Anytime Safety Verification of Autonomous Vehicles

Felix Gruber and Matthias Althoff

***Abstract*— We propose a procedure to formally verify the safety of autonomous vehicles online, i.e., during operation, that considers the uniqueness of each traffic situation. A challenging aspect of online verification is the varying number of surrounding traffic participants, which causes significant variations in computational demand. To guarantee timely safe motion plans, we propose an anytime approach that provides rapid conservative verification results based on coarse model abstractions, which are refined continually if computation time is available. Reachability analysis, which over-approximates all possible behaviors of other traffic participants, is performed for each abstraction. We demonstrate the usefulness of the proposed procedure using the CommonRoad benchmark suite.**

I. Introduction

Predicting the movement of other traffic participants is crucial for motion planning, threat assessment, and formal safety verification of (partially) automated vehicles [1]. Several techniques have been developed based on their intended use. Rather simple surrogate measures have been proposed to warn drivers based on predicting a single behavior of other traffic participants, e.g., time to collision [2], [3], as well as combinations of several surrogate measures [4]. In addition, collision mitigation systems that typically require short prediction horizons often rely on the prediction of a single future behavior [5], [6]. Threat assessments mainly employ stochastic predictions, either by performing Monte Carlo simulations [7], [8], which consider a finite number of future trajectories, or by predicting occupancy probability distributions [9], [10], which account for infinitely many possible behaviors. A comparison between Monte Carlo simulation and probabilistic occupancy prediction is provided in [11].

However, none of these methods can verify whether a collision is impossible under certain assumptions about the behavior of other traffic participants, e.g., assuming traffic rules are obeyed. In the concept presented in [12], the importance of set-based prediction of other traffic participants is highlighted, but no prediction algorithm to formally obtain occupancies is provided. In our previous work [13], we use reachability analysis to compute over-approximations of the occupancy of surrounding traffic participants. A similar concept has also been applied to mobile robots [14], [15]. When the trajectory of the ego-vehicle, i.e., the vehicle performing the predictions, does not intersect the over-approximative occupancies of other traffic participants at any time, we can formally guarantee that no collision occurs given the considered assumptions.

An aspect of set-based prediction that has not yet received much attention is the large dependence of computation time on the number of surrounding traffic participants. At a busy intersection in an urban area, this number can easily vary from only a few to more than 100 due to many surrounding pedestrians and cyclists. However, the computing resources of the ego-vehicle are limited. Using conventional prediction techniques requires, e.g., performing fewer simulations in a Monte Carlo simulation or simply omitting some traffic participants from the prediction. However, these measures reduce the safety of the ego-vehicle.

The situation differs for set-based techniques, which work with nondeterministic models to capture all possible behaviors; therefore, such models can be abstracted easily. The obtained abstractions result in over-approximations and thus more conservative behavior. To attenuate overly conservative results, we propose an anytime algorithm [16] for set-based safety verification of traffic participants. When computation time is available, we refine the results and reduce over-approximation. In this manner, we can formally guarantee safety while making optimal use of available computational resources.

The remainder of this paper is organized as follows. Definitions are given in Section II. In Section III, we provide an overview of our set-based safety verification procedure. The proposed anytime method is described in Section IV, and example traffic benchmarks are discussed in Section V. Conclusions and suggestions for future work are provided in Section VI.

II. Preliminaries

In this section, we introduce some general notation and the concepts of reachable and occupancy sets. Please note that throughout this paper, we assume that all safety-relevant traffic participants are detected by the sensors of the ego-vehicle.

A. Notation

Let $\mathbb{R}^n$ represent the n-dimensional Euclidean space. Given an n-dimensional vector a of any set $\mathcal{A}$ or a list a of length $|a| = n$, a_i represents its i^{th} component or element for $i \in \{1, \dots, n\}$, and $\mathcal{P}(\mathcal{A})$ denotes the power set of $\mathcal{A}$, i.e., the set of all subsets of $\mathcal{A}$. The Minkowski addition of two sets $\mathcal{A}$ and $\mathcal{B}$ is defined by $\mathcal{A} \oplus \mathcal{B} = \{a + b \mid a \in \mathcal{A},\ b \in \mathcal{B}\}$.

The set of Booleans $\mathbb{B}$ comprises two elements, i.e., true $\top$ and false $\bot$. Let $\wedge$ and $\vee$ denote logical conjunction and disjunction, respectively. The logical equality and nonequality are denoted respectively by $\equiv$ and $\not\equiv$.

The authors are with the Department of Informatics, Technical University of Munich, Boltzmannstr. 3, 85748 Garching, Germany.
Email: {felix.gruber, althoff}@tum.de
URL: https://www6.in.tum.de

The sensor measurements are updated only at discrete time steps t_k for $k \in \mathbb{Z}_{\geq 0}$. To simplify parallelization, we consider only a fixed step size, i.e., $t_{k+1} - t_k = \Delta t$, as shown in Fig. 1. Moreover, the constant receding prediction horizon is denoted by $h \in \mathbb{Z}_{>0}$, which corresponds to the number of time intervals evaluated by prediction at each initial time step t_k.

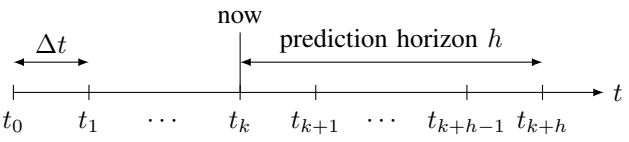


Fig. 1. Fixed receding prediction horizon h at current initial time step t_k, and constant step size Δt.

B. Reachable Set

Typically, an exact mathematical model M^{exact} of another traffic participant is not known by the ego-vehicle unless transmitted via vehicle-to-vehicle communication. Therefore, we use nondeterministic dynamic models that capture real physical behavior. All considered models of different complexity for another traffic participant are contained in a list M of length $|M|$.

The nominal behavior of model M_i for $i \in \{1, \dots, |M|\}$ is given by the following ordinary differential equation:

$$\dot{x}^{(i)}(t) = f^{(i)}\big(x^{(i)}(t), u^{(i)}(t)\big), \tag{1}$$

where $x^{(i)}(t) \in \mathbb{R}^{n_x^{(i)}}$ and $u^{(i)}(t) \in \mathbb{R}^{n_u^{(i)}}$ denote the state and input, respectively. The inputs, e.g., steering rate and acceleration, are uncertain but bounded by the set $\mathcal{U}^{(i)} \subset \mathbb{R}^{n_u^{(i)}}$ for all times, i.e., $\forall t\colon u^{(i)}(t) \in \mathcal{U}^{(i)}$ which is denoted by $u^{(i)}(\cdot) \in \mathcal{U}^{(i)}$. Based on new sensor measurements, the uncertain initial state set $\mathcal{X}_0^{(i)}(t_k) \subset \mathbb{R}^{n_x^{(i)}}$ is updated at each time step t_k, and the corresponding initial state is $x_0^{(i)}(t_k) \in \mathcal{X}_0^{(i)}(t_k)$. The solution of (1) beginning from initial time step t_k is denoted by $\xi^{(i)}\big(t, x_0^{(i)}(t_k), u^{(i)}(\cdot)\big)$.

The exact reachable set, i.e., the set of states $x^{(i)}$ that are reachable, based on model M_i, initial time step t_k, and prediction time interval $[t_{k+j-1},\ t_{k+j}]$ is

$$\begin{aligned}\mathcal{R}^e(M_i, t_k, t_{k+j}) = \big\{&\xi^{(i)}\big(t, x_0^{(i)}(t_k), u^{(i)}(\cdot)\big) \mid \\ &t \in [t_{k+j-1},\ t_{k+j}], \\ &x_0^{(i)}(t_k) \in \mathcal{X}_0^{(i)}(t_k), \\ &u^{(i)}(\cdot) \in \mathcal{U}^{(i)}\big\},\end{aligned} \tag{2}$$

where $i \in \{1, \dots, |M|\}$ and $j \in \{1, \dots, h\}$. Typically, (2) cannot be computed exactly [17]. Thus, we use over-approximations $\mathcal{R} \supseteq \mathcal{R}^e$ and want $\mathcal{R}$ to enclose $\mathcal{R}^e$ as tightly as possible. In the following, we assume that tight over-approximations are provided for all models.

Model M_i is called an abstraction of the unknown model M^{exact} of another traffic participant if the exact reachable set of M_i over-approximates the set of M^{exact}, i.e., $\forall j, \forall k\colon \mathcal{R}^e(M^{\text{exact}}, t_k, t_{k+j}) \subseteq \mathcal{R}^e(M_i, t_k, t_{k+j})$. In this paper, we assume all models M_i are abstractions, i.e., conformant with the real physical system. If this assumption is invalid, more uncertainty must be added to the nondeterministic model abstractions, as done in reachset conformance [18], [19].

C. Occupancy Set

We introduce the mapping

$$\Pi\big(x^{(i)}\big) : \mathbb{R}^{n_x^{(i)}} \to \mathcal{P}\big(\mathbb{R}^2\big),$$

which projects the state $x^{(i)}$ of a traffic participant to its set of occupied X-Y-positions. For a given set $\mathcal{A}$, the projection is applied element-wise, i.e., $\Pi(\mathcal{A}) = \{\Pi(a) \mid a \in \mathcal{A}\}$. The predicted occupancy set of another traffic participant based on model M_i, initial time step t_k, and prediction interval $[t_{k+j-1},\ t_{k+j}]$ is denoted by

$$\mathcal{O}_j^k(M_i) = \Pi\big(\mathcal{R}(M_i, t_k, t_{k+j})\big), \tag{3}$$

i.e., given by projecting the reachable set $\mathcal{R}$ to the two-dimensional set of occupied X-Y-positions. The relation

$$\mathcal{O}_j^k(M^{\text{exact}}) \subseteq \bigcap_{i=1}^{|M|} \mathcal{O}_j^k(M_i) \tag{4}$$

allows us to predict the occupancy set of another traffic participant efficiently by intersecting the occupancies of $|M|$ different abstractions [20, Prop. 5.1]. Thus, the over-approximation becomes tighter each time a new model is added.

Similar to (3), the occupancy set of the ego-vehicle based on the known reference trajectory at time step t_k and prediction interval $[t_{k+j-1},\ t_{k+j}]$ is denoted by $\mathcal{E}_j^k$. The uncertainties due to a non-perfect tracking controller and the dimensions of the ego-vehicle are included in the set $\mathcal{E}_j^k$ [20].

Example 1: A very simple model M_1 can be obtained by allowing infinite acceleration and assuming maximum velocity $v_{\max}$. Under this model, it is possible to compute the projected exact reachable set $\Pi(\mathcal{R}^e)$ of a point mass as a circular disk with radius $r = (t_{k+j} - t_k)v_{\max}$ corresponding to the prediction interval $[t_{k+j-1},\ t_{k+j}]$. A simple over-approximation $\Pi(\mathcal{R})$ is a square with length $2r$. Finally, to obtain the occupancy set of the considered traffic participant, the vehicle dimensions must be added via Minkowski addition. □

III. Safety Verification of Autonomous Vehicles

In this section, we provide an overview of our formal safety verification procedure that uses set-based prediction of other traffic participants [13], [21]. In addition, the concept of fail-safe motion planning is presented [22].

A. Set-based Formal Verification

Our formal verification method in Algorithm 1 is executed in parallel for each surrounding traffic participant and has two return values. The first output of Algorithm 1 is Boolean true $\top$ if there exists a possible collision for the ego-vehicle with the considered traffic participant, otherwise Boolean

false $\bot$. To this end, we introduce the function any, which returns $\top$ if any element of the considered input vector $c \in \mathbb{B}^h$ is $\top$, otherwise $\bot$. At time step t_k, the list of occupancies of another traffic participant and the ego-vehicle for the prediction horizon h are denoted by $\mathcal{O}^k = \left[\mathcal{O}_1^k, \mathcal{O}_2^k, \ldots, \mathcal{O}_h^k\right]$ and $\mathcal{E}^k = \left[\mathcal{E}_1^k, \mathcal{E}_2^k, \ldots, \mathcal{E}_h^k\right]$, respectively. The list $\mathcal{O}^k$ is the second output of Algorithm 1.

Algorithm 1 Standard Safety Verification

1: **function** standardVerification$(\mathcal{E}^k, M, k, h)$
2: updateParameters( )
3: **for all** $j \in \{1, \ldots, h\}$ **do**
4: $\mathcal{O}_j^k \leftarrow \bigcap_{i=1}^{|M|} \mathcal{O}_j^k(M_i)$
5: $c_j \leftarrow$ checkCollision$\left(\mathcal{O}_j^k, \mathcal{E}_j^k\right)$
6: **return** any(c), $\mathcal{O}^k$

Throughout this paper, we use the following three model abstractions for other traffic participants:

- an infinite-acceleration-based model M_1 (Section II-C);
- a finite-acceleration-based model M_2 [13]; and
- a lane-following model M_3 [13].

Although $\mathcal{O}_j^k(M_2) \subseteq \mathcal{O}_j^k(M_1)$ holds for arbitrary k and j, we do not require the occupancy set of a model to be the subset of another one or vice versa, e.g., $\mathcal{O}_j^k(M_2) \nsubseteq \mathcal{O}_j^k(M_3)$ and $\mathcal{O}_j^k(M_3) \nsubseteq \mathcal{O}_j^k(M_2)$ generally hold.

The parameters of these models are primarily based on traffic rules and physical constraints. For example, M_1 and M_2 assume that another traffic participant does not exceed maximum velocity $v_{\max}$, e.g., given by an exact or relaxed speed limit. Moreover, it is checked whether the other traffic participant obeys traffic rules, such as staying in their own lane. If a violation is detected by the ego-vehicle, the corresponding parameter is adapted or removed, e.g., by increasing the individual speed limit or disabling the assumption that the other traffic participant will follow lanes in the future. Otherwise, the abstractions become nonconformant with the real system. The described parameter updating is handled by the function updateParameters, which is called first in Algorithm 1.

Second, our set-based verification procedure predicts the occupancies of other traffic participants at time step t_k for all h consecutive prediction intervals via reachability analysis, as shown in Fig. 2. In line 4 of Algorithm 1, the overall occupancy set $\mathcal{O}_j^k$ for another traffic participant at time step t_k and prediction interval $[t_{k+j-1},\ t_{k+j}]$ is computed. Based on (4), in order to reduce the over-approximation error, the occupancy sets $\mathcal{O}_j^k(M_i)$ of all $|M|$ models are intersected.

Third, collision checks are performed for each prediction interval in line 5 of Algorithm 1. The set-based method checkCollision$\left(\mathcal{O}_j^k, \mathcal{E}_j^k\right)$ returns Boolean true $\top$ if $\mathcal{O}_j^k \cap \mathcal{E}_j^k \not\equiv \emptyset$, otherwise false $\bot$. If no intersection is detected for any of the h prediction intervals, the motion plan of the ego-vehicle is formally verified as safe with respect to the considered traffic participant. Otherwise, the ego-vehicle must modify the intended trajectory or perform a fail-safe maneuver to ensure safety, as explained in the next section.

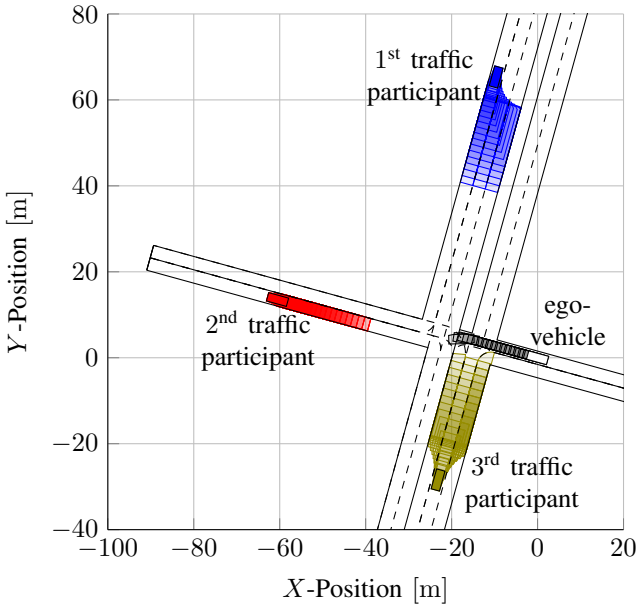


Fig. 2. Predicted future occupancy sets of all surrounding traffic participants and ego-vehicle. The time step size is $\Delta t = 0.1$ s, and the prediction horizon is $h = 17$.

B. Fail-safe Motion Planning

The consideration of all possible future behaviors increasingly restricts the solution space of the ego-vehicle's trajectory the larger the prediction horizon h is chosen. Thus, the formal set-based verification procedure in Section III-A is primarily used to verify maneuvers with short time horizons. Nevertheless, there exist non-formal long-term trajectories that are initially not safe for all parts of the maneuver while considering all possible future behaviors of the other traffic participants. However, such motion plans can become safe because uncertainty about the other traffic participants' future maneuvers is reduced significantly as time proceeds.

Thus, we use an off-the-shelf trajectory planner to compute a long-term reference motion plan based on the most likely maneuvers of the other surrounding traffic participants, as shown in Fig. 3. Our presented safety verification method is only applied to the first part of the computed motion plan and a consecutive fail-safe maneuver. If formally verified, this part of the long-term plan can be executed safely; then, the next part along the trajectory is checked for potential collisions. However, the previously verified fail-safe maneuver, e.g., realizing a sufficient distance behind another traffic participant or a standstill, is executed if the verification fails. Please note that the existence of fail-safe trajectories can be proven by induction [22]. Similar to our approach, the braking inevitable collision state concept ensures that the ego-vehicle is always in a legally safe state when a collision occurs, i.e., in a state where it is not causing a collision [14].

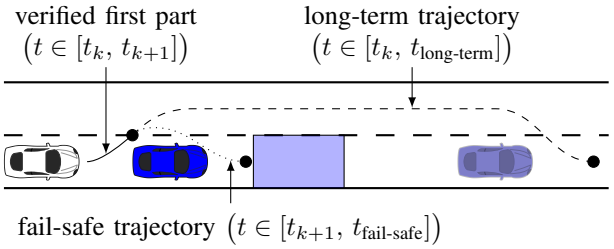


Fig. 3. Comparison of long-term and fail-safe trajectory planning of the white ego-vehicle, which wants to overtake the blue traffic participant. The predicted occupancy set of the other vehicle at $t_{\text{fail-safe}}$ and the most likely position at $t_{\text{long-term}}$ are shown by the transparent blue rectangle and vehicle, respectively.

IV. Anytime Safety Verification

In Section IV-A, we propose a novel anytime safety verification scheme that attempts to verify that the trajectory of the ego-vehicle is collision-free as quickly as possible. While previous works provide a formal concept, none of these approaches are anytime capable, i.e., the algorithm can be interrupted at any time after completing a short start-up phase, and the quality of the results improves as more computation time is available [16]. To design an efficient algorithm, we

- reuse the list of predicted occupancies $\mathcal{O}^{k-1}$ obtained at the previous time step t_{k-1} (Section IV-B),
- order the list of models M based on computational complexity and perform collision checks immediately after a new occupancy set has been computed (Section IV-C), and
- refine the predicted occupancies $\mathcal{O}_j^k$ for as long as computation time is available (Section IV-D).

A. Anytime Algorithm

Our anytime safety verification procedure is presented in Algorithm 2. It has the same inputs and outputs as Algorithm 1 with one exception, i.e., we use the occupancy list of the other traffic participant of the previous time step $\mathcal{O}^{k-1}$ for $k \in \mathbb{Z}_{>0}$ as an additional input.

First, we check in line 2 of Algorithm 2 if the updated model parameters at time step t_k have changed compared to those at t_{k-1} based on new sensor data. If altered, the occupancy sets obtained at the previous time step t_{k-1} are based on models M_i that are possibly no longer conformant with the real system. Therefore, we modify the function updateParameters compared to Algorithm 1 by adding a return value that is Boolean false $\bot$ if the model parameters have changed or $t = t_0$, otherwise true $\top$. If the return value is $\top$, our procedure reuses the list of occupancies $\mathcal{O}^{k-1}$ obtained at the previous time step t_{k-1} to quickly obtain over-approximations of $\mathcal{O}_j^k$ for $j \in \{1, \dots, h-1\}$, as described in Section IV-B. Otherwise, all h occupancies are initialized with $\mathbb{R}^2$ in line 7 of Algorithm 2.

In lines 8 to 14, we attempt to verify that no collision occurs for any prediction interval as quickly as possible. This is achieved by ordering the list of models M and performing

Algorithm 2 Anytime Safety Verification

```
1:  function anytimeVerification(E^k, M, k, h, O^{k-1})
2:      if updateParameters() ≡ ⊤ then
3:          O^k ← lazyUpdate(O^{k-1})
4:          O_h^k ← ℝ^2
5:      else
6:          for all j ∈ {1, ..., h} do
7:              O_j^k ← ℝ^2
8:      for all j ∈ {1, ..., h} do
9:          m_j ← 0
10:         c_j ← checkCollision(O_j^k, E_j^k)
11:         while (c_j ≡ ⊤) ∧ (m_j < |M|) do
12:             m_j ← m_j + 1
13:             O_j^k ← O_j^k ∩ O_j^k(M_{m_j})
14:             c_j ← checkCollision(O_j^k, E_j^k)
15:     for all j ∈ {1, ..., h} do
16:         for all i ∈ {m_j + 1, ..., |M|} do
17:             O_j^k ← O_j^k ∩ O_j^k(M_i)
18:     return any(c), O^k
```

collision checks (line 14) immediately after obtaining new sets $\mathcal{O}_j^k(M_{m_j})$, as described in Section IV-C. Similar to Algorithm 1, the Boolean collision vector $c \in \mathbb{B}^h$ stores the formal verification result for all h prediction intervals. Furthermore, the variable $m_j \in \mathbb{Z}_{\geq 0}$ for $j \in \{1, \dots, h\}$ corresponds to the number of models required to verify safety for the prediction interval $[t_{k+j-1},\ t_{k+j}]$.

If more computation time is available, the remaining abstractions are used additionally to refine the occupancies $\mathcal{O}_j^k$ in lines 15 to 17, as described in Section IV-D. Finally, our anytime method returns the safety verification result $\text{any}(c)$ in addition to the list $\mathcal{O}^k$ in line 18 of Algorithm 2.

B. Reuse of Occupancy Lists

We can quickly predict future occupancies of another traffic participant at time step t_k for $k \in \mathbb{Z}_{>0}$ by reusing the list $\mathcal{O}^{k-1}$ obtained at the previous time step t_{k-1}. As a result, we only need to compute the occupancy set $\mathcal{O}_h^k$ corresponding to the last prediction interval $[t_{k+h-1},\ t_{k+h}]$ while using elements of $\mathcal{O}^{k-1}$ as over-approximations corresponding to the other intervals, as described subsequently.

Proposition 1: At time step t_k for $k \in \mathbb{Z}_{>0}$, the relation $\mathcal{O}_{j-1}^k(M_i) \subseteq \mathcal{O}_j^{k-1}(M_i)$ holds for all models M_i and $j \in \{2, \dots, h\}$. □

Proof: This relation is valid because the considered time intervals are the same, i.e., $[t_{k+(j-1)-1},\ t_{k+(j-1)}] \equiv [t_{(k-1)+j-1},\ t_{(k-1)+j}]$. In addition, the prediction uncertainty for the identical interval is reduced after each time step because more information about the other traffic participant has been gathered. ■

As mentioned previously, the list of occupancies of another traffic participant at time step t_k is given by $\mathcal{O}^k = \left[\mathcal{O}_1^k, \mathcal{O}_2^k, \mathcal{O}_3^k, \dots, \mathcal{O}_{h-1}^k, \mathcal{O}_h^k\right]$. Then, the lazy update func-

tion, which is called in line 3 of Algorithm 2 with $\mathcal{O}^{k-1}$ as input, is defined by

$$\mathsf{lazyUpdate}(\mathcal{O}^k) = \left[\mathcal{O}_2^k, \mathcal{O}_3^k, \ldots, \mathcal{O}_{h-1}^k, \mathcal{O}_h^k, \mathcal{O}_1^k\right],$$

i.e., the method performs a circular shift.

Based on Proposition 1, by executing $\mathsf{lazyUpdate}(\mathcal{O}^{k-1})$, we quickly obtain an over-approximative result for all prediction time intervals $[t_{k+j-1},\ t_{k+j}]$ with $j \in \{1, \ldots, h-1\}$ at initial time step t_k. Therefore, only the element $\mathcal{O}_h^k$ must be computed based on new sensor measurements at t_k to obtain a valid over-approximative list $\mathcal{O}^k$.

Example 2: In the upper plot of Fig. 4, all occupancy sets at time step t_0 for $h = 3$ and the rightward moving vehicle are illustrated. Based on Proposition 1, we exploit the fact that $\mathcal{O}_1^1 \subseteq \mathcal{O}_2^0$ and $\mathcal{O}_2^1 \subseteq \mathcal{O}_3^0$ hold to quickly obtain an over-approximative result for the first two prediction intervals at the subsequent time step t_1, as shown in the lower plot of Fig. 4. Thus, only the occupancy set $\mathcal{O}_3^1$ must be computed at time step t_1. □

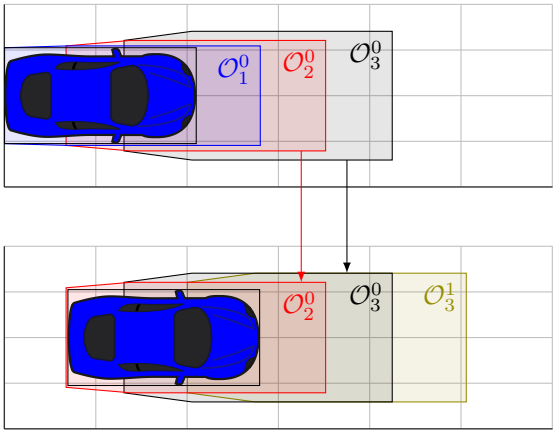
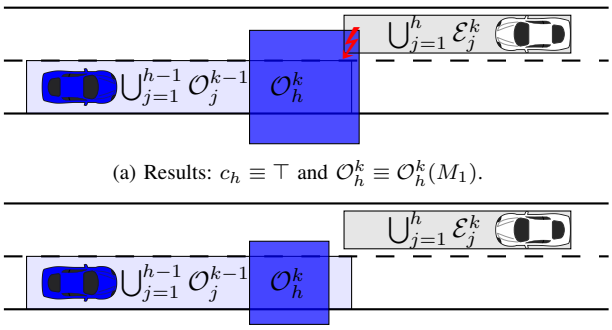


Fig. 4. Occupancy sets $\mathcal{O}_2^0$ and $\mathcal{O}_3^0$ computed at time step t_0 (upper plot) are reused at t_1 (lower plot) to over-approximate $\mathcal{O}_1^1$ and $\mathcal{O}_2^1$, respectively. Only the set $\mathcal{O}_3^1$ is computed at time step t_1.

C. Fast Safety Verification

In lines 8 to 14 of Algorithm 2, we attempt to verify that the motion plan of the ego-vehicle is safe for $t \in [t_k,\ t_{k+h}]$ as quickly as possible. First, the set $\mathcal{O}_j^k$ for $j \in \{1, \ldots, h-1\}$, which is possibly over-approximated by a reused set as explained in Section IV-B, is checked for collision with the ego-vehicle using $\mathcal{E}_j^k$. If the trajectory of the ego-vehicle is unchanged, i.e., $\mathcal{E}_j^k \subseteq \mathcal{E}_{j+1}^{k-1}$, and we can reuse $\mathcal{O}_{j+1}^{k-1}$, the collision check in line 10 always returns $\bot$ and can thus be omitted. However, if a collision is detected in line 10 for a reused set and a changed motion plan, it is unclear whether this is a true or spurious collision due to the reuse of over-approximations. In this case, we verify safety for the first $h-1$ prediction intervals exactly as done for $[t_{k+h-1},\ t_{k+h}]$, which is described in the following.

To speed up the safety verification, we order the list of models M such that M_i has lower computational complexity than M_{i+1} for all $i \in \{1, \ldots, |M|-1\}$. As a complexity measure, we use the number of floating point operations required to obtain the corresponding occupancy set. Then, we compute $\mathcal{O}_h^k(M_1)$ corresponding to the simplest abstraction M_1 and intersect this set with the overall occupancy $\mathcal{O}_h^k$ in line 13 of Algorithm 2. Subsequently, a collision check is performed in line 14. If a collision is detected for model M_1, as illustrated in Fig. 5a, we compute $\mathcal{O}_h^k(M_2)$ for the second abstraction M_2, intersect it with the overall set $\mathcal{O}_h^k$ to reduce the over-approximation based on (4), and perform a collision check. This procedure is repeated until safety, i.e., $c_h \equiv \bot$, is eventually verified for model M_{m_h} with $m_h \in \{1, \ldots, |M|\}$, as illustrated in Fig. 5b. Therefore, we formally verify the motion plan of the ego-vehicle as safe using as few model abstractions as possible, beginning with the coarsest ones.



(a) Results: $c_h \equiv \top$ and $\mathcal{O}_h^k \equiv \mathcal{O}_h^k(M_1)$.

(b) Results: $c_h \equiv \bot$ and $\mathcal{O}_h^k \equiv \bigcap_{i=1}^{m_h} \mathcal{O}_h^k(M_i)$.

Fig. 5. Use of m_h models to verify safety for the last prediction interval $[t_{k+h-1},\ t_{k+h}]$, i.e., to show that $\mathcal{O}_h^k \cap \mathcal{E}_h^k \equiv \emptyset$. The first $h-1$ occupancies are over-approximated by the collision-free reused sets obtained at time step t_{k-1}.

The procedure above produces different verification results, i.e., different collision vectors $c \in \mathbb{B}^h$, for the same input data depending on the amount of available computation time. Nevertheless, our interruptible Algorithm 2 can formally verify the safety of the ego-vehicle's trajectory for $t \in [t_k,\ t_{k+h}]$ much faster than Algorithm 1, as shown in Section V. In case we cannot verify an intended motion plan in time, we execute the verified fail-safe maneuver, as explained in Section III-B.

D. Occupancy Set Refinements

In lines 15 to 17 of Algorithm 2, our anytime procedure continues computing the occupancy sets $\mathcal{O}_j^k(M_i)$ based on the more complex models M_i for $i \in \{m_j + 1, \ldots, |M|\}$ and the sensor data obtained at t_k, even though the collision vector c no longer changes. This is done to reduce the over-approximation of the occupancy sets for future reuse of these sets, i.e., at initial time steps $t_{k+\tilde{k}}$ for $\tilde{k} \in \{1, \ldots, h\}$. Thus, if more computation time is available, the other abstractions are additionally used to refine the overall occupancies $\mathcal{O}_j^k$ for all $j \in \{1, \ldots, h\}$. Finally, after all occupancy sets are refined, as illustrated in Fig. 6, Algorithm 2 returns the formal verification result $\mathsf{any}(c)$ and the list of computed occupancies $\mathcal{O}^k$, which are identical to the two outputs of Algorithm 1.

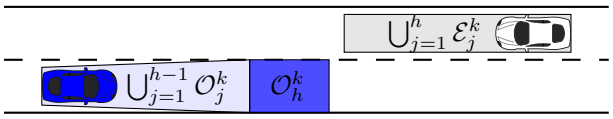


Fig. 6. Refined occupancy sets.

V. Examples

In this section, we compare the performance of the formal safety verification Algorithms 1 and 2 on two benchmarks. Our proposed anytime procedure has been integrated into the open-source MATLAB® tool SPOT [21], which represents occupancies by polygons and implements Algorithm 1. Since collision detection involving polygons is relatively slow, we plan to speed it up using bounding volume hierarchies [23], [24] and pre-computed collision checks [25]. To generate a long-term trajectory, as described in Section III-B, we use the sampling-based approach in [26]. All computations are run on a single thread of an Intel® Core™ i7-7820HQ with 32 GB RAM.

To determine the computational speed-up potential, we terminate Algorithm 2 as soon as the motion plan is verified. To easily reproduce our results, we use the freely available motion planning benchmark suite CommonRoad[1] [27], since performance comparisons are highly dependent on the specific traffic scenario. Each benchmark is specified by a unique identifier and contains detailed information about the ego-vehicle, road network, and other traffic participants.

A. PM1:MW1:DEU_Muc-3_1_T-1 Benchmark

To visualize the computed occupancy sets for two consecutive time steps, we compare the two verification algorithms using the CommonRoad benchmark PM1:MW1:DEU_Muc-3_1_T-1. The considered traffic scenario comprises an uncontrolled intersection with three other traffic participants and specifies that the ego-vehicle makes the left turn. The initial configuration and the occupancies computed at time step t_0 are shown in Fig. 2. The step size is $\Delta t = 0.1\,\mathrm{s}$, and the prediction horizon is $h = 17$, i.e., we predict the occupancies for all surrounding vehicles for the next $1.7\,\mathrm{s}$.

The predicted occupancy sets computed by our interruptible Algorithm 2 at time step t_1 are shown in Fig. 7. As described in Section III-A, we use models M_1, M_2, and M_3, which are ordered by computational complexity. In addition to reusing the occupancies obtained at t_0, it is sufficient to consider only the simplest model M_1 for the first and second vehicles in order to guarantee safety. However, for the third traffic participant, we have to use all three models to formally verify the motion plan.

By averaging the results over 10 simulation runs, we obtain computational speed-ups of Algorithm 2 compared to Algorithm 1 of 33.6, 28.4, and 3.4 for the first, second, and third traffic participants, respectively. This results in an overall speed-up of 7.9 and takes $12\,\mathrm{ms}$ in total.

[1] commonroad.in.tum.de

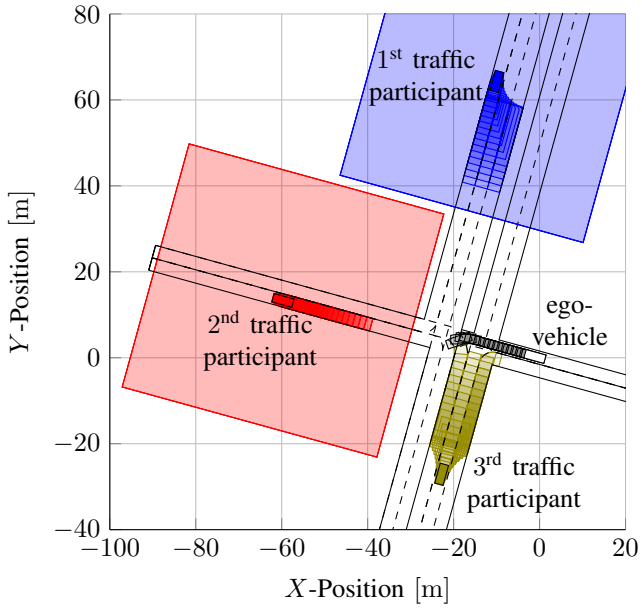


Fig. 7. Occupancy sets of all surrounding traffic participants and ego-vehicle computed by interrupted Algorithm 2 for CommonRoad benchmark PM1:MW1:DEU_Muc-3_1_T-1 at time step t_1.

There are multiple reasons why our proposed anytime method is not even faster. For example, the collision check, which is currently computationally expensive due to the intersection of polygons, is performed each time after a new set $\mathcal{O}_h^k(M_i)$ is intersected with the overall $\mathcal{O}_h^k$ in line 13 of Algorithm 2. In contrast, Algorithm 1 performs a single collision check for only the final occupancy set $\mathcal{O}_h^k$. Thus, if computing the occupancy set for M_{i+1} has lower complexity than performing the collision detection for M_i, it may be beneficial to skip this check to optimize, e.g., the expected overall computation time. More importantly, some computations, e.g., obtaining the reachable lanes for model M_3, must be performed regardless of whether the result is only used for the last prediction time interval $[t_{k+h-1},\ t_{k+h}]$ or for all h intervals. However, the complexity of these computations will be reduced in future implementations.

B. PM1:MW1:DEU_A9-2_1_T-1 Benchmark

The second vehicular traffic example is given by the CommonRoad benchmark PM1:MW1:DEU_A9-2_1_T-1. It features a three-lane highway, where the ego-vehicle is initially located in the middle lane and must perform a lane change to the right one, as shown in Fig. 8. Furthermore, this scenario includes two other traffic participants.

Similar to the previous benchmark in Section V-A, the step size is $\Delta t = 0.1\,\mathrm{s}$, and the prediction horizon is $h = 17$. By averaging the results over 10 simulation runs for the whole lane change maneuver, we obtain computational speed-ups of Algorithm 2 compared to Algorithm 1 of 43.8 and 50.4 for the first and second traffic participants, respectively. This results in an overall speed-up of 47.4 and takes $3\,\mathrm{ms}$ in total. In contrast to the previous benchmark, it is unnecessary to consider the most complex model M_3 for either of the

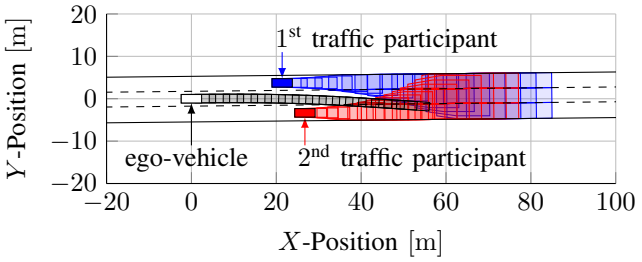


Fig. 8. Initial occupancies of all traffic participants and ego-vehicle for CommonRoad benchmark PM1:MW1:DEU_A9-2_1_T-1.

two other traffic participants in order to guarantee safety, which is the primary reason why a higher overall speed-up is obtained.

VI. Conclusions and Future Work

We have proposed an anytime safety verification approach that attempts to verify that the planned trajectory of the ego-vehicle is collision-free as quickly as possible. The quality of the verification results is continuously improved as long as computation time is available. However, our set-based anytime method would not be possible using simulation-based techniques, such as Monte Carlo simulations, because the concept of abstraction is only applicable when all possible maneuvers can be obtained. Since the benefits of our proposed approach are most apparent in complex traffic scenarios when computational resources are particularly scarce, in future, we plan to use urban traffic data for further evaluation.

Acknowledgments

The authors gratefully acknowledge financial support from the European Commission project UnCoVerCPS under grant number 643921.

References

[1] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH journal*, vol. 1, no. 1, pp. 1–14, 2014.

[2] J. Hayward, "Near-miss determination through use of a scale of danger," in *Highway Research Record*, vol. 384, 1972, pp. 24–34.

[3] K. Vogel, "A comparison of headway and time to collision as safety indicators," *Accident Analysis & Prevention*, vol. 35, pp. 427–433, 2003.

[4] A. Tamke, T. Dang, and G. Breuel, "A flexible method for criticality assessment in driver assistance systems," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2011, pp. 697–702.

[5] M. Brännström, E. Coelingh, and J. Sjöberg, "Model-based threat assessment for avoiding arbitrary vehicle collisions," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 658–669, 2010.

[6] J.-H. Kim and D.-S. Kum, "Threat prediction algorithm based on local path candidates and surrounding vehicle trajectory predictions for automated driving vehicles," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2015, pp. 1220–1225.

[7] A. E. Broadhurst, S. Baker, and T. Kanade, "Monte Carlo road safety reasoning," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2005, pp. 319–324.

[8] A. Eidehall and L. Petersson, "Statistical threat assessment for general road scenes using Monte Carlo sampling," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, pp. 137–147, 2008.

[9] M. Althoff, O. Stursberg, and M. Buss, "Model-based probabilistic collision detection in autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 2, pp. 299–310, 2009.

[10] T. Gindele, S. Brechtel, and R. Dillmann, "Learning driver behavior models from traffic observations for decision making and planning," *IEEE Intelligent Transportation Systems Magazine*, vol. 7, no. 1, pp. 69–79, 2015.

[11] M. Althoff and A. Mergel, "Comparison of Markov chain abstraction and Monte Carlo simulation for the safety assessment of autonomous cars," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1237–1247, 2011.

[12] B. Vanholme, D. Gruyer, B. Lusetti, S. Glaser, and S. Mammar, "Highly automated driving on highways based on legal safety," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 1, pp. 333–347, 2013.

[13] M. Althoff and S. Magdici, "Set-based prediction of traffic participants on arbitrary road networks," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 2, pp. 187–202, 2016.

[14] S. Bouraine, T. Fraichard, and H. Salhi, "Provably safe navigation for mobile robots with limited field-of-views in dynamic environments," *Autonomous Robots*, vol. 32, no. 3, pp. 267–283, 2012.

[15] H. Täubig, U. Frese, C. Hertzberg, C. Lüth, S. Mohr, E. Vorobev, and D. Walter, "Guaranteeing functional safety: design for provability and computer-aided verification," *Autonomous Robots*, vol. 32, no. 3, pp. 303–331, 2012.

[16] S. Zilberstein, "Using anytime algorithms in intelligent systems," *AI magazine*, vol. 17, no. 3, pp. 73–83, 1996.

[17] G. Lafferriere, G. J. Pappas, and S. Yovine, "Symbolic reachability computation for families of linear vector fields," *Symbolic Computation*, vol. 32, pp. 231–253, 2001.

[18] H. Roehm, J. Oehlerking, M. Woehrle, and M. Althoff, "Reachset conformance testing of hybrid automata," in *Proc. of Hybrid Systems: Computation and Control*, 2016, pp. 277–286.

[19] B. Schürmann, D. Heß, J. Eilbrecht, O. Stursberg, F. Köster, and M. Althoff, "Ensuring drivability of planned motions using formal methods," in *Proc. of the 20th IEEE International Conference on Intelligent Transportation Systems*, 2017, pp. 1–8.

[20] M. Althoff and J. M. Dolan, "Online verification of automated road vehicles using reachability analysis," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 903–918, 2014.

[21] M. Koschi and M. Althoff, "SPOT: A tool for set-based prediction of traffic participants," in *IEEE Intelligent Vehicles Symposium*, 2017, pp. 1686–1693.

[22] S. Magdici and M. Althoff, "Fail-safe motion planning of autonomous vehicles," in *Proc. of the 19th International IEEE Conference on Intelligent Transportation Systems*, 2016, pp. 452–458.

[23] J. T. Klosowski, M. Held, J. S. B. Mitchell, H. Sowizral, and K. Zikan, "Efficient collision detection using bounding volume hierarchies of k-DOPs," *IEEE Transactions on Visualization and Computer Graphics*, vol. 4, no. 1, pp. 21–36, 1998.

[24] C. Ericson, *Real-time collision detection*. CRC Press, 2004.

[25] A. Rizaldi, S. Söntges, and M. Althoff, "On time-memory trade-off for collision detection," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2015, pp. 1173 – 1180.

[26] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a frenét frame," in *IEEE Conference on Robotics and Automation*, 2010, pp. 987–993.

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