2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

Trajectory Planning based on Model Predictive Control with Dynamic Obstacle Avoidance in Unstructured Environments

Giulio Borrello, Luca Lorusso, Michele Basso and Antonio Acernese

Abstract— **Trajectory planning at low speed applications can involve a large variety of different scenarios, including structured and unstructured environments, pedestrians, cyclists, etc. In this context, real-time planning is crucial to the dexterity of autonomous vehicles. In this paper, a real-time trajectory planner based on model predictive control (MPC) is proposed. Moreover, a dynamic obstacle avoidance and narrow passages control are designed in a scalable way through continuous updates of the optimization constraints. The performance of the proposed methodology are evaluated with a V-cycle model-based approach, through both model-in-the-loop (MIL) simulations, and real prototype in-vehicle experiments.**

I. Introduction

Nowadays the research and industry communities on autonomous driving at low speeds are advancing at a steady and fast pace. The applications cover different sectors and domains, such as military and defense [1], industrial logistics, e.g., autonomous trucks and delivery vehicles [2], working vehicles, e.g., fire truck, agricultural, and engineering machinery [3], [4], public transport, e.g., airport and minibus shuttle [5], and private and/or shared ones. All these heterogeneous applications share the common task of following a path to reach the desired endpoint by moving at low speed (maximum 20 km/h), and reacting in real-time to unexpected situations.

Generally speaking, the trajectory planning (TP) problem in control and automation engineering aims to find a motion law along a given geometric path, while taking into account some requirements, e.g., kinematic and dynamic constraints. Particularly to the field on autonomous vehicles, TP and trajectory tracking represent two important functions, and a variety of strategies have been proposed in the literature for both functionalities. A comprehensive review of vehicle control techniques can be found in [6]. In general, vehicle controllers are classified into three categories: path stabilization, trajectory tracking control, and predictive control approaches. Path stabilization methods aim to control the vehicle to follow a path by using geometric approaches, as it is reported in [7], where a pure pursuit control generates a desired turning radius of the vehicle based on its current state (position and orientation), and the state of a point ahead on the planned path. These types of controls are quite popular due to their low computational cost, ease of implementation, and acceptable performance at low speeds. Trajectory tracking controllers aim at tracking a time-variant reference trajectory while guaranteeing asymptotic zero error. Most of these methods relies on static and dynamic linearization of state feedback laws [8], PID controllers [9], and optimal LQR controllers [10]. However, these methods generally do not consider the physical constraints of the vehicle, and suffer from high level of uncertainty. Predictive control approaches use the state-space model of the vehicle to describe its kinematics. One of the most powerful and effective TP controllers is the model predictive control (MPC), which is formulated as an optimization problem which can handle multiple variables and constraints. Besides, it has inherent robustness against uncertainties. These advantages, combined with the optimality guarantees, made the MPC very popular in the control community, and several approaches have been proposed to design MPCs for TP, including the absence (or presence) of one or more constraints (actuator's limits, obstacles, etc.) [11], [12], the number of controllers [13], [14], [15], and the type of vehicle model used, linear or nonlinear [16], [17], [18].

This paper solves the TP problem for low speed applications, such as parking valets and home-zones, using an MPC-based approach. This problem covers different challenging scenarios. For example, consider a driver that leaves or picks up the vehicle in the drop-off areas, such as an airport or a train station, and the vehicle autonomously navigates in the parking area and performs the parking, or exits from it. Another example includes the private contexts, in which the driver records a complex maneuver and the vehicle re-executes it while optimizing the maneuver according to specific criteria. This work takes the foundations from a recently published paper [19] that proposes an MPC-based TP in unstructured environments. Although the formulation of the TP problem in terms of the methodology that we use does not change, we improve the existing paper in three main aspects. Particularly, the proposed TP: i) is independent on the choice of the higher-level motion planning module; ii) automatically optimizes the length of the prediction horizon used in the MPC, through the definition of a proper roto-translation of the environments; iii) takes into account the real size of the vehicle and the presence of dynamic obstacles through the dynamic constraints. Hence, we strongly believe that this paper would be helpful for the autonomous vehicle research community.

The paper is structured as follows: Section 2 presents the vehicle model used to design the controllers; Section 3 provides the control problem formulation and the solution design; Section 4 reports the simulations and the experimen-

G. Borrello, L. Lorusso, M. Basso, and A. Acernese are with the Department of Automated Driving Control, CRF, Strada Torino 50, 10043 Orbassano, Italy. {giulio.borrello, luca.lorusso, michele.basso, antonio.acernese}@stellantis.com.

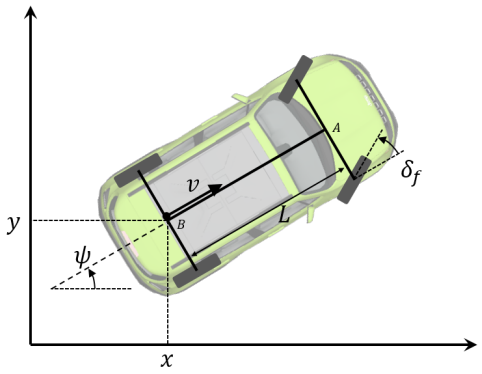


Fig. 1: 4 D.o.F. kinematic bicycle model.

tal results, validated in different real-world scenarios. Some conclusions are drawn in Section 5.

II. Preliminaries: Vehicle Model

In this section we present the model of the vehicle, assumed to be a car, and we limit the speed limit to the range $0 \div 20\ km/h$, i.e., at low speed.

As shown in [16] and [20], at low speed the 4 D.o.F. kinematic bicycle model is one of the simplest and well-conditioned models used to approximate the motion of the vehicle in the context of autonomous vehicles. In this model, the left and right wheels are approximated with two single wheels at points A and B, respectively for the front and the rear axle (Fig. 1). Front and rear steering angles are represented by $\delta_f \in \mathbb{R}$ and $\delta_r \in \mathbb{R}$, respectively. Moreover, i) the side-slip angle of the front and rear wheels are assumed to be negligible, and ii) $\delta_r = 0$, hence $\delta := \delta_f \in [\delta_{min}, \delta_{max}]$, with δ_{min} and δ_{max} minimum and maximum steering angles.

The resulting kinematic bicycle model (Fig. 1) is described with the following state-space equations in the inertial frame:

$$\dot{x} = v\ cos(\psi) \tag{1}$$

$$\dot{y} = v\ sin(\psi) \tag{2}$$

$$\dot{\psi} = \frac{v}{L}\ tan(\delta) \tag{3}$$

$$\dot{v} = a. \tag{4}$$

In particular, $x \in \mathbb{R}$, $y \in \mathbb{R}$ are the Cartesian coordinates of the vehicle's rear wheel, and $\psi := [-\pi, \pi] \subseteq \mathbb{R}$ is the yaw angle. $v := [v_{min}, v_{max}] \subseteq \mathbb{R}$ and $a := [a_{min}, a_{max}] \subseteq \mathbb{R}$ denote the velocity and the longitudinal acceleration, respectively, within acceptable ranges. Thus, the state and input vectors of this model can be defined as $X = [x, y, \psi, v]$ and $U = [\delta, a]$, respectively.

The non linear model described above can be divided into longitudinal and lateral dynamics, to obtain two independent yet simpler sub-models. Moreover, a nonlinear state and input transformation can be applied to the lateral model, namely the time-state control form (T-SCF), [21], [22], which allows to represent equations (2) and (3) as linear differential equations w.r.t. the space, namely:

$$\begin{bmatrix} \frac{\partial z_2}{\partial z_1} \\ \frac{\partial z_3}{\partial z_1} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} z_2 \\ z_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mu_2, \tag{5}$$

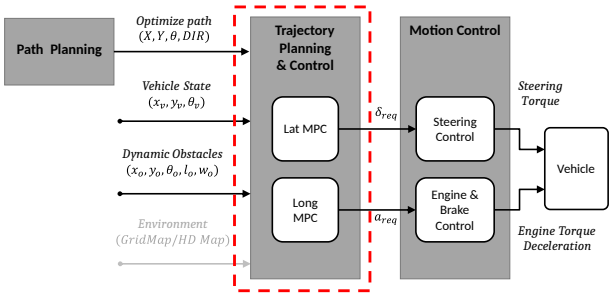


Fig. 2: General automated driving control scheme.

where $z_1 = x$, $z_2 = y$, $z_3 = tan(\psi)$, and $\mu_2 = \frac{tan(\delta)}{L\ cos^3(\psi)}$.

Furthermore, the longitudinal dynamics of the vehicle can be accurately approximated with a double integrator system, in which the states are the travelled distance $\xi \in \mathbb{R}$ and v, and the control input is a:

$$\begin{bmatrix} \frac{\partial \xi}{\partial t} \\ \frac{\partial v}{\partial t} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \xi \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} a. \tag{6}$$

III. Trajectory Planning Design

In this section we provide an introduction of the TP module, and we propose our MPC-based TP methodology.

The general scheme of the proposed control architecture is shown in Fig. 2. It is designed as a hierarchical structure comprising three main modules. The higher-level module is the motion planning, which outputs a static path $P := [p_1, p_2, \ldots, p_N]$, that guarantees a collision-free and drivable path from the starting point p_1 to the ending state p_N, where the i-th generic waypoint is $p_i := (x_i, y_i, \psi_i, r_i)$, and $r_i \in [0, 1]$ represents the vehicle's motion direction, backward and forward, respectively. Note that the motion planning considers a constant velocity and does not provide any information on how this path should be followed. The time can be included in the lower level TP module in terms of velocity profile to be followed. Given the static path P, the TP post-processes it to (i) enhance comfort, (ii) combine the static generated path with the time, and (iii) manage dynamic objects, e.g., pedestrians, bicyclists, other vehicles, etc. To this end, TP requires additional inputs coming from the perception module, revealing the actual states of such objects, in terms of their positions and orientations, as well as their dimensions in the space. The output of the TP module is to provide, at each discrete time-step $k \in \mathbb{N}$, the high-level vehicle commands, i.e., δ and a. The connection of such control requests to the actuators of the vehicle is demanded to the lowest-level control block, called motion control, which computes the torque requests of the electronic power steering system, the engine torque, and the deceleration requests of the brake system module, respectively.

In the following, we formulate the problem of TP as the combination of two independent MPC problems, for lateral and longitudinal dynamics.

A. Lateral MPC

Given the vehicle dynamic model (5), the lateral controller aims to provide the steering angle to follow the optimal path according to the vehicle's constraints on δ_{min} and δ_{max},

and the environmental constraints on the free space corridor. Applying the Tustin discretization to (5), we obtain:

$$\begin{bmatrix} z_2(k+1) \\ z_3(k+1) \end{bmatrix} = \begin{bmatrix} 1 & \Delta_s \\ 0 & 1 \end{bmatrix} \begin{bmatrix} z_2(k) \\ z_3(k) \end{bmatrix} + \begin{bmatrix} \Delta_s^2/2 \\ \Delta_s \end{bmatrix} \mu_2(k), \quad (7)$$

where Δ_s is the discretization step w.r.t. the time-state $z_1 = x$, and it is positive when the vehicle moves forward, and negative otherwise. The lateral control problem can be formulated as an MPC problem as follows:

$$\begin{aligned} &\mathbf{J}(z_2, z_3) \\ &= \begin{bmatrix} y_{ref_{H_{lat}}} - z_2(H_{lat}) \\ -z_3(H_{lat}) \end{bmatrix}^T \mathbf{Q_f} \begin{bmatrix} y_{ref_{H_{lat}}} - z_2(H_{lat}) \\ -z_3(H_{lat}) \end{bmatrix} \\ &+ \sum_{k=0}^{H_{lat}-1} \begin{bmatrix} y_{ref_k} - z_2(k) \\ -z_3(k) \end{bmatrix}^T \mathbf{Q} \begin{bmatrix} y_{ref_k} - z_2(k) \\ -z_3(k) \end{bmatrix} + \mathbf{R}\mu_2^2(k), \end{aligned} \quad (8)$$

subject to:

$$\begin{aligned} &Y_{min_k} \leq z_2(k) \leq Y_{max_k} \quad \forall k \in (0, H_{Lat}] \\ &\tan(\delta_{min}) \leq \hat{\Gamma}(z_3, \mu_2) \leq \tan(\delta_{max}) \\ &\text{system (7)}, \end{aligned}$$

where Q, Q_r are the semi-positive definite weighting matrix of the states, R is the positive definite weighting matrix of the inputs, $H_{lat} \in \mathbb{N}$ is the prediction horizon, and $y_{ref_k} \in \mathbb{R}^{H_{lat}+1}$ is the centerline of the left and right free-spaces $Y_{max_k} \in \mathbb{R}^{H_{lat}+1}$ and $Y_{min_k} \in \mathbb{R}^{H_{lat}+1}$. They are obtained by discretizing (by Δ_s) the segments that connect the path planning waypoints. The combination of these two vectors is generally referred to as the free corridor of the TP. Moreover, $\hat{\Gamma}(\cdot, \cdot)$ is the linear approximation of $\Gamma(z_3, \mu_2) = lcos^3(tan^{-1}(z_3))\mu_2$, obtained through a first-order Taylor expansion of $\Gamma(\cdot, \cdot)$, defined to include the constraint on the steering angle ($\delta_{min} \leq \delta \leq \delta_{max}$) in the transformed system.

The choice of using a linear MPC with linear constraints guarantees that the solver computes optimal solutions in real time application. However, it does not allow to handle explicitly nonlinear constraints in (8), e.g., take into account the vehicle size, or avoid collisions with dynamic obstacles. In the next subsections we provide a solution to indirectly include such constraints by modifying the free corridor (Y_{min_k}, Y_{max_k}).

Remark 1: It is worth mentioning that in Problem (8) and throughout the rest of the paper, with a slight abuse of the notation we use the variables x, y to represent the position of the car in a convenient roto-translated frame. Generally speaking, the MPC problem could be solved in the vehicle coordinate system. However, the T-SCF introduces singularities, and strongly limits the prediction horizon. To obtain better performance, one can apply an *ad-hoc* roto-translation of the references and the lateral constraints, by translating w.r.t. the vehicle position, and rotating of an angle θ_{Rot}. The choice of this angle has significant effects on the performance of the MPC.

B. Longitudinal MPC

On the same stream of (7), the longitudinal dynamics (6) can be discretized leveraging the Tustin rule, obtaining the following discrete-time longitudinal model:

$$\begin{bmatrix} \xi(k+1) \\ v(k+1) \end{bmatrix} = \begin{bmatrix} 1 & \Delta_t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \xi(k) \\ v(k) \end{bmatrix} + \begin{bmatrix} \Delta_t^2/2 \\ \Delta_t \end{bmatrix} a(k), \quad (9)$$

where $\Delta_t \in \mathbb{R}$ is the discretization step w.r.t. time.
The MPC-based TP problem for the longitudinal control can be then formulated as:

$$\begin{aligned} &\mathbf{J}(\xi, v) \\ &= \begin{bmatrix} \xi_{ref_{H_{long}}} - \xi(H_{long}) \\ v_{ref_{H_{long}}} - v(H_{long}) \end{bmatrix}^T \mathbf{Q_f} \begin{bmatrix} \xi_{ref_{H_{long}}} - \xi(H_{long}) \\ v_{ref_{H_{long}}} - v(H_{long}) \end{bmatrix} \\ &+ \sum_{k=0}^{H_{long}-1} \begin{bmatrix} \xi_{ref_k} - \xi(k) \\ v_{ref_k} - v(k) \end{bmatrix}^T \mathbf{Q} \begin{bmatrix} \xi_{ref_k} - \xi(k) \\ v_{ref_k} - v(k) \end{bmatrix} + \mathbf{R}a^2(k), \end{aligned} \quad (10)$$

subject to:

$$\begin{aligned} &\xi_{min_k} \leq \xi(k) \leq \xi_{max_k} \quad \forall k \in (0, H_{Long}] \\ &v_{min} \leq v(k) \leq v_{max} \\ &a_{min} \leq a(k) \leq a_{max} \\ &j_{min}\Delta_t \leq a(k) - a(k-1) \leq j_{max}\Delta_t \\ &\text{system (9)}, \end{aligned}$$

where $H_{long} \in \mathbb{N}$ is the prediction horizon, $j_{min} \in \mathbb{R}$, $j_{max} \in \mathbb{R}$ are minimum and maximum allowed jerks, and v_{ref_k} is the vehicle speed reference. The reference on the travelled distance ξ_{ref_k} acts as an integral action to handle stationary errors due to exogenous disturbances and/or inaccuracies in the motion control module. It is computed as

$$\xi_{ref_k} = (D_{ref} - D_{meas}) + v_{ref_k}\Delta_t k, \quad (11)$$

where $D_{ref} = \sum_{i=t_0}^{t_k} v_{ref}(i)\Delta_t$, D_{meas} is the measured travelled distance (from sensors), and D_{ref} is the expected travelled distance computed from the longitudinal control activation time (t_0) to the current time step (t_k). Moreover, ξ_{max} determines the maximum value of the travelled distance (ξ_{Goal}), and, whenever reached, it constrains the vehicle to stop.

C. Rotation angle optimization

Given the set of waypoints provided by the path planning, P, the proposed TP firstly re-samples it in order to consider only the necessary ones, based on a trade-off between accuracy in the approximation of the path, and dimensions of the vector that will then be used by the MPC. The outputs of this stage are the vectors $X_{SEG}, Y_{SEG}, \Theta_{SEG}$. Afterwards, leveraging such vectors, the TP module realizes the so-called free corridor, (blue and red lines in Fig. 4) inside which the vehicle has to be during the maneuver. Such limits (Y_{min}, Y_{max}) are designed based on a user-defined distance from the corresponding waypoints, and are then roto-translated in the MPC-frame (see remark 1), and updated based on the current vehicle position and rotation

angle. Although problem (8) can be solved efficiently and guarantees a deviation from the reference path that shrinks to zero over the time-steps, it has two well-known limitations, namely: i) the T-SCF-based model shall work for $-\pi/2 < \psi < \pi/2$, since it introduces singularities in $\psi = \pm\pi/2$, and ii) there is a linearization error, despite small, related to the first-order approximation of $\Gamma(z_3, \mu_2)$.

A natural solution to this problem, that has been proposed in [19], is to rotate the path in the vehicle frame in order to manage cases in which the vehicle has to follow tight turns, thus avoiding singularities. Particularly, assume that at a certain time-step the vehicle is located at the i^{th} segment, as shown in Fig. 3a, where a segment is the straight line linking two consecutive waypoints. Then, the problem of the singularities can be easily addressed by dynamically determining a rotation angle θ_{Rot}, defined as a linear interpolation of the actual segment orientation, θ_i, and the following two, θ_{i+1} and θ_{i+2}, eventually weighting it by a "corrective factor" $W \in \mathbb{R}$. The main drawback of this approach is that it considers only few consecutive waypoints, thus potentially obtaining a myopic predictive controller.

Algorithm 1 proposes an alternative to the described solution. It aims to improve and optimize the dynamic choice of the three orientations used in the computation of θ_{Rot}, and thus optimizes and maximizes the path section used for the prediction in the MPC and avoids unnecessary rotations.

Algorithm 1 Rotation angle optimization

Input: $X_{SEG}, Y_{SEG}, \Theta_{SEG}, i_{MAX}, i_k$
Output: θ_{Rot_k}
  $\Theta_{Rot_0} \leftarrow \Theta_{SEG}(0)$
  $\Theta_{OUT}(0) \leftarrow \Theta_{SEG}(0)$
  $I_{OUT}(0) \leftarrow 0$
  $\theta_{ref} \leftarrow \Theta_{SEG}(0)$
  $n \leftarrow 1, i \leftarrow 1, j \leftarrow i+1$
  while $j \leq i_{MAX}$ **do**
    $\theta_{new} \leftarrow ComputeAngle(X_i, Y_i, X_{j+1}, Y_{j+1})$
    while $|\theta_{ref} - \theta_{new}| \leq \theta_{th}$ **do**
      $\theta_{new} \leftarrow ComputeAngle(X_i, Y_i, X_{j+1}, Y_{j+1})$
      $j \leftarrow j+1$
    end while
    $\Theta_{OUT}(n) = \theta_{new}$
    $\theta_{ref} = \theta_{new}$
    $I_{OUT}(n) = j$
    $n \leftarrow n+1, i \leftarrow j, j \leftarrow j+1$
  end while
  $i_{0_k}, i_{1_k} \leftarrow PickIndex(i_k, I_{OUT})$
  $\theta_{0_k}, \theta_{1_k}, \theta_{2_k} \leftarrow PickOrientation(i_k, \Theta_{OUT})$
  $\theta_{Rot_k} \leftarrow ComputeRotationAngle$
    $(i_{0_k}, i_{1_k}, \theta_{0_k}, \theta_{1_k}, \theta_{2_k}, \theta_{Rot_{k-1}})$, through (12)

Starting from the first segments and taking its orientation as reference θ_{ref}, Algorithm 1 computes and stores in Θ_{OUT} the highest orientation of the equivalent segment created between the first point and the future ones that satisfies the condition $|\theta_{ref} - \theta_{new}| \leq \theta_{th}$, where $\theta_{th} < \pi/2$ is a user-defined threshold. The obtained orientation is then used as a new reference θ_{ref}, and the process repeates until the last waypoint is reached. Then, at each time-step k, the functions $PickIndex$ and $PickOrientation$, given the index i_k of the segment where the vehicle is located, extract the information necessary to compute the rotation angle at that time, through the formula:

$$\theta_{Rot_k} = W \frac{(\theta_{2_k} - \theta_{1_k})d_k}{d_{1,0_k}} + \frac{(\theta_{1_k} - \theta_{Rot_{k-1}})d_k}{d_{1,0_k}} + \theta_{Rot_{k-1}}, \tag{12}$$

where d_k is the euclidean distance (at time k) between the vehicle rear center axle and the end point of the segment i_k, $d_{1,0_k}$ is the distance between the end point of the segment i_k and the next end point, and θ_{1_k}, θ_{2_k} are the first two orientations of Θ_{OUT} w.r.t. i_k. A one-step graphical representation of Algorithm 1 is represented in Fig. 3b.

Note that in the algorithm, X_{SEG} and Y_{SEG} are vectors containing the waypoints (x_i, y_i) provided by the higher level path planning module, that we rename to allow an eventual resampling, if needed. Moreover, Θ_{SEG} is the vector of orientations of the segments derived from such waypoints, and i_k is the segment i in which the vehicle is located at that time. Finally, i_{MAX} represents a stopping iteration condition, that can take into account the end of the trajectory or a change in the direction of motion, for example.
The proposed algorithm solves the problems on the singularities without limiting the number of segments to take into account, providing two main benefits, i.e., i) maximize the prediction horizon, and ii) allow arbitrary dense path as inputs (see Fig. 4b), i.e., improve the trajectory accuracy.

D. Obstacle avoidance

Another task that is demanded to the MPC-based TP is to react to dynamic objects that could be present in the environment. To handle these situations, the control needs additional information about the objects, which are generally provided by the perception module, namely their location and orientation (x_o, y_o, θ_o), and their lengths and widths (l_o, w_o). To simplify the description and without loss of generality, in the sequel we assume that the free corridor width ($|Y_{min} - Y_{max}|$) is equal to the vehicle's width w.
The general flow-chart that manages the obstacle-avoidance task is reported in Fig. 5. At each time instant, for each surrounding obstacle inside the prediction horizon, the procedure first evaluates if the corners of the object are outside the free corridor, i.e:

$$\left|y_o - w_o/2 - W_{safe_{offset}}\right| \geq w/2, \tag{13}$$

where $W_{safe_{offset}} \in \mathbb{R}$ is a safety margin considered to include possible measurement errors coming from data fusion. If (13) is satisfied, the obstacle is simply neglected since no collision is expected (Fig. 6a). Otherwise the obstacle is considered as "dangerous", and an obstacle avoidance maneuver is necessary. In the last case, it is evaluated whether the obstacle is on the right or left side of the reference trajectory, and since we have to guarantee that the maneuver can be concluded correctly, it is also considered a maximum acceptable lateral error $MAX_{Lat_{err}}$ (that depends on the environment [1]) from the original planned path.

If this maximum lateral error is achieved (Fig. 6c) a "Safe Stop" is requested to the longitudinal MPC through $\xi_{obstacle}$,

[1]For example, $MAX_{Lat_{err}}$ can be decreased if the vehicle is close to the end point of the maneuver or from a change of the motion direction, and increased if the vehicle is travelling along a long straight segment.

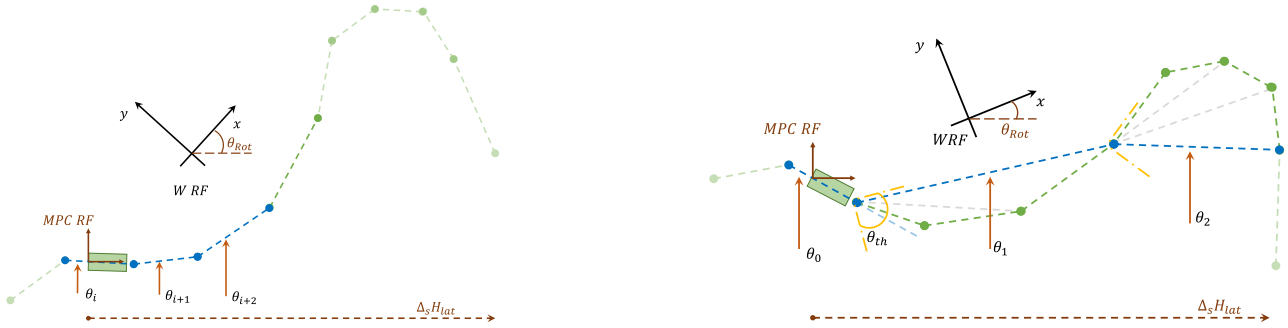


Fig. 3: Proposed rotation angle optimization. Light green points represent the available wayponts; light green dashed lines are the available segments; dark green dashed lines represent the segments exploitable in the MPC horizon; blue dashed lines are the generated segments that establish θ_0, θ_1, θ_2; yellow cone forms the angle θ_{th}.

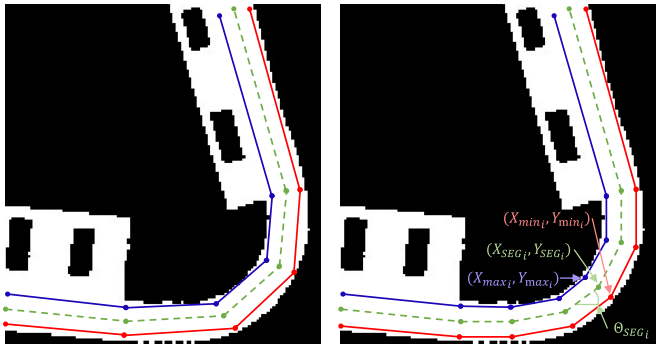


Fig. 4: (a) Traditional free corridor construction. (b) New general free corridor. Path waypoints (in green), left and right free corridor bounds (in blue and red).

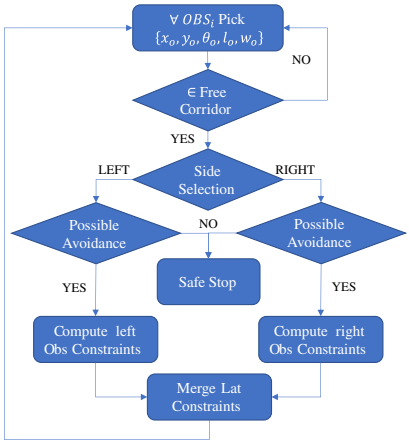


Fig. 5: Flow chart of obstacle avoidance procedure.

which leads to stop the vehicle at a distance $X_{safe_{stop}} \in \mathbb{R}$ (that depends on the environment [1]). In this case both conditions are satisfied:

$$\left|y_o - w_o/2 - W_{safe_{offset}}\right| < w/2,$$

$$\left|y_o - w_o/2 - W_{safe_{offset}} - w/2\right| \geq MAX_{Lat_{err}}.$$

Thus, the stop condition of the longitudinal MPC takes into account both the goal position (ξ_{Goal}), and possible dynamic object constraints ($\xi_{Obstacle}$), as:

$$\xi_{max_k} = min(\xi_{Goal}, \xi_{Obstacle}). \tag{14}$$

As a last case (Fig. 6b), an obstacle constraint is generated around the obstacle. In order to maintain the obstacle avoidance procedure as simple as possible, we design a trapezoidal shape around the obstacle, starting from the two vehicle corners that protrude more on the free corridor, and considering safety offsets $off_{x_{front}} \in \mathbb{R}$ and $off_{x_{back}} \in \mathbb{R}$, on the side $off_y \in \mathbb{R}$ of the obstacle (Fig. 7).

In order to realize the new lateral constraints used by the lateral MPC controller, the trapezoidal shape, Y_{obs_1}, is

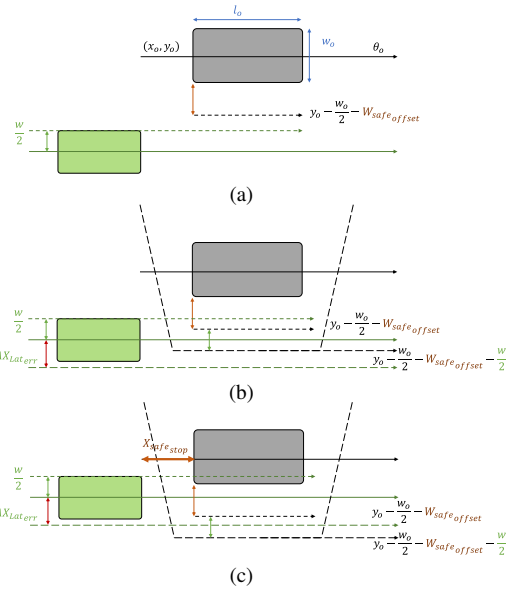


Fig. 6: (a) Obstacle not dangerous. (b) Obstacle dangerous and avoidable. (c) Obstacle dangerous and not avoidable.

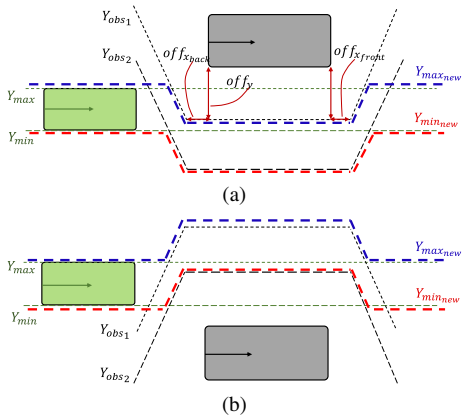
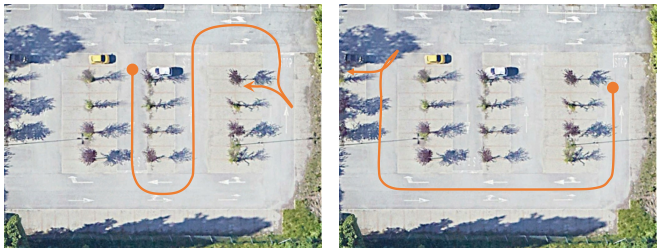


Fig. 7: Obstacle constraints construction on right side (a) and on left side (b).



(a) (b)

Fig. 8: Selected scenarios for low speed maneuvers, tests have been performed at parking area of Centro Ricerche Fiat, Orbassano, TO.

duplicated and translated (Y_{obs_2}) of the same width of the free corridor, and then merged into the original one as:

$$\begin{cases} Y_{max_{new_k}} = min\left(Y_{max_k}, Y_{obs_{1_k}}\right) \\ Y_{min_{new_k}} = min\left(Y_{min_k}, Y_{obs_{2_k}}\right) \end{cases},$$

if the obstacle is on the left side (Fig. 7a), and

$$\begin{cases} Y_{max_{new_k}} = max\left(Y_{max_k}, Y_{obs_{2_k}}\right) \\ Y_{min_{new_k}} = max\left(Y_{min_k}, Y_{obs_{1_k}}\right) \end{cases},$$

if the obstacle is on the right side (Fig. 7b).
It is worth to highlight that the proposed obstacle avoidance procedure takes into account the static information on the environment (HD Map or Binary Gridmap) to check in advance if the constraints generated to escape the obstacle overlap with static objects. Moreover, it is clear that other obstacle avoidance shapes are allowed.

IV. Simulation and Experimental Validation

A. Prototypal vehicle and test setup

The proposed TP module has been validated both in simulation, and on a real Proof of Concept (Poc), namely a Jeep Renegade. Besides the nominal measurement available from the vehicle CAN, additional sensors have been used to design a proper perception module, namely a PwrPak7 Novatel dual antenna Global Navigation Satellite System (GNSS), 12 Valeo Ultra Sound Scan (USS) sensors, 6 Arbe Imaging Radar sensors, and one RoboSense Lidar. As far as the actuators are concerned, the vehicle is equipped with a TRW/ZF column EPS, controlled using the Parking Assistance Module (PAM) interface from the vehicle C-CAN network. This system, which usually assists the driver in steering through an electric motor torque, has been modified in order to use it as a mechatronic unit able to autonomously steer the vehicle. Furthermore, the vehicle uses a modified Braking System Module (BSM), which provides a functional channel on the vehicle C-CAN network able to be a gateway of acceleration/deceleration to the powertrain ECU (ECM) and the brakes.

The proposed TP algorithm has been first validated via simulation using IPG CarMaker in a Matlab/Simulink environment. Afterwards, the same scenarios have been tested on the real vehicle equipped with a dSPACE MicroAutobox II.

Fig. 9: Jeep Renegade prototypal vehicle.

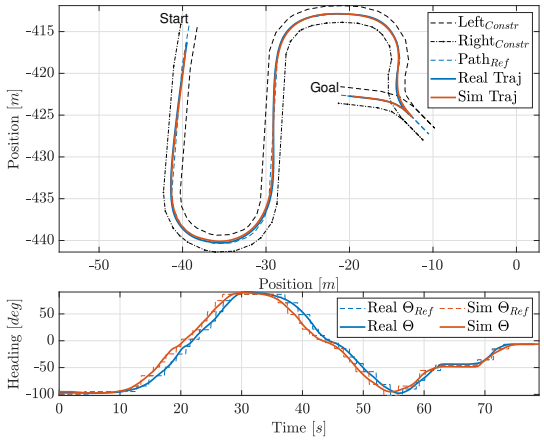


Fig. 10: MIL and in-vehicle lateral states in ex. 4.1.

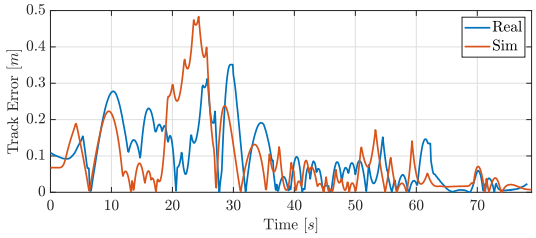


Fig. 11: MIL and in-vehicle tracking error in ex. 4.1.

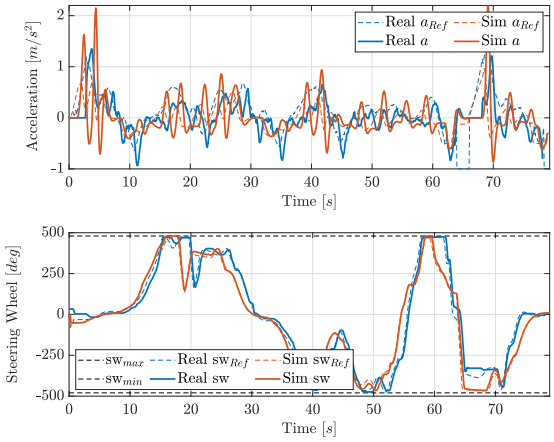


Fig. 12: MIL and in-vehicle control inputs in ex. 4.1.

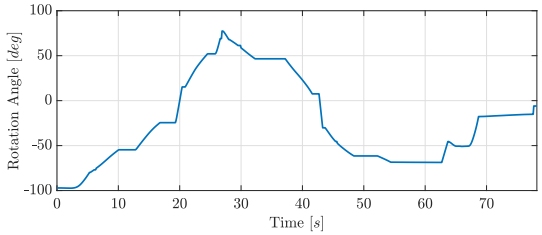


Fig. 13: Rotation angle of MPC Reference Frame in ex. 4.1.

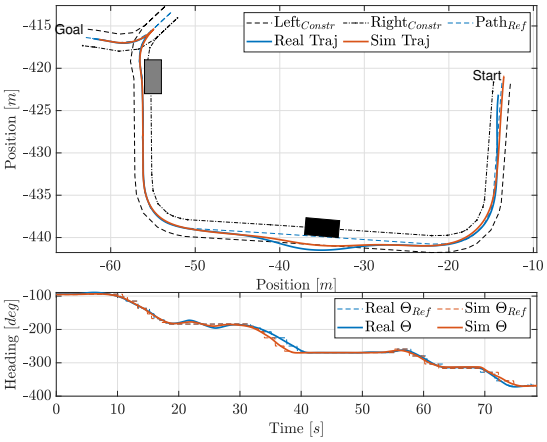


Fig. 14: MIL and in-vehicle lateral states in ex. 4.2.

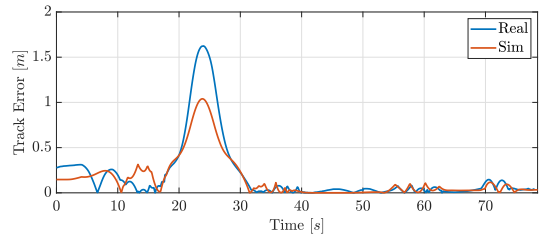


Fig. 15: MIL and in-vehicle tracking error in ex. 4.2.

The solvers used for the MPC problems rely on CVXGEN Code Generation for Convex Optimization [23] for real time execution. Both simulations and in-vehicle tests have been performed selecting a prediction horizon $H_{lat} = 60$ steps, and step widths $\Delta_s = 0.5m$ for the lateral MPC, and $H_{long} = 40$ steps, $\Delta_t = 100ms$ for the longitudinal one. The chosen prediction horizon is a reasonable choice to represent a quasi-infinite horizon for the vehicle kinematics optimization (minimum between $30m$ and $4sec$). Particularly to the selected scenarios, two different home-zone maneuvers have been considered (Fig. 8).

Example 4.1: The first scenario (Fig. 8a) represents a nominal search and park maneuver of an automated valet parking function. It consists of two narrow U-turns and a multi-maneuver. In particular, the two MPCs use variable weights dependent on the distance to the goal position to improve passenger comfort, at the cost of worsening the tracking error when the vehicle is far from the end point of the maneuver. Fig. 10 shows the comparison between the MIL and the vehicle states, in terms of $x - y$ position, and heading angle. From the figure, one can appreciate the reliability of the designed MIL system (in red) which accurately replicates the real vehicle behaviour (in blue), while following the reference path provided by the higher level motion planning module, and reaching the desired parking spot. This behaviour is also confirmed in Fig. 11, reporting the tracking error w.r.t. the nominal path. Indeed, the vehicle is able to solve the task with a maximum lateral error of about 0.3 m, confirming the effectiveness of the proposed MPC-based approach. Moreover, from the figure, one can appreciate that the error shrinks to zero toward the end of the trajectory, meaning that the MPC weights are designed to take into account passenger comfort during the maneuver, but constrain the MPC to very low lateral errors when approaching the goal position, since the methodology is designed to work in narrow passages, and in these applications the precision of the controller is very important. Fig. 13 reports the rotation angle of the MPC computed from Algorithm 1, and Fig. 12 displays the control signals provided from the MPC, appreciating a clear similarity between the MIL and the PoC behaviours.

Example 4.2: The second scenario (Fig. 8b) represents an automated valet parking function with the presence of obstacles along the path. In particular, during the maneuver, the ego vehicle falls into two obstacle avoidance procedures, the first one with an overtaking on the left, the second generating a stop request until the obstacle in front moves away (Fig. 14). This experiment highlights the repeatability and reliability of the proposed TP algorithm. Indeed, very similar MIL and real vehicle behaviours are confirmed, as well as the tracking precision. From Fig. 15, one can appreciate that the maximum lateral error is below 0.3 m, although there is a time window ($t \in [20-30]$ sec) in which the error is about 1.5 m. This case, however, is related to an obstacle avoidance procedure. Indeed, since there is a stand still vehicle on the right of the ego vehicle, it gets away from the reference path to overtake it and continue its maneuver. The same behaviour is repeated in the neighborhood of the goal position (Fig. 14). In such a situation, however, the ego vehicle raises a safe stop request to the actuators since the

dynamic obstacle is near a change of the motion direction: in this case an avoidance of the obstacle could lead the ego vehicle too far from the reference path in the surrounding of the goal position, which could compromise the proper completion of the maneuver. Once the obstacle proceeds with its own motion, the ego vehicle continues the maneuver, and correctly stops the car in the available spot with a near-zero position error.

V. Conclusion

This paper proposed a real-time trajectory planner based on MPC for low speed maneuvers. The methodology is designed for home-zone scenarios, narrow passages, drop-off areas, where the tracking accuracy is crucial to assure effective parking maneuvers. Moreover, a dynamic obstacle avoidance and narrow passages control are designed in a scalable way through continuous updates of the optimization constraints in the MPC problem. The performance of the proposed methodology are evaluated with a V-cycle model-based approach, through both MIL simulations, and real prototype in-vehicle experiments.

References

[1] J. E. Naranjo, M. Clavijo, F. Jiménez, O. Gómez, J. L. Rivera, and M. Anguita, "Autonomous vehicle for surveillance missions in off-road environment," in *2016 IEEE Intelligent Vehicles Symposium (IV)*, 2016, pp. 98–103.

[2] H. Flamig, *Autonomous Vehicles and Autonomous Driving in Freight Transport*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 365–385.

[3] R. Keicher and H. Seufert, "Automatic guidance for agricultural vehicles in europe," *Computers and Electronics in Agriculture*, vol. 25, no. 1, pp. 169–194, 2000.

[4] J. V. Frasch, T. Kraus, W. Saeys, and M. Diehl, "Moving horizon observation for autonomous operation of agricultural vehicles," in *2013 European Control Conference (ECC)*, 2013, pp. 4148–4153.

[5] C. Iclodean, N. Cordoş, and B. O. Varga, "Autonomous shuttle bus for public transportation: A review," *Energies*, vol. 13, no. 11, pp. 2917–, 2020.

[6] 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 Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[7] J. Wit, C. D. Crane III, and D. Armstrong, "Autonomous ground vehicle path tracking," *Journal of Robotic Systems*, vol. 21, no. 8, pp. 439–449, 2004.

[8] B. d'Andréa Novel, G. Campion, and G. Bastin, "Control of nonholonomic wheeled mobile robots by state feedback linearization," *The International Journal of Robotics Research*, vol. 14, no. 6, pp. 543–559, 1995.

[9] W. Farag, "Complex trajectory tracking using pid control for autonomous driving," *International Journal of Intelligent Transportation Systems Research*, vol. 18, 09 2019.

[10] F. Lin, Z. Lin, and X. Qiu, "Lqr controller for car-like robot," in *2016 35th Chinese Control Conference (CCC)*, 2016, pp. 2515–2518.

[11] M. Rokonuzzaman, N. Mohajer, S. Nahavandi, and S. Mohamed, "Model predictive control with learned vehicle dynamics for autonomous vehicle path tracking," *IEEE Access*, vol. 9, pp. 128 233–128 249, 2021.

[12] E. Kim, J. Kim, and M. Sunwoo, "Model predictive control strategy for smooth path tracking of autonomous vehicles with steering actuator dynamics," *International Journal of Automotive Technology*, vol. 15, no. 7, pp. 1155–1164, nov 2014.

[13] J. Ji, A. Khajepour, W. W. Melek, and Y. Huang, "Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 2, pp. 952–964, 2017.

[14] J.-H. Jhang and F. Lian, "An autonomous parking system of optimally integrating bidirectional rapidly-exploring random trees* and parking-oriented model predictive control," *IEEE Access*, vol. 8, pp. 163 502–163 523, 2020.

[15] T. Keviczky, P. Falcone, F. Borrelli, J. Asgari, and D. Hrovat, "Predictive control approach to autonomous vehicle steering," in *2006 American Control Conference*, 2006, pp. 6 pp.–.

[16] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," *2015 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1094–1099, 2015.

[17] P. Polack, F. Altché, B. d'Andréa Novel, and A. de La Fortelle, "Guaranteeing consistency in a motion planning and control architecture using a kinematic bicycle model," *2018 Annual American Control Conference (ACC)*, pp. 3981–3987, 2018.

[18] K. Sakaeta, T. Oda, K. Nonaka, and K. Sekiguchi, "Model predictive parking control with on-line path generations and multiple switching motions," *2015 IEEE Conference on Control Applications (CCA)*, pp. 804–809, 2015.

[19] G. Borrello, E. Raffone, C. Rei, and M. Fossanetti, "Trajectory planning and vehicle control at low speed for home zone manoeuvres," *IFAC-PapersOnLine*, vol. 53, pp. 15 516–15 523, 2020.

[20] P. Polack, F. Altché, B. d'Andréa Novel, and A. de La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 812–818, 2017.

[21] M. Sampei, "A control strategy for a class of nonholonomic systems - time-state control form and its application," in *Proceedings of 1994 33rd IEEE Conference on Decision and Control*, vol. 2, 1994, pp. 1120–1121 vol.2.

[22] K. Oyama and K. Nonaka, "Model predictive parking control for nonholonomic vehicles using time-state control form," *2013 European Control Conference (ECC)*, pp. 458–465, 2013.

[23] J. Mattingley and S. P. Boyd, "Cvxgen: a code generator for embedded convex optimization," *Optimization and Engineering*, vol. 13, pp. 1–27, 2012.