Probabilistic Decision-Making under Uncertainty for Autonomous Driving using Continuous POMDPs

Sebastian Brechtel, Tobias Gindele, and Rüdiger Dillmann
Humanoids and Intelligence Systems Laboratories, Karlsruhe Institute of Technology
Adenauerring 2, D-76131 Karlsruhe, Germany
Email: { brechtel | gindele | dillmann}@kit.edu

***Abstract*— This paper presents a generic approach for tactical decision-making under uncertainty in the context of driving. The complexity of this task mainly stems from the fact that rational decision-making in this context must consider several sources of uncertainty: The temporal evolution of situations cannot be predicted without uncertainty because other road users behave stochastically and their goals and plans cannot be measured. Even more important, road users are only able to perceive a tiny part of the current situation with their sensors because measurements are noisy and most of the environment is occluded. In order to anticipate the consequences of decisions a probabilistic approach, considering both forms of uncertainty, is necessary. We address this by formulating the task of driving as a continuous Partially Observable Markov Decision Process (POMDP) that can be automatically optimized for different scenarios. As driving is a continuous-space problem, the belief space is infinite-dimensional. We do not use a symbolic representation or discretize the state space a priori because there is no representation of the state space that is optimal for every situation. Instead, we employ a continuous POMDP solver that learns a good representation of the specific situation.**

I. Introduction

Decision problems in the real world are aggravated by incomplete and noisy perception and uncertain knowledge about how the world evolves over time. The domain of autonomous driving is a very good example: On the one hand, the intentions of other traffic participants are unclear and their behavior cannot be predicted certainly. On the other hand, the major part of the environment is occluded. Even when they are visible to the sensors, measurements underlie errors. In the future, *car2X* communication might reduce these uncertainties by actively sharing knowledge between road users. Nevertheless, unconnected road users like pedestrians and variables that remain hidden to sensors like intentions always induce uncertainty. We argue that for safe driving decisions, uncertainty must not be neglected.

Manually modeling safe or even optimal decisions that consider all these properties, e.g., with rule-based systems, is very difficult and error-prone because of the arising complexity. POMDPs provide a principled and powerful framework for modeling sequential decision problems under uncertainties [1], [2]. In a POMDP, the agent can choose an action in every time step and this way take influence on the uncertain development of the situation. The task is additionally aggravated because he is never able to observe the state of the world fully.

Solving POMDPs exactly is computationally intractable. POMDPs with discrete state and observation spaces of medium complexity can be approximately solved in reasonable time. However, autonomous driving, like most other real-world applications, has a continuous and thus uncountably infinite state space. Finding a suitable symbolic representation is very difficult, as it heavily depends on the specific task and situation. The usual approach is to use an equidistant discretization of the continuous space. Especially in higher dimensional problems this can be problematic: At some places, e.g., in narrow passages, such a discretization is often too coarse and cannot represent enough detail to find a solution to the problem. At other places it represents details that are completely irrelevant, like if the distance to a leading car is $200\,\mathrm{m}$ or $201\,\mathrm{m}$. Here, the information that it is *far away* would be sufficient. Often even complete dimensions in the state space can be ignored. E.g., oncoming traffic can usually be neglected, but if the task is to make a left turn it becomes very important. For those reasons it is more general and preferable to formulate and solve the problem directly in its natural continuous space.

II. Overview

In this paper, we model the task of choosing safe, efficient, and goal-directed driving behaviors as a continuous POMDP. We employ a continuous POMDP solver [3] to find an approximately optimal policy and at the same time a problem-specific symbolic space representation. Related work to the task of tactical decision-making for self-driving cars is discussed in Section III. One of the dominating challenges for decision problems in general and also in the traffic domain is the representation of situations. This issue is discussed in Section IV. A brief introduction to continuous POMDPs is given in Section V. Our approach allows to directly use the natural continuous representation of traffic situations and their development over time without tailoring a symbolic representation for every special driving task. The spaces and models for the prediction, the perception, and the goals of the optimization form the POMDP. These models are presented in Section VI. Traffic situations are represented by the pose and velocity of the involved road users. The same basic formulation of the POMDP can be used for many different driving tasks. In Section VII the results are

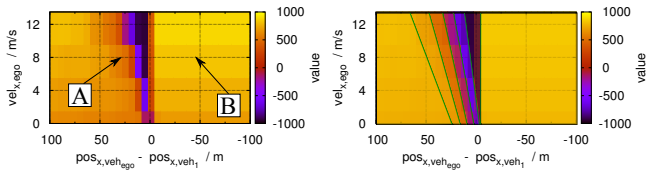
evaluated in a merging scenario with special emphasis on the question how the algorithm implicitly solves the problem of situation representation.

III. Related Work

The most common approach to decision-making today is to manually model reactions to situations. Mostly state machines are used to assess situations and decide in a single framework. One example is the hierarchical state machine used by Team *AnnieWAY* [4] in the Urban Challenge. Alternatives to state machines are, e.g., behavior networks [5], distributed voting-based behavior architectures [6] or hybrids of rule-based decision-making and behavior models [7]. With this approach, individual complex tasks, like following a leading vehicle or merging into traffic, usually need tailored solutions. Ziegler et al. [8] used a hierarchical, concurrent state machine to generate basic behaviors. A trajectory planning unit conducts the acc-/decceleration for maneuvers like merging. It is assumed that other cars maintain their distance to the right bound of the lane and do not accelerate. Uncertainty in the process or in the observation is not considered and it is not clear, how branching roads are handled. The winner of the Urban Challenge *BOSS* performed several manually defined checks that used metrics, e.g., checking the spacing and if the car can accelerate quickly enough, to assure that merging is safe [9]. Team *Junior* used a threshold test, based on velocity and proximity, to find *critical zones* where the vehicle has to wait [10].

Planning and learning based approaches for decision-making in various driving tasks have been proposed that are able to consider uncertainty. Wei et al. [11] presented a QMDP-based approach for single-lane behaviors. They show that considering uncertainty in the behavior of the leading vehicle as well as limitations of the perception improves robustness. However, they used a state space that is tailored for single-lane driving. Also, because only a QMDP and no POMDP was used, no active information gain policy can be achieved and the sensor constraints had to be modeled by hand to obtain the wanted behaviors. Ulbrich et al. [12] showed how POMDPs with a discrete state space can be used to derive lane change decisions. They use a small set of only 8 manually defined states to describe highway situations. While this approach keeps the complexity of the solving process manageable, the tailored representation restricts the application to highway lane change decisions and the capabilities of the resulting policies can be limited.

In a previous work [3], we embeded a continuous-state hierarchical Bayesian transition model in an MDP. Decisions for an autonomous car in a two-lane highway scenario with multiple other cars were automatically derived by solving the MDP. The evaluation showed that sophisticated behavior predictions, for example of the probability of lane changes, are essential for good decisions. A state space representation with relative distances and velocities was used. For this reason, the same approach was used for overtaking with oncoming traffic and it could be transferred to other situations like intersections. For complexity reasons, it is difficult to directly



(a) Equidistant discretization. (b) Improved discretization.

Fig. 1: MDP value when approaching a standing vehicle. The value depends on the distance (x-axis) to the vehicle and the velocity of the ego vehicle (y-axis). Based on [15].

extend this approach to partial observability (see Section IV). Bandyopadhyay et al. [13] suggest that for navigation tasks the intentions of other road users should be considered. They present a Mixed Observable MDP (MOMDP) approach that treats pedestrians' intentions as hidden variables. The state of the predestrians, however, is assumed to be fully observable. Additionally, due to the computational complexity, multiple pedestrians were treated as independent entities.

Recent developments in the area of solving continuous POMDPs led to more capable solvers. Their capabilities are often demonstrated with tasks that are similar to driving. These methods solve the tactical decision making problem without needing symbolic representations and under full consideration of process uncertainty and partial observability. Bai et al. [14] present a sampling based continuous POMDP solver using a Generalized Policy Graph (GPG) representation. It is evaluated for an intersection-like scenario with a single other vehicle that can only be observed with noisy measurements. Velocities and inertia were not considered in this example. In [3] a Value Iteration method for continuous POMDPs is presented. Their evaluation shows that this approach is able to solve the decision-making problem for an intersection-like scenario with limited sensor range and capabilities. The 8D space of the POMDP consisted of two cars and their velocities. Consequently, also physical effects like the constrained acceleration because of mass inertia were considered.

IV. Space Representation in Decision Processes

A decision problem can be formulated as an optimization of a reward or a cost function over time. The expected sum over the future rewards is often called *value*. A corresponding *policy* can be derived, which defines what reactions to situations have to be chosen in order to reach the value. Finding suitable representations for solving decision problems and policies is a challenging task. A *good* representation needs to be able to represent every decision relevant detail. At the same time, for the sake of efficiency, a space representation should abstract from details that are not not important to solve the given decision problem.

Even for simpler tasks, like highway-driving without considering partial observability, it is difficult to find a good representation. We demonstrate this in the example of the MDP policy from [15]. Figure 1a shows the value (the expected future reward), when driving on the right lane

together with a stopping vehicle. It uses an equidistant discretization of the distance to the vehicle with $6\,\mathrm{m}$ steps. $4\,\frac{\mathrm{m}}{\mathrm{s}}$ steps for the velocity of the vehicles are used. On the one hand, it is evident that the results could have been improved if a more detailed discretization had been used. Especially in the area around A, where the distance is small and the chance of a collision is quite high, a finer granularity would have been useful to assure a correct reaction. On the other hand, many states were discriminated although their value and as a consequence their policy is very similar. For example in area B, where the ego vehicle is driving away from the standing vehicle, the precise distance is not important.

The MDP formulation contained several thousands of states, rendering it computationally expensive to solve. The worst case complexity for the running time of one MDP value iteration or policy iteration step is quadratic in the cardinality of the space.

Solving the problem with a discretized space that is better suited to the problem, e.g., a linear combination of the distance and the velocity like it is sketched in Figure 1b, would be able to represent more details and at the same time reduce the cardinality of the space. The ability to represent more relevant details improves the quality of the resulting policy and consequently yields higher average rewards. Partial observability even increases this complexity to the point where it is practically impossible to solve driving problems, if an unsuitable representation is used. What constitutes a suitable or *good* representation, however, differs very much depending on the underlying problem. If, for instance, the other vehicle is driving with high speed (instead of stopping), the representation proposed in Figure 1b fails because then negative distance values would have to be discriminated. It is important that a simplified representation preserves enough information to select the optimal policy [16]. Feng et al. showed that in discrete POMDPs states with the same value can be agglomerated [17]. For decision processes, the value function serves as a mathematically sound basis to deduce what is relevant to the decision-making and what is not.

In previous work, we [3] transferred this idea to continuous POMDPs by applying a learning algorithm during value iteration in order to only distinguish states if the discrimination is necessary to represent the optimal decision. This way, a suitable representation is found during the process of solving the continuous POMDP. Using this approach, there is no need for discretizing the state space a priori. In this paper, we use the continuous solver presented in [3] to solve the problem directly in a formulation that is solely based on the pose and velocity states of cars.

V. Continuous Partially Observable Markov Decision Processes

A discrete time POMDP models the decision process of an agent acting in a dynamic environment. See Figure 2 for a representation of a POMDP as a Dynamic Bayesian Network. The *state* of the world $s \in \mathcal{S}$ changes with every time step following the stochastic *process model* $p(s'|s,a) = p(s_{t+1}|s_t,a)$. The Markov Assumption applies, hence the state s' only depends on the previous state s and the current action a. By deciding for an action a from the set of actions $\mathcal{A}$ the agent can influence the behavior of the world in order to optimize the expectation of the *reward function* $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. Analogously to Hidden Markov Models the agent can only perceive the state partially through *observations* $o \in \mathcal{O}$ following the *observation model* $p(o|s') = p(o|s_{t+1})$. Thus, in contrast to MDPs, he does not know the state of the world exactly, but only has an estimation called the *belief* $b \in \mathcal{B}$ with $b : \mathcal{S} \to \mathbb{R}_{\geq 0}$ and $\int_{s\in\mathcal{S}} b(s)\mathrm{d}s = 1$. The goal of the agent is to maximize the *value* $V : \mathcal{B} \to \mathbb{R}$ for an initial belief b_0. It therefore has to find an optimal policy $\pi^* : \mathcal{B} \to A$ and act accordingly. The value is defined as the expected future sum of rewards for an *initial belief* b_0 discounted by $\gamma \in [0,1)$ under the policy π:

$$V_\pi(b_0) = E\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, \pi(b_t))\right]. \tag{1}$$

Value Iteration is a popular method to solve discrete POMDPs based on the paradigm of dynamic programming [18]. It uses Bellman backups to draw conclusions for the current value and the current best decision by anticipating future beliefs and their values. Recursive application of this form of backward induction converges to the optimal value and solves the problem of finding an optimal policy [1]. Further, the value function is piecewise linear and convex (PWLC) and can be represented as the maximum of a finite set of linear α-vectors, which is beneficial for the application of Value Iteration. Porta et al. [19] showed that the same holds for continuous POMDPs where states and observations can be from the set of real numbers. Then a policy π can be expressed as the supremum of a finite set Γ_π of linear α-functions:

$$V(b) = \sup_{\alpha\in\Gamma_\pi} \langle \alpha, b\rangle = \sup_{\alpha\in\Gamma_\pi} \int_{s\in S} b(s)\alpha(s)\mathrm{d}s. \tag{2}$$

In this paper, the Monte Carlo Value Iteration algorithm presented in [3] is used to optimize the continuous POMDP model. The backup step in this algorithm is based on Importance Sampling and similar to Sequential Monte Carlo methods, also known as Particle Filtering. As a consequence, a particle-based representation of the distributions is used. This enables the method to handle multi-modal and highly non-linear distributions well. This is especially interesting when dealing with sensor occlusions. The method integrates machine learning into a Value Iteration with α-functions in order to automatically find a problem-adequate discrete representation of the continuous space as part of solving the POMDP, thereby realizing the ideas discussed in Section IV.

We use the solver to create an offline policy, represented by a state space representation in form of a decision tree and a set of discrete α-vectors. The precalculated policy can be executed by looking up the dominating α-function for the current belief and its associated action following equation (1). This can be accomplished in milliseconds: First the continuous belief has to be discretized using the learned state

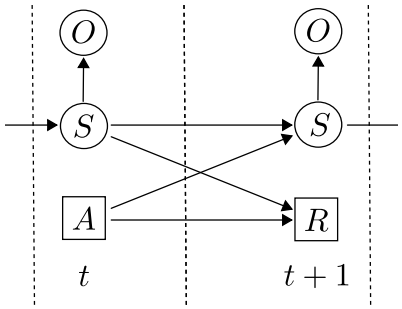


Fig. 2: A POMDP model as a Dynamic Bayesian Network. The action A can be set by the solver. The reward function is deterministic.

representation. Then the (sparse) dot-product of the discrete representation with all α-vectors has to be computed.

VI. A POMDP Model for Driving

In order to formulate the task of driving as a continuous POMDP the spaces and models have to be defined. Figure 2 gives an overview on the relationships of the spaces and models.

A. Spaces

1) State Space: The state space $\mathcal{S}$ must be sufficient for the Markov Property and hold sufficient information for decision-making. Because a continuous POMDP solver is used, there is no need to discretize the state space a priori. Hence, we can use a similar representation as in filtering and prediction tasks. Information like the road topology and geometry is assumed to be static background knowledge of the models and consequently not covered by the state.

The state of the environment $x \in \mathcal{X}$ is the vector of the states of all n road users and the ego vehicle. The state $x_i \in \mathcal{X}_{\text{obj}}$ of every road user including the controlled agent vehicle x_e is represented by its global position $(\mathrm{x}_1, \mathrm{x}_2)$, orientation ψ, and velocity v:

$$x = \left(x_e, x_1, \ \ldots, \ x_n\right)^\mathsf{T}, \tag{3}$$

$$\text{with} \quad x_i = \left(\mathrm{x}_1, \mathrm{x}_2, \psi, \mathrm{v}\right)^\mathsf{T} \in \mathcal{X}_{\text{obj}} \subseteq \mathbb{R}^4. \tag{4}$$

Additionally to X that represents normal driving situations a special *terminal* state s_{term} is introduced. The full $s \in \mathcal{S}$ of the world is either in an environment state $s \in \mathcal{X}$ or terminal $s = s_{\text{term}}$:

$$\mathcal{S} = \mathcal{X} \cup \{s_{\text{term}}\}. \tag{5}$$

The terminal state s_{term} models the semantic that the planning problem is over, e.g., after crashing or after reaching the goal. This state plays a very important role because it effectively prevents the system to draw wrong conclusions such as that it is possible to reach the goal after a collision or that the goal can be reached multiple times. Its properties, which are described in the according models, assure that the future reward collected in the terminal state is always zero. Consequently, to improve the solver's performance, the planning can be stopped if the terminal state is reached without negative effect on the results.

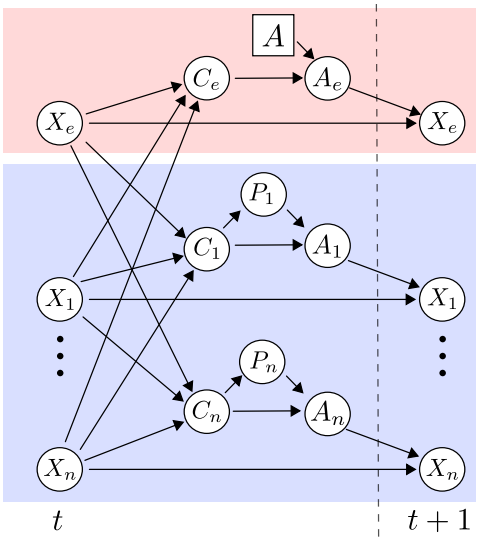


Fig. 3: Bayesian Network represntation of the process model. The area marked red is the ego vehicle and the area marked blue are the other road users.

2) Observation Space: The ego vehicle is assumed to directly observe its own state and the states of other road users x_i. Additionally, there is the special case o_{inv} that a road user is invisible to the sensors due to occlusions or other sensor limitations.

$$o = \left(o_e, o_1, \ \ldots, \ o_n\right)^\mathsf{T}, \text{ with} \quad o_i \in \mathcal{X}_{\text{obj}} \cup \{o_{\text{inv}}\}. \tag{6}$$

3) Action Space: The goal of the decision-making process is to decide for a tactical maneuver like stopping in front of an intersection or holding a safety distance. Such reactions are covered by the discrete action A consisting of a finite set of acc-/deceleration. The action space is the only space that needs to be discrete. The information that the vehicle follows the road is part of the process model. The discrete decision for an abstract action $a \in \mathcal{A}$ implies a continuous control policy for $a_e \in \mathcal{A}_{\text{c}}$ under consideration of the context and the underlying road network. Every car in the model has such a continuous control variable A_i with $a_i \in \mathcal{A}_{\text{c}}$, but only the action of the ego vehicle can be actively set.

B. Models

The process and observation model and the reward function can be seen as the *glue* between the spaces, as they define the meaning of the spaces for the decision process.

1) Process Model: The process model describes the dynamics of the system and the influence of the chosen actions on the system. It is represented by the conditional density function $p(s'|s, a)$ to account for the uncertainty in the transition. For the presented driving task, the process model is the most complex one because it has to comprise the behavior and interaction of the road users as well as the underlying driving physics and background knowledge, e.g., from maps. We use a hierarchical Bayesian model that is structured and implemented similarly to the one presented in [20]. Additionally to this model, an autonomously controlled car (the ego car) is added. The ego car's action is not part

of the models, but can be set directly by the solver to be able anticipate the consequences and to conduct an optimized policy.

Figure 3 shows a simplified overview on the extended Bayesian Network representation of the process model. In the first place it has to model that the motion of cars is restricted by kinematic and dynamic constraints. Therefore an underlying pyhsical model $p(x_i'|x_i, a_i)$ concludes the next state of every car $i \in \{e, 1, \dots, n\}$ from its individual control action A_i and the previous state. For the not-controlled vehicles, the decision-making process of their drivers is resembled as presented in [20]. This includes the fact that the other drivers $j \in \{1, \dots, n\}$ usually follow the course of the road and interact with each other and the ego car. Therfore a planned route P_j is estimated from their situational context C_j and map knowledge. From this information, the control actions A_j are derived by $p(a_j|c_j, p_j)$. The ego vehicle is an exception because its high-level behavior is preset by the discrete action $a \in \mathcal{A}$. The model $p(a_e|a, c_e)$ implements the discrete actions. It depends on the context of the ego vehicle to be able to define actions that account for other road users like *'hold the velocity of the car in front'*.

The time step of the tactical decision-making and thus the POMDP is $1.5\,\mathrm{s}$. To get a good simulation of the road users, a time step smaller than $0.2\,\mathrm{s}$ is necessary. For this reason several internal prediction steps are executed to compute a single PODMP prediction step.

The terminal state has the special property that there is no way out: $\forall a \in \mathcal{A} : p(s = s_{\text{term}'}|s_{\text{term}}, a) = 1$. The terminal state is reached when a collision between the ego vehicle and another road user happened or when the goal (given by the mission) is reached.

The detection of collisions is an essential part of the process model and also very important for the reward function. Collisions need to be detected also if they happen between a time step and the evaluation needs to be efficient. For this reason we perform a time-continuous collision check by assuming the motion to be linear between the steps and using simplified car shapes.

2) Observation Model: The observation model has to simulate the measurement process. The measurements of the objects are modeled to be independent from each other by $p(o_i|x_e, x_i')$. Sensor inaccuracies are modeled with additive, Gaussian distributed noise. More important for decision-making is the fact that other road users j often are completely invisible to the sensors. This might be the case because the angle of view or the range of the sensors is limited. Also the view can be occluded by other road users or environment objects like houses. Occlusions are detected by checking if there is a direct line of sight to the road user that does not intersect with other objects, which are represented as polygons. See Figure 4 for an illustration.

Only when the observation model accounts for non-visibility, the optimized policies will conduct information-aware behaviors like stopping at an intersection to look behind a building or not overtaking a vehicle because the ego vehicle cannot see behind it.

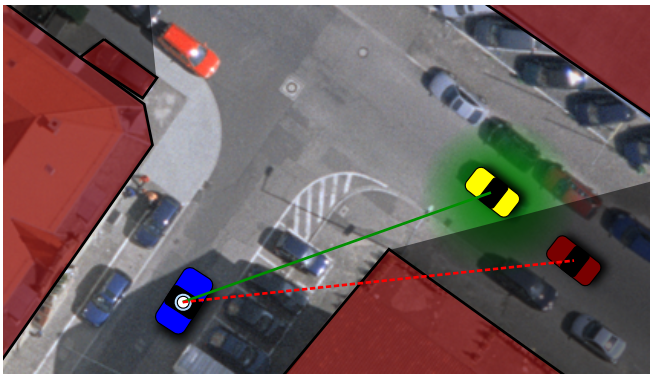


Fig. 4: Observation model. Red polygons represent opaque areas. The yellow car is visible and measured with normal distributed noise. The red car is not visible because the house occludes the view.

If the object x_j' is visible from the ego state x_e, the observation is modeled by

$$p(o_j|x_e, x_j') = \mathcal{N}(x_j', \mathbf{\Sigma}_O) \ , \ p(o_j = o_{\text{inv}}|x_e, x_j') = 0 \ . \quad (7)$$

The perception noise can be adapted according to the angle or to the distance, if that is necessary to model the used sensors. Using, e.g., stereo based systems, Σ_O should be increased with the distance. Also the variance in the dimension of the depth should be higher than in the horizontal dimension. If the object is invisible, the model is

$$p(o_j|x_e, x_j') = \begin{cases} 1 & \text{if } o_j = o_{\text{inv}} \\ 0 & \text{else} \end{cases} \ . \quad (8)$$

The ego vehicle e can safely be assumed to measure its pose and velocity with very little noise, using state of the art methods from the area of Simultaneous Localization And Mapping (SLAM).

3) Reward Function: The resulting policies must satisfy several criteria simultaneously. The main goal of the autonomous car is to reach its target destination as quickly as possible, while minimizing the risk of crashing. At the same time, it should drive as comfortable and economical as possible and comply with the traffic rules. The reward function quantifies these objectives. It is a real-valued deterministic function of the previous environment state, the conducted action and the resulting situation.

For driving, it can be separated into two parts. The first part only depends on the ego vehicle's state x_e, the chosen action a and background knowledge, i.e. it defines egocentric goals. In the presented system it aims to optimize comfort and efficiency by returning a cost for acc- and deceleration. A reward is given, if the car successfully reaches a goal area, e.g., located behind an intersection. Additionally, if the car leaves the road, a cost for a collision is given. In future work, compliance with traffic rules such as speed limits can be incorporated. The second part also considers other traffic participants. The most important aspect here is to return high costs for collisions with other road users.

The terminal state is never rewarded

$$\forall a \in \mathcal{A} : r(s_{\text{term}}, a) = 0 \, . \quad (9)$$

The reward function sums all these factors to a global scalar driving objective. They are weighted to balance the different criteria according to special needs or preferences such as risk averse or sporty driving.

VII. Evaluation

We evaluate the presented approach in a merging scenario: The self-driving car has to control its velocity to merge safely into a gap between two other cars that it has to yield to. The task is aggravated because the perception of the autonomous car is blocked by the red occlusion areas. In order to solve this situation, the car has to stop about $12\,\mathrm{m}$ in front of the intersection to see where the gap between the other cars is and if it is large enough to merge safely. A simple hand-coded strategy that approaches the intersection first does not work here because the view is blocked by a wrongly parked truck (rectangular occlusion area at the intersection). Also, the acceleration is limited and the car would not be able to reach the speed of the gap in time.

To evaluate the system, we precomputed a policy for this scenario and all possible outcomes of it offline. The policy was derived by solving the model described in Section VI using the POMDP solver presented in [3]. The models were parametrized as follows.

A. Parameters

In the initial belief the pose of the cars is set as shown in Figure 5 and 6 at $t = 0$. A timestep of $1.5\,\mathrm{s}$ is used. The uncertainty in the positions is uniformly distributed. The self-driving car e (*car 0*) can choose to accelerate or decelerate with $1\,\frac{\mathrm{m}}{\mathrm{s}^2}$. Initially, *car 0* has the speed $3\,\frac{\mathrm{m}}{\mathrm{s}}$. *Car 1* and *2* have the speed $5\,\frac{\mathrm{m}}{\mathrm{s}}$. In the process model all vehicles have additive white noise in the velocity dimension. Further, noise is added to the position of the vehicle in the vehicles driving direction. Both are scaled with the velocity and normally distributed. If a car is visible, the covariance of the observation of the position dimension is $\Sigma_O = 1\,\mathrm{m}^2$. The velocity and the orientation are assumed to be not measured directly. To account for that these dimensions have very high covariances (approximating a uniform distribution). *Car 0* is assumed to localize itself at any point in time precisely. A reward of 5 is given, if *car 0* reaches the goal area after the intersection and a negative reward of -1 else. If a collision happens, it receives a negative reward of -10.

B. Behavior Examples

In Figure 5 and Figure 6 the reactions of the resulting POMDP policy for two different trials are shown. The current belief is represented by a particle set. A single state consists of the poses and velocities of *car 0* to *2*. The association of the car's states can be seen from the connecting lines. The color of the lines indicates whether *car 0* can see the other cars (green lines) or not (red lines). The current *true* underlying state is highlighted by the position of the text "car i". This true state is not known to the self-driving car. It has to gain the information about the situation indirectly

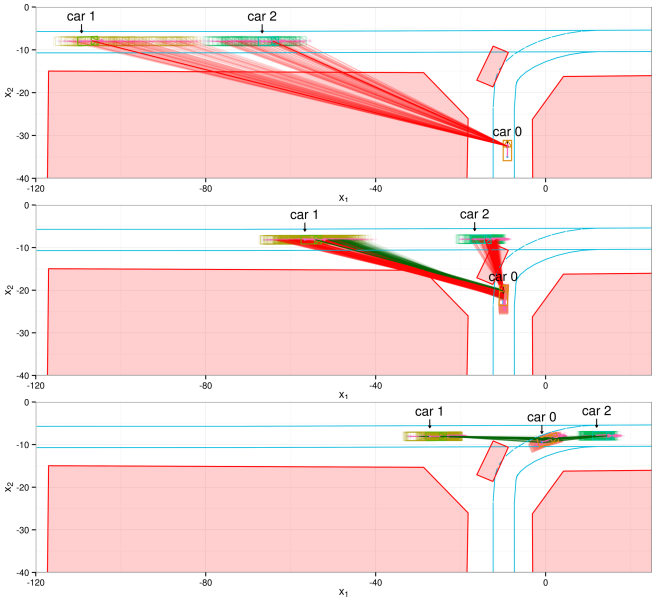


Fig. 5: Policy simulation at $t = 0, 7$ and 11.

using observations and build its belief distributions using Particle Filtering.

The first plot at time step $t = 0$ in Figure 5 shows the initial distribution of the problem. The true state shows a big gap between *car 1* and *car 2*. The POMDP policy makes the controlled car stop and wait before the truck that blocks the view, until it can gain enough information about the gap. In the second plot at $t = 7$ ($10.5\,\mathrm{s}$ after $t = 0$) it starts to accelerate. Note that it was not able to see *car 1* at this point in time. Yet, it can be certain that the gap is big enough because it would have seen *car 1* otherwise. This example shows the ability of the POMDP policy to gain information from negative observations.

The true state at the beginning of Figure 6 shows a gap of about $25\,\mathrm{m}$. This size would be sufficient for the car to fit in. However, the POMDP policy decides in $t = 7$ to not merge and wait for the next gap instead. The last time step shows that this is a good decision: Because of the stochastic driving of the other cars, the gap shrunk to a size of about $10\,\mathrm{m}$. Thus, merging would have been dangerous.

C. Overall Performance

To evaluate the overall performance of the precalculated POMDP policy, we repeated 1000 simulations with different initial true states sampled from the initial belief. In Figure 7 the summed discounted rewards the self-driving car received and the true size of the gap at the beginning of the simulation are plotted. If the car collided during the simulation, it receives a negative reward of about -6. The precise value depends on the time when it hit the other vehicle because of the discounting. If it reaches the goal area, it receives a reward of about 1.3. If it has to wait for the next gap, it receives about -1.8 because it has only costs. As the development of the situation is uncertain, these values can be interpreted as a joint probability density over the summed reward and the initial gap size. This is indicated by the kernel

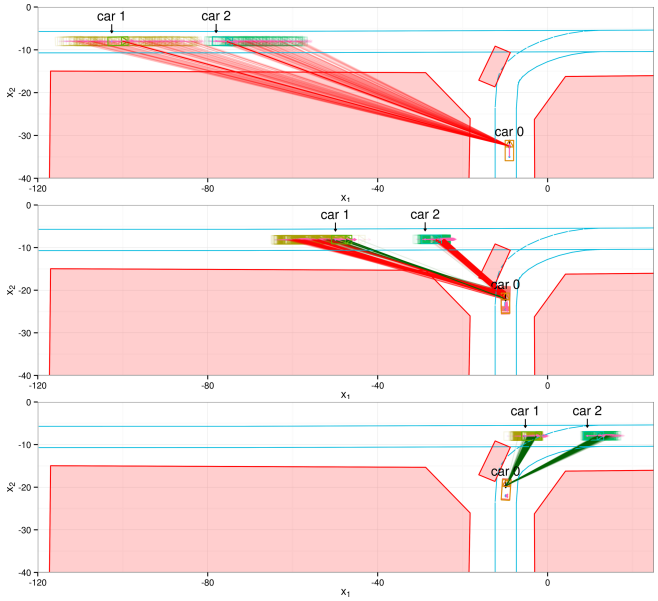


Fig. 6: Policy simulation at $t = 0, 7$ and 13.

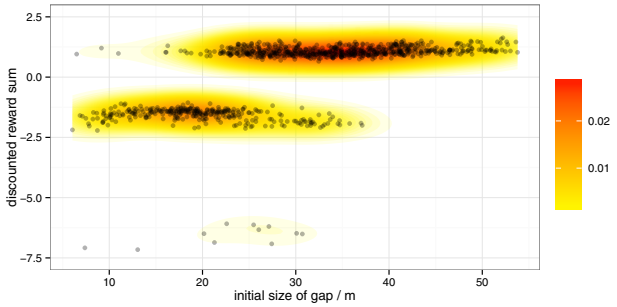
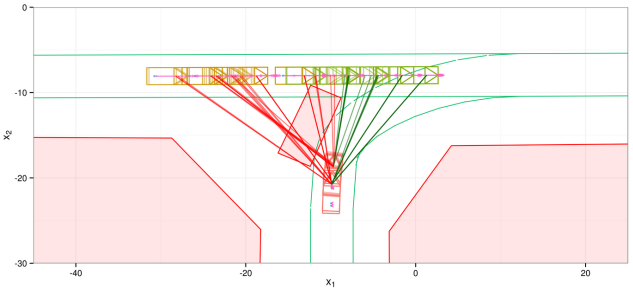


Fig. 7: Relation of initial gap size of a simulation trial and its result for the summed discounted reward.

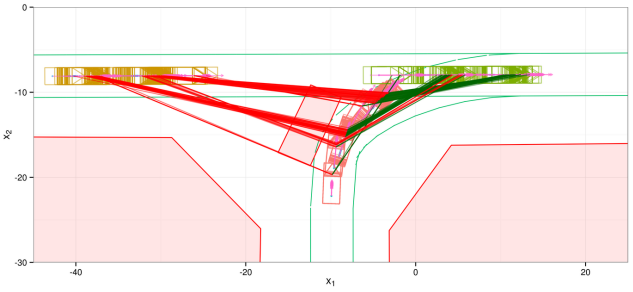
density estimation.

It can be seen that the policy chooses to merge into gaps that are bigger than ca. 27 m and waits if they are smaller. There are varying results for the gap size around 27 m. The different decisions have their reason in the stochastic development of the situation and details: The exact position and velocity of *car 2* and if the self-driving car can accelerate quickly enough in the particular situation, determines if merging is safe. The collision particles in the lower part of the plot show that collisions, especially if the gap is just big enough, cannot be completely prevented. The reason for these also lie in the normal distributed uncertainty of the other cars driving behavior. In some cases, the policy succeeds for smaller gap sizes because of the same uncertainty in the process. Because of this uncertainty, any policy can only minimize the probability for accidents, but never prevent them.

Space Representation To represent the 12-dimensional combined space of all three cars efficiently for this problem, the POMDP solver learned a decision tree with 442 leafs. Every leaf encodes one POMDP state. To reduce the complexity of the planning space in this dramatic manner, many



(a) POMDP state: Gap is too small to merge.



(b) POMDP state: Gap is sufficiently large to merge.

Fig. 8: POMDP state representation. 12 D states that are mapped to the same POMDP state and not differentiated.

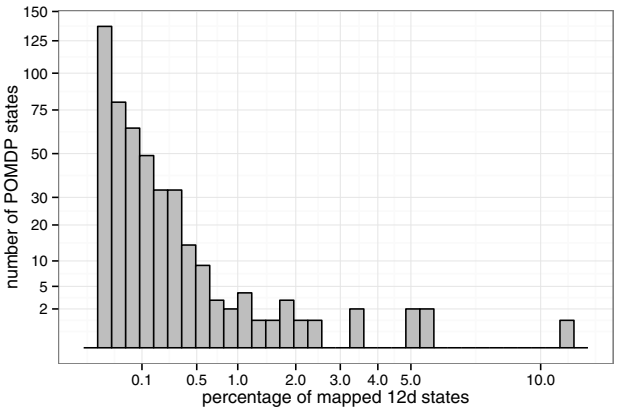


Fig. 9: Histogram showing the large spectrum of the size of the discrete POMDP states.

continuous states have to be summarized to a single discrete state. Figure 8 shows two discrete states of the final policy. In both graphics several continuous samples of the 12 D states are shown that are mapped to the same POMDP state. The state in Figure 8a subsumes several 12 D states where the gap is not big enough to merge safely. From the decision-making point of view, distinguishing those is not valuable because the outcome of those situations is the same: If the car does not stop, it will receive a cost for the collision. The state in Figure 8b contains continuous states where accelerating (and merging) will probably be successful. They can only be interpreted correctly with the connecting lines that show which car configurations form one joint continuous state. Note that the merged states are quite far away from each other in the euclidean 12 D space.

The histogram in Figure 9 shows how *big* the learned POMDP states are. Therefore we collected $3\,876\,880$ continuous state samples from 1000 simulations. These samples cover most reachable and relevant states of the problem. Then we discretized them to the POMDP representation and plotted how many percent of these states were mapped to the same discrete POMDP state. The histogram shows that most discrete states represent *small* areas, meaning that only a small number of continuous states is covered by the discrete state. These states usually model situations were the decision is a very close call. However, about $58\,\%$ of the sampled particles were mapped to only 20 discrete POMDP states. These results indicate that the discrete states are very small at some positions to represent details. At other positions they can aggregate many particles that need not be differentiated, which makes the computation feasible.

VIII. Conclusion

In this paper a generic approach to tactical decision-making for self driving cars is presented. We argue that uncertainties, which have their origin mainly in the stochastic behavior of other road users and the limited perception of the environment, must not be neglected to make safe driving decisions. We identify finding a suited space representation for driving as one of the prevailing challenges for automatic decision-making. An approach based on solving a continuous POMDP is proposed that automatically learns a suited representation depending on the specific given problem. The decision problem is modeled as a continuous POMDP, which is directly based on the pose and velocities of the involved road users. The evaluation shows that the approach is capable of merging into traffic under non-trivial occlusions of the perception. Further, the results indicate that the approach is able to find a sufficient and efficient discrete representation for this task.

In future work, other situations like following, highway and intersection behaviors need to be evaluated. The acc-/decceleration actions need to be extended with lane change actions for driving on multiple lanes. As the system uses a low-level state space, there is no need to define new models or symbolic states for these tasks. It is also interesting how the presented approach scales to cooperative driving where several cars can be controlled. This could be realized with a centralized POMDP planning with combined actions for all controllable road users.

An interesting idea for future approaches is to utilize an occupancy-based representation like in [21] with POMDP planning. This could mitigate the effect of the number of road users on the computational performance.

Besides fully autonomous driving, the approach could be used for drivers assistance systems. The behavior and intentions of the driver of the assisted vehicle would in this scenario be modeled with uncertainty as a conditional density function. Assistance actions would only indirectly influence the assisted car through the driver.

Acknowledgement The authors gratefully acknowledge the support of the Software Campus, the German Federal Ministry of Education and Research (BMBF), and the German Research Foundation (DFG).

References

[1] E. Sondik, "The Optimal Control of Partially Observable Markov Decision Processes," *PhD thesis, Stanford University*, 1971.

[2] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Journal on Artificial Intelligence*, vol. 101, no. 1–2, pp. 99 – 134, 1998.

[3] S. Brechtel, T. Gindele, and R. Dillmann, "Solving continuous POMDPs: Value iteration with incremental learning of an efficient space representation," in *International Conference on Machine Learning (ICML)*, Atlanta, GA, USA, 2013, pp. 370–378.

[4] T. Gindele, D. Jagszent, B. Pitzer, and R. Dillmann, "Design of the planner of Team AnnieWAY's autonomous vehicle used in the DARPA Urban Challenge 2007," in *Intelligent Vehicles Symposium*, 2008, pp. 1131–1136.

[5] J. Schröder, M. Hoffmann, M. Zöllner, and R. Dillmann, "Behavior Decision and Path Planning for Cognitive Vehicles using Behavior Networks," in *Intelligent Vehicles Symposium*, Istanbul, Türkei, 2007, pp. 710–715.

[6] J. K. Rosenblatt, "Damn: A distributed architecture for mobile navigation," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 9, no. 2-3, pp. 339–360, 1997.

[7] F. W. Rauskolb, K. Berger, C. Lipski, M. Magnor, K. Cornelsen, J. Effertz, T. Form, F. Graefe, S. Ohl, W. Schumacher, *et al.*, "Caroline: An autonomously driving vehicle for urban environments," *Journal of Field Robotics*, vol. 25, no. 9, pp. 674–724, 2008.

[8] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, T. Strauss, C. Stiller, T. Dang, U. Franke, N. Appenrodt, C. Keller, *et al.*, "Making Bertha Drive? An Autonomous Journey on a Historic Route," *Intelligent Transportation Systems Magazine*, vol. 6, no. 2, pp. 8–20, 2014.

[9] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, *et al.*, "Autonomous driving in urban environments: Boss and the urban challenge," *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[10] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, *et al.*, "Junior: The stanford entry in the urban challenge," *Journal of Field Robotics*, vol. 25, no. 9, pp. 569–597, 2008.

[11] J. Wei, J. M. Dolan, J. M. Snider, and B. Litkouhi, "A point-based mdp for robust single-lane autonomous driving behavior under uncertainties," in *International Conference on Robotics and Automation*. IEEE, 2011, pp. 2586–2592.

[12] S. Ulbrich and M. Maurer, "Probabilistic Online POMDP Decision Making for Lane Changes in Fully Automated Driving," in *Intelligent Transportation Systems*, 2013, pp. 2063–2070.

[13] T. Bandyopadhyay, C. Z. Jie, D. Hsu, M. H. Ang Jr, D. Rus, and E. Frazzoli, "Intention-aware pedestrian avoidance," in *Experimental Robotics*. Springer, 2013, pp. 963–977.

[14] H. Bai, D. Hsu, and W. S. Lee, "Integrated perception and planning in the continuous space: A POMDP approach," in *Robotics: Science and Systems*, 2013.

[15] S. Brechtel, T. Gindele, and R. Dillmann, "Probabilistic MDP-behavior planning for cars," in *Intelligent Transportation Systems*, 2011, pp. 1537–1542.

[16] P. Poupart and C. Boutilier, "Value-directed compression of POMDPs," *Advances in Neural Information Processing Systems*, vol. 15, pp. 1547–1554, 2002.

[17] Z. Feng and E. Hansen, "An Approach to State Aggregation for POMDPs," in *Workshop on Learning and Planning in Markov Processes*, 2004, pp. 7–12.

[18] R. Bellman, "A Markovian decision process," *Journal of Applied Mathematics and Mechanics*, vol. 6, pp. 679–684, 1957.

[19] J. Porta, N. Vlassis, M. Spaan, and P. Poupart, "Point-based value iteration for continuous POMDPs," *The Journal of Machine Learning Research*, vol. 7, pp. 2329–2367, 2006.

[20] T. Gindele, S. Brechtel, and R. Dillmann, "Learning context sensitive behavior models from observations for predicting traffic situations," in *Intelligent Transportation Systems*, 2013, pp. 1764–1771.

[21] S. Brechtel, T. Gindele, and R. Dillmann, "Recursive Importance Sampling for Efficient Grid-Based Occupancy Filtering in Dynamic Environments," in *Int. Conf. on Robotics and Automation*, 2010, pp. 3932–3938.