2018 IEEE Intelligent Vehicles Symposium (IV)
Changshu, Suzhou, China, June 26-30, 2018

Exploiting Hierarchy for Scalable Decision Making in Autonomous Driving

Ekhlas Sonu, Zachary Sunberg, and Mykel J. Kochenderfer
Department of Aeronautics and Astronautics
Stanford University
Stanford, USA
{esonu,zsunberg,mykel}@stanford.edu

***Abstract*—A major challenge in autonomous driving has been the intractability of planning algorithms. Research has largely focused on simple, short-term scenarios with few interacting traffic participants. We propose a hierarchical approach for long-horizon tactical planning in large-scale autonomous driving settings. Our approach exploits the locality of interactions with other agents by sequentially setting and accomplishing short-term goals involving fewer agents and hence is able to scale to more traffic participants. We demonstrate the effectiveness of our approach on an example highway driving problem where the ego vehicle must safely transit to the farthest lane in order to exit the highway at a designated exit.**

I. Introduction

There are now commercially available vehicles that are capable of performing simple tasks such as lane keeping, adaptive cruise control, and supervised lane changes. However, planning under uncertainty has been mostly of academic interest. A few recent efforts [1], [2] have demonstrated the benefits of accounting for uncertainty in tactical planning for short-term goals, such as lane changes. It has been shown that it can be beneficial to model the behavior of road participants in the decision making process [3].

Even a simple task such as lane changing may require predicting behavior of other traffic participants and computing a safe, smooth, and possibly cooperative trajectories. Due to the uncertainties inherent in the driving environments arising from an incomplete view of the world and ignorance about other drivers' driving behavior and their intentions, Markov decision processes (MDPs) [4] and partially observable MDPs (POMDPs) [5], [6] have emerged as a promising tool for modeling decision making for autonomous driving.

Brechtel et al. [7] proposed the use of continuous state POMDPs for intersection navigation using an offline approach. While their approach makes a compelling case for POMDPs, offline approaches are not easily scalable to general driving problems. Ulbrich and Mauer [1], [2] utilized POMDPs for online planning for single lane change decisions with real-time performance in urban settings. However, their approach was restricted to a short planning horizon with few traffic participants. Longer horizon decisions such as computing an optimal sequence of lane changes were delegated to the strategic level of route planning. Solution algorithms tend to scale poorly to longer horizon problems. This limits their application to short-term tactical tasks such as executing a single lane change.

Currently, there a disconnect between strategic planning (e.g. route planning) and tactical planning (e.g. safe lane changing). We present a hierarchical MDP [8] framework to bridge this gap. Our approach exploits the insight that regardless of the total number of agents involved, the ego vehicle interacts with only a small subset of all agents over a short time horizon. Hence, by decomposing the larger problem into smaller sub-problems and sequentially solving them, the goal can be achieved near-optimally.

Other hierarchical formulation of model predictive control [9] and Markov decision processes [10]–[12] have also been proposed. Unlike these approaches that solely focus on either offline or online solutions, our approach uses both forms of decision making while clearly outlining the division of task between each level. The lower level, which is responsible for ensuring safety, smoothness, and legality of driving actions, is solved online and can override the directives received from the upper level which is solved offline. The resulting conflicts are modeled as uncertainty in state transitions at the upper level. The hierarchy proposed in this paper is most similar to the hierarchical structure MAXQ value decomposition [12] with a focus is on planning rather than learning. In context of the hierarchical structure of behavior for a road user [13], our approach is localized to planning in the maneuver level.

We consider an example highway driving scenario in which the ego vehicle must exit the highway at a specified exit. The ego vehicle must navigate safely through the traffic to reach the farthest lane to exit the highway. We demonstrate the scalability and improved performance of our hierarchical approach compared to a flat MDP solved over a longer horizon and a heuristic approach.

II. Background

This section reviews MDPs and Monte Carlo tree search for modeling and solving decision making problems.

Markov Decision Processes

A Markov decision process is represented as the tuple $\langle S, A, T, R, \gamma\rangle$, where S is the set of the environment states, A is the set of actions available to the agent, T is the transition function, R is the reward function, and γ is the discount factor. The transition function represents the probability of transitioning from the current state s to the next state s' after performing action a, i.e. $T(s, a, s') = \Pr(s' \mid s, a)$. The reward function represents the desirability of taking an action from a given state ($R : S \times A \rightarrow \Re$).

A solution to an MDP is a policy π^*, which maps states to actions that maximizes the discounted rewards received by the agent (also called its value function). For small, discrete-state MDPs, the solution can be computed iteratively using a process known as value iteration. Alternatively, sampling based methods such as Monte Carlo tree search (MCTS) [14] have shown promise in solving continuous or large discrete state problems.

POMDPs generalize MDPs to scenarios with partial and imperfect observability. Since the exact state of the environment is unknown, the solution to a POMDP maps distributions over state space (also known as agent's *beliefs*) to actions. Continuous state POMDPs are a natural representation of decision making problems in autonomous driving scenarios. Although the ideas presented in this work are demonstrated using continuous state MDPs, their generalization to partially observable scenario is straightforward, albeit intractable.

Monte Carlo Tree Search

Monte Carlo tree search is a widely used online approach for approximating MDPs [14]. Starting with the initial state, MCTS progressively constructs a tree with alternating layers of state and action nodes. MCTS associates a value with each action node and returns the action corresponding to the maximum valued child of the root node. We consider a variant of MCTS using upper confidence tree (UCT) for action selection and double progressive widening (DPW) [15] to control the depth of the tree. UCT value is computed as:

$$UCB(s, a) = Q(s, a) + c\sqrt{\frac{\ln N(s)}{N(s, a)}}$$

where $Q(s, a)$ is the current estimate of the action-value, $N(s, a)$ is the number of times action a was taken from state s, $N(s)$ is the number of times state s was visited, i.e. $N(s) = \sum_a N(s, a)$, and c is an exploration constant.

One drawback of MCTS when applied to problems with large or continuous state spaces is that a new state is sampled every time and hence the solution tree tends to be a single level deep. DPW overcomes this problem by limiting the number of children of a state-action node pair (s, a) to $kN(s, a)^\alpha$, where k and α are tunable parameters. When the number of children of the node pair (s, a) are fewer than this value, a new state is added as a child of the node, otherwise one of the existing children is sampled. As $N(s, a)$ grows, so does the number of children, thereby allowing gradual widening of the tree. A similar widening process is also used for sampling the actions allowable from a given node. However, for a problem with few actions, this step has less significance.

III. Problem Definition

We consider a highway driving problem in which the ego vehicle must navigate through traffic to reach the farthest lane in order to take an exit. The vehicle may potentially interact with many other traffic participants during. We simulate this scenario as a discrete time, continuous state MDP. In this section, we describe the environment model, the behavior model of other traffic participants, and the MDP model of the highway driving problem.

Environment Model

We model the state of the environment as the joint local state of individual vehicles, $\boldsymbol{s} = (s_0, s_1, \ldots, s_n)$ where s_0 is the local state of the ego vehicle and s_i is the local state of ith agent in the environment. An agent's local state consists of its pose and longitudinal velocity, i.e. $s_i = \langle x_i, y_i, \dot{x}_i \rangle$. Its control actions consist of its longitudinal acceleration and lateral velocity, i.e. $a_i = \langle \ddot{x}_i, \dot{y}_i \rangle$. The state transition model for each agent is defined using a generative model that uses the kinematic equations with a Gaussian noise added to it.

Driver Model

For both planning and experiment stages, we use a widely used parametric driver model to derive the control actions of other drivers. Our driver models have two components: the *intelligent driver model* (IDM) [16] to control the vehicle's longitudinal acceleration and a variation of the *minimizing overall braking induced by lane-change* (MOBIL) model [17] to control its lane change behavior. Both models have a few parameters that define the driver's behavior. To model a range of driving behavior, we classify the other vehicles' behavior into one of three categories, *timid*, *normal*, and *aggressive*. The parameter values for each class is defined in Table I. Although better models are available in literature, we find IDM and MOBIL models to sufficiently model traffic scenarios to demonstrate the scalability of our approach.

Longitudinal Acceleration Model: The intelligent driver model is a simple model for simulating collision-free in-lane highway and urban driving. The IDM computes the value of longitudinal acceleration of a vehicle based on distance from the leading vehicle (g), the absolute velocity of the vehicle ($\dot{x}$), and velocity relative to the leading vehicle ($\Delta(\dot{x})$).

$$\ddot{x}_{IDM} = a\left[1 - \left(\frac{\dot{x}}{\dot{x}_d}\right)^\delta - \left(\frac{g^*(\dot{x}, \Delta\dot{x})}{g}\right)^2\right]$$

where $g^*(\cdot, \cdot)$ is the desired gap defined as follows:

$$g^*(\dot{x}, \Delta\dot{x}) = g_d + T\dot{x} + \frac{\dot{x}\Delta\dot{x}}{2\sqrt{ab}},$$

where $\dot{x}_d$ is the desired velocity, g_d is the minimum desired gap from the leading vehicle (also known as collision distance), T is the desired time gap from the leading vehicle, and a and b are desired comfortable acceleration and deceleration values. The values of IDM parameters are given in Table I.

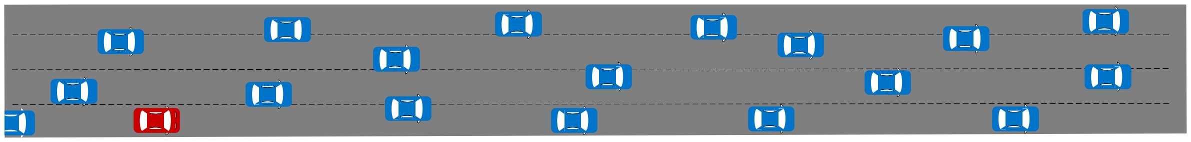
Fig. 1. Illustration of the highway driving problem used. The ego vehicle is shown in red and the other vehicles in blue.

TABLE I
DRIVER MODEL PARAMETERS

IDM Parameter	Aggr.	Timid	Normal
$\dot{x}_d$ (m/s)	27.24	22.76	25.00
T (s)	1.5	1.5	1.5
g_d (m)	2.0	0.5	1.0
a (m/s^2)	1.4	1.4	1.4
b (m/s^2)	2.0	2.0	2.0
MOBIL Parameter			
p	0.0	0.0	0.0
b_{safe} (m/s^2)	2.0	2.0	2.0
a_{thr} (m/s^2)	1.0	2.0	1.5

Lateral Velocity Model: The MOBIL model is used alongside IDM for lane change decisions. The model has two components. The first is a safety check with respect to the next following vehicle which is done by ensuring that the new IDM deceleration of the next following vehicle is greater than a certain threshold, i.e. $\tilde{\ddot{x}}_{follow} \geq -b_{safe}$, which is set to reflect a comfortable braking deceleration. The second is to ensure that the lane change is smooth and non-detrimental to the local traffic flow, which is done by checking that $\tilde{\ddot{x}}_c - \ddot{x}_c + p(\tilde{\ddot{x}}_n - \ddot{x}_n + \tilde{\ddot{x}}_o - \ddot{x}_0) \geq a_{thr}$. Here the terms with tildes are computed as IDM accelerations assuming that the lane change was made instantaneously. The terms without tildes are computed on the current configuration of the environment state. Subscripts c, n, and o represent the vehicle performing lane change, the following vehicle in the target lane and the following vehicle in the current lane respectively. The second condition ensures that when the vehicle changes lane, the combined acceleration (local flow) of the traffic participants is not adversely affected. Our choice of parameter values were dictated by our desire to model self-centered, goal-driven driving behavior by traffic participants. The politeness factor (p) is set to 0 in Table I.

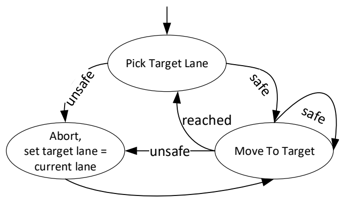

Fig. 2. Controller to augment the MOBIL model.

While MOBIL can determine if a lane change should be initiated, it does not provide a temporally extended control policy for executing the lane change. For extended discrete time control of lane change behavior, we augment the MOBIL model with a simple finite state controller (Figure 2). The controller uses the MOBIL criteria to determine whether to initiate a lane change, continue it if conditions are still favorable, or abort it.

Flat MDP Representation

We present a flat representation of the planning problem that serves as a baseline for our hierarchical approach. A natural representation of the planning problem is as a POMDP. The physical state of immediately neighboring vehicles may be fully observable to the ego vehicle through reliable short-range sensors, but occlusion and sensor uncertainties lead to erroneous observation of distant vehicles. However, such a representation introduces complexity that has little relevance to the hierarchical decomposition that this research focuses on, so instead an MDP representation is used. We propose a scalable MDP representation as baseline, which serves as a theoretical upper bound to the corresponding POMDP. We make a further simplifying assumption of full observability of both physical state and model of other agents without affecting the upper-bound property. We have:

$$MDP = \langle \mathcal{S}, A_0, T, R_0 \rangle \tag{1}$$

- $\mathcal{S} = S_0 \times \prod_{i=1}^{n} S_i$ is the set of states such that for $s_0 \in S_0$, $s_0 = \langle x_0, y_0, \dot{x}_0 \rangle$ is the continuous physical state of the ego vehicle; the state of any other vehicle is $\boldsymbol{s}_i \in S_i$ such that $\boldsymbol{s}_i = \langle s_i, \theta_i \rangle$ consists of the physical state and the driver model of agent i, i.e. $s_i = \langle x_i, y_i, \dot{x}_i \rangle$.
- A_0 is the set of actions available to the ego vehicle agent such that $a_0 = \langle \ddot{x}_0, \dot{y}_0 \rangle$. We restrict the set of actions to 10 discrete actions: three possible values of longitudinal acceleration, $\{-2.0, 0.0, 2.0\}$ (m/s^2), three values of lateral velocities, $\{-2.0, 0.0, 2.0\}$ (m/s), and one additional action of hard braking $\langle -6.0, 0.0 \rangle$.
- $T : \mathcal{S} \times A_0 \times \mathcal{S} \rightarrow [0, 1]$ is the transition function represented in a factorized as a product of individual vehicles' transitions as follows:
$$\Pr(s' \mid s, a_0) = \Pr(s_0' \mid s_0, a_0) \prod_{i=1}^{n} \Pr(a_i \mid s) \Pr(s_i' \mid s_i, a_i)$$
Each agent's action is derived from its driver model and its state is updated using the kinematic equations.
- $R_0 : \mathcal{S} \times A_0 \times \mathcal{S} \rightarrow \Re$ is the reward function, which depends on safety, smoothness, and legality of the ego vehicles' actions besides achievement of the goal, i.e. $R_0(s, a_0, s') = R_{goal}(s') + R_{collide}(s') + R_{smooth}(s, a_0, s') + R_{legal}(s')$.

We model collision states and goal states as terminal states. $R_{goal}(s')$ is 200 if s' is a goal state, $R_{collide}$ is -500 if s'

is a collision state, R_{smooth} is -5 if the ego vehicle brakes hard or by its action causes another vehicle to brake hard, and R_{legal} is $2\times \mid y_0' - y_{lc} \mid$, where y_0' is the y-coordinate of end state and y_{lc} is the center of ego vehicle's lane. R_{legal} discourages the vehicle to drive between two lanes.

Theoretically, the flat representation may be solved using MCTS algorithm to yield a near optimal trajectory for the ego vehicle. However, the large dimensionality and partial observability precludes the use of such a representation in the real world, yet it serves as a baseline for comparison.

IV. Hierarchical Planning Framework

Two factors aggravate the solution complexity of the flat MDP representation: the number of agents present in the environment and the long time horizon of the problem to be solved. To overcome these challenges, we propose a hierarchical framework that exploits the modularity in autonomous driving. We conjecture that many tactical behaviors in autonomous driving can be decomposed as a sequence of simpler subtasks. For example, our highway driving problem may be represented as a sequence of multiple sequential lane changes. Our approach models a two-level hierarchy of actions. The macro-actions [18, Chapter 9] at the upper level serve as the sub-task to be achieved at the lower level. Henceforth, we use the terms subtasks and macro-actions interchangeably.

The key sources of improved scalability obtained by decomposing the problem are: (1) the problem is partially solved offline; (2) the online planning for the subtasks is carried over significantly smaller horizons and hence is more tractable; and (3) during the execution of the subtask, the agent is likely to interact with only a small subset of agents, and so the behavior of fewer agents needs to be modeled.

Upper Level MDP

The upper level MDP consists of a coarse representation of the environment and guides the path planning towards the overall goal. The state space at this level represents a fuzzy view of the world that either ignores the presence of other vehicles or includes partial traffic information. The actions used at this level are the subtasks that are accomplished by solving the lower level planning problem. For our example problem domain, we model the upper level planning as a discrete state MDP on a finite grid with three (macro-)actions for the ego vehicle: one for changing to an adjacent lane in either direction and one for maintaining the current lane. Although additional macro-actions may be added for acceleration or deceleration, we choose a simpler definition for our illustration.

Formally, the upper level MDP is defined as follows:

$$MDP_{ul} = \langle S_{ul}, A_{ul}, T_{ul}, R_{ul} \rangle$$

where S_{ul} is the set of ego vehicle's positions on the grid, A_{ul} = {change-left, keep-lane, change-right} is the set of macro-actions, $T_{ul} : S_{ul} \times A_{ul} \times S_{ul} \rightarrow [0, 1]$ is the transition function where the transition probabilities are set manually (or estimated by running lower level MDP simulations), and R_{ul} is the reward function.

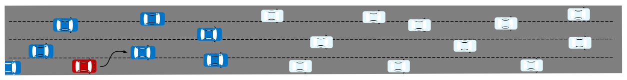
Fig. 3. Illustration of the ego vehicle's neighborhood. During execution of a macro-action, the ego vehicle may only interact with its neighbors.

Since the state and action spaces of the upper level MDP are small and discrete, the upper level MDP can be solved offline using discrete value iteration to yield a compact policy $\pi_{ul} : S_{ul} \rightarrow A_{ul}$ in the form of a lookup table. However, the upper level MDP model does not account for the safety, legality, and comfort of the vehicle's actions, which depend on the exact state of surrounding agents. Such considerations are delegated to the online planning at the lower level. As a result, the macro-action picked may be overridden by the lower level MDP leading to uncertainty in state transition at the upper level.

Lower Level MDP

Given that the macro-action to perform is obtained from the upper-level decision making, the goal of the lower level MDP is to implement it online using atomic actions. The lower level MDP is solved online over a shorter planning horizon (on the order of 3–5 seconds). A key concept at this level of abstraction is that of the ego-vehicle's neighborhood (ν_0). The definition of neighborhood could be tailored depending on the subtask to be performed and the agents it may interact with directly during the execution of the macro-action. However, for simplicity, we maintain a single definition for all subtasks. Formally, in our example, we define an agent's neighborhood as the set of vehicles immediately leading and following the agent on any lane (Figure 3).

For any given subtask, we define the lower level MDP analogous to the flat MDP described in the Section III as follows:

$$MDP_{a_{ul}} = \langle \mathcal{S}_{ll}, A_0, T_{ll}, R_{ll} \rangle$$

where the terms are analogous to Equation (1). $\mathcal{S}_{ll} = S_0 \times \prod_{i \in \nu_0} S_i$, A_0 is the set of atomic actions available to the ego vehicles, T_{ll} is the transition function defined over the smaller set of agents, and R_{ll} is the reward function. The reward function of the lower level MDP ensures safety, smoothness, and legality of driving actions (see Section III for values). Depending on the macro-action, the goal is set to be at the center of the target lane and any value of x and $\dot{x}$.

The lower level decision making is represented as a continuous state, discrete time MDP and can be solved approximately using online sampling-based approaches such as MCTS. Compared to the flat MDP, the decision horizon and number of agents are significantly smaller for the low level MDP.

V. Algorithm

Assuming that ego vehicle's route is precomputed at a strategic level. The upper level MDP is formulated and solved offline using discrete value iteration to yield a policy π_{ul}, the lower level MDP is solved online, hence the exact state of the local environment is available to the agent. Algorithm 1 outlines the hierarchical controller for the ego vehicle.

Algorithm 1 Hierarchical planning for highway driving.

```
Input: $MDP$, $s \in \mathcal{S}$, $MDP_{ul}$, $\pi_{ul}$
Output: $a_0 \in A_0$
1: $s_{ul} \leftarrow$ get_ego_grid_location($MDP_{ul}, s$)
2: $a_{ul} \leftarrow \pi_{ul}(s_{ul})$
3: $\nu_0 \leftarrow$ extract_neighborhood($s, 0$)
4: $s_{ll} \leftarrow \{s_0\} \cup_{i \in \nu_0} \{s_i\}$
5: Formulate $MDP_{a_{ul}}$
6: return $a_0 \leftarrow$ solve_mcts($MDP_{ll}$, $s_{ll}$)
```

The input to the algorithm is the overall driving problem formulated as an MDP, the current state of the environment s, and the upper level MDP MDP_{ul} and its optimal solution π_{ul}. The algorithm begins by extracting the grid location of the ego vehicle s_{ul} using MDP_{ul} and s (line 1). Then, the optimal macro-action a_{ul} is extracted from π_{ul} (line 2). Next, the algorithm extracts the set of neighboring agents ν_0 and the state of the low level MDP s_{ll} (line 3–4). Finally, the lower level MDP is formulated and solved to yield the optimal driving action a_0 (line 5–6) at the current instance. The reward function of the lower level MDP is tailored to prioritize collision avoidance and driving quality constraints over the satisfaction of the sub-goals.

VI. Experiments

The experiments were carried out on a $1200\,\mathrm{m}$ four lane highway where each lane is $4\,\mathrm{m}$ wide. The initial position of the ego vehicle was set at the center of the first lane $1000\,\mathrm{m}$ away from the exit at the end of the highway. Its initial velocity was set at $25\,\mathrm{m\,s^{-1}}$. The goal was set at the end of the highway on the lane furthest from the ego vehicle (i.e. 1000m away in lane 4). For simulations, we assumed a fixed number of agents on the highway distributed uniformly with small Gaussian noise added to their initial positions. Each vehicle was randomly assigned one of three types: *aggressive*, *normal*, and *timid*. The initial velocity of each vehicle was sampled using a normal distribution with a mean at the desired velocity of its type (Table I) and a specified variance of $2.5\,\mathrm{m^2/s^2}$. Here, we report results for varying number of agents (40 to 140) and draw conclusions based on them.

We defined the upper level MDP as a discrete state problem where grids are $75\,\mathrm{m}$ long and one lane width ($4\,\mathrm{m}$) wide, i.e. 64 states. The goal was set on the last cell on the fourth lane. The agent received a reward of 200 for reaching the goal. The upper level MDP was solved offline using discrete value iteration algorithm until convergence. The online planning approach outlined in Algorithm 1 was carried out every $0.3\,\mathrm{s}$ and output a low-level action. At each step, lower level MDP was formulated for the optimal macro-action and solved online using MCTS-DPW to obtain optimal action for the time step (Algorithm 1). The parameters of MCTS-DPW algorithm are defined in Table II. The goal of the low level MDP was set as a range of physical states centered on the y-axis at the middle of the target lane.

TABLE II
MCTS-DPW PARAMETERS

Parameter	Value
Time step (Δt)	0.3s
Horizon (h)	15
Iterations (iter.)	1500
Exploration Constant (c)	10
DPW Linear Action Parameter (k_a)	10.0
DPW Exp. Action Parameter (α_a)	0.1
DPW Linear State Parameter (k_s)	5.0
DPW Exp. State Parameter (α_s)	0.1

We compare performance of the hierarchical approach with the flat MDP and a heuristic controller. The solution to the flat MDP was obtained using MCTS-DPW with identical parameters as in Table II except the planning horizon was set to 105. For the heuristic approach, the ego vehicle was modeled using the driver model described in Section III with the same parameter values as a *normal* agent. However, for the heuristic agent, lane changes were restricted to the direction of the goal. The results of the experiments are shown in Table III.

TABLE III
PERFORMANCE RESULTS

n		Avg. Sol. Time (s)	Succ./ rate	Coll./ rate	#HB/100 steps	Avg. Dev. (m)
40	Heur.	0.03	0.85	0.09	0.14	0.60
	Flat*	1308	0.90	0.00	0.10	0.94
	Hier.	44.36	1.00	0.00	0.05	0.80
60	Heur.	0.04	0.60	0.25	0.17	0.47
	Flat*	1905	1.00	0.00	0.07	0.93
	Hier.	61.74	0.96	0.01	0.03	0.75
80	Heur.	0.06	0.50	0.25	0.14	0.34
	Flat*	2262	1.00	0.00	0.07	1.00
	Hier.	82.74	0.89	0.00	0.03	0.70
100	Heur.	0.10	0.17	0.05	0.08	0.14
	Flat*	—	—	—	—	—
	Hier.	115.66	0.62	0.01	0.03	0.65
120	Heur.	0.12	0.09	0.05	0.05	0.08
	Flat*	—	—	—	—	—
	Hier.	145.41	0.42	0.01	0.02	0.58
140	Heur.	0.14	0.08	0.04	0.04	0.07
	Flat*	—	—	—	—	—
	Hier.	171.82	0.27	0.00	0.02	0.53

The experiments were run with 100 random starting configurations for the heuristics and the hierarchical approaches. We started with 10 random starting configurations for the flat MDP representation of the problem due to the runtime constraints. The relevant metrics used to determine the quality of driving are:

- **Average solution time**, which measures the scalability of the approach and hence its practicality as a real-time decision making tool for autonomous driving.
- **Success rate**, which determines how effective the approach was in achieving its goal.

- **Collision rate**, which is a measure of the safety property.
- **Hard brake rate** and **average deviation** from lane center, which are a measure of smoothness and legality.

The heuristic method requires negligible computation time, which is expected since it takes a shortsighted approach, ignoring the future motion of other agents. On the other hand, the flat MDP representation takes a farsighted approach modeling the motion of every agent for a long time horizon. This level of modeling impacts the solution time, rendering it impractical for real-time driving. On average, the flat MDP needed over an hour to solve for a single instance of the problem involving 100 or more agents. Hence, the results are not included. The hierarchical approach strikes a balance between the two approaches. While the solution time is up to 4 times slower than real-time ($\sim$40s), significant scalability can be achieved by parallelizing MCTS [19] is a possibility.

In terms of success rate, the performance the heuristic approach is evidently inferior to the other approaches once again due to lack of foresight in modeling the goal and the behavior on of the other agents. Although the results for the flat MDP and the hierarchical approaches seem comparable, conclusions may not be drawn based on the insufficient number of simulations for the flat MDP. We observe a decline in success rate as the number of agents increases. This is expected because the increasing number of agents adversely affects the traffic density and consequently the success rate.

Similar trends continue when comparing collision rate and hard braking rate. While IDM and MOBIL models generally yield collision free driving behavior, most collisions observed in the heuristic approach were a result of the ego vehicle and a second vehicle moving into the same lane at the same time from different directions, a case not handled by IDM and MOBIL. We observe a decline in collision rate in experiments with $\geq$100 vehicles because the number of lane change attempted by the ego vehicle were fewer. The few collision cases in the hierarchical approach may be avoided by adding additional collision avoidance measures such as reactively aborting the current upper-level directive from states that are deemed dangerous. Finally, in terms of average distance from lane center, larger values for the hierarchical approach suggest that the transition between lanes are not smooth. This effect may be overcome by using model predictive control (MPC) to smooth the trajectory. Additionally, the upper-level MDP may be augmented by including partial information about neighbor state that would yield a more conservative lane change policy.

VII. Conclusion

We presented a simple and scalable hierarchical framework for autonomous driving and demonstrated its merits on an example highway driving problem. We argue that since autonomous driving is inherently modular, such frameworks may be used for many driving scenarios. Cooperative tasks such as negotiating right-of-way may be delegated to lower levels that capture short-term interactions with neighboring agents. Upper level planning can guide the vehicle to achieve its overall goal. Furthermore, the online planning could be augmented with safety overrides and the state representation at the upper level could be augmented with discrete neighborhood information to dissuade lane changes in precarious situations and hence ensure smooth lane transition.

Acknowledgments Toyota Research Institute (TRI) provided funds to assist the authors with their research, but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity.

References

[1] S. Ulbrich and M. Maurer, “Probabilistic online POMDP decision making for lane changes in fully automated driving,” in *IEEE Conference on Intelligent Transportation Systems*, Oct 2013, pp. 2063–2067.

[2] ——, “Situation assessment in tactical lane change behavior planning for automated vehicles,” in *IEEE Conference on Intelligent Transportation Systems*, Sept 2015, pp. 975–981.

[3] Z. N. Sunberg, C. Ho, and M. J. Kochenderfer, “The value of inferring the internal state of traffic participants for autonomous freeway driving,” in *American Control Conference*, 2017.

[4] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley & Sons, 1994.

[5] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, “Planning and acting in partially observable stochastic domains,” *Artificial intelligence*, no. 1, pp. 99–134, 1998.

[6] M. J. Kochenderfer, *Decision Making Under Uncertainty: Theory and Application*. MIT Press, 2015.

[7] S. Brechtel, T. Gindele, and R. Dillmann, “Probabilistic decision-making under uncertainty for autonomous driving using continuous POMDPs,” in *IEEE Conference on Intelligent Transportation Systems*, Oct 2014, pp. 392–399.

[8] M. Hauskrecht, N. Meuleau, L. P. Kaelbling, T. Dean, and C. Boutilier, “Hierarchical solution of Markov decision processes using macro-actions,” in *Conference on Uncertainty in Artificial Intelligence*, 1998, pp. 220–229.

[9] R. Scattolini and P. Colaneri, “Hierarchical model predictive control,” in *IEEE Conference on Decision and Control*, 2007, pp. 4803–4808.

[10] A. McGovern, D. Precup, B. Ravindran, S. Singh, and R. S. Sutton, “Hierarchical optimal control of MDPs,” in *Proceedings of the Tenth Yale Workshop on Adaptive and Learning Systems*, 1998, pp. 186–191.

[11] R. Parr and S. J. Russell, “Reinforcement learning with hierarchies of machines,” in *Advances in Neural Information Processing Systems*, 1998, pp. 1043–1049.

[12] T. G. Dietterich, “Hierarchical reinforcement learning with the maxq value function decomposition,” *Journal of Artificial Intelligence*, pp. 227–303, 2000.

[13] J. A. Michon, *A critical view of driver behavior models: what do we know, what should we do?* Springer, 1985, pp. 485–524.

[14] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, “A survey of Monte Carlo tree search methods,” *IEEE Transactions on Computational Intelligence and AI in Games*, no. 1, pp. 1–43, March 2012.

[15] A. Couëtoux, J.-B. Hoock, N. Sokolovska, O. Teytaud, and N. Bonnard, *Continuous Upper Confidence Trees*. Springer, 2011, pp. 433–445.

[16] M. Treiber, A. Hennecke, and D. Helbing, “Congested traffic states in empirical observations and microscopic simulations,” *Phys. Rev. E*, pp. 1805–1824, 2000.

[17] M. Treiber and A. Kesting, *Modeling lane-changing decisions with MOBIL*. Springer, 2009, pp. 211–221.

[18] M. Wiering and M. van Otterlo, *Reinforcement Learning: State-of-the-Art*. Springer, 2014.

[19] G. Chaslot, M. H. Winands, and H. J. van Den Herik, “Parallel Monte-Carlo tree search,” *International Conference on Computers and Games*, pp. 60–71, 2008.