Towards Time-Optimal Race Car Driving using Nonlinear MPC in Real-Time

Robin Verschueren, Stijn De Bruyne, Mario Zanon, Janick V. Frasch, Moritz Diehl

Abstract— **This paper addresses the real-time control of autonomous vehicles under a minimum traveling time objective. Control inputs for the vehicle are computed from a nonlinear model predictive control (MPC) scheme. The time-optimal objective is reformulated such that it can be tackled by existing efficient algorithms for real-time nonlinear MPC that build on the generalized Gauss-Newton method. We numerically validate our approach in simulations and present a real-world hardware setup of miniature race cars that is used for an experimental comparison of different approaches.**

I. INTRODUCTION

In the past few years, various advanced driver assistance systems (ADAS) based on conventional control schemes have been introduced in commercial passenger vehicles, such as semi-autonomous parking systems, autonomous cruise control, or last-second crash prevention systems. While these systems work rather well in their typical use case, challenging situations may require more sophisticated control algorithms, such as model-predictive control (MPC) [19], cf. [4], [8], [12]. Particularly for the task of fully autonomous driving (nonlinear) MPC permits the use of first-principle models that accurately predict the driving behavior even in extreme conditions.

Consequently, MPC for autonomous driving has received growing attention in the research community over the past years. Some earlier attempts were due to [9], [8], [13], [15], and include experimental validation. These approaches use various model simplification techniques to reduce the computational complexity. In [11], [23] the use of tailored nonlinear MPC algorithms was proposed to render real-time feasible significantly higher-fidelity vehicle models, featuring 14 differential states and a Pacejka-type tire model; however, only simulation results are provided. All these approaches have in common that they aim for reference-tracking, i.e., road-following objectives while satisfying certain safety constraints.

This research was supported by Research Council KUL: PFV/10/002 Optimization in Engineering Center OPTEC, GOA/10/09 MaNet and GOA/10/11 Global real- time-optimal control of autonomous robots and mechatronic systems. Flemish Government: FWO: PhD/postdoc grants; IWT: PhD Grants, projects: Eurostars SMART; Belgian Federal Science Policy Office: IUAP P7 (DYSCO, Dynamical systems, control and optimization, 2012-2017); EU: FP7-SADCO (MC ITN-264735), FP7-TEMPO(MC ITN-607957), ERC HIGHWIND (259 166).

R. Verschueren, M. Zanon, J. Frasch, and M. Diehl are with the Department of Electrical Engineering, KU Leuven, 3001 Leuven, Belgium robin.verschueren@student.kuleuven.be

S. De Bruyne is with LMS, A Siemens Business. LMS International NV, Interleuvenlaan 68, 3001, Leuven, Belgium.

R. Verschueren, M. Zanon and M. Diehl are also with the Department of Microsystems Engineering IMTEK, University of Freiburg, Georges-Koehler-Allee 102, 79110 Freiburg, Germany

In this paper, by contrast, we target time-optimal driving. Due to the natural antagonism of safety and speed in driving, MPC formulations that aim for minimum-time driving while satisfying limitations, such as the boundaries of a race track, seem particularly challenging. Additionally, minimum time objectives do not naturally fit into the class of least-squares objectives for which highly efficient algorithms exist, such as the ones used in [11], [23].

Time-optimal MPC approaches have received notable attention in the area of robotics for path following, cf., e.g. [22] and the references therein. There however, the geometric path to be tracked in a minimum-time fashion is predetermined, which renders the problem significantly simpler than the driving problem, where the exact path to be taken by the vehicle is part of the optimization.

Existing advanced control based attempts to minimum-time driving include [17], [20]. There, however, only offline open-loop solutions were computed. In [5] a cascading controller scheme was used, generating trajectory references from a simple geometric model and tracking these using a higher-detail model. In this paper, instead, we directly use a one-level approach, solving the nonlinear MPC problem with an economic cost function in real-time. The tight real-time bounds imposed on computational times make it necessary to reformulate the problem so as to allow for the use of efficient algorithms.

We provide real-world experimental results of the proposed nonlinear MPC scheme from a miniature race-car setup that is detailed in the paper. We propose this approach as a compromise for rapid prototyping of MPC controllers under more realistic conditions without the need for cost-intensive full-scale test vehicles and proving grounds.

The rest of the paper is organized as follows: in Section II, the vehicle model is described, based on previous work. The reformulation of the vehicle dynamics into spatially dependent coordinates is be developed in Section III, together with a corresponding time-optimal objective formulation. The algorithmic tools used for the solution of the nonlinear MPC problem in real-time are briefly reviewed in Section IV. In Section V the experimental setup with miniature race cars is described. The results of our method are presented in Section VI, and Section VII concludes the paper.

II. VEHICLE MODEL

In this paper, experimental results are obtained on a real-world setup (cf. Section V) with small-scale electric race cars. Due to the limited availability of model parameters for miniature cars (particularly regarding road-tire interaction)

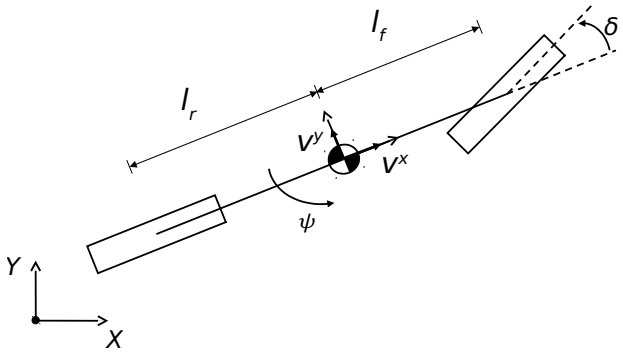

Fig. 1: Bicycle vehicle model.

Parameter	Unit	Physical meaning	Value
C_1	-	geometrical $(l_r/(l_r+l_f))$	0.5
C_2	m^{-1}	geometrical $(1/(l_r+l_f))$	17.06
C_{m_1}	m/s^2	motor parameter	12.0
C_{m_2}	1/s	motor parameter	2.17
C_{r_2}	1/m	second order friction parameter	0.1
C_{r_0}	m/s^2	zero order friction parameter	0.6

TABLE I: Bicycle Model Parameters

and due to the fact that the underlying physics differ slightly from passenger cars, we use a rather simple model, similar to the one used in [21]. In particular, road-tire interaction and load transfer are neglected in this paper, yielding a rather standard slip-free bicycle model, which we briefly describe in the following. The extentsion of the proposed work to higher fidelity models such as the one proposed in [23] is the object of ongoing research.

A. Time dependent car model

The chassis is modeled as a rigid body, described by the global position of its center of gravity in the X-Y plane, its global orientation ψ and its absolute velocity v, see Fig. 1. The model equations thus read:

$$\dot{X} = v\cos(\psi + C_1\delta), \tag{1a}$$
$$\dot{Y} = v\sin(\psi + C_1\delta), \tag{1b}$$
$$\dot{\psi} = v\,\delta\,C_2, \tag{1c}$$
$$\dot{v} = (C_{m_1} - C_{m_2}v)D - C_{r_2}v^2 - C_{r_0} - (v\,\delta)^2 C_2 C_1^2, \tag{1d}$$

where δ and D we denote the steering angle and the dutycycle applied to the DC motor, respectively. Parameters $C_\cdot$ are explained in Table I. Assuming a small steering angle δ, we used the following approximations, cf. [21], [18]:

$$v^x \approx v, \qquad v^y \approx v\,C_1\,\delta.$$

The car velocity is then given by $v^2 = (v^x)^2 + (v^y)^2$.

Equations (1a)-(1c) follow from the kinematics of the rigid body. The longitudinal dynamics in (1d) depend on the duty-cycle D and a commonly used approximation [14] for the resistive force, which consists of a constant and a second order term in the velocity.

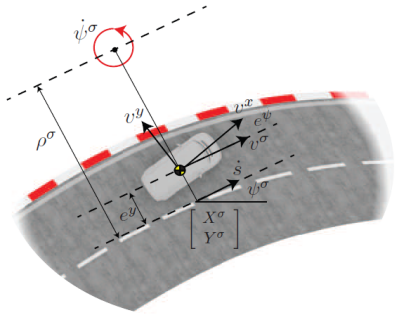

Fig. 2: Definition of the coordinate system used in the spatial reformulation of the vehicle dynamics. The s coordinate denotes the arc-length along the track. Source: [11].

B. Spatial Reformulation

Model (1) is a dynamic system with *time* being the independent variable. A reparameterization is therefore required to render time an optimization variable. We propose to employ the transformation to spatial coordinates from [11] for this purpose. Consequently, track limitations become simple (convex) state bounds, which are independent of the vehicle speed.

We state the spatial transformation for completeness. Vehicle coordinates in the global frame are denoted by $[X, Y]^T$. We project these $X-Y$ coordinates on a curve σ, given as a reference trajectory and parametrized by its arc length $\sigma(s)$. States X, Y and ψ can be replaced by

$$e^y = \cos(\psi^\sigma)(Y - Y^\sigma) - \sin(\psi^\sigma)(X - X^\sigma), \text{ and}$$
$$e^\psi = \psi - \psi^\sigma,$$

where $[X^\sigma, Y^\sigma]^T$ and ψ^σ are the position and orientation of the reference point on the path given by s, see Fig. 2. If we assume that the car is not at rest at any time instant ($\dot{s} > 0$), then it holds for the state vector $\xi = [e^y, e^\psi, v, t]$ that

$$\xi' = \frac{d\xi}{ds} = \frac{d\xi}{dt}\frac{dt}{ds} = \frac{1}{\dot{s}}\dot{\xi}, \tag{2}$$

where $\dot{\xi}$ is defined in (1).

In order to compute the vehicle speed with respect to the reference, $\dot{s}$, note that from Fig. 2 we have

$$v^\sigma = (\rho^\sigma - e^y)\,\dot{\psi}^\sigma, \text{ and}$$
$$v^\sigma = v^x\,\cos e^\psi - v^y\,\sin e^\psi,$$

where ρ^σ is the radius of local curvature of σ. The velocity along the path, $\dot{s}$, is then given by

$$\dot{s} = \rho^\sigma\,\dot{\psi}^s = \frac{1}{1 - \frac{e^y}{\rho^\sigma}}\,(v^x\cos(e^\psi) - v^y\sin(e^\psi)).$$

Note that $e^y < \rho^\sigma$ always needs to be fulfilled in order to guarantee uniqueness of the projection onto the centerline. This essentially means that the car needs to drive sufficiently close to the reference when the curvature is high.

TABLE II: STATES AND CONTROL INPUTS OF THE SPATIAL VEHICLE MODEL

State	Unit	Description
e^y	m	Deviation from centerline
e^ψ	rad	Yaw angle relative to path
v	m/s	Absolute velocity
t	s	Time

Control	Range	Unit	Description
δ	[-0.44, 0.44]	rad	Steering Angle
D	[-1,1]	-	Dutycycle of DC motor

We obtain the spatial dynamic system as:

$$e^{y\prime}(s) = (v\sin(e^\psi) + v\,C_1\,\delta\,\cos(e^\psi))/\dot{s} \tag{3a}$$
$$e^{\psi\prime}(s) = \dot{\psi}/\dot{s} - \kappa^\sigma(s) \tag{3b}$$
$$v'(s) = \dot{v}/\dot{s} \tag{3c}$$
$$t'(s) = 1/\dot{s}, \tag{3d}$$

where $\kappa^\sigma(s) = 1/\rho^\sigma(s)$ is the local curvature of the track. Note that the states in the global coordinate system can always be recovered by

$$\begin{aligned} X &= X^\sigma - e^y\,\sin(\psi^\sigma) \\ Y &= Y^\sigma - e^y\,\cos(\psi^\sigma) \\ \psi &= \psi^\sigma + e^\sigma. \end{aligned}$$

A summarizing list of the states and inputs of the spatial model can be found in Table II.

III. NONLINEAR MPC FORMULATION

A. Trajectory Tracking

For tracking a reference, the receding horizon optimal control problem subject to track and input limitations then reads:

$$\begin{aligned} \min_{\xi(\cdot),u(\cdot)} \quad & \int_{s_0}^{s_f} \|\xi(\tau) - \xi_{\text{ref}}(\tau)\|_Q^2 + \|u(\tau) - u_{\text{ref}}(\tau)\|_R^2\,\mathrm{d}\tau \\ & + \|\xi(s_f) - \xi_{\text{ref}}(s_f)\|_P^2 \\ \text{s.t.} \quad & \xi'(s) = f(s,\xi(s),u(s)) \\ & e^y(s) \in [e_L^y(s), e_U^y(s)] \\ & u(s) \in [\delta_L,\delta_U]\times[-1,1] \\ & \xi(0) = \xi_0, \end{aligned} \tag{4}$$

where $\xi = [e^y, e^\psi, v, t]^T$ is the state vector and $u = [\delta, D]^T$ is the control input vector. The interval $[s_0, s_f]$ is the prediction horizon and $\|\cdot\|$ is the Euclidean norm with weighting matrices Q, R, P, respectively. The road boundaries are denoted by $e_L^y(s), e_U^y(s)$. Note that the ODE system is stated in terms of its spatial derivatives $\xi' = \dfrac{d\xi}{ds}$, cf. (3).

B. Time-optimal formulation

For an approximate solution of the time-optimal driving problem, we aim at minimizing the time required for the race car to reach the end of the fixed-length spatial prediction horizon,

$$T = \int_{t_0}^{t_f} 1\,\mathrm{d}t = \int_{s_0}^{s_f} \frac{1}{\dot{s}(\tau)}\,\mathrm{d}\tau\ . \tag{5}$$

For prediction horizons which tend to infinity, this objective tends to the goal of driving time-optimally. As a consequence, long horizons are expected to yield a good approximation of the original problem in practice.

In order to allow for the use of efficient algorithms which rely on least-squares formulations and the generalised Gauss-Newton method, we propose to modify the objective formulation as follows.

Observation 1: Let all track data be fixed. Let T^* be the minimum time required by a vehicle that satisfies the dynamics and constraints of Problem (4). Then, for any $0 < T_{\text{ref}} < T^*$, the global optimum (if existent) of the optimization problem

$$\begin{aligned} \min_{\xi(\cdot),u(\cdot),T} \quad & \|T - T_{\text{ref}}\|_{q_{t_N}}^2 \\ \text{s.t.} \quad & \xi'(s) = f(s,\xi(s),u(s)) \\ & e^y(s) \in [e_L^y(s), e_U^y(s)] \\ & u(s) \in [\delta_L,\delta_U]\times[-1,1] \\ & \xi(0) = \xi_0, \end{aligned} \tag{6}$$

$(\hat{\xi}, \hat{u})$, satisfies $\hat{T} = T^*$.

By providing a suffciently small (i.e., infeasible) "target time" T_{ref} we can therefore have an approximate time-optimal MPC formulation in least-squares form. Note that, by fixing the sampling grid, the existence of a KKT point is given under rather mild conditions.

IV. SOLUTION METHOD

We adopt the software framework based on the ACADO Toolkit [1] that was already successfully used for high-speed nonlinear MPC of autonomous vehicles in [23].

In particular, we use the ACADO Code Generation tool [16], which is an open-source software environment that exports an instance of the *Real-Time Iteration Scheme*, which is based on Bock's multiple shooting method, (see [3], [6] for the algorithmic details) tailored to the model dynamics of the MPC problem. The exported solver is provided as self-contained plain ANSI C code, and includes an explicit Runge Kutta integrator of order 4, an efficient condensing algorithm [2], as well as a static memory version of the parametric QP solver qpOASES [10].

V. EXPERIMENTAL SETUP

One of the main contributions of this paper is the validation of the proposed methods on an experimental setup with small-scale race cars. A brief description of this setup is given in the following. In our experiments, we use the Kyosho dNaNo model race cars in their 2008 version. Measurements on the car's current position (X, Y), orientation (ψ) velocity (v) and yaw rate $(\dot{\psi})$ are obtained through a custom-made camera-based infrared sensing system with a

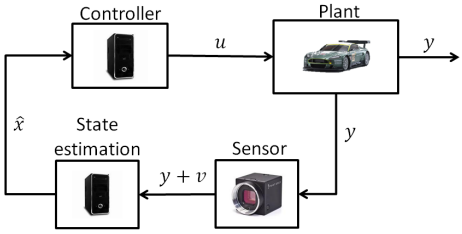

Fig. 3: Schematical view of the experimental setup.

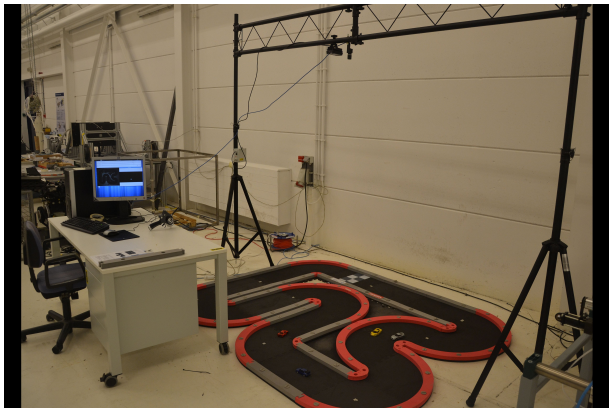

Fig. 4: The experimental setup.

sampling speed of 100 Hz. The feedback scheme of the setup is depicted in Fig. 3.

State measurements are obtained from a Kalman filter, which also accounts for possibly missing measurements that may arise from sensor defaulting due to external influences. The Kalman filter algorithm assumes a point mass vehicle model to estimate the position, orientation and velocity of the car.

The controller runs on a desktop computer featuring Debian LINUX 7.0 'Wheezy' with its *RT Preempt* real-time kernel patch installed that implements soft real-time guarantees.

The control inputs are sent to the car via a wireless connection. The dNano race cars have been extended with a custom communication module, such that the control signals can be sent over an Asynchrounous Connection-Less Bluetooth (ACL) communication link. Further technical details can be found in Appendix A. The considered race track features a chicane, a U-turn and a longer straight section with the car is travelling in clockwise direction. An overview of the setup is shown in Fig. 4.

VI. RESULTS

In this section, the performance of the trajectory tracking and the time-optimal driving formulation is compared, both in simulation and in real-world experiments.

A. Simulation

All simulations are carried out using the ACADO Code Generation tool in MATLAB. The same nominal dynamics were used for the simulation.

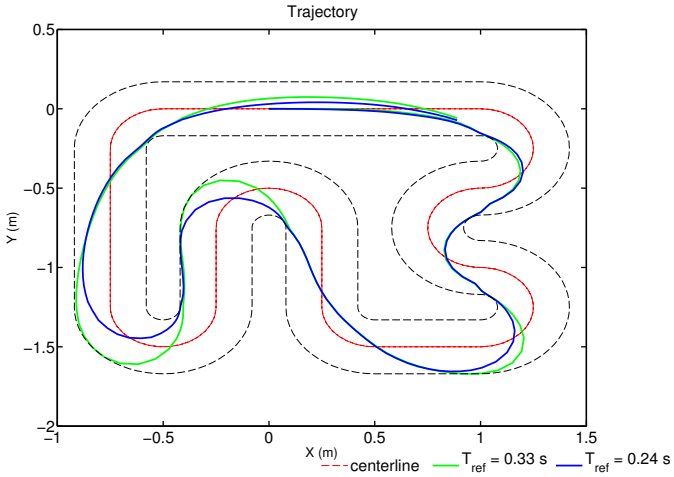

Fig. 5: Comparison of time-optimal driving at different target velocities.

a) Trajectory Tracking: The centerline of the track is taken as reference path. For the reference speed, three scenarios have been considered, $v_{\text{ref}} = 1.0\,\text{m/s}$, $2.0\,\text{m/s}$, and $4.0\,\text{m/s}$. The spatial prediction horizon is fixed to $1.0\,\text{m}$, composed of $N = 20$ intervals. The weighting matrices are chosen as follows:

$$Q = \text{diag}([q_{e^y}, q_{e^\psi}, q_v, q_t]) = \text{diag}([1, 0.01, 0.1, 0]),$$
$$R = \text{diag}([r_\delta, r_D]) = \text{diag}([10^{-4}, 10^{-4}]),\ P = Q$$

The centerline is tracked exactly at all selected reference velocities. This is due to the absence of model-plant mismatch, but also due to the simplicity of the model which lacks the influence of slip and thus is able to turn at an arbitrarily high speed.

b) Time Optimal Formulation: In this formulation, the objective tracks the reference on the final time t_{ref}. In order to improve the numerical reliability of the algorithms used, a tracking term is kept in the objective function with very small weights. The weighting matrices are given by

$$Q = \text{diag}([5 \cdot 10^{-4}, 10^{-10}, 10^{-10}, 10^{-10}]),$$
$$R = \text{diag}([10^{-3}, 10^{-10}]),$$
$$P = \text{diag}([10^{-10}, 10^{-10}, 10^{-10}, 1]).$$

The performance of the car with this formulation is shown in Fig. 5 for several laps, where it can be seen that the cost function allows the car to deviate from the centerline in order to minimize the time tracking error. As expected, the car now increasingly cuts the corners. As the target time T_{ref} is decreased, the car turns more aggressively (Fig. 5).

The performance in terms of speed is plotted in Fig. 6. We can see that for a target time $T_{\text{ref}} = 0.33\,\text{s}$ the vehicle speed is not tracking an average speed. The optimizer exploits the freedom to travel at higher speeds in some parts of the track to track the given time, regardless of tracking the centerline. Due to the engine-induced constraint on the maximum velocity of the model cars, this effect cannot be observed anymore for a target time of $0.24\,\text{s}$ (i.e. slightly

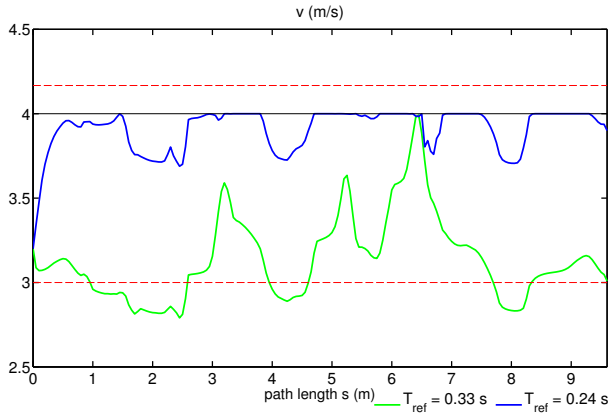

Fig. 6: Comparison of time-optimal speed profiles. The dashed lines are the velocities which correspond to the target times T_{ref} on a horizon of 1 m. Also the velocity constraint at 4m/s is shown.

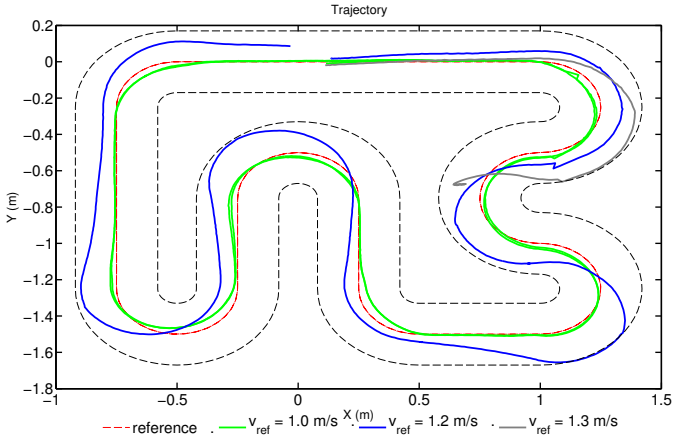

Fig. 7: Comparison of trajectory tracking on the experimental setup at different reference velocities.

infeasible target time), where the constraint becomes active thus limiting the freedom in choosing the vehicle speed.

B. Experimental Validation

Using the experimental setup described above, we assess the performance of the considered methods in a real-world environment.

a) Trajectory Tracking: The driving at different reference velocities is plotted in Fig. 7. Here, we use the same weighting matrices as in simulation. We can see that for a velocity of $v_{\text{ref}} = 1.0\,\text{m/s}$, the tracking is quite satisfactory, with only a larger deviation in the lower right hand corner. This is due to the fact that the real car is not able to turn as swiftly as the model. The discontinuities in the measurements come from the vision system, which is not always able to capture the car accurately. When measurements are missing, the Kalman filter provides a rough prediction of the position of the car, but once the car is seen again by the vision system, the position is corrected in a discontinuous fashion. At a slightly higher reference speed, $v_{\text{ref}} = 1.2\,\text{m/s}$, the car still manages to complete the lap, but gets dangerously close to the track boundary twice. At an even higher reference speed, the car crashes after the first corner.

b) Time Optimal Driving: To compare the time-optimal approach with trajectory tracking in experiments, we choose a target time of $T_{\text{ref}} = 0.24\,\text{s}$. The results are displayed in Fig. 8. Note that the time-optimal driving induces the car to drive on the inside of a curve when entering it and on the outside when exiting. The weights need to be tuned slightly in experiment: the new weights are $Q = \text{diag}([10^{-4}, 10^{-10}, 10^{-10}, 5 \cdot 10^{-3}]), R = \text{diag}([8 \cdot 10^{-5}, 10^{-4}]), P = \text{diag}([10^{-10}, 10^{-10}, 10^{-10}, 10^{2}])$.

The lap times of the tracking scheme is 9.11 s, while the one of the time-optimal approach is 8.21 s. Both times refer to periodic trajectories, which are obtained after an initial transient of several rounds.

Comparing the closed-loop trajectory with the predicted trajectory, one can see that the prediction performance of

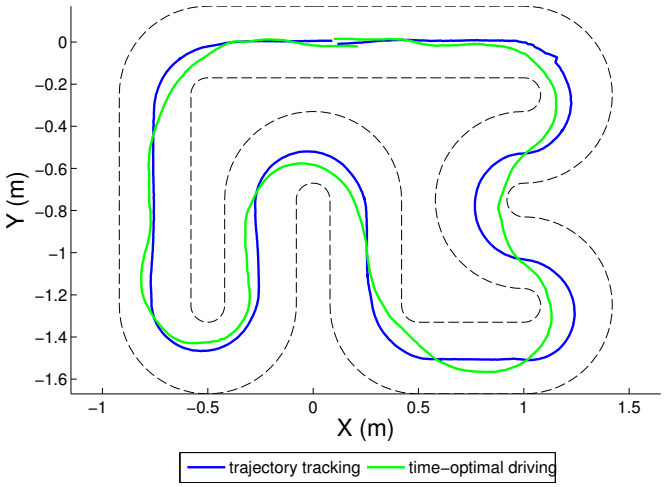

Fig. 8: Comparison of the performance of trajectory tracking and time-optimal driving on the experimental setup. Note that for tracking $v_{\text{ref}} = 1.0$ m/s and $T_{\text{ref}} = 0.24$ s for time-optimal driving.

the bicycle model (1) is quite satisfactory, see Fig. 9. Here, a close-up of the chicane, which is considered the most challenging part of the track, is shown.

The lap times and the computational times obtained with prediction horizons $N = 30$ and $N = 50$ are shown in Tab. III for different horizon lengths. It can be seen that, for the current setup, a long horizon $N = 50$ does not improve the lap time dramatically compared to a shorter horizon $N = 30$. Future work will aim at using higher fidelity models, for which a longer prediction horizon might be necessary. As the horizon length increases, the computational times increase dramatically. This is a known limitation of the condensing/qpOASES approach; the use of structure-exploiting sparse QP solvers, such as qpDUNES [**?**] or FORCES [7] are envisaged in the future to significantly reduce the computational times for long horizons.

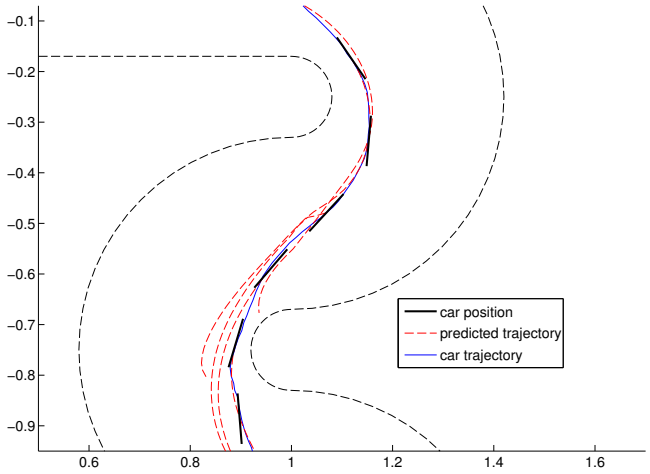

Fig. 9: Predicted vs. actual trajectory of the car for a time-optimal MPC formulation with a horizon of 1.0 m.

Prediction horizon length	Computational time (ms)	Lap time (s)
$N = 30$	1.8	8.3
$N = 50$	9.2	8.2

TABLE III: Computational times and resulting lap times for the time-optimal method for different horizon lengths.

VII. CONCLUSION

In this paper, we experimentally compared different time-optimal nonlinear MPC formulations based on least squares objectives in a setup of small-scale model race cars. We employed a nonlinear bicycle model in a reformulation to spatial coordinates to this end and proposed an infeasible-time-tracking objective for the best practical results.

The real-world experiments showed that this approach has potential, but the bicycle model proved insufficient at high velocities due to slip effects not being modelled. Ongoing research includes the transition to a higher fidelity model, like the one used in [23] and the use of structure exploiting QP solvers [**?**] that showed to be well suited for long prediction horizons.

APPENDIX A: TECHNICAL DETAILS OF THE LMS SETUP

[Infrared camera:] PointGrey Flea 3 Infrared Camera
Imaging: 1280x1024 at 100 FPS
Connection with computer: USB 3.0
[Computer:] Real-time Debian system
Intel Core i3-3220 @ 3.3 GHz
OS: Debian 7.0 'Wheezy'
Kernel: Linux 3.8.13 with Preempt RT Patch
Driver software: ni_pcimio from Comedi library
[Race Car:] Kyosho dNaNo FX-101 ASF2.4GHz System

REFERENCES

[1] ACADO Toolkit, 2009–2013. [Online; accessed 23-February-2009].
[2] Joel A. E. Andersson, Janick V. Frasch, Milan Vukov, and Moritz Diehl. A Condensing Algorithm for Nonlinear MPC with a Quadratic Runtime in Horizon Length. *Automatica*, 2013. Submitted.
[3] H.G. Bock and K.J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *Proceedings 9th IFAC World Congress Budapest*, pages 242–247. Pergamon Press, 1984.
[4] Francesco Borrelli, Paolo Falcone, Tamas Keviczky, and Jahan Asgari. Mpc-based approach to active steering for autonomous vehicle systems. *International Journal of Vehicle Autonomous Systems*, 3(2):265–291, 2005.
[5] F. Braghin, F. Cheli, S. Melzi, and E. Sabbioni. Race driver model. *Computers & Structures*, 86:1503–1516, July 2008.
[6] M. Diehl, H.G. Bock, J.P. 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.
[7] A. Domahidi, A. Zgraggen, M.N. Zeilinger, M. Morari, and C.N. Jones. Efficient Interior Point Methods for Multistage Problems Arising in Receding Horizon Control. In *IEEE Conference on Decision and Control (CDC)*, pages 668 – 674, Maui, HI, USA, December 2012.
[8] P. Falcone, F. Borrelli, J. Asgari, H.E. Tseng, and D. Hrovat. Low complexity MPC schemes for integrated vehicle dynamics control problems. 9^{th} *International Symposium on Advanced Vehicle Control*, 2008.
[9] Paolo Falcone, Francesco Borrelli, H Eric Tseng, Jahan Asgari, and Davor Hrovat. Integrated braking and steering model predictive control approach in autonomous vehicles. In *Advances in Automotive Control*, volume 5, pages 273–278, 2007.
[10] H. J. Ferreau, H. G. Bock, and M. Diehl. An online active set strategy to overcome the limitations of explicit MPC. *International Journal of Robust and Nonlinear Control*, 18(8):816–830, 2008.
[11] 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*, pages 4136–4141, 2013.
[12] Y. Gao, A. Gray, J. V. Frasch, T. Lin, E. Tseng, J.K. 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.
[13] Yiqi Gao, Theresa Lin, Francesco Borrelli, Eric Tseng, and Davor Hrovat. Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads. In *ASME 2010 Dynamic Systems and Control Conference*, pages 265–272. American Society of Mechanical Engineers, 2010.
[14] G. Genta. *Motor vehicle dynamics: modeling and simulation*. World Scientific, 1997.
[15] A. Gray, Y. Gao, T. Lin, J.K. Hedrick, E. Tseng, and F. Borrelli. Predictive control for agile semi-autonomous ground vehicles using motion primitves. *Proceedings American Control Conference*, 2012.
[16] B. Houska, H. J. Ferreau, and M. Diehl. An Auto-Generated Real-Time Iteration Algorithm for Nonlinear MPC in the Microsecond Range. *Automatica*, 47(10):2279–2285, 2011.
[17] 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.
[18] K. Popp and W.O. Schiehlen. *Ground Vehicle Dynamics*. Springer, 2010.
[19] J.B. Rawlings and D.Q. Mayne. *Model Predictive Control: Theory and Design*. Nob Hill, 2009.
[20] A. Rucco, G. Notarstefano, and J. Hauser. Computing minimum lap-time trajectories for a single-track car with load transfer. In *51st IEEE Conference on Decision and Control*, pages 6321–6326, 2012.
[21] P. Spengler and C. Gammeter. Modeling of 1:43 scale race cars. Master's thesis, ETH Zrich, 2010.
[22] D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, and M. Diehl. Time-Optimal Path Tracking for Robots: a Convex Optimization Approach. *IEEE Transactions on Automatic Control*, 54:2318–2327, 2009.
[23] M. Zanon, J. V. Frasch, M. Vukov, S. Sager, and M. Diehl. Model Predictive Control of Autonomous Vehicles. In *Proceedings of the Workshop on Optimization and Optimal Control of Automotive Systems*, pages 41–57. 2014.