IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 21, 2019, accepted December 9, 2019, date of publication December 13, 2019,
date of current version December 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2959432

Real-Time Motion Planning Approach for Automated Driving in Urban Environments

ANTONIO ARTUÑEDO, JORGE VILLAGRA, AND JORGE GODOY
Centre for Automation and Robotics (CSIC-UPM), 28500 Arganda del Rey, Spain

Corresponding author: Antonio Artuñedo (antonio.artunedo@csic.es)

This work was supported in part by the Spanish Ministry of Science, Innovation and Universities with National Project COGDRIVE under Grant DPI2017-86915-C3-1-R, in part by the European Commission through the Projects PRYSTINE under Grant ECSEL-783190-2, in part by SECREDAS under Grant ECSEL-783119-2, and in part by the Community of Madrid through SEGVAUTO 4.0-CM Programme under Grant S2018-EMT-4362.

ABSTRACT Autonomous vehicles must be able to react in a timely manner to typical and unpredictable situations in urban scenarios. In this connection, motion planning algorithms play a key role as they are responsible of ensuring driving safety and comfort while producing human-like trajectories in a wide range of driving scenarios. Typical approaches for motion planning focus on trajectory optimization by applying computation-intensive algorithms, rather than finding a balance between optimatily and computing time. However, for on-road automated driving at medium and high speeds, determinism is necessary at high sampling rates. This work presents a trajectory planning algorithm that is able to provide safe, human-like and comfortable trajectories by using cost-effective primitives evaluation based on quintic Bézier curves. The proposed method is able to consider the kinodynamic constrains of the vehicle while reactively handling dynamic real environments in real-time. The proposed motion planning strategy has been implemented in a real experimental platform and validated in different real operating environments, successfully overcoming typical urban traffic scenes where both static and dynamic objects are involved.

INDEX TERMS Autonomous driving, trajectory planning, obstacle avoidance, collision checking.

I. INTRODUCTION

Among the wide range of technologies involved in automated driving, decision-making systems aim to both provide the understanding of the vehicle environment and generate a safe and efficient action plan in real-time [1]. Accordingly, tasks such as prediction of nearby traffic participants actions, motion planning, and obstacle avoidance must be carried out within the decision-making sub-systems.

Whilst considerable efforts have been addressed in the perception and localization domains, planning in urban scenarios is still in an earlier development stage. Understanding the spatio-temporal relationship between the subject vehicle and the surrounding entities, while being constrained by the road network is a very difficult challenge. With the aim of reacting safely even in complex urban situations, autonomous vehicles require methods to generalize unpredictable situations and reason in a timely manner. In this context, motion planning is particularly relevant as it plays a key role in ensuring driving safety [2], [3], while producing human-like trajectories in a wide range of driving scenarios. Most of the motion planning algorithms proposed in the literature focuses on path and speed optimization by applying computation-intensive algorithms, rather than finding a balance between optimatily and computing time. However, for on-road automated driving at medium and high speeds, determinism is necessary at high sampling rates while ensuring safety and comfort.

On the basis of the modular architecture for automated driving proposed in [4], which enables the vehicle to handle dynamic urban scenarios, this work presents a trajectory planning algorithm that is able to provide safe and comfortable paths and speed profiles by using cost-effective primitives evaluation and selection. The main contributions of the trajectory generation strategy proposed in this paper are listed below:

- A novel sampling method for generating sets of curvature-continuous path candidates based on quintic Bézier curves.
- A motion planning algorithm that computes feasible, safe and human-like trajectories in real-time, while

The associate editor coordinating the review of this manuscript and approving it for publication was Razi Iqbal.

considering the kinodynamic constrains of the vehicle. This allows the vehicle to reactively handle dynamic scenarios in real environments.

The proposed motion planning strategy has been implemented in a real experimental platform and validated in different real operating environments, successfully overcoming typical urban traffic scenes where both static and dynamic objects are involved.

The remainder of the article is organized as follows. In section II, an overview of the most relevant motion planning approaches for automated driving is given. Section III introduces the motion planning architecture in which the proposed trajectory generation strategy is integrated. Section IV focuses on the path planning as well as the collision checking methods and V describes the applied speed planning method. Section VI focuses on the trajectory generation algorithm. In section VII, the results are shown and discussed. Finally, some conclusions and future works are presented in section VIII.

II. RELATED WORK

Motion planning is required to provide feasible and collision-free trajectories in a short period of time while considering the surrounding environment and the current state of the vehicle. For most planning problems of interest in urban environments, optimal algorithms with practical computational complexity are unavailable [5], [6]. In that regard, significant research attention has been directed towards studying approximate methods or particular solutions of the general motion planning problem. In the following subsections a review of the relevant works on path and speed planning is carried out.

A. PATH PLANNING

The path planning techniques proposed in the literature generally do not find an exact answer to the problem, but attempt to find a satisfactory solution or a sequence of feasible solutions that converge to the optimal solution. The utility and performance of these approaches are typically quantified by the class of problems for which they are applicable as well as their evidences for converging to an optimal solution. These approximate methods for path planning can be divided in three main families [2]: (i) variational methods, that project the infinite-dimensional function space of trajectories to a finite-dimensional vector space [7]–[11], (ii) incremental search methods sample the configuration space and incrementally build a reachability graph (often a tree) that maintains a discrete set of reachable configurations and feasible transitions between them. One of the most well-known and used techniques are the RRT [12] and their variants (e.g. [13]), always looking for the best trade-off between completeness and computational cost. Finally, (iii) the graph-based search methods discretize the configuration space of the vehicle as a graph, where the vertices represent a finite collection of vehicle configurations and the edges represent transitions between vertices. The graph is then used to find a minimum-cost path based on defined cost function. There is a significant number of methods to build that graph. They can be grouped in two main families: geometric methods, such as cell decomposition [14], visibility graphs [15] or Voronoi diagrams [16], and sampling-based methods [17], [18]. The latter are particularly relevant in structured environments, where different motion primitives (e.g [19]) or steering functions (e.g [20], [21]) can be applied to explore the reachability of the free configuration space [22], [23]. Once the graph is built, different strategies exist also to conduct the graph search in the most appropriate way (e.g. Dijkstra [24], A* [25], D* [26], etc.). In the case of automated driving, the road structure provides strong heuristics, where sampling-based planning methods are very often sufficient to produce a feasible solution [2]. An evolution of these methods, where spatio-temporal constraints are considered, propose to formulate the problem as a trajectory ranking and search problem, where multiple cost terms are combined to produce a specific behaviour.

The applicability of variational methods is limited by their convergence to only local minima. In order to try to overcome this problem, graph-search methods perform global searches in a discretized version of the path space, generated by motion primitives. In some specific situations, this fixed graph discretization may lead to wrong or suboptimal solutions. In this cases, incremental search strategies may be useful, providing a feasible path to any motion planning problem instance, if one exists. In exchange, the required computation time to verify this completeness property may be unacceptable for a real-time system as required in the scope of automated driving.

To overcome these limitations, some recent approaches propose the use of a double stage planning strategy [27]–[29] that aims at limiting the search space to the region where the optimal solution is likely to exist, while keeping a high degree of reactivity. To that end, a first step performs the spatial space only considering the road geometry, resulting in a first reference trajectory that does not consider obstacles. Then, a traffic-based planning produces a path based on the reference plan to take into account other traffic and interfering objects. Finally, the final trajectory is generated for the most appropriate manoeuvre. Although two-stage optimization approaches give good results in terms of curve smoothness, as discussed in the comparison carried out in [23], a significant number of possible variations can significantly affect the resulting path and the computation times are generally high as well as indeterminate since the applied optimization algorithms do not ensure to obtain a solution for the motion planning problem in a constrained time horizon. Based on the comparison of path planning methods in [23], the approaches using quintic Bézier curves and not applying optimization algorithms have been shown to be able to find solutions with low values of the cost functions used, by evaluating a limited number of possible candidates and therefore obtaining low and limited computation times.

B. SPEED PLANNING

Regarding speed planning techniques, two different approaches can be found in the literature: those that jointly compute the path and speed and those that decouple the path and speed planning in two different stages. The methods in this first group are generally not able to work in real time. This is because considering both the spatial an temporal motion components at one, the problem dimensionality is tipically high. For example, in [30] a search-based approach is applied over three dimensional space (two spatial dimensions and time) in order to adapt the speed of the vehicle based on the spatial distance among the nodes. Although this strategy presents a small computing time, the search space is too small to obtain a near-optimal trajectory. In another approach [31], the authors use a discrete spatio-temporal state lattice to combine the configuration space and time and set the reachable states. Both approaches have only been tested in simulation.

Most of the motion planning strategies found in the literature in the scope of automated driving use decoupled path and speed planning. Within these approaches, the one proposed in [32] the geometric properties of the primitives applied for computing the path (a concatenation of clothoids, arcs of circles and straight lines) are used to simplify the speed profile computation that limits the maximum speed, both longitudinal and lateral accelerations and longitudinal jerk. Thus, the final trajectory is obtained by concatenating the different path and speed sections. The main drawback of this approach is the dependency on the use of that same type of primitives. Another approach proposed in [33], propose an optimization-based strategy for path planning and then states the speed planning as a convex optimization problem. This approach limits the maximum speed, lateral and longitudinal accelerations. However, the computing time of the speed profile is dependent on the applied optimization algorithm.

Since the computing time of each planning cycle plays a key role in driving safety and comfort, the motion planning strategy proposed in this paper focuses on computing an optimal trajectory in a limited time without using computationally intensive optimization, while evaluating large search space. To that end, a decoupled method is used. Quintic Bézier curves are used as geometric primitives for path planning and a fast and primitive-agnostic speed planning algorithm computes a speed profile that limits both maximum speed and maximum longitudinal and lateral accelerations.

III. MOTION PLANNING ARCHITECTURE

The trajectory generation proposed in this paper is integrated in an architecture that consists of a set of modules with different functionalities in order to provide the whole system with automated driving capabilities. These components are depicted in the general functional diagram of Fig. 1, where dashed lines represent event-driven actions and continuous lines correspond to continuously applied actions.

Within the architecture, the decision modules are structured in two groups: global and local planning. On the one hand, global planning and mapping modules address deliberative features such as the computation of the route to reach a given destination. Instead of requiring high-definition maps, the proposed approach uses low-fidelity map data to plan a global route and then automatically generate an extended data structure from OSM that enables the computation of driving corridors [34].

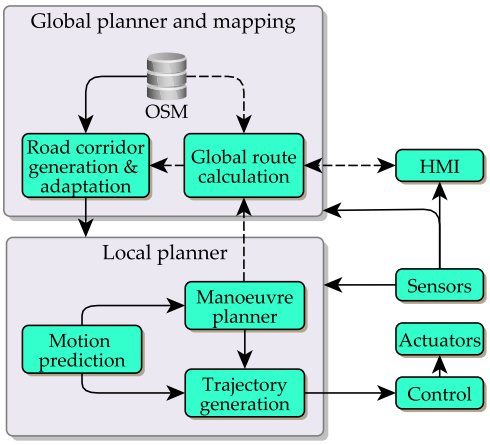


FIGURE 1. Motion planning architecture.

On the other hand, local planning modules deal with reactive decisions such as final trajectory generation and obstacle avoidance. Self-generated driving corridors are used by the local planner modules to finally plan the trajectories that the vehicle will follow. To that end, three different elements have been integrated in the local planner. Firstly, a motion prediction component predicts the future motion of perceived objects. A manoeuvre planner is then responsible of analysing the output of the motion prediction module by checking possible spatio-temporal collisions with the current planned trajectory, and consequently trigger a new trajectory planning request, if needed. Finally, based on the predicted motion of nearby objects and the manoeuvre request, a trajectory generation module computes the final path and speed profile.

The trajectory generation is the last step of the motion planning architecture. The main goal of this module is to provide a new trajectory when requested by the manoeuvre planner, with the aim of achieving the best trade-off between smoothness, safety and planning time. Note that in this document, it is referred to as trajectory the composition of a path with an associated speed profile. Bearing this in mind, the computed trajectory must meet a set of requirements:

- To ensure comfort inside the vehicle, steering and pedals behaviour must be smooth and continuous. In order words, lateral and longitudinal accelerations must not exceed specified maximum values along the trajectory.
- The trajectory generator must be able to provide feasible trajectories to avoid static or dynamic objects.
- The trajectory must be computed in a reasonable amount of time in order to be reactive enough to avoid collisions in dangerous situations.

The following sections focus on the trajectory generation module, detailing the proposed algorithms for path and speed planning.

IV. PATH PLANNING APPROACH

This section describes the details of both the approach to generate the path candidates and the used collision checking method.

Based on the requirements stated in section III, the primitive used for path planning must be able to generate a continuous curvature path. In geometric terms, that means that G^2 continuity must be guaranteed. Furthermore, the path must be computed as fast as possible, since an optimization algorithm should evaluate a large number of paths in the shortest possible time.

Taking into account the extensive comparison presented in [23], quintic Bézier curves are chosen to generate the final path. Some of the main advantages of fifth order Bézier curves over cubic ones are the higher control of the curve shape and the possibility to impose curvature at both extremes of the curve. Let us recall that it is possible to concatenate quintic Bézier sections to achieve curvature continuity along the path [35], thus complying with comfort requirements. Moreover, these primitives allow to define a wide range of curves from given initial and final poses, leading to a straightforward generation of a number of possible paths. The analytic expression of a quintic Bézier curve is obtained as follows:

$$C_{d_b}(t) = \sum_{i=0}^{d_b} \mathbf{P}_i B_{i,d_b}(t), \quad t \in [0, 1] \tag{1}$$

being $B_{i,d_b}(t) = \binom{d_b}{i} t^i (1-t)^{d_b - 1}$ the Bernstein polynomials, $\mathbf{P}_i$ the control points of Bézier curve, and d_b the degree of the Bézier curve. In the case of a quintic curve ($d_b = 5$), it can be expressed explicitly as:

$$\begin{aligned} C_5(t) = {} & (1-t)^5 \mathbf{P}_0 + 5t(1-t)^4 \mathbf{P}_1 \\ & + 10t^2(1-t)^3 \mathbf{P}_2 + 10t^3(1-t)^2 \mathbf{P}_3 \\ & + 5t^4(1-t)\mathbf{P}_4 + t^5 \mathbf{P}_5, \ t \in [0, 1] \end{aligned} \tag{2}$$

Using this primitive, a sampling-based motion planning approach is able to generate continuous curvature paths, i.e. maintaining G^2 continuity.

In view of previous works carried out to compare path planning strategies with different primitives [23], the approach chosen for the path generation in this work is the following: Firstly, a modified version of the Douglas-Peucker algorithm is applied to obtain a set of reference points from the centreline of the road corridor. Besides the maximum distance between the original and the simplified curve (ϵ_{simp}), the modified algorithm also limits the distance between two consecutive simplified points (d_{simp}^{max}) in order to extend the search space increasing the amount of reference points to explore. Then, the reference points obtained after appling the Douglas-Peucker algorithm are used to create sets of path candidates to be explored by the planning algorithm.

As stated, quintic Bézier curves provides a higher controllability of the curve shape over cubic ones. The higher polynomial order allows to impose the position, orientation and curvature at the extreme curve points but also two additional degrees of freedom are still available, which are used to generate a large variety curves with the same initial ($p_0 = [x_0, y_0, \theta_0, \kappa_0]$) and final ($p_f = [x_f, y_f, \theta_f, \kappa_f]$) poses. That is achieved by varying velocity and acceleration vectors while maintaining the initial and final poses.

In order to generate a set of curves with the same orientation at their extremes, the length of the initial and final velocity vectors ($\vec{\mathbf{t}}_0$ and $\vec{\mathbf{t}}_f$) is varied. Firstly, to make independent the modules of the tangent vectors from each different curve cases (where the distance between extreme points is not constant), both lengths are normalised with respect to the distance between both curve extremes (d_{0f}). Then, a set of N_t points is generated between the interval $[m_t^{min}, m_t^{max}]$, where m_t^{min} and m_t^{max} are the minimum and maximum normalised lengths of the tangent vectors, respectively. Finally the length of the tangent vector is calculated as follows:

$$|\vec{\mathbf{t}}_0^n| = |\vec{\mathbf{t}}_f^n| = m_{tn} \cdot d_{0f} \quad \forall m_{tn} \in [m_t^{min}, m_t^{max}] \ n = 1, \ldots, N_t \tag{3}$$

Furthermore, the influence of the curvature on the acceleration vector is also used to generate more candidates. Note that the curvature (κ_n) only affects the normal component of the acceleration vector:

$$\vec{\mathbf{a}}_n = |\vec{\mathbf{t}}_n|' \cdot \vec{\mathbf{t}}_n^u + \kappa_n |\vec{\mathbf{t}}_n|^2 \cdot \vec{\mathbf{n}}_n^u \tag{4}$$

where $\vec{\mathbf{t}}_n^u$ and $\vec{\mathbf{n}}_n^u$ are the unit tangent and normal vectors at point P_n of the curve respectively. With this in mind, the magnitude of the tangential component of the acceleration vector can be varied to generate different curve candidates that maintain the same initial and final velocity vectors ($\vec{\mathbf{t}}_0$ and $\vec{\mathbf{t}}_f$) and curvatures (κ_0 abd κ_f) at the curve extremes.

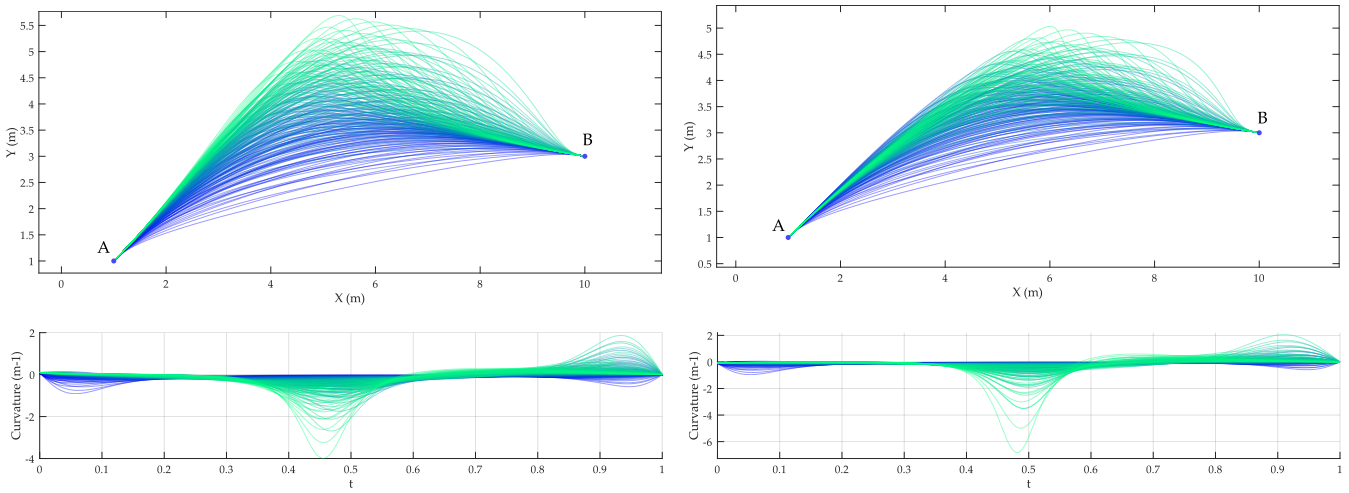
Just like in the case of the velocity vector length, the tangential component of the acceleration vector is modified proportionally to distance d_{0f} as follows, in order to generate N_κ different values of the tangential component of the acceleration vector (a_n^t):

$$a_n^t = m_{\kappa n} \cdot d_{0f} \quad \forall m_{\kappa n} \in [m_\kappa^{min}, m_\kappa^{max}] \ n = 1, \ldots, N_\kappa \tag{5}$$

Thus, the following expression is used to generate N_κ different acceleration vectors for both the initial ($\vec{\mathbf{a}}_0^n$) and final ($\vec{\mathbf{a}}_f^n$) points of the path:

$$\begin{aligned} \vec{\mathbf{a}}_0^n &= a_n^t \cdot \vec{\mathbf{t}}_0^u + \kappa_0 |\vec{\mathbf{t}}_0|^2 \cdot \vec{\mathbf{n}}_0^u \\ \vec{\mathbf{a}}_f^n &= a_n^t \cdot \vec{\mathbf{t}}_f^u + \kappa_f |\vec{\mathbf{t}}_f|^2 \cdot \vec{\mathbf{n}}_f^u \end{aligned} \tag{6}$$

Once velocity and acceleration vectors are calculated for two given poses, a set of quintic Bézier curves can be generated by imposing all combinations of velocity and acceleration vectors at both extremes. Let $P_0 = [x_0, y_0]$ and $P_f = [x_f, y_f]$ be the position of the initial and final poses,



(a) Initial and final poses: $A = \left[1m, 1m, 45°, 0.1m^{-1}\right]$, $B = \left[10m, 3m, -10°, 0m^{-1}\right]$

(b) Initial and final poses: $A = \left[1m, 1m, 45°, -0.1m^{-1}\right]$, $B = \left[10m, 3m, -10°, 0m^{-1}\right]$

FIGURE 2. **Top: Quintic Bézier curves generated with the same initial and final poses. Bottom: Curvatures of the curves.**

respectively. Thus, the position of the six control points of each Bézier segment can be obtained as follows:

$$\mathbf{P}_0 = P_0 \tag{7}$$

$$\mathbf{P}_1 = P_0 + \frac{1}{5}\vec{\mathbf{t}}_0 \tag{8}$$

$$\mathbf{P}_2 = \frac{1}{20}\vec{\mathbf{a}}_0 + 2\mathbf{P}_1 - P_0 \tag{9}$$

$$\mathbf{P}_3 = \frac{1}{20}\vec{\mathbf{a}}_f + 2\mathbf{P}_4 - P_f \tag{10}$$

$$\mathbf{P}_4 = P_f - \frac{1}{5}\vec{\mathbf{t}}_f \tag{11}$$

$$\mathbf{P}_5 = P_f \tag{12}$$

where $\mathbf{P}_m$ ($m \in \mathbb{N} : m \in [0, 5]$ are control points of the Bézier curve.

This method allows to generate a set of different curves maintaining the initial and final poses, allowing a better space exploration from the same inputs (p_0 and p_f). Fig. 2 illustrates a graphic example of this method. Fig. 2a shows a set of curves generated with the same initial and final poses and the curvature of each curve, where a range of colours has been used to relate each curve (at the top of the figure) with its curvature (at the bottom). Fig. 2b shows another example with a different initial curvature value ($-0.1m^{-1}$ instead of $0.1m^{-1}$). In this figures, the green/blue color scale is used to visualize the correspondence between each 2D curve (top) and its respective curvature (bottom).

A. COLLISION CHECKING

In sampling-based motion planning approaches, collision checking should be carried out for each sampled system state. Thus, collision checking is the most computationally expensive process in most of search-based motion planning algorithms [36].

To mitigate this problem, some approaches such as those presented in [33] introduce two collision checking stages in order to: firstly make a fast approximation of the possible colliding states; then a second and more accurate collision computation is performed.

Several collision checking approaches start from the rectangular vehicle shape and then approximate this rectangle through a set of circles (typically 3, 4, 6 or 8 circles) [33], [36]. The main motivation for these approaches is the low computing time of the collision checking as only the computation of euclidean distances is needed. Their main drawback is the loss of accuracy when few circles are used to approximate the vehicle shape. The key challenge is therefore to balance computing time and collision checking accuracy.

Instead of circle-based approximations of the vehicle footprint, the approach for collision checking presented in this paper uses the bounding rectangle of the vehicle. The method is based on the generation of a polygon that represents the space that the vehicle would take while driving along the calculated path. This occupancy polygon is used to firstly check if the path is inside the road corridor and then if any obstacle collides with it.

In order to obtain the occupancy polygon, the dimensions of the vehicle and the path that is being evaluated are needed. Taking advantage of the fact that the path is a Bézier curve or a concatenation of them, the tangent vector and the curvature can be obtained analytically.

Fig. 3 shows an example of occupancy polygon for a given path, where l_{tw} is the vehicle axle track, l_{la} is the distance from the rear axle to the front bumper and l_{lb} is the distance from rear bumper to the rear axle. Based on the path, the right and left bounds of the area occupied by the vehicle can be computed as follows: the right bound will be composed of the points of right corner of the front of the vehicle when the vehicle is turning left and of the points of right corner of the rear axle when turning right. The left bound is computed analogously: it is composed of the points of left corner of

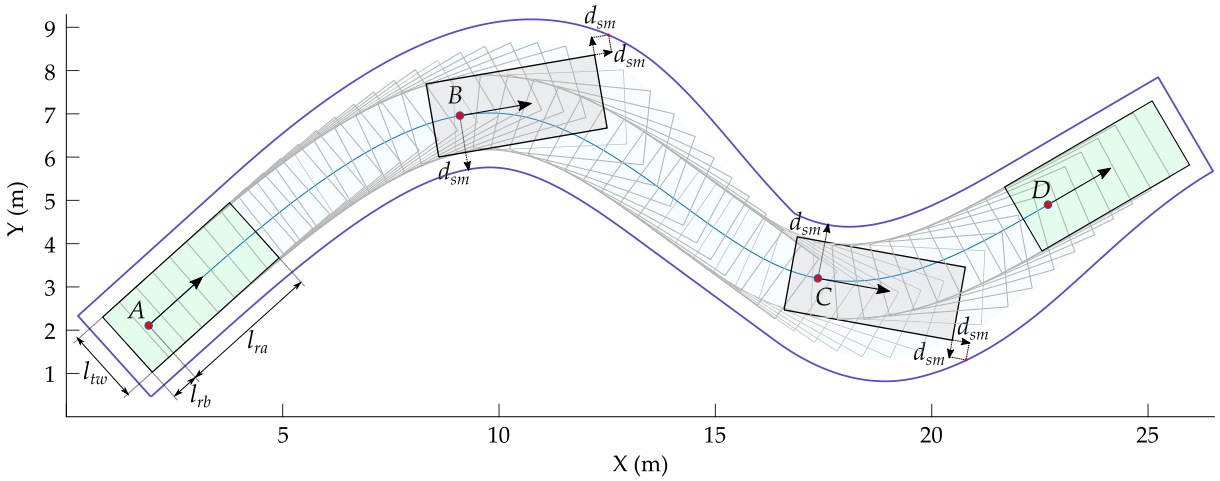


FIGURE 3. Example of the path occupancy polygon calculation for collision checking.

the front of the vehicle when the vehicle is turning right and of the points of left corner of the rear axle when turning left.

It is worth to mention that a safety margin is added around the vehicle (d_{sm}). To determine if vehicle is turning right or left, the sign of the curvature is used. Finally, the polygon is conformed by joining the points of right and left sides to obtain a closed shape.

In point *B* of Fig. 3 it can be seen how the vehicle is turning right and the extreme left point of the front belongs to the left bound of the polygon, while the right extreme point of the rear axle is used for the right bound.

Once the path occupancy polygon is obtained, it is used to check if it is inside the road corridor and also if it collides with some obstacle. Both verification strategies rely on two different modifications of algorithm 5 described in [37] to solve the ''point in polygon'' problem:

- **Verify if the path polygon is inside a given road corridor:** It is checked that all vertexes of a simplified path polygon are inside the road corridor. When a vertex of the path polygon is outside the road corridor polygon, the execution stops.
- **Verify if the path polygon collides with some obstacle:** In this case all vertexes of the obstacles are checked to be outside the path polygon. When a vertex of any obstacle is inside the path polygon the execution stops as in the previous case.

V. SPEED PROFILE GENERATION

The speed profile is calculated over a given path so that a longitudinal speed value is associated to each of path points.

In order to meet the requirements stated in section III, the generated speed profile must not exceed given bounds for longitudinal acceleration, lateral acceleration and speed to comply with traffic rules and to ensure comfort inside the vehicle (see table 1). In this connection, initial and final speed must be also imposed.

TABLE 1. Speed profile generation parameters.

Symbol	Description
v_0	Initial speed
v_f	Final speed
v^{max}	Maximum allowed speed
a_{lat}^{max}	Maximum lateral acceleration
a_{acc}^{max}	Maximum positive longitudinal acceleration
a_{dec}^{max}	Maximum negative longitudinal acceleration

Since the road corridor and the centreline are composed of Bézier curves, the curvature in each of the simplified points over the centreline is analytically calculated when a new corridor is generated. The curvature at each reference point (κ_{R_n}) is then used to compute a maximum speed value ($v_{R_n}^{max}$) by limiting the lateral acceleration (a_{lat}^{max}) as follows:

$$v_{R_n}^{max} = \sqrt{\frac{a_{lat}^{max}}{|\kappa_{R_n}|}} \tag{13}$$

These maximum speed values at reference points will be used to set the final speed of the final trajectory computation:

$$v_f = v_{R_n}^{max} \tag{14}$$

where R_n is the reference point over the centreline used to generate the candidate selected in the final trajectory).

The speed profile calculation is carried out in several stages:

1) Firstly, a speed limit curve (v^{lim}) is computed based on the allowed maximum lateral acceleration (a_{lat}^{max}). To that end, the speed limit at each point of the path (P_n) is computed considering the maximum speed and the circular motion with the curvature at point P_n (κ_n):

$$v_n^{lim} = \min\{v^{max}, \sqrt{\frac{a_{lat}^{max}}{|\kappa_n|}}\} \tag{15}$$

where $n = 1, \ldots, N$ is the index of each path point (P_n) and N is the number of points.

TABLE 2. Speed profile parameters used in example of Fig. 4.

Parameter	Value
$v_0(km/h)$	0
$v_f(km/h)$	0
$v^{max}(km/h)$	25
$a_{lat}^{max}(m/s^2)$	1.0
$a_{acc}^{max}(m/s^2)$	1.0
$a_{dec}^{max}(m/s^2)$	2.0

2) After that, the longitudinal accelerations a_n at every point is checked. To do that, initial and final speeds (v_0 and v_f) are imposed and the acceleration profile is computed assuming uniform acceleration between two consecutive points of the path following this expression:

$$v_n = \sqrt{v_{n-1}^2 + 2a_n d_p} \tag{16}$$

where d_p is the distance between points P_{n-1} and P_n of the path.

3) Then, the accelerations computed for each path point are traversed from the starting to the final point in order to verify that they are lower than the maximum acceleration value (a_{acc}^{max}) and that the speed is below the maximum at that point ($v_n < v_n^{lim}$). In case the acceleration at point P_n exceeds the limit or the speed exceed the limit, both the speed and acceleration at point P_n are thresholded to the maximum values and the speed at point P_{n+1} is recalculated using equation (16).

4) Finally, the same procedure followed in the previous step is performed backwards imposing a deceleration limit of a_{dec} along the whole path.

Fig. 4 presents an example of speed profile computation considering the parameters shown in table 2. At the top of the figure, the example path is shown. As can be seen, the speed profile is always under the speed limit curve, which considers the maximum lateral acceleration and the maximum speed. In this case, the speed limit curve is highly influenced by the lateral acceleration limitation as can be seen when the absolute value of the curvature of the path is high. Moreover, note that the maximum positive and negative longitudinal accelerations are not exceeded along the speed profile, as shown at the bottom of Fig. 4.

When a speed planning is requested, the initial speed is set to the current vehicle speed and the final speed is taken from the speed estimation of the road corridor centreline performed when a new road corridor is set. It is important to emphasize the importance of speed imposed at the end of the trajectory, as it ensures the anticipation of the speed calculation over a path taking into account the road corridor topology beyond where the current planed path ends.

It is also remarkable that this approach allows to change the speed planner parameters (a_{lat}^{max}, a_{acc}^{max}, a_{dec}^{max} and v^{max}). That means that it is possible to adapt the speed behaviour over the path based on passengers needs allowing them to choose the driving abruptness level.

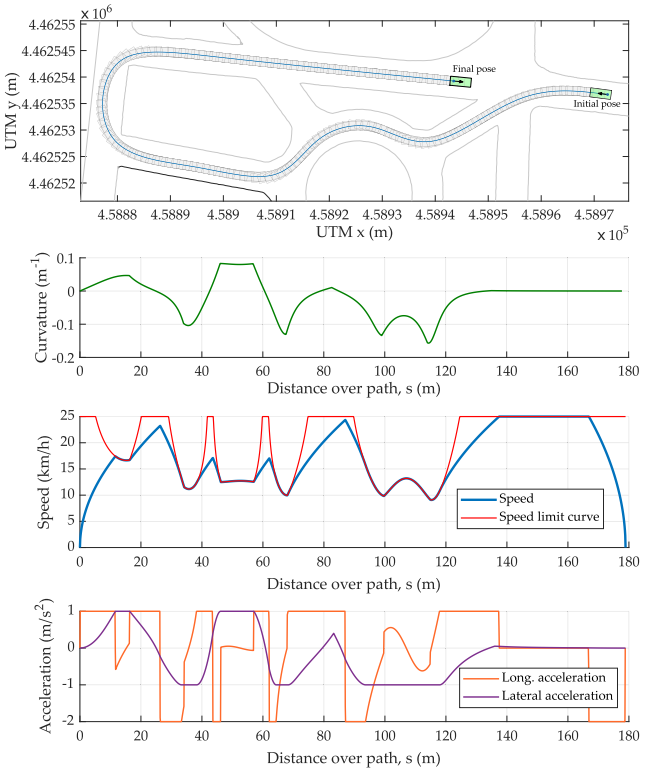


FIGURE 4. In order top-bottom: (i) Example path, (ii) curvature of the path, (iii) speed profile and speed limit curve, and (iv) resulting longitudinal and lateral accelerations.

VI. TRAJECTORY GENERATION ALGORITHM

The main goal of the trajectory generation module is to provide a new trajectory when needed. To that end, the manoeuvre planner module is in charge of analysing the current state of the vehicle and the perceived traffic scene and consequently request a new trajectory to the trajectory generation module.

The proposed strategy comprises several stages:

1) **Motion planning problem initialization**: At this first stage, the motion planning solver defines the search space to explore depending on the planning mode that has been set. The planning modes are introduced below.
2) **Candidates evaluation**: At this stage all the path candidates are evaluated by checking their validity and calculating their costs based on previously defined cost functions.
3) **Candidate selection**: Among the valid evaluated candidates, one is selected based on their costs values.
4) **Final trajectory calculation**: Once the best candidate is chosen, the speed profile is calculated taking into account the maximum accelerations to ensure comfort inside the vehicle.

The general motion planning algorithm is depicted in Fig. 5. The following subsections provides detailed descriptions of all the algorithm stages enumerated above.

This algorithm is launched when a new trajectory is requested by the manoeuvre planner (see Fig. 1). Since the

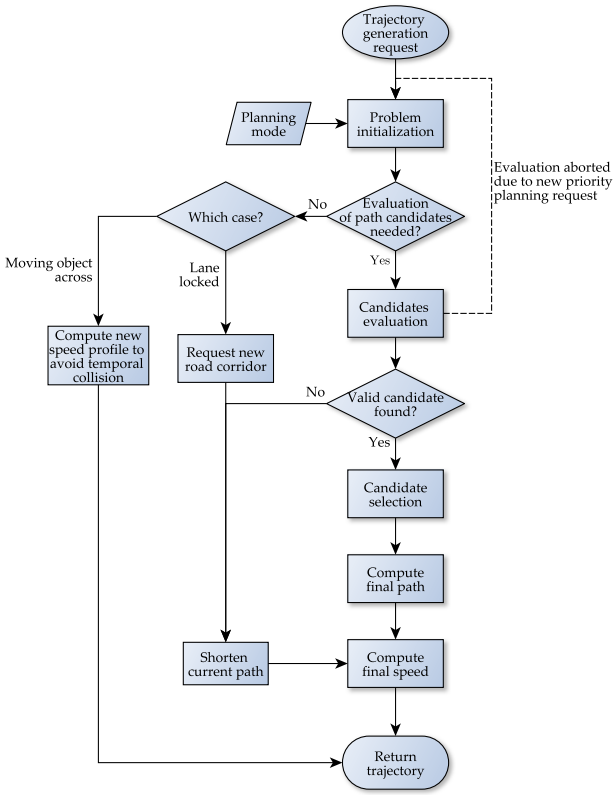


FIGURE 5. General flow diagram of motion planning algorithm.

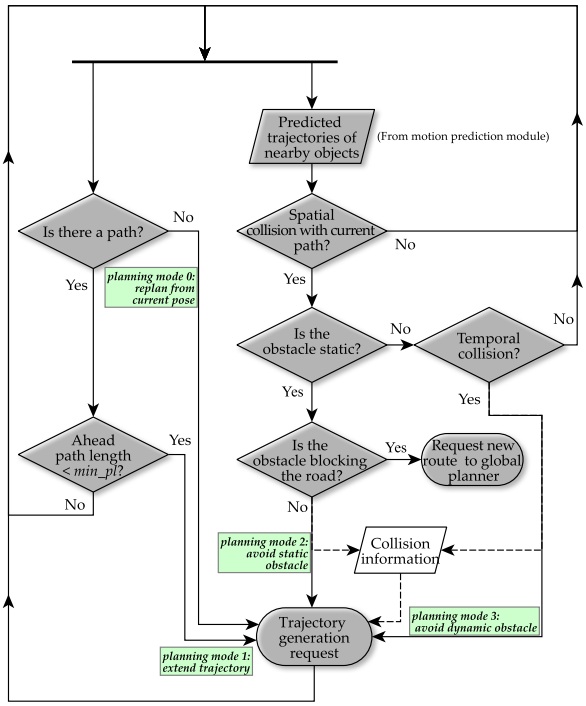


FIGURE 6. Flow diagram of the manoeuvre planner.

manoeuvre planner module is continuously analysing the current situation (collision checking of the current trajectory with perceived objects, and the remaining trajectory length), a mechanism has been included in the trajectory generation in order to prioritize a new planning request to avoid a static or dynamic obstacle (planning modes 2 or 3) in case a non-priority request (planning modes 0 or 1) is being computed.

1) MOTION PLANNING PROBLEM INITIALIZATION

The initialization is the first task performed when a new trajectory is requested. Its main function is to set up the rest planning process based on the current vehicle state and the planning mode requested. When the vehicle follows a planned trajectory, the manoeuvre planner uses the path occupancy polygon to continuously check for collisions with new obstacles that may arise. Thus, a quick response is given to avoid collisions when new obstacles appear (e.g. when detecting an object in a previously occluded area, etc.). To address all possible situations when analysing the predicted motion of nearby objects together with the current planned trajectory of the ego-vehicle, four different planning modes have been stipulated (see Fig. 6): (0) plan from current pose, (1) extend current trajectory, (2) avoid static obstacle and (3) avoid dynamic obstacle. These planning modes influence the initialization of the trajectory generation algorithm that is executed in each new planning request:

- **Plan from current pose**: This planning mode is used when planning for the first time or in emergency situations e.g. when the vehicle is not being able to follow the current trajectory with small control errors.
  When this planning mode is used, the initial pose for path planning is set to the current vehicle pose. The final poses for the candidates of this trajectory section are determined by the next N_{rp} reference points.
- **Extend current trajectory**: If the length of the remaining path is lower that a stated value (min_{pl}) the trajectory generator is called with this planning mode. In this case, a point placed at the 90% of the remaining path length is taken as the initial point for the new trajectory section. As in the previous case, the final poses for the candidates are determined by the next N_{rp} reference points.
- **Avoid static obstacle**: In order to state the reference points for the candidates when a collision of the current trajectory with static obstacle is detected, the free distance from the obstacle to both boundaries of the road corridor (left and right boundaries) is measured to determine if the vehicle is able to pass the obstacle by none, one of both sides. If there is no space enough to avoid the obstacle, it is assumed that the lane is locked and the trajectory is shortened to stop ahead the obstacle maintaining a safe distance, and a new speed profile is calculated with $v_f = 0\ km/h$. In this case, an alternative route to reach the same destination is requested to the global planner. In the cases that the vehicle is able to avoid the obstacle, a set of equidistant reference points R_n is computed over the free space of the perpendicular

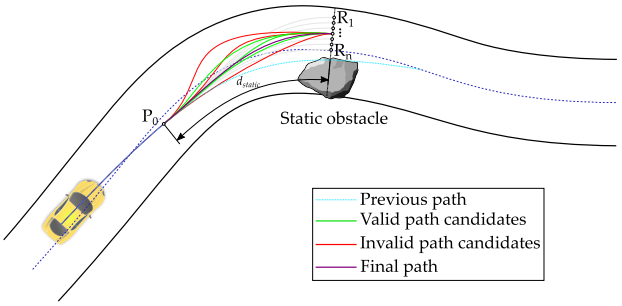


FIGURE 7. Proposed candidates in case an static obstacles must be avoided.

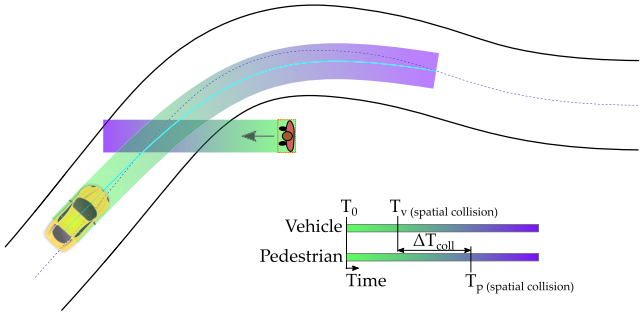


FIGURE 8. Schematic example of motion prediction and collision detection with a dynamic object.

line to the centreline of the road corridor that pass through the centre of the obstacle, as shown in Fig. 7. The point of the current path placed at a distance d_{static} from the obstacle is set as initial point of the new trajectory section. In case the distance from the vehicle to the obstacle is lower than d_{static}, the closest point of the current path to the vehicle is taken as the initial point for candidates.

- **Avoid dynamic obstacle**: In order to detect a future collision with a dynamic obstacle, the predicted trajectory of each dynamic obstacle is used to firstly check if there is a spatial collision with the current trajectory ego-vehicle. If there is a spatial collision, then the time at which the ego-vehicle (T_v) and the obstacle (T_p) will pass through the spatial collision point are compared. When the time difference ($\Delta T_{coll} = |T_v - T_p|$) is lower than a stated safety threshold (T_{th}), i.e. $\Delta T_{coll} < T_{th}$, a future collision is detected. In these cases, the motion direction of the obstacle is used to determine the strategy to avoid it:
  - In the event that the object moves perpendicularly to the trajectory of the vehicle, it is assumed that it will outside the road in the near future, so a new speed profile that avoid the future collision is searched. In case a new speed profile cannot be found, a new path is computed using the same strategy that is described for the static obstacles above.
  - If the obstacle moves in the same direction that the vehicle, a new speed profile is computed using the obstacle speed as the maximum speed.
  - If the obstacle moves in the opposite direction to the vehicle, the strategy used is similar to the one described to avoid static obstacles. Nevertheless, instead of considering the current pose of the obstacle, the predicted pose in the collision point is used.

  As depicted in the schematic example of Fig. 8, a new speed profile is needed when a moving obstacle moves perpendicularly to the trajectory of the vehicle. In order to choose a speed profile that avoids the temporal collision with the obstacle, a set of speed profiles are calculated by decreasing iteratively the maximum speed in the section from the current position vehicle to the collision point of the path, maintaining the design maximum speed (v^{max}) for the speed profile of the remaining path. Thus, when a speed profile complies with $\Delta T_{coll} > T_{th}$ in the spatial collision point, the speed profile of the current trajectory is updated to avoid the collision, and then, after the vehicle reach the collision point, the speed profile calculation is resumed.

Once the problem initialization is finalized, the candidates evaluation is performed, if needed. Note that in cases where the trajectory is shortened or only the speed profile is updated, there is no need to evaluate path candidates.

2) CANDIDATES EVALUATION

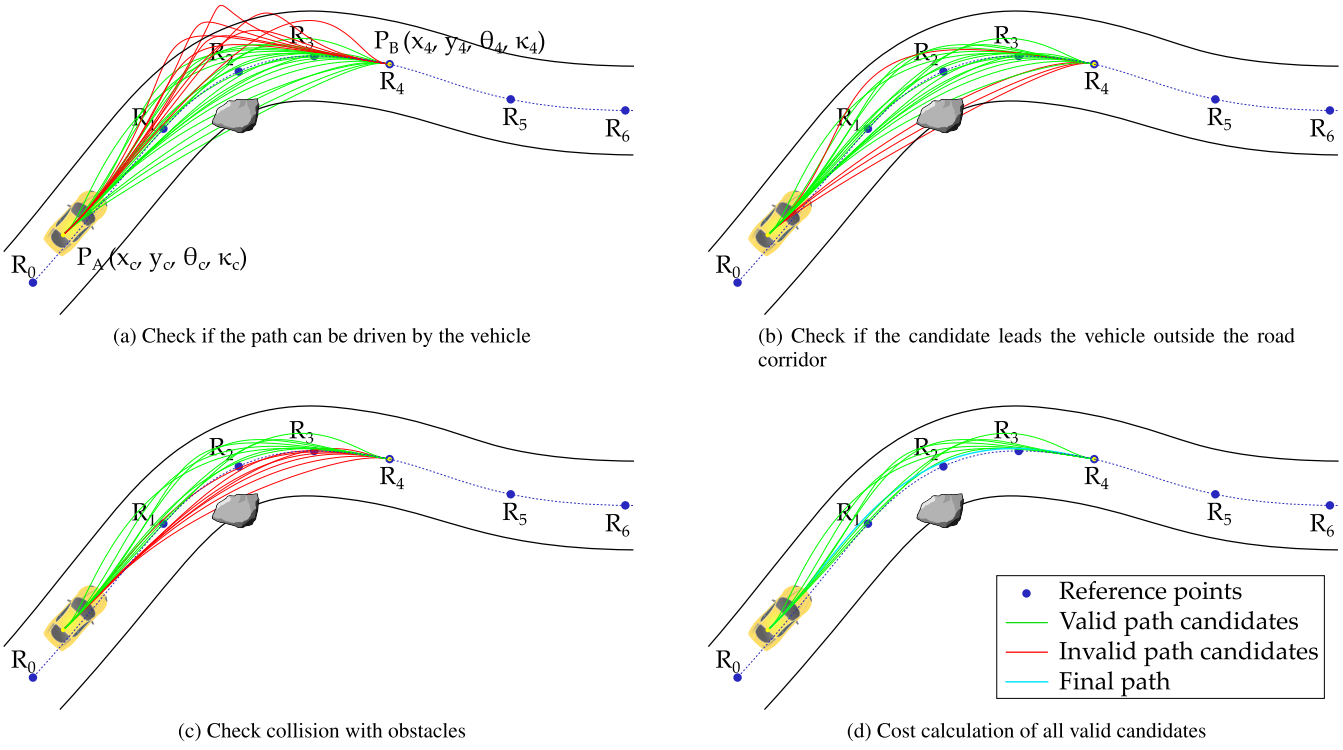
In this stage, all the path candidates that have been selected in the problem initialization are computed and evaluated.

The first step in the evaluation of a candidate is to check its validity by verifying if the path can be driven by the vehicle. To that end, it has to be analyzed that (i) the maximum curvature of the path candidate is below the maximum feasible curvature by the car ($\kappa^{p_c}_{max} < \kappa^{v}_{max}$, see Fig. 9a); (ii) if it would lead the vehicle outside the road corridor (see Fig. 9b) or (iii) if it would lead the vehicle to collide with some obstacle (see Fig. 9c).

After the validity of the path candidate is satisfactorily checked, the cost function used to evaluate the quality of the path is computed (see Fig. 9d). It is not a trivial task to choose a cost function that define the goodness of a path in a given context. Nevertheless, based on the comparison carried out in [23], the following cost function is used in the evaluation of the candidates.

$$J_p = \frac{1}{w_{L_p} L_p} \int_{s_0}^{s_f} \dot{\kappa}(s)^2 + w_{\ddot{\kappa}} \ddot{\kappa}(s)^2 \, ds \tag{17}$$

On the one hand, the first and second derivatives of the curvature reflect the smoothness of the path along the curve. Moreover, the length of the path (L_p) is used to normalize its result. Note also the use of the weight parameter w_{L_p}. The motivation for adding this weight is that in case $w_{L_p} = 1$ (i.e. this weight is not considered), the path that minimizes the cost functions tends to be straight and short in spite of being in curved road sections, thus obtaining almost straight paths in cases in which it should not. In order to avoid that, w_{L_p} is used with values greater than 1.



(a) Check if the path can be driven by the vehicle

(b) Check if the candidate leads the vehicle outside the road corridor

(c) Check collision with obstacles

(d) Cost calculation of all valid candidates

FIGURE 9. Candidates evaluation steps.

3) BEST CANDIDATE SELECTION AND FINAL TRAJECTORY CALCULATION

Among all valid candidates evaluated in the previous stage, the candidate with minimum cost is selected as the path for the final trajectory (see Fig. 9d). In order to obtain the final trajectory, firstly the Bézier curve of the selected path candidate is evaluated to obtain equidistant points.

To do that, firstly the curve polynomials are evaluated using a fine discretization of the parameter t to approximate the relationship between t and the distance over the curve (s). Finally, the values of t to obtain equidistant points are calculated by interpolation.

Depending on the planning mode, the new path section can be concatenated with a section of the previous path or not.

Once a path of equidistant points is obtained, the speed profile is computed as described in section V.

VII. RESULTS AND DISCUSSION

This section presents the experiments carried out in the real platform at proving grounds of the Centre for Automation and Robotics (CAR). A set of different trials have been conducted to validate the proposed motion planning approaches proposed.

A. EXPERIMENTAL PLATFORM

The vehicle used in the trials is a Citroën DS3 which includes hardware modifications for the automated control of throttle, brake, gearbox and steering systems (see Fig. 10). The

FIGURE 10. Experimental platform.

localization relies on a RTK DGPS receiver. To perceive the environment, the vehicle is equipped with a stereo camera and a four-layers LiDAR (Ibeo Lux 4l) installed at the front of the vehicle. In these tests only the LiDAR was used to perceive the environment. The vehicle also includes an on-board computer with an Intel Core i7-3610QE and 8Gb RAM.

B. TRAJECTORY GENERATION RESULTS

The proposed trajectory generator is evaluated in two different driving scenes: (i) an urban-like scenario with sharp

TABLE 3. Path planning setup.

Parameter	Description	Value
d_{sm}	Safety distance around vehicle (m)	0.4
min_{pl}	Minimum trajectory length threshold (m)	55
ϵ_{simp}	Tolerance for centreline simplification (m)	0.25
d_{simp}^{max}	Maximum distance between reference points (m)	7
N_{rp}	Number of reference points used to create candidates	15
N_t	Initial and final tangent vector magnitude evaluation number	10
N_κ	Initial curvature vector magnitude evaluation number	3
m_t^{min}	Minimum normalized tangent vector magnitude	0.3
m_t^{max}	Maximum normalized tangent vector magnitude	1.7
m_κ^{min}	Minimum normalized curvature vector magnitude	0
m_κ^{max}	Maximum normalized curvature vector magnitude	10

TABLE 4. Speed planning setup.

Parameter	Description	Value
v^{max}	Maximum allowed speed (km/h)	20
a_{lat}^{max}	Maximum lateral acceleration (m/s^2)	1.0
a_{acc}^{max}	Maximum positive longitudinal acceleration (m/s^2)	0.4
a_{dec}^{max}	Maximum negative longitudinal acceleration (m/s^2)	0.7

curves and (ii) an urban-like scenario where static and dynamic obstacles must be avoided.

In the experiments, the trajectory generator was set up to compute a total of 4500 path candidates in each planning request. The detailed list of parameters and values used in these experiments are shown in tables 3 and 4.

1) SCENARIO 1: URBAN-LIKE ROUTE THROUGH TIGHT CURVES

This scenario presents a highly sharp road corridor in which the trajectory generator is tested. The route includes the entrance and exit of a small roundabout and several 90°curves in a single narrow lane of 5 meters wide approximately. Fig. 11 shows the valid candidates evaluated in different planning requests during the trial. Notice that invalid candidates has not been plotted in this sub-figures and still the trajectory generation algorithm can choose the final candidate among a number of valid candidates that evaluated in each planning request

Fig. 12 shows the path seen by the vehicle controllers and the real vehicle path during the performed trial. As can be visually noticed, the whole path presents a smooth shape similar to typical vehicle paths obtained when a human is driving.

Fig. 13 shows information of the trajectory tracking during the full trial in scenario 1. In addition to the visual perception of the path smoothness observed in Figs. 11 and 12, in Fig. 13a it can be seen that both lateral and angular errors used in the lateral control present a smooth behaviour, even though the trajectory is updated 13 times during the trial.

TABLE 5. Additional information related to relevant planning requests during the trial in scenario 1.

Planning request ID	Request time-stamp (s)	Planning mode	Planning time (ms)	Valid candidates (%)
1	5.37	0	73.96	33.51
2	5.47	1	64.43	21.80
3	5.56	1	61.04	16.11
4	5.67	1	65.50	12.38
5	8.77	1	64.24	5.93
6	13.15	1	52.98	3.44
7	16.61	1	57.44	17.27
8	20.25	1	51.61	4.04
9	23.96	1	58.29	7.04
10	32.33	1	76.77	35.11
11	42.15	1	42.90	9.80
12	43.50	1	42.01	12.00
13	45.83	1	21.61	14.42

Moreover, Fig. 13d shows the instant and mode of the planning requests performed. The mode of the first planning request is *0* - plan from the current vehicle pose - as there is not any initial trajectory. Since this scenario does not include obstacles, the rest of the planning requests are performed in mode *1* (extend current trajectory). In addition, it can be observed that the first four planning requests were triggered in consecutive samplig periods (100ms). This is caused by the minimum remaining path length requirement, that makes the manoeuvre planner to send planning requests to the trajectory generator with planning mode *1* (extend current trajectory) until this requirement is met.

Finally, Fig. 13c shows the reference speed and measured vehicle speed during the test.

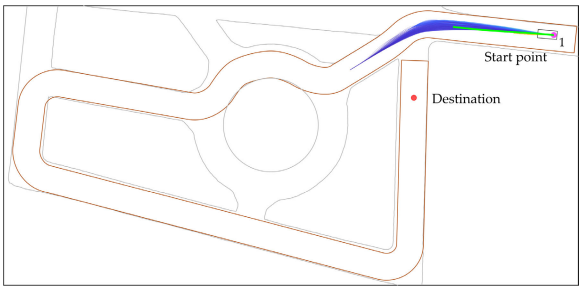
Regarding the computing time, the mean of the total processing time per planning request is 56.37 ms with a standard deviation of 14.69 ms, where the speed planning takes less than 1 ms in all planning requests. More information about each planning request is shown in table 5.

Regarding the speed planning, the maximum positive longitudinal, negative longitudinal and lateral accelerations were set to 0.4 m/s^2, 0.7 m/s^2 and 1.0 m/s^2, respectively. To analyze the resulting behaviour of the vehicle in terms of occupant comfort, Fig. 14 shows a density plot of the real longitudinal and lateral accelerations measured along the trial. This figure shows that most of the acceleration measurements fall within the dashed white rectangle that represent the design acceleration limits.

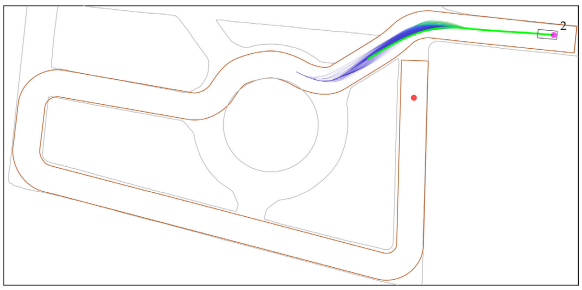
To conclude, this experiment showed that the proposed algorithm is able to generate a number of valid candidates and select the optimum candidates in a few milliseconds even in sharp areas where consecutive curves must be overcome by the vehicle.

2) SCENARIO 2: STATIC AND DYNAMIC OBSTACLES AVOIDANCE

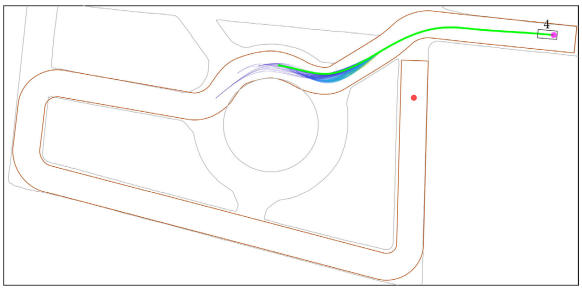
The scenario of this experiment includes static and dynamic obstacles located at different places of the route that the vehicle is following to reach the destination point, which



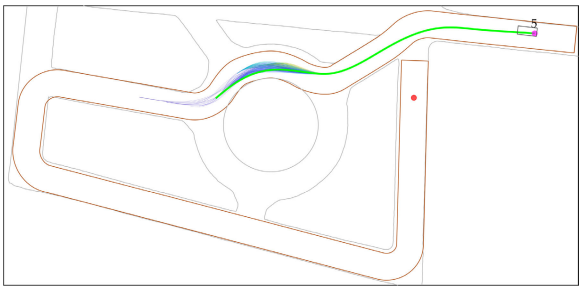
(a) Planning request 1. Instant t=5.367 s



(b) Planning request 2. Instant t=5.467 s

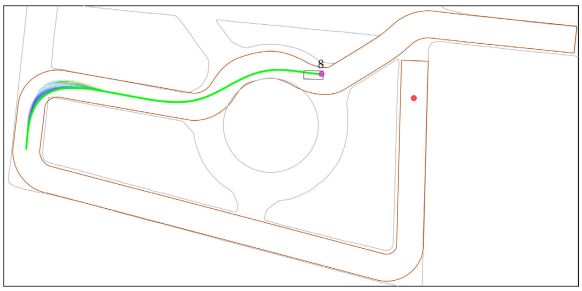


(c) Planning request 4. Instant t=5.666 s

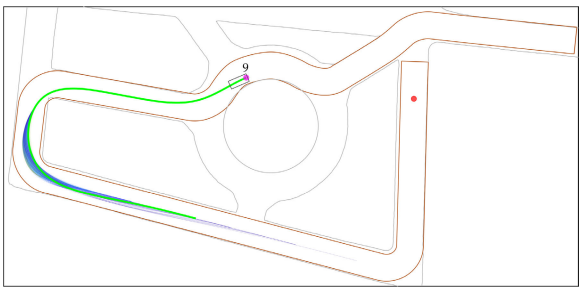


(d) Planning request 5. Instant t=8.765 s

(e) Planning request 6. Instant t=13.154 s

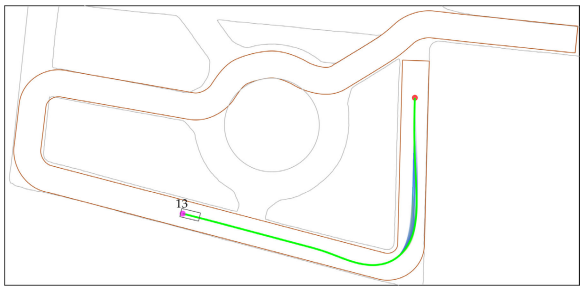


(f) Planning request 8. Instant t=20.253 s

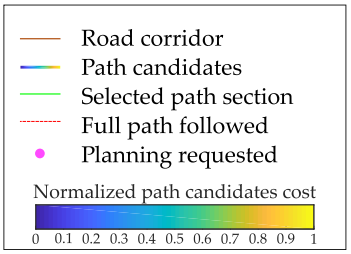


(g) Planning request 9. Instant t=23.960 s

(h) Planning request 12. Instant t=43.495 s



(i) Planning request 13. Instant t=45.825 s



(j) Legend

FIGURE 11. Valid and selected candidates at some of the planning requests during the trial in scenario 1.

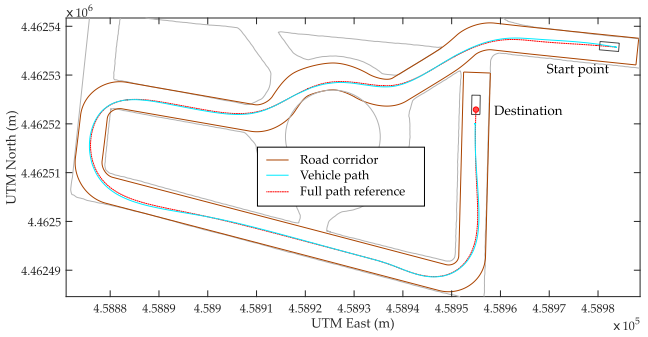
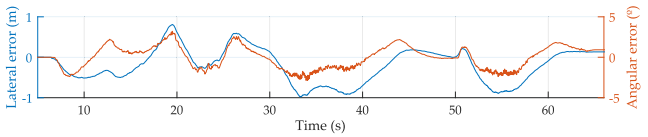
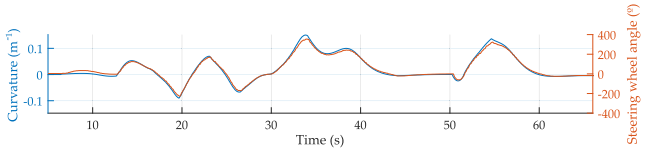


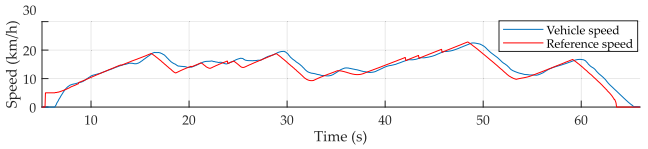
FIGURE 12. Final reference path and vehicle path in the scenario 1.



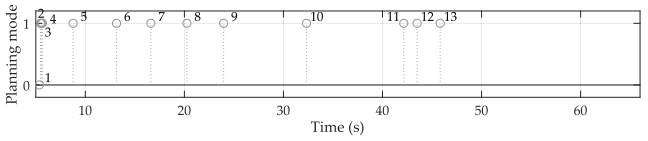
(a) Lateral and angular control errors during the trial.



(b) Path curvature and steering wheel angle during the trial.



(c) Reference and vehicle speed during the trial.



(d) Mode of the planning requests carried out during the trial.

FIGURE 13. Trajectory tracking in scenario 1 trial.

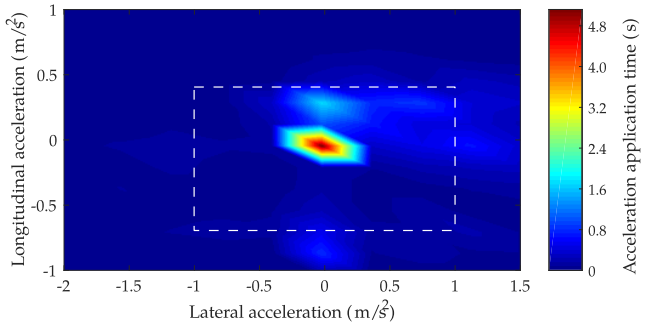


FIGURE 14. Density graph of measured acceleration in the vehicle during the trial in scenario 1.

was set a few meters behind the initial point, as can be seen in Fig. 15. As this scenario includes a greater complexity with respect to the previous one, and consequently a higher

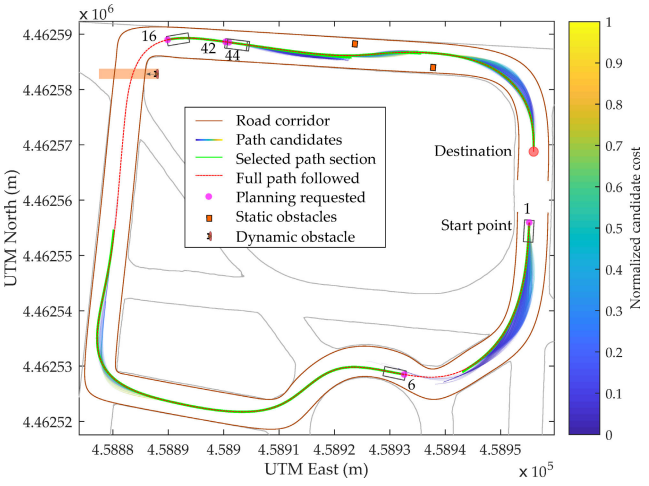


FIGURE 15. Resulting paths in scenario 2.

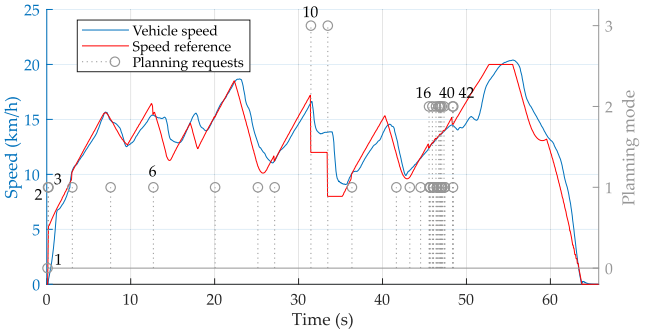


FIGURE 16. Resulting speed profile and planning requests in scenario 2.

number of planning request is carried out, only the valid path candidates of five representative planning requests (*1, 6, 16, 40, 42*) during the trial are plotted to improve the readability of this figure. To complement the understanding of this plot, table 6 shows relevant information about the representative planning requests of Fig. 15. Moreover, the percentage of valid candidates and the processing time in milliseconds at each request are also shown in this table. The average planning time for the whole trial was 37 ms with a standard deviation of 25 ms for the whole experiment.

Furthermore, Fig. 16 shows the speed profile and vehicle speed (left ordinate axis) together with the planning requests during the experiment (right ordinate axis). In this figure it can be seen how and when the planning requests are triggered with different planning modes depending on the situation: at the beginning, the first trajectory is planned from the current vehicle pose (mode *0*). At instant t=31.47 s, a dynamic obstacle is detected and consequently a new planning request is carried out (request ID 10). Later, at instant t=45.54 s, the first static obstacle is detected and a planning request of mode *2* is sent (request ID 16). Afterwards, successive planning requests are closely triggered (request IDs 17-42). This is caused by the small shape and localization changes of the perceived obstacles at different consecutive instants, which leads to launch the candidates evaluation procedure at a

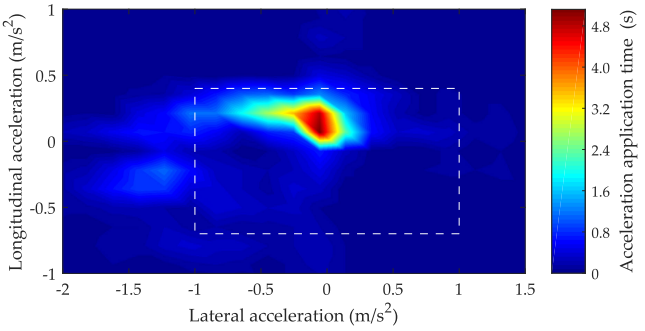


FIGURE 17. Density graph of measured acceleration in the vehicle during the trial in scenario 2.

TABLE 6. Additional information related to relevant planning requests during the trial in scenario 2.

Planning request ID	Request time-stamp (s)	Planning mode	Planning time (ms)	Valid candidates (%)
1	0.092	0	54.45	30.18
6	12.70	1	36.66	33.22
10	31.47	3	1.01	-
16	45.54	2	67.23	34.24
40	48.34	2	70.40	27.56
42	48.43	2	14.49	41.47

higher frequency. Note that the trajectory is quickly corrected to avoid the obstacles satisfactorily even with highly noisy perception information. It is also to be noticed that although the speed profile is smooth and continuous almost at every moment, there are two instants (planning requests 10 and 11), where the sudden incursion of the pedestrian in the road forces to reduce the target speed so that the vehicle keeps in the safe envelope.

In order to analyse the comfort inside the vehicle during the test, Fig. 17 shows a density map to represent the measured longitudinal and lateral accelerations.

Fig. 17 shows how most of the measured acceleration values fall within the limits (marked with a dashed white rectangle in Fig. 17) established in the planning. However, some values are outside mainly due to the joint effect of vibrations induced by road imperfections, slopes and cants in some stretches appearing in this scenario 2. It can also be noticed that since the vehicle must go through more right turns than left ones to reach the destination, more negative (acceleration applied in left direction) than positive lateral acceleration values are measured.

VIII. CONCLUSION AND FUTURE WORK

Based on a modular architecture for automated driving, a real-time motion planning approach for automated driving in urban environments is proposed in this work.

On the one hand, a novel sampling method for generating sets of continuous-curvature path candidates based on quintic Bézier curves is proposed. The proposed path planning method uses a cost-effective primitives evaluation that allows an efficient exploration of a big search space. On the other hand, the speed planning algorithm computes the appropriate speed for each point of the generated path considering limits on the speed, longitudinal and lateral accelerations to ensure comfort inside the vehicle.

The proposed motion planning approach has been implemented in a real automated vehicle and successfully tested in a different scenarios where both static and dynamic obstacles had to be avoided. The results showed the ability of the proposed method in computing feasible, safe and human-like trajectories in real-time, allowing the vehicle to reactively handle dynamic scenarios in real environments.

In order to increase the robustness of the approach in environments where the uncertainties in localization and perception can lead to hazardous situations, future work will focus on considering both uncertainties in the motion planning strategy.

REFERENCES

[1] S. Liu, J. Tang, Z. Zhang, and J.-L. Gaudiot, "Computer architectures for autonomous driving," *Computer*, vol. 50, no. 8, pp. 18–25, 2017.
[2] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Trans. Intell. Vehicles*, vol. 1, no. 1, pp. 33–55, Mar. 2016.
[3] C. Katrakazas, M. Quddus, W.-H. Chen, and L. Deka, "Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions," *Transp. Res. C, Emerg. Technol.*, vol. 60, pp. 416–442, Nov. 2015. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0968090X15003447
[4] A. Artuñedo, J. Godoy, and J. Villagra, "A decision-making architecture for automated driving without detailed prior maps," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 1645–1652.
[5] J. Canny and J. Reif, "New lower bound techniques for robot motion planning problems," in *Proc. 28th Annu. Symp. Found. Comput. Sci.*, Oct. 1987, pp. 49–60.
[6] S. Lazard, J. Reif, and H. Wang, "The complexity of the two dimensional curvatureconstrained shortest-path problem," in *Proc. 3rd Int. Workshop Algorithmic Found. Robotics*, Houston, TX, USA, 1998, pp. 49–57.
[7] J. Villagra and H. Mounier, "Obstacle-avoiding path planning for high velocity wheeled mobile robots," *IFAC Proc. Volumes*, vol. 38, no. 1, pp. 49–54, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667016372901
[8] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, T. Strauss, C. Stiller, T. Dang, U. Franke, N. Appenrodt, and C. G. Keller, "Making bertha drive—An autonomous journey on a historic route," *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 2, pp. 8–20, Apr. 2014.
[9] C. L. Darby, W. Hager, and A. V. Rao, "An hp-adaptive pseudospectral method for solving optimal control problems," *Optim. Control Appl. Methods*, vol. 32, no. 4, pp. 476–502, 2011.
[10] J. T. Betts, "Survey of numerical methods for trajectory optimization," *J. Guid. Control Dyn.*, vol. 21, no. 2, pp. 193–207, Mar./Apr. 1998.
[11] L. S. Pontryagin, *Mathematical Theory of Optimal Processes*. Boca Raton, FL, USA: CRC Press, 1987.
[12] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Dept. Comput. Sci., Iowa State Univ., Ames, IA, Iowa, USA, Tech. Rep. 98-11, 1998.
[13] S. Karaman and E. Frazzoli, "Optimal kinodynamic motion planning using incremental sampling-based methods," in *Proc. 49th IEEE Conf. Decis. Control (CDC)*, Dec. 2010, pp. 7681–7687.
[14] B. Chazelle, "Approximation and decomposition of shapes," in *Algorithmic and Geometric Aspects of Robotics*, J. T. Schwartz and C. K. Yap, Eds. London, U.K.: Lawrence Erlbaum Associates, 1987, pp. 145–185.
[15] J.-C. Latombe, *Robot Motion Planning*, vol. 124. New York, NY, USA: Springer, 2012.
[16] O. Takahashi and R. J. Schilling, "Motion planning in a plane using generalized Voronoi diagrams," *IEEE Trans. Robot. Autom.*, vol. 5, no. 2, pp. 143–150, Apr. 1989.
[17] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Automat.*, vol. 12, no. 4, pp. 566–580, Aug. 1996.

[18] L. Janson, E. Schmerling, A. Clark, and M. Pavone, "Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions," *Int. J. Robot. Res.*, vol. 34, no. 7, pp. 883–921, Jun. 2015.
[19] S. Fleury, P. Soueres, J.-P. Laumond, and R. Chatila, "Primitives for smoothing mobile robot trajectories," *IEEE Trans. Robot. Autom.*, vol. 11, no. 3, pp. 441–448, Jun. 1995.
[20] J. A. Reeds and L. A. Shepp, "Optimal paths for a car that goes both forwards and backwards," *Pacific J. Math.*, vol. 145, no. 2, pp. 367–393, Oct. 1990.
[21] S. Petti and T. Fraichard, "Safe motion planning in dynamic environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Aug. 2005, pp. 2210–2215.
[22] A. Artuñedo, J. Godoy, and J. Villagra, "Smooth path planning for urban autonomous driving using OpenStreetMaps," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2017, pp. 837–842. [Online]. Available: http://ieeexplore.ieee.org/document/7995820/
[23] A. Artuñedo, J. Godoy, and J. Villagra, "A primitive comparison for traffic-free path planning," *IEEE Access*, vol. 6, pp. 28801–28817, 2018.
[24] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numer. Math.*, vol. 1, no. 1, pp. 269–271, Dec. 1959.
[25] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, Jul. 1968.
[26] A. Stentz, "Optimal and efficient path planning for partially-known environments," in *Proc. IEEE Int. Conf. Robot. Automat.*, May 1994, pp. 3310–3317.
[27] T. Gu, J. M. Dolan, and J.-W. Lee, "On-road trajectory planning for general autonomous driving with enhanced tunability," in *Advances in Intelligent Systems and Computing*, Switzerland: Springer, 2016, pp. 247–261. [Online]. Available: http://link.springer.com/10.1007/978-3-319-08338-4_19
[28] T. Gu, J. Snider, J. M. Dolan, and J.-W. Lee, "Focused trajectory planning for autonomous on-road driving," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2013, pp. 547–552. [Online]. Available: http://ieeexplore.ieee.org/document/6629524/
[29] X. Li, Z. Sun, D. Cao, Z. He, and Q. Zhu, "Real-time trajectory planning for autonomous urban driving: Framework, algorithms, and verifications," *IEEE/ASME Trans. Mechatronics*, vol. 21, no. 2, pp. 740–753, Apr. 2016.
[30] Z. Boroujeni, D. Goehring, F. Ulbrich, D. Neumann, and R. Rojas, "Flexible unit A-star trajectory planning for autonomous vehicles on structured road maps," in *Proc. IEEE Int. Conf. Veh. Electron. Saf. (ICVES)*, Jun. 2017, pp. 7–12.
[31] J. Ziegler and C. Stiller, "Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2009, pp. 1879–1884.
[32] J. Villagra, V. Milanés, J. Pérez, and J. Godoy, "Smooth path and speed planning for an automated public transport vehicle," *Robot. Auto. Syst.*, vol. 60, no. 2, pp. 252–265, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S092188901100203X
[33] Y. Zhang, H. Chen, S. L. Waslander, J. Gong, G. Xiong, T. Yang, and K. Liu, "Hybrid trajectory planning for autonomous driving in highly constrained environments," *IEEE Access*, vol. 6, pp. 32800–32819, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8375948/
[34] J. Godoy, A. Artuñedo, and J. Villagra, "Self-generated OSM-based driving corridors," *IEEE Access*, vol. 7, pp. 20113–20125, 2019.
[35] X. Qian, I. Navarro, A. de La Fortelle, and F. Moutarde, "Motion planning for urban autonomous driving using Bézier curves and MPC," in *Proc. IEEE 19th Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2016, pp. 826–833. [Online]. Available: http://ieeexplore.ieee.org/document/7795651/
[36] J. Ziegler and C. Stiller, "Fast collision checking for intelligent vehicle motion planning," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2010, pp. 518–522. [Online]. Available: http://ieeexplore.ieee.org/document/5547976/
[37] K. Hormann and A. Agathos, "The point in polygon problem for arbitrary polygons," *Comput. Geometry*, vol. 20, no. 3, pp. 131–144, 2001. [Online]. Available: https://www.inf.usi.ch/hormann/papers/Hormann.2001.TPI.pdf

ANTONIO ARTUÑEDO received the B.Sc. degree in electrical engineering from the Universidad de Castilla—La Mancha, Spain, in 2011, the M.Sc. degree in industrial engineering from the Universidad Carlos III de Madrid, in 2014, and the Ph.D. degree in automation and robotics from the Technical University of Madrid (UPM), Spain, in 2019, in the AUTOPIA Program. He joined the Centre for Automation and Robotics (CSIC-UPM), in 2013, where he has been working on both national and European research projects in the scope of autonomous vehicles. His research interests include system modeling and simulation, intelligent control, motion planning, and decision making systems.

JORGE VILLAGRA graduated in industrial engineering from the Universidad Politécnica of Madrid, in 2002, and the Ph.D. degree in real-time computer science, robotics and automatic control from the École des Mines de Paris, France, in 2006. He was first granted with a three years CIFRE Program in PSA-Peugeot-Citroën and then with a Postdoctoral Fellowship at a Joint Research Unit INRIA-Mines ParisTech, France. The results of the Ph.D. were granted with the Prize for the Best dissertation in Automatic Control in France, in 2006. From 2007 to 2009, he was a Visiting Professor at the University Carlos III, Spain. He then received a three years JAEDoc Fellowship at the AUTOPIA Program in the Center for Automation and Robotics UPM-CSIC, Spain, where he spent one additional year funded by a research contract. From 2013 to August 2016, he led the Department of ADAS and Highly Automated Driving Systems, Ixion Industry and Aerospace SL, where he also coordinated all the activities in the EU R&D Funding Programmes. He has been leading AUTOPIA Program, CSIC-UPM, since October 2016. He has developed his research activity in six different entities with a very intense activity in project setup and management, through more than 30 international and national R&D projects, where he is or has been IP of ten of these projects. He has published over 85 articles in international journals and conferences on autonomous driving, intelligent transportation systems, dmodel-free control, and new probabilistic approaches for embedded components in autonomous vehicles.

JORGE GODOY was born in Maracay, Venezuela, in 1986. He received the degree in electronic engineering from the Universidad Simón Bolívar, in 2008, and the M.E. and Ph.D. degrees in automation and robotics from the Universidad Politécnica de Madrid, in 2011 and 2013, respectively. In 2009, he was granted with a Predoctoral JAE Fellowship from CSIC for researching on autonomous vehicles at the Centre of Automation and Robotics, CSIC-UPM. From 2013 to 2017, he was the Technical Coordinator of the AUTOPIA Program, funded by research contracts from National and European research projects. In November 2017, he was granted with a Juan de la Cierva fellowship for postdoctoral research at the Universidad Politécnica de Madrid. His research interests include intelligent transportation systems, autonomous driving, path planning, and embedded AI-based control for autonomous vehicles.