Path and velocity trajectory selection in an anticipative kinodynamic motion planner for autonomous driving

Jordi Pérez Talamino and Alberto Sanfeliu

Institut de Robòtica i Informàtica Industrial, (CSIC-UPC). Parc Tecnològic de Barcelona, 08028, Barcelona, Spain
joperez@iri.upc.edu, sanfeliu@iri.upc.edu,
http://www.iri.upc.edu/

Abstract. This paper presents an approach for plan generation, selection and pruning of trajectories for autonomous driving, capable of dealing with dynamic complex environments, such as driving in urban scenarios. The planner first discretizes the plan space and searches for the best trajectory and velocity profile of the vehicle. The main contributions of this work are the use of G^2-splines for path generation and a method that takes into account accelerations and passenger comfort for generating and pruning velocity profiles based on 3rd order splines, both fulfilling kinodynamic constraints. The proposed methods have been implemented in a motion planner in MATLAB and tested through simulation in different representative scenarios, involving obstacles and other moving vehicles. The simulations show that the planner performs correctly in different dynamic scenarios, maintaining the passenger comfort.

Keywords: autonomous driving, urban, anticipation, kinodynamic motion planning, path planning, G^2-splines, velocity profiles.

1 Introduction

Nowadays autonomous driving is becoming more and more popular due to its numerous benefits, and motion planning is a key element [1] [2]. Motion planners need to generate candidate geometric paths that will be followed by a low level control system. Popular techniques, such as Bézier curves, present a high complexity generation from the road shape, with the presence of non-intuitive geometric waypoints and multitude of parameters [3].

On the other hand, velocity profile generation methods often use trapezoidal profiles due to their simplicity despite its dynamic limitations because they have discontinuities in the acceleration [4]. Better approaches compute spline-based velocity trajectories over time, fixing the total maneuver time [5] or also over the road length [6]. Nevertheless, these time conditions make the discretization process difficult and lead even to the need of a post-optimization.

We explain in section 2 of this article the anticipative kinodynamic motion planner that is used; in section 2.1, the G^2-splines for path generation; in section 2.2, the velocity profile generation; and in section 2.3 the cost structure. Finally in section 3 we explain the simulations and in section 4, the conclusions.

2 Anticipative kinodynamic motion planner framework

Fig. 1 shows the general scheme of the proposed motion planner.

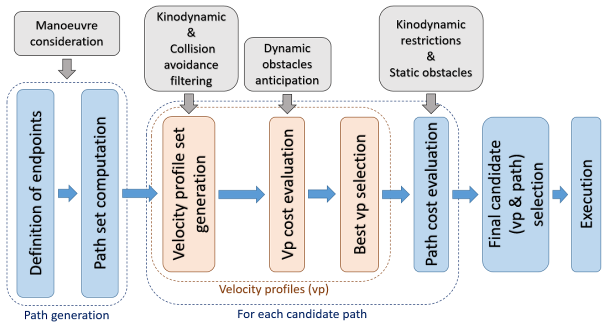

Fig. 1. Motion planner framework

First of all, the planner discretizes the environment choosing several endpoints, which are state configurations $\mathbf{X} = [x, y, \theta, \kappa]$, where x and y are the position coordinates, θ is the heading and κ is the curvature, which is related with the steering wheel angle using the bicycle kinematic model.

Then candidate paths are generated from the host vehicle state to the endpoints (section 2.1). The motion planner uses a novel approach to generate paths, G^2-splines [7]. This method only needs a basic geometric state $\mathbf{X}$ for the initial and the final points. It can approximate any kind of path shape preserving a smooth curvature continuity and minimizing curvature variability, and this implies kinematic feasibility for a vehicle following it: circular segments, straight lines, clothoids, complete lane changes, etc.

Once the planner has a set of candidate paths, for each one of them it computes a set of velocity profiles in order to anticipate dynamic obstacles (section 2.2). Each velocity profile has an associated cost and the one with the minimum cost is chosen, for a certain path. Finally, all the candidate paths with its associated velocity profiles are compared with a cost structure (section 2.3) and the minimum cost solution (path and velocity profile) is chosen to be the executed one. The velocity profiles are computed using 3rd order splines,taking into account the dynamic restrictions of the candidate path and the road path shape, and they also are kinematically validated.

The output of the proposed motion planner is a kinematically and dynamically feasible path with an associated velocity profile for the host vehicle, also fulfilling comfort restrictions for the passengers, such as bounded accelerations and jerk.

2.1 G^2-splines path generation

The G^2-*splines* are geometric polynomials of 5th order presenting second order geometric continuity (G^2), so the curvature κ is continuous [7]. In order to build a general path $\mathbf{p}(u)$, understood as a path with arbitrary defined starting endpoint $\mathbf{X}_A = [x_A, y_A, \theta_A, \kappa_A]$, and ending endpoint $\mathbf{X}_B = [x_B, y_B, \theta_B, \kappa_B]$, the equations to define these splines are:

$$\mathbf{p}(u) = \begin{bmatrix} x(u) \\ y(u) \end{bmatrix} := \begin{bmatrix} x_0 + x_1 u + x_2 u^2 + x_3 u^3 + x_4 u^4 + x_5 u^5 \\ y_0 + y_1 u + y_2 u^2 + y_3 u^3 + y_4 u^4 + y_5 u^5 \end{bmatrix} \tag{1}$$

where $u \in [0,1]$ and:

$$\begin{aligned}
x_0 &= x_A \\
x_1 &= \eta_1 \cos\theta_A \\
x_2 &= \frac{1}{2}(\eta_3 \cos\theta_A - \eta_1^2 \kappa_A \sin\theta_A) \\
x_3 &= 10(x_B - x_A) - (6\eta_1 + \frac{3}{2}\eta_3)\cos\theta_A - (4\eta_2 - \frac{1}{2}\eta_4)\cos\theta_B + \\
&\quad + \frac{3}{2}\eta_1^2 \kappa_A \sin\theta_A - \frac{1}{2}\eta_2^2 \kappa_B \sin\theta_B \\
x_4 &= -15(x_B - x_A) + (8\eta_1 + \frac{3}{2}\eta_3)\cos\theta_A + (7\eta_2 - \eta_4)\cos\theta_B - \\
&\quad - \frac{3}{2}\eta_1^2 \kappa_A \sin\theta_A + \eta_2^2 \kappa_B \sin\theta_B \\
x_5 &= 6(x_B - x_A) - (3\eta_1 + \frac{1}{2}\eta_3)\cos\theta_A - (3\eta_2 - \frac{1}{2}\eta_4)\cos\theta_B + \\
&\quad + \frac{1}{2}\eta_1^2 \kappa_A \sin\theta_A - \frac{1}{2}\eta_2^2 \kappa_B \sin\theta_B \\
y_0 &= y_A \\
y_1 &= \eta_1 \sin\theta_A \\
y_2 &= \frac{1}{2}(\eta_3 \sin\theta_A + \eta_1^2 \kappa_A \cos\theta_A) \\
y_3 &= 10(y_B - y_A) - (6\eta_1 + \frac{3}{2}\eta_3)\sin\theta_A - (4\eta_2 - \frac{1}{2}\eta_4)\sin\theta_B - \\
&\quad - \frac{3}{2}\eta_1^2 \kappa_A \cos\theta_A + \frac{1}{2}\eta_2^2 \kappa_B \cos\theta_B \\
y_4 &= -15(y_B - y_A) + (8\eta_1 + \frac{3}{2}\eta_3)\sin\theta_A + (7\eta_2 - \eta_4)\sin\theta_B + \\
&\quad + \frac{3}{2}\eta_1^2 \kappa_A \cos\theta_A - \eta_2^2 \kappa_B \cos\theta_B \\
y_5 &= 6(y_B - y_A) - (3\eta_1 + \frac{1}{2}\eta_3)\sin\theta_A - (3\eta_2 - \frac{1}{2}\eta_4)\sin\theta_B - \\
&\quad - \frac{1}{2}\eta_1^2 \kappa_A \cos\theta_A + \frac{1}{2}\eta_2^2 \kappa_B \cos\theta_B
\end{aligned}$$

The resulting path depends on the parameter vector $\boldsymbol{\eta}$ that affect its shape, $\boldsymbol{\eta} = [\eta_1, \eta_2, \eta_3, \eta_4]$.

An important characteristic of the resulting spline is its curvature $\kappa(u)$, which can be computed using the following equation:

$$\kappa(u) = \frac{\dot{x}(u)\ddot{y}(u) - \ddot{x}(u)\dot{y}(u)}{(\dot{x}(u)^2 + \dot{y}(u)^2)^{3/2}} \tag{2}$$

G^2-splines optimization algorithm The general case for the optimization of the $\boldsymbol{\eta}$ values requires numerical optimization [7]. However, due to the type of the required vehicle motion maneuvers, we can apply a faster solution that does not need this numerical optimization, but a short iterative process. We have realized that these trajectories have symmetrical behaviour between the starting and ending points, then we can impose the following constraints to the $\boldsymbol{\eta}$ parameters. $\eta_1 = \eta_2$ and $\eta_3 = -\eta_4$ so only 2 parameters are needed to be tuned: $\eta_{1\_2} = \eta_1 = \eta_2$ and $\eta_{3\_4} = \eta_3 = -\eta_4$.

$\eta_{3\_4} \in (-\infty, +\infty)$, and it affects to the curvature changes. It can be concluded that the best value for this parameter is 0 in all the cases. In Fig. 2, different η_{3_4} values are tested performing a lane change maneuver. As it can be seen, if $\eta_{3\_4} > 0$ (green trajectories) the curvature changes are concentrated in the center of the trajectory, whereas $\eta_{3\_4} < 0$ (blue trajectories) concentrate it on the extreme initial and final points. The red trajectory represents $\eta_{3\_4} = 0$, and it is the smoothest and more balanced trajectory, presenting the minimum curvature variability.

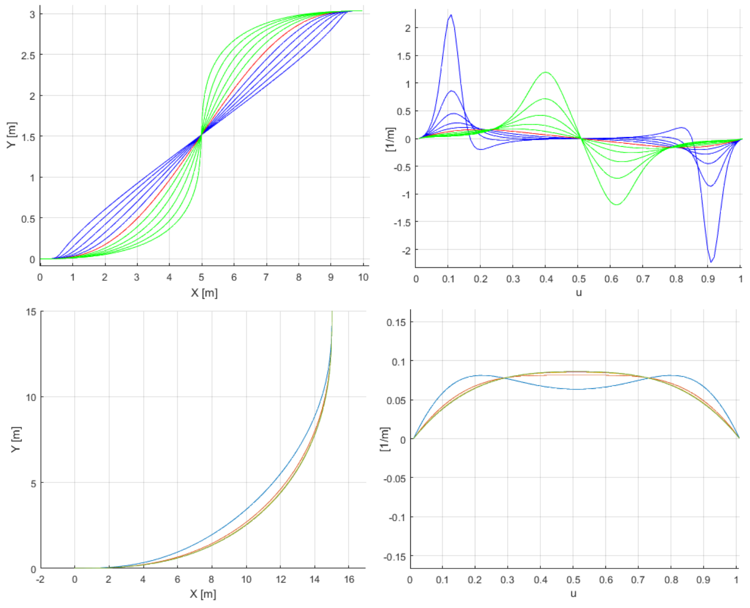

Fig. 2. η_{3_4} variations in a lane change maneuver (top) and η_{1_2} iterations approximating a circular path (bottom)

On the other hand, $\eta_{1\_2} \in (0, +\infty)$, and it forces θ and κ to stay close to the initial and final values. It has been found that the smoothest trajectory is reached when η_{1_2} is close to the trajectory length (meters). For this reason, an iterative method is performed in order to converge into the best possible trajectory, giving to η_{1_2} the length of the path. After a few iterations, usually 3 or 4, all the computed trajectories converge into the smoothest one (Fig. 2). This process obtains similar results as the proposed numerical optimization in [7], which minimizes the curvature variability $\left(\min\limits_{\boldsymbol{\eta}} \dfrac{d\kappa}{du}\right)$, but with much less computation allowing to reach real time performance.

2.2 Velocity profile generation

We explain three different situations for the velocity profile generation.

3rd order spline without initial acceleration The proposed 3rd order spline in velocity has 4 variables: a, b, c, d (Eq. 3). The fifth unknown is the total time of the trajectory, T. The equation system is solved by applying initial and final conditions, and also using the fact that the velocity spline is symmetric, so the maximum acceleration is achieved at time equal to $T/2$.

$$\begin{aligned} x(t) &= \tfrac{a}{4}t^4 + \tfrac{b}{3}t^3 + \tfrac{c}{2}t^2 + dt + x_0 \\ v(t) &= at^3 + bt^2 + ct + d \\ a(t) &= 3at^2 + 2bt + c \end{aligned} \tag{3}$$

$$\begin{cases} v(0) = v_0 \\ a(0) = 0 \\ v(T) = v_f \\ a(T) = 0 \\ a(\frac{T}{2}) = a_{max} \end{cases} \quad \begin{cases} b = \dfrac{4a_{max}^2}{3(v_f - v_0)} \\ a = -\dfrac{b^2}{3a_{max}} \\ c = 0 \\ d = v_0 \\ T = \dfrac{3(v_f - v_0)}{2a_{max}} \end{cases} \tag{4}$$

3rd order spline with arbitrary initial acceleration The 3rd order spline in velocity is the same as in the case without initial acceleration. The equation system is slightly different because the spline is not symmetric and the maximum acceleration is reached in the time t_1. The 6 unknowns are the spline variables a, b, c, d, the total time of the trajectory T and the time t_1.

$$\begin{cases} v(0) = v_0 \\ a(0) = a_0 \\ v(T) = v_f \\ a(T) = 0 \\ a(t_1) = a_{max} \\ a'(t_1) = 0 \end{cases} \quad \begin{cases} 0 = \left[\dfrac{4a_0^2}{a_0 - a_{max}} - 3a_0\right] T^2 + \\ \quad + \left[6(v_f - v_0) - 12\dfrac{(v_f - v_0)a_0}{a_0 - a_{max}}\right] T + \left[\frac{9(v_f - v_0)^2}{a_0 - a_{max}}\right] \\ b = \dfrac{1}{T}\left[\frac{3(v_f - v_0)}{T} - 2a_0\right] \\ a = \dfrac{b^2}{3(a_0 - a_{max})} \\ c = a_0 \\ d = v_0 \\ t_1 = -\dfrac{b}{3a} \end{cases} \tag{5}$$

We have to solve a 2nd order equation to find T_1 and T_2. Not in all cases there exists a solution, due to the denominator value. a_{max} must be always greater than a_0 if accelerating or lower when decelerating.

When only one of the T_i is positive, then this is the unique solution. In the case when both T_i are positive corresponds to two possible solutions: increasing acceleration up to a_{max} and finishing faster the maneuver, or decreasing the acceleration from a_0 and finishing the maneuver in longer time. In this case, the fastest maneuver is always chosen, so $T = min(T_1, T_2)$.

The cases when the spline has no solution correspond to situations when the vehicle is accelerating in the opposite direction of the desired final speed, or *contradictory cases*. This issue is solved by adding a linear section that starts in a_0 and ends in 0 acceleration, with a desired slope that guarantees comfort and feasibility.

Velocity profile generation algorithm The complete algorithm for generating a velocity profile is detailed in Algorithm 1.

Algorithm 1 Generate a velocity profile

```
if $a_0 = 0$ then
    Spline case without $a_0$
else
    if $v_f = v_0$ then
        Add linear acceleration from $a_0$ to 0
        Spline case without $a_0$
    else
        if contradictory case then
            Add linear acceleration from $a_0$ to 0
            Spline case without $a_0$
        else
            Spline case with arbitrary $a_0$
        end if
    end if
end if
```

Set of velocity profile candidates This spline-based method gives a smooth transition between two different velocities, caused by the derivative of the acceleration, the jerk, which is continuous. In addition, the possibility of directly adjusting the parameter of maximum desired acceleration in the trajectory is very useful for this application, as it is directly related with the time needed for the trajectory and also with the comfort of the passengers and its dynamic feasibility.

For each candidate path, the motion planner computes a set of velocity profile candidates, discretizing the final velocities and also the accelerations, taking into account the kinodynamics of the vehicle. Starting in the current state of the host vehicle, several final velocities are chosen from 0 to the maximum allowed road/street velocity. Then, for each final velocity, different accelerations are also chosen, from the maximum deceleration limit to the desired acceleration value. Examples of these velocities and accelerations can be seen in Fig. 3.

In order to reduce the computation, we impose some constraints in the velocity profile using the splines to reject the non-feasible candidates and restrict the search space. The constraints are:

1. The distance to collide with a static obstacle, by using the distance in length of the track from the host vehicle. If a collision is detected, the only allowed final velocity is 0.
2. The maximum allowed velocity due to the planned path curvature.
3. The maximum allowed velocity due to the center lane path curvature. This restriction assures safety, in order to be more conservative and check a static property of the track geometry, because the planned path could be smoothed when re-launching.
4. The longitudinal dynamics of the vehicle. This restricts the accelerations to ensure feasibility.

With this approach, we can have the required accelerations, and then we can reduce the search space and prune the velocity profile selecting the best ones.

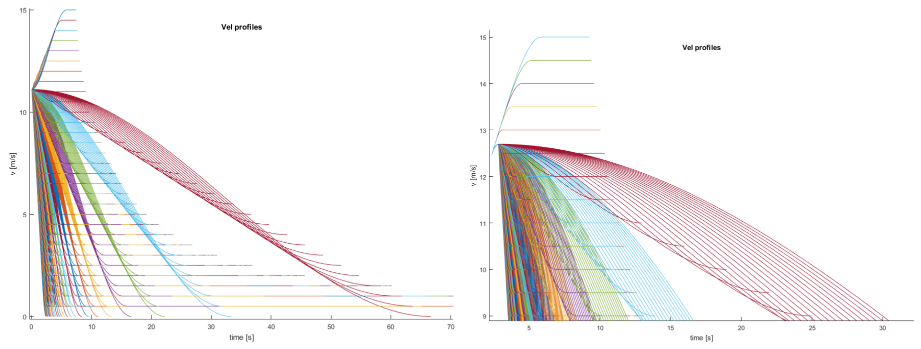

Fig. 3. Sets of all the velocity profiles, with a discretization of 0.5 m/s and different accelerations. In the right figure, cases with a lower final velocity are *contradictory cases*

2.3 Cost structure

The chosen velocity profile is the one with minimum dynamic cost. The motion planner uses two kind of cost terms: static and dynamic. Dynamic costs are related to the temporal dimension and they are associated to a velocity profile and include the dynamic obstacles (Table 1) and is computed as $J_{dynamic} = \sum_i w_i c_i$. The costs are based on the method proposed in [6], but adapted to fit the proposed approach.

Cost	Formula	Physical interpretation	Impact
c_v	$1 - v_{f,vp}/v_{max,desired}$	velocity	Behaviour
c_a	$abs(a_{max,vp}/a_{max,braking})$	acceleration	Comfort & Efficiency
$c_{obs,d}$	$f \cdot e^{(-1/\lambda)\cdot d}$	dynamic obstacles repulsion	Safety & Behaviour

Table 1. Dynamic costs

$v_{f,vp}$ is the final velocity of the candidate velocity profile. $v_{max,desired}$ is the maximum allowed velocity at the current moment. $a_{max,vp}$ is the maximum acceleration value of the candidate velocity profile. $a_{max,braking}$ is the acceleration value (always < 0) of the maximum allowed braking.

In the other hand, static costs are the terms related to path geometry and static obstacles and they are associated to a candidate path (Table 2). The cost function is computed as the weighted sum of all the statics terms, $J_{static} = \sum_i w_i c_i$.

Cost	Formula	Physical interpretation	Impact
c_l	$l/s_{maneuver}$	path length	Efficiency
c_κ	$max(\kappa) \cdot r_{min}$	maximum path κ	Comfort & Kinematic feasibility
$c_{\dot{\kappa}}$	$max(\dot{\kappa}) \cdot r_{min}$	maximum path $\dot{\kappa}$	Comfort & Kinematic feasibility
c_{off}	o/o_{max}	lateral offset from centerline	Behaviour
$c_{obs,s}$	$f \cdot e^{(-1/\lambda)\cdot d}$	static obstacles repulsion	Safety

Table 2. Static costs

All the terms (static and dynamic) with the exception of the obstacle terms, are normalized between 0 and 1. $s_{maneuver}$ is the longitudinal distance from the host vehicle and the ending point of the path. κ is the curvature value of a specific point in the path. r_{min} is the minimum turning radius of the host vehicle. o_{max} is the lateral distance from the farthest endpoint to the center of the lane (the point with more lateral offset). f is a scale value and λ is the decay of the exponential function. d is the distance from a obstacle (static or dynamic) to the host vehicle (they have been modelled as circles). If d is smaller than a threshold, then the cost is penalized:

$$c_{obs} = \begin{cases} f \cdot e^{(-1/\lambda)\cdot d} + Penalization, & if \;\; d < threshold \\ f \cdot e^{(-1/\lambda)\cdot d}, & otherwise \end{cases} \tag{6}$$

Unlike other approaches that penalize collisions with an infinite cost, as [6], it is preferred to preserve the value. Then, in a situation where all the candidate solutions present high cost (for instance, an unavoidable obstacle), the planner will give always the minimum cost solution, so it will be the less dangerous one.

Finally, for several maneuver candidates, formed by a path-velocity profile pair, a total cost function J_{total} is computed as the minimum cost maneuver, $J_{total} = min_{maneuver}(J_{dynamic} + J_{static})$.

3 Simulations

The motion planner and the low level vehicle control have been implemented in MATLAB. The Stanley's lateral controller [8] has been used together with a longitudinal controller which takes into account the dynamics of the vehicle. The next figures show representative road scenarios (with straight and curves lanes), with the lines (in black), its center-lines (in blue) and the path followed by the host vehicle (in green). The host vehicle (in red and purple) and the velocity profile are plotted at each interval of time. The simulations were made in MATLAB with a slow CPU and the average computation for each simulation cycle with static and dynamic obstacles was of 250 ms. Taken into account that programming in C++ is around 10 times faster, this method can run at 25 ms.

Static environment Fig. 4 shows a simulation in a road environment without any obstacle in the lanes. Fig. 5 shows the same environment, but with several static obstacles, represented by circles in the lanes. This simulates an urban scenario, for instance with vehicles parked in double row.

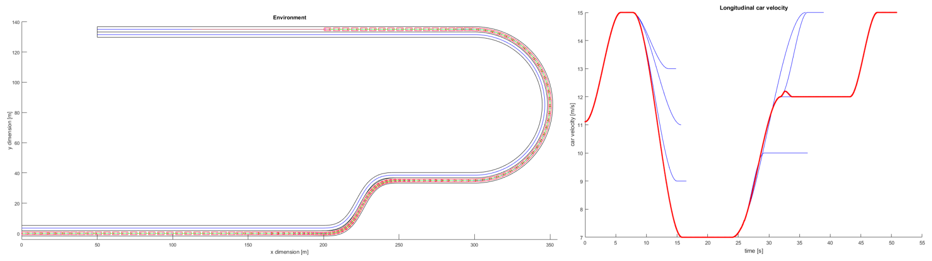

Fig. 4. Simulation without any obstacle. Followed path (left) and vehicle velocity profile (right) in red and planned in blue

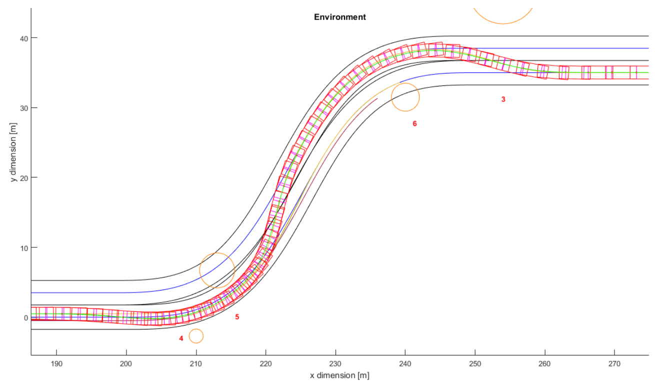

Fig. 5. Detail of a static obstacle avoidance maneuver

Dynamic environment Fig. 6 shows two different overtaking maneuvers involving dynamic obstacles.

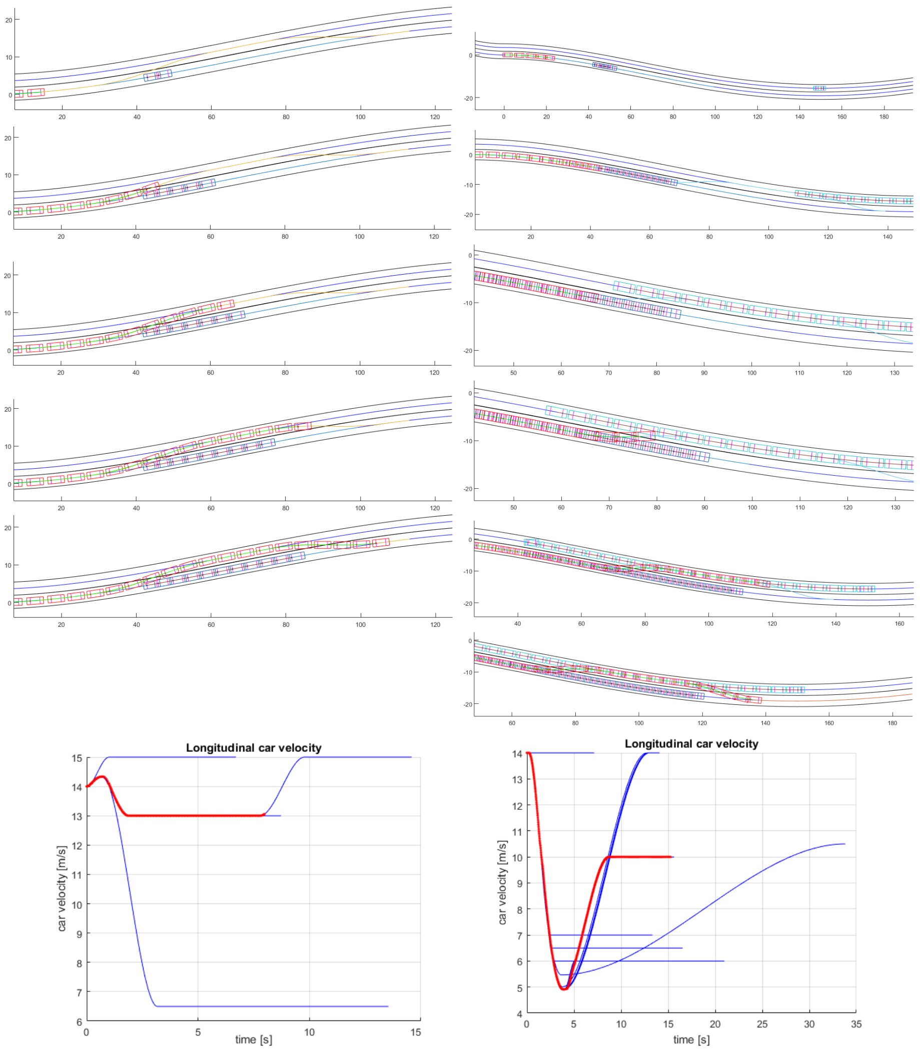

Fig. 6. Overtaking of a slower vehicle in a one way track (left) and in a two way track with oncoming traffic (right). Red vehicle is the host.

T-intersection To enter in a T-intersection is mandatory to anticipate the oncoming vehicles, which come in both directions. Fig. 7 is a conservative case, where the host vehicle slows down and let the traffic pass. On the other hand, Fig. 8 is a more aggressive case, where the host vehicle takes advantage of a gap between vehicles.

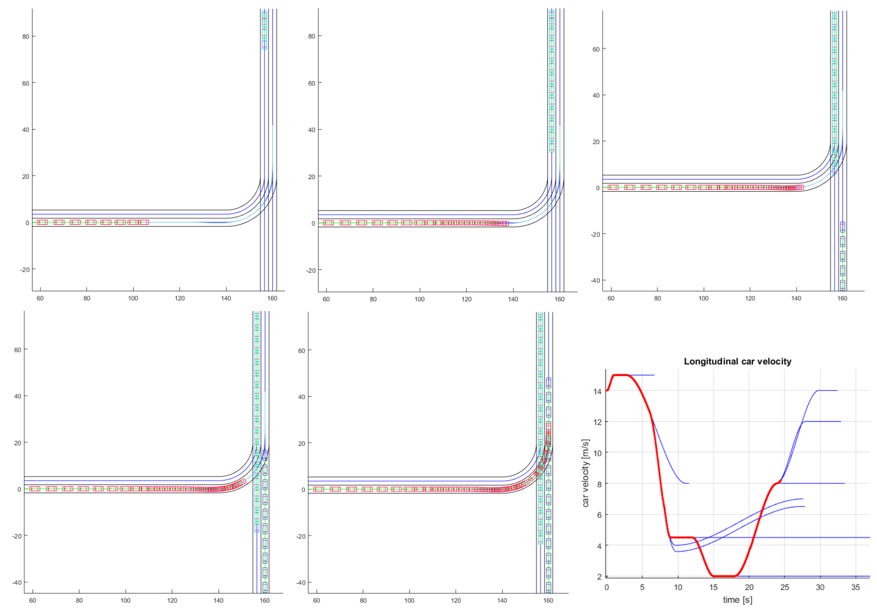

Fig. 7. Incorporation in a T-intersection, in several instants of time. Conservative case

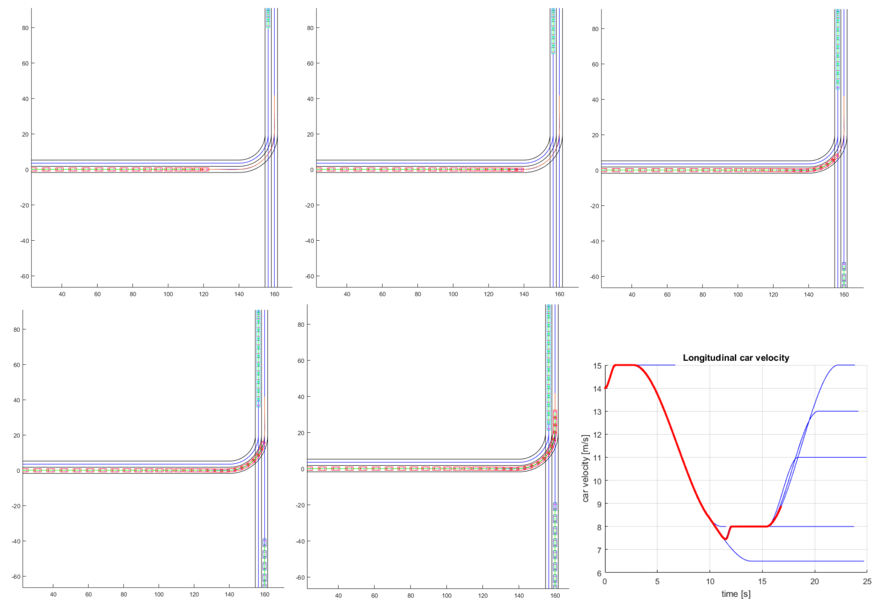

Fig. 8. Incorporation in a T-intersection, in several instants of time. Aggressive case

4 Conclusions

After analyzing the simulations, it is verified that the proposed methods of path and velocity profile generation behaves correctly and help to overcome successfully all the presented situations, which correspond to representative cases that occur in on-road driving.

The proposed approach for the path generation, using the G^2-splines, performs above expectations and a real-time computation can be obtained.

The proposed method for the velocity profile generation behaves smoothly and provides a fully analytic solution that can deal with any arbitrary situation. Moreover, it allows to directly adjust the desired acceleration instead of the maneuver time and this helps to the selection and pruning of a candidate set of velocity profiles, taking into account kinodynamic and dynamic constraints.

References

1. B. Paden, M. Cáp, S. Zheng Yong, D. Yershov, E. Frazzoli: A Survey of Motion Planning and Control Techniques for Self-driving Urban Vehicles. IEEE Transactions on Intelligent Vehicles, vol. 1, no. 1, pp. 33-55, 2016.
2. C. Katrakazas, M. Quddus, W.-H. Chen, L. Deka: Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. Transportation Research Part C, vol. 60, pp. 416-442, 2015.
3. D. G. Bautista, J. P. Rastelli, R. Lattarulo, V. Milanés, F. Nashashibi: Continuous curvature planning with obstacle avoidance capabilities in urban scenarios. 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), pp. 1430-1435, 2014.
4. D. Ferguson, T. M. Howard, M. Likhachev: Motion Planning in Urban Environments, Part I and II. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1063-1069, 1070-1076, 2008.
5. T. Gu, J. M. Dolan: On-Road Motion Planning for Autonomous Vehicles. International Conference on Intelligent Robotics and Applications (ICIRA), vol. 3, pp. 588-597, 2012.
6. W. Xu, J. Wei, J. M. Dolan, H. Zhao, H. Zha: A Real-Time Motion Planner with Trajectory Optimization for Autonomous Vehicles. IEEE International Conference on Robotics and Automation (ICRA), pp. 2061-2067, 2012.
7. C. G. Bianco, A. Piazzi: Optimal trajectory planning with quintic G^2- splines. Proceedings of the IEEE Intelligent Vehicles Symposium, pp. 620-625, 2000.
8. G. M. Hoffmann, C. J. Tomlin, M. Montemerlo, S. Thrun, Autonomous Automobile Trajectory Tracking for Off-Road Driving: Controller Design, Experimental Validation and Racing. American Control Conference (ACC), pp. 2296-2301, 2007.