Compositional and Contract-based Verification for Autonomous Driving on Road Networks

Lucas Liebenwein, Wilko Schwarting, Cristian-Ioan Vasile, Jonathan DeCastro, Javier Alonso-Mora, Sertac Karaman and Daniela Rus

Abstract Recent advances in autonomous driving have raised the problem of safety to the forefront and incentivized research into establishing safety guarantees. In this paper, we propose a safety verification framework as a *safety standard* for driving controllers with full or shared autonomy based on compositional and contract-based principles. Our framework enables us to synthesize safety guarantees over entire road networks by first building a library of locally verified models, and then composing local models together to verify the entire network. Composition is achieved using assume-guarantee contracts that are synthesized concurrently during verification. Thus, we can reuse local models within and across networks, add additional models to cover local road geometries without re-verifying the entire library, and perform all computations in a parallel and distributed way, which enables computational tractability. Furthermore, we employ *controller contracts* such that any controller satisfying them can be certified safe. We demonstrate the practical effectiveness of our framework by certifying controllers over parts of the Manhattan road network.

Keywords: Verification, Safety, Autonomous Car, Composition, Contracts

1 Introduction

The way we use and think about mobility and transportation has changed significantly in the last years due to the developments in autonomous driving. A major emerging challenge is to provide safety guarantees to gain customers' trust. Current driver-assisting technologies have mostly advisory roles with limited autonomy and decision making responsibilities [9, 25]. Vehicles with full or shared autonomy have the potential to reduce the high incidence of vehicle-related deaths (over 3000 each month [23]), most of them caused by driver error (over 70% [1]). Thus, *safety standards*, which can be used for development by companies, and certification by legislators and administrators, can be very beneficial for autonomous driving.

Lucas Liebenwein, Wilko Schwarting, Cristian-Ioan Vasile, Sertac Karaman, Daniela Rus
Massachusetts Institute of Technology, Cambridge, MA 02139
e-mail: (lucasl,wilkos,cvasile,sertac)@mit.edu,rus@csail.mit.edu

Jonathan DeCastro
Toyota Research Institute, Cambridge, MA 02139 e-mail: jonathan.decastro@tri.global

Javier Alonso-Mora
Delft University of Technology, 2628 CD, Delft, Netherlands e-mail: J.AlonsoMora@tudelft.nl

Toyota Research Institute ("TRI") provided funds to assist the authors with their research, but this article solely reflects the opinions and conclusions of its authors, and not TRI or any other Toyota entity.

Fig. 1 The network shows the Manhattan street grid. We verified a part of Manhattan consisting of ca. 130 blocks with 180 intersections and 330 straight road segments (marked black) using a library of 22 verified segments. This underlines the gain in efficiency of the compositional-based approach compared to that of a direct naive approach.

Formal verification can fill the gap of certification by providing a platform to assess safety with clear assumptions and guarantees. While simulation and testing are undoubtedly essential tools for deployment of complex systems, they lack the completeness, and therefore the guarantees, of verification, potentially missing out on rare and hard-to-characterize events. Recent studies [19] have indicated that the requirement to demonstrate safety for an autonomous car is *hundreds of millions* of miles of testing taking possibly *tens of years* to complete. To meet these proof-of-safety demands, testing and simulation, which provide very detailed insights for specific events, can be supplemented with verification frameworks, which provide insights for an entire set of events, though often less detailed. In such a sense, verification provides a way to check over an infinite number of simulated trajectories as opposed to straightforward case-based simulation at the cost of reduced model complexity.

Despite its appeal, verification can become computationally intractable for large complex systems. Informally speaking, verification is a tool, where a *system* is checked over all possible executions with respect to a *model* to ensure a *specification* [32]. Checking for safety of a given controller over a city, e.g. Fig. 1, might take years to process. Moreover, the process must be repeated for each controller we want to verify, thus heavily limiting re-usability. In this paper, we propose a method to verify a broad class of controllers for autonomous vehicles navigating in urban environments. We leverage *compositional verification*, *assume-guarantee contracts*, and *reachability* theory to verify space-time properties of controllers. Instead of performing the verification over entire road networks, we verify local road models, such as road segments and intersections, against traffic models for other cars. Concurrently, *assume-guarantee contracts*, i.e., pairs of safe entry and exit sets, are synthesized that guarantee safe traversal of road models (reaching safe exit sets) if the vehicle enters the components in the safe entry sets. A controller is certified safe over a road network if all of its local road models can be composed together using their associated contracts. An important feature of the verification framework is that it applies to all controllers that abide by a *controller contract*, i.e., a set of safety constraints. The key idea is to propagate reachable sets that guarantee safety through the contract, without explicitly considering the controller itself. The framework can be used to verify entire networks offline, or as a roll-out procedure to sequentially check routes as these are traversed.

As an example of a verification task, consider the road network in Fig. 1. Checking the entire network is intractable. We therefore decompose it into intersection and road

segments such as the ones shown in Fig. 2. Thus, the verification reduces to checking a limited number of smaller models over short time horizons. The assume-guarantee contracts are computed during verification (Fig. 2(b) and 2(e)), and are used to certify the road network through composition (Fig. 2(f)).

Contributions and Impact We propose a *safety standard* for full or shared autonomous driving controllers in the form of a verification framework based on compositional and contract-based principles. The benefits of the approach are: (1) reusability of road models across road networks, (2) extendibility of the library to additional local road models without re-verification of existing models, and (3) ability to perform computations in a parallel and distributed way.

Thus, we are able to efficiently leverage computational resources over the entire life cycle of the standard: the initial construction of the library, its use for checking road networks, and its incremental extension. Furthermore, *controller contracts* ensure that any controller abiding by them can be certified safe, regardless of whether motion primitive, sampling, or receding horizon based. For shared autonomy [30], safety is assured under human input as long as the vehicle satisfies the controller contract. The contributions of the paper are:

1. we formalize the vehicle safety verification problem;
2. we design a compositional framework that defines local road models, controller contracts, assume-guarantee contracts, and composition;
3. we develop the verification and contract-synthesis procedure, and the compositional verification algorithm;
4. we provide domain-specific methods and implementation to overcome tractability issues;
5. we demonstrate effectiveness of our approach on a case study of the Manhattan road network.

The work presented in this paper may impact: (1) car companies, providing them with a safety standard for developed controllers; (2) road administrators and mapping businesses, providing road safety certificates; and (3) legislators, policy makers and insurers, enabling them to assess and certify safety guarantees for autonomous cars.

Related works Challenges such as DARPA-sponsored competitions [7] have pushed automated driving to near real-world conditions, but the question of safety and reliability remains, preventing widespread adoption up to now.

Local motion planning: Methods to compute safe trajectories for autonomous vehicles in dynamic environments [27] include input space discretization [15, 33], rapidly exploring random trees [20], and receding horizon control [13, 31]. The latter can be applied for shared-control of highly automated vehicles [30]. These methods work well in practice, but usually compute valid and safe trajectories only up to a predefined time horizon, with no global and long term guarantees. In this work, we aim to obtain guarantees via road certificates, which are employed by the local planner.

Reachability and Safety Games: Safety can be guaranteed by avoiding sets of states that *inevitably* lead to collisions. These sets are referred to as the capture set [18], the inevitable collision states (ICS) [6, 14], the region of inevitable collision (RIC) [8], and the target set [22]. Such sets are expensive to compute, thus they are mostly applied to simplified systems, such as highway driving. Numerical reachability analysis was used to ensure safety in reach-avoid control problems [12], though only over lim-

ited domains, in symbolic abstractions of system models [35], and for overtaking maneuvers in autonomous driving [3], though for specific ones only. We use reachability analysis to concurrently solve local road model verification and contract synthesis problems to synthesize guarantees over larger domains.

Formal Verification: Verification has been employed in a variety of safety-critical domains such as aerospace [5], and automotive [24, 26, 34]. In the automotive field, prior work has focused on time-bounded behaviors and simple motion primitives rather than verification of controlled systems over complex, structured environments. Recent verification and synthesis techniques exploit composition [4, 10, 21, 28], assume-guarantee contracts [16, 29], and reachability [11]. These approaches are limited in the scope of the system and/or model considered. We leverage compositional and contract-based verification to simultaneously achieve scalability and computational tractability in the model (large networks) and the system (control system of car).

Overview In Sec. 2, we formalize the verification problem of a controller for a (semi-)autonomous vehicle. The verification framework is presented in Sec. 3, its implementation is described in Sec. 4, and a case study involving parts of the Manhattan road network in Sec. 5. Conclusions and future work are discussed in Sec. 6.

2 Problem Formulation

In this section, we introduce the safety verification problem of controllers for autonomous cars. We define models for the ego-car, traffic, road network, controller, and safety that form the context of the verification problem.

Ego-car and road network The ego-car is defined as a dynamical system $V = (\mathscr{Z}, \mathscr{R}, \mathscr{U}, f, h)$ evolving according to $\mathbf{z}_{k+1} = f(\mathbf{z}_k, \mathbf{u}_k)$, where $\mathscr{Z}$ is the state space, $\mathscr{R} \subset \mathbb{R}^2$ the workspace, $\mathscr{U}$ the control space, $\mathscr{C} \subset SE(2)$ the configuration space (pose) of the car, and $\mathbf{z}_k$, $\mathbf{p}_k = h(\mathbf{z}_k)$ and $\mathbf{q}_k = g(\mathbf{z}_k)$ are the state, location and configuration of the car at time k. Further, let $f : \mathscr{Z} \times \mathscr{U} \to \mathscr{Z}$, $h : \mathscr{Z} \to \mathscr{R}$, and $g : \mathscr{Z} \to \mathscr{C}$ be the Lipschitz continuous (invertible) dynamics, observation function, and configuration space submersion, respectively. When it is clear from the context, we denote the ego-car's state by $\mathbf{z}_k^0$ instead. The workspace $\mathscr{R}$, which is a planar compact connected region, corresponds to the roadway of the road network the car operates in, see Fig. 1. Let $\mathscr{B}(\mathbf{z}_k) \subset \mathscr{R}$ be the ego-car's footprint.

Traffic The road network associated with the ego-car is also populated by other traffic participants, e.g., pedestrians, bikes, and cars. For brevity, we only consider other cars. We denote the state of car i present in the road network $\mathscr{R}$ at time k by $\mathbf{z}_k^i$, $i \in \{1, \ldots, N(k)\}$, where $N(k)$ is the number of cars in $\mathscr{R}$ at time k. We consider a traffic model $T = (\mathscr{V}(0), \mathfrak{I}, \mathfrak{O}, S)$, where $\mathscr{V}(0) = \{V^i\}_{i=1}^{N(0)}$ is the set of vehicles present in $\mathscr{R}$ at initial time $k = 0$, $\mathfrak{I} \subset \mathscr{Z}$ and $\mathfrak{O} \subset \mathscr{Z}$ are sets of states/regions for entering and leaving the road network, and S is a scheduler that generates cars at $\mathfrak{I}$ and destroys them at $\mathfrak{O}$, see Sec. 5 for an example in the form of a hybrid system.

Controllers and driving behaviors The behaviors of all the cars are defined by controllers (feedback or open-loop). Formally, a controller for car i is a map from

Table 1 Symbols table.

V, $\mathscr{Z}$, $\mathscr{U}$, $\mathscr{R}$, $\mathscr{C}$	vehicle model and its state space, control space, workspace, configuration space
f, h, g	dynamics, observation, configuration map
$\mathbf{z}_k^i$	state of car i at time k, $i=0$ denotes the ego-car
$\mathscr{B}^i(\mathbf{z}_k^i)$, $\mathscr{O}_k$	footprint of car i at state $\mathbf{z}_k^i$, footprint of static and dynamic obstacles
$\mathscr{V}(k)$, $N(k)$	set and number of cars present in $\mathscr{R}$ at time k
T, $\mathfrak{I}$, $\mathfrak{O}$, S	traffic model, regions where cars enter and exit a road model, traffic scheduler
C^i	controller of car i
Z_k, $\widetilde{Z}_k$, $\widehat{Z}_k$	reachable, safe reachable, and backward safe reachable set of ego-car at time k
$\neg\mathscr{R}$	exterior (complement) of road network
$G=(I,R)$	topological graph of the road network
$\mathscr{M}$, m, A	the set of verified road models, road model and associated parameters
$\mathscr{S}$, $(entry, exit)$	controller contract, pair of assume-guarantee contracts

all the car states $\mathbf{z}_k^{0:N}$ to a control value $\mathbf{u}_k^i$, i.e., $C^i : \mathscr{Z}^{N+1} \to \mathscr{U}$ such that $\mathbf{z}_{k+1}^i = f^i(\mathbf{z}_k^i, C^i(\mathbf{z}_k^{0:N}))$, $\forall i \in \{0,\dots,N\}$, where f^i defines the dynamics of car i. Throughout the paper, we will tacitly assume that the other cars' models are given together with the controllers that define their behavior and are known a priori for verification.

Safety The controller for the ego-car is said to be *safe* at time k if it does not collide with environment obstacles, the road boundary, and other vehicles. Formally, the ego-car is safe at time k if $\inf_{\mathbf{z}\in\mathscr{B}(\mathbf{z}_k),\mathbf{o}\in\mathscr{O}_k} \|\mathbf{z}-\mathbf{o}\| > \pi_{safety}$, where $\mathscr{O}_k = \neg\mathscr{R} \cup \bigcup_{i=1}^{N} \mathscr{B}^i(\mathbf{z}_k^i)$ is the footprint of static and dynamic obstacles, $\neg\mathscr{R}$ is the exterior (complement) of the road network, $\mathscr{B}^i(\mathbf{z}_k^i)$ is the footprint of car i in the workspace at state $\mathbf{z}_k^i$, and $\pi_{safety} \in \mathbb{R}_{\geq 0}$ is the safety margin. Similarly, the ego-car is *safe* on $\{0,\dots,K\}$, $K \in \mathbb{N}\cup\{0,\infty\}$, if it is safe for all times $k \in \{0,\dots,K\}$. The set of constraints representing safety are hereby said to be the *controller contract* $\mathscr{S}$, see Sec. 5 for more details.

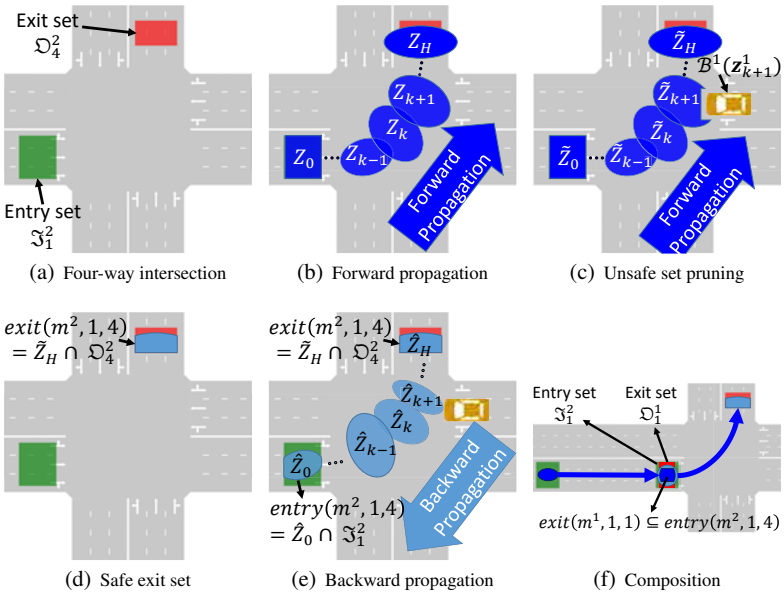
The problem that we address in this paper is checking the *safety* of executing a controller C on the ego-car with respect to given car, road network, and traffic models. The controller C is hereby represented by the controller contract $\mathscr{S}$, which it is supposed to enforce. This not only allows for an abstract representation of specific controllers, but also enables to concurrently verify a broad class of controllers.

Problem 1 (Controller safety). Given a ego-car model V operating in road network $\mathscr{R}$, using controller C that abides by the controller contract $\mathscr{S}$, under the assumption of a traffic model T, determine whether the ego-car is *safe* under control of C in time interval $\{0,\dots,K\}$ starting from some subset of the initial states $Z_0 \subseteq \mathscr{Z}$.

3 Composition-based Verification Framework

In this section, we propose a verification framework based on decomposition of the problem into smaller verification tasks corresponding to topological features of road networks, mainly road segments and intersections. Verification over entire networks is achieved by composition of models using synthesized assume-guarantee contracts.

The framework has two steps: (1) local verification of the controller contract $\mathscr{S}$ with synthesis of assume-guarantee contracts, and (2) fitting local models and composition with the assume-guarantee contracts. First, the parameterized local road models



(a) Four-way intersection (b) Forward propagation (c) Unsafe set pruning

(d) Safe exit set (e) Backward propagation (f) Composition

Fig. 2 Consider the library $\mathscr{M} = \{m^1, m^2\}$ composed of a straight road m^1 and a four-way intersection m^2. (a) The four-way intersection m^2 is shown. (b) The entry set is propagated forward, and, (c), concurrently pruned of unsafe states induced by other cars. (d) The safe exit set is the intersection of the safe reachable set at time step H, the verification horizon, and the exit set. (e) The safe entry set is computed by backward propagating the safe exit set. (f) Lastly, the composition of the models using the associated assume-guarantee contracts enables us to certify road networks.

are verified, i.e., $\mathscr{S}$ is verified, and a pair of *safe* entry and exit states is synthesized for each model, which forms the *assume-guarantee contract* associated with each model. These tasks may be performed off-line in parallel. The collection of local models is called a *library*. Second, given a road network, locally verified models from the library are fitted to the roads and intersections of the network. The assume-guarantee contracts are used to check the composition of the models based on the topology of the road network.

Library of Parameterized Models The verification process is decomposed into smaller local problems to enable computational tractability as well as facilitate parallel and distributed solutions. Moreover, we want to reuse the local computations both within and across road networks. Thus, we propose constructing a library of parameterized models that can be verified a priori and used in any road network to decide the safety of a controller. Local verification of models is valid for *any* controller that implements the controller contract $\mathscr{S}$.

Formally, each road element model is a tuple $m = (\mathscr{Z}, \mathfrak{I}, \mathfrak{O}, S, A)$, where $\mathscr{Z}$ and $\mathscr{R} = h(\mathscr{Z}) \subset \mathbb{R}^2$ are the local state space and workspace (roadway) of an ego-car, $\mathfrak{I} = \{\mathfrak{I}_j\}_{j=1}^{n_I} \subset \mathscr{Z}$ is the set of the n_I possible entry regions, $\mathfrak{O} = \{\mathfrak{O}_j\}_{j=1}^{n_O} \subset \mathscr{Z}$ is the set of the n_O possible exit regions, S is the traffic scheduler that dictates when cars are generated into $\mathscr{R}$, and A is the set of parameters associated with the model. The pa-

Algorithm 1: Verification of Road Models

Input: $m = (\mathscr{Z}, \mathfrak{I}, \mathfrak{O}, S, A)$ – road model, $\mathfrak{I}_j \in \mathfrak{I}$, $\mathfrak{O}_{j'} \in \mathfrak{O}$ – pair of entry/exit regions, H – verification horizon

Output: $entry(m, j, j')$, $exit(m, j, j')$ – pair of safe entry/exit regions (assume-guarantee contract)

1 $\widetilde{Z}_0 \leftarrow \mathfrak{I}_j$
2 $\{Z_0^i\}_{i=1}^{N(0)} \leftarrow S.init()$
3 **for** $k \in \{0, \dots, H-1\}$ **do** `// forward propagation`
4 $\quad Z_{k+1}^i \leftarrow f^i(Z_k^i, C^i(\widetilde{Z}_k, Z_k^{1:N(k)})), \forall i \in \{1, \dots, N(k)\}$
5 $\quad N(k+1) \leftarrow S.update()$
6 $\quad \mathscr{O}_{k+1} \leftarrow \neg\mathscr{R} \cup \bigcup_{i=1}^{N(k+1)} \mathscr{B}^i(Z_{k+1}^i)$
7 $\quad Z_{k+1} \leftarrow f(\widetilde{Z}_k, \mathscr{U})$
8 $\quad \widetilde{Z}_{k+1} \leftarrow Z_{k+1} \setminus \mathscr{O}_{k+1}$ `// enforcing the controller contract`
9 $exit(m, j, j') \leftarrow \widetilde{Z}_H \cap \mathfrak{O}_{j'}$
10 $\widehat{Z}_H \leftarrow exit(m, j, j')$
11 **for** $k \in \{H, \dots, 1\}$ **do** `// backward propagation`
12 $\quad \widehat{Z}_{k-1} \leftarrow f^{\dagger}(\widehat{Z}_k, \mathscr{U}) \cap \widetilde{Z}_{k-1}$
13 $entry(m, j, j') \leftarrow \widehat{Z}_0 \cap \mathfrak{I}_j$
14 **return** *entry(m, j, j'), exit(m,j,j')*

rameters of a road model can be related to its geometry, such as the width of the lanes, the angles of an intersection's branches, and the pose of the model within a global road network. Note that, depending on the parameter (e.g. lane width), the safety guarantee does not automatically hold for the entire set of parameters, implying that the verification procedure has to be conducted for a (finite) set of potential parameter values. The *library* of all available road element models is denoted by $\mathscr{M} = \{m^p\}_p$, where upper indices are used to distinguish between multiple road models if necessary. An example of a four-way intersection is shown in Fig. 2(a). Composition of road models is done such that the exit region of the current model overlaps with the entry region of the next one, see Fig. 2(f) for an example.

Verification of Controller Contract and Assume-Guarantee Contracts Verification of the local road models in the *library* is based on reachability analysis, which allows concurrent synthesis of the assume-guarantee contracts.

An assume-guarantee contract of a road model m, traversed from entry region j to exit region j', is a pair of safe entry and exit sets $(entry(m, j, j'), exit(m, j, j'))$, where $entry(m, j, j') \subseteq \mathfrak{I}_j \subset \mathscr{Z}$ and $exit(m, j, j') \subseteq \mathfrak{O}_{j'} \subset \mathscr{Z}$. The contract is interpreted as follows: if the system starts in $entry(m, j, j')$, then it is guaranteed that the controller can drive the ego-car safely to $exit(m, j, j')$ within the exit region $\mathfrak{O}_{j'}$.

The verification procedure for a road model m is shown in Alg. 1 for given entry and exit regions. Overall, all entry-exit pairs need to be verified. The algorithm has two parts. In the first part, the entry set is propagated forward over the given time horizon H, starting from the initial entry set $\mathfrak{I}_j$ (line 1). The initial state sets of the other cars are initialized by the scheduler using its $init()$ method (line 2). At each step the other vehicles' state sets are propagated using their controllers C^i (line 4). Next, the scheduler's $update()$ method is called (line 5) that spawns and removes vehicles,

Algorithm 2: Verification with Library of Verified Road Models

Input: $V = (\mathscr{X}, \mathscr{R}, \mathscr{U}, f, h)$, $\mathscr{M}$ – library of verified parameterized models
Output: Boolean value indicating *safety*

1 extract topology graph $G = (I, R)$ of road network $\mathscr{R}$
2 fit each node $\iota \in I$ (intersection) to $m^{\iota} \in \mathscr{M}$ with parameters A^{ι}
3 fit each edge $r \in R$ (road) to $m^{r} \in \mathscr{M}$ with parameters A^{r}
4 **for** $r_1 = (\iota_1, \iota_2), r_2 = (\iota_2, \iota_3) \in R$ **do**
5 **if** $\neg\big(exit(m^{r_1}, 1, 1) \subseteq entry(m^{\iota_2}, r_1, r_2) \wedge exit(m^{\iota_2}, r_1, r_2) \subseteq entry(m^{r_2}, 1, 1)\big)$ **then**
6 **return** $\bot$
7 **return** $\top$

initializes new vehicles, and returns the number of vehicles, according to the traffic model T. The unsafe states $\mathscr{O}_{k+1}$, which arise from $\mathscr{S}$, are the union over the road complement $\neg\mathscr{R}$, i.e., the exterior, and the union of possible footprints $\mathscr{B}^i(Z^i_{k+1})$ of each of the other cars (line 6). Then, the ego-car's reachable set Z_{k+1} is computed (line 7) and the unsafe states $\mathscr{O}_{k+1}$ are pruned from Z_{k+1} (line 8) to obtain the *safe* reachable set $\widetilde{Z}_{k+1}$. Note that the reachable set Z_{k+1} is computed over all possible control inputs $u \in \mathscr{U}$, since the controller is only implicitly represented via $\mathscr{S}$, i.e., the safety constraints. In other words, we check for all solutions Z_{k+1} and then prune them accordingly to obtain the possible solutions $\widetilde{Z}_{k+1}$ under $\mathscr{S}$. The safe exit set $exit(m, j, j')$ is the set of safe reachable solutions within the exit set $\mathfrak{O}_{j'}$ (line 9). Next, to compute the safe entry set for $exit(m, j, j') \subset \mathfrak{O}_{j'}$, backwards propagation is employed starting from the safe exit set (line 10). The safe exit set is backpropagated via the inverse dynamics $\mathbf{z}_{k-1} = f^{\dagger}(\mathbf{z}_k, \mathbf{u}_{k-1})$ of the ego-car and intersected with the safe forward reachable set $\widetilde{Z}_{k-1}$, since these are the only relevant solutions (line 12). The controller contract $\mathscr{S}$ is enforced at all times since $\widetilde{Z}_{k-1}$ abides by $\mathscr{S}$. The safe entry set is thus $\widehat{Z}_0$ (line 13). The necessity for this additional step arises from the fact that, although we can infer the set of initial conditions ($\mathfrak{I}_j$) of the safe reachable set $\widetilde{Z}_H$, we cannot infer the set of *safe* initial conditions ($entry(m, j, j')$) of the safe exit set $exit(m, j, j')$. A graphic representation of the procedure is shown in Fig. 2. The forward propagation procedure (line 7) is shown in 2(b), and the pruning step (line 8) is shown in 2(c). Once the safe-reachable set at step H is computed, it is trimmed (line 9) to lie within the exit region, see 2(d). The second part of the procedure, the backward propagation (line 10-13) shown in 2(e), computes the safe entry set.

Road Network Verification Given a library of verified road models $\mathscr{M}$, we can verify road networks via composition using the models' assume-guarantee contracts. The procedure is summarized in Alg. 2. First, we extract the topology graph G of network $\mathscr{R}$ (line 1), and then fit models to all intersections $\iota \in I$ (line 2) and road segments $r \in R$ (line 3) corresponding to the graph's nodes and edges. Finally, we check for each two incident road segments $r_1, r_2 \in R$ if: (a) the safe exit set $exit(m^{r_1}, 1, 1)$ of the ingoing road r_1 is included in the safe entry set $entry(m^{\iota_2}, r_1, r_2)$ of the common intersection m^{ι_2}, and (b) the safe exit set of the intersection $exit(m^{\iota_2}, r_1, r_2)$ is included in the safe entry set $entry(m^{r_2}, 1, 1)$ of the outgoing road r_2. If all checks pass, then the network is certified safe. In other words, if the safe set remains non-empty during

the propagation throughout the network, the network is certified safe. In the case that a (pairwise) composition is deemed unsafe, Alg .1 can be run for the composition.

4 Implementation

In the following, we introducce the tools for reachability analysis and set operations. Relevant details for the implementation of Alg. 1 are mentioned as well.

Reachability Tool We employ CORA [2] to compute the reachable set $Z_k \subset \mathscr{Z}$ for each time k (line 7 of Alg. 1). CORA is a reachability tool for linear, nonlinear and hybrid dynamical systems. Starting from some set $\widetilde{Z}_k$, represented as zonotope, it computes the reachable set Z_{k+1} at time $k+1$ by solving $Z_{k+1} = f(\widetilde{Z}_k, \mathbf{u} = 0)$, and, subsequently, appropriately enlarging Z_{k+1} to account for the variable control input $\mathbf{u} \in \mathscr{U}$. Nonlinear systems, such as the dynamics model used in our case study in Sec. 5, are abstracted to polynomial systems and the abstraction error is accounted for through an additional enlargement of Z_{k+1} to obtain an overapproximation.

Set Representation The road segments $\mathscr{R}$ and the collision constraints $\mathscr{O}$ are described via polytopes. Since polytopes, as opposed to zonotopes, are closed under intersection, they can be used to prune unsafe solutions from the reachable set. We use the MPT toolbox [17] for polytope operations. To be able to convert sets back between zonotopes (for reachability) and polytopes (for other purposes) representation, we use CORA to compute an overapproximation, i.e., an encompassing convex hull of the polytope. Note that non-convex regions are stored as an array of convex regions. Thus, any error caused by the conversion can be neglected.

Set Pruning As expressed in Alg. 1, we check for unsafe states and prune solutions that are in collision with either road boundaries or other cars, which generally yields non-convex sets (line 8). These operations are performed using polytopic representations. Note that any non-convex safe reachable set $\widetilde{Z}_k = \bigcup_{\ell=1}^{L_k} \widetilde{Z}_{k,\ell}$ is segmented into L_k convex regions $\widetilde{Z}_{k,\ell}$ and stored accordingly. This is crucial as CORA requires convex input sets. The next reachable set is computed as $Z_{k+1} = \bigcup_{\ell=1}^{L_k} f(\widetilde{Z}_{k,\ell}, \mathscr{U})$ applying CORA separately to each convex region $\widetilde{Z}_{k,\ell}$, see Fig. 5 for an example.

Reduction of Complexity Representing sets as a union of convex sets, however, leads to an exponential growth in the number of convex segments L_k, because: (1) each pruning $Z_{k,\ell} \setminus \mathscr{O}_k$ can lead to a split into more convex regions, and (2) the reachability tool splits large partial sets $\widetilde{Z}_{k,\ell}$ into further subregions to minimize the error associated with the underlying approximation procedure. Exponential growth in L_k will inevitably yield exponential growth in runtime as we must compute the reachable set individually for each of the L_k subsets. Thus, this calls for an efficient and effective reduction method in order for the procedure to maintain computational tractability.

Due to the constraint checks during each time k, many subsets $\widetilde{Z}_{k,\ell}$ are reduced to a negligible size, e.g. subsets that overlap the roadway, i.e., $Z_{k,\ell} \cap \neg\mathscr{R} \neq \emptyset$. This can be exploited by omitting any subsets for which $\mu(\widetilde{Z}_{k,\ell}) \ll \max_\ell \mu(\widetilde{Z}_{k,\ell})$ at each time k (1). Moreover, neighboring subsets $\widetilde{Z}_{k,\ell}$, $\widetilde{Z}_{k,\ell'}$ that will be propagated separately,

are likely to overlap at time $k+1$, i.e., $\widetilde{Z}_{k+1,\ell} \cap \widetilde{Z}_{k+1,\ell'} \neq \emptyset$. A simple iteration over all possible pairwise combinations of sets can be used to check if the one set is a subset of the other; If so, that set may be removed from the list. The downside of this approach is that it runs in $O(L_k^2)$ time. Since we may expect a large reduction in the number of sets, a *divide and conquer* approach is applied, where we recursively divide the overall set $\{\widetilde{Z}_{k,\ell}\}_\ell$ to perform pairwise checks on smaller sets of subsets (2).

Our empirical results show that the runtime can be significantly reduced (up to 99%) when methods (1) and (2) are applied at the end of each time step. The same principles are applied during backpropagation to maintain a tractable representation of the relevant sets.

We also note that the runtime is dependent on the number of vehicles, since each additional obstacle increases the complexity of $\mathscr{O}_k$. Empirically, we found that up to 5-7 vehicles can be added before noticing significant increases in the runtime.

5 Case Study

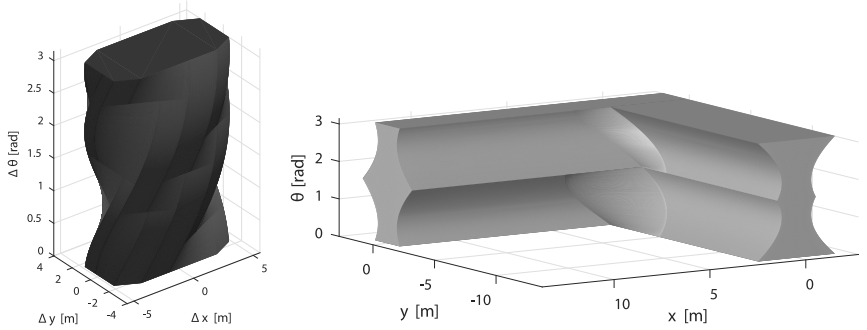
In this section, we will instantiate the verification framework to verify a specific receding horizon controller, also referred to as model predictive controller (MPC) that abides by the controller contract $\mathscr{S}$. In the following, we specify the dynamic motion model and the vehicle's dynamic limitations, as well as, the collision constraints from other vehicles and the drivable space limitations are formulated, resulting in the controller contract $\mathscr{S}$. At the end of the section, the Manhattan road library and traffic model, which are to be verified in the case study, together with the results, are presented. In the following, the Minkowski sum is denoted by $\oplus$.

Dynamic Motion Model and Dynamic Constraints We follow the nonlinear MPC formulation in [30] and employ a car model with a fixed rear wheel and a steerable front wheel with state $\mathbf{z}$ and controls $\mathbf{u}$. At time k, we denote the state of the ego-vehicle, typically position $\mathbf{p}_k = [x_k, y_k] \in \mathscr{R} \subseteq \mathbb{R}^2$, linear velocity v_k, orientation θ_k and steering angle δ_k, by $\mathbf{z}_k = [\mathbf{p}_k, \theta_k, \delta_k, v_k] \in \mathscr{Z} \subseteq \mathbb{R}^5$, and the configuration by $\mathbf{q}_k = [\mathbf{p}_k, \theta_k] \in \mathscr{C} \subseteq SE(2)$. Its control input, typically steering velocity $\dot{\delta}_k$ and acceleration a_k, is labeled $\mathbf{u}_k = [u_k^\delta, u_k^a] \in \mathscr{U} \subset \mathbb{R}^2$. The rear-wheel driven vehicle with inter-axle distance L and continuous kinematic model

$$\dot{\mathbf{z}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\delta} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \frac{v}{L}\tan(\delta) \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \underbrace{\begin{bmatrix} u^\delta \\ u^a \end{bmatrix}}_{\mathbf{u}}, \tag{1}$$

is described by a discrete time model by integration $\mathbf{z}_{k+1} = \mathbf{z}_k + \int_{k\Delta t}^{(k+1)\Delta t} \dot{\mathbf{z}}\,\mathrm{d}t = f(\mathbf{z}_k, \mathbf{u}_k)$, where Δt is the sampling period.

We limit the steering angle, $|\delta| \leq \delta_{\max}$, steering speed, $|u^\delta| \leq \dot{\delta}_{\max}$, longitudinal speed, $|v| \leq v_{\max}$, breaking and accelerations $a_{\min} \leq u^a \leq a_{\max}$, such that they conform to the dynamical limitations and the rules of the road. The yaw-rate is limited to $|\dot{\theta}| \leq \dot{\theta}_{\max}$, allowing to neglect slip. The modification is in line with our main goal: driver safety. While our choice of motion model considers a more conservative yaw-



(a) Car-collision configurations (b) Drivable space

Fig. 3 (a) The exact Minkowski swept volume between the ego-car and another vehicle, which represents configurations in collision, is shown. (b) Drivable space in a four-way intersection left-turn scenario, obtained by taking the Minkowski sum between the rectangular ego-car and the road over $\Delta\mathbf{q}$, i.e., $\mathfrak{D}(SE(2))$. This is used to trim the admissible state space. Because of symmetry and for brevity θ is only shown for $[0,\pi]$. Approximations to 10 θ-slices are used in the implementation to reduce computational complexity.

rate constraint, the verification framework allows for straight-forward integration of more advanced motion models including slip and load-transfers. Uncertainty in the dynamical model may be accounted for through proper enlargement of the reachable set.

Other vehicles In the following, we will derive the representation of the safety constraints with respect to the other vehicles. To ensure real-time operation, motion planners frequently approximate their own or other cars' footprint $\mathscr{B}^i(\mathbf{z}_k^i)$ by simpler geometries such as rotation invariant bounding boxes, enclosing ellipses, or polygons. For the reachability analysis, we approximate the shape of other vehicles by a polygon, enclosing the ellipse used by the MPC of [30]. Note that accounting for the ego-car's shape in the reachable set can become intractable due to the non-convex, disjoint nature of the set. We propose an approach, where we instead compute the Minkowski sum of the other vehicle's polygon and the ego vehicle's rectangular shape for each possible difference in configuration $\Delta\mathbf{q}_k^i = \mathbf{q}_k - \mathbf{q}_k^i$ to form a single representation of the collision region $\mathfrak{C}_k^i(\mathbf{q}_k) = \mathscr{B}^i(\mathbf{z}_k^i) \oplus \mathscr{B}(\mathbf{z}_k), \mathfrak{C}_k^i(\mathbf{q}_k) \subset \mathscr{R}$. The resulting volume $\mathfrak{C}_k^i(SE(2)) \subset \mathscr{C}$ can be represented as a single, invariant shape in the $\Delta\mathbf{q}$-space, where $\Delta\mathbf{q} = \mathbf{q}^i - \mathbf{q}$, see Fig. 3(a). This volume is translated and rotated according to each of the other vehicles' poses $\mathbf{q}_k^i$ to obtain the actual constraint in the configuration space of the ego-car. To reduce the computational complexity in the implementation, a coarser overapproximation is chosen instead.

Drivable space We take a similar approach to obtain the drivable space, where we compute the Minkowski sum between the ego-car's footprint, and the non-road surface over all configurations $\mathbf{q}$ of the ego-car. The complement, i.e., the drivable space $\mathfrak{D}(\mathbf{q}) = \neg(\neg\mathscr{R} \oplus \mathscr{B}(\mathbf{q}))$, can then be lifted into a configuration volume $\mathfrak{D}(SE(2))$ containing the allowed ego-configurations, see Fig. 3(b). An underapproximation with fewer θ-slices is used in the implementation to reduce the computational effort.

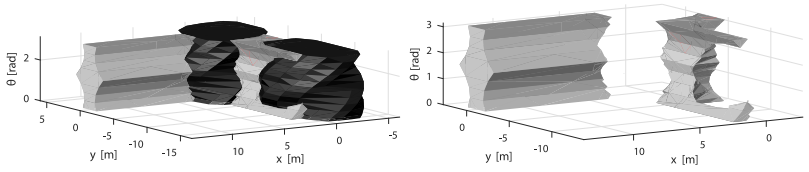


Fig. 4 The dynamic obstacles (black) and the drivable space (grey) are shown for time k on the left. Subtraction yields the admissible configuration volume $\mathfrak{A}_k$ (right).

Admissible configurations An intuitive visualization of the final admissible configuration volume $\mathfrak{A}_k \subset \mathscr{C}$ can be obtained by subtracting the other vehicle's swept volume from the drivable configuration volume: $\mathfrak{A}_k = \mathfrak{D}(SE(2)) \setminus \bigcup_{i \in \{1,\dots,N(k)\}} \mathfrak{C}_k^i(SE(2))$, see Fig. 4. After lifting the admissible configuration volume into the state space $\mathscr{X}$, we now have the unsafe states $\mathscr{O}_k = g^{-1}(\neg\mathfrak{A}_k)$ in a simple form, which enables us to efficiently compute the collision constraints once and enforce them by simply applying a set difference operation to the forward propagated state set Z_k, as shown in Alg. 1, line 8.

Controller Contract $\mathscr{S}$ We summarize the specific instantiation of the controller contract $\mathscr{S}$ to be verified below:

- Safety (obstacle avoidance): $\mathscr{O}_k \cap \widetilde{Z}_k = \emptyset, \forall k$,
- Speed limit (traffic rules): $|v| \leq v_{\max}$,
- Dynamic limits: $|\delta| \leq \delta_{\max}$, $|u^\delta| \leq \dot{\delta}_{\max}$, $a_{\min} \leq u^a \leq a_{\max}$.

Road Library and Network We showcase the framework for a Manhattan-style library consisting of 17 intersections and 5 straight road segments. Roads can have one or more lanes and incoming roads into an intersection may be tilted with respect to each other, see Fig. 7 for a selection of verified models. The library is used to verify the network shown in Fig. 1, which consists of ca. 130 blocks in Mid-Manhattan with ca. 180 intersections and ca. 330 straight road segments.

Traffic Model We describe other traffic participants as a hybrid system, i.e., a set of traffic flows. Each traffic flow is specified by a predefined path, including possible lane changes, connecting regions for entering ($\mathfrak{I}_j$) and leaving ($\mathfrak{O}_{j'}$) road model m as well as a velocity profile, together forming a trajectory. The traffic scheduler S spawns vehicles at some region $\mathfrak{I}_j$ with a fixed frequency ($0.1s^{-1}$ to $0.4s^{-1}$) [1] and removes them once they reach $\mathfrak{O}_{j'}$. Trajectories are defined as arc-length parametrized, continuously differentiable clothoid spline paths and velocity profiles generated by cubic Hermite spline interpolation. For simplicity, we restrict the class of controllers associated with the other traffic cars to open-loop.

Results The proposed procedure was successfully applied to the aforementioned library, including the four-way intersection m in Fig. 5, where the verified left-turn maneuver is shown. The considered time horizon is $6.0s$ with a discrete timestep $\Delta t = 0.05s$ resulting in 120 iterations. In the shown example there are two streams of cars occupying the intersection, both of which the ego-car avoids. When the second

[1] This range covers a large variety of situations, from occasional vehicles to dense car streams.

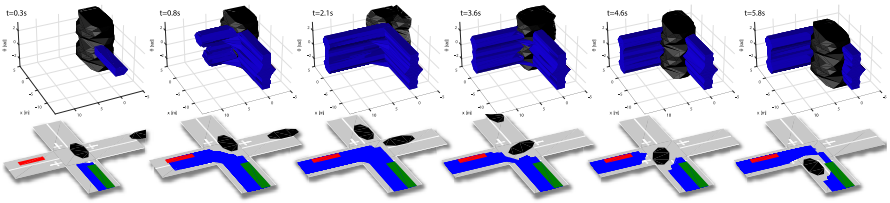


Fig. 5 The forward propagation of the reachable set is shown for a left turning maneuver. Blue sets indicate the safely reachable configuration set $g(\widetilde{Z}_k)$ (top row), respectively the position set $h(\widetilde{Z}_k)$ (bottom row) of the ego-car for various times k. Black sets mark the swept volume $\mathfrak{C}_k^i(SE(2))$ (top row), respectively the footprint $\mathscr{B}^i(\mathbf{z}_k^i)$ (bottom row) of other traffic participants. The entry and exit sets are shown in green and red, respectively. Note how the ego-car maintains a safe distance to the other cars and the road boundaries at all times.

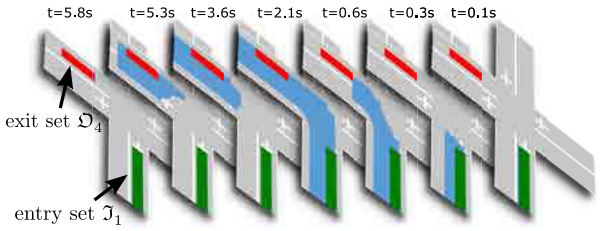


Fig. 6 Backward propagation on the four-way intersection m for various times k. At $t = 6.0s$, we start out at $\widehat{Z}_H = exit(m,1,4) = \mathfrak{O}_4$ and compute the backward reachable set $\widehat{Z}_k$ (marked blue) for each time k to obtain the safe entry set $entry(m,1,4)$.

car crosses the intersection all states in collision are pruned causing the reachable set to become disjoint ($t = 3.6s, t = 4.6s$). However, as the ego-car controller has to abide by the controller contract $\mathscr{S}$ safety remains verified. Upon the exit of the second car, the safe reachable set is expanding again reflecting the fact that the intersection has become available ($t = 5.8s$).

As visible from Fig. 5 the ego-car reaches the entire exit set, i.e., $exit(m,1,4) = \widetilde{Z}_H \cap \mathfrak{O}_4 = \mathfrak{O}_4$. Next, the safe entry set $entry(m,1,4)$ is computed by means of backward propagation, see Alg. 1 for details. The results of the backward progagation are shown in Fig. 6.

The remaining library was verified analogously. Note that for intersections, where there are multiple entry and exit regions, all combinations must be tested in order for the model to be deemed safe. This results in a total of 83 experiments for the 22 road models considered. In Fig. 7, a selection of models with various geometries is shown together with the reachable set and traffic for some time step of the verification. Consequently, the library was matched with the topology graph of Mid-Manhattan, see Fig. 1, and the compositions were tested for safety according to Alg. 2. All compositions have been deemed *safe*. Henceforth, any car, abiding by the controller contract and the assume-guarantee contracts, can safely transit through the network under the assumption of the used traffic model. [2]

In Fig. 8, average computation times are shown for one iteration. We observe that the largest cost comes from the reachability analysis itself. We also observe large variations in the computation times, arising from the varying level of complexity, i.e., level of fragmentation of the reachable sets. The backward propagation shows a sig-

[2] Further details of the case study and the library may be found in the supplementary material.

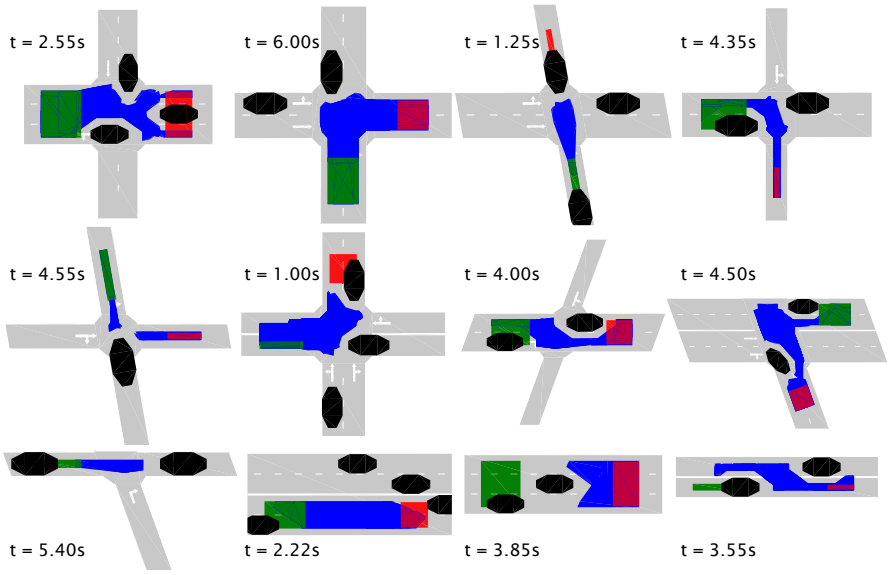
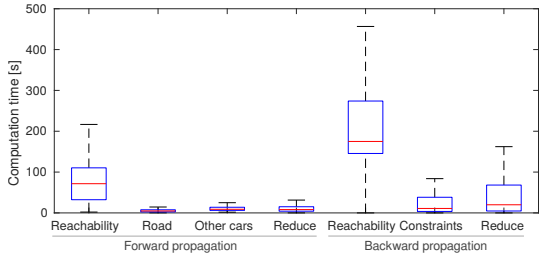


Fig. 7 A selection of verified road models, comprised of various intersections and straight roads, is shown together with the reachable set (blue) and other traffic participants (black) at the indicated timestep. The initial and final set are marked green, respectively, red.

Fig. 8 The box plot indicates computation times for various parts of one iteration with a fixed timestep of $\Delta t = 0.05s$ averaged over all conducted experiments.



nificantly higher computational demand due to the generally more fragmented sets caused by the computationally more intense constraints. We conducted our experiments on an Intel Xeon E5-2680 2.8GHz 16 Core CPU with each experiment running on a separate core. On average one experiment took 21 hours.

6 Conclusions

We studied the problem of safety verification of controllers for autonomous vehicles and proposed a novel framework for synthesizing safety guarantees for entire road networks building upon compositional verification and assume-guarantee contracts. Our framework is based on verifying a library of local road models against given ego-car and traffic models, concurrently with synthesizing assume-guarantee contracts used for composition. The library can then be used to certify the safety of executing

ego-car controllers satisfying a controller contract over road networks. We further demonstrated the effectiveness of our approach on a case study involving a library of local road models, which enabled us to verify a substantial part of Mid-Manhattan.

Avenues for future work include extending the library of locally verified road models to capture a wider range of road geometries. We plan to use the extended library to certify larger, more complex road networks. At the traffic level, we want to extend the framework to handle more realistic behaviors of other vehicles and traffic situations, as well as relax the assumptions about the knowledge of other traffic participants. Finally, we are interested in complementing the verification framework with a roll-out strategy for controllers that leverages the local road models' safety certificates. The goal is to provide online constraints that ensure the long-term safety of the ego-car.

References

[1] NHTSA traffic safety facts. https://crashstats.nhtsa.dot.gov/ (2015)
[2] Althoff, M.: An Introduction to CORA. In: ARCH, CPSWeek (2015)
[3] Althoff, M., Dolan, J.M.: Online Verification of Automated Road Vehicles Using Reachability Analysis. IEEE T-RO **30**(4) (2014)
[4] Alur, R., Moarref, S., Topcu, U.: Compositional synthesis with parametric reactive controllers. In: ACM HSCC (2016)
[5] Balachandran, S., Ozay, N., Atkins, E.M.: Verification Guided Refinement of Flight Safety Assessment and Management System for Takeoff. Journal of Aerospace Information Systems **13** (2016)
[6] Bautin, A., Martinez-Gomez, L., Fraichard, T.: Inevitable Collision States: A probabilistic perspective. In: IEEE ICRA (2010)
[7] Buehler, M., Iagnemma, K., Singh, S.: The DARPA Urban Challenge: Autonomous Vehicles in City Traffic, vol. 56. Springer (2009)
[8] Chan, N., Kuffner, J., Zucker, M.: Improved Motion Planning Speed and Safety using Regions of Inevitable Collision. In: 17th CISM-IFToMM Symposium on Robot Design, Dynamics, and Control (2008)
[9] Dagan, E., Mano, O., Stein, G.P., Shashua, A.: Forward Collision Warning with a Single Camera. In: IEEE IV (2004)
[10] Dallal, E., Tabuada, P.: Decomposing controller synthesis for safety specifications. In: IEEE CDC (2016)
[11] DeCastro, J.A., Kress-Gazit, H.: Synthesis of Nonlinear Continuous Controllers for Verifiably-Correct High-Level, Reactive Behaviors. IJRR **34**(3) (2015)
[12] Ding, J., et al.: Hybrid Systems in Robotics: Toward Reachability-Based Controller Design. IEEE Robotics & Automation Magazine **18**(3) (2011)
[13] Erlien, S.M., Fujita, S., Gerdes, J.C.: Shared steering control using safe envelopes for obstacle avoidance and vehicle stability. IEEE T-ITS **17**(2) (2016)
[14] Fraichard, T., Asama, H.: Inevitable collision states. A step towards safer robots? In: IEEE/RSJ IROS (2003)
[15] Furgale, P., et al.: Toward automated driving in cities using close-to-market sensors: An overview of the V-Charge Project. In: IEEE IV (2013)
[16] Henzinger, T.A., Qadeer, S., Rajamani, S.K.: You assume, we guarantee: Methodology and case studies. Springer (1998)

[17] Herceg, M., Kvasnica, M., Jones, C., Morari, M.: Multi-Parametric Toolbox 3.0. In: Proc. of the European Control Conference (2013). http://control.ee.ethz.ch/~mpt
[18] Hoehener, D., Huang, G., Vecchio, D.D.: Design of a lane departure driver-assist system under safety specifications. In: IEEE CDC (2016)
[19] Kalra, N., Paddock, S.M.: Driving to Safety: How Many Miles of Driving Would It Take to Demonstrate Autonomous Vehicle Reliability? Transportation Research Part A: Policy and Practice **94** (2016)
[20] Karaman, S., Frazzoli, E.: Optimal kinodynamic motion planning using incremental sampling-based methods. In: IEEE CDC (2010)
[21] Kim, E.S., Arcak, M., Seshia, S.A.: Compositional controller synthesis for vehicular traffic networks. In: IEEE CDC (2015)
[22] Mitchell, I.M., Bayen, A.M., Tomlin, C.J.: A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games. IEEE Transactions on Automatic Control **50**(7) (2005)
[23] National Safety Council: NSC motor vehicle fatality estimates 2012-2015
[24] Nilsson, P., et al.: Correct-by-Construction Adaptive Cruise Control: Two Approaches. IEEE Transactions on Control Systems Technology **24**(4) (2016)
[25] Nohmi, M., Fujikura, N., Ueda, C., Toyota, E.: Automatic braking or acceleration control system for a vehicle (1978). US Patent 4,066,230
[26] O'Kelly, M., et al.: APEX: A Tool for Autonomous Vehicle Plan Verification and Execution. In: SAE World Congress and Exhibition (2016)
[27] Paden, B., Cap, M., Yong, S.Z., Yershov, D., Frazzoli, E.: A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles. IEEE T-IV **1**(1) (2016)
[28] Rungger, M., Zamani, M.: Compositional Construction of Approximate Abstractions of Interconnected Control Systems. IEEE Transactions on Control of Network Systems (2016)
[29] Sadraddini, S., Rudan, J., Belta, C.: Formal Synthesis of Distributed Optimal Traffic Control Policies. In: Intl Conf on Cyber-Physical Systems (2017)
[30] Schwarting, W., et al.: Parallel Autonomy in Automated Vehicles: Safe Motion Generation with Minimal Intervention. In: IEEE ICRA (2017)
[31] Shia, V.A., Gao, Y., Vasudevan, R., Campbell, K.D., Lin, T., Borrelli, F., Bajcsy, R.: Semiautonomous vehicular control using driver modeling. IEEE T-ITS **15**(6) (2014)
[32] Tabuada, P.: Verification and control of hybrid systems: a symbolic approach. Springer Science & Business Media (2009)
[33] Urmson, C., et al.: Autonomous driving in urban environments: Boss and the Urban Challenge. Journal of Field Robotics **25**(8) (2008)
[34] Wongpiromsarn, T., Topcu, U., Murray, R.: Receding Horizon Temporal Logic Planning. IEEE Transactions on Automatic Control **57** (2012)
[35] Zamani, M., Pola, G., Mazo, M., Tabuada, P.: Symbolic models for nonlinear control systems without stability assumptions. IEEE Transactions on Automatic Control **57**(7) (2012)