Autonomous racing using Linear Parameter Varying-Model Predictive Control (LPV-MPC)

Eugenio Alcalá [a,*], Vicenç Puig [a], Joseba Quevedo [a], Ugo Rosolia [b]

[a] Center of Supervision, Security and Automatic Control, Universitat Politècnica de Catalunya (UPC), Rambla Sant Nebridi, 10, 08222, Terrassa, Spain
[b] Department of Mechanical Engineering, University of California, Berkeley, CA 94701, USA

ARTICLE INFO

Keywords:
Autonomous vehicle
MPC
LPV
Autonomous racing
Self-driving

ABSTRACT

This article presents an innovative control approach for autonomous racing vehicles. Linear Parameter Varying (LPV) theory is used to model the dynamics of the vehicle and implement an LPV-Model Predictive Controller (LPV-MPC) that can be computed online with reduced computational cost. The optimal time problem is solved by an optimal off-line trajectory planner that calculates the best trajectory under the constraints of the circuit. An identification of the system model based on optimization is also carried out. The planning and control scheme is validated in simulation and experimentally in a real platform where the effectiveness of the proposed LPV-MPC is demonstrated.

1. Introduction

In the last few years, we have witnessed a rapid advance in the area of autonomous driving. Currently, we can find different advanced driver assistance systems (ADAS) in commercial vehicles such as cruise control or lane keeping which are based on classic control strategies. While these systems work very well for the mentioned applications, new challenges appear on the horizon which need more sophisticated control techniques.

We have recently seen solutions to the problem of path tracking in Brown, Funke, Erlien, and Gerdes (2017) and Laurense, Goh, and Gerdes (2017). We also find strategies in Li, Li, Li, Zhu and Dai (2017) and Li, Sun, Cao, Liu and He (2017) for the resolution of a still more complex problem such as trajectory tracking. In this paper, we tackle a more challenging task: autonomous racing driving.

Autonomous racing has generated attention in the last years in the area of robotics, Brunner, Rosolia, Gonzales, and Borrelli (2017), Caporale et al. (2018), Liniger, Domahidi, and Morari (2015), Shin and McKay (1985) and Verschueren, Zanon, Quirynen, and Diehl (2016). The main objective is to make the lap in the shortest possible time while keeping a smooth driving behavior. However, this is not as simple as it seems to be. Trying to minimize the time in a circuit implies going as fast as possible without exceeding the limits of maximum overall acceleration and thus avoid slipping. This implies knowing the physical limits of the car which is sometimes difficult to achieve. Thus, this involve working sometimes outside the vehicle comfort zone, which adds complexity to the control problem. Ni and Hu (2017) describes the difficulty of operating the vehicle at driving limits and present their results for a Formula race car. Today there are several motion control techniques applied to autonomous driving with a quite satisfactory result in practice. Particularly, among all these control strategies, Model Predictive Control (MPC) is one of the most promising since it allows to deal with states and input constraints as well as to accurately predict the behavior of the vehicle. Consequently, MPC has been generating more attention among researchers in this area. In Law, Dalal, and Shearrow (2018), a MPC approach for controlling front steering of an autonomous vehicle is presented using a successive online linearization of a non-linear vehicle model. In Verschueren, De Bruyne, Zanon, Frasch, and Diehl (2014), a real-time MPC control scheme was presented that solve the racing problem and test it in miniature race cars. Also in Rosolia, Carvalho, and Borrelli (2017), Learning MPC was introduced to solve the racing problem.

In this work, we propose the Linear Parameter Varying-Model Predictive Control (LPV-MPC) approach as a novel option to solve the driving control problem. In Bujarbaruah, Zhang, Tseng, and Borrelli (2018) an explicit version of this idea is introduced for lateral control. The authors perform a comparison against the corresponding non-linear MPC version showing the promising results of the LPV technique. The advantage of the LPV approach is that the non-linear model can be expressed as a combination of linear models that depend on some scheduling variables without using linearization (Sename, Gaspar, & Bokor, 2013).

Time-optimal problems have received great interest in recent years. A trajectory planning approach for robot manipulators based on minimum-time optimization is presented in Liu, Lai, and Wu (2013).

* Corresponding author.
E-mail addresses: eugenio.alcala@upc.edu (E. Alcalá), vicenc.puig@upc.edu (V. Puig), joseba.quevedo@upc.edu (J. Quevedo), ugo.rosolia@berkeley.edu (U. Rosolia).

Fig. 1. Vehicle used for experimental tests (BARC).

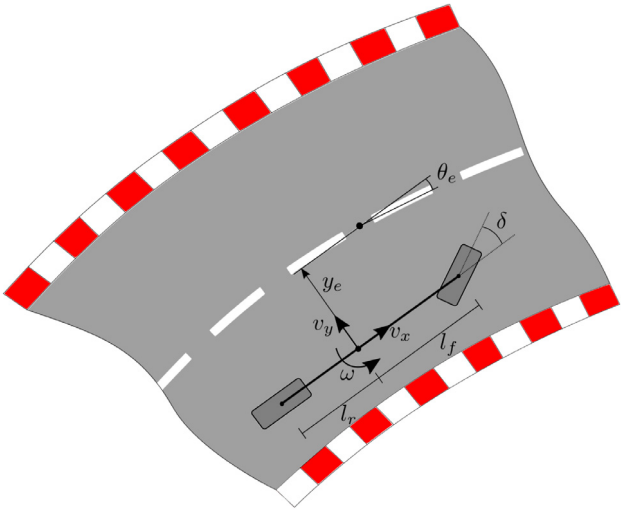

Fig. 2. Overview of the bicycle vehicle model and its variables with respect to the road.

Consequently, the huge interest to rise the autonomous driving technology to the highest level makes the time-optimal control of especial interest in the field of car racing. Some works solve this problem using an optimal control technique, as e.g. Rosolia et al. (2017) and Verschueren et al. (2014). Unlike them, we propose a planning layer where the complete optimal time trajectory is calculated, thus allowing to reduce the complexity inside the motion control layer.

The contribution of this paper is twofold. First, we solve the planning task using an offline non-linear Model Predictive Planner (NLMPP) taking into account the vehicle dynamics and its limits for racing. Then, we compute the online LPV-MPC strategy for tracking the racing trajectory which enables real-time embedded systems computation.

The paper is structured as follows: The testing vehicle and modeling section presents the experimental platform describing the different types of modeling used for planning and control. In the NLMP Planner formulation section, the optimal planning strategy is stated as an offline algorithm for racing vehicles. The LPV-MPC formulation section presents the online control algorithm and the core of this work. The system identification section shows the experimental process performed to adjust the model to the real vehicle. The racing results section presents and discusses the results obtained in simulation and experimentally. The last section concludes the paper.

2. Testing vehicle & modeling

The Berkeley Autonomous Race Car (BARC[1]) is a development platform for autonomous driving. It is a 1/10 Scale RC vehicle (see Fig. 1) that has been modified to operate autonomously. This vehicle includes a basic net of sensors for performing localization. A fusion of IMU, encoders and indoor GPS data is made by a Kalman filter in order to achieve an accurate localization while testing. An Odroid XU4 is used to run the Robotic Operating System (ROS) and the control and planning algorithms.

2.1. Vehicle modeling

When a car is running in a race it is subjected to lateral and longitudinal acceleration levels much higher than those produced in normal driving situations. It is expected that the vehicle works within its dynamic limits obtaining then an optimal response in terms of traceability and therefore in terms of lap time. However, working at these limits is complicated if a vehicle model, that faithfully represents all its dynamics, is not used. The dynamic model of the vehicle used in this work is a standard bicycle model obtained from Schramm, Hiller, and Bardini (2014)

$$\begin{aligned} \dot{v}_x &= a + \frac{-F_{yf}\sin\delta - \mu m g}{m} + \omega v_y \\ \dot{v}_y &= \frac{F_{yf}\cos\delta + F_{yr}}{m} - \omega v_x \\ \dot{\omega} &= \frac{F_{yf} l_f \cos\delta - F_{yr} l_r}{I}, \end{aligned} \tag{1}$$

where lateral tire forces, produced in front and rear wheels, respectively, are computed as

$$\begin{aligned} F_{yf} &= C_f\left(\delta - \frac{v_y}{v_x+\epsilon} - \frac{l_f\omega}{v_x+\epsilon}\right) \\ F_{yr} &= C_r\left(-\frac{v_y}{v_x+\epsilon} + \frac{l_r\omega}{v_x+\epsilon}\right), \end{aligned} \tag{2}$$

where v_x, v_y and ω are the body frame velocities, linear in x, linear in y and angular, respectively. δ and a are the control actions, steering angle and longitudinal acceleration, respectively. Variables C_f and C_r represent the tire stiffness coefficients for the front and rear wheels. m and I represent the vehicle mass and inertia and l_f and l_r are the distances from the center of gravity to the front and rear wheel axes, respectively. μ and g are the friction coefficient and the gravity value, respectively. Note that, the dynamic states (v_x, v_y and ω) are estimated by means of the fusion with a Kalman filter of the respective sensors measurements (see Fig. 3). Parameter ϵ is a small value properly introduced to avoid the singularity given at $v_x = 0$.

Regarding the kinematic representation, we have used two different error-based models for two different tasks: control and planning. A time-dependent representation is used for control while a spatial representation is employed for planning purposes.

2.1.1. Modeling for control

For control purposes, an error-based model represented in the curvilinear coordinate system is obtained as the error of heading angle and lateral position with respect to the center line of the track (Verschueren

[1] http://www.barc-project.com/.

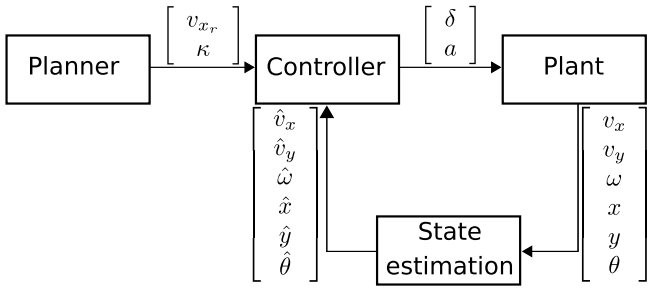

Fig. 3. Schematical view of the experimental set up.

et al., 2014). This model considers the following equations

$$\begin{aligned}\dot{y}_e &= v_x \sin\theta_e + v_y \cos\theta_e \\ \dot{\theta}_e &= \omega - \frac{v_x \cos\theta_e - v_y \sin\theta_e}{1 - y_e\kappa}\kappa \\ \dot{s} &= \frac{v_x \cos\theta_e - v_y \sin\theta_e}{1 - y_e\kappa} \ ,\end{aligned} \tag{3}$$

where θ_e and y_e are the orientation error and the lateral position error in the curvilinear frame, see Fig. 2. Variable s is the projected traveled distance along the track center line. κ represents the road curvature being function of s and is assumed to be known. The complete model for control is the union of the set of equations in (1) and (3).

2.1.2. Modeling for planning

Model (3) is convenient for control purposes due to its time-domain representation. However, optimal time racing focuses on minimizing lap time and thus, the time variable needs to be converted into an optimization variable. For that reason, we propose using the model reparametrization from Gao et al. (2012). Such a new formulation is based on the space domain and the road curvature is given as a function of s hence, the trajectory optimization under the space-based domain is then feasible. On the contrary, using a time-based domain would not be possible to solve a predictive optimization problem like the one presented in next section because time would not appear explicitly as an optimization variable.

Denoting $x_c = \begin{bmatrix} y_e & \theta_e & s \end{bmatrix}$ as the state vector of the control model (3), then, a new state vector $\tilde{x_c} = \begin{bmatrix} \tilde{y_e} & \tilde{\theta_e} & \tilde{t} \end{bmatrix}$ is obtained by applying ($\tilde{(\cdot)}$ denotes a variable in the space domain)

$$\tilde{x_c'} = \frac{dx_c}{ds} = \frac{dx_c}{dt}\frac{dt}{ds} = \dot{x_c}\frac{1}{\dot{s}} = x_c'\dot{\tilde{t}} \ , \tag{4}$$

leading consequently to the following model equations

$$\begin{aligned}\tilde{y_e}' &= \frac{v_x \sin\theta_e + v_y \cos\theta_e}{\dot{s}} \\ \tilde{\theta_e}' &= \frac{\omega}{\dot{s}} - \kappa \\ \tilde{t}' &= \frac{1}{\dot{s}} \ .\end{aligned} \tag{5}$$

Note the time ($\tilde{t}$) is now in the space domain and hence, is a function of s. The complete model used for solving the trajectory planning task is presented in the Appendix A.2.

3. NLMP planner formulation

The trajectory planning task achieves higher relevance in the self-driving field when the vehicle performs in racing mode. Planning references must consider the reachable dynamics of the vehicle, thus providing a traceable set of variables to the motion control strategy.

The main function of this racing planning task is to find a trajectory within the circuit that minimizes the total lap time and provides relevant information to the motion control. To do so, the time dependent error model (3) is reformulated to be space dependent (5). In addition, the dynamic model (1)–(2) is also rewritten to be in the same space frame (see Appendix A.2). Then, the following non-linear optimization problem is solved using a constant sampling space (ds)

$$\underset{\Delta\mathcal{U},\tilde{\mathcal{X}}}{\text{minimize}} \quad \mathrm{J} = \sum_{i=0}^{N-1}\left(\tilde{x}_i^T Q\tilde{x}_i + \Delta u_i R\Delta u_i\right) + \tilde{x}_N^T P\tilde{x}_N$$

s. t.

$$\begin{aligned} &\tilde{x}_{i+1} = \tilde{x}_i + f(\tilde{x}_i, u_i, \kappa_i)ds, \quad i = 0, \dots, N-1 \\ &u_i = u_{i-1} + \Delta u_i \ , \quad i = 0, \dots, N-1 \\ &N = L_{track}/ds \\ &\tilde{y_e} \in [\underline{\tilde{y_e}}, \overline{\tilde{y_e}}] \\ &\left(\frac{\tilde{v}_{x_{i+1}} - \tilde{v}_{x_i}}{\overline{\Delta v_x}}\right)^2 + \left(\frac{\tilde{v}_{y_{i+1}} - \tilde{v}_{y_i}}{\overline{\Delta v_y}}\right)^2 - 1 \le 0 \ , \end{aligned} \tag{6}$$

where the decision vector variables are

$$\begin{aligned}\tilde{\mathcal{X}} &= (\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_N) \\ \Delta\mathcal{U} &= (\Delta u_1, \Delta u_2, \dots, \Delta u_N) \ .\end{aligned} \tag{7}$$

Note that, with the aim of smoothing the performance effort and providing a null tracking error in steady state, the variation of the input model variables (Δu_i) is used as an optimization variable. State and control input vectors are defined as $\tilde{x} = \begin{bmatrix} \tilde{v_x} & \tilde{v_y} & \tilde{\omega} & \tilde{y_e} & \tilde{\theta_e} & \tilde{t} \end{bmatrix}^T$ and $u = \begin{bmatrix} \delta & a \end{bmatrix}^T$, respectively. The continuous-time non-linear model $f(\tilde{x}_i, u_i, \kappa_i)$ is presented in the Appendix A.2. N is the prediction horizon and L_{track} is the total length of the circuit. The last inequality constraint in (6) bounds the longitudinal and lateral vehicle acceleration. This constraint defines an ellipse limited by $\overline{\Delta v_x}$ and $\overline{\Delta v_y}$ which are found experimentally.

Finally, before providing the obtained trajectory references to the motion control strategy, a time-based interpolation is made in order to convert the space-based variables into time-based variables. Before applying the linear interpolation, we have the space-based state vector ($\tilde{x}$) obtained from the optimization problem (6) and among them the time depending on the space ($\tilde{t}$). Now, we define the query time points as a finer sampling in the range of $\tilde{t}$. Then, the next interpolation is performed at those query points (t_i) to obtain the vehicle state variables as a function of time (x_i)

$$x_i = \tilde{x}_i + \frac{\tilde{x}_{i+1}(t_i - \tilde{t}_i) - \tilde{x}_i(t_i - \tilde{t}_i)}{\tilde{t}_{i+1} - \tilde{t}_i} \ , \ i = 0, \dots, N \ , \tag{8}$$

where x_i is the new time-dependent variable and t_i is the accumulative sampling time. Note that, this interpolation procedure is carried out for variables $\tilde{v_x}$, $\tilde{v_y}$, $\tilde{\omega}$, $\tilde{y_e}$ and $\tilde{\theta_e}$, but not for $\tilde{t}_i$.

4. LPV-MPC formulation

In this section, we present a novel formulation for the MPC technique using the LPV representation of the non-linear vehicle model. A LPV system is a Linear Time Variant (LTV) plant of finite dimension whose state space matrices are fixed functions of a vector of measurable scheduling variables. When the MPC technique performs the prediction of future vehicle states, it uses the LPV model. This imply that, at every time instant, an instantiation of the non-linear vehicle model computed with a known scheduling vector is required. Such a vector can be given by the trajectory planner or by the prediction made in the previous iteration. The LPV matrices, i.e. $A(\zeta)$ and $B(\zeta)$, of the control model (1)–(3) are obtained using the non-linear embedding approach (Rotondo, Puig, Nejjari, & Witczak, 2015) (see Appendix A.1). The vector of scheduling variables is $\zeta = \begin{bmatrix} v_x & v_y & \theta_e & \kappa & y_e & \delta \end{bmatrix}$.

This allows to formulate the MPC problem as a quadratic optimization problem that is solved at each time k to determine the control

actions considering that the values of x_k and u_{k-1} are known

$$\begin{aligned}\min_{\Delta U_k} \quad & \mathrm{J_k} = \sum_{i=0}^{H_p-1} \Big((r_{k+i} - x_{k+i})^T Q (r_{k+i} - x_{k+i}) \\ & + \Delta u_{k+i} R \Delta u_{k+i} \Big) + x_{k+H_p}^T Q x_{k+H_p} \\ \text{s.t.} \quad & \\ & x_{k+i+1} = x_{k+i} + \big(A(\zeta_{k+i}) x_{k+i} + B(\zeta_{k+i}) u_{k+i}\big) dt \\ & u_{k+i} = u_{k+i-1} + \Delta u_{k+i} \\ & \Delta U_k \in \Delta \Pi \\ & U_k \in \Pi \\ & y_e \in [\overline{y_e}, \underline{y_e}] \\ & x_{k+0} = \hat{x}_k \ ,\end{aligned} \tag{9}$$

where $x = \begin{bmatrix} v_x & v_y & \omega & \theta_e & s & y_e \end{bmatrix}^T$ is the state vector, $\hat{x}$ is the estimated state vector, $r = \begin{bmatrix} v_{x_r} & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T$ is the reference vector provided by the trajectory planner, $u = \begin{bmatrix} \delta & a \end{bmatrix}^T$ is the control input vector and H_p is the control prediction horizon. The tuning matrices $Q \in \mathbb{R}^{6\times6}$ and $R \in \mathbb{R}^{2\times2}$, are semi-positive definite in order to obtain a convex cost function. The time discretization is carried out using Euler approach and the constant sampling time dt. Constant sets Π and $\Delta\Pi$ constraint the model inputs and their variations, respectively.

5. System identification

In this section, we present the identification methodology used for adjusting the parameters of the vehicle dynamic model. The parameter estimation procedure considers the non-linear model (1)–(2) for the vehicle dynamics and the goal is to identify the tire stiffness coefficients C_f and C_r using a least-squares approach. The rest of parameters are assumed to be known for the particular vehicle and it is assumed to have at disposal M data samples. The identification procedure determines the unknown parameters that provides the minimum of the following objective function

$$\begin{aligned}\underset{C_f, C_r}{\text{minimize}} \quad & \mathrm{J_k} = \sum_{i=0}^{M} \Big(\frac{1}{\overline{v_x}} (v_{x_k} - \hat{v}_{x_k})^2 + \frac{1}{\overline{v_y}} (v_{y_k} - \hat{v}_{y_k})^2 \\ & + \frac{1}{\overline{\omega}} (\omega_k - \hat{\omega}_k)^2 \Big) \ ,\end{aligned} \tag{10}$$

where $\hat{v}_{x_k}$, $\hat{v}_{y_k}$ and $\hat{\omega}_k$ are the one-step predictions based on the non-linear equations (1)–(2) after the corresponding discretization in time and $\overline{v_x}$, $\overline{v_y}$ and $\overline{\omega}$ are their maximum dynamic values.

Note that since the computational cost of this optimization-based approach is high to be run in real-time, this is solved offline using IPOPT non-linear optimization solver (Wächter & Biegler, 2006).

6. Racing results

The process of evaluating the planning and control strategies for racing is by first simulating the whole autonomous driving system and then testing it in a real framework. To do so, we have proposed a circuit where the objective is to minimize the lap time while fulfilling the road constraints. First, the off-line planning is obtained by solving the problem (8) using Matlab, Yalmip (https://yalmip.github.io/download/) and IPOPT non-linear optimization solver (Wächter & Biegler, 2006). The tuning is basically focused on minimizing total lap time. However, in order to achieve numerical reliability, very small weights are used for the rest of variables in the cost function. The diagonal values of the weighting matrices, found to obtain the best trade-off among the different objectives, are

$$\begin{aligned} & Q = [10^{-8} \quad 10^{-8} \quad 10^{-3} \quad 10^{-5} \quad 10^{-8} \quad 10^{-8}], \\ & R = [0.05 \quad 0.01], \\ & P = [10^{-8} \quad 10^{-8} \quad 10^{-8} \quad 10^{-8} \quad 10^{-8} \quad 1], \\ & ds = 0.1m \ , \quad L_{track} = 18m. \end{aligned} \tag{11}$$

The solution of this optimization problem will provide reference variables for the control loop. Future work will address online computation of this planning strategy. Then, at every sampling period, i.e. 30 Hz, the control problem (9) is solved to find the appropriate control actions (δ and a).

The LPV-MPC algorithm is programmed in Python 2.7 language over the ROS (Robotic Operating System) platform and solved in real time employing the Operator Splitting Quadratic Program (OSQP) solver (Stellato, Banjac, Goulart, Bemporad, & Boyd, 2018) running on a DELL inspiron 15 (Intel core i7-8550U CPU @ 1.80 GHzx8). The tuning aims to minimize the velocity and lateral errors while computing smooth control actions. The diagonal terms of the weighting matrices in the cost function and prediction horizon of (9), found by iterative tuning until the desired performance is achieved, are

$$\begin{aligned} & Q = [120 \quad 1 \quad 1 \quad 40 \quad 0 \quad 800], \\ & R = [6 \quad 2], \\ & N = 20. \end{aligned} \tag{12}$$

Before validating the presented algorithms in an experimental way, they have been tested in simulation. In addition, the dynamic system has been properly identified using the identification method presented in (10). The next subsections present simulation and experimental results from a control perspective.

6.1. Simulation test

All simulations are carried out in the ROS platform. We use a high fidelity vehicle model for simulation (see Appendix A.3). This model considers a more precise lateral tire force formulation using the simplified Magic Formula (Pacejka, 2005) for modeling the non-linear relationship between front and rear slip angles and lateral tire forces. The parameters b, c and d define the shape of the semi-empirical curve. Also, a more accurate computation of the tire slip angles is given. The three algorithms shown in Fig. 3 are executed every 30 ms. First, the controller instantiates the LPV model matrices for the prediction stage. Then, the optimal problem is solved using the current state variables and the references coming from the planner. Once, the optimal control actions are computed they are applied to the simulated vehicle. As a consequence, the vehicle changes its state and this is measured by the sensor net. The sensors are simulated to be realistic by adding Gaussian noise. Finally, the state estimator algorithm deals with the current available measurements to obtain a precise and complete state vector.

Fig. 4 depicts the reference and the response longitudinal velocity profiles for three laps, i.e. the acceleration lap and two consecutive laps.

It can be seen that the controller has some troubles when the vehicle is accelerating but it works acceptably after 27 s. Such problems are related to the lack of modeling of the traction motor resulting in the controller unable to follow the speed reference perfectly. Fig. 5 shows the performance of the controller in terms of errors. Note that, the MPC is able to make the velocity error to converge to zero but not the longitudinal position error since x_e is not minimized in the cost function of (9). In addition, this shows how controlling the lateral error (y_e) in section B is difficult for the controller and the lateral error presents a slow convergence due to the difficulty of the racing scenario and the fast changing reference.

In Fig. 6, two simulated racing laps are depicted where circles represent the real vehicle position and dashed lines the planning references. From this top view, we observe the largest lateral error in section B which coincides with the most difficult section of the track. In the rest of the circuit it is seen that the controller is able to delimit the lateral error to a tighter interval in spite of the complexity of driving in a high acceleration situation.

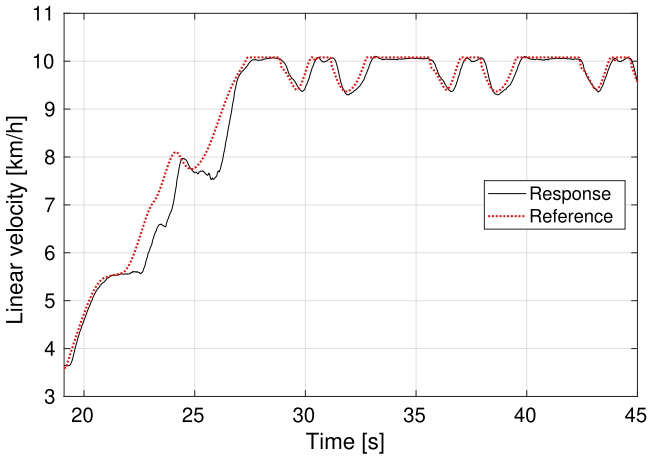

Fig. 4. Linear velocities in simulation. The reference is provided by the NLMP Planner. The response is the result after treating the measured data from the vehicle sensors.

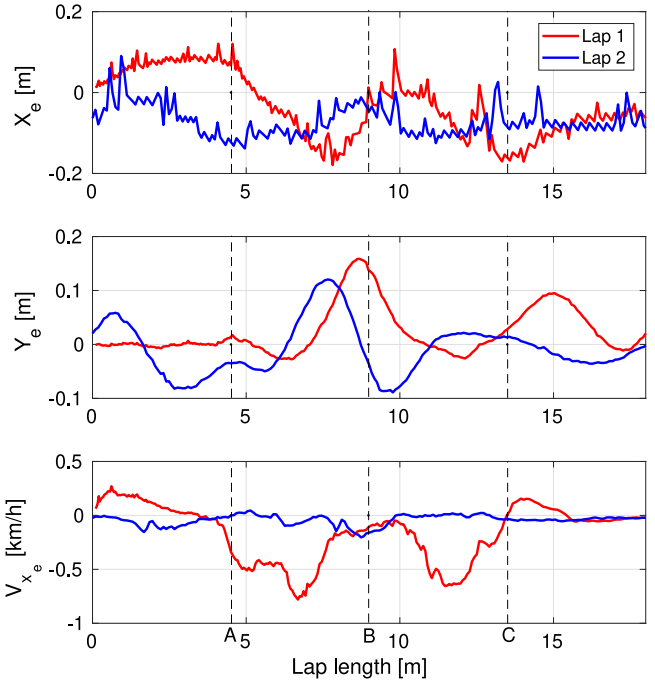

Fig. 5. Errors achieved during two simulated racing laps. A, B and C represent circuit sections.

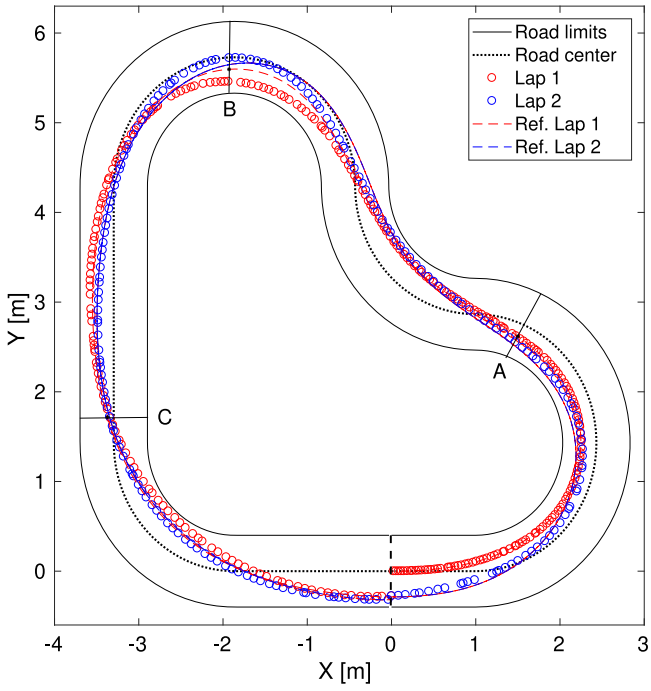

Fig. 6. Two racing laps in simulation controlling the simulation vehicle.

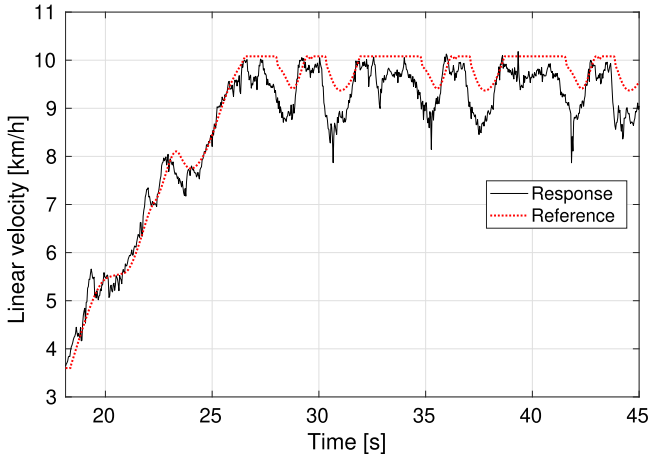

Fig. 7. Linear velocities in experimental test. The reference is provided by the NLMP Planner. The response is the result after treating the measured data from the vehicle sensors.

6.2. Experimental test

Using the same setup than in simulation, we assess the performance of the LPV-MPC technique in an experimental way using the BARC platform (see Fig. 1). The racing planning is the same as in the simulation tests since the track is the same. A resulting video can be watched at https://www.youtube.com/watch?v=MXz9InvoVBw.

The result of the controlled longitudinal velocity for three laps is shown in Fig. 7. It shows a good reference tracking although with a bit of steady state error from $t = 27$ s and up. In spite of the identification performed, it is possible to attribute this error to modeling and estimation errors.

Fig. 8 presents the resulting trajectory during the test. The mean lap time achieved disregarding the first accelerating lap is 6.97 s. Some jumps can be observed along the vehicle way which are totally attributed to issues in the GPS system. The indoor GPS works by using ultrasonic sensors allowing to have up to ~2 cm of error in localization using triangulation. However, the ultrasonic sensors sometimes experience interference by external signals which results in little jumps in localization. These jumps are treated by the Kalman filter, however, when they are very large and continuous it is very difficult to filter them.

The computational time of this approach is one of the most interesting advantages. Fig. 9 shows the elapsed time when computing the LPV-MPC strategy with a mean time of 0.0149 s for a prediction horizon of 20 steps. The peaks that can be seen outside the permitted area (real-time constraint) are due to sudden locations jumps of the indoor-GPS system, causing the optimizer to solve a more complex and therefore more computationally expensive problem. In these particular cases the applied control action corresponds with the one predicted in the previous optimization. Finally, we can perform a comparison against the control strategies presented in Liu et al. (2013) and Rosolia, Zhang, and Borrelli (2019). In both references and this work, the results are obtained after testing under the same conditions, i.e. the

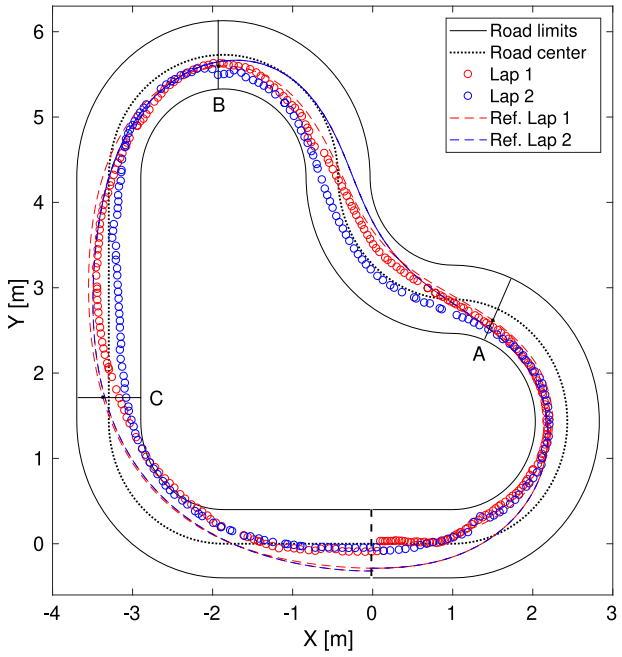

Fig. 8. Two experimental racing laps using the BARC vehicle.

same track and the same vehicle. The resulting comparison shows us a lap time reduction using the proposed method in this work. The best lap time achieved in these works is more than 7 s however, in the presented strategy the mean lap time achieved disregarding the first accelerating lap is 6.97 s. In addition, the average computational time is also improved being in this work 14.9 ms while in the other works is around 30 ms.

7. Conclusions

In this paper, we propose an LPV-MPC strategy as a novel approach to solve autonomous driving control problems under realistic conditions in real-time. In addition, an optimal planning algorithm is presented that is responsible for computing the optimal trajectory along the time. For a good control performance, an offline identification of unknown vehicle coefficients is carried out. The strategies are tested in simulation and in real experiments that show potential and similar results among them, thus strengthening the task of the simulator. In the real test, we showed the contribution of the controller which is able to solve a 20 steps prediction at 33 Hz and thus follow the trajectory although with a certain error due to the non-modeled dynamics (see https://www.youtube.com/watch?v=MXz9InvoVBw). The main disadvantage of this strategy is the initialization due to the need to instantiate the LPV model.

For future work, the study of an online planning strategy using the LPV-MPC approach and able to deal with multiple obstacles and vehicles is being performed. Furthermore, the study of more accurate vehicle formulations such as a four wheels model considering roll and pitch motions are open for future research.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The experimental tests were developed in the MPC LAB (http://www.mpc.berkeley.edu/) at the Department of Mechanical Engineering, University of California, Berkeley, CA 94701, USA. Project website: http://www.barc-project.com/. This work has been funded by the Spanish Ministry of Economy and Competitiveness (MINECO) and FEDER through the projects SCAV (Ref. DPI2017-88403-R) and HARCRICS (Ref. DPI2014-58104-R). The author is supported by a FI AGAUR grant (Ref. 2017 FI B00433).

Appendix

A.1. Continuous-time LPV model

$$A(v_x, v_y, \theta_e, \kappa, y_e, \delta) = \begin{bmatrix} A_{11} & A_{12} & A_{13} & 0 & 0 & 0 \\ 0 & A_{22} & A_{23} & 0 & 0 & 0 \\ 0 & A_{32} & A_{33} & 0 & 0 & 0 \\ A_{41} & 0 & 1 & 0 & 0 & 0 \\ A_{51} & 0 & 0 & 0 & 0 & 0 \\ A_{61} & A_{62} & 0 & 0 & 0 & 0 \end{bmatrix} \tag{13}$$

$$B(\delta) = \begin{bmatrix} -\frac{1}{m}\sin\delta C_f & 1 \\ \frac{1}{m}\cos\delta C_f & 0 \\ \frac{1}{I}\cos\delta C_f l_f & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{14}$$

being

$$\begin{aligned} A_{11} &= \frac{-\mu g}{v_x}\ ,\ A_{12} = \frac{C_f \sin\delta}{m v_x}\ ,\ A_{13} = \frac{C_f l_f \sin\delta}{m v_x} + v_y \\ A_{22} &= -\frac{C_r + C_f\cos\delta}{m v_x}\ ,\ A_{23} = -\frac{C_f l_f \cos\delta - C_r l_r}{m v_x} - v_x \\ A_{32} &= -\frac{C_f l_f \cos\delta - l_r C_r}{I v_x}\ ,\ A_{33} = -\frac{C_f l_f^2 \cos\delta + l_r^2 C_r}{I v_x} \\ A_{41} &= -\frac{v_x\cos\theta_e - v_y\sin\theta_e}{(1-y_e\kappa)v_x}\kappa\ ,\ A_{51} = \frac{v_x\cos\theta_e - v_y\sin\theta_e}{(1-y_e\kappa)v_x} \\ A_{61} &= \sin\theta_e\ ,\ A_{62} = \cos\theta_e \end{aligned} \tag{15}$$

All parameters are properly defined in Table 1.

A.2. Continuous-time model for trajectory planning

For motion planning purposes we use a high fidelity vehicle bicycle model represented in the space domain (see Section 2). This considers the Pacejka "Magic Formula" tire model where the parameters b, c and d define the shape of the non-linear curve. The explicit model equations are given by

$$\dot{\tilde{x}} = \begin{bmatrix} \dot{v}_x \\ \dot{v}_y \\ \dot{\omega} \\ \dot{y}_e \\ \dot{\theta}_e \\ \dot{t} \end{bmatrix} = f(\tilde{x}, u, \kappa) \tag{16}$$

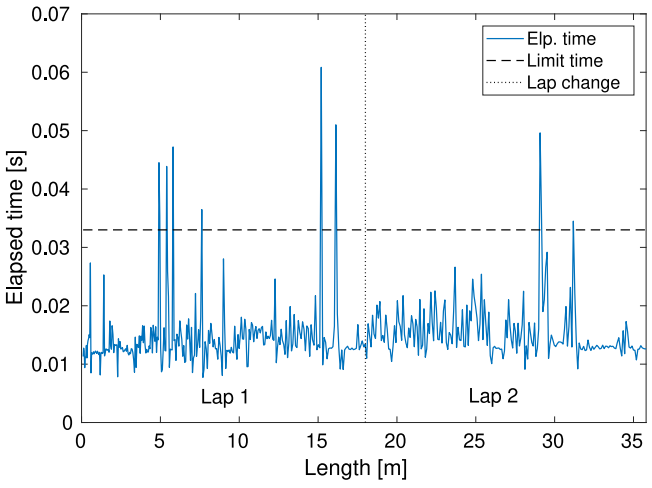

Fig. 9. LPV-MPC computational time during two experimental racing laps.

Table 1
Dynamic model parameters.

Parameter	Value	Parameter	Value
l_f	0.125 m	l_r	0.125 m
m	1.98 kg	I	0.03 kg m^2
C_f	68 $\frac{\text{N}}{\text{rad}}$	C_r	71 $\frac{\text{N}}{\text{rad}}$
d	8.255	c	1.6
b	6.1	μ	0.85

$$= \begin{bmatrix} \frac{a - \frac{\sin\delta d \sin\left(c\tan^{-1}\left(b\delta - b\tan^{-1}\left(\frac{\tilde{v_y}}{\tilde{v_x}} - \frac{l_f\tilde{\omega}}{\tilde{v_x}}\right)\right)\right)}{m} - \mu g + \tilde{\omega}\tilde{v_y}}{\dot{s}} \\ \frac{\frac{d\cos\delta\sin\left(c\tan^{-1}\left(b\delta - b\tan^{-1}\left(\frac{\tilde{v_y}}{\tilde{v_x}} - \frac{l_f\tilde{\omega}}{\tilde{v_x}}\right)\right)\right)}{m} + \frac{d\sin\left(c\tan^{-1}\left(-b\tan^{-1}\left(\frac{\tilde{v_y}}{\tilde{v_x}} + \frac{l_r\tilde{\omega}}{\tilde{v_x}}\right)\right)\right)}{m} - \tilde{\omega}\tilde{v_x}}{\dot{s}} \\ \frac{l_f d\cos\delta\sin\left(c\tan^{-1}\left(b\delta - b\tan^{-1}\left(\frac{\tilde{v_y}}{\tilde{v_x}} - \frac{l_f\tilde{\omega}}{\tilde{v_x}}\right)\right)\right) - l_r d\sin\left(c\tan^{-1}\left(-b\tan^{-1}\left(\frac{\tilde{v_y}}{\tilde{v_x}} + \frac{l_r\tilde{\omega}}{\tilde{v_x}}\right)\right)\right)}{I} \\ \frac{\tilde{v_x}\sin\tilde{\theta}_e + \tilde{v_y}\cos\tilde{\theta}_e}{\dot{s}} \\ \frac{\tilde{\omega}}{\dot{s}} - \kappa \\ \frac{1}{\dot{s}} \end{bmatrix},$$

where

$$\dot{s} = \frac{\tilde{v_x}\cos\tilde{\theta}_e - \tilde{v_y}\sin\tilde{\theta}_e}{1 - \tilde{y}_e\kappa}. \tag{17}$$

All parameters are properly defined in Table 1.

A.3. Continuous-time simulation vehicle

For simulation purposes we use a high fidelity vehicle bicycle model. This considers the simplified "Magic Formula" model for simulating lateral tire forces where the parameters b, c and d define the shape of the curve.

$$\begin{aligned}
\dot{x} &= \cos\theta v_x - \sin\theta v_y \\
\dot{y} &= \sin\theta v_x + \cos\theta v_y \\
\dot{\theta} &= \omega \\
\dot{v}_x &= a - \frac{F_{yF}\sin\delta}{m} - \mu g + \omega v_y \\
\dot{v}_y &= \frac{F_{yF}\cos\delta}{m} + \frac{F_{yR}}{m} - \omega v_x \\
\dot{\omega} &= \frac{F_{yF} l_f \cos\delta - F_{yR} l_r}{I} \\
F_{yF} &= d\sin\left(c\tan^{-1}(b\alpha_f)\right) \\
F_{yR} &= d\sin\left(c\tan^{-1}(b\alpha_r)\right) \\
\alpha_f &= \delta - \tan^{-1}\left(\frac{v_y}{v_x} - \frac{l_f\omega}{v_x}\right) \\
\alpha_r &= -\tan^{-1}\left(\frac{v_y}{v_x} + \frac{l_r\omega}{v_x}\right) \\
F_f &= \mu m g
\end{aligned} \tag{18}$$

All parameters are properly defined in Table 1.

References

Brown, M., Funke, J., Erlien, S., & Gerdes, J. C. (2017). Safe driving envelopes for path tracking in autonomous vehicles. *Control Engineering Practice*, *61*, 307–316.

Brunner, M., Rosolia, U., Gonzales, J., & Borrelli, F. (2017). Repetitive learning model predictive control: An autonomous racing example. In *2017 IEEE 56th annual conference on decision and control (CDC)* (pp. 2545–2550). IEEE.

Bujarbaruah, M., Zhang, X., Tseng, H. E., & Borrelli, F. (2018). Adaptive MPC for autonomous lane keeping. arXiv preprint arXiv:1806.04335.

Caporale, D., Fagiolini, A., Pallottino, L., Settimi, A., Biondo, A., Amerotti, F., et al. (2018). 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.

Gao, Y., Gray, A., Frasch, J. V., Lin, T., Tseng, E., Hedrick, J. K., et al. (2012). Spatial predictive control for agile semi-autonomous ground vehicles. In *Proceedings of the 11th international symposium on advanced vehicle control* (pp. 1–6).

Laurense, V. A., Goh, J. Y., & Gerdes, J. (2017). Path-tracking for autonomous vehicles at the limit of friction. In *2017 American control conference (ACC)* (pp. 5586–5591). IEEE.

Law, C. K., Dalal, D., & Shearrow, S. (2018). Robust model predictive control for autonomous vehicles/self driving Cars. arXiv preprint arXiv:1805.08551.

Li, C., Li, X., Li, J., Zhu, Q., & Dai, B. (2017). Trajectory planning for autonomous ground vehicles driving in structured environments. In *2017 9th international conference on intelligent human-machine systems and cybernetics (IHMSC), Vol. 2* (pp. 41–46). IEEE.

Li, X., Sun, Z., Cao, D., Liu, D., & He, H. (2017). Development of a new integrated local trajectory planning and tracking control framework for autonomous ground vehicles. *Mechanical Systems and Signal Processing*, *87*, 118–137.

Liniger, A., Domahidi, A., & Morari, M. (2015). Optimization-based autonomous racing of 1: 43 scale RC cars. *Optimal Control Applications & Methods*, *36*(5), 628–647.

Liu, H., Lai, X., & Wu, W. (2013). Time-optimal and jerk-continuous trajectory planning for robot manipulators with kinematic constraints. *Robotics and Computer-Integrated Manufacturing*, *29*(2), 309–317.

Ni, J., & Hu, J. (2017). Dynamics control of autonomous vehicle at driving limits and experiment on an autonomous formula racing car. *Mechanical Systems and Signal Processing*, *90*, 154–174.

Pacejka, H. (2005). *Tire and vehicle dynamics*. Elsevier.

Rosolia, U., Carvalho, A., & Borrelli, F. (2017). Autonomous racing using learning model predictive control. In *2017 American control conference (ACC)* (pp. 5115–5120). IEEE.

Rosolia, U., Zhang, X., & Borrelli, F. (2019). Simple policy evaluation for data-rich iterative tasks. In *2019 American control conference (ACC)* (pp. 2855–2860). IEEE.

Rotondo, D., Puig, V., Nejjari, F., & Witczak, M. (2015). Automated generation and comparison of takagi–sugeno and polytopic quasi-LPV models. *Fuzzy Sets and Systems*, *277*, 44–64.

Schramm, D., Hiller, M., & Bardini, R. (2014). Vehicle dynamics. In *Modeling and simulation, Vol. 151*. Berlin, Heidelberg: Springer.

Sename, O., Gaspar, P., & Bokor, J. (2013). *Robust control and linear parameter varying approaches: application to vehicle dynamics, Vol. 437*. Springer.

Shin, K., & McKay, N. (1985). Minimum-time control of robotic manipulators with geometric path constraints. *IEEE Transactions on Automatic Control*, *30*(6), 531–541.

Stellato, B., Banjac, G., Goulart, P., Bemporad, A., & Boyd, S. (2018). OSQP: An operator splitting solver for quadratic programs. In *2018 UKACC 12th international conference on control (CONTROL)* (p. 339). IEEE.

Verschueren, R., De Bruyne, S., Zanon, M., Frasch, J. V., & Diehl, M. (2014). Towards time-optimal race car driving using nonlinear MPC in real-time. In *53rd IEEE conference on decision and control* (pp. 2505–2510). IEEE.

Verschueren, R., Zanon, M., Quirynen, R., & Diehl, M. (2016). Time-optimal race car driving using an online exact hessian based nonlinear MPC algorithm. In *2016 European control conference (ECC)* (pp. 141–147). IEEE.

Wächter, A., & Biegler, L. T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, *106*(1), 25–57.