Optimal control of Formula 1 race cars in a VDrift based virtual environment

Florian Kehrle * **Janick V. Frasch** * **Christian Kirches** *
Sebastian Sager *

* *Interdisciplinary Center for Scientific Computing (IWR), University of Heidelberg, Im Neuenheimer Feld 368, 69120 Heidelberg, Germany. (e-mail: [firstname.lastname]@iwr.uni-heidelberg.de)*

Abstract: Control of autonomous vehicles and providing recommendations to drivers in real time are challenging tasks from an algorithmic point of view. To include realistic effects, such as nonlinear tire dynamics, at least medium-sized mathematical models need to be considered. Yet, fast feedback is of utmost importance. Existing Nonlinear Model Predictive Control (NMPC) algorithms need to be enhanced to comply with these two contradictory requirements.
As the testing of algorithms in an automatic driving context is cumbersome and expensive, we propose a virtual testbed for NMPC of driving cars. We use the open source race simulator `VDrift` as virtual real world, in which algorithms need to cope with the mismatch between the detailed physical model in the simulator and a coarser approximative model used for NMPC.
We present the general framework of this virtual environment and an optimal control problem based on a medium-sized ordinary differential equation model and a generic and flexible parameterization of the track constraint. We discuss one possible algorithmic approach to the task of minimum time driving including gear shifts and give preliminary open loop numerical results for a Porsche on Germany's Formula One racing circuit Hockenheimring. This can be used as a reference against which other (closed loop) solutions can be compared in the future.

1. INTRODUCTION

Due to progress in control algorithms and computational power, automatic driving of cars and driving recommendations in real time are more than a theoretical idea. See, e.g., Falcone et al. [2007] for references to recent developments in car steering control. However, testing algorithms in such a context requires a lot of effort. Therefore we propose to use an open source race simulator, in our case the platform-independent software `VDrift` by Venzon [2010b], as a virtual testbed for control, in particular nonlinear model predictive control (NMPC) of driving cars.

Additionaly, we propose an optimal control based approach to calculate controls for time-optimal driving. We use it to calculate open loop controls for a Porsche on the Hockenheim ring as a proof of concept. The huge potential of the simultaneous optimization of driving trajectory and gear selection that was shown for energy-optimal driving in Terwen et al. [2004] or Hellström et al. [2009] indicates potential performance gains also for time-optimal driving.

The paper is organized as follows. In Section 2 we discuss the open source race simulation software `VDrift` that serves as a virtual test environment. In Section 3 we present a smaller-scale mathematical model that we propose to use to calculate controls. This includes a discussion of the track parameterization and the formulation of an open loop optimal control problem. In Section 4 we propose a solution approach, based on Bock's direct multiple shooting method and a partial outer convexification of the gear choice control function. Numerical results are given in Section 5. An outlook to future work concludes the paper.

2. A VDRIFT BASED VIRTUAL ENVIRONMENT

In order to provide a flexible and realistic simulation environment as a testbed for control algorithms, we make use of the cross-platform and open-source driving simulator `VDrift` by Venzon [2010b]. We chose `VDrift` among the variety of available car simulation packages for various reasons. It is a well-documented open source, cross-platform software with active support of the developers, in particular from the creator Joe Venzon himself. It provides a variety of currently about 30 fully modeled tracks and 30 car models. The `VDrift` developers put particular effort in its driving physics engine, which is inspired by and loosely based on the Vamos physics engine (see Vamos Team [2010]).

We extended the source code of `VDrift` to permit a direct or file-based exchange of data. This allows to couple automatic controllers to the simulation software. We intend to include this interface in future releases of `VDrift`. Unfortunately the `VDrift` documentation includes no mathematical car model. Although for our setting this is not necessary as algorithms are explicitly meant to cope with model uncertainty, we describe the most important relationships in the following that have been deduced from a manual interpretation of the source code. Parameters can be found in the source code, in `VDrift`'s "Documentation Wiki" by Venzon [2010a] and in the diploma thesis of Kehrle [2010].

2.1 Pacejka's Magic Formula Tire Model

For the dynamic behavior of a road vehicle, tire characteristics are of essential importance. However, accurate friction models of road and tire interfaces are very difficult to obtain. Most of today's racing simulations, just as professional tire research, use a version of Pacejka's so-called *Magic Formula* as the state-of-the-art in realistic tire modeling (cf. Pacejka [2006]).

The *Magic Formula* is a semi-empirical (based on measured data, but with physical structures) tire model to calculate steady-state tire force and moment characteristics. The forces are generated by the model as a result of different wheel angles and parameters. The main input variables are the camber angle γ, which is the inclination of the wheel to the vertical plane, the side slip angle α as the angle between the wheel's orientation and the actual direction of movement, the slip ratio $\kappa = \frac{\omega R_\mathrm{e} - v}{v}$ that is used as longitudinal slip while accelerating and braking and the normal load F_z that influences the grip of the tire on the road. The wheel's angular velocity is denoted by ω, the current wheel speed with respect to the ground by v and the effective tire rolling radius by R_e. The friction factor $f_\mathrm{s} \in [0,1]$ is a simplified way to model friction, depending on what surface the car is driving on.

As a result, three varying forces act on the wheel and accordingly affect the motion of the vehicle. The force F_x is directed in longitudinal direction. By pressing the throttle, the wheel speed increases and gets minimally higher than the current ground speed, so the car accelerates. If the wheel spins too fast, grip gets lost, resulting in less acceleration. For braking instead, the same force exists in opposite direction.

$$\begin{aligned}
F_\mathrm{x} &= D\ \sin(b_0\ \arctan(SB + E(\arctan(SB) - SB))) \\
B &= \frac{(b_3\ F_\mathrm{z} + b_4)\ e^{-b_5 F_\mathrm{z}}}{(b_1\ F_\mathrm{z} + b_2)\ b_0} \\
D &= (b_1\ F_\mathrm{z} + b_2)\ F_\mathrm{z}\ f_\mathrm{s} \\
E &= b_6\ F_\mathrm{z}^2 + b_7\ F_\mathrm{z} + b_8 \\
S &= 100\ \kappa + b_9\ F_\mathrm{z} + b_{10}
\end{aligned}$$

The lateral force F_y depends on the side slip angle, which describes the direction of the wheel compared to the actual vehicle direction on the ground.

$$\begin{aligned}
F_\mathrm{y} &= D\ \sin(a_0\ \arctan(SB + E(\arctan(SB) - SB))) + S_v \\
B &= \frac{a_3\ \sin\left(2 \arctan\left(\frac{F_\mathrm{z}}{a_4}\right)\right)\ (1 - a_5|\gamma|)}{a_0\ (a_1\ F_\mathrm{z} + a_2)\ F_\mathrm{z}} \\
D &= (a_1\ F_\mathrm{z} + a_2)\ F_\mathrm{z}\ f_\mathrm{s} \\
E &= a_6\ F_\mathrm{z} + a_7 \\
S &= \alpha + a_8\ \gamma + a_9\ F_\mathrm{z} + a_{10} \\
S_\mathrm{v} &= \big((a_{11}\ F_\mathrm{z} + a_{12})\ \gamma + a_{13}\big)\ F_\mathrm{z} + a_{14}
\end{aligned}$$

The self-aligning moment M_z acts on the steered wheel, trying to center the tire back to straight ahead driving.

$$\begin{aligned}
M_\mathrm{z} &= D\ \sin(c_0\ \arctan(SB + E(\arctan(SB) - SB))) + S_v \\
B &= \frac{\left(c_3\ F_\mathrm{z}^2 + c_4\ F_\mathrm{z}\right)\ (1 - c_6\ |\gamma|)\ e^{-c_5\ F_\mathrm{z}}}{c_0\ D} \\
D &= (c_1\ F_\mathrm{z} + c_2)\ F_\mathrm{z}\ f_\mathrm{s} \\
E &= \left(c_7\ F_\mathrm{z}^2 + c_8\ F_\mathrm{z} + c_9\right)\ (1 - c_{10}\ |\gamma|) \\
S &= \alpha + c_{11}\ \gamma + c_{12}\ F_\mathrm{z} + c_{13} \\
S_\mathrm{v} &= \left(c_{14}\ F_\mathrm{z}^2 + c_{15}\ F_\mathrm{z}\right)\ \gamma + c_{16}\ F_\mathrm{z} + c_{17}
\end{aligned}$$

2.2 `VDrift` *Engine Model*

An overview of the global functionality of an engine and its transmission of energy to the wheels within `VDrift` is illustrated in Figure 1. Within the diagram, blue diamonds illustrate the individual vehicle parts with their parameters in gray, rounded boxes. Controls are displayed in form of red circles. The green rectangles show the actual calculations of the vehicle parts, which combine the different input values.

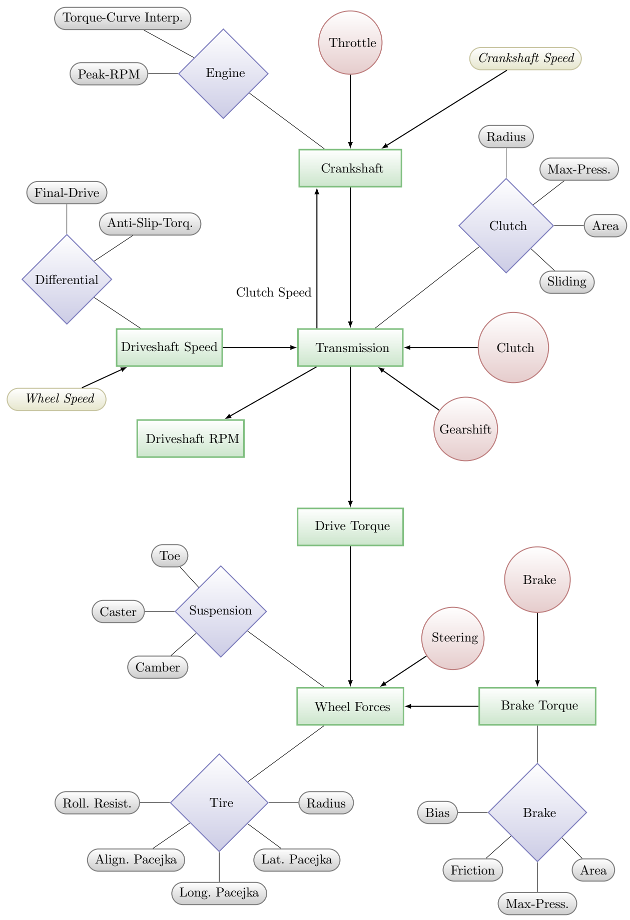


Fig. 1. Overview of `VDrift`'s car model

In order to accelerate a car, on pressing the throttle gas is injected into a piston and ignited. The amount of energy which is released by the combustion can be transmitted by the crankshaft from linear motion of the pistons into rotational motion. The engine is controlled by accelerator pedal position ϕ.

The total engine torque $M_\mathrm{E} = M_\mathrm{cb} + M_\mathrm{fr} - M_\mathrm{Cl}$ is applied to the crankshaft from combustion M_cb, internal friction M_fr, and the clutch M_Cl. To compute the accelerating combustion torque, a cubic spline interpolates torque curve $g(\nu_\mathrm{E})$ at the current number of revolutions-per-minute (*rpm*) of the crankshaft. Combined with the throttle it results in the combustion torque $M_\mathrm{cb} = \phi g(\nu_\mathrm{E})$. The engine's number of rpm comes with its angular velocity ω_E, implemented in `VDrift` as a modified Euler method, see Venzon [2010a], and $\nu_\mathrm{E} = \omega_\mathrm{E} \frac{30}{\pi}$. The peak engine speed is achieved at ω_p as $\omega_\mathrm{p} = \nu_\mathrm{p} \frac{\pi}{30}$. This leads to the friction factor f_E, which allies with the engine's angular velocity to friction torque $f_{\mathrm{E}} = \frac{g(\nu_{\mathrm{p}})}{\omega_{\mathrm{p}}^2}$ and $M_{\mathrm{fr}} = -1.3 \cdot 10^3 \omega_{\mathrm{E}} f_{\mathrm{E}}(1-\phi)$. Before we are able to calculate the total engine torque subject to the throttle position, current driving speed is necessary for backward computation of the clutch's angular velocity via the driveshaft. Assuming a rear wheel drive, the current driving speed can be computed from the average of the rear wheel's rotational velocity (ω_{rl}, ω_{rr}), differential ratio, and gear transmission ratio $\omega_{\mathrm{D}} = i_{\mathrm{t}} \frac{1}{2}(\omega_{\mathrm{rl}} + \omega_{\mathrm{rr}})$ and $\omega_{\mathrm{Cl}} = \omega_{\mathrm{D}} i_{\mathrm{g}}^{\mu}$. The clutch torque represents the friction that arises any time one side of the clutch is spinning faster than the other side. The sign of the friction depends on which side is spinning faster. If the engine speed is faster than the driveshaft, the clutch friction will generate torque to slow the engine and accelerate the wheels. Otherwise, if the engine speed is slower than the driveshaft, the clutch friction will generate torque to speed up the engine and slow the wheels,

$$M_{\mathrm{Cl}} = \chi(\omega_{\mathrm{E}} - \omega_{\mathrm{Cl}})(A_{\mathrm{Cl}} R_{\mathrm{Cl}} \cdot f_{\mathrm{Cl}} p_{\mathrm{Cl}}).$$

The actual force transmitted by resulting total engine torque, over driveshaft M_{Dr}, to the wheel, can be computed with the current gear transmission ratio. Finally this leads to the drive torque at rear right and left wheel M_{rr}, M_{rl} for a rear wheel driven car $M_{\mathrm{Dr}} = M_{\mathrm{E}} i_{\mathrm{g}}^{\mu}$ and $M_{\mathrm{rr}} = \frac{1}{2} i_{\mathrm{t}} M_{\mathrm{Dr}}$. The maximum brake torque is counteractive on the front wheels as $M_{\mathrm{Bf}} = A_{\mathrm{Bf}} R_{\mathrm{Bf}} \cdot f_{\mathrm{Bf}} \rho_{\mathrm{f}} p_{\mathrm{Bf}}$ and on the other as rear wheel torque M_{Br}, applied accordingly with its rear parameters. To get the current brake torque, the input control value of the brake pedal is required. In addition to effective rolling radius R_{e}, the braking force F_{B} at front and rear wheels is $F_{\mathrm{Bf}} = \frac{M_{\mathrm{Bf}} \xi}{R_{\mathrm{ef}}}$ and $F_{\mathrm{Br}} = \frac{M_{\mathrm{Br}} \xi}{R_{\mathrm{er}}}$.

3. OPTIMIZATION PROBLEM FORMULATION

In this Section we formulate a mathematical model that will be used within our optimization approach. It is based on previous work by Gerdts [2005, 2006], Sager et al. [2008b], Kirches et al. [2010]. Furthermore we introduce objective and constraint functions, in particular for track constraints. They are the basis for an optimal control problem that we will use to calculate controls that are used within the `VDrift` environment described in Section 2.

3.1 Car Model

The car model used in this setting is closely related to the one described in Kirches et al. [2010]. We model only a single front and rear wheel located in the virtual center between each original pair of wheels, neglecting any rolling or pitching of the car body against the track.

The four control functions are listed in Table 1). Note that the gear choice μ that influences the effective engine torque's transmission ratio is an integer control function.

Control	Range	Unit	Description
w_δ	$[-0.5, 0.5]$	$\frac{\mathrm{rad}}{\mathrm{s}}$	Steering wheel angular velocity
ξ	$[0, 1]$	–	Break pedal position
ϕ	$[0, 1]$	–	Accelerator pedal position
μ	$\{1, \ldots, n_\mu\}$	–	Selected gear

Table 1. Controls used in the car model.

The single-track dynamics are described by a system of ODEs, see one of the aforementioned publications for details. The individual system states are listed in Table 2. The car's center of gravity is determined by a

State	Unit	Description
d	m	Car's deviation from centerline
v	$\frac{\mathrm{m}}{\mathrm{s}}$	Magnitude of directional velocity of the car
δ	rad	Steering wheel angle
β	rad	Side slip angle
ψ	rad	Yaw angle
w_{z}	$\frac{\mathrm{rad}}{\mathrm{s}}$	Yaw angle velocity
t	s	Elapsed time

Table 2. States used in the car model.

coordinate pair (σ, d), where σ describes the car's progress on and d denotes its deviation from the track's centerline. This is different from prior modeling approaches (e.g., Gerdts [2005, 2006], Kirches et al. [2010]), which used a Cartesian coordinate description, yet necessary to allow arbitrary tracks. Only slight changes to the car dynamics are necessary: from a time-dependent ODE system (with states $x(\cdot)$ and controls $u(\cdot)$ as defined in Tables 1 & 2 and Section 3.3)

$$\frac{\mathrm{d}x}{\mathrm{d}t}(t) = f_t\Big(t, x(t), u(t), \mu(t)\Big). \tag{4}$$

as it can be found in Kirches et al. [2010], we obtain a position-dependent system simply by taking the inner derivative $\frac{\mathrm{d}t}{\mathrm{d}\sigma}(\sigma) = \frac{1}{v}$. It holds

$$\frac{\mathrm{d}x}{\mathrm{d}\sigma}(\sigma) = f_\sigma\left(\sigma, x(\sigma) \cdot \frac{1}{v(\sigma)}, u(\sigma), \mu(\sigma)\right). \tag{5}$$

By the coordinate system transformation described in Section 3.2, the centerline of the track is mapped onto the horizontal x-axis in the model from Kirches et al. [2010]. Consequently, the dynamics of c_{x} w.r.t. σ are redundant. The deviation d from the centerline is mapped onto c_y in the original model. Therefore, the car's yaw angle ψ has to be reduced by the track curvature at the current position. The new dynamics for $d(\cdot)$ read

$$\frac{\mathrm{d}d}{\mathrm{d}\sigma}(\sigma) = \sin\Big(\psi(\sigma) - curv(\sigma) - \beta(\sigma)\Big). \tag{6}$$

Finally, we introduce the elapsed time t as a new state of the ODE system:

$$\frac{\mathrm{d}t}{\mathrm{d}\sigma}(\sigma) = \frac{1}{v(\sigma)}. \tag{7}$$

3.2 Track Parametrization

As mentioned in Section 3.1, the car's position is determined by a pair (σ, d). In order to achieve high flexibility concerning the track choice, we propose a general coordinate system transformation first approximating the original track description by bicubic Bézier surfaces before extracting boundary curves, centerline curve and level curve (measuring altitude changes) as cubic Bézier splines. For a set of $(n+1)(m+1)$ control points $P_{i,j}$ given in Cartesian coordinates, the order (n, m) Bézier patch (again in Cartesian coordinates) reads as

$$C(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} P_{i,j}\ B_i^n(u)\ B_j^m(v) \tag{8}$$

with parameters $u, v \in [0,1]$ and Bernstein polynomials

$$B_i^n(u) = \binom{n}{i}\, u^i\, (1-u)^{n-i}. \qquad (9)$$

Accordingly, a bicubic Bézier patch is defined by 16 control points, whereas a cubic Bézier spline is defined by 4 control points. For a general discussion on Bézier spline interpolation, we refer to Stoer and Bulirsch [2007].

The patch boundaries' control points define a Bézier spline, approximating the boundary curves. Jointly they determine the track centerline as the Bézier spline defined by the arithmetic middle of the 2D projection of each pair of boundary control points. The track constraints now merely are lower and upper bounds on the deviation state d at each centerline control point. The level curve is computed analogously as 1D Bézier spline.

Note that the computation of the car's progress in longitudinal direction, u, as well as the computation of the deviation from the centerline d ultimately leads to the minimization of a polynomial of degree six in u, which has to be solved numerically, e.g., using Newton's method.

3.3 Optimal Control Problem

Summing up, the state vector x and the vector u of continuous controls read

$$x := \Big(d, v, \delta, \beta, \psi, w_{\mathrm{z}}, t\Big)^{\mathsf{T}}, \quad u := \Big(w_\delta, \xi, \phi\Big)^{\mathsf{T}}.$$

The integer controls are denoted by a separate vector $\mu(\cdot)$. If we use f to denote the right-hand-side function of the ODE system, the resulting mixed-integer optimal control problem reads

$$
\begin{aligned}
\min_{x(\cdot),u(\cdot),\mu(\cdot)} \quad & t(\sigma_{\mathrm{f}}) + \int_{\sigma_0}^{\sigma_{\mathrm{f}}} w_\delta^2(\sigma)\, \mathrm{d}\sigma && (10a)\\
\text{s.t.} \quad & \dot{x}(\sigma) = f\big(\sigma, x(\sigma), u(\sigma), \mu(\sigma)\big) && (10b)\\
& d(\sigma) \in \big[-P_{\mathrm{u}}(\sigma) + \tfrac{B}{2}, P_{\mathrm{u}}(\sigma) - \tfrac{B}{2}\big] && (10c)\\
& n_{\mathrm{eng}}^{\min} \le n_{\mathrm{eng}}(v(\sigma), \mu(\sigma)) \ \le n_{\mathrm{eng}}^{\max} && (10d)\\
& w_\delta(\sigma) \in [-0.5, 0.5] && (10e)\\
& \xi(\sigma) \in [0,1] && (10f)\\
& \phi(\sigma) \in [0,1] && (10g)\\
& \mu(\sigma) \in \{1, \ldots, n_\mu\} && (10h)\\
& x(\sigma_0) = \big(\text{free}, 10, 0, 0, 0, 0, 0\big)^{\mathsf{T}} && (10i)\\
& \psi(\sigma_{\mathrm{f}}) = 0 && (10j)
\end{aligned}
$$

for $\sigma \in [\sigma_0, \sigma_f]$. The objective function (10a) strives for minimization of the loop time $t(\sigma_{\mathrm{f}})$ while keeping the steering effort $w_\delta(\sigma)$ minimal. At any time, the full car of width B must be positioned within the track course's boundaries (cf. (10c)) and the engine speed is bounded (10d). Note that it is sufficient to use w.l.o.g. only the upper bound due to symmetry of upper and lower bound at the centerline. The system's initial values are fixed in (10i); the car's initial vertical position on the track however remains a free and is only subject to track boundaries. At the end of the track, straight ahead driving is guaranteed by (10j).

4. GENERAL SOLUTION FRAMEWORK

We propose to use the optimal control problem from Section 3 to obtain controls. We start with an open loop offline solution and intend to extend this in future work to feedback based NMPC.

4.1 The Direct Multiple Shooting Method

Neglecting the integer control $\mu(\cdot)$ for the moment, the optimal control problem (10) can be solved using Bock's direct multiple shooting method (see Bock and Plitt [1984]), which transforms the optimal control problem into a finite dimensional optimization problem. This is achieved by a reduction of the feasible control space to a finite-dimensional one using basis functions with local support, and a relaxation of the path constraints to a finite time grid. A highly structured nonlinear program (NLP) is obtained that is solved by a tailored sequential quadratic programming (SQP) method. This includes an extensive exploitation of the arising structures, in particular using block-wise high-rank updates and condensing for a reduction of the size of the quadratic problems (QP) to that of a single-shooting method. For more details see Bock and Plitt [1984], Leineweber et al. [2003].

We use an efficient implementation of this method, the optimal control software package MUSCOD-II, for computational results.

4.2 Convex Relaxation of Integer Controls

We partially convexify the original optimal control problem with respect to the integer control functions $\mu(\cdot)$ as first suggested in Sager [2005]. We assign one control function $\omega_i(\cdot)$ to every possible control choice $\mu^i \in \{1, \ldots, n_\mu\}$ and obtain

$$\dot{x}(\sigma) = \sum_{i=1}^{n_\mu} f(x(\sigma), \mu^i, u(\sigma))\, \omega_i(\sigma) \qquad (11)$$

instead of (10b) and

$$\omega(\sigma) \in \{0,1\}^{n_\mu} \qquad (12a)$$

$$1 = \sum_{i=1}^{n_\mu} \omega_i(\sigma) \qquad (12b)$$

instead of (10h). There is a bijection $\mu(t) = \mu^i \Leftrightarrow \omega_i(t) = 1$ between the solutions of the original and the partially convexified problem, see Sager [2005]. The relaxation of the latter is obtained by replacing the constraint (12a) by

$$\omega(\sigma) \in [0,1]^{n_\mu}. \qquad (13)$$

This formulation has two main advantages over other methods for the calculation of integer solutions for mixed-integer optimal control problems that suffer from a combinatorial explosion when the number of discretized binary control variables increases. First, for many optimal control problems the optimal solution will have a bang–bang character, therefore the solution of the relaxed problem will yield the optimal integer solution. Second, for cases in which path-constraints or sensitivity-seeking arcs occur, the integer gap can be made arbitrarily small by refinement of the underlying control discretization grid. We recommend to use a sum up rounding strategy as developed in Sager [2005] in combination with a switching time optimization approach (cf. Kirches et al. [2010]). Details can be found, e.g., in Sager et al. [2009].

5. NUMERICAL RESULTS

As a proof of concept, we compute open loop optimal controls for a Porsche 911 Club Sport (Porsche CS) stock car and the Formula One racing circuit Hockenheimring, Germany. Numerical data for both are included in `VDrift`. The Porsche CS, built in the years 1987 through '89, is a licensed street car, but the CS-series come with a racing engine. Car specific parameters are listed in Kehrle [2010]. The circuit's length measures $\sigma_f = 4574m$ and is implemented in `VDrift` by a set of 7776 Bézier points depicted in Figure 2, together with the calculated trajectory.

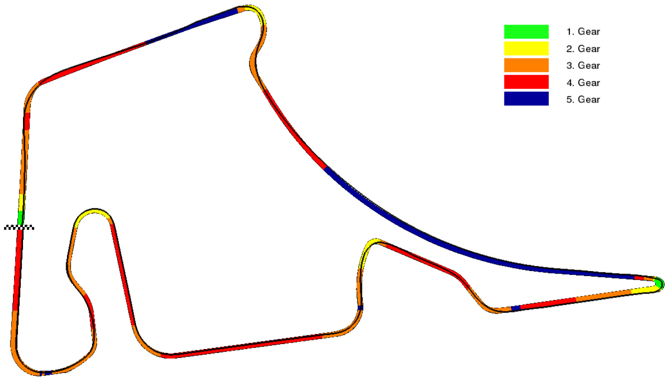


Fig. 2. Track data of Hockenheimring racing circuit, given as Bézier points in `VDrift` with a suboptimal open loop solution: gear choice and trajectory.

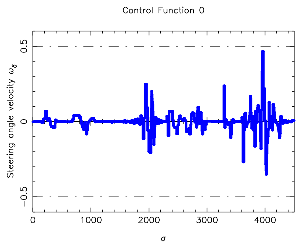
The Porsche CS features a five-speed gearbox ($n_\mu = 5$), which is to be controlled manually, i.e., by the optimization algorithm. For other car-specific parameters, we refer to Kehrle [2010] or the `VDrift` implementation and documentation Venzon [2010b].

Computing optimal controls for the described setting is significantly more complex than for the settings considered in Gerdts [2005, 2006], Sager et al. [2008b], Kirches et al. [2010]. This is due to the length of the track and difficult track parts like hairpin curves. Every second Bézier patch was chosen as a Multiple Shooting grid point, resulting in 324 shooting intervals. Note that naturally the Bézier points are placed more densely in difficult areas. We applied a homotopy to generate start values for the optimization.

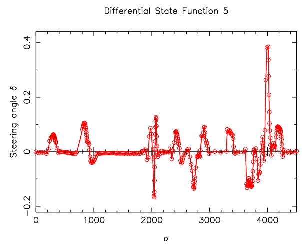
Optimization starts as the car crosses the scratch line, using straight ahead driving at full acceleration in first gear as default control up to that point. Thus, the initial values for the Porsche CS on Hockenheimring racing circuit are given by $v_0 = 11.4509\frac{\mathrm{m}}{\mathrm{s}}$, $d = 4.05983\mathrm{m}$ and $\psi_0 = -1.55632$, 0 for all other states.

Figures 3 through 5 show trajectory details corresponding to Figure 2. As can be seen, the gear choice control functions are often bang-bang and hence integer feasible. On some arcs procedures as described in Sager et al. [2009], Sager [2009] need to be applied. However, at the price of a possible frequent switching, the objective function value of the relaxed problem can always be obtained with an integer solution, Sager et al. [2011]. For illustration we depict only the not-yet integer trajectory. The resulting lap time for the relaxed solution is $t = 112.7175\mathrm{s}$.

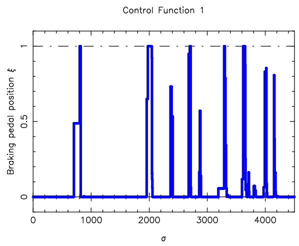
It is beyond the scope of this paper to discuss the details of the actuators and trajectory in Figures 3 to 5. Instead, we refer to a video visualization that can be found on the webpage `http://mathopt.de/RESEARCH/automotive.php`.



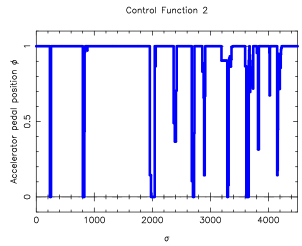
(a) Steering angle velocity ω_δ.



(b) Steering angle δ.

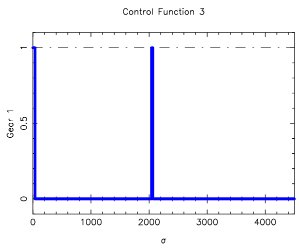


(c) Braking pedal position ξ.

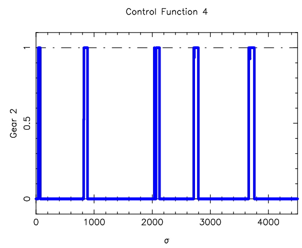


(d) Acceleration pedal ϕ.

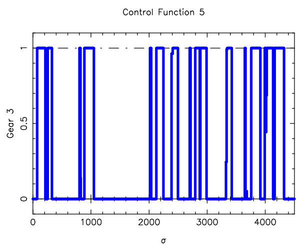
Fig. 3. Optimal steering angle velocity, steering angle, braking and acceleration given as differential state of the optimal integer solution for the Porsche CS.



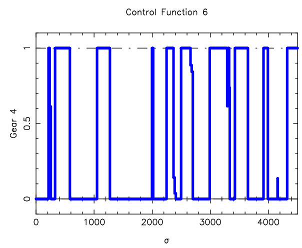
(a) 1^{st} gear.



(b) 2^{nd} gear.



(c) 3^{rd} gear.



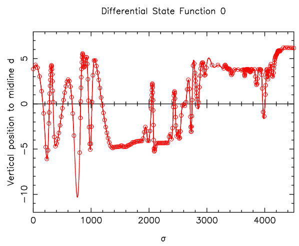
(d) 4^{th} gear.

Fig. 4. Relaxed gear choice functions as in Section 4.2: control is 1 whenever the gear is active. Gear 5 is 1 whenever all others are 0, compare (12b).

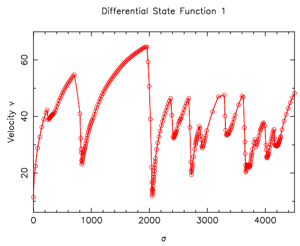
When we apply the controls to a Porsche in our virtual testbed `VDrift`, the model mismatch leads to a violation of the path constraint and a subsequent crash already in the first curve, compare Figures 2 and 6. Future work will hence include feedback.

6. DIRECTIONS OF FUTURE RESEARCH

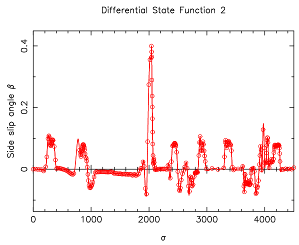
We presented a testbed for control algorithms and an open loop solution for a Porsche on Hockenheimring. We observed the expected model-plant mismatch effect for the



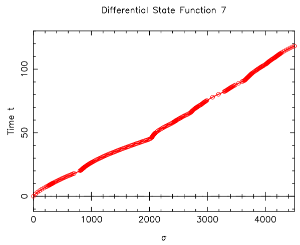
(a) Vertical position d.



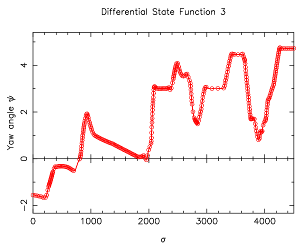
(b) Velocity v.



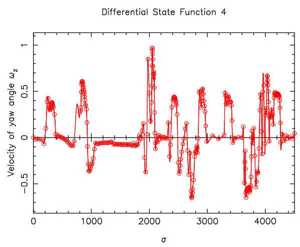
(c) Side slip angle β.



(d) Time t.



(e) Yaw angle ψ.



(f) Yaw angle ω_z.

Fig. 5. Differential states of the open loop trajectory for the PORSCHE CS.

Fig. 6. Application of the open loop controls to the leading red car in `VDrift` leads to violation of track constraints, making the need to include feedback information obvious.

calculated controls: the car crashes shortly after the first curve. Future work will focus on the inclusion of feedback in an NMPC context. To achieve real-time capability we want to evaluate the potential of multi-level optimization approaches with different levels of model accuracy that are intertwined in one algorithm. A comparison with other control algorithms, e.g., based on linearizations, is expected to give further insight and shall be performed within the `http://embocon.org` EU FP7 project.

ACKNOWLEDGEMENTS

This work was partially supported by the Heidelberg Graduate School of Mathematics and Computational Methods for the Sciences. We acknowledge funding from the EU Seventh Framework Programme FP7/2007-2013 under grant agreement n° FP7-ICT-2009-4 248940. We thank Joe Venzon for providing `VDrift` and active support.

REFERENCES

H.G. Bock and K.J. Plitt. A Multiple Shooting algorithm for direct solution of optimal control problems. In *Proceedings of the 9th IFAC World Congress*, 243–247, Budapest, 1984. Pergamon Press.

P. Falcone, F. Borrelli, J. Asgari, H.E. Tseng, and D. Hrovat. Predictive active steering control for autonomous vehicle systems. *IEEE Trans. on Control Systems Techn.*, 15(3):566–580, 2007. ISSN 1063-6536.

M. Gerdts. Solving mixed-integer optimal control problems by Branch&Bound: A case study from automobile test-driving with gear shift. *Optimal Control Applications and Methods*, 26:1–18, 2005.

M. Gerdts. A variable time transformation method for mixed-integer optimal control problems. *Optimal Control Applications and Methods*, 27(3):169–182, 2006.

E. Hellström, M. Ivarsson, J. Aslund, and L. Nielsen. Look-ahead control for heavy trucks to minimize trip time and fuel consumption. *Control Engineering Practice*, 17:245–254, 2009.

F. Kehrle. Optimal control of vehicles in driving simulators. Diploma thesis, Uni Heidelberg, March 2010.

C. Kirches, S. Sager, H.G. Bock, and J.P. Schlöder. Time-optimal control of automobile test drives with gear shifts. *Optimal Control Applications and Methods*, 31 (2):137–153, March/April 2010.

D.B. Leineweber, I. Bauer, A.A.S. Schäfer, H.G. Bock, and J.P. Schlöder. An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimization (Parts I and II). *Computers and Chemical Engineering*, 27:157–174, 2003.

H. B. Pacejka. *Tyre and Vehicle Dynamics*. Elsevier Ltd., Oxford Burlington, 2nd edition, 2006.

S. Sager. *Numerical methods for mixed–integer optimal control problems*. Der andere Verlag, Tönning, Lübeck, Marburg, 2005. ISBN 3-89959-416-9.

S. Sager. Reformulations and algorithms for the optimization of switching decisions in nonlinear optimal control. *Journal of Process Control*, 19(8):1238–1247, 2009.

S. Sager, C. Kirches, and H.G. Bock. Fast solution of periodic optimal control problems in automobile test-driving with gear shifts. In *47th IEEE CDC, Cancun, Mexico*, 1563–1568, 2008. ISBN: 978-1-4244-3124-3.

S. Sager, G. Reinelt, and H.G. Bock. Direct methods with maximal lower bound for mixed-integer optimal control problems. *Mathematical Programming*, 118(1):109–149, 2009.

S. Sager, H.G. Bock, and M. Diehl. The integer approximation error in mixed-integer optimal control. *Mathematical Programming*, 2011. doi: 10.1007/s10107-010-0405-3.

J. Stoer and R. Bulirsch. *Numerische Mathematik 1*. Springer-Verlag, Berlin Heidelberg New York, 10th edition, 2007.

S. Terwen, M. Back, and V. Krebs. Predictive powertrain control for heavy duty trucks. In *Proceedings of IFAC Symposium in Advances in Automotive Control*, 451–457, Salerno, Italy, 2004.

Various Authors. Vamos physics engine. `http://vamos.sourceforge.net/`.

Various Authors. VDrift Documentation Wiki. `http://wiki.vdrift.net/Numerical_Integration`.

J. Venzon. VDrift Website. `http://vdrift.net`.