Safe Linear Encoding of Vehicle Dynamics for the Instantiation of Abstract Scenarios

Jan Steffen Becker

German Aerospace Center (DLR) e.V.
Institute of Systems Engineering for Future Mobility
Oldenburg, Germany
jan.becker@dlr.de

Abstract. In the automotive domain, scenario-based development is the answer to the increasing complexity of highly automated driving functions. Scenario-based methods cluster the large scenario space by so-called abstract scenarios which can be used to sample an infinite number of concrete scenarios. Because abstract scenarios constrain driving maneuvers by excessive use of additional constraints, constraint solving techniques may be required to find concrete scenario instances. In order to guarantee correctness of simulation runs with respect to the abstract scenario, realistic vehicle dynamics need to be considered during the instantiation process. This paper proposes a new encoding scheme for simple, but realistic vehicle dynamics into linear constraint systems for the generation of correct-by-construction concrete scenarios. The correctness of the construction is formally proven, and applicability shown by demonstrating the method on a set of basic driving maneuvers.

Keywords: scenario-based testing · simulation · vehicle dynamics · constraint solving · abstract scenario

1 Introduction

In the scenario-driven development of highly automated vehicles, scenarios are used at different levels of abstraction [24, 26]. This paper looks at two abstraction levels: *abstract* and *concrete* scenarios. While a concrete scenario [24] describes one specific event sequence in a fixed scenery, abstract scenarios [26] focus on the relations between traffic participants and the environment. Thereby, an abstract scenario may cover a variety of scenes and event sequences – it subsumes an infinite number of concrete scenarios. In a scenario-based testing process [27], for example, abstract scenarios are used to cluster the test space into adequate scenario classes. Scenario class mining may be data-driven or based on expert knowledge, and supported by criticality analysis [26]. Test execution, and experimental evaluation (e.g., in the context of criticality analysis), however, requires sampling of concrete scenarios from the scenario classes.

Formal languages for abstract scenarios, such as OpenSCENARIO DSL [33], SCENIC [14], or Traffic Sequence Charts [7], allow the user to specify complex

spatial and temporal relations between traffic participants. These various constraints may interact in a way that turns instantiation of abstract scenarios into a tough challenge. As a simple example, speed and distance constraints later on in a scenario can restrict valid placements of traffic participants at the beginning of the scenario.

As a quite generic solution, this work proposes to use constraint solving to generate concrete scenarios that satisfy all the constraints. Given an abstract scenario, the constraint system must guarantee that

- the concrete scenario is executable on a simulator with realistic vehicle dynamics, and
- the simulation conforms to the abstract scenario, i.e., all the constraints in the abstract scenario are fulfilled.

This requires that knowledge about vehicle dynamics is incorporated into the constraint solving problem. In this work an approach is presented, how both abstract scenarios and vehicle dynamics can be encoded into a linear constraint solving problem. The focus is on the encoding of vehicle dynamics.

The reminder of the paper is structured as follows: Section 2 summarizes related work, and Section 3 gives an overview on the SMT-based scenario instantiation approach. In Section 4, the complete construction for encoding vehicle dynamics is presented, followed by a correctness proof in Section 5. An experimental evaluation is described in Section 6. The paper concludes with a short discussion in Section 7.

2 Related Work

A range of description languages for traffic scenarios has been proposed so far [22]. Most of them follow the concepts defined by Ulbrich et al. [37], who define a scenario as a sequence of scenes, where the temporal evolution between the scenes is described by events and actions. Bach et al. [1] present a model-based scenario description language in a graphical modeling environment. Zhang et al. [38] decide for a textual language for scripting driving scenarios on two different abstraction levels. Menzel et al. [23] propose to derive so-called logical scenarios from natural language scenario descriptions in an engineering process. Logical scenarios are equipped with parameter ranges and probability distributions that allow to sample test cases that follow well-known characteristics. Schuldt et al. [35] propose testing strategies based on logical scenarios. ASAM OpenSCENARIO XML has evolved to a widely accepted industry standard for logical and concrete driving scenarios and has been adopted by academic and industrial driving simulators such as ESMINI, Vires VTD, and CARLA. These description languages all use a similar set of driving maneuvers as scenario building blocks. This allows an efficient execution by simulation software, but only a limited level of abstraction.

Neurohr et al. [27, 26] therefore propose the use of formalized abstract scenario descriptions. The authors present concepts for explointing abstract scenarios for testing [27] and criticality analysis [26, 21] of automated driving functions.

Textual domain-specific languages for abstract scenarios are M-DSL [11], OpenSCENARIO DSL [33], and Scenic [14]. These languages combine maneuvers and textual constraints. Because they still rely on maneuver actions, their interpretation is simulator-dependent. While Scenic comes with an execution engine that implements a guided simulation approach, the support of OpenSCENARIO DSL is still limited. Since version 0.9.15, the CARLA ScenarioRunner[1] implements a subset of OpenSCENARIO DSL.

Zone graphs [5] and traffic sequence charts (TSCs) [7, 8] are graphical specification languages for abstract scenarios. TSCs have a mathematically defined semantics [9], and have been proposed for a wide range of applications, including simulation [2] and monitoring [16]. In contrast to the aforementioned languages, TSCs rely exclusively on seamless declarative scene descriptions. Therefore, simulation requires constraint solving [3, 2]. Similar approaches are followed by, e.g., Eggers et al. [10], Klischat and Althoff [19], and Goyal et al. [15].

There are different approaches for solving constraint systems arising from traffic scenarios. A pure discrete approach is presented by Schwammberger [36]. Above cited work [10, 19, 3, 2] sketches some solutions that consider vehicle dynamics to some extend. General-purpose numeric solvers for hybrid systems such as PHAVer [12] and SpaceEx [13] generate precise solutions, but scale bad for the large state space induced by traffic scenarios. Plaku et al. [29, 30] overcome this limitation by combining simulation techniques with temporal model checking.

Bézier curves are widely used for path planning (e.g., in [17, 32, 6]). The cited works use them in combination with a single track vehicle dynamics model (sometimes also called bicycle model). Kong et al. [20] show experimentally that a more complex dynamic model (that considers tire forces) has no advantages over the simple model with respect to control design.

3 Encoding of Abstract Scenarios

According to Ulbrich et al., "a scenario describes the temporal development between several scenes in a sequence of scenes"[37]. This is realized differently in the different scenario description languages. In this paper, a simplified form of traffic sequence charts (TSCs) [7] is used as an example specification language. A TSC describes an abstract scenario as a sequence of so-called *spatial views*. A spatial view is a graphical representation of a scene with a well-defined mathematical semantics. Figure 1 shows a TSC describing an overtake maneuver. The placement of symbols (roads, crossings, vehicles) in a spatial view describes spatial relations between the depicted objects in both an x and y direction. For simplicity, we assume straight roads and orthogonal crossings, and describe spatial relations with respect to vehicle bounding boxes (in a Cartesian coordinate system) and lane boundaries. The dashed boxes are so-called somewhere boxes and are used to finer control the spatial relation—the objects outside the box are put in relation to the box borders, instead of the object inside the box.

[1] `https://github.com/carla-simulator/scenario_runner/tree/v0.9.15` accessed on 2024-04-19

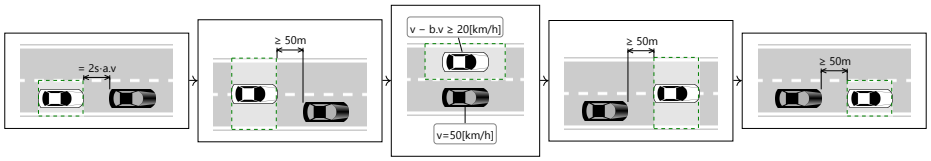


Fig. 1: An overtaking maneuver as TSC

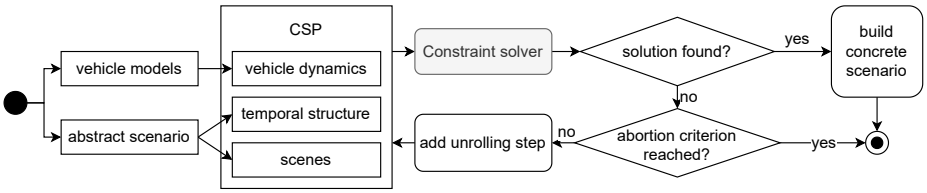


Fig. 2: Strategy for scenario instantiation

Example 1. Using the dashed lane marking as the x-axis of the global coordinate system, the first spatial view in Figure 1 can be expressed as

$$-w < b.\underline{\mathrm{y}} < b.\bar{\mathrm{y}} < 0 \wedge -w < a.\underline{\mathrm{y}} < a.\bar{\mathrm{y}} < 0 \wedge b.\underline{\mathrm{x}} - a.\bar{\mathrm{x}} = 2\,\mathrm{s} \cdot a.v\,,$$

where $\underline{\mathrm{x}}$, $\bar{\mathrm{x}}$, $\underline{\mathrm{y}}$, and $\bar{\mathrm{y}}$ denote bounding boxes (see Figure 3) of the white (a) and the black (b) vehicles. The variable w denotes lane width, so the interval $[-w, 0]$ is the y-range covered by the lane that contains the vehicles.

The encoding process for an abstract scenario into a constraint solving problem (CSP) follows the idea of bounded model checking and is displayed in Figure 2. First, we split the time domain of the scenario into N intervals $T_i = [t_{i-1}, t_i)$ (for $i = 1, 2, \ldots, N$) of fixed length $t_i - t_{i-1} = \Delta$, called slices. Note, that a scene may hold during more then one slice, so the number of slices can be larger than the number of scenes. Usually, one starts with the minimal number of slices, and adds further slices until a solution is found or an abortion criterion (e.g., maximum number of slices or timeout) reached. On each time slice $i = 1, 2, \ldots, N$, we describe each scene (indexed by j) in the scenario with a constraint ϕ^i_j. This constraint is further described later on in Section 4. The temporal structure of the abstract scenario is encoded with boolean state variables. A seamless sequence of five scenes – as in our running example – can be encoded by introducing a Boolean variable s^i_j for each scene $j = 1, 2, \ldots, 5$ and slice index $i = 1, 2, \ldots, N$ together with a constraint

$$\overbrace{\bigwedge_{i=1}^{N-1}\Big((s_1^{i+1} \to s_1^i) \wedge \bigwedge_{j=1}^{4}(s_{j+1}^{i+1} \to (s_{j+1}^i \vee s_j^i))\Big)}^{\text{(a)}} \wedge \overbrace{\Big(s_1^1 \wedge \bigwedge_{j=2}^{5} \neg s_j^1\Big)}^{\text{(b)}} \wedge \overbrace{s_5^N}^{\text{(c)}}\,.$$

Part (a) of the formula says that scene $j+1$ may be 'active' on slice $i+1$ only if itself or its predecessor was active on slice i. The sub-term $(s_1^{i+1} \to s_1^i)$ handles

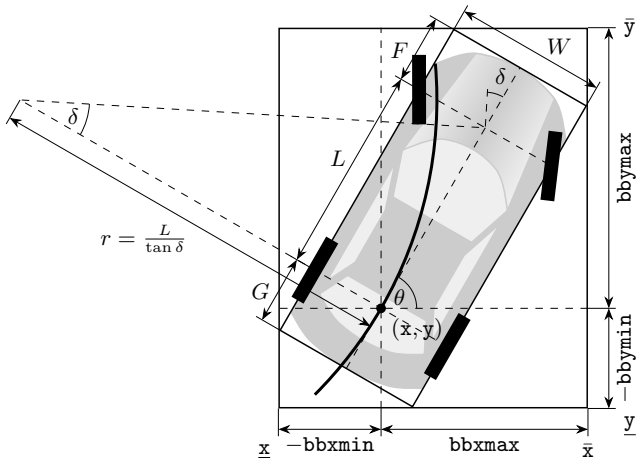


Fig. 3: The simple single track model

this rule for the first scene which has no predecessor. Furthermore, only the first scene is active on the first slice (part (b)), and the last scene in the end (part (c)). The link between the spatial and temporal aspects of the scenario is done by adding a series of implications $\bigwedge_{j,i} \left(sc_j^i \rightarrow \phi_j^i\right)$ ranging over all slices and scenes.

4 Encoding Vehicle Dynamics and Scenes

In this work, a simple single track vehicle dynamics model [34] is used. The single track model (Figure 3) assumes that the longitudinal axis of the vehicle is tangential to the vehicles trajectory, and touches the trajectory at the center of the (stable) rear axis, which is used as a reference point for the vehicle position. The direction of the longitudinal axis is called the *heading angle* θ and is at the same time the direction of movement. In steady state cornering, the vehicle follows a circle with radius $r = \frac{L}{\tan \delta}$ depending on wheel base L and steering angle δ.

4.1 Encoding Trajectories as Splines

We model vehicle trajectories as Bézier splines of fixed degree $n \geq 2$. Bézier curves [31] of degree n are defined by interpolation between $n+1$ points $\mathbf{p}_0, \ldots \mathbf{p}_n$, where $\mathbf{p}_0$ and $\mathbf{p}_n$ are endpoints of the curve, and $\mathbf{p}_1, \ldots, \mathbf{p}_{n-1}$ are bend-points.

Definition 1 (Bézier curve). *A Bézier curve of degree n with control points $\mathbf{p}_0, \ldots, \mathbf{p}_n \in \mathbb{R}^2$ is a parametric curve $\mathbf{p} : [0,1] \rightarrow \mathbb{R}^2$ given by*

$$\mathbf{p}(s) := \sum_{k=0}^{n} \mathcal{B}_k^n(s)\mathbf{p}_k \quad \textit{with} \quad \mathcal{B}_k^n(s) := \binom{n}{k}(1-s)^k s^{n-k}$$

called the Bernstein basis polynomial. A Bézier spline is a connected curve that is piecewise defined by Bézier curves.

Bézier curves have the following properties [31] that are of interest for this work:

(P1) A Bézier curve is always located within the polygon (convex hull) spanned by its control points.

(P2) Bézier curves are closed under linear combinations and affine transformations; Bézier curves of same degree can be combined/transformed by applying the transformation/combination (pairwise) to the control points.

(P3) The first derivative is again Bézier curve of degree $n-1$:

$$\dot{\mathbf{p}}(s) = \sum_{k=0}^{n-1} \mathcal{B}_k^{n-1}(s) n(\mathbf{p}_{k+1} - \mathbf{p}_k)$$

By applying *degree elevation*, we can express it as a Bézier curve of degree n with control points $\dot{\mathbf{p}}_0 = n(\mathbf{p}_1 - \mathbf{p}_0)$, $\dot{\mathbf{p}}_n = n(\mathbf{p}_n - \mathbf{p}_{n-1})$, and $\dot{\mathbf{p}}_k = k(\mathbf{p}_k - \mathbf{p}_{k-1}) + (n-k)(\mathbf{p}_{k+1} - \mathbf{p}_k)$ for $k = 1, \ldots, n-1$.

We describe the trajectories of vehicles by Bézier splines, using one spline segment per time slice. The spline uniquely defines both the path and the speed profile of a vehicle. A trajectory $\mathbf{p} : [t_0, t_N] \to \mathbb{R}^2$ of N slices is described by a sequence of $N \in \mathbb{N}$ Bézier curves $\mathbf{p}^1, \ldots, \mathbf{p}^N$ such that

$$\mathbf{p}(t) = \mathbf{p}^i(s) \qquad \text{for } t \in [t_{i-1}, t_i) \text{ and } s = \frac{t - t_{i-1}}{\Delta} \ . \tag{1}$$

Note that velocity and acceleration at time point $t \in [t_{i-1}, t_i)$ scale by Δ^{-1} (resp. Δ^{-2}) and are given by

$$\mathbf{v}(t) = \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{p}^i((t - t_{i-1})\Delta^{-1}) = \Delta^{-1}\dot{\mathbf{p}}^i(s) \text{ and } \mathbf{a}(t) = \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{v}(t) = \Delta_i^{-2}\ddot{\mathbf{p}}^i(s) \ . \tag{2}$$

4.2 Encoding of Scenes

In the following, we consider only a single time slice and vehicle. So, we can drop the superscripts and refer to the Bézier curve for the current slice as $\mathbf{p}$. It is described by $n+1$ control points $\mathbf{p}_0, \ldots, \mathbf{p}_n$. We encode the kth control point $\mathbf{p}_k$ by a pair of variables x_k and y_k. Furthermore, we approximate speed v, longitudinal acceleration acc, heading angle θ, and the bounding box dimensions $bound \in BB := \{\texttt{bbxmin}, \texttt{bbxmax}, \texttt{bbymin}, \texttt{bbymax}\}$ by save lower and upper bounds $bound^l$, $bound^u$ (for $bound \in BB$), $v^l(s)$, $v^u(s)$, $acc^l(s)$, $acc^u(s)$, θ^l, and θ^u (lower bounds are denoted by superscript l and upper bounds by superscript u). In case of bounding box dimensions and heading angle, these are variables; in case of v and acc these are Bernstein polynomials. For simplicity, w.l.o.g. we assume all used Bézier splines and Bernstein polynomials (including derivatives such as velocity and acceleration) having same the degree n. A practical implementation, however, will reduce the number of free variables by encoding velocity and acceleration bounds with at most n (resp. $n-1$) control points.

Given some scene as a monotone Boolean combination of linear constraints, we can utilize properties (P1) and (P2) of Bézier splines and conservatively approximate each linear constraint using the Bézier control points and lower/upper bounds introduced above.

Example 2. A sufficient condition for a 2-seconds distance between two vehicles a and b can be expressed as

$$\bigwedge_{k=0}^{n} \left((b.x_k + b.\texttt{bbxmin}^l) - (a.x_k + a.\texttt{bbxmax}^u) \geq 2\,\mathrm{s} \cdot (a.v_k^u - b.v_k^l) \right) \ .$$

4.3 Encoding of Vehicle Dynamics

The CSP constructed from the abstract scenario must ensure that

- the trajectory is possible within the single track model, and
- the variables, resp. values, $bound^l$, $bound^u$ (for $bound \in BB$), $v^l(s)$, $v^u(s)$, $acc^l(s)$, $acc^u(s)$, θ^l, and θ^u are valid lower/upper bounds for $bound(\theta(t))$, $v(t)$, $\dot{v}(t)$, and $\theta(t)$.

This is done by constructing some constraint system denoted Φ in the following, that is instantiated for each vehicle and time slice. It includes a special linear approximation of the single track model equations of motion.

Definition 2. *The single track model's equations of movement are*

$$\dot{x}(t) = v(t)\cos\theta(t), \qquad \dot{y}(t) = v(t)\sin\theta(t), \qquad \dot{\theta}(t) = v(t)\frac{\tan\delta(t)}{L}$$

and are subject to the constraints

$$|\delta(t)| \leq \delta_{\max} \quad \textit{and} \quad a_{lat}(t) := |\dot{\theta}(t)v(t)| \leq a_{lat,\max} := 0.4\mathrm{g} \ .$$

The construction for Φ uses a discretization of the heading angle θ. We choose a finite set $\mathcal{I}$ of heading intervals. Each interval $I = [\bar{\theta} - \eta, \bar{\theta} + \eta] \in \mathcal{I}$ is characterized by a *primary heading angle* $\bar{\theta} \in [-\pi, \pi]$ and a *heading angle deviation* $\eta \in [0, \pi/2)$. The selection of heading intervals depends on the abstract scenario. For example, a U-turn scenario requires that the union of heading intervals contains at least a half circle, while in an overtaking scenario a small heading range may be sufficient. The to-be-constructed constraint Φ ensures that on each slice the vehicle heading is within one of the intervals. From this interval, linear constraints for the vehicle movement within the slice are derived. In order to simplify the construction, we project the vehicle trajectory into a rotated coordinate system where the x-axis points into the primary heading direction.

Construction 1. *Given some interval $I = [\bar{\theta} - \eta, \bar{\theta} + \eta]$ and variables x_k, y_k (for $k = 0, \ldots, n$) describing Bézier curve control points, define*

$$X(I)_k := x_k \cos\bar{\theta} + y_k \sin\bar{\theta} \ \textit{and}\ Y(I)_k := -x_k \sin\bar{\theta} + y_k \cos\bar{\theta} \ .$$

Analogously, denote the control points of the projected first and second derivatives (see property (P3)) by $(\dot{X}(I)_k, \dot{Y}(I)_k)^T$ resp. $(\ddot{X}(I)_k, \ddot{Y}(I)_k)^T$.

In the rotated coordinate system, the vehicle moves roughly in X-direction, with a deviation of $\pm\eta$. Because the speed on a curve segment is determined by the distance between control points, the curvature increases when the vehicle accelerates (or brakes) on a curved segment (visually, the segment is skewed which leads to a sharper bend). In order to make handling of the curvature easier, the following construction distinguishes between straight and curved segments. Acceleration and deceleration along the rotated $X(I)$ axes are allowed on straight ($\eta = 0$) segments only, otherwise $\ddot{X}(I)$ is kept constant. Furthermore, a Boolean variable rws differentiates forward ($\mathsf{rws} = 0$) from reverse ($\mathsf{rws} = 1$) driving.

Construction 2. *Given a set $\mathcal{I}$ of heading intervals, the simple single track model is approximated by a constraint*

$$\Phi :\equiv \Phi_{cont} \wedge \bigvee_{I\in\mathcal{I}} ((\neg\mathsf{rws} \to \Phi_{fwd}(I)) \wedge (\mathsf{rws} \to \Phi_{rws}(I)))$$

with $\Phi_{fwd}(I) :\equiv \Phi_H(I) \wedge \Phi_{BB}(I) \wedge \Phi_{valid}(I) \wedge \Phi_v(I) \wedge \Phi_{acc}(I)$ *where*

$$\Phi_{cont} :\equiv \mathbf{p}_n{=}\mathbf{p}_0' \wedge \dot{\mathbf{p}}_n{=}\dot{\mathbf{p}}_0' \wedge (\dot{\mathbf{p}}_n{=}\mathbf{0} \to \theta^l{=}\theta^u{=}\theta^{l\prime}{=}\theta^{u\prime}) \wedge (\dot{\mathbf{p}}_n{\neq}\mathbf{0} \to (\mathsf{rws}{\leftrightarrow}\mathsf{rws}'))$$

$$\Phi_{BB}(I) :\equiv \bigwedge_{bound\in BB} \big(bound^l \le \min_{\theta\in I} bound(\theta) \wedge bound^u \ge \max_{\theta\in I} bound(\theta)\big)$$

$$\Phi_H(I) :\equiv [\theta^l, \theta^u] \supseteq I \wedge \bigwedge_{k=0}^{n} \big(\dot{X}(I)_k \ge 0 \wedge |\dot{Y}(I)_k| \le \dot{X}(I)_k \tan\eta\big)$$

$$\Phi_{valid}(I) :\equiv \bigwedge_{k=0}^{n} \left(\underbrace{\ddot{X}(I)_k = 0}_{\eta > 0 \ only} \wedge \Big|\frac{\ddot{Y}(I)_k}{\Delta^2}\Big| \le a_{lat,\max} \wedge \dot{X}(I)_k \ge \frac{2(n-1)L\tan(\eta)}{\tan(\delta_{\max})} \right)$$

$$\Phi_v(I) :\equiv \bigwedge_{k=0}^{n} \begin{cases} v_k^l = v_k^u = \Delta^{-1}\dot{X}(I)_k & \text{for } \eta = 0 \\ v_k^l \le \Delta^{-1}\dot{X}(I)_k \wedge v_k^u \ge \frac{\dot{X}(I)_k + \tan\left(\frac{\eta}{2}\right)|\dot{Y}(I)_k|}{\Delta} & \text{for } \eta > 0 \end{cases}$$

$$\Phi_{acc}(I) :\equiv \bigwedge_{k=0}^{n} \begin{cases} acc_k^l = acc_k^u = \Delta^{-2}\ddot{X}(I)_k & \text{for } \eta = 0 \\ acc_k^l \le -|\Delta^{-2}\ddot{Y}(I)_k| \wedge acc_k^u \ge |\Delta^{-2}\ddot{Y}(I)_k| & \text{for } \eta > 0 \end{cases}$$

The constraint $\Phi_{rws}(I)$ is the same as $\Phi_{fwd}(I)$ except for the following replacements: $\dot{X}(I)_k$ by $-\dot{X}(I)_k$, v_k^l by $-v_k^u$, and v_k^u by $-v_k^l$.

In the construction, Φ_{cont} ensures that the next segment starts with the heading and speed at the end of the current segment. On each segment, $\Phi_{fwd}(I)$ and $\Phi_{rws}(I)$ encode forward resp. backwards movement within a heading interval I. Here, $\phi_H(I)$ ensures that the movement direction on the current slice is within the given interval. The remaining constraints ensure that curvature and lateral acceleration are within validity range of the single track model ($\phi_{valid}(I)$), and that bounding box ($\phi_{BB}(I)$), velocity ($\phi_v(I)$), and acceleration ($\phi_{acc}(I)$) bounds are valid.

5 Correctness

This section sketches the correctness proofs for the construction. Recall that the constraint Φ from Construction 2 is instantiated for every slice and vehicle. Each solution of the resulting constraint system contains control point values that can be used to construct a vehicle trajectory according to Equation (1). In the following, it is shown that the generated trajectory is a solution for the simple single track model equations. Then, it is shown that the values for $v^l, v^u, acc^l, acc^u, \ldots$ are correct lower and upper bounds for velocity, acceleration, heading, and bounding box dimensions. For reasons of space, the proofs show the case for forward driving only. Proofs for reversing are analogous.

Lemma 1. *A trajectory* $\mathbf{p}$ *as defined in Equation* (1) *satisfies the simple single track model differential equations with*

$$v(t) = |\dot{\mathbf{p}}(t)|,\ \delta(t) = \arctan(\kappa'(t)L),\ \textit{and}\ \kappa'(t) = \frac{\dot{x}(t)\ddot{y}(t) - \ddot{x}(t)\dot{y}(t)}{(\dot{x}(t)^2 + \dot{y}(t)^2)^{3/2}}\ \textit{if}\ v(t) \neq 0$$

(else $\kappa'(t) = 0$*), provided the control points* $\mathbf{p}_k = \mathbf{p}_k^i$, $\mathbf{p}'_k = \mathbf{p}_k^{i+1}$ $(i = 0, \ldots, n)$ *satisfy* Φ *as defined in Construction 2.*

Proof. Given some time point t, call the ith slice, with $t_{i-1} \leq t \leq t_i$, the *current slice*. Assuming that the constraint Φ holds, there is some heading range $I_i = [\bar{\theta}_i - \eta_i, \bar{\theta}_i + \eta_i] \in \mathcal{I}$ such that $\Phi_H(I_i)$ and $\Phi_{valid}(I_i)$ hold. Furthermore, Φ_{cont} holds. The ODE system for the single track model can also be formulated as a linear ODE

$$\dot{\mathbf{h}}(t) = \begin{bmatrix} 0 & -v(t)\kappa(t) \\ v(t)\kappa(t) & 0 \end{bmatrix} \mathbf{h}(t) \qquad \dot{\mathbf{p}}(t) = v(t)\mathbf{h}(t) \tag{3}$$

with the constraint $|\mathbf{h}(t)| = 1$. It can be easily verified that the single track model ODEs solve the above linear ODE if $\mathbf{h}(t) = (\cos\theta(t), \sin\theta(t))^T$ and $\kappa(t) = \frac{\tan\delta(t)}{L}$. We show that the ODEs in Equation 3 hold on the current slice when setting

$$\mathbf{h}(t) = \begin{cases} (\cos\bar{\theta}_i, \sin\bar{\theta}_i)^T & \text{if } v(t) = 0 \\ \dot{\mathbf{p}}(t)/v(t) & \text{else.} \end{cases} \tag{4}$$

and $\delta(t)$ as in the Lemma. In case $v(t) \neq 0$, this is true by some basic analysis:

$$\begin{aligned} \dot{\mathbf{h}}(t) = \frac{d}{dt}\frac{\dot{\mathbf{p}}(t)}{v(t)} = \frac{d}{dt}\begin{bmatrix} \frac{\dot{x}(t)}{\sqrt{\dot{x}(t)^2+\dot{y}(t)^2}} \\ \frac{\dot{y}(t)}{\sqrt{\dot{x}(t)^2+\dot{y}(t)^2}} \end{bmatrix} &= \begin{bmatrix} \frac{\ddot{x}(t)\dot{y}(t)^2 - \ddot{y}(t)\dot{x}(t)\dot{y}(t)}{(\dot{x}(t)^2+\dot{y}(t)^2)^{3/2}} \\ \frac{\ddot{y}(t)\dot{x}(t)^2 - \ddot{x}(t)\dot{x}(t)\dot{y}(t)}{(\dot{x}(t)^2+\dot{y}(t)^2)^{3/2}} \end{bmatrix} \\ &= \begin{bmatrix} -\kappa'(t)\dot{y}(t) \\ \kappa'(t)\dot{x}(t) \end{bmatrix} = \begin{bmatrix} 0 & -v(t)\kappa(t) \\ v(t)\kappa(t) & 0 \end{bmatrix} \mathbf{h}(t) \end{aligned}$$

(note that $\kappa'(t) = \kappa(t)$). In case $v(t) = 0$, we have to show that $\dot{\mathbf{h}}(t) = 0$. Constraint $\Phi_{valid}(I)$ allows $v(t) = 0$ only if $\eta_i = 0$. By $\Phi_H(I)$ and Construction 1,

this implies that we are on a straight segment with $\dot{\mathbf{p}}(t') = (\cos\bar{\theta}_i, \sin\bar{\theta}_i)^T v(t')$, and therefore $\mathbf{h}(t') = (\cos\bar{\theta}_i, \sin\bar{\theta}_i)^T$ constant for all $t' \in [t_{i-1}, t_i]$. So, $\dot{\mathbf{h}}(t) = 0$.

The remaining part, showing that no discontinuities at time slice boundaries occur, is a direct consequence of the constraint Φ_{cont}. □

For the following lemmas and proofs, we fix some Bézier spline trajectory and construct $v(t)$ and $\delta(t)$ as in Lemma 1. Furthermore, we fix some time slice $[t_{i-1}, t_i)$ and some heading interval $I = [\bar{\theta} - \eta, \bar{\theta} + \eta]$ with $\bar{\theta} \in \mathbb{R}, \eta \in [0, \pi/2)$. Again, we drop the superscripts. Recall that Construction 1 uses a rotated version of the trajectory segment given by control points $(X_k, Y_k)^T$, depending on $\bar{\theta}$. We use it in form of Bernstein polynomials $X(s) := \sum_{k=0}^{n} \mathcal{B}_k^n(s) X_k$ and $Y(s) := \sum_{k=0}^{n} \mathcal{B}_k^n(s) Y_k$.

Lemma 2. *If both the heading constraint $\Phi_H(I)$ and the constraint $\Phi_{valid}(I)$ hold, then $|\kappa(t)| \leq \frac{\tan(\delta_{\max})}{L}$ and $|a_{lat}(t)| \leq a_{lat,\max}$ for all $t \in [t_{i-1}, t_i]$.*

Proof. Fix some $t \in [t_{i-1}, t_i]$ and let $s = (t - t_{i-1})\Delta^{-1}$. By property (P1) of Bézier curves, the constraints $\Phi_{valid}(I)$ and $\Phi_H(I)$ imply

$$\Delta v(t) = \sqrt{\dot{X}^2(s) + \dot{Y}^2(s)} \geq \dot{X}(s) \geq 0 \text{ and } |\dot{Y}(s)| \leq \dot{X}(s)\tan\eta$$

for all $s \in [0, 1]$ (recall that speed is invariant to rotation and, by Equation (2), scaled by Δ^{-1}). The curvature of the trajectory is invariant to rotation and, with the above inequalities, bounded by

$$|\kappa(t)| = \frac{|\dot{X}(s)\ddot{Y}(s) - \ddot{X}(s)\dot{Y}(s)|}{(\dot{X}^2(s) + \dot{Y}^2(s))^{3/2}} \leq \frac{|\ddot{Y}(s)| + |\ddot{X}(s)\tan\eta|}{\dot{X}^2(s) + \dot{Y}^2(s)} \leq \frac{|\ddot{Y}(s)| + |\ddot{X}(s)\tan\eta|}{\Delta^2 v^2(s)}$$

Similar, lateral acceleration $a_{lat}(t) = \kappa(t)v^2(t)$ is bounded by

$$a_{lat}(t) \leq \frac{|\ddot{Y}(s)| + |\ddot{X}(s)\tan\eta|}{\Delta^2} \ .$$

In the case $\eta = 0$, the bound $|\dot{Y}(s)| \leq \dot{X}(s)\tan\eta$ implies $\dot{Y}(s) = 0$ for all $s \in [0, 1]$, so $\ddot{Y}(s) = 0$. As a consequence of the above equations, both $a_{lat}(t) = 0$ and $\kappa(t) = 0$.

For $\eta > 0$, the constraint $\Phi_{valid}(I)$ forces $\dot{X}(s) > 0$ and $\ddot{X}(s) = 0$. This simplifies the above approximation for a_{lat} to $a_{lat}(t) \leq |\ddot{Y}(s)|\Delta^{-2}$. Furthermore, it means that $\dot{X}(s)$ is constant. Recall that $\dot{X}$ and $\dot{Y}$ are Bernstein polynomials of degree n with $\dot{X}_0 = \dot{X}_1 = \cdots = \dot{X}_{n-1} = \dot{X}(s) > 0$, and control points of $\dot{Y}$ can be bounded by $|\dot{Y}_i| \leq \dot{X}(s)\tan(\eta)$ for $i = 0, 1, \ldots, n-1$. By Bézier curve properties, $\ddot{Y}(s)$ is bounded by

$$|\ddot{Y}_i(s)| \leq (n-1)|\dot{Y}_{i+1} - \dot{Y}_i| \leq 2(n-1)\dot{X}(s)\tan(\eta) \ .$$

Using $\dot{X}(s)$ as a lower bound for $\Delta \cdot v(t)$, it is

$$|\kappa(s)| \leq \frac{2(n-1)\dot{X}(s)\tan(\eta)}{\dot{X}^2(s)} = \frac{2(n-1)\tan(\eta)}{\dot{X}(s)} \ .$$

With the bounds $\left|\frac{\ddot{Y}(s)}{\Delta^2}\right| \leq a_{lat,\max}$ and $\dot{X}(s) \geq \frac{2(n-1)L\tan(\eta)}{\tan(\delta_{\max})}$ implied by $\Phi_{valid}(I)$ follows $|a_{lat}(s)| \leq a_{lat,\max}$ and $|\kappa(t)| \leq \frac{\tan(\delta_{\max})}{L}$. □

Lemma 3. *If both the heading constraint $\Phi_H(I)$ and $\Phi_v(I)$ hold, then $v^l(t) \leq v(t) \leq v^u(t)$ for all $t \in [t_{i-1}, t_i]$.*

Proof. Fix some time point $t \in [t_{i-1}, t_i]$ and let $s := (t - t_{i-1})\Delta^{-1}$. Define $v_X := \Delta^{-1}\dot{X}(s)$ and $v_Y = \Delta^{-1}\dot{Y}(s)$ and recall that $v(t) = \sqrt{v_X^2 + v_Y^2}$ and $v_X \geq 0$. So, $v^l(t) \leq v_X \leq v(t)$.

In case $\eta = 0$, the constraint $|\dot{Y}(s)| \leq \dot{X}(s)\tan\eta$ implied by $\Phi_H(I)$ enforces $v_Y = 0$. Hence, $v(t) = v_X$ and by definition $v^u(t) = v_X = v(t)$.

The remaining case is $\eta > 0$. Recall (from the above proofs) that then $v_X > 0$. Because $v(t) = \sqrt{v_X^2 + v_Y^2} > 0$ and $0 < |v_Y| \leq v_X \tan\eta$ we can find some (unique) $\eta' \in [-\eta, +\eta]$ such that $v_X = v(t)\cos\eta'$ and $v_Y = v(t)\sin\eta'$. It is

$$\begin{aligned} v(t) &= v(t)\cos(\eta') + v(t)(1-\cos(\eta')) = v(t)\cos(\eta') + \frac{1-\cos(\eta')}{\sin(\eta')}v(t)\sin(\eta') \\ &= v_X + \frac{1-\cos(\eta')}{\sin(\eta')}v_Y = v_X + \tan\left(\frac{\eta'}{2}\right)v_Y \end{aligned}$$

and by monotony of tan on the interval $(-\pi, +\pi)$

$$\leq v_X + \tan\left(\frac{\eta}{2}\right)|v_Y| = \Delta^{-1}\dot{X}(s) + \Delta^{-1}\tan\left(\frac{\eta}{2}\right)|\dot{Y}(s)| \leq v^u(s)$$

by definition of $\Phi_v(I)$. So, the lemma holds. □

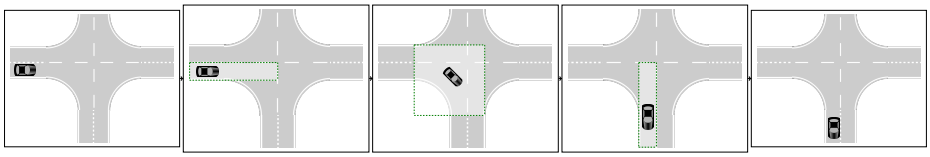
Lemma 4. *If both $\Phi_H(I)$, $\Phi_{valid}(I)$, and $\Phi_{acc}(I)$ hold, then $acc^l(t) \leq acc(t) \leq acc^u(t)$ for all $t \in [t_{i-1}, t_i]$.*

Proof. Fix some time point $t \in [t_{i-1}, t_i]$ and let $s := (t - t_{i-1})\Delta^{-1}$. The acceleration $acc(t) = \dot{v}(t)$ is invariant to rotation of the spline segment. Therefore

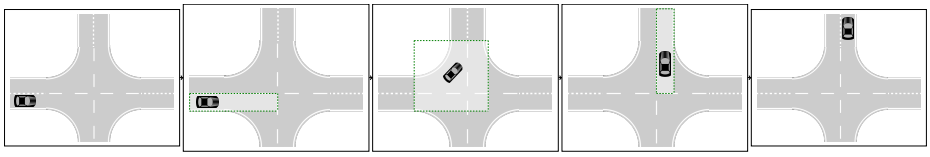
$$acc(t) = \frac{\ddot{X}(s)\dot{X}(s) + \ddot{Y}(s)\dot{Y}(s)}{\Delta^2\sqrt{\dot{X}(s)^2 + \dot{Y}(s)^2}} \ .$$

If $\eta = 0$, then by constraint $\Phi_H(I)$, it is $\dot{Y}(s) = \ddot{Y}(s) = 0$ and therefore $acc(t) = \Delta^2\ddot{X}(s) = acc^l(t) = acc^u(t)$. If $\eta > 0$, then $\Phi_{valid}(I)$ enforces $\ddot{X}(s) = 0$. Together with $\sqrt{\dot{X}(s)^2 + \dot{Y}(s)^2} \geq |\dot{Y}(s)|$ follows $|acc(t)| \leq \Delta^{-2}|\ddot{Y}(s)|$. The lemma follows with the definition of $\Phi_{acc}(I)$. □

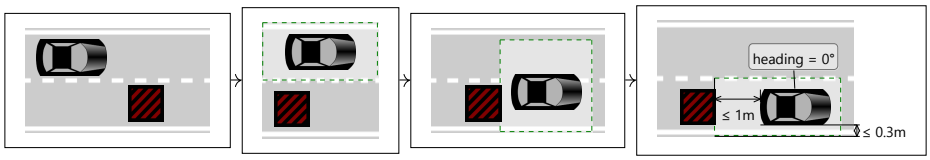
Corollary 1 (Correctnes of Construction 2). *Bézier splines satisfying Construction 2 are valid trajectories for the simple single track model, with valid bounds for velocity and acceleration.*



(a) right turn



(b) left turn



(c) parking

Fig. 4: Additional demonstration scenarios

6 Demonstration

In order to demonstrate the applicability of the approach, this section presents trajectories generated from four different abstract scenarios: an overtaking (Figure 1), a right turn (Figure 4a), a left turn (Figure 4b), and a parking scenario (Figure 4c). The four scenarios have been selected in order to contain a majority of the basic driving maneuvers that commonly occur in urban traffic scenarios. In particular, they map to the basic maneuvers identified by Hartjen et al. [18] as follows:

- Overtaking: keep velocity, follow lane, change lane, follow object[2], passing
- Left/right turn: follow lane, approach/cross junction, left/right turn[3]
- Parking: driveaway[4], halt, approach object, reversing, park

The vehicle dynamics encoding given in this paper has been integrated into the scenario instantiation framework for TSCs that has been presented in previous work [3, 2]. As a CSP solver, Z3 [25] is used. The TSCs (temporal structure and spatial views) given in Figures 1 and 4 are translated to BMC problems using the encoding scheme sketched in Section 3 and the related work [3, 2]. Vehicle dynamics are encoded with quadratic Bézier splines ($n = 2$) using Construction 2 given in this paper. Table 2 lists the heading intervals $\mathcal{I}$, unrolling depth N, and

[2] keep distance to other vehicle
[3] 90° turn on junction
[4] acceleration from standstill

Table 2: Heading intervals, number of slices (N), and slice length (Δ) for the demonstration scenarios

scenario	**intervals** $\mathcal{I}$	**#slices** N	**slice length** Δ
overtaking	$[0,0]^\circ$, $[-10,10]^\circ$	20	2 s
parking	$[0,0]^\circ$, $[-15,15]^\circ$, $[15,15]^\circ$, $[0,30]^\circ$	8	3 s
left turn	$[0,0]^\circ$, $[0,45]^\circ$, $[45,90]^\circ$, $[90,90]^\circ$	8	3 s
right turn	$[0,0]^\circ$, $[0,-45]^\circ$, $[-45,-90]^\circ$, $[-90,-90]^\circ$	8	3 s

step size Δ used in each scenario. All scenarios use a fictive vehicle model with a maximum speed of 120 km/h, (20 km/h when reversing), maximum acceleration of 1 ms^{-2} (both forward and reverse), and braking deceleration of $-5\,\mathrm{ms}^{-2}$. The single track model vehicle parameters (except heading ranges which are scenario-dependent) are listed in Figure 1. All scenarios use a lane width of 3 m. The crossing in the turning scenario has a corner radius of 6 m as shown in Figure 5.

For each of the SMT problems, the solver can find a satisfying model within less than 10 seconds[5]. The resulting trajectories are plotted in Figure 6. The blue shaded areas show the outer bounding box approximation calculated by the solver. Note that the inequalities $\Phi_{BB}(I)$ in Construction 2 allow the solver to assume any bounding approximation box that fits the vehicle, so the bounding boxes are not necessarily tight. It can be observed that the solver tends to choose the maximum bounding box that satisfies the current spatial view.

The presented concrete scenarios show that trajectories can be found for all basic maneuvers. The trajectories are correct by construction with respect to the chosen vehicle model and abstract scenario. However, the single track vehicle model simplifies actual vehicle dynamics. For example, side slip of the tires is

[5] The results have been produced using Z3 v4.8.14 running on a Windows 10 notebook with Intel Core i7-1185G7 CPU @ 3.00 GHz and 16 GB RAM.

Table 1: Vehicle parameters for the demonstration scenarios

parameter	**variable**	**value**
max. Ackermann steering angle	$\delta_{\max}$	45.0°
front	F	0.8 m
wheel base	L	2.8 m
rear	G	0.7 m
width	W	1.8 m

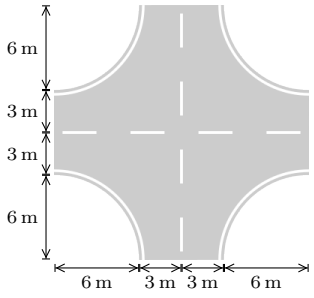
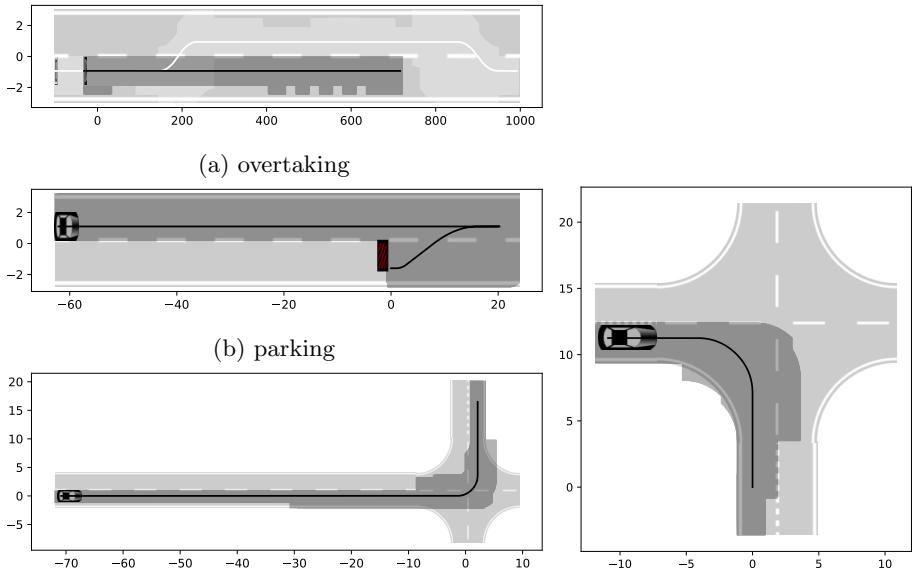


Fig. 5: Dimensions of the crossing in the demonstration scenarios



(a) overtaking

(b) parking

(c) left turn

(d) right turn

Fig. 6: Generated trajectories; the solid lines show vehicle trajectories, and the shaded areas the bounding box approximation; axes are in Meters

neglected. As a consequence, when a simulator executes the scenario with a more accurate vehicle physics model, the simulated trajectory may differ from the ideal one calculated by the CSP solver. In order to give an upper bound for the simulation error for the demonstration scenarios, the generated trajectories have been executed with the CARLA 14.0 driving simulator. As a vehicle, the CARLA built-in model of an Audi A2 has been chosen, which has nearly the same wheel base as our fictive vehicle. In the experiment, a custom controller implementation has been used that operates in the simulation loop. In each simulation step, the controller calculates throttle/brake and steering values by comparing the current actual and calculated vehicle states. Table 3 shows the maximum error in each experiment, in terms of lateral (lat) and longitudinal (lon) distance of the actual reference point to the calculated trajectory, heading angle (heading), and absolute distance error in the corners of the vehicle bounding box (bounds).

The results show that the lateral error is with a maximum of 4.0 cm in the right turn maneuver quite moderate. They are comparable to what has been achieved by Kong et al. [20] using a model predictive controller. This indicates that the single track model is adequate for trajectory planning. The comparatively larger longitudinal error is due to the sensitivity of the implemented controller with respect to acceleration changes, which are quite prominent in the overtaking maneuver for `other`. This can probably be reduced, e.g., by replacing the used discrete PID controller by a model predictive control algorithm

Table 3: Error between ideal and simulated trajectories

scenario (vehicle)	lat	lon	heading	bounds
overtaking (ego)	4.0 cm	14.9 cm	0.6°	14.9 cm
overtaking (other)	< 0.1 cm	11.8 cm	< 0.1°	11.9 cm
right turn (ego)	2.0 cm	3.6 cm	1.5°	9.0 cm
left turn (ego)	2.8 cm	10.0 cm	1.7°	10.0 cm
parking (ego)	0.5 cm	13.4 cm	0.4°	13.4 cm

[20]. The controller aims at minimizing the error in the vehicle reference point located at the rear axes. As a consequence, the bounding box error is dominated by the heading and longitudinal errors. Choosing another reference point in the simulation may reduce this effect.

7 Conclusion

This paper describes a method for encoding abstract scenarios to linear constraint solving/BMC problems. The encoding of vehicle dynamics, given in detail in Section 4, uses a simple single track model. The main ideas are to describe the vehicle trajectories as Bézier splines that are parameterized by time. So, the solution of the constraint problem determines the position of every vehicle to every point in time. Linear constraints give safe bounds for velocity and acceleration, by a discretization of the heading angle.

Due to the linearizations and over-approximations, heading intervals, number, and length of time slices must be appropriately chosen for the abstract scenario. While finer discretizations increase the covered scenario space, they also increase the complexity of the CSP problem. The practical demonstration in Section 6 shows that the approach can generate trajectories for a majority of basic driving maneuvers, and that the simple vehicle dynamics yields trajectories that can be executed under realistic driving physics. In this paper only one solution has been generated for each abstract scenario. The approach may be combined with model enumeration techniques (e.g. all-SMT [28] or recursive blocking [4]) in order to systematically generate different scenarios. Because of the enormous search space, strategies for finding an optimal coverage need to be developed. This subject to ongoing research.

So far, the approach has been applied to simulation [2] (where also first scalability results have been published) and consistency analysis [3] of TSCs. Future applications include the application to maritime and railway scenarios. This future research will address the above shortcomings of the approach.

References

1. Bach, J., Otten, S., Sax, E.: Model based scenario specification for development and test of automated driving functions. In: 2016 IEEE Intelligent Vehicles Symposium (IV). pp. 1149–1155 (2016). https://doi.org/10.1109/IVS.2016.7535534

2. Becker, J., Koopmann, T., Neurohr, B., Neurohr, C., Westhofen, L., Wirtz, B., Böde, E., Damm, W.: Simulation of Abstract Scenarios: Towards Automated Tooling in Criticality Analysis. pp. 42–51 (Feb 2022). https://doi.org/10.5281/zenodo.5907154
3. Becker, J.S.: Partial consistency for requirement engineering with traffic sequence charts. In: Automotive Software Engineering (ASE2020) (2020)
4. Bjørner, N., de Moura, L., Nachmanson, L., Wintersteiger, C.M.: Programming Z3, pp. 148–201. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-17601-3˙4
5. Butz, M., Heinzemann, C., Herrmann, M., Oehlerking, J., Rittel, M., Schalm, N., Ziegenbein, D.: Soca: Domain analysis for highly automated driving systems. In: 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC). pp. 1–6 (2020). https://doi.org/10.1109/ITSC45102.2020.9294438
6. Choi, J.w., Curry, R., Elkaim, G.: Path planning based on bézier curve for autonomous ground vehicles. In: Advances in Electrical and Electronics Engineering-IAENG Special Edition of the World Congress on Engineering and Computer Science 2008. pp. 158–166. IEEE (2008)
7. Damm, W., Kemper, S., Möhlmann, E., Peikenkamp, T., Rakow, A.: Using Traffic Sequence Charts for the Development of HAVs. In: ERTS 2018. 9th European Congress on Embedded Real Time Software and Systems (ERTS 2018), Toulouse, France (Jan 2018), https://hal.science/hal-01714060
8. Damm, W., Kemper, S., Möhlmann, E., Peikenkamp, T., Rakow, A.: Traffic sequence charts: A visual language for capturing traffic scenarios. In: Embedded Real Time Software and Systems - ERTS2018 (February 2018)
9. Damm, W., Möhlmann, E., Peikenkamp, T., Rakow, A.: A Formal Semantics for Traffic Sequence Charts, pp. 182–205. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-95246-8“˙11
10. Eggers, A., Stasch, M., Teige, T., Bienmüller, T., Brockmeyer, U.: Constraint systems from traffic scenarios for the validation of autonomous driving. In: Third International Workshop on Satisfiability Checking and Symbolic Computation, Part of FLOC 2018 (2018)
11. Foretellix Ltd.: Measurable scenario description language reference. Tech. rep., Foretellix Ltd. (2020), https://www.foretellix.com/wp-content/uploads/2020/07/M-SDL_LRM_OS.pdf, accessed on 2024-04-19
12. Frehse, G.: Phaver: Algorithmic verification of hybrid systems past hytech. In: International workshop on hybrid systems: computation and control. pp. 258–273. Springer (2005)
13. Frehse, G., Le Guernic, C., Donzé, A., Cotton, S., Ray, R., Lebeltel, O., Ripado, R., Girard, A., Dang, T., Maler, O.: Spaceex: Scalable verification of hybrid systems. In: International Conference on Computer Aided Verification. pp. 379–395. Springer (2011)
14. Fremont, D.J., Kim, E., Dreossi, T., Ghosh, S., Yue, X., Sangiovanni-Vincentelli, A.L., Seshia, S.A.: Scenic: a language for scenario specification and data generation. Machine Learning **112**(10), 3805–3849 (2023). https://doi.org/10.1007/s10994-021-06120-5, https://doi.org/10.1007/s10994-021-06120-5
15. Goyal, S., Griggio, A., Kimblad, J., Tonetta, S.: Automatic generation of scenarios for system-level simulation-based verification of autonomous driving systems. arXiv preprint arXiv:2311.09784 (2023)
16. Grundt, D., Köhne, A., Saxena, I., Stemmer, R., Westphal, B., Möhlmann, E.: Towards runtime monitoring of complex system requirements for autonomous driving functions (2022). https://doi.org/10.48550/arXiv.2209.14032

17. Han, L., Yashiro, H., Nejad, H.T.N., Do, Q.H., Mita, S.: Bezier curve based path planning for autonomous vehicle in urban environment. In: 2010 IEEE intelligent vehicles symposium. pp. 1036–1042. IEEE (2010)
18. Hartjen, L., Philipp, R., Schuldt, F., Friedrich, B., Howar, F.: Classification of driving maneuvers in urban traffic for parametrization of test scenarios. In: 9. Tagung Automatisiertes Fahren (2019)
19. Klischat, M., Althoff, M.: Synthesizing traffic scenarios from formal specifications for testing automated vehicles. In: 2020 IEEE Intelligent Vehicles Symposium (IV). pp. 2065–2072. IEEE (2020)
20. Kong, J., Pfeiffer, M., Schildbach, G., Borrelli, F.: Kinematic and dynamic vehicle models for autonomous driving control design. pp. 1094–1099 (06 2015). https://doi.org/10.1109/IVS.2015.7225830
21. Kramer, B., Neurohr, C., Büker, M., Böde, E., Fränzle, M., Damm, W.: Identification and quantification of hazardous scenarios for automated driving. In: Zeller, M., Höfig, K. (eds.) Model-Based Safety and Assessment. pp. 163–178. Springer International Publishing, Cham (2020)
22. Ma, J., Che, X., Li, Y., Lai, E.M.K.: Traffic scenarios for automated vehicle testing: A review of description languages and systems. Machines **9**(12), 342 (2021)
23. Menzel, T., Bagschik, G., Isensee, L., Schomburg, A., Maurer, M.: From functional to logical scenarios: Detailing a keyword-based scenario description for execution in a simulation environment. In: 2019 IEEE Intelligent Vehicles Symposium (IV). pp. 2383–2390 (2019). https://doi.org/10.1109/IVS.2019.8814099
24. Menzel, T., Bagschik, G., Maurer, M.: Scenarios for development, test and validation of automated vehicles. In: 2018 IEEE Intelligent Vehicles Symposium (IV). pp. 1821–1827. IEEE (2018). https://doi.org/10.1109/IVS.2018.8500406
25. de Moura, L., Bjørner, N.: Z3: An efficient SMT solver. In: Tools and Algorithms for the Construction and Analysis of Systems: 14th International Conference, TACAS 2008. Proceedings. pp. 337–340. Springer (2008). https://doi.org/10.1007/978-3-540-78800-3"˙24
26. Neurohr, C., Westhofen, L., Butz, M., Bollmann, M.H., Eberle, U., Galbas, R.: Criticality analysis for the verification and validation of automated vehicles. IEEE Access **9**, 18016–18041 (2021). https://doi.org/10.1109/ACCESS.2021.3053159
27. Neurohr, C., Westhofen, L., Henning, T., de Graaff, T., Mohlmann, E., Bode, E.: Fundamental considerations around scenario-based testing for automated driving. In: 2020 IEEE Intelligent Vehicles Symposium (IV). pp. 121–127. IEEE (2020). https://doi.org/10.1109/IV47402.2020.9304823
28. Phan, Q.S., Malacaria, P.: All-solution satisfiability modulo theories: Applications, algorithms and benchmarks. In: 2015 10th International Conference on Availability, Reliability and Security. pp. 100–109 (2015). https://doi.org/10.1109/ARES.2015.14
29. Plaku, E., Kavraki, L.E., Vardi, M.Y.: Hybrid systems: From verification to falsification. In: International Conference on Computer Aided Verification. pp. 463–476. Springer (2007)
30. Plaku, E., Kavraki, L.E., Vardi, M.Y.: Falsification of ltl safety properties in hybrid systems. International Journal on Software Tools for Technology Transfer **15**(4), 305–320 (2013)
31. Prautzsch, H., Boehm, W., Paluszny, M.: Bézier and B-spline techniques. Springer Science & Business Media (2013)
32. Qian, X., Navarro, I., de La Fortelle, A., Moutarde, F.: Motion planning for urban autonomous driving using bézier curves and mpc. In: 2016 IEEE 19th international conference on intelligent transportation systems (ITSC). pp. 826–833. Ieee (2016)

33. Rauschert, A., Amid, G.: ASAM OpenSCENARIO DSL 2.1.0: Release presentation. Presentation (2024), https://www.asam.net/standards/detail/openscenario-dsl/, last accessed on 2024-05-08
34. Schramm, D., Hiller, M., Bardini, R.: Single track models. In: Vehicle Dynamics, pp. 223–253. Springer (2014)
35. Schuldt, F., Reschka, A., Maurer, M.: A Method for an Efficient, Systematic Test Case Generation for Advanced Driver Assistance Systems in Virtual Environments, pp. 147–175. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-61607-0˙7, https://doi.org/10.1007/978-3-319-61607-0_7
36. Schwammberger, M.: An abstract model for proving safety of autonomous urban traffic. Theoretical Computer Science **744**, 143–169 (2018)
37. Ulbrich, S., Menzel, T., Reschka, A., Schuldt, F., Maurer, M.: Defining and substantiating the terms scene, situation, and scenario for automated driving. In: 2015 IEEE 18th international conference on intelligent transportation systems. pp. 982–988. IEEE (2015)
38. Zhang, X., Khastgir, S., Jennings, P.: Scenario description language for automated driving systems: A two level abstraction approach. In: 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC). pp. 973–980. IEEE (2020)