IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 18, 2018, accepted May 18, 2018, date of publication May 24, 2018, date of current version June 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2839884

A Primitive Comparison for Traffic-Free Path Planning

ANTONIO ARTUÑEDO[1], JORGE GODOY[2], AND JORGE VILLAGRA[1]
[1]Centre for Automation and Robotics CSIC-UPM, Spanish National Research Council, 28500 Madrid, Spain
[2]Centre for Automation and Robotics CSIC-UPM, Universidad Politécnica de Madrid, 28040 Madrid, Spain

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

This work was supported by the Spanish Ministry of Economy, Industry and Competitiveness through the National Project TCAP-AUTO under Grant RTC-2015-3942-4.

ABSTRACT Motion planning in on-road urban driving is usually stated as an optimization problem in a multi-dimensional space that presents a high complexity in obtaining a global optimal solution. In that sense, a great amount of different approaches to solve this problem coexist in the literature. However, to the best of our knowledge, there is no prior work studying how to choose the best strategy in this multi-dimensional space. This paper presents an in-depth analysis of interpolation curve planners based on continuous curvature Bézier compositions. To that end, a comparison framework to benchmark different path-planning primitives for on-road urban driving is proposed, and the evaluation of different primitive configurations and optimization techniques for path-planning is carried out. In addition, the results are openly published together with its consequent analysis, based on a set of key performance indicators related to the aforementioned main features.

INDEX TERMS Path planning, path optimization, autonomous driving.

I. INTRODUCTION

Autonomous driving involves the integration of a number of technologies related to perception, localization, decision-making, motion-planning and control. Among them motion-planning is particularly relevant as it plays a key role in ensuring driving safety and comfort [1].

The robotics community has been intensively working over the last 30 years in path planning problems. Although many of the proposed algorithms are able to cope with a wide range of situations and contexts, they often demand computation-intensive algorithms, feasible for low speed motion patterns. However, for on-road autonomous driving, determinism is necessary at high sampling rates. In this context, optimality can be slightly sacrificed at the expense of safe human-adapted paths. 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 incommensurable cost terms are combined to produce a specific behavior.

Two main drawbacks emerge from this approach: (i) these techniques often provide scenario-dependant solutions, which may cause wrong behaviors in general real driving on urban roads [3]; (ii) to guarantee reactivity, the trajectories need to be exhaustively sampled and evaluated in a high-dimensional space, which is computationally expensive. To cope with these limitations, some works (e.g. [4], [5]) propose a higher-level decision maker able to select the right cost set and sampling scale for different situations.

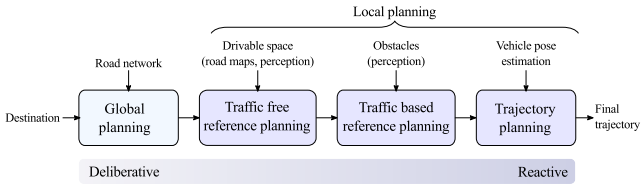

FIGURE 1. Diagram of the motion-planning architecture.

This hybrid architecture aims at achieving the best trade-off between deliberative and reactive architectures in a computationally efficient planning framework, structured around the following three phases, whose inter-connections are depicted in Fig. 1:

- A traffic-free reference planning, that generates a traffic-free reference trajectory for each drivable lane assuming that the geometry of the map is known and that the vehicle will follow the reference without any traffic interference.

- A traffic-based reference planning, that provides a traffic-based motion reference by performing reference variation to respond to static and moving objects.
- A local trajectory planning, that generates a parametric trajectory for tracking control to carry out the planned reference.

There exist many path generation patterns useful for the traffic-free reference stage, namely clothoids [6], arc-lines [7], spiral [8] and different variations of splines (e.g [9]–[11]). The criteria to choose the most adapted primitive are of course computational cost, safety and comfort, but also tunability and stability. Indeed, the resulting path has to be not only confined to the drivable space and needs to be compliant with fixed comfort-based acceleration and jerk bounds. Tunability also matters as designers may need to modify the planning strategy to avoid parameter overfitting following the considered scenarios and/or different user preferences. In this connection, stability is also very important to avoid jerkiness in steering/braking when such context-based retuning is conducted.

However, to the best of our knowledge there is no prior work studying how to choose the best strategy in this multidimensional space. This article presents an in-depth analysis of interpolation curve planners, which use the current pose of the vehicle and intermediate waypoints within a short-medium time horizon. More specifically, continuous curvature Bézier-based path planners will be exhaustively compared in different driving scenarios, shedding some light into the best design choice for each particular situation and goal.

The main contributions of this paper are (i) a comparison framework to benchmark different path-planning primitives for on-road urban driving, (ii) the evaluation of different primitive configurations and optimisation techniques for path-planning, and (iii) the open publication of the results and its consequent analysis, based on a set of key performance indicators (KPI) related to the aforementioned main features.

The remainder of the article is organized as follows. Section II gives an overview of the curve primitives used for human-like path-planning in autonomous driving as well as the most similar works aiming at proposing benchmarking frameworks for different path-planning strategies. Section III is devoted to formally state the problem to be solved, introducing the primitives to compare and the considered path planning approaches and algorithms. The adopted comparison framework is detailed in Section IV. Section V shows firstly some relevant results of the full set of conducted experiments, and then a discussion to propose some guidelines for practitioners. Finally, Section VI draws some concluding remarks.

II. LITERATURE OVERVIEW

The problem of finding an optimal path subject to holonomic constraints avoiding obstacles is known to be PSPACE-hard [12]. Significant research attention has been directed towards studying approximate methods or particular solutions of the general motion planning problem.

Since for most problems of interest in autonomous driving, exact algorithms with practical computational complexity are unavailable [13], numerical methods are often used. These techniques 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 guarantees for converging to an optimal solution. These approximate methods for path and trajectory planning can be divided in three main families [2]:

- Variational methods, that project the infinite-dimensional function space of trajectories to a finite-dimensional vector space. Direct methods approximate the solution to the optimal path with a set of parameters obtained with different types of non-linear continuous optimization techniques, often collocation-based integrators [14] or pseudoespectral approaches [15]. Indirect methods [16], in turn, solve the problem by finding solutions that satisfy the optimality conditions established by the Pontryagin's minimum principle [17].
- Graph-based search methods, that 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 desired path is found by performing a search for a minimum-cost path in such a graph. There is a significant number of strategies to construct that graph in the most efficient way, but they can be grouped in two main families: geometric methods, such as cell decomposition [18], visibility graphs [19] or Voronoi diagrams [20], and sampling-based methods [21], [22]. The latter deserves particular focus, as is particularly adapted to structured environments, where different steering functions (e.g [23], [24]) or motion primitives (e.g [25]) explore the reachability of the free configuration space. Once the graph is built, different strategies exist also to conduct the graph search in the most dependable way (e.g. Dijkstra [26], A* [27], D* [28]...).
- 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 [29] and their variants (e.g. [30]), always looking for the best trade-off between completeness and computational cost.

Since the applicability of variational methods is limited by their convergence to only local minima, graph-search methods try to overcome the problem by performing global search 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 which case incremental search techniques 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.

Given this context, a double stage planning approach, where a computationally-efficient reasonable traffic-free path is computed, seems the best trade-off for urban environments, where roads are usually well-structured and reasonably digitalized. Some authors have deepened into the potential and limitations of this approach, considering to that end a very specific variety of graph-based search methods, with different path primitives and optimization schemes (e.g. [3], [31]). However, as mentioned in Section I, there exist a significant number of possible variations within this scheme that can significantly affect the resulting path.

In some cases, the curvature continuity cannot be guaranteed, and in many others the primitives are either difficult to parameterize or non-analytical, and therefore computationally expensive and/or unpredictable. Bézier curves have a closed-form expression and an intuitive way to choose their parameters. Although some previous work proposed combinations of symmetric curves [32] or smooth concatenations of cubic Bézier curves and segments [33], in this work uniform relaxed B-splines are used, concatenating continuous-curvature Bézier curves, so that more flexible solutions can be obtained.

Once a set of path primitive is selected, the reference path planning becomes an optimization problem whose goal is to select the minimum number of waypoints needed to connect an origin and a destination, taking into consideration road geometry and comfort constraints. Multiple optimization criteria and methods can be applied to that end [34]–[36]. However, the lack of an absolute trajectory quality indicator makes it hard to determinate the most appropriate optimality criteria. This paper presents a comparison framework for an on-road traffic-free path planning scheme. The need of benchmarks in robotics is formulated in [37] and some well-known exists for perception (e.g. KITTI [38]), but is rare to find systematic studies for the decision-making stage of automated vehicles. An exception is CommonRoad [39], a recent approach to benchmark motion-planning in autonomous driving. In this work, the focus is on the specific path planning aforementioned presented, but adopting a similar syntax upon which CommonRoad is based: composable aspects of the comparison (scenario, optimization method, primitive, function to be minimized, etc.) that can be referenced to with a unique ID.

III. PROBLEM STATEMENT

The goal of path planning algorithms is to find a feasible path to drive from an initial point (typically the current pose of the vehicle) to a target point, while often minimizing a predefined criteria. This work focuses on path planning for autonomous driving in typical structured environments such as roads or highways, where the non-holonomic constraints of the vehicle cannot be ignored. The dynamic restrictions that are commonly taken into account are (i) the maximum curvature that the vehicle is able to handle and (ii) the continuity of the curvature along the planned path. It is worth to remark the importance of (ii), since discontinuities in the curvature do not allow automated vehicles to track the path. In addition to these constraints, the path is required to be comfortably driven by the vehicle i.e. the turning angle and turning speed of steering maneuvers should not lead to strong lateral accelerations. As a result, the path planning strategy should minimize the variability of the curvature along the computed path.

Since elevation increment of the considered space for path planning is insignificant in most of the structured environments, the path planning problem is typically performed in a 2D plane. This allows to formally define the path planning problem in a general way as follows:

$$\begin{aligned} &\arg\min_{x_a} J(x_a,\ D_s,\ V) \\ &\text{subject to } l_b \leq x_a \leq u_b \end{aligned} \tag{1}$$

where:

- x_a is a vector containing all variables to be optimized. Its size and variables can vary depending on the path planning approach, as presented in following subsections.
- l_b and u_b are the lower and upper bounds of the values of x_a in order to constrain the search space of the algorithms. The bounds values depend on the scenario and the approach.
- $D_s \subset \mathbb{R}^2$ is the drivable space of scenario s.
- $V = [p_0,\ p_f,\ l_{tw},\ \kappa^v_{max}]$ includes vehicle-related information:
  - $p_0 = [x_0, y_0, \theta_0, \kappa_0]$ is the initial vehicle pose where x_0 and y_0 are the initial coordinates, θ_0 the heading and κ_0 curvature.
  - $p_f = [x_f, y_f, \theta_f, \kappa_f]$ is the final vehicle pose where x_f and y_f are the initial coordinates, θ_f the heading and κ_f curvature.
  - l_{tw} is the track width of the vehicle.
  - κ^v_{max} is the maximum curvature the vehicle is able to handle.

In this work, we consider different types of primitives, optimization methods and algorithms, cost functions as well as different initial and final heading and curvature configurations. These are described in detail in the following subsections.

A. PRIMITIVES TO COMPARE

Interpolation curve planners use the current pose and curvature of the vehicle and some waypoints in order to obtain the final path to be followed. This path is required to have continuous curvature and has to be as much efficient as possible. To that end, different interpolation methods based on Bézier curves will be compared, always guaranteeing G^2 continuity along the path. These primitives present some advantages that make them suitable for path planning in autonomous driving: fast curve and curvature calculation using analytic expressions, fast collision-checking using Bézier curves

properties such as convex hull property, curve-line intersection, etc. Nevertheless, diverse piecewise Bézier curves feature different stability [40], which is an important property that defines the impact of a small local change in the position of one waypoint on the whole curve shape. In general, with interpolating splines there is a trade-off between stability and higher-order [41].

In this work, two possible variations will be explored: (i) piecewise B-splines, defined as composites of cubic Bézier curves, and (ii) quintic Bézier curves.

1) CUBIC B-SPLINES CURVES

They allow to generate a curve that goes through a set of given waypoints. The curve generated is a concatenation of n plane cubic Bézier sections which are defined generically as follows:

$$C_j(t) = \sum_{i=0}^{d_b} P_i^j B_{i,d_b}(t), \quad t \in [0, 1],\ j = 1...n \tag{2}$$

being $B_{i,d_b}(t) = \binom{d_b}{i} t^i (1-t)^{d_b - 1}$ the Bernstein polynomials, P_i^j the control points of Bézier section j, and d_b the degree of the Bézier curve ($d_b = 3$ in the case of cubic curves).

Continuity at joints can be guaranteed in B-splines by forcing the first and second derivatives of two contiguous Bézier sections to be equal in the joints, so that the following expression is verified:

$$2P_2^i - P_1^i = 2P_1^{i+1} - P_2^{i+1} = A_i, \quad i = 1...n-1 \tag{3}$$

where A_i are intermediate control points. The position of these intermediate points is fixed by the $n+1$ points to be interpolated S_i, solving the next linear equation system:

$$\begin{pmatrix} 1 & & & & \\ 4 & 1 & & & \\ 1 & 4 & 1 & & \\ & 1 & 4 & 1 & \\ & & & \ddots & 1 \\ & & & 1 & 4 \\ & & & & 1 \end{pmatrix} \begin{pmatrix} A_0 \\ A_1 \\ A_2 \\ A_3 \\ \vdots \\ A_{n-1} \\ A_n \end{pmatrix} = \begin{pmatrix} S_0 \\ 6S_1 - S_0 \\ 6S_2 \\ 6S_3 \\ \vdots \\ 6S_{n-1} - S_n \\ S_n \end{pmatrix} \tag{4}$$

Since the curvature of a cubic Bézier section is continuous, the C^2 continuity is guaranteed along the whole curve (and consequently G^2 continuity).

The curve (2) whose coefficients are computed solving equations (3) and (4) implicitly verifies that the second derivative is zero at its initial and end points ($A_0 = S_0$ and $A_n = S_n$). However, the linear system 4 can be modified so as to set additional boundary conditions. In that sense, the possibilities we consider include: setting the initial and/or end tangent vector as well as setting the initial tangent and curvature vectors. These end conditions are further explained bellow. It is worth to mention that cubic B-splines present excellent stability as a change in one of its waypoints only affects its adjacent Bézier sections.

a: INITIAL HEADING SETTING

To set the initial heading of the curve, the tangent vector at the initial point must be forced. To that end, the first derivative of (2) can replace the second equation of the linear system (in eq. 4):

In this case, the stability is lower than the previous case, where initial tangent is not imposed.

$$2A_0 + A_1 = 3S_0 + \vec{t}_0 \tag{5}$$

where $\vec{t}_0$ is the tangent vector at the initial point.

Then, the resulting linear system to solve is in Eq. 6.

$$\begin{pmatrix} 2 & 1 & & & \\ 1 & 4 & 1 & & \\ & 1 & 4 & 1 & \\ & & & \ddots & 1 \\ & & & 1 & 4 \\ & & & & 1 \end{pmatrix} \begin{pmatrix} A_0 \\ A_1 \\ A_2 \\ \vdots \\ A_{n-1} \\ A_n \end{pmatrix} = \begin{pmatrix} 3S_0 + \vec{t}_0 \\ 6S_1 \\ 6S_2 \\ \vdots \\ 6S_{n-1} - S_n \\ S_n \end{pmatrix} \tag{6}$$

b: INITIAL AND FINAL HEADING SETTING

The final heading can be also forced in the same way yielding

$$\begin{pmatrix} 2 & 1 & & & \\ 1 & 4 & 1 & & \\ & 1 & 4 & 1 & \\ & & & \ddots & \\ & & 1 & 4 & 1 \\ & & & 1 & 2 \end{pmatrix} \begin{pmatrix} A_0 \\ A_1 \\ A_2 \\ \vdots \\ A_{n-1} \\ A_n \end{pmatrix} = \begin{pmatrix} 3S_0 + \vec{t}_0 \\ 6S_1 \\ 6S_2 \\ \vdots \\ S_{n-1} \\ 3S_n - \vec{t}_n \end{pmatrix} \tag{7}$$

where $\vec{t}_n$ is the tangent vector at the last point.

c: INITIAL HEADING AND CURVATURE SETTING

In addition to the initial heading , the curvature can be also set at the initial point. In this particular case, the first and second derivatives of the B-spline equation (2) evaluated at S_0 can be written as follows:

$$A_1 - A_0 = \vec{t}_0 - \frac{1}{2}\vec{\kappa}_0 \tag{8}$$

$$6A_0 = 6S_0 + \vec{\kappa}_0 \tag{9}$$

where $\vec{\kappa}_0$ is the second derivative vector (acceleration) at the initial point of the curve. This vector can be obtained as

$$\vec{\kappa}_0 = \frac{d|\vec{t}_0|}{dt}\vec{T}_0 + \kappa_0 |\vec{t}_0|^2 \vec{N}_0 \tag{10}$$

where $\frac{d|\vec{t}_0|}{dt}$ is the rate of change of the tangent module with respect to the independent variable t of the parametric curve, and $\vec{T}_0$ and $\vec{N}_0$ are the unit tangent and normal vector at the initial point, respectively.

If constant speed (in terms of differential geometry) is considered, the tangential part of eq. (10) is null, and therefore

the resulting linear system (4) can be written as follows

$$\begin{pmatrix} -1 & 1 & & & \\ 1 & 4 & 1 & & \\ & 1 & 4 & 1 & \\ & & & \ddots & \\ & & 1 & 4 & 1 \\ 6 & & & & \end{pmatrix} \begin{pmatrix} A_0 \\ A_1 \\ A_2 \\ \vdots \\ A_{n-1} \\ A_n \end{pmatrix} = \begin{pmatrix} \vec{t}_0 - \frac{1}{2}\vec{\kappa}_0 \\ 6S_1 \\ 6S_2 \\ \vdots \\ S_{n-1} \\ 6S_0 + \vec{\kappa}_0 \end{pmatrix} \quad (11)$$

Note that in this case the stability is totally lost since a small change in the position of one waypoint changes the shape of the whole curve.

2) QUINTIC BÉZIER SPLINES

They have more degrees of freedom than cubic ones, which has some advantages but also drawbacks. A higher degree provides more control points, improving the controllability of the curve and consequently the controllability of the set of Bézier sections that make up the complete path. Thus, the initial and final pose (including initial and final curvature) could be imposed, making easier re-planning tasks when the vehicle is on motion and the curvature cannot suddenly change. However, coherent values for tangent and curvature vectors at intermediate joints are also needed to place the control points of each Bézier section. As they are not known in advance, heuristic rules can be used, as proposed in [42], to estimate convenient values of first and second derivatives at intermediate waypoints ($S_i \in [S_1, S_{n-1}]$) of the quintic Bézier spline, able to guarantee t the curve smoothness at joints. These estimations can be used to calculate the final curve but also can be taken as initial guesses to apply an optimization algorithm. The considered heuristics to calculate the first and second derivatives are described below.

a: FIRST DERIVATIVE

The first derivative at waypoint S_i of the spline (tangent vector $\vec{t}_i$) is determined as follows: On the one hand, the orientation of $\vec{t}_i$ is perpendicular to the bisector of the angle formed by vector $\vec{v}_a$ and $\vec{v}_b$ where $\vec{v}_a = S_i - S_{i-1}$ and $\vec{v}_b = S_{i+1} - S_i$. On the other hand, the magnitude of $\vec{t}_i$ is set to the minimum euclidean distance between the current point (S_i) and its two neighboring points (S_{i-1}, S_{i+1}) multiplied by a scaling factor f_t. Thus $|\vec{t}_i| = f_t\, min(|\vec{v}_a|, |\vec{v}_b|)$.

Both magnitude and orientation of $\vec{t}_i$ have a high influence on the final curve geometry. Therefore, they are variables to be optimized in some of the methods compared in this work.

b: SECOND DERIVATIVE

To determine the second derivative the heuristic proposed in [42] is applied. To estimate the curvature $\vec{\kappa}_i$ at S_i, this approach uses two cubic Bézier sections (one from S_{i-1} to S_i and another from S_i to S_{i+1}). The second derivative is applied at S_i in both curves, using the tangent vectors at the previously mentioned points ($\vec{t}_{i-1}, \vec{t}_i$ and $\vec{t}_{i+1}$:

$$\vec{\kappa}_i^a = 6S_{i-1} + 2\vec{t}_{S_{i-1}} + 4\vec{t}_{S_i} - 6S_i \quad (12)$$

$$\vec{\kappa}_i^b = -6S_i - 4\vec{t}_{S_i} - 2\vec{t}_{S_{i+1}} + 6S_{i+1} \quad (13)$$

Then a weighted average is calculated with the curvature vectors of both curves at S_i, in order to obtain the estimated curvature vector $\vec{\kappa}_i^e$:

$$\vec{\kappa}_i^e = \alpha\, \vec{\kappa}_i^a + (1-\alpha)\, \vec{\kappa}_i^b \quad (14)$$

where $\alpha = \frac{|S_i - S_{i-1}|}{|S_i - S_{i-1}| + |S_{i+1} - S_i|}$.

The six control points of each Bézier segment can be calculated equalling first and second derivatives of the quintic Bézier equations in two consecutive sections:

$$P_0^i = S_i = P_5^{i-1} \quad (15)$$

$$P_1^i = S_i + \frac{1}{5}\vec{t}_i \quad (16)$$

$$P_2^i = \frac{1}{20}\vec{\kappa}_i + 2P_1^i - S_i \quad (17)$$

$$P_3^i = \frac{1}{20}\vec{\kappa}_{i+1} + 2P_4^i - S_{i+1} \quad (18)$$

$$P_4^i = S_{i+1} - \frac{1}{5}\vec{t}_{i+1} \quad (19)$$

$$P_5^i = S_{i+1} = P_0^{i+1} \quad (20)$$

where P_m^i ($m \in \mathbb{N} : m \in [0, 5]$ are control points of Bézier section i.

3) HEADING AND CURVATURE AT THE INITIAL AND END POINTS

A summary of the cases covered depending on the primitive used are shown in the table below:

TABLE 1. Cases covered regarding the imposition of initial/final heading and curvature.

	Initial heading (h_0)	Final heading (h_f)	Initial curvature (κ_0)	Final curvature (κ_f)
cubic B-spline	No	No	No	No
	Yes	No	No	No
	Yes	Yes	No	No
	Yes	No	Yes	No
quintic Bézier spline	Yes	Yes	Yes	Yes

B. CONSIDERED PATH PLANNING APPROACHES

The approaches considered and compared in this work cover the most common state of the art path planning techniques that are based on Bézier curve primitives, as well as some proposed novel strategies.

They all intend to find the most suitable set of intermediate waypoints. To that end, some steps are usually carried out: firstly, the centreline of the drivable space is estimated from the drivable space boundaries. Over the centreline (i) a set of *reference points* is selected. After that, (ii) the position of the *reference points* is optimized. Some existing approaches stop at this point and compute the final path by interpolating among the optimized *reference points* by means of different curve primitives. Other approaches use the optimized *reference points* to calculate intermediate waypoints usually,

called *seeding points*. These latter approaches then (iii) optimize the *seeding points* based on different methods.

Different techniques used for each of the three steps stated above are further described in following subsections.

1) REFERENCE POINTS SELECTION METHOD (*RS*)

The first step is to select a set of *reference points* over the centreline. To that end, three different methods are considered:

- Equidistant points over the centreline (*E*).
- Douglas-Peucker simplification algorithm [43] (*D*). This algorithm is based on tolerance of perpendicular point-to-edge distance to extract the simplified line.
- Opheim simplification algorithm [44] (*O*). Unlike Douglas-Peucker algorithm, the search area in Opheim algorithm is constrained by both a perpendicular and a radial maximum distances.

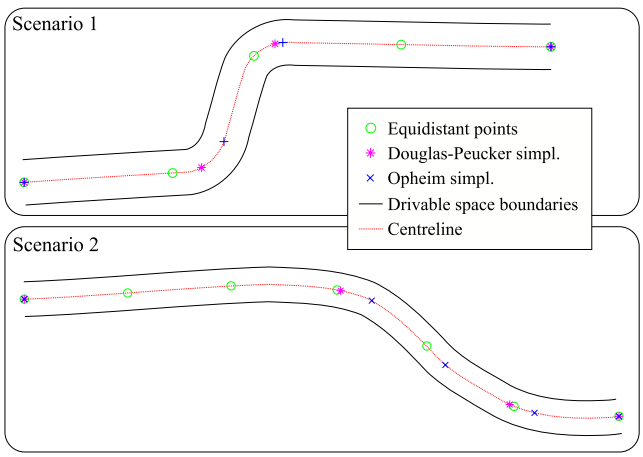

FIGURE 2. Reference points selection: Equidistant points, Douglas-Peucker and Opheim algorithms.

Fig. 2 shows the results of the application of the three methods over two different scenarios. The number of points selected by each algorithms is shown in table 2.

TABLE 2. Resultant number of selected points in both scenarios.

Method	Scenario 1	Scenario 2
Equidistant points	5	7
Douglas-Peucker	4	4
Opheim	4	5

2) WAYPOINTS OPTIMIZATION STRATEGIES

Both *reference points* and *seeding points* are subjected to optimization processes. Taking into account the wide range of refinement possibilities offered by the primitives considered, the approaches explained below have been addressed. For ease of reference, each method has been named with an abbreviated term.

a: A SEARCH (A*)*

This approach is only considered for *reference points* optimization. It uses the selected *reference points* to place a set of possible waypoints over their perpendiculars to the centreline inside the drivable space boundaries. The set of possible waypoints is connected through a directed graph from the initial vehicle pose to the final intended pose, as shown in Fig. 3.

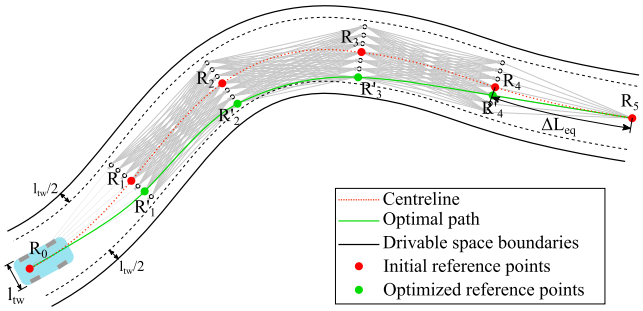

FIGURE 3. Reference points optimization based on directed graph search.

This graph is used to apply the A* algorithm in order to find the waypoints that minimize a cost function $f(n)$

$$\begin{aligned} f(m) &= g(m) + h(m) \\ g(m) &= w_{koff}\ |k_e| + (1 - w_{koff})\ d_{offset} \\ h(m) &= w_d\ d_{end} \end{aligned} \tag{21}$$

where k_e is the estimation of the curvature at the current point (m) being evaluated by using the two predecessor nodes ($m-1$ and $m-2$) to determine a circumference radius r_k ($k_e = 1/r_k$), d_{offset} is the distance from the point being evaluated (m) to the centreline, d_{end} is the distance from the point being evaluated (m) to the goal point, and weighting values w_{koff} and w_d are set to 0.5 and 0.001, respectively.

This approach is similar to [4] regarding reference trajectory planning, but in the present work curvature estimation and central offsets are considered in the cost function.

b: LATERAL DISPLACEMENTS (LA)

The method uses lateral displacements of waypoints in order to optimize the given cost function with a specific optimization algorithm. This approach considers one continuous variable per waypoint (lateral displacements). To calculate their optimal positions R'_i, the normal vector at each waypoint $R_i, i \in [1, N]$, $N+1$, being N the number of waypoints, is used (see Fig. 4):

$$R'_i = R_i + d_{lat_i} \vec{u}_{n_i}$$

where $\vec{u}_{n_i}$ and d_{lat_i} are respectively the normal unit vectors and the lateral distance to the optimum point, computed at R_i, as shown in Fig. 4.

Using the notation introduced in (1), the optimization variables and bounds can be written for this specific case as:

$$\begin{aligned} x_a^{LA} &= d_{lat} \\ l_b^{LA} &= -(\frac{l_w}{2} - \frac{l_{tw}}{2}) \\ u_b^{LA} &= (\frac{l_w}{2} - \frac{l_{tw}}{2}) \end{aligned}$$

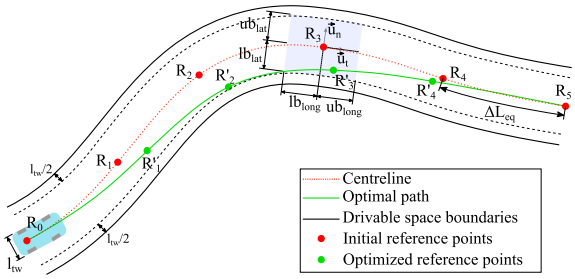

FIGURE 4. Reference points optimization based on longitudinal and lateral movements.

where $d_{lat}, l_b, u_b \in \mathbb{R}^{N-1}$, being d_{lat} the set of real values containing lateral displacements of points $R_i, i \in [1, N-1]$, l_w the lane width and l_{tw} the vehicle track width.

c: LONGITUDINAL DISPLACEMENTS (LO)

This method is similar to the previous one. The only difference is the use of longitudinal displacements instead of lateral ones:

$$R'_i = R_i + d_{long_i}\vec{u}_{t_i}$$

where $\vec{u}_{t_i}$ are the tangential vectors at points R_i (Fig. 4).

In this case, the optimization variables and bounds are:

$$\begin{aligned} x_a^{LO} &= d_{long} \\ l_b^{LO} &= -(-\Delta L/3) \\ u_b^{LO} &= (\Delta L/3) \end{aligned}$$

where $d_{long}, l_b, u_b \in \mathbb{R}^{N-1}$, being d_{long} the set of real values containing longitudinal displacements of points $R_n, n \in [1, N-1]$, and ΔL the distance to the closest waypoint of the two adjacent waypoints.

d: LATERAL AND LONGITUDINAL DISPLACEMENTS (LL)

This case is a combination of the two above, where lateral and longitudinal variations are considered:

$$R'_i = R_i + d_{long_i}\vec{u}_{t_i} + d_{lat_i}\vec{u}_{n_i}$$

In this case, two optimization variables are needed for each waypoint. As s result, the vectors of bounds and variables to be optimized are composed by the concatenation of the ones from both combined methods.

$$\begin{aligned} x_a^{LL} &= (x_a^{LA}, x_a^{LO}) \\ l_b^{LL} &= (l_b^{LA}, l_b^{LO}) \\ u_b^{LL} &= (u_b^{LA}, u_b^{LO}) \end{aligned}$$

e: LATERAL DISPLACEMENTS WITH SELECTION OPTION (LAS)

In this case, besides lateral displacements of each waypoint as in *LA* method, the problem includes additional binary variables to decide if a waypoint is used or not:

$$\begin{aligned} x_a^{LAS} &= (x_a^{LA}, b_{N-1}) \\ l_b^{LAS} &= (l_b^{LA}, 0_{N-1}) \\ u_b^{LAS} &= (u_b^{LA}, 1_{N-1}) \end{aligned}$$

where $b \in 0, 1$ is a binary vector of size $N-1$ that indicates whether a point R_i is used as waypoint to calculate the path or not, 0_{N-1} and 1_{N-1} are all-zeros and all-ones vectors of size $N-1$, respectively.

f: LONGITUDINAL DISPLACEMENTS WITH SELECTION OPTION (LOS)

This method is similar to the previous one but using longitudinal displacements instead of lateral ones:

$$\begin{aligned} x_a^{LOS} &= (x_a^{LO}, b_{N-1}) \\ l_b^{LOS} &= (l_b^{LO}, 0_{N-1}) \\ u_b^{LOS} &= (u_b^{LO}, 1_{N-1}) \end{aligned}$$

g: LATERAL AND LONGITUDINAL DISPLACEMENTS WITH SELECTION OPTION (LLS)

This approach was presented in [45]. It is a combination of the two above, where binary variables are introduced to decide if a waypoint is used or not, in addition to lateral and longitudinal displacements. Three variables per waypoint are used by the optimization algorithm in this case.

$$\begin{aligned} x_a^{LLS} &= (x_a^{LA}, x_a^{LO}, b_{N-1}) \\ l_b^{LLS} &= (l_b^{LA}, l_b^{LO}, 0_{N-1}) \\ u_b^{LLS} &= (u_b^{LA}, u_b^{LO}, 1_{N-1}) \end{aligned}$$

h: TANGENT VECTOR MAGNITUDE (TM)

As explained in subsection III-A, the magnitude of tangent vector has a high impact on the curve geometry. Using this method, the magnitude of tangent vector at each waypoint is optimized within a constrained range.

$$\vec{t}'_i = f_{t_i}\vec{u}_{t_i}$$

where $\vec{u}_{t_i}$ is the tangent vector at point R_i, and f_{t_i} is the magnitude of the new tangent vector.

In this case, the path planning problem variables and bounds are:

$$\begin{aligned} x_a^{TM} &= f_t \\ l_b^{TM} &= f_{t_{min}} \cdot 1_{N-1} \\ u_b^{TM} &= f_{t_{max}} \cdot 1_{N-1} \end{aligned}$$

where $f_t \in \mathbb{R}^{N-1}$, f_t is the set of real values containing the magnitudes of tangent vectors at intermediate waypoints, $f_{t_{min}}$ and $f_{t_{max}}$ are the minimum and maximum values for the scaling factor.

i: TANGENT VECTOR ORIENTATION (TD)

The orientation of the tangent vector highly affects the final curve too. Its value at each waypoint is optimized within a constrained range centered in the initial tangent orientation:

$$\vec{t}'_i = 1\angle\theta_i\,\vec{t}_i$$

where θ_{t_i} is the tangent vector orientation with respect to initial tangent orientation at point R_i.

In this case, the path planning problem variables and bounds are:

$$\begin{aligned} x_a^{TD} &= \theta_t \\ l_b^{TD} &= (-\Delta\theta_t \cdot 1_{N-1}) \\ u_b^{TD} &= (\Delta\theta_t \cdot 1_{N-1}) \end{aligned}$$

where $\theta_t \in \mathbb{R}^{N-1}$, θ_t is the set of real values containing the orientation of tangent vectors at intermediate waypoints and $\Delta\theta_t$ is the maximum allowed variation of tangent orientation.

j: TANGENT VECTOR MAGNITUDE AND ORIENTATION (TT)

This method is a combination of the two above, where both magnitude and orientation of the tangent vector are optimized. Just like in *LL* case, the vectors of bounds and variables to be optimized are composed by the concatenation of the ones from both combined methods.

$$\begin{aligned} x_a^{TT} &= (x_a^{TM}, x_a^{TD}) \\ l_b^{TT} &= (l_b^{TM}, l_b^{TD}) \\ u_b^{TT} &= (u_b^{TM}, u_b^{TD}) \end{aligned} \tag{22}$$

k: CURVATURE AT JOINTS (KJ)

In this method, the curvature at intermediate waypoints is optimized. As explained in Section III-A, curvature can be imposed at the joints of Bézier sections when quintic curves a re used. The approach is similar to the adopted in the *TM* method, i.e. a proportional factor (in this case f_{κ_i}) is used to modify the curvature at each intermediate waypoint as expressed in (22). The path planning problem variables and bounds are:

$$\begin{aligned} x_a^{KJ} &= f_\kappa \\ l_b^{KJ} &= f_{\kappa_{min}} \cdot 1_{N-1} \\ u_b^{KJ} &= f_{\kappa_{max}} \cdot 1_{N-1} \end{aligned}$$

where $f_\kappa, l_b, u_b \in \mathbb{R}^{N-1}$ is the set of real values containing the magnitude of tangent vectors at intermediate waypoints, $f_{\kappa_{min}}$ and $f_{\kappa_{max}}$ are the minimum and maximum values for the scaling factor. $f_{\kappa_{max}}$ is determined so as to ensure the maximum feasible curvature of the vehicle is not exceeded ($\kappa_i \cdot f_{\kappa_{max_i}} \leq \kappa_{max}^v$).

l: TANGENT VECTOR MAGNITUDE AND CURVATURE AT JOINTS (MK)

A combination of *TM* and *KJ* method is also considered. In this case, the optimization problem variables and bounds are:

$$\begin{aligned} x_a^{MK} &= (x_a^{TM}, x_a^{KJ}) \\ l_b^{MK} &= (l_b^{TM}, l_b^{KJ}) \\ u_b^{MK} &= (u_b^{TM}, u_b^{KJ}) \end{aligned}$$

m: TANGENT VECTOR ORIENTATION AND CURVATURE AT JOINTS (DK)

The last considered approach combines *TD* and *KJ* methods. the optimization problem variables and bounds are:

$$\begin{aligned} x_a^{DK} &= (x_a^{TD}, x_a^{KJ}) \\ l_b^{DK} &= (l_b^{TD}, l_b^{KJ}) \\ u_b^{DK} &= (u_b^{TD}, u_b^{KJ}) \end{aligned}$$

Note that methods *TM, TD, TT, MK* and *DK* can only be applied when using quintic Bézier splines, as cubic B-splines do not allows to impose tangent at intermediate waypoints.

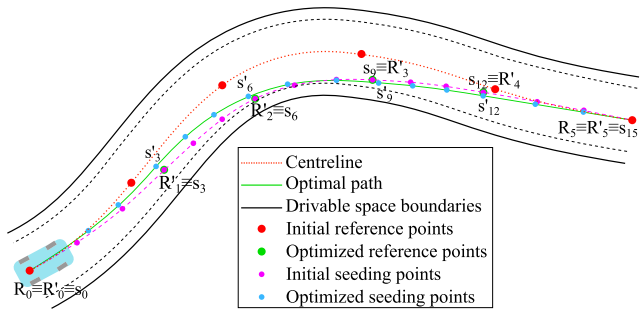

FIGURE 5. Seeding points optimization.

As exposed at the beginning of this section, cases with two optimization stages are also considered in this study. In these cases the first stage is used to optimize the position of *reference points* from which a set of new intermediate points (*seeding points*) will be obtained. To that end, the corresponding primitive is discretized with a fixed amount of points, as depicted in Fig. 5.

In cases with two optimization stages only methods *A*, LA, LO, LL, LAS, LOS, LLS* are considered for *reference points* optimization, while all of them are considered for the second stage and for the cases with only one optimization process.

3) OPTIMIZATION ALGORITHMS

Four algorithms to solve constrained non-linear multivariable optimization problems are compared in this work: Interior-point [46], Levenberg-Marquardt [47], Simple Multi-Objective Cross-Entropy (SMOCE) [48] and Non-linear Optimization by Mesh Adaptive Direct Search (NOMAD) [49].

SMOCE is an evolutionary multi-objective optimization algorithm which presents remarkable performance in solving complex problems with many decision variables. In this work the algorithm is applied to solve single-objective problems.

NOMAD is able to solve mixed-integer non-linear programming problems (MINLP), as the ones defined in *LAS, LOS* and *LLS* methods. This algorithm is highly configurable and is designed for constrained optimization of non-linear functions.

As in path planning methods, the algorithms are referenced using shorted names: interior-point (*IP*), Levenberg-Marquardt (*LM*), NOMAD (*NM*), SMOCE (*CE*).

4) COST FUNCTIONS

Based on the related work and the tests carried out in this work, five cost functions are proposed based on curvature that are typically used to path planning optimization processes:

1) $J_1 = \int_{s_0}^{s_f} \dot{\kappa}(s)^2 \, ds + h$
2) $J_2 = \int_{s_0}^{s_f} \ddot{\kappa}(s)^2 \, ds + h$
3) $J_3 = \int_{s_0}^{s_f} \dot{\kappa}(s)^2 + w_{J3} \cdot \ddot{\kappa}(s)^2 \, ds + h$
4) $J_4 = \int_{s_0}^{s_f} d_{off} \, ds + h$
5) $J_5 = \int_{s_0}^{s_f} d_{off} + w_{J5} \cdot \dot{\kappa}(s)^2 \, ds + h$

where κ is the scalar curvature of the path, $s \in [s_0, s_f] \in \mathbb{R}$ is the curve length over the initial (s_0) and final (s_f) values of the path, d_{off} is the perpendicular distance from the path to the centreline of the driving corridor, w_{J3} is the weight of the component related to the second derivative of the curvature in J_3, w_{J5} is the weight of the component related to the first derivative of the curvature in J_3, and h is a non-smooth function describing the relation between the path and the drivable space.

$$h = \begin{cases} 0 & \text{if path is within boundaries} \\ \infty & \text{if boundaries/obstacle collision} \\ \infty & \text{if } \kappa^p_{max} \geq \kappa^v_{max} \end{cases}$$

The value of κ is computed using the generic curvature equation of a given planar curve $c(t) = [x(t), y(t)]$:

$$\kappa = \frac{\dot{x}(t)\ddot{y}(t) - \dot{y}(t)\ddot{x}(t)}{\left(\dot{x}(t)^2 + \dot{y}(t)^2\right)^{\frac{3}{2}}} \tag{23}$$

where x and y are the parametric equations of the Bézier curves for both dimensions of curve c.

Note that the value of the objective function is infinity if: (i) the any part of the final path is outside the boundaries, (ii) the final path collides with obstacles, or (iii) the maximum curvature along the path κ^p_{max} is greater than the maximum curvature the vehicle can handle κ^v_{max}.

IV. COMPARISON FRAMEWORK DESCRIPTION

Due to the large number of possibilities when composing a followable path, one of the main issues to compare different approaches to solve the same path-planning problem is to extract objective measurements that characterize them. Furthermore, the performance can be evaluated both in terms of the tracking quality of the resulting path, which can be hard to objectively define, or in run-time terms.

In order to compare all path planning methods, a common framework is specified. On the one hand, some KPIs are defined with the aim of reflecting the quality of the approach employed, regardless of the type of primitive used, and not only in terms of functional performance but also of the execution time of the approaches.

A. DEFINITION OF KPIs

The definition of representative KPIs for benchmarking different approaches is not a trivial task. There is no a clear objective way of assessing a path planner. In fact, it is common to find a useful KPI for a small set of cases in specific scenarios that is useless or gives a wrong indication in other cases.

Besides computational cost, safety and comfort will be considered with different metrics related to the curvature and its variation. In addition to that, tunability and stability of the resulting path will be also taken into account through the offset to the centreline. Based on the tests carried out when developing this work, the most suitable KPIs contemplating all these aspects are the following:

1) Execution time: $K_t = t_{exc}$
2) Maximum curvature: $K_{\kappa_{max}} = \kappa_{max}$
3) Normalized accumulated curvature along the path:

$$K_{\kappa 0} = \frac{1}{L_p} \int_{s_0}^{s_f} \kappa(s)^2 \, ds \tag{24}$$

4) Normalized accumulated first derivative of the curvature along the path:

$$K_{\kappa 1} = \frac{1}{L_p} \int_{s_0}^{s_f} \dot{\kappa}(s)^2 \, ds \tag{25}$$

5) Normalized accumulated second derivative of the curvature along the path:

$$K_{\kappa 2} = \frac{1}{L_p} \int_{s_0}^{s_f} \ddot{\kappa}(s)^2 \, ds \tag{26}$$

6) Centreline offset along the path:

$$K_{cl} = \frac{1}{L_p} \int_{s_0}^{s_f} d_{off} \, ds \tag{27}$$

where $L_p = \sum_{i=1}^{N} \|p_i - p_{i-1}\|$ is the length of the path, p_i ($i \in \mathbb{N} : i \in [1, N]$) is the point i of the path, κ is the curvature of the path, κ_{max} is the maximum value of κ along the path, s ($s \in \mathbb{R} : s \in [s_0, s_f]$) is the curve length over the path, and t_{exc} is the total execution time.

B. PATH PLANNING PROBLEM SPECIFICATION

Each path planning strategy is defined as a set of parameters related to its main characteristics. As previously described, each case is characterized by: (i) the *reference points* selection method (over a given centreline), (ii) the type of primitive curve, (iii) the method of the first optimization process, (iv) method of second optimization process, and (v) the setting of initial and/or final heading and/or curvature as described in SectionIII.

For the sake of clarity, each test carried out in this work is identified with a unique text string composed of a set of sub-IDs separated by colons. The sub-IDs are taken from the abbreviated terms specified in Section III-B for each path planning step. The complete test ID is composed as follows:

$$\text{ID} = \text{S:RS:P:O1:O2:H:K}$$

where

- *S* is the scenario number.
- *RS* is *reference points* selection method: *E*, *D* or *O*.
- *P* is the primitive type: *3* if cubic B-spline or *5* if quintic Bézier splines is used.
- *Reference points* optimization process (*O1*): It is composed of the optimization method, the optimization algorithm and the cost function. If it is set to 0, no *reference points* optimization process is carried out. *O1* is therefore defined by the concatenation of:
  - Optimization method: *A*, LA, LO, LL, LAS, LOS, LLS*.
  - Optimization algorithm: *IP, LM, NM, CE*.
  - Optimization cost function: *J1, J2, J3, J4, J5*.
- *Seeding points* optimization process (*O2*): It is composed of the optimization method, the optimization algorithm and the cost function. If it is set to 0, no *seeding points* optimization process is carried out. *O2* is therefore defined by the concatenation of:
  - Optimization method: *LA, LO, LL, LAS, LOS, LLS, TM, TD, TT, KJ, MK, DK*.
  - Optimization algorithm: *IP, LM, NM, CE*.
  - Optimization cost function: *J1, J2, J3, J4, J5*.
- *H* indicates if the initial and final heading is imposed. It is defined using two binary digits: the first one refers to the initial heading and the second one to the final heading. The considered possibilities are: *00, 10, 11*, as specified in table 1 ($h_0, h_f = \{0, 1\}$).
- *K* indicates if the initial and final curvature is imposed in the same way that *H*. The considered possibilities are: *00, 10, 11*, as specified in table 1 ($\kappa_0, \kappa_f = \{0, 1\}$).

One sample of ID could be: 1:D:3:0:LA-IP-J1:11:00. This case addresses the scenario 1, the *reference points* are selected with the Douglas-Peucker algorithm, cubic B-splines are used as primitive curve, there is no *reference points* optimization (*O1* = 0, so the *seeding points* = *reference points*), the *seeding points* are optimized using *LA* method (lateral position optimization) with the interior-point algorithm (*IP*) and minimizing *J1* cost function. The initial and final orientation are set and the initial and final curvatures are not set.

V. EXPERIMENTS AND RESULTS

A. TESTS CASES SETUP

In order to compare the performance of all the strategies presented in Section III-B, all feasible combinations among the considered methods, primitives, optimization algorithms, etc. are tested in two different scenarios. As some combinations are not possible, the tests cases that include the configurations listed below are excluded:

- Impose initial curvature without imposing initial tangent.
- Impose final curvature without imposing final tangent.
- Impose final tangent without imposing initial tangent.
- Use cubic B-splines and
  - Impose final curvature.
  - Impose final tangent when initial tangent and curvature are already imposed.
  - The second optimization method is one of these: *TM, TD, TT, KJ, MK, DK* (They only apply to quintic Bézier spline cases).
- Use quintic Bézier splines and initial and/or final tangent and/or curvature are not imposed.
- There is only one optimization stage and it is defined as the first one instead of second one (*seeding points*).
- Tests cases in which there are two optimization stages and *CE* optimization algorithm is used in the second one. These cases are excluded because *CE* does not allow to set as the initial point the output of the first optimization stage.

Once the above test cases were excluded, 90417 tests per scenario were executed (180834 for both scenarios). All path planning approaches were implemented in Matlab and the experiments were executed on an Intel Core i7-3770 3.4GHz machine with 8GB RAM.

In order to test the path planning methods in realistic scenarios, both driving environments were extracted from real roads, which are shown in Fig. 2. Scenario *1* contains two tight curves and a centreline length of 40.02m, while *Scenario 2* comprises a roundabout entrance, with a centreline length of 54.79m. The lane with in both scenarios is 3m and a vehicle track width of 1.71m (l_{tw}) is considered.

The motivation to choose scenarios with these values of centreline length is mainly due to the trade-off between computational cost and anticipation capabilities, as larger paths may involve more intermediate waypoints and therefore longer computation time. Based on the tests carried out in this study, paths with a length around 50m are computed in a reasonable amount of time as shown in Section V-B.

To determine the acceptable values for the large amount of parameters of different methods, algorithms, etc. employed in this work, specific tests were carried out. Regarding reference points selection, the length to obtain equidistant point was set to 7.5m. In Douglas-Peucker algorithm, ξ was set to 1m. Opheim algorithm was parametrized with minimum and maximum tolerance of 1.8m and 30m, respectively. The maximum function evaluation of optimization algorithms were limited in order to avoid large execution times in cases were the algorithms cannot find a solution. In the case of NOMAD algorithm the maximum optimization time was set to 20s. SMOCE algorithm parameters was set to 50 epochs, 100 solutions and 0.1 as elitist fraction.

Finally, the weights w_{J3} in cost function *J3* and w_{J5} in cost function *J5* were set to 60 and 100, respectively.

B. RESULTS AND DISCUSSION

Fig. 6 represents the tests cases distribution of both scenarios using the percentiles against the value of each KPI plotted in logarithmic scales. The distribution of the tests regarding K_t and K_{cl} is similar in both scenarios. However, the values of $K_{\kappa 0}$, $K_{\kappa 1}$, $K_{\kappa 2}$ and $K_{\kappa max}$ in *Scenario 2* are generally

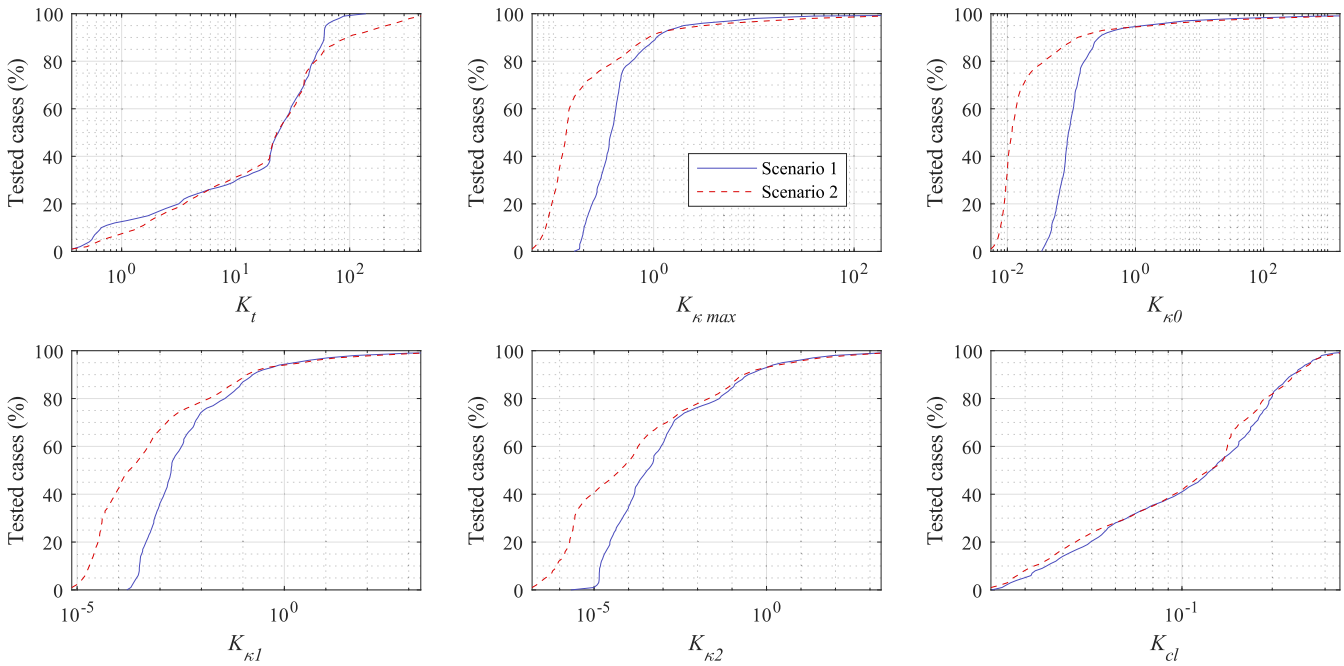

FIGURE 6. Percentiles of all KPIs against their values for both scenarios (*1* and *2*).

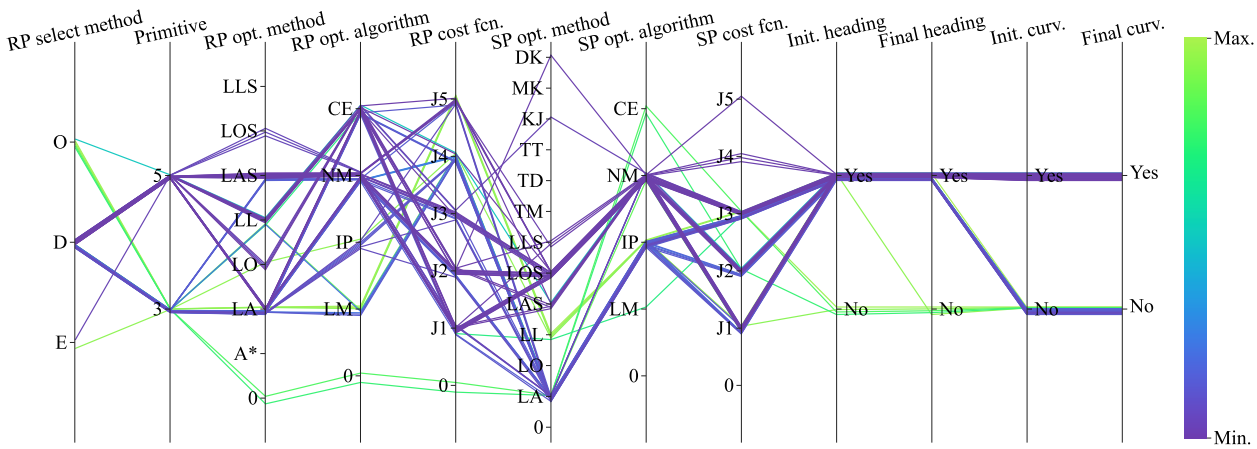

FIGURE 7. Results of filtered tests cases with thresholds on K_t, $K_{\kappa max}$ and path length, and percentile 50th on K_{cl}, $K_{\kappa 0}$, $K_{\kappa 1}$ and $K_{\kappa 2}$ in Scenario *1*. Color is based on KPI K_t, ranging from 0s to 50s.

below those in *Scenario 1*, probably due to its tighter curves. On another issue, the noticeable change in the distribution at $K_t = 20s$ is caused by the maximum imposed optimization time when using NOMAD algorithm.

In order to extract relevant results from all tested cases, a set of test cases of each scenario is filtered separately, based on the values of their KPIs. The thresholds used to select the minimum acceptable results with respect to the KPIs are listed below:

- $K_t \leq 50s$.
- $K_{\kappa max} \leq 0.4m^{-1}$
- $K_{\kappa 0},\ K_{\kappa 1}, K_{\kappa 2} \leq 3$.
- The path length is also constrained such that it does not differs more than a 5% with respect to the centreline length of the scenario (L_{cl}): $|L_p - L_{cl}| \leq L_{cl} \cdot 0.05$

The wide range of combinations of tests configurations results in high dimensional data, and therefore specific graphic representation tools are needed. By using parallel coordinates plots, Fig. 7 and 8 allow to represent the resulting KPI values in terms of its configuration values, as specified in the test ID (see section IV-B), for *Scenario 1* and *Scenario 2*, respectively. The color of each line represent the value of a particular KPI (in these two particular cases K_t). In addition to the filtering above described, the 50th percentiles of $K_{cl}, K_{\kappa 0}, K_{\kappa 1}$ and $K_{\kappa 2}$ were used to select the cases plotted in Figs. 7 and 8.

As can be noticed, after applying the filters there are significantly more valid tests cases in *Scenario 2* than in *Scenario 1* (1117 and 77, respectively). Furthermore, it is also remarkable that almost all cases in *Scenario 1* needed two

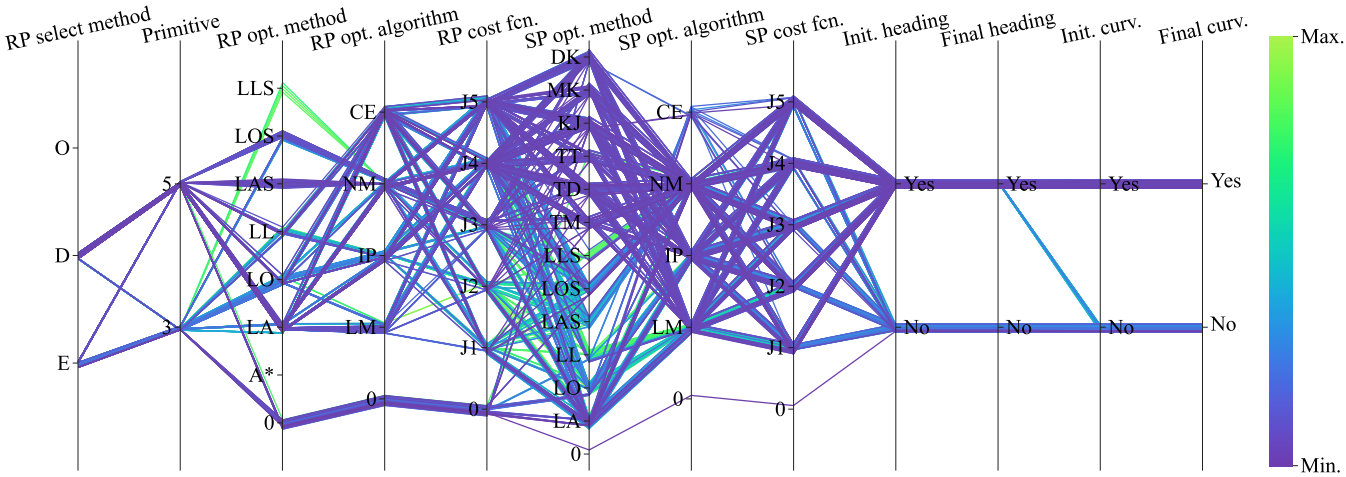

FIGURE 8. Results of filtered test cases in Scenario 2. Color is based on KPI K_t, ranging from 0s to 50s.

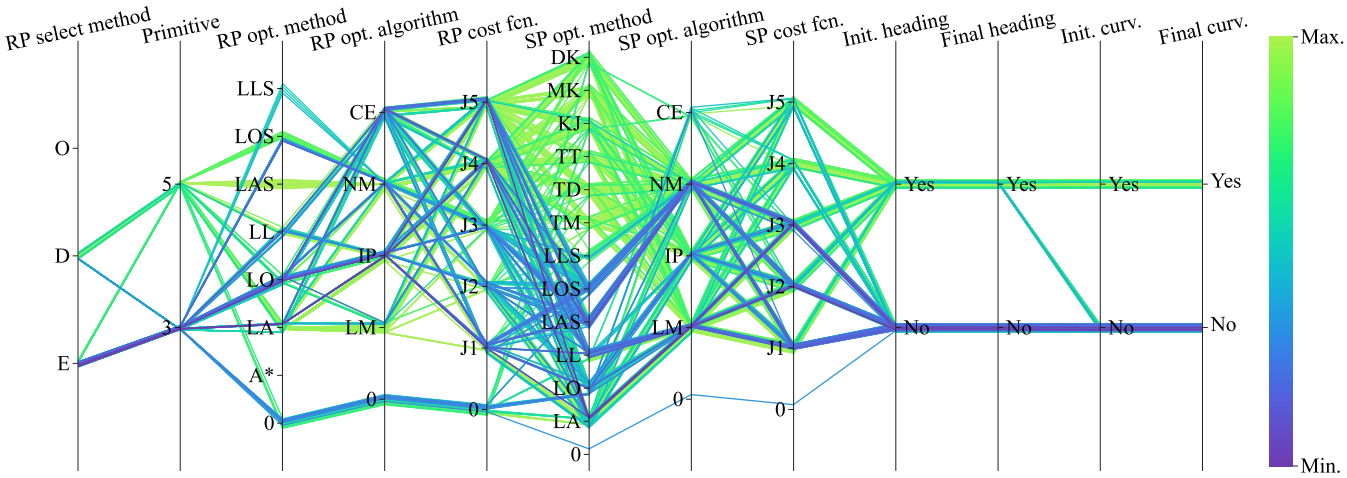

FIGURE 9. Results of filtered test cases in Scenario 2. Line colors are based on KPI $K_{\kappa 0}$.

optimization stages. Just a few of them used only one stage but with a bad performance regarding the execution time. In contrast, although most of the cases used two optimization stages, the ones with only one optimization stage have a good timing performance in *Scenario 2*. These are signs of the greater complexity of the path planning problem in *Scenario 1* compared to *Scenario 2*.

Comparing both scenarios regarding the execution time, it is generally lower when using this Douglas-Peucker algorithm as *reference points* selection method (as shown in table 2). This is non surprising, as a lower number of optimization variables is used in those cases.

Since the richness of information in *Scenario 2* is higher, i.e. there is a greater number of valid tests cases compared to *Scenario 1*, the subsequent discussion is focused on this particular information. The color of the lines in Figs. 9, 10, 11 and 12 show the values of KPIs $K_{\kappa 0}$, $K_{\kappa 1}$, $K_{\kappa 2}$ and K_{cl}, respectively. It can be appreciated that the cases using *LA, LO, LL, LAS, LOS, LLS* methods as *seeding points* optimization present a better performance regarding $K_{\kappa 0}$, $K_{\kappa 1}$, $K_{\kappa 2}$ and K_{cl}. Contrastingly, a worse performance regarding K_t is observed when these methods are used (Fig. 8), where methods *TM, TD, TT, KJ, MK, DK* (only applied when using quintic Bézier splines) present better results.

The resulting paths of two selected tests cases are shown in Figs.13 and 14. The high smoothness of optimized paths is observed in both cases. To demonstrate that, as curve smoothness is hard to see on a 2D plane curve directly, the resultant path curvature is shown together with its first derivative at the bottom of both figures. As can be noticed, the curvature remains continuous along the path in both figures. These are two random test cases, but an insight into the best configurations is shown below.

1) COMPARING BEST CASES IN BOTH SCENARIOS

Delving deeper into the results, several cases with the minimum value in all KPIs are selected. The resulting KPIs are shown in table 3. Moreover, the normalized KPIs of both scenarios are represented in radar charts of Fig. 15.

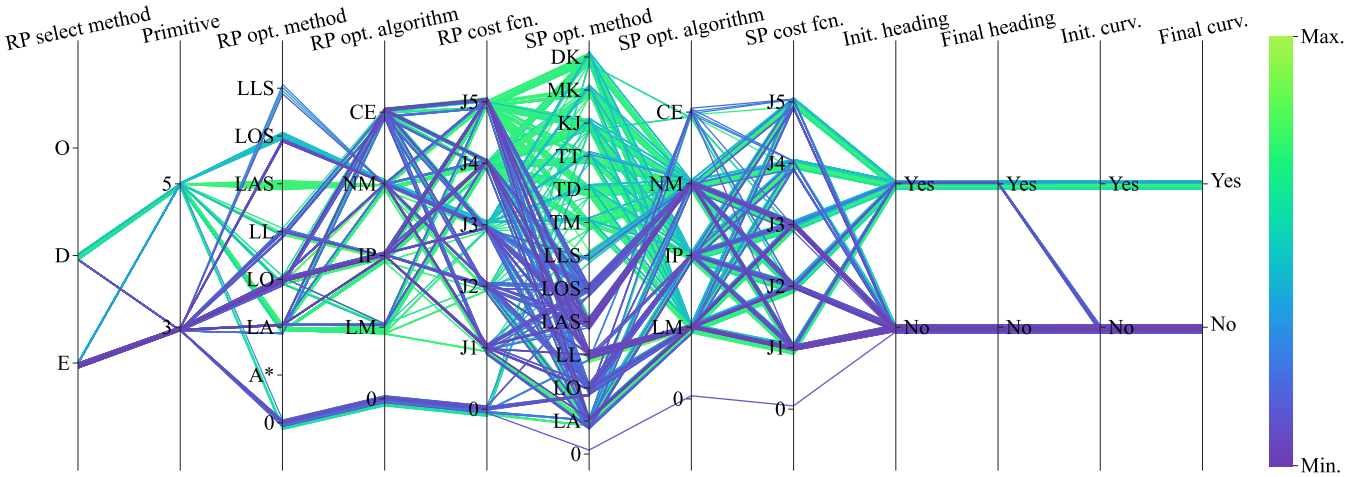

FIGURE 10. Results of filtered test cases in Scenario 2. Line colors are based on KPI $K_{\kappa 1}$.

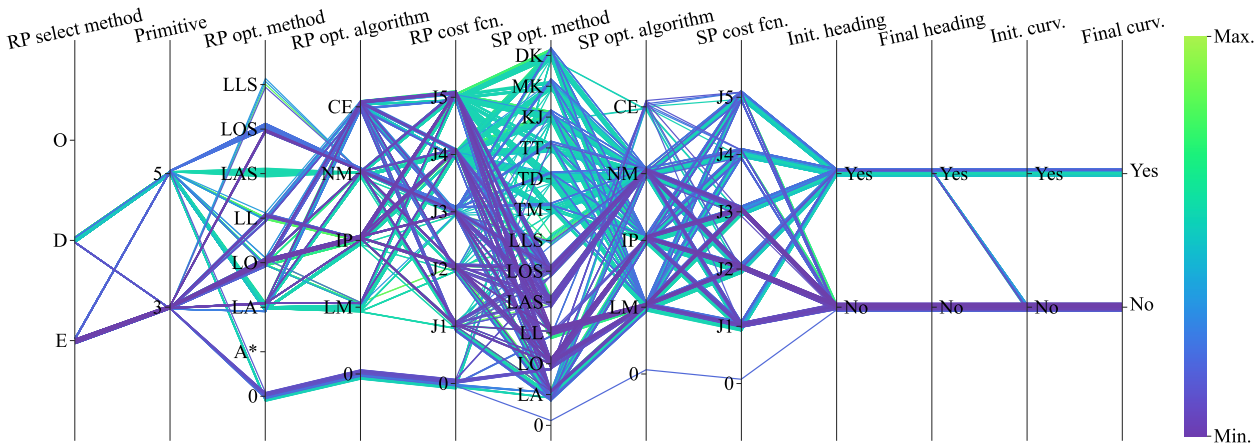

FIGURE 11. Results of filtered test cases in Scenario 2. Line colors are based on KPI $K_{\kappa 2}$.

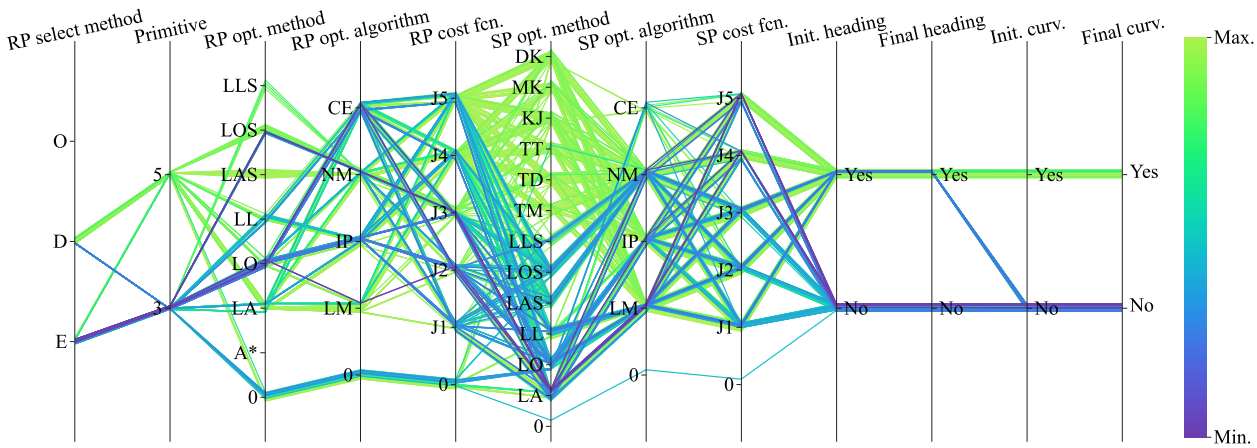

FIGURE 12. Results of filtered test cases in Scenario 2. Line colors are based on KPI K_{cl}.

Note that the normalization was done to achieve scenario-independent results, as *Scenario 1* has tighter curves, and curvature-related KPIs values are therefore higher than in *Scenario 2*.

The first thing that can be noticed from the table 3 is that in both scenarios the *reference points* selection method and primitive used in all selected cases are Douglas-Peucker and quintic Bézier splines, respectively. Furthermore, it is also

TABLE 3. KPI values of selected tests from both scenarios.

Test case ID	K_t	$K_{\kappa max}$	$K_{\kappa 0}$	$K_{\kappa 1}$	$K_{\kappa 2}$	K_{cl}
1:D:5:LA-CE-J1:LOS-NM-J1:11:11	0.3478	0.2146	6.3975E-02	4.3119E-04	1.8174E-05	0.1774
1:D:5:LA-CE-J1:LOS-NM-J2:11:11	0.3489	0.2127	6.1818E-02	4.3765E-04	1.8818E-05	0.1560
1:D:5:LA-CE-J2:LOS-NM-J1:11:11	0.3491	0.2115	6.1428E-02	4.3518E-04	1.9052E-05	0.1629
1:D:5:LA-CE-J1:LOS-NM-J3:11:11	0.3510	0.2116	6.1511E-02	4.3510E-04	1.9103E-05	0.1653
1:D:5:LL-CE-J4:LOS-NM-J3:11:11	0.3519	0.2207	6.0106E-02	4.8143E-04	2.0956E-05	0.1769
1:D:5:LA-CE-J2:LOS-NM-J3:11:11	0.3551	0.2077	5.8191E-02	4.4287E-04	1.9494E-05	0.1524
2:D:5:LO-CE-J4:LOS-NM-J1:11:11	0.3442	0.1164	9.5620E-03	2.8543E-05	1.2807E-06	0.1393
2:D:5:LA-CE-J2:TD-NM-J3:11:11	0.3464	0.1251	8.9540E-03	3.1399E-05	1.7702E-06	0.1380
2:D:5:LA-CE-J2:KJ-NM-J1:11:11	0.3474	0.1302	9.6590E-03	3.3555E-05	1.8731E-06	0.1388
2:D:5:LO-CE-J2:KJ-NM-J3:11:11	0.3495	0.1007	9.5130E-03	2.6601E-05	8.4680E-07	0.1297
2:D:5:LA-CE-J3:TD-NM-J3:11:11	0.3495	0.1269	9.1370E-03	3.2077E-05	1.8669E-06	0.1409
2:D:5:LO-CE-J3:TD-NM-J5:11:11	0.3501	0.1255	9.4750E-03	3.2300E-05	1.7255E-06	0.1328

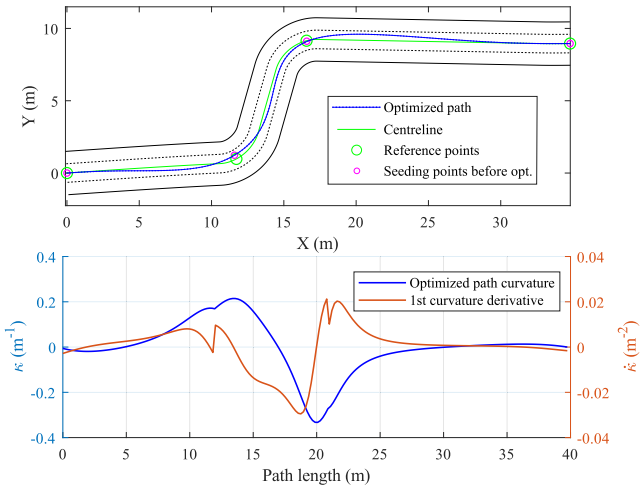

FIGURE 13. Results of test 1:D:5:LA-CE-J1:LOS-NM-J1:11:11. Scenario and final optimized path (top) and curvature of the final path and its first derivative (bottom).

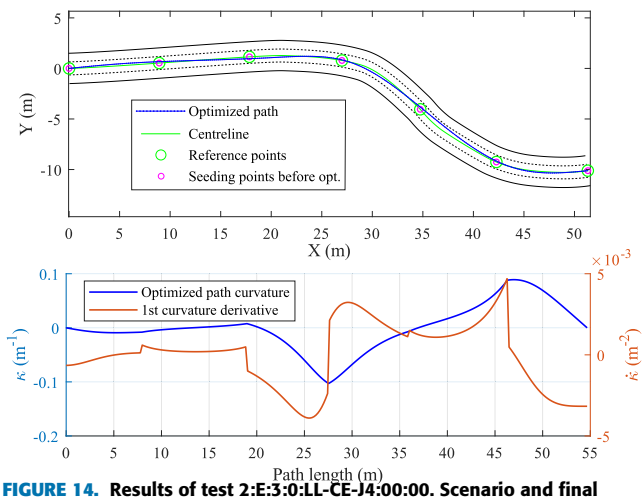

FIGURE 14. Results of test 2:E:3:0:LL-CE-J4:00:00. Scenario and final optimized path (top) and curvature of the final path and its first derivative (bottom).

remarkable that SMOCE and NOMAD algorithms are also used in all scenarios for the first and the second optimization stage, respectively.

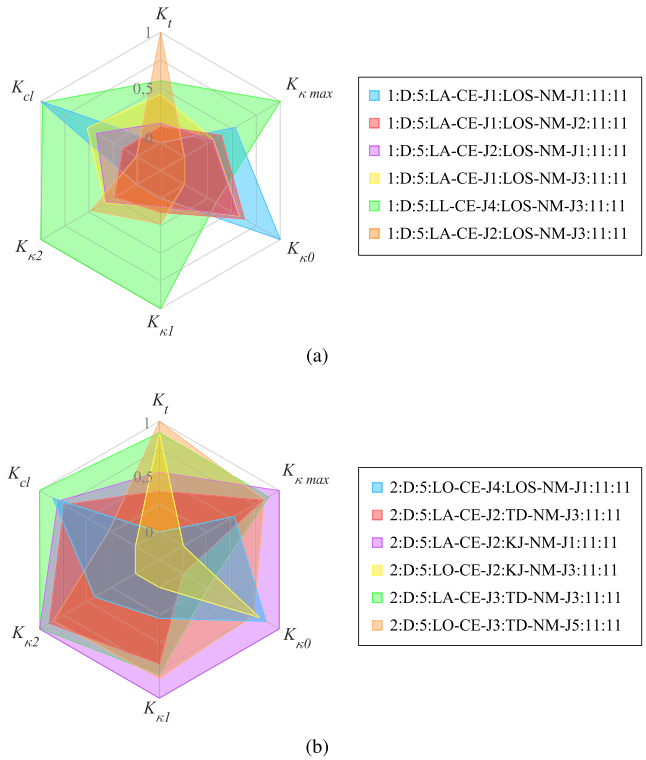

FIGURE 15. Results of selected test cases of each scenario. (a) Scenario *1*. (b) Scenario *2*.

Regarding (K_t) it can be seen that test cases with ID = 1:D:5:LA-CE-J2:LOS-NM-J3:11:11 and 2:D:5:LO-CE-J3:TD-NM-J5:11:11 are those which take longer to perform the optimization of the selected tests in each scenario. However, in these 2 specific configurations almost all values of the KPIs related to the quality of the path are much lower than in the other cases. Focusing on *Scenario 1*, it is also remarkable the influence of a greater execution time in the higher quality of the final path. Regarding *Scenario 2* it can be observed that the cases 2:D:5:LA-CE-J2:TD-NM-J3:11:11 and 2:D:5:LA-CE-J2:KJ-NM-J1:11:11 present a high value in almost all KPIs, i.e. a higher execution time did not lead to better results in the quality of the final path.

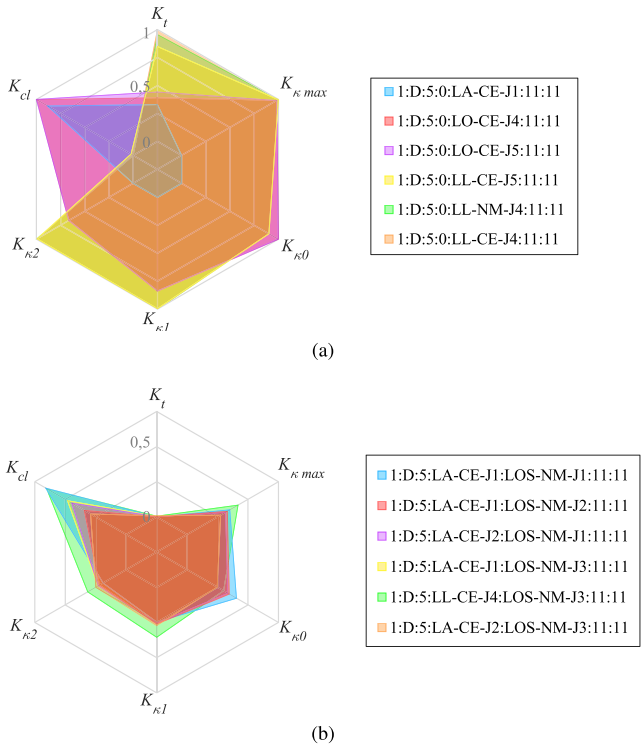

FIGURE 16. Results of selected test cases of *Scenario 1* with one (a) and two (b) optimization stages.

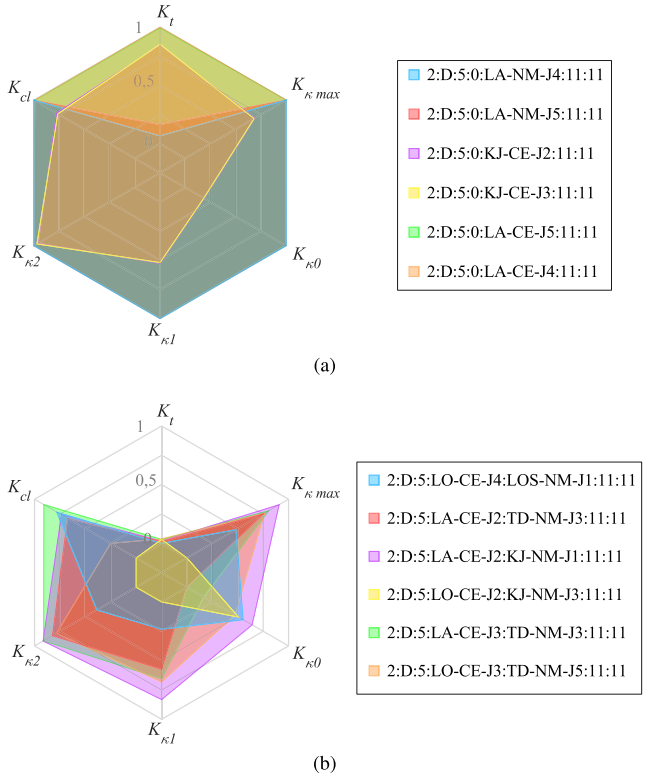

FIGURE 17. Results of selected test cases of *Scenario 2* with one (a) and two (b) optimization stages.

2) COMPARING BEST CASES WITH ONE AND TWO OPTIMIZATION STAGES IN BOTH SCENARIOS

In order to analyze the impact of each approach, the best results with one and two optimization stages are selected from both scenarios separately. Fig. 16 shows the cases with one and two optimization stages in *Scenario 1*, while Fig. 17 depict those in *Scenario 2*.

The values of the KPIs are normalized with respect to each scenario separately, as in the previous analysis.

In this case, the tests selected in Fig. 17b are the same than were selected in Fig. 15b as the best results present two optimization stages. However, the KPIs are normalized together with the selected tests with one optimization stage for this scenario. As a result, the represented values are different between both figures.

As can be noticed taking into account the results in both scenarios, the KPIs reflect a better overall performance when two optimization stages are carried out. It is also noteworthy that even the KPI K_t is lower in cases where two optimization stages were performed. Moreover, it is also remarkable that, again, the best cases with one optimization stage use quintic Bézier splines optimized through different approaches, instead of cubic B-splines. This can be caused by the lower tunability and stability of cubic B-splines compared to quintic Bézier splines.

Note that the all shown cases in these figures are the best selected after the filtering explained above. Therefore, despite the normalized value of some KPIs is close to 1, all test cases present acceptable absolute values.

Considering the results obtained, it can be concluded that some of the better approaches from those proposed in this work are those using quintic Bézier splines as primitive and two optimization stages. Furthermore, some of the better approaches for the first optimization stage are *LA, LO,* and *LL*; while for the second one are *LOS, KJ,* and *TD*.

Regarding optimization algorithms, SMOCE and NOMAD seem to deal better with the strong non-linearity of the optimization problem (since most of the cost function are based on the curvature equation) obtaining better overall results compared with interior-point and Levenberg-Marquardt algorithms.

With regard to the cost functions, *J1, J2, J3,* and *J4* are present in the first optimization stage of the best results, while all of them are present in the second stage. In fact, it can be observed that some of the best results use the same configuration for the first optimization stage and different configurations in the second. This observation highlights the higher impact of the cost function in the first optimization stage when compared with the second one.

The results of all tested cases are publicly available at https://autopia.car.upm-csic.es/antonio/comparison_results.html. In this url, parallel coordinates plots as those shown in Figs. 7-12 can be seen. The interface allows to select the color of the lines based on KPIs values as well as selecting the scenario and the percentile to filter data. Moreover, additional filters can be applied over the plot coordinates. In addition,

the resulting KPIs values of selected test cases are shown in a table at the bottom of the web page.

VI. CONCLUSIONS

An insight on a number of different approaches for path planning is carried out in this work, where a wide range of possible combinations among several primitives, optimization methods and algorithms are compared. The results are intended to help in future decisions about the most appropriate approach for local path planning in different environments or applications. To that end, the main contributions of this paper are (i) a comparison framework to benchmark different path-planning primitives for on-road urban driving, (ii) the evaluation of different primitive configurations and optimisation techniques for path-planning, and (iii) the open publication of the results and its consequent analysis, based on a set KPIs related to the aforementioned main features.

ACKNOWLEDGMENT

J. Godoy wants to especially thank the Juan de la Cierva fellowship program (Spanish Ministry of Economy, Industry and Competitiveness) for its support in the development of this work.

REFERENCES

[1] 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.

[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. Veh.*, vol. 1, no. 1, pp. 33–55, Mar. 2016.

[3] T. Gu, J. M. Dolan, and J.-W. Lee, "On-road trajectory planning for general autonomous driving with enhanced tunability," in *Intelligent Autonomous Systems 13*. Cham, Switzerland: Springer, 2016, pp. 247–261.

[4] T. Gu, J. Snider, J. M. Dolan, and J.-W. Lee, "Focused trajectory planning for autonomous on-road driving," in *Proc. IEEE Intell. Veh. Symp. (IV)*, Jun. 2013, pp. 547–552.

[5] J. Wei, J. M. Snider, T. Gu, J. M. Dolan, and B. Litkouhi, "A behavioral planning framework for autonomous driving," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2014, pp. 458–464.

[6] 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.

[7] J. Horst and A. Barbera, "Trajectory generation for an on-road autonomous vehicle," *Proc. SPIE*, vol. 6230, p. 62302J, May 2006.

[8] T.-C. Liang, J.-S. Liu, G.-T. Hung, and Y.-Z. Chang, "Practical and flexible path planning for car-like mobile robot using maximal-curvature cubic spiral," *Robot. Auto. Syst.*, vol. 52, no. 4, pp. 312–335, 2005.

[9] 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.

[10] A. Piazzi, C. G. L. Bianco, and M. Romano, "η^3 splines for the smooth path generation of wheeled mobile robots," *IEEE Trans. Robot.*, vol. 23, no. 5, pp. 1089–1095, Oct. 2007.

[11] J. Connors and G. Elkaim, "Manipulating b-spline based paths for obstacle avoidance in autonomous ground vehicles," in *Proc. ION Nat. Tech. Meeting*, vol. 5, 2007, pp. 1081–1088.

[12] J. Canny and J. Reif, "New lower bound techniques for robot motion planning problems," in *Proc. 28th Annu. Symp. Found. Comput. Sci.*, 1987, pp. 49–60.

[13] S. Lazard, J. Reif, and H. Wang, "The complexity of the two dimensional curvatureconstrained shortest-path problem," in *Proc. 3rd Int. Workshop Algorithmic Found. Robot.*, Houston, TX, USA, 1998, pp. 49–57.

[14] J. Ziegler *et al.*, "Making Bertha drive—An autonomous journey on a historic route," *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 2, pp. 8–20, Feb. 2014.

[15] C. L. Darby, W. W. Hager, and A. V. Rao, "An hp-adaptive pseudospectral method for solving optimal control problems," *Optimal Control Appl. Methods*, vol. 32, no. 4, pp. 476–502, 2011.

[16] J. T. Betts, "Survey of numerical methods for trajectory optimization," *J. Guid., Control, Dyn.*, vol. 21, no. 2, pp. 193–207, 1998.

[17] L. S. Pontryagin, *Mathematical Theory of Optimal Processes*. Boca Raton, FL, USA: CRC Press, 1987.

[18] B. Chazelle, "Approximation and decomposition of shapes," in *Advances in Robotics: Algorithmic and Geometric Aspects of Robotics*, vol. 1, J. T. Schwartz and C. K. Yap, Eds. Lawrence Erlbaum Associates, 1987, pp. 145–185.

[19] J.-C. Latombe, *Robot Motion Planning*, vol. 124. New York, NY, USA: Springer, 2012.

[20] 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.

[21] 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.

[22] 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, 2015.

[23] 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, 1990.

[24] S. Petti and T. Fraichard, "Safe motion planning in dynamic environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Aug. 2005, pp. 2210–2215.

[25] 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.

[26] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numer. Math.*, vol. 1, no. 1, pp. 269–271, Dec. 1959.

[27] 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. TSSC-4, no. 2, pp. 100–107, Jul. 1968.

[28] A. Stentz, "Optimal and efficient path planning for partially-known environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 1994, pp. 3310–3317.

[29] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Dept. Comput. Sci., Iowa State Univ., Ames, IA, USA, Tech. Rep. TR 98-11, 1998.

[30] 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.

[31] 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.

[32] K. Yang and S. Sukkarieh, "An analytical continuous-curvature path-smoothing algorithm," *IEEE Trans. Robot.*, vol. 26, no. 3, pp. 561–568, Jun. 2010.

[33] J.-W. Choi, R. Curry, and G. Elkaim, "Path planning based on bézier curve for autonomous ground vehicles," in *Proc. World Congr. Eng. Comput. Sci. (WCECS)*, 2008, pp. 158–166.

[34] D. Q. Tran and M. Diehl, "An application of sequential convex programming to time optimal trajectory planning for a car motion," in *Proc. 48th IEEE Conf. Decis. Control, Jointly 28th Chin. Control Conf. (CDC/CCC)*, Oct. 2009, pp. 4366–4371.

[35] C. Dimitrakakis, "Online statistical estimation for vehicle control," IDIAP, Martigny, Switzerland, Tech. Rep. EPFL-REPORT-83327, 2006.

[36] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for Bertha—A local, continuous method," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2014, pp. 450–457.

[37] J. Baltes, "A benchmark suite for mobile robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, vol. 2, Feb. 2000, pp. 1101–1106.

[38] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[39] M. Althoff, M. Koschi, and S. Manzinger, "CommonRoad: Composable benchmarks for motion planning on roads," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 719–726.

[40] M. Ganesh, *Basics of Computer Aided Geometric Design: An Algorithmic Approach*. Delhi, India: I.K. International, 2008.

[41] R. Levien and C. H. Séquin, "Interpolating splines: Which is the fairest of them all?" *Comput.-Aided Des. Appl.*, vol. 6, no. 1, pp. 91–102, Jan. 2009. [Online]. Available: http://www.tandfonline.com/doi/abs/10.3722/cadaps.2009.91-102

[42] B. Lau, C. Sprunk, and W. Burgard, "Kinodynamic motion planning for mobile robots using splines," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2009, pp. 2427–2433. [Online]. Available: http://ieeexplore.ieee.org/document/5354805/

[43] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Classics Cartogr., Reflections Influential Articles Cartographica*, vol. 10, no. 2, pp. 15–28, Dec. 2011. [Online]. Available: http://utpjournals.press/doi/10.3138/FM57-6770-U75U-7727

[44] H. Opheim, "Smoothing a digitized curve by data reduction methods," in *Proc. Int. Conf.*, 1981, p. 127. [Online]. Available: http://diglib.eg.org/EG/DL/Conf/EG81/papers/EUROGRAPHICS_81pp127-135.pdf.abstract.pdf;internal&action=paperbibtex.action

[45] A. Artuñedo, J. Godoy, and J. Villagra, "Smooth path planning for urban autonomous driving using OpenStreetMaps," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 837–842. [Online]. Available: http://ieeexplore.ieee.org/document/7995820/

[46] R. H. Byrd, J. C. Gilbert, and J. Nocedal, "A trust region method based on interior point techniques for nonlinear programming," *Math. Program.*, vol. 89, no. 1, pp. 149–185, 2000.

[47] J. J. Moré, "The Levenberg-Marquardt algorithm: Implementation and theory," in *Numerical Analysis*, G. A. Watson, Ed. Berlin, Germany: Springer, 1978, pp. 105–116.

[48] R. E. Haber, G. Beruvides, R. Quiza, and A. Hernandez, "A simple multi-objective optimization based on the cross-entropy method," *IEEE Access*, vol. 5, pp. 22272–22281, 2017.

[49] S. Digabel, "Algorithm 909: NOMAD: Nonlinear optimization with the MADS algorithm NOMAD: Nonlinear optimization with the MADS algorithm," *ACM Trans. Math. Softw.*, vol. 37, no. 4, pp. 15–44, 2011. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1916461.1916468

ANTONIO ARTUÑEDO received the B.Sc. degree in electrical engineering from the Universidad de Castilla-La Mancha, Spain, in 2011, and the M.Sc. degree in industrial engineering from the Universidad Carlos III de Madrid in 2014. He is currently pursuing the Ph.D. degree in automation and robotics with the Centre for Automation and Robotics (CSIC-UPM), Spain, where he is involved in the AUTOPIA Program. His research interests include system modeling and simulation, intelligent control, planning, and decision making in autonomous vehicles.

JORGE GODOY was born in Maracay, Venezuela, in 1986. He received the Degree in electronic engineering from 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. Since 2009, he has been with the Centre of Automation and Robotics, researching on the autonomous vehicles. His research interests include fuzzy-logic control and intelligent transportation systems.

JORGE VILLAGRA received the Degree 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-year CIFRE Program with PSA-Peugeot-Citroën and then with a post-doctoral fellowship with a joint research unit INRIA-Mines ParisTech, France. From 2007 to 2009, he was a Visiting Professor with University Carlos III, Spain. He then received a three-year JAEDoc Fellowship from the AUTOPIA Program, Center for Automation and Robotics UPM-CSIC, Spain, where he spent one additional year funded by a research contract. From 2013 to 2016, he led the Department of ADAS and Highly Automated Driving Systems, Ixion Industry & Aerospace SL, where he also coordinated all the activities in the EU research and development funding programs. He has been leading the AUTOPIA Program with CSIC since 2016. He developed his research activity in six different entities with a very intense activity in project setup and management, through over 30 international and national research and development projects, where he is or has been IP of 10 of these projects. He has authored over 85 papers in international journals and conferences on autonomous driving, intelligent transportation systems, dmodel-free control, and new probabilistic approaches for embedded components in autonomous vehicles. He received the Prize for the Best Dissertation in Automatic Control in France in 2006 for his Ph.D. results.