Time-optimal Race Car Driving using an Online Exact Hessian based Nonlinear MPC Algorithm

Robin Verschueren[1], Mario Zanon[3], Rien Quirynen[1,2], Moritz Diehl[1]

***Abstract*— This work presents an embedded nonlinear model predictive control (NMPC) strategy for autonomous vehicles under a minimum time objective. The time-optimal control problem is stated in a path-parametric formulation such that existing reliable numerical methods for real-time nonlinear MPC can be used. Building on previous work on time-optimal driving, we present an approach based on a sequential quadratic programming type algorithm with online propagation of second order derivatives. As an illustration of our method, we provide closed-loop simulation results based on a vehicle model identified for small-scale electric race cars.**

I. Introduction

To this day, a variety of advanced driver assistance systems (ADAS), such as automatic parking, autonomous cruise control and lane keeping functionality, have been introduced in consumer vehicles. The mentioned assistance systems are often based on traditional control schemes, and their scope is limited to specific simple tasks. Fully autonomous driving, on the other hand, requires more advanced control paradigms. For instance, model predictive control (MPC) has been shown to be a reliable and efficient control technique for self-driving ground vehicles. Some earlier attempts were due to [7], [11], and include experimental validation. These approaches use various model simplification techniques to reduce the computational complexity. A detailed vehicle model consisting of 14 states, and with inclusion of a Pacejka tire model, used in a nonlinear MPC (NMPC) scheme is presented in [10]. In there, the control problem is rendered real-time feasible by using fast tailored NMPC algorithms.

Time-optimal driving is one of the most challenging tasks, due to the obvious antagonism between safety and speed: the controller needs to act fast while coping with the nonlinear vehicle dynamics and satisfying the track boundaries. In this paper we focus on efficient algorithms and tailored problem formulations for time-optimal driving.

A related task to time-optimal driving is path following. Given a prespecified nominal trajectory, path following calculates a *timing law*, that determines *when* to be *where* on this trajectory. Nonlinear MPC is a suitable control technique for online path following, as showed in [8] and experimentally validated in e.g. [17]. A convex formulation for path following for general vehicles is presented in [24], [18].

[1]Department IMTEK, University of Freiburg, Germany. robin.verschueren@imtek.uni-freiburg.de

[2]Department ESAT-STADIUS, KU Leuven University, Belgium.

[3]Department of Signals and Systems, Chalmers University of Technology, Göteborg, Sweden.

In contrast to path following, driving time-optimally means to deviate largely from a nominal path (e.g. the center-line of the race track) in order to save time. A conventional solution strategy to time-optimal driving consists of a two-level scheme. On the higher level, a dynamic optimiziation routine calculates the optimal geometric trajectory, while the lower level tracking controller uses the resulting trajectory as a time-dependent reference. Such a scheme is worked out in [4] and in [17] including experimental results on small-scale race cars. The work in [16] displays experimental results on a full-scale racing car on a track with low friction coefficient. Others have proposed a one-level approach, as [15], [23]. In [12], offline computed time-optimal trajectories are fed as reference to an actual vehicle.

In [25], a real-time NMPC scheme is presented, based on a multiple shooting discretization in the real-time iteration (RTI) framework [5]. The time-optimal problem is reformulated as a path-parametric system, as proposed in [10]. Similar to existing efficient algorithms for real-time NMPC problems with least squares cost function, a Gauss-Newton Hessian approximation is used to solve the time-optimal problem.

Note that time-optimal driving is an economic MPC problem as the stage cost is not bounded from below by a $\mathcal{K}_\infty$ function [1]. Therefore, stability is harder to guarantee and efficient algorithms developed for tracking MPC cannot be deployed. In this paper we focus on the second issue.

The contribution of this paper is as follows. In contrast to the previous work in [25], we solve a nonlinear economic MPC (EMPC) problem directly, taking into account nonlinear vehicle dynamics, path constraints and control bounds. We compute second order derivatives online and use them in a sequential quadratic programming (SQP) type algorithm based on a multiple shooting discretization. Recent algorithmic advances have made it possible to generate second order sensitivities efficiently [21], which makes it possible to use them in real-time algorithms. Furthermore, we present a slightly more sophisticated version of the vehicle model used in [25].

The remainder of the paper is structured as follows. In Section II, the vehicle model and the spatial system reformulation is presented. Thereafter, we show the time-optimal nonlinear MPC problem under consideration and formulate the exact Hessian based solution strategy. In Section V, numerical simulations are introduced to present the effectiveness of our method. The paper concludes with an outreach to future research interests.

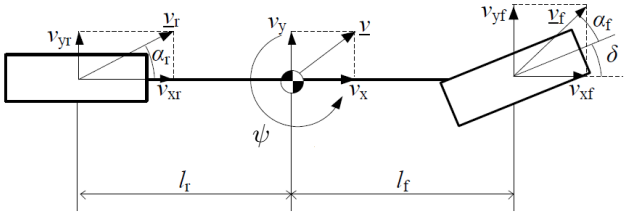

Fig. 1: Geometric properties of the vehicle model of Section II.

II. Vehicle Dynamics

In this section, we present the model equations derived as an ordinary differential equation (ODE) in the time domain first. In a second step, we make use of a spatial transformation in order to formulate the model in the spatial domain. This approach has been proposed and motivated in [11], [10], [25].

A. Vehicle Model

In this paper, we consider an extension of the bicycle model used in [25]. We model the lateral tire forces using a Pacejka model [20]. The model is based on an experimental test setup with race cars of scale 1:43. The system dynamics in the time domain are given by

$$m\dot{v}_x = m\dot{\psi}v_y + F_x^{\mathrm{d}} - F_x^{\mathrm{r}}, \tag{1a}$$
$$m\dot{v}_y = -m\dot{\psi}v_x + F_y^{\mathrm{r}} + F_y^{\mathrm{f}}\cos(\delta), \tag{1b}$$
$$J\ddot{\psi} = l_{\mathrm{f}}F_y^{\mathrm{f}}\cos(\delta) - l_{\mathrm{r}}F_y^{\mathrm{r}}, \tag{1c}$$

where v_x and v_y are the velocities along the x and y axes in the car reference frame. Angle δ is the steering angle which we assume to control directly. Angle ψ denotes the orientation of the vehicle in the absolute reference frame and $\psi = 0\,\mathrm{rad}$ indicates that the axes x and y of the vehicle reference frame are aligned with the axes X and Y of the absolute reference frame. Parameters m and J denote the vehicle's mass and moment of inertia respectively, while l_{f} and l_{r} denote the distance of the vehicle's center of mass from the front and rear wheel respectively, see Figure 1.

The engine drive force of the DC motor in the race cars is given by

$$F_x^{\mathrm{d}} = (C_{\mathrm{m1}} - C_{\mathrm{m2}}v_x)\,D,$$

where D is the engine duty-cycle, and the rolling friction force is given by $F_x^{\mathrm{r}} = C_{\mathrm{r0}} + C_{\mathrm{r2}}v_x^2$. The lateral tire forces are given by the Pacejka formula

$$F_y^{\bullet} = D_{\mathrm{c}}\sin(C_{\mathrm{c}}\tan^{-1}(\alpha^{\bullet}B_{\mathrm{c}} - (\alpha^{\bullet}B_{\mathrm{c}} - \tan^{-1}(\alpha^{\bullet}B_{\mathrm{c}}))E_{\mathrm{c}})),$$

with $\bullet = \mathrm{f}, \mathrm{r}$. The slip angles are given by

$$\alpha^{\mathrm{f}} = -\tan^{-1}\left(\frac{v_y + \dot{\psi}l_{\mathrm{f}}}{v_x}\right) + \delta,$$
$$\alpha^{\mathrm{r}} = -\tan^{-1}\left(\frac{v_y - \dot{\psi}l_{\mathrm{r}}}{v_x}\right).$$

TABLE I: Parameters of the vehicle model.

C_{m1}	C_{m2}	C_{r0}	C_{r2}
0.48 N	0.087 kg/s	0.024 N	0.004 kg/m
B_{c}	C_{c}	D_{c}	E_{c}
8	2.1	0.1 N	1
m	J	l_{f}	l_{r}
0.04 kg	$1.6 \cdot 10^{-5}$ kg m^2	0.028 m	0.028 m

Note that, as a first approximation, we assume no longitudinal slip. This simplification is also due to the impossibility to measure the rotational speed of the wheels in our setup, which impedes the use of longitudinal tire models and, in turn, of combined slip models.

The parameters of the Pacejka tire force model have been identified on the real setup and are reported in Table I together with all other model parameters. Please note that a basic identification method has been used, a more elaborate method would be needed in order to obtain reliable experimental results.

B. Spatial Reformulation of Dynamics

The vehicle model presented in the previous subsection is formulated in the time domain. Taking into account track information in the system dynamics directly leads to a more natural formulation of the control problem. We will make use of a path-parametric system reformulation, as presented in [10], and briefly explained here for clarity. This reformulation results in a dynamic model depending on a path parameter, such that time becomes a state variable that can be optimized for. Additionally, by using the curvilinear coordinate system around the centerline of the track, road constraints become simple bounds. The advantages relative to this choice are discussed in [10].

We project the $X - Y$ absolute coordinates on the centerline of the track, parametrised as $\sigma(s)$, and replace them with the lateral displacement e_y from $\sigma(s)$. Similarly, ψ is replaced by the angular deviation e_ψ. The time derivative is related to the spatial derivative $(\cdot)' = \frac{\mathrm{d}\cdot}{\mathrm{d}s} = \frac{1}{\dot{s}}\frac{\mathrm{d}\cdot}{\mathrm{d}t}$, where $\dot{s} = \frac{1}{1-\kappa_\sigma \cdot e_y}\left(v_x\cos(e_\psi) - v_y\sin(e_\psi)\right)$, and κ_σ is the local curvature of the curve σ. For the newly introduced states we obtain

$$e_y'(s) = (v_x\sin(e_\psi) + v_y\cos(e_\psi))/\dot{s},$$
$$e_\psi'(s) = \dot{\psi}/\dot{s} - \kappa_\sigma(s).$$

All details of the spatial transformation can be found in [11], [10], [25]. In the following, we lump the system dynamics in $\xi' = f(\xi, u)$, where the state vector is $\xi = (e_y, e_\psi, v_x, v_y, \dot{\psi}, t)$ and the control vector is $u = (\delta, D)$.

III. Exact Hessian Based Nonlinear MPC

In this section, we consider the following continuous and nonlinear OCP formulation:

$$\begin{aligned}
\min_{\xi(\cdot),\,u(\cdot)} \quad & E(\xi(S)) && \text{(2a)}\\
\text{s.t.} \quad & 0 \;=\; \xi(0) - \hat{\xi}_0, && \text{(2b)}\\
& 0 \;=\; f(\xi'(s), \xi(s), u(s)), \;\forall s \in [0, S], && \text{(2c)}\\
& 0 \;\leq\; c(\xi(s), u(s)), \qquad \forall s \in [0, S], && \text{(2d)}
\end{aligned}$$

where s is usually time but, due to the use of a spatial reformulation in our case it is the curvilinear coordinate of the centerline of the track. We therefore also denote the state derivatives as $\xi'(s)$ instead of $\dot{\xi}(t)$. The control horizon length is denoted as S, $\xi(s) \in \mathbb{R}^{n_\xi}$ denotes the differential states and $u(s) \in \mathbb{R}^{n_u}$ are the control inputs. This parametric OCP depends on the current state estimate $\hat{\xi}_0 \in \mathbb{R}^{n_\xi}$ through the initial value condition of Eq. (2b). The objective in (2a) is defined by the terminal cost $E(\cdot)$ and the path constraints in (2d) are assumed to be affine for simplicity of notation. The nonlinear dynamics in Eq. (2c) are formulated as a system of Ordinary Differential Equations (ODE). The functions $E(\cdot)$ and $f(\cdot)$ for this discussion will need to be twice continuously differentiable in all arguments within the domain of interest, i.e. $v_x > 0, \dot{s} > 0$, which corresponds to forward motion of the vehicle.

A direct *multiple shooting* discretization [3] of OCP (2) which divides the horizon in N control intervals, results in the following NLP:

$$\begin{aligned}
\min_{\Xi,\,U} \quad & E(\xi_N) && \text{(3a)}\\
\text{s.t.} \quad & 0 \;=\; \phi(\xi_i, u_i) - \xi_{i+1}, \quad i = 0, \ldots, N-1, && \text{(3b)}\\
& 0 \;\leq\; c_i + C_i \begin{bmatrix} \xi_i \\ u_i \end{bmatrix}, \qquad i = 0, \ldots, N-1, && \text{(3c)}
\end{aligned}$$

with state trajectory $\Xi := [\xi_0^\top, \ldots, \xi_N^\top]^\top$ and control trajectory $U := [u_0^\top, \ldots, u_{N-1}^\top]^\top$. Function $\phi(\cdot)$ represents a sufficiently accurate numerical simulation of the nonlinear dynamics in (2c). To arrive at a more compact notation, the initial value condition (2b) has been included in the affine inequality constraints for $i = 0$ in Equation (3c).

A. Exact Hessian Based Sequential Quadratic Programming

In this paper, we solve NLP (3) by means of SQP, i.e. by sequentially solving the following Quadratic Program (QP):

$$\begin{aligned}
\min_{\Delta W} \quad & \sum_{i=0}^{N-1} \left(\frac{1}{2} \Delta w_i^\top H_i \, \Delta w_i \right) && \text{(4a)}\\
& + g_N^\top \Delta \xi_N + \frac{1}{2} \Delta \xi_N^\top H_N \, \Delta \xi_N, && \text{(4b)}\\
\text{s.t.} \quad & 0 \;=\; d_i + \frac{\mathrm{d}\phi(\bar{w}_i)}{\mathrm{d}w_i} \Delta w_i - \Delta \xi_{i+1}, && \text{(4c)}\\
& 0 \;\leq\; \tilde{c}_i + C_i \, \Delta w_i, \quad i = 0, \ldots, N-1, && \text{(4d)}
\end{aligned}$$

where $\Delta W := (\Delta w_0, \ldots, \Delta w_N)$, $\Delta w_i := w_i - \bar{w}_i$ for $i = 0, \ldots, N-1$ and $\Delta w_N := \Delta \xi_N$. The constraint values are rewritten using $d_i := \phi(\bar{w}_i) - \bar{\xi}_{i+1}$ and $\tilde{c}_i := c_i + C_i \, \bar{w}_i$. The notation $\bar{w}_i := (\bar{x}_i, \bar{u}_i)$ is used to denote the current optimization values, which are updated in each iteration by solving the latter QP subproblem, i.e., $\bar{W}^+ = \bar{W} + \Delta W$ in case of a full SQP step [19].

The Lagrangian of OCP (3) is separable in time and the terms at each stage i are defined as $\mathcal{L}_i(w_i, \lambda_i, \nu_i) := \lambda_i^\top \left(\phi(w_i) - \xi_{i+1}\right) + \nu_i^\top \left(c_i + C_i \, w_i\right)$ for $i = 0, \ldots, N-1$. The quadratic term in the objective (4a) is defined by $H_i := \nabla^2_{w_i} \mathcal{L}_i(\bar{w}_i, \bar{\lambda}_i, \bar{\nu}_i)$, when using an exact Hessian based SQP method [19]. The contribution to the Hessian can be written as

$$H_i = \nabla^2_{w_i} \left(\bar{\lambda}_i^\top \phi(\bar{w}_i) \right), \quad i = 0, \ldots, N-1, \qquad (5)$$

where the Lagrange multipliers of the continuity constraints (3b) are defined as $\Lambda := [\lambda_0^\top, \ldots, \lambda_{N-1}^\top]^\top$.

The second order derivatives $\nabla^2_{w_i} \left(\bar{\lambda}_i^\top \phi(\bar{w}_i) \right)$ are the result of a propagation of sensitivities through the system dynamics, which can be performed efficiently as discussed in [21].

We define the contribution of the terminal cost to the gradient and Hessian in (4b) as $g_N := \nabla_{\xi_N} E(\bar{\xi}_N)$ and $H_N := \nabla^2_{\xi_N} E(\bar{\xi}_N)$ respectively.

B. Efficient Convexification Procedures

The QP subproblem (4) is convex only when the Hessian block H_i is positive semi-definite for $i = 0, \ldots, N$, which is not necessarily the case in general. In what follows, we therefore target a suitable Hessian regularization based on [19], [21].

An application of the eigenvalue decomposition to regularize the Hessian, would consist in computing the approximate Hessian $\tilde{H}$ as

$$\tilde{H} := V \, \mathrm{abs}\left(\Gamma\right) V^\top \quad \text{with} \quad H = V \Gamma V^\top, \qquad (6)$$

where the function $\mathrm{abs}\left(\gamma\right)$ is defined as:

$$\mathrm{abs}\left(\gamma\right) := \begin{cases} \gamma & \text{if } \gamma > \epsilon \\ \epsilon & \text{otherwise} \end{cases}$$

where $\epsilon > 0$ denotes a small constant in order to ensure positive definiteness of the Hessian approximation. Note that also other heuristic definitions would be possible [21]. We have introduced the eigenvalue decomposition $H = V \Gamma V^\top$, i.e. Γ is a diagonal matrix and V is an orthogonal matrix. The absolute value of the diagonal matrix Γ is computed component-wise.

Clearly, the computational complexity of this approach does not scale well with the size of the problem. We therefore propose two different approaches, both based on the eigenvalue decomposition, but not applied to the full Hessian directly. The first one considers regularizing the condensed Hessian, i.e. the Hessian obtained after eliminating the state variables $\Delta \xi_i$ in the structured QP (4) by means of condensing [3]. The second one considers exploiting the block-diagonal structure of the full Hessian in order to regularize each block independently.

TABLE II: Computational complexity: Hessian regularization.

	Algorithm 1	Algorithm 2
EVD computation	$9\ (Nn_u)^3$	$9\,N\ (n_x+n_u)^3 + 9\,n_x^3$

1) Regularizing the Condensed Hessian: The first approach that we propose consists in applying Equation (6) to the condensed Hessian. Algorithm 1 summarizes the resulting approach, using the compact notation $\mathrm{reg}\,(H) = V\,\mathrm{abs}\,(\Gamma)\,V^\top$. Note that in the absence of (active) inequality constraints, the condensed Hessian is positive definite whenever the reduced Hessian is positive definite [19]. This entails that, in the absence of active inequality constraints, Algorithm 1 will not regularize unless it is strictly necessary to do so.

Algorithm 1 The condensed regularization approach

Input: Hessian blocks H_i for $i = 0, \ldots, N$.
Output: The condensed and regularized Hessian $\tilde{H}^{\mathrm{c}} \succ \epsilon I$.
1: Perform the condensing routine [3] to the original QP (4).
2: Regularize the condensed Hessian matrix $\tilde{H}^{\mathrm{c}} = \mathrm{reg}\,(H^{\mathrm{c}})$.

While Algorithm 1 has a reduced numerical complexity , it still scales cubically with the horizon length N, as detailed in Table II.

2) Regularizing the Hessian Blocks Independently: Alternatively, one can decide to regularize each block H_i for $i = 0, \ldots, N$ of the original Hessian, as described in Algorithm 2. As opposed to regularizing the condensed Hessian, this second approach has the disadvantage that there is no equivalence with the positive definiteness of the reduced Hessian, not even in case there are no (active) inequality constraints at the solution. This means that, by using Algorithm 2, one might need to regularize the Hessian even in cases in which the condensed Hessian is positive definite. On the other hand, as shown in Table II, the block regularization approach scales linearly with the horizon length N. This technique does in principle not require the use of condensing to solve the QP and the regularization of each Hessian block can additionally be performed in parallel, similar to the numerical simulation routines within direct multiple shooting [3].

Algorithm 2 The block regularization approach

Input: Hessian blocks H_i for $i = 0, \ldots, N$.
Output: The condensed and regularized Hessian $\tilde{H}^{\mathrm{c}} \succ \epsilon I$.
1: **for** $i = 0 : N$ **do**
2: Regularize the Hessian block $\tilde{H}_i = \mathrm{reg}\,(H_i)$.
3: **end for**
4: Perform the condensing routine [3] to the regularized QP (4).

Remark 1: For the computation of the eigenvalue decomposition (EVD) of the symmetric Hessian matrix in Table II, we considered the efficient combination of the Householder tridiagonalization which reduces the matrix to a tridiagonal form, followed by the symmetric QR algorithm. The latter method requires overall about $9n^3$ flops to compute the complete eigenvalue decomposition, where n denotes the dimension of the symmetric matrix [13].

Remark 2: In general, only a linear convergence rate can be expected for the SQP method based on the two computationally cheap but conservative convexification procedures proposed in this paper, see [19]. It is important to mention that such a result on the local convergence rate itself is of less practical importance when implementing fast Nonlinear MPC in a real-world test setup as targeted in this work.

IV. Real-Time Nonlinear MPC Implementation

This section discusses the details of deploying a real-time implementation of the presented Nonlinear MPC scheme on the experimental setup.

A. Exact Hessian Based Real-Time Iterations

Algorithm 3 presents a detailed implementation of an exact Hessian based version of the Real-Time Iteration (RTI) scheme [5], as presented earlier in [21]. It can be based either on Algorithm 2 or 1 to convexify and condense each QP subproblem in Eq. (4). Note that the SQP iterations are divided into a preparation and a feedback phase like in the classical Gauss-Newton based RTI algorithm.

Remark 3: A full proof of local convergence and stability for the exact Hessian based RTI variant in Algorithm 3 is outside the scope of this paper. The results in [6] for the classical RTI scheme based on Gauss-Newton Hessian approximations could however be extended, by applying the convergence theory for exact Hessian based SQP [19].

Algorithm 3 Real-Time Exact Hessian RTI

Input: An initial guess for the trajectories $\bar{W}$, $\bar{\Lambda}$.
1: **while** `sampling instant` **do**
2: Prepare the QP in (4) using AD [2] and a suitable integrator [22].
3: Apply Algorithm 1 or 2 to obtain the corresponding convexified and condensed QP subproblem.
4: Wait until the new measurement $\hat{x}_t$ arrives.
5: Solve the dense QP, update $\bar{W}^+ = \bar{W} + \Delta W$ and $\bar{\Lambda}^+$ is defined by the Lagrange multipliers with respect to the constraints in (4c).
6: Send the new control input $\bar{u}_0^+$ to the process.
7: **end while**

B. ACADO Code Generation Software

This paper relies on the open-source software for auto-generating an Exact Hessian based RTI scheme as part of the ACADO Toolkit [14]. The software is free of charge and can be downloaded from `www.acadotoolkit.org` for reproducing the numerical results which are presented in this paper. The tool allows the export of highly efficient C-code, implementing Algorithm 3 based on algorithmic ideas that have been presented in [14], [21]. In our implementation, the QP solver is based on a condensing technique that has originally been proposed in [3]. The resulting smaller and dense QP is then solved with an online active set method using the

software qpOASES [9]. The numerical performance of this algorithm is discussed in the next section.

V. Numerical Results

In this section, we detail the MPC formulation used and we present the simulation results obtained.

A. MPC Setup

Because we use the spatial transformation, we formulate the OCP in space. The continuous formulation is given by

$$\begin{aligned}
\min_{\xi(\cdot),u(\cdot)} \quad & t(S) && && \text{(7a)}\\
\text{s.t.} \quad & \xi'(s) = f(s,\xi(s),u(s)), && \forall s\in[0,S] && \text{(7b)}\\
& e^y(s) \in [\underline{e^y}, \overline{e^y}], && \forall s\in[0,S] && \text{(7c)}\\
& \alpha^\bullet(s) \in [\underline{\alpha}, \overline{\alpha}],\ \bullet = \mathrm{f},\mathrm{r}, && \forall s\in[0,S] && \text{(7d)}\\
& u(s) \in [\underline{\delta}, \overline{\delta}] \times [-1,1], && \forall s\in[0,S] && \text{(7e)}\\
& \xi(0) = \xi_0. && && \text{(7f)}
\end{aligned}$$

We discretize the OCP (7) using 35 control intervals with a piecewise constant control parametrization. For discretizing the system dynamics, we use a fixed stepsize explicit Runge-Kutta integrator of order 4 with 10 integration steps over a sampling length of 0.025 m.

Note that we introduced the somehow artificial bounds (7d) on the slip angles. This choice is necessary to avoid that the OCP solver brings the system in configurations in which we cannot trust the model. A justification of this choice is given in the following subsection. The bounds are given by $\overline{\alpha} = -\underline{\alpha} = 0.16$ rad, $\overline{\delta} = -\underline{\delta} = 25$ deg and $\overline{e^y} = -\underline{e^y} = 0.13$ m. We introduced two slack variables as ficticious controls, one for the constraints on e_y and the other one for the constraints on $\alpha^\bullet,\ \bullet = \mathrm{f},\mathrm{r}$.

B. Simulation Results

In this section, we present simulation results obtained with the proposed time-optimal approach. We first solved the periodic time-optimal problem offline with a multiple shooting discretization using CasADi [2] and the NLP solver Ipopt [26]. We used the obtained result as a baseline for comparison for the performance of the closed-loop MPC trajectory.

We first analyze the results obtained using the regularization Algorithm 2. The simulation results obtained by the closed-loop system are displayed in Figure 2 in thick line. In the same figure, the offline-computed periodic optimal trajectory is also displayed in thin line. Due to the finite horizon, when exiting some curves, the next curve is not yet in the MPC prediction horizon and therefore, the trajectory computed by the MPC controller differs from the periodic optimal one. It can be seen that, due to time-optimality, the trajectory touches the constraints on e_y. For safety reasons, for these lateral road bounds a backoff of 0.04 m has been taken with respect to the road boundary. This backoff allows us to impose the road bound avoidance constraint on the center of mass of the vehicle only. Clearly, while this choice makes the formulation simpler and more efficient from a computational point of view, it also introduces some conservatism and other strategies can lead to trajectories that get closer to the real bounds of the road and, therefore, shorter lap times. Future research will investigate alternative constraint formulations.

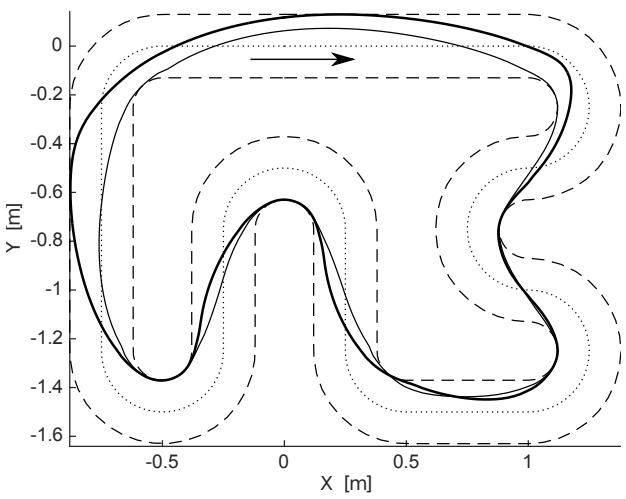

Fig. 2: Closed-loop MPC simulations (thick line) and periodic optimal trajectory (thin line). The centerline of the track is displayed in dotted line and the track boundaries are displayed in dashed lines.

The slip angles, lateral forces and controls for both the MPC closed loop simulations and the periodic optimal trajectory are displayed in Figure 3. It can be seen that MPC delivers a solution which is similar to the periodic optimal one for the slip angles and lateral forces.

Concerning the controls, the steering angle δ is similar for the MPC closed loop and the periodic optimal trajectory, but the engine duty cycle D has a very different profile. It is interesting to note how, due to time-optimality, for most of the time there is at least one active constraint. By looking e.g. at $s = 5\,\mathrm{m}$, one can see that the duty-cycle is not bang-bang, but has a singular arc and so does the slip angle α^r, while the constraint on the steering angle δ becomes active. As soon as the constraint on δ becomes inactive, both constraints on D and α^r become active again.

The lap times for the periodic optimal trajectory and for the MPC closed-loop are 4.27 s and 4.55 s respectively. RTI-based MPC therefore yields lap times which are $6.6\,\%$ higher than the optimum. The largest part of this loss is due to the prediction horizon being finite. By converging the algorithm, it is possible to only marginally reduce the lap time to 4.52 s. We remark, moreover, that such a control law would not be real-time feasible and can therefore only be used as an ideal term of comparison.

We remark that the introduction of the constraints on the slip angle is motivated by the simplification introduced in the model. It is important to underline that the lateral forces are not constrained by our formulation: because we choose the upper bound on the slip angle as the value which yields the peak of the Pacejka force, the vehicle is allowed to exploit

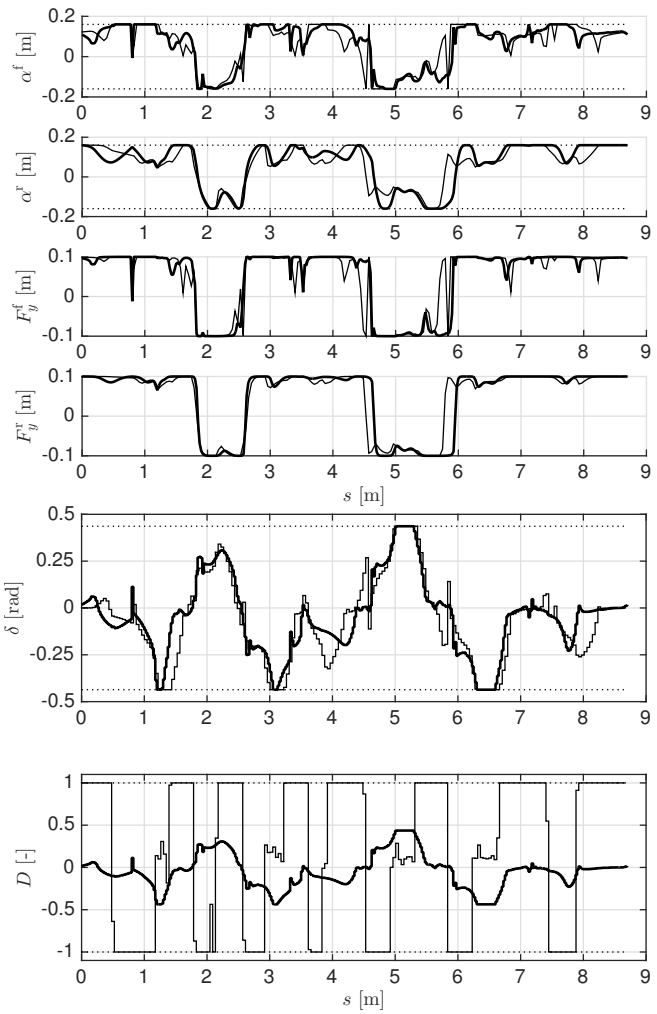

Fig. 3: Slip angles, lateral forces and controls in simulations: MPC closed loop in thick line and periodic optimal trajectory in thin line.

the full available lateral force and it is only prevented from drifting the wheels sidewise. The periodic optimal trajectory obtained without imposing these constraints yields a lap time of 4.03 s, with a gain of 5.6 % on the constrained trajectory. The OCP solver obtains this improvement by letting the tires drift sideways. However, because our tire model is not very accurate and we do not take into account the longitudinal tire dynamics, we decided not to trust the model beyond the peak of Pacejka's force. We remark that at the moment it is not possible to account for the longitudinal tire dynamics, due to the impossibility to measure the wheel rotational speed on the real setup.

We now turn to analyzing the closed-loop results obtained when using regularization Algorithm 1. The closed-loop trajectory obtained are displayed in Figure 4 in thick line. The closed-loop trajectory obtained with Algorithm 2 and the periodic optimal trajectory are reported in thin and dotted line respectively.

It can be seen that in this second case the trajectory is farther from the periodic optimal one on almost all track length. This is also reflected in the lap time, which is 5.58 s, i.e. 22% higher than that obtained when using Algorithm 2. This can be explained as follows. Because we rely on the RTI framework for approximately solving the MPC NLP, we take a single full Newton step at each sampling instant, i.e. we do not perform any globalization strategy. For this reason, we need to tune the threshold ϵ by choosing a high enough value in order to guarantee contraction. In particular for Algorithm 1, we needed to choose a rather high value for ϵ, which guarantees contraction, but is likely to result to slow convergence. Such a high value for ϵ produces worse results, as described above and plotted in Figure 4.

Unfortunately, an automatic procedure for tuning the regularization parameter ϵ is currently not available. Future research will aim at investigating regularization strategies further. We remark that the implementation of exact Hessian based RTI algorithms is fairly recent and the first result is due to [21], in 2014.

To conclude this section, we measure the computational load of Algorithm 2, as is shown in Table III. We remark that there is no large difference between Algorithms 1 and 2, which is to be expected as the solver spends most of the time inside of the online propagation of second order sensitivities; in this case the method of regularization is not the computational bottleneck; in other cases that might depend on the problem dimensions, see Table II.

TABLE III: Computation times spent inside the tailored ACADO algorithm. All computations done on a Macbook Pro with Intel Core i7 2.5Ghz, 16GB RAM.

Computation times (ms)	min	max	mean
One RTI step	3.1	5.6	3.6

VI. CONCLUSIONS

In this paper, we presented a nonlinear model predictive control scheme for time-optimal driving of race cars based on an exact Hessian SQP-type optimization algorithm. We made use of a vehicle model based on a Pacejka model for the lateral forces, while we assume no longitudinal slip. The model was identified using measurements from an experimental setup.

We show the algorithmic results with simulations, where we have compared its performance to the periodic optimal trajectory computed offline. As expected, the MPC trajectory has a slightly worse performance than the periodic optimal one, due to the prediction horizon being finite. Also, two methods of regularization of the possibly indefinite Hessian were presented and compared with one another.

As a subject of future work we envision the implementation of the control strategy on an experimental small-scale race car setup. As can be seen in Table III, and considering the experimental setup runs at a fixed control rate of 50 Hz, the algorithm enables real-time experiments. Furthermore, we intend to pursue a broader class of system identification

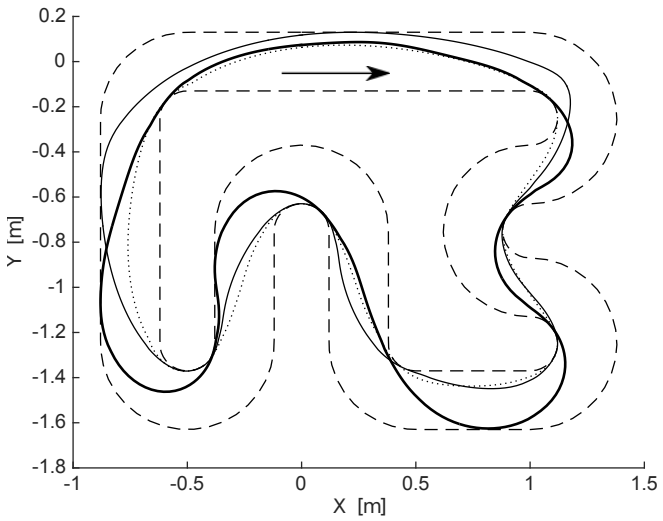

Fig. 4: Closed-loop MPC simulations with Algorithm 1 (thick line), Algorithm 2 (thin line) and periodic optimal trajectory (dotted line). The centerline of the track is displayed in dotted line and the track boundaries are displayed in dashed lines.

experiments, in order to caption better the physical behavior of the tire-road interaction.

ACKNOWLEDGMENTS

This research benefits from ERC-HIGHWIND (259 166), FP7-ITN-TEMPO (607 957), and H2020-ITN-AWESCO (642 682). R. Verschueren is a fellow from the TEMPO Initial Training Network funded by the People Programme (Marie Curie Actions) of the European Unions Seventh Framework Programme (FP7/2007-2013) under REA grant agreement no 607957. R. Quirynen holds a PhD fellowship of the Research Foundation – Flanders (FWO).

REFERENCES

[1] R. Amrit, J. Rawlings, and D. Angeli. Economic optimization using model predictive control with a terminal cost. *Annual Reviews in Control*, 35:178–186, 2011.

[2] J. Andersson. *A General-Purpose Software Framework for Dynamic Optimization*. PhD thesis, Arenberg Doctoral School, KU Leuven, October 2013.

[3] H. G. Bock and K. J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *Proceedings of the IFAC World Congress*, pages 242–247. Pergamon Press, 1984.

[4] F. Braghin, F. Cheli, S. Melzi, and E. Sabbioni. Race driver model. *Computers & Structures*, 86:1503–1516, July 2008.

[5] M. Diehl, H. G. Bock, J. Schlöder, R. Findeisen, Z. Nagy, and F. Allgöwer. Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *Journal of Process Control*, 12(4):577–585, 2002.

[6] M. Diehl, R. Findeisen, F. Allgöwer, H. G. Bock, and J. P. Schlöder. Nominal stability of the real-time iteration scheme for nonlinear model predictive control. *IEE Proc.-Control Theory Appl.*, 152(3):296–308, 2005.

[7] P. Falcone, F. Borrelli, H. E. Tseng, J. Asgari, and D. Hrovat. Integrated braking and steering model predictive control approach in autonomous vehicles. In *Advances in Automotive Control*, volume 5, pages 273–278, 2007.

[8] T. Faulwasser, B. Kern, and R. Findeisen. Model predictive path-following for constrained nonlinear systems. In *Joint 48th IEEE conference on Decision and Control and 28th Chinese Control Conference*, 2009.

[9] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl. qpOASES: a parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, 6(4):327–363, 2014.

[10] J. V. Frasch, A. J. Gray, M. Zanon, H. J. Ferreau, S. Sager, F. Borrelli, and M. Diehl. An auto-generated nonlinear MPC algorithm for real-time obstacle avoidance of ground vehicles. In *Proceedings of the European Control Conference (ECC)*, pages 4136–4141, 2013.

[11] Y. Gao, A. Gray, J. V. Frasch, T. Lin, H. E. Tseng, J. Hedrick, and F. Borrelli. Spatial predictive control for agile semi-autonomous ground vehicles. In *Proceedings of the 11th International Symposium on Advanced Vehicle Control*, 2012.

[12] M. Gerdts, S. Karrenberg, B. Müller-Beßler, and G. Stock. Generating locally optimal trajectories for an automatically driven car. *Optimization and Engineering*, 10(4):439–463, 2009.

[13] G. H. Golub and C. F. Van Loan. *Matrix computations. 4th ed.* Baltimore, MD: The Johns Hopkins University Press, 4th ed. edition, 2013.

[14] B. Houska, H. J. Ferreau, and M. Diehl. ACADO toolkit – an open source framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods*, 32(3):298–312, 2011.

[15] F. Kehrle, J. V. Frasch, C. Kirches, and S. Sager. Optimal control of formula 1 race cars in a VDrift based virtual environment. In S. Bittanti, A. Cenedese, and S. Zampieri, editors, *Proceedings of the 18th IFAC World Congress*, pages 11907–11912, 2011.

[16] K. Kritayakirana and C. J. Gerdes. Autonomous vehicle control at the limits of handling. *International Journal of Vehicle Autonomous Systems*, 10(4):271–296, 2012.

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

[18] T. Lipp and S. Boyd. Minimum-time speed optimisation over a fixed path. *International Journal of Control*, 87(6):1297–1311, 2014.

[19] J. Nocedal and S. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer, 2 edition, 2006.

[20] H. B. Pacejka. *Tyre and Vehicle Dynamics*. Elsevier, 2006.

[21] R. Quirynen, B. Houska, M. Vallerio, D. Telen, F. Logist, J. V. Impe, and M. Diehl. Symmetric algorithmic differentiation based exact Hessian SQP method and software for economic MPC. In *Proceedings of the IEEE Conference on Decision and Control (CDC)*, pages 2752–2757, 2014.

[22] R. Quirynen, M. Vukov, M. Zanon, and M. Diehl. Autogenerating microsecond solvers for nonlinear MPC: a tutorial using ACADO integrators. *Optimal Control Applications and Methods*, 36:685–704, 2014.

[23] A. Rucco, G. Notarstefano, and J. Hauser. An efficient minimum-time trajectory generation strategy for two-track car vehicles. *IEEE Transactions on Control Systems Technology*, 23(4):1505–1519, 2015.

[24] D. Verscheure, B. Demeulenaere, J. Swevers, J. D. Schutter, and M. Diehl. Time-optimal path tracking for robots: a convex optimization approach. *IEEE Transactions on Automatic Control*, 54:2318–2327, 2009.

[25] R. Verschueren, S. D. Bruyne, M. Zanon, J. V. Frasch, and M. Diehl. Towards time-optimal race car driving using nonlinear MPC in real-time. In *Proceedings of the IEEE Conference on Decision and Control (CDC)*, pages 2505–2510, 2014.

[26] A. Wächter. *An Interior Point Algorithm for Large-Scale Nonlinear Optimization with Applications in Process Engineering*. PhD thesis, Carnegie Mellon University, 2002.