Online Verification of Cognitive Car Decisions

Matthias Althoff, Olaf Stursberg and Martin Buss

Abstract—Verifying a safe locomotion of cognitive cars is indispensable for their participation in road traffic. This paper suggests an approach for verifying safety by computing reachable sets for the position of relevant traffic participants, i.e. a cognitive car as well as moving objects in its environment. In order to account for the uncertainty in the behavior of traffic participants, a stochastic setting is chosen, in which Markov chains represent the positions probabilistically. An efficient online algorithm is presented that leads to the result whether the reachable sets of different traffic participants can intersect, meaning that the control strategy of the cognitive car is possibly unsafe.

I. INTRODUCTION

Cognitive cars navigate autonomously through traffic based on measured information (cameras, lidar, radar,...) without intervention of human drivers. Analog to human drivers, cognitive cars need a sense of safety, i.e. they have to know which behaviors possibly lead to an accident. Behavior is referred to a computed reference trajectory that the cognitive car tries to follow. Driving along the path of the reference trajectory may cause an accident if the possible behavior of other traffic participants is not appropriately considered. Additionally, it is important to account for the deviation between the actual behavior of the cognitive car and the reference trajectory due to disturbances.

An approach towards collision free path planning in static environments can be found, e.g. in [1]. For dynamic environments, current literature shows that safe navigation of intelligent vehicles is largely an open research problem, see e.g. [2], [3], [4], and forward collision avoidance systems still exhibit deficiencies [5]. A novel approach for safe motion planning is established by avoiding inevitable collision states in [6]. However, this work differs from the approach presented here by applying simulation instead of verification techniques. A major drawback of simulation is that it can only proof that a system with uncertain behavior is unsafe, but not that it is safe. This is due to an infinite number of initial and disturbance values that have to be simulated if the uncertainties are modeled by sets of initial and disturbance values.

Algorithmic verification, as it is applied in this work, has been developed for hybrid systems in recent years. Hybrid systems evolve according to a mixed discrete and continuous dynamics. The model framework of hybrid systems is very useful in the context of traffic modeling: traffic participants, e.g. cars, trucks and bicycles make logic decisions, such as lane changing, turning and stopping which are appropriately modeled by discrete dynamics. Additionally, the vehicle dynamics is best described by continuous differential equations. Most hybrid verification algorithms compute the set of (discrete and continuous) states which are reachable by the investigated system. If the set does not intersect sets of unsafe states, safety can be concluded. However, the verification problem is known to be decidable for a limited class of hybrid systems only [7]. For this reason, sets of reachable states are overapproximated which allows to conclude safety, but possibly leads to the result of unsafe behavior for safe systems. However, this conservative approach is suitable in traffic since only confident trajectories of the cognitive car should be executed. Prominent verification algorithms using overapproximated reachable sets compute ellipsoids [8], polytopes [9], oriented rectangular hulls [10] and zonotopes [11]. In contrast to the common use of verification techniques for offline safety analysis, verification algorithms are applied for online safety analysis in this paper. The online application is necessary as unsafe states originating from the reachable sets of moving obstacles (other traffic participants) are not known a priori. Along with the online application comes the demand for real time constraints of the verification algorithms. In order to speed up the verification process for online application, the continuous system dynamics of the cognitive car and other traffic participants is abstracted by Markov chains, similar as used in a different context in [12]. This conservative transformation is based on a discretization of the state and input space of the model. However, Markov chains provide fast and probabilistic computations of reachable sets which has been shown in [13] and for stochastic aircraft conflict situations in [14]. The presented approach is an extension of the previous work in [15] by

- allowing disturbances bounded by hyperrectangles (interval hulls) instead of hypercubes,
- probabilistically modeled disturbances,
- improving the abstraction process from nonlinear to linear systems,
- more efficient execution of Markov chains due to partial transition executions,
- advanced construction of reachable sets of other traffic participants.

In contrast to [15], not only the state space is discretized, but the input space, too, which allows faster computation of probabilistic reachable sets.
An overview of the approach from the nonlinear dynamics of the cognitive car and its surrounding traffic participants to the probabilistic computation of reachable sets is given in Fig. 1. The offline computation consists of the following steps: first, the nonlinear continuous dynamics is conservatively abstracted by a linear differential inclusion (Sec. III). The linear uncertain model is then further abstracted to Markov chains for different input and disturbance sets as well as for discrete time and time intervals (Sec. IV and V). During online application, the Markov chains obtained from the offline computation are executed. They compute the reachable sets of the cognitive car and of other traffic participants based on behavior assumptions (Sec. VI) between the time indices \( l - 1 \) and \( l \), see Fig. 1. Each verification process ends with intersecting the reachable sets of other traffic participants with the one of the cognitive car as shown exemplarily for a traffic scenario (Sec. VII). This allows to calculate the probability of a crash and due to the conservative computation of reachable sets, safety can be guaranteed if the set intersection is empty. After the verification process is finished, it is reset and started with actual sensor values so that the safety of locomotion is continuously evaluated.

![Fig. 1. Verification process overview](image)

### II. PROBLEM STATEMENT

At the beginning of each verification process, one has the following situation: Given are the uncertain positions and velocities of other traffic participants that are identified by the cognitive car and modeled by sets. The position, velocity, direction and angular speed of the cognitive car are also known but subject to uncertainties. Additionally, the reference trajectory of the cognitive car is assumed to be given. The goal of the presented method is to determine the probability that the cognitive car will crash into another traffic participant within certain time intervals \( t \in [0, t_f] \), for a given horizon \( t \leq \Delta t k, k \in \mathbb{N}^+, \Delta t \in \mathbb{R}^{>0} \) when following the reference trajectory, see Fig. 2(a). The time interval \( \Delta t \) specifies the time span after which actual sensor values are read out and \( k \) is the factor for \( \Delta t \) determining the time horizon. The probability of a crash is computed by the reachable set of the cognitive car and the ones of other traffic participants starting from the uncertain initial states under disturbances. This is illustrated exemplarily for the cognitive car and one additional traffic participant in Fig. 2(b). If the intersections of the reachable sets are empty, the trajectory of the cognitive car is safe and otherwise, the probability of the crash is computed. The verification process is repeated after each time step \( \Delta t \) using an update of the dynamic models according to the current sensor readings. Note that the verification has to be terminated after the time \( \Delta t \), i.e. the procedure has to run \( k \) times faster than real time.

![Fig. 2. Verification results](image)

### III. ABSTRACTION FROM NONLINEAR TO LINEAR CONTINUOUS DYNAMICS

In order to simplify the computation of the reachable sets, the dynamics of the nonlinear systems is conservatively abstracted to linear but uncertain ones, i.e. the reachable sets of the nonlinear systems are enclosed by the ones of the uncertain linear systems. In order to apply the presented abstraction method, the state \( x \) of the nonlinear system, the input \( u \) and the disturbance \( v \) are limited to sets: \( x \in X \subset \mathbb{R}^n, u \in U \subset \mathbb{R}^m \) and \( v \in V \subset \mathbb{R}^o \). In case of the cognitive car, the input \( u \) is interpreted as the reference trajectory, or respectively, as the acceleration for other cars, see Sec. VII. The nonlinear system has the following form:

\[
\dot{x} = f(x, u) + v, \quad x(0) \in X, u \in U, v \in V
\]  

For the presented approach it is necessary to discretize the sets \( X, U, \) and \( V \). The discretization \( D_X : X \rightarrow \mathbb{I} \) is a map which assigns to each value \( x \in X \subset \mathbb{R}^n \) an identifier \( i \in \mathbb{I} \subset \mathbb{N}^+ \) where \( \mathbb{I} \) is the finite set of identifiers. The connected subset that is mapped to an identifier \( i \) is denoted by \( X_i = \{x|D_X(x) = i\} \) and referred to as a cell. The state space is discretized rectangurally and equidistant by \( D_X \) so that all cells \( X_i \) are interval hulls \((\mathbb{I}^i, \mathbb{I}^j)\) of equal lengths with \( x, \mathbb{I} \in \mathbb{R}^n \). Analogously, there exists a discretization function \( D_U : U \rightarrow \mathbb{I} \) that assigns to each identifier \( j \) a cell \( U_j \) and another discretization function \( D_V : V \rightarrow \mathbb{I} \) that assigns identifiers \( m \) to cells \( V_m \). The sets of initial states \( x(0) \), inputs \( u \in U_j \), and \( v \in V_m \) are expressed in terms of

\(^1\)The discretization can lead to a large number of cells for high-dimensional systems, requiring to use order reduction techniques.
these cells. This allows to conservatively approximate (1) by a linear system that is modeled as a differential inclusion:

\[
\dot{x} \in A^i_j x + B^i_j u + b^i_j + v + E^i_j, \quad 1\text{st order Taylor expansion} \\
\text{Lagrangian remainder} \\
x(0) \in X_i, u \in U_j, v \in V_m, t \in [0, T] \tag{2}
\]

\(A^i_j\) is the system matrix, \(B^i_j\) the input matrix, \(b^i_j\) a constant vector and \(T\) is the time horizon for which the above equation holds. The values \(A^i_j, B^i_j, b^i_j\) are obtained by a first order Taylor expansion of the nonlinear system (1) and the indices \(i, j\) refer to the indices of the sets \(X_i\) and \(U_j\). Note, that \(b^i_j \neq 0\) as the nonlinear system is not linearized in a steady state. In order to abstract the nonlinear dynamics conservatively, the linearization error \(E^i_j\) is added by Minkowski addition. In contrast to \(A^i_j, B^i_j\) and \(b^i_j\), the result of \(E^i_j\) also depends on the uncertainty cell \(V_m\). The set \(E^i_j\) can be obtained by evaluating the Lagrange remainder of the first order Taylor expansion using interval arithmetics [16] as shown in [15]. A disadvantage of this method are the relatively conservative bounds of \(E^i_j\). In order to tighten these bounds, branch and bound methods known from global optimization [17] have been applied for the traffic scenario in this paper. This method is based on selective division of intervals (branching) so that interval analysis returns better bounds (bounding).

A remaining task is the proper selection of the linearization point to reduce the linearization term \(E^i_j\). In order to suggest a selection of the linearization point, the set of overapproximated reachable states of (2) in the time interval \([0, T]\) is introduced. This set is denoted by \(R^i_j \{0, T\}\) and defined over an auxiliary set \(R^i_j \{0, T\}\):

**Definition 1:** \(R^i_j \{0, T\}\) is an overapproximated set of the exact reachable set \(R^i_j \{0, T\}\) at time \(t = T\): \(R^i_j \{0, T\}\) is solution of (2), \(t = T, x(0) \in X_i, u \in U_j, v \in V_m\) and \(R^i_j \{0, T\} \supset R^i_j \{0, T\}\).

**Definition 2:** \(R^i_j \{0, T\}\) is the union of all overapproximated reachable sets \(R^i_j \{t\}\) for \(t \in [0, T]\): \(R^i_j \{0, T\} = \bigcup_{t \in [0, T]} R^i_j \{t\}\).

The description of the computation of \(R^i_j \{0, T\}\) is given in Sec. IV. The selection of the linearization error is motivated by the observation that it usually grows with increasing distance to the linearization point. As all states are within \(R^i_j \{0, T\}\), the maximum linearization error is reduced by the heuristics that the volumetric center of \(R^i_j \{0, T\}\) is chosen as the linearization point \(x^*\) (this choice is different from the one in [15]). The linearization point for the input \(u^*\) is chosen as the center of \(U\).

IV. REACHABILITY

The reachable set of the continuous evolution \(R^i_j \{0, T\}\) is computed by zonotopes [11], [15], see Fig. 3(a). Zonotopes are used as they are closed under Minkowski sum which results in an efficient computation of reachable sets under uncertain inputs. The difference to [11] and [15] is that the input \(B^i_j u + b^i_j + v \in E^i_j \in B^i_j U_j \oplus b^i_j \oplus V_m \oplus E^i_j =: W^*\) is within a hyperrectangle (interval hull) instead of a hypercube. For further computations, the uncertain input \(W^*\) is split up into an interval hull \(W = W^* - \text{mid}(W^*)\) with the volumetric center at the origin and the constant input \(\text{mid}(W^*)\), where the operator \(\text{mid}(\cdot)\) returns the volumetric center of a set. The reachable set occurring due to the input \(W\) is denoted \(\hat{R}^i_j \{0, T\}\) and the reachable set resulting from the solution of the dynamics for the initial state \(x(0)\) and the constant input \(\text{mid}(W^*)\) is denoted \(\hat{R}^i_j \{0, T\}\). The superposition principle allows to compute the reachable set \(\hat{R}^i_j \{0, T\}\) of the linear system (2) by Minkowski addition of \(\hat{R}^i_j \{0, T\}\) and \(\hat{R}^i_j \{0, T\}\):

\[
\hat{R}^i_j \{0, T\} = \hat{R}^i_j \{0, T\} \oplus \hat{R}^i_j \{0, T\}
\]

The computation of \(\hat{R}^i_j \{0, T\}\) is presented in [11], and the reachable set \(\hat{R}^i_j \{0, T\}\) is computed in modal space:

\[
\dot{x} \in \hat{A}^i_j \hat{x} \oplus \hat{W}
\]

with \(\hat{x} = M^{-1} x, \hat{A}^i_j = M^{-1} A^i_j M, \hat{W} = M^{-1} W\) and \(M\) is the matrix of eigenvectors of \(A^i_j\). The transformation to modal coordinates is done as the \(k\)-th dimension of the input \(W\) exclusively affects the \(k\)-th component of \(\hat{x}\). The \(k\)-th component of the interval hull, denoted \(W_k\), is an interval \([-\hat{w}_k, \hat{w}_k]\) with \(\hat{w} \in \mathbb{R}^n\). The trace of \(\hat{A}^i_j\) is represented by a vector \(\alpha\), and \(\alpha_k\) is the \(k\)-th element of \(\alpha\). The reachable interval hull \(F\) is obtained elementwise by intervals \(F_k\) (\(k\)-th dimension of \(F\)) by the following estimates:

\[
F_k = \int_0^T e^{\alpha_k (t-\tau)} d\tau [-\hat{w}_k, \hat{w}_k] \\
\|F_k\|_\infty \leq \int_0^T \|e^{\alpha_k (t-\tau)}\|_\infty d\tau \|[-\hat{w}_k, \hat{w}_k]\|_\infty \\
\leq \int_0^T e^{\|\alpha_k\|_\infty (t-\tau)} d\tau \|\hat{w}_k\|_\infty \\
= \|\alpha_k\|_\infty (e^{\|\alpha_k\|_\infty T} - 1) \|\hat{w}_k\|_\infty
\]

The norm estimates are necessary as the trace \(\alpha\) may contain conjugate complex values. The infinity norm is chosen as the set of maximum size fulfilling the infinity norm is an interval hull: \(F_k = [-f_k, f_k] = \{x : \|x\|_\infty < f_k\}\). The reachable set \(\hat{R}^i_j \{0, T\}\) results in \(\hat{R}^i_j \{0, T\} = MF\).

V. MARKOV CHAINS

The reachable set \(\hat{R}^i_j \{0, T\}\) is used to obtain the transition probabilities of the Markov chain abstracting the behavior of the linear system. The Markov chain consists of states \(i \in I\) which are the cells of the discretized state space, and \(p_i\) is the probability that the system is in cell \(i\). The transition matrix \(F\) specifies the transitions between states: \(p(l+1) = \Phi p(l)\) and \(p(l)\) is the probability vector at time step \(l\). The conversion from continuous dynamics to Markov chains is based on the assumption that the continuous state
of the linear system is evenly distributed within the reachable set \( R_i^m([0, T]) \):

\[
\Phi_{oi}(m, [0, T]) = \frac{V(R_i^m([0, T]) \cap X_o)}{V(R_i^m([0, T]))}
\]

where \( V() \) is an operator determining the volume of a geometric object. The transition matrix \( \Phi_{oi}(m, [0, T]) \) contains the probabilities that a trajectory starting in cell \( X_i \) with input \( u \in U_j \) and disturbance \( v \in V_m \) can be found in cell \( X_o \) within the time span \([0, T]\). This is in contrast to [12], where time is not explicitly considered. A two dimensional example of computing probabilistic reachable sets of time is not explicitly considered. A two dimensional example in cell probabilities are introduced. The probability that the input is probability that the disturbance is in cell \( \phi \in [0, T] \) based on the reachable set in Fig. 3(a) is shown in Fig. 3(b). In order to obtain the transition matrix for a certain input \( u(t) \) and disturbance \( v(t) \), \( t \in [T, (l+1)T] \), additional probabilities are introduced. The probability that the input is in cell \( U_j \) for \( t \in [T, (l+1)T] \) is denoted by \( q_j(l) \) and the probability that the disturbance is in cell \( V_m \) is denoted by \( c_m \). In contrast to the probability vector \( q(l) \), the probability vector \( c \) is modeled time invariant. Applying the rule for the computation of unconditional probabilities, the transition matrix under input \( u \) and disturbance \( v \) is computed as:

\[
\Phi_{oi}(lT, (l+1)T) = \sum_{j=1}^{[j]} q_j(l) \sum_{m=1}^{[m]} c_m \Phi_{oi}(m, [0, T])
\]

The transition probabilities for the time point solution \( \Phi_{oi}(lT) \) are calculated in an analogous way. The time point solution is computed in order to provide the initial probabilistic set for the time interval solution after each time step. This approach differs from the one in [12] and improves the accuracy compared to the exclusive use of \( \Phi_{oi}(lT, (l+1)T) \) for two reasons: First, reachable sets for linear systems at time points without uncertain inputs can be computed exactly, see e.g. [9]. Consequently, the worse approximation of the time interval solution is not propagating as it is computed based on the time point solution. Second, the computation of \( R_i^m([0, T]) \) and hence for \( \Phi_{oi}(lT, (l+1)T) \) is based on an initial set of states at a time point so that an initial set obtained from a time interval solution would be more conservative. The equations for the computation of the probability vector \( p(l+1) \) in the time interval \( t \in [T, (l+1)T] \) and the auxiliary probability vector \( \tilde{p}_o(l) \) for the time point \( t = lT \) are:

\[
\begin{align*}
\tilde{p}_o(l+1) &= \Phi_{oi}(lT)\tilde{p}_o(l) \\
p_o(l+1) &= \Phi_{oi}([lT, (l+1)T])\tilde{p}_o(l)
\end{align*}
\]

In order to save computation time for evaluating (3), transitions of the Markov chain are executed depending on the original continuous system dynamics (1). After defining \( \theta_i(T) = \max_{j=m}^T ||E_{i}^{m}(T)||_{\infty} \), one can choose time constants \( T_i = \rho T, \rho \in \mathbb{N}^+ \) that are assigned to cells \( X_i \) in order to ensure that the linearization error stays below a specified bound \( \theta_i(T_i) < \theta \). The time varying set containing the admissible states \( k \) at time step \( l \) is denoted \( \mathbb{K}(l) \). Note that \( \mathbb{K}(l) \) enables transitions at the beginning of any time interval \([T, (l+1)T]\) to ensure conservativeness of the probabilistic reachable set. The extended probability update function is:

\[
\begin{align*}
\tilde{p}_o(l+1) &= \Phi_{oi}(lT)\tilde{p}_o(l) + p_m(l) \\
p_o(l+1) &= \Phi_{oi}([lT, (l+1)T])\tilde{p}_o(l) + p_m(l) \\
k &= \mathbb{K}(l), \quad m \in \mathbb{N} \mathbb{K}(l)
\end{align*}
\]

VI. BEHAVIOR MODELING OF OTHER TRAFFIC PARTICIPANTS

For safety assessment of cognitive cars, prediction of the behavior of other traffic participants is crucial. Similar to human driving, the cognitive car expects a certain behavior of other traffic participants which is addressed in the following.

A. Assumptions

The most important assumption about the behavior of other road users is that road traffic regulations are met. This excludes behaviors such that an approaching car from the opposite lane steers into the cognitive car. If such behavior is observed, the verification algorithms have to take the physically possible instead of the permitted behavior of this traffic participant into account.

B. Path Generation and Path Following

The behavior of other traffic participants is modeled in two stages: path generation and path following. Possible paths of traffic participants can be composed by elementary actions such as lane following, turn left/right or lane changing. This is illustrated in Fig. 4 for a car approaching a crossing. The paths consist of clothoid segments [18] of length \( s \) or \( s \leq s' \leq 2s \) in front of branching points. In a next step, the finite set of paths is enhanced by considering deviations from these which represents an infinite set of possible paths. Deviation is modeled as a static piecewise constant probability distribution that varies between road user types. Examples for these probabilities are given for cars and bicycles in Fig. 5. Path following is also modeled by elementary actions, like accelerating, braking, stand still and drive at speed limit. Note that elementary actions cover a set of behaviors, e.g. accelerating encloses all behaviors in between minimum and maximum acceleration. An exemplary model of the longitudinal dynamics along possible paths is presented in Sec. VII-B.
The car parameters $c_1, c_2$ and the constant speed $c_3$ of the car can be found in table I.

### B. Model of the Other Car

The longitudinal dynamics of the other car for path following is modeled as a switching system with the modes standstill, speed limit, brake and accelerate, see Fig. 7. Invariant sets are denoted by $I$ and transitions by $t$. The transition guards or probabilities can be found next to the transition arrows. When a transition is taken, the continuous states are not reset. The continuous dynamics is described by the position $y_1$ and the velocity $y_2$. The brake model $h_1(u)$ and the acceleration model $h_2(y, u)$ are given as

$$h_1(u) = -c_4 u, \quad u \in [0, 1]$$
$$h_2(y, u) = c_4 (1 - \frac{y_2}{c_5}) u, \quad u \in [0, 1]$$

Note, that the input $u$ of the brake and acceleration model is uncertain in its bounds. The discrete dynamics of the switching model also contains uncertainty. In opposite to transitions $t_1, t_3$, all other transitions are taken by probabilities $p$. It is believed that the stochastic driver model is best suited as only few information is available about other traffic participants.

### C. Reachable Sets

The parameters for the reachable sets of both cars are given as follows: The set of initial conditions is listed in table II. The variables $p_{\text{acc}}, p_{\text{brake}}, p_{\text{sl}}, p_{\text{ss}}$ refer to the probability that the discrete state is in $\text{acceleration}(\text{acc}), \text{brake}(\text{brake}), \text{speed limit}(\text{sl})$ or $\text{standstill}(\text{ss})$ mode at $t = 0$. The state space discretization is summarized in table III.
and the cognitive car is disturbed by $\|v\|_\infty < 0.01$. The computation time was 2.7s for the cognitive car and 0.5s for the other car on a dual core processor (1.66 GHz) for 4s in real time. The probability of a crash is 0.01% for $t \in [3,4]$s and 0% for all other times. The probability of crash is simply determined by the scalar product of the probability vector of the cognitive and the other car.

The resulting reachable sets of both cars are visualized in Fig. 8 for four time intervals. The car starting from the right lane is the cognitive car, the one starting from the left one is the other car. The green line shows the reference trajectory of the cognitive car. The red box is a static obstacle on the road. Dark blue color indicates high probability and light blue color small probability that a car is located on the road. Note, that the probabilities refer to the presence of the cognitive car. Dark blue color indicates high probability and light blue color small probability that a car is located on the road. The red box is a static obstacle on the road. Dark blue color indicates high probability and light blue color small probability that a car is located on the road.

TABLE II

<table>
<thead>
<tr>
<th>Set of Initial Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>cognitive car</td>
</tr>
<tr>
<td>$x_1$ [-0.3, 0.3] m</td>
</tr>
<tr>
<td>$x_2$ [-0.3, 0.3] m</td>
</tr>
<tr>
<td>$x_3$ [-0.05, 0.05] rad</td>
</tr>
<tr>
<td>$x_4$ [-0.2, 0.2] rad/s</td>
</tr>
</tbody>
</table>

TABLE III

<table>
<thead>
<tr>
<th>Discretization Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>variable</td>
</tr>
<tr>
<td>cognitive car: 512 cells</td>
</tr>
<tr>
<td>$x_1$</td>
</tr>
<tr>
<td>$x_2$</td>
</tr>
<tr>
<td>$x_3$</td>
</tr>
<tr>
<td>$x_4$</td>
</tr>
<tr>
<td>other car: 1500 cells</td>
</tr>
<tr>
<td>$y_1$</td>
</tr>
</tbody>
</table>

It has been shown that the probability of an accident for the above verification example can be computed faster than real time (3.2s computation for 4s in real time), resulting in a $k$-factor (Sec. II) of $k = 1.25$ which is planned to be increased to $k \approx 4$. Besides future improvements, the computation time can be reduced by enlarging the cell size of the discretized state space at the expense of decreasing accuracy (while keeping the conservativity of the computation).

VIII. Conclusion

REFERENCES