Is minimalistic MPC based on centroidal dynamics enough to race complex vehicles?

Author Names Omitted. Paper-ID [add your ID here]

Abstract—**MPC is often used to control in real time various systems and processes in the chemical, mechanical and electrical fields. Here we focus on its use in autonomous driving. In particular, we investigate whether a minimalistic MPC based on a point mass model is able to drive a complete multibody vehicle model. Then we propose a method, called MPC offset-free, useful to compensate the differences between the MPC internal model and the driven vehicle.**

I. Introduction

Autonomous driving is one of the most discussed and actual topic in research and industrial fields. Here, we focus our attention on autonomous race car. In particular, in literature, three different approaches are employed in order to compute real-time control input for an autonomous car. In the first one the control of the vehicle is realized with a deep dense neural network, which is trained with recorded vehicle data to represent the vehicle's dynamic behaviour **cita1-2-3**. In the second approach a model predictive control (MPC) solves at each sampling instant, a finite horizon open-loop control problem, using the current state of the system as the initial state; the optimization yields an optimal control sequence and the first control in this sequence is applied as an input **cita4-5-6**. The last method is a mix of the previous approaches, in this case a learning model predictive control (LNMPC) is used in order to find optimal control and to learn the system model and the disturbance acting on it through the data from the previous trajectories **cita7-8-9**.

In our work, we start from the classical MPC approach, and we employ a minimalistic MPC, hence a very simplified vehicle model is used as MPC internal model in order to reduce computational time. After that the offset-free theory developed in **citaPannocchia**, is implemented in order to compensate the differences between the MPC model and the driven vehicle.

II. Proposed approach

A. Model of the simulated vehicle

B. MPC internal model

In order to race the fully-fledged vehicle we build an MPC dynamic model from first principles which strives to be as minimalistic as possible for maximum computational efficiency yet able to capture the fundamentals of vehicle dynamics. To this sake, a point-mass model is devised, embodying the dynamics of the center of mass (CoM) of the full model. This model has 5 DoFs and, worthily enough, it employs the very same basic parameters of the full vehicle, such as: mass, aerodynamic coefficients, grip and power limits for braking and traction phases. These are usually readily available even for a complex vehicle.

Control inputs are the longitudinal force F_x, and the angular acceleration along the vertical axes $r_p = dr/dt$, where r is the yaw rate [1].

The steering angle, needed to control the simulated vehicle, can be extracted by assuming that

$$\delta = \alpha_1 + \beta_1 = F_{y_{11}}/C_\alpha + (v + r a_1)/u \tag{1}$$

where, exploiting the same notation in **?**], C_α is the cornering stiffness of the front tire and a_1 is the distance between the Centre of Mass (CoM) and the front axle. Tire slip angles α_{1i} and vehicle slip angles β_{1i} are assumed equal for each wheel $i = 1, 2$ of the same axle, hence $\alpha_{11} = \alpha_{12} = \alpha_1$ and $\beta_{11} = \beta_{12} = \beta_1$. Front wheel lateral forces $F_{y_{1i}}$ are estimated within the steady-state assumption, leading to $F_{y_{11}} = F_{y_{12}} = F_y a_2/(2l)$. F_y, included as an algebraic state, summarizes the total lateral force acting on the vehicle model. Then, a_2 is the distance between the CoM and the rear axle and l is the wheelbase.

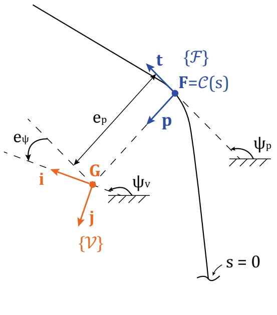
C. Spatial formulation

The race track, assumed planar, is modelled through the parametric 2D curve

$$\mathcal{C}(\alpha) = \{\boldsymbol{x}(\alpha) = [x(\alpha), y(\alpha)]^T \in \mathbb{R}^2 : \alpha \in [\alpha_0, \alpha_f]\} \tag{2}$$

that identifies the road centerline (aka track spine), and the 1D curve $\mathcal{W}(\alpha)$ that specifies the track width. With reference to Figure 1a, the *curve parameter* α uniquely selects a point $\boldsymbol{F} = \boldsymbol{x}(\alpha)$ that defines the origin of the *Frenet-Serret frame* $\mathcal{F} = \{\boldsymbol{F}, (\boldsymbol{t}, \boldsymbol{p})\}$ with unit tangent and normal vectors $\boldsymbol{t}$ and $\boldsymbol{p}$ of $\mathcal{C}$ at point $\boldsymbol{F}$. The vehicle reference system $\mathcal{V} = \{\boldsymbol{G}, (\boldsymbol{i}, \boldsymbol{j})\}$ can be expressed with the aid of the moving frame $\mathcal{F}$ with a *lateral displacement* e_p along the track normal direction $\boldsymbol{p}$ and the *heading error* e_ψ. In order to maintain $\mathcal{F}$ side-by-side with $\mathcal{V}$, the Frenet-Serret system has to proceed together with the vehicle: this leads to a relation between vehicle and Frenet-Serret velocities that ultimately imposes a bound between time and α increments.

[1] It is worth remarking that the apparent nonsense of introducing the yaw rate for a point mass is resolved if one thinks of its dynamics written in the body-fixed reference frame. This justifies the introduction of a yaw rate for the point mass which is the same as the body-fixed reference frame's yaw rate.



(a)

Fig. 1. (**a**) vehicle pose respect to the Frenet-Serret reference system identified on the track curve.

D. OCP problem formulation

The centroidal dynamics is formulated in spatial domain instead of the time, and the *direct collocation* is used to transform the *Optimal Control Problem* (OCP) into a *Non-Linear Program* (NLP). The NLP is coded in a scripting environment using the Matlab interface to the open-source CasADi framework written by Andersson et al. [1], which provides building blocks to efficiently formulate and solve large-scale optimization problems, and solved through IPOPT [**?**].

The OCP is formulated with the goal of finding the optimal control sequence necessary to minimize the travel time of the next l_N meters of the track ahead of the vehicle.

E. Offset-free MPC

III. Preliminary results

The overall system is modeled in Matlab-Simulink, where five different macro-blocks are developed. In particular, referring to 2, we have *Adams & States Measures* in which the multibody vehicle model is imported and its states are measured; *Lateral Displacement & Heading Error Estimation* where we localize through e_p and e_ψ the vehicle respect to the track; *Track Visualization* necessary to visualize the l_n meters of the track ahead of the vehicle; *MPC Parameters* and *MPC*, where in the first one necessary data of the track as control point for NURBS **cita nurbs** representation are calculated, and in the second one the MPC controller finds the optimal control inputs. The MPC controller is tested on the Indianapolis oval racing track. In particular, three different simulations are performed: (i) lap-time simulation with MPC controller, (ii) lap-time simulation with MPC controller in which the aerodynamic drag is neglected in the MPC internal model, (iii) lap-time simulation with offset-free MPC, where the aerodynamic drag is estimated through the technique explained in II-E.

For each simulation an horizon length of $l_N = 75$m is chosen and the OCP problem is discretized in $N = 40$ intervals. Hence, having a number of states $n_x = 7$, a number of controls $n_u = 2$ and a number of algebrical variables $n_z = 4$ each optimal control problem has 533 variables, 413 equality constraints and 82 inequality constraints. The solution of these problems on a laptop with 2.30GHz Intel(R) Core(TM) i7-10875H CPU and 32 GB di RAM requires a computing time $t_c \simeq 60$ms.

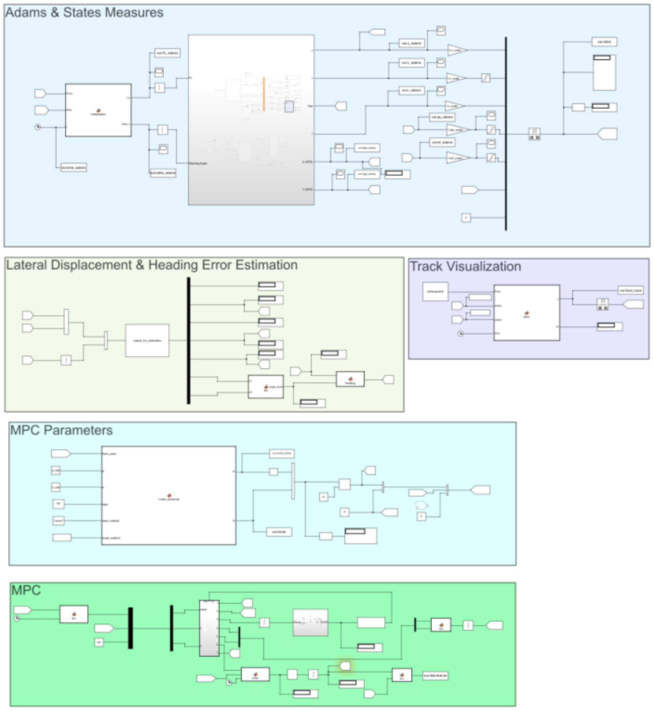


Fig. 2. Simulink model of the simulated system

Solutions in terms of controlled vehicle trajectories and inputs are compared, focusing on the entry phase of the first curve.

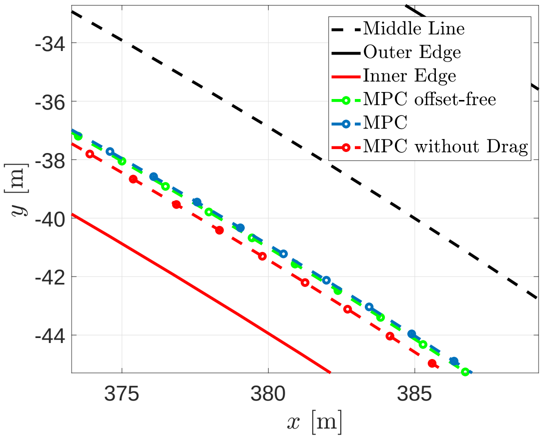


Fig. 3. Trajectories comparison between lap simulation with MPC (blue), MPC offset-free (green), MPC without drag (red)

Figures.3 - 4 highlight how MPC with drag and MPC offset-free are capable to drive the Adams vehicle in a practically identical way, with a lap-time difference of 6.5ms. Instead, if the aerodynamic drag is not modelled inside the MPC, the

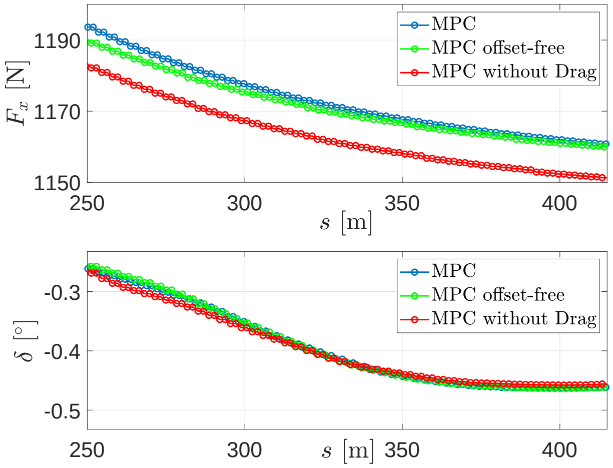


Fig. 4. Comparison between control inputs, longitudinal force (top), steering angle of the front wheel (bottom), obtained with MPC (blue), MPC offset-free (green) and MPC without drag (red)

controller is not able to maintain the real vehicle on the track: in fact the simulation fails on the entry phase of the second curve.

IV. Conclusion

The minimalistic MPC based solely on the vehicle centroidal dynamics seems enough to race complex vehicle counterpart. In particular, parameters of the MPC model such as mass, aerodynamic coefficients or power limits, have to be equal to those of the driven car: in this way the point-mass model can be considered as a condensed model of the complex vehicle itself.

Nevertheless, the offset-free technique is a powerful and efficient addendum that allows to relax even further the internal model. It is also forgiving enough to adjust some uncertain yet fundamental parameters of the complex vehicle in a convenient way at run time. As a proof of concept, an internal model agnostic of the aerodynamic drag has shown to dramatically benefit from the estimation of the drag force through the disturbance state d.

Further developments to the MPC internal model, as well as new approaches to the estimation of the steer angle, can lead to drive the full vehicle in conditions even closer to handling limits.

References

[1] Joel A.E. Andersson, Joris Gillis, Greg Horn, James B. Rawlings, and Moritz Diehl. CasADi: a software framework for nonlinear optimization and optimal control. *Math. Program. Comput.*, 11(1):1–36, mar 2019. ISSN 18672957. doi: 10.1007/s12532-018-0139-4. URL https://link.springer.com/article/10.1007/s12532-018-0139-4.