Curious Autonomous Robots in Uncertain Environments

Ran Tian, Haiming Gang, and David Isele

***Abstract*— Autonomous robots often need to operate in uncertain environments. Giving an agent the ability to be *curious* about its environment could improve its understanding and potentially aid in future planning. In this paper, we present a novel algorithm that provides a goal oriented agent with artificial curiosity, enabling the agent to explore as needed to better achieve its goal. The algorithm uses model-based open-loop planning which lends itself to use on robotic platforms where repeated trials are expensive, and safety is critical. We demonstrate how the proposed framework can be applied to enable autonomous vehicles to effectively negotiate traffic merges, and to improve the navigation of a Fetch robot under pose uncertainty.**

I. Introduction

Curiosity is closely linked to the learning processes in humans and other animals [1], [2], and it has been hypothesized that curiosity is a key component of general intelligence [3]. Here we take the view that curious behavior optimizes an intrinsic goal to actively collect information from the environment. This strategy has recently shown great improvements in reinforcement learning (RL) [4]–[6]. In this work, we show that these benefits extend to other forms of motion planning under uncertainty, where artificial curiosity gives a goal oriented agent a better means to reason about uncertainty by actively seeking to improve its understanding.

We show that in uncertain environments, where uncertainty has an impact on performance, curious behaviors can efficiently learn about the environment and incorporate that knowledge into the planning process. This produces improved performance compared to agents that select actions that are optimal for achieving a goal under the current expectations, but do not try to influence the uncertainty directly.

The benefits of augmenting goal seeking behavior with exploration is a growing focus in RL. An assortment of computational models have been proposed to formalize the seeking of novelty and surprise [4], [5], [7], [8] to better balance the trade-off between exploration and exploitation [9], [10]. But the difficulty of training still restricts the use of RL on most physical systems. D* and its variants [11], [12] propose search strategies that work in unknown environments based on expected costs, but do not allow for actions that explicitly reduce uncertainty. We look at information gathering in robotics [13], where particle filters

Ran Tian is with the University of California, Berkeley, CA 94720 USA (rantian@berkeley.edu).

Haiming Gang and David Isele are with Honda Research Institute USA, 70 Rio Robles, San Jose, CA 95134, USA ({hgang,disele}@honda-ri.com). Ran Tian conducted the research during an internship at HRI.

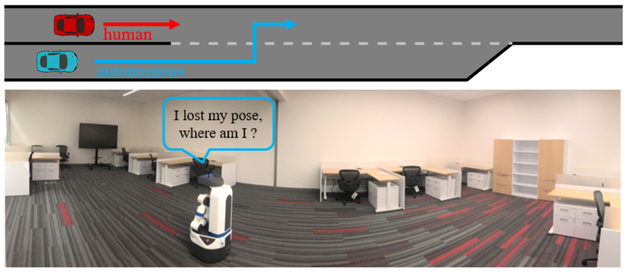

Fig. 1. Robots interact with uncertain environments. (Top) An autonomous vehicle (blue) is attempting to merge to the left lane but while a traffic participant is driving in a leading position. (Bottom) A Fetch robot is "kidnapped" to a room, it must know where it is in order to plan a path.

[14] and other data-driven belief propagation methods [15] have been used for solving global localization problems.

A common approach to handle planning in the face of uncertainty is to use Partially Observable Monte-Carlo Planning (POMCP). POMCP uses Monte-Carlo samples to simultaneously improve beliefs and search for optimal behaviors, and is known to converge to an optimal policy when the belief is correct [16]. However errors in the belief degrade performance. When coupled with the computation limits of running online, other strategies become preferable [17]–[19]. In particular, open-loop planning strategies that seek suboptimal sequences of open-loop actions, have been shown to perform well in real-world robotic planning problems [20]–[23] and sometimes optimally in these situations [20].

In this paper, we consider the decision-making of a goal oriented agent in uncertain environments. We formulate the problem as a Belief-state Markov Decision Process and adopt Receding-horizon Control (RHC) to solve for open-loop controls. This formulation provides the agent with artificial curiosity, enabling the agent to explore as needed to better achieve its goal. Using our general purpose formulation, we present interactive decision-making as an application: showing how curiosity can be useful when interacting with other agents in the case of autonomous vehicles. To emphasize the generality of our framework, we also formulate robot navigating as an application, demonstrating how a robot can efficiently navigate to a goal when uncertain about its location.

The contributions of this paper are as follows: 1) We present a curiosity driven open-loop planning algorithm for autonomous robot in uncertain environments. 2) We illustrate the uses of the proposed algorithm for both interactive decision-making on an autonomous vehicle and motion planning for robot navigation problem.

II. Problem formulation

Consider an ego agent whose behaviors are governed by a Partially Observable Markov Decision Process (POMDP) defined by tuple $< \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O} >$, where $\mathcal{S}$ is a finite state space, $\mathcal{A}$ is a finite action space, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0,1]$ is a state transition probability function, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is a reward function, Ω is a finite observation set, and $\mathcal{O} : \mathcal{S} \times \Omega \mapsto [0,1]$ is an observation probability function. As the agent's knowledge of its state $s \in \mathcal{S}$ is incomplete, it must maintain a belief distribution over $\mathcal{S}$. Then, the original POMDP can be represented using an equivalent Belief-state Markov Decision Process (B-MDP) defined by tuple $< \mathcal{S}, \mathcal{B}, \mathcal{A}, \mathcal{T}', \mathcal{R}', \Omega, \mathcal{O} >$. The notation $\mathcal{B} = \{b \in [0,1]^{|\mathcal{S}|} \mid \|b\|_1 = 1\}$ is the $(|\mathcal{S}|-1)$-dimensional probability simplex representing the belief state space ($|\mathcal{S}|$ denotes the cardinality of $\mathcal{S}$, and $\|(\cdot)\|_1$ denotes the $\mathcal{L}_1$ norm). A belief state $b_t := [\mathbb{P}(s_t = s_1), \dots, \mathbb{P}(s_t = s_{|\mathcal{S}|})]^\intercal$ is a vector representing the belief distribution over the set $\mathcal{S}$ at the time instant t, $t \in \mathbb{N} \cup \{0\}$. The notation $\mathcal{T}' : \mathcal{B} \times \mathcal{A} \mapsto \mathcal{B}$ is a belief state transition function. The notation $R' : \mathcal{B} \times \mathcal{A} \mapsto \mathbb{R}$ is a reward function on belief states and it is defined as follows:

$$\mathcal{R}'(b,a) = \sum_{s \in \mathcal{S}} b(s) R(s,a). \tag{1}$$

We let $\pi^* : \mathcal{B} \times \mathcal{A} \mapsto [0,1]$ denote the optimal policy of the B-MDP, and it is defined $\pi^*(b,a) := \mathbb{P}(a_t = a | b_t = b)$ for $\forall b \in \mathcal{B}, \forall a \in \mathcal{A}$ and is modeled:

$$\pi^*(b,a) = \frac{\exp(Q^*(b,a))}{\sum_{a' \in \mathcal{A}} \exp(Q^*(b,a'))}. \tag{2a}$$

The state-action value function, $Q^* : \mathcal{B} \times \mathcal{A} \mapsto \mathbb{R}$, represents the expected total future reward and is defined:

$$Q^*(b,a) = \mathcal{R}'(b,a) + \gamma \max_{a' \in \mathcal{A}} \left[Q^*\big(\mathcal{T}'(b,a), a'\big)\right], \tag{3}$$

where γ is a discount factor. At each time instant t, the agent applies the action $a_t \sim \pi^*$ (a_t is sampled from π^*). Then after the time instant t, the agent receives an observation $o_t \in \Omega$ induced by the action a_t and updates its belief state at the time instant $t+1$ following:

$$b_{t+1} = \rho(b_t, a_t, o_t), \tag{4}$$

where $\rho : \mathcal{B} \times \mathcal{A} \times \Omega \mapsto \mathcal{B}$ is a function that maps belief state $b_t \in \mathcal{B}$, action $a_t \in \mathcal{A}$, and observation $o_t \in \Omega$ to belief state $b_{t+1} \in \mathcal{B}$. Each element in b_{t+1} is computed using the recursive Bayesian inference formula:

$$b_{t+1}(s) = \frac{\mathcal{O}(s,o) \sum_{s' \in \mathcal{S}} \big(\mathcal{T}(s',a,s) b_t(s')\big)}{\sum_{s'' \in \mathcal{S}} \Big(\mathcal{O}(s'',o) \sum_{s' \in \mathcal{S}} \big(\mathcal{T}(s',a,s'') b_t(s')\big)\Big)}. \tag{5}$$

We note that solving Q^* directly using stochastic dynamic programming can be prohibitively expensive in real-world problems. In this work, we propose a curiosity-driven open-loop planning approach that exploits RHC and uses a curiosity heuristic to approximate a suboptimal Q value online, avoiding the need to solve the Bellman equation.

III. Curiosity-driven Open-loop Planning

A. Open-loop planning via Receding-horizon control

RHC is a predictive control scheme that is widely applied to control systems with state and control constraints [24]. The key idea of RHC is to obtain a finite sequence of open-loop control actions by solving a finite horizon optimal control problem online at each time instant, using an approximate system dynamics model to predict the responses. The agent applies the first action in the sequence, measures the new state, and re-plans. We define a local Q-value function that approximates Q^* based on RHC:

$$\hat{Q}(b,a) = \max_{\mathbf{a} \in \tilde{\mathbf{A}}} \sum_{\tau=0}^{N-1} \gamma^\tau \mathcal{R}'(b_{\tau|t}, a_{\tau|t}) \big| a_{0|t} = a, b_{0|t} = b, \tag{6}$$

where $N \in \mathbb{Z}_+$ is a specified planning horizon, the notation $(\cdot)_{\tau|t}$ means a predicted value of the variable $(\cdot)$ at the time $t+\tau$. The notation $\mathbf{a} = \{a_{0|t}, \dots, a_{N-1|t}\}$ is a sequence of open-loop control actions along the planning horizon, $\tilde{\mathbf{A}}$ is a sampled subset of $\mathcal{A}^N$, and $b_{\tau|t}$ is the predicted belief state at the time $t+\tau$ based on the belief state transition function $\mathcal{T}'$ (defined in Section III-C). Using $\tilde{\mathbf{A}}$ reduces the computation load when running the algorithm on real platforms.

B. Belief state entropy as curious motivation

RHC based open-loop planning strategies have been used to treat POMDP problems in [21]–[23]. The strategies in [22], [23], only exploit the available information, where optimizing (6) can easily lead to negative effects when belief estimation errors result in different planned behaviors. The strategy in [21], seeks to reduce system uncertainty, but, because it does not adaptively balance the extrinsic rewards, it can produce overly cautious exploration.

Seeking to combine goal-directed objectives with the perceived benefits of human curiosity, we aim to "instill" curiosity into the agent by adding an intrinsic curiosity reward into the Q-value approximation. Similar ideas have been proposed for RL methods [5], [25], but these are generally impractical for use on physical systems.

In this work, we model the curiosity reward as the accumulative belief state entropy over the planning horizon. The local Q-value function is defined:

$$\tilde{Q}(b,a) = \max_{\mathbf{a} \in \tilde{\mathbf{A}}} \sum_{\tau=0}^{N-1} \gamma^\tau \mathcal{R}'(b_{\tau|t}, a_{\tau|t}) + \lambda(b) \sum_{\tau=1}^{N-1} \gamma^\tau \mathcal{C}(b_{\tau|t}) \big| a_{0|t} = a, b_{0|t} = b, \tag{7}$$

where the function $\mathcal{C} : \mathcal{B} \mapsto \mathbb{R}$ measures the negative entropy of a belief state, and the function $\lambda : \mathcal{B} \mapsto \mathbb{R}$ is a regulation function that trades off curious exploration and exploitation:

$$\mathcal{C}(b) = \sum_{s \in \mathcal{S}} b(s) \log(b(s)),\ \forall b \in \mathcal{B}, \tag{8a}$$

$$\lambda(b) = \alpha \cdot \mathrm{abs}(\mathcal{C}(b)), \tag{8b}$$

where $\alpha > 0$ is a tuning parameter. Intuitively, the second summation of (7) is the confidence of the expected knowledge gained when taking a given trajectory, and $\lambda(b)$ weights the importance of this knowledge by our current uncertainty. The ego agent's policy with curious exploration at each time t is then modeled:

$$\tilde{\pi}_t(b,a) = \frac{\exp(\tilde{Q}(b,a))}{\sum_{a' \in \mathcal{A}} \exp(\tilde{Q}(b,a'))}. \tag{9}$$

We refer to the planning algorithm that uses $\tilde{\pi}_t$ as Curiosity Driven Open-Loop Planning (CDOLP). Once $\tilde{\pi}_t$ is obtained, the agent applies an action sampled from $\tilde{\pi}_t$, updates b_{t+1},

Algorithm 1: Curiosity Driven Open-Loop Planning

```
1 Given POMDP model $(\mathcal{S},\mathcal{A},\mathcal{T},\mathcal{R},\Omega,\mathcal{O})$, planning horizon $N$,
    discount factor $\gamma$, and parameter $\alpha$.
2 Initialize $b_0$.
3 for $t = 1, 2, \ldots$ do
4     Compute a suboptimal policy $\tilde{\pi}_t$ following (9).
5     Execute $a_t \sim \tilde{\pi}_t$ and collect new observation $o_t$.
6     Update belief state $b_t$ following (4) and (5).
7 end for
```

and solves for $\tilde{\pi}_{t+1}$. The planning algorithm is summarized in Algorithm 1.

C. Monte-Carlo belief propagation

Optimizing $\tilde{Q}$ relies on the belief transition function $\mathcal{T}'$ to predict $b_{\tau|t}$ along the planning horizon. The predicted belief state $b_{\tau|t}$ must be a posterior distribution that accounts for the possible future observations at $t+\tau$.We define $\mathcal{T}'$ as $\mathcal{T}'(b_{\tau|t}, a_{\tau|t}) = b_{\tau+1|t}$, where each element in $b_{\tau+1|t}$ can be computed recursively:

$$b_{\tau+1|t}(s \mid b_{\tau|t} = b, a_{\tau|t} = a) = \sum_{o\in\Omega}\Big(\rho(s|b,a,o)\sum_{s'\in\mathcal{S}}\sum_{s''\in\mathcal{S}}\big(\mathcal{O}(s',o)\cdot\mathcal{T}(s'',a,s')b(s'')\big)\Big), \quad (10)$$

where $\rho(s|b,a,o)$ is the belief in s based on the output of $\rho(b,a,o)$. The full derivation is in App. VII-A. For convenience, we rewrite (10) as:

$$b_{\tau+1|t}(s \mid b_{\tau|t} = b, a_{\tau|t} = a) = \sum_{o\in\Omega}\big(\rho(s|b,a,o)\tilde{O}^{b,a}_{\tau|t}(o)\big), \quad (11)$$

where the notation $\tilde{O}^{b,a}_{\tau|t} := [\mathbb{P}(o_{\tau|t} = o_1 | b_{\tau|t} = b, a_{\tau|t} = a), \ldots, \mathbb{P}(o_{\tau|t} = o_{|\Omega|} | b_{\tau|t} = b, a_{\tau|t} = a)]^\intercal$ is a vector representing the predicted observation distribution after the time instant $t+\tau$, given the belief state and the action of the ego agent at the time instant $t+\tau$.

We note that (11) imposes computational challenges since it checks each possible future observation. In order to reduce the complexity, we make use of a Monte Carlo strategy to sample observations from the predicted observation probability distribution and approximate $b_{\tau+1|t}$ as follows:

$$b_{\tau+1|t}(s \mid b_{\tau|t} = b, a_{\tau|t} = a) = \sum_{k=1, o_k\sim\tilde{O}^{b,a}_{\tau|t}}^{n} \rho(s|b,a,o_k)\tilde{O}^{b,a}_{\tau|t}(o^k), \quad (12a)$$

$$b_{\tau+1|t} = \frac{b_{\tau+1|t}}{\|b_{\tau+1|t}\|_1}, \quad (12b)$$

where $n \in \mathbb{N}$ is a specified sample number. Intuitively, (12a) draws sample observations using importance sampling to approximate $b_{\tau+1|t}$, and (12b) re-normalizes the approximated belief state.

IV. Application to Interactive Decision-Making for Autonomous Vehicle

A. Problem statement

We apply CDOLP to the problem of interactive decision making. We consider the problem of forced merging: an autonomous vehicle must control longitudinal and lateral speeds to merge into an occupied lane as quickly as possible.

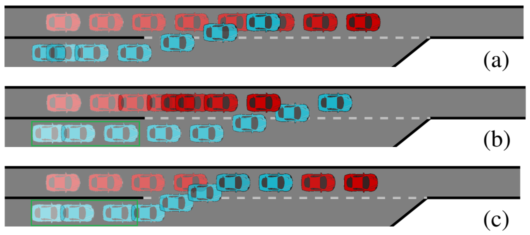

Fig. 2. Visualization of a forced merging scenario. (a) shows the ego vehicle (blue) controlled by a pure exploitation strategy with a cautious/aggressive traffic vehicle (red); (b-c) show the ego vehicle controlled by CDOLP with a (b) cautious and (c) aggressive traffic vehicle.

Forced merging is an active research topic in autonomous driving literature [26], [27]. Most existing approaches target an open gap and maneuver the autonomous vehicle into the open gap. However, in dense traffic, the ego agent (autonomous vehicle) may be forced to wait arbitrarily long for a gap to appear, frustrating the occupants and any traffic participants stuck behind the ego agent. In this paper, we consider a two-agent[1] forced merging scenario. We apply the proposed CDOLP algorithm to enable the autonomous vehicle to actively interact with a traffic vehicle: indicating a desire to merge, monitoring the responses of traffic vehicle, and promptly seizing on any opportunity.

B. Observation function and state transition function

We first define the observation function and the state transition function in the POMDP model described in Sec. II.

In this merging scenario, the state is composed of two parts $s = (\tilde{s}, \theta)$: $\tilde{s} = (\tilde{s}^{\mathrm{e}}, \tilde{s}^{\mathrm{h}}) \in \tilde{\mathcal{S}}$ is a collection of the physical states of the ego vehicle (denoted by the superscript e) and the traffic vehicle (denoted by the superscript h) and represents the state of the merging scenario; $\theta \in \Theta$ parameterizes the traffic vehicle's decision strategy. We define the state evolution using a discrete-time model:

$$\tilde{s}_{t+1} = \mathcal{F}(\tilde{s}_t, a^{\mathrm{e}}_t, a^{\mathrm{h}}_t), \quad (13)$$

where $\tilde{s}^i = (x^i, y^i, v^i)$, $i \in \mathcal{P} = \{\mathrm{e}, \mathrm{h}\}$, denotes the kinematics of agent i, and is modeled using a unicycle model described in App. VII-C. The action of agent i is $a^i = (a^{i,1}, a^{i,2}) \in \mathcal{A}^i$, with $a^{i,1}$ denoting the longitudinal acceleration and $a^{i,2}$ denoting the lateral acceleration.

We note that $\tilde{s}$ is observable while θ represents the internal state of the traffic participant and cannot be observed. We assume the ego vehicle can perfectly measure position (i.e., $o := \tilde{s}$), and we define the observation function as a one-hot vector encoding:

$$\mathcal{O}(s,o) = \begin{cases} 1 & \text{if } \tilde{s} = o, \\ 0 & \text{otherwise,} \end{cases} \quad \forall s \in \mathcal{S}, \forall o \in \Omega. \quad (14)$$

[1]Though we consider two vehicles, the method can be extended to multi-vehicle interactions at a greater computational cost.

We define the state transition function as follows:

$$\mathcal{T}(s', a, s) = \mathbb{P}(s_{t+1} = s | s_t = s', a_t^{\mathrm{e}} = a)$$
$$= \sum_{a' \in \mathcal{A}^{\mathrm{h}}} \Big(\mathbb{I}\big(\tilde{s} = \mathcal{F}(\tilde{s}', a, a') \text{ and } \theta = \theta'\big) \cdot \mathbb{P}(a_t^{\mathrm{h}} = a' | s_t = s')\Big),$$
$$\forall s, s' \in \mathcal{S}, \forall a \in \mathcal{A}^{\mathrm{e}}, \tag{15}$$

where $\mathbb{I}(\cdot)$ is an indicator function (taking 1 if the event $(\cdot)$ is true, and 0 otherwise), $\tilde{s}$ is the first component in s, $\tilde{s}'$ is the first component in s', θ is the second component in s, and θ' is the second component in s'. Intuitively, the indicator function governs that the state transition follows the dynamic model (13) and ensures the consistency of θ during transitions. The full derivation is in App. VII-B. The notation $\mathbb{P}(a_t^{\mathrm{h}} = a' | s_t = s') = \pi^{h,\theta'}(s', a^{\mathrm{h}})$ represents a stochastic policy of the traffic vehicle. In this work, we adopt a game-theoretic approach to model traffic vehicle decision-making.

C. Game-theoretic interaction modeling

Game theory is a framework that models strategic interactions among rational human decision-makers. The k-level game framework is a cognitive model that characterizes human agent behavior in strategic games based on the assumptions of bounded rationality [28], [29]. In the k-level game framework, a human agent is assumed to make decisions based on her level of intelligence (*i.e*, cognitive-level). Specifically, a level-k human agent, $k \in \mathbb{N}$, assumes that other agents in the game can be modeled as level-$(k-1)$ agents, predicts the other agents' actions based on this assumption, and optimally responds to the predicted actions. The k-level framework has been exploited for modeling human behavior in autonomous driving [22], [30], [31] and competitive games [32]. We use the k-level framework to model the traffic vehicle. In particular, we let k take values in $K = \{1, 2\}$, as most human drivers' cognitive-level is level-1 or level-2 [29].

The key idea of k-level game theory is that a human's policy can be well-defined by her cognitive-level. If we let the parameter θ model a human's cognitive-level ($\theta = k$ and $\theta \in K$), then $\pi^{h,\theta}$ can be defined. The construction of a level-k policy follows from [22], [31] and is shown in App. VII-D. In short, a level-1 traffic vehicle represents a cautious driver and a level-2 traffic vehicle represents an aggressive driver.

Finally, the state transition model can be well-defined:

$$\mathcal{T}(s', a, s) = \sum_{a' \in \mathcal{A}^{\mathrm{h}}} \Big(\mathbb{I}\big(\tilde{s} = \mathcal{F}(\tilde{s}', a, a') \text{ and } \theta = \theta'\big) \cdot \pi^{\mathrm{h},\theta'}(\tilde{s}', a')\Big),$$
$$\forall s, s' \in \mathcal{S}, \forall a \in \mathcal{A}^{\mathrm{e}}, \tag{16}$$

where $\pi^{\mathrm{h},\theta'}$ is the level-θ' policy of the traffic vehicle. With $\mathcal{O}$ and $\mathcal{T}$ defined, the belief state transition function $\mathcal{T}'$ (10) is now well-defined.

Note that the uncertainty in forced merging arises from the traffic vehicle's uncertain cognitive-level. Our CDOLP algorithm promotes the ego agent to actively learn the traffic vehicle's level while planning for the main task. Modeling a human's cognitive level as a hidden state and learning it through Bayesian inference has been proposed in [22], [32], but the active learning considered in this paper has not been incorporated in these works.

D. Reward function

The reward functions of the ego vehicle and the traffic vehicle used to define the level-k policies are:

$$R^i(s_t, a_t^i, a_t^j) = w^\intercal \Phi^i(s_{t+1}), \tag{17}$$

where $i \in \mathcal{P}$, s_{t+1} is obtained based on the dynamic model (13), $w = [1, 10]^\intercal$ is a weight vector, $\Phi^i(s_{t+1}) = [\phi^{i,1}(s_{t+1}), \phi^{i,2}(s_{t+1})]^\intercal$ is a feature vector defined:

$$\phi^{i,1}(s) = -|x_{\text{goal}} - x^i| - |y_{\text{goal}} - y^i|, \tag{18a}$$
$$\phi^{i,2}(s) = \mathbb{I}_{\text{colli}}(s), \tag{18b}$$

where feature $\phi^{i,1}$ characterizes the vehicle's behavior in reaching its target location, and feature $\phi^{i,2}$ characterizes the safe status of the vehicle, taking -1 if two vehicle collide, and 0 otherwise.

The ego agent's B-MDP reward function is defined:

$$\mathcal{R}(s, a^{\mathrm{e}}) = \mathbb{E}\big(R^{\mathrm{e}}(s, a^{\mathrm{e}}, a^{\mathrm{h}} | a^{\mathrm{h}} \sim \pi^{\mathrm{h},\theta})\big), \tag{19}$$

where a^{h} is sampled from the traffic vehicle's level-θ policy $\pi^{\mathrm{h},\theta}$ and θ is a component in s.

E. Simulation results

The action space of the ego vehicle is $\mathcal{A}^{\mathrm{e}} = \{(-6,0),(0,0),(6,0),(0,3)\}m/s^2$, and that of the traffic vehicle is $\mathcal{A}^{\mathrm{h}} = \{(-6,0),(0,0),(6,0)\}m/s^2$. The planning horizon is $N = 5$, the reward discount factor is 0.9, the value of the parameter α in (8b) is $\alpha = 10$, and the sampling time Δt is chosen as $0.5s$.

We show a forced merging scenario where the traffic vehicle is in a leading position to highlight the functionality of CDOLP. Fig. 2 (a) shows the interactions between the ego vehicle (blue) controlled by a pure exploitation strategy in [22][2] and either a cautious (level-1) or aggressive (level-2) traffic vehicle (red). In both cases, the ego vehicle yields. In Fig. 2 (b-c), we show an ego vehicle controlled by CDOLP. The agent actively indicates a desire to merge by accelerating (see the green box in Fig. 2 (b-c)), as it predicts that, by accelerating, it can receive more information of the traffic participant's cognitive-level. The time history of the belief state b_t entropy is shown in App. VII-E. The curiosity reward effectively guides the ego agent to reduce the ambiguity of the uncertain environment.

V. Application to robot navigation under uncertainty

A. Problem statement

In this section, we show how curiosity can be used to solve a variant of the global localization problem [14]. These experiments allows us to show the generality of our approach as well more extensive quantitative analysis. We consider a

[2]The strategy is similar to CDOLP, but Q-value is approximated using (6) with only extrinsic reward and the belief propagation process does not account for future observations.

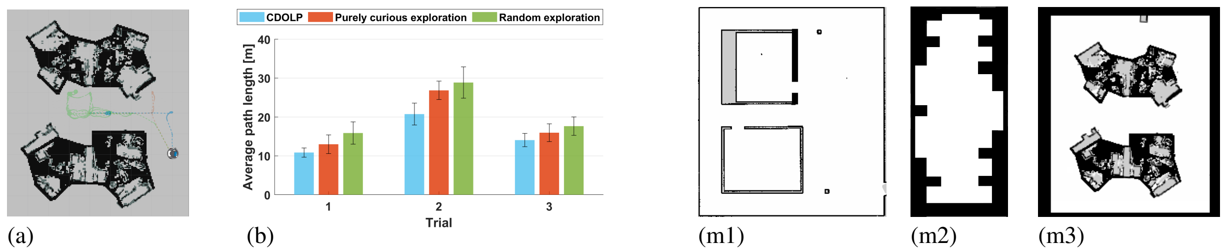

Fig. 3. (a) Robot trajectories using three different navigation algorithms. (b) Average path length with three different start and goal locations. Three maps used to evaluate our algorithms. (m1) is the playground environment in Fetch robotics's Gazebo simulator; (m2) and (m3) are two indoor offices.

scenario where an autonomous robot is placed at an unknown location in a known environment. The robot needs to go to a desired location, but is uncertain about its location, as illustrated in the bottom of Fig 1. Unlike the traditional localization problems where the objective is only to localize, here the objective is to reach a goal. While localization is required to make a valid path, the time and behaviors spent on localizing effect the performance of the agent. In this paper, we apply CDOLP to efficiently navigate the Fetch robot to the goal.

B. Observation function and state transition function

In this problem, the state is composed of two parts: $s = (s^1, \xi) \in \mathcal{S}$. The first part $s^1 = (x, y, \theta) \in \mathcal{S}^1$ is the kinematics of the robot and is modeled:

$$\begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix} = f(s^1_t, a_t) = \begin{bmatrix} x + a^1 \\ y + a^2 \\ \theta + a^3 \end{bmatrix}, \tag{20}$$

where $a_t = (a^1, a^2, a^3) \in \mathcal{A}$ is the action of the robot and is composed by the desired displacements along the x-axis, y-axis, and heading angle. The actions of the robot are high-level actions and the low-level control is handled by [33]. The second part of the state $\xi \in \mathcal{X}$ is a feature vector containing key information about the environment which can be measured by the robot. In this work, the Fetch robot is equipped with a Laser Range Finder and can measure the occupancy status of the five cells in the occupancy map in front of the robot (illustrated in App. VII-F). The resolution of the occupancy map is Δl and we set $\Delta l = 1m$.

Formally, We define feature vector as follows:

$$\xi_t = \{\xi^i_t\},\ i = 1, \ldots, 5, \tag{21}$$

where ξ^i_t is an indicator, taking 1 if the cell i is occupied, and 0 otherwise, based on the laser scan at t. The feature vector is observable to the robot (i.e., $o = \xi$, $o \in \Omega = \mathcal{X}$), and we define the observation probability function as follows:

$$\mathcal{O}(s, o) = \begin{cases} \delta & \text{if } o = \xi, \\ \frac{1-\delta}{|\Omega|} & \text{otherwise,} \end{cases},\ \forall s \in \mathcal{S}, \forall o \in \Omega, \tag{22}$$

where $\delta \in [0, 1]$ is a tuning parameter used to account for the sensory noise. We model the state transition probability function as follows:

$$\mathcal{T}(s, a, s') = \begin{cases} \delta' & \text{if } s' = f(s, a), \\ \frac{1-\delta'}{|\mathcal{S}|} & \text{otherwise,} \end{cases},\ \forall s, s' \in \mathcal{S}, \forall a \in \mathcal{A}, \tag{23}$$

where $\delta' \in [0, 1]$ is a tuning parameter used to account for the low-level control error.

C. Reward function

The reward function $\mathcal{R}$ of the robot (ego agent) is:

$$\mathcal{R}(s, a) = \sum_{s' \in \mathcal{S}} \mathcal{T}(s, a, s') r(s'),\ \forall s \in \mathcal{S}, \forall a \in \mathcal{A}, \tag{24a}$$

$$r(s') = -|x - x_g| - |y - y_g| - |\theta - \theta_g| - 10 \cdot \mathbb{I}_{\text{colli}}(s'), \tag{24b}$$

where (x_g, y_g, θ_g) denotes the goal state of the robot, $\mathbb{I}_{\text{colli}}(s')$ is a binary indicator, taking 1 if the robot state is in an occupied cell, and taking 0 otherwise. The reward function encourages the robot to reach the goal state as quickly as possible while avoiding unsafe states.

We consider navigation in three different maps shown in Fig. 3. The action space of the robot is $\mathcal{A} = \{\pm 1, \pm 1, \pm \frac{\pi}{2}\}$, the planning horizon is $N = 5$, the reward discount factor is 0.9, the value of the parameter α in (8a) is $\alpha = 10$, the values of the parameters δ and δ' in (22) and (23) are 0.8 and 0.9, respectively.

We evaluate three navigation algorithms: 1) **random exploration**: the robot selects random actions to explore the environment and updates its belief state using (5); 2) **purely curious exploration**: the robot approximates the Q-value following (7) but with only a curiosity reward and plans actions using the procedure in Algorithm 1; 3) **CDOLP**: the robot plans with curiosity *and* balances the exploration/exploitation trade-off. In all three algorithms, once the pose confidence reaches 99%, an $\mathbf{A}^*$ planning algorithm is activated to plan a valid path to the goal.

We first simulate the Fetch robot in the Gazebo simulator to evaluate the three algorithms on three different maps. For each map, we run 20 trials. Each trial consists of a different start and goal location. Trials are repeated three times to collect statistics. The results of m3 are shown in Fig. 4, m1 and m2 are shown in VII-G. CDOLP is consistently the most efficient at reaching the goal.

We verify the results on the physical Fetch robot. The physical Fetch robot is run in the indoor office environment shown in Fig. 3(m3). Fig. 3(a) shows a trace of the different robot behaviors. The random exploration strategy (green) spends a long time exploring the same area and gaining little new information. The purely curious strategy (red) is effective at gathering information, but the exploration behavior does not take into account the potential costs

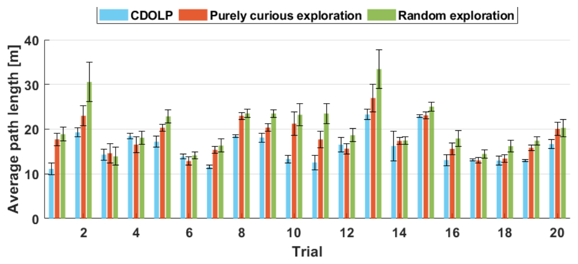

Fig. 4. Average path length of the three algorithms with various start and goal locations in the indoor office environment in Fig. 3(m3).

associated with exploration and therefore takes actions that are good at improving the belief, but put the agent at a disadvantage when it comes to exploiting the knowledge. With CDOLP (blue), the robot simultaneously balances the cost of exploration with reaching the goal and can decide when additional information is no longer required to act optimally. In Fig. 3(b), we show the average path length taken across 3 trials (each consists of a different start and goal location; trials are repeated three times to collect statistics) and the error bars indicate the standard error.

VI. Conclusion

We presented a novel algorithm that provides a goal oriented agent with artificial curiosity, enabling the agent to explore as needed to better achieve its goal. The algorithm uses model-based open-loop planning which lends itself for use on robotic platforms where repeated trials are expensive, and safety is critical. We demonstrated how the proposed framework can be applied to enable autonomous vehicles to effectively negotiate traffic merges, and to improve the navigation of a Fetch robot under pose uncertainty.

References

[1] P. Waelti, A. Dickinson, and W. Schultz, "Dopamine responses comply with basic assumptions of formal learning theory," *Nature*, vol. 412, no. 6842, p. 43, 2001.
[2] P. Dayan and B. W. Balleine, "Reward, motivation, and reinforcement learning," *Neuron*, vol. 36, no. 2, pp. 285–298, 2002.
[3] J. Schmidhuber, "Formal theory of creativity, fun, and intrinsic motivation (1990–2010)," *IEEE Transactions on Autonomous Mental Development*, vol. 2, no. 3, pp. 230–247, 2010.
[4] M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos, "Unifying count-based exploration and intrinsic motivation," in *Advances in Neural Information Processing Systems*, 2016, pp. 1471–1479.
[5] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *International Conference on Machine Learning (ICML)*, 2017.
[6] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *arXiv preprint arXiv:1801.01290*, 2018.
[7] M. Lopes, T. Lang, M. Toussaint, and P.-Y. Oudeyer, "Exploration in model-based reinforcement learning by empirically estimating learning progress," in *Advances in Neural Information Processing Systems*, 2012, pp. 206–214.
[8] B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine, "Diversity is all you need: Learning skills without a reward function," *arXiv preprint arXiv:1802.06070*, 2018.
[9] T. L. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Advances in applied mathematics*, vol. 6, no. 1, pp. 4–22, 1985.
[10] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "The nonstochastic multiarmed bandit problem," *SIAM journal on computing*, vol. 32, no. 1, pp. 48–77, 2002.
[11] S. Koenig and M. Likhachev, "Dˆ* lite," *Aaai/iaai*, vol. 15, 2002.
[12] M. Przybylski and B. Putz, "D* extra lite: A dynamic a* with search–tree cutting and frontier–gap repairing," *International Journal of Applied Mathematics and Computer Science*, vol. 27, no. 2, pp. 273–290, 2017.
[13] J. R. Souza, R. Marchant, L. Ott, D. F. Wolf, and F. Ramos, "Bayesian optimisation for active perception and smooth navigation," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 4081–4087.
[14] D. Fox, S. Thrun, W. Burgard, and F. Dellaert, "Particle filters for mobile robot localization," in *Sequential Monte Carlo methods in practice*. Springer, 2001, pp. 401–428.
[15] D. W. Cheng Peng, "Map as the hidden sensor: fast odometry-based global localization," *arXiv preprint arXiv:1910.00572*, 2019.
[16] D. Silver and J. Veness, "Monte-carlo planning in large pomdps," in *Advances in neural information processing systems*, 2010, pp. 2164–2172.
[17] A. Weinstein and M. L. Littman, "Open-loop planning in large-scale stochastic domains," in *Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.
[18] D. Perez Liebana, J. Dieskau, M. Hunermund, S. Mostaghim, and S. Lucas, "Open loop search for general video game playing," in *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*. ACM, 2015, pp. 337–344.
[19] E. Lecarpentier, G. Infantes, C. Lesire, and E. Rachelson, "Open loop execution of tree-search algorithms," 2018.
[20] C. Yu, J. Chuang, B. Gerkey, G. Gordon, and A. Ng, "Open-loop plans in multi-robot pomdps," Technical report, Stanford CS Dept, Tech. Rep., 2005.
[21] M. Lauri and R. Ritala, "Planning for robotic exploration based on forward simulation," *Robotics and Autonomous Systems*, vol. 83, pp. 15–31, 2016.
[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," *arXiv preprint arXiv:1908.04005*, 2019.
[23] N. Li, A. Girard, and I. Kolmanovsky, "Stochastic predictive control for partially observable markov decision processes with time-joint chance constraints and application to autonomous vehicle control," *Journal of Dynamic Systems, Measurement, and Control*, vol. 141, no. 7, 2019.
[24] M. Morari and J. H. Lee, "Model predictive control: past, present and future," *Computers & Chemical Engineering*, vol. 23, no. 4-5, pp. 667–682, 1999.
[25] S. Mohamed and D. J. Rezende, "Variational information maximisation for intrinsically motivated reinforcement learning," in *Advances in neural information processing systems*, 2015, pp. 2125–2133.
[26] M. Bouton, A. Nakhaei, K. Fujimura, and M. J. Kochenderfer, "Cooperation-aware reinforcement learning for merging in dense traffic," *arXiv preprint arXiv:1906.11021*, 2019.
[27] D. Isele, "Interactive decision making for autonomous vehicles in dense traffic," in *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2019.
[28] D. O. Stahl and P. W. Wilson, "On players' models of other players: Theory and experimental evidence," *Games and Economic Behavior*, vol. 10, no. 1, pp. 218–254, 1995.
[29] 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, Dec. 2006.
[30] N. Li, D. W. Oyler, M. Zhang, Y. Yildiz, I. Kolmanovsky, and A. R. Girard, "Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems," *IEEE Transactions on control systems technology*, vol. 26, no. 5, pp. 1782–1797, 2017.
[31] R. Tian, S. Li, N. Li, I. Kolmanovsky, A. Girard, and Y. Yildiz, "Adaptive game-theoretic decision making for autonomous vehicle control at roundabouts," in *2018 IEEE Conference on Decision and Control (CDC)*. IEEE, 2018, pp. 321–326.
[32] R. Tian, N. Li, I. Kolmanovsky, and A. Girard, "Beating humans in a penny-matching game by leveraging cognitive hierarchy theory and bayesian learning," in *2020 Annual American Control Conference (ACC)*, 2020.
[33] M. Wise, M. Ferguson, D. King, E. Diehr, and D. Dymesich, "Fetch and freight: Standard platforms for service robot applications," in *Workshop on autonomous mobile service robots*, 2016.

VII. Appendix

A. Full derivation of belief propagation

$$
\begin{aligned}
&b_{\tau+1|t}(s \mid b_{\tau|t} = b, a_{\tau|t} = a) \\
&= \sum_{o\in\Omega} \mathbb{P}(s_{\tau+1|t} = s, o_{\tau|t} = o | b_{\tau|t} = b, a_{\tau|t} = a) \\
&= \sum_{o\in\Omega} \big(\mathbb{P}(s_{\tau+1|t} = s | o_{\tau|t} = o, b_{\tau|t} = b, a_{\tau|t} = a) \\
&\qquad \cdot \mathbb{P}(o_{\tau|t} = o | b_{\tau|t} = b, a_{\tau|t} = a)\big), \\
&= \sum_{o\in\Omega} \big(\rho(s|b,a,o)\mathbb{P}(o_{\tau|t} = o | b_{\tau|t} = b, a_{\tau|t} = a)\big) \\
&= \sum_{o\in\Omega} \Big(\rho(s|b,a,o) \sum_{s'\in\mathcal{S}} \sum_{s''\in\mathcal{S}} \big(\mathcal{O}(s',o)\cdot\mathcal{T}(s'',a,s')b(s'')\big)\Big).
\end{aligned} \tag{25}
$$

B. Full derivation of state transition function

$$
\begin{aligned}
&\mathcal{T}(s',a,s) = \mathbb{P}(s_{t+1} = s | s_t = s', a_t^{\mathrm{e}} = a) \\
&= \sum_{a'\in\mathcal{A}^{\mathrm{h}}} \big(\mathbb{P}(s_{t+1} = s | s_t = s', a_t^{\mathrm{e}} = a, a_t^{\mathrm{h}} = a')\cdot \\
&\qquad \mathbb{P}(a_t^{\mathrm{h}} = a' | s_t = s')\big) \\
&= \sum_{a'\in\mathcal{A}^{\mathrm{h}}} \Big(\mathbb{I}\big(\tilde{s} = \mathcal{F}(\tilde{s}', a, a') \text{ and } \theta = \theta'\big)\cdot \\
&\qquad \mathbb{P}(a_t^{\mathrm{h}} = a' | s_t = s')\Big),\ \forall s, s' \in \mathcal{S}, \forall a \in \mathcal{A}^{\mathrm{e}}.
\end{aligned} \tag{26}
$$

C. Unicycle model

The unicycle model used for the autonomous vehicles follows:

$$
\begin{bmatrix} x_{t+1}^i \\ y_{t+1}^i \\ v_{t+1}^{i,1} \\ v_{t+1}^{i,2} \end{bmatrix} = f(s_t^i, a_t^i) = \begin{bmatrix} x_t^i + v_t^{i,1}\,\Delta t \\ y_t^i + v_t^{i,2}\,\Delta t \\ v_t^{i,1} + a_t^{i,1}\,\Delta t \\ v_t^{i,2} + a_t^{i,2}\,\Delta t \end{bmatrix}, \tag{27}
$$

where $(x^i, y^i, v^{i,1}, v^{i,2})$ represent, respectively, the x coordinate in the ground-fixed frame, the y coordinate in the ground fixed frame, the longitudinal speed, and the lateral speed of the agent i,

D. Human level-k policy construction

For agent i, $i \in \mathcal{P}$, its level-k policy, $k \in K$, is defined as follows:

$$
\pi^{i,k}(\tilde{s}, a) = \frac{\exp\big(Q^{i,k}(\tilde{s}, a)\big)}{\sum_{a'\in\mathcal{A}^i} \exp\big(Q^{i,k}(\tilde{s}, a')\big)},\quad \forall \tilde{s} \in \tilde{\mathcal{S}},\ \forall a \in \mathcal{A}^i, \tag{28a}
$$

$$
\begin{aligned}
&Q^{i,k}(\tilde{s}, a) = \max_{a_{1|t}^i,\dots,a_{N-1|t}^i} \mathbb{E}\Big(\\
&\sum_{\tau=0}^{N-1} \gamma^\tau R^i(\tilde{s}_{\tau|t}, a_{\tau|t}^i, a_{\tau|t}^j) | \tilde{s}_{0|t} = \tilde{s}, a_{0|t}^i = a, a_{\tau|t}^j \sim \pi^{j,k-1}\Big),
\end{aligned} \tag{28b}
$$

where $R^i : \tilde{\mathcal{S}} \times \mathcal{A}^i \times \mathcal{A}^j \mapsto \mathbb{R}$ is the reward function of agent i, $\pi^{j,k-1}$ denotes the level-$(k-1)$ policy of agent j, $j \in \mathcal{P}$ and $j \neq i$. We remark that the level-k policy of agent i is constructed based on the level-$(k-1)$ policy of agent j, as agent i predicts agent j's actions using $\pi^{j,k-1}$.

In order to construct the level-k policy of the traffic vehicle ($\pi^{\mathrm{h},k}$, $k \in K$), we require the level-0 policies of the ego vehicle and the traffic vehicle. Following [30], [31], we let $\pi^{i,0}$, $i \in \mathcal{P}$, select actions that maximize agent i's reward function and assume agent j, $j \in \mathcal{P}$ and $j \neq i$, is stationary along its planning horizon (*i.e.*, a level-0 agent plans without considering interactions). With $\pi^{i,0}$ defined, $\pi^{\mathrm{h},k}$ can be constructed sequentially and iteratively following (28).

E. Time history of the belief state entropy

In Fig. 5, we show the time history of the belief state entropy. It can be observed that curiosity reward can effectively guide the ego agent to reduce the ambiguity of the uncertain environment while planning for the main task.

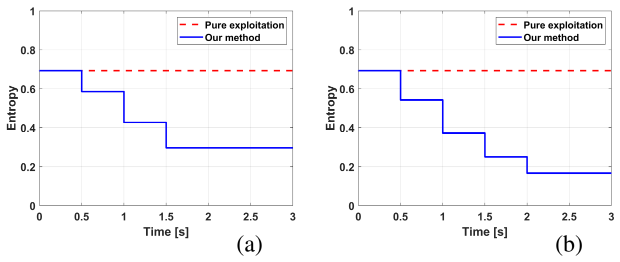

Fig. 5. Entropy of the belief over the human driver's cognitive level; (a) AV interacts with a cautious driver; (b) AV interacts with an aggressive driver.

F. Visualization of the robot's perception system

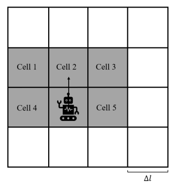

Fig. 6. Visualization of the robot's perception system relative to the occupancy grid; the arrow represents the facing direction.

G. Simulation results

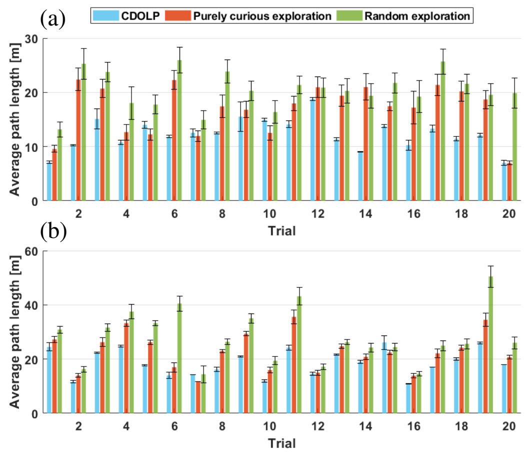

Fig. 7. Average path length of the three algorithms with various start and goal locations; (a) playground environment in Fig. 3 (m1); (b) indoor office environment in Fig. 3 (m2).