Proceedings of the
2007 IEEE Intelligent Vehicles Symposium
Istanbul, Turkey, June 13-15, 2007

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.

Research supported by the German Research Council (DFG) through the Collaborative Research Center SFB-TR 28 (Cognitive Automobiles).

All authors are with the Institute of Automatic Control Engineering (LSR) , Technische Universität München, 80290 München, Germany.

1-4244-1068-1/07/$25.00 ©2007 IEEE.

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.

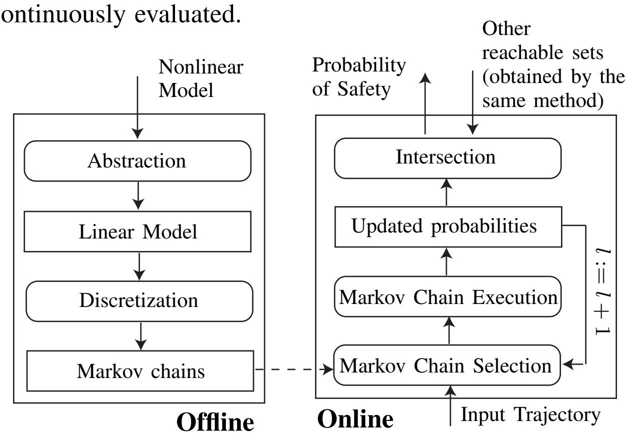
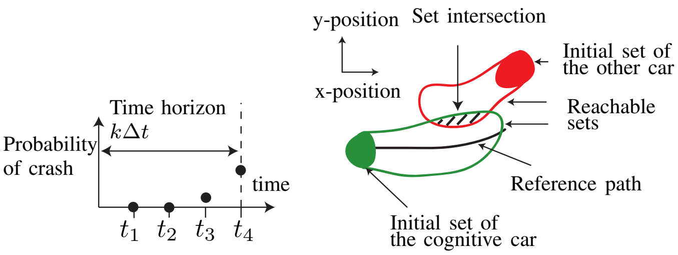


Fig. 1. Verification process overview

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 $t \in [0, t_i]$, for a given horizon $t \leq k\Delta t, k \in \mathbb{N}^+, \Delta t \in \mathbb{R}^{>0}$ when following the reference trajectory, see Fig. 2(a). The time interval Δt specifies the time span after which actual sensor values are read out and k is the factor for Δt 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 Δt using an update of the dynamic models according to the current sensor readings. Note that the verification has to be terminated after the time Δt, i.e. the procedure has to run k times faster than real time.



(a) Probability of a crash for the time horizon $k\Delta t$

(b) Problem setup

Fig. 2. Verification results

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 $v \in V \subset \mathbb{R}^n$. 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:

$$\dot{x} = f(x,u) + v, \quad x(0) \in X, u \in U, v \in V \qquad (1)$$

For the presented approach it is necessary to discretize the sets X, U and V. The discretization $D_X : X \to \mathbb{I}$ 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[1]. 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 rectangularly and equidistant by D_X so that all cells X_i are interval hulls $(]\underline{x}, \overline{x}])$ of equal lengths with $\underline{x}, \overline{x} \in \mathbb{R}^n$. Analogously, there exists a discretization function $D_U : U \to \mathbb{J}$ that assigns to each identifier j a cell U_j and another discretization function $D_V : V \to \mathbb{M}$ 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:

$$\dot{x} \in \underbrace{A_i^j x + B_i^j u + b_i^j + v}_{1^{st} \text{ order Taylor expansion}} \oplus \underbrace{E_i^{jm}}_{\substack{\text{Lagrange} \\ \text{remainder}}} , \tag{2}$$
$$x(0) \in X_i,\, u \in U_j,\, v \in V_m, t \in [0,T]$$

A_i^j is the system matrix, B_i^j the input matrix, b_i^j is 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 and 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^{jm} is added by Minkowski addition[2]. In contrast to A_i^j,B_i^j and b_i^j, the result of E_i^{jm} also depends on the uncertainty cell V_m. The set E_i^{jm} 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^{jm}. 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^{jm}. 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 $R_i^{jm}([0,T])$ and defined over an auxiliary set $R_i^{jm}(T)$:

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

Definition 2: $R_i^{jm}([0,T])$ is the union of all overapproximated reachable sets $R_i^{jm}(t)$ for $t \in [0,T]$: $R_i^{jm}([0,T]) = \bigcup_{t\in[0,T]} R_i^{jm}(t)$

The description of the computation of $R_i^{jm}([0,T])$ 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 $R_i^{jm}([0,T])$, the maximum linearization error is reduced by the heuristics that the volumetric center of $R_i^{jm}([0,T])$ 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 $R_i^{jm}([0,T])$ 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 $B_i^j u + b_i^j + v \oplus E_i^{jm} \in B_i^j U_j \oplus b_i^j \oplus V_m \oplus E_i^{jm} =: W^*$ is within an hyperrectangle (interval hull) instead of a hypercube. For further computations, the uncertain input W^* is split up into an interval hull $W = W^* - mid(W^*)$ with the volumetric center at the origin and the constant input $mid(W^*)$, where the operator $mid()$ returns the volumetric center of a set. The reachable set occurring due to the input W is denoted $\bar{R}_i^{jm}([0,T])$ and the reachable set resulting from the solution of the dynamics for the initial state $x(0)$ and the constant input $mid(W^*)$ is denoted $\hat{R}_i^{jm}([0,T])$. The superposition principle allows to compute the reachable set $R_i^{jm}([0,T])$ of the linear sytstem (2) by Minkowski addition of $\hat{R}_i^{jm}([0,T])$ and $\bar{R}_i^{jm}([0,T])$:

$$R_i^{jm}([0,T]) = \hat{R}_i^{jm}([0,T]) \oplus \bar{R}_i^{jm}([0,T])$$

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

$$\dot{\hat{x}} \in \hat{A}_i^j \hat{x} \oplus \hat{W}$$

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 $\hat{W}$ exclusively affects the k-th component of $\hat{x}$. The k-th component of the interval hull, denoted $\hat{W}_k$ 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 α, and α_k is the k-th element of α. The reachable interval hull F is obtained elementwise by intervals F_k (k-th dimension of F) by the following estimates:

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

The norm estimates are necessary as the trace α 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 $\bar{R}_i^{jm}([0,T])$ results in $\bar{R}_i^{jm}([0,T]) = MF$.

V. Markov Chains

The reachable set $R_i^{jm}([0,T])$ is used to obtain the transition probabilities of the Markov chain abstracting the behavior of the linear system. The Markov chain consists of states $i \in \mathbb{I}$ 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 Φ 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

[2] $A \oplus B = \{a + b | a \in A, b \in B\}$

of the linear system is evenly distributed within the reachable set $R_i^{jm}([0,T])$:

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

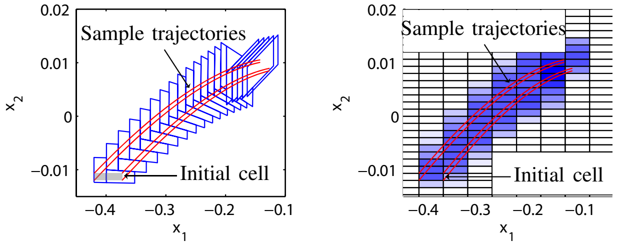
where $V()$ is an operator determining the volume of a geometric object. The transition matrix $\Phi_{oi}^{jm}([0,T])$ contains the probabilities that a trajectory starting in cell X_i with input $u \in U_j$ and disturbance $v \in V_m$ can be found in cell X_o within the time span $[0,T]$. This is in contrast to [12], where time is not explicitly considered. A two dimensional example of computing probabilistic reachable sets of $x = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T$ 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 $v(t)$, $t \in [lT,(l+1)T]$, additional probabilities are introduced. The probability that the input is in cell U_j for $t \in [lT,(l+1)T]$ 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[3], the transition matrix under input u and disturbance v is computed as:

$$\Phi_{oi}([lT,(l+1)T]) = \sum_{j=1}^{|\mathbb{J}|} q_j(l) \sum_{m=1}^{|\mathbb{M}|} c_m \Phi_{oi}^{jm}([0,T])$$

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 $R_i^j([0,T])$ 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 $t \in [lT,(l+1)T]$ and the auxiliary probability vector $\tilde{p}_o(l)$ for the time point $t = lT$ are:

$$\begin{aligned} \tilde{p}_o(l+1) &= \Phi_{oi}(lT)\tilde{p}_i(l) \\ p_o(l+1) &= \Phi_{oi}([lT,(l+1)T])\tilde{p}_i(l) \end{aligned} \tag{3}$$

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 $\theta_i(T) = \max_{j,m} \|E_i^{jm}(T)\|_\infty$, 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 $\bar{\theta}$: $\theta_i(T_i) < \bar{\theta}$. 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 $[lT, lT+T_i]$ to ensure conservativeness of the probabilistic reachable set. The extended probability update function is:



(a) Reachable sets described by zonotopes

(b) Probabilistic reachable set

Fig. 3. From zonotopes to transition probabilities

$$\begin{aligned} \tilde{p}_i(l+1) &= \Phi_{ik}(lT)\tilde{p}_k(l) + \tilde{p}_m(l) \\ p_i(l+1) &= \Phi_{ik}([lT,(l+1)T])\tilde{p}_k(l) + p_m(l) \\ k &\in \mathbb{K}(l), \quad m \in \mathbb{I}\backslash\mathbb{K}(l) \end{aligned}$$

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.

[3] $P(\beta) = \sum_{a=1}^{b} P(\beta|\alpha_a)P(\alpha_a)$, where α_i are mutually exclusive events and $\bigcup_{a=1}^{b} \alpha_a = \Omega$ is the certain event Ω

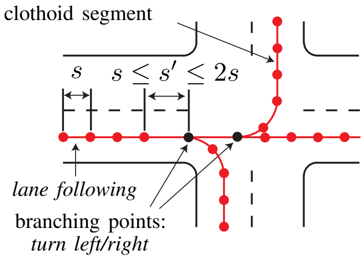


Fig. 4. Path Generation

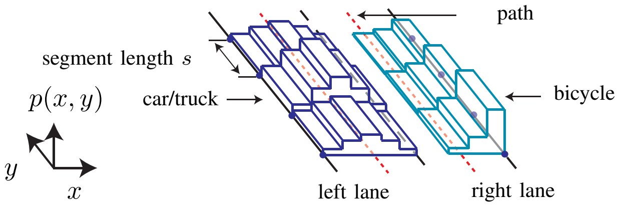


Fig. 5. Deviation probabilities

VII. Verification of an Exemplary Traffic Scenario

To demonstrate the presented method, a typical traffic scenario is investigated, see Fig. 6. The cognitive car is controlled along a reference trajectory to avoid the static obstacle and the oncoming car on the opposite lane. As discussed before, it is assumed that the other car respects the traffic regulations, i.e. it does not leave its lane.

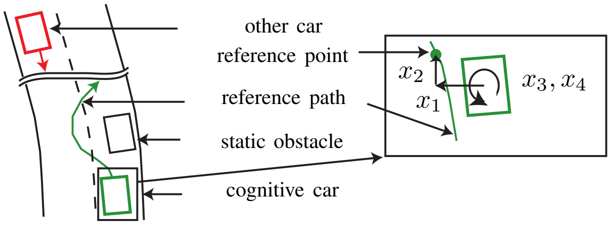


Fig. 6. Verification scenario

A. Model of the Cognitive Car

The lateral dynamics is modeled by a simplified bicycle model [19] with yaw angle x_3 and yaw rate x_4, see Fig. 6. The position deviation of the center of gravity to the reference point on the reference path in road-fixed coordinates is denoted x_1 and x_2, see Fig. 6. The lateral control is given as $u = w - x_3 - 0.1(x_2 \cos(w) - x_1 \sin(w))$, where $u(t)$ is the steering wheel angle and $w(t)$ is the orientation of the reference trajectory. The controlled car model is:

$$\begin{aligned}
\dot{x}_1 &= c_3(\cos(x_4) - \cos(w)) \\
\dot{x}_2 &= c_3(\sin(x_4) - \sin(w)) \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= \frac{c_1}{c_3} x_4 + c_2(w - x_3 - 0.1(x_2 \cos(w) - x_1 \sin(w)))
\end{aligned}$$

The car parameters c_1, c_2 and the constant speed c_3 of the car can be found in table I.

TABLE I
Parameter values

cognitive car			other car		
c_1	160	$\frac{m}{s^2 \cdot rad}$	c_4	10	$\frac{m}{s^2}$
c_2	53	$\frac{1}{s^2 \cdot rad}$	c_5	60	$\frac{m}{s}$
c_3	15	$\frac{m}{s}$	c_6	15	$\frac{m}{s}$
			p_1, p_2, p_3, p_4	0.5	–

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

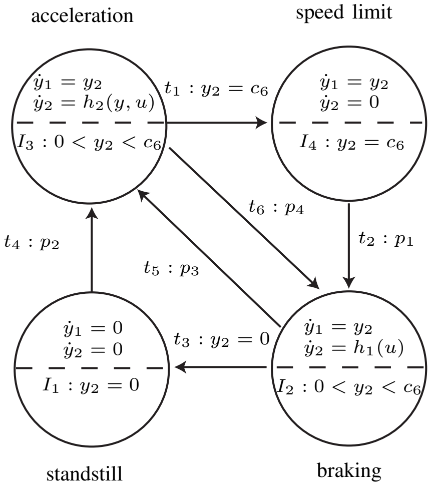


Fig. 7. Other car model

$$h_1(u) = -c_4 u, u \in [0, 1]$$

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

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 $p_{acc}, p_{brake}, p_{sl}, p_{ss}$ refer to the probability that the discrete state is in *acceleration*(acc), *brake*(brake), *speedlimit*(sl) or *standstill*(ss) mode at $t = 0$. The state space discretization is summarized in table III

TABLE II
SET OF INITIAL CONDITIONS

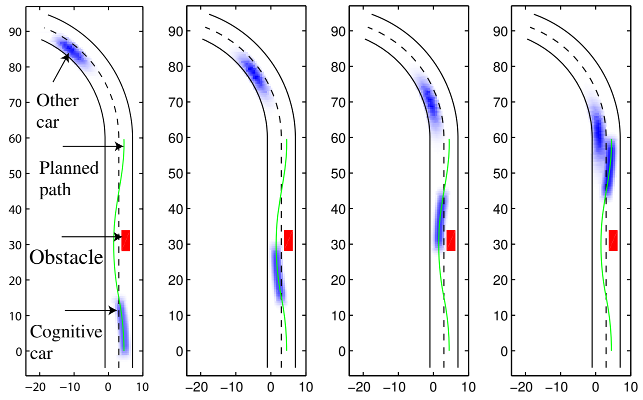
cognitive car			other car		
x_1	$[-0.3, 0.3]$	m	y_1	$[90, 95]$	m
x_2	$[-0.3, 0.3]$	m	y_2	$[8, 10]$	m/s
x_3	$[-0.05, 0.05]$	rad	p_{acc}, p_{brake}	0.5	–
x_4	$[-0.2, 0.2]$	rad/s	p_{sl}, p_{ss}	0	–

TABLE III
DISCRETIZATION PARAMETERS

variable	segment length	segments
cognitive car: 512 cells		
x_1	$0.5[m]$	4
x_2	$0.5[m]$	4
x_3	$0.05[rad]$	8
x_4	$0.4[rad/s]$	4
other car: 1500 cells		
y_1	$1[m]$	100
y_2	$1[m/s]$	15

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 whole car and not to its center of mass only (car length: 4m, car width: 2m).



(a) t=0-1 sec (b) t=1-2 sec (c) t=2-3 sec (d) t=3-4 sec

Fig. 8. Reachable sets of the traffic scenario

VIII. CONCLUSION

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).

REFERENCES

[1] I. Ulrich and J. Borenstein, "Vfh*: Local obstacle avoidance with look-ahead verification," in *In Proc. of the International Conference on Robotics and Automation*, 2000, pp. 2505–2511.

[2] C. Laugier, S. Petti, D. Vasquez, M. Yguel, T. Fraichard, and O. Aycard, "Steps towards safe navigation in open and dynamic environments," in *Autonomous Navigation in Dynamic Environments: Models and Algorithms*. Springer, 2006.

[3] J. van den Berg, D. Ferguson, and J. Kuffner, "Anytime path planning and replanning in dynamic environments," in *Proc. of the International Conference on Robotics and Automation*, 2006, pp. 2366–2371.

[4] A. E. Broadhurst, S. Baker, and T. Kanade, "A prediction and planning framework for road safety analysis, obstacle avoidance and driver information," in *Proc. of the 11th World Congress on Intelligent Transportation Systems*, October 2004.

[5] K. Lee and H. Peng, "Evaluation of automotive forward collision warning and collision avoidance algorithms," *Vehicle System Dynamics*, vol. 43, no. 10, pp. 735–751, 2005.

[6] S. Petti and T. Fraichard, "Safe motion planning in dynamic environments," in *Proc. of the Conference on Intelligent Robots and Systems*, 2005.

[7] G. Lafferiere, G. Pappas, and S. Yovine, "A new class of decidable hybrid systems," in *Hybrid Systems: Computation and Control*, ser. LNCS 1569. Springer, 1999, pp. 137–151.

[8] O. Botchkarev and S. Tripakis, "Verification of hybrid systems with linear differential inclusions using ellipsoidal approximations," in *Hybrid Systems - Computation and Control*, ser. LNCS 1790. Springer, 2000, pp. 73–88.

[9] A. Chutinan and B. H. Krogh, "Computational techniques for hybrid system verification," in *IEEE Transactions on Automatic Control*, vol. 48, no. 1, 2003, pp. 64–75.

[10] O. Stursberg and B. H. Krogh, "Efficient representation and computation of reachable sets for hybrid systems," in *Hybrid Systems - Computation and Control*, ser. LNCS 2623. Springer, 2003, pp. 482–497.

[11] A. Girard, "Reachability of uncertain linear systems using zonotopes," in *Hybrid Systems : Computation and Control*, vol. 3414, 2005, pp. 291–305.

[12] J. Lunze and B. Nixdorf, "Representation of hybrid systems by means of stochastic automata," *Mathematical and Computer Modeling of Dynamical Systems*, vol. 4, pp. 383–422, 2001.

[13] X. Koutsoukos and D. Riley, "Computational methods for reachability analysis of stochastic hybrid systems," in *Hybrid Systems: Computation and Control*, 2006, pp. 377–391.

[14] M. Prandini and J. Hu, "A stochastic approximation method for reachability computations," Final Report of the Hybridge Project, pp. 115–147, 2005.

[15] M. Althoff, O. Stursberg, and M. Buss, "Safety assessment of autonomous cars using verification techniques," in *to appear in the Proc. of the American Control Conference*, 2007.

[16] L. Jaulin, M. Kieffer, and O. Didrit, *Applied Interval Analysis*. Springer, 2006.

[17] E. Hansen, W. Walster, and G. W. Walster, *Global Optimization Using Interval Analysis*. CRC Press, 2003.

[18] H. Delingette, M. Hebert, and K. Ikeuchi, "Trajectory generation with curvature constraint based on energy minimization," in *International Workshop on Intelligent Robots and Systems*, 1991, pp. 206–211.

[19] S. Brennan and A. Alleyne, "Dimensionless robust control with application to vehicles," *IEEE Transactions on Control Systems Technology*, vol. 13, no. 4, pp. 624–630, 2005.