Ensuring drivability of planned motions using formal methods — **Source link** ⧉

Bastian Schürmann, Daniel Heß, Jan Eilbrecht, Olaf Stursberg ...+2 more authors

Institutions: Technische Universität München, German Aerospace Center, University of Kassel

Published on: 01 Oct 2017 - International Conference on Intelligent Transportation Systems

Topics: Motion planning and Collision avoidance

Related papers:

- Online Verification of Automated Road Vehicles Using Reachability Analysis

- Ensuring Drivability of Planned Motions from Simple Models Using Formal Methods

- A Review of Motion Planning Techniques for Automated Vehicles

- A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles

- CommonRoad: Composable benchmarks for motion planning on roads

Ensuring Drivability of Planned Motions 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 motion planning is usually done using simplified models, 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 of simple dynamics and set-based uncertainty; conformance testing is used to determine the required uncertainty. To quickly provide the set of solutions for these uncertain models, we use pre-computed reachable sets (i.e., the union of all possible solutions) for pre-selected motion primitives. The reachable sets of vehicles are obtained through the novel combination of optimization techniques and reachability analysis in the controller synthesis – 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 only ensure feasibility to a lesser extent, since not all aspects of the vehicle's dynamics are considered.

In this work, we address exactly this problem: By computing motion plans from a simple point-mass model we enable efficient computation. We ensure 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 can follow a planned maneuver, c) control approaches which can provide guarantees, and d) conformance testing methods.

a) Motion planning: The last decades have witnessed significant progress in the development of motion planning algorithms for both general problems [2] and in the field of autonomous driving in particular [3]. 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 or which lack the guarantee of convergence to a global optimum. These methods comprise graph-searching in a discretized state space of a nonlinear model [4], sampling-based approaches such as RRT* [5], nonlinear model-predictive control [6], and optimal control procedures like hp-adaptive collocation [7]. In contrast to these detailed models, much simpler ones are used in many approaches, which decouple longitudinal and lateral dynamics and do not enforce kinematic constraints explicitly [8], [9], [10], [11]. While these models only give a coarse representation of a real vehicle's dynamics, they allow the application of efficient planning algorithms.

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. Some works formally prove correctness, but only consider specific aspects of automated driving. An approach for safely entering an intersection is presented in [12]. In [13], automated cruise control is formally verified by automated theorem proving. Automated theorem proving has also been applied to overtaking maneuvers in [14] under the assumption of perfect knowledge of the environment. A verified synthesis for driving assistance in traffic merging is presented in [15].

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 was first performed theoretically for cooperating vehicles in [16], for mixed traffic in [17], and on a real vehicle in [18]. In our previous work [19], we present a much simpler method, in comparison to the approach shown here, for pre-computing drivable maneuvers (so called maneuver primitives).

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 [20], [21]. However, these methods work mainly for linear systems. Abstraction-based control [22] is a formal control approach which is able to guarantee complex specifica-

tions. Since this is often achieved by discretizing the state and input spaces, these approaches are limited to low-dimensional systems. In [23], the authors compute so-called LQR-trees to control sets of initial states using sums-of-squares methods which have rather high computational costs.

Recently, we proposed several new controller design approaches [24], [25], [26], which optimize over sets of solutions and guarantee constraint satisfaction, even for disturbed nonlinear systems, by incorporating reachability analysis in the controller design. While [24], [25] interpolate open-loop trajectories and therefore result in non-continuous control laws, the approach in [26] directly optimizes over the closed-loop dynamics. In [26], we restrict the approach to disturbed, linear systems; here we apply it 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 [27, p. 30]. We use conformance testing to find the set of possible deviations between a model and a real system. There exists rather limited work for conformance testing of automated vehicles. Although research exists for general hybrid systems [28], this work does not quantify the difference between the real system and the model. In [29], rapidly-exploring random trees are used for test generation. Rather than quantifying the model error, the authors investigate the compliance with a specification. First works on rigorously bounding model errors for systems with continuous dynamics have been developed by the authors themselves: In [30], methods have been developed to create simple models with uncertainty that capture all behaviors of complex models and in [31], this concept has been generalized to 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 also present the first work ensuring that the used uncertain model contains all behaviors recorded from test drives of a 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.

We begin with an overview in Sec. II. Next, we describe the three main parts of our approach, starting with the reference trajectory in Sec. III, followed by the controller design in Sec. IV, and by 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 described, we use a simple point-mass model for motion planning to save computation time. Since the dynamics of a real vehicle does not exactly match that 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
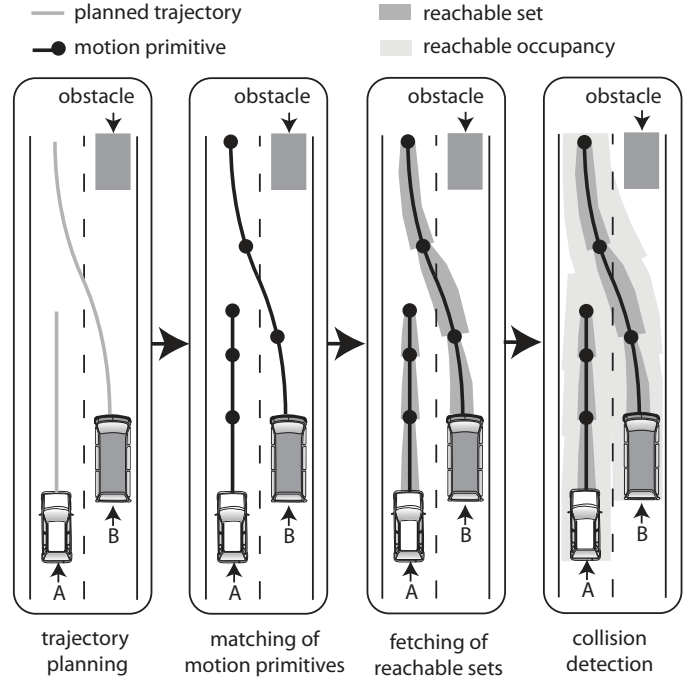

Fig. 1. Checking drivability from left to right: a) trajectory planning of vehicles (here: vehicle A and B), b) matching motion primitives with the obtained trajectories, c) fetching pre-computed reachable sets and reachable occupancies (reachable occupancies not yet displayed), d) collision checking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

introduce the state $x \in \mathbb{R}^n$, the control input $u \in \mathbb{R}^m$, and 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 measurement of the system is modeled by a function h, returning the measurement vector $y \in \mathbb{R}^o$ subject to 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 $\mathcal{R}^e$ starting from the set of possible initial states $x(0) \in \mathcal{X}_0$:

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

The superscript e on $\mathcal{R}^e(t)$ denotes the exact reachable set, which cannot be computed for arbitrary nonlinear systems [32]. 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.
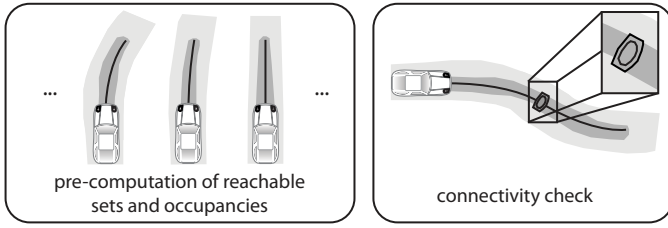
Fig. 2. Pre-computation of reachable sets and reachable occupancies for all motion primitives. It is checked which motion primitives can be combined (last reachable set of preceding motion primitive has to be enclosed in intial set of the proceeding motion primitive).

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, we define the reachable occupancy 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 $\mathcal{S}_{\text{free}}$, 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

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

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 [18], [33], [34]. In order to speed up the computation of reachable occupancies, we pre-compute them for so-called motion primitives [35]. After computing reachable occupancies for each motion primitive, one can combine them when the final set of the preceding motion primitive is a subset of the initial set of the proceeding one (see Fig. 2 and [19]). We concatenate motion primitives so that the resulting movement is as close 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. Conformance testing can either be done using measurements from a real vehicle or using a high-fidelity model. It is even possible to combine both, as demonstrated later. 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 using more sophisticated models during motion planning would not result in behavior identical to the real system. 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 [8], [9]. This success has been enabled by the fact that for simplified vehicle models, efficient algorithms exist which facilitate far better computation times than the theoretic worst-case run-times. Solving the planning problem in a receding-horizon fashion further reduces the computational load and also allows one to react to changes in the environment. Plans are generated by minimizing

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

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 and where x_{ref} contains velocity and lateral position references. Let us introduce the argument $t_{k+j}|t_k$ to denote the prediction at time instant t_k for a quantity at time instant t_{k+j}. The search for appropriate input and state sequences,

$$u(\cdot|t_k) = \big(\ u(t_k|t_k) \quad u(t_{k+1}|t_k) \quad \ldots \quad u(t_{k+H-1}|t_k) \ \big),$$
$$x(\cdot|t_k) = \big(\ x(t_{k+1}|t_k) \quad x(t_{k+2}|t_k) \quad \ldots \quad x(t_{k+H}|t_k) \ \big),$$

is subject to linear dynamical 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 $\Delta u(t_{k+j-1}|t_k) = u(t_{k+j-1}|t_k) - u(t_{k+j-2}|t_k)$, $u(t_{-1}|t_0) = 0$:

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

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 $x = [p_x, v_x, p_y, v_y]^\top$ in (4) contains the position and velocity in both longitudinal direction x and lateral direction y of the road, while the input vector

$u = [a_x, a_y]^\mathsf{T}$ consists of longitudinal and lateral acceleration. Accordingly, the discrete-time system matrices read

$$A = \begin{bmatrix} 1 & T_s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} \frac{1}{2}T_s^2 & 0 \\ T_s & 0 \\ 0 & \frac{1}{2}T_s^2 \\ 0 & T_s \end{bmatrix},$$

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 $a_x^2 + a_y^2 \leq a_{\max}^2$, though 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.

For 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 a 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 us to consider it as a point mass from now on. In addition, we enlarge the bounding rectangle by the safety margins $l_{\text{safe}}^{(i)} = |v_x^{(i)}| C_1$ to avoid collisions. The design parameter $C_1 \in \mathbb{R}^+$ permits adjusting 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:

$$\delta_1^{(q)}(t_k) = 1 \Leftrightarrow p_x^{(q)}(t_k) + l_{\text{safe}}^{(q)}(t_k) + \tfrac{1}{2}L \leq p_x^{(e)}(t_k)$$
$$\delta_2^{(q)}(t_k) = 1 \Leftrightarrow p_x^{(e)}(t_k) \leq p_x^{(q)}(t_k) - l_{\text{safe}}^{(e)}(t_k) - \tfrac{1}{2}L$$
$$\delta_3^{(q)}(t_k) = 1 \Leftrightarrow p_y^{(q)}(t_k) + \tfrac{1}{2}W \leq p_y^{(e)}(t_k)$$
$$\delta_4^{(q)}(t_k) = 1 \Leftrightarrow p_y^{(e)}(t_k) \leq p_y^{(q)}(t_k) - \tfrac{1}{2}W.$$

Here, L and W denote the length and width, respectively, of the enlarged bounding box before adding the safety margins. Requiring that $\sum_{l=1}^{4} \delta_l^{(q)}(t_k) \geq 1$ 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].

IV. CONTROLLER DESIGN

To be able to follow the trajectory generated in Sec. III, we pre-compute motion primitives by extending the technique presented in [26] for the first time to nonlinear systems. We design controllers offline for motion primitives as described in Sec. II to steer all states of an initial set $\mathcal{X}_0$ along a reference
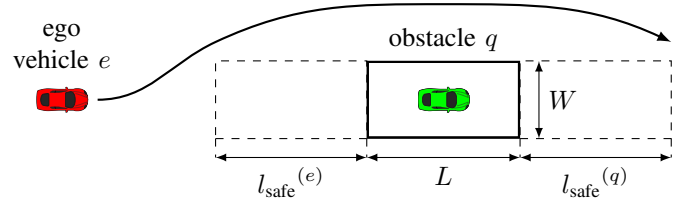

Fig. 3. Obstacles and their safety zones are represented by rectangles

trajectory despite disturbances and measurement noise. Since we want to minimize the size of the reachable set (see Fig. 2) 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 or 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:

$$\dot{p}_x = v \cos(\theta), \tag{8}$$
$$\dot{p}_y = v \sin(\theta), \tag{9}$$
$$\dot{\theta} = \frac{v}{l\left(1 + \left(\frac{v}{v_{\text{ch}}}\right)^2\right)}(\delta + w_\delta), \tag{10}$$
$$\dot{v} = a + w_a, \tag{11}$$

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 $v_{\text{ch}} = \sqrt{\frac{l^2 c_f c_r}{m(c_r l_r - c_f l_f)}}$, 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; $l = l_f + l_r$ their sum; and m the vehicle mass [37].

Without loss of generality, all reference trajectories of the motion primitives start at 0 for the position and orientation due to positional and rotational invariance. We only have to sample different velocity ranges. To maximize flexibility, 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 $x_{\text{ref}}(0)$ with $\mathcal{S}_{\text{init}} = \{x = x_{\text{ref}}(0) + \hat{x} \mid -x_i^{\max} \leq \hat{x}_i \leq x_i^{\max}\}$, where $\hat{x}_i$ denotes the i-th element of vector $\hat{x}$.

We fix the controller to be a linear tracking controller:

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

where $x_{\text{ref}}(\cdot), u_{\text{ref}}(\cdot)$ 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 to our approach is including reachability analysis [38] into a nonlinear programming problem. We use zonotopes to represent the reachable sets due to their favorable computational complexity for required operations introduced subsequently. Zonotopes are defined as

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

Therein $c \in \mathbb{R}^n$ is 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 $\mathcal{Z}$.

To be able to connect as many motion primitives with each other as possible, we minimize the reachable set at the final time t_f while restricting it to be inside the shifted initial set. Due to the box shape of the initial set, we introduce the cost function for minimizing the reachable set based on the minimal bounding box as

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

where we denote, for a set $\mathcal{S} \subset \mathbb{R}^n$, the sum of the edges of its axis-aligned bounding box by $\|\mathcal{S}\|_1$, 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 $\|\mathcal{S}\|_1 = 2 \sum_{i=1}^{n} \sum_{j=1}^{p} |g_i^{(j)}|$ [39]. We can include a weighting matrix which we multiply the generators by in order to weight certain dimensions more than others or to normalize the final set with respect to the size of the initial set.

Since initial states are bounded by a box around $x_{\text{ref}}(0)$, 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 generators of the final set, where the set of x and y positions are rotated by $\theta_{\text{ref}}(t_f)$. This can easily be extended for other types of initial sets, see [26] for details. Note that any dynamical state constraints, such as avoiding other vehicles, are taken care of by the online planner (see Sec. II), which uses the pre-computed reachable sets of the motion primitives.

In addition to the final set constraint, we consider the input constraint which results from the friction circle

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

with the longitudinal acceleration $a_{\text{long}} = a + w_a$ and lateral acceleration $a_{\text{lat}} = v\dot{\theta} \stackrel{(10)}{=} \frac{v^2}{l\left(1 + \left(\frac{v}{v_{\text{ch}}}\right)^2\right)} (\delta + w_\delta)$, both with respect to the orientation of the vehicle. 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.

From the reachability analysis, we not only obtain reachable states, 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) in a coupled way, we check if

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

By $\mathcal{Z}_u^*([t_k, t_{k+1}])$, 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 $\frac{v_{\max}^2}{l(1 + (\frac{v_{\min}}{v_{\text{ch}}})^2)}$, where $v_{\min}$ and $v_{\max}$ denote the minimum and maximum value of the reachable set $\mathcal{R}([t_k, t_{k+1}], \mathcal{X}_0, u_{\text{track}}(\cdot), \mathcal{W})$ of the time interval $[t_k, t_{k+1}]$ in the velocity 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 transferring formal properties from the model to the real, physical system. In order to justify 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 $U_i = [u(t_1), \ldots, u(t_K)] \in \mathbb{R}^{m \times K}$ applied to the system and a measurement trace $Y_i = [y(t_1), \ldots, y(t_K)] \in \mathbb{R}^{o \times K}$ recorded from the system at discrete points in time $t_1, \ldots, t_K$. A test suite is defined as a set of test cases $\{\langle U_1, Y_1 \rangle, \ldots, \langle U_r, Y_r \rangle\}$. 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 $X_i = [x(t_1), \ldots, x(t_K)] \in \mathbb{R}^{n \times K}$, $V_i = [v(t_1), \ldots, v(t_K)] \in \mathbb{R}^{o \times K}$, $W_i = [w(t_1), \ldots, w(t_K)] \in \mathbb{R}^{q \times K}$, exists, for which the following holds:

$$\forall t_k \in \{t_1, \ldots, t_K\} :$$
$$x(t_{k+1}) = x(t_k) + \int_{t_k}^{t_{k+1}} f(x(\tau), u(t_k), w(t_k)) \, d\tau \tag{14}$$
$$\wedge \quad y(t_k) = h(x(t_k), u(t_k), \nu(t_k)) \tag{15}$$
$$\wedge \quad \nu(t_k) \in \mathcal{V} \wedge w(t_k) \in \mathcal{W}. \tag{16}$$

Our conformance testing process involves recording a test suite with the real system and then solving a constraint satisfaction problem for equations (14)–(16) for each test case. The problem is similar to a constrained optimal control problem, which uses the disturbances as system inputs and can thus be solved accordingly. We use local linearization of the system model and the measurement constraints, a quadratic cost function for the disturbances, and MATLAB quadprog

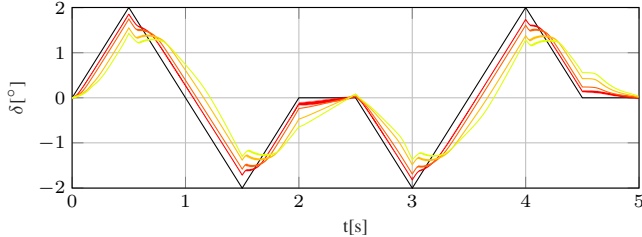Fig. 4. Automated Vehicle FASCar2, which is used to obtain the test drives for the conformance testing.

Fig. 5. 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 $v = 10\,\mathrm{m/s}$ (red) to $v = 25\,\mathrm{m/s}$ (green).

Fig. 6. Physical test drives: Lane change, five repetitions.

TABLE I
CONFORMANCE TESTING: MEASUREMENT ERRORS AND DISTURBANCES

	$e_{p_x}^m\,[m]$	$e_{p_y}^m\,[m]$	$e_\theta^m\,[°]$	$e_v^m\,[\frac{m}{s}]$	$e_a^d\,[\frac{m}{s^2}]$	$e_\delta^d\,[°]$
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

to find candidate disturbance traces. These can be applied to an initial value problem with the nonlinear model, solved at the desired numerical precision, and tested against the measurement error bounds. 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 by increasing the uncertainty in the model by changing $\mathcal{V}$ or $\mathcal{W}$.

VI. NUMERICAL EXAMPLE

In this section, we consider a numerical example in order to demonstrate our approach for ensuring drivability of a maneuver, which has been computed for a simple *planning model*. First, the conformance between the 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. Second, we verify the drivability of the computed trajectory for the point-mass model by matching it with the motion primitives computed for the SSM model with disturbances and measurement noise.

As mentioned in Sec. II, our approach is capable of obtaining the measurement error and disturbance error sets $\mathcal{V}, \mathcal{W}$ used for reachability analysis, either from real vehicle data or from simulations of a high-fidelity model. To demonstrate this, we compare the verification model against a test suite of five test drives with a life-sized automated vehicle (see Fig. 4, FASCar2, detailed technical description in [40]) and against a test suite of 480 Monte-Carlo simulations with a multi-body vehicle model. While the experiments with a real
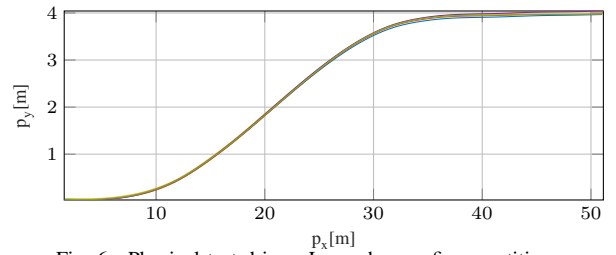
vehicle serve to demonstrate applicability of our approach to a physical system, the simulated test suite underpins reliability by successful execution of a bigger number of tests. In both test suites, the control input $u = [a, \delta]^{\mathsf{T}}$ contains the requested steering angle as well as the requested acceleration, and the measured output $y = [p_x, p_y, v, \theta]^{\mathsf{T}}$ contains the positions in x and y directions, the direction of movement θ, and the absolute velocity v (values are recorded at $100\,\mathrm{Hz}$). Here, the *requested* control input refers to the control input sent from the controller to the actuator, implying that actuator errors are incorporated in the disturbances of the resulting, conformant model. A lane-change maneuver with velocity $v = 10\,\mathrm{m/s}$ and lateral acceleration $a_y \approx 2\,\mathrm{m/s}^2$ is used for conformance testing, as shown in Fig. 6.

For the simulation experiments, we use a multi-body model (MBM) with 29 states described in [41]. To represent a test drive, a Monte-Carlo simulation is executed with 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; this consists of two types of maneuvers (double lane change and slalom), which are executed both in a $7\,\mathrm{s}$ and a $10\,\mathrm{s}$ time interval, with lateral accelerations from the set $a_{\mathrm{lat}} \in \{1, 2, 4\}\mathrm{m/s}^2$, test velocities from the set $v \in \{10, 15, 20, 25\}\mathrm{m/s}$, and 10 repetitions per combination. The double lane change maneuver is specified by a piece-wise linear steering angle profile presented in Fig. 5 (black) and the slalom maneuver is specified by a sine-wave steering angle profile with frequency $0.2\,\mathrm{Hz}$.

We match the four-dimensional SSM with differential equation f_{SSM} (8)–(11) and measurement function $h(x, u, \nu) = x + \nu$ against the physical system and the simulated test drives. As shown in Fig. 5, the deviations from the reference inputs have to be increased with higher velocities and higher lateral accelerations to maintain similar outputs, e.g., by increasing the disturbances. In order to achieve a conformant, non-
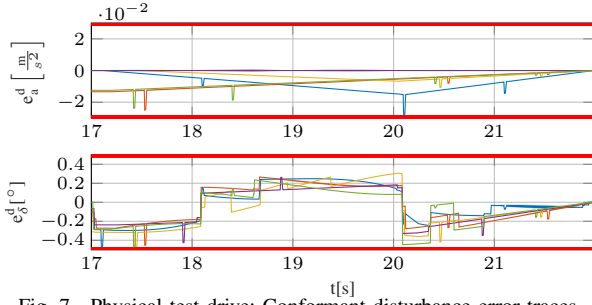
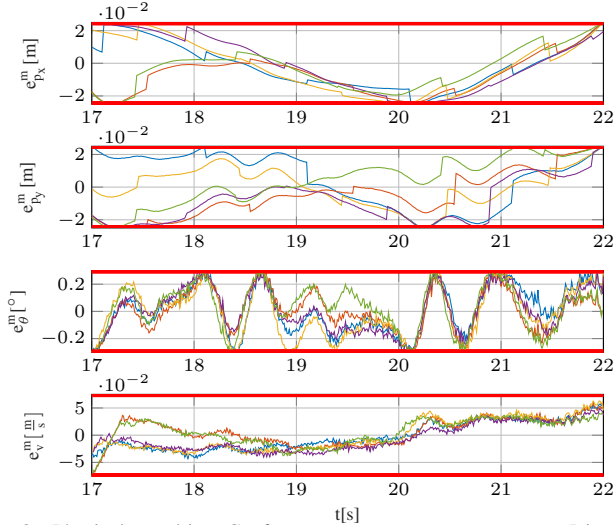Fig. 7. Physical test drive: Conformant disturbance error traces.

Fig. 8. Physical test drive: Conformant measurement error traces. Discontinuities due to jumps in GPS position signal.

Fig. 9. Steering angle error (e_δ^d) for test-cases with $v = 25\,\mathrm{m/s}$, $a_y \approx 4\,\mathrm{m/s}^2$, and duration $10\,\mathrm{s}$. Results for double lane-change maneuver in blue and slalom in green, admissible error bound in red.

Fig. 10. Double lane-change maneuver driven using motion primitives along planned trajectory (green). Initial sets of motion primitives are shown in blue, final sets in red, reachable sets in gray, and their reference trajectories in black. The planned trajectory is hard to see, as in most parts the reference trajectories are very close to it.

deterministic model $\langle f_{\mathrm{SSM}}, h, \mathcal{V}, \mathcal{W} \rangle$, the disturbance set $\mathcal{W}$ and the measurement error set $\mathcal{V}$ are increased until (14)-(16) can be satisfied by the corresponding error traces $\langle V_i, W_i \rangle$ for all test cases $\langle U_i, Y_i \rangle$, with $X_i = Y_i + V_i$. The bounds of error sets, for which conformance can be shown for all test cases, are provided in Tab. I in line 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 are shown in Fig. 9. 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. 7 and 8, respectively. Each trace of a given color corresponds to one test drive. 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. As previously described (see Sec. IV), all reference trajectories of the motion primitives start at 0 for p_x, p_y, and θ and have a duration of $2\,\mathrm{s}$ each. The initial set for all maneuvers is a box with size $[-0.2, 0.2]\mathrm{m} \times [-0.2, 0.2]\mathrm{m} \times [-1.15, 1.15]° \times [-0.2, 0.2]\mathrm{m/s}$. We compute a robust controller for each maneuver and assume thereby that the disturbances and measurement errors belong in the sets corresponding to the maximum values from the confor-
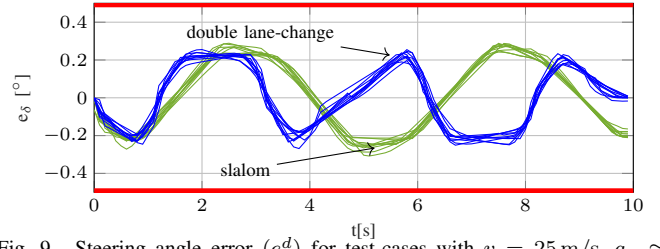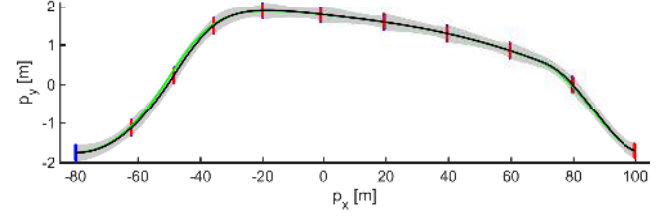
mance testing for the MBM and for the real driving data. We restrict the maximum acceleration in (13) to $a_{\max} = 10\,\mathrm{m/s}^2$. All considered maneuvers end in a final set, which is contained in the initial set shifted and rotated by $x_{\mathrm{ref}}(2\,\mathrm{s})$. Therefore, we can concatenate any one maneuver with any other.

The combination of all parts is shown in Fig. 10. Therein, we show in green 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, the reachable set in between in gray, and their reference trajectories in black. In Fig. 11 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 and formal path-planning-and-tracking approach for autonomous vehicles. Since autonomous vehicles act in
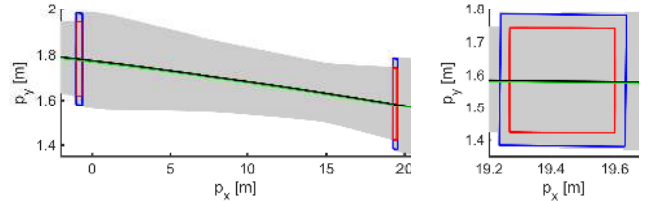

Fig. 11. Zoom into a single motion primitive: Full motion primitive on the left, final set (red) with initial set from following maneuver (blue) on the right. Initial set is a box rotated by the orientation of the reference trajectory.

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 so, the planning algorithms are mostly restricted to simplified models with no guarantees whether the planned paths are drivable and safe for the real vehicle. We therefore combine path planning with motion primitives, which include the pre-computed controllers and reachable sets for car models which incorporate the real vehicle dynamics in the 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 our planning algorithm uses a simple model and since the motion primitives are pre-computed, we can perform the online planning in real-time, while still ensuring safety despite disturbances, sensor noise, and complex vehicle dynamics.

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 Trans. on Intelligent Vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

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

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

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

[5] 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 *Proc. of the American Control Conf.*, 2013, pp. 188–193.

[6] 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.

[7] 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.

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

[9] 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 *Proc. of the 19th Int. Conf. on Intelligent Transportation Systems*. IEEE, 2016, pp. 205–210.

[10] J. Eilbrecht and O. Stursberg, "Cooperative driving using a hierarchy of mixed-integer programming and tracking control," in *Proc. of the Intelligent Vehicles Symposium*. IEEE, 2017, pp. 673–678.

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

[12] 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.

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

[14] 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.

[15] 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.

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

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

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

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

[20] 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.

[21] 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.

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

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

[24] 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, pp. 121–130.

[25] ——, "Guaranteeing constraints of disturbed nonlinear systems using set-based optimal control in generator space," in *Proc. of the 20th IFAC World Congress*, 2017, pp. 12 020–12 027.

[26] ——, "Optimal control of sets of solutions to formally guarantee constraints of disturbed linear systems," in *Proc. of the American Control Conf.*, 2017, pp. 2522–2529.

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

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

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

[30] 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 Conf.*, 2012, pp. 3559–3566.

[31] 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.

[32] 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.

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

[34] D. Heß, C. Löper, and T. Hesse, "Safe cooperation of automated vehicles," in *Proc. of the AAET Automatisiertes & Vernetztes Fahren*. ITS automotive nord e.V., 2017, pp. 309–334.

[35] E. Frazzoli, M. A. Dahleh, and E. Feron, "Maneuver-based motion planning for nonlinear systems with symmetries," *IEEE Trans. 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 *Proc. of the 51st Conf. on Decision and Control*. 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.

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

[40] G. Temme, T. Hesse, C. Löper *et al.*, "EU-project HAVEit Deliverable D41.3: Joint system validation in vehicle (2nd version)," DLR, Institute of Transportation Systems, Tech. Rep., 2010. [Online]. Available: http://elib.dlr.de/65782/

[41] M. Althoff, M. Koschi, and S. Manzinger, "CommonRoad: Composable benchmarks for motion planning on roads," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2017, pp. 719–726.