Ensuring Drivability of Planned Motions from Simple Models Using Formal Methods

Bastian Schürmann, Daniel Heß, Jan Eilbrecht, Olaf Stursberg, Frank Köster, and Matthias Althoff

***Abstract*—Motion planning of automated vehicles requires dynamical models to ensure that obtained trajectories are drivable. An often overlooked aspect is that usually simplified models are used for motion planning, which do not always sufficiently conform to the real behavior of vehicles. Thus, collision avoidance and drivability is not necessarily ensured. We address this problem by modeling vehicles as differential inclusions composed from simple dynamics plus set-based uncertainty; conformance testing is used to determine the required uncertainty. To quickly provide the set of solutions of these uncertain models, we provide pre-computed reachable sets (i.e. union of all possible solutions) for pre-selected motion primitives. The reachable sets of vehicles are obtained by a novel combination of optimization techniques and reachability analysis – they enable us to guarantee safety by checking their mutual non-intersection for consecutive time intervals. The benefits of our approach are demonstrated by numerical experiments.**

I. Introduction

Motion planning is one of the key technologies for automated driving [1], requiring efficient approaches for reacting timely to changes in traffic. The computation time in motion planning significantly depends on the underlying model chosen to generate feasible trajectories [2]: while a detailed, general problem formulation may be computationally intractable in real-time due to its nonlinear, non-convex nature, simple models require much less computation time, but ensure feasibility to a lesser extend since not all aspects of the vehicle dynamics are considered.

In this work, we address exactly this problem: By computing motion plans from a simple point-mass model we ensure efficient computation, while ensuring drivability by utilizing formal methods, namely *reachability analysis* and *conformance testing*. In the following, we review previous work on a) motion planning of automated road vehicles, b) formal techniques to ensure that an uncertain vehicle model (including disturbances, sensor noise, and uncertain parameters) can follow a planned maneuver, and c) conformance testing methods to ensure that the used uncertain model contains all behaviors recorded from test drives of the real vehicle. Those test drives are performed before the deployment of our approach and we do not consider adapting the model online in this work.

a) Motion planning: The last decades have witnessed significant progress in the development of motion planning algorithms both for general problems [3] and the field of autonomous driving in particular [4]. Existing approaches can be classified according to the complexity of the dynamical system employed for planning: Detailed models which account for, e. g., kinematic constraints or nonlinear tire dynamics, require solution methods which may not be applicable in real-time and lack guarantee of convergence to a global optimum. These methods comprise graph-search in a discretized state space of a nonlinear model [5], sampling-based approaches such as RRT* [6], nonlinear model-predictive control [7], and optimal control procedures like hp-adaptive collocation [8]. In contrast to these detailed models, a large group of approaches uses much simpler, often linear vehicle models with point mass properties, decoupling longitudinal and lateral dynamics, and not enforcing kinematic constraints explicitly [9], [10], [11], [12]. While these models give only a coarse representation of a real vehicle's dynamics, they allow the application of efficient planning algorithms with stronger theoretical guarantees.

b) Formally ensuring drivability: Despite the wide use of simplistic vehicle models, there has been very limited research in formally proving drivability of planned paths. There exist some works that formally prove correctness, but only specific aspects of automated driving are considered. An approach for safely entering an intersection is presented in [13]. In [14], automated cruise control is formally verified by automated theorem proving. Automated theorem proving has been applied to overtaking maneuvers in [15] under the assumption of perfect knowledge on the environment. A verified synthesis for driving assistance in traffic merging is presented in [16].

The problem of ensuring that a vehicle can follow a desired trajectory has mostly been addressed by the authors themselves in previous work. To our best knowledge, online verification of drivability has first been performed theoretically for cooperating vehicles in [17], for mixed traffic in [18], and on a real vehicle in [19]. A much simpler approach for precomputing drivable maneuvers (so called maneuver primitives) compared to this work, are presented in our previous work [20].

c) Formal controller design: Computing controllers which provide formal guarantees for the satisfaction of state and input constraints despite disturbances, measurement noise, and nonlinear dynamics is a hard task. Using optimal control online for disturbed systems is done in tube model predictive control [21], [22]. However, these methods work mainly for

B. Schürmann and M. Althoff are with the Department of Informatics, Technische Universität München, Boltzmannstr. 3, 85748 Garching, Germany (e-mail: {bastian.schuermann, althoff}@tum.de)

D. Heß, F. Köster are with the Institute of Transportation Systems, German Aerospace Center (DLR), Braunschweig, Germany (e-mail: {daniel.hess, frank.koester}@dlr.de)

J. Eilbrecht and O. Stursberg are with the Control and System Theory Group, Department of Electrical and Computer Science, Universität Kassel, 34121 Kassel, Germany (e-mail: {jan.eilbrecht, stursberg}@uni-kassel.de)

linear systems. Abstraction based control [23] is a formal control approach which is able to guaranteeing complex specifications. Since this is often achieved by discretizing the state and input space, these approaches are limited to small dimensional systems. In [24], the authors use sums-of-squares programming to compute so-called LQR-trees to control sets of initial states. While this approach can take nonlinear dynamics as well as disturbances into account, the computational effort of sum-of-squares methods grows very fast with the system dimension which limits this approach.

Recently, we proposed several new controller design approaches [25], [26], [27] which optimize over sets of solutions and guarantee constraint satisfaction even for disturbed nonlinear systems by incorporating reachability analysis in the controller design. While [25], [26] interpolate open-loop trajectories and therefore result in non-continuous control laws, the approach in [27] directly optimized over the closed-loop dynamics. In the previous work, we restricted the approach to disturbed, linear systems, while we apply it in this work for the first time to disturbed, nonlinear systems.

d) Conformance testing: conformance testing is a systematic process for finding whether the real behavior of a system and its mathematical model fulfill a conformance relation [28, p. 30]. We use conformance testing to find the set of possible deviations between a model and a real system, which are added as disturbances and sensor noise to the corresponding model so that it captures all real behaviors. There exists rather limited work for conformance testing of automated vehicles. Although work exists for general hybrid systems [29], this work does not quantify the difference between the real system and the model. In [30], rapidly-exploring random trees are used for the test generation, but the compliance with a specification is investigated rather than quantifying the model error. First works on rigorously bounding model errors for systems with continuous dynamics have been developed by the authors themselves: In [31] methods have been developed to create simple models plus uncertainty that capture all behaviors of complex models and in [32] this concept has been generalized for arbitrary dynamical systems.

e) Concluding remark and paper organization: To the best of our knowledge, no previous work formally checks the feasibility of motions planned by a simplified model.

We begin with an overview in Sec. II. We continue by describing the three main parts of our combined approach, starting with the reference trajectory in Sec. III, followed by the controller design in Sec. IV, and concluded by the conformance checking in Sec. V. In Sec. VI, we show the results of our approach for a numerical example based on measured data from real test drives.

II. Overview

An overview of our proposed approach is illustrated in Fig. 1. As previously motivated, we use simple models for motion planning of collaborative vehicles to save computation time. In particular, we use a simple point-mass model in this work, but any other model would also be feasible.

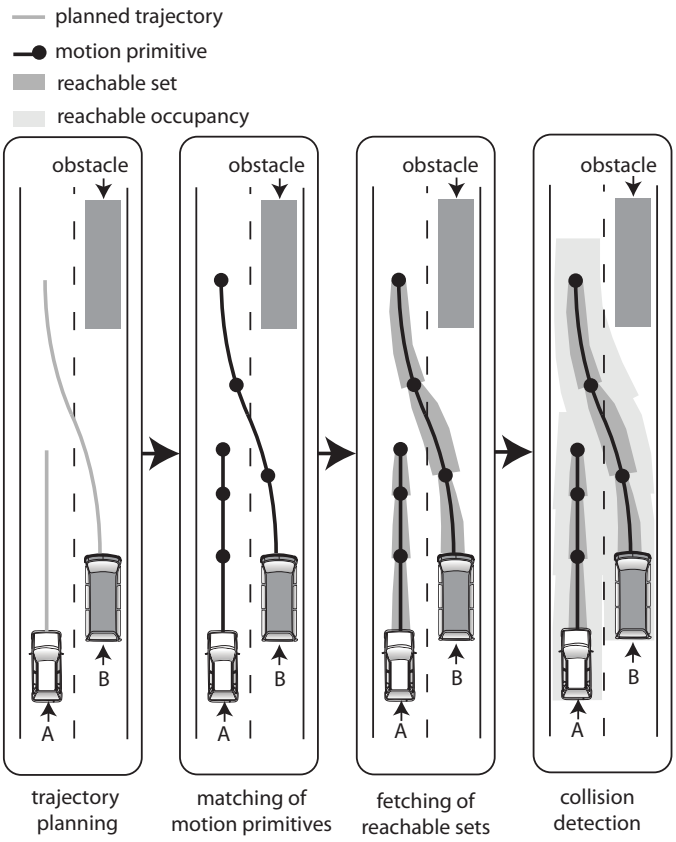

Fig. 1. Checking drivability from left to right: a) trajectory planning of vehicles (here: vehicle A and B), b) matching motion primitives with the obtained trajectories, c) fetching pre-computed reachable sets and reachable occupancies (reachable occupancies not yet displayed), d) collision checking by using reachable occupancies of consecutive time intervals.

Since the dynamics of a real vehicle does not exactly match the one of a point mass model, we prove whether the planned maneuver is realizable by a non-deterministic system modeled as a differential inclusion. To properly define this model, we introduce the state $x \in \mathbb{R}^n$, the control input $u \in \mathbb{R}^m$, the set of disturbances $\mathcal{W} \subset \mathbb{R}^q$. The differential inclusion of our uncertain model is

$$\dot{x} \in \{f(x, u, w) | w \in \mathcal{W}\}, \tag{1}$$

where $\mathcal{W}$ is obtained from conformance testing as detailed in Sec. V. The state of the system is defined to be measured by a function h, with a measurement vector $y \in \mathbb{R}^o$ and a set of measurement errors $\mathcal{V} \subset \mathbb{R}^o$:

$$y \in \{h(x, u, \nu) | \nu \in \mathcal{V}\} \tag{2}$$

We denote a possible solution of (1) by $\gamma(t, x(0), u(\cdot))$, where $x(0) \in \mathbb{R}^n$ is the initial state and $u(\cdot)$ is an input trajectory. Unlike a differential equation, a differential inclusion has infinitely many solutions that can be bounded by its reachable set starting from the set of possible initial states $x(0) \in \mathcal{X}_0$:

$$\mathcal{R}^e(t, \mathcal{X}_0, u(\cdot), \mathcal{W}) := \Big\{ \gamma(t, x(0), u(\cdot)) \Big| x(0) \in \mathcal{X}_0, \\ \forall \tau \in [0, t] : w(\tau) \in \mathcal{W} \Big\}.$$

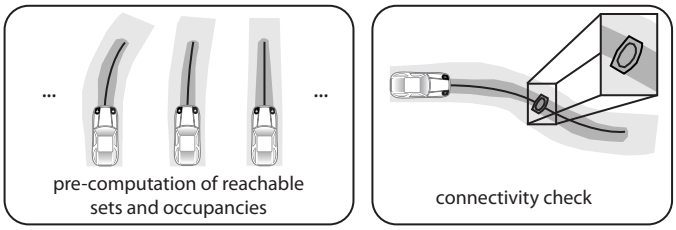

Fig. 2. Pre-computation of reachable sets and reachable occupancies for all motion primitives. It is checked which motion primitives can be combined (last reachable set of preceding motion primitive has to be enclosed in intial set of the proceeding motion primitive)

The superscript e on $\mathcal{R}^e(t)$ denotes the exact reachable set, which cannot be computed for arbitrary nonlinear systems [33]. For this reason, we aim to compute overapproximations $\mathcal{R}(t) \supseteq \mathcal{R}^e(t)$ which are as accurate as possible. From now on, we often only say *reachable set* when referring to an *overapproximative reachable set* to simplify the wording. To check whether the movement of an uncertain vehicle model is collision-free, we also have to define the reachable occupancy. After introducing the mapping $\Gamma(x) : \mathbb{R}^n \rightarrow P(\mathbb{R}^2)$ ($P()$ denotes the power set), which maps a state x to a subset of $\mathbb{R}^2$ representing the occupancy of the vehicle, the reachable occupancy is defined as

$$\mathcal{O}(t, \mathcal{X}_0, u(\cdot), \mathcal{W}) := \Big\{ \Gamma(x) \Big| x \in \mathcal{R}(t, \mathcal{X}_0, u(\cdot), \mathcal{W}) \Big\}.$$

Let us denote the free space of a road scene as $\mathcal{S}_{\texttt{free}}$, which excludes all obstacles and space beyond lane/road boundaries. We consider a trajectory $u(\cdot)$ as drivable up to a time horizon t_f if

$$\forall t \in [0, t_f] : \mathcal{O}(t, \mathcal{X}_0, u(\cdot), \mathcal{W}) \subseteq \mathcal{S}_{\texttt{free}},$$

i.e. the trajectory can be followed closely enough to avoid any possible collisions. An extension to an infinite time horizon is possible when considering fail-safe maneuvers as explained in [19], [34]. In order to speed up the computation of reachable occupancies, we pre-compute those for so called motion-primitives [35]. After computing reachable occupancies for each motion primitive, one can combine those when the final set of the preceding motion primitive is a subset of the initial set of the proceeding one (see Fig. 2 and [20]). We concatenate motion primitives so that the resulting movement is as closely as possible to the originally planned trajectory.

In this work, we consider three different vehicle models: A simple point-mass model for fast online planning, a kinematic model for the controller design, and a high-dimensional multi-body model for the conformance testing. As a result, we can efficiently plan motions based on simple models, while still guaranteeing that the movement is collision-free despite disturbances and other uncertainties. Even when using more sophisticated models during motion planning, the real system would not exactly behave as the used model. Thus, even for complicated models, conformance testing would be required – an extra effort that is rarely done in most other work. A byproduct of our approach is that we do not have to prove stability of our trajectory tracking controller, since safety is already guaranteed by the embedded reachability analysis.

III. Reference Trajectory

Generating collision-free paths for vehicles in a dynamically changing environment, such as on-road traffic, is a challenging problem. This is due to the non-convex nature of the solution space, which is caused by obstacle-avoidance constraints and both nonlinear dynamics and complex geometries of the vehicles under consideration. Thus, in its general form, the problem may not be solvable in real-time. However, approaches based on the solution of simplified problems exist. In this paper, we employ a formulation based on the solution of mixed-integer programs in order to generate reference trajectories, which are then matched by motion primitives and followed by their feedback controllers.

Despite unfavorable theoretic run-time properties, mixed-integer programming has already been applied successfully in the area of path planning [9], [10]. This success has been enabled by the fact that for simplified vehicle models, efficient algorithms exist which facilitate computation times far better than the theoretic worst-case run-times. Solving the planning problem in a receding-horizon fashion further reduces the computational load and also allows to react to changes in the environment. In the following, we briefly describe how we adapt existing approaches to the needs of the problem at hand. Plans are generated by minimizing:

$$J(x(\cdot|t_k), u(\cdot|t_k)) = \sum_{j=1}^{H} ||x(t_{k+j}|t_k) - x_{\text{ref}}(t_{k+j})||_Q^2 + ||u(t_{k+j-1}|t_k) - u_{\text{ref}}(t_{k+j-1})||_R^2 + ||\Delta u(t_{k+j-1}|t_k)||_S^2 \quad (3)$$

repeatedly over a receding horizon of length $H \in \mathbb{N}^+$, in which the positive semi-definite matrices $Q \in \mathbb{R}^{n \times n}$, $R \in \mathbb{R}^{m \times m}$, and $S \in \mathbb{R}^{m \times m}$ serve as user-defined weighting matrices. The search for appropriate input and state sequences,

$$u(\cdot|t_k) = \begin{pmatrix} u(t_k|t_k) & u(t_{k+1}|t_k) & \dots & u(t_{k+H-1}|t_k) \end{pmatrix},$$
$$x(\cdot|t_k) = \begin{pmatrix} x(t_{k+1}|t_k) & x(t_{k+2}|t_k) & \dots & x(t_{k+H}|t_k) \end{pmatrix},$$

is subject to linear dynamic constraints:

$$x(t_{k+j}|t_k) = Ax(t_{k+j-1}|t_k) + Bu(t_{k+j-1}|t_k), \quad (4)$$
$$x(t_k|t_k) = x(t_k),$$

and box constraints on states $x \in \mathbb{R}^n$, inputs $u \in \mathbb{R}^m$, and input increments $\Delta u(t_{k+j-1}|t_k) = u(t_{k+j-1}|t_k) - u(t_{k+j-2}|t_k)$, $u(t_{k-1}|t_k) = 0$:

$$x_{\min} \leq x(t_{k+j}|t_k) \leq x_{\max}, \quad (5)$$
$$u_{\min} \leq u(t_{k+j-1}|t_k) \leq u_{\max}, \quad (6)$$
$$\Delta u_{\min} \leq \Delta u(t_{k+j-1}|t_k) \leq \Delta u_{\max}. \quad (7)$$

The argument $t_{k+j}|t_k$ denotes the prediction at time instant t_k for a quantity at time instant t_{k+j}.

A linear prediction model is chosen in order to retain computational tractability. The vehicle is modeled by double integrator dynamics for both longitudinal and lateral motion,

such that the state vector $x = \begin{bmatrix} p_x & v_y & p_y & v_y \end{bmatrix}^\mathsf{T}$ in (4) contains the position and velocity in both longitudinal direction x and lateral direction y of the road, while the input vector $u = \begin{bmatrix} a_x & a_y \end{bmatrix}^\mathsf{T}$ consists of longitudinal and lateral acceleration. Accordingly, the discrete-time system matrices read:

$$A = \begin{bmatrix} 1 & T_\mathrm{s} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_\mathrm{s} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; B = \begin{bmatrix} \frac{1}{2}T_\mathrm{s}^2 & 0 \\ T_\mathrm{s} & 0 \\ 0 & \frac{1}{2}T_\mathrm{s}^2 \\ 0 & T_\mathrm{s} \end{bmatrix},$$

where T_s denotes the sampling time. In this model, longitudinal and lateral motion are decoupled by neglecting explicit constraints from non-holonomic kinematics and the friction circle. Because the latter represents a convex constraint, it could be accounted for by introducing a quadratic inequality constraint such as $a_x^2 + a_y^2 \leq a_{max}^2$, however at the expense of higher computation times. In order to implicitly capture these constraints and thus maintain drivability of the generated plans, proper choices of the weights in (3) and constraints in (5) are essential.

Given a set $\mathcal{C}$ of vehicles, collision avoidance is based on an obstacle representation as exemplified in Fig. 3: At first, all vehicles are over-approximated by rectangles in order to abstract from more complex geometric shape. Then, the bounding rectangles of both the ego-vehicle and other vehicles are projected onto the axes of a coordinate system aligned with the road orientation. The bounding rectangles of other vehicles are then enlarged by the dimensions of the bounding box of the ego-vehicle, thus allowing to consider it as a point mass from now on. In addition, we enlarge the bounding rectangle by safety margins:

$$l_\mathrm{safe}{}^{(i)} = |v_x^{(i)}|C_1,$$

depending on the velocities of the involved vehicles $i \in \mathcal{C}$, thus always allowing the execution of emergency maneuvers. The design parameter $C_1 \in \mathbb{R}^+$ permits to adjust the size of the margins.

Obstacle avoidance is then achieved by constraining the position $p_x^{(e)}$ of the ego-vehicle $e \in \mathcal{C}$ in the optimization problem to lie outside the bounding rectangles of obstacles $q \in \mathcal{C}$. A standard procedure to enforce this is to assign a binary variable δ to each edge of a bounding rectangle, which is set to 1 in the optimization problem if a position is on the non-critical side and to 0 otherwise:

$$\begin{aligned}
\delta_1^{(q)}(t_k) = 1 &\Leftrightarrow p_x^{(q)}(t_k) + l_\mathrm{safe}{}^{(q)}(t_k) \leq p_x^{(e)}(t_k) \\
\delta_2^{(q)}(t_k) = 1 &\Leftrightarrow p_x^{(e)}(t_k) \leq p_x^{(q)}(t_k) - l_\mathrm{safe}{}^{(e)}(t_k) \\
\delta_3^{(q)}(t_k) = 1 &\Leftrightarrow p_y^{(q)}(t_k) + W \leq p_y^{(e)}(t_k) \\
\delta_4^{(q)}(t_k) = 1 &\Leftrightarrow p_y^{(e)}(t_k) \leq p_y^{(q)}(t_k) - W.
\end{aligned}$$

Here, L and W denote the length respectively width of the enlarged bounding box before addition of the safety margins. Requiring that $\sum_{l=1}^{4} \delta_l^{(q)}(t_k) \geq 1$ then ensures that the position lies on at least one non-critical side of the rectangle and therefore outside of the obstacle. Implementation of these logical relations is based on the so-called Big-M-procedure [36].

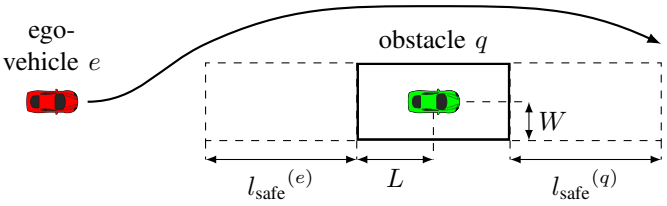

Fig. 3. Representation of obstacles and their safety zones by rectangles

IV. Controller Design

To obtain a tracking controller to compute motion primitives which are able to follow the trajectory generated in Sec. III, we extend the technique presented in [27] for the first time to nonlinear systems. The controller has to steer all states of an initial set $\mathcal{X}_0$ along a reference trajectory despite disturbances and measurement noise. Since we want to minimize the size of the reachable set while satisfying the input and state constraints, we formulate the controller design problem as an optimization problem. In contrast to classical optimal controller design, we do not optimize over open-loop control inputs, nor single trajectories. Instead, we directly minimize the over-approximative reachable set of the closed-loop dynamics including disturbances and measurement noise, included in the sets $\mathcal{W}$ and $\mathcal{V}$, respectively, see (1) and (2).

For computing the motion primitives, we consider the steady state vehicle model (SSM) proposed in [37] together with disturbances and input noise, which accounts for any uncertainties and model mismatch to the real vehicle:

$$\dot{p}_x = v\, cos(\theta), \tag{8}$$

$$\dot{p}_y = v\, sin(\theta), \tag{9}$$

$$\dot{\theta} = \frac{v}{l\left(1 + \left(\frac{v}{v_{ch}}\right)^2\right)}(\delta + w_\delta), \tag{10}$$

$$\dot{v} = a + w_a, \tag{11}$$

with the positions in x and y directions p_x and p_y, the orientation θ and velocity v as states; the control inputs acceleration a and steering angle δ; and disturbances w_a and w_δ. Moreover, we assume that there is a measurement uncertainty for each state, see (2). The characteristic velocity v_{ch} is a parameter which computes as $v_{ch} = \sqrt{\frac{l^2 c_f c_r}{m(c_r l_r - c_f l_f)}}$, with c_f, c_r denoting the cornering stiffness of the front and rear wheels, l_f, l_r the distances between the front and rear axis to the center of gravity, their sum $l = l_f + l_r$, and m the vehicle mass [37].

All reference trajectories of the motion primitives can start at 0 for the position and orientation, and we only have to sample different velocity ranges due to positional and rotational invariance. In order to maximize the flexibility of our maneuver automaton, we choose the same initial set for

all maneuvers and include the constraint, that the shifted and rotated final set must be inside this initial set, see Fig. 2. We choose the initial set to be a box around $x_{ref}(0)$ with $\mathcal{S}_{init} = \{x = x_{ref}(0) + \hat{x} | -x_i^{max} \leq \hat{x}_i \leq x_i^{max}\}$, where $\hat{x}_i$ denotes the i-th element of vector $\hat{x}$.

For the controller structure, we consider the following linear tracking controller:

$$u_{track}(x(t)) = u_{ref}(t) + K(t)(x(t) - x_{ref}(t)), \tag{12}$$

where $x_{ref}(\cdot), u_{ref}(\cdot)$ denote the reference state and input trajectories, respectively. In our case, the reference states and inputs are defined by the desired motion primitive. Our goal is to find the time-varying feedback controller $K(t)$ which tracks this reference trajectory despite disturbances and sensor noise.

The key of the approach is to formulate it as a nonlinear programming problem, in which we include reachability analysis. Therefore, we are able to formulate the cost function and the constraints directly in terms of reachable sets as proposed in [38] where we use zonotopes due to their favorable computational complexity. Zonotopes are sets of the following form

$$\mathcal{Z} = \left\{ x \in \mathbb{R}^n \middle| x = c + \sum_{i=1}^{p} \alpha_i g^{(i)}, \alpha_i \in [-1, 1] \right\}.$$

Therein $c \in \mathbb{R}^n$ defines the center of the zonotope, and $g^{(i)} \in \mathbb{R}^n, i \in \{1, \ldots, p\}$, are p generators. We use $< c, g^{(1)}, \ldots, g^{(p)} >$ as a more concise notation of $\mathcal{Z}$.

Since we start in a box, we want the reachable set to be enclosed in a box which is as small as possible. Therefore, we can formulate the cost function for the reachable set at the final time t_f as

$$\min_{K(\cdot)} \|\mathcal{R}(t_f, \mathcal{X}_0, u_{track}(\cdot), \mathcal{W}))\|_1,$$

where we denote for a set $\mathcal{S} \subset \mathbb{R}^n$ by $\|\mathcal{S}\|_1$ the sum of the edges of the axis-aligned bounding box, i.e.,

$$\|\mathcal{S}\|_1 = \sum_{i=1}^{n} (\sup_{x \in \mathcal{S}} x_i - \inf_{x \in \mathcal{S}} x_i).$$

If $\mathcal{S} =< c, g^{(1)}, \ldots, g^{(p)} >$ is a zonotope, which is the case for our reachable sets, then $\|\mathcal{S}\|_1$ can be efficiently computed by $\|\mathcal{S}\|_1 = \sum_{i=1}^{n} \sum_{j=1}^{p} |g_i^{(j)}|$. We can include a cost matrix which we multiply the generators in order to weight certain dimensions more than others or to normalize the final set with respect to the size of the initial set.

For the constraint function, we have to take two constraints into account: The final reachable set must be inside the shifted initial set and the input constraint must be satisfied at all times despite disturbances. The input constraint results from the friction circle

$$\sqrt{a_{long}^2 + a_{lat}^2} \leq a_{max}, \tag{13}$$

with the longitudinal acceleration $a_{long} = a + w_a$ and lateral acceleration $a_{lat} = v\dot{\theta} = \frac{v}{l\left(1+\left(\frac{v}{v_{ch}}\right)^2\right)}(\delta + w_\delta)$. While we treat the input constraints decoupled for time reasons during the online planning, we take the coupled constraints into account for the motion primitives for more accuracy. Note that any dynamical state constraints, such as avoiding other vehicles, are taken care of by the online-planner (see Sec. III), which is verified using the pre-computed reachable sets of the motion primitives. Since the initial set is a box around $x_{ref}(0)$, the the final set constraint can be written as

$$\sum_{j=1}^{p} |g_i^{(j)}| \leq x_i^{max}, \forall i \in \{1, \ldots, n\},$$

where $g^{(j)}$ are the generator of the final set, where the set of x and y positions are rotated by $\theta_{ref}(t_f)$. This can easily be extended for other types of initial sets, see [27] for details.

From the reachability analysis, we not only obtain the set of reachable staties, but also the set of applied input $\mathcal{Z}_u([t_k, t_{k+1}])$ for each time interval $[t_k, t_{k+1}]$. To check (13) now in a coupled way, we have to check if

$$\|\mathcal{Z}_u^*([t_k, t_{k+1}])\|_2 \leq a_{max}.$$

With $\mathcal{Z}_u^*([t_k, t_{k+1}])$, we denote the zonotope $\mathcal{Z}_u([t_k, t_{k+1}])$ which is projected into the a_{long} and a_{lat} space by multiplying the δ dimension with $\frac{v_{max}^2}{l(1+(\frac{v_{min}}{v_{ch}})^2)}$, where v_{min} and v_{max} denote the minimum and maximum value of the reachable set $\mathcal{R}([t_k, t_{k+1}], \mathcal{X}_0, u_{track}(\cdot), \mathcal{W})$ of the time interval $[t_k, t_{k+1}]$ in the v dimension. We take advantage of the fact that the norm of a zonotope can be exactly computed, see [39]. By checking the input constraint for all time intervals, we ensure that the real inputs satisfy the input constraints at all times despite disturbances and sensor noise.

V. Conformance Testing

One of the difficulties of applying formal methods to automated vehicles is the transference of formal properties to the real physical system. In order to make plausible, why results derived for the model also apply to the physical system, the conformance between a physical system and a model is investigated in this section. A model is said to conform to a system, if it reacts similarly to the system, when the same inputs are applied. *Testing* conformance refers to applying exemplary inputs to the system, recording observations of the system's behavior and investigating, whether the model can reproduce similar observations under these inputs.

We use the following definition of trace conformance: A test-case $\langle U_i, Y_i \rangle$ is understood as a combination of a control input trace $U_i = [u(t_1), ..., u(t_K)] \in \mathbb{R}^{m \times K}$ applied to the system and a measurement trace $Y_i = [y(t_1), ..., y(t_K)] \in \mathbb{R}^{o \times K}$ recorded from the system at discrete points of times $t_k, l \in \{1, \ldots, K\}$. A test suite is defined as a set of test cases $\{\langle U_1, Y_1 \rangle, ..., \langle U_r, Y_r \rangle\}$. A model $\langle f, h, \mathcal{V}, \mathcal{W} \rangle$ is said to be trace conformant, if for every test case $\langle U_i, Y_i \rangle$ a model trace $\langle X_i, V_i, W_i \rangle$ with $X_i = [x(t_1), ..., x(t_K)] \in \mathbb{R}^{n \times K}$, $V_i =$

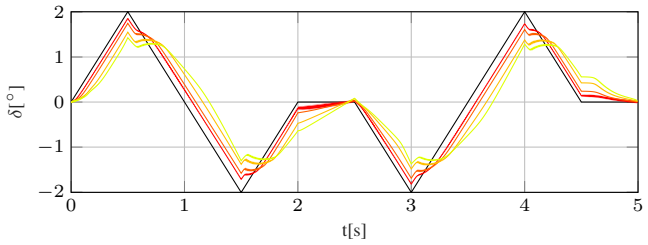

Fig. 4. Difference between multi-body and steady-state vehicle model. A steering angle error is applied to the steady-state model inputs, in order to match the output of the multi-body model. The deviation from the reference steering angle (black) increases with increasing velocity: Shown from $v = 15m/s$ (red) to $v = 25m/s$ (yellow).

$[v(t_1), ..., v(t_K)] \in \mathbb{R}^{o \times K}$, $W_i = [w(t_1), ..., w(t_K)] \in \mathbb{R}^{q \times K}$, exists, for which holds:

$$\forall k = \{1, \ldots, K\} :$$

$$x(t_{k+1}) = x(t_k) + \int_{t_k}^{t_{k+1}} f\left(x(\tau), u(t_k), w(t_k)\right) d\tau \quad (14)$$

$$\wedge \quad y(t_k) = h(x(t_k), \nu(t_k)) \quad (15)$$

$$\wedge \quad \nu(t_k) \in \mathcal{V} \wedge w(t_k) \in \mathcal{W} \quad (16)$$

Our conformance testing process involves recording a test-suite with the system and then solving a constraint satisfaction problem for equations (14)-(16) for each test case. If a valid model trace exists for each test case, the model is trace conformant. If the constraint satisfaction problem cannot be solved for one or more test-cases, the model is not conformant. In this case, the system has to be modeled more precisely by choosing a more appropriate f or h, or the uncertainty in the model has to be increased by changing $\mathcal{V}$ and $\mathcal{W}$.

VI. Numerical Example

In the following, a numerical example is exercised in order to demonstrate our approach to ensuring drivability of a maneuver. As a first step of ensuring drivability of a maneuver, which has been computed for a simple *planning-model*, the conformance between physical system and a *verification-model* is established. In the following we use the steady state vehicle model (SSM) (8)–(11) as a verification model. This includes quantification of measurement noise and disturbance errors, under which the verification model conforms to the physical system. The second step is verification of the drivability of the computed plan for the point-mass model by matching it with the motion primitives computed for the SSM model with disturbances and measurement noise.

To motivate the measurement error and disturbance error sets $\mathcal{V}, \mathcal{W}$ used for reachability analysis, we compare the verification model against a test-suite of five test-drives with a live-sized automated vehicle and against a test-suite of 480 simulated test-drives with a multi-body vehicle model. In both cases, the control input $u = [a, \delta]^T$ contains the requested steering angle as well as the requested acceleration and the measured output $y = [p_x, p_y, v, \theta]^T$ contains the position in x as well as y direction, the direction of movement θ, and the absolute velocity v (values are recorded at $100Hz$).

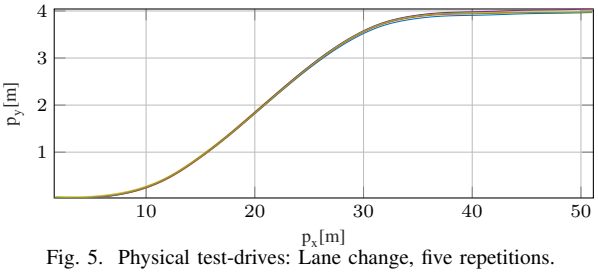

Fig. 5. Physical test-drives: Lane change, five repetitions.

TABLE I
Conformance testing: Measurement noise and disturbances

	$e^m_{p_x}[m]$	$e^m_{p_y}[m]$	$e^m_\theta[°]$	$e^m_v[\frac{m}{s}]$	$e^d_a[\frac{m}{s^2}]$	$e^d_\delta[°]$
MBM/MC	0.015	0.015	0.15	0.05	0.025	0.2
SSM2MBM	0.025	0.025	0.25	0.075	0.1	0.5
SSM2VEH	0.025	0.025	0.3	0.075	0.03	0.5

The automated vehicle (VEH) is equipped with an Inertial Navigation System (INS), which is connected to a differential GPS receiver filtering the measurements with an Extended Kalman Filter. A lane-change maneuver with velocity $v = 10\frac{m}{s}$ and lateral acceleration $a_y \approx 2\frac{m}{s^2}$ is defined as a trajectory and tracked in closed-loop, as shown in Fig. 5. The recorded inputs correspond to the inputs requested by the trajectory tracking controller and the recorded outputs are the outputs of the INS.

For the simulation experiments, we use a multi-body model (MBM) with 29 states described in [40]. To represent a test drive, a Monte-Carlo simulation is executed with open-loop control inputs $U_i \in \mathbb{R}^{2 \times K}$ according to the tested maneuver as well as with additive, uniformly distributed disturbances with bounds e^d given in row MBM/MC of Tab. I. A simulated output trace Y_i is generated by adding uniformly distributed measurement errors e^m. For the simulated test-drives, a test-suite is designed, which consists of two types of maneuvers, (double lane change and slalom), which are executed both in a $7s$ and a $10s$ time interval, with lateral accelerations from the set $a_{lat} \in \{1, 2, 4\}\frac{m}{s^2}$, test velocities from the set $v \in \{10, 15, 20, 25\}\frac{m}{s}$ and 10 repetitions per combination. The double lane change maneuver is specified by a piece-wise linear steering angle profile presented in Fig. 4 (black), the slalom maneuver is specified by a sine-wave steering angle profile with frequency $\frac{2}{7}Hz$ or $\frac{2}{10}Hz$.

We match the four dimensional steady state vehicle model (SSM) with differential equation f_{SSM} and measurement function $h(x) = x$ against the physical and the simulated test drives. As shown in Fig. 4, the deviations from the reference inputs (e.g. the disturbances) has to be increased with higher velocities and higher lateral accelerations to maintain similar outputs. In order to achieve a conformant, non-deterministic model $\langle f_{SSM}, h, \mathcal{V}, \mathcal{W} \rangle$, the error sets $\mathcal{V}$ and $\mathcal{W}$ are increased until (14)-(16) can be satisfied for all test cases $\langle U_i, Y_i \rangle$ by corresponding error traces $\langle V_i, W_i \rangle$, with $X_i = Y_i + V_i$ in this case. The bounds of error sets, for which conformance can be shown for all test cases are provided in Tab. I in line

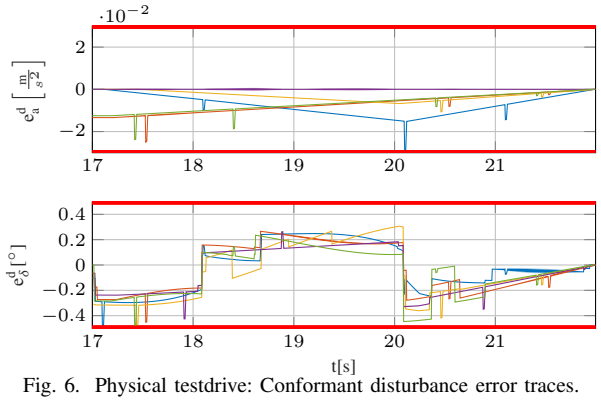

Fig. 6. Physical testdrive: Conformant disturbance error traces.

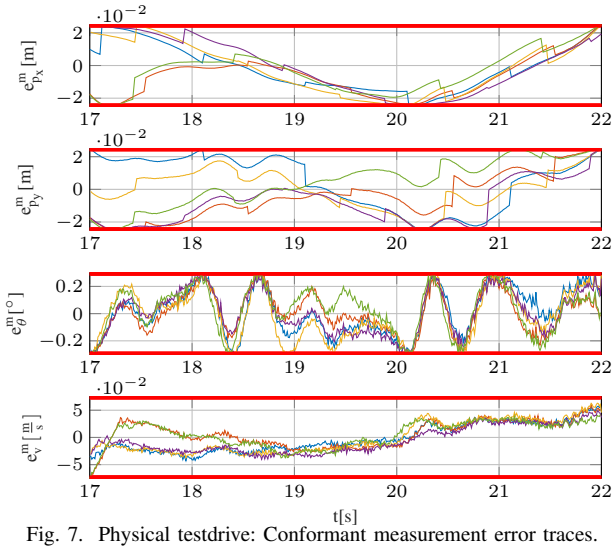

Fig. 7. Physical testdrive: Conformant measurement error traces.

SSM2MBM for the simulation test suite and in line SSM2VEH for the physical test-drives. The disturbance error traces W_i for the steering angle error of a subset of the simulation test suite is shown in Fig. 8. The measurement error traces of the simulation test suite are indistinguishable from random noise and are therefore omitted. The disturbance error traces and the measurement error traces for the physical test drives are given in Fig. 6 and 7 respectively. As can be seen, the given error bounds (red) are never exceeded.

We use our results from conformance testing for the controller design of the motion primitives. All reference trajectories start at 0 for the p_x, p_y, and θ and have a duration of $2s$ each. The initial set is for all maneuvers a box with size $[-0.2, 0.2]m \times [-0.2, 0.2]m \times [-1.15, 1.15]° \times [-0.2, 0.2]\frac{m}{s}$. We compute for each maneuver a robust controller by computing four pairs of Q and R matrices. Thereby, we assume the disturbances and measurement errors to belong in the sets corresponding to the maximum values from the conformance testing for the MBM and for the real driving data. We restrict the maximum acceleration in (13) to $a_{max} = 10\frac{m}{s^2}$. All considered maneuvers end in a final set, which is contained in initial set shifted and rotated by the $x_{ref}(2s)$, therefore, we can concatenate any maneuver with each other.

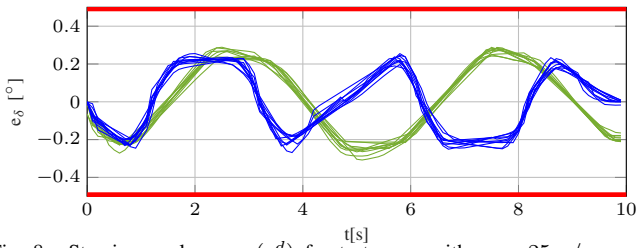

Fig. 8. Steering angle error (e_δ^d) for test-cases with $v = 25m/s$, $a_y \approx 4m/s^2$ and duration $10s$. Results for double lane-change maneuver in blue and slalom in green, admissible error bound in red.

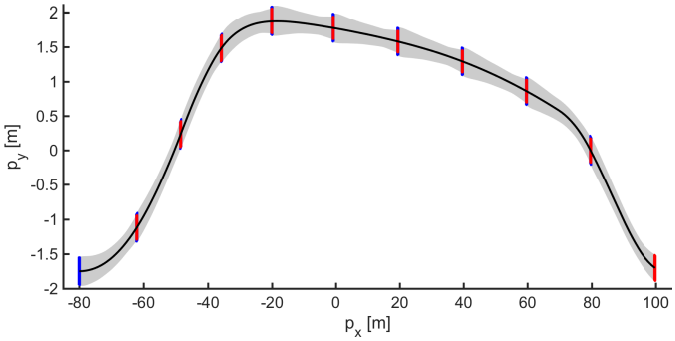

Fig. 9. Double lane-change maneuver driven using motion primitives along reference trajectory (black). Initial sets of motion primitives are shown in blue, final sets in red, and reachable sets in gray.

The combination of all parts is shown in Fig. 9. Therein, we show in black the planned path for a double-lane change, which is planned using our online planner. In the next step, we fit the planned path with our motion primitives. We show the initial sets of each motion primitive in blue, the final sets in red, and the reachable set in between in gray. In Fig. 10 we show a single maneuver. One can see, how the final set is contained in the initial set of the second maneuver, which is the rotated version of the initial set of the first maneuver.

VII. Conclusion

In this paper we present for the first time all parts of an efficient, formal path planning and tracking approach for autonomous vehicles. Since autonomous vehicles act in complex and safety-critical environments, it is important to have algorithms which solve the path planning problem in real-time despite other traffic participants and obstacles. To be able to do

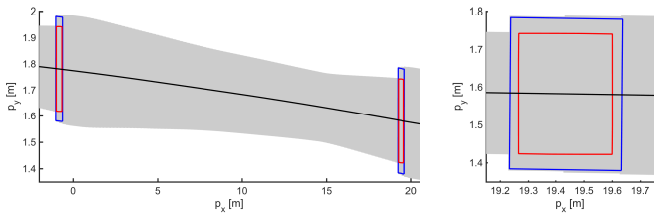

Fig. 10. Zoom into into an arbitrary motion primitive: Full motion primitive on the left, only the final set (red) with initial set from following maneuver (blue) on the right. Initial set is a box rotated by the orientation of the reference trajectory.

so, most of the times, the planning algorithms are restricted to simplified models with no guarantees if the planned paths are drivable and safe for the real vehicle. Therefore, we combine path planning with motion primitives, which include the pre-computed controllers and reachable sets for car models which include the real vehicle dynamics in form of disturbance sets. The disturbance sets are obtained from a combination of simulations with more complex models and real vehicle data to guarantee conformance of the controller model to the real vehicle dynamics. Since the planning algorithm uses a simple model and since the motion primitives are pre-computed, the online planning can be done in real-time, while we still ensure safety despite disturbances, sensor noise, and complex vehicle dynamics. We present the whole chain of methods to obtain these safe paths and show in an numerical example which uses real measurement data the applicability of our approach.

REFERENCES

[1] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[2] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[3] ——, *Planning Algorithms*. Cambridge University Press, 2006.

[4] D. González, J. Pérez, V. Milanés, and F. Nashashibi, "A review of motion planning techniques for automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1135–1145, 2016.

[5] H. Rewald and O. Stursberg, "Cooperation of autonomous vehicles using a hierarchy of auction-based and model-predictive control," in *2016 IEEE Int. Vehicles Symp.*, pp. 1078–1084.

[6] J. hwan Jeon, R. V. Cowlagi, S. C. Peters, S. Karaman, E. Frazzoli, P. Tsiotras, and K. Iagnemma, "Optimal motion planning with the half-car dynamical model for autonomous high-speed driving," in *2013 American Control Conf.* IEEE, pp. 188–193.

[7] M. Werling, L. Groll, and G. Bretthauer, "Invariant trajectory tracking with a full-size autonomous road vehicle," *IEEE Trans. on Robotics*, vol. 26, no. 4, pp. 758–765, 2010.

[8] D. J. N. Limebeer and A. V. Rao, "Faster, higher, and greener: Vehicular optimal control," *IEEE Control Systems*, vol. 35, no. 2, pp. 36–56, 2015.

[9] T. Schouwenaars, B. De Moor, E. Feron, and J. How, "Mixed integer programming for multi-vehicle path planning," in *2001 European Control Conf.* IEEE, pp. 2603–2608.

[10] X. Qian, F. Altché, P. Bender, C. Stiller, and A. de La Fortelle, "Optimal trajectory planning for autonomous driving integrating logical constraints: An miqp perspective," in *2016 IEEE 19th Int. Conf. on Intelligent Transportation Systems*, 2016, pp. 205–210.

[11] J. Eilbrecht and O. Stursberg, "Auction-based cooperation of autonomous vehicles using mixed-integer planning," in *AAET Automatisiertes & Vernetztes Fahren*. ITS automotive nord e.V., 2017.

[12] Y. Kuwata and J. P. How, "Cooperative distributed robust trajectory optimization using receding horizon milp," *IEEE Transactions on Control Systems Technology*, vol. 19, no. 2, pp. 423–431, March 2011.

[13] A. Colombo and D. Del Vecchio, "Efficient algorithms for collision avoidance at intersections," in *Proc. of Hybrid Systems: Computation and Control*, 2012, pp. 145–154.

[14] S. M. Loos, A. Platzer, and L. Nistor, "Adaptive cruise control: Hybrid, distributed, and now formally verified," in *Proc. of the 17th International Symposium on Formal Methods*, ser. LNCS 6664. Springer, 2011, pp. 42–56.

[15] M. Hilscher, S. Linker, and E.-R. Olderog, *Theories of Programming and Formal Methods*. Springer, 2013, ch. Proving Safety of Traffic Manoeuvres on Country Roads, pp. 196–212.

[16] W. Damm, H. J. Peter, J. Rakow, and B. Westphal, "Can we build it: Formal synthesis of control strategies for cooperative driver assistance systems," *Mathematical Structures in Computer Science*, vol. 23, no. 4, pp. 676–725, 2013.

[17] M. Althoff, D. Althoff, D. Wollherr, and M. Buss, "Safety verification of autonomous vehicles for coordinated evasive maneuvers," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2010, pp. 1078–1083.

[18] M. Althoff and J. M. Dolan, "Set-based computation of vehicle behaviors for the online verification of autonomous vehicles," in *Proc. of the 14th IEEE Conference on Intelligent Transportation Systems*, 2011, pp. 1162–1167.

[19] ——, "Online verification of automated road vehicles using reachability analysis," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 903–918, 2014.

[20] D. Heß, M. Althoff, and T. Sattel, "Formal verification of maneuver automata for parameterized motion primitives," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 1474–1481.

[21] D. Q. Mayne, M. M. Seron, and S. V. Raković, "Robust model predictive control of constrained linear systems with bounded disturbances," *Automatica*, vol. 41, no. 2, pp. 219 – 224, 2005.

[22] S. V. Rakovic, B. Kouvaritakis, M. Cannon, C. Panos, and R. Findeisen, "Parameterized tube model predictive control," *IEEE Transactions on Automatic Control*, vol. 57, no. 11, pp. 2746–2761, 2012.

[23] M. Zamani, G. Pola, M. Mazo Jr., and P. Tabuada, "Symbolic models for nonlinear control systems without stability assumptions," *IEEE Transactions on Automatic Control*, vol. 57, no. 7, pp. 1804–1809, July 2012.

[24] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, "LQR-trees: Feedback motion planning via sums-of-squares verification," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 1038–1052, 2010.

[25] B. Schürmann and M. Althoff, "Convex interpolation control with formal guarantees for disturbed and constrained nonlinear systems," in *Proc. of Hybrid Systems: Computation and Control*, 2017, submitted.

[26] ——, "Efficient set-based optimal control for disturbed nonlinear systems with formal guarantees," in *Proc. of the 20th World Congress of the International Federation of Automatic Control*, 2017.

[27] B. Schürmann and M. Althoff, "Optimal control of sets of solutions to formally guarantee constraints of disturbed linear systems," in *Proc. of the American Control Conference*, 2017.

[28] J. Tretmans, "A formal approach to conformance testing," Ph.D. dissertation, Universiteit Twente, 1992.

[29] M. P. W. J. van Osch, "Automated model-based testing of hybrid systems," Ph.D. dissertation, Technische Universiteit Eindhoven, 2009.

[30] T. Dang, *Model-Based Testing for Embedded Systems*. CRC Press, 2011, ch. Model-based Testing of Hybrid Systems, pp. 383–423.

[31] M. Althoff and J. M. Dolan, "Reachability computation of low-order models for the safety verification of high-order road vehicle models," in *Proc. of the American Control Conference*, 2012, pp. 3559–3566.

[32] H. Roehm, J. Oehlerking, M. Woehrle, and M. Althoff, "Reachset conformance testing of hybrid automata," in *Proc. of Hybrid Systems: Computation and Control*, 2016, pp. 277–286.

[33] G. Lafferriere, G. J. Pappas, and S. Yovine, "Symbolic reachability computation for families of linear vector fields," *Symbolic Computation*, vol. 32, pp. 231–253, 2001.

[34] S. Magdici and M. Althoff, "Adaptive cruise control with safety guarantees for autonomous vehicles," in *Proc. of the 20th World Congress of the International Federation of Automatic Control*, 2017.

[35] E. Frazzoli, M. A. Dahleh, and E. Feron, "Maneuver-based motion planning for nonlinear systems with symmetries," *IEEE Transactions on Robotics*, vol. 21, no. 6, pp. 1077–1091, 2005.

[36] H. P. Williams, *Model Building in Mathematical Programming*. John Wiley & Sons, 2013.

[37] M. Werling and D. Liccardo, "Automatic collision avoidance using model-predictive online optimization," in *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*. IEEE, 2012, pp. 6309–6314.

[38] M. Althoff, "Reachability analysis and its application to the safety assessment of autonomous cars," Dissertation, Technische Universität München, 2010, http://nbn-resolving.de/urn/resolver.pl?urn:nbn:de:bvb:91-diss-20100715-963752-1-4.

[39] ——, "An introduction to CORA 2015," in *Proc. of the Workshop on Applied Verification for Continuous and Hybrid Systems*, 2015, pp. 120–151.

[40] M. Althoff, M. Koschi, and S. Manzinger, "Commonroad – composable benchmarks for motion planning on roads," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2017.