Model-Based Probabilistic Collision Detection in Autonomous Driving

Matthias Althoff, Olaf Stursberg, *Member, IEEE,* and Martin Buss, *Member, IEEE,*

***Abstract*—Safety of planned paths of autonomous cars with respect to the movement of other traffic participants is considered. Thereto, the stochastic occupancy of the road by other vehicles is predicted. The prediction considers uncertainties originating from the measurements and the possible behaviors of other traffic participants. In addition, the interaction of traffic participants as well as the limitation of driving maneuvers due to the road geometry is considered. The result of the presented approach is the probability of a crash for a specific trajectory of the autonomous car. The presented approach is efficient as most intensive computations are performed offline, resulting in a lean online algorithm for real-time application.**

***Index Terms*—Safety assessment, threat level, autonomous cars, behavior prediction, interaction, Markov chains, uncertain models, reachable sets.**

I. Introduction

In the past years, new driver assistant systems have successfully emerged into the market as they compensate shortcomings of human drivers, such as inevitable reaction times for emergency brakes or deficiencies for vehicle stabilization. However, the cognitive capabilities of humans are excellent and valuable in unexpected driving situations as well as for the correct interpretation of a traffic situation. Consequently, the next step towards intelligent vehicles is the implementation of basic cognitive capabilities as it is tried in many research projects, among them the collaborative research center *Cognitive Automobiles* [29], in which this work has been carried out.

One of the human abilities in traffic is the estimation of the threat level of planned actions. Maneuvers, such as overtaking, lane changing, or intersection crossing, are mainly evaluated according to a ratio of risk and time efficiency. As the consequences of a started maneuver affect the future development of a traffic situation, prediction is inevitable in order to assess the danger of the taken action. In this work, a technical realization for the safety assessment of driving maneuvers of autonomous cars is proposed, which is based on the prediction of traffic situations. In contrast to predictive approaches, non-predictive methods are based on the record and evaluation of traffic situations that have resulted in dangerous situations, see e.g. [1].

Behavior prediction of human drivers has been widely investigated, e.g. in [21], [33], [25]. As reported in the literature, human driver prediction within the *ego vehicle* (i.e. the vehicle for which the safety assessment is performed) can be conducted with the help of learning mechanisms such as neural networks or filter techniques as e.g. Kalman filters. The same approaches can be applied for traffic scenes that are observed from a fixed location, such as intersections, allowing to learn typical behavior patterns of the specific scene, see e.g. [17], [30].

Another possibility to predict traffic situations is to simulate single behaviors of traffic participants [10], [15], resulting in measures like *time to collision* or *predicted minimum distance*. Due to the efficiency of single simulations, these approaches are already widely implemented in cars. However, single simulations do not consider uncertainties in the measurements and actions of other traffic participants, which may lead to unsatisfying collision predictions [20]. A more sophisticated threat assessment considers multiple simulations of other vehicles, considering different initial states and changes in their inputs (steering angle and acceleration). These so called Monte-Carlo methods have been studied in [11], [12], [14].

Another method to investigate possible behaviors of traffic participants is reachability analysis, see e.g. [9], [3], [7]. For a given set of initial states and disturbance values, a reachable set contains all possible states that the system trajectories can evolve into. If for a traffic scenario, the reachable positions of the ego vehicle do not intersect the reachable positions of another vehicle, these vehicles cannot crash. Reachable sets for vehicles have been investigated in [27], [31]. It has been shown that planned paths of autonomous vehicles are too often evaluated as unsafe by this method, because the reachable sets of other vehicles rapidly cover all positions the autonomous vehicle could possibly move to. For this reason, the reachable sets are enhanced by stochastic information to so called *stochastic reachable sets* in previous works of the authors [4], [2], [6]. The stochastic information allows not only to check if a planned path of the ego vehicle may result in a crash, but also with which probability. Consequently, possible driving strategies of autonomous cars can be evaluated according to their safety. Stochastic reachable sets have also been investigated for air traffic safety [16] and fault diagnosis [28], [23].

It is further emphasized that traffic prediction has to consider interaction between vehicles. Interaction between vehicles has been widely studied in microscopic traffic simulations [24] and within Monte-Carlo techniques [11], [12], [14]. The problem one faces with interaction is that the simulations result in a fast growing tree of possible situations to be considered. If only discrete actions (e.g. lane change) at discrete points of time are taken into account, the

All authors are with the Institute of Automatic Control Engineering (LSR), Technische Universität München, 80290 München, Germany. {althoff,stursberg,mb}@tum.de

Manuscript received 22. September 2008; revised xxx.

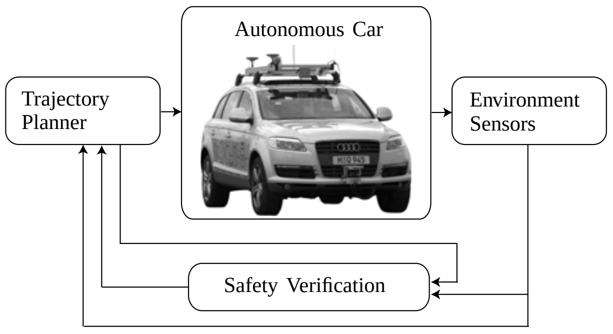


Fig. 1. Conception of the safety assessment.

computational complexity is $O(\mu^{\rho\cdot\nu})$, where μ is the number of possible actions, ρ is the number of time steps of the prediction and ν is the number of traffic participants, see [13]. This worst case complexity, however, can be reduced by discarding impossible or unrealistic behaviors. The consideration of interaction also causes additional complexity for the computation of stochastic reachable sets, however, the incorporated interaction mechanism of the presented approach is efficient in the sense that the computations can be applied online using a desktop PC.

II. Motivation and Problem Statement

Clearly, autonomous driving requires a control loop containing a perception and planner module, see Fig. 1. The perception module detects traffic situations and extracts relevant information, such as the road geometry as well as static and dynamic obstacles. In order to fulfill the driving task, the planner module computes trajectories that the autonomous car is tracking with the use of low level controllers. A major constraint for the trajectory planner is that the generated trajectories have to be safe, i.e. no static and dynamic obstacles must be hit. The task of circumventing static obstacles can be ensured by checking whether the static obstacle intersects with the vehicle body of the autonomous car following the planned path. For dynamic obstacles, the safety assessment is much more intricate as their future actions are unknown. For this reason, sets of possible behaviors of other traffic participants are considered, which are checked with the planned path of the autonomous car in a dedicated safety verification module (see Fig. 1). Paths that fulfill the safety requirements are executed and are conservatively replanned otherwise, e.g. by braking the car.

The safety verification module which is described in this work requires the description of a traffic situation containing the following information gathered by the perception module:

- the planned trajectory of the autonomous car,
- the geometric description of the relevant road sections,
- the position and geometry of static obstacles,
- as well as the position, velocity, and classification of dynamic obstacles.

Static obstacles are a special case of dynamic obstacles with zero velocity, and for that reason, the discussion is continued for dynamic obstacles only. The classification groups the

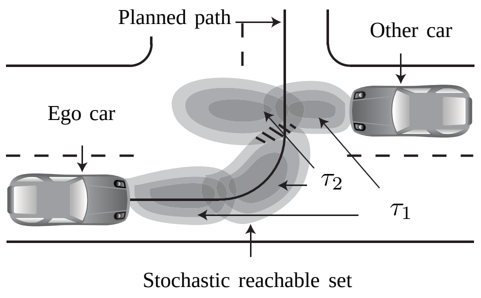


Fig. 2. Stochastic reachable sets of traffic participants.

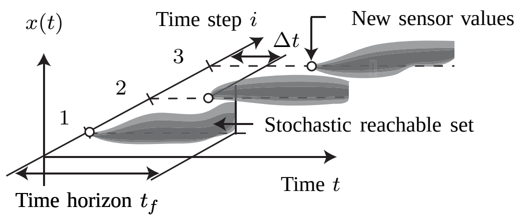


Fig. 3. Repetitive computation of reachable sets.

dynamic obstacles ($\hat{=}$ traffic participants) into *cars*, *trucks*, *motorbikes*, *bicycles*, and *pedestrians*. As the measurement of positions and velocities of other traffic participants is uncertain, the presented approach allows the measured data to be specified by a probability distribution. However, the minimum requirement is that all relevant traffic participants are detected at all. The region where the probability density is non-zero is referred to as the initial set. Given this set, the future set of positions possibly occupied by the traffic participant is denoted *reachable set*. Analogously, given the initial probability distribution of a traffic participant, the future probability distribution is also referred to as the *stochastic reachable set*.

Reachable sets of other traffic participants allow to guarantee the safety of the planned trajectory for a prediction horizon t_f if they do not intersect the reachable set of the ego car within the specified horizon. For the case of a possible crash, the probability distribution within the reachable set is used to determine the probability of the crash. This is illustrated in Fig. 2, where stochastic reachable sets are shown for the time intervals $\tau_1 = [0, t_1]$, $\tau_2 = [t_1, t_2]$ (dark color indicates high probability density). Within the time interval τ_1, a crash between both cars is impossible, while for the second time interval τ_2, the crash probability is non-zero. It is obvious that the computation of the crash probability has to be faster than real-time for online application. In order to update the crash probability prediction after a time interval Δt based on new sensor values, its computation has to be faster than real-time by a factor of $t_f/\Delta t$. This is illustrated in Fig. 3 for the reachable set of a single variable $x(t)$.

III. Modeling of Traffic Participants

This work focuses on the safety assessment of autonomous cars driving on a road network, i.e. the motion of traffic participants is constrained along designated roads. On that account, the possible paths of traffic participants are determined by the finite set of decisions {*left turn*, *right turn*, *go straight*}. The extension to multi-lane roads with the additional actions {*left lane change*, *right lane change*} is subject to future work. The deviation along these major paths is modeled by a piecewise constant probability density function $f(\delta)$, where δ is the lateral deviation from a driving path. Possible driving paths of a road network section, as well as the deviation probability distribution $f(\delta)$ are shown in Fig. 4. The deviation probability can be adjusted to different classes of traffic participants: Bicycle drivers are more likely to be found close to the curb, whereas cars and trucks are driving more likely in the center of a lane. For unstructured environments, such as parking spaces or pedestrians on a square, the motion of vehicles/people cannot be described along paths. For these kinds of scenarios, the approach presented in [26] is suggested, which uses the same mathematical principles as presented in this work.

The longitudinal probability distribution of the position of the vehicles is obtained by a dynamic model. After denoting the position of the volumetric center of the vehicles along a path with s, the velocity with v, and the absolute acceleration with a, the longitudinal dynamics for the driving input u can be described as follows:

$$\dot{s} = v, \quad \dot{v} = \begin{cases} c_1 \cdot (1 - (v/c_2)^2) \cdot u, & u > 0 \\ c_1 \cdot u, & u \leq 0 \\ 0, & v \leq 0 \end{cases} \tag{1}$$

subject to the constraint

$$a \leq \bar{a}^{max}, \text{ where } a = \sqrt{a_N^2 + a_T^2},\ a_N = v^2/\rho(s),\ a_T = \dot{v}. \tag{2}$$

The constant c_1 models the maximum possible acceleration due to tire friction and c_2 the top speed - these constants are chosen according to the specific properties of the different classes of traffic participants. The acceleration input varies from $[-1, 1]$, where -1 represents full braking and 1 represents full acceleration. Backwards driving on a lane is not considered, see (1) ($\dot{v} = 0,\ v \leq 0$). The function $\rho(s)$ maps the path coordinate s to the radius of curvature of the path and a_N, a_T is the normal and tangential acceleration respectively. The constraint in (2) models that the tire friction of a vehicle only allows a limited absolute acceleration $\bar{a}^{max}$ (Kamm's circle). In case, a traffic participant is violating the traffic regulations, such as driving on the wrong lane, the approach for unstructured motion in [26] is applied for this specific traffic participant. The dynamic equations in (1) are chosen exemplarily and can be easily exchanged against a different set of equations. Based on the dynamics specification, the longitudinal probability distribution $f(s)$ is obtained. Assuming that the lateral distribution is independent from the longitudinal one, the overall probability distribution is computed as $f(s, \delta) = f(s) \cdot f(\delta)$, see also Fig. 4. Thus, the probability distribution is described in a curved, path-aligned coordinate system, as also used in e.g. [14].

It is emphasized that the lateral and the longitudinal distribution $f(\delta)$ and $f(s)$ refer to the volumetric center of the vehicles. However, for visualization reasons, all figures in this work show the vehicle body density which is taking the vehicle size into account, see Fig. 5.

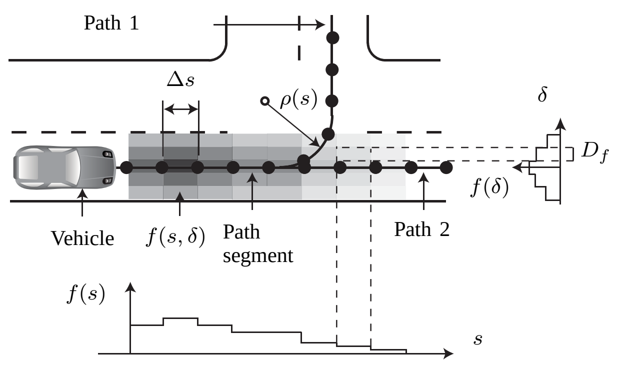


Fig. 4. Probability distribution of the position of a vehicle along a path-aligned coordinate system.

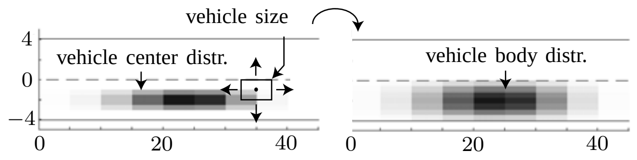


Fig. 5. Probability distribution of the vehicle center and the vehicle body.

IV. Reachable Sets of Traffic Participants

This section deals with the computation of reachable sets for traffic participants (no stochastic information). Given the dynamics of a traffic participant as $\dot{x} = f^{TP}(x(t), u(t))$, where $x \in \mathbb{R}^n$ is the state and $u \in U \subset \mathbb{R}^m$ is a Lipschitz continuous input constrained by the set U, the exact reachable set $R^e(r)$ at time $t = r$ can be defined as:

$$R^e(r) = \Big\{ x(r) \big| x(r) = x(0) + \int_0^r f^{TP}(x(\tau), u(\tau)) d\tau, \\ x(0) \in X_0, u(\tau) \in U \Big\}.$$

In general, the exact reachable set of a system cannot be computed [19]. However, one can always compute an over-approximation, which is denoted $R(r) \supseteq R^e(r)$. The over-approximated reachable set for a time interval is defined as $R([0, r]) := \bigcup_{t \in [0,r]} R(t)$.

In this work, over-approximations of reachable sets of nonlinear dynamic systems are computed as presented in [5]. An example of the over-approximated reachable set of (1) for $u \in [0.5, 1]$ and $t \in [0, 2]$ sec is given in Fig. 6 for two different initial sets. Additionally, sample trajectories starting from the initial set are shown, where the states at times $k \cdot \Delta t^*$, $k = 0 \ldots 4, \Delta t^* = 0.5$ sec are marked by a circle. If one is only interested in the reachable interval of the position and velocity of a vehicle driving along a straight path, the following special case can be formulated:

Proposition 1: Given is a vehicle driving along a straight path with dynamics subject to (1) and the initial condition $x(0) \in X(0) = S(0) \times V(0)$ where $S(0) = [\underline{s}(0), \overline{s}(0)]$ and $V(0) = [\underline{v}(0), \overline{v}(0)]$ are the position and velocity intervals.

The reachable, two-dimensional interval $X(t) = [\underline{x}(t), \overline{x}(t)]$ of position and velocity is given by:

$$\underline{x}(t) = \underline{x}(0) + \int_0^t f^{TP}(x(\tau), u(\tau)) d\tau, \quad u(\tau) = -1$$
$$\overline{x}(t) = \overline{x}(0) + \int_0^t f^{TP}(x(\tau), u(\tau)) d\tau, \quad u(\tau) = 1. \quad \square$$

The proof is omitted, as it is clear that the greatest position and velocity is reached when the vehicle starts with the greatest initial position and velocity within the initial set, under full acceleration. The analogous argumentation holds for the lowest position and velocity. Note that this argumentation is only applicable if there exists an initial state that jointly contains the maximum initial position and velocity. This is always the case when the initial set is a two-dimensional interval, which is in contrast to the left example of Fig. 6, for which proposition 1 is not applicable. In the left example, the maximum reachable position at different times is reached from trajectories starting from different initial states. However, if one is only interested in the reachable position – and the initial set is a two-dimensional interval, as shown in the right example of Fig. 6, the result of proposition 1 results in the exact reachable interval of the position coordinate.

For the case of a curved path, one has to consider the tire friction constraint in (2). For a given radius profile $\rho(s)$, the minimum and maximum admissible input $\underline{u}(s)$ and $\overline{u}(s)$ can be obtained as presented e.g. in [32]. By changing $u(\tau) = -1$ to $u(\tau) = \underline{u}(s(\tau))$ and $u(\tau) = 1$ to $u(\tau) = \overline{u}(s(\tau))$ in proposition 1, one can compute the reachable positions for a curved road. Speed limits on a road can be handled by cutting off the previously computed speed profile $v(s)$ at v^{max} by assigning $\overline{u}(s) = 0$ if $v(s) > v^{max}$.

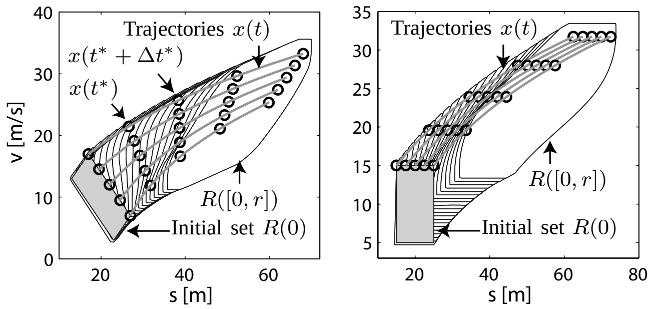


Fig. 6. Reachable sets of a vehicle for different initial sets.

V. Stochastic Reachable Sets of Traffic Participants

As stated in the previous section, the exact computation of reachable sets is only possible for a limited class of systems [19]. Hence, it is obvious that for the more general problem of stochastic reachable sets, one has to apply (conservative) approximation techniques, too. One of the most frequently used techniques is to approximate stochastic processes by Markov chains. The techniques for the abstraction to Markov chains are manifold, where many of them couple the time interval between the updates of the probability distribution with the accuracy of the abstraction, see e.g. [16], [18]. However, this coupling is unfavorable in terms of real-time applicability, as a required approximation accuracy may lead to short update times of the probabilities – and consequently to too many update iterations that cannot be handled in real-time. For this reason, the update intervals and the approximation accuracy is decoupled as in [22], [28]. The presented abstraction method for nonlinear systems with unknown inputs is conservative, which means that the reachable set of the Markov chain is an over-approximation of the reachable set of the original stochastic process. Note that in contrast to the reachable set, the probability distribution is an approximation as in case of an over-approximation its integral would be greater than one and thus destroys an axiom of the probability calculus.

The abstraction of the continuous dynamics of the traffic participants to Markov chains is computed offline. During online execution of the algorithm, for each traffic participant, a Markov chain is instantiated. The interaction between traffic participants is established by influencing the acceleration probabilities of traffic participants by the state (position, velocity) and the acceleration of other traffic participants. This means, that the traffic participants of a traffic situation are not combined in a single Markov chain, but represented by separate Markov chains. Thus, the complexity of the algorithm is N times the worst-case complexity of the computation of a single traffic participant and N is the number of considered traffic participants.

It is also emphasized, that the abstraction to Markov chains discussed from Sec.V-A to Sec.V-D is only performed for other vehicles, as the motion of the ego vehicle is planned within the vehicle and thus known. The ego vehicle is discussed separately in Sec. V-E.

A. Abstraction by Markov Chains

A Markov chain is a stochastic dynamic system with discrete states $z \in \mathbb{N}^+$. There are discrete time and continuous time Markov chains. In this work, discrete time Markov chains are used, such that $t \in \{t_1, t_2, \ldots, t_f\}$ where t_f is the prediction horizon and $t_{k+1} - t_k = T \in \mathbb{R}^+$ is the time step increment. In Markov chains the current state is not exactly known but probabilities $p_i = P(z = i)$ describe that the system is in state $z = i$, leading to a probability vector p. By definition, the probability vector for the next time step t_{k+1} is a linear combination of the probability vector of the previous time step t_k:

$$p(t_{k+1}) = \Phi \cdot p(t_k), \tag{3}$$

where Φ is referred to as the transition matrix. The generation of a Markov chain model can be divided into two steps: First, the state space of the original continuous system is discretized into cells representing the discrete states. Second, the transition probabilities from one cell to another cell have to be determined, which are stored in the transition matrix of the Markov chain.

1) Discretization of the State and Input Space: In this work, a region $X \subset \mathbb{R}^2$ of the continuous state space of (1) is discretized in orthogonal cells of equal size, resulting in

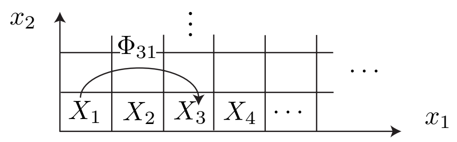


Fig. 7. Discretization of the state space.

rectangular cells with the coordinates $x_1 := s$ and $x_2 := v$, see Fig. 7. This discretization allows to describe the cells X_i (with index i referring to the represented discrete state) by the two dimensional interval $X_i =]\underline{x}_i, \overline{x}_i]$, $\underline{x}_i, \overline{x}_i \in \mathbb{R}^2$ and $X = \bigcup X_i$. In an analogous way, a region $U \subset \mathbb{R}$ of the input space of (1) is discretized into intervals U^α such that $U = \bigcup U^\alpha$. The index α refers to the value of the discrete input, which is denoted by y. In order to distinguish between indices referring to states or inputs, state indices are subscripted and Latin, where input indices are superscripted and Greek.

2) Transition Probabilities of the Markov Chain: The transition probabilities store the probabilities that the system state changes from i to j: $\Phi_{ji} = P(z(t_{k+1}) = j | z(t_k) = i)$. In this work, the transition probabilities depend on the discrete input α, too. For this reason, a different transition probability matrix Φ^α is computed for each discrete input α. The value of a transition probability Φ^α_{ji}, is obtained by the reachable set $R^\alpha_i(T)$ starting from $x(0) \in X_i$ under the effect of the input $u \in U^\alpha$, where the cells X_i and U^α represent the corresponding discrete state and input of Φ^α_{ji}. Note that the time T of the reachable set is equal to the time step increment T of the Markov chain. The probability of reaching cell j is computed as the volumetric fraction of the reachable set intersecting with the cell X_j:

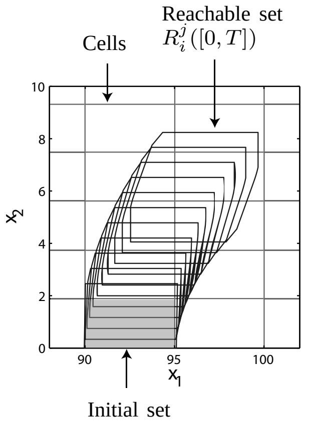
$$\Phi^\alpha_{ji}(T) = \frac{V(R^\alpha_i(T) \cap X_j)}{V(R^\alpha_i(T))}, \tag{4}$$

where $V()$ is an operator returning the volume. As soon as a cell is reached by $R^\alpha_i(T)$, the probability that the cell can be reached is non-zero. As only the probability of reaching a cell is stored, the information is lost which part of the cell can be reached. For this reason, the reachable cells (cells with non-zero probability) of the Markov chain over-approximate each corresponding reachable set $R^\alpha_i(T)$. The probability values itself are approximative (in contrast to over-approximative reachable sets) as an equal distribution of the system state within the reachable sets is assumed in (4).

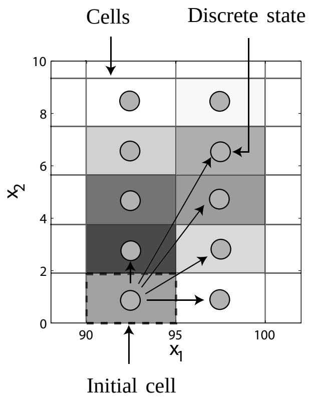
In an analogous way, one can compute the transition probabilities $\Phi^\alpha_{ji}([0,T])$ for $t \in [0,T]$ by substituting $R^\alpha_i(T)$ with $R^\alpha_i([0,T])$ which is the reachable set for $t \in [0,T]$. Due to computational reasons [5], the reachable set $R^\alpha_i([0,T])$ is obtained from $\bigcup_{l=0}^{T/r-1} R^\alpha_i([l \cdot r, (l+1) \cdot r])$ where T is a multiple of r, such that

$$\Phi^\alpha_{ji}([0,T]) = \frac{r}{T} \sum_{l=0}^{T/r-1} \Phi^\alpha_{ji}([l \cdot r, (l+1) \cdot r]).$$

Note that the transition probabilities in $\Phi^\alpha_{ji}(T)$ and $\Phi^\alpha_{ji}([0,T])$ are computed offline, such that computationally expensive operations are performed beforehand. The reachable set of $R^\alpha_i([0,T])$ is exemplarily shown for the case of $T/r = 10$ in Fig. 8(a). The corresponding stochastic reachable set, resulting from $\Phi^\alpha_{ji}([0,T]) \cdot p(0)$ with $p_m(0) = 0$ for $m \neq i$ and $p_i(0) = 1$ ($i \hat{=}$ Initial cell) is illustrated in Fig. 8(b). The circles symbolize the discrete states, which are assigned to the corresponding cells. A transition to a cell is the more likely, the darker the color of the cell is.



(a) Reachable set for a time interval.



(b) Reachable cells for a time interval.

Fig. 8. Reachable set of the original system and the corresponding stochastic reachable set of the abstracting Markov chain.

B. Computing Stochastic Reachable Sets using Markov Chains

After the discretization of the state space, the stochastic reachable set can be fully described by the probability vector p of the Markov chain. For a single Markov chain, the evolution of the probability vector can be computed as in (3). However, in this work, Markov chains for different discrete inputs α as well as for the time point and time interval solution are computed. Hence, (3) has to be extended. The first extension is, that the probability distribution within a time interval is computed based on the distribution at a time point (for a given input α). Thus, the probability distribution at certain time points serves as a support for the time interval computations:

$$\begin{aligned} p(t_{k+1}) &= \Phi^\alpha(T) \cdot p(t_k), \\ p([t_k, t_{k+1}]) &= \Phi^\alpha([0,T]) \cdot p(t_k). \end{aligned} \tag{5}$$

This is justified by the fact that the transition probability matrix $\Phi^\alpha([0,T])$ is also computed based on the initial cell at the beginning of the considered time interval. Note that the indices of the matrix Φ^α and the vector $p(t_k)$ are neglected as commonly done in linear algebra. In (5), the input α is known and globally applied to all states i of the Markov chain. For the next extension, the conditional probability $q^\alpha_i = P(y = \alpha | z = i)$ is defined. The joint probability of the state and input is abbreviated as $p^\alpha_i := P(z = i, y = \alpha) = P(y = \alpha | z = i) \cdot P(z = i) = q^\alpha_i \cdot p_i$. It is also clear, that $p_i = \sum_\alpha p^\alpha_i$ where $\sum_\alpha$ denotes the sum over all possible values of α. As a novelty compared to earlier work in [6], the probability distribution of the input is dynamically changed by another Markov chain with transition matrix Γ_i, depending on the system state i. This allows a more accurate modeling of driver behavior by considering how frequently and how intense the acceleration command is changed. The input probability distribution changes instantly at the discrete times t_k, which is indicated by a prime:

$$q_i^\beta(t_k)' = \Gamma_i^{\beta\alpha} \cdot q_i^\alpha(t_k) \rightarrow p_i^\beta(t_k)' = \Gamma_i^{\beta\alpha} \cdot p_i^\alpha(t_k). \quad (6)$$

The mapping has to be performed for each discrete state value $z = i$, unless $p_i^\alpha = 0, \forall\alpha$. For a better distinction to Φ, the indices of Γ describing the input transitions are superscripted – as also done for the input indices of the probability p. The introduction of the joint probability p_i^α requires a slight modification of (5), where the joint probabilities p_i^α have to be updated for each input value α instead of only a single input value:

$$p^\alpha(t_{k+1}) = \Phi^\alpha(T) \cdot p^\alpha(t_k),$$
$$p^\alpha([t_k, t_{k+1}]) = \Phi^\alpha([0,T]) \cdot p^\alpha(t_k).$$

In contrast to the computation of the transition matrices Φ^α for the states, the transition matrices Γ_i for the inputs cannot be computed based on a dynamic model. This is because the input/driving commands to other vehicles are provided by humans or complex computer systems (when vehicles will drive autonomously in the future), for which the system dynamics is unknown. As a consequence, the transition matrices Γ_i have to be learned by observation or set by a combination of simulations and heuristics, where the latter is used in this work. The input transition matrix is composed of an input dynamics matrix Ψ and a priority vector λ, where the prioritization results from many aspects, such as speed limits or interaction with other vehicles. The input dynamics matrix Ψ and the priority vector λ are combined to the following:

$$\begin{aligned} \Gamma_i^{\beta\alpha} &= \mathtt{norm}(\hat{\Gamma}_i^{\beta\alpha}) := \frac{\hat{\Gamma}_i^{\beta\alpha}}{\sum_\beta \hat{\Gamma}_i^{\beta\alpha}}, \\ \hat{\Gamma}_i^{\beta\alpha} &= \mathtt{diag}(\lambda_i^\beta) \cdot \Psi^{\beta\alpha}, \quad \forall i : \sum_\beta \lambda_i^\beta = 1,\ 0 \leq \lambda_i^\beta \leq 1 \end{aligned} \quad (7)$$

The intermediate result $\hat{\Gamma}_i^{\beta\alpha}$ is normalized by the sums of the columns such that they are summing up to 1, in order to ensure that the sum of the probability vector stays 1 after the multiplication with Γ, see (6). The state dependence is modeled by the priority vector λ, while the input dynamics matrix Ψ is independent of the state. The above formula has the following special cases:

- $\lambda_i^\beta = 0$: Regardless of the input dynamics matrix Ψ, the input β of state i is prohibited ($q_i^\beta = 0$) as the corresponding row of Γ becomes 0.
- $\lambda_i^\beta = \frac{1}{\kappa}, \forall i, \beta$ (κ is the number of inputs): No input is prioritized, such that $\Gamma_i = \Psi$.
- $\Psi = I$ (I is the identity matrix): $\Gamma_i = I$, regardless of λ, such that the input probability is unchanged.
- $\Psi = \frac{1}{\kappa} O$ (O is a matrix of ones): The multiplication $\Gamma_i \cdot q_i$ results into $\mathtt{norm}\big(\mathtt{diag}(\lambda_i)\frac{1}{\kappa}O\big) \cdot q_i = \lambda_i$. Thus, a certain input probability distribution $q_i = \lambda_i$ is enforced, regardless of the probability distribution of the previous time step.

In order to discuss the effect of Ψ separately, λ is set to $\lambda_i^\beta = \frac{1}{\kappa}, \forall i, \beta$, such that $\Gamma_i = \Psi$, so that the input dynamics matrix Ψ specifies with which probability the input is changed from input α to input β, see (6). These transition probabilities are set according to the heuristics, that the bigger the change of the input[1], the more unlikely is this change. A transition matrix that considers this aspect and further contains the special cases of $\Psi = I$ and $\Psi = \frac{1}{\kappa} O$ is:

$$\Psi^{\beta\alpha}(\gamma) = \mathtt{norm}\big(\hat{\Psi}^{\beta\alpha}(\gamma)\big), \quad \hat{\Psi}^{\beta\alpha}(\gamma) = \frac{1}{(\beta-\alpha)^2 + \gamma}.$$

The parameter γ allows to gradually interpolate the extreme cases $\Psi = I$ and $\Psi = \frac{1}{\kappa} O$, which are represented by the limit $\lim_{\gamma\rightarrow 0} \Psi(\gamma) = I$ and the other limit $\lim_{\gamma\rightarrow\infty} \Psi(\gamma) = \frac{1}{\kappa} O$. Informally speaking, a low value of γ models drivers that do not change their input often, whereas a high value models drivers, that change their input often. The higher the value of γ, the faster converges the initial input probability distribution to a steady state distribution, which can be changed by the priority vector λ. This is illustrated in Fig. 9 for 3 inputs, where the high input numbers represent high positive acceleration, such that the first input $y = 1$ represents full braking and the last input $y = 3$ full acceleration. The initial probabilities are set to $P(y = 1) = 0$, $P(y = 2) = 0.8$, $P(y = 3) = 0.2$ and the probabilities converge to $\frac{1}{3}$ as no prioritization is specified.

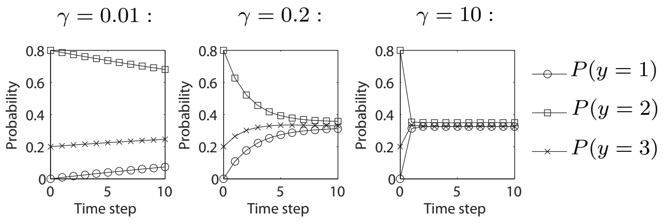


Fig. 9. Input evolution for $\gamma = 0.01, 0.2, 10$.

The effect of the priority vector λ is discussed for $\lim_{\gamma\rightarrow\infty} \Psi(\gamma)$, such that $\lambda_i^\alpha = q_i^\alpha = P(y = \alpha | z = i)$, which is also the setting used in [2], [6]. The state dependent priority vectors control the vehicles in a way, such that the constraints due to other traffic participants or the road geometry are met, as it is done for controlled Markov chains, see e.g. [8].

In order to break down the effects of the priority vector λ, two different aspects are considered. One of them is the characteristic probability distribution m in the absence of constraints, such that $m_i^\alpha = \lambda_i^\alpha$ which is modeled as independent of the state ($m_i^\alpha = m_j^\alpha \forall i, j$). However, not all inputs meet constraints (are drivable), such that the event C of constraint satisfaction is introduced. Due to the uncertain modeling of traffic situations, the event of constraint satisfaction C is subject to probability and stored in the constraint vector $c_i^\alpha := P(C | z = i, y = \alpha)$. As at least the constraints have to be met, the probability vector c_i^α serves as an upper bound for the priority vector λ:

$$\lambda_i^\beta = \begin{cases} m_i^\beta, \text{ if } m_i^\beta \leq c_i^\beta \\ c_i^\beta, \text{ otherwise.} \rightarrow m_i^{\beta-1} := m_i^{\beta-1} + m_i^\beta - c_i^\beta \end{cases}$$

With other words, m_i^β is cut off at c_i^β and the cut-off probability is added to the next lower acceleration interval, see

[1]As the discrete inputs are numbered in increasing order according to the acceleration intervals, the difference between the input numbers is a measure for the change of the acceleration interval.

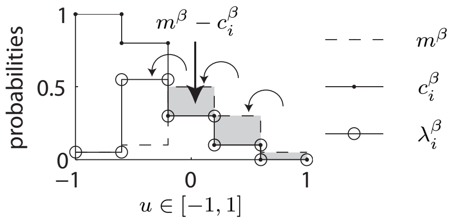


Fig. 10. Combining a probability distribution with a probabilistic constraint.

Fig. 10. This is motivated by the fact that for the considered situations of vehicle and road following (Sec. V-C and V-D), drivers have to accelerate less or brake stronger in order to fulfill the safety constraints.

C. Interaction of Traffic Participants

In this subsection the computation of the constraint vector $c_i^\alpha = P(C|z=i, y=\alpha)$ with respect to the interaction of two vehicles driving on the same lane is presented. The state and input of the following vehicle A is denoted z^A, y^A and analogously z^B, y^B for the leading vehicle B. The constraint for this scenario is that the probability of the following vehicle crashing into the leading vehicle has to be less than ϵ, which is typically very low. Thereto, it is firstly checked if a certain initial situation results in a crash. As the probability distribution of vehicles is approximate (in contrast to the reachable set), the check for a crash under initial states and inputs chosen from the cells belonging to z^A, y^A, z^B and y^B is also approximately done for a single sample of possible initial states and inputs:

$$x^A(0) = \mathtt{center}(X_i^A),\ x^B(0) = \mathtt{center}(X_j^B),$$
$$u^A(0) = \mathtt{center}(U^{A^\alpha}),\ u^B(0) = \mathtt{center}(U^{B^\beta}),$$

where `center` returns the volumetric center of a set. The single simulation run approximately checking for a crash is performed as follows:

1) Simulate both vehicles for the time $\Delta t = \nu \cdot T$, $\nu \in \mathbb{N}^+$, starting from $x^A(0)$, $x^B(0)$ under the effect of u^A, u^B.
2) Simulate a sudden brake beginning at $t = \nu \cdot T$ of the leading and following vehicle until the following vehicle has stopped at $t = t_S$.
3) Check if the following vehicle has crashed into the leading vehicle for $t \in [0, t_S]$.

Events such as $z^A = i$ are abbreviated by index notation as $(z^A = i) \hat{=} z_i^A$ in the following in order to obtain a concise notation. The outcome of the simulation determines the conditional probability for satisfying the constraint that a crash occurs with probability less than ϵ which is motivated by driver inattentiveness:

$$P(C|z_i^A, z_j^B, y^{A^\alpha}, y^{B^\beta}, \Delta t = \nu \cdot T) = \begin{cases} 1, & \text{no crash simulated} \\ \epsilon, & \text{otherwise} \end{cases}$$

The conditional probabilities $P(C|z_i^A, z_j^B, y^{A^\alpha}, y^{B^\beta}, \Delta t = \nu \cdot T)$ can be obtained for different intervals $\Delta t = \nu \cdot T$ of constant acceleration. Long time intervals Δt model the behavior of foresighted drivers, who adjust their acceleration early to changes of other drivers. More sporty drivers change their acceleration in shorter time intervals Δt. The probability distribution for time intervals in which the acceleration interval is unchanged $P(\Delta t = \nu \cdot T)$ allows to compute

$$P(C|z_i^A, z_j^B, y^{A^\alpha}, y^{B^\beta}) = \sum_\nu P(C|z_i^A, z_j^B, y^{A^\alpha}, y^{B^\beta}, \Delta t = \nu \cdot T) \cdot P(\Delta t = \nu \cdot T).$$

Note that the probability $P(\Delta t = \nu \cdot T)$ is not obtained online by observation of other drivers, but set according an average distribution of all drivers. The conditional probabilities $P(C|z_i^A, z_j^B, y^{A^\alpha}, y^{B^\beta})$ are computed offline and stored in an $(c \times c)$-array of $(d \times d)$-matrices $\Theta_{ji}^{\alpha\beta}$ where c and d are the numbers of discrete inputs and states. Using the values in $\Theta_{ji}^{\alpha\beta}$, the constraint vector for vehicle interaction is computed online as:

Proposition 2: Under the assumption that $P(z_i^A, y^{A^\alpha})$ and $P(z_j^B, y^{B^\beta})$ are independent, one obtains

$$c_i^\alpha = \sum_{j,\beta} \Theta_{ji}^{\alpha\beta} p_j^{B^\beta}. \qquad \square$$

Proof: After defining the events $A = (z_i^A, y^{A^\alpha})$ and $B_j^\beta = (z_j^B, y^{B^\beta})$, one can write:

$$\begin{aligned} P(C, A) &= \sum_{j,\beta} P(C|A, B_j^\beta) P(A, B_j^\beta) \\ &\overset{\text{independence}}{=} \sum_{j,\beta} P(C|A, B_j^\beta) P(A) P(B_j^\beta) \\ \rightarrow c_i^\alpha &= P(C|A) = \frac{P(C,A)}{P(A)} = \sum_{j,\beta} P(C|A, B_j^\beta) P(B_j^\beta) \\ &= \sum_{j,\beta} \Theta_{ji}^{\alpha\beta} p_j^{B^\beta}. \end{aligned}$$

■

It is clear, that the independence assumption only approximates the joint probability $P(A, B_j^\beta) \approx P(A)P(B_j^\beta)$, however this assumption simplifies the computation and has produced reasonable results for numerical examples. In traffic situations with more than two vehicles on a lane, each vehicle is only interacting with the next nearest vehicle driving in front, such that the proposed algorithm is simply applied for each vehicle that follows another vehicle.

D. Behavior Restriction due to Road Geometry and Speed Limits

Besides the interaction with other vehicles, possible acceleration inputs are restricted due to speed limit and the road geometry in combination with limited tire friction. For these restrictions, the velocity profile resulting from the minimum and maximum possible acceleration with respect to (2) is introduced and denoted $\underline{v}(s)$ and $\overline{v}(s)$, respectively. An additional labeling with brackets indicates the maximum absolute acceleration, e.g. $\overline{v}(s)^{[0.5g]}$ is the fastest possible velocity profile for $a^{max} = 0.5g$ and g is the gravity constant. Exemplary

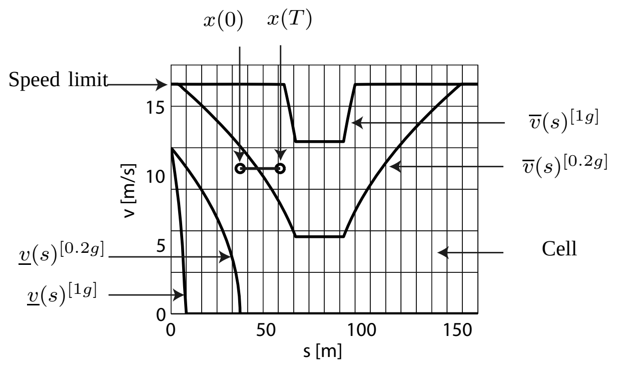


Fig. 11. Velocity profiles for two straights connected by a 90° curve (15.5 m radius) and a speed limit of $v^{max} = 16.7$ m/s $\hat{=}$60 km/h.

velocity profiles with a speed limit (possibly greater than the official speed limit to account for sporty drivers), are shown in Fig. 11.

The velocity profiles allow to compute the constraint vector $c_i^\alpha = P(C|z = i, y = \alpha)$ with respect to road geometry and speed limit, where the event C holds when the velocity is within the velocity profile bounds ($\underline{v}(s) \leq v \leq \overline{v}(s)$). Analogously to the vehicle interaction, the compliance of a constraint for state $z = i$, input $y = \alpha$ and given velocity bounds $\underline{v}(s)$ and $\overline{v}(s)$ is approximately checked by a single simulation run:

1) Simulate the vehicle for the time interval $\Delta t = T$, starting from $x(0) = \texttt{center}(X_i)$ under the effect of $u = \texttt{center}(U^\alpha)$.
2) Check whether the velocity is within the minimum and maximum velocity profile after one time increment T (see also Fig. 11):

$$\underline{v}(s(T))^{[a_d^{max}]} \leq v(T) \leq \overline{v}(s(T))^{[a_d^{max}]}. \qquad (8)$$

The constraint vector for road geometry and speed limit is then obtained as:

$$P(C|z_i, y^\alpha, a < a_d^{max}) = \begin{cases} 1, & \text{if (8) holds} \\ 0, & \text{otherwise} \end{cases}.$$

In contrast to the vehicle interaction, the constant ϵ for inattentiveness is not applied here as the physical constraint is either met or not. After introducing the probability distribution for applied accelerations $P(a < a_d^{max})$ among all drivers, where $0 < a_d^{max} \leq \bar{a}^{max}$ and $\bar{a}^{max}$ is the physically possible acceleration, the constraint vector is

$$c_i^\alpha = P(C|z_i, y^\alpha) = \sum_d P(C|z_i, y^\alpha, a < a_d^{max}) P(a < a_d^{max}).$$

It remains to combine the constraint vectors for vehicle interaction and road geometry/ speed limit, which are denoted $c_i^{int,\alpha}$ and $c_i^{road,\alpha}$ respectively. As it is sufficient that the most restrictive constraint is active, the total constraint is computed as

$$c_i^\alpha = \min(c_i^{int,\alpha}, c_i^{road,\alpha}).$$

E. Stochastic Reachable Set of the Ego Car

In contrast to other traffic participants, the trajectory of the ego car is known since the trajectory planner module is connected to the safety verification module, see Fig. 1. This means, that for time points t_k, the position and velocity is known up to a certain inaccuracy Λ of the vehicle controller for trajectory tracking, such that $x(t) \in \hat{x}(t) + \Lambda$, where $\hat{x}(t)$ is the planned trajectory and the addition is performed as a Minkowski sum[2]. For time intervals $t \in [t_k, t_{k+1}]$, the reachable set is given by the two-dimensional interval

$$R([t_k, t_{k+1}]) = \bigcup_{t \in [t_k, t_{k+1}]} \hat{x}(t) + \Lambda$$

The reachable set is intersected with the cells of the discretized state space, similar to (4), in order to obtain the discretized probability distribution $\hat{p}^\alpha([t_k, t_{k+1}])$ of the ego car.

VI. Crash Probabilities of the Ego Car

The probabilities $p_i^\alpha([t_k, t_{k+1}])$ computed in the previous section refer to the probability that the continuous state is in a certain cell X_i, under the effect of the input interval U^α. The total probability without input information is obtained by summation:

$$p_i([t_k, t_{k+1}]) = \sum_\alpha p_i^\alpha([t_k, t_{k+1}]).$$

Each state space cell i represents a position and velocity interval S_e and V_m ($e, m \in \mathbb{N}^+$ indexing position and velocity segments), such that $X_i = S_e \times V_m$, where only the probability of the position interval S_e is of interest for the computation of crash probabilities. The position probability for a path segment is obtained from the joint probabilities by summation, resulting in

$$\begin{aligned} p_e^{path}([t_k, t_{k+1}]) :=& P(s \in S_e, t \in [t_k, t_{k+1}]) \\ =& \sum_m P(s \in S_e, v \in V_m, t \in [t_k, t_{k+1}]). \end{aligned}$$

As the deviation probability is piecewise constant, one can also define intervals D_f in which the deviation probability is constant, see Fig. 4. Introducing the probability for a deviation segment $p_f^{dev}([t_k, t_{k+1}]) = P(\delta \in D_f, t \in [t_k, t_{k+1}])$, the probability that $s \in S_e$ and $\delta \in D_f$ is $p_{ef}^{pos} = p_e^{path} \cdot p_f^{dev}$ due to the independency assumption (see Sec. III). The set obtained from $s \in S_e$ and $\delta \in D_f$ is a polygon $Y(e, f)$, which is indexed by its path segment $e \in \mathbb{N}^+$ and deviation segment $f \in \mathbb{N}^+$, see Fig. 12.

Without loss of generality, the crash probability is considered only for the ego car with one other traffic participant for simplicity in the following. In case there are more traffic participants, the following procedure has to be repeated for each one of them. For a better distinction of variables from the other vehicle with the ones of the ego car, the variables referring to the ego car are indexed by a hat ($\hat{\square}$). In order to compute the crash probability, the pairs of polygons $(Y(g, h), \hat{Y}(g', h'))$ for which the vehicle bodies intersect, have to be determined. Thereto, the possible set of vehicle bodies $B(g, h)$ for a set of vehicle centers in a polygon $Y(g, h)$ is introduced, which is the union of car bodies whose centers are within $Y(g, h)$ and whose orientation equals the direction of the path segment g,

[2]Minkowski addition of two sets A, B: $A + B = \{a + b | a \in A, b \in B\}$.

see Fig. 12. In order to efficiently compute the polygon pairs $(Y(g,h), \hat{Y}(g',h'))$ resulting in intersecting vehicle bodies ($B(g,h) \cap \hat{B}(g',h') \neq \emptyset$), three steps are suggested allowing to efficiently discard non-intersecting pairs:

1) First, it is checked if the vehicle bodies $\bigcup_f B(e,f)$ of a certain path segment e can possibly intersect the vehicle bodies belonging to a path segment g of the ego car. Thereto it is checked if the circles enclosing the set of vehicle bodies intersect, where circles are chosen since checking for their intersection is computationally cheap.
2) Next, the set of vehicle bodies $B(g,h)$, $\hat{B}(g',h')$ belonging to pairs of path segments passing the first test, are again checked for intersection by the same procedure using enclosing circles.
3) Finally, the polygon pairs passing the two previous tests are directly checked for intersection. However, polytope intersection is computationally expensive, such that look-up tables are generated in advance, which contain relative positions and angles for which polygons intersect. There are different look-up tables depending on checking intersection with a bicycle/bike, a car or a truck.

The set of combinations of path and deviation segments resulting in a crash are stored in a list Ω. The crash probability is obtained by adding the probabilities for the pairs of path and deviation segments in Ω:

$$p^{crash} = \sum_{(g,h,e,f)\in\Omega} \hat{p}^{pos}_{gh} \cdot p^{pos}_{ef}.$$

Note that the probability of a crash is computed in an over-approximated way as the sets of vehicle bodies $B(g,h)$ are over-approximations due to the union of possible vehicle bodies within the uncertain vehicle center $Y(g,h)$.

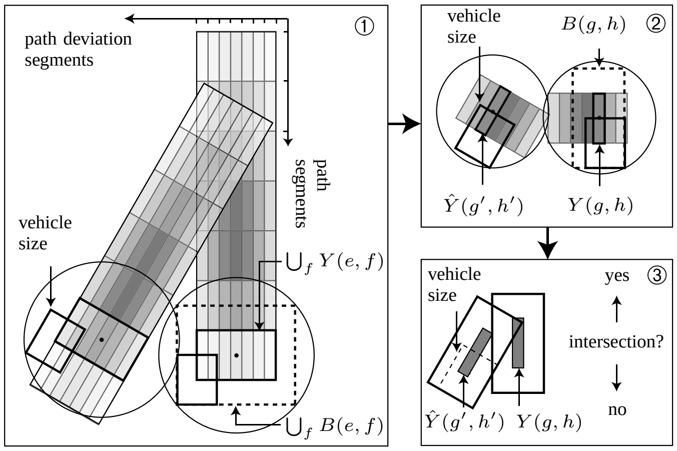


Fig. 12. Crash probability obtained from stochastic reachable sets.

VII. Numerical Examples

The presented methods for the safety verification of planned paths of autonomous cars are demonstrated for two traffic situations. For both examples, the same discretization of the state and input space is used for all vehicles. The position is discretized in 40 segments of 5 m length and the velocity is discretized in 10 segments, each representing an interval of 2.2 m/s. Each vehicle is subject to 5 input intervals and the time increment of the Markov chains is chosen as $T = 0.5$ sec. The γ-value for the input dynamics has been chosen as $\gamma = 0.2$ and the values of the characteristic input probability distribution are chosen as $m^{1\ldots5} = \begin{bmatrix}0.01 & 0.04 & 0.5 & 0.4 & 0.05\end{bmatrix}$.
In the first example, it is checked if the ego car can safely overtake a bicycle before reaching a T-intersection, where another car is approaching. The stochastic reachable sets of the traffic participants are given in Fig. 13. Dark regions indicate high probability, while bright regions represent areas of low probability. In order to improve the visualization, the colors are normalized for each vehicle separately, i.e. the highest probability value of a vehicle is plotted in black. Note, that both options, the left and right turn are considered with probability one each, for the other car. The second example shows a situation, where the ego car plans to merge into another road. It is checked if it hits the oncoming cars A, B and C, see Fig. 14. The stochastic reachable sets also show the interaction between car B and C, as car C is forced to brake due to the higher velocity compared to car B. The crash probabilities for both examples are found in Fig. 15.

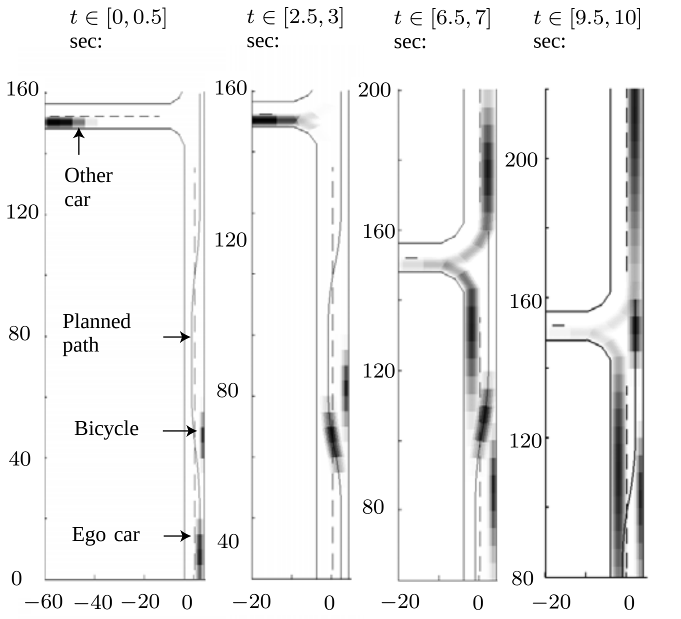


Fig. 13. Stochastic reachable sets for the overtaking scenario.

The examples are implemented in C++[3] and are computed on a 3.7 GHz single core desktop computer, for which the computation times are listed in Tab. I. The $\eta = t_f/t_c$ values, where t_f is the time horizon and t_c is the computation time, state how much faster the computation is than real-time. In addition to the discretization of the state and input space used so far, a coarser discretization with 6 instead of 10 velocity intervals and 3 instead of 5 input intervals is computed. The computation times for the more coarse discretization are also shown in Tab. I. The difference in accuracy for the high and low resolution discretization is tried to familiarize with by a scenario, where a vehicle is driving on a straight and free road.

[3]For convenience, the plots have been created in Matlab.

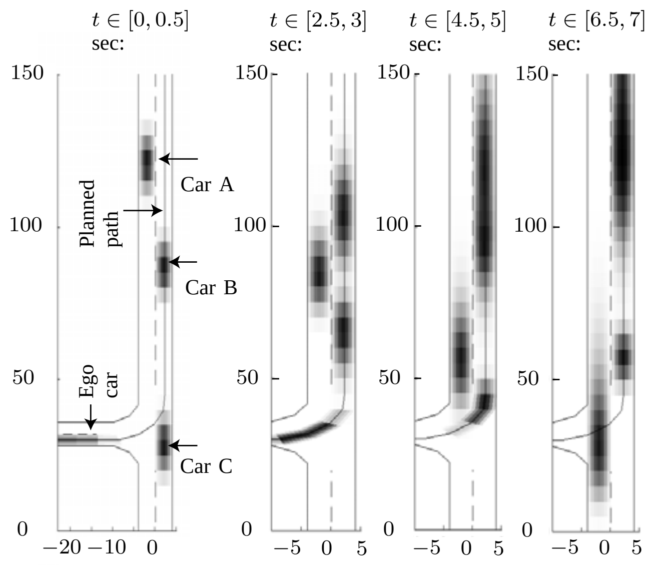
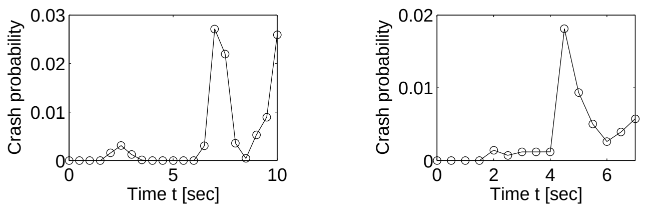


Fig. 14. Stochastic reachable sets for the merging scenario.

The probability distribution for the high and low resolution model is shown in Fig. 16. By showing the results for different initial velocities, it is shown that the accuracy of the coarser discretization significantly depends on the given situation. The bigger difference in the probability distributions for the low initial velocities is due to the fact, that the vehicle is more likely to reach standstill as the probability for the input interval including full braking increased after the mapping from 5 to 3 discrete inputs.

Another aspect, which is considered in Tab. I is the necessary time for the computation of crash probabilities if the trajectory planner forwards a replanned trajectory of the ego car, while the initial situation of the other traffic participants is unchanged. This time is referred to as replanning time in Tab. I and does not account for the time that the planner module requires for path planning of the ego car. The obtained times are for the case that the velocity of the ego car has been decreased by 20 %.



(a) Overtaking scenario. (b) Merging scenario.

Fig. 15. Crash probabilities of the overtaking and merging scenario.

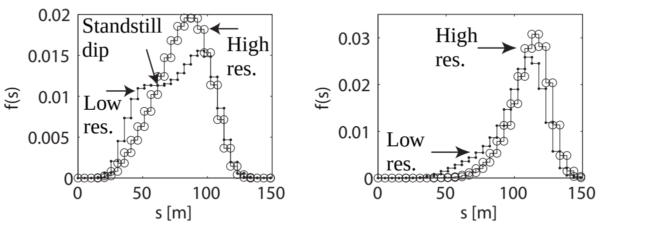
VIII. Conclusion

The presented approach allows to assess the safety of planned trajectories of autonomous cars by predicting the traffic situation. Uncertainties originating from measurements of other traffic participants and their unknown intention are considered in a stochastic way. In order to improve the prediction accuracy, interaction of traffic participants is included. The resulting crash probabilities of a planned path are conservative, i.e. if the crash probability is zero, it can be guaranteed that the ego car will not cause a crash if the traffic participants stay within the predefined physical bounds, such as maximum tire friction.

It has been shown that the approach is scalable and can be applied online. If not much computational power is available, the discretization resolution can be easily adapted. Further, as most of the computations are based on matrix multiplications, a specialized hardware (e.g. graphics chips) could provide a low cost solution for this approach. This would allow to use the proposed safety assessment algorithm for driver assistance systems by substituting the planned path of the autonomous car with a predicted path of the human driver.

TABLE I
Computational times of the numerical examples.

	Overtaking scenario		Merging scenario	
Discretization	Total time [sec]	Replanning time [sec]	Total time [sec]	Replanning time [sec]
High	0.56 (η : 17.9)	0.127	1.31 (η : 5.30)	0.074
Low	0.28 (η : 35.7)	0.086	0.51 (η : 13.7)	0.053



(a) Low initial velocity $v(0) \in [4, 6]$. (b) High initial velocity $v(0) \in [12, 14]$.

Fig. 16. Comparison of stochastic reachable sets with low/high discretization resolution.

Acknowledgment

The authors gratefully acknowledge the partial financial support of this work by the Deutsche Forschungsgemeinschaft (German Research Foundation) within the Transregional Collaborative Research Center 28 ”Cognitive Automobiles”.

References

[1] M. Abdel-Aty and A. Pande. ATMS implementation system for identifying traffic conditions leading to potential crashes. *IEEE Transactions on Intelligent Transportation Systems*, 7(1):78–91, 2006.

[2] M. Althoff, O. Stursberg, and M. Buss. Online verification of cognitive car decisions. In *Proc. of the 2007 IEEE Intelligent Vehicles Symposium*, pages 728–733, 2007.

[3] M. Althoff, O. Stursberg, and M. Buss. Reachability analysis of linear systems with uncertain parameters and inputs. In *Proc. of the 46th IEEE Conference on Decision and Control*, pages 726–732, 2007.

[4] M. Althoff, O. Stursberg, and M. Buss. Safety assessment of autonomous cars using verification techniques. In *Proc. of the American Control Conference*, pages 4154–4159, 2007.

[5] M. Althoff, O. Stursberg, and M. Buss. Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization. In *Proc. of the 47th IEEE Conference on Decision and Control*, pages 4042–4048, 2008.

[6] M. Althoff, O. Stursberg, and M. Buss. Stochastic reachable sets of interacting traffic participants. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 1086–1092, 2008.

[7] M. Althoff, O. Stursberg, and M. Buss. Verification of uncertain embedded systems by computing reachable sets based on zonotopes. In *Proc. of the 17th IFAC World Congress*, pages 5125–5130, 2008.
[8] A. Arapostathis, R. Kumar, and S.-P. Hsu. Control of markov chains with safety bounds. *IEEE Transactions on Automation Science and Engineering*, 2(4):333–343, 2005.
[9] E. Asarin, T. Dang, G. Frehse, A. Girard, C. Le Guernic, and O. Maler. Recent progress in continuous and hybrid reachability analysis. In *Proc. of the 2006 IEEE Conference on Computer Aided Control Systems Design*, pages 1582–1587, 2006.
[10] A. Barth and U. Franke. Where will the oncoming vehicle be the next second? In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 1068–1073, 2008.
[11] 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.
[12] A. E. Broadhurst, S. Baker, and T. Kanade. Monte carlo road safety reasoning. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 319–324, 2005.
[13] I. Dagli and D. Reichardt. Motivation-based approach to behavior prediction. In *Proc. of the Intelligent Vehicles Symposium*, pages 227–233, 2002.
[14] A. Eidehall and L. Petersson. Statistical threat assessment for general road scenes using monte carlo sampling. *IEEE Transactions on Intelligent Transportation Systems*, 9:137–147, 2008.
[15] J. Hillenbrand, A. M. Spieker, and K. Kroschel. A multilevel collision mitigation approach - its situation assessment, decision making, and performance tradeoffs. *IEEE Transactions on Intelligent Transportation Systems*, 7:528–540, 2006.
[16] J. Hu, M. Prandini, and S. Sastry. Aircraft conflict detection in presence of a spatially correlated wind field. *IEEE Transactions on Intelligent Transportation Systems*, 6:326–340, 2005.
[17] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank. A system for learning statistical motion patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:1450–1464, 2006.
[18] X. Koutsoukos and D. Riley. Computational methods for reachability analysis of stochastic hybrid systems. In *Hybrid Systems: Computation and Control*, pages 377–391, 2006.
[19] G. Lafferriere, G. J. Pappas, and S. Yovine. A new class of decidable hybrid systems. In *Hybrid Systems: Computation and Control*, LNCS 1569, pages 137–151. Springer, 1999.
[20] K. Lee and H. Peng. Evaluation of automotive forward collision warning and collision avoidance algorithms. *Vehicle System Dynamics*, 43(10):735–751, 2005.
[21] C.-F. Lin, A. G. Ulsoy, and D. J. LeBlanc. Vehicle dynamics and external disturbance estimation for vehicle path prediction. *IEEE Transactions on Control Systems Technology*, 8:508–518, 2000.
[22] J. Lunze and B. Nixdorf. Representation of hybrid systems by means of stochastic automata. *Mathematical and Computer Modeling of Dynamical Systems*, 7:383–422, 2001.
[23] J. Lunze and J. Schröder. Sensor and actuator fault diagnosis of systems with discrete inputs and outputs. *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, 34(2):1096–1107, 2004.
[24] J. Maroto, E. Delso, J. Flez, and J. M. Cabanellas. Real-time traffic simulation with a microscopic model. *IEEE Transactions on Intelligent Transportation Systems*, 7(4):513–527, 2006.
[25] A. Polychronopoulos, M. Tsogas, A. J. Amditis, and L. Andreone. Sensor fusion for predicting vehicles path for collision avoidance systems. *IEEE Transactions on Intelligent Transportation Systems*, 8(3):549–562, 2007.
[26] F. Rohrmüller, M. Althoff, D. Wollherr, and M. Buss. Probabilistic mapping of dynamic obstacles using markov chains for replanning in dynamic environments. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2504–2510, 2008.
[27] C. Schmidt, F. Oechsle, and W. Branz. Research on trajectory planning in emergency situations with multiple objects. In *Proc. of the IEEE Intelligent Transportation Systems Conference*, pages 988–992, 2006.
[28] J. Schröder. *Modelling, State Observation and Diagnosis of Quantised Systems*. Springer, 2003.
[29] C. Stiller, G. Färber, and S. Kammel. Cooperative cognitive automobiles. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pages 215–220, 2007.
[30] N. Sumpter and A. J. Bulpitt. Learning spatio-temporal patterns for predicting object behaviour. *Image and Vision Computing*, 18:679–704, 2000.
[31] J. van den Berg. *Path Planning in Dynamic Environments*. PhD thesis, Utrecht University, 2007.
[32] E. Velenis and P. Tsiotras. Optimal velocity profile generation for given acceleration limits: theoretical analysis. In *Proc. of the American Control Conference*, pages 1478 – 1483, 2005.
[33] Y. U. Yim and S.-Y. Oh. Modeling of vehicle dynamics from real vehicle measurements using a neural network with two-stage hybrid learning for accurate long-term prediction. *IEEE Transactions on Vehicular Technology*, 53:1076–1084, 2004.

Matthias Althoff Matthias Althoff received the diploma engineering degree in Mechanical Engineering in 2005 from the Technische Universität München, Germany. Currently he is a PhD student at the Institute of Automatic Control Engineering, Faculty of Electrical Engineering and Information Technology, Technische Universität München, Germany. His research interests include (stochastic) reachability analysis of continuous and hybrid systems, and safety analysis of driving strategies of autonomous cars.

Olaf Stursberg Olaf Stursberg received a Master degree in Chemical Engineering from University of Dortmund (Germany) in 1996 and the PhD degree from the same university in 2000. He was a postdoc in the department of Electrical and Computer at Carnegie Mellon University (USA) in 2001-2002 and a senior researcher and lecturer at University of Dortmund from 2003 to 2006. Since 2006 he holds the position of an associate professor in the Department of Electrical Engineering and Information Technology at the Technische Universität München (Germany), where he heads the group of industrial automation systems. His main research interests include the control, optimization and analysis of automated systems as well as hybrid and discrete event systems, formal verification, and intelligent and networked control systems.

Martin Buss Martin Buss received the diploma engineer degree in Electrical Engineering in 1990 from the Technical University Darmstadt, Germany, and the Doctor of Engineering degree in Electrical Engineering from the University of Tokyo, Japan, in 1994. In 2000 he finished his habilitation in the Department of Electrical Engineering and Information Technology, Technische Universität München, Munich, Germany. In 1988 he was a research student at the Science University of Tokyo, Japan, for one year. As a postdoctoral researcher he stayed with the Department of Systems Engineering, Australian National University, Canberra, Australia, in 1994/5. From 1995-2000 he has been senior research assistant and lecturer at the Institute of Automatic Control Engineering, Department of Electrical Engineering and Information Technology, Technical University Munich, Germany. He has been appointed full professor, head of the control systems group, and deputy director of the Institute of Energy and Automation Technology, Faculty IV – Electrical Engineering and Computer Science, Technical University Berlin, Germany, from 2000-2003. Since 2003 he is full professor (chair) at the Institute of Automatic Control Engineering, Faculty of Electrical Engineering and Information Technology, Technische Universität München, Germany; 2008- also in the medical faculty. Since 2006 he is the coordinator of the DFG Excellence Research Cluster ”Cognition for Technical Systems” CoTeSys. His research interests include automatic control, mechatronics, multi-modal human-system interfaces, optimization, nonlinear, and hybrid discrete-continuous systems.