2019 IEEE Intelligent Vehicles Symposium (IV)
Paris, France. June 9-12, 2019

Generating Critical Test Scenarios for Automated Vehicles with Evolutionary Algorithms

Moritz Klischat and Matthias Althoff

Abstract— **Virtual testing of automated vehicles using simulations is essential during their development. When it comes to the testing of motion planning algorithms, one is mainly interested in challenging, critical scenarios for which it is hard to find a feasible solution. However, these situations are rare under usual traffic conditions, demanding an automatic generation of critical test scenarios. We present an approach that automatically generates critical scenarios based on a minimization of the solution space of the vehicle under test. By formulating a scenario parametrization and automatic determination of relevant parameter intervals, we are able to optimize the criticality of complex scenarios. We use evolutionary algorithms to tackle the resulting highly nonlinear optimization problem. Compared to our previous approach, we are now able to handle complex situations, in particular those involving intersections. Finally, we demonstrate our approach by generating critical scenarios from initially uncritical scenarios.**

I. Introduction

When automated vehicles are deployed in the real world, they are subjected to an unforeseeable number of situations. This requires extensive testing during their development to ensure safety, especially with regards to motion planning. To demonstrate that automated vehicles have a better performance than humans with a 95% confidence level, they need to be driven for 440 million km [1]. Because this cannot be achieved through real-world testing alone, virtual tests are a common practice, making it possible to test many aspects faster than in real-time, see e.g., [2]–[4]. Even virtual tests, however, can be very time-consuming since dangerous or interesting situations are rare events. This motivates automatic generation of critical test cases for automated vehicles—in this work, we focus on test cases for motion planning. These tests cases expose the vehicle under test to short traffic scenarios for which a feasible motion needs to be found.

A. Related work

One straightforward approach for increasing the efficiency of virtual testing is the extraction and classification of relevant scenarios from large databases of recorded traffic data as demonstrated in, e.g., [5], [6]. In [7], scenarios are grouped by unsupervised learning to find situations where small deviations of the environment lead to changes in behavioral modes, e.g., when the vehicle under test is forced to take a different path. Despite the fact that collecting data at this scale is challenging, one is restricted to observed situations, which are typically not critical most of the time.

All authors are with the Technische Universität München, Fakultät für Informatik, Lehrstuhl für Robotik, Künstliche Intelligenz und Echtzeitsysteme, Boltzmannstraße 3, 85748, Garching, Germany. {moritz.klischat, althoff}@in.tum.de

For generating new scenarios based on existing data, test cases are created automatically based on observed cause-effect relations, which are checked subsequently during test execution in [8]. Combination and mutation of recorded data is used in [9] to create new test cases. In [10] the authors propose using learned behavior from recorded or simulated traffic data to generate traffic scenarios with neural networks, which are subsequently searched for accidents. However, an accident does not necessarily imply that a situation is critical; it might have been easily avoided. In return, a critical situation might be resolved by a good driver and not be classified as critical.

Another practice for more systematic testing is to define scenarios on an abstract level and by deducting test cases through the variation of parameters, see e.g., [11], [12]. However, without considering the criticality of a scenario, the number of generated test cases quickly becomes uneconomical. Criticality is explicitly considered in [13]–[15], where automated vehicles are tested by optimizing scenarios towards a short Time-To-Collision or using related cost functions. Similarly, in [16], systems are forced into faulty behavior with respect to previously-defined specifications. In [17] and [18], test case generation for automated vehicles is realized using S-TaLiRo. In [19], an approach using S-TaLiRo is applied to motion planners based on machine learning, including simulated camera processing using deep neural networks.

Our previous work [20] is the first approach that generates critical scenarios with a small solution space for the vehicle under test, which we refer to as the ego vehicle from now on. In that work, the drivable area is used as a measure for criticality. The drivable area denotes the solution space in which the ego vehicle can operate safely without colliding. This measure enables the quantification of criticality in many more driving situations than the Time-To-Collision. However, [20] only allows the generation of simple scenarios on straight, non-intersecting roads and only realized translation of other traffic participants.

B. Contributions

We present an approach to generate critical scenarios for testing motion planners in complex traffic situations. Based on the optimization of the drivable area of the ego vehicle, our method provides the following improvements compared to [20]:

1) A new parametrization of scenarios is presented for realizing more complex and diverse scenarios.

978-1-7281-0560-4/19/$31.00 ©2019 IEEE

2) By using evolutionary algorithms (EA), we find better local minima over a larger range of scenarios. This is especially advantageous compared to the local linearization used in [20] despite the highly nonlinear optimization problem at hand.
3) To improve the optimization process and to consider only relevant driving situations, a method to prune the parameter space of a scenario is presented. As a result, only parameter regions where traffic participants can possibly interfere with the ego vehicle are considered.
4) By formulating a repair algorithm based on a linear program, obtained scenarios do not contain collisions among other traffic participants.

Our approach considers all feasible trajectories of the ego vehicle unlike previous approaches that generate scenarios based on a closed-loop simulation with the motion planner of the ego vehicle [13]–[15]. Thus, our approach yields test cases whose criticality is not depending on the performance of the motion planner. This enables a more objective comparison of multiple motion planners and the creation of a database with generic critical scenarios. Since we only generate scenarios with a non-empty drivable area, we also ensure that a collision-free trajectory exists, compared to approaches that focus on finding accident scenarios.

II. Problem Statement

The subsequent formulation of the underlying optimization problem is similar to [20], except that we consider collisions among other traffic participants. For an illustration of the introduced variables, we refer to Fig. 2a.

A. Traffic Participants

We define the list $\mathcal{V}$ of traffic participants $V^{(i)}$. In the remainder of this work, the superscript $\square^{(i)}$ refers to the i-th traffic participant. For each traffic participant, we assume a parametrized trajectory $x^{(i)}(t;p) \in \mathbb{R}^n$, where the parameter vector $p \in \mathcal{P}$ and parameter space $\mathcal{P} \in \mathbb{R}^{1\times q}$ are presented in detail in Sec. III-A. We introduce the operator $\mathrm{occ}(x)$ returning the occupied space of a vehicle with state x. The occupancy set of a traffic participant is denoted by $\mathcal{O}^{(i)}(t,p) = \mathrm{occ}(x^{(i)}(t;p))$. To each traffic participant we assign a lane $\mathcal{L}^{(i)} \in \mathbb{R}^2$ of the road network.

B. Motion Planning Problem of the Ego Vehicle

Let us introduce the motion planning problem for the ego vehicle as follows: By $f(x(t),u(t))$ we denote the right-hand side of the state-space model of the ego vehicle so that

$$\dot{x}(t) = f(x(t),u(t)),$$

where $x \in \mathbb{R}^n$ is the state vector and $u \in \mathbb{R}^m$ is the input vector. We further require the initial state $x_0 = x(t_0)$, the initial time t_0, and the time horizon t_f. The possibly time-varying, allowed space on the road surface is denoted by $\mathcal{W}_{\mathrm{lanes}}(t) \subset \mathbb{R}^2$. The occupancy of the ego vehicle has to lie within the allowed space, while avoiding other traffic participants $\forall t \in [t_0,t_f] : \mathrm{occ}(x(t)) \subseteq \mathcal{W}_{\mathrm{lanes}}(t)\backslash\mathcal{O}(t,p)$. We also require constraints $g(x(t),u(t),t) \leq 0$, such as speed limits or other traffic rules [21].

After introducing an input trajectory as $u(\cdot)$ (in contrast to a value $u(t)$ at time t) and the cost function of the obtained solution $J(x(t),u(t),t_0)$, we can finally formulate the motion planning problem as finding

$$u^*(\cdot) = \underset{u(\cdot)}{\arg\min}\ J(x(t),u(t),t_0)$$

subject to

$$\begin{aligned} &\dot{x}(t) = f(x(t),u(t)), \quad \mathrm{occ}(x(t)) \subseteq \mathcal{W}_{\mathrm{lanes}}(t)\backslash\mathcal{O}(t,p),\\ &g(x(t),u(t),t) \leq 0, \quad x(t_0) = x_0\,. \end{aligned} \tag{1}$$

Finally, we denote a scenario by the tuple

$$S(p) = (x_0, \mathcal{O}(\cdot,p), \mathcal{W}_{\mathrm{lanes}}(\cdot)).$$

C. Drivable Area

To consider not only the optimal solution of the motion planning problem, but the space of all solutions, we require the set of reachable states [22]. In particular, we use a so-called *anticipated reachable set*, which excludes states that will inevitably result in an accident [23] in the time interval $t \in [t_0,t_f]$. We denote a feasible trajectory as $\chi(t;x_0,u(\cdot))$, which meets all constraints in (1). After introducing the set of input trajectories $\mathcal{U}$, we define the anticipated reachable set as

$$\mathcal{R}(t;x_0,\mathcal{O}(\cdot,p),\mathcal{W}_{\mathrm{lanes}}(\cdot),t_f) = \Big\{ \chi(t;x_0,u(\cdot)) \Big| \exists u(\cdot) \in \mathcal{U}, \\ \forall \tau \in [t_0,t_f] : \mathrm{occ}\big(\chi(\tau;x_0,u(\cdot))\big) \subseteq \mathcal{W}_{\mathrm{lanes}}(\tau)\backslash\mathcal{O}(\tau,p) \Big\}.$$

By applying the projection operator for projecting to the position domain in Euclidean space $\underset{xy}{\mathrm{proj}}(x) : \mathbb{R}^n \to \mathbb{R}^2$, the drivable area becomes

$$\mathcal{D}(t;x_0,\mathcal{O}(\cdot,p),\mathcal{W}_{\mathrm{lanes}}(\cdot)) = \\ \bigcup_{x\in\mathcal{R}(t;x_0,\mathcal{O}(\cdot,p),\mathcal{W}_{\mathrm{lanes}}(\cdot),t_f)} \underset{xy}{\mathrm{proj}}(x). \tag{2}$$

To quantify the solution space over time, we introduce the function $\mathrm{area}(\mathcal{X}) : \mathbb{R}^2 \to \mathbb{R}^+$, returning the area of a set. We write

$$A(p,t) \coloneqq \mathrm{area}\big(\mathcal{D}(t;x_0,\mathcal{O}(\cdot,p),\mathcal{W}_{\mathrm{lanes}}(\cdot))\big)$$

to obtain the area profile of the drivable area over time.

D. Optimization Problem

The goal of this work is to create a scenario $S(p)$ with a desired area profile $A_{ref}(t)$ by optimizing

$$\underset{p}{\arg\min}\,\kappa(S(p)), \quad \kappa(S(p)) = \int_0^{t_f} \big(A(p,t) - A_{ref}(t)\big)^2 \tag{3}$$

$$\text{subject to} \qquad \forall t, \forall i, \forall j : \quad \mathcal{O}^{(i)}(t,p) \cap \mathcal{O}^{(j)}(t,p) = \emptyset. \tag{4}$$

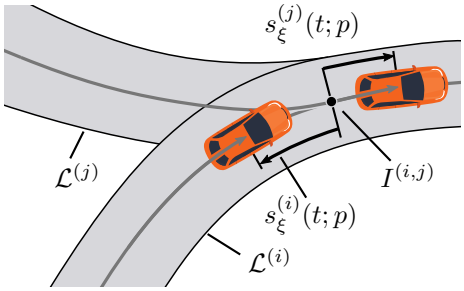


Fig. 1: Longitudinal coordinates $s_\xi(t_k;p)$ formulated relative to the intersection point $I^{(i,j)}$ of lanes $\mathcal{L}^{(i)}$ and $\mathcal{L}^{(j)}$.

The constraint in (4) ensures that no traffic participants collide with each other. We refer to the set of parameters, for which this constraint holds, as the *feasible set*. In this work, we use for $A_{ref}(t)$ the drivable area computed without any traffic participants and the scalar $\gamma \in]0,1[$, which quantifies the reduction of the drivable area: $A_{ref}(t) = \gamma \cdot \text{area}(\mathcal{D}(t; x_0, p, \emptyset, \mathcal{W}_{\text{lanes}}(\cdot)))$.

III. Parametrization of the Optimization Problem

For the trajectories $x^{(i)}(t;p)$ of all traffic participants $\mathcal{V}$, we require a parametrization that can be applied to complex road networks and enables efficient handling of collision constraints in (4), yet is lightweight enough for being solved in reasonable time. To this end, we introduce the curvilinear coordinate system C, in which a state is defined as $x_C = [s_\xi, \dot{s}_\xi, s_\eta, \dot{s}_\eta]^T$, where ξ denotes longitudinal and η denotes lateral coordinates with respect to the center line of a lane $\mathcal{L}$. The operator $\underset{\xi}{\text{proj}}(x) : \mathbb{R}^2 \to \mathbb{R}$ projects the Euclidean space to the longitudinal position domain. Furthermore, we use discretized time $t_k = \Delta t \cdot k$ with time steps $k \in \mathbb{N}$ and step size $\Delta t \in \mathbb{R}^+$.

A. Parametrization of Traffic Participants

For each vehicle, we assume an initial trajectory to be given and independent dynamics in lateral and longitudinal direction. We parametrize the longitudinal position trajectory as

$$s_\xi^{(i)}(t_k; p^{(i)}) = \hat{s}_\xi(t_k) + p_s^{(i)} + t_k p_v^{(i)} + \frac{1}{2} t_k^2 p_a^{(i)} \quad (5)$$

with the initial longitudinal trajectory $\hat{s}_\xi(t_k) \in \mathbb{R}$ and the parameter vector

$$p = \left[p_s, p_v, p_a\right]$$

for translations $p_s \in \mathbb{R}^{1\times\nu}$, initial velocity variations $p_v \in \mathbb{R}^{1\times\nu}$, and acceleration variations $p_a \in \mathbb{R}^{1\times\nu}$. The parameter p is bounded by the multidimensional interval set

$$\mathcal{P} = \{[p_s, p_v, p_a] \mid p_s \in \mathcal{S}, p_v \in \mathcal{B}, p_a \in \mathcal{A}\}$$

with

$$\mathcal{S} = \left[\underline{p_s}, \overline{p_s}\right], \mathcal{B} = \left[\underline{p_v}, \overline{p_v}\right], \mathcal{A} = \left[\underline{p_a}, \overline{p_a}\right].$$

By $\overline{\square}$ we denote the supremum and by $\underline{\square}$ the infimum of interval sets.

B. Collision Constraints

In order to solve the collision constraints in (4) efficiently, we approximate them by a formulation as linear inequality constraints of the form $d(t_k, p) \leq \Delta r$ with $d(t_k, p), \Delta r \in \mathbb{R}^\rho$, which represent a convex feasible set. We introduce the solution candidate $\tilde{p} \in \mathcal{P}$, which is repaired using an Euclidean projection of $\tilde{p}$ onto the convex feasible set. This projection is trivial and can be solved by a quadratic program [24]. The repair mechanism is used during optimization as described in Sec. IV.

The elements of $d(t_k, p)$ are obtained by pair-wise formulations of collision constraints between traffic participants. Due to the scenario parametrization in (5), not all traffic participants can collide, e.g., if the lanes of two vehicles never intersect. Therefore, we first identify all pairs of traffic participants $V^{(i)}, V^{(j)}, \forall i \neq j$, which can possibly collide by checking for intersection of their lanes, $\mathcal{L}^{(i)} \cap \mathcal{L}^{(j)}$.

Constraints are composed of the distances between vehicles along lanes. While obtaining distances between traffic participants in the same lane is trivial, defining longitudinal distances of merging or intersecting lanes is not obvious. To this end, we define the intersection point of lanes of traffic participants $V^{(i)}$ and $V^{(j)}$ as $I^{(i,j)} \in \mathbb{R}^2$ as shown in Fig. 1, which serves as the origin of the curvilinear coordinate systems. Longitudinal distances in these coordinate systems are then classically obtained as $|s_\xi^{(i)}(t_k;p) - s_\xi^{(i)}(t_k;p)|$.

Depending on $\tilde{p}$, two configurations of two traffic participants are possible: either $V^{(i)}$ is passing before (case ①) or behind (case ②) $V^{(j)}$. In order to preserve the configuration, which results from the parameter $\tilde{p}$ at hand, we distinguish the cases by

$$d^{(i,j)}(t_k, p) = \begin{cases} s_\xi^{(j)}(t_k;p) - s_\xi^{(i)}(t_k;p) & \text{for case ①} \\ s_\xi^{(i)}(t_k;p) - s_\xi^{(j)}(t_k;p) & \text{for case ②} \end{cases}.$$

Finally, we write the collision constraint for each t_k as

$$d^{(i,j)}(t_k, p) \leq (r^{(i)} + r^{(j)}), \quad (6)$$

where r is the radius of the circle inscribing the shape of a traffic participant, including a safety margin.

C. Pruning of the Parameter Space

Solving our optimization problem is complicated since a traffic participant can only reduce the drivable area $\mathcal{D}$ if it intersects it at some point in time. However, large regions of possibly occupied positions $\{\mathcal{O}(\cdot, p) \mid p \in \mathcal{P}\}$ of traffic participants might never intersect the drivable area $\mathcal{D}$ like vehicle $V^{(1)}$ in Fig. 2a. To quickly converge to solutions reducing the drivable area, we automatically want to remove regions from the translational parameter space $\mathcal{S}$ which have no influence on the drivable area, as shown in Fig. 2b for $V^{(1)}$. In contrast, there exist parameters within $\mathcal{P}$ that can drastically reduce the drivable area; this typically depends largely on the traffic participant. Thus, we first identify the best traffic participants $\mathcal{V}^B$ which have a large influence on

the reduction of the drivable area and separate them from the worse remaining traffic participants $\mathcal{V}^W$:

$$\mathcal{V} = \mathcal{V}^B \cup \mathcal{V}^W . \tag{7}$$

From now on, we denote by superscripts $\square^B$ and $\square^W$ the relation to traffic participants of the corresponding sets.

For identifying traffic participants with a large influence on the drivable area, i.e., the cost function κ, we define the criterion

$$\lambda(V^{(i)}, \tilde{p}) = \frac{\kappa(S(x_0, \mathcal{O}(\cdot, \tilde{p}), \mathcal{W}_{\text{lanes}}(\cdot)))}{\kappa(S(x_0, \hat{\mathcal{O}}(\cdot, \tilde{p}), \mathcal{W}_{\text{lanes}}(\cdot))},$$
$$\hat{\mathcal{O}}^{(i)}(\cdot, \tilde{p}) = \mathcal{O}(\cdot, \tilde{p}) \setminus \mathcal{O}^{(i)}(\cdot, \tilde{p}),$$

which expresses the ratio of costs for the scenario with and without the i-th traffic participant for the parameter $\tilde{p}$. As shown in Algorithm 1, $\mathcal{V}^B$ is initially empty and new members $\mathcal{V}_{b,new}$ are selected by

$$V_{b,new} = \underset{V^{(i)} \in \mathcal{V}^W}{\arg\min}\, \lambda(V^{(i)}, \tilde{p}) \tag{8}$$

to obtain the traffic participant with the largest impact on the cost function.

After adding $V_{b,new}$ to $\mathcal{V}^B$, we want to identify the intervals $\mathcal{S}^{W,(i)} = [\underline{p_s}^{(i)}, \overline{p_s}^{(i)}]$ for the remaining $V^{(i)} \in \mathcal{V}^W$, which contain all parameters that can intersect with $\mathcal{D}(\cdot, x_0, \mathcal{O}(\cdot, [\tilde{p}^B, p^W]), \mathcal{W}_{\text{lanes}}(\cdot))$. By computing $\mathcal{D}^B(\cdot, x_0, \mathcal{O}^B(\cdot, \tilde{p}^B), \mathcal{W}_{\text{lanes}}(\cdot))$ considering the occupancies $\mathcal{O}^B(\cdot, \tilde{p}^B)$ only, we obtain a superset of the drivable area

$$\forall p^W \in \mathcal{P}^W : \mathcal{D}(\cdot, x_0, \mathcal{O}(\cdot, [\tilde{p}^B, p^W]), \mathcal{W}_{\text{lanes}}(\cdot)) \subseteq \mathcal{D}^B(\cdot, x_0, \mathcal{O}^B(\cdot, \tilde{p}^B), \mathcal{W}_{\text{lanes}}(\cdot)), \tag{9}$$

where $[\tilde{p}^B, p^W]$ denotes the combined parameter vector of $\tilde{p}^B$ and p^W. From now on, the shortened notation $\mathcal{D}^B(\cdot, \tilde{p}^B) = \mathcal{D}^B(\cdot, x_0, \mathcal{O}^B(\cdot, \tilde{p}^B), \mathcal{W}_{\text{lanes}}(\cdot))$ is used. By restricting $\mathcal{S}^W$ to the interval which results in an intersection with $\mathcal{D}^B(\cdot, \tilde{p}^B)$, we consequently guarantee that $\mathcal{S}^W$ contains the desired parameters only. This is formalized as

$$\underline{p_s}^{(i)} = \min \{p_s^{(i)} \mid \mathcal{O}^{(i)}(\cdot, [p_s^{(i)}, p_v^{(i)}, p_a^{(i)}]) \cap \mathcal{D}^B(\cdot, \tilde{p}^B) \neq \emptyset, \forall p_v^{(i)} \in \mathcal{B}^{(i)}, \forall p_a^{(i)} \in \mathcal{A}^{(i)}\},$$

$$\overline{p_s}^{(i)} = \max \{p_s^{(i)} \mid \mathcal{O}^{(i)}(\cdot, [p_s^{(i)}, p_v^{(i)}, p_a^{(i)}]) \cap \mathcal{D}^B(\cdot, \tilde{p}^B) \neq \emptyset, \forall p_v^{(i)} \in \mathcal{B}^{(i)}, \forall p_a^{(i)} \in \mathcal{A}^{(i)}\}.$$

Since this is a challenging problem, we formulate an over-approximative representation using an over-approximation of $\mathcal{D}^B(\cdot, \tilde{p}^B)$ in curvilinear coordinates, thus guaranteeing that the entire drivable area is considered when computing $\mathcal{S}^W$. For that purpose, we intersect the drivable area $\mathcal{D}^B(\cdot, \tilde{p}^B)$ with the corresponding lane $\mathcal{L}^{(i)}$ and project it to the longitudinal position domain to obtain the over-approximative interval of $\mathcal{D}^B(\cdot, \tilde{p}^B)$ in curvilinear coordinates (see Fig. 2b):

$$d_\xi^{(i)}(t_k) = \left[\inf \left\{ \underset{\xi}{\text{proj}} \left(\mathcal{D}^B(t_k, \tilde{p}^B) \cap \mathcal{L}^{(i)} \right) \right\}, \sup \left\{ \underset{\xi}{\text{proj}} \left(\mathcal{D}^B(t_k, \tilde{p}^B) \cap \mathcal{L}^{(i)} \right) \right\} \right].$$

Afterwards, we obtain the bounds of $\mathcal{S}^{W,(i)}$ for the time interval $[t_0, t_f]$ as

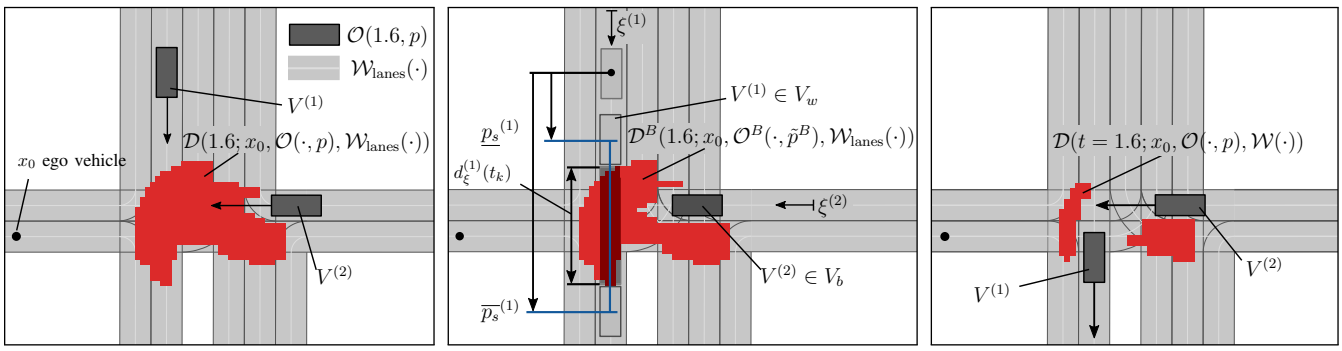
$$\underline{p_s}^{(i)} = \min_{t_k \in [t_0, t_f]} \Big(\underline{d_\xi}^{(i)}(t_k) - \hat{s}_\xi(t_k) \underbrace{-(t_k - t_0)\overline{p_v} - \frac{1}{2}(t_k - t_0)^2 \overline{p_a}}_{\text{minimizes } \underline{p_s}^{(i)}} \Big), \tag{10}$$

$$\overline{p_s}^{(i)} = \max_{t_k \in [t_0, t_f]} \Big(\overline{d_\xi}^{(i)}(t_k) - \hat{s}_\xi(t_k) \underbrace{-(t_k - t_0)\underline{p_v} - \frac{1}{2}(t_k - t_0)^2 \underline{p_a}}_{\text{maximizes } \overline{p_s}^{(i)}} \Big) \tag{11}$$

which directly follows from solving (5) for $p_s^{(i)}$.

IV. Optimization Routine

For solving the optimization problem in (3), we utilize evolutionary algorithms. These algorithms are especially suited for global optimization problems for which no analytic gradient can be formulated [6], as is the case for the cost function $\kappa(S(p))$. One can incorporate any EA in our approach as shown in Algorithm 1; we exemplarily



(a) Initial scenario. (b) Pruning parameter region $\mathcal{S} = [\underline{p_s}, \overline{p_s}]$. (c) Solution of optimization: critical scenario.

Fig. 2: Generating a critical scenario. Depicted are drivable area D and traffic participants $V^{(i)}$ at $t = 1.6\,\text{s}$.

use differential evolution (DE) [25] and particle swarm optimization (PSO) [26] and compare them.

During optimization, we perform intermediate tightening of parameter bounds $\mathcal{P}$ according to Sec. III-C. For the optimization, we define the population Q as the set consisting of all n_p solution candidates p_e, $e \in \{0, ..., n_p\}$ for the EA.

Initially, all traffic participants are assigned to $\mathcal{V}^W$, and initial bounds are computed according to Sec. III-C. After conducting n_b iterations with the respective solver, all solution candidates $p_e \in Q$ which violate the constraints in (6) are repaired by computing the closest projection to the feasible set as described in Sec. III-B. Afterwards, intermediate updating of parameter bounds $\mathcal{S}$ is performed by first selecting the most relevant member $V_{b,new} \in \mathcal{V}^W$ with respect to $\kappa(S(p))$ for adding it to $\mathcal{V}^B$. With the updated sets $\mathcal{V}^B$ and $\mathcal{V}^W$, the parameter bounds $\mathcal{S}^W$ are tightened in the subsequent iteration step l using (10) and (11). Next, elements of solution candidates $p_e \in Q$, which violate $p_e \in \mathcal{P}$, are resampled within the newly computed bounds. This routine is repeated until either all traffic participants are assigned to $\mathcal{V}^B$ or the optimization algorithm converged. For a step-by-step example that illutstrates the iterative computation of bounds, we refer to Sec. V-A.

Algorithm 1 IterativeBoundingOptimization

Require: scenario $S(p)$, initial solution $\tilde{p}$, traffic participants $\mathcal{V}$, number of traffic participants n_o, initial bounds $\mathcal{P}$
Ensure: Critical Scenario S

$\mathcal{V}^W \leftarrow \mathcal{V}$
$\mathcal{V}^B \leftarrow \emptyset$
$Q \leftarrow$ INITPOPULATION
$l \leftarrow 0$
$converged \leftarrow false$
while $l < n_o$ and $\neg converged$ **do**
  $\mathcal{P}^W \leftarrow$ TIGHTENPARAMETERBOUNDS$(\mathcal{V}^W, \mathcal{V}^B, \tilde{p})$ ▷ see (10) and (11)
  $Q, converged \leftarrow$ EA$(\mathcal{P}, Q)$
  $Q \leftarrow$ REPAIRINFEASIBLE(Q) ▷ see Sec. III-B
  $\tilde{p} \leftarrow \arg\min_{p_e \in Q} \kappa(S(p_e))$
  $V_{b,new} \leftarrow$ SELECTRELEVANT$(\mathcal{V}^W, \mathcal{V}^B, \tilde{p})$ ▷ see (8)
  $\mathcal{V}^B \leftarrow \mathcal{V}^B \cup V_{b,new}$
  $\mathcal{V}^W \leftarrow \mathcal{V}^W \backslash V_{b,new}$
  $l \leftarrow l + 1$
end while
return $S(\tilde{p})$

V. RESULTS

The proposed approach is demonstrated by two initial scenarios from the CommonRoad benchmark collection [27]. We also compare results from the two evolutionary algorithms DE and PSO. Due to the lack of comparable algorithms, no comparison to existing work is possible. For the computation of the drivable area, we use the method presented in [23]. Used parameters are listed in Table I. We used different settings for the solvers in both scenarios due to their different complexities. The settings are described in the respective paragraphs. Computation times were measured on a machine with an Intel i7-8650U 1.90 GHz processor.

TABLE I: Scenario Parameters

max. acceleration ego vehicle $\lvert a_{max} \rvert$	$5.0\,\mathrm{m/s^2}$
time step size Δt	$0.1\,\mathrm{s}$
time horizon t_f	$3.0\,\mathrm{s}$
initial velocity variation $\mathcal{B}$	$[-3, 3]\,\mathrm{m\,s^{-1}}$
acceleration variation $\mathcal{A}$	$[-5, 5]\,\mathrm{m/s^2}$

A. Scenario I: Intersection

The first scenario is a hand-crafted, unregulated urban intersection that can be found in the CommonRoad benchmark collection[1] under ID DEU_Ffb-1_3_T-1. The ego vehicle has an initial velocity $v_0 = 7.1\,\mathrm{m\,s^{-1}}$ and is surrounded by 3 other vehicles. This results in 9 parameters, for which a population of 90 individuals is used during the optimization. We conduct 45 iterations with both solvers, while performing parameter bounding every $n_b = 15$ steps. The computation time for this scenario is 22.09 minutes.

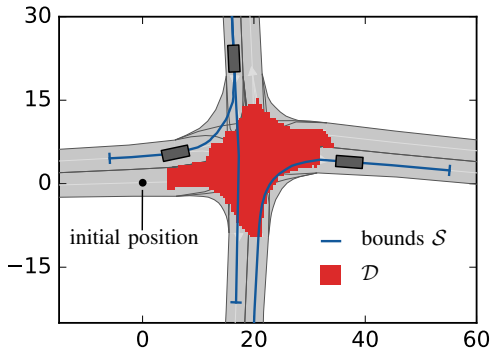
The initial configuration at $t = 2.5\,\mathrm{s}$ is depicted in Fig. 3a. The other vehicles almost do not restrict the drivable area $\mathcal{D}$ and thus the situation is uncritical. The final result of the particle swarm algorithm is shown in Fig. 3. To illustrate the optimization routine, we show three intermediate solutions at different iterations of the optimization. In the initial configuration, the depicted position bound $\mathcal{S}$ for all vehicles are large due to the large drivable area. However, after two adaptation iterations, the bounds could be decreased due to the smaller drivable area. After the final iteration $k = 3$, the drivable area has decreased even further.

The optimized scenario shows a vehicle coming from the left and ignoring the right of way of the ego vehicle while breaking with $a = -2\,\mathrm{m\,s^{-1}}$. The ego vehicle either needs to perform an emergency braking maneuver or evade to the right. Even though another traffic participant would be blamed for the potential collision in this case, one is still interested in protecting the passengers of the ego vehicle through a safe maneuver of the motion planner. When comparing the convergence of solvers in Fig. 4, DE and PSO perform almost equally in the beginning; however, after 12 iterations, DE almost does not improve anymore, while the PSO still improves substantially.

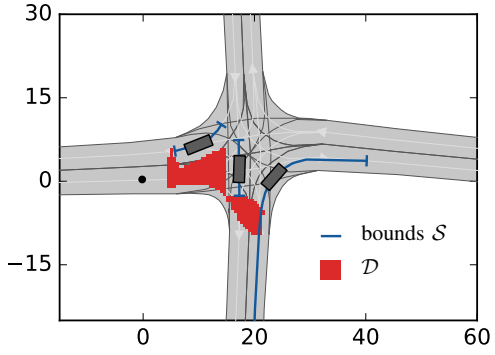
B. Scenario II: Highway scenario

The second scenario is a highway scenario taken from the NGSIM US 101 dataset[2]. This scenario can be found in the CommonRoad benchmark collection under ID USA_US101-14_1_T-1. The ego vehicle has an initial velocity of $14.85\,\mathrm{m\,s^{-1}}$ and is surrounded by similarly paced vehicles in the initial configuration. In order to simplify the scenario,

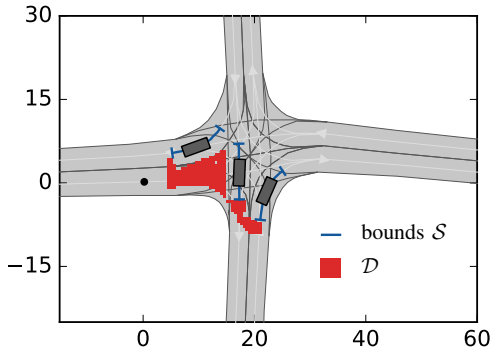
[1] https://commonroad.in.tum.de
[2] http://www.fhwa.dot.gov/publications/research/operations/07030/



(a) Initial configuration at $t = 2.5\,\mathrm{s}$ with initial bounds $\mathcal{S}$.



(b) Configuration after iteration step $l = 2$ at $t = 2.5\,\mathrm{s}$ with adapted bounds $\mathcal{S}$.



(c) Final optimized configuration at $t = 2.5\,\mathrm{s}$ with final bounds $\mathcal{S}$.

Fig. 3: Scenario I: initial configuration, intermediate result during optimization, and the final configuration at $t = 2.5\,\mathrm{s}$. Positional bounds $\mathcal{S}$ of respective iteration k are depicted in blue.

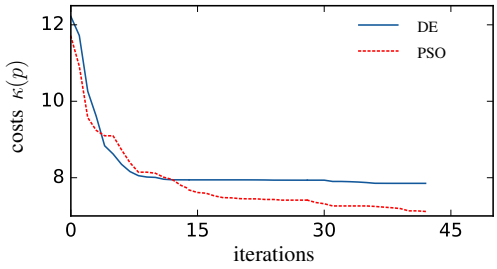


Fig. 4: Comparison of solver convergence for scenario I.

irrelevant vehicles are removed prior to the optimization. As a result, 13 vehicles with a total number of 39 parameters are optimized. This scenario is especially demanding with respect to collision constraints (4) and the high number of parameters in total.

All three algorithms are initialized with a population of 195 individuals and computed for 45 iterations, while parameter bounds are adapted and the repair algorithm is applied every 9 iterations. The computation time is 201.7 minutes. In Fig. 5, the resulting convergence of all solvers

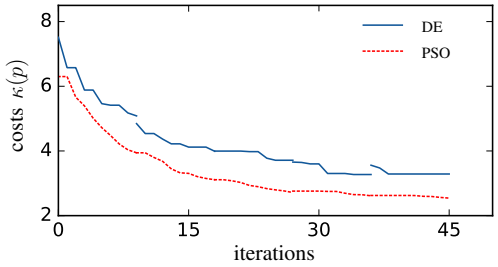


Fig. 5: Comparison of solver convergence for scenario II.

are compared. It shows that PSO has the highest convergence rate and yields the best solution. In comparison, DE exhibits premature convergence. The occasional increase of the cost function can be attributed to the repair algorithm in line 9 of Algorithm 1.

The result of the PSO algorithm is depicted in Fig. 6. While in the beginning the ego vehicle has enough space to maneuver, at $t = 2.6\,\mathrm{s}$ there is little space left due to a lane-changing vehicle and several closely navigating vehicles. For comparison, the initial scenario prior to the optimization, where the vehicle has considerably more space to maneuver, is shown. As this scenario also demonstrates, no vehicles collide, despite the crowded driving situation.

VI. Conclusions

In this work, we present an optimization-based approach to generate critical scenarios for complex traffic situations. Unlike previous works, our approach can handle complex road layouts and dynamics for a high number of involved traffic participants. We ensure that all relevant configurations of a scenario can be reached due to the automatic computation of relevant parameter intervals and evolutionary algorithms. We demonstrate that we can generate critical scenarios for urban scenarios and many involved traffic participants. The obtained scenarios can be used for testing arbitrary motion planners.

Acknowledgements

The authors gratefully acknowledge financial support by the Central Innovation Programme of the German Federal Government under grant ZF4086007BZ8.

References

[1] N. Kalra and S. M. Paddock, "How many miles of driving would it take to demonstrate autonomous vehicle reliability?" RAND Corporation, Santa Monica, CA, Tech. Rep., 2016. [Online]. Available: http://www.rand.org/pubs/research_reports/RR1478.html

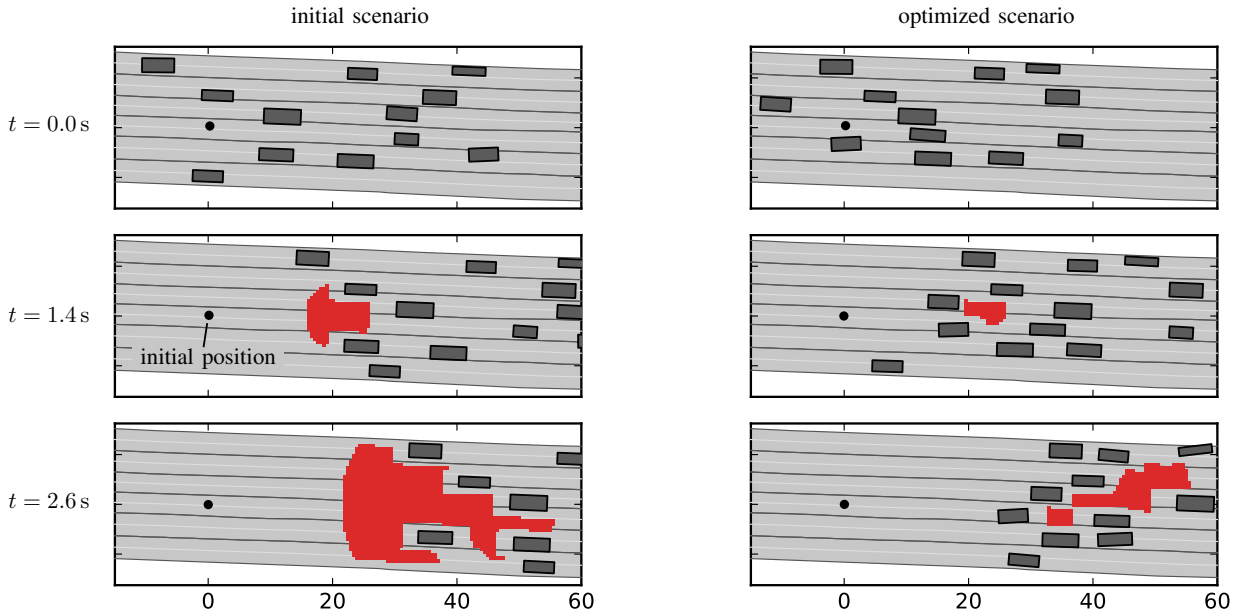


Fig. 6: Scenario II: optimized scenario compared to initial scenario at different times t.

[2] B. Kim, Y. Kashiba, S. Dai, and S. Shiraishi, "Testing autonomous vehicle software in the virtual prototyping environment," *IEEE Embedded Systems Letters*, vol. 9, no. 1, pp. 5–8, 2017.

[3] M. R. Zofka, S. Klemm, F. Kuhnt, T. Schamm, and J. M. Zöllner, "Testing and validating high level components for automated driving: Simulation framework for traffic scenarios," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2016, pp. 144–150.

[4] R. Math, A. Mahr, M. M. Moniri, and C. Müller, "OpenDS: A new open-source driving simulator for research," in *Proc. of Automotive meets Electronics*, 2013.

[5] A. Pütz, A. Zlocki, J. Küfen, J. Bock, and L. Eckstein, "Database approach for the sign-off process of highly automated vehicles," in *25th International Technical Conference on the Enhanced Safety of Vehicles (ESV) National Highway Traffic Safety Administration*, 2017.

[6] J. Vesterstrom and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *Proc. of the Congress on Evolutionary Computation*, vol. 2, 2004, pp. 1980–1987.

[7] G. E. Mullins, P. G. Stankiewicz, and S. K. Gupta, "Automated generation of diverse and challenging scenarios for test and evaluation of autonomous vehicles," in *Proc. of the IEEE International Conference on Robotics and Automation*, 2017, pp. 1443–1450.

[8] C. Wolschke, D. Rombach, P. Liggesmeyer, and T. Kuhn, "Mining test inputs for autonomous vehicles," in *Proc. of Commercial Vehicle Technology*, 2018, pp. 102–113.

[9] V. De Oliveira Neves, M. E. Delamaro, and P. C. Masiero, "An environment to support structural testing of autonomous vehicles," in *European Signal Processing Conference*, 2014, pp. 19–24.

[10] I. R. Jenkins, L. O. Gee, A. Knauss, H. Yin, and J. Schroeder, "Accident scenario generation with recurrent neural networks," in *Proc. of IEEE Conf. on Intelligent Transportation Systems*, 2018, pp. 3340–3345.

[11] F. Schuldt, F. Saust, B. Lichte, M. Maurer, and S. Scholz, "Effiziente systematische Testgenerierung für Fahrerassistenzsysteme in virtuellen Umgebungen," in *Automatisierungssysteme, Assistenzsysteme und eingebettete Systeme für Transportmittel*, 2013, pp. 114 – 134.

[12] G. Bagschik, T. Menzel, C. Körner, and M. Maurer, "Wissensbasierte Szenariengenerierung für Betriebsszenarien auf deutschen Autobahnen," in *Workshop Fahrerassistenzsysteme und automatisiertes Fahren. Bd*, vol. 12, 2018.

[13] F. Hauer and B. Holzmüller, "Szenario-Optimierung für die Absicherung von automatisierten und autonomen Fahrsystemen," in *FKFS AutoTest Fachkonferenz*, 2018.

[14] H. Beglerovic, M. Stolz, and M. Horn, "Testing of autonomous vehicles using surrogate models and stochastic optimization," in *Proc. of the IEEE Conf. on Intelligent Transportation Systems*, 2018, pp. 1129–1134.

[15] O. Buehler and J. Wegener, "Evolutionary functional testing of an automated parking system," in *Proc. of the Int. Conf. on Computer, Communication and Control Technologies*, 2003, pp. 26–31.

[16] T. Hempen, S. Biank, W. Huber, and C. Diedrich, "Model based generation of driving scenarios," in *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, vol. 222, 2018, pp. 153–163.

[17] C. E. Tuncali, T. P. Pavlic, and G. Fainekos, "Utilizing S-TaLiRo as an automatic test generation framework for autonomous vehicles," in *Proc. of the IEEE 19th International Conference on Intelligent Transportation Systems*, 2016, pp. 1470–1475.

[18] H. Abbas, M. E. O'Kelly, A. Rodionova, and R. Mangharam, "A driver's license test for driverless vehicles," *Mechanical Engineering*, vol. 139, no. 12, pp. 13–16, 2017.

[19] C. E. Tuncali, G. Fainekos, H. Ito, and J. Kapinski, "Simulation-based adversarial test generation for autonomous vehicles with machine learning components," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2018, pp. 1555–1562.

[20] M. Althoff and S. Lutz, "Automatic generation of safety-critical test scenarios for collision avoidance of road vehicles," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2018, pp. 1326–1333.

[21] A. Rizaldi and M. Althoff, "Formalising traffic rules for accountability of autonomous vehicles," in *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, 2015, pp. 1658–1665.

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

[23] S. Söntges and M. Althoff, "Computing the drivable area of autonomous road vehicles in dynamic road scenes," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 6, pp. 1855–1866, 2018.

[24] J. Dattorro, *Convex optimization & Euclidean distance geometry*. USA: Meboo Publishing, 2011.

[25] R. Storn and K. Price, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[26] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, 1995, pp. 39–43.

[27] 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.