Game Theoretic Decision Making by Actively Learning Human Intentions

Siyu Dai, Sangjae Bae, and David Isele

***Abstract*— The ability to estimate human intentions and interact with human drivers intelligently is crucial for autonomous vehicles to successfully achieve their objectives. In this paper, we propose a game theoretic planning algorithm that models human opponents with an iterative reasoning framework and estimates human latent cognitive states through probabilistic inference and active learning. By modeling the interaction as a partially observable Markov decision process with adaptive state and action spaces, our algorithm is able to accomplish real-time lane changing tasks in a realistic driving simulator. We compare our algorithm's lane changing performance in dense traffic with a state-of-the-art autonomous lane changing algorithm to show the advantage of iterative reasoning and active learning in terms of avoiding overly conservative behaviors and achieving the driving objective successfully.**

I. INTRODUCTION

Interacting with human drivers safely and efficiently is a major challenge faced by autonomous vehicles (AVs). Human driving behaviors can be very diverse, and are influenced by many factors including the driver's personality and the traffic scenario. Obtaining an accurate estimate of the other agents' behavior is crucial to safe interactions in dense traffic. Learning-based approaches for modeling human driving behaviors [1]–[3] have achieved impressive results in recent literature, but inevitably face the demand for a large amount of training data and produce trained models that are nontrivial to interpret. Methods based on probabilistic inference [4]–[6] provide more interpretable models, but interactions often requires pre-defining the states of the models. Game theory allows us to capture the mutual influence between the traffic agents while modeling human-like interactions, and making fewer restrictive assumptions about state. However, existing work on game-theoretic AV planning often assume agents to be perfectly rational utility maximizers that operate under equilibrium strategies [7], [8], whereas human agents are known to deviate from equilibrium behaviors due to cognitive limitations [9].

In this paper, we propose an approach that combines game theory and probabilistic inference in order to reason about human drivers' behavior models, and achieve the AV's objective through successful interactions. In order to take human drivers' diverse driving styles and cognitive levels into account, we utilize the quantal level-k reasoning theory [10] and model human's distinct behaviors with *intelligence levels* and *rationality coefficients*. Intelligence levels describe how sophisticated an agent believes its opponents are when reacting to their actions, whereas rationality coefficients describe how close an agent is to a perfect maximizer. We model the interactions between an AV with human drivers as a partially observable Markov decision process (POMDP) and apply probabilistic inference to reason about their intelligence levels and degrees of rationality. Instead of passively adapting to humans' behaviors, our approach actively learns the humans' latent states in order to plan for actions that optimize the AV's own objective. Although similar ideas have previously been discussed in the context of abstract grid-based driving simulators [11], our proposed approach can adapt to diverse scenarios with multiple opponent agents and we demonstrate its performance in CARLA [12] (Fig. 1), a more realistic driving simulator. In addition, since our framework employs a standard POMDP formulation, it can be intuitively adapted to many other human-robot interaction applications beyond autonomous driving.

Fig. 1: A visualization of our algorithm running in the CARLA simulation environment. The ego car (Right) negotiates with other traffic to merge into a lane of dense on the left.

II. RELATED WORK

Game theory has been widely applied in autonomous driving for handling vehicle interactions. Stackelberg equilibrium [13] and Nash equilibrium [14] are commonly used for modeling traffic scenarios. However, when modeled as pure *followers* in Stackelberg games [8], [15]–[17], human drivers are assumed to be fully accommodating, rather than being treated as agents that may try to influence others. Additionally, human drivers are required to accurately predict AVs' behaviors, which is hardly realistic in practical scenarios. On the other hand, when adopting the Nash equilibrium solution [7], [18], [19], players are commonly assumed to have unlimited computational resources and respond optimally to the others, whereas humans are known to have cognitive limitations and don't act with perfect rationality. Williams et al. [20] proposed Iterative Best Response (IBR) and formulated vehicle interactions as differential games that

*Authors are with Honda Research Institute, USA can be solved approximately using optimization, but their approach requires knowledge of the other vehicles' dynamics, objectives, and current state. Zhan et al. [21] applies game theoretic techniques to actively gather information during the interaction, but focused on observing occluded opponents instead of predicting and reacting to opponents' behavior. Li et al. [22] and Tian et al. [11] applied hierarchical cognitive frameworks based on level-k reasoning [10] to explicitly model the cognitive limitations and irrationality of human agents, but limited the evaluation of their models to abstract grid-based driving simulators. In contrast, our work adapts game-theoretic formulations to realistic driving scenarios, and we demonstrate the real-time planning performance of our approach in lane-changing scenarios in dense traffic.

III. PRELIMINARIES

A. Quantal Level-k Reasoning

Quantal level-k reasoning [23] is a game theoretic modeling technique that combines the ideas of quantal response equilibrium (QRE) models [24] and level-k reasoning [10]. QRE models the imperfect maximization of expected payoffs using stochastic choices, hence it is a more realistic model for human players than the sharp best responses in standard game theory. Level-k reasoning theory additionally models how players form conjectures about their co-players' strategies and assumes players choose actions according to their belief. In the quantal level-k (ql-k) reasoning model, players are divided into k intelligence levels and they react to opponent actions by assuming the opponents are at a lower intelligent level than themselves. A ql-0 agent is assumed to be non-strategic, and a ql-k agent, $k \in \mathbb{N}^+$, believes that its opponent is a ql-$(k-1)$ agent and chooses actions according to the QRE model. The stochastic behaviors of each agent is modeled by a rationality coefficient $\lambda \in \mathbb{R}^+$. Let $Q^i(s, a^i|a^{-i})$ denote the expected total reward agent i achieves when executing action a^i at state s against an action a^{-i} from its opponent $-i$, then the policy of a ql-k agent with rationality λ can be expressed as:

$$\pi^i(a^i|s, a^{-i}) = \frac{\exp(\lambda Q^i(s, a^i|a^{-i}))}{\sum_{a' \in \mathcal{A}^i} \exp(\lambda Q^i(s, a'|a^{-i}))}, \tag{1}$$

where $\mathcal{A}^i$ denotes the action space of agent i.

B. Monte Carlo Tree Search (MCTS)

MCTS [25] is a heuristic search approach widely used in problems where the exact solutions are difficult or impossible to solve for. It breaks the curse of dimensionality by sampling state transitions instead of considering all possible state transitions when estimating the potential long-term reward. It converges to the optimal policy when exploration is controlled appropriately [26]. During rollouts, MCTS selects the next node to expand greedily according to the UCT algorithm [27] which augments the action reward by an exploration bonus. It continues to conduct rollouts until timeout and returns the current best action. Similar to Silver et al. [26], we apply MCTS to solve POMDPs, but we adapt it to work with multi-player games and incorporate an active learning component.

IV. PROBLEM STATEMENT

We model the decision making problem faced by a robotic agent when interacting with humans as a POMDP. $\mathbf{s} = [\mathbf{x}, \theta]$ represents a state in the POMDP, where $\mathbf{x}$ denotes the physical states (including $\mathbf{x}^e$ for ego agent and $\mathbf{x}^i$, $i = 1, 2, \ldots, n$ for n human agents) and θ denotes the internal states that dominate the opponents' behavior. We assume all physical states are fully observable and all opponents' internal states are hidden, and denote the observations as $\mathbf{o}$. We denote the action space of the ego agent as $\mathcal{A}^{\mathcal{E}}$, and assume all human agents have the same action space $\mathcal{A}^{\mathcal{H}}$. We assume the opponents hidden state θ doesn't change with t during the decision making process. The ego agent maintains a belief on the probability distribution over states $\mathbf{b}(\mathbf{s})$. Then the POMDP problem can be written as follows:

Problem 1.

$$\begin{aligned} \underset{\pi}{\text{maximize}} \quad & V(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{b}_t, \mathbf{a}_t^e) | \mathbf{a}_t^e \sim \pi\right] \\ \text{subject to} \quad & \mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{a}_t^e, \mathbf{a}_t^1, \ldots, \mathbf{a}_t^n) \\ & \mathbf{a}_t^e \in \mathcal{A}^{\mathcal{E}},\ \mathbf{a}_t^i \in \mathcal{A}^{\mathcal{H}}, \quad i = 1, 2, \ldots, n \\ & \mathbf{o} = \mathbf{x},\ \mathbf{s} = [\mathbf{x}, \theta] \\ & \mathbf{b}_{t+1}(\mathbf{s}_{t+1}) = \rho(\mathbf{b}_t(\mathbf{s}_t), \mathbf{o}_{t+1}), \end{aligned}$$

where r is the reward function, γ is the discount factor, f is a known dynamics function that governs the transition of physical states and ρ is the belief update function.

V. APPROACH

To solve Problem 1, we propose to model human opponents as ql-k agents and pre-compute their policies offline. We then let the hidden state $\theta = [k^i, \lambda^i]$, $i = 1, 2, \ldots, n$ and solve the POMDP online with MCTS by propagating beliefs according to the pre-computed policies.

A. Quantal Level-k Policies

In order to achieve real-time belief update during online planning, we pre-compute the Q value function for perfectly rational agents at each level k to store in memory, and then compute the quantal policies using Equation 1 according to the ego agent's belief on k^i and λ^i during online belief propagation. We assume the non-strategic level-0 policy is given, and a perfectly rational level-k agent maximizes the objective $V^{i,k}_{\pi^{i,k}}(\mathbf{s}_0) = \mathbb{E}_{\pi^{-i,k-1}}[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t^{i,k}) | \mathbf{a}_t^{i,k} \sim \pi^{i,k}]$. Note that we assume all humans follow the ql-k policies and only differ in terms of the internal states θ, hence during the value function pre-computation, we only consider a two-player game with one human and one ego agent. $V^{i,k}$ can be computed using value iteration [28] when the state space is discrete, and fitted value iteration [29] when the state space is continuous. With the value function $V^{i,k}$, we can then compute the Q function using the following equation:

$$Q^{i,k}(\mathbf{s}_t, \mathbf{a}_t^{i,k}) = \mathbb{E}_{\pi^{-i,k-1}}[r(\mathbf{s}_t, \mathbf{a}_t^{i,k}) + \gamma V^{i,k}(\mathbf{s}_{t+1})]. \tag{2}$$

In contrast to the quantal level-k dynamic programming approach proposed by Tian et al. [11] which directly computes the level-k policies during offline training, we only precompute the Q functions offline and compute ql-k policies during online planning. This allows us to save memory space because we don't need to store the level-k policies for every possible λ value. In addition, it also makes it more intuitive to extend our framework to incorporate the entire continuous set of λ values instead of pre-specifying a few discrete values.

B. Game Theoretic Planning with MCTS

During online planning, we construct a search tree which stores the current observation and belief with the root node as $root.\mathbf{x}$ and $root.\mathbf{b}$ respectively. In order to avoid the exponential increase of the tree expansion breadth as the number of opponents increases, we select up to two opponents during tree search according to their relative positions to the ego vehicle. We first identify the nearest two human agents, and if both the nearest agents are in front of or behind the ego agent, we only consider the nearest one as the opponent. Otherwise, both the nearest agents are considered as opponents. All other agents in the scene are considered as obstacles.

Algorithm 1 describes the game theoretic planning procedure. It relies on a tree structure that conducts rollouts to a depth that equals to the planning horizon H until timeout and returns the current best action based on the rewards of the rollouts. The tree consists of nodes that store the state $node.\mathbf{x}$, a list of children indexed by the ego action $\mathbf{a}^e$ and opponent actions $\mathbf{a}^o$, belief on opponents' hidden states $node.\mathbf{b}$, and the values of different ego actions $node.values$. During rollout, ego actions that have never been sampled ($node.unsampled\_as$) are prioritized, and if there are no more unsampled actions, an action with the highest UCT score [27] will be selected, which considers both its value and the number of times it has been sampled (line 18-21). Human agents that are not viewed as opponents are assumed to take the no-op action, i.e. maintaining constant velocity. Opponent agents' ql-k policies are computed using Equation 1 with pre-computed Q^k value functions, and their actions are sampled from the policy distribution weighted by the belief state $node.\mathbf{b}$. Each node keeps track of the total number of times it has been sampled ($node.N$) and the number of times each action has been sampled ($node.N_a$). The value of each action is computed by averaging the reward each rollout has achieved (line 33).

During node expansion, the state of the child node $child.\mathbf{x}$ and the reward r are estimated by the simulator, and the belief state of the child node $child.\mathbf{b}$ is updated following the standard POMDP belief update procedure based on the opponent actions $\mathbf{a}^o$ that led to the expansion of this child node (line 37-38 in Algorithm 1). In order to facilitate active learning of the human latent states, we augment the reward with an information gain reward that measures the reduction in the total entropy of the belief states on each opponent agent. We use an information gain reward coefficient ϕ to control the weight of this information gain reward during rollouts (line 39-40).

Algorithm 1: Game Theoretic Planning

```
Input:
$\mathcal{E}$ : environment
$n$: total number of other vehicles
$H$: planning horizon
$T$: time allowance for MCTS
$\gamma$: discount factor
$\phi$: information gain reward coefficient
$\mathbf{b}_0$: initial belief
$Q^k$: precomputed $Q$ function, $k = 1, \ldots, k_{max}$
1  $\mathbf{o} \leftarrow$ take initial observation
2  $oppo\_id \leftarrow$ SelectOpponent($\mathbf{o}$)
3  $Tree \leftarrow$ ConstructTree($\mathbf{o}, \mathbf{b}_0, oppo\_id$)
4  while not success do
5      $best\_action \leftarrow Tree$.search($H, T$)
6      Execute $best\_action$ and take observation $\mathbf{o}$ in $\mathcal{E}$
7      $oppo\_id \leftarrow$ SelectOpponent($\mathbf{o}$)
8      $Tree$.UpdateRoot($\mathbf{o}$)
9      $Tree$.UpdateOppo($oppo\_id$)
10
11 Function Tree.search($H, T$):
12     while $time\_elapsed < T$ do $Tree$.rollout($root, H$)
13     $best\_action \leftarrow$ argmax$_\mathbf{a}$ $root.values(\mathbf{a})$
14     return $best\_action$
15
16 Function Tree.rollout($node, H$):
17     if $node.depth = H$ then return 0
18     if len($node.unsampled\_as$) $> 0$ then
19         $\mathbf{a}^e \leftarrow$ random action from $node.unsampled\_as$
20     else
21         $\mathbf{a}^e \leftarrow$argmax$_\mathbf{a}$ $node.values(\mathbf{a}) + c\sqrt{\frac{\log node.N}{node.N_a(\mathbf{a})}}$
22     for $i$ in $n$ do
23         if $i$ in $oppo\_id$ then
24             $\pi^o \leftarrow$ ComputePolicy($node.\mathbf{b}[i], Q^k$)
25             Sample $\mathbf{a}^o[i] \sim \pi^o$
26         else
27             $\mathbf{a}^o[i] \leftarrow default\_action$
28     if $(\mathbf{a}^e, \mathbf{a}^o)$ not in $node.children$ then expand($\mathbf{a}^e, \mathbf{a}^o$)
29     $child \leftarrow node.children(\mathbf{a}^e, \mathbf{a}^o)$
30     $R = child.r + \gamma * Tree$.rollout($child, H$)
31     $node.N\ += 1$
32     $node.N_a(\mathbf{a}^e)\ += 1$
33     $node.values(\mathbf{a}^e)\ += \frac{R - node.values(\mathbf{a}^e)}{node.N_a(\mathbf{a}^e)}$
34     return $R$
35
36 Function node.expand($\mathbf{a}^e, \mathbf{a}^o$):
37     $child.\mathbf{x}, r \leftarrow$ Simulate($node.\mathbf{x}, \mathbf{a}^e, \mathbf{a}^o$)
38     $child.\mathbf{b} \leftarrow$ BeliefUpdate($node.\mathbf{x}, node.\mathbf{b}, child.\mathbf{x}$)
39     $info\_gain = \sum_{i \in oppo\_id}(\mathcal{H}(node.\mathbf{b}) - \mathcal{H}(child.\mathbf{b}))$
40     $child.r = r + \phi * info\_gain$
41     $node.children(\mathbf{a}^e, \mathbf{a}^o) \leftarrow child$
```

After a best action is returned by the search tree, ego agent executes it and observes the environment's physical state $\mathbf{o}$. When planning for the next action, we reevaluate which human agents should be considered as opponents, and prune sections of the tree that are no longer possible by reassigning the current observation as the new root node. This search procedure is repeated until the goal is achieved. An example illustration of our algorithm with planning horizon $H = 2$

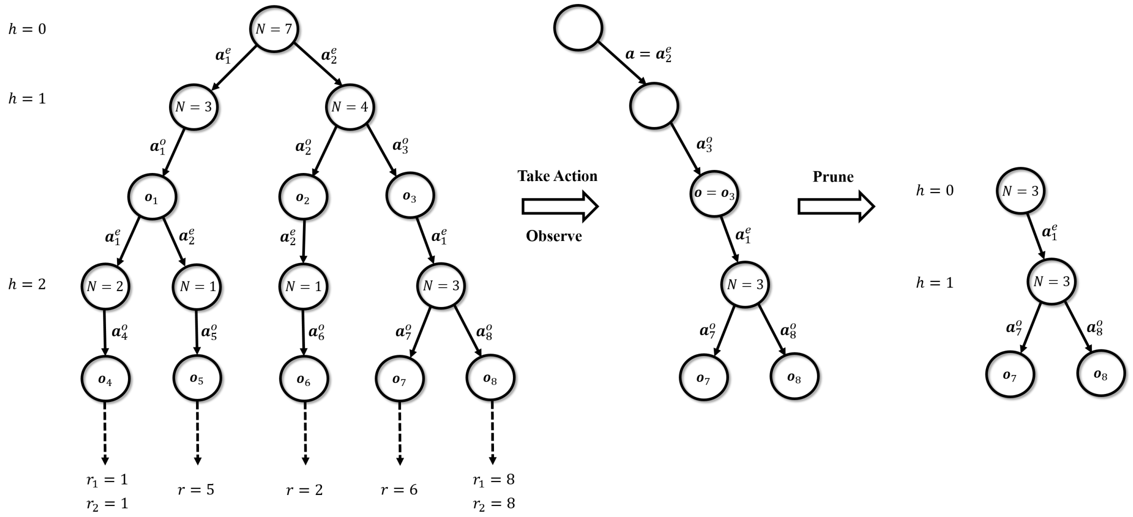

Fig. 2: An illustrative diagram of one search step of our algorithm with planning horizon $H = 2$ and two available ego vehicle actions. The leftmost graph shows the search tree after conducting 7 rollouts within the time allowance T (line 12 in Algorithm 1), where h denotes the depth, N denotes the total number of times a node has been expanded, and r denotes the total reward of a rollout trajectory. a_1^e and a_2^e represent the two available ego actions, and a_i^o includes the actions of all other vehicles. After the algorithm decides that a_2^e is the best action to take at the current time step, it executes a_2^e and receives a real observation from the environment. The algorithm identifies the real observation $\mathbf{o} = \mathbf{o}_3$, hence it prunes the section of the search tree that is no longer possible and reassigns the root node.

and two available ego actions is shown in Fig. 2.

Our game theoretic planning algorithm differs from the one proposed by Tian et al. [11] in three major aspects. First, instead of pre-computing and storing in memory the entire transition matrix of the POMDP, we takes advantage of the discretized state and action space and the known dynamics function to conduct online tree expansion and rollouts. The approach in [11] is constrained within abstract grid-world applications due to its high memory consumption and requirement of re-training the level-k policies and transition matrices when opponents have unseen rationality coefficient values. In contrast, our approach is adaptable to various scenarios through state-space mapping and can still maintain real-time planning performance through MCTS. Second, the ego vehicle in [11] is restricted to follow a ql-k policy and the values of tree nodes are estimated using pre-computed ql-k value functions, whereas our ego agent is able to plan for the best action according to the true value without being restricted by the ql-k policy. Additionally, the approach in [11] can only be applied in scenarios with one other agent, whereas our approach can be adapted to scenarios with an arbitrary number of human agents in the scene.

VI. EMPIRICAL EVALUATION

We evaluate the performance of our proposed approach on a lane changing task in the CARLA [12] simulator. All experiments are conducted on a 16-core Intel i9 2.4 GHz laptop with 32 GB RAM and one GeForce RTX 2060 GPU.

A. Experiment Setup

In order to achieve real-time planning performance, we use a small discretized state and action space instead of the real CARLA simulator for MCTS rollouts and pre-training ql-k policies. We assume $k_{max} = 2$ based on the findings on human intelligence levels in the literature [30], [31], and set the planning horizon depth $H = 4$ which corresponds to a $4T$-second look ahead. The agent receives a negative reward at every step where it's not on the target lane and a large negative reward for collision. The non-strategic level-0 policy is computed by assuming all other agents in the scene are static obstacles. The Q functions for ql-k policies are pre-trained by first obtaining the value function through value iteration assuming opponent agents are ql-$(k-1)$ agents, and then computing Q using Equation 2. The game theoretic planner will publish high-level target trajectories, and a low-level PID controller is used to track the target trajectory. To ensure adaptability in various scenarios, we use a 40 m long two-lane wide grid centered around the ego vehicle as the state space, and conduct real-time state mapping between the grid state space and the true physical state space during online planning. The size of the grid system also means that only vehicles that are on the ego lane and target lane with a distance of less than 20 m to the ego vehicle will be considered by the planner. The ego agent has five actions: "accelerate", "decelerate", "maintain velocity" (no-op), "indicate lane change intention" and "lane change". The "accelerate" action causes the ego vehicle to increase its velocity by 1 m/s and the "decelerate" action causes to decrease velocity by 1 m/s. The ego agent starts at rest, and traffic is initialized at 3 m/s. The "indicate lane change intention" action will trigger a target trajectory that moves the ego vehicle 0.5 m towards the target lane while maintaining the same longitudinal speed, whereas the "lane change" action will publish a target trajectory that brings the ego vehicle to the center of the target lane. To simplify the scenario, we encode only three actions for the human agents: accelerate, decelerate and maintain velocity (no-op).

The scenarios we tested on are in the `Town03` environment in CARLA with an illegal bicycle and a stopped vehicle

TABLE I: Evaluation on Belief Accuracy

# Opponents	T (s)	With Info Gain	W/O Info Gain
One	0.5	87.5%	81%
	1	85%	80.5%
	2	85.5%	80.5%
Two	0.5	82.5%	79.8%
	1	83.9%	79.3%
	2	83.8%	80.9%

100 m ahead of the ego vehicle's starting pose, forcing a lane change. We evaluate our planner in three non-cooperative game scenarios with one, two and six driving vehicles in the target lane respectively, where the driving vehicles follow the Intelligent Driver Model (IDM) [32]. The task goal is to reach within 0.5 m in the lateral direction from the center of the target lane before reaching the stopped vehicle in the ego lane. The driving vehicles are placed in the scene to simulate human drivers, and their controller tracks a randomly generated velocity while maintaining a distance of $5-10$ m with the front vehicle. We program the other vehicles to not follow ql-k policies in order to show the generalization behavior of our planner to unseen behaviors. We simulate the behaviors of aggressive drivers and yielding drivers by allowing these driving vehicles to yield when the ego vehicle is close in half of the tests for each set of experiments. We compare our planner's lane changing performance with NNMPC [33], a state-of-the-art autonomous lane changing planner that uses Recurrent Neural Networks (RNNs) for predicting other vehicles' behaviors and guiding the Model Predictive Controller (MPC). We also conduct an ablation study by comparing our approach with a version without the information gain reward (Algorithm 1 line 40) in order to show the effect of active learning.

B. Evaluation on Belief Accuracy

We first evaluate whether our game theoretic planner can accurately estimate the intelligence level of opponent agents. Since up to two agents can be considered as opponents at a time, we experiment with two scenarios: one opponent in the scene and two opponents in the scene. We compare the performance using time allowances $T = 0.5$, $T = 1$ and $T = 2$, and conduct an ablation study by comparing our full model against a model without the information gain. In each scenario, we let the opponent follow a ql-k policy for all $k = 1, \ldots, k_{max}$ (or all combinations of k's when there are two opponents) and run 100 simulations with rationality coefficients λ randomly sampled from [1, 3, 5] for each k. We view a belief state as accurate if it assigns a larger than 0.5 probability for the correct intelligence level. We average the accuracy for all k combinations in each set of experiments and show the results in Table I. We see that with information gain reward the average performance is better. This shows that active learning improves the ego agent's estimation accuracy on the opponents' latent states. Even though accurately estimating belief is more challenging when there are two opponents in the scene, our approach is still able to achieve an accuracy of above 80%.

TABLE II: Evaluation on Lane Changing Performance

Scenario	Metric[1]	Ours with Info Gain	Ours w/o Info Gain	NNMPC
1 other vehicle	time to merge (s)	5.52	7.02	20.08
	collision rate	2%	6%	0%
	timeout rate	0%	4%	0%
2 other vehicles	time to merge (s)	7.5	8.96	24.69
	collision rate	4%	12%	2%
	timeout rate	2%	8%	0%
6 other vehicles	time to merge (s)	10.2	12.6	31.5
	collision rate	8%	12%	4%
	timeout rate	6%	10%	16%

[1] Time to merge refers to the time it takes the ego vehicle to reach within 0.5 m from the center of the target lane, and timeout includes failure to merge when reached within 10 m from the stopped vehicle or having stopped for more than 15 s in the ego lane.

C. Evaluation on Lane Changing Performance

Table II compares the autonomous lane changing performance of our game theoretic planner, its version without information gain reward, and NNMPC, where each entry is the average of 50 test runs in the CARLA simulator. In the experiments in this section, the time allowance in our algorithm is set to $T = 1$ s. From the results in Table II and the qualitative performance in the simulation videos we observed that NNMPC is a very conservative planner that will sacrifice the time to merge to obtain low collision rate. It takes more than three times as much time to change lane compared to our approach in all the scenarios we tested, and we often observe it stopping to wait for all vehicles to pass before merging in, which causes it to have a higher timeout rate in dense traffic (i.e. scenarios with 6 other vehicles). This is because even though NNMPC is able to predict other agents' paths and plan for a safe trajectory to merge, it lacks the ability to actively interact with the other agents and signal a lane-changing intention. On the contrary, our approach leverages game theory to generate interactive behaviors that help the ego agent achieve its goal as soon as possible. For example, we observe that when our planner meets aggressive opponents, it will slow down to let the opponent go first and then merge in after, whereas NNMPC in the same situation will continue going forward with the same speed and reaches a dead-lock situation with the opponent vehicle. Similarly, when our planner signaled the intention to merge and found that the opponent vehicle is yielding, it then accelerated to a safer location to accomplish the lane change. NNMPC never makes the attempt to change lane as long as the opponent vehicles are close and therefore is not always able to take advantage of potentially yielding opponents.

Another finding from Table II is that the time to merge and failure rate are both higher when our approach is used without the information gain reward, which is likely caused by the less accurate estimation of opponent intelligence level. In general, our approach has a relatively higher collision rate compared to NNMPC. This issue can potentially be solved by adding chance constraints [34], [35] to our planner or by using a collision-aware low-level controller to execute the high-level target trajectories.

VII. CONCLUSIONS

In this paper, we propose a game theoretic autonomous driving algorithm that can actively reason about human drivers' latent cognitive states in real time and achieve its driving objective through successful interactions with other vehicles. By modeling human drivers as level-k reasoning agents and modeling the interaction as a POMDP, our algorithm is able to estimate humans' latent cognitive states through probabilistic inference. By augmenting the reward function with an information gain reward, our agent can actively take actions to encourage the accurate estimation of latent states instead of passively reacting to human behaviors. We apply adaptive state and action spaces in the POMDP model and utilize MCTS to achieve real-time planning performance in lane changing scenarios in the CARLA driving simulator. Experiment results comparing our algorithm with another state-of-the-art autonomous lane changing algorithm show that our approach is able to avoid overly conservative behaviors and succeed more frequently in mandatory lane changing scenarios in dense traffic.

REFERENCES

[1] R. P. Bhattacharyya, D. J. Phillips, C. Liu, J. K. Gupta, K. Driggs-Campbell, and M. J. Kochenderfer, "Simulating emergent properties of human driving behavior using multi-agent reward augmented imitation learning," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 789–795.

[2] M. Bouton, A. Nakhaei, D. Isele, K. Fujimura, and M. J. Kochenderfer, "Reinforcement learning with iterative reasoning for merging in dense traffic," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–6.

[3] M. Naumann, L. Sun, W. Zhan, and M. Tomizuka, "Analyzing the suitability of cost functions for explaining and imitating human driving behavior based on inverse reinforcement learning," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 5481–5487.

[4] S. Sekizawa, S. Inagaki, T. Suzuki, S. Hayakawa, N. Tsuchida, T. Tsuda, and H. Fujinami, "Modeling and recognition of driving behavior based on stochastic switched arx model," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 4, pp. 593–606, 2007.

[5] G. Agamennoni, J. I. Nieto, and E. M. Nebot, "A bayesian approach for driving behavior inference," in *2011 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2011, pp. 595–600.

[6] D. Isele, "Interactive decision making for autonomous vehicles in dense traffic," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 3981–3986.

[7] A. Dreves and M. Gerdts, "A generalized nash equilibrium approach for optimal control problems of autonomous cars," *Optimal Control Applications and Methods*, vol. 39, no. 1, pp. 326–342, 2018.

[8] J. Yoo and R. Langari, "A game-theoretic model of human driving and application to discretionary lane-changes," *arXiv preprint arXiv:2003.09783*, 2020.

[9] J. R. Wright and K. Leyton-Brown, "Level-0 meta-models for predicting human behavior in games," in *Proceedings of the fifteenth ACM conference on Economics and computation*, 2014, pp. 857–874.

[10] D. O. Stahl II and P. W. Wilson, "Experimental evidence on players' models of other players," *Journal of economic behavior & organization*, vol. 25, no. 3, pp. 309–327, 1994.

[11] R. Tian, L. Sun, M. Tomizuka, and D. Isele, "Anytime game-theoretic planning with active reasoning about human's latent states for human-centered robots," in *2021 International Conference on Robotics and Automation (ICRA)*. IEEE, 2021.

[12] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

[13] H. Von Stackelberg, *Market structure and equilibrium*. Springer Science & Business Media, 2010.

[14] X. Vives, "Nash equilibrium with strategic complementarities," *Journal of Mathematical Economics*, vol. 19, no. 3, pp. 305–321, 1990.

[15] J. F. Fisac, E. Bronstein, E. Stefansson, D. Sadigh, S. S. Sastry, and A. D. Dragan, "Hierarchical game-theoretic planning for autonomous vehicles," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 9590–9596.

[16] Q. Zhang, R. Langari, H. E. Tseng, D. Filev, S. Szwabowski, and S. Coskun, "A game theoretic model predictive controller with aggressiveness estimation for mandatory lane change," *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 1, pp. 75–89, 2019.

[17] P. Hang, C. Lv, Y. Xing, C. Huang, and Z. Hu, "Human-like decision making for autonomous driving: A noncooperative game theoretic approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 4, pp. 2076–2087, 2020.

[18] A. Liniger and J. Lygeros, "A noncooperative game approach to autonomous racing," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 3, pp. 884–897, 2019.

[19] M. Wang, Z. Wang, J. Talbot, J. C. Gerdes, and M. Schwager, "Game-theoretic planning for self-driving cars in multivehicle competitive scenarios," *IEEE Transactions on Robotics*, 2021.

[20] G. Williams, B. Goldfain, P. Drews, J. M. Rehg, and E. A. Theodorou, "Best response model predictive control for agile interactions between autonomous ground vehicles," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 2403–2410.

[21] Z. Zhang and J. F. Fisac, "Safe Occlusion-Aware Autonomous Driving via Game-Theoretic Active Perception," in *Proceedings of Robotics: Science and Systems*, Virtual, July 2021.

[22] S. Li, N. Li, A. Girard, and I. Kolmanovsky, "Decision making in dynamic and interactive environments based on cognitive hierarchy theory, bayesian inference, and predictive control," in *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE, 2019, pp. 2181–2187.

[23] Y. Breitmoser, J. H. Tan, and D. J. Zizzo, "On the beliefs off the path: Equilibrium refinement due to quantal response and level-k," *Games and Economic Behavior*, vol. 86, pp. 102–125, 2014.

[24] R. D. McKelvey and T. R. Palfrey, "Quantal response equilibria for normal form games," *Games and economic behavior*, vol. 10, no. 1, pp. 6–38, 1995.

[25] G. Chaslot, S. Bakkes, I. Szita, and P. Spronck, "Monte-carlo tree search: A new framework for game ai." *AIIDE*, vol. 8, pp. 216–217, 2008.

[26] D. Silver and J. Veness, "Monte-carlo planning in large pomdps," *Advances in Neural Information Processing Systems*, vol. 23, pp. 2164–2172, 2010.

[27] L. Kocsis and C. Szepesvári, "Bandit based monte-carlo planning," in *European conference on machine learning*. Springer, 2006, pp. 282–293.

[28] R. Bellman, "A markovian decision process," *Journal of mathematics and mechanics*, vol. 6, no. 5, pp. 679–684, 1957.

[29] M. Lutter, S. Mannor, J. Peters, D. Fox, and A. Garg, "Value iteration in continuous actions, states and time," in *Proceedings of the 38th International Conference on Machine Learning*, 2021, pp. 7224–7234.

[30] M. A. Costa-Gomes and V. P. Crawford, "Cognition and behavior in two-person guessing games: An experimental study," *American economic review*, vol. 96, no. 5, pp. 1737–1768, 2006.

[31] 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.

[32] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical review E*, vol. 62, no. 2, p. 1805, 2000.

[33] S. Bae, D. Saxena, A. Nakhaei, C. Choi, K. Fujimura, and S. Moura, "Cooperation-aware lane change maneuver in dense traffic based on model predictive control with recurrent neural network," in *2020 American Control Conference (ACC)*. IEEE, 2020, pp. 1209–1216.

[34] M. Ono and B. C. Williams, "Iterative risk allocation: A new approach to robust model predictive control with a joint chance constraint," in *2008 47th IEEE Conference on Decision and Control*. IEEE, 2008, pp. 3427–3432.

[35] S. Dai, S. Schaffert, A. Jasour, A. Hofmann, and B. Williams, "Chance constrained motion planning for high-dimensional robots," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8805–8811.