of layers and the sampling distribution of nodes in one layer. Suppose the number of layers is N, and a layer is denoted by L_i $(i = 1, 2, \ldots, N)$. We define the number of nodes on layer L_i is M, the location of a node on L_i is denoted by $\mathbf{n}_{ij}$ $(j = 1, 2, \ldots, M_i)$.

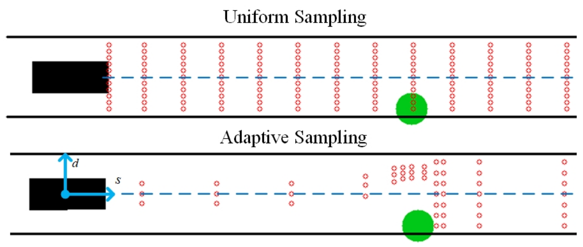

Fig. 2. Uniform sampling and adaptive sampling

Each node in this graph has a cost. We compute the cost of each node by applying repulsive forces and attractive forces on it. This cost computation way is similar to a discrete elastic band method proposed by Gu et al. (2015). The repulsive forces are generated by static obstacles to make the path collision-free. The repulsive force F_r on node n_{ij} are computed by this equation (1) proposed by Quinlan and Khatib (1993):

$$F_r = \begin{cases} (d_0 - d)\, \partial d / \partial \mathbf{n}_{ij}, d < d_0 \\ 0, d \geq d_0 \end{cases}, \tag{1}$$

where d_0 is the distance from obstacles up to which the force is applied and k_r is the repulsive force gain. The attractive forces are generated by the centerline to make the path on the road. Attractive forcesF_a are computed by the distance from the centerline. Then the cost of node n_{ij} is:

$$Cost\,(n_{ij}) = (k_a F_a + k_c F_c)^2. \tag{2}$$

3.2 Adaptive sampling method

The core idea of this adaptive sampling method is to sample with a non-uniform resolution. This method sample more useful areas with a high resolution to help us find an optimal path faster. We believe that usually important areas are near obstacles. Take the scenario where there is one obstacle as an example. To make more layers near obstacles, two normal distributions are added at O_{start} and O_{end} respectively, as shown in Fig. 3. Where O_{start} and O_{end} are the locations of the start edge and the end edge of the obstacle respectively.

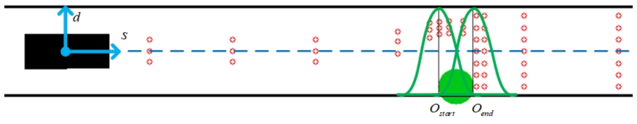

Fig. 3. Normal distributions added near obstacles

These two normal distribution is denoted by:

$$\begin{aligned} f_1(s) &= exp\left(-(s-\mu_1/\sigma_1)^2\Big/2\right)\Big/\left(\sqrt{2\pi}\sigma_1\right) \\ f_2(s) &= exp\left(-(s-\mu_2/\sigma_2)^2\Big/2\right)\Big/\left(\sqrt{2\pi}\sigma_2\right) \end{aligned} \tag{3}$$

Then the distribution of layers is

$$F(s) = N\left(f_1(s) + f_2(s) + \omega/l\right)/(2+\omega). \tag{4}$$

The area of this new distribution is

$$H(s) = N\left(\Phi_1(s) + \Phi_2(s) + \omega x/l\right)/(2+\omega), \tag{5}$$

where N is the number of layers, ω is the weight of uniform distribution and two normal distributions and l is the length of road. Note that $H(l) = N$. To find the location of layers, we can divide the area of this new distribution into N parts. Then the problem of finding the location of layers can be transformed into this mathematic problem: Find all the $s \in [0, l]$ which satisfy

$$H(s) - k = 0, (k = 1, 2, \ldots, N). \tag{6}$$

This equation can be solved it by newton method whose update step in every loop is limited by $\min(3\sigma_1, 3\sigma_2)$. Besides, the value of $\Phi_1(s), \Phi_2(s)$ can be calculated by the std::erf function in C++. The area of standard normal distribution can be solved by this equation:

$$\Phi(s) = 0.5 + 0.5erf(s\Big/\sqrt{2}), \tag{7}$$

where the error function $erf(s)$ can be expressed as:

$$erf(s) = 2\int_0^s exp\left(-t^2\right)dt/\sqrt{\pi}. \tag{8}$$

After determining the location of layers, the location of nodes on each layer can be determined by a biased sampling method. The uniform sampling way of nodes on one layer is also inefficient. For example, it is obvious that we do not need to sample too many points in the places under centerline if there is an obstacle under centerline in front of us. We also do not need to sample points within or near obstacles. The biased sampling method has two steps. Firstly, the costs of nodes sampled by a uniform distribution on one layer are calculated by equation 2. Define the costs of these nodes on one layer as $C_m\,(m = 1, 2, \cdots, M)$, where M is the number of nodes on one layer. Define the location of these corresponding nodes of these costs on one layer as $d_m\,(m = 1, 2, \cdots, M)$. A cost distribution function $f_{\cos t}(d)$ can be generated by connecting the cost value of adjacent nodes with line segments, which is calculated by equation 9.

In the interval $[d_i, d_{i+1}]\,(1 \le i \le M-1)$, $f_{\cos t}(d)$ can be represented as

$$f_{\cos t}(d) = \frac{d - d_{i+1}}{d_i - d_{i+1}}C_i + \frac{d - d_i}{d_{i+1} - d_i}C_{i+1}. \tag{9}$$

Then the distribution of nodes $f_{node}(d)$, in one layer can be obtained by this cost distribution by equation 10, which is shown in Fig. 4.

$$f_{node}(d) = 1/\left[f_{\cos t}(d) + 1\right]. \tag{10}$$

To determine the location of nodes on one layer, the method is the same as the method used in the determination of location of layers.

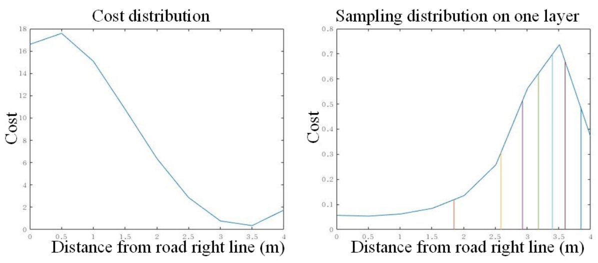

Fig. 4. Biased sampling guided by cost distribution on one layer

To find a collision-free path is to find a node sequence $\{n_k | n_k \in L_k, k = 1, 2, \ldots N\}$, which minimize the total cost of these nodes:

$$\underset{\{n_k | n_k \in L_k, k=1,2,\ldots N\}}{\arg\min} \sum_{k=1}^{N} Cost(n_k). \tag{11}$$

Then this problem can be solved by dynamic programming. The problem can be divided into this sub problem that find a node on layer L_k that minimize the total cost from this node to the end node:

$$\begin{aligned} &Cost_{\min}^{toend}(n_k) = Cost_{\min}^{toend}(n_{k+1}) \\ &+Cost(n_k) + |\mathbf{n_k} - \mathbf{n_{k+1}}| \end{aligned}, \tag{12}$$

where $|\mathbf{n_k} - \mathbf{n_{k+1}}|$ denotes the distance between node n_k and node n_{k+1}. This item is added to smooth the path, which has the same effects with the contractive forces in discrete elastic band method.

3.3 Smooth by pure pursuit controller

The collision-free path consisting of nodes sequence is piecewise-linear, which is not smooth in curvature and sometimes kinematically infeasible. The kinematic bicycle model proposed by Gillespie (1992) and the pure pursuit controller proposed by Coulter (1992) is used to smooth this path. The kinematic bicycle model can be written in this two-input form:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \\ \tan(\delta)/L \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \dot{\delta}, \tag{13}$$

where v and $\dot{\delta}$ are the longitudinal velocity and the angular velocity of the steered wheel respectively. We use the pure pursuit controller to track this piecewise-linear path. After computing the steering angle δ by pure pursuit controller in each loop, we set the speed v to a small constant value. Then we can update the state of the car model by equation (13). The final path is consisted of all the states during the loop of update by pure pursuit controller. The smoothed path is not equal to the original collision-free path, which means that it has a very small possibility that the smoothed path is not collision-free. Therefore, a final evaluation on this path is conducted. The calculation of the cost of this path is similar to the calculation of cost of nodes. If the cost exceeds the threshold of safety, a replanning whose sampling distributions need to be modified is required. After this smoothing part, the path is both collision-free and kinematically feasible.

4. SPEED PLANNER WITH THE NON-CONSERVATIVELY DEFENSIVE STRATEGY

4.1 Trajectory sampling

Suggested speed profile generation Till now, a smooth and collision-free path is generated, but the speed information is still unavailable. A suggested speed profile is generated by finding a time-optimal speed plan under several constraints. This suggested speed profile can be used to calculate the cost of difference to speed reference. The constraints include Max speed limitation, jerk constraint, lateral acceleration constraint, longitudinal acceleration and deceleration constraint and obstacle proximity constraint. The details can be found in Gu (2017).

Local trajectory sampling The trajectory sampling includes local spatial sampling and speed sampling. The spatial sampling set is generated using cubic interpolation, which is shown in Fig. 5.

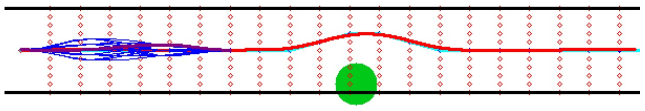

Fig. 5. Local spatial sampling

This speed sampling method adopts a non-conservatively defensive strategy (NCDS) proposed by Zhan et al. (2016). For a typical sampling-based trajectory planner, the time horizon of a planned trajectory is quite long compared with the time actually executed by the host vehicle. Different long-term motions are considered to deal with the uncertainty of different cases in future. However, because the motion that will be executed at the next planning cycle should be determined, the short-term motion should be the same these different cases in future.

The key idea of this sampling strategy is to sample the trajectories under both the yielding and passing situation. In the short term horizon, these two trajectories under these two situations have the same speed profile. In the long term horizon, the final speed of the trajectory under passing situation is not smaller than the final speed of the trajectory in the short term horizon. The final speed of the trajectory under yielding situation is smaller than the final speed of the trajectory in the short-term horizon. Besides, we also sample some speed profiles whose stop time is shorter than the long-term horizon. The sampled speed profiles whose short-term final speed is $5m/s$ are shown in Fig. 6.

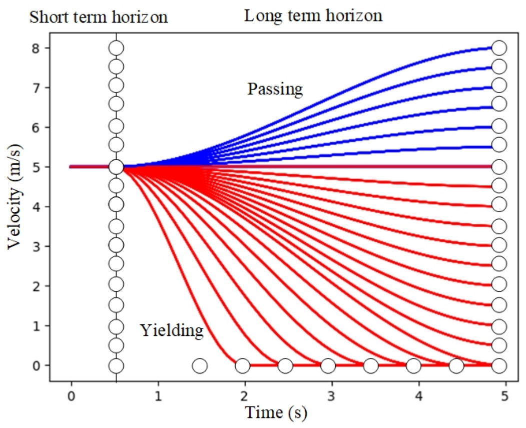

Fig. 6. Speed profiles generation by non-conservatively defensive strategy

To sample the speed in short-term horizon, we use cubic curve to represent the speed profile:

$$v_{short}(t) = s_0 + s_1 t + s_2 t^2 + s_3 t^3. \tag{14}$$

We have four constraints: start and final speed constraints, start acceleration constraints, final jerk constraints. Given the start speed v_0 and acceleration a_0, final speed v_{f0} and jerk J_{f0}, and the short term horizon t_{f0}, the unknown parameters $\{s_0, s_1, s_2, s_3\}$ of short-term speed can be solved by the following equations:

$$\begin{aligned} &v_{short}(0) = v_0 = s_0 \\ &a_{short}(0) = a_0 = s_1 \\ &v_{short}(t_{f0}) = v_{f0} = s_0 + s_1 t_{f0} + s_2 t_{f0}^2 + s_3 t_{f0}^3 \\ &J_{short}(t_{f0}) = 0 = 2s_2 + 6 s_3 t_{f0} \end{aligned} \tag{15}$$

To sample the speed in long-term horizon, we also use the cubic curve to represent the speed profile:

$$v_E(t) = l_0 + l_1 t + l_2 t^2 + l_3 t^3, \tag{16}$$

where $E \in \{pass, yield\}$. We have four constraints: start and final speed constraints, start and final acceleration constraints. Given the start speed v_{f0} and acceleration a_{f0}, final speed v_{f1} and acceleration a_{f1}, and the long term horizon t_{f1}, the unknown parameters $\{l_0, l_1, l_2, l_3\}$ of short-term speed can be solved by the following equations:

$$\begin{aligned} &v_E(0) = v_{f0} = l_0 \\ &a_E(0) = a_{f0} = l_1 \\ &v_E(t_{f1}) = v_{f1} = l_0 + l_1 t_{f1} + l_2 t_{f1}^2 + l_3 t_{f1}^3 \\ &a_E(t_{f1}) = a_{f1} = l_1 + 2 l_2 t_{f1} + 3 l_3 t_{f1}^2 \end{aligned} \tag{17}$$

Note that the long term horizon t_{f1} in passing situation is fixed, while in yielding situation t_{f1} can be many different values to make speed profiles in yielding situation have different stop time.

Cost computation After sampling speed profiles and path, we can get many trajectory candidates. Then we can calculate the cost of different feature terms on every trajectory candidate, which is shown in table 1.

Table 1. Cost computation on different features

Cost feature	Computation
Static obstacles	$\sum 1/d_{static}^2$
Moving obstacles	$\sum 1/d_{moving}^2$
Lateral acceleration	$\sum \left(v^2\kappa\right)^2$
Longitudinal acceleration	$\sum \left(a_{long}\right)^2$
Jerk	$\left(a_0 - a_0^{last}\right)^2$
Temporal diff. from ref	$\sum \left(v - v_{ref}\right)^2$
Spatial diff. from ref	$\sum \left[\left(x - x_{ref}\right)^2 + \left(y - y_{ref}\right)^2\right]$

Where d_{static} and d_{moving} denote the distance from static and moving obstacles respectively, κ represents the curvature of the road, a_{long} is the longitudinal acceleration, a_0 and a_0^{last} are the start acceleration and the start acceleration in the last planning cycle respectively, v_{ref} represents the reference velocity which is generated in suggested speed profile generation part, (x_{ref}, y_{ref}) denotes the reference path generated by the adaptive sampling method .

Then we can add the cost of the passing and yielding situation together with probability using this equation:

$$\begin{aligned} Cost_{total} = Cost_{short} + P(pass)Cost_{pass} \\ + P(yield)Cost_{yield} \end{aligned}, \tag{18}$$

where $P(pass)$ and $P(yield)$ denote the possibility of passing and yielding for the host vehicle, $Cost_{pass}$ and $Cost_{yield}$ represent the cost of each feature in passing and yielding situation respectively. The possibility is computed using the method proposed by Schulz et al. (2018).

4.2 Cascaded ranking

After generating the trajectory candidates, the cascaded ranking method is used to select an optimal trajectory. A cascaded ranking method which was used by Gu et al. (2016) and Wang et al. (2011) is used because of its good tunability. In this method, we use bucket to select candidates. All the buckets correspond to one of features with different priorities. One bucket can be thought of as a filter of its corresponding feature. After the trajectory candidates are put into one bucket, only the candidates satisfying some constraints can be returned, which is illustrated in Fig. 7. After using several bucket to select, we can get some feasible trajectories that have a relative good performance. Then all the feature values of these trajectories are compared and the trajectory that has the maximum value in the feature with top priority is selected as the final trajectory.

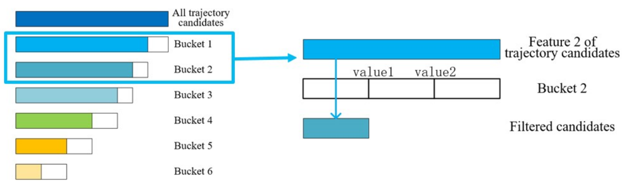

Fig. 7. Cascaded ranking

5. RESULTS

5.1 Simulation

We tested the adaptive sampling method on four scenarios, which was compared with uniform sampling. In this test, the distribution of some layers was still uniform, and we sampled 15 points in one layer during uniform sampling. In the cost and runtime figures shown in Fig. 8, the red lines represent the cost and runtime of final path generated by adaptive sampling and the blue lines represents path generated by uniform sampling. Fig. 8 demonstrates the efficiency of this adaptive sampling method compared with the uniform one. With fewer sampling layers and fewer sampling nodes, the adaptive sampling method can generate a path that is close to the optimal one. The runtime of adaptive algorithm was close to the runtime of uniform sampling. Besides, sometimes it had a shorter runtime because the biased sampling method could reduce the search space of dynamic programming.

Fig. 9 demonstrates the sampling distribution of layers in long box obstacles. This distribution showed the advantages of adding normal distribution at locations of the start edge and the end edge of the obstacle respectively because this adaptive sampling distribution did not need to focus too much on the middle of long box obstacles. That was why we do not choose to add normal distribution directly on the middles of obstacles.

5.2 Experiments

The experiments of this planning framework were conducted on a Lincoln MKZ test vehicle. The test scenario was extracted from a roundabout contained in the INTERACTION dataset Zhan et al. (2019). The driving behavior of the other vehicles were generated either by human drivers using a driving simulator, or replaying the data in the INTERACTION dataset.

In the first case, the motion of the obstacle car was generated by a driving simulator. This obstacle car was in the roundabout when the autonomous car was going to enter this roundabout. The obstacle car had two routes to choose. One was to keep itself in this roundabout, while another was to exit the roundabout. According to the traffic rule, the autonomous car should yield to the obstacle car. We used sample-based planning with this non-conservatively defensive strategy to deal with this situation.

In Fig. 10, the figures on the right are the speed profiles generated by our planner. In these pictures, the green curve is the speed profile in passing situation, and the red curve is the speed profile in yielding situation. The passing possibility is also shown in the top right corner of the picture. In Fig. 10, the left pictures show this roundabout scenario. The long blue line represents the road reference. The short green line represents the path planned in future three seconds. The host vehicle is red one that is on the blue line, while another vehicle was considered as the obstacle.

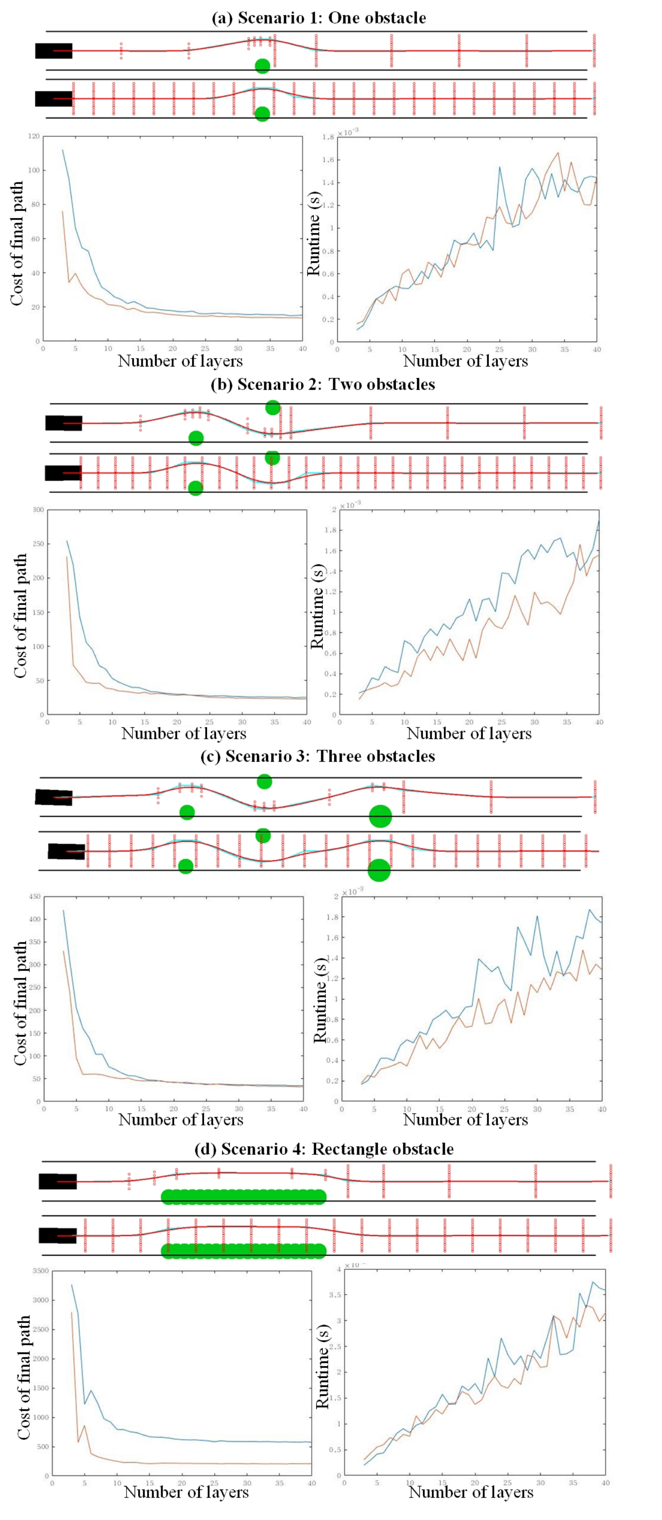

Fig. 8. Simulation results of path generation using adaptive sampling and uniform sampling method

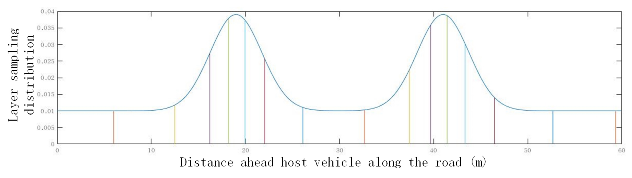

Fig. 9. Sampling distribution in long box obstacles scenario

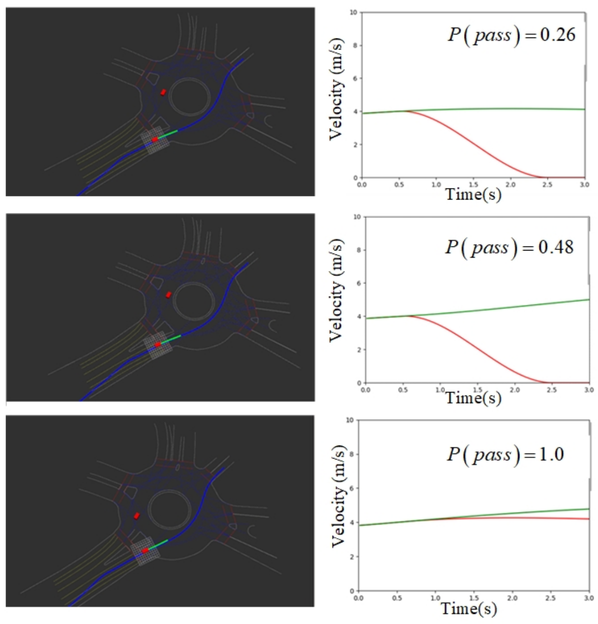

Fig. 10. Roundabout scenario

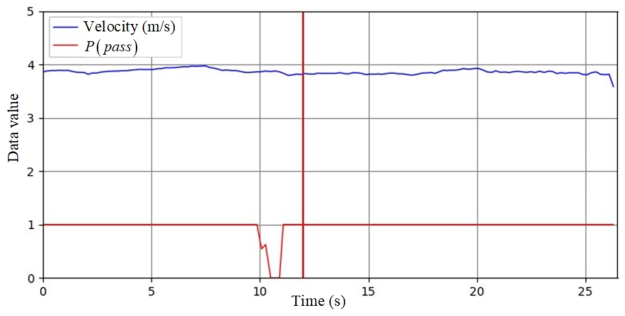

Fig. 11. Velocity profile and passing possibility profile.

When this obstacle car was in the roundabout and had a chance to interact with the host vehicle, our planner generated two speed profiles in both passing and yielding situation. The speed profile in the yielding situation guaranteed the safety of the host vehicle, while the passing situation enabled the host autonomous vehicle not to overreact to the potential danger.

In the second case, the host vehicle is the red cuboid and the blue cuboids represent other vehicles that interact with the host vehicle. The trajectories of blue cars were replayed according to the INTERACTION dataset. Since the obstacles did not tend to yield to the host vehicles, the host vehicle needed to slow down to keep safe. The velocity profile is shown in Fig. 12.

6. CONCLUSION

In this paper, we proposed a planning framework including two main novelties: adaptive path sampling method with non-uniform resolution and non-conservatively defensive speed profiles sampling method. This novel planning framework addressed the inefficiency of current sampling methods and the overcautious behaviors of current trajectory planning methods.

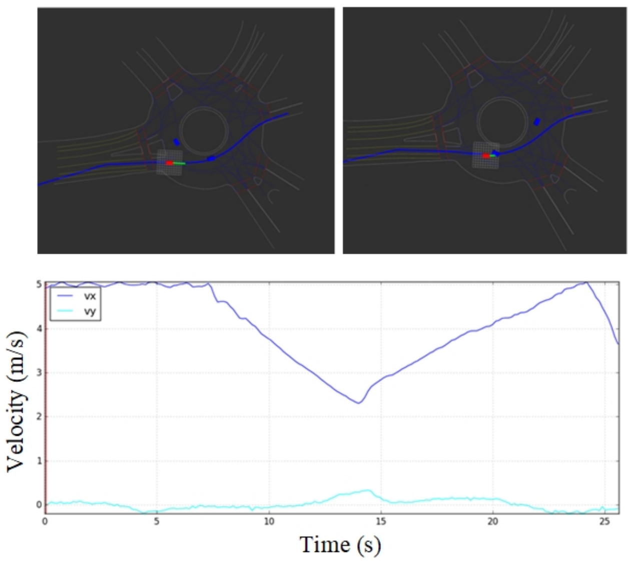

Fig. 12. Yielding when other cars still go around of this roundabout.

For future work, a sampling distribution based on history planning information will be tested. How to combine the history information with the current environment information will also be explored. For trajectory sampling part, a more efficient sampling way need more research work. More complex scenarios will also be used to test the planning framework.

REFERENCES

Broggi, A., Cerri, P., Debattisti, S., Laghi, M.C., Medici, P., Panciroli, M., and Prioletti, A. (2014). Proud-public road urban driverless test: Architecture and results. In *2014 IEEE Intelligent Vehicles Symposium Proceedings*, 648–654. IEEE.

Claussmann, L., Revilloud, M., Gruyer, D., and Glaser, S. (2019). A review of motion planning for highway autonomous driving. *IEEE Transactions on Intelligent Transportation Systems.*

Coulter, R.C. (1992). Implementation of the pure pursuit path tracking algorithm. Technical report, Carnegie-Mellon UNIV Pittsburgh PA Robotics INST.

Gillespie, T.D. (1992). *Fundamentals of vehicle dynamics*, volume 400. Society of automotive engineers Warrendale, PA.

Gu, T. (2017). *Improved trajectory planning for on-road self-driving vehicles via combined graph search, optimization & topology analysis.* Ph.D. thesis, Carnegie Mellon University.

Gu, T., Atwood, J., Dong, C., Dolan, J.M., and Lee, J.W. (2015). Tunable and stable real-time trajectory planning for urban autonomous driving. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 250–256. IEEE.

Gu, T., Dolan, J.M., and Lee, J.W. (2016). Runtime-bounded tunable motion planning for autonomous driving. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, 1301–1306. IEEE.

Gu, T., Snider, J., Dolan, J.M., and Lee, J.w. (2013). Focused trajectory planning for autonomous on-road driving. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, 547–552. IEEE.

Ichter, B., Harrison, J., and Pavone, M. (2018). Learning sampling distributions for robot motion planning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 7087–7094. IEEE.

Kumar, S. and Chakravorty, S. (2012). Adaptive sampling for generalized probabilistic roadmaps. *Journal of Control Theory and Applications*, 10(1), 1–10.

LaValle, S.M. (2006). *Planning algorithms.* Cambridge university press.

Lindemann, S.R. and LaValle, S.M. (2005). Current issues in sampling-based motion planning. In *Robotics Research. The Eleventh International Symposium*, 36–54. Springer.

Quinlan, S. and Khatib, O. (1993). Elastic bands: Connecting path planning and control. In *[1993] Proceedings IEEE International Conference on Robotics and Automation*, 802–807. IEEE.

Schulz, J., Hubmann, C., Löchner, J., and Burschka, D. (2018). Interaction-aware probabilistic behavior prediction in urban environments. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3999–4006. IEEE.

Wang, L., Lin, J., and Metzler, D. (2011). A cascade ranking model for efficient ranked retrieval. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, 105–114. ACM.

Zhan, W., Liu, C., Chan, C.Y., and Tomizuka, M. (2016). A non-conservatively defensive strategy for urban autonomous driving. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, 459–464. IEEE.

Zhan, W., Sun, L., Wang, D., Shi, H., Clausse, A., Naumann, M., Kummerle, J., Konigshof, H., Stiller, C., de La Fortelle, A., et al. (2019). Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps. *arXiv preprint arXiv:1910.03088.*