2016 IEEE 55th Conference on Decision and Control (CDC)
ARIA Resort & Casino
December 12-14, 2016, Las Vegas, USA

Hierarchical Reasoning Game Theory Based Approach for Evaluation and Testing of Autonomous Vehicle Control Systems

Nan Li[1], Dave Oyler[1], Mengxuan Zhang[1], Yildiray Yildiz[2], Anouck Girard[1] and Ilya Kolmanovsky[1]

***Abstract*— A hierarchical game theoretic decision making framework is exploited to model driver decisions and interactions in traffic. In this paper, we apply this framework to develop a simulator to evaluate various existing autonomous driving algorithms. Specifically, two algorithms, based on Stackelberg policies and decision trees, are quantitatively compared in a traffic scenario where all the human-driven vehicles are modeled using the presented game theoretic approach.**

I. INTRODUCTION

Predictive models of driver actions in complex traffic scenarios have several potential applications for autonomous vehicle control. Firstly, these models can be employed in simulators to generate realistic traffic scenarios, which can be utilized for testing, verification and validation, and comparison of competing autonomous driving control algorithms. Such simulators can save time in the development phase by providing a model-based testing environment, before the actual road tests. Secondly, these models can be used in the design of hierarchical control schemes for driverless cars: typically, in an autonomous vehicle, a higher level outer loop controller generates the reference trajectories for the lower level inner loop controller, which determines the steering angles, acceleration/deceleration inputs, etc., required to follow the reference trajectory [1]. Predictive driver models can be utilized in the higher level outer loop controller generating the reference trajectories for the lower level inner loop controller, thereby ensuring similar behavior to that of a human-driven vehicle and improving the comfort level of the passengers [1]. In addition, these models can provide predictions of the future trajectories of the vehicles in the vicinity of the host autonomous vehicle and be used as inputs for the inner loop controllers such as model predictive controllers (MPC) [2]–[4].

The literature on driver modeling is vast. In [5] and [6], Hidden Markov Model (HMM) based driver models are considered, which are developed using real driving data. In [7] and [8], k-means clustering is used to determine the driving mode, and an approach to predict and overbound future vehicle trajectory is proposed. It is shown that a prediction of the driver inputs can improve the performance of an assisted driving algorithm. In [9], a "cognitive architecture" approach, which is "a computational framework that incorporates built-in, well-tested parameters and constraints on cognitive and perceptual-motor processes," is utilized for driver modeling. Built in logical (if-then-else) rules are used to represent the decision making process. In [10], lane change behavior of drivers is modeled using a multi agent simulation system called "Simulation of Intelligent TRAnsport Systems (SITRAS)." Several logical algorithms are used to model the decision making during lane changes. The resulting actions of the drivers are therefore predefined with strict rules. Driver aggressiveness can also be incorporated into the model by tuning certain parameters.

Some other references represent drivers as feedback controllers (e.g., see [11]–[15]). In [16], another driver model is proposed, in which support vector machines together with a Bayesian filter are used to capture the intention of a driver for a lane change, which can then be used as an input to a driver assistance system. The method uses local measurements such as the lateral position in a lane, the steering angle, and the derivatives of these variables, to predict a lane change before it occurs. A comprehensive list of existing human driver models, control based or behavioral based, can be found in [17], [18].

With respect to the existing approaches, the present paper is distinguished by advanced modeling of driver-to-driver interactions in traffic scenarios using a specific game theoretic formulation which is scalable to multiple vehicles. The proposed method of traffic modeling has the following advantages: a) driver responses to environmental changes are determined utilizing a human decision making process, instead of assuming that the actions of the drivers are known in advance for a given state of the system, b) multiple human-human and human-automation (e.g., autonomous cars) interactions can be modeled simultaneously, which helps investigate traffic scenarios with several vehicles and c) all the vehicles in a traffic scenario can simultaneously be modeled as decision makers (as opposed to predicting the decisions of one vehicle while assuming that the rest of the vehicles move based on certain kinematic and dynamic constraints), in a computationally tractable way.

The exploited game theoretic model makes it possible to conduct a quantitative analysis of the traffic. For example, a) the increase in the number of accidents corresponding to the increase in the traffic density can be estimated, b) the effect of certain parameter value selections in an autonomous driving algorithm on the safety of the vehicle can

*The research of Ilya Kolmanovsky and Nan Li is supported by the National Science Foundation under Award Number CNS-1544844. The research of Yildiray Yildiz is sponsored by the Scientific and Technological Research Council of Turkey under grant number 114E282.

[1]Nan Li, Dave Oyler, Mengxuan Zhang, Anouck Girard and Ilya Kolmanovsky are with the Department of Aerospace Engineering, University of Michigan, 1320 Beal Avenue, 48109-2140 Ann Arbor, MI, USA {nanli, dwoyler, mengxuan, anouck, ilya}@umich.edu

[2]Yildiray Yildiz is with the Department of Mechanical Engineering, Bilkent University, 06800 Cankaya, Ankara, Turkey yyildiz@bilkent.edu.tr

be evaluated, c) competing autonomous driving algorithms can be compared quantitatively in a multi-vehicle time-extended scenario, based on certain safety and performance metrics, and finally, d) these quantitative analyses can be used for optimization purposes based on a predefined objective function including safety and performance measures. What makes the above mentioned analyses possible is a method that uniquely combines game theory, which is used to model human interactions, and reinforcement learning, which is used to obtain the policy which models the driver actions. At the core of this method is an approach known as "semi network-form games" [19], which helps us obtain the probable outcomes of a complex traffic scenario in the presence of multiple driver-driver interactions.

There have been other game theoretic approaches proposed to model highway driving, such as [20] and [21]. Although these approaches exploit driver interaction models developed in a game theoretic setting, they did not consider dynamic (multi-move) scenarios. The latter are exploited in [22] for Hybrid Electric Vehicle (HEV) energy management where the driver and the powertrain are considered to be two players in a game. However, increasing the number of players (drivers, in our case) beyond three complicates computing a Stackelberg solution, especially in a time extended (multi-move) scenario. On the other hand, the hierarchical reasoning based game theoretic approach exploited in this paper is easily scalable. Indeed, an implementation of the proposed approach for a 50 player game can be found in [23], and scenarios with up to 25 vehicles are treated in this paper.

Some preliminary results of using our game theoretic approach for driver modeling presented in this paper have been published in [24], including the application of reinforcement learning and the demonstration of the effect of driver level on the number of lane changes and driving safety. In this paper, we show that a traffic simulator consisting of interacting drivers that are modeled using the proposed approach can be utilized to quantitatively and comparatively evaluate different autonomous driving methods. As case studies, we present simulation results for two different autonomous driving methods, based on Stackelberg equilibrium policies and decision trees, and compare them in terms of safety and performance.

The organization of this paper is as follows: The problem formulation is given in Section II. The process of obtaining driver policies by exploiting game theory and reinforcement learning is explained in Section III. The autonomous driving algorithms to be tested and compared are described in Section IV. Simulation results are reported in Section V. Finally, a summary is given in Section VI.

II. Problem formulation

The problem we treat in this paper is to model the behavior of drivers in a traffic scenario where the cars are driven on a 3-lane highway. We then demonstrate that such models can be used in simulators to evaluate autonomous vehicle policies. Fig. 1 shows an example scenario with 6 cars. Note that using the method we propose, scenarios with more cars and more lanes can be handled.

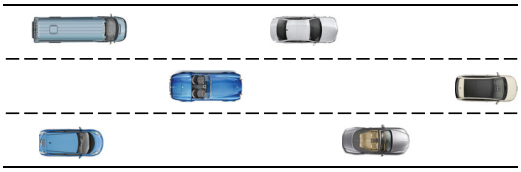

Fig. 1: Traffic in a 3-lane highway.

In this scenario, the cars are assumed to be traveling in the same direction and driven by human drivers who obey the general traffic laws.

A. Action space

Drivers are assumed to have 5 basic actions:

1) "Maintain" current speed,
2) "Accelerate" at rate = 2.5 $\mathrm{m/s^2}$, provided velocity does not exceed 98 $\mathrm{km/h}$,
3) "Decelerate" at rate = -2.5 $\mathrm{m/s^2}$, provided velocity is above 62 $\mathrm{km/h}$,
4) Change lane to the left,
5) Change lane to the right.

This action space may represent typical actions that human drivers may use in highway traffic. A larger action space may improve the fidelity of the model, while also increasing its complexity. The exploration of larger action spaces is left to future work.

B. Observation space

In real traffic flows, a driver can neither observe nor process all the information about all the cars on the road. A human can possibly observe and use the information he/she obtains from the cars in a certain vicinity of him/her. Therefore, we assign the following observation space to the drivers:

1) The longitudinal distance (range) to the car in front and in the same lane (front center), quantified as "close" (range $\leq$ 30m), "nominal" (30m$<$ range $\leq$ 60m) or "far" (range $>$ 60m),
2) The range to the car in front and to the left (front left), quantified as "close," "nominal" or "far,"
3) The range to the car in front and to the right (front right), quantified as "close," "nominal" or "far,"
4) The range to the car in the rear and to the left (rear left), quantified as "close," "nominal" or "far,"
5) The range to the car in the rear and to the right (rear right), quantified as "close," "nominal" or "far,"
6) The relative motion of the car in the front center, quantified as "approaching" (range decreasing), "stable" (range not changing), or "moving away" (range increasing),
7) The relative motion of the car in the front left, quantified as "approaching," "stable" or "moving away,"

8) The relative motion of the car in the front right, quantified as "approaching," "stable" or "moving away,"
9) The relative motion of the car in the rear left, quantified as "approaching," "stable" or "moving away,"
10) The relative motion of the car in the rear right, quantified as "approaching," "stable" or "moving away,"
11) The lane of the car, quantified as "left lane," "center lane" or "right lane."

Similar to the action space, a larger observation space with more observations may improve the performance of the model, but may also entail heavier computational effort.

C. Reward function

A "reward function" is a mathematical representation of the goals of a driver. Basic goals of the drivers in real traffic are 1) to not have a collision (safety), 2) to minimize the time needed to reach the destination (performance), 3) to keep a reasonable headway from preceding cars (comfort) and 4) to minimize driving effort (comfort).

These goals can be reflected in a reward function given by

$$R = w_1 c + w_2 v + w_3 h + w_4 e, \tag{1}$$

where w_i, $i = 1,2,3,4$, are the weights for each term and c, v, h and e represent "collision," "velocity," "headway" and "effort" metrics, respectively.

The weighting terms w_i may change depending on the aggressiveness of the driver, but intuitively, collision avoidance should be the most important factor. Thus, the following relationship between the weights should be kept: $w_1 \gg w_2, w_3, w_4$.

These terms are further explained below.

c **(collision):** The term c gets the value of -1 when a collision occurs and the value of 0 otherwise. This term enforces driver's safety.

v **(velocity):** The term v gets the value of $v =$ (current velocity $-$ nominal velocity) $\div$ (acceleration rate), where "nominal velocity" = 80 km/h. This term encourages the driver to pursue higher traveling speed and shorten travel time.

h **(headway):** The term h is set to the following values depending on the headway distance (range to the car in front)

$$h = \begin{cases} -1 & \text{if} \quad \text{headway is "close,"} \\ 0 & \text{if} \quad \text{headway is "nominal,"} \\ 1 & \text{if} \quad \text{headway is "far."} \end{cases} \tag{2}$$

This term reflects the observation that a larger headway distance improves the safety and comfort of the driver.

e **(effort):** The term e gets the value of 0 if the driver's action is "maintain" and -1 otherwise. This term discourages the driver from making unnecessary maneuvers.

D. Car dynamics

It is assumed that all cars accelerate/decelerate at $\pm 2.5 \text{m/s}^2$, and lane changes occur with constant lateral velocity such that the total time to change lanes is 3s. We also assume that during lane changes, the longitudinal velocity remains constant, and once a lane change begins, it always continues to completion.

III. Obtaining Hierarchical Reasoning Based Driver Models

A policy is defined as a stochastic map from the observation space (see Section II-B), to the action space (see Section II-A), i.e., the drivers have a probability distribution over their possible actions corresponding to each observation state. To obtain driver policies, two main tools are exploited: the hierarchical reasoning game theoretic (also called "level-k") approach, and the Jaakkola reinforcement learning algorithm. In this section, both of these tools are explained.

A. Level-k reasoning

The driver models developed in this work are based on the observation that humans make decisions exploiting various levels of reasoning. The lowest level, level 0, represents an intelligent agent (driver) who chooses his/her actions without considering the possible actions of other agents. For example, if a driver decides to make a lane change, say, from lane A to lane B, without considering the possible actions of the other drivers in the vicinity, that driver is referred to as a level 0 driver. However, if the same driver assumes that the other drivers are level 0 drivers, and then chooses his/her actions as the best response to the possible actions of other drivers, then he/she is referred to as a level 1 driver. So, a level k driver assumes that the rest of the drivers are level $(k-1)$ and acts accordingly. More detailed explanations about this approach can be found in [25] and [26].

Level 0 policy: In general, level 0 policies are considered as "reflexive" behavior, the kind of actions one takes without really taking into account other players' possible actions. These actions can be random, meaning that every possible action is given the same probability of realization given a state, or it can be a simple but reasonable policy that is formed using very basic principles of the scenario one is in. In our scenario, the level 0 policy is formulated as follows:

$$\text{action}_{l0} = \begin{cases} \text{"Decelerate,"} & \text{if the car in front is "close"} \\ & \text{and "approaching,"} \\ \text{"Maintain,"} & \text{otherwise.} \end{cases} \tag{3}$$

Note that a level 0 driver would never change lanes. As a matter of fact, it may not be an uncommon assumption in the simulations of autonomous driving algorithms that other vehicles in traffic do not change lanes, e.g., see [27], [28]. However, in this work, the level k ($k \geq 1$) drivers would make lane changes to pursue higher rewards, which makes the traffic model developed by our approach of higher fidelity compared to some of previous traffic models.

B. Jaakkola reinforcement learning

The Jaakkola reinforcement learning (RL) algorithm (see [29]) is similar to other conventional RL methods (see [30]). It involves a policy evaluation step, where state-action pairs of a policy are assigned values based on the cumulative rewards gained, and a policy improvement step, where the existing policy is refined so that actions that have higher expectations of cumulative reward values have

increased probability of actually being chosen. A feature of the Jaakkola RL algorithm is that when the underlying dynamics of the problem is Markov while the system states are only partially observable to the agents, i.e., the problem is modeled as a Partially Observable Markov Decision Process (POMDP), the Jaakkola RL was shown to be able to converge at least to a local maximum in terms of the average over infinite horizon rewards under certain conditions [29]. Note that the highway problem defined in this paper is modeled as a POMDP, due to the drivers' restricted observation spaces (see Section II-B).

C. Complete algorithm

The complete algorithm is a combination of the level-k reasoning game theory and the Jaakkola reinforcement learning approach.

To obtain a level k driver policy, we assign level $(k-1)$ policies to all the drivers in the traffic except the driver we want to "train." By training we mean that we run the Jaakkola RL algorithm where the trained driver is the *learner* and the rest of the drivers, together with the vehicles, constitute the *environment*.

To start the procedure, we first assign a level 0 policy (see Section III-A) to all the drivers except the *learner*, and then train a level 1 policy and save it. We then train a level 2 policy by assigning the level 1 policy we just saved to all the other drivers, and train and save a newly obtained level 2 policy. This can continue until we reach the depth of reasoning (level k) we want to achieve. It is shown in some experimental studies (see [26]) that humans are generally level 0, 1, or 2 players in a game. Therefore, we train driver policies up to level 2 in this paper.

IV. Evaluating Autonomous Driving Approaches

Two autonomous driving strategies, based on Stackelberg policies and decision trees, are evaluated and compared using a simulator in which the traffic, other than the host vehicle, consists of drivers modeled using the above level-k approach. Below, brief explanations of the Stackelberg and decision tree approaches are provided. See [20], [21], [27], and [31] for further details.

A. Stackelberg policies

To generate Stackelberg policies for the autonomous vehicle, we consider three vehicles as players, and the rest of the vehicles are considered to be a part of the environment. The three players are assigned roles as the "leader," "first follower," and "second follower," and they choose their actions sequentially, where the leader chooses its action first, followed by the first follower, and, finally, the second follower. Each player evaluates its actions according to a utility function that consists of two parts. The first, referred to as the positive utility, is defined as follows:

$$U_{pos} = \begin{cases} \min(d_{\triangle}, d_v), & \text{if there is a vehicle ahead,} \\ d_v, & \text{otherwise,} \end{cases} \quad (4)$$

where $d_{\triangle}$ is the distance to the car in front, and d_v is the maximum visibility distance. The second part of the utility is referred to as the negative utility:

$$U_{neg} = d_{\nabla} - v_r T - d_{min}, \quad (5)$$

where d_{∇} and v_r are, respectively, the distance to and the relative velocity of the car behind, T is a prediction time window, and d_{min} is the minimum distance required to allow a lane change; here, d_{min} is set to the car length. Thus, overtaking vehicles are taken into consideration, and lane changes that cut off overtaking vehicles are discouraged.

The actions chosen by the leader, first follower, and second follower, denoted by γ_ℓ, γ_{f1}, and γ_{f2}, respectively, are the Stackelberg equilibrium actions, i.e., the leader chooses its actions to maximize its utility for the worst-case actions that the two followers might choose. Thus, the leader chooses:

$$\gamma_\ell^* \in \underset{\gamma_\ell}{\operatorname{argmax}} \min_{\gamma_{f1},\gamma_{f2}} \left[U'_{pos} + U'_{neg} \right], \quad (6)$$

where U'_{pos} and U'_{neg} are the utilities that correspond to a specific set of actions $\{\gamma_\ell, \gamma_{f1}, \gamma_{f2}\}$. The two followers maximize their own utilities with the known choice of γ_ℓ. In this paper, when constructing Stackelberg policies, the host vehicle is the leader, and the two cars immediately behind are the followers. Alternatively, the host vehicle could be one of the followers instead of the leader, and this can be treated similarly.

Note that [20], [21] consider different vehicle dynamics than the ones in this paper. Additionally, [20], [21] consider uncertainties in recognition distance, side-viewing, and response delays, but these are not considered in this paper.

B. Decision tree policies

In the decision tree approach to autonomous driving, all vehicles except the host vehicle are a part of the environment, and it is assumed that the environment evolves deterministically over a planning horizon (the vehicles maintain their current lanes and velocities), independently of the host vehicle's actions. The host vehicle maintains its current lane and velocity unless there is an obstruction in front of it within a critical headway distance (60m, in this work). If an action is required, a path planner is triggered that evaluates a specified number of pre-selected action profiles by building a tree of potential action sequences and evaluating each branch according to a specified metric. Branches are pruned from the decision tree if they lead to collisions or other unsafe behaviors.

In this paper, the decision tree consists of two layers, where each layer allows the five actions listed in Section II-A. Thus, the action profiles consist of two actions each, and 25 profiles are evaluated. The evaluation metric is the sum of the headway distances after each action.

V. Simulation Results

A. Environment and set-up

We model the environment as follows: the width of a lane is 3.6m, and all cars are modeled as 6m x 2m boxes.

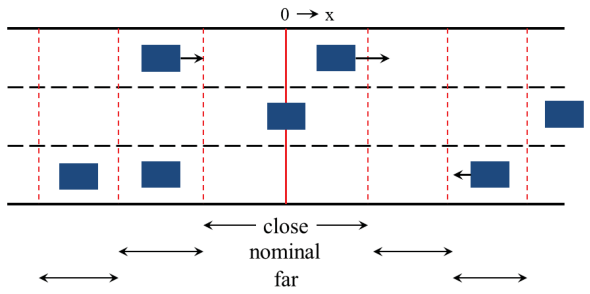

Fig. 2: Simulation Environment.

Cars always drive at the center of a lane unless they are changing lanes. All cars accelerate or decelerate at $\pm 2.5\text{m/s}^2$, and lane changes occur with constant lateral velocity such that the total time to change lanes is 3s. During lane changes, the longitudinal velocity remains constant, and once a lane change begins, it always continues to completion. The longitudinal axis is called x, and its origin is collocated with the car that is to be trained or evaluated.

Five cars are observable, as described in Section II-B, and a car is considered "close" if its relative longitudinal position, $x_c \leq 30$, "nominal" if $30 < x_c \leq 60$, and "far" if $60 < x_c \leq 90$, where 90m is considered to be the maximum visibility distance. Cars more than 90m away are considered to be out of visual range and unobservable. If no car can be observed in a position, it is considered equivalent to a car that is "far" and "moving away."

Fig. 2 shows a snapshot of an example simulation with three lanes. The rectangles represent cars, which are all moving to the right, and the arrows show the velocities of the cars relative to the car under evaluation, which is located in the center lane at $x = 0$. The observation is as follows:

- Front left: close, moving away.
- Front center: far, moving away.
- Front right: far, approaching.
- Rear left: nominal, approaching.
- Rear right: nominal, stable.
- Lane: center.

Note that two cars are unobservable in this scenario. The car in the front center position is beyond visual range, so the corresponding observed status is "moving away" even though it is actually "stable." Also, the car in the rear right "far" position is hidden by the car in the rear right "nominal" position. This reflects the POMDP nature of the problem, as discussed previously.

Initialization of a simulation requires the specification of the following values:

1) the number of lanes, n_ℓ,
2) the number of cars, n_c,
3) the maximum allowable initialization distance, x^0_{max},
4) the simulation duration, t_f.

When a car is initialized, its position is assigned to a lane randomly based on uniform distribution, and then it is placed within that lane randomly based on uniform distribution in $[-x^0_{max}, x^0_{max}]$ such that its distance to any previously initialized cars is at least 30m. Also, the car is initialized with random longitudinal velocity uniformly distributed in the range $[62, 98]$ km/h and assigned the action "Maintain." The car is then assigned a policy to follow (level 0, 1, or 2). This process repeats until all cars have been initialized, and then the simulation proceeds according to Algorithm 1.

```
t = 0;
while t < t_f do
    foreach car do
        Get observation from environment;
        Select action according to policy and
         observation;
        Update position and relative velocity according
         to action;
    end
    if training a policy then
        Evaluate reward function for trainee;
        Update value function;
    end
    if the host car is in a collision state then
        End the simulation;
    end
    t = t + Δt;
end
```

Algorithm 1: Single Episode Simulation.

B. Training level-k driver models

When training a new policy, the observation value function, V, for an observed message m, and the action value function, Q, for a message/action pair (m,a), are initialized as follows:

$$\begin{aligned} &\forall m, \quad V(m) = 0; \\ &\forall m, \forall a, \quad Q(m,a) = 0. \end{aligned} \tag{7}$$

For each observation, the actions are assigned equal probability of selection at initialization. And during each policy improvement step, if

$$\max_a Q(m,a) > V(m), \tag{8}$$

then 0.01 is added to the probability of selecting $\text{argmax}_a Q(m,a)$, after which the action probabilities are normalized.

The observation space described in Section II-B has 3^{11} different observations. In order to ensure that the learning algorithm is exposed to a large portion of the observation space, the trainee needs to be exposed to both sparse and dense traffic. Therefore, during training, the number of cars in the traffic is selected randomly, based on uniform distribution, where $0 \leq n_c \leq n_c^{max}$. The maximum number of cars, n_c^{max}, is chosen based on the number of lanes and x^0_{max} such that if n_c^{max} cars are placed in the environment, the road is near full capacity.

Finally, after sufficient training time, we assign the level 0 policy to any observations that were encountered too few

times during the training for the policy to converge (i.e., not well trained), so that in such rarely encountered observation states, a conservative action is performed.

Training then proceeds according to Algorithm 2.

```
step=0;
while step < desired training cycles do
    Randomly select n_c ∈ [0, n_c^max];
    Initialize all cars with level (k − 1) policies;
    Evaluate the level k policy using Algorithm 1;
    Improve the policy;
    step=step+1;
end
Assign level 0 policy to the not well trained observation states.
```

Algorithm 2: Training Process.

Fig. 3 shows the average reward evolution during level 1 and level 2 training. The weights we choose are $w_1 = 10000$, $w_2 = 5$, $w_3 = w_4 = 1$. They can alternatively be calibrated based on real traffic data, but this will be addressed in our future work.

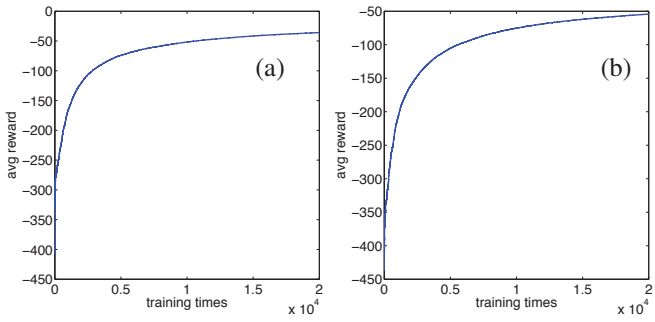

Fig. 3: Average rewards. (a) Training for level 1. (b) Training for level 2.

After training, we run simulations to check the collision rate of the trained policy, where collision rate is a metric for safety defined as the proportion of simulation runs during which the host vehicle touches another vehicle. Each simulation is 200s long, and 1,000 simulations are run for each number of cars, which represents traffic density. As Fig. 4 shows, the level 2 player has higher collision rates because it is in a level 1 environment, whose dynamics are harder to predict than a level 0 environment where the level 1 player performs well.

C. Comparison of Stackelberg and decision tree policies

We consider a traffic environment where 10% of the drivers make decisions based on level 0 policies, 60% of the drivers act based on level 1 policies and 30% use level 2 policies. These percentages of various levels are based on an experimental study conducted in [26]. Fig. 5 shows the collision rates for the Stackelberg and decision tree policies vs. the number of cars in the environment. Again, each

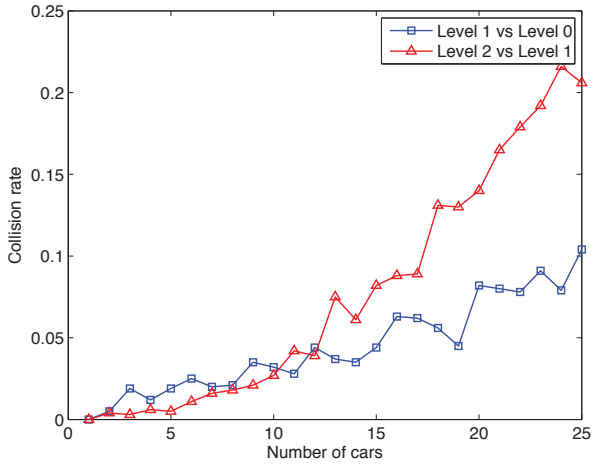

Fig. 4: Collision rates for level 1 and level 2 policies.

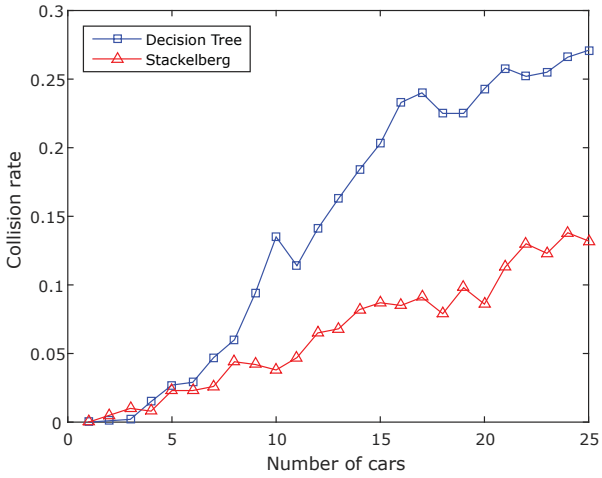

Fig. 5: Collision rates observed in the traffic simulator configured with 10% level 0, 60% level 1, and 30% level 2 drivers.

simulation is 200s long, and 1,000 simulations are run for each number of cars.

As Fig. 5 shows, both approaches experience collisions in the simulations. One explanation for this is that both algorithms may make erroneous assumptions about the environment. The Stackelberg approach assumes that only the host vehicle and two others are decision makers, while the decision tree approach assumes that only the host vehicle is a decision maker. In these simulations, however, all vehicles make observations and take actions to maximize their rewards. Furthermore, not all collisions represented in Fig. 5 result from poor decisions of the host vehicle. Other vehicles in the environment may also be at fault.

In Fig. 5, the Stackelberg policy has fewer collisions. One explanation for this is that the Stackelberg approach considers two other vehicles as decision makers, whereas in the decision tree approach, the host vehicle is assumed to be the only decision maker. However, both approaches can

be paired with lower level controllers to provide additional collision avoidance capability as is done in [31]. Note that the Stackelberg policy requires more measurements because it must constantly measure the headway in each of the three lanes as well as the positions and velocities of the two followers. On the other hand, during most of the simulation time, the decision tree policy only requires a measurement of the headway in its current lane, and additional measurements are only required when changing lanes.

Note that the above implementations of the two policies are relatively simple, and serve as case studies to show the functionality of the traffic simulator developed using the level-k driver models in testing and evaluation of autonomous vehicle policies. More evolved implementations of these policies and their comparisons are left to future publications.

VI. Summary

In this paper, we described a simulator for testing and comparing autonomous driving algorithms in terms of predefined metrics. The simulator consists of human-driven vehicles whose drivers are modeled as strategic decision makers using a game theoretic decision making process. As case studies, autonomous driving algorithms based on Stackelberg policies and decision tree policies were tested. Their performances were compared according to a safety metric.

References

[1] A. Carvalho, S. Lefevre, G. Schildbach, J. Kong, and F. Borrelli, "Automated driving: The role of forecasts and uncertainty-a control perspective," *European Journal of Control*, vol. 24, pp. 14–32, 2015.

[2] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 3, pp. 566–580, 2007.

[3] P. Falcone, F. Borrelli, H. Tseng, J. Asgari, and D. Hrovat, "Linear time varying model predictive control and its application to active steering systems: Stability analysis and experimental validation," *International Journal of Robust and Nonlinear Control*, vol. 18, no. 8, pp. 862–875, 2008.

[4] A. Carvalho, Y. Gao, A. Gray, H. E. Tseng, and F. Borrelli, "Predictive control of an autonomous ground vehicle using linearization approach," in *Proceedings of the 16th IEEE Annual Conference on Intelligent Transportation Systems*, The Hague, The Netherlands, October 2013.

[5] S. Lefevre, Y. Gao, D. Vasquez, E. Tseng, R. Bajcsy, and F. Borrelli, "Lane keeping assistance with learning-based driver model and model predictive control," in *Proceedings of the 12th international symposium on advanced vehicle control*, 2014.

[6] S. Lefevre, A. Carvalho, and F. Borrelli, "Autonomous car following: A learning-based approach," in *IEEE Intelligent Vehicles Symposium*, 2015, pp. 920–926.

[7] R. Vasudevan, V. Shia, Y. Gao, R. Cervera-Navarro, R. Bajcsy, and F. Borrelli, "Safe semi-autonomous control with enhanced driver modeling," in *Proceedings of American Control Conference (ACC)*, 2012, pp. 2896–2903.

[8] V. Shia, Y. Gao, R. Vasudevan, K. D. Campbell, T. Lin, F. Borrelli, and R. Bajcsy, "Semiautonomous vehicular control using driver modeling," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 6, pp. 2696–2709, 2014.

[9] D. Salvucci, E. Boer, and A. Liu., "Toward an integrated model of driver behavior in cognitive architecture," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1779, pp. 9–16, 2001.

[10] P. Hidas, "Modelling lane changing and merging in microscopic traffic simulation," *Transportation Research Part C: Emerging Technologies*, vol. 10, no. 5, pp. 351–371, 2002.

[11] R. A. Hess and A. Modjtahedzadeh, "A control theoretic model of driver steering behavior," *IEEE Control Systems Magazine*, vol. 10, no. 5, pp. 3–8, 1990.

[12] R. S. Sharp, D. Casanova, and P. Symonds, "A mathematical model for driver steering control, with design, tuning and performance results," *Vehicle System Dynamics*, vol. 33, no. 5, pp. 289–326, 2000.

[13] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical Review E*, vol. 62, no. 2, pp. 1805–1824, 2000.

[14] D. D. Salvucci and R. Gray, "A two-point visual control model of steering," *Perception*, vol. 33, no. 10, pp. 1233–1248, 2004.

[15] A. Y. Ungoren and H. Peng, "An adaptive lateral preview driver model," *Vehicle System Dynamics*, vol. 43, no. 4, pp. 245–259, 2005.

[16] P. Kumar, M. Perrollaz, S. Lefevre, and C. Laugier, "Learning-based approach for online lane change intention prediction," in *IEEE Intelligent Vehicles Symposium*, 2013, pp. 797–802.

[17] C. C. Macadam, "Understanding and modeling the human driver," *Vehicle System Dynamics*, vol. 40, no. 1-3, pp. 101–134, 2003.

[18] M. Plchl and J. Edelmann, "Driver models in automobile dynamics application," *Vehicle System Dynamics*, vol. 45, no. 7-8, pp. 699–741, 2007.

[19] R. Lee and D. Wolpert, *Chapter: Game theoretic modeling of pilot behavior during mid-air encounters*. in Decision making with multiple imperfect decision makers. Intelligent Systems Reference Library Series. Springer, 2011.

[20] J. H. Yoo and R. Langari, "Stackelberg game based model of highway driving," in *Proc. ASME Dynamic Systems and Control Conference joint with JSME Motion and Vibration Conference*, Fort Lauderdale, Florida, Oct. 2012.

[21] ——, "A stackelberg game theoretic driver model for merging," in *Proc. ASME Dynamic Systems and Control Conference*, Palo Alto, California, Oct. 2013.

[22] C. Dextreit and I. V. Kolmanovsky, "Game theory controller for hybrid electric vehicles," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 2, pp. 652–663, 2014.

[23] Y. Yildiz, A. Agogino, and G. Brat, "Predicting pilot behavior in medium-scale scenarios using game theory and reinforcement learning," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 4, pp. 1335–1343, 2014.

[24] D. W. Oyler, Y. Yildiz, A. R. Girard, N. I. Li, and I. V. Kolmanovsky, "A game theoretical model of traffic with multiple interacting drivers for use in autonomous vehicle development," in *Proceedings of American Control Conference (ACC)*. IEEE, 2016, pp. 1705–1710.

[25] D. Stahl and P. Wilson, "On players models of other players: Theory and experimental evidence," *Games and Economic Behavior*, vol. 10, no. 1, p. 218254, 1995.

[26] M. A. Costa-Gomes, V. P. Crawford, and N. Iriberri, "Comparing models of strategic thinking in Van Huyck, Battalio, and Beil's coordination games," *Journal of the European Economic Association*, vol. 7, no. 2-3, pp. 365–376, 2009.

[27] I. Miller, M. Campbell, D. Huttenlocher, F.-R. Kline, A. Nathan, S. Lupashin, J. Catlin, B. Schimpf, P. Moran, N. Zych, E. Garcia, M. Kurdziel, and H. Fujishima, "Team cornell's skynet: Robust perception and planning in an urban environment," *Journal of Field Robotics*, vol. 25, no. 8, pp. 493–527, 2008.

[28] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2722–2730.

[29] T. Jaakkola, P. S. Satinder, and I. Jordan., "Reinforcement learning algorithm for partially observable markov decision problems," *Advances in Neural Information Processing Systems 7: Proceedings of the 1994 Conference*, 1994.

[30] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. Cambridge: MIT press, 1998.

[31] L. Claussman, A. Carvalho, and G. Schildbach, "A path planner for autonomous driving on highway human mimcry approach with binary decision diagrams," in *Proceedings of the European Control Conference*, Linz, Austria, July 2015.