Concretization of Abstract Traffic Scene Specifications Using Metaheuristic Search

Aren A. Babikian, *Graduate Student Member, IEEE*, Oszkár Semeráth, and Dániel Varró, *Senior Member, IEEE*

***Abstract*—Existing safety assurance approaches for autonomous vehicles (AVs) perform system-level safety evaluation by placing the AV-under-test in challenging traffic scenarios captured by abstract scenario specifications and investigated in realistic traffic simulators. As a first step towards scenario-based testing of AVs, the initial scene of a traffic scenario must be concretized. In this context, the scene concretization challenge takes as input a high-level specification of abstract traffic scenes and aims to map them to concrete scenes where exact numeric initial values are defined for each attribute of a vehicle (e.g. position or velocity). In this paper, we propose a traffic scene concretization approach that places vehicles on realistic road maps such that they satisfy an extensible set of abstract constraints defined by an expressive scene specification language which also supports static detection of inconsistencies. Then, abstract constraints are mapped to corresponding numeric constraints, which are solved by metaheuristic search with customizable objective functions and constraint aggregation strategies. We conduct a series of experiments over three realistic road maps to compare eight configurations of our approach with three variations of the state-of-the-art SCENIC tool, and to evaluate its scalability.**

***Index Terms*—Assurance for autonomous vehicles, scenario description language, traffic scene concretization, metaheuristic search.**

Manuscript received 16 August 2022; revised 17 October 2023; accepted 23 October 2023. Date of publication 13 November 2023; date of current version 9 January 2024. This work was supported in part by the NSERC under Grants RGPIN-2022-04357 and PGSD3-546810-2020, in part by the NRDI Fund based on the charter of bolster issued by the NRDI Office under the auspices of the Ministry for Innovation and Technology and by the ÚNKP-21-4 New National Excellence Program of the Ministry for Innovation and Technology from the source of the National Research, Development and Innovation Fund, in part by the Wallenberg AI, Autonomous Systems and Software Program (WASP), Sweden, and in part by an Amazon Research Award. During the development of the achievements, we took into consideration the goals set by the Balatonfüred System Science Innovation Cluster and the plans of the "BME Balatonfüred Knowledge Center" under Grant EFOP 4.2.1-16-2017-00021. Recommended for acceptance by D. Bianculli. *(Corresponding author: Oszkár Semeráth.)*

Aren A. Babikian is with the Department of Electrical and Computer Engineering, McGill University, Montreal H3A 0G4, Canada (e-mail: aren.babikian@mail.mcgill.ca).

Oszkár Semeráth is with the Department of Measurement and Information Systems, Budapest University of Technology and Economics, 1111 Budapest, Hungary (e-mail: semerath@mit.bme.hu).

Dániel Varró is with the Department of Electrical and Computer Engineering, McGill University, Montreal H3A 0G4, Canada, also with the Department of Measurement and Information Systems, Budapest University of Technology and Economics, 1111 Budapest, Hungary, and also with the Department of Computer and Information Science, Linköping University, 58183 Linköping, Sweden (e-mail: daniel.varro@liu.se).

This article has supplementary downloadable material available at https://doi.org/10.1109/TSE.2023.3331254, provided by the authors.

Digital Object Identifier 10.1109/TSE.2023.3331254

I. Introduction

The increasing popularity of autonomous vehicles (AVs) has resulted in a rising interest in their safety assurance. As such, rigorous certification criteria must be met to ensure the safe, widespread use of AVs from a societal perspective. A high-level certification objective may be formulated for AVs as follows: if an AV-under-test is intending to execute *any valid target maneuver*, at *any valid location* on Earth and alongside *any valid placement* of external actors performing *valid maneuvers*, the AV-under-test executes the maneuver *safely* (e.g. without getting into an accident).

In this context, existing safety assurance approaches [1], [2] test AVs by placing them in challenging traffic scenarios to evaluate their system-level safety. Graph models (e.g. scene graphs) are frequently used to define such test scenarios along qualitative abstractions (relations) of concrete values and positions of scenario actors. Such a formal representations particularly allows for the analysis of various properties at the level of test scenario suites through high-level metrics such as situation coverage [3], [4].

On the one hand, safety experts and standards typically express scenarios at a high-level of abstraction using abstract relations between various actors to evaluate situation coverage of a test suite. On the other hand, modern traffic simulators (like CARLA [5] or DriveSim [6]) necessitate concrete scenarios with exact numeric values provided for the various actors in order to evaluate the safety compliance of each test scenario. To derive such concrete test scenarios, first, an initial concrete scene needs to be derived from the abstract scenario representation. The initial scene is then augmented with concrete behavior before being run in simulation. As a key challenge, *automated concretization of an initial scene* takes an abstract scene specification with numerous high-level constraints as input, and automatically derives concrete scenes by providing concrete parameter values for each actor. Since the relevance of certain test scenarios may depend on the physical location (e.g. in case of geofencing for AVs), *scene concretization parameterized within a designated geographical location* is particularly challenging.

Graph model generation has been used extensively in research to derive models that satisfy high-level (abstract) constraints. Existing approaches may rely on logic solvers [7], metaheuristic search [8] or a dedicated graph solver [9]. However, such approaches derive abstract graph models as output without any numeric information, which is insufficient for scene concretization. To derive numeric solutions, modern model generators [2], [10] propose a hybrid search technique that integrates a back-end numeric reasoning tool. However, their use in traffic scene concretization is limited to generating instances of a specific type of traffic scene (selected a priori) over a simple pre-defined map.

Specialized traffic scene concretization approaches such as the state-of-the-art SCENIC tool [11] have been developed with a custom scene specification language to *capture arbitrary abstract constraints* over a *custom road map* (both given as input). However, the limited expressiveness of these languages (e.g. related to the allowed constraint structures) prevents adequate measurement of situation coverage as necessitated in safety standards for road vehicles [12], [13]. Furthermore, as shown in our paper, the underlying exploration strategies are not scalable enough to provide effective assurance for AVs aligned with these safety standards.

In this paper, we propose a scene concretization approach that automatically derives concrete scenes in accordance with an abstract scene specification as input. The specific contributions of the paper are as follows:

- (C1) **Scene specification language:** We propose an expressive (abstract) functional scene specification language with 4-valued partial model semantics that generalizes the SCENIC language [11] and enables static detection of inconsistencies at specification time.
- (C2) **Mapping for abstract constraints:** We define an extensible mapping from abstract (relational) constraints to corresponding numeric constraints to derive a numeric scene concretization problem.
- (C3) **Integration of metaheuristic search:** We formalize the scene concretization problem as a customizable optimization problem which we solve using metaheuristic search algorithms.
- (C4) **Extensive evaluation:** We evaluate eight configurations of our proposed approach over three realistic road maps to assess success rate, runtime and scalability compared to the SCENIC tool.

The rest of the paper is structured as follows: Section II summarizes core concepts related to traffic scenes and scenarios used in safety assurance of AVs. Section III introduces the novel, expressive abstract scene specification language. Section IV presents the mapping from abstract to numeric constraints. Section V details the adaptation of metaheuristic search for traffic scene concretization. Section VI provides evaluation results of our proposed approach for three case studies. Section VII discusses the practical implications of our approach. Section VIII overviews related approaches available in the literature. Finally, Section IX concludes the paper.

II. PRELIMINARIES

A. Traffic Scenes and Scenarios

Traffic scene is defined by Ulbrich et al. [14] as a snapshot of the environment, including the *scenery* and *dynamic elements*, as well as the *relations* between those entities.

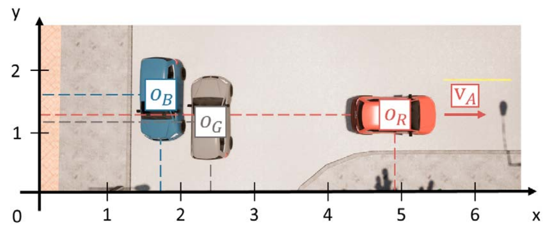


Fig. 1. A traffic scene involving three vehicles.

- The *scenery* is comprised of the lane network, stationary components such as traffic lights and curbs, vertical elevation of roads and environmental conditions.
- *Dynamic elements* (or *actors*) include the various vehicles and pedestrians involved in a scene such as the ego vehicle. A scene may contain information about the state (e.g. position and speed) and attributes (e.g. vehicle color, whether a car door is open) of actors.
- *Relations* are defined between scenery elements and actors. For example, two vehicles may be *far from each other*, or a vehicle may be placed *on* a specific lane. See Section III for further details.

A sequence of consecutive traffic scenes together with related temporal developments corresponds to a *scenario*. A scenario is defined by an *initial scene*, followed by a sequence of *actions and events* performed by the actors according to individual *goals and values*. *Actions and events* may refer to traffic maneuvers (e.g. a lane change maneuver), while *goals and values* may be transient (e.g. reaching a certain area on a map) or permanent (e.g. driving in a safe manner).

Existing safety standards (e.g. ISO 26262-1 [12] and SOTIF [13]) place system-level safety requirements and restrictions on autonomous vehicles (AVs) under test. Such requirements are often formalized as high-level constraints between actors. Adherence to such safety requirements is often evaluated by using sophisticated traffic simulators like CARLA [5] or DRIVE Sim [6] which can only handle a lower-level representation of the investigated scenarios.

Example 1: Fig. 1 depicts a scene with three *actors* (i.e. vehicles) $\circ_B$, $\circ_G$ and $\circ_R$ at an intersection, which composes the *scenery*. Their respective concrete positions are $\langle 1.7, 1.6\rangle$, $\langle 2.4, 1.2\rangle$ and $\langle 4.9, 1.3\rangle$ according to the indicated coordinate system. $\circ_B$ and $\circ_G$ are positioned behind $\circ_R$, and to the left of each other. They have *opposite* headings and they are placed on *adjacent*, but *opposite* lanes. $\circ_R$ faces in the direction of the x axis and has a forward speed v_A of 0.5 units, depicted by a red arrow, thus it is moving *away from* $\circ_B$ and $\circ_G$, both of which are static.

B. Levels of Abstraction in Traffic Scenarios

Menzel et al. [15] define three abstraction levels to adequately describe traffic scenarios for simulating AVs [16].

1) *Functional Scenarios* include abstract (qualitative) constraints pertaining to traffic concepts. For example, such abstract constraints may be used to describe geospatial concepts (e.g. two vehicles are *close to* or *far from* each other), causal concepts (e.g. vehicle A stopped moving

Functional (initial) scene specification

$\circ_R$: *Car*. $\circ_B$: *Car*.

ahead$(\circ_R, \circ_B)$. *medDist*$(\circ_B, \circ_R)$. ! *right*$(\circ_B, \circ_R)$.

Logical (initial) scene specification

$$\circ_R \mapsto \vec{a}_R = \langle \vec{a}_R.\mathrm{x}, \vec{a}_R.\mathrm{y}, \vec{a}_R.\mathrm{h} \rangle$$

$$\circ_B \mapsto \vec{a}_B = \langle \vec{a}_B.\mathrm{x}, \vec{a}_B.\mathrm{y}, \vec{a}_B.\mathrm{h} \rangle$$

$$\exists r > 0 : \substack{\vec{a}_B.\mathrm{x} = \vec{a}_R.\mathrm{x} + r\cos \vec{a}_R.\mathrm{h} \wedge \\ \vec{a}_B.\mathrm{y} = \vec{a}_R.\mathrm{y} + r\sin \vec{a}_R.\mathrm{h}}$$

$$c_l \leqslant \sqrt{(\vec{a}_R.\mathrm{x} + \vec{a}_B.\mathrm{x})^2 + (\vec{a}_R.\mathrm{y} + \vec{a}_B.\mathrm{y})^2} \leqslant c_u$$

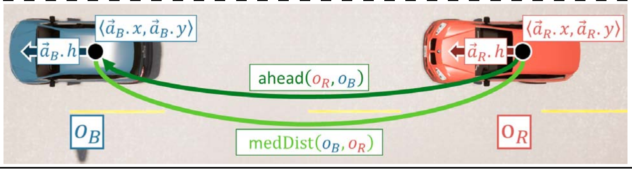
$$\neg\left(\exists r > 0 : \substack{\vec{a}_R.\mathrm{x} = \vec{a}_B.\mathrm{x} + r\cos(\vec{a}_B.\mathrm{h} - \pi/2) \wedge \\ \vec{a}_R.\mathrm{y} = \vec{a}_B.\mathrm{y} + r\sin(\vec{a}_B.\mathrm{h} - \pi/2)}\right)$$

Concrete (initial) scene

$$\vec{a}_R = \langle \vec{a}_R.\mathrm{x} = 56.20, \vec{a}_R.\mathrm{y} = 188.48, \vec{a}_R.\mathrm{h} = 1.57 \rangle$$

$$\vec{a}_B = \langle \vec{a}_B.\mathrm{x} = 42.44, \vec{a}_B.\mathrm{y} = 188.48, \vec{a}_B.\mathrm{h} = 1.57 \rangle$$

Visualisation



$\langle \vec{a}_B.x, \vec{a}_B.y \rangle$
$\vec{a}_B.h$
$\langle \vec{a}_R.x, \vec{a}_R.y \rangle$
$\vec{a}_R.h$
ahead(o_R, o_B)
o_B
medDist(o_B, o_R)
o_R

Fig. 2. An initial traffic scene described at various levels of abstraction.

because it encountered a red light) and temporal concepts (e.g. event A occurred *before* or *after* event B).

2) *Logical Scenarios* refine the abstract constraints of functional scenarios into constraints over parameter ranges or intervals, optionally accompanied by probability distributions. For example, geospatial functional constraints may be refined to areas on a map, and temporal functional constraints may be refined to time intervals.

3) *Concrete Scenarios* substitute concrete numeric values from the parameter ranges/intervals defined in a logical scenario. For example, concrete scenarios contain exact values for the position coordinates of actors, as well as exact times and durations for event executions.

Given a specific concrete scenario executable in a traffic simulator, any abstract relation in functional scenarios can be derived by (i) identifying relevant logical constraints (e.g. geospatial, temporal), and (ii) assigning the truth value of abstract relations accordingly by predicate abstraction.

Example 2: An initial traffic scene containing two actors $\circ_R$ and $\circ_B$ is described at various levels of abstraction in Fig. 2 (c_l and c_u are constants). Functional and logical scenes define constraints (i.e. the *problem*), while concrete scene defines an instance (i.e. a *solution*) that satisfies these constraints. The problem is exclusively comprised of geospatial constraints, since no dynamic behavior is involved in the initial scene. The functional scene is defined using the abstract language proposed in Section III. The logical and concrete scenes are handled in Section IV, while Section V describes our solution to such problems.

C. Scenario-Based Testing by Simulation

Scenario-based testing by simulation [11], [17], [18], [19] is commonly used to evaluate the adherence of AVs under test to traffic safety requirements. In line with the definition of a traffic scenario proposed in Section II-A [14], scenario-based test cases are composed of an abstract *Initial Scene Specification* (ISS), abstract *Behavioral/Temporal Constraints* (BTCons) over actors along with *Evaluation Criteria* (ECrit), often defined as oracles [20] based on safety requirements.

For a test case to be executable in simulation, its abstract constraints must be concretized. As a first step, (1) the abstract ISS is refined into a concrete scene. Then, (2) concrete behaviors, such as exact trajectories to follow, are assigned to each actor in accordance to BTCons. Finally, (3) the concrete scenario is simulated, and its success (pass/fail) is evaluated according to ECrit. For instance, a test may be considered successful if the ego vehicle can navigate its assigned trajectory without colliding with any other actor.

In this paper, we exclusively focus on *Step (1)*, i.e., the automated concretization of abstract ISSs into realistic *initial scenes*. This is an important aspect of AV testing as the initial scene may *have a direct impact on the outcome of test execution while assuming identical behavior for all actors*. For instance, consider two test scenarios with different initial scenes (ISSs) but identical actor behaviors. In both cases, the ego actor $\circ_{ego}$ and a non-ego actor $\circ_B$ are driving at a high speed inside their lane ($\circ_{ego}$ is following $\circ_B$) while another actor $\circ_R$ abruptly cuts in front $\circ_B$, forcing $\circ_B$ to do an abrupt brake. If, according to the ISS, $\circ_{ego}$ is close behind $\circ_B$, these actor will collide. However, if the ISS specifies that $\circ_B$ and $\circ_{ego}$ are initially far from each other, collision will be avoided since $\circ_B$ will have time to slow down. Videos depicting the two test scenarios are included in an online publication page[1] dedicated to this paper.

D. Scene Concretization in Scenario-Based Testing

Our paper proposes a *scene concretization* approach where a functional-level (initial) scene specification is given as input and the concrete positioning of vehicles is constructed as output. For that purpose, abstract constraints specifying the functional scene are first mapped into an equivalent numeric problem (i.e. the logical scene) along the mapping of Section IV-B. Then, metaheuristic search (MHS) is used to derive a numeric solution (i.e. a concrete scene) that satisfies all related constraints.

Search-based test generation techniques [17], [19], [21], [22], [23] have been actively used to provide potentially dangerous concrete test scenarios as input for traffic simulators. As general assumption of these approaches, a single search process is conducted to find concrete scene parameters and actor behavior that leads to potential danger. Our paper investigates *scene concretization as a standalone subproblem* of the complex challenge of scenario-based testing, which complements existing work in three key aspects.

- Our abstract scene representation enables to *evaluate the coverage of arbitrary automatically generated test scenarios* with formal precision by qualitative abstraction for similar behavior of actors.
- When a potentially dangerous scenario is found by existing test generators, our approach can provide *what-if* analysis

[1]https://doi.org/10.5281/zenodo.6345282

by deriving a diverse set of initial scenes to investigate similar behavior of actors. Such analysis can help better understand what contextual parameters of the scene itself can contribute to potential danger.

- Our approach enables to investigate *traffic scenarios in a realistic context by concretizing scenes in concrete map locations*. Demonstrating safe behavior of AV at a specific location can help *geofencing*, e.g. an AV is allowed to take a particular route on the map but not other routes.

III. Functional Scene Specification

Functional scene specifications (FSSs) are often captured by an abstract constraint language [11] that leverages qualitative abstractions [24], [25] of concrete scene attributes. In this paper, we adapt (4-valued) partial graph models as a FSS language using the syntax and formal semantics defined in [26]. As a key benefit over state-of-the-art traffic scene concretization approaches, e.g. Scenic [11], partial models enable the detection of inconsistencies at the FSS level.

A. Scene Specification Language

Vocabulary: Objects in a partial model correspond to actors of a scene. The relations between actors are captured by a finite set of *relation symbols* $\Sigma = \{\mathtt{R}_{pos} \cup \mathtt{R}_{dist} \cup \mathtt{R}_{vis} \cup \mathtt{R}_{coll} \cup \mathtt{R}_{road}\}$ grouped into 5 *geospatial relation categories*:

- $\mathtt{R}_{pos} = \{\textit{left}, \textit{right}, \textit{ahead}, \textit{behind}\}$ are **positional relations** denoting the relative position of the target actor with respect to the heading of the source actor.
- $\mathtt{R}_{dist} = \{\textit{close}, \textit{medDist}, \textit{far}\}$ is a set of **distance relations** which qualitatively characterize the Euclidean distance between two actors (using x and y coordinates).
- $\mathtt{R}_{vis} = \{\textit{canSee}\}$ is the **visibility relation** to capture if the target actor is in the field of view of the source actor.
- $\mathtt{R}_{coll} = \{\textit{noColl}\}$ is the **collision avoidance relation** which denotes that two actors are positioned such that they are not overlapping (colliding).
- $\mathtt{R}_{road} = \{\textit{onAnyRd}\}$ represents the unary **road placement relation** which denotes that an actor is placed on any driveable road segment of the map (i.e. any segment which can be used by vehicles).

The abstract relations listed above are adapted from the Scenic specification languages [11], and similar abstract relations have been proposed in [24], [25], [27]. For simplicity, abstract relations are restricted to binary relations as they are the most common in traffic scene specifications. Thus, the unary *onAnyRd* relation is represented as a (binary) self-loop relation. However, the proposed formalism can be generalized to *n-ary* relations and constraints, such as a ternary constraint specifying that the line of sight between actors $\mathtt{o}_A$ and $\mathtt{o}_B$ is obstructed by an actor $\mathtt{o}_C$.

Our approach assumes that these relations can be derived from concrete scenes by qualitative abstractions. Note that our approach is independent from the included concrete relations, therefore we can extend the set of relations by various sorts of abstractions from the concrete scenes. Moreover, we can also adjust relation categories accordingly.

Syntax and semantics: Given a vocabulary of geometric relations Σ, a FSS is a partial model $P = \langle \mathcal{O}_P, \mathcal{I}_P \rangle$, where $\mathcal{O}_P$ is the finite set of objects (each object corresponds to an actor), and $\mathcal{I}_P$ gives a 4-valued logic interpretation for each symbol $\mathtt{r} \in \Sigma$ as $\mathcal{I}_P(\mathtt{r})\colon \mathcal{O}_P \times \mathcal{O}_P \to \{\mathtt{false}, \mathtt{true}, \mathtt{unknown}$ (unspecified), $\mathtt{error}$ (inconsistent)$\}$.

A FSS consists of *relation assertions*, which explicitly assign a 4-valued truth-value to a binary relation over a pair of objects [28], [29]. Syntactically, a relation assertion can be prefixed by the ? (`unknown`) and ! (`false`) symbols, while no prefix represents `true`.

When multiple assertions to the same relation instance exist, the interpretation value is obtained by the 4-valued *information merge* operator $\oplus$ [26], where contradictory information results in `error` while unspecified relations result in `unknown`. For instance, if a FSS contains both *right*($\mathtt{o}_A$, $\mathtt{o}_B$) and !*right*($\mathtt{o}_A$, $\mathtt{o}_B$), then the relation will be interpreted as $\mathcal{I}_P(\textit{right})(\mathtt{o}_A, \mathtt{o}_B) = \mathtt{true} \oplus \mathtt{false} = \mathtt{error}$.

Such `error` values detect inconsistencies in the FSS, i.e. they detect sets of constraints that cannot be satisfied by any concrete scene. For instance, *right*($\mathtt{o}_A$, $\mathtt{o}_B$) and !*right*($\mathtt{o}_A$, $\mathtt{o}_B$) cannot hold at the same time for any pair of actors $\langle \mathtt{o}_A, \mathtt{o}_B \rangle$. `error` values also arise from more complex inconsistencies when enforcing domain-specific validity rules.

Validity rules: For a partial model $P = \langle \mathcal{O}_P, \mathcal{I}_P \rangle$ to represent a valid FSS, P must be refined according to five validity rules ($V_{road}, V_{loop}, V_{sym}, V_{pos}, V_{dist}$). V_{road} states that all *onAnyRd* relations between two distinct actors are known to be `false`, since only self-looping *onAnyRd* relations are valid. V_{loop} states that any self-loop relation (other than *onAnyRd*) is known to be `false`. V_{sym} states that if a **distance** or **collision avoidance** relation *r* holds, then the same relation in the opposite direction is known to be `true`. V_{pos} states that if a **positional relation** *r1* holds between a given directed pair of actors, then all other **positional relations** *r2* between those actors are known to be `false`. V_{dist} is analogous to V_{pos}, but is applied to **distance relations**. Formally:

$$V_{road} \quad \frac{a, b \in \mathcal{O}_P \wedge a \neq b}{\mathcal{I}_P(\textit{onAnyRd})(a, b) := \mathcal{I}_P(\textit{onAnyRd})(a, b) \oplus \mathtt{false}}$$

$$V_{loop} \quad \frac{o \in \mathcal{O}_P}{\mathcal{I}_P(\textit{r})(o, o) := \mathcal{I}_P(\textit{r})(o, o) \oplus \mathtt{false}}; \text{ for } \textit{r} \in \Sigma \backslash \{\textit{onAnyRd}\}$$

$$V_{sym} \quad \frac{a, b \in \mathcal{O}_P \wedge \mathcal{I}_P(\textit{r})(a, b) = \mathtt{true}}{\mathcal{I}_P(\textit{r})(b, a) := \mathcal{I}_P(\textit{r})(b, a) \oplus \mathtt{true}}; \text{ for } \textit{r} \in \mathtt{R}_{dist} \cup \{\textit{noColl}\}$$

$$V_{pos} \quad \frac{a, b \in \mathcal{O}_P \wedge \mathcal{I}_P(\textit{r1})(a, b) = \mathtt{true}}{\mathcal{I}_P(\textit{r2})(a, b) := \mathcal{I}_P(\textit{r2})(a, b) \oplus \mathtt{false}}; \text{ for } \textit{r1} \in \mathtt{R}_{pos}, \textit{r2} \in \mathtt{R}_{pos} \backslash \textit{r1}$$

$$V_{dist} \quad \frac{a, b \in \mathcal{O}_P \wedge \mathcal{I}_P(\textit{r1})(a, b) = \mathtt{true}}{\mathcal{I}_P(\textit{r2})(a, b) := \mathcal{I}_P(\textit{r2})(a, b) \oplus \mathtt{false}}; \text{ for } \textit{r1} \in \mathtt{R}_{dist}, \textit{r2} \in \mathtt{R}_{dist} \backslash \textit{r1}$$

In this paper, we provide a sound but incomplete set of validity rules. If the enforcement of these rules produces an error, then the scene specification is surely inconsistent. However, if no error is produced, the scene specification is not ensured to be consistent.

The above validity rules are provided in accordance with the included abstract relations and they can easily be extended with new relations in the future. Additionally, custom validity rules may be defined to prevent semantic inconsistencies (i.e. physically infeasible specifications) or to enforce additional requirements (e.g. traffic laws). For instance, a custom validity rule V_{cust} can capture that if a vehicle b is behind a vehicle a, then a cannot see b.

$$V_{cust}\ \frac{a, b \in \mathcal{O}_P \wedge \mathcal{I}_P(\mathit{behind})(a, b)}{\mathcal{I}_P(\mathit{canSee})(a, b) := \mathcal{I}_P(\mathit{canSee})(a, b) \oplus \mathtt{false}}$$

Example 3: A functional-level scene specification is defined as a partial model $P = \langle \mathcal{O}_P, \mathcal{I}_P \rangle$ by the relation assertions in Fig. 3(a). The scene (P) contains three car objects ($\circ_G, \circ_B, \circ_R \in \mathcal{O}_P$), which are placed on a road and do not collide with each other.

All unspecified relations are originally interpreted as `unknown`. Additionally, positional and distance relations are included for every pair of actors (every oriented pair, in the case of positional relations). As such, according to the validity rules, all unspecified positional and distance relations are refined to `true` or to `false`.

For instance, the inclusion of *close*($\circ_G$, $\circ_B$) implies:

- $\mathcal{I}_P(\mathit{close})(\circ_B, \circ_G) = \mathtt{true}$; *(from V_{sym})*,
- $\mathcal{I}_P(\mathit{medDist})(\circ_G, \circ_B) = \mathtt{false}$, $\mathcal{I}_P(\mathit{far})(\circ_G, \circ_B) = \mathtt{false}$; *(from V_{dist})*, and
- $\mathcal{I}_P(\mathit{medDist})(\circ_B, \circ_G) = \mathtt{false}$, $\mathcal{I}_P(\mathit{far})(\circ_B, \circ_G) = \mathtt{false}$; *(from both V_{sym} and V_{dist})*.

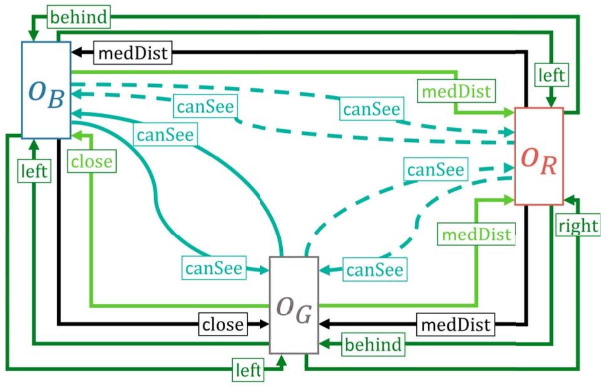
Validity rules enable the detection of inconsistencies in the FSS. For instance, had the FSS also included *far*($\circ_G$, $\circ_B$), the application of V_{dist} would result in an inconsistency:

$$\frac{\circ_G, \circ_B \in \mathcal{O}_P \wedge \mathcal{I}_P(\mathit{close})(\circ_G, \circ_B) = \mathtt{true}}{\begin{aligned}\mathcal{I}_P(\mathit{far})(\circ_G, \circ_B) &:= \mathcal{I}_P(\mathit{far})(\circ_G, \circ_B) \oplus \mathtt{false}\\ &:= \mathtt{true} \oplus \mathtt{false} = \mathtt{error}\end{aligned}}$$

Example 4: The visibility, positional and distance relations of the partial model are represented as a graph in Fig. 3(c), where nodes represent actors. Road placement and collision avoidance relations are not represented for simplicity. Solid and dashed lines represent relations that are interpreted as `true` and as `unknown`, respectively. `false` relations are not depicted, and there are no `error` relations. Relations and nodes are colored according to Fig. 3(a). Relations derived from validity rules are shown in black.

# road placement	# road placement
onAnyRd($\circ_G$, $\circ_G$)	*onAnyRd*($\circ_G$, $\circ_G$)
onAnyRd($\circ_B$, $\circ_B$)	*onAnyRd*($\circ_B$, $\circ_B$)
onAnyRd($\circ_R$, $\circ_R$)	*onAnyRd*($\circ_R$, $\circ_R$)
# collision avoidance	# collision avoidance
noColl($\circ_G$, $\circ_B$)	*noColl*($\circ_G$, $\circ_B$)
noColl($\circ_G$, $\circ_R$)	*noColl*($\circ_G$, $\circ_R$)
noColl($\circ_B$, $\circ_R$)	*noColl*($\circ_B$, $\circ_R$)
# visibility	# visibility
canSee($\circ_G$, $\circ_B$)	*canSee*($\circ_G$, $\circ_B$)
canSee($\circ_B$, $\circ_G$)	*canSee*($\circ_B$, $\circ_G$)
# position	# position
left($\circ_G$, $\circ_B$)	*left*($\circ_G$, $\circ_B$)
right($\circ_G$, $\circ_R$)	~~*right*($\circ_G$, $\circ_R$)~~
left($\circ_B$, $\circ_G$)	~~*left*($\circ_B$, $\circ_G$)~~
left($\circ_B$, $\circ_R$)	*left*($\circ_B$, $\circ_R$)
behind($\circ_R$, $\circ_G$)	~~*behind*($\circ_R$, $\circ_G$)~~
behind($\circ_R$, $\circ_B$)	~~*behind*($\circ_R$, $\circ_B$)~~
# distance	# distance
close($\circ_G$, $\circ_B$)	*close*($\circ_G$, $\circ_B$)
medDist($\circ_G$, $\circ_R$)	*medDist*($\circ_G$, $\circ_R$)
medDist($\circ_B$, $\circ_R$)	*medDist*($\circ_B$, $\circ_R$)

(a) Full scene specification (b) SCENIC-expressible subset



(c) Graph representation of the functional scene

Fig. 3. A functional-level scene specification.

B. Static Analysis of Scene Specifications

Inconsistency detection: SCENIC compiles a functional scene specification into logical constraints, most of which define probability distributions over areas of the road map, and attempts to solve it by using rejection sampling [11]. This involves randomly sampling the desired distributions until all constraints are satisfied. A known drawback of this approach is that when a FSS is inconsistent (e.g. if both *right*($\circ_A$, $\circ_B$) and !*right*($\circ_A$, $\circ_B$) are included), such an inconsistency can only be suspected when the solver repeatedly fails to provide a solution.

A main benefit of the 4-valued semantics of partial models introduced in this paper for FSSs is to *detect such semantic inconsistencies statically* (at specification time). When contradictory assignments are given to a particular relation, they are merged automatically into the `error` value, which can be detected easily. Note that such a static detection of inconsistent specifications is a unique feature of our technique compared to related FSS approaches (e.g. SCENIC).

A sound qualitative abstraction from logical constraints to abstract relations (to be discussed in Section IV) ensures that whenever there is a (concrete) solution to the logical constraints, then the respective abstract relations will also evaluate to `true`. Consequently, our 4-valued static analysis technique also guarantees that if an inconsistency is detected in the FSS (by the `error` value), then no concrete solutions may exist for the logical constraints. In such a case, the underlying solver does not need to be called at all, which can result in significant time savings. Unsurprisingly, our static analysis technique does not have completeness guarantees, i.e. there may be a set of inconsistent logical constraints, which cannot be detected on the abstract level.

Restrictions of the SCENIC *FSS language:* While our FSS language builds on the scene specification language of SCENIC, it is important to note that the original SCENIC language has limitations wrt. (i) error detection capabilities, as discussed above, as well as (ii) soundness.

Due to limitations of the underlying scene concretization approach, the SCENIC language rejects certain valid (consistent)

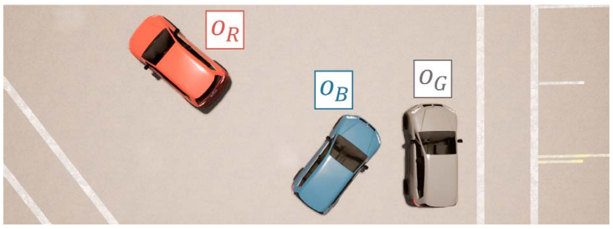


Fig. 4. Concrete scene that does not satisfy Fig. 3(a), but satisfies the corresponding SCENIC-expressible subset.

constraint structures, which poses limitations to the soundness of the approach. For example, actors may be the target of at most one positional relation, pointing from an already instantiated actor. As such, positional relations may only form a tree structure (and not an arbitrary graph), which, in certain cases, is not enough to formally distinguish two semantically different scenes.

Example 5: Fig. 3(b) presents a SCENIC-expressible subset of the relation assertions of Fig. 3(a) (inexpressible constraints are crossed out). Of the 6 positional relations, at most two can be expressed by SCENIC. Note that the order of actor definitions ($\circ_G$, $\circ_B$, then $\circ_R$) influences which relations can be included. Along this ordering, *right*($\circ_G$, $\circ_R$) could be included instead of *left*($\circ_B$, $\circ_R$).

Fig. 4 depicts a concrete scene that satisfies the SCENIC-expressible subset. When compared to Fig. 1, which satisfies the full FSS proposed in Fig. 3(a), the benefits of a more expressive language become apparent. *None* of the relations excluded from the SCENIC-expressible subset are satisfied by the the scene depicted in Fig. 4. As such, both of these scenes, which are semantically different, cannot be formally distinguished using the default SCENIC FSS language as they would both be represented by the FSS shown in Fig. 3(b).

C1 We propose a high-level traffic scene representation language with 4-valued partial model semantics. This enables static detection of inconsistencies at specification time, which is not offered by related FSS approaches.

IV. Logical Scenes as Numeric Problems

The scene concretization problem can be represented as a numeric constraint satisfaction problem over actors on a logical-level scene. We introduce a formalization for logical-level scenes, and propose a novel mapping from a FSS to a corresponding numeric constraint satisfaction problem.

As key benefit, our mapping is *extensible* to take any abstract functional relation as input and yields *customizable* numeric constraints as output. Additionally, our approach can be contextualized in *any underlying road map*. Existing such mappings often either provide restricted numeric constraints as output, or they are limited to approach-specific input constraints defined over a simplistic road map.

A. Numeric Concretization Problem

Formalization: A logical scene defines a numeric rectangle layouting problem that yields a concrete scene as solution. Formally, such a numeric problem N corresponds to a tuple $N = \langle \mathcal{A}_N, \mathcal{C}_N, m_N, D_N \rangle$, where:

- $\mathcal{A}_N$ is a finite set of actors where each $\vec{\mathtt{a}}_i \in \mathcal{A}_N$ is an oriented rectangle defined as *5-tuples* (see below),
- $\mathcal{C}_N$ is a set of binary numeric (geometric) constraints $c_i(\vec{\mathtt{a}}_0, \vec{\mathtt{a}}_1)$ over actors (oriented rectangles) $\vec{\mathtt{a}}_i \in \mathcal{A}_N$,
- m_N is a road map that restricts the range of position variables for actors in $\mathcal{A}_N$, and
- D_N contains valid bounding box sizes (width, length) of actors as a finite set of floating-point pairs $\langle \mathtt{w}_i, \mathtt{l}_i \rangle$.

Note that we only handle binary numeric constraints in this paper to stay consistent with the functional-level binary relations proposed in Section III-A. Nevertheless, our formalization can be generalized to *n-ary* constraints.

We approximate actors $\vec{\mathtt{a}}_i \in \mathcal{A}_N$ as oriented rectangles over a map m_N given as input. Formally, an actor $\vec{\mathtt{a}}_i$ is represented by a tuple $\vec{\mathtt{a}}_i = \langle \mathtt{x}, \mathtt{y}, \mathtt{h}, \mathtt{w}, \mathtt{l} \rangle$, where:

- $\mathtt{x}$ and $\mathtt{y}$ are (floating point) variables that represent the center point of $\vec{\mathtt{a}}_i$ (where $\mathtt{x} \in [0, m_N.x]$ and $\mathtt{y} \in [0, m_N.y]$). Here, $m_N.x$ and $m_N.y$ respectively represent the width and length of the map m_N,
- $\mathtt{h}$ is a (floating point) variable for the heading angle of $\vec{\mathtt{a}}_i$ (in *radians*, i.e. $\mathtt{h} \in [-\pi, \pi]$),
- $\mathtt{w}, \mathtt{l}$ are (floating point) variables that represents the width and length of $\vec{\mathtt{a}}_i$ (where $\langle \mathtt{w}, \mathtt{l} \rangle \in D_N$).

Deriving a numeric problem: A numeric problem $N = \langle \mathcal{A}_N, \mathcal{C}_N, m_N, D_N \rangle$ is derived from a partial model $P = \langle \mathcal{O}_P, \mathcal{I}_P \rangle$ of a FSS through a mapping $f2l : P \mapsto N$ between functional relations and numeric constraints such as the one proposed in Section IV-B. We derive $f2l$ by

1) providing the map m_N and possible actor bounding box sizes D_N as external inputs (parameters),
2) mapping each abstract object $\mathtt{o}_i \in \mathcal{O}_P$ to a corresponding numeric actor $\vec{\mathtt{a}}_i \in \mathcal{A}_N$,
3) populating $\mathcal{C}_N$ such that
   - for every positive relation $\mathtt{r} \in \Sigma$ over objects $\mathtt{o}_i \in \mathcal{O}_P$ (i.e. where $\mathcal{I}_P(\mathtt{r})(\mathtt{o}_0, \mathtt{o}_1) = \mathtt{true}$), a corresponding numeric constraint $c_i(\vec{\mathtt{a}}_0, \vec{\mathtt{a}}_1)$ (as defined in Section IV-B) is included in $\mathcal{C}_N$.
   - for every negative relation $\mathtt{r} \in \Sigma$ over objects $\mathtt{o}_i \in \mathcal{O}_P$ (i.e. where $\mathcal{I}_P(\mathtt{r})(\mathtt{o}_0, \mathtt{o}_1) = \mathtt{false}$), the *negation* of $c_i(\vec{\mathtt{a}}_0, \vec{\mathtt{a}}_1)$ is included in $\mathcal{C}_N$.

Defining a numeric solution: Given a numeric problem $N = \langle \mathcal{A}_N, \mathcal{C}_N, m_N, D_N \rangle$ derived from a partial model $P = \langle \mathcal{O}_P, \mathcal{I}_P \rangle$ of a FSS, a solution $s_N : \mathcal{A}_N \rightarrow \mathbb{R}^5$ is a value assignment of the variables associated to all actors $\vec{\mathtt{a}}_i \in \mathcal{A}_N$ (within the respective ranges) such that all constraints $c_i \in \mathcal{C}_N$ are satisfied. The numeric values assigned to variables represent a concrete scene which is a concretization of the FSS corresponding to P.

The concrete solution s_N of a numeric problem N can be abstracted into the partial model P_s (of a functional scene) using the same set of logical constraints. The numeric constraint corresponding to each instance of relation $\mathtt{r} \in \Sigma$ is evaluated on s_N. Each constrain evaluation yields a Boolean truth value (true/false). These Boolean values derived from such a mapping

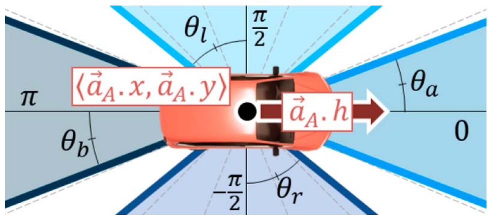
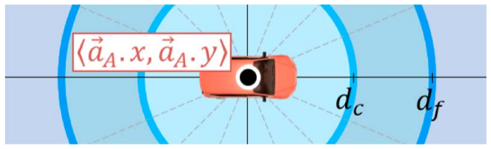
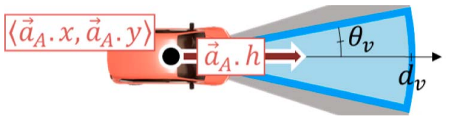
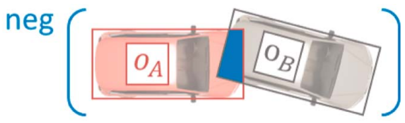
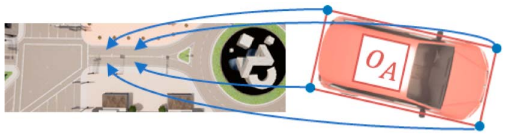
Functional rel.	Logical constraint (with $\circ_A \mapsto \vec{a}_A$, $\circ_B \mapsto \vec{a}_B$)	Visualisation
left($\circ_A$, $\circ_B$)	$\exists r_l \geqslant 0, \vec{a}_A.\mathrm{h} + \pi/2 - \theta_l \leqslant \alpha_l \leqslant \vec{a}_A.\mathrm{h} + \pi/2 + \theta_l :$ $\vec{a}_B.\mathrm{x} = \vec{a}_A.\mathrm{x} + r_l \cos\alpha_l \wedge \vec{a}_B.\mathrm{y} = \vec{a}_A.\mathrm{y} + r_l \sin\alpha_l$	 θ_l $\frac{\pi}{2}$ θ_a π $\langle\vec{a}_A.x, \vec{a}_A.y\rangle$ $\vec{a}_A.h$ 0 θ_b $-\frac{\pi}{2}$ θ_r
right($\circ_A$, $\circ_B$)	$\exists r_r \geqslant 0, \vec{a}_A.\mathrm{h} - \pi/2 - \theta_r \leqslant \alpha_r \leqslant \vec{a}_A.\mathrm{h} - \pi/2 + \theta_r :$ $\vec{a}_B.\mathrm{x} = \vec{a}_A.\mathrm{x} + r_r \cos\alpha_r \wedge \vec{a}_B.\mathrm{y} = \vec{a}_A.\mathrm{y} + r_r \sin\alpha_r$	
ahead($\circ_A$, $\circ_B$)	$\exists r_a \geqslant 0, \vec{a}_A.\mathrm{h} - \theta_a \leqslant \alpha_a \leqslant \vec{a}_A.\mathrm{h} + \theta_a :$ $\vec{a}_B.\mathrm{x} = \vec{a}_A.\mathrm{x} + r_a \cos\alpha_a \wedge \vec{a}_B.\mathrm{y} = \vec{a}_A.\mathrm{y} + r_a \sin\alpha_a$	
behind($\circ_A$, $\circ_B$)	$\exists r_b \geqslant 0, \vec{a}_A.\mathrm{h} + \pi - \theta_b \leqslant \alpha_b \leqslant \vec{a}_A.\mathrm{h} + \pi + \theta_b :$ $\vec{a}_B.\mathrm{x} = \vec{a}_A.\mathrm{x} + r_b \cos\alpha_b \wedge \vec{a}_B.\mathrm{y} = \vec{a}_A.\mathrm{y} + r_b \sin\alpha_b$	
close($\circ_A$, $\circ_B$)	$0 \leqslant \sqrt{(\vec{a}_A.\mathrm{x} + \vec{a}_B.\mathrm{x})^2 + (\vec{a}_A.\mathrm{y} + \vec{a}_B.\mathrm{y})^2} < d_c$	
medDist($\circ_A$, $\circ_B$)	$d_c \leqslant \sqrt{(\vec{a}_A.\mathrm{x} + \vec{a}_B.\mathrm{x})^2 + (\vec{a}_A.\mathrm{y} + \vec{a}_B.\mathrm{y})^2} < d_f$	
far($\circ_A$, $\circ_B$)	$d_f \leqslant \sqrt{(\vec{a}_A.\mathrm{x} + \vec{a}_B.\mathrm{x})^2 + (\vec{a}_A.\mathrm{y} + \vec{a}_B.\mathrm{y})^2}$	
canSee($\circ_A$, $\circ_B$)	$\exists 0 \leqslant r_v \leqslant d_v, \vec{a}_A.\mathrm{h} - \theta_v \leqslant \alpha_v \leqslant \vec{a}_A.\mathrm{h} + \theta_v,$ $\langle \mathrm{cx}, \mathrm{cy} \rangle \in \vec{a}_B.\mathrm{corners} :$ $\mathrm{cx} = \vec{a}_A.\mathrm{x} + r_v \cos\alpha_v \wedge \mathrm{cy} = \vec{a}_A.\mathrm{y} + r_v \sin\alpha_v$	 $\langle\vec{a}_A.x, \vec{a}_A.y\rangle$ $\vec{a}_A.h$ θ_v d_v
noColl($\circ_A$, $\circ_B$)	$\neg$ *intersects*($\circ_A$, $\circ_B$)	
onAnyRd($\circ_A$, $\circ_A$)	$\forall \vec{a}_A.\mathrm{c} \in \vec{a}_A.\mathrm{corners} : \exists r_{map} \in m_i :$ *contains*$(r_{map}, \vec{a}_A.\mathrm{c})$	

Fig. 5. Mapping from functional relations to logical constraints.

$l2f : s_N \mapsto P_s$ define a concrete partial model that does not contain `unknown` values.

Soundness of a numeric solution: In the Appendix (see the supplementary material), we prove the soundness of our approach, as captured in Theorem 1, given that every individual relation implementation soundly captures the geometrical requirements of the corresponding constraint.

Theorem 1: For a numeric problem $N = f2l(P)$ derived from a FSS, the concrete partial model P_s abstracted from a solution s_N of N as $P_s = l2f(s_N)$ satisfies all relations in P (formally, P_s refines P, i.e. $P \sqsubseteq P_s$[26]).

Example 6: The scene concretization problem in Fig. 2 is defined for 2 actors $\circ_R$ and $\circ_B$, and 3 functional relations *ahead*($\circ_R$, $\circ_B$) and *medDist*($\circ_B$, $\circ_R$) (visualised as arrows), and a negative relation !*right*($\circ_B$, $\circ_R$). For better presentation, identical colors relate actors and constraints on different levels, and numeric actor representations on logical and concrete levels exclude width and length variables. A concrete scene (solution) is defined as an assignment of actor variables to satisfy the logical constraints, e.g. $c_l \leqslant \sqrt{(\vec{a}_R.\mathrm{x} + \vec{a}_B.\mathrm{x})^2 + (\vec{a}_R.\mathrm{y} + \vec{a}_B.\mathrm{y})^2} \leqslant c_u$ in case of *medDist*($\circ_B$, $\circ_R$).

B. Mapping Functional Relations to Logical Constraints

In this section, we describe the formal mapping from functional relations in a partial model (i.e. *relation symbols* in the abstract vocabulary Σ) to corresponding logical (numeric) constraints, alongside a visualisation depicted over actors is proposed in Fig. 5. Our mapping builds on but also generalizes previous work by Menzel et al. [25].

In our notation, $\circ_A$ and $\circ_B$ define actors at the functional level (i.e. partial model objects), which are respectively mapped to actors $\vec{a}_A$ and $\vec{a}_B$ at the logical level to find a concrete solution (see Section IV-A). Corresponding relations and numeric attributes are color-coded. Customizable constants, such as θ_l and d_c, are depicted in black.

Positional relations: When actor $\circ_A$ is connected to actor $\circ_B$ via a positional relation $c \in \mathtt{C}_{pos}$, this signifies that the center point $\langle \vec{a}_B.\mathrm{x}, \vec{a}_B.\mathrm{y} \rangle$ of $\vec{a}_B$ is located in a circular sector centered at $\langle \vec{a}_A.\mathrm{x}, \vec{a}_A.\mathrm{y} \rangle$, with infinite radius. The orientation and central angle of the sector is defined according to the specific positional relation. For example, the *left*($\circ_A$, $\circ_B$) relation denotes a circular sector that covers the region located at the left of $\vec{a}_A$ (with respect to its heading $\vec{a}_A.\mathrm{h}$). With that respect, *left*($\circ_A$, $\circ_B$) means that $\vec{a}_B$ is positioned *to the left of* $\vec{a}_A$.

Distance relations: A distance relation between actors $\circ_A$ and $\circ_B$ signifies that the Euclidean distance between the center point of each actor falls within a certain numeric range. For example, *medDist*($\circ_A$, $\circ_B$) requires that the Euclidean distance between $\langle \vec{a}_A.\mathrm{x}, \vec{a}_A.\mathrm{y} \rangle$ and $\langle \vec{a}_B.\mathrm{x}, \vec{a}_B.\mathrm{y} \rangle$ is a value in $[d_c, d_f]$. The thresholds of distance relations can be customized, e.g., the definition of *closeness* can be different for pedestrians and vehicles on a highway.

$$\vec{a}_A.\texttt{corners} \equiv \Big\{\langle \vec{a}_A.\texttt{x} + L\cdot\cos\vec{a}_A.\texttt{h} - W\cdot\sin\vec{a}_A.\texttt{h}, \vec{a}_A.\texttt{y} + L\cdot\sin\vec{a}_A.\texttt{h} + W\cdot\cos\vec{a}_A.\texttt{h}\rangle, \text{ where } L = \pm\frac{\vec{a}_A.\texttt{l}}{2}, W = \pm\frac{\vec{a}_A.\texttt{w}}{2}\Big\}$$

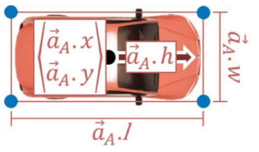


Fig. 6. Formalization for the $\vec{a}_A.\texttt{corners}$ parameter.

Visibility relations: Visibility relations consider that the shape of an actor $\vec{a}_B$ is approximated as a rectangle (bounding box) with width $\vec{a}_B.\texttt{w}$ and length $\vec{a}_B.\texttt{l}$. As such, $\vec{a}_B$ has four corners (i.e. the four corners of the rectangular approximation). Their coordinates are collected in $\vec{a}_B.\texttt{corners}$ according to the definition proposed in Fig. 6. For example, relation *canSee*($\texttt{o}_A$, $\texttt{o}_B$) requires for at least one of the corners of $\vec{a}_B$ to be in a circular sector (i.e. the field of view of $\vec{a}_A$) centered at $\langle\vec{a}_A.\texttt{x}, \vec{a}_A.\texttt{y}\rangle$ in the direction of $\vec{a}_A.\texttt{h}$. Unlike the *ahead*($\texttt{o}_A$, $\texttt{o}_B$) relation, the circular sector defined by visibility relations has a finite radius r_v.

Collision avoidance relations: We reuse the definition of collision avoidance proposed in the SCENIC framework [11]. Informally, the *noColl*($\texttt{o}_A$, $\texttt{o}_B$) relation states that the area within the bounding boxes of $\vec{a}_A$ and of $\vec{a}_B$ do not intersect. Bounding box analysis is outside the scope of this paper, therefore we define collision avoidance through a call to the SCENIC library function *intersects*($\vec{a}_A$, $\vec{a}_B$).

Road placement relations: We also rely on the SCENIC library functions to define road placement relations. Informally, *onAnyRd*($\texttt{o}_A$, $\texttt{o}_A$) states that the four corners of $\vec{a}_A$ (i.e. $\vec{a}_A.\texttt{corners}$) must be located on (contained by) a road that is part of the scene map m_i provided as input. The scene map is segmented into multiple connected roads which are represented as complex polygons (e.g. curved roads, or roads with varying width). Given a road segment r_{map} that is part of the scene map, the *contains*(r_{map}, $\vec{a}_A.\texttt{c}$) function call checks whether the point $\vec{a}_A.\texttt{c}$, which is a corner point of $\vec{a}_A$, is placed within the bounds of the polygon represented by r_{map}. Once again, polygon detection is outside the scope of this paper, therefore we refer to SCENIC library functions.

C. Benefits of the Mapping

While our mapping is based on existing research [25], it conceptually extends this baseline in three different aspects.

First, thanks to the use of partial models as FSSs, our mapping defined in Fig. 5 is *extensible* to arbitrary qualitative abstractions from a concrete scene to an abstract scene on the functional level. As such, we can seamlessly incorporate additional relations proposed by safety experts that can be observed and measured over a concrete model. Furthermore, the implementation of additional relations may be iteratively validated by (1) creating a simple FSS containing only the newly implemented relation, (2) concretizing the FSS, (3) visually detecting implementation issues, if any, and (4) adjusting the implementation accordingly.

Moreover, our approach maps abstract functional relations to numerical constraints over 2-dimensional space with no geometric assumptions. Thus, our mapping is *independent from the underlying road map*, and it can be contextualized to any real physical location on a map. Existing $f2l$ mappings [1], [2] are typically hard-coded to the specific geometry of a particular (often simplistic) road map.

Finally, our mapping is *customizable* to the needs of a given scene. For instance, parameters such as the thresholds for closeness, or the angle for the field of view of an actor can be adjusted according to its types (e.g. pick-up truck or pedestrians). Such flexibility allows the generation of more realistic scenes compared to existing approaches that map functional relations to numeric constraints.

C2 We provide an *extensible* and *customizable* mapping from abstract functional relation to numeric constraints, which can incorporate arbitrary qualitative abstractions, and can be *contextualized in any physical locations of a map*.

V. USING MHS TO DERIVE CONCRETE SCENES

To derive a concrete scene from a FSS, the numeric scene concretization problem defined in Section IV needs to be solved. Since exact algorithms, such as quadratic solvers, have failed to provide scalable results for similar problems [2], in this paper, we adopt metaheuristic search (MHS) algorithms, which are commonly used in the domain of AV testing [1], [17], [21], [22], [23], [30].

Formalization: A numeric scene concretization problem N is mapped to a metaheuristic minimization (MIN) problem that can be solved by a MHS algorithm. A MIN problem is formalized as $M_{min} = \langle\mathcal{V}_M, \mathcal{OF}_M\rangle$, where:

- $\mathcal{V}_M$ represents a set of variables $\{v_{1,1}, \dots, v_{m,5}\}$, where m is the number of actors (represented as *5-tuples*) in N. The domain of each variable $v_i \in \mathcal{V}_M$ is defined by a numeric range $[l_i, b_i]$.
- $\mathcal{OF}_M$ represents a set of objective functions $\{OF_1, \dots, OF_n\}$ defined over variables in $\mathcal{V}_M$ that return *non-negative* numeric values.

At each iteration of the MHS algorithm, a set of *candidate solutions* is derived, which are assignments for each $v_i \in \mathcal{V}_M$ to a value within the corresponding range. A candidate solution is a valid *solution* s_{min} to M_{min} iff all objective functions are minimal (i.e. zero). To resolve potential issues with precision of floating point variables, we require that the value of an objective function should be below a given threshold $\epsilon > 0$.

As is the case for existing research [31], [32], [33], we propose an *unconstrained* MIN problem formalization: all logical-level constraints are directly incorporated into objective functions. As such, no additional handling of constraints is required by the underlying MHS algorithm.

Deriving a MIN problem: Given a numeric problem $N = \langle\mathcal{A}_N, \mathcal{C}_N, m_N, D_N\rangle$ representing a logical scene as input, we define a corresponding metaheuristic minimization problem $M_{min} = \langle\mathcal{V}_M, \mathcal{OF}_M\rangle$ as follows:

1) $\mathcal{V}_M$ collects the variables $v \mid \vec{a}_i.\texttt{v}$ that define all actors $\vec{a}_i \in \mathcal{A}_N$
2) $\mathcal{OF}_M$ is a set of objective functions derived from distance functions associated to numeric constraints $\mathcal{C}_N$.

Relation category	Distance functions ($DF_{pos}, \ldots, DF_{road}$)
$r(\circ_A, \circ_B) \mid r \in \mathrm{R}_{pos}$	(DF_{pos}): If $\circ_B$ is within the circular sector defined by r, DF_{pos} returns 0. Otherwise, DF_{pos} returns the angle relative to $\circ_A$ between the $\overline{\circ_A \circ_B}$ segment and the closest edge of the circular sector defined by r.
$r(\circ_A, \circ_B) \mid r \in \mathrm{R}_{dist}$	(DF_{dist}): If both actors are at an appropriate distance, DF_{dist} returns 0. Otherwise, DF_{dist} denotes the shortest distance that $\circ_B$ must traverse to be positioned within the distance bounds required by r.
$r(\circ_A, \circ_B) \mid r \in \mathrm{R}_{vis}$	(DF_{vis}): If $\circ_A$ can see $\circ_B$, DF_{vis} returns 0. Otherwise, DF_{vis} denotes the shortest distance that $\circ_B$ must traverse for at least one of its corners to be in the field of view of $\circ_A$.
$r(\circ_A, \circ_B) \mid r \in \mathrm{R}_{coll}$	(DF_{coll}): DF_{coll} returns 0 if $\circ_A$ and $\circ_B$ are positioned such that they do not collide, 1 otherwise.
$r(\circ_A, \circ_A) \mid r \in \mathrm{R}_{road}$	(DF_{road}): If all four corners of $\circ_A$ are placed on a road contained in *RoadMap*, DF_{road} returns 0. Otherwise, DF_{road} denotes the shortest distance that $\circ_A$ must traverse for $\circ_A$ to be placed on a road.

Fig. 7. Informal overview of distance functions derived from positive functional relations.

Distance functions: Each numeric constraint $c \in \mathcal{C}_N$ has a corresponding distance function $DF_i(c)$ which is derived according to the relation category of c (e.g. different functions for visibility and positional constraints). A distance function $DF_i(c)$ returns a non-negative number that represents how far a candidate solution CS is from satisfying c. $DF_i = 0$ iff the corresponding numeric constraint c holds for the variable assignment defined by CS.

In case of positive constraints (i.e. derived from positive functional relations), DF_i is computed according to Fig. 7. Negative constraints are defined similarly, where 0 is returned if the negation of the corresponding (positive) constraint is satisfied. The set of distance functions is easily extensible by other relations defined by experts without further change in the underlying MIN problem.

Objective functions: Many MHS algorithms are designed to handle a reduced number of objective functions. As such, fitness values of multiple distance functions may be combined into an aggregate objective function $OF_{P_i} \in \mathcal{OF}_M$ according to various aggregation strategies, such as:

- **Global aggregation:** All constraints are aggregated into a single objective function.
- **Category aggregation:** All constraints corresponding to the same functional relation category are aggregated.
- **Actor aggregation:** All constraints applied to the same source actor are aggregated.

Each aggregation strategy partitions the set of numeric constraints $\mathcal{C}_N$ into distinct subsets $P_i \subseteq \mathcal{C}_N$ that contain all constraints associated to a specific objective function. For instance, **category aggregation** would create a subset for each functional relation category, and **actor aggregation** would create a subset for each actor.

The choice of objective functions influences the appropriate search algorithms. For instance, single-objective optimization algorithms (i.e. genetic algorithm) are ideal for **global aggregation**, while multi-objective (e.g. NSGA-II [34]), many-objective (e.g. NSGA3 [35]) or custom optimization algorithms are ideal for other aggregation strategies.

Formally, given a subset P_i of numeric constraints, an objective function is defined as:

$$OF_{P_i} = f_{P_i}\left(\sum_{c \in P_i} DF_i(c)\right)$$

- $f_{P_i}()$ is an arbitrary (non-negative) weight function that does not modify the minima.
- $DF_i(c)$ is the output of a distance function measured over a single constraint c, as detailed in Fig. 7.

Our objective function is a weighted aggregation of distance functions. Different weights can help fine-tune different characteristics of solutions (e.g. realisticness, diversity).

Soundness: The soundness of our approach is captured by Theorem 2 (with a formal proof in the Appendix, available online):

Theorem 2: For a MIN problem M_{min} derived from a numeric problem N, a solution s_{min} to M_{min} is also a solution to N (i.e. all numeric constraints are satisfied).

C3 We solve scene concretization as a metaheuristic minimization problem where objective functions are derived from various aggregation strategies.

VI. Evaluation

We conducted various measurements to address the following research questions:

RQ1: Which MHS configuration provides the best scene concretization results in terms of success rate and runtime?

RQ2: How does our approach compare to state-of-the-art scene concretization approaches with respect to success rate (**RQ2.1**), runtime (**RQ2.2**) wrt. different maps, and success rate wrt. increasing number of actors (**RQ2.3**)?

RQ3: How does our approach scale/fail wrt. an increasing number of constraints?

RQ4: How does our approach scale/fail when concretizing scenes with large number of actors?

A. Case Studies

To answer these questions, we execute scene concretization campaigns over three road maps.

Carla: The Carla simulator framework [5] includes multiple road networks alongside realistic depictions of the surrounding environment. We perform experiments over the *Town02* road map (215×217 units^2) included in Carla, which represents a simple town consisting of "T junctions".

Zalazone: The *ZalaZONE Automotive Proving Ground* [36] is a physical test track located in Zalaegerszeg, Hungary designed to conduct experiments related to the safety assurance of AVs. The test track consists of various subsections adapted to the testing of different facets of AVs. We perform experiments over a digital twin of the *Smart city* portion of the test track (270×474 units^2), which features a focused urban layout with multiple complex intersections, roundabouts, parking lots and curved roads.

Tramway: We also perform experiments over a real-life road network (258×181 units^2) in Budapest, Hungary. A digital twin for this road network is derived together with an industrial partner using the public *OpenStreetMap* database. This road network features many multi-lane, complex intersections located over a dedicated tramway lane.

B. Compared Approaches

We compare our proposed approach with three variations of the baseline **Scenic** approach. Our evaluation does not consider manual or semi-automatic scene concretization for comparative evaluation considering that, despite being conceptually simple for humans, manually concretizing scenes into simulation-ready representations (1) is very time-consuming, as it relies on trial and error, and (2) does not ensure formal correctness. Automated scene concretization approaches, such as the ones presented below, address both difficulties by (1) automatically synthesizing simulator-friendly scenes (2) that are guaranteed (by the underlying algorithm) to satisfy the mathematical definitions of the abstract constraints.

Scenic: As a baseline reference, we concretize FSSs represented through the SCENIC [11] specification language. For this purpose, we use the integrated approach based on rejection sampling proposed by the framework. However, **Scenic** has expressiveness limitations (i.e. certain scene specifications cannot be expressed, see Section III-B). Therefore, we identify three variations of the **Scenic** approach which use different constructs to represent abstract relations. In our experiments, we remove the minimum number of relations to make the resulting scene expressible.

- **SceDef** (Default): **Positional** (and accompanying **distance**) relations are represented by the built-in constructs (referred to in the framework as **VENEER**s) at actor definition time which restricts the sample space to a line segment on the map relative to the position of other actors. An example of the numeric constraint corresponding to a **VENEER** is given in Fig. 2 for the $ahead(\circ_R, \circ_B)$ relation. When distance relations cannot be expressed, they are represented by **REQUIRE** clauses, which are handled as acceptance conditions. **Visibility** relations are also handled through **REQUIRE** clauses. Neither cyclic positional relations nor actors with multiple dependencies can be handled in this variation.
- **SceReg** (Regions): **Positional**, **distance** and **visibility** relations are represented as **REGION** instances, which restrict the sample space to certain regions of the map relative to the position of other actors. In this variation, functional relations are mapped to numeric constraints as proposed in this paper. This variation cannot handle cycles of positional and/or visibility relations.
- **SceHyb** (Hybrid): This variation combines the previous variations by handling **positional** relations as **REGION** instances, and representing **distance** and **visibility** relations as acceptance conditions (**REQUIRE** clauses). As such, this variation stays consistent with the $f2l$ mapping proposed in this paper. Additionally, the only expressiveness restriction is that cycles of positional relations cannot be represented.

Once a FSS is defined, the functional scene is concretized by rejection sampling. **Collision avoidance** and **road placement** relations are included as additional acceptance conditions. However, to ensure that the approach can solve the entire scene concretization problem (i.e. not only the expressible subset of the problem), we check whether a derived concretization satisfies the removed, inexpressible relations. This is implemented as an additional acceptance condition for the concretized scene that is evaluated after the termination of the default sampling approach.

MHS: We implement the MHS-based approach proposed in this paper using various objective function aggregation strategies. Aside from the *global* $\mathtt{G}$, *category* $\mathtt{C}$ and *actor* $\mathtt{A}$ strategies proposed in Section V, we also implement:

- *weighted category aggregation* $\mathtt{WC}$ (a variant of $\mathtt{C}$)
- *weighted dependency aggregation* $\mathtt{WD}$ (two objective functions are defined according to the dependency structure of constraints type (see Section VI-F): **collision avoidance** and **road placement** combine for one objective function, while the remaining constraint types form the other)
- *no aggregation* $\varnothing$ (each constraint in the FSS defines a separate objective function).

In case of $\mathtt{WC}$ and $\mathtt{WD}$, higher weight is given to constraint categories that have less dependencies (i.e. **collision avoidance** and **road placement** constraints). Specifically, the objective functions for $\mathtt{WC}$ are:

- $OF_{road}, OF_{coll} = (\sum_{c \in \mathtt{R}_i} DF_i(c))^3$, where i is $road$ and $coll$ respectively, and
- $OF_{pos}, OF_{dist}, OF_{vis} = (\sum_{c \in \mathtt{R}_i} DF_i(c))^2$, where i is pos, $dist$ and vis respectively.

The objective functions for $\mathtt{WD}$ are:

- $OF_1 = (\sum_{c \in \mathtt{R}_{road} \cup \mathtt{R}_{coll}} DF_j(c))^3$, and
- $OF_2 = (\sum_{c \in \mathtt{R}_{pos} \cup \mathtt{R}_{dist} \cup \mathtt{R}_{vis}} DF_j(c))^2$,

where j represents the relation category of c, and $c \in \mathtt{R}_i$ is shorthand for $c \in \mathcal{C}_N | c = f2l(r(\circ_A, \circ_B)) \wedge r \in \mathtt{R}_i$.

We evaluate our approach using three underlying MHS algorithms through the PYMOO Python library [39]:

- a single-objective *genetic algorithm* $\mathtt{GA}$
- a multi-objective *NSGA-II* algorithm $\mathtt{N2}$ [34]
- a many-objective *NSGA-3* algorithm $\mathtt{N3}$ [35]

Table I provides an overview of the relevant hyperparameters and genetic operators used as part of our experimentation. Genetic operators are selected according to the default PYMOO settings. Population size and number of offsprings are selected according to preliminary measurements, which are included on the publication page. While small population sizes are rather unusual for MHS algorithms, we believe their good performance is attributed to the particularities of our experimental setup (e.g. MHS approaches are often designed to provide multiple partial solution, whereas in our experimentation a single, complete solution is retrieved).

The implemented MHS configurations handle all functional constraints of our scene specification language, and handle logical constraints of Section IV. As such, input scenes do

TABLE I
OVERVIEW OF THE HYPERPARAMETERS AND GENETIC OPERATORS USED FOR EACH EVALUATED MHS ALGORITHM. THE TABLE REFERS TO *SIMULATED BINARY CROSSOVER* (SBX) [37], *POLYNOMIAL MUTATION* (PM) [37], THE *DAS-DENNIS* (DD) [38] APPROACH FOR DETERMINING THE NUMBER OF REFERENCE DIRECTIONS $n_{refDirs}$, *BINARY TOURNAMENT* (BT), AND *NON-DOMINATED SORTING* (NDS). FURTHERMORE, *PR* REFERS TO PROBABILITY AND η REFERS TO THE DISTRIBUTION INDEX, WHILE IN OUR IMPLEMENTATION, $n_{var} = 2 \times n_{actors}$

	Pop. Size	$n_{offsprings}$	**Crossover op.**	**Mutation op.**	**Selection op.**	**Survival op.**
GA	5	5	SBX(pr=0.9, η=3)	PM($pr=\frac{1}{n_{var}}$, η=5)	Fitness BT	Fitness
N2	5	5	SBX(pr=0.9, η=15)	PM($pr=\frac{1}{n_{var}}$, η=20)	Fitness Domination and Crowding Distance BT	NDS Rank and Crowding Distance
N3	$n_{refDirs}$	$n_{refDirs}$	SBX(pr=1.0, η=30)	PM($pr=\frac{1}{n_{var}}$, η=20)	Random BT	NDS Rank and Reference Direction DD($n_{dimensions}$=$n_{objectives}$, $n_{partitions}$=1)

TABLE II
COMPARISON OF RELATION REPRESENTATION AND HANDLING OF RELATIONS BY CONCRETIZATION APPROACHES

Approach	Representing Relations		
	Positional	Distance	Visibility
SceDef	**VENEER**	**VENEER/REQUIRE**	**REQUIRE**
SceReg	**REGION**	**REGION**	**REGION**
SceHyb	**REGION**	**REQUIRE**	**REQUIRE**
MHS	**NATIVE**	**NATIVE**	**NATIVE**

Approach	Handled relation structures		
	Positional cycles	Visibility cycles	Multiple dependencies
SceDef	**NO**	**YES**	**NO**
SceReg	**NO**	**NO**	**YES**
SceHyb	**NO**	**YES**	**YES**
MHS	**YES**	**YES**	**YES**

not need to be adjusted, as is the case for **Scenic** approaches. A comparison of relation representations, and the handling of relation structures is provided in Table II. Furthermore, to stay consistent with the baseline **Scenic** approaches, (1) our experimentation runs are terminated once *a single solution* to the problem defined by the FSS is found, and (2) particular handling for same-scene concretization tasks is not implemented.

C. General Measurement Setup

To evaluate various concretization approaches, we randomly generated FSSs to be used as input:

1) Given a number of required actors, we create a preliminary FSS P_{pre} used to derive the input FSSs for our experiments. P_{pre} contains road placement constraints for each actor and collision avoidance constraints for each pair of actors, to yield realistic concrete scenes. Additionally, a maximum distance r between actors is established to avoid randomly generating scenes where no relations exist between actors.
2) We use the default sampling-based scene concretization approached proposed in SCENIC [11] to derive a numeric solution s_{pre} for a numeric problem $N_{pre} = f2l(P_{pre})$. Note that different runs yield different numeric solutions for a same P_{pre} given as input.
3) We derive a FSS P_{in} by applying qualitative abstractions over s_{pre} (i.e. $P_{in} = l2f(s_{pre})$). Qualitative abstractions are applied for all relation categories. As such, any **positional**, **distance** and **visibility relations** that hold in s_{pre} are included in P_{in}. Additionally, we know that P_{in} is not contradictory, as it has at least one feasible solution (s_{pre}).
4) We use P_{in} as input for our measurements.

Throughout our measurements, for simplicity, we consider actors with pre-defined length and width. Additionally, we refer to the road network to determine the expected heading at a given position in the map. We assume that each position has a single expected heading (we avoid non-determinism at intersections, where vehicles may take multiple paths that cross each other). As such, our scene concretization runs derive the position coordinates of each actor, and the heading is determined accordingly.

We performed the measurements on an enterprise server[2]. Measurements are run in a Python environment, and the garbage collector is called explicitly between runs. All generated FSSs, concrete scenes along with related measurements and figures are included in the publication page.

D. RQ1: Comparison of MHS Configurations

Measurement setup: *This experiment aims to determine which optimization algorithm and objective function aggregation strategy combination provides the best results when implementing our proposed approach.* We perform measurements over the **TRAMWAY** map (which is shown in **RQ2** to be of intermediate difficulty for the evaluated approaches) using scenes with 2, 3 and 4 actors (size) to compare 8 MHS configurations. We set target scene sizes aligned with the level of scalability offered by many recent publications [1], [17], [19], [21], [22], [23], [40], [41], [42] that handle local traffic constraints over individual actors (such as the ones proposed in our paper).

Each evaluated configuration has an objective function aggregation strategy and a MHS algorithm selected accordingly. Single-objective optimization `GA` is used with `G`, which yields one objective function. Multi-objective optimization `N2` is used for at most 3 objective functions: `A`, `WD`. Many-objective optimization `N3` is used for more than 3 objective functions: `C`, `WC` and `∅`. Moreover, we evaluate the `N2-WC` configuration (as initial measurements provided promising results) and the `N3-A` configuration (as the number of objective functions increases with the number of actors).

For each scene size, we randomly generate 10 FSSs as inputs (see Section VI-C), and run each approach 10 times, for a total of 100 runs. A 10-minute time-out is set for each run.

[2]12 × 2.2 GHz CPU, 64 GiB RAM, CentOS 7, Java 1.8, 12 GiB Heap

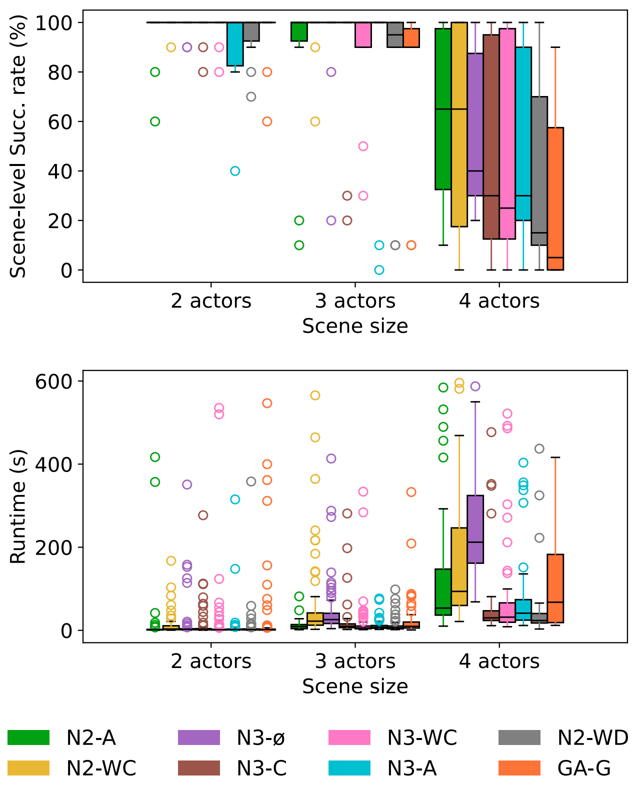


Fig. 8. Scene-level success rate and runtime measurements for various MHS configurations on the **Tramway** map.

Analysis of results: Success rate and runtime measurements comparing the MHS configurations are provided in Fig. 8. Each configuration is depicted with a uniquely colored box. For each configuration, we evaluate scene-level aggregate success rate: we derive a cumulative success rate for each of the 10 FSSs (each FSS is subject to 10 runs) and depict their distribution.

We determine which configuration is best suited for our proposed approach by evaluating the statistical significance of our results. For *success rate measurements*, we perform the *Fisher exact test* [43] to determine p-value and measure the *odds ratio* [44] for effect size, as suggested in existing guidelines [45] for comparing algorithms with dichotomous outcomes (i.e. success or failure). Analysis shows that (despite underperforming for 3-actor scenes), the `N2-A` configuration provides better success rate with statistical significance ($p < 0.05$) compared to all other configurations except for `N2-WC` and `N3-ø`. However, effect size is not large for our measurements (between 1.845 and 4.378).

For *runtime measurements*, we perform the *Mann-Whitney U-test* [46] to determine *p-value* and measure the Vargha and Delaney's $\hat{A}_{12}$ [47] for effect size. Additionally, following existing guidelines [45], we only consider the times of successful runs. Analysis shows that the `N2-A` configuration provides better runtime with statistical significance ($p < 0.05$) than the `N2-WC` and `N3-ø` configurations for all three scene sizes. Effect size is medium to large for our measurements (between 0.662 and 0.853).

Considering these results, we identify `N2-A` as the best configuration for our experimental setup and use it for comparison with **Scenic** approaches in **RQ2**. Furthermore, we select `N2-A`, `N2-WC` and `N3-C` as the top three promising configurations to be evaluated in scalability measurements (**RQ3** and **RQ4**). Despite its slightly worse success rate, we select `N3-C` over `N3-ø` due to its significantly faster runtimes.

RQ1: *Out of the 8 evaluated MHS configurations, `N2-A` either provides significantly better success rate or better runtime compared to all other configurations.*

E. RQ2: Comparing MHS With Scenic Approaches

Measurement setup: *This experiment aims to determine which approach completes scene concretization in reasonable time (**RQ2.1**, **RQ2.2**). Given a particular scene specification as input, we also determine (**RQ2.3**) which approach is most likely to succeed in concretizing it.* We perform measurements over the 3 road maps using scenes with 2, 3 and 4 actors to compare the 4 concretization approaches. As in **RQ1**, we exclude larger scene sizes to enable cross-approach comparisons with higher success rates. Additionally, as discussed in **RQ1**, **MHS** refers to the `N2-A` configuration of our approach.

For each size and map, we randomly generate 10 FSSs as inputs (see Section VI-C), and run each approach 10 times (each with a time-out of 10 minutes), for a total of 100 runs.

Analysis of results (RQ2.1): We compare the overall success rate wrt. different maps in the top row of Fig. 9. Each figure contains cumulative success rate results for all four approaches, depicted in the corresponding color (scene-level success rate is addressed in **RQ2.3**). For **Scenic** approaches, a run is considered to be successful if the approach provides a solution to the input partial problem that also satisfies the removed relations (i.e. the provided solution solves the complete problem).

Among the **Scenic** approaches, **SceHyb** consistently provides relatively high success rates. Nevertheless, for all maps, and for all scene sizes, the success rate is dominated by **MHS**. Particularly for 4-actor scenes, **Scenic** approaches are generally unable to provide any solutions, while the success rate of **MHS** varies between 56-75%.

We evaluate statistical significance of our results according to existing guidelines [45] as detailed in Section VI-D. For each configuration (map and scene size), we evaluate the statistical difference between **MHS** and the **Scenic** approach with the highest success rate. The statistical test results are shown in Table III. *p-values* are lower than 0.05 for all configurations, and the lowest *effect size* is 7.0. Hence, the success rates are significantly higher (with large effect size) for **MHS** compared to **Scenic** approaches.

RQ2.1: *For success rates, **MHS** dominates all **Scenic** approaches with statistical significance. **Scenic** approaches reach their scalability limit at 4 actors with close to 0 success rate, while the success rate of **MHS** is still 56-75%.*

Analysis of results (RQ2.2): We compare the runtimes wrt. different maps in the middle row of Fig. 9. Each figure

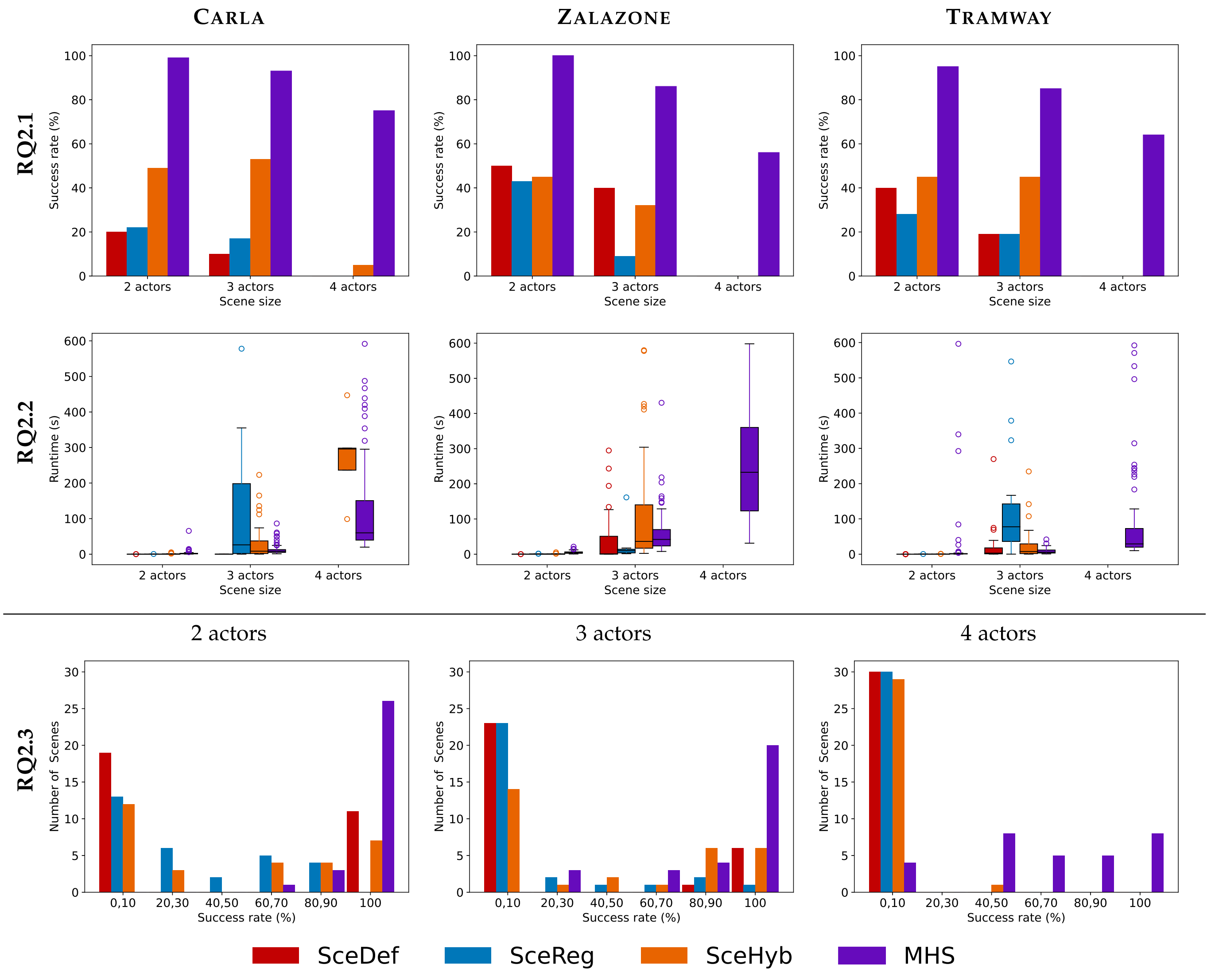


Fig. 9. Measurement data for success rate (**RQ2.1**) and runtime (**RQ2.2**) wrt. different maps, and success rate wrt. increasing number of actors (**RQ2.3**).

TABLE III
STATISTICAL TEST RESULTS COMPARING SUCCESS RATES: **MHS** VS. THE BEST **SCENIC** APPROACH WRT. MAP AND SCENE SIZE (*p*: *p-Value*, *e*: *Effect Size*). EFFECT SIZE IS *LARGE* FOR ALL DATA POINTS

Map	Scene size					
	2 actors		3 actors		4 actors	
	p	*e*	*p*	*e*	*p*	*e*
CARLA	5.22e-18	103.0	1.01e-10	11.8	2.26e-26	57.0
ZALAZONE	4.45e-19	∞	1.35e-11	9.2	4.94e-22	∞
TRAMWAY	1.37e-15	23.2	3.55e-09	7.0	2.33e-26	∞

contains measurement results for scenes with up to 4 actors. Results for all four of the approaches are depicted as color-coded box plots.

Is Scenic faster than MHS? For scenes with 2 or 3 actors, the **MHS** approach is generally slower than the **Scenic** approaches. We evaluate the statistical significance of our results according to existing guidelines [45], see Section VI-D.

For each configuration (map and scene size), we evaluate the statistical difference between the **MHS** approach and each **Scenic** approach (pairwise). For 2-actor scenes, all pairwise comparisons show a statistically significant difference ($p < 0.05$) with large effect size ($\hat{A}_{12} > 0.85$) in favor of the **Scenic** approaches. For 3-actor scenes, there is only a statistically significant difference in favor of **SceDef** for the **CARLA** and **ZALAZONE** maps and of **SceReg** for the **ZALAZONE** map (with large effect size, $\hat{A}_{12} > 0.7$).

How much faster is Scenic? To evaluate how much faster **Scenic** approaches are, we multiply the **Scenic** runtimes by a constant factor c, then we evaluate the statistical difference between the new runtimes and the **MHS** runtimes (pairwise). If no statistically significant difference is detected, we conclude that the given **Scenic** approach is at most c times faster than the **MHS** approach.

For 2-actor scenes, our results show that **SceDef** is 141 times faster, **SceReg** is 19 times faster, and **SceHyb** is 13 times faster than **MHS**. However, while **Scenic** approaches are often very fast (1-10 milliseconds), **MHS** also provide results in reasonable time.

For 3-actor scenes, **Scenic** approaches are at most 2.7 times faster than **MHS** approaches. An exception is for the **SceDef** approach applied to the **Carla** map, where **SceDef** is 122 times faster. However, in this case, there is a significant difference in success rate between the two approaches, favoring **MHS**.

For 4-actor scenes, very few data points exist for **Scenic** approaches as they failed to provide a solution in most runs. **MHS** data shows that the median runtimes of successful runs are 29.1s for **Tramway**, 60.0s for **Carla** and 232.8s for **Zalazone**. As such, we notice that the **Zalazone** map provides the most challenging scene concretization problem for **MHS** in terms of runtime, while also providing comparable or lower success rates as other maps (see **RQ2.1**). This is attributed to the large map size and complex structure (containing many unusual road segments), which affects the search space for **MHS**.

RQ2.2: *For scenes with 2 or 3 actors, **Scenic** approaches are 1-2 orders of magnitude faster than **MHS**, which can still provide results in reasonable time (with better success rates). For 4-actor scenes, only **MHS** is successful, with a median runtime of at most 232.8s (reported for the **Zalazone** map).*

Analysis of results (RQ2.3): Success rates wrt. increasing number of actors are shown in the last row of Fig. 9. Each figure *aggregates data for all measurements* (i.e. for all maps and scenes) performed for *a given number of actors*.

For this research question, we evaluate scene-level success rate, as in **RQ1**. Specifically, each bar in these figures represents the number of scenes where the associated success rate corresponds to the x-axis label. For instance, the bottom-left subfigure of Fig. 9 shows that there are 19 (2-actor) scenes where **SceDef** provides a success rate of 0% or 10% (the leftmost bar). Similarly, there are 26 scenes where **MHS** provides a success rate of 100% (the rightmost bar).

For 2-actor and 3-actor scenes, (1) **SceDef** provides distributions skewed towards lower and higher success rates, (2) **SceReg** and **SceHyb** provide more uniform distributions, and (3) **MHS** provides a distribution skewed towards higher success rates. This shows that certain scenes cannot be concretized by **Scenic** approaches, while **MHS** provides at least 20% success rate for every 2-actor or 3-actor scene.

For 4-actor scenes, **MHS** provides a uniform distribution of success rates, while **Scenic** approaches most often cannot solve the concretization problems. In fact, **MHS** was able to provide at least 1 solution (i.e. at least 10% success rate) for *every scenes* with 4 actors.

RQ2.3: *The **MHS** approach provides at least one solution (i.e. a 10%+ success rate) for every input scenes (i.e. concretization problems) with increasing number of actors. For an arbitrary practical scene concretization problem, **MHS** is more likely to find a solution than **Scenic** approaches.*

F. RQ3: Scalability Analysis wrt. Constraints

Measurement setup: *This experiment aims to determine how the inclusion of additional constraints influences runtime and success rate for three promising **MHS** configuration.* For this research question, our measurements are restricted to the most challenging 4-actor **Zalazone** configuration (see **RQ2**). As discussed in **RQ1**, we evaluate the `N2-WC`, `N2-A` and `N3-C` configurations of our approach. Furthermore, since **Scenic** approaches failed to handle scenes with 4 actors, we exclude them from our scalability measurements.

We use the same 10 scenes used for **RQ2**, however, we gradually build up the scenes by including all constraints of a certain type one by one and then performing measurements after adding each constraint type. Specifically, we start with scenes containing no constraints (∅), then we gradually add road placement (R), collision avoidance (C), positional (P), distance (D) and visibility (V) constraints until we reach the complete scene specification.

The order of adding constraint types is based on their dependencies: (1) collision avoidance (C) is irrelevant for realistic initial scene generation if vehicles are not placed on roads (R), (2) distance (D) and position (P) cannot be measured if vehicles are overlapping (C), and (3) visibility (V) is based on position (P). While, for each constraint type, the exact number of added constraints may vary between scenes, adding constraints by type ensures that, for a given scene, the number of constraints gradually increases.

As in **RQ1** and **RQ2**, we perform 10 iterations per scene with a 10-minute time-out.

Analysis of results: Measurement results for **RQ3** are shown in the top row of Fig. 10. As in **RQ1**, a distribution of scene-level success rate is depicted. For `N2-WC`, the gradual inclusion of constraint types results in a decrease in success rate and an increase in median runtime. Similar trends are also observed for `N3-C` and for `N2-A`, however, only up to the inclusion of distance (D) constraints. In all cases, these trends become particularly noticeable after including positional (P) constraints, which constitute *complex constraints* involving *multiple vehicles* (unlike e.g. road placement constraints).

For `N3-C`, the further inclusion of visibility (V) constraints results in an *increase* in success rate and a *stagnation* in runtime. Although these results might seem counter-intuitive, they are in line with the behavior of popular SAT solvers where the increasing number of constraints does not necessarily correlate with the increasing complexity of the underlying constraint satisfaction problem [48].

For `N2-A`, the further inclusion of visibility (V) constraints results in a *stagnation* in both success rate and median runtime, which is attributed to the use of `A` as the objective function aggregation strategy. `C` and `WC` both produce a new objective function for each newly added constraint type, which explains the observed variation in success rate and runtime throughout the experiment. However, `A` produces a constant number of aggregation functions regardless of the included constraints. Furthermore, visibility constraints have a similar formalization as *ahead()* positional (P) constraints. As such, according to our

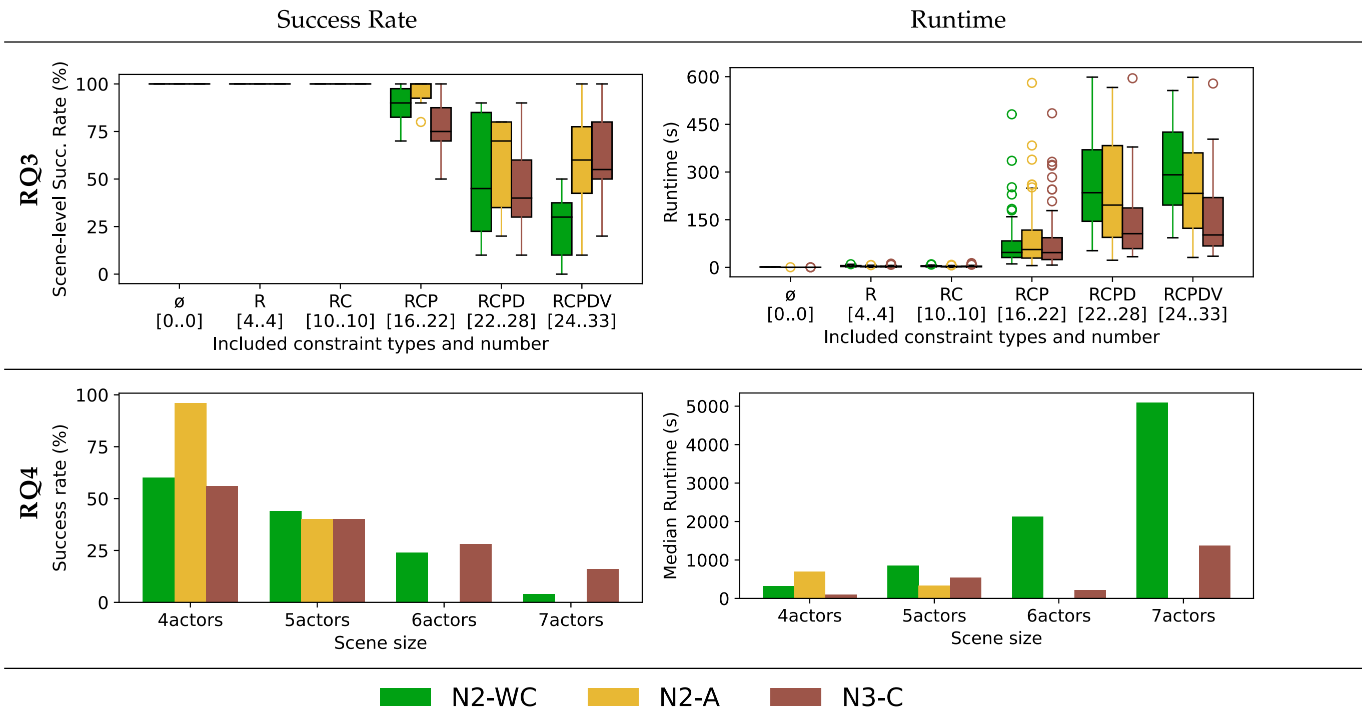


Fig. 10. Scalability results (for **MHS** runs on the **Zalazone** map) wrt. constraints (for 4 actors) (**RQ3**) and wrt. number of actors (**RQ4**). We report measurement data for success rate and runtime.

random FSS generation approach described in Section VI-C, scenes that contain visibility constraints likely also contain *ahead*() constraints. Considering that *ahead*() constraints are already handled with the inclusion of positional (P) constraints, the addition of visibility constraints should not significantly influence success rate and runtime, as shown in our results.

RQ3: *Gradual inclusion of constraint types generally results in a performance decrease for the* ***MHS*** *approaches, particularly with the addition of positional constraints. Furthermore, the choice of objective function aggregation strategy influences scalability results wrt. constraints.*

G. RQ4: Scalability Analysis wrt. Actors

Measurement setup: *This experiment aims to determine the maximum size of a scene specification that our approach can successfully concretize in reasonable time.* For this research question, we measure the scalability of the **MHS** approach by introducing more than 4 actors. As in **RQ3**, we evaluate the N2-WC, N2-A and N3-C configurations on the most challenging **Zalazone** map.

We perform measurements over 5 randomly generated input scene specifications with increasing size up to the point where the **MHS** approach is predominantly failing (i.e. under 10% success rate). We run concretization 5 times for each scene specification, for a total of 25 runs per scene size per **MHS** configuration. However, we increase the time-out to 2 hours for each run and measure the success rate and runtime of the concretization runs.

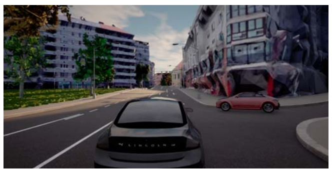
Analysis of results: Measurement results for **RQ4** are shown in the bottom row of Fig. 10. Despite a high success rate for 4-actor scenes, the N2-A configuration is only capable of concretizing scenes with up to 5 actors. For N2-WC, scalability is limited to 6-actor scenes. N3-C is the most scalable configuration and can concretize scenes with up to 7 actors (our results show a 16% success rate for 7-actor scenes). Additionally, despite the 2-hour time-out, the median runtime for the N3-C configuration for 7-actor scenes is 1375s (23 minutes).

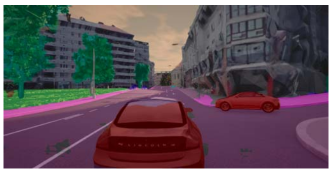
Note that increasing the number of actors by one represents an exponential increase in the overall complexity of the scene concretization problem. Consider a scene containing m actors defined over a set of n directed binary relation symbols. The size of the entire search space is estimated (i.e. over-approximated) as $(2^n)^{m(m-1)}$, where (i) 2^n is the number of possible relation combinations over an ordered pair of actors, and (ii) $m(m-1)$ is the number of ordered actor pairs in the scene. As such, the search space complexity is $\mathcal{O}(2^{nm^2})$. In particular, the largest search space handled by **MHS** (for 7 actors) is 2^{360} times (over 100 orders of magnitude) larger than the 3-actor space handled by **Scenic** approaches.

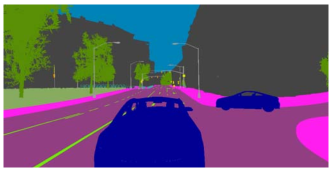
RQ4: *The scalability of* ***MHS*** *approaches is limited to scenes with 7 actors over the* ***Zalazone*** *map which solves a search space with 2^{420} states. As such,* ***MHS*** *can handle an exponentially (over 100 orders of magnitude) larger search space compared to* ***Scenic*** *approaches.*

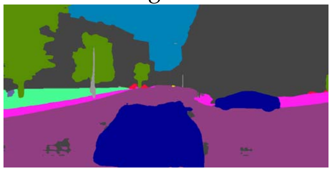
H. Towards Testing Vision-Based ML Components for Semantic Segmentation

Although our initial scene concretization approach is a first step towards complete scenario-based testing of AVs, concrete scenes are commonly used for testing various components of


(a) RGB image as simulated in CARLA


(b) Predicted SEG overlayed on Figure 11a


(c) Ground truth semantic segmentation


(d) Predicted semantic segmentation using *ANN*

Class	Precision	Recall	Intersection-over-union
Vehicles	0.991	0.941	0.934
Sidewalks	0.946	0.488	0.474
Traffic Signs	0.6	0.094	0.089

(e) Metrics measurement data

Fig. 11. Artifacts derived from using a generated scene to test the semantic segmentation capabilities of *ANN*.

AVs, such as cameras and LiDAR sensors. As a proof of concept, we provide initial results for the testing of the semantic segmentation functionality of a computer-vision component by integrating our proposed approach with the realistic CARLA simulator [5], which includes a simple AV stack that may be tested. Additionally, in an ongoing work, we use our scene concretization technique to evaluate three computer-vision components, namely *SegFormer* [49], *ANN* [50] and *BiSeNet V2* [51]. However, the complete evaluation of these components is outside of the scope of this paper, which primarily focuses on the core scene concretization technique.

Fig. 11 shows an example where a 3-actor scene generated by our approach is used as a test case to evaluate a computer-vision component. For such scenes, we provide:

1) an RGB image of the scene (Fig. 11a) from the ego vehicle's viewpoint (dashcam footage),
2) ground truth semantic segmentation (SEG) of the RGB image as provided by CARLA (Fig. 11c),
3) predicted SEGs of the RGB image using the three listed computer vision components (a sample SEG using *ANN* is shown in (Fig. 11d),
4) an overlay of the predicted SEGs over the RGB image (Fig. 11b), and
5) initial measurement data for *precision*, *recall* and *intersection-over-union* (*IoU*) [52] for different classes of objects in the scene (Fig. 11e). This measurement data is used to evaluate the performance of the computer-vision component under test.

The preliminary measurements shown in Fig. 11 indicate that *ANN* most successfully detects vehicles in the test scene. *ANN* also successfully identifies two of the three sidewalk segments in the image, while it performs poorly in the identification of traffic signs. As such, our preliminary results show that the depicted scene may act as a good test case for traffic sign and sidewalk detection. As future work, we plan to build upon these preliminary results to guide scene generation towards challenging object detection test cases.

In addition to the previously listed artifacts, we also include (online on the publication page):

6) a **Scenic** description containing exact positions of each vehicle (for reproducibility), and
7) a video of the scene running with the default AV stack included in CARLA.

Further initial results (for 30 scenes located on the ZALAZONE map containing 2, 3 or 4 actors) are also available on the publication page. These scenes are derived such that all non-ego vehicles can be seen by the ego vehicle, which increases the relevance of our auto-generated scenes for the testing of vision-based ML components (as each object has an impact on the test).

I. Threats to Validity

Construct validity. Our approach represents a traffic scene as a set of pre-defined abstract relations. In this paper, we excluded certain relations associated to, e.g., vehicle size and orientation, but our proposed approach may be generalized to include those relations. Additionally, we use various approximations (i.e. actors are modeled as rectangles) in our definition of numeric constraints to simplify the scene concretization task. Nevertheless, we provide a complete conceptual basis for scene concretization with an *extensible specification language* and *customizable numeric constraints*.

We compare various MHS algorithms and objective function aggregation strategies to solve the derived MIN problem. We selected population size and number of offsprings based on preliminary measurements. Default values are used for all other parameters.

Internal validity. To strengthen internal validity, we explicitly call the garbage collector between scene concretization runs. Additionally, we derive the input functional scenes in such a way to ensure its feasibility on the tested map (to avoid contradicting relations). Our input scene derivation approach is only limited by an enforced maximum relative distance between actors, which does not restrict the diversity of the derived scenes.

External validity. We mitigate threats to external validity by performing measurements over 3 maps with different characteristics, derived from different sources (including a real test track and a real world location). We perform comparative evaluation of up to 8 MHS configurations, composed of 3 MHS algorithms and of 6 objective function aggregation strategies. We also compare our proposed approach to three variations of the state-of-the-art baseline approach, until their scalability limits are reached.

For **RQ1** and **RQ2**, we perform a thorough evaluation (100 concretization runs per approach/configuration, per map, per scene size) to adequately compare the approaches and to cover a diverse set of scenes for each setting. We accompany our measurement with statistical significance analysis in accordance with the best practices. We also perform thorough evaluation for

RQ3 (100 concretization runs per **MHS** configuration, per set of constraint types). Our scalability analysis for **RQ4** is restricted to MHS approaches (as the baseline **Scenic** approaches failed to provide solutions in **RQ2**). Here, we limit our evaluation to 25 concretization runs, over 5 scenes per size, which is still sufficient to illustrate the scalability of our MHS approach.

VII. Discussion

In this section, we discuss the practical implications of the proposed approach and we situate it within a longer-term research strategy towards rigorous certification of AVs.

A. Practical Implications

Complexity of modeling: Despite handling large search spaces with up to 2^{420} states, the complexity of modeling within our approach is only internal. In other words, the burden on users of our tool is minimal: users only need to provide a compact model to define functional scene specifications at a high-level of abstraction (Section III).

The large, complex internal models are observed only while concretizing scenes and are handled by (third party) MHS algorithms (in Section V). Furthermore, the large internal complexity of modeling is a necessary consequence of our highly expressive input language. As a core contribution, our proposed approach handles a wide breadth of possible input scenarios, which justifies the complexity of the internal models that our approach can handle.

Scalability: Our proposed approach reliably generates scenes with up to 4 actors (and even up to 7 actors, with a longer timeout). In the context of state-of-the-art AV testing, this level of scalability seems to be the standard. Our paper clearly shows that our proposed approach dominates Scenic, which has been used in various practical applications. Additionally, many recent publications that handle local traffic constraints over individual actors (such as the ones proposed in our paper) are limited to scenes with, 1 actor [19], 2 actors [17], [21], 3 actors [1], [22], [23], [40], [41] or 4 actors for an intersection testing scenario [42].

Naturally, there may be other practically applicable AV testing approaches that require significantly more actors. For instance, test scenarios on busy highways (also presented in [42]), or in bumper-to-bumper traffic. However, these test cases do not require the handling of local constraints over specific vehicles such as the ones proposed in this paper (they may handle global constraints, e.g. traffic density).

Customizability and extensibility: As defined in Section IV-B, our proposed mapping is *customizable*: mathematical formulae used to represent a functional relation may be adjusted (i.e. its constants may be adjusted) according to use case requirements. For instance, the visibility angle may vary according to the type of scenarios (e.g. field of view is reduced for high-speed scenarios).

As discussed in Section IV-C, our mapping is *extensible*: experts may define any new (arbitrary) functional relations (by qualitative abstractions), along with the corresponding numeric constraints and fitness functions. Our framework can automatically apply MHS over the defined fitness functions while evaluating all other fitness functions from the input functional scene specification.

Our framework also provides support for manual validation of newly added functional relations. For that purpose, a developer may (1) implement the new functional relation (e.g. a *veryFar*($\circ_A$, $\circ_B$) distance constraint), (2) create a functional scene specification containing only the newly implemented constraint, (3) run the scene generation, (4) after observing the generated scene, visually validate whether the involved actors are indeed satisfying the newly implemented constraint (i.e. if two actors are indeed very far from each other, which would be easy to observe), and (5) make adjustments if necessary. For further validation, developers may also follow a similar approach to address the negation of the newly implemented constraint. Such validation approaches are common for constraint handling tools such as Alloy [53].

B. Research Outlook

Our scene concretization and two-step scenario generation approach can contribute to a long-term strategy for rigorous AV certification to ensure AV safety criteria related to all possible (practically relevant) scenarios.

Existing research: Given the high-level AV certification challenge presented in Section I, one way to address this objective is to automatically derive a suite of relevant test scenarios. As an initial step towards AV certification, existing approaches generate tests to investigate a *specific* scenario configuration, which entails (1) a given target maneuver for the AV-under-test (2) at a given location (3) with a given placement of actors (defined as a set of abstract relations) and a corresponding maneuver assignment.

For instance, Calo et al. [22] generate scenarios at a 4-way intersection where “the [ego vehicle] is proceeding on its lane and two cars are crossing the main road from left to right”. Other approaches [1], [41] generate scenarios on a straight road leading up to an intersection which feature an ego vehicle on the straight road, a leading non-ego vehicle and a pedestrian that is crossing the intersection.

The practical benefit of existing approaches is to derive scenarios where the AV is subject to dangerous situations to expose bugs in AV behavior. However, it is unclear how such test scenario generation approaches customized for a given configuration (provided a priori) would generalize (out-of-the-box) to satisfy broader certification criteria that involve other, unrelated scenarios. For instance, it is unclear whether an approach for test generation at intersections [22] would be applicable to e.g. overtaking scenarios on a highway without further customization. As such, existing approaches address *a part of the certification objective* mentioned above by *fixing a specific scenario configuration*.

Our contribution: Rigorous certification of AVs (potentially over arbitrary scenarios) requires an approach that *inherently handles* arbitrary scenarios. It is not scalable to list all possible scenarios and to design customized scenario generation approaches for each of them. Our approach addresses *a different fragment of the certification challenge* by inherently handling *arbitrary scenes* and *arbitrary locations*.

We believe that the contributions of the paper, particularly those related to handling arbitrary scenes over arbitrary map locations, form a *novel and necessary* conceptual base towards AV certification with *guarantees over the entire space of valid scenarios that can be represented with a given vocabulary of relations* (as opposed to guarantees limited to a given scenario, as offered by existing research). This fits into our *long-term research strategy* that follows the *divide-and-conquer principle to gradually approach rigorous certification*: first, we address the challenge of handling *arbitrary scenes at arbitrary locations*, and we later focus on the concepts required for adequate handling of arbitrary AV maneuvers.

VIII. Related Work

First, we overview existing *abstract scenario specification languages* that describe scenes and scenarios at a functional level (Fig. 12a). We evaluate them with respect to *(a) expressiveness* (to represent arbitrary scenarios and constraints), *(b) extensibility* with custom traffic concepts, *(c)* handling of *temporal*, e.g. behaviors, *(d)* support for static *error detection*, and *(e)* prior use in scenario *concretization*.

We then evaluate existing (concrete) *scenario generation approaches* (Fig. 12b) (as a generalisation of scenario concretization approaches) to check if they can handle *(a) arbitrary abstract scenario specifications* as input (i.e. assumptions about interactions between actors are not hard-coded into the approach), *(b) adjustable numeric constraints*, and *(c)* any underlying *road map given as input*.

		Expressive	Extensible	Temporal	Err. det.	Concret.
Ont.s & models	[24], [57], [58]	●	●	●	○	○
Temporal scen.s	[59]–[61]	◐	◐	●	○	◐
Gener. appr.s	[1], [62], [63]	○	○	◐	○	●
Scenic	[11]	◐	◐	●	○	●
Our approach		●	●	○	●	●

(a) Comparison of *abstract scenario specification languages*

		Arbitrary scen.	Adj. num. cons.	Map as input
Search-based	[1], [2], [23]	◐	○	○
Sampling-based	[62], [64], [65]	◐	◐	○
Path Planning	[42], [63]	○	◐	◐
Scenic	[11]	●	◐	●
Our approach		●	●	●

(b) Comparison of *scenario generation approaches*

Fig. 12. Comparison of our approach with the existing state of the art. Notation – ●: yes, ◐: to a certain extent, ○: no.

Input languages for generation approaches: Existing scenario generation approaches (detailed in Section VIII-B), often define functional scenarios with a custom specification language that may include temporal concepts. However, such input languages are often tailored to represent a specific type of scenario (e.g. with a single maneuver decided a priori). As such, they lack in expressiveness and extensibility. An exception is Scenic [11], which is thoroughly discussed in this paper. Despite its expressiveness limitations, Scenic handles arbitrary scenario specifications as input.

A. Scenario Specification Languages

Existing scenario specification languages often build upon a conceptual basis for traffic scenarios proposed by Ulbrich et al. [14], by Menzel et al. [15] and by Steimle et al. [54]. These approaches describe the various components and abstraction levels required for scenario specification. At a conceptual level, scenarios are often defined through multi-layer representations [24], [27], [55], [56] which provide a hierarchy for traffic scenario components.

Ontologies and models: *Ontologies* [24], [27], [57], [58] can provide a formal basis for functional scenario specification. A similar level of formality is provided by *model-based scenario* specification approaches [66], [67]. Conceptually, these approaches represent an expressible and extensible specification language, but they are not often used in existing research as inputs for scenario concretization yet.

Temporal scenario concepts: Other specification approaches use *temporal concepts* as building blocks for scenarios. Such temporal concepts include (1) reasoning over vehicle paths [59], (2) sequence of vehicle behaviors [60] or (3) conditional state transitions between scenes [61]. The expressivity and extensibility of these approaches are limited, but initial concretization results are often provided.

B. Scenario Generation Approaches

Search-based approaches are most commonly used for scenario generation. Many-objective search can be used to test feature interactions in AVs [1], to perform efficient *online* testing [18] and to address the branch coverage of test suite generation approaches [68]. Additionally, Ben Abdessalem et al. rely on multi-objective search [21], and learnable evolutionary algorithm [17] to guide scenario generation towards *critical scenarios*. Critical scenarios have also been derived using a weighted search-based approach [22] and genetic algorithm [23]. Similarly, DeepJanus [19] combines evolutionary and novelty search to derive test inputs at the behavioral frontier of AVs, while Babikian et al. [2] use a hybrid, graph and numeric solver-based approach to concretize a limited-visibility pedestrian crossing scenario. Note that different search-based approaches may represent domain-specific constraints differently in the underlying algorithm [69] (e.g. objectives vs. hard constraints).

Existing search-based approaches are often used to guide scenario generation towards test cases with particular characteristics. Our approach can complement such existing work as discussed in Section II-D. Nevertheless, existing search-based approaches as standalone components are limited in various aspects. Such approaches are often designed for the concretization of a specific scenario type, which is selected upfront and then hard-coded into the search problem. In particular, numeric constraints are hard-coded according to (1) the specific interactions between actors and (2) the fixed underlying map structure. While our measurements clearly highlighted that the actual map location has a substantial influence on the de facto complexity of a scenario concretization problem, *all existing search-based approaches are limited to a pre-defined map location*, i.e. the underlying map is not given as input. Search-based approaches exist for map generation [30], but no actors are involved in such cases.

Sampling-based approaches have also been used for scenario generation. Certain approaches [62], [64] sample over a parametric (discrete) representation of arbitrary functional input scenarios, but they often avoid numeric (continuous) constraints (i.e. the logical scenario). O'Kelly et al. [65] use Monte Carlo sampling (over a continuous domain) to simulate scenarios with rare events, thus reasoning directly at the logical scenario level. However, all these sampling-based approaches are limited to a fixed map location.

The SCENIC framework [11] also provides sampling-based concretization, but it improves on other sampling-based approaches by handling any road map as an input parameter. Limitations of the input language and of the underlying functional-to-numeric constraint mapping are discussed in Section III-B and in Section IV-C, respectively.

Path-planning approaches [42], [63] address scenario generation directly at the level of numeric constraints. As such, they cannot handle abstract constraints as input. These approaches often use formalizations of safety requirements as guiding metrics for scenario generation. Furthermore, despite the lack of experimental results, such path-planning approaches are in principle adaptable to any road maps.

IX. CONCLUSION

In this paper, we proposed a traffic scene concretization approach that leverages metaheuristic search to place vehicles on an arbitrary road map (given as input) such that they satisfy a set of input constraints. Our approach handles traffic scenes on three different levels of abstractions in compliance with safety assurance best practices for autonomous vehicles. The input is a functional scene specification (represented in a novel scene specification language with 4-valued partial model semantics) that captures the scene concretization problem by abstract relations, and enables early detection of inconsistent specifications. Then, the functional scene specification is mapped to a complex numeric constraint satisfaction problem on the logical level. Finally, we use metaheuristic search with customizable objective functions and constraint aggregation strategies to solve the numeric problem in order to derive a concrete scene that can be investigated in the popular CARLA simulator [5].

We carried out a detailed experimental evaluation comparing eight configurations of our proposed approach over three realistic road maps to assess success rate, runtime and scalability. Our results show that despite higher runtimes, our approach provides significantly better success rate and scalability than the state-of-the-art SCENIC tool, while traversing a search space with 2^{420} states.

(Initial) scene concretization is a subproblem of the complex challenge of scenario-based testing of AVs. As such, our future work aims to integrate the handling of behaviors and dynamic constraints, potentially representable through temporal logic languages. Moreover, we plan to address the systematic synthesis of abstract functional scene specifications with certain coverage guarantees over the auto-generated specification suites.

ACKNOWLEDGMENT

The authors would like to thank (1) Kristóf Marussy for his help in formalizing the logic structures, (2) Catriona McIntosh for her help in running preliminary measurements, as well as (3) Attila Ficsor and Boqi Chen for their work in vision-based ML component testing.

REFERENCES

[1] R. B. Abdessalem, A. Panichella, S. Nejati, L. C. Briand, and T. Stifter, "Testing autonomous cars for feature interaction failures using many-objective search," in *Proc. 33rd ACM/IEEE Int. Conf. Automated Softw. Eng. (ASE)*, New York, NY, USA: ACM, 2018, vol. 18, pp. 143–154.
[2] A. A. Babikian, O. Semeráth, A. Li, K. Marussy, and D. Varró, "Automated generation of consistent models using qualitative abstractions and exploration strategies," *Softw. Syst. Model.*, vol. 21, no. 5, pp. 1763–1787, 2022.
[3] R. Alexander, H. Hawkins, and D. Rae, "Situation coverage – A coverage criterion for testing autonomous robots," Department of Computer Science, University of York, York, U.K., Technical Report YCS-2015-496, 2015.
[4] A. A. Babikian, "Automated generation of test scenario models for the system-level safety assurance of autonomous vehicles," in *Proc. 23rd ACM/IEEE Int. Conf. Model Driven Eng. Lang. Syst. (MoDELS), Companion Proc.*, New York, NY, USA: ACM, 2020, pp. 1–7.
[5] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proc. 1st Annu. Conf. Robot Learn.*, 2017, pp. 1–16.
[6] "Virtual-based safety testing for self-driving cars from NVIDIA drive sim." NVIDIA. [Online]. Available: https://www.nvidia.com/en-us/self-driving-cars/simulation/
[7] S. Sen, B. Baudry, and J. Mottu, "Automatic model generation strategies for model transformation testing," in *Proc. Theory Pract. Model Transformations, 2nd Int. Conf. (ICMT), Companion Proc.*, R. F. Paige, Ed., Berlin, Germany: Springer-Verlag, 2009, vol. 5563, pp. 148–164.
[8] G. Soltana, M. Sabetzadeh, and L. C. Briand, "Synthetic data generation for statistical testing," in *Proc. 32nd IEEE/ACM Int. Conf. Automated Softw. Eng. (ASE)*, Piscataway, NJ, USA: IEEE Press, 2017, pp. 872–882.
[9] O. Semeráth, A. S. Nagy, and D. Varró, "A graph solver for the automated generation of consistent domain-specific models," in *Proc. Int. Conf. Softw. Eng. (ICSE)*, New York, NY, USA: IEEE Computer Society, 2018, vol. 1, pp. 969–980.
[10] G. Soltana, M. Sabetzadeh, and L. C. Briand, "Practical constraint solving for generating system test data," *ACM Trans. Softw. Eng. Methodol.*, vol. 29, no. 2, pp. 1–48, 2020.
[11] D. J. Fremont, X. Yue, T. Dreossi, A. L. Sangiovanni-Vincentelli, S. Ghosh, and S. A. Seshia, "Scenic: A language for scenario specification and scene generation," in *Proc. ACM SIGPLAN Conf. Program. Lang. Des. Implementation (PLDI)*, New York, NY, USA: ACM, 2019, pp. 63–78.
[12] *Road Vehicles — Functional Safety*, ISO Standard 26262-1, 2018.
[13] *Road Vehicles — Safety of the Intended Functionality*, ISO/PAS Standard 21448, 2019.
[14] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, "Defining and substantiating the terms scene, situation, and scenario for automated driving," in *Proc. IEEE 18th Int. Conf. Intell. Transp. Syst.*, Piscataway, NJ, USA: IEEE Press, 2015, pp. 982–988.

[15] T. Menzel, G. Bagschik, and M. Maurer, "Scenarios for development, test and validation of automated vehicles," in *Proc. IEEE Intell. Vehicles Symp. (IVS)*, Piscataway, NJ, USA: IEEE Press, 2018, pp. 1821–1827.

[16] I. Majzik et al., "Towards system-level testing with coverage guarantees for autonomous vehicles," in *Proc. ACM/IEEE 22nd Int. Conf. Model Driven Eng. Lang. Syst. (MODELS)*, 2019, pp. 89–94.

[17] R. B. Abdessalem, S. Nejati, L. C. Briand, and T. Stifter, "Testing vision-based control systems using learnable evolutionary algorithms," in *Proc. Int. Conf. Softw. Eng. (ICSE)*, New York, NY, USA: IEEE Computer Society, 2018, vol. 11, pp. 1016–1026.

[18] F. U. Haq, D. Shin, and L. Briand, "Efficient online testing for DNN-enabled systems using surrogate-assisted and many-objective optimization," in *Proc. Int. Conf. Softw. Eng. (ICSE)*, 2022, pp. 811–822.

[19] V. Riccio and P. Tonella, "Model-based exploration of the frontier of behaviours for deep learning system testing," in *Proc. 28th ACM Joint Meeting Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng. (ESEC/FSE)*, 2020, pp. 876–888.

[20] G. Jahangirova, A. Stocco, and P. Tonella, "Quality metrics and oracles for autonomous vehicles testing," *Proc. 14th IEEE Int. Conf. Softw. Testing, Verification Validation (ICST)*, 2021, pp. 194–204.

[21] R. B. Abdessalem, S. Nejati, L. C. Briand, and T. Stifter, "Testing advanced driver assistance systems using multi-objective search and neural networks," *Proc. 31st IEEE/ACM Int. Conf. Automated Softw. Eng. (ASE)*, 2016, pp. 63–74.

[22] A. Calo, P. Arcaini, S. Ali, F. Hauer, and F. Ishikawa, "Generating avoidable collision scenarios for testing autonomous driving systems," in *Proc. IEEE 13th Int. Conf. Softw. Testing, Validation Verification (ICST)*, Piscataway, NJ, USA: IEEE Press, 2020, pp. 375–386.

[23] S. Wu, H. Wang, W. Yu, K. Yang, D. Cao, and F. Wang, "A new SOTIF scenario hierarchy and its critical test case generation based on potential risk assessment," in *Proc. IEEE 1st Int. Conf. Digit. Twins Parallel Intell. (DTPI)*, 2021, pp. 399–409.

[24] G. Bagschik, T. Menzel, and M. Maurer, "Ontology based scene creation for the development of automated vehicles," in *Proc. IEEE Intell. Vehicles Symp.,* Piscataway, NJ, USA: IEEE Press, 2018, pp. 1813–1820.

[25] T. Menzel, G. Bagschik, L. Isensee, A. Schomburg, and M. Maurer, "From functional to logical scenarios: Detailing a keyword-based scenario description for execution in a simulation environment," in *Proc. IEEE Intell. Vehicles Symp*., Piscataway, NJ, USA: IEEE Press, 2019, pp. 2383–2390.

[26] K. Marussy, O. Semeráth, A. A. Babikian, and D. Varró, "A specification language for consistent model generation based on partial models," *J. Object Technol.*, vol. 19, no. 3, pp. 1–22, 2020.

[27] I. Urbieta et al., "Design and implementation of an ontology for semantic labeling and testing: Automotive global ontology (AGO)," *Appl. Sci.*, vol. 11, no. 17, 2021, Art. no. 7782.

[28] M. Sagiv, T. Reps, and R. Wilhelm, "Parametric shape analysis via 3-valued logic," *ACM Trans. Program. Lang. Syst.*, vol. 24, no. 3, pp. 217–298, 2002.

[29] D. Gopan, F. DiMaio, N. Dor, T. Reps, and M. Sagiv, "Numeric domains with summarized dimensions," in *Proc. Int. Conf. Tools Algorithms Construction Anal. Syst. (TACAS)*, Springer-Verlag, 2004, vol. 2988, pp. 512–529.

[30] A. Gambi, M. Mueller, and G. Fraser, "Automatically testing self-driving cars with search-based procedural content generation," in *Proc. 28th ACM SIGSOFT Int. Symp. Softw. Testing Anal. (ISSTA)*, ACM, 2019, vol. 11, pp. 273–283.

[31] Z. Cai and Y. Wang, "A multiobjective optimization-based evolutionary algorithm for constrained optimization," *IEEE Trans. Evol. Computation*, vol. 10, no. 6, pp. 658–675, Dec. 2006.

[32] Y. Wang, Z. Cai, Y. Zhou, and W. Zeng, "An adaptive tradeoff model for constrained evolutionary optimization," *IEEE Trans. Evol. Computation*, vol. 12, no. 1, pp. 80–92, Feb. 2008.

[33] T. Ray, H. K. Singh, A. Isaacs, and W. Smith, "Infeasibility driven evolutionary algorithm for constrained optimization," in *Studies in Computational Intelligence,* vol. 198. Berlin, Germany: Springer-Verlag, 2009, pp. 145–165.

[34] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Computation*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[35] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints," *IEEE Trans. Evol. Computation*, vol. 18, no. 4, pp. 577–601, Aug. 2014.

[36] Z. Szalay, Z. Hamar, and P. Simon, "A multi-layer autonomous vehicle and simulation validation ecosystem axis: Zalazone," in *Proc. Adv. Intell. Syst. Comput.*, 2018, vol. 867, pp. 954–963.

[37] K. Deb, K. Sindhya, and T. Okabe, "Self-adaptive simulated binary crossover for real-parameter optimization," in *Proc. Genetic Evol. Computation Conf. (GECCO)*, 2007, pp. 1187–1194.

[38] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems," *SIAM J. Optim.*, vol. 8, pp. 631–657, Aug. 1998.

[39] J. Blank and K. Deb, "Pymoo: Multi-objective optimization in python," *IEEE Access*, vol. 8, pp. 89 497–89 509, 2020.

[40] F. U. Haq, D. Shin, and L. C. Briand, "Many-objective reinforcement learning for online testing of DNN-enabled systems," in *Proc. IEEE/ACM 45th Int. Conf. Softw. Eng. (ICSE)*, 2023, pp. 1814–1826.

[41] Z. Zhong, G. Kaiser, and B. Ray, "Neural network guided evolutionary fuzzing for finding traffic violations of autonomous vehicles," *IEEE Trans. Softw. Eng.*, vol. 49, no. 4, pp. 1860–1875, Apr. 2023.

[42] M. Klischat and M. Althoff, "Generating critical test scenarios for automated vehicles with evolutionary algorithms," in *Proc. IEEE Intell. Vehicles Symp. (IVS)*, Piscataway, NJ, USA: IEEE Press, 2019, pp. 2352–2358.

[43] R. A. Fisher, "Statistical methods for research workers," in *Breakthroughs in Statistics*. New York, NY, USA: Springer-Verlag, 1992, pp. 66–70.

[44] C. K. Haddock, D. Rindskopf, and W. R. Shadish, "Using odds ratios as effect sizes for meta-analysis of dichotomous data: A primer on methods and issues," *Psychological Methods*, vol. 3, no. 3, pp. 339–353, 1998.

[45] A. Arcuri and L. Briand, "A practical guide for using statistical tests to assess randomized algorithms in software engineering," in *Proc. Int. Conf. Softw. Eng. (ICSE)*, 2011, pp. 1–10.

[46] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *Ann. Math. Statist.*, vol. 18, no. 1, pp. 50–60, 1947.

[47] A. Vargha and H. D. Delaney, "A critique and improvement of the CL common language effect size statistics of McGraw and Wong," *J. Educ. Behav. Statist.*, vol. 25, no. 2, pp. 101–132, 2000.

[48] H. Katebi, K. A. Sakallah, and J. P. Marques-Silva, "Empirical study of the anatomy of modern SAT solvers," in *Lecture Notes in Computer Science*, vol. 6695. Berlin, Germany: Springer-Verlag, 2011, pp. 343–356.

[49] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "SegFormer: Simple and efficient design for semantic segmentation with transformers," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, vol. 15, pp. 12 077–12 090.

[50] Z. Zhu, M. Xu, S. Bai, T. Huang, and X. Bai, "Asymmetric non-local neural networks for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vision*, 2019, pp. 593–602.

[51] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen, and N. Sang, "BiSeNet V2: Bilateral network with guided aggregation for real-time semantic segmentation," *Int. J. Comput. Vision*, vol. 129, no. 11, pp. 3051–3068, 2020.

[52] M. Everingham et al., "The PASCAL visual object classes (VOC) challenge," *Int. J. Comput. Vision,* vol. 88, no. 2, pp. 303–338, 2009.

[53] D. Jackson, "Alloy: A lightweight object modelling notation," *ACM Trans. Softw. Eng. Methodol.*, vol. 11, no. 2, pp. 256–290, 2002.

[54] M. Steimle, T. Menzel, and M. Maurer, "Toward a consistent taxonomy for scenario-based development and test approaches for automated vehicles: A proposal for a structuring framework, a basic vocabulary, and its application," *IEEE Access*, vol. 9, pp. 147 828–147 854, 2021.

[55] F. Schuldt, A. Reschka, and M. Maurer, "A method for an efficient, systematic test case generation for advanced driver assistance systems in virtual environments," in *Automotive Systems Engineering II.* New York, NY, USA: Taylor and Francis, 2018, pp. 147–175.

[56] M. Scholtes et al., "6-layer model for a structured description and categorization of urban traffic and environment," *IEEE Access*, vol. 9, pp. 59 131–59 147, 2021.

[57] S. Geyer et al., "Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance," *IET Intell. Transp. Syst.*, vol. 8, no. 3, pp. 183–189, 2014.

[58] F. Klueck, Y. Li, M. Nica, J. Tao, and F. Wotawa, "Using ontologies for test suites generation for automated and autonomous driving functions," in *Proc. IEEE Int. Symp. Softw. Rel. Eng. Workshops (ISSREW)*, Piscataway, NJ, USA: IEEE Press, 2018, pp. 118–123.

[59] R. Queiroz, T. Berger, and K. Czarnecki, "GeoScenario: An open DSL for autonomous driving scenario representation," in *Proc. IEEE Intell. Vehicles Symp.,* Piscataway, NJ, USA: IEEE Press, 2019, pp. 287–294.

[60] B. Schütt, T. Braun, S. Otten, and E. Sax, "SceML: A graphical modeling framework for scenario-based testing of autonomous vehicles," in *Proc. 23rd ACM/IEEE Int. Conf. Model Driven Eng. Lang. Syst.,* ACM, 2020, pp. 114–120. [Online]. Available: https://doi.org/10.1145/3365438.3410933

[61] 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*, vol. 222. Cham, Switzerland: Springer-Verlag, 2017, pp. 153–163.

[62] R. Majumdar, A. Mathur, M. Pirron, L. Stegner, and D. Zufferey, "Paracosm: A test framework for autonomous driving simulations," in *Proc. Int. Conf. Fundam. Approaches Softw. Eng.,* Cham, Switzerland: Springer-Verlag, 2021, pp. 172–195.

[63] M. Althoff and S. Lutz, "Automatic generation of safety-critical test scenarios for collision avoidance of road vehicles," in *Proc. IEEE Intell. Vehicles Symp.,* Piscataway, NJ, USA: IEEE Press, 2018, pp. 1326–1333.

[64] E. Rocklage, H. Kraft, A. Karatas, and J. Seewig, "Automated scenario generation for regression testing of autonomous vehicles," in *Proc. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Piscataway, NJ, USA: IEEE Press, 2018, pp. 476–483.

[65] M. O'Kelly, J. Duchi, A. Sinha, H. Namkoong, and R. Tedrake, "Scalable end-to-end autonomous vehicle testing via rare-event simulation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 9827–9838.

[66] J. Bach, S. Otten, and E. Sax, "Model based scenario specification for development and test of automated driving functions," in *Proc. IEEE Intell. Vehicles Symp.,* Piscataway, NJ, USA: IEEE Press, 2016, pp. 1149–1155.

[67] C. A. González, M. Varmazyar, S. Nejati, and L. C. Briand, "A SysML-based methodology for model testing of cyber-physical systems," University of Luxembourg, Esch-sur-Alzette, Luxembourg, pp. 1–42, 2018.

[68] A. Panichella, F. M. Kifetew, and P. Tonella, "Reformulating branch coverage as a many-objective optimization problem," in *Proc. 8th IEEE Int. Conf. Softw. Testing, Verification Validation (ICST)*, 2015.

[69] Z. Fan, F. Yi, W. Li, J. Lu, X. Cai, and C. Wei, "A comparative study of constrained multi-objective evolutionary algorithms on constrained multi-objective optimization problems," in *Proc. IEEE Congr. Evol. Computation (CEC),* 2017, pp. 209–216.

Aren A. Babikian (Graduate Student Member, IEEE) is working toward the Ph.D. degree with the Department of Electrical and Computer Engineering, McGill University. His research focuses on using model generation techniques for the safety assurance of autonomous vehicles. He has published a related research paper at the international FASE 2020 conference and *Journal on Software and Systems Modeling* 2021 journal. He has also done related internships at NVIDIA and at AWS.

Oszkár Semeráth is an Assistant Professor with Budapest University of Technology and Economics. His research focuses on modeling technologies and logic solvers. He won IEEE/ACM Best Paper Award at MODELS conference, Young Researcher Award from the National Academy of Science, John George Kemeny Award from John von Neumann Computer Society, Amazon Research Award, and the New National Excellence Program scholarship three times.

Dániel Varró (Senior Member, IEEE) is a Full Professor with Linköping University and an Adjunct Professor with McGill University and with BME. He is a Co-Author for more than 180 scientific papers with seven Distinguished Paper Awards and three Most Influential Paper Awards. He serves on the editorial board in software and systems modeling. He served as a Program Co-Chair of MODELS 2021, SLE 2016, ICMT 2014, and FASE 2013 conferences. He is a Co-Founder of the VIATRA, an open-source model transformation framework, and the IncQuery Labs, a technology-intensive company.