A local trajectory planning and control method for autonomous vehicles based on the RRT algorithm

Stefano Feraco
Department of Mechanical and Aerospace Engineering
Politecnico di Torino
Torino, Italy
stefano.feraco@polito.it

Sara Luciani
Department of Mechanical and Aerospace Engineering
Politecnico di Torino
Torino, Italy
sara.luciani@polito.it

Angelo Bonfitto
Department of Mechanical and Aerospace Engineering
Politecnico di Torino
Torino, Italy
angelo.bonfitto@polito.it

Nicola Amati
Department of Mechanical and Aerospace Engineering
Politecnico di Torino
Torino, Italy
nicola.amati@polito.it

Andrea Tonoli
Department of Mechanical and Aerospace Engineering
Politecnico di Torino
Torino, Italy
andrea.tonoli@polito.it

***Abstract*—This paper presents a local trajectory planning and control method based on the Rapidly-exploring Random Tree algorithm for autonomous racing vehicles. The paper aims to provide an algorithm allowing to compute the planned trajectory in an unknown environment, structured with non-crossable obstacles, such as traffic cones. The investigated method exploits a perception pipeline to sense the surrounding environment by means of a LIDAR-based sensor and a high-performance Graphic Processing Unit. The considered vehicle is a four-wheel drive electric racing prototype, which is modeled as a 3 Degree-of-Freedom bicycle model. A Stanley controller for both lateral and longitudinal vehicle dynamics is designed to perform the path tracking task. The performance of the proposed method is evaluated in simulation using real data recorded by on-board perception sensors. The algorithm can successfully compute a feasible trajectory in different driving scenarios.**

Keywords—Trajectory planning, Autonomous driving, Rapidly-exploring Random Tree, Vehicle control, Environment perception, Local planning.

I. Introduction

Self-driving vehicles are experiencing an increasing interest worldwide during the last years, which motivates a constant research effort to continuously address design challenges related to safety and performances of the next generation of automated cars [1]. Fully autonomous projects are currently in various stages of development and testing, which include on-road validation, and recent vehicles can already feature many on-board Advanced Driver Assistance Systems (ADAS), such as Lane Keeping Assist (LKA), Smart Cruise Control (SCC), Intelligent Speed Adaptation (ISA), Lane Departure Warning (LWA) [2][3], or intelligent devices devoted to virtual sensing [4][5]. Nevertheless, fully autonomous vehicles are still far from entering the market in the brief period, since significant concerns from the potential customers include not only privacy and cybersecurity, but also the elevated level of performance expectations which is only partially achieved in the experimental validation of the proposed techniques at the current stage [6][7]. Considering this framework, trajectory planning is one of the fundamental tasks that a fully autonomous vehicle must perform to be compliant with its requirements in terms of autonomy level [8]. In autonomous driving, trajectory planning is defined as the real-time computation of the planned vehicle's motion from an initial state to the next one, according to the maneuver's feasibility with respect to the vehicle dynamics limits. Considering the whole collection of trajectory planning techniques investigated in the recent literature, the optimal trajectory is typically computed at each time step after the evaluation of a set of feasible trajectories with respect to a cost function [3][9]. To this end, many trajectory planning methods have been investigated during the last decades for autonomous driving. The Probabilistic Road Map (PRM) algorithm is used for path planning in autonomous driving, especially for curved tracks [10]. However, it could feature a high computational complexity, which may yield to the installation of a very demanding hardware mounted on-board in the real application [9][11]. State Lattice-based methods are also exploited for motion planning, although their application is mainly limited to indoor or static driving scenarios since they could be inappropriate in the case of demanding maneuvers [9][12]. Furthermore, local search algorithms are investigated for trajectory planning in the case of structured road scenarios or within a limited road distance [9][13][14]. A global trajectory planning algorithm exploiting ArcGIS maps and a GPS sensor is investigated in [15] for an autonomous electric bus, although it requires the exact knowledge of the destination point. Another trajectory planning method based on the application of Jump Point Search (JPS) on GPS data in ArcGIS maps is presented in [16] for urban environments. Alternative approaches based on simplified polynomial parametrizations of trajectories are investigated in [17] and [18], while a method based on the cubic spline approximation for computing the planned trajectory is presented in [19]. Furthermore, an empirical polynomial method based on clothoid tentacles is studied in [20], by exploiting occupancy grid-maps generated from LIDAR data. A global trajectory optimization method for a racecar is described in [21]. Nevertheless, it requires a robust information about the whole race track, which is obtained by means of a preliminary global mapping procedure. Considering a highway driving scenario, the Adaptive Potential Field (APF) approach is used in [22] for trajectory planning in the presence of obstacles, as well as an algorithm based on Partial Motion Planning (PMP) is proposed by [23]. Rapidly-exploring Random Tree (RRT) methods features a robust real-time kinematic feasibility and quick search of free space, also in driving scenarios characterized by demanding maneuvers, such as race tracks [9][11][13]. A modified RRT-based motion planning algorithm for automated vehicle participating at the DARPA Urban Challenge (DUC) is presented in [24] and [25]. A hybrid method based on both the A* search and RRT algorithms is proposed in [26] for a local trajectory planner. An application of the RRT method in a highway driving scenario is presented in [27]. A complete review of other trajectory planning algorithm can be found in [3][9][11][13]. This paper proposes a RRT-based local trajectory planning and control method for autonomous racing vehicles. Specifically, the algorithm computes the planned trajectory in an unknown environment without the need of a preliminary computed global map representing the driving scenario. The driving scenario is structured with non-crossable obstacles, such as traffic cones according to the Formula Student Germany (FSG) driverless regulations (2019) [28]. Moreover, the proposed method allows to control the vehicle exploiting a combination of a Stanley and a PID controllers, which is proven to be effective for both lateral and longitudinal vehicle dynamics, although it is simpler with respect to other methods based on Model Predictive Control (MPC) [29][30]. Stanley controller was used in the autonomous vehicle of Stanford University which won the DARPA Grand Challenge in 2005 [31]. This control method consists of two main compensation tasks: a compensation stage for the angular error with respect to the target orientation and another one for minimizing the front lateral distance error from the center of the front axle to the nearest point on the path [29][31]. The considered vehicle is a four-wheel drive (4WD) electric racing prototype, which is modeled as a 3 Degree-of-Freedom (3-DOF) bicycle model [32]. The method exploits a perception stage to sense the surrounding environment by means of a LIDAR-based sensor and a high-performance Graphic Processing Unit (GPU). The perception pipeline includes a preliminary ground-plane filtering of LIDAR data using semantic segmentation, which is a fundamental stage to get rid of unnecessary points [33][34]. In the driving scenario, obstacles are detected with a clustering technique that exploits an algorithm based on Support Vector Machine (SVM), that is applied to the filtered point-cloud. Also, alternative voxel-based Artificial Neural Networks (ANNs) have been investigated in the recent literature for object detection exploiting 3D point-cloud for autonomous driving [35]. The performance of the proposed trajectory planning method is evaluated in simulation by using real data recorded by sensors mounted on-board the vehicle while driving a lap on a race track structured according to [28].

II. Method

A. Vehicle modeling

The considered 4WD electric racing vehicle is represented in Fig. 1. The LIDAR sensor is mounted on the front wing of the vehicle. The sensor is fixed at a height of about 8 cm from the ground. The vehicle has an integral carbon fiber chassis built with honeycomb panels, double wishbone push-rod suspensions, an on-wheel planetary transmission system, and a custom aerodynamic package. The vehicle can reach a maximum speed equal to 120 km/h with longitudinal acceleration peaks reaching up to 1.6 g. The main vehicle's parameters are listed in Table 1.

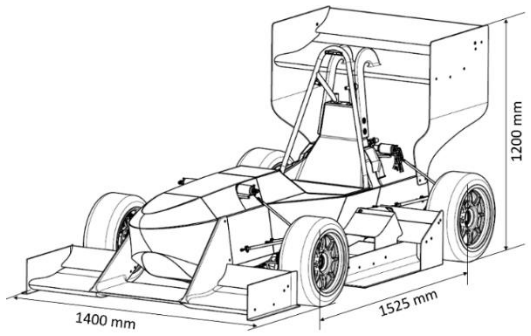

Fig. 1. Considered 4WD electric racing vehicle.

To validate the proposed local trajectory planning and control method, a simplified vehicle model is considered. Thus, the longitudinal and the lateral dynamics of the race car are modeled by means of a 3-DOF bicycle model, because it is proven to be effective among the different mathematical formulations which are present in the literature [30][36]. The relevant states of the vehicle are depicted in Fig. 2. The considered formulation features a rigid two axle vehicle model and it accounts for the linear motion in the xy plane and the rotation about the z-axis. The three equations of motion are developed in a reference frame xy fixed to the Centre-of-Mass of the vehicle (CoM in Fig. 2):

$$ma_x = mV_y\dot{\psi} + F_{xf,w}cos\delta + F_{yf,w}sin\delta + F_{xr,w} \tag{1}$$

$$ma_y = -mV_x\dot{\psi} + F_{yf,w}cos\delta + F_{xf,w}sin\delta + F_{yr,w} \tag{2}$$

$$I_z\ddot{\psi} = a(F_{yf,w}cos\delta + F_{xf,w}sin\delta) - bF_{yr,w} \tag{3}$$

where x and y are the positions along the x-axis and y-axis respectively, ψ is the heading angle, δ is the front wheel steering angle, $F_{xf,w}$ and $F_{xr,w}$ are the longitudinal tire forces applied to front and rear wheels, $F_{yf,w}$ and $F_{yr,w}$ are the lateral tires forces applied to front and rear wheels.

TABLE I. Main vehicle's parameters (* driver included).

Parameter	Symbol	Value	Unit
Mass*	m	253	[kg]
Moment of Inertia about z-axis*	Iz	95.81	[kgm^2]
Vehicle wheelbase	l	1.525	[m]
Overall length	L	2.873	[m]
Front axle distance to CG	a	0.839	[m]
Rear axle distance to CG	b	0.686	[m]
Vehicle track width	t	1.4	[m]
Overall width	W	1.38	[m]
Height of CG*	h_{CG}	0.242	[m]
Wheel radius	R_W	0.241	[m]
Maximum power (total vehicle)	P_{max}	80	[kW]
Motors peak torque (total vehicle)	T_{max}	84	[Nm]
Transmission ratio	τ	14.82	[-]
Maximum energy stored (battery pack)	E_{bp}	6.29	[kWh]

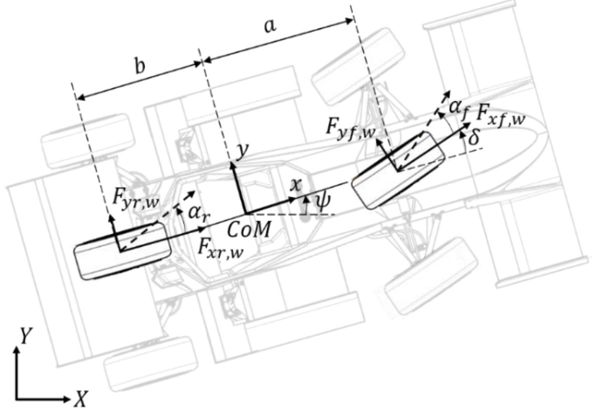

Fig. 2. Bicycle vehicle model applied to the 4WD electric racing vehicle.

The relation between the inertial reference frame XYZ and the vehicle-fixed reference frame is described by the following equation:

$$X = xcos\psi - ysin\psi \tag{4}$$

$$Y = xsin\psi + ycos\psi \tag{5}$$

$$\Psi = \psi \tag{6}$$

The lateral tire forces at the front and rear wheels are considered perpendicular to the rolling direction of the tire, and proportional to the lateral slip angle, α_f and α_r, between its velocity vector in the vehicle fixed reference frame and its forward direction. Considering the assumption of small slip angles, the lateral tire forces are modelled as:

$$F_{yf,w} = 2C_{\alpha f}\alpha_f \tag{7}$$

$$F_{yr,w} = 2C_{\alpha r}\alpha_r \tag{8}$$

where $C_{\alpha f}$ and $C_{\alpha r}$ are the tire stiffness and α_f and α_r are tires slip angles of the front and rear tires, respectively. The tires slip angles are defined as follows:

$$\alpha_f = \text{atan}\left(\frac{V_y + \dot{\psi}a}{V_x}\right) - \delta \tag{9}$$

$$\alpha_r = \text{atan}\left(\frac{V_y - \dot{\psi}b}{V_x}\right) \tag{10}$$

where a and b are distance from the CoM and the front and rear axle, respectively.

B. Perception

The driving scenario consists of a race track structured with traffic cones according to [28]. Each traffic cone has a height equal to 0.325 m and a square base, with a side length equal to 0.228 m.

The cones of the left lane boundary are blue with a white stripe, while the cones delimiting the right lane boundary are yellow with a black stripe. Two couples of bigger orange cones indicate the starting and the ending points of the track. The perception algorithm layout is represented in Fig. 3. The driving environment is sensed with a LIDAR-based sensor (block 1 in Fig. 3). The sensor is a Velodyne VLP-16 Puck, that provides a full 360-degree view of the surrounding environment at 10 Hz to obtain an accurate real-time 3D data reconstruction recorded by 16 light channels. It ranges up to 100 m with 30° vertical field-of-view (FOV) and an angular resolution up to 0.1° in the horizontal plane. The LIDAR sensor is connected to a NVIDIA Jetson AGX Xavier high-performance computing platform with embedded GPUs through an Ethernet connection. The computing platform creates a Robot Operating System (ROS) network, which allows to process the information streaming from the LIDAR-based sensor. This information contains point-cloud data computed at 10 Hz. Each point cloud consists of thousands of 3D points in a 360° range on the horizontal plane, while the vertical FOV is ±15°. Each point-cloud contains the distance of each point in the 3D space along with the intensity of the reflected light in that point. Then, the raw point cloud is filtered by removing all the points out of the region-of-interest (ROI) and the points belonging to the ground-plane (block 2 in Fig. 3). The adopted ground-plane filtering algorithm is developed in the ROS environment. It is based on a standard point-cloud semantic segmentation technique adapted to the specific driving environment and vehicle, setting the sensor position and height. The ground plane filtering is necessary to avoid considering ground points in the following obstacles detection process. A robust theoretical background of the adopted method can be found in [37], [38] and [39]. After the application of the ground removal algorithm to the point-cloud, a ROI is identified in the 3D sensed point-cloud. This stage is convenient to reduce the data dimension and remove outliers in the ROI. Therefore, $ROI(x, y, z)$ is defined in (11):

$$ROI(x,y,z) = \begin{Bmatrix} 0 < x < 25 \\ -10 < y < 10 \\ -0.5 < z < 1.5 \end{Bmatrix} [m] \quad (11)$$

Once the point-cloud has been filtered, a SVM-based algorithm for point-cloud clustering is applied to detect clusters of points (block 3 in Fig. 3). Therefore, each cluster represents a detected object delimiting the race track. The designed perception method is based on the research described in [40]. Nevertheless, the algorithm is properly adapted to the actual vehicle environment structured with traffic cones, by tuning all the parameters relative to the search for clusters of points in the ROI. Thus, multiple clusters of points are extracted from the point cloud at each frame and each of the clusters represents a cone. The position of each cone is estimated as the centroid of the corresponding cluster (block 4 in Fig. 3).

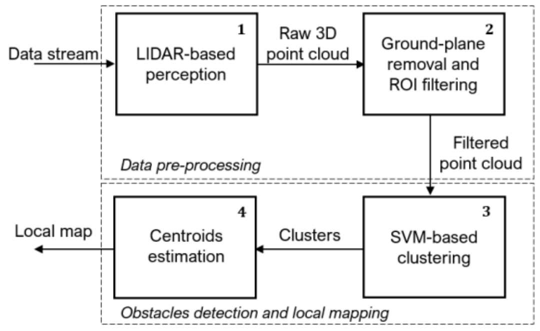

Fig. 3. Layout of the perception algorithm.

The centroid of the cluster is computed as the geometric center of all the points belonging to that cluster. The z-axis is neglected since the aim of the proposed perception algorithm is to find a 2D local map at each frame. Therefore, considering a cluster C in the two-dimensional ROI, its centroid $\gamma(x_\gamma, y_\gamma)$ is computed as in (12).

$$\gamma(x_\gamma, y_\gamma) = \left(\frac{x_1+x_2+\cdots+x_{n_c}}{n_c}, \frac{y_1+y_2+\cdots+y_{n_c}}{n_c}\right) \quad (12)$$

where n_c indicates the number of points in the cluster. Thus, the local 2D map is built with the clusters centroids representing the detected cones with respect to the vehicle's reference frame.

C. Trajectory planning

The purpose of the proposed algorithm is to implement a motion planner that provides a path and a speed profile to the controller, thus the racing vehicle can travel within the track boundaries while avoiding hitting cones. The path planning algorithm is based on the RRT method that takes into account kinematic and dynamic constraints of the vehicle, in addition to the pure geometric problem of obstacles avoidance. This method allows to search non-convex high-dimensional spaces by randomly building a space-filling tree [41]. Starting from a single initial configuration, the path planner algorithm explores the environment around the vehicle by computing a tree of random collision-free poses. An incremental process of sampling makes the tree expand toward sampled configurations in the configuration space. Initially, the search tree only knows the initial states of the vehicle, and terminates when the tree has reached a goal pose. A pseudo-code implementation of the implemented RRT algorithm is shown in Fig. 4, where τ indicates the local search tree that is a directed acyclic graph. The aim of the RRT algorithm is to find a path from the vehicle's initial configuration x_{init} to the vehicle's goal configuration x_{goal}. The search algorithm iteratively tries to add new configurations to the search tree until it reaches n iterations [42]. In detail, the algorithm performs sampling, node selection, expansion, and constraint check until n iterations are reached, and the tree expansion is updated each time new vehicle states are available. The planned path for each link must be feasible and collision-free.

```
**Algorithm** RRT − LocalPathPlanner ($x_{init}$; $x_{goal}$)
$\tau_{INIT}(x_{init})$
**for** $i = 1,2,\dots,n$ **do**
  $x_{samp} \leftarrow sample\_configuration$
  $x_{near} \leftarrow nearest\_neighbour(x_{samp}, \tau)$
  $u_{new} \leftarrow select\_input(x_{near}, x_{samp})$
  $x_{new} \leftarrow extend(x_{near}, u_{new})$
  $dubins_{segment} = generate\_curve(x_{near}, x_{new})$
  **if** $dubins_{segment} \neq nil$**, then**
    $\tau_{ADD-EDGE}(x_{near}, x_{new}, u_{new})$
    **if** $x_{new} \in x_{goal}$**, then**
      *end for*
**return** $\tau$
```

Fig. 4 Pseudo-code for the RRT algorithm.

To this end, a cost map should be generated. As explained in Section B, the data generated by the LIDAR sensor and the estimated vehicle's position create a 2D representation of the approximate locations of the obstacles, which are the left and right cones in this case. The size of the resulting cost map depends both on the quality of the collected data and on the vehicle's position. When the vehicle is traveling on the center line, the road is straight and, the cones are exactly aligned, the longitudinal size of the map can reach up to 60 m. As the vehicle approaches a bending road, the size of the cost map will decrease. Then, each cell in the occupancy grid has a value representing the probability of the occupancy of the considered cell. These values range from 0 to 1. When the value in a single cell is close to 1, there is a high probability that the cell contains an obstacle. On the contrary, if this value is around 0, the cell is probably not occupied and can be considered as obstacle-free. In this work, the highest value is assigned to each identified cone and an inflation coefficient is applied to inflate higher probability across the grid. Since the sensor is mounted in the center of the vehicle front wing, the radius of the inflation area has been heuristically selected equal to 0.5 m. Finally, the initial position and the computed final goal position are computed as the current vehicle's position and as the middle point between the farthest couple of left and right cones. Each time the frame is updated, the initial and final positions are changed accordingly. Afterwards, the RRT is applied to compute the best path avoiding the racing vehicle hits the cones or lies on the inflated areas.

The reference path generated by the path planner is composed of Dubins segments [43]. Therefore, the curvature between two consecutive segments could be not sufficiently smooth. This may yield to abrupt changes in the computed steering angle command. To produce a natural motion leading to improvement on the passenger comfort and vehicle's performance, the path is interpolated by a parametric cubic spline. Once the path is smoothed, the speed profile is computed based on the maximum allowable acceleration and jerk of the vehicle. Thus, when the vehicle is approaching a curve, the velocity is constrained by the maximum lateral acceleration threshold ($a_{y,max} = 2$ g). On the other hand, when the vehicle is traveling on a straight path, the reference speed profile is constrained by the longitudinal speed, acceleration and jerk. In the first case, the minimum value of the velocity is imposed directly by the curvature κ of the computed path:

$$V_{min} = \sqrt{\frac{a_{y,max}}{\kappa}} \tag{13}$$

In the second case, the reference speed profile consists of a trapezoidal transition from the minimum value V_{min} to a maximum allowed speed V_{max}, and eventually again to V_{min} while considering the imposed constraints on accelerations and jerk.

D. Control

The control layout implemented in this paper is presented in Fig. 5. In detail, the control problem has been studied decoupling the longitudinal and lateral dynamics of the racing vehicle. The longitudinal dynamic control is obtained by means of a PID controller that minimizes the error between the actual vehicle's speed and reference speed while the lateral dynamics control aims to minimize the deviation from the desired path [31]. The commands to the vehicle model are the longitudinal acceleration a_x and the steering angle of the front wheels δ. The inputs to the longitudinal controller are the actual vehicle's speed and the reference speed profile from the trajectory computation, as shown in Fig. 5. The mathematical formulation is described by:

$$u(t) = K_P e(t) + K_I \int_0^t e(t)dt \tag{14}$$

where $u(t)$ is the acceleration/deceleration command to the vehicle at time t, K_p and K_I are the proportional and integral gains respectively and $e(t)$ is the velocity error at time t. At each time t, this error is equal to the difference between the current velocity and the reference velocity. Conversely, the lateral controller exploits a non-linear control law to follow the trajectory by considering the orientation of the front wheels with respect to the desired trajectory.

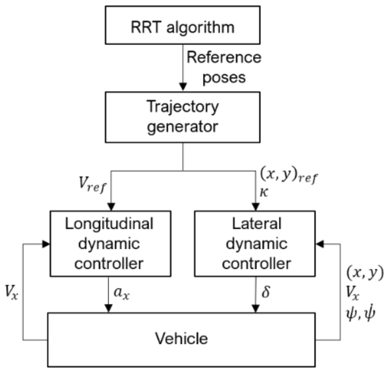

Fig. 5. Block scheme of the control method.

Some variables are defined as follows, before introducing the mathematical formulation of the control law:

$$\psi_{ss} = \frac{m V_x \dot{\psi}_{des}}{C_\alpha\left(1+\frac{a}{b}\right)} \tag{15}$$

$$\dot{\psi}_{des} = V_x \kappa \tag{16}$$

$$\dot{e}_1 = V_y + V_x e_2 \tag{17}$$

where $\dot{\psi}_{des}$ is the rate of change of the desired orientation of the vehicle, V_x is the vehicle's longitudinal velocity, ψ_{ss} is the steady state yaw relative to a constant curvature path, e_2 is the heading angle of the vehicle with respect to the closest trajectory segment, e_1 is the distance of the vehicle's center of mass with respect to the center line and V_y is the vehicle's lateral velocity. Considering (15), (16) and (17), the lateral dynamic controller computes the front wheel steering angle as follows:

$$\delta(t) = (e_2 - \psi_{ss}) + \text{atan}\left(\frac{k e_1}{1+V_x}\right) + k_{yaw}\left(\dot{\psi} - \dot{\psi}_{des}\right) \tag{18}$$

where k and k_{yaw} are two tuneable gains. The presented control law is composed of three main terms: the first one guarantees that the vehicle turns producing a non-zero yaw setpoint when the vehicle is approaching a bend; the second modifies the steering angle such that the future trajectory lies on the tangent to the desired path at a certain distance from the front axle; the latter introduces a damping coefficient at increasing speed. Until the final goal is reached, the following procedure is implemented. Initially, the front wheel steering angle is computed along with the acceleration/deceleration commands required to track the desired trajectory by the lateral and longitudinal decoupled dynamic controllers, respectively. Then, they are provided to the vehicle model. Eventually, the states of the vehicle are recorded to feed them into the controllers at the successive iteration, as represented in Fig. 5.

III. Results

The proposed method is validated in a simulation software environment, which has been developed exploiting both ROS and MATLAB/Simulink. The validation dataset has been recorded during a real acquisition stage performed on-board the vehicle represented in Fig. 1, instrumented with a LIDAR-based sensor and a high-performance GPU. The considered dataset consists of over a thousand of frames. In each frame, a set of possible trajectories, whose number varies between 200 and 300, are randomly generated during the planning phase. The whole validation dataset includes a wide range of maneuvers, including sudden accelerations while turning or coasting, performed by the vehicle in a racing scenario that was properly structured with traffic cones according to regulations listed in [28]. In the following figures (from Fig. 6 to Fig. 10), the planned trajectory is represented by a blue solid line, with waypoints indicated as blue points. The goal vehicle position is indicated by the red dot. The detected cones are represented with black triangles, and inflated areas are illustrated by pink dots. The planned vehicle's motion is indicated by blue shapes. Moreover, the longitudinal acceleration a_x and the steering angle δ commands are shown in subfigures b) and c), respectively. Subfigures d) illustrate the reference longitudinal speed V_{ref} (indicated with red dashed line) with respect to the actual vehicle's speed V_x (black solid line). The vehicle's yaw rate $\dot{\psi}$ is shown in subfigure e) for each maneuver.
An example of the recorded maneuvers is represented in Fig. 6. The vehicle is accelerating on a straight portion of the racing track. Fig 6.a illustrates the planned vehicle's motion (successive blue shapes) that starts from standstill in the origin. The vehicle performs the maneuver according to the planned trajectory computed by the investigated RRT-based algorithm (blue line in Fig. 6.a), until it reaches the goal position (red dot in Fig. 6.a). The cones detected by the LIDAR-based perception algorithm are represented with black triangles in Fig. 6.a, while the inflated areas used by the RRT algorithm are represented by pink circles. The vehicle receives the steering and longitudinal acceleration commands from the designed control algorithm, which are illustrated in Fig. 6.b and Fig. 6.c, respectively. The wheels steering angle δ is always negligible during the maneuver. The vehicle reaches over 15 m/s in about 2.5 s, as illustrated in Fig. 6.d. The reference longitudinal speed V_{ref} is accurately followed by the actual vehicle's longitudinal speed V_x. The vehicle's yaw rate $\dot{\psi}$ reaches 2.5 deg/s at the end of the maneuver, as represented in Fig. 6.e. Fig. 7 illustrates the results of

an acceleration maneuver while cornering to the right. The vehicle accelerates in a curved portion of the racing track delimited by traffic cones (black triangles in Fig. 7.a). Fig 7.a illustrates the planned vehicle's motion (successive blue shapes) that starts from standstill, as well as the planned trajectory computed by the proposed algorithm (blue line in Fig. 7.a). The vehicle accelerates up to 10 m/s in about 2 s, as illustrated in Fig. 7.d, while the wheels steering angle is represented in Fig. 7.c. The actual vehicle's speed is accurate with respect to V_{ref}. The vehicle's yaw rate $\dot{\psi}$ reaches up to 5 deg/s during the maneuver as represented in Fig. 7.e. An acceleration maneuver while cornering to the left is represented in Fig. 8. The vehicle starts from standstill and reaches up to 10 m/s. Fig 8.a shows the computed trajectory and the planned vehicle's motion. During the cornering maneuver, the wheels steering angle δ reaches up to -10 deg in Fig. 8.c. The actual vehicle's speed is accurate with respect to V_{ref}, as illustrated by Fig 8.d. The vehicle's yaw rate $\dot{\psi}$ is represented in Fig. 8.e. Results obtained for two coasting maneuvers while cornering to the right and left are represented in Fig. 9 and Fig. 10, respectively. The vehicle follows the planned trajectory in both cases, while keeping the longitudinal speed constant at 8 m/s (Fig. 9.d and Fig. 10.d). The wheels steering angle δ reaches up to 20 deg and -18 deg in the two cases, respectively (Fig. 9.c and Fig. 10.c). The vehicle's yaw rates $\dot{\psi}$ for the two maneuvers are represented in Fig. 9.e and Fig. 10.e, respectively.

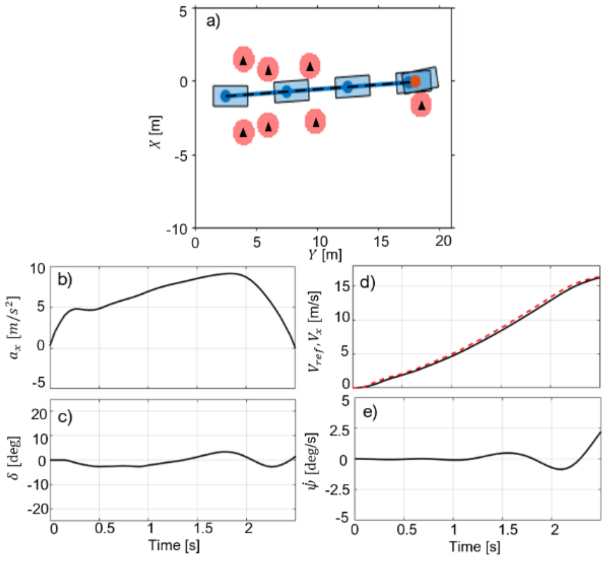

Fig. 6. Acceleration test on straight road.

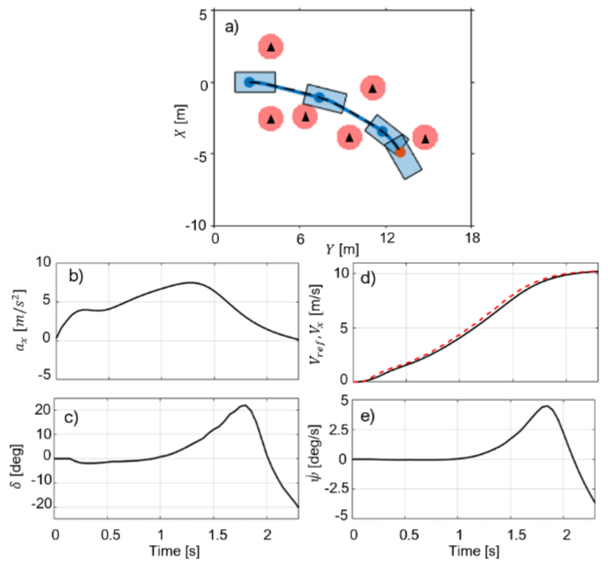

Fig. 7. Acceleration while cornering to the right.

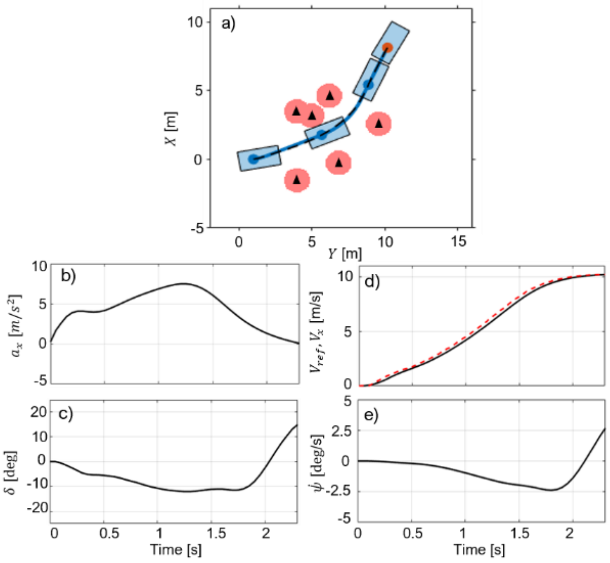

Fig. 8. Acceleration while cornering to the left.

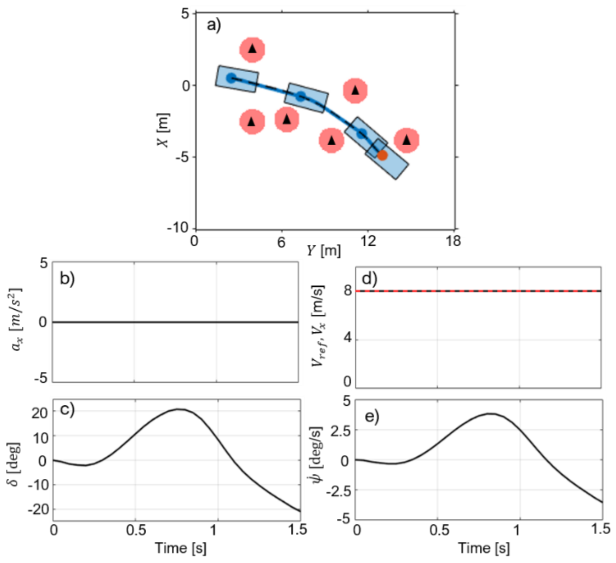

Fig. 9. Coasting while cornering to the right.

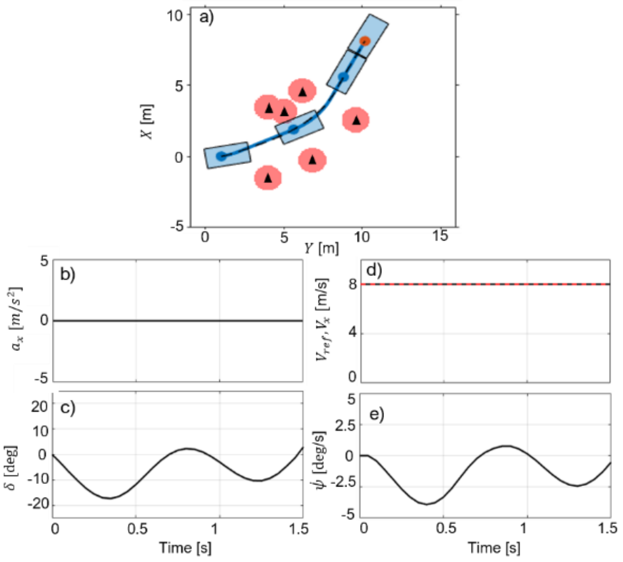

Fig. 10. Coasting while cornering to the left.

Conclusion

A local trajectory planning and control method for autonomous racing vehicles was presented. To this end, an algorithm based on the RRT approach was exploited. The considered 4WD electric prototype is modeled with a 3-DOF linear bicycle model. The vehicle dynamics was controlled with a Stanley control strategy. The method has been validated in a simulation environment built with real data, which are recorded by perception sensors mounted on-board the vehicle. The investigated algorithm computes feasible trajectories during different maneuvers performed in the driving scenario. The method requires a further testing stage to real-time validate the achieved results on the instrumented vehicle.

Acknowledgments

This work was developed in the framework of the activities of the Interdepartmental Center for Automotive Research and Sustainable Mobility (CARS) at Politecnico di Torino (www.cars.polito.it).

References

[1] Hussain, R., and Sherali Z. "Autonomous Cars: Research, Results, Issues, and Future Challenges." IEEE Communications Surveys & Tutorials 21.2: 1275-1313, 2018.

[2] Koopman, P., & Wagner, M. “Challenges in autonomous vehicle testing and validation”. SAE International Journal of Transportation Safety, 4(1), 15-24. 2016.

[3] Claussmann, L., Revilloud, M., Gruyer, D., & Glaser, S. “A review of motion planning for highway autonomous driving”. IEEE Transactions on Intelligent Transportation Systems. 2019.

[4] Bonfitto, A., Feraco, S., Amati, N., & Tonoli, A. “Virtual Sensing in High-Performance Vehicles with Artificial Intelligence.” In ASME 2019 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. American Society of Mechanical Engineers Digital Collection, 2019.

[5] Bonfitto, A., Feraco, S., Tonoli, A., & Amati, N. “Combined regression and classification artificial neural networks for sideslip angle estimation and road condition identification”. Vehicle System Dynamics, 1-22, 2019.

[6] Yurtsever, E., Lambert, J., Carballo, A., & Takeda, K.“A Survey of Autonomous Driving: Common Practices and Emerging Technologies”. arXiv preprint arXiv:1906.05113, 2019.

[7] Kaur, K. and Rampersad G., “Trust in driverless cars: Investigating key factors influencing the adoption of driverless cars”, Journal of Engineering and Technology Management, Vol.48, pp. 87-96, 2018.

[8] Litman, T. “Autonomous vehicle implementation predictions”. Victoria, Canada: Victoria Transport Policy Institute, 2017.

[9] Katrakazas, C., Quddus, M., Chen, W. H., & Deka, L. “Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions”. Transportation Research Part C: Emerging Technologies, 60, 416-442. 2015.

[10] Kavraki, L., Svestka, P., Latombe, J.-C., and Overmars, M. “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” IEEE Trans. on Robotics and Automation, vol. 12, no. 4, pp. 566–580, 1996.

[11] Paden, B., Čáp, M., Yong, S. Z., Yershov, D., & Frazzoli, E. “A survey of motion planning and control techniques for self-driving urban vehicles”. IEEE Transactions on intelligent vehicles, 1(1), 33-55, 2016.

[12] González-Sieira, A., Mucientes, M. and Bugarín, A. "A state lattice approach for motion planning under control and sensor uncertainty." ROBOT2013: First Iberian robotics conference. Springer, Cham, 2014.

[13] Schwarting, W., Alonso-Mora, J., & Rus, D. “Planning and decision making for autonomous vehicles”. Annual Review of Control, Robotics, and Autonomous Systems, 1, 187-210. 2018.

[14] Wang, M., Ganjineh, T., & Rojas, R. “Action annotated trajectory generation for autonomous maneuvers on structured road networks.” In The 5th International Conference on Automation, Robotics and Applications (pp. 67-72). IEEE, 2011.

[15] Yu, L., Kong, D., & Yan, X. “A driving behavior planning and trajectory generation method for autonomous electric bus”. Future Internet, 10(6), 51. 2018.

[16] Zhou, K., Yu, L., Long, Z., & Mo, S. “Local path planning of driverless car navigation based on jump point search method under urban environment”. Future Internet, 9(3), 51. 2017.

[17] Minh, V. T. “Trajectory Generation for autonomous vehicles.” In Mechatronics 2013 (pp. 615-626). Springer, Cham. 2014.

[18] Yin, G., Li, J., Jin, X., Bian, C., & Chen, N. “Integration of motion planning and model-predictive-control-based control system for autonomous electric vehicles.” Transport, 30(3), 353-360. 2015.

[19] Horst, J., & Barbera, A. (2006, May). Trajectory generation for an on-road autonomous vehicle. In Unmanned Systems Technology VIII (Vol. 6230, p. 62302J). International Society for Optics and Photonics.

[20] Alia, C., Gilles, T., Reine, T., & Ali, C. “Local trajectory planning and tracking of autonomous vehicles, using clothoid tentacles method.” In 2015 IEEE Intelligent Vehicles Symposium (IV) (pp. 674-679). IEEE, 2015.

[21] Caporale, D. et al. “A planning and control system for self-driving racing vehicles”. In 2018 IEEE 4th International Forum on Research and Technology for Society and Industry (RTSI) (pp. 1-6). IEEE, 2018.

[22] Kim, D., Kim, H., & Huh, K. “Local trajectory planning and control for autonomous vehicles using the adaptive potential field.” In 2017 IEEE Conference on Control Technology and Applications (CCTA) (pp. 987-993) IEEE, 2017.

[23] Resende, P., & Nashashibi, F. “Real-time dynamic trajectory planning for highly automated driving in highways.” In 13th International IEEE Conference on Intelligent Transportation Systems (pp. 653-658). IEEE, 2010.

[24] Kuwata, Y., Teo, J., Fiore, G., Karaman, S., Frazzoli, E., & How, J. P. “Real-time motion planning with applications to autonomous urban driving.” IEEE Transactions on control systems technology, 17(5), 1105-1118. 2009.

[25] Buehler, M., Iagnemma, K., & Singh, S. (Eds.). “The DARPA urban challenge: autonomous vehicles in city traffic” (Vol. 56). Springer. 2009

[26] Tanzmeister, G., Friedl, M., Wollherr, D., & Buss, M. “Path planning on grid maps with unknown goal poses.” In 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013) (pp. 430-435). IEEE. 2013.

[27] Ma, L., Xue, J., Kawabata, K., Zhu, J., Ma, C., & Zheng, N. “A fast RRT algorithm for motion planning of autonomous road vehicles.” In 17th International IEEE Conference on Intelligent Transportation Systems (ITSC) (pp. 1033-1038). IEEE. 2014.

[28] Formula Student Germany, “FSG Competition Handbook 2019”, 2019.

[29] Dominguez, S., Ali, A., Garcia, G., & Martinet, P. “Comparison of lateral controllers for autonomous vehicle: Experimental results.” In 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC) (pp. 1418-1423). IEEE. 2016.

[30] Feraco, S., Bonfitto, A., Amati, N., & Tonoli, A. (2019, August). Combined Lane Keeping and Longitudinal Speed Control for Autonomous Driving. In International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (Vol. 59216, p. V003T01A018). American Society of Mechanical Engineers.

[31] Thrun, Sebastian, et al. "Stanley: The robot that won the DARPA Grand Challenge." Journal of field Robotics 23.9: 661-692. 2006.

[32] Rajamani R., “Vehicle Dynamics and Control”, Mechanical Engineering Series, 2012.

[33] Liu, K., Wang, W., Tharmarasa, R., Wang, J., & Zuo, Y. “Ground surface filtering of 3D point clouds based on hybrid regression technique.” IEEE Access, 7, 23270-23284. 2019.

[34] Wang, L., & Zhang, Y. “LiDAR Ground Filtering Algorithm for Urban Areas Using Scan Line Based Segmentation.” arXiv preprint arXiv:1603.00912. 2016.

[35] Zhou, Y., & Tuzel, O “Voxelnet: End-to-end learning for point cloud based 3d object detection.” In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 4490-4499). 2018.

[36] Amer, Noor & Zamzuri, Hairi & Hudha, Khisbullah & Kadir, Zulkiffli. “Modelling and Control Strategies in Path Tracking Control for Autonomous Ground Vehicles: A Review of State of the Art and Challenges.” Journal of Intelligent & Robotic Systems.2016.

[37] Miądlicki, K., Pajor, M., & Saków, M. “Ground plane estimation from sparse LIDAR data for loader crane sensor fusion system.” In 2017 22nd International Conference on Methods and Models in Automation and Robotics (MMAR) (pp. 717-722). IEEE. 2017.

[38] Himmelsbach, M., Hundelshausen, F. V., & Wuensche, H. J., “Fast segmentation of 3D point clouds for ground vehicles.” In 2010 IEEE Intelligent Vehicles Symposium (pp. 560-565). IEEE. 2010.

[39] Zermas, D., Izzat I., and Papanikolopoulos, N.. "Fast segmentation of 3d point clouds: A paradigm on lidar data for autonomous vehicle applications." 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2017.

[40] Yan, Z., Duckett, T., & Bellotto, N. “Online learning for 3D LiDAR-based human detection: experimental analysis of point cloud clustering and classification methods.” Autonomous Robots, 44(2), 147-164. 2020.

[41] S. Karaman and E. Frazzoli, ‘Incremental sampling-based algorithms for optimal motion planning’, Robotics: Science and Systems, vol. 6, pp. 267–274, 2011.

[42] Walker, A. “Hard real-time motion planning for autonomous vehicles.” PhD thesis, Swinburne University, 2011.[43] D. Živojević and J. Velagić, "Path Planning for Mobile Robot using Dubins-curve based RRT Algorithm with Differential Constraints," 2019 International Symposium ELMAR, Zadar, Croatia, 2019, pp. 139-142, doi: 10.1109/ELMAR.2019.8918671.