Anytime Game-Theoretic Planning with Active Reasoning About Humans’ Latent States for Human-Centered Robots

Ran Tian, Liting Sun, Masayoshi Tomizuka, and David Isele

Abstract—A human-centered robot needs to reason about the cognitive limitation and potential irrationality of its human partner to achieve seamless interactions. This paper proposes an anytime game-theoretic planner that integrates iterative reasoning models, a partially observable Markov decision process, and chance-constrained Monte-Carlo belief tree search for robot behavioral planning. Our planner enables a robot to safely and actively reason about its human partner’s latent cognitive states (bounded intelligence and irrationality) in real-time to maximize its utility better. We validate our approach in an autonomous driving domain where our behavioral planner and a low-level motion controller hierarchically control an autonomous car to negotiate traffic merges. Simulations and user studies are conducted to show our planner’s effectiveness.

I. INTRODUCTION

Human-centered robots (e.g., self-driving cars, assistive robots, etc.) operate in close proximity to humans. When designing planning algorithms for human-centered robots, it is critical for the robot to reason about the mutual influence between itself and human actors. Such a mutual dependency can be formulated as a general-sum game, in which a standard approach is to assume that each agent is a perfectly rational, expected utility maximizer, who simultaneously responds optimally to all the others (i.e., operates under equilibrium strategies) [1], [2]. However, experimental studies [3]–[5] suggest that human behaviors often systemically deviate from equilibrium behaviors due to their latent cognitive states: bounded intelligence (cognitive limitation) and irrationality (tendency to make errors). Therefore, a robot must account for its human partner’s cognitive states for seamless and safe interactions.

Recent works exploited the leader-follower model [6]–[9] and the level-k model [10]–[12] to equip robots with the ability to reason about humans’ non-equilibrium behaviors. These planners either assign humans’ latent states a priori, omitting humans’ distinct cognitive characteristics, or passively adapt to the humans’ latent states, sacrificing the benefits from actively learning the latent states (Sec. [M]).

In this work, we propose an anytime game-theoretic planning framework that integrates iterative reasoning models, a partially observable Markov decision process (POMDP), and chance-constrained Monte-Carlo belief tree search. Drawing inspiration from behavioral game theory, we model humans’ intelligence levels and degrees of rationality as their latent cognitive states, capturing their heterogeneous cognitive limitations and tendencies to make errors. Rather than passively adapting to humans’ latent states when planning, our approach enables the robot to actively and safely learn the latent states to achieve its goal more effectively without losing the ability of real-time execution. Our key insight is: Human-centered robots can exploit the mutual influence in interactions to design actions that reveal their human partners’ cognitive limitations and degrees of rationality. By actively reasoning about these latent states, robots can achieve more effective planning. Overall, we make the following contributions:

An active anytime game-theoretic planner. We formalize the robot planning problem in a human-robot team as a POMDP with the human’s cognitive states as latent states. The POMDP is approximately solved by an open-loop Monte-Carlo belief tree search algorithm in an anytime manner. Coupled with explicit realization of active information gathering on the latent states, chance-constrained branching, and tailored terminal value functions, our planner enables the robot to safely and adaptively balance between exploration (learning the human’s latent states) and exploitation (maximizing utility).

Application of our framework to autonomous driving. The proposed behavioral planner is connected with an off-the-shelf low-level motion control layer [13] to achieve feedback control for an autonomous car represented by a high-fidelity model. Simulations and user studies are conducted to show the effectiveness of our planner compared to baselines.

II. RELATED WORK

Game-theoretic planning for human-centered robots. Our work is related to [6]–[12]. In [7]–[9], the robot exploits the Stackelberg game [14] and models its human partner as a follower who accommodates the robot’s planned actions. The follower model is homogeneous (the human is always the follower w.r.t. the robot), thus the robot may behave poorly when the human does not behave like a follower. The approach in [6] allows the robot to actively “probe” the human’s latent states, but the underlying human model is still a pure follower model. In addition, the safety of the planner is not explicitly enforced in [6]. In [10]–[12], the robot exploits the level-k model [15] to model the human as an agent who reasons under various intelligence levels. While the level-k model is heterogeneous, it assumes that humans with lower-levels of intelligence best respond to higher-level humans, omitting humans’ potential irrationality. Furthermore, the planners in [10]–[12] passively adapt to the human’s intelligence level, leading to less effective plans. Our approach is different from [6]–[9] since our planner

Ran Tian, Liting Sun, and Masayoshi Tomizuka are with the UC Berkeley, \{rantian, liting, tomizuka\}@berkeley.edu. David Isele is with Honda Research Institute. (disele@honda-ri.com).
does not assume humans’ cognitive characteristics in interactions a priori but reasons about them when planning. Our work is also distinguished from [10]–[12] by enabling the robot to safely and actively learn the human’s cognitive states to maximize utility more effectively without losing the ability of real-time execution.

Solution methods for POMDP. A POMDP is a framework for planning under uncertainty. Various approaches have been proposed to approximate POMDPs, including point-based methods [16]–[18], open-loop strategies [19]–[22], and Monte-Carlo tree search [23], [24]. Partially observable Monte-Carlo planning (POMCP) [23] performs well, though building a closed-loop search tree in large games is computationally expensive. Open-loop strategies condition action selection on previous action sequences. They use a much smaller search space by sacrificing the ability of active information gathering, and achieve competitive performance though building a closed-loop search tree in large games is computationally expensive. Open-loop strategies condition action selection on previous action sequences. They use a much smaller search space by sacrificing the ability of active information gathering, and achieve competitive performance compared to closed-loop planning under computational constraints [20]. Our approach combines the strengths from POMCP and open-loop strategies, achieving real-time active game-theoretic planning.

III. PROBLEM FORMULATION

Human-robot team formalization. We formalize the human-robot team as a two-player dynamic game represented by the tuple $\mathcal{G} = < \mathcal{P}, \mathcal{S}, \mathcal{A}, f, r_{\mathcal{R}}, r_{\mathcal{H}} >$, where $\mathcal{P} = \{ \mathcal{R}, \mathcal{H} \}$ represents the two players with $\mathcal{R}$ denoting the robot and $\mathcal{H}$ denoting the human; $\mathcal{S} = S_{\mathcal{R}} \times S_{\mathcal{H}}$ and $\mathcal{A} = A_{\mathcal{R}} \times A_{\mathcal{H}}$ are, respectively, the joint fully-observable state and action spaces of the two agents; the function $f$ governs the evolution of the joint fully-observable state and is defined by the following dynamic model: $s_{t+1} = f(s_t, a^R_t, a^H_t)$ ($s_t \in S, a^R_t \in A_{\mathcal{R}}, a^H_t \in A_{\mathcal{H}}$); $r(\cdot) : \mathcal{S} \rightarrow \mathbb{R}$ is the reward function of an agent.

Robot planning as a POMDP. We consider planning from the robot’s perspective. The fully-observable state $s$ of the human-robot team represents measurable variables (e.g., position, speed, etc.). In addition, the human also has latent states (e.g., preference, trust, cognitive limitation, etc.) that characterize his/her cognition and reasoning; such latent states cannot be observed directly, and therefore must be inferred from interactions. We let $\Theta$ denote the human’s latent states, and we consider an augmented state space $S = S \times \Theta$. As the robot’s knowledge about the augmented state $s \in S$ is incomplete, it maintains a belief distribution over $S$ at each discrete time step $t$, namely, the robot maintains the belief state $b_t := [P(s_t = s_1), \ldots, P(s_t = s_N)]^{T}$. We formulate the robot planning problem as a POMDP defined by the tuple $< \mathcal{G}, \mathcal{B}, \Omega, Z_{\mathcal{R}}, r_{\mathcal{R}}, \mathcal{O}_{\mathcal{safe}} >$, where $\mathcal{G}$ denotes the dynamic game model defined above; $\mathcal{S}$ is the augmented state space; $\mathcal{B}$ represents the space of probability distributions over $S$ ($b_t \in \mathcal{B}$); $\Omega$ is the finite observation space; $Z : \Omega \times S \rightarrow [0, 1]$ is a probability function specifying the probability of receiving an observation in a state; the belief dynamics function $\rho : \mathcal{B} \times A_{\mathcal{R}} \times \Omega \rightarrow \mathcal{B}$ governs the belief state evolution and is defined as $b_{t+1} = \rho(b_t, a^R_t, o_{t+1})$. Given an initial belief state $b_t$, the robot executes the action $a^R_t$, receives the observation $o_{t+1}$ at time step $t + 1$, and updates its belief accordingly. $r^R_t : \mathcal{B} \times A_{\mathcal{R}} \rightarrow \mathbb{R}$ denotes the reward function of the robot in belief space (defined in Sec. V-A); $\mathcal{O}_{\mathcal{safe}} \subseteq S$ represents the set of safe states of the robot.

We let $\pi_{\mathcal{R}} : \mathcal{B} \rightarrow A_{\mathcal{R}}$ denote a robot’s deterministic policy. Given belief state $b_t$, the robot maximizes its value:

$$\pi^*_R = \arg \max b_t V^R(b_t),$$

(1)

where $V^R(b_t) = \mathbb{E}_z \left[ \sum_{t=0}^{\infty} \gamma^t r^R_t(b_{t+1}, a_{t+1}) | o_{t+1} = \pi(b_{t+1}) \right]$ is the value function representing the robot’s expected return starting from $b_t$, subject to its policy and the belief dynamics function. Note the robot needs to reason about both its own actions and its human partner’s responses due to the mutual dependence. The POMDP formulation allows us to condition the robot behaviors on the inferred latent states.

IV. HUMAN LATENT STATES MODELING

A. Iterative Reasoning Model

When studying interactions in games, players are commonly assumed to adopt the Nash equilibrium solution: each player has unlimited computational resources and responds optimally to the others. In real life however, humans are known to act irrationally and have cognitive limitations. The iterative reasoning models from behavioral game theory have been proven to show better performance in characterizing humans’ reasoning capabilities in simultaneous games [5]. Examples of iterative models include: the level-$k$ model [15], the cognitive hierarchy model [25], and the quantal level-$k$ model [26]. All these models aim to capture humans’ cognitive limitations and share a common feature: they model humans as agents with heterogeneous bounds on their reasoning abilities, i.e., human agents can only perform a finite number of iterations of reasoning, and such an intelligence bound is referred as the intelligence level.

Among the various iterative reasoning models, the quantal level-$k$ model is the state-of-the-art [27].

B. Human Quantal Level-$k$ Reasoning Model

Quantal best response and rationality coefficient. One of the key components of the quantal level-$k$ (ql-$k$) model is quantal best response. The notion behind quantal best response is that human players are more likely to select actions with higher expected future rewards [28]. Formally, we define the quantal best response function as follows: let $Q^i(\tilde{s}, a^i | a^{-i})$ denote agent $i$’s expected total reward ($i \in \mathcal{P}$) when executing $a^i$ in $\tilde{s}$ against an action $a^{-i}$ from his/her opponent $-i$. Then a quantal best response by agent $i$ to agent $-i$ is a mixed policy:

$$\pi^i(\tilde{s}, a^i | a^{-i}) = \frac{\exp(\lambda^i Q^i(\tilde{s}, a^i | a^{-i}))}{\sum_{a^i' \in A_i} \exp(\lambda^i Q^i(\tilde{s}, a^i' | a^{-i}))},$$

(2)

where $\lambda^i \in (0, 1]$ is the rationality coefficient that controls the degree of agent $i$ conforming to optimal behaviors. In general, the larger the $\lambda$ is, the more rational the human is.

Human quantal level-$k$ policies. In the ql-$k$ model, the iterative reasoning process starts from ql-0 agents who are non-strategic reasoners. Then, a ql-$k$ agent, $k \in \mathbb{N}^+$, assumes the other agents are ql-$(k-1)$ agents, predicts their ql-$(k-1)$ policies, and quantally best responds to the predicted ql-$(k-1)$ policies. On the basis of ql-0 policies, the ql-$k$ policies
are defined for every \( i \in \mathcal{P} \), for every \( \lambda \in \Lambda \), and for every \( k = 1, \ldots, k_{\text{max}} \) through a sequential and iterative process. Specifically, given an initial state \( \hat{s}_t \in \mathcal{S} \), a qI-k agent \( i \) maximizes the following objective: max \( \pi_{\sigma,\theta,k,\lambda} V^{i,k}(\hat{s}_t) \), where \( V^{i,k}(\hat{s}_t) = \mathbb{E}_{s \sim \mathcal{S}} V^{i,k}(\hat{s}_t) \), is the qI-k value function of agent \( i \) and \( \pi_{\sigma,\theta,k,\lambda} : \hat{s} \times \mathcal{A} \rightarrow [0,1] \) is the probability of the latent states (\( \sigma \) denotes the model parameters), and \( \pi_{\mathcal{H},k,\lambda} \) denotes the human’s qI-k policy with rationality coefficient \( \lambda \) (recall \( \theta = (k, \lambda) \)). Then, we can define a prior belief state prediction function that predicts the future belief state without accounting for the possible observations: \( \hat{b}_{t+1} = \tilde{\rho}(b_t, a_t) \), where each element in \( \hat{b}_{t+1} \) is computed following \( \hat{b}_{t+1} = \sum_{s \in \mathcal{S}} T(s, a, b_t) \hat{b}_t(s) \).

Prior probability of future observations. With the observation function and the prior belief prediction function defined, given an initial belief state \( b_t \), we can predict the probabilities of the robot’s future observations after executing an action \( a \in \mathcal{A}_R \) following: \( \mathcal{O}(o, b, a) = \mathbb{P}(a_{t+1} = a | b_t = b, a_t^R = a) = \sum_{s' \in \mathcal{S}} \mathcal{O}(a, s') \hat{b}_{t+1}(s') \).

Summary. The human’s behavioral model is embedded into the belief transition. With \( \tilde{\rho} \), the robot infers the human’s latent states, and in turn uses the inference result to predict belief evolution via the belief prediction function \( \tilde{\rho} \).

B. Robot Planning Algorithm

Algorithm 1: Quantal level-k dynamic programming

1. **Input:** The highest intelligence level \( k_{\text{max}} \), a set of rationality coefficients \( \Lambda \), and the level-0 model \( \pi_{\theta,0} \), \( i \in \mathcal{P} \).
2. for \( k = 1 : k_{\text{max}} \) do
3.   for \( (i, \lambda) \in \mathcal{P} \times \Lambda \) do
4.     while \( V^{i,k} \) not converged do
5.       for \( \hat{s} \in \mathcal{S} \) do \( V^{i,k}(\hat{s}) \leftarrow BV^{i,k}(\hat{s}) \);
6.         for \( (\hat{s}, a^i) \in \hat{s} \times \mathcal{A} \) do
7.             Compute \( \pi^{i,k}(\hat{s}, a^i) = \frac{\exp(\lambda \pi_{\sigma,\theta,k,\lambda}(\hat{s}, a^i))}{\sum_{a^i \in \mathcal{A}} \exp(\lambda \pi_{\sigma,\theta,k,\lambda}(\hat{s}, a^i))} \) using (3).
8.             Compute \( \pi_{\sigma,\theta,k,\lambda} : \hat{s} \times \mathcal{A} \rightarrow [0,1] \) for \( \hat{s} \in \mathcal{S} \) and \( \mathcal{A} \).
9.     end do
10.   end do
11. end do
12. end do

V. ANYTIME ACTIVE GAME-THEORETIC PLANNING

In this section, we embed the human qI-k model in the POMDP through belief dynamics following the procedure in [12, 29] and present our anytime game-theoretic planner.

A. Embed the Human Behavioral Model in Robot Planning Observation function. We define an observation made by the robot as \( o := \hat{s} \), i.e., the robot can measure the joint physical state \( \hat{s} \). Then the observation function is defined as: \( \mathcal{Z}(o' = \hat{s}) \), where \( \hat{s} \) is the joint physical state in \( s \), and \( \mathcal{I}(\cdot) \) is an indicator function, taking 1 if the event \( \cdot \) is true and taking 0 otherwise.

Prior belief state. We define the probability of arriving to state \( s' = (s', \theta') \in \mathcal{S} \) from state \( s = (\hat{s}, \theta) \in \mathcal{S} \) after executing \( a \in \mathcal{A}_R \) as:

\[
\mathcal{T}(s, a, s') := \mathbb{P}(s_{t+1} = s' | s_t = s, a_t^R = a) = \sum_{a' \in \mathcal{A}_N} \mathcal{P}(\theta_{t+1} = \theta' | \theta_t = \theta, s_t = s, \theta) \pi_{\mathcal{H},k,\lambda}(\hat{s}, a'),
\]

where \( f \) is the dynamics function described in Sec. III. \( \mathcal{P}(\theta_{t+1} | \theta_t, \hat{s}_t, \hat{\sigma}) \) represents an explicit probabilistic model that governs the dynamics of the latent states (\( \hat{\sigma} \) denotes the
where $H(b) = -\sum_{s \in S} b(s) \log b(s)$, and $O$ is the observation prediction function defined in \cite{5}. In general, the higher the $\mathcal{I}(b, a)$ is, the more expected information the robot obtains about the human’s latent states if the robot executes $a$ in $b$. With the augmented reward function $\hat{\mathcal{r}}_R(b, a)$, (7) becomes:

$$a^*_R = \arg \max_a E_z \left[ \hat{V}(b_{t+1}) + \sum_{\tau = 0}^{T} \hat{\mathcal{r}}_R(b_{t+1}, a_{t+\tau}) \right].$$

(9a)

$$\hat{\mathcal{r}}_R(b, a) = \hat{r}_R(b, a) + \eta \mathcal{L}(b, a),$$

(9b)

subject to $\mathbb{P}(\hat{s}_{t+1} \in \mathcal{O}_{\text{safe}} | b_t) \geq 1 - \Delta, \sum_{\tau = 1}^{T} \Delta\tau \leq \Delta,$

(9c)

where $\eta \propto H(b)$ is an adaptive term that enables the robot to learn the latent states as needed. The constraint (9c) requires that the probability that the predicted state $\hat{s}_{t+1}$ is in the safe set $\mathcal{O}_{\text{safe}}$ is larger than $1 - \Delta$ for all steps over the planning horizon, with $\Delta$ being a design parameter. The overall safety is bounded by $\Delta$ via risk allocation \cite{35}, \cite{36} (evaluation of risk is shown in Alg. 2). \cite{12} exploited continuous relaxation techniques to solve (9) with a time-joint chance constraint. However, with the information reward \cite{95}, the approach in \cite{12} could become intractable.

Algorithm 2: Open-Loop Monte-Carlo Belief Tree Search

1. **Input**: POMDP model, planning horizon $T$, discount factor $\gamma$, exploration parameter $\epsilon$, and initial belief state $b_0$.
2. **while** goal not reached **do**
   3. Initialize root node $v_{\text{root}} = (0, 0, 0)$.
   4. **while** not TimeOut() **do**
   5. Simulate($v_{\text{root}}, b_t, 0$); $N(v_{\text{root}}, a) + 1 = 1$;
   6. $a^*_R = \arg \max_a \mathcal{V}(a)$;
   7. Execute $a^*_R$ and collect the new observation $o_{t+1}$;
   8. Update belief state: $b_t \leftarrow \rho(b_t, a_t, o_{t+1})$;
   9. **end while**
10. **end while**
11. **Function** Simulate($v, b, \tau$)
12. **if** $\tau = T - 1$ **then**
13. return $\sum_{k \in K} \mathbb{P}(k^H = b | \sum_{s \in S} \mathbb{P}(\hat{s} | b) V^*, R, k+1(\hat{s}))$
14. **else**
15. for all $a \in A$ **do**
16. Append $v_{\text{succ}} = \langle v.a.\text{add}(a), 0, 0 \rangle$ in the tree;
17. return Rollout($b_t$, $\tau$);
18. **end if**
19. $v_{\text{succ}}^* = \arg \max_a \mathcal{V}(v_{\text{succ}}, a) + \epsilon \frac{\log(N(v_{\text{succ}}, a))}{N(v_{\text{succ}}, a)}$;
20. $a = v_{\text{succ}}^*. \text{last}()$; Sample observation $o \sim \mathcal{O}(a, b, a)$;
21. $V = \hat{r}_R(b_t, a) + \gamma \text{Simulate}(v_{\text{succ}}, \rho(b_t, a, o), \tau + 1),$
22. $N(v_{\text{succ}}, a) + 1 = 1; V(v_{\text{succ}}, a) = \frac{\sum_{a} N(v_{\text{succ}}, a)}{N(v_{\text{succ}}, a)}$; return $V$;
23. **end Function**
24. **Function** ComputeRisk($b$)
25. return $\sum_{a \in A} \mathcal{P}(\hat{s}^H = b | \sum_{s \in S} \mathbb{P}(\hat{s} | b) V^*, R, k+1(\hat{s}))$
26. **end Function**
27. **Function** Rollout($b, \tau$)
28. **if** $\tau = T - 1$ **then**
29. return $\sum_{k \in K} \mathbb{P}(k^H = b | \sum_{s \in S} \mathbb{P}(\hat{s} | b) V^*, R, k+1(\hat{s}))$
30. **else**
31. $a \sim \pi_{\text{rollout}}(b) \sim \mathcal{O}(a, b, a)$;
32. return $r_R(b, a) + \gamma \text{Rollout}(b, a, o), \tau + 1$;
33. **end if**
34. **end Function**

Open-loop chance-constrained Monte-Carlo belief tree search. POMCP \cite{23} combines Monte-Carlo simulation and game-tree search. Building upon POMCP, we propose an algorithm (Alg. 2) to solve (2) in an anytime manner. Our algorithm differs from POMCP in the following ways: 1) a search node, $v = (a, V(a), N(a))$, stores an action sequence $a$ and its associated statistics: $V(a)$ is the mean return of all simulations that execute $a$, and $N(a)$ counts the number of times that $a$ has been visited; 2) leaf node expansions must enforce the safety chance constraint (line 14); 3) terminal values are estimated using the pre-computed ql-$k$ values (line 11); 4) the active information gathering on latent states is explicitly realized via the augmented reward function $\hat{r}_R$.

Benefits of Algorithm 2. A search node only stores an action sequence since the open-loop optimization \cite{9} searches for an action sequence. In contrast to POMCP, in which a node stores a history of actions and observations, the search space in Alg. 2 is significantly reduced. Hence Alg. 2 can run in real-time under computational constraints. Quantal level-$k$ reasoning (Sec. IV) is exploited to estimate the terminal value when the maximum planning depth is reached (line 11). Specifically, the terminal belief state $b_{t+T}$ is used to determine the human’s intelligence level distribution $\mathbb{P}(k^H | b_{t+T})$; then, we assume the robot behaves as a ql-$(k^H + 1)$ agent (recall that a ql-$(k+1)$ agent quantally best responds to a ql-$k$ agent) with rationality coefficient $\lambda = 1$ and estimates the terminal value using the pre-computed robot’s ql-$(k^H + 1)$ value weighted by $\mathbb{P}(k^H | b_{t+T})$. By actively learning the belief state, the planner can quickly reduce the estimation error on the terminal values and maximize its performance.

Infeasibility handling mechanism. When the root node has no safe successors, we relax the chance constraint and find the action that minimizes the degree of constraint violation.

VI. EXPERIMENTS

A. Implementation Details

Test domain. We use autonomous driving as the test domain. In particular, we consider a forced merging scenario \cite{37}, where an autonomous car must merge to an adjacent (upper) lane that is occupied by a human-driven car (Fig. 1).

![Control feedback](image)

Fig. 1. An autonomous car interacts with a human-driven car controlled by a user through a racing wheel and a pedal in a forced merging scenario.

Reward functions. The reward function of a car is a linear combination of features $\phi : S \rightarrow \mathbb{R}^n$, satisfying $r(\hat{s}) = \omega^T \cdot \phi(\hat{s})$. The features encode safety, comfort, progress, etc. \cite{11}. The weights $\omega \in \mathbb{R}^n$ can be recovered via an inverse learning algorithm \cite{38}-[40]. Note that when the autonomous car runs Alg. 2, the safety feature in its reward function is deactivated since the safety is handled by (2c).

Level-0 policy. Alg. 1 requires a ql-0 policy to initiate the iterative reasoning process. Similar to [10], \cite{11}, we let a ql-0 agent be a non-strategic agent who treats others as static obstacles when making decisions.

Latent state dynamics. Recall (2c) that an explicit probabilistic model is required to govern the dynamics of the latent states. In this work, we consider single-shot games, i.e., the human’s latent states are assumed to be constant during
interaction. Hence, the latent states’ transition model is reduced to \( P(\theta_{t+1} | \theta_t, s_t, \sigma) = \mathbb{1}(\theta_{t+1} = \theta_t) \). In general repeated games, the transition model can be represented as a Markov chain and its parameters \( \sigma \) can be embedded in the POMDP and learned simultaneously as in [41].

**High-level robot planning.** The dynamics of the human-robot system are represented as

\[
\begin{bmatrix}
\dot{x} & \dot{y} & \dot{x}_H & \dot{y}_H & \dot{v}_x & \dot{v}_y & \dot{v}_H & \dot{a}_x & \dot{a}_y & \dot{a}_H
\end{bmatrix}, \quad (10)
\]

where \( x (y) \) is the longitudinal (lateral) position, \( v (w) \) is the longitudinal (lateral) speed, and \( a \) is the acceleration.

The sampling period is \( \Delta t = 0.5 [s] \). We use a state grid of the size \( 40 \times 40 \) to represent the discrete states of the human-robot system. The safety set \( \Omega_{safe} \) includes states in which the boundaries of the two agents do not overlap. We use \( \Delta = 0.05 \) and \( \Delta_s = \frac{1}{100} \) as the chance constraint thresholds in [9]. We let the highest intelligence level of the human be \( k_{max} = 2 \) based on experimental results in [42], [43]. The rationality coefficients take value from the set \( L = \{0.5, 0.8, 1.0\} \). The planning horizon in [9] is \( T = 8 \).

**Hierarchical planning and control.** Behavioral planning and control of the autonomous car are hierarchically connected. The planning layer (Alg. [2]) uses a low-fidelity model (10) to generate behavioral commands, and runs at \( 8Hz \) on a laptop with 2.8 GHz CPU. In the low-level control layer (running at \( 8Hz \)), the vehicle dynamics are represented by a high-fidelity bicycle model [44], and we use a model predictive controller [13], [45] to generate continuous controls that drive the system to the desired states generated by Alg. [2]. In Alg. [2] when the actual system state deviates from the state grid, the nearest neighboring grid will be exploited.

**Baselines.** We consider two baseline planners: 1) our planner without the feature of active information gathering, i.e., the autonomous car passively infers the human’s latent states (BLP-1); 2) the strategic game-theoretic planner in [8] that treats the human-driven car as a follower who accommodates the actions from the autonomous car (BLP-2). Both BLP-2 and our planner use a closed-loop feedback structure when building human behavioral models, but our planner reasons about the heterogeneity in the human’s cognitive limitations and irrationality through active inference rather than treating the human as a follower.

**B. The Human Behavioral Model**

**Human intelligence level interpretation.** The ql-\( k \) model is exploited to reason about human behaviors under bounded intelligence. Recall that the level-0 agent represents a non-strategic agent who treats others as static obstacles. Thus, a ql-1 agent can be interpreted as a cautious agent since it believes that its opponent is an aggressive non-strategic agent. On the contrary, a ql-2 agent behaves aggressively since it believes its opponent is a cautious ql-1 agent.

Fig. [2] shows the interactions between two cars modeled as ql-k agents in the forced merging task. The heat-map displays the ql-\( k \) value function \( (V^{*; i, k}) \) described in Sec. [IV-B] of the lower-lane car, indicating the preferred states; colder color means higher value. It can be observed in (a-b) that the interactions are seamless between a ql-1 agent and a ql-2 agent, which is expected since the ql-2 agent’s belief in the model of the ql-1 agent matches the ground truth (note that the high-value region in the upper lane encourages the red car to merge ahead of the white car (a); the low-value region in front of the white car guides it to yield (b)). However, when an agent’s true model deviates from the other agent’s belief, conflicts may occur. For instance, (e) shows the interaction between two ql-1 agents with a dead-lock because both agents prefer to yield. The low-value region in front of the yellow car discourages it to merge although it is safe. Similarly, when two ql-2 agents interact (d), collisions may occur as both agents think their opponents will be likely to yield. Note that the high value region in the upper lane encourages the yellow car to merge even if it is not safe.

![Fig. 2. Interactions between ql-k agents.](image)

**C. Case Studies and Quantitative Results**

We compare our planner against the baselines via simulations. The human-driven car is modeled as a ql-\( k \) agent, and is also controlled by the hierarchical planning and control scheme described in Sec. [VI-A] with behavioral commands obtained directly from the corresponding ql-\( k \) policies.

**Hypotheses.** We state the following two hypotheses: 1) active exploration improves efficiency of the robot’s planning; 2) our planner is robust to the human’s heterogeneous intelligence levels and irrational behaviors.

![Fig. 3. Planner comparison.](image)
car. With our planner, the autonomous car nudges in to explore the human’s latent states, then quickly decides to yield after observing the humans’ aggressive reactions. With BLP-2, the autonomous car initiates a dangerous merge as BLP-2 incorrectly assumes the human-driven car will likely yield. Note that when humans behave under heterogeneous cognitive states, it is critical for a robot to reason about such a heterogeneity to better predict human behaviors.

Metrics for quantitative results. In quantitative studies, we use two metrics for validating the hypotheses: 1) the rate of success (RS), which measures the percentage of simulations in which no collision or deadlock occurs; 2) the time used by the autonomous car to complete the merge (TM).

Results. We run 50 simulations for each of the scenarios in the case studies using our planner and two baseline planners. We first compare our planner against BLP-1 in terms of inference performance (BLP-2 has no inference capability). In Fig. 4 (a-b), we show the time history of the autonomous car’s belief in the ground truth human-driven car’s latent states. It can be observed that our planner can more effectively identify the human’s latent states by exploiting the mutual influence to make its human partner reveal his/her hidden states. In Fig. 4 (c), we show the average TM for the two scenarios. Note that our planner achieves the lowest TM, due to the active inference. Furthermore, our planner achieves the highest confidence on TM since our planner strategically generates actions that aim for triggering the most informative reactions from the human, this regulates human behaviors in a game-theoretic sense (such a phenomena is also observed in user studies Fig. 5(b)).

Next, we evaluate the robustness of our planner under various human intelligence levels and degrees of irrationality. We let the simulated ql-κ human take his/her rationality coefficient from Λ and run 100 simulations for each (λ, κ) combination. The human starts at a random position within 10 m around the autonomous car. In Fig. 4 (d-e), we show the RS of each planner for each scenario. Note that in all cases, our planner and BLP-1 achieve more than 95% RS. This is attributed to reasoning about the human’s bounded intelligence and irrationality, and enforcing the safety chance constraint. BLP-2 shows satisfactory RS when the simulated human is a ql-1 agent, but the RS decreases as the human becomes more irrational. When the human is a ql-2 agent, BLP-2 performs noticeably worse. The results indicate that our planner enables the robot to learn the human’s latent states, plan more effectively, and be robust to the human’s various intelligence levels and irrational behaviors.

D. User Study

Objective. We conduct user studies in which we let the autonomous car interact with a real human driver in a simulator (Fig. 1), showcasing the effectiveness of our planner.

Experiment setup. We recruited 10 human participants. For each participant, we ran Scenario 1 for 3 times using each of the planners. Here, the accelerations of the upper lane car are provided by human participants directly through a pedal.

Results. Fig. 5(a) shows the RS of each planner. It can be observed that both our planner and BLP-1 outperform the BLP-2 in terms of safety. Fig. 5(b) shows the time histories of the step reward collected by the autonomous car. Note that although the autonomous car pays more costs in the first few steps using our planner (due to the probing actions) it is able to complete the task much more effectively and safely compared with the baselines. In Fig. 5(c), we show the distributions of inferred intelligence levels and rationality coefficients of the human participants.

VII. CONCLUSION

We proposed an anytime game-theoretic planning framework that integrates iterative reasoning models, POMDP, and chance-constrained Monte-Carlo belief tree search for robot behavioral planning. Our planner enables a robot to safely and actively reason about its human partner’s latent cognitive states in real-time to maximize its utility more effectively. We applied the proposed approach to an autonomous driving domain where our behavioral planner and a low-level motion controller hierarchically control an autonomous car to negotiate traffic merges. Both simulation and user study results demonstrated the effectiveness of our planner compared with baseline planners.
