Local trajectory planning for autonomous vehicle with static and dynamic obstacles avoidance

Abdallah Said, Reine Talj, Clovis Francis, Hassan Shraim

Local trajectory planning for autonomous vehicle with static and dynamic obstacles avoidance

Abdallah Said [1,2], Reine Talj [1], Clovis Francis [2] and Hassan Shraim [2]

***Abstract*— Trajectory planning is one of the most complex tasks that should be accomplished in order to ensure vehicle autonomous driving. Trajectory planning can be classified into local and global planning. The purpose of local trajectory planning is to find the optimal trajectory to follow a global reference trajectory while avoiding obstacles in a smooth and comfortable way, within the constraints of road driving. This paper presents a trajectory planning algorithm that calculates a path according to a set of predefined way-points describing a global map. The predefined way-points provide the basic reference frame of a curvilinear coordinate system to generate candidate paths, which start with a transient phase, followed by a curve parallel to the road. Each candidate path, associated to a desired velocity profile, is evaluated via a cost function against several criteria including passenger's comfort, static and dynamic obstacles avoidance and overall trajectory tracking. The chosen trajectory is then applied to a full vehicle model using a coupled longitudinal/lateral controller validated on SCANeR studio (OKtal) simulator. A challenging test scenario of SCANeR studio is used to validate the proposed algorithm under Matlab.**

I. INTRODUCTION

For nearly two decades, autonomous vehicles have been an important area of academic and industrial research that aims to improve vehicles safety and passenger's comfort. Various competitions have been organized around the autonomous vehicle: The $DARPA$ challenges in the United States (2004, 2005, 2007), the Korean competitions for autonomous vehicles, the European $GCDC$ competitions (2013, 2016), and many others. Autonomous driving can be divided into different stages: environment perception and localization, trajectory planning, and vehicle control. This paper presents a local trajectory planning algorithm for autonomous navigation. The planning algorithm must generate a smooth and safe trajectory based on several criteria. To treat this problem, the perception module must provide information about the vehicle's environment, then, by applying the planning algorithm, an optimal trajectory is generated. Finally, the vehicle's controller ensures tracking the chosen trajectory by generating the reliable control laws.

Different trajectory planning approaches have been developed for the navigation of autonomous vehicles. State lattices can be considered as a grids generalization [1] . They are constructed by regularly repeating primitive paths that connect the possible states for the vehicle, in terms of position, curvature or time. The planning problem is then reduced to a boundary value problem. A cost function determines the best path between the pre-calculated networks. State networks [2], [3] overcome the efficiency limitations of network-based techniques without increasing computing power. However, It is not clear how to move the robot from one cell to another adjacent cell when its dynamics are non-linear and complex.

Another approach is sampling-based planning that relies on sampling the configuration space. This randomized approach has the advantage of providing quick solutions to difficult problems. The disadvantage is that the solutions are widely regarded as sub-optimal and The execution time is unpredictable. As a result, the resulting paths tend to be jerky, redundant, and non-curved, which is not acceptable for autonomous driving applications, especially at high speeds. The most commonly used algorithms are the roadmap method (PRM) [4] and Rapid Exploration Trees (RRT) [5] .

Numerical optimization methods, adopted in [6], [7] and [8], aim at minimizing or maximizing a function subject to different constraints by using non-linear optimization techniques to solve the trajectory planning problem where the non-holonomy and differential constraints are represented by a system of equality and inequality such as speed, steering speed, acceleration, jerk, etc. They are widely used in autonomous driving systems for their rapid convergence towards locally optimal solutions; however, they are generally not able to find globally optimal solutions, unless an appropriate initial estimate is provided. In addition, they are very time consuming since the optimization of the function takes place at each planning sampling time.

Starting from a set of points describing a global map, the curve interpolation method seeks to generate trajectories in a certain horizon having a specific geometric shape and responding to one or more conditions such as vehicle dynamics and kinematics, comfort, road shape, continuity of curvature, etc. Each candidate is then evaluated via a cost function taking into account several considerations such as distance and time costs, acceleration and collision verification. Geometric representations of trajectories include polynomials [9], polynomial spirals [10], spline curves [11], Bézier curves [12] and tentacles [13].

In [14], the path planner generates a finite set of paths candidate parallel to the base path with different offset shifts and selects the optimal path based on a discrete optimization approach. It is a fast reactive method that reduces the solution space and is therefore well suited for real-time implementation. In our work, the concept of this method was adopted. Several improvements have been proposed on the

[1]Université de Technologie de Compiègne, CNRS, Heudiasyc (Heuristics and Diagnosis of Complex Systems), CS 60 319, 60 203 Compiègne Cedex, France. Email: (abdallah.said, reine.talj)@hds.utc.fr

[2]Université Libanaise, Faculté de Génie, Centre de Recherche Scientifique en Ingénierie (CRSI), Liban. Email: (cfrancis, hassan.shraim) @ul.edu.lb

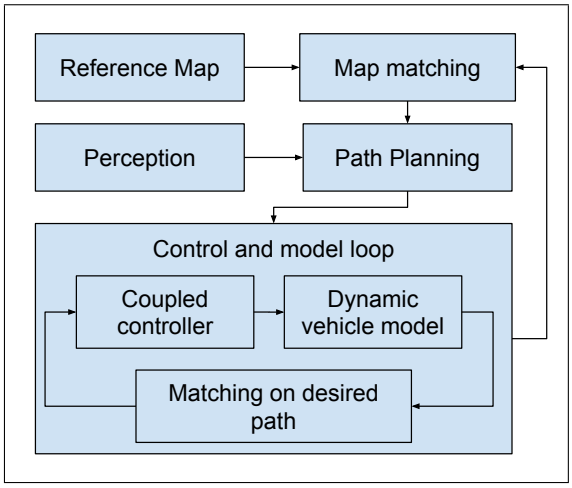


Fig. 1: Functional diagram of the algorithm

method of generating the path candidates and on the selection criteria. Our approach is less conservative due to safety considerations in relation to the longitudinal distance of the obstacle. In addition, a collision checking method has been used to reduce the time required. Our method also handles moving obstacles and multi-lane structured environments. The proposed algorithm was applied on a representative scenario based on real-world driving situations and using a fully controlled vehicle model developed in [19]. We present in Section II the developed navigation strategy and explain how each module works. The proposed planning module is detailed in section III. Section IV reports some simulation results, while the final Section V concludes the paper.

II. Presentation of the Navigation Strategy

The autonomous driving system consists of several complementary modules. Each module receives information, processes it and sends it to another one. In Fig. 1, the functional diagram of the developed system is presented where all the blocks are combined together:

- a reference map block which interpolates the reference trajectory and calculates its characteristics which are used in the generation of candidate paths,
- a map matching block that attempts to match the vehicle position on the reference map,
- a perception block that generates a local occupancy grid taking into consideration the new position of the vehicle and mobile obstacles in order to check the navigability of candidate paths,
- a planning block that generates the candidate paths and corresponding desired velocity profile then selects the best trajectory to be executed,
- a four-wheeled vehicle dynamic model implemented with a coupled longitudinal/lateral controller and a sub block to match the vehicle position on the desired trajectory.

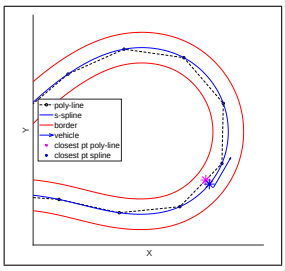
Since path planning is the goal of this paper, it is detailed in the section III.

A. Reference Map

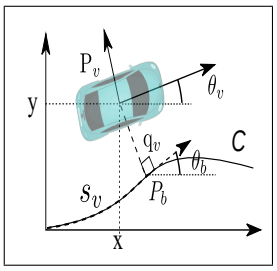
Generally, a road is represented by a sequence of points and modeled by a parametric curve to define the trajectory of the movement. There are different approaches, such as bezier curve, poly-line model, lanlet model, cubic spline and many others [16]. In our work, we use Piece-wise cubic spline curves for map interpolation based on the given way-points P_{bf} to ensure the continuity of the curvature profile and to avoid discontinuities and differentiation problems when passing over one of the way-points. However, it is essential to efficiently link the values of the spline curve parameters to the curvilinear abscissa s in order to simplify and improve the planning and trajectory tracking performance of the vehicle (see [15]). A series of arc length parameterized cubic spline curves $P(s) = (x(s), y(s))$ are generated. Using these curves equations, the curvature profile $\rho_{P_{bf}}$ is calculated first. Then, the base frame velocity profile $V_{P_{bf}}$ is calculated based on the desired speed and road speed limit that guarantees the stability of the vehicle.

B. Map matching

The correspondence of the vehicle's position on the reference map is an unavoidable step. It allows to identify the curvilinear coordinates of the vehicle (or any interesting point) from the Cartesian coordinates in order to use the reference information. In order to make the mapping neither computationally demanding nor difficult to be implemented in real-time embedded systems, we proposed a composite algorithm that starts with the poly-line mapping method [16] to find a rough estimate that serves as an initial guess for the quadratic minimization method in [17] (Fig. 2a). It consists in finding on the reference trajectory C, the closest point P_b to the vehicle P_v. At the end, in a local curvilinear coordinate system ($\overrightarrow{s}$, $\overrightarrow{q}$), the coordinates (s_v, q_v) of the vehicle P_v are obtained (Fig. 2b).



(a) Correspondence on the map



(b) Curvilinear coordinates

Fig. 2: Map matching

C. Perception

To avoid collisions and maintain safe conditions, a candidate path must be checked to determine whether or not it crosses roadsides or obstacles. The perception block is out of the scope of this paper. The present work considers the output of the perception block as an occupancy grid representation. The occupancy grid is a discrete and metric representation of the surrounding environment, where it is represented by

a set of square cells. Black cells correspond to a navigable area, and white one may belong to the roadside or obstacles. However, for validation by simulation, the occupancy grids are generated in $Matlab$. Starting from a global map (Fig. 3), a global occupancy grid is generated (Fig. 4). The local

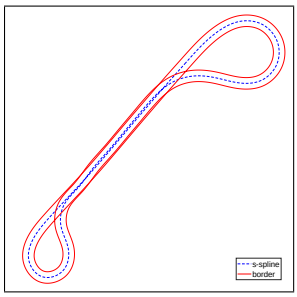


Fig. 3: Reference map

Fig. 4: Global occupancy grid

occupancy grid is obtained from the global occupancy grid according to the position and orientation of the vehicle to obtain the local occupancy grid. This latter is constituted of 400 * 400 cells, each of which measures 25 * 25 cm (Fig. 6). These dimensions are related to the horizon of the perception system. At each planning iteration, the local occupancy grid is updated by the new footprint occupied by the moving obstacles existing in its perception zone. However, to take their velocity into consideration, a longitudinal widening of their occupancy is carried out in the grid (Fig. 5). It's equal to the predicted traveled distance during one iteration.

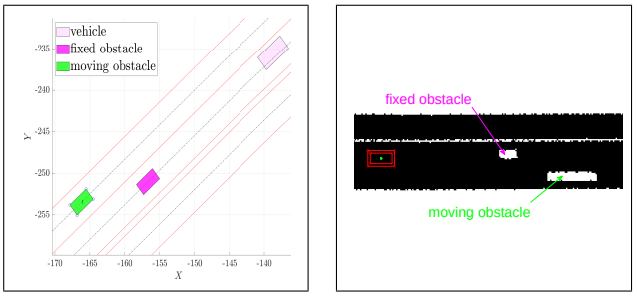


Fig. 5: Vehicle and obstacles occupancy

In order to improve the accuracy, efficiency and time consumption for collision checking, the local occupancy grid (Fig. 6) is transformed into a clearance map as shown in Fig. 7. This map represents the distance to the nearest obstacle using the method explained in [18]. The validation of a circular area of the grid is then transformed into a single-point validation: if the distance from the center of a circle to the nearest obstacle is greater than its radius, then there is no collision, and vice-versa.

D. Controlled vehicle dynamic model

A full longitudinal and lateral vehicle model is implemented in our algorithm. It has been developed using the multi-body formalism in [19]. The model inputs are the wheel driving/braking torque τ_w and the steering angle δ. Using Dugoff model to estimate tires forces and model matrix calculations, the model outputs are calculated: the

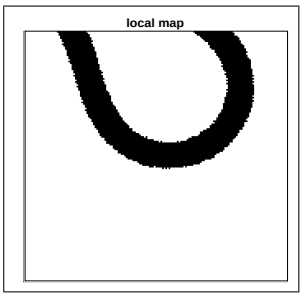


Fig. 6: Local occupancy grid

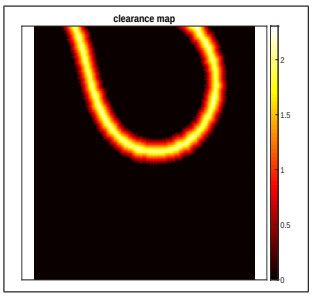


Fig. 7: Clearance map

longitudinal ($\ddot{x}$) and lateral ($\ddot{y}$) vehicle accelerations, the yaw rate ($\dot{\psi}$), and the wheels angular velocities ($\dot{w}_{ij}$). For control purposes, we choose to proceed with a higher order sliding mode (second order) based on the super-twisting algorithm to ensure robust stability while reducing chattering. The full model has been validated in SCANeR studio simulator, under several driving conditions.

III. Path Planning

Our path planning algorithm must provide the optimal path from a set of paths that helps the vehicle to track a reference trajectory while avoiding static and mobile obstacles and ensuring the safety and comfort of passengers. The navigation strategy is composed of several stages: generation of a set of candidate paths, obstacle detection and paths classification according to their navigability, costs calculation and selection of the optimal one from this set of candidate paths. This selected trajectory is considered as the desired trajectory to be executed by the vehicle at each planning iteration. The entire procedure of the path planning is shown in Fig. 8 and explained in the following.

A. Generation of candidate paths

The objective is to generate candidate paths while avoiding obstacles shifted by a lateral offset from the reference trajectory (Base Frame). The planning algorithm generates a finite number of candidates, each of them having

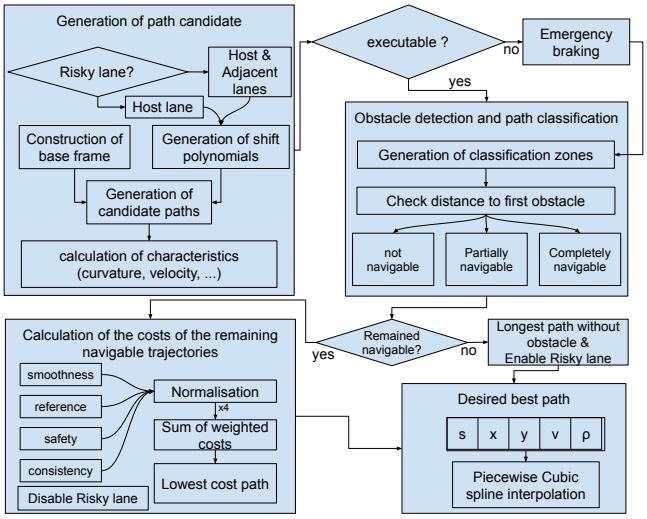


Fig. 8: Path planning diagram

a separate lateral offset q_f to cover the entire width of the lane (Fig. 9). The generation of these candidate paths consists of two stages: the generation of candidate path in the coordinate system ($\overrightarrow{s}$, $\overrightarrow{q}$) in order to facilitate their representation and then their transformation into Cartesian coordinates in the global coordinate system ($\overrightarrow{X}$, $\overrightarrow{Y}$).

In the curvilinear coordinate system ($\overrightarrow{s}$, $\overrightarrow{q}$):

The candidate path starts from the last executed point P_{des}^{t-1} of the desired trajectory at the previous planning iteration to ensure its continuity. It consists of a transient phase to reach the offset already defined q_f, followed by a permanent phase where it will be parallel to the road to continue with this offset. The P_{des}^{t-1} is matched on the global map, so the curvilinear coordinate ($s_{P_{des}^{t-1}}$, $q_{P_{des}^{t-1}}$) and other characteristics of the first corresponding point P_{bf}^0 on the base frame of the new candidate path are obtained (Fig. 10). First, from the width of the lane (the vehicle width is subtracted to avoid the generation of candidate paths near borders) and the desired lateral sampling resolution Δq, the number of last points (s_f^i, q_f^i) on the candidate paths are determined. The arc length s_f is that traveled by the candidate path all along the reference trajectory. It consists of two phases, s_{f1} for transient phase and s_{f2} for permanent phase. It is proportional to the speed of the vehicle v_V with the gain K_v, bounded by the minimum Δs_{f-min} and the maximum l_{max} acceptable length and given by:

$$\begin{aligned} s_{f1} &= \Delta s_{f1-min} + K_{v1} V_v, \\ s_{f2} &= max(\Delta s_{f2-min}, 2\ d_{ss} - s_{f1}) \\ s_f &= min(l_{max}, s_{f1} + s_{f2}) \end{aligned} \tag{1}$$

where the safe stop distance d_{ss} is the distance required to stop the vehicle traveling at a speed V_v with the maximum longitudinal deceleration $a_x^{dec-max}$ and defined by:

$$d_{ss} = d_{ss0} + \frac{V_v^2}{2a_x^{dec-max}} \tag{2}$$

where d_{ss0} is a minimum safety gap.

Next, for each of candidate path, a fourth order polynomial (eq. 3) describes the lateral shift polynomial $q_M^i(s)$ to provide a smooth change (Fig. 9) ensuring the $\mathscr{C}^2$ continuity while avoiding the discontinuity of the curvature with respect to the desired one in the last iteration.

$$q_M^i(s) = \begin{cases} a_0^i + a_1^i \Delta s + a_2^i \Delta s^2 & s_{P_{des}^{t-1}} \leqslant s \leqslant s_{f1}, \\ \quad + a_3^i \Delta s^3 + a_4^i \Delta s^4, & \\ q_f^i, & s_{f1} \leqslant s \leqslant s_f \end{cases} \tag{3}$$

where $\Delta s = s - s_{P_{des}^{t-1}}$. The coefficients of fourth order polynomials a_i are determined based on the boundary conditions below:

$$\begin{aligned} q^i(s_{P_{des}^{t-1}}) &= q_{P_{des}^{t-1}}, \quad \frac{\partial q^i}{\partial s}(s_{P_{des}^{t-1}}) = \tan(\theta_{P_{des}^{t-1}} - \theta_{bf}), \\ \rho^i(s_{P_{des}^{t-1}}) &= \rho_{P_{des}^{t-1}}, \\ q^i(s_{f1}^i) &= q_f^i, \quad \frac{\partial q^i}{\partial s}(s_{f1}^i) = 0 \end{aligned} \tag{4}$$

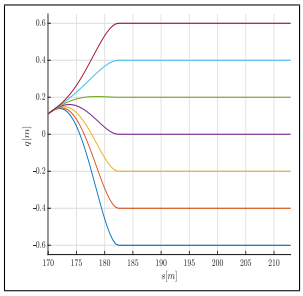


Fig. 9: Lateral offset polynomial in ($\overrightarrow{s}$, $\overrightarrow{q}$)

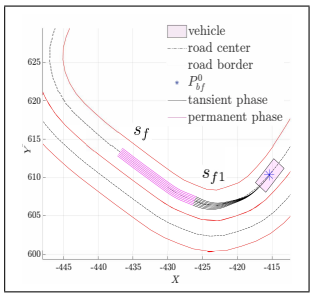


Fig. 10: Candidate paths in ($\overrightarrow{X}$, $\overrightarrow{Y}$)

The analytic expression of the coefficients a_i according to the boundary conditions are calculated offline using the function $Matlab$ ($SOLVE$) since there is an implicit boundary condition.

In the global coordinate system ($\overrightarrow{X}$, $\overrightarrow{Y}$):

The candidate paths are generated in the curvilinear coordinate system, but the planning results should be converted to the Cartesian coordinate system approved by the maneuver system (Fig. 10).

First, from P_{bf}^0, a base frame trajectory is constructed along the reference trajectory starting from $s_{P_{des}^{t-1}}$ and ending on s_f, then its characteristics are calculated (curvature $\rho_{P_{bf}}$, limit velocity $V_{P_{bf}}^{limit}$, desired velocity $V_{P_{bf}}^{des}$ and normal vector $\overrightarrow{N_{P_{bf}}}$).

Second, for each candidate path, we proceed, point by point, by the transformation between the Cartesian coordinate system and the curvilinear coordinate system. The trajectories are deduced as follows:

$$\overrightarrow{OM^i(s)} = \overrightarrow{OP_{bf}} + q^i(s)\overrightarrow{N_{P_{bf}}} \tag{5}$$

If the point M belongs to a circle around the vehicle (P_v, L_{PA}) which represents the perception area of the vehicle, we calculate its characteristics. First, we calculate the curvature of each point on the candidate path ρ_M^i as follows:

$$\rho_M^i = \frac{S_M^i}{{Q_M^i}^2} \left(\rho_{P_{bf}} + \frac{(1 - q^i \rho_{P_{bf}}) \frac{\partial^2 q^i}{\partial s^2} + \rho_{P_{bf}} \frac{\partial q^i}{\partial s}^2}{{Q_M^i}^2} \right) \tag{6}$$

with:

$$S_M^i = sgn(1 - q^i \rho_{P_{bf}}), Q_M^i = \sqrt{\frac{\partial q^i}{\partial s}^2 + (1 - q^i \rho_{P_{bf}})^2}. \tag{7}$$

Now, we check if the candidate path is executable by the vehicle due to the steering limitation: ρ_{max} is the curvature corresponding to the maximum steering angle that can be executed by the vehicle. So, if at any point M of the candidate path, $\rho_M > \rho_{max}$, the candidate path is considered as not executable and it is rejected from the available set of paths. If all candidate paths are not executable, an emergency braking mode is activated: the planning algorithm selects the longest navigable path providing the maximum distance to the obstacle, (III-B), and brake with high deceleration and zero desired speed.

Determination of the desired velocity profile:

The vehicle must follow a predefined velocity profile or brakes in case of obstacle detection or no executable trajectory. Starting from desired velocity profile of the base frame $V_{P_{bf}}^{des}$ and for each point of candidate path, we calculate the desired initial velocity at a point M for the candidate i, ${V_M^{des0}}^i$, and its limit ${V_M^{limit}}^i$ imposed by the velocity limit of the base frame $V_{P_{bf}^{limit}}$ as follows:

$$ {V_M^{des0}}^i = S_M^i Q_M^i V_{P_{bf}}^{des}, \qquad {V_M^{limit}}^i = S_M^i Q_M^i V_{P_{bf}^{limit}} \quad (8) $$

Another limitation ${V_M^{\rho}}^i$ is imposed by the lateral acceleration to improve vehicle stability and passenger comfort:

$$ {V_M^{\rho}}^i = \sqrt{\frac{a_y^{max}}{\rho_M^i}} \quad (9) $$

where a_y^{max} is the maximum acceptable lateral acceleration in the comfort zone.

Finally, the desired velocity profile of each point on the candidate path ${V_M^{des}}^i$ is calculated as follows:

$$ {V_M^{des}}^i = \min({V_M^{limit}}^i, {V_M^{\rho}}^i, {V_M^{des0}}^i) \quad (10) $$

Due to perception limit, the safe stop distance must be less than the perception horizon of the vehicle. So, the desired velocity is also maximized by the value calculated using equation (2) for $d_{ss} = L_{PA}$.

B. Obstacle detection and paths classification

For the rest of the executable candidate paths, an obstacle detection procedure is carried out. So, on each point of this path and as described in the sub-Section II-C, we validate the footprint of the vehicle represented by several circles as shown in Fig. 11: a large circle and six other small circles.

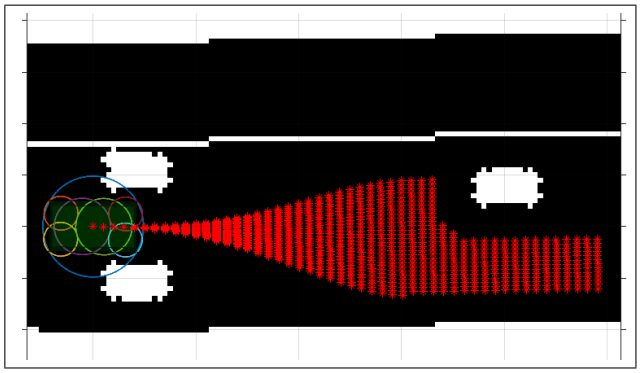


Fig. 11: Example of collision checking

First, an evaluation of the large circle is carried out: if there are no obstacles, we pass to the next point on the path (Fig. 11). In the case where an obstacle is present, a more precise validation is carried out based on the test of the 6 small circles. Once the first obstacle is detected, the free distance traveled on the path to reach the obstacle is called the collision distance d_{obs}. The obstacle type, fixed or moving, is also determined and is also associated with its speed V_{obs}, if it is a moving obstacle. The next step is to classify the trajectory based on the collision distance into three classes as follows:

$$ \begin{cases} d_{obs}^i < L_s & \rightarrow no\ navigable\ trajectory \\ L_s < d_{obs}^i < l_{max}^i & \rightarrow partially\ navigable\ trajectory \\ d_{obs}^i \geq l_{max}^i & \rightarrow navigable\ trajectory \end{cases} \quad (11) $$

where l_{max}^i is the length of corresponding candidate path and L_s is the adapted security distance [20] that the vehicle must keep with the vehicles that precede it and is calculated using the following formula :

$$ L_s = \begin{cases} d_{ss}, & \text{for fixed obstacle} \\ d_{ss0} + V_f \Delta t + \frac{V_f^2 - V_p^2}{2a_x^{dec-max}}, & \text{for mobile obstacle} \end{cases} \quad (12) $$

where Δt is the average reaction time of autonomous vehicles, V_f is the following vehicle's speed and V_p is the preceding vehicle's speed.

If there is no navigable paths on the reference lane, the generation of an additional set of paths candidate is performed on the adjacent lanes (Fig. 12). Note that braking is performed whenever there is no navigable paths on all the lanes.

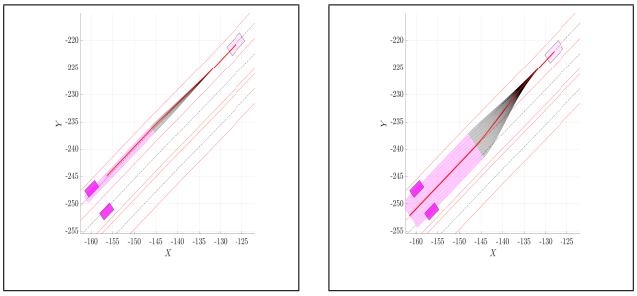


Fig. 12: Generation of path candidate on one and two lanes

C. Selection of the desired trajectory

The desired path is either the optimal one from the executable and navigable set of paths. Or, if there are none, the path who has the greatest distance to the obstacle is chosen to brake the vehicle. Among all remained candidate paths, we seek to choose the optimal trajectory to execute it. These trajectories are evaluated according to different criteria including smoothness, safety, consistency with the initial state of the vehicle and the reference trajectory tracking. Once these criteria have been estimated and the corresponding cost function is evaluated, they are normalized, weighted and combined in a linear manner within a cost function which we seek to minimize.

Smoothness cost $C_K[i]$: Low smoothness induces additional lateral acceleration which degrades passenger comfort. The lateral acceleration of the vehicle is related to the curvature especially at constant speed (eq. 9). So, to reduce the slackness of a path, the integration of the curvature squared along a trajectory i is chosen as a criterion of smoothness for this trajectory as follows:

$$ C_K[i] = \int {\rho_M^i}^2 \, ds_M^i = \int {\rho_M^i}^2 \, Q_M^i \, ds \quad (13) $$

where ds^i_M is the curvilinear abscissa along the i path and ds is that along the base frame.

Consistency cost $C_c[i]$: A sudden change in the trajectory requires excessive energy and an extra effort of control. Therefore, to avoid generating a trajectory that is significantly different from that of the previous step, a measure of similarity between the previous trajectory and the current candidate path is required. We simplify it in our algorithm to the difference between the index of the current path i and that of desired path at previous planning iteration.

$$C_c[i] = |i - i^{t-1}_{des}| \tag{14}$$

Cost of tracking the reference trajectory $C_r[i]$: This criterion leads the vehicle to follow the global reference trajectory in order to position itself in the center of the desired lane. As the candidate paths are generated around this center by a defined lateral offset, the cost of following the reference trajectory is proportional to this offset. The further you go, the more the cost increases and is given by:

$$C_r[i] = d^2_{i-bf} \tag{15}$$

where d_{i-bf} is the lateral distance between the path i and the base frame.

safety cost $C_s[i]$: we define two safety costs: a longitudinal cost C_{sl} and a lateral cost C_{sL}. The first one measures the navigability of the trajectory itself and how far the obstacle is, while the lateral one takes into account the navigability of the adjacent trajectories. The collision distance d_{obs} (11), is transformed into a safety cost as follows:

$$C_{sl}[i] = \begin{cases} 0 & \textit{obstacle-free trajectory} \\ 2 - \frac{2}{1+e^{-c_1\ d^i_{obs}}} & \textit{otherwise} \end{cases} \tag{16}$$

$$g[k] = \frac{1}{\sqrt{2\pi}\sigma}\ exp\Big(-\frac{(k\ \Delta q)^2}{2\sigma^2}\Big) \tag{17}$$

$$C_{sL}[i] = \frac{\sum_{j\in\Gamma^i} C_{sl}[j]\ g[i-j]}{n_{\Gamma^i}} \tag{18}$$

$$C_s[i] = C_{sl}[i] + W_{SlL} C_{sL}[i] \tag{19}$$

where c_1 is a cost setting parameter, Δq is the desired lateral sampling resolution, σ is the standard deviation of the discrete Gaussian convolution for the risk of collision and is calculated by considering that $g[\frac{L_v}{\Delta q}] = 0.5$. Γ^i is the set of indexes of executable paths without i, n_{Γ^i} is their number and W_{SlL} is a weighting coefficient between the two costs.

Total cost and optimal path selection:
As our different costs criteria are measured in different units, we use the normalization technique to make them dimensionless. Hence, we apply the following formula:

$$\overline{C_.}[i] = \frac{C_.[i] - min(C_.)}{max(C_.) - min(C_.)} \tag{20}$$

Then, the total cost function $C_T[i]$ of a given trajectory is defined as the weighted sum of the normalized cost functions defined above:

$$C_T[i] = W_K\overline{C_K}[i] + W_c\overline{C_c}[i] + W_r\overline{C_r}[i] + W_s\overline{C_s}[i] \tag{21}$$

with W_K, W_c, W_r and W_s are the weighting coefficients of smoothness, consistency, reference tracking and safety respectively.

At this final step, the selected trajectory is represented by a series of points defined by the curvilinear abscissa, x and y coordinates, velocity and curvature. This trajectory is interpolated using the same method explained in II-A to facilitate the vehicle's matching in the control/model block.

IV. Simulation results

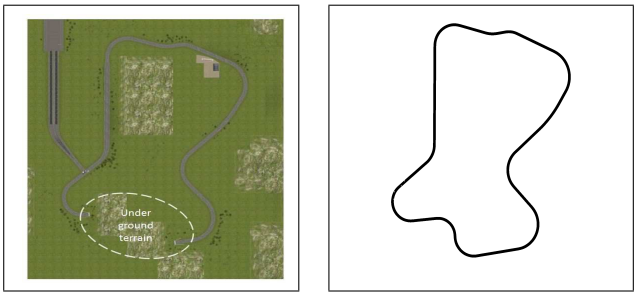


Fig. 13: $SCANeR\ studio$: global map

To validate our planning algorithm, it was applied to a fairly realistic scenario taken from SCANeR studio simulator. Several fixed and moving obstacles are introduced so as to cover several commonly encountered driving scenarios. It was simulated in $MATLAB$ by taking into account the vehicle's dynamics. The vehicle block (model and controller) is running at a frequency of 50 Hz, and the rest of the system is running at 5 Hz. At each simulation step, the local occupancy grid is updated based on the new position of the vehicle and mobile obstacles are also added if there are any.

To show the flexibility and the safety aspects of the presented approach, we present in Fig. 14 a one challenging test scenario, instead of several simple scenarios, which shows:

- passing through a tight passage
- overtaking fixed obstacles
- lane change maneuver
- vehicle speed adaptation
- overtaking a moving obstacle

The trajectory of the vehicle is represented by its colored footprint relative to its instantaneous longitudinal speed.

As shown in Fig. 14, the vehicle starts traveling along the center road at speed of 30 km/h. It passes through a tight passage (t=4s) to avoid static obstacles in a straight road then accelerate to reach the desired speed. Or, when entering the curvy road, the vehicle encounter a fixed obstacle blocking the lane and a mobile obstacle on the other lane (t=8s), so the planning algorithm slows down the vehicle, leaving the way for the moving obstacle (t=10s). Then, the vehicle is accelerated to pass the fixed obstacle (t=15s). At t=18s, we see an overtaking of a mobile obstacle traveling at $v_{obs} = 25km/h$ (t=18s). The overtaking and returning maneuvers

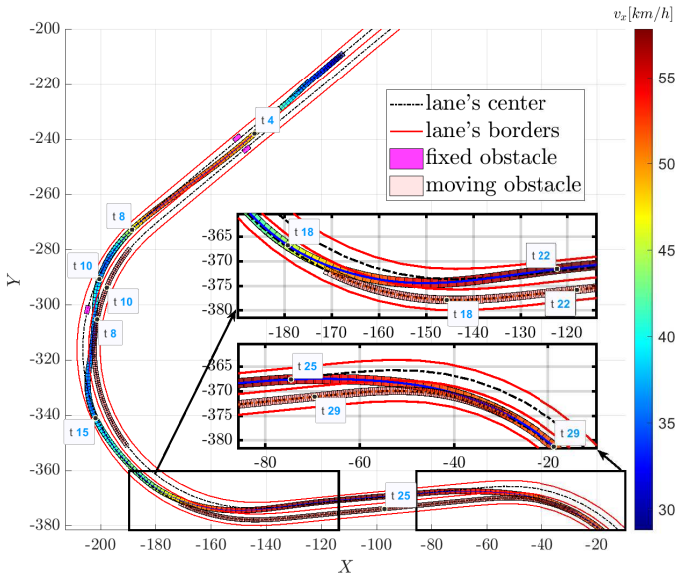


Fig. 14: Simulation results: trajectories and vehicle's speed

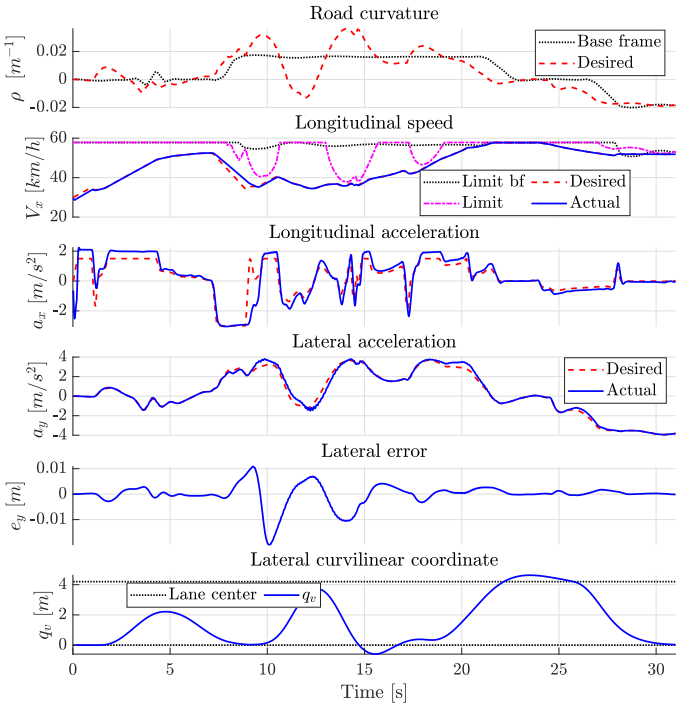


Fig. 15: Simulation results: vehicle dynamic variables

are shown in the boxes. The overtaking is permissible while the speed difference between vehicle and obstacle is greater than 20 km/h. The planner start overtaking (t=22s) after approaching from obstacle to reach the security distance (eq.12) and selects a smooth and comfortable path on the left lane to pass the mobile obstacle while increasing its speed. The vehicle reach the second lane's center at t=22s and travel along to reach a sufficient safety distance (t=25s) to start returning to the host lane. At t=29s, the vehicle succeeds to reach the center of the road and travel at the desired speed while entering a curvy road.

In Fig. 15, the base frame curvature profile is shown in dotted line. Smooth motion can be seen from the generated continuous curvature profile ensured by the order 4 of the lateral offset polynomial. We see also the longitudinal speed profile. The base frame limit speed is around 57 km/h. Based on selected trajectories and driving situations, the desired speed profile is shown in dashed line. Its limit based on comfort acceleration, shown in Dash-dot line, is well respected, especially at t=10, 15 and 18s in the three different maneuvers. The vehicle is traveling at the desired speed with an acceptable longitudinal speed error and track perfectly the maximal acceptable speed between t=22 and 25s when passing mobile obstacle on the second lane. Note that the longitudinal acceleration is suitable for comfortable travel and limited between $\pm 1.5 m/s^2$ for comfort travel and can reach $-3 m/s^2$ in case of emergency braking (t=8s). From the lateral acceleration profile, we can deduce that the vehicle stability is ensured, which improves passenger comfort. The maximal lateral acceleration ($\pm 4 m/s^2$) reached at t=10, 15, 20 and 29s in the different scenarios shows the performance of our proposed method on the limit of stability. Also, the small lateral error shows that the vehicle is able to follow accurately the trajectory generated by the planning algorithm during the entire course. The lateral curvilinear coordinate is presented to show its smooth variation. The vehicle position in the mobile obstacle overtaking maneuver is displayed between t=18 and 29s.

V. CONCLUSION

A reactive trajectory planning algorithm for on-road navigation has been developed. It's able to follow a reference trajectory while passing fixed and mobile obstacles. Within the road driving rules and vehicle's dynamic constraints, the velocity is adapted to ensure passenger's comfort. Many other scenarios are tested and the results show good behavior in rather complicated driving situations. In the future, we look to implement our planning algorithm on a real experimental vehicle in *Heudiasyc* laboratory and to optimize the cost function coefficients in eq. 21 in order to take into consideration the driving situation in an optimal way (comfort oriented, stability, reference tracking, trajectory smoothness,...).

REFERENCES

[1] PIVTORAIKO, Mihail et KELLY, Alonzo. Efficient constrained path planning via search in state lattices. In International Symposium on Artificial Intelligence, Robotics, and Automation in Space. Germany, Munich, 2005. p. 1-7.

[2] HOWARD, Thomas M. et KELLY, Alonzo. Optimal rough terrain trajectory generation for wheeled mobile robots. The International Journal of Robotics Research, 2007, vol. 26, no 2, p. 141-166.

[3] ZIEGLER, Julius et STILLER, Christoph. Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios. In 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2009. p. 1879-1884.

[4] PLAKU, Erion, BEKRIS, Kostas E., CHEN, Brian Y., et al. Sampling-based roadmap of trees for parallel motion planning. IEEE Transactions on Robotics, 2005, vol. 21, no 4, p. 597-608.

[5] KARAMAN, Sertac et FRAZZOLI, Emilio. Sampling-based algorithms for optimal motion planning. The international journal of robotics research, 2011, vol. 30, no 7, p. 846-894.

[6] WERLING, Moritz, ZIEGLER, Julius, KAMMEL, Sören, et al. Optimal trajectory generation for dynamic street scenarios in a frenet frame. In 2010 IEEE International Conference on Robotics and Automation. IEEE, 2010. p. 987-993.

[7] LIM, Wonteak, LEE, Seongjin, SUNWOO, Myoungho, et al. Hierarchical trajectory planning of an autonomous car based on the integration of a sampling and an optimization method. IEEE Transactions on Intelligent Transportation Systems, 2018, vol. 19, no 2, p. 613-626.

[8] ZIEGLER, Julius, BENDER, Philipp, DANG, Thao, et al. Trajectory planning for Bertha—A local, continuous method. In 2014 IEEE intelligent vehicles symposium proceedings. IEEE, 2014. p. 450-457.

[9] LEE, Jin-Woo et LITKOUHI, Bakhtiar. A unified framework of the automated lane centering/changing control for motion smoothness adaptation. In 2012 15th International IEEE Conference on Intelligent Transportation Systems. IEEE, 2012. p. 282-287.

[10] McNaughton Matthew. Parallel Algorithms for Real-time Motion Planning (Doctoral dissertation, Carnegie Mellon University).

[11] WANG, Miao, GANJINEH, Tinosch, et ROJAS, Raúl. Action annotated trajectory generation for autonomous maneuvers on structured road networks. In The 5th International Conference on Automation, Robotics and Applications. IEEE, 2011. p. 67-72.

[12] YANG, Kwangjin et SUKKARIEH, Salah. An analytical continuous-curvature path-smoothing algorithm. IEEE Transactions on Robotics, 2010, vol. 26, no 3, p. 561-568.

[13] CHEBLY, Alia, TAGNE, Gilles, Talj, Reine, et al. Local trajectory planning and tracking of autonomous vehicles, using clothoid tentacles method. In 2015 IEEE intelligent vehicles symposium (IV). IEEE, 2015. p. 674-679.

[14] CHU, Keonyup, LEE, Minchae, et SUNWOO, Myoungho. Local path planning for off-road autonomous driving with avoidance of static obstacles. IEEE Transactions on Intelligent Transportation Systems, 2012, vol. 13, no 4, p. 1599-1616.

[15] WANG, Hongling, KEARNEY, Joseph, et ATKINSON, Kendall. Arc-length parameterized spline curves for real-time simulation. In Proc. 5th International Conference on Curves and Surfaces. 2002.

[16] HÉRY, Elwan, MASI, Stefano, XU, Philippe, et al. Map-based curvilinear coordinates for autonomous vehicles. In 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2017. p. 1-7.

[17] WANG, Hongling, KEARNEY, Joseph, et ATKINSON, Kendall. Robust and efficient computation of the closest point on a spline curve. In Proceedings of the 5th International Conference on Curves and Surfaces. 2002. p. 397-406.

[18] FELZENSZWALB, Pedro F. et HUTTENLOCHER, Daniel P. Distance transforms of sampled functions. Theory of computing, 2012, vol. 8, no 1, p. 415-428.

[19] CHEBLY, Alia, TALJ, Reine, et CHARARA, Ali. Coupled longitudinal/lateral controllers for autonomous vehicles navigation, with experimental validation. Control Engineering Practice, 2019, vol. 88, p. 79-96.

[20] Althoff, Matthias, and Robert Lösch. "Can automated road vehicles harmonize with traffic flow while guaranteeing a safe distance?." In 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), pp. 485-491. IEEE, 2016.