Stochastic Dynamic Games in Belief Space

Wilko Schwarting, Alyssa Pierson, *Member, IEEE*, Sertac Karaman, *Member, IEEE*, and Daniela Rus, *Fellow, IEEE*

***Abstract*—Information gathering while interacting with other agents under sensing and motion uncertainty is critical in domains such as driving, service robots, racing, or surveillance. The interests of agents may be at odds with others, resulting in a stochastic noncooperative dynamic game. Agents must predict others' future actions without communication, incorporate their actions into these predictions, account for uncertainty and noise in information gathering, and consider what information their actions reveal. Our solution uses local iterative dynamic programming in Gaussian belief space to solve a game-theoretic continuous POMDP. Solving a quadratic game in the backward pass of a game-theoretic belief-space variant of iterative linear-quadratic Gaussian control (iLQG) achieves a runtime polynomial in the number of agents and linear in the planning horizon. Our algorithm yields linear feedback policies for our robot, and predicted feedback policies for other agents. We present three applications: Active surveillance, guiding eyes for a blind agent, and autonomous racing. Agents with game-theoretic belief-space planning win 44% more races than without game theory and 34% more than without belief-space planning.**

***Index Terms*—Game-theoretic planning, motion and path planning, multirobot systems, optimization and optimal control.**

I. Introduction

We aim to develop planners for multiagent systems that are robust under uncertainty and combine information-seeking behavior with game-theoretic reasoning. While game theory can model the interaction and dependency among agents, it does not address the quality of the information available to the agent for decision-making. Agents must plan and act within a game, remain robust to uncertainty, gain information, and leverage the information gain to improve their control policies. We propose an approach that combines game-theoretic planning with belief-space planning (BSP), leveraging the interaction models from game theory while incorporating uncertainties in the modeled dynamics and perception. In multiagent systems, we find that agents gather information to reduce uncertainty while maintaining decision-making strategies that support complex interactions. Applications include assistive robotics, surveillance, pursuer–evader games, and racing, see Fig. 1.

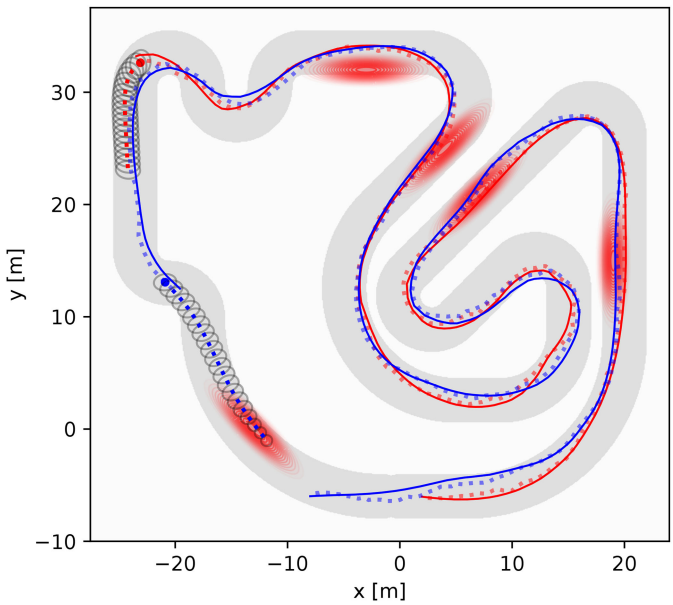

Fig. 1. One application we present is dynamic racing. Here, the blue agent starts with a disadvantage but is equipped with better acceleration and capable of moving faster through corners than the red agent. Our approach allows the blue agent to overtake and win the race. Planned trajectories and chance constraints are shown in dashed lines and ellipses. The traces correspond to the true state (solid) and the noisy EKF-estimate (dashed) available to each agent during the race. The red areas are zones with low noise observations and reduce uncertainty.

While each agent operates independently and does not reveal plans or intentions through communication, agents have approximate models of the other agents. Dynamic models allow us to infer the ability of other agents to move in the environment, approximate cost models encode the agents' objectives. Models of how other agents perceive the world from observations allow estimating what other agents know and with what certainty. Our work is related to model predictive receding horizon planners: Agents start from an initial belief about themselves, others, and the environment and imagine how the future will evolve if they and other agents were to execute certain actions. These models can be prescribed, observed, sensed, or communicated. As plans are not shared among agents, we propose a game-theoretic framework that predicts the interactive policies of other agents while optimizing for our own policy.

Within the game-theoretic framework, agents take actions that increase their information gain, which in turn results in the ability to improve their control policies with reduced uncertainty. For example, an assistive robot tasked with guiding a human may

Manuscript received October 22, 2020; accepted February 15, 2021. Date of publication May 24, 2021; date of current version December 6, 2021. This article was recommended for publication by Associate Editor S. J. Chung and Editor P. R. Giordano upon evaluation of the reviewers' comments. *(Corresponding author: Wilko Schwarting.)*

Wilko Schwarting, Alyssa Pierson, and Daniela Rus are with the Computer Science, and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: wilkos@csail.mit.edu; apierson@csail.mit.edu; rus@csail.mit.edu).

Sertac Karaman is with the Laboratory of Information, and Decision Systems (LIDS), Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: sertac@mit.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TRO.2021.3075376.

Digital Object Identifier 10.1109/TRO.2021.3075376

explore the environment to reduce uncertainty and better navigate. Conversely, a game-theoretic setting can model adversarial agents. Here, agents may choose to *hide* to prevent others from gathering information about themselves, which is relevant to surveillance applications. In the context of racing, agents may force others to increase their uncertainty, such as by pressuring them to drive too fast in a corner which increases uncertainty in their state, or by simply pushing them into the dark. It is therefore not only important to reason about a robot's own uncertainty but also the uncertainty of other agents in the environment, and even more so how one's actions impact the change in the uncertainty of others.

Game-theoretic models have not only proven useful to model interactions between autonomous systems but also in integrating interactive human predictions into autonomous decision-making and planning. We can model the actions of humans as expected cost-minimizing and estimate human cost functions from past observed trajectories with inverse reinforcement learning (IRL) [1]. Consequently, computing expected cost-minimizing actions based on the learned cost functions generates human predictions. The expected cost-minimizing behavior can also be interpreted as the best response to an autonomous agent's actions. This best response setting allows us to estimate how the autonomous system's actions influence human actions. The autonomous system can therefore implicitly control the human's actions to a certain degree. This technique has been applied to predict interactive human behavior for autonomous vehicles [2]–[4], and to predict pedestrians [5]. The combination of game-theoretic modeling of human behavior and information-seeking planning is therefore even more promising.

For instance, home service robots can provide assistance and support to humans, particularly the elderly population. These robots need to work near humans, gauge the human's intent, and understand the state of mind of others to better perform tasks. They have to avoid confusion and misunderstandings and will need to seek information about both their environment and surrounding humans. Additionally, these autonomous systems need to also reason about the amount of information and understanding the human has about the robot. The robot can aid the human's understanding through explicit communication, as well as implicitly through behavior, such as moving to visible locations or indicating intent by unambiguously moving in the desired direction.

We propose a solution that combines multiagent game-theoretic decision-making under uncertainty and BSP. Our approach supports robust solutions to a wide range of multirobot applications in which dealing with uncertainty, the need to gather information, and game-theoretic decision-making are fundamental. We build on important advancements in two areas: Game-theoretic planning and BSP. Game-theoretic planning successfully solves problems where an agent's objective is at odds with the objective of other agents, such as in modeling human behavior in traffic [4], [6], [7], and leveraging the effects on humans by autonomous cars [2]. The authors in[8] give a recent review on game theory and control. In game theory, the Nash equilibrium is a proposed solution of a noncooperative game involving two or more players. Each player is assumed to know the equilibrium strategies of the other players, and no player has anything to gain by changing only their own strategy. Solving for Nash equilibria has been applied to competitive racing [9]–[11] and guiding vehicles through intersections [12]. Solution methods include iterated best response [9], [11], [13], [14], iterative quadratic approximations [15], [16], using discrete payoff matrices [10], or solving the necessary conditions [4]. We will solve the necessary condition of a static quadratic game at each stage in the backward-pass of iterative linear-quadratic Gaussian control (iLQG) to solve for the Nash equilibrium of the dynamic game.

While game-theoretic planning models the interaction and dependency among agents, it does not address the quality of information available to the agent for decision-making. BSP [17] uses beliefs, which are the distribution of the robot's state estimate, to represent the uncertainties in the perception of the robot. The problem of computing a control policy over the space of belief states is formally described as a partially observable Markov decision process (POMDP), and has been studied extensively. Solutions to POMDPs are known to be very complex. Solving a POMDP to global optimality is NP-hard: Solutions such as point-based algorithms [18]–[21] in discrete space are bound to the curse of history, as well as sampling-based solvers [22]–[24]. Optimization-based approaches have been developed for planning in continuous belief space [25]–[29], by approximating beliefs as Gaussian distributions and computing a value function valid in local regions of the belief space. Similarly to [28], [29], we avoid the common maximum-likelihood observation assumption [26], [27]. In comparison to point-based algorithms which scale exponentially in the planning horizon l, optimization-based methods scale linearly, $\mathcal{O}(l)$.

A. Main Assumptions

To generate reasonable predictions about other agents, we build on approximate prior information. Consider the analogy of a race car driver. A driver knows that other race cars will have comparable driving characteristics, while different classes of vehicles, like trucks, will have different handling dynamics. They also know that the other racing drivers desire to go as fast as possible around the track to win the race, without crashing into other vehicles or sliding off the road, similar to a cost model. Lastly, they have experience in how other drivers observe the track and that the quality of perception decreases in the dark. For our autonomous systems, Assumption 1 describes that we assume *common knowledge* of models about the world.

Assumption 1 (Common Knowledge): Agents have models for cost, dynamics, and observations of other agents.

Related game-theoretic works, with applications ranging from pedestrian-robot interactions to autonomous racing, make similar assumptions by either prescribing dynamics and cost models [6], [9]–[11] or learning cost models through IRL [2], [4], [5]. Assumption 1 allows agents to imagine how the future will evolve: If the robot and other agents were to execute given policies, how would model-based predictions of motions and observations in the world impact the beliefs over time? We do not assume any form of direct communication between agents

and therefore do not have access to the policies of other agents. Instead, we predict interactive policies of other agents through game-theory and leveraging the models of costs, dynamics, and observations. The robot, as part of the game, can then leverage the influence of its actions on the predicted actions of other agents to their advantage. We will show in a competitive racing example that Assumption 1 can be relaxed in practice and that approximate models of other agents prove sufficient to improve performance.

We refer to beliefs as distributions over states and draw inspiration over how we design our system from the cognitive theory of mind. The cognitive theory of mind [30] defines the ability to attribute mental states, such as beliefs, intents, or desires to oneself and others. It is integral to understanding that others have beliefs that are different from one's own. Single-agent planning in belief space, reasoning about the uncertainty of only the own state, is limited to zero-order beliefs (e.g., I think...). In contrast, we will also reason about the uncertainty of other agents. The theory of mind refers to this as first-order belief spaces (e.g., I think they think...). Higher-order beliefs such as second-order belief spaces (e.g., I think they think that I think...) are beyond the scope of this article as they quickly become computationally intractable by essentially defining beliefs over beliefs. We find that parametrizing belief spaces efficiently is essential to generating real-time capable algorithms. Assumption 2, keeps computation complexity at a reasonable level and avoids an explosion in parameters in the recursive beliefs over beliefs.

Assumption 2 (First-Order Beliefs): Planning and prediction are limited to first-order beliefs: Any robot i's belief over another agent j is the same as that agent j's belief about themselves.

In Section IV, we evaluate cases, such as competitive racing, where this assumption is a simplification of the true system dynamics. In the racing scenario, all agents execute separate instances of our algorithm and therefore maintain separate beliefs. Thus, an agent's belief about themselves does not necessarily match the beliefs that others have about them. However, while these belief mismatches may occur, we see performance improvements over a game-theoretic baseline without BSP, see Section IV-C, which highlights the importance of accounting for uncertainty and information gain in competitive racing and other applications.

The purpose of Assumptions 1 and 2 is to enable interactive predictions of other agents in belief space while maintaining computational tractability. Since our approach is executed continuously in a receding horizon fashion, and we compute policies that are reactive to deviations from the predicted beliefs, the proposed method can adapt if the observed behavior differs from the predicted ones. Our approach continues to successfully control the agent under reasonable violations of the presented assumptions, such as if the dynamics, observation, or cost models are inaccurate, if their own beliefs do not exactly match the beliefs of others, or if the other agent's optimization is suboptimal.

B. Contributions

We present a computationally tractable solution to multiagent planning that combines game-theoretic planning and BSP to interact within a problem formulated as a game, gain information, and leverage the information gain to improve the agents' control policies. The main limiting factor in applying either game theory or BSP, and even more so the combination of both to robotic control problems lies in the associated computational complexity. To the best of our knowledge, this is the first work to combine general dynamic games and planning in belief space into an efficient real-time algorithm. The main contributions of this paper are: 1) A method for computing Nash equilibria for dynamic games in belief space; 2) A linear feedback policy, similar to linear-quadratic Gaussian control (LQG), for the robot resulting from the solution, and also a predicted linear feedback policy for all other agents; 3) Belief and control trajectory based regularization to ensure convergence; 4) Evaluation of the proposed method in three stochastic dynamic games: racing with autonomous vehicles, active surveillance, and guiding eyes for a blind agent.

We organize the remainder of the article as follows: Section II introduces dynamic games in belief space, including a general definition of best response POMDPs and a Nash equilibrium formulation of the noncooperative dynamic game. We give the resulting problem definition in Section II-A and, assuming beliefs can be represented in the form of Gaussian distributions, approximate the belief dynamics based on an extended Kalman filter (EKF) detailed in Section III-A. Our method computes a locally optimal solution to the best response POMDP problem with continuous state and action spaces and nonlinear dynamics and observation models by iteratively solving for a local Nash equilibrium, outlined in Section III. We utilize a belief-space variant of iLQG to compute the Nash equilibrium, Section III-C, by solving for a local Nash equilibrium at each stage of the backward pass, see Section III-B. At each iteration, each agent's value function is approximated based on a quadratization around a nominal trajectory, and the belief dynamics are approximated with an extended Kalman filter. We describe regularization techniques in Section III-D to ensure that the algorithm converges regardless of initial conditions. Based on these findings, we introduce Algorithm 1 in Section III-E describing the full belief-space Nash equilibrium computation.

We show the potential of our approach in Section IV by presenting three multiagent problems that combine our game-theoretic formulation with information-seeking behavior: Active surveillance, guiding blind agents, and racing with autonomous vehicles.

II. Dynamic Games in Belief Space

We first define POMDPs in their most general form (following notation of [28] and [31]), then formulate the resulting game, derive the Nash Equilibrium, and present an iterative solution method. The following notation is summarized in Tables I.

We write the BSP problem as a stochastic optimal control problem. Consider a system of N agents $i \in \{1, \ldots, N\}$, with agent i's state at time k denoted $\mathbf{x}_k^i \in \mathrm{R}^{n_{\mathbf{x}^i}}$, measurement as $\mathbf{z}_k^i \in \mathrm{R}^{n_{\mathbf{z}^i}}$, and control input $\mathbf{u}_k^i \in \mathrm{R}^{n_{\mathbf{u}^i}}$. Here, $n_{\mathbf{x}^i}$, $n_{\mathbf{z}^i}$, $n_{\mathbf{u}^i}$ define the dimensionality of agent i's state, measurement, and control. For brevity we refer to $\mathbf{x}_k = [\mathbf{x}_k^{1,\top}, \ldots, \mathbf{x}_k^{N,\top}]^\top \in \mathrm{R}^{n_\mathbf{x}}$ as the joint state, $\mathbf{z}_k = [\mathbf{z}_k^{1,\top}, \ldots, \mathbf{z}_k^{N,\top}]^\top \in \mathrm{R}^{n_\mathbf{z}}$ as the joint

TABLE I
MAIN SYMBOLS AND NOTATION

$\mathbf{x}, \mathbf{u}, \mathbf{z}$	State, control input, and measurement
$\mathbf{b},\ \mathbf{s} = [\mathbf{b}^\top, \mathbf{u}^\top]^\top$	Belief, short for belief and controls
Q^i, V^i	Action-value and value function of agent i
π^i	Optimal control policy of agent i
$j_k,\ K_k$	Feedforward and feedback gains at time k
$c^i_k(\mathbf{b}_k, \mathbf{u}_k),\ c^i_l(\mathbf{b}_l)$	Cost of agent i at time k, and terminal cost
$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{m}_k)$	State transition with process noise $\mathbf{m}_k$
$\mathbf{z}_k = h(\mathbf{x}_k, \mathbf{n}_k)$	Measurement function with meas. noise $\mathbf{n}_k$
$\mathbf{b}_{k+1} = \beta(\mathbf{b}_k, \mathbf{u}_k, \mathbf{z}_{k+1})$	Belief transition
$\mathbf{b} = \bar{\mathbf{b}} + \delta\mathbf{b}$	Nominal + perturbation, similar for $\mathbf{u}, \mathbf{s}$
$c^i_{\mathbf{s},k},\ c^i_{\mathbf{ss},k}$	Gradiant and Hessian of c^i evaluated at $\bar{\mathbf{s}}_k$
$g_{\mathbf{s},k},\ W_{\mathbf{s},k}$	Jacobians of g and W evaluated at $\bar{\mathbf{s}}_k$
$V_{\mathbf{b},k},\ V_{\mathbf{bb},k}$	Gradient and Hessian of value at time k
$Q_{\mathbf{s},k},\ Q_{\mathbf{ss},k}$	Gradient and Hessian of action-value at k

measurement, and $\mathbf{u}_k = [\mathbf{u}_k^{1,\top}, \ldots, \mathbf{u}_k^{N,\top}]^\top \in \mathrm{R}^{n_\mathbf{u}}$ as the joint control, consisting of all agents. We refer to the joint dimensions as $n_\mathbf{x} = \sum_i n_{\mathbf{x}^i}, n_\mathbf{z} = \sum_i n_{\mathbf{z}^i}$, and $n_\mathbf{u} = \sum_i n_{\mathbf{u}^i}$. The notation $\neg i$ indicates all agents except i, e.g., $\mathbf{u}_k^{\neg i}$ relates to the controls of all other agents except i. We will refer to $\mathbf{u} = [\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{l-1}]$ as the control trajectory until time l. The joint *belief* $\mathbf{b}(\mathbf{x}_k)$ is defined as the distribution of the state $\mathbf{x}_k$ given all past control inputs and sensor measurements, and consists of individual beliefs $\mathbf{b}^i$. For brevity, we define $\mathbf{s} = [\mathbf{b}^\top, \mathbf{u}^\top]^\top$.

Following [28] and [31], we compute the belief by

$$\mathbf{b}(\mathbf{x}_k) = p(\mathbf{x}_k | \mathbf{u}_0, \ldots, \mathbf{u}_{k-1}, \mathbf{z}_1, \ldots, \mathbf{z}_k) \tag{1}$$

from all past control inputs and sensor measurements. The stochastic dynamics and observation model, here formulated in probabilistic notation as

$$\mathbf{x}_{k+1} \sim p(\mathbf{x}_{k+1} | \mathbf{x}_k, \mathbf{u}_k), \quad \mathbf{z}_k \sim p(\mathbf{z}_k | \mathbf{x}_k) \tag{2}$$

allow us to forward propagate the belief given a control input $\mathbf{u}_k$ and a measurement $\mathbf{z}_{k+1}$ through Bayesian filtering

$$\mathbf{b}(\mathbf{x}_{k+1}) = \eta p(\mathbf{z}_{k+1} | \mathbf{x}_{k+1}) \int p(\mathbf{x}_{k+1} | \mathbf{x}_k, \mathbf{u}_k) \mathbf{b}(\mathbf{x}_k) \mathrm{d}\mathbf{x}_k. \tag{3}$$

In (3), η is a normalizer independent of $\mathbf{x}_{k+1}$ and $\mathbf{b}(\mathbf{x}_{k+1})$ and, contains the uncertainty originating from the stochastic dynamics, the uncertain measurement and the uncertainty in the belief at the previous time step. We employ the shorthand $\mathbf{b}_k$ to refer to $\mathbf{b}(\mathbf{x}_k)$. The stochastic *belief dynamics* are defined by (3) and are written as

$$\mathbf{b}_{k+1} = \beta(\mathbf{b}_k, \mathbf{u}_k, \mathbf{z}_{k+1}). \tag{4}$$

The expected return of each individual agent i under a control trajectory of all agents $\mathbf{u}$, including its own control trajectory $\mathbf{u}^i$, subject to uncertainty on the observed measurements $\mathbf{z}$ over the horizon l is determined by the action-value function Q^i, defined

$$Q^i(\mathbf{b}_0, \mathbf{u}) = \mathop{\mathbb{E}}_{\mathbf{z}} \left[c^i_l(\mathbf{b}_l) + \sum_{k=0}^{l-1} c^i_k(\mathbf{b}_k, \mathbf{u}_k) \right]. \tag{5}$$

Here $c^i_k(\cdot)$ denotes the cost at time k and $c^i_l(\cdot)$ denotes the terminal cost of agent i. Since there exists an action-value function for each agent, there are N distinct action-value functions Q^i for $i \in \{1, \ldots, N\}$.

We will first formulate the two problems of (1) solving the general POMDP best response game, and then (2) finding the Nash equilibrium of this game.

Problem 1 (POMDP Best Response Game): Given an initial belief $\mathbf{b}_0$, for agents $i \in \{1, \ldots, N\}$, we need to solve the stochastic optimal control problem

$$\pi^i = \arg\min_{\mathbf{u}^i} \ Q^i(\mathbf{b}_0, \mathbf{u}) \quad \forall i \in \{1, \ldots, N\} \tag{6}$$

$$\text{s.t. } \mathbf{b}_{k+1} = \beta(\mathbf{b}_k, \mathbf{u}_k, \mathbf{z}_{k+1}) \tag{7}$$

for each agent by minimizing each agent's expected cost with respect to their own controls $\mathbf{u}^i$, where $Q^i(\mathbf{b}_0, \mathbf{u})$ is the action-value function of agent i.

Note that all agents' optimal policies π^i depend on the actions of all other agents because each agent i minimizes their own action-value function $Q^i(\mathbf{b}_0, \mathbf{u})$. The result is a noncooperative game [32] in which all agents' policies depend on the optimal policies of all other agents $\pi^i(\pi^{\neg i})$. Since all policies are optimized jointly and severally, the dependence of agent i's policy π^i on other agents' controls $\mathbf{u}^{\neg i}$ is resolved by inserting their optimal policy $\pi^{\neg i}$. We therefore denote π^i instead of $\pi^i(\mathbf{u}^{\neg i})$.

A general solution to (6) can be defined recursively by the Bellman equation

$$\begin{aligned} V^i_l(\mathbf{b}_l) &= c^i_l(\mathbf{b}_l) \\ Q^i_k(\mathbf{b}_k, \mathbf{u}_k) &= c^i_k(\mathbf{b}_k, \mathbf{u}_k) + \mathop{\mathbb{E}}_{\mathbf{z}_{k+1}} \left[V^i_{k+1}(\beta(\mathbf{b}_k, \mathbf{u}_k, \mathbf{z}_{k+1})) \right] \\ V^i_k(\mathbf{b}_k) &= \min_{\mathbf{u}^i_k} Q^i_k(\mathbf{b}_k, \mathbf{u}_k) \\ \pi^i_k(\mathbf{b}_k) &= \arg\min_{\mathbf{u}^i_k} Q^i_k(\mathbf{b}_k, \mathbf{u}_k) \end{aligned} \tag{8}$$

where $V^i_k(\mathbf{b}_k)$ is the value function and $\pi^i_k(\mathbf{b}_k)$ the optimal policy at time k. Note that in (8) the cost $c^i_k(\mathbf{b}_k, \mathbf{u}_k)$, the reached value function $V^i_{k+1}(\beta(\mathbf{b}_k, \mathbf{u}_k, \mathbf{z}_{k+1}))$, and therefore the action-value function $Q^i_k(\mathbf{b}_k, \mathbf{u}_k)$ of agent i depends not only on its own action but also on all other players' actions. This interdependence is analogous to (6) but formulated recursively over time.

To better capture how an agent's action-value function depends on the controls of all other actions, we can equivalently write $Q^i(\mathbf{b}_0, \mathbf{u}) = Q^i(\mathbf{b}_0, \mathbf{u}^i, \mathbf{u}^{\neg i})$. More precisely, the interdependence of all players optimal policies is captured in the Nash equilibrium of Problem 1, defined in Problem 2. Problem 2 formulates a sufficient condition for Nash equilibria [32], [33] in belief space.

Problem 2 (Nash Equilibrium): Find the optimal control policy $\pi = [\pi^{1,\top}, \ldots, \pi^{N,\top}]^\top$ that yields a local Nash equilibrium of the POMDP best response game in Problem 1, such that it satisfies

$$Q^i(\mathbf{b}_0, \mathbf{u}^i, \pi^{\neg i}) \geq Q^i(\mathbf{b}_0, \pi^i, \pi^{\neg i}), \forall i \in \{1, 2, \ldots, N\} \tag{9}$$

for all $\mathbf{u}^i$ in the neighborhood of π^i.

More intuitively, in the Nash equilibrium no player has anything to gain by changing only their own strategy. Based on the necessary condition of Problem 2, we will derive a local necessary condition for each subproblem in the backward pass of our game-theoretic variant of belief iLQG.

A. Problem Formulation

The difficulty in solving POMDPs stems from the infinite-dimensional space of all beliefs, and that in general the value function cannot be expressed in parametric form. To overcome these challenges we describe beliefs by Gaussian distributions, approximating the belief dynamics using an EKF, and a quadratic approximation of the value function about a nominal trajectory through the belief space. We iteratively compute a local Nash equilibrium over all agents in the proximity of the nominal trajectory by solving the necessary condition (9) of Problem 2 at each timestep during a belief-space variant of iLQG to perform the Bellman backward recursion in (8). Due to its similarity to iLQG we benefit from linear scaling $\mathcal{O}(l)$ in the planning horizon l, in contrast to point-based POMDP algorithms which scale exponentially.

We are given nonlinear stochastic dynamics and observation models in state-transition notation

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{m}_k), \qquad \mathbf{m}_k \sim \mathcal{N}(0, I) \tag{10}$$

$$\mathbf{z}_k = h(\mathbf{x}_k, \mathbf{n}_k) \qquad \mathbf{n}_k \sim \mathcal{N}(0, I) \tag{11}$$

where $\mathbf{m}_k$ and $\mathbf{n}_k$ are the motion and measurement noise, respectively. Without loss of generality, we draw both the motion and measurement noise from independent Gaussian distributions with zero mean and unit variance since the noise can be arbitrarily transformed inside these functions. Depending on the system, motion, and sensing noise may be state and control dependent.

Note that formulating the general dynamics and measurement functions jointly of all agents includes, but is not limited to, the special case of independent functions for each agent i as in

$$f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{m}_k) = [f^1(\mathbf{x}_k^1, \mathbf{u}_k^1, \mathbf{m}_k^1)^\top, \ldots, f^N(\mathbf{x}_k^N, \mathbf{u}_k^N, \mathbf{m}_k^N)^\top]^\top \tag{12}$$

$$h(\mathbf{x}_k, \mathbf{n}_k) = [h^1(\mathbf{x}_k^1, \mathbf{n}_k^1)^\top, \ldots, h^N(\mathbf{x}_k^N, \mathbf{n}_k^N)^\top]^\top. \tag{13}$$

We define the Gaussian belief as $\mathbf{b}_k = (\hat{\mathbf{x}}_k^\top, \Sigma_k)$, by the mean state $\hat{\mathbf{x}}_k$ and the variance Σ_k of the normal distribution describing the stochastic state $\mathbf{x}_k \sim \mathcal{N}(\hat{\mathbf{x}}_k, \Sigma_k)$.

III. Technical Approach

Before detailing the value iteration method for the Nash equilibrium solution based on a game-theoretic belief-space variant of iLQG in Section III-C, we need to derive two important components. First, we describe the approximation of the general Bayesian filter update (4) by an EKF in Section III-A to formulate the Gaussian belief dynamics. This allows us to forward propagate Gaussian beliefs given an initial belief and a control trajectory which we utilitze in the game-theoretic variant of belief-space iLQG. Second, we show that the necessary condition of Problem 2, the Nash equilibrium, is equivalent to a local necessary condition at each timestep in the Bellman recursion in Section III-B. The full algorithm is detailed in Section III-E.

A. Bayesian Filter and Belief Dynamics

The Bayesian filter in (4) defines the general belief dynamics of a current belief $\mathbf{b}_k$ and measurement $\mathbf{z}_{k+1}$. To make the belief propagation tractable we follow [28] and approximate the Bayesian filter by an EKF, suitable for nonlinear Gaussian beliefs as well as nonlinear dynamics and measurement models. For well-defined transition models, the EKF is the standard for nonlinear state estimation [34], [35]. The EKF makes a first-order approximation of f with respect to the stochastic variable $\mathbf{x}$, such that for a given belief $\mathbf{b}_k = (\hat{\mathbf{x}}_k, \Sigma_k)$ we have the standard EKF update equations [28], [31]

$$\begin{aligned} \hat{\mathbf{x}}_{k+1} &= f(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0) + K_k(\mathbf{z}_{k+1} - h(f(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0), 0)) \\ \Sigma_{k+1} &= \Gamma_{k+1} - K_k H_k \Gamma_{k+1} \end{aligned} \tag{14}$$

with corresponding matrices defined by

$$\begin{aligned} \Gamma_{k+1} &= A_k \Sigma_k A_k^T + M_k M_k^T \\ K_k &= \Gamma_{k+1} H_k^\top (H_k \Gamma_{k+1} H_k^\top + N_k N_k^\top)^{-1} \\ A_k &= \frac{\partial f}{\partial \mathbf{x}}(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0), \quad M_k = \frac{\partial f}{\partial \mathbf{m}}(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0) \\ H_k &= \frac{\partial h}{\partial \mathbf{x}}(f(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0), 0), \quad N_k = \frac{\partial h}{\partial \mathbf{n}}(f(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0), 0). \end{aligned} \tag{15}$$

The noisy measurement $\mathbf{z}_k$ in the belief update makes the belief dynamics stochastic. We define $\mathbf{b}_k = [\hat{\mathbf{x}}_k^\top, \text{vec}(\Sigma_k)^\top]^\top$, where $\text{vec}(\Sigma_k)$ is the matrix Σ_k reshaped into vector form and formulate the stochastic belief dynamics

$$\mathbf{b}_{k+1} = g(\mathbf{b}_k, \mathbf{u}_k) + W(\mathbf{b}_k, \mathbf{u}_k)\xi_k, \quad \xi_k \sim \mathcal{N}(0, I) \tag{16}$$

with

$$g_k(\mathbf{b}_k, \mathbf{u}_k) = \begin{bmatrix} f(\hat{\mathbf{x}}_k, \mathbf{u}_k, 0) \\ \text{vec}(\Gamma_{k+1} - K_k H_k \Gamma_{k+1}) \end{bmatrix} \tag{17}$$

$$W_k(\mathbf{b}_k, \mathbf{u}_k) = \begin{bmatrix} \sqrt{K_k H_k \Gamma_{k+1}} \\ \mathbf{0} \end{bmatrix}. \tag{18}$$

Here, ξ_k is a Gaussian with dimension $n_{\mathbf{x}}$ that is applied to the stochastic part of $\mathbf{b}_k$, i.e., the stochastic state variable $\mathbf{x}_k$. In this form ξ_k represents both measurement noise $\mathbf{n}_k$ and motion noise $\mathbf{m}_k$ mapped onto the belief transition. The stochastic Gaussian belief dynamics allow us to propagate beliefs efficiently during the forward pass of the game-theoretic variant of belief-space iLQG.

B. Nash Equilibrium Necessary Condition

While formulating how to propagate uncertainty for the continous POMDP, we also need to define a tractable procedure to solve for Nash equilibria. One common method to solve for Nash equilibria is the method of iterated best response [9], [11], where control policies are exchanged after each agent's separate and independent optimization iteration. In contrast, we directly integrate the necessary condition of the Nash equilibrium into the backward pass of a belief space variant of iLQG. Specifically, we solve a quadratic game at each stage of the backward pass with a unique solution. First, we formulate the necessary condition of Problem 2 as

$$\frac{\partial Q^i(\mathbf{b}_0, \mathbf{u})}{\partial \mathbf{u}^i} = 0 \;\; \forall i \in \{1, 2, \ldots, N\} \tag{19}$$

which allows us to compute local Nash equilibria by solving (19). Theorem 1 states an equivalent condition for $Q_k^i(\mathbf{b}_k, \mathbf{u}_k)$, the value function from time k to l, defined in the Bellman recursion (8).

Theorem 1: The necessary condition of the local Nash equilibrium (19) is equivalent to

$$\frac{\partial Q_k^i(\mathbf{b}_k, \mathbf{u}_k)}{\partial \mathbf{u}_k^i} = 0 \tag{20}$$

for all $i \in \{1, 2, \ldots, N\}$, and $k \in \{0, 1, \ldots, l-1\}$.

Proof: Recall that the conventional POMDP formulation (6) in Problem 1 is equivalent to the recursive Bellman equation (8). Maximizing $Q_k^i(\mathbf{b}_k, \mathbf{u}_k)$ with respect to $\mathbf{u}_k^i$ in (8) yields the corresponding necessary optimality condition $\frac{\partial Q_k^i(\mathbf{b}_k, \mathbf{u}_k)}{\partial \mathbf{u}_k^i} = 0$, the same as (20). Therefore, the necessary optimality condition (19) of Problem 1 is equivalent to the recursive Bellman necessary optimality condition (20) in Theorem 1. ■

Alternatively, we can split the action-value from time 0 into the action-value from k and the cost accumulated until k

$$Q^i(\mathbf{b}_0, \mathbf{u}) = Q_k^i(\mathbf{b}_k, \mathbf{u}_k) + \mathop{\mathbb{E}}_{\mathbf{z}}\left[\sum_{t=0}^{k-1} c_t^i(\mathbf{b}_t, \mathbf{u}_t)\right]. \tag{21}$$

Taking the derivative of both sides with respect to $\mathbf{u}_k^i$ directly implies that $\frac{\partial Q_k^i(\mathbf{b}_k, \mathbf{u}_k)}{\partial \mathbf{u}_k^i} = \frac{\partial Q^i(\mathbf{b}_0, \mathbf{u})}{\partial \mathbf{u}_k^i}$, since the cost accumulated until k, the second term on the right hand side, does not depend on $\mathbf{u}_k^i$. Intuitively, current actions cannot affect costs accumulated in the past.

Concluding, if each agent i finds an optimizing policy π_k^i to the Bellman recursion, all $\frac{\partial Q_k^i(\mathbf{b}_k, \mathbf{u}_k)}{\partial \mathbf{u}_k^i} = 0$ necessary conditions are fulfilled at time k. Note that each agents' policy $\pi^i(\mathbf{u}^{\neg i})$ depends on the other agents' inputs $\mathbf{u}^{\neg i}$, where $\neg i$ indicates all other agents. Therefore, solving the Bellman recursion simultaneously for all agents defines a static game [32], but more importantly a game at each stage k of the backward-pass.

In the next section, we describe our solution for integrating the Nash equilibrium necessary condition at every time k into the backward pass of a belief-space variant of iLQG.

C. Iterative Dynamic Programming

In this section we describe our belief-space variant of iLQG for computing local Nash equilibria by solving the Bellman recursion defined in (8). We denote the nominal belief as $\bar{\mathbf{b}} = \mathbf{b} - \delta\mathbf{b}$, the nominal controls $\bar{\mathbf{u}} = \mathbf{u} - \delta\mathbf{u}$, and $\bar{\mathbf{s}} = \mathbf{s} - \delta\mathbf{s}$, with $\bar{\mathbf{s}} = [\bar{\mathbf{b}}^\top, \bar{\mathbf{u}}^\top]^\top$ and local perturbations $\delta\mathbf{u}$, $\delta\mathbf{b}$, $\delta\mathbf{s}$. At each iteration, the algorithm performs a backward pass and a forward pass on the current estimate of the belief $\bar{\mathbf{b}} = [\bar{\mathbf{b}}_0, \bar{\mathbf{b}}_1, \ldots, \bar{\mathbf{b}}_l]$ and control trajectory $\bar{\mathbf{u}} = [\bar{\mathbf{u}}_0, \bar{\mathbf{u}}_1, \ldots, \bar{\mathbf{u}}_{l-1}]$, i.e., the nominal trajectories. In the backward pass, the algorithm approximates the value functions for each agent as a quadratic function

$$V_k^i(\bar{\mathbf{b}}_k + \delta\mathbf{b}_k) \approx V_k^i + V_{\mathbf{b},k}^{i,\top}\delta\mathbf{b}_k + \frac{1}{2}\delta\mathbf{b}_{k+1}^\top V_{\mathbf{bb},k}^i \delta\mathbf{b}_k$$

along the nominal trajectory, and computes a linear feedback policy π^1 for the robot and predicted linear feedback policies $\pi^{\neg 1}$ for all other agents. The value function is propagated backward in time. In the forward pass we produce a new nominal trajectory based on the value function computed in the backward pass and apply the associated feedback policy. This iterative process continues toward a locally optimal solution to the Nash equilibrium in belief space. The key idea is to maintain a quadratic approximation of $Q_k^i(\mathbf{b}_k, \mathbf{u}_k)$ and the value functions $V_k^i(\mathbf{b}_k)$.

We first derive the quadratic form of $Q_k^i(\mathbf{b}_k, \mathbf{u}_k)$ in Theorem 2 by a Taylor expansion of the dynamics and costs, then find the minimizing control policy $\pi_k = [\pi_k^{1,\top}, \ldots, \pi_k^{N,\top}]^\top$ by solving the static game and computing the Nash equilibrium over all agents. From this result we compute the value functions $V_k^i(\mathbf{b}_k) = Q_k^i(\mathbf{b}_k, \pi_k)$ and derive an update law for V backward in time.

Theorem 2: By linear expansion of the belief dynamics and quadratic expansion of the cost and value function, $Q_k^i(\mathbf{s}_k)$ is a quadratic of the form

$$Q_k^i(\bar{\mathbf{s}}_k + \delta\mathbf{s}_k) \approx Q_k^i + Q_{\mathbf{s},k}^{i,\top}\delta\mathbf{s}_k + \frac{1}{2}\delta\mathbf{s}_k^\top Q_{\mathbf{ss},k}^i \delta\mathbf{s}_k \tag{22}$$

where

$$Q_k^i = c_k^i + V_{k+1}^i + \frac{1}{2}\sum_{j=1}^{n_x} W_k^{(j),\top} V_{\mathbf{bb},k+1}^i W_k^{(j)} \tag{23}$$

$$Q_{\mathbf{s},k}^i = c_{\mathbf{s},k}^i + g_{\mathbf{s},k}^\top V_{\mathbf{b},k+1}^i + \sum_{j=1}^{n_x} W_{\mathbf{s},k}^{(j),\top} V_{\mathbf{bb},k+1}^i W_k^{(j)} \tag{24}$$

$$Q_{\mathbf{ss},k}^i = c_{\mathbf{ss},k}^i + g_{\mathbf{s},k}^\top V_{\mathbf{bb},k+1}^i g_{\mathbf{s},k} + \sum_{j=1}^{n_x} W_{\mathbf{s},k}^{(j),\top} V_{\mathbf{bb},k+1}^i W_{\mathbf{s},k}^{(j)}. \tag{25}$$

Similar derivations in iLQG [36], and belief-space iLQG [28] showed an agent's action-value Q function to be quadratic with respect to the agent's controls and state or belief. In contrast, we show that agent i's action-value function Q^i is also a quadratic with respect to the *joint* state and controls which is critical to formulate the static quadratic game in the backward pass.

Proof: We start by expanding the terms of the action-value function of the Bellman recursion (8)

$$Q_k^i(\mathbf{b}_k, \mathbf{u}_k) = c_k^i(\mathbf{b}_k, \mathbf{u}_k) + \mathop{\mathbb{E}}_{\xi_k}\left[V_{k+1}^i(g_k(\mathbf{b}_k, \mathbf{u}_k) + W_k(\mathbf{b}_k, \mathbf{u}_k)\xi_k)\right] \tag{26}$$

to second order around the nominal control and belief $\bar{\mathbf{s}}_k = [\bar{\mathbf{b}}_k^\top, \bar{\mathbf{u}}_k^\top]^\top$. The term $c_k^i(\mathbf{b}_k, \mathbf{u}_k)$ becomes

$$c_k^i(\bar{\mathbf{s}}_k + \delta\mathbf{s}_k) \approx c_k^i + c_{\mathbf{s},k}^{i,\top}\delta\mathbf{s}_k + \frac{1}{2}\delta\mathbf{s}_k^\top c_{\mathbf{ss},k}^i \delta\mathbf{s}_k \tag{27}$$

with $c_k^i = c_k^i(\bar{\mathbf{s}})$, where $c_{\mathbf{s},k}^i$ and $c_{\mathbf{ss},k}^i$ are the Jacobian and Hessian of c_k^i evaluated at $\bar{\mathbf{s}}_k$. To expand the second term on the right hands side of (26) we first expand the stochastic joint belief dynamics to

$$g_k(\bar{\mathbf{s}}_k + \delta\mathbf{s}_k) \approx g_k + g_{\mathbf{s},k}\delta\mathbf{s}_k \tag{28}$$

$$W_k^{(j)}(\bar{\mathbf{s}}_k + \delta\mathbf{s}_k) \approx W_k^{(j)} + W_{\mathbf{s},k}^{(j)}\delta\mathbf{s}_k \tag{29}$$

with terms $g_k = g_k(\bar{\mathbf{s}}_k)$, $W_k^{(j)} = W_k^{(j)}(\bar{\mathbf{s}}_k)$, and $g_{\mathbf{s},k}$, $W_{\mathbf{s},k}^{(j)}$ the respective Jacobians evaluated at $\bar{\mathbf{s}}_k$. $W_k^{(j)}$ denotes the jth column of matrix W_k.

We now formulate the second term of (26). We define the value function as a quadratic around $\bar{\mathbf{b}}_{k+1}$

$$V_{k+1}^i(\bar{\mathbf{b}}_{k+1} + \delta\mathbf{b}_{k+1}) \tag{30}$$

$$\approx V_{k+1}^i + V_{\mathbf{b},k+1}^{i,\top}\delta\mathbf{b}_{k+1} + \frac{1}{2}\delta\mathbf{b}_{k+1}^\top V_{\mathbf{bb},k+1}^i \delta\mathbf{b}_{k+1}$$

$$= V_{k+1}^i + V_{\mathbf{b},k+1}^{i,\top}(\mathbf{b}_{k+1} - \bar{\mathbf{b}}_{k+1}) + \frac{1}{2}(\mathbf{b}_{k+1} - \bar{\mathbf{b}}_{k+1})^\top V_{\mathbf{bb},k+1}^i (\mathbf{b}_{k+1} - \bar{\mathbf{b}}_{k+1}) \tag{31}$$

with $\delta\mathbf{b}_{k+1} = \mathbf{b}_{k+1} - \bar{\mathbf{b}}_{k+1}$ for convenience. Inserting the expanded dynamics (28) and (29) into the second term of (26), defined by (31), and evaluating the expectation over ξ_k yields

$$\mathop{\mathbb{E}}_{\xi_k}\left[V_{k+1}^i(g_k(\mathbf{s}_k) + W_k(\mathbf{s}_k)\xi_k)\right] \tag{32}$$

$$\approx \mathop{\mathbb{E}}_{\xi_k}\Big[V_{k+1}^i + V_{\mathbf{b},k+1}^{i,\top}\left(g_k(\mathbf{s}_k) + W_k(\mathbf{s}_k)\xi_k - \bar{\mathbf{b}}_{k+1}\right) + \frac{1}{2}\left(g_k(\mathbf{s}_k) + W_k(\mathbf{s}_k)\xi_k - \bar{\mathbf{b}}_{k+1}\right)^\top V_{\mathbf{bb},k+1}^i \left(g_k(\mathbf{s}_k) + W_k(\mathbf{s}_k)\xi_k - \bar{\mathbf{b}}_{k+1}\right)\Big] \tag{33}$$

$$= V_{k+1}^i + V_{\mathbf{b},k+1}^{i,\top}\left(g_k(\mathbf{s}_k) - \bar{\mathbf{b}}_{k+1}\right) + \frac{1}{2}\left(g_k(\mathbf{s}_k) - \bar{\mathbf{b}}_{k+1}\right)^\top V_{\mathbf{bb},k+1}^i \left(g_k(\mathbf{s}_k) - \bar{\mathbf{b}}_{k+1}\right) + \frac{1}{2}\mathrm{tr}\left(W_k(\mathbf{s}_k)^\top V_{\mathbf{bb},k+1}^i W_k(\mathbf{s}_k)\right) \tag{34}$$

$$= V_{k+1}^i + V_{\mathbf{b},k+1}^{i,\top} g_{\mathbf{s},k}\delta\mathbf{s}_k + \frac{1}{2}\delta\mathbf{s}_k^\top g_{\mathbf{s},k}^\top V_{\mathbf{bb},k+1}^i g_{\mathbf{s},k}\delta\mathbf{s}_k + \frac{1}{2}\sum_{j=1}^{n_x}(W_k^{(j)} + W_{\mathbf{s},k}^{(j)}\delta\mathbf{s}_k)^\top V_{\mathbf{bb},k+1}^i (W_k^{(j)} + W_{\mathbf{s},k}^{(j)}\delta\mathbf{s}_k)^\top. \tag{35}$$

Here we use the value function expansion (31) in (33), and the fact that $\bar{\mathbf{b}}_{k+1} = g_k(\bar{\mathbf{s}}_k)$ in (34) in the form of

$$g_k(\mathbf{s}_k) - \bar{\mathbf{b}}_{k+1} = g_k(\mathbf{s}_k) - g_k(\bar{\mathbf{s}}_k) \approx g_{\mathbf{s},k}\delta\mathbf{s}_k. \tag{36}$$

Collecting and grouping all first and second-order terms of (35) and (27) we have that the resulting $Q_k^i(\bar{\mathbf{s}}_k + \delta\mathbf{s}_k)$ is a quadratic with coefficients given by (23)–(25). ■

For notational convenience we will drop the time index k for the Q matrices. We can also recover other partial derivatives of Q^i from (23)–(25)

$$Q_{\mathbf{s}}^i = \begin{bmatrix} Q_{\mathbf{b}}^i \\ Q_{\mathbf{u}^1}^i \\ \vdots \\ Q_{\mathbf{u}^N}^i \end{bmatrix}, \; Q_{\mathbf{ss}}^i = \begin{bmatrix} Q_{\mathbf{bb}}^i & Q_{\mathbf{bu}^1}^i & \cdots & Q_{\mathbf{bu}^N}^i \\ Q_{\mathbf{u}^1\mathbf{b}}^i & Q_{\mathbf{u}^1\mathbf{u}^1}^i & \cdots & Q_{\mathbf{u}^1\mathbf{u}^N}^i \\ \vdots & \vdots & \ddots & \vdots \\ Q_{\mathbf{u}^N\mathbf{b}}^i & Q_{\mathbf{u}^N\mathbf{u}^1}^i & \cdots & Q_{\mathbf{u}^N\mathbf{u}^N}^i \end{bmatrix}. \tag{37}$$

With $Q_k^i(\bar{\mathbf{s}}_k + \delta\mathbf{s}_k)$ in quadratic form from Theorem 2, at stage k each agent i solves the quadratic problem

$$\delta\mathbf{u}_k^{i,*} = \arg\min_{\delta\mathbf{u}_k^i} \; Q_{\mathbf{s},k}^{i,\top}\delta\mathbf{s}_k + \frac{1}{2}\delta\mathbf{s}_k^\top Q_{\mathbf{ss},k}^i \delta\mathbf{s}_k \tag{38}$$

yielding a quadratic game in the variables δu_k. Note that each agent's optimal $\delta\mathbf{u}_k^{i,*}$ depends on all other agents' $\delta\mathbf{u}_k^{\neg i}$ as they are contained in $\delta\mathbf{s}_k$. In comparison to other related methods such as iLQR [37], iLQG [36], or differential dynamic programming (DDP) [38] that solve a single quadratic optimization in the backward pass, we have to solve N interdependent quadratic optimizations. Nonetheless, we obtain a unique and simple solution to the quadratic game [32] by stacking the N optimality conditions of each interdependent optimization. Solving the resulting system of equations amounts to solving all interdependent quadratic optimizations at once. Theorem 3 presents this solution.

Theorem 3: The solution to the quadratic game (38) is

$$\delta\mathbf{u}_k^* = -\hat{Q}_{\mathbf{uu}}^{-1}\left(\hat{Q}_{\mathbf{u}} + \hat{Q}_{\mathbf{ub}}\delta\mathbf{b}_k\right) \tag{39}$$

where $\hat{Q}_{\mathbf{uu}}$, $\hat{Q}_{\mathbf{ub}}$, $\hat{Q}_{\mathbf{u}}$, are populated from (37), and defined

$$\hat{Q}_{\mathbf{uu}} = \begin{bmatrix} Q_{\mathbf{u}^1\mathbf{u}}^1 \\ Q_{\mathbf{u}^2\mathbf{u}}^2 \\ \vdots \\ Q_{\mathbf{u}^N\mathbf{u}}^N \end{bmatrix}, \; \hat{Q}_{\mathbf{ub}} = \begin{bmatrix} Q_{\mathbf{u}^1\mathbf{b}}^1 \\ Q_{\mathbf{u}^2\mathbf{b}}^2 \\ \vdots \\ Q_{\mathbf{u}^N\mathbf{b}}^N \end{bmatrix}, \; \hat{Q}_{\mathbf{u}} = \begin{bmatrix} Q_{\mathbf{u}^1}^1 \\ Q_{\mathbf{u}^2}^2 \\ \vdots \\ Q_{\mathbf{u}^N}^N \end{bmatrix}. \tag{40}$$

Proof: By taking the derivative of the objective of (38) and equating it to zero, the stationarity condition of (38) yields

$$\begin{bmatrix} Q_{\mathbf{u}^i\mathbf{u}^i}^i & Q_{\mathbf{u}^i\mathbf{u}^{\neg i}}^i \end{bmatrix}\begin{bmatrix} \delta\mathbf{u}_k^i \\ \delta\mathbf{u}_k^{\neg i} \end{bmatrix} + Q_{\mathbf{u}^i\mathbf{b}}^i\delta\mathbf{b}_k + Q_{\mathbf{u}^i}^i = 0. \tag{41}$$

Stacking the stationarity conditions of all N agents into a single system of equations we find the joint stationarity condition for all interdependent quadratic optimizations

$$\hat{Q}_{\mathbf{uu}}\delta\mathbf{u}_k + \hat{Q}_{\mathbf{ub}}\delta\mathbf{b}_k + \hat{Q}_{\mathbf{u}} = 0 \tag{42}$$

where (39) is the solution to this system of equations. ■

The local necessary condition of Problem 2, derived in Section III-B holds as shown below.

Corollary 1: The solution (39) fulfills the necessary condition of the local Nash equilibrium (19) at time k.

Proof: From (41), we see that $\frac{\partial Q_k^i(\mathbf{b}_k, \mathbf{u}_k)}{\partial \mathbf{u}_k^i} = 0$. ■

We can immediately derive the linear feedback policy for all agents at planning time k of the form

$$\pi_k = \bar{\mathbf{u}}_k + j_k + K_k\delta\mathbf{b}_k \tag{43}$$

with $j_k = -\hat{Q}_{\mathbf{uu}}^{-1}\hat{Q}_{\mathbf{u}}$ the feed forward term and $K_k = -\hat{Q}_{\mathbf{uu}}^{-1}\hat{Q}_{\mathbf{ub}}$ the feedback term. Note that π_k contains the optimal policy of the robot π_k^1 and also the predicted policies for all other $(N-1)$ agents $\pi_k^{\neg 1}$. The interdependence has been resolved by solving (42). The predicted linear policies $\pi_k^{\neg 1}$ depend on the change in *joint* belief $\delta\mathbf{b}_k$. The predicted actions will adapt if the robot, the other agents, or the environment behave differently as expected, causing the estimated belief $\mathbf{b}_k$ at future times k

to diverge from the predicted nominal belief $\bar{\mathbf{b}}_k$. Similarly, the robot's linear policy π_k^1 will allow it to adapt if other agents deviate from predicted behavior. In contrast, this flexibility would be impossible with a static optimal control trajectory instead of a policy.

We now formulate the backward equations to propagate the value functions V^i backward, hence defining the backward pass.

Corollary 2: The discrete backward differential equations of the value functions V^i are

$$V_k^i = Q^i + Q_{\mathbf{u}}^{i,\top} j_k + \frac{1}{2} j_k^\top Q_{\mathbf{uu}}^i j_k \tag{44}$$

$$V_{\mathbf{b},k}^i = Q_{\mathbf{b}}^i + K_k^\top Q_{\mathbf{uu}}^i j_k + K_k^\top Q_{\mathbf{u}}^i + Q_{\mathbf{ub}}^{i,\top} j_k \tag{45}$$

$$V_{\mathbf{bb},k}^i = Q_{\mathbf{bb}}^i + K_k^\top Q_{\mathbf{uu}}^i K_k + K_k^\top Q_{\mathbf{ub}}^i + Q_{\mathbf{ub}}^{i,\top} K_k \tag{46}$$

with terminal constraints

$$V_l^i = c_l^i(\bar{\mathbf{b}}_l), \quad V_{\mathbf{b},l}^i = \left.\frac{\partial c_l^i(\mathbf{b})}{\partial \mathbf{b}}\right|_{\mathbf{b}=\bar{\mathbf{b}}_l}, \quad V_{\mathbf{bb},l}^i = \left.\frac{\partial^2 c_l^i(\mathbf{b})}{\partial \mathbf{b}^2}\right|_{\mathbf{b}=\bar{\mathbf{b}}_l}. \tag{47}$$

Proof: Substituting the solution (39) and (43) back into the quadratic (22) yields the value function $V_k^i(\bar{\mathbf{b}}_k + \delta\mathbf{b}_k)$.

$$\begin{aligned} V_k^i(\bar{\mathbf{b}}_k + \delta\mathbf{b}_k) &= Q_k^i(\bar{\mathbf{b}}_k + \delta\mathbf{b}_k, \pi_k) \\ &= Q^i + Q_{\mathbf{u}}^{i,\top}(j_k + K_k\delta\mathbf{b}_k) + Q_{\mathbf{b}}^{i,\top}\delta\mathbf{b}_k \\ &\quad + \frac{1}{2}(j_k + K_k\delta\mathbf{b}_k)^\top Q_{\mathbf{uu}}^i(j_k + K_k\delta\mathbf{b}_k) + \frac{1}{2}\delta\mathbf{b}_k^\top Q_{\mathbf{bb}}^i\delta\mathbf{b}_k \\ &\quad + \frac{1}{2}(j_k + K_k\delta\mathbf{b}_k)^\top Q_{\mathbf{ub}}^i\delta\mathbf{b}_k + \frac{1}{2}\delta\mathbf{b}_k^\top Q_{\mathbf{bu}}^i(j_k + K_k\delta\mathbf{b}_k). \end{aligned}$$

Collecting first and second-order terms in $\delta\mathbf{b}_k$ gives (44)–(46) in the form of (31). The terminal constraints (47) result from a Taylor expansion of the final cost c_l^i around the final nominal belief $\bar{\mathbf{b}}_l$. ■

Based on results of Theorem 3 and Corollary 2 we can propagate the quadratic value functions backward in time starting from the terminal constraints at time l.

D. Regularization

With any Newton-like method, care must be taken when the Hessian $\hat{Q}_{\mathbf{uu}}$ is not positive-definite or when the minimum is not close and the quadratic model inaccurate. To ensure that the algorithm converges regardless of initial conditions, we implement a Levenberg–Marquardt style regularization [39].

1) Control Regularization: The control regularization is achieved by adding a diagonal term of magnitude $\mu_{\mathbf{u}}$ to the diagonal of $\hat{Q}_{\mathbf{uu}}$, yielding

$$\tilde{Q}_{\mathbf{uu}}^i = \hat{Q}_{\mathbf{uu}}^i + \mu_{\mathbf{u}} I. \tag{48}$$

This simple Levenberg–Marquardt style modification results in adding a quadratic cost around the current control sequence, which forces the new optimal control inputs computed by the backward pass to stay closer to the previous iteration.

2) Belief Regularization: The drawback of the control-based regularization scheme is that even small control perturbations can cause large deviations in the state trajectory potentially inhibiting convergence. To ensure that the updated belief trajectory does not deviate too far from the previous iteration, we introduce a scheme that penalizes deviations from beliefs rather than controls with parameter $\mu_{\mathbf{b}}$

$$\begin{aligned} \tilde{Q}_{\mathbf{ss},k}^i = c_{\mathbf{ss},k}^i &+ g_{\mathbf{s},k}^T \left(V_{\mathbf{bb},k+1}^i + \mu_{\mathbf{b}} I\right) g_{\mathbf{s},k} \\ &+ \sum_{i=1}^{n} W_{\mathbf{s},k}^{(j),T} \left(V_{\mathbf{bb},k+1}^i + \mu_{\mathbf{b}} I\right) W_{\mathbf{s},k}^{(j)}. \end{aligned} \tag{49}$$

The belief-based regularization results in placing a quadratic belief-cost around the previous belief trajectory, similarly to [40], where a state-based regularization was employed. In contrast to the standard control-based regularization, the feedback gains K_k do not go to zero as $\mu_{\mathbf{b}} \to \infty$, but rather force the new trajectory closer to the old one. In practice, we find this to improve the robustness of convergence.

E. Algorithm for Dynamic Game Belief Space Planning

We summarize our findings of solving Nash equilibria of dynamic games in belief space in Algorithm 1. Theorem 2 lays the foundation for the quadratic game solved in the backward pass of Algorithm 1. The solution to the quadratic game presented in Theorem 3 yields a linear feedback policy π_k for all agents. We propagate the value function in the backward pass according to Corollary 2 starting with the terminal conditions from the terminal cost.

Algorithm 1 starts from the current belief estimate $\mathbf{b}_0$, in our experiments provided from an EKF, and an initial control trajectory guess. We found initializing controls to all zeros to work well in practice. We update the nominal control and belief trajectories in the forward pass based on rolling out the belief dynamics model and applying the updated feedback policy π_k. If all agents' action-value functions improved, we accept the updated nominal belief and control trajectories and reduce regularization. Otherwise, we reject the trajectories and increase regularization. The iteration of backward and forward pass continues until each agents' action value function Q^i has converged and changes less than a specified threshold ϵ.

The algorithm yields a linear feedback policy π^1 and a predicted belief trajectory $\mathbf{b}^1$ of the robot. It also gives predicted feedback policies $\pi^{\neg 1}$ and predicted belief trajectories $\mathbf{b}^{\neg 1}$ for all other agents over the full time horizon.

F. Runtime Analysis

The dominant runtime complexity in a single backward step is $\mathcal{O}(N^7 n_{\mathbf{x}}^{i,6})$. A full iteration of Algorithm 1 solves l quadratic games leading the final runtime complexity to $\mathcal{O}(lN^7 n_{\mathbf{x}}^{i,6})$. Scaling linearly in the planning horizon l enables real-time deployment whereas other POMDP algorithms scale exponentially, even without taking any game dynamics into account. The following provides a brief summary of our runtime analysis.

We analyze the runtime by first recalling that the dimension of the joint state is $\mathcal{O}(n_{\mathbf{x}})$, and assume for the sake of analysis that the agents' state dimensions are equal, such that $\mathcal{O}(n_{\mathbf{x}}) = \mathcal{O}(Nn_{\mathbf{x}}^i)$. To simplify the analysis, we also assume the joint input $(n_{\mathbf{u}})$ and the joint measurement dimensions $(n_{\mathbf{z}})$ to be $\mathcal{O}(n_{\mathbf{x}})$.

Algorithm 1: Nash Equillibrium of Dynamic Games in Belief Space.

Input: Initial belief $\mathbf{b}_0$, control $\bar{\mathbf{u}}$, models c_k^i, c_l^i, f, h
Output: Predicted trajectories $\bar{\mathbf{b}}$, $\bar{\mathbf{u}}$, feedback law π

$\bar{\mathbf{b}} \leftarrow$ Propagate $\mathbf{b}_0$ with g and $\bar{\mathbf{u}}$
while $|Q^i(\bar{\mathbf{b}}_{\text{new}}, \bar{\mathbf{u}}_{\text{new}}) - Q^i(\bar{\mathbf{b}}, \bar{\mathbf{u}})| > \epsilon$ **do**
  Backward pass:
  $V^i_{\mathbf{b},l}, V^i_{\mathbf{bb},l} \leftarrow$ From terminal boundary conditions (47)
  for k from $l-1$ to 0 **do**
    $\pi^i_k, j^i_k, K^i_k \leftarrow$ Solve quadratic game (43)
    $V^i_{\mathbf{b},k}, V^i_{\mathbf{bb},k} \leftarrow$ Propagate value function (45), (46)
  end for
  Forward pass:
  $\bar{\mathbf{b}}_{\text{new}}, \bar{\mathbf{u}}_{\text{new}} \leftarrow$ Propagate $\mathbf{b}_0$ with g and π
  if $Q^i(\bar{\mathbf{b}}_{\text{new}}, \bar{\mathbf{u}}_{\text{new}}) \leq Q^i(\bar{\mathbf{b}}, \bar{\mathbf{u}})$ **then**
    $\bar{\mathbf{b}}, \bar{\mathbf{u}} \leftarrow \bar{\mathbf{b}}_{\text{new}}, \bar{\mathbf{u}}_{\text{new}}$,
    lower regularization (48), (49)
  else increase regularization
  end if
end while

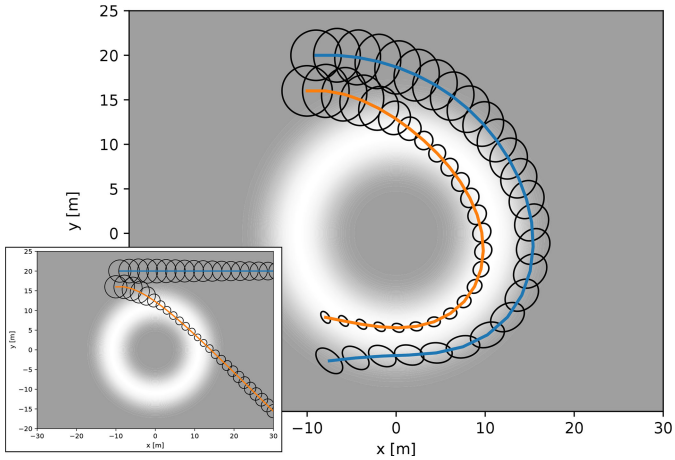

Fig. 2. By nudging Agent 2 onto the circular light source Agent 1 is able to reduce the uncertainty over Agent 1's state at the end of the planning horizon. The lower left inset shows the same scenario without any information gain. As a result, Agent 1 has no incentive to manipulate Agent 2's behavior since there is no way to influence its uncertainty. Both agents start with positive velocity in the x-direction.

The covariance matrix of the joint state contains $n_{\mathbf{x}}^2/2$ unique elements. Since the joint belief $\mathbf{b}$ contains the covariance of the state in addition to the state itself, it entails $\mathcal{O}(n_{\mathbf{x}}^2)$ elements.

Now consider the matrix multiplicative terms in the iterative dynamic programming procedure. A computational bottleneck occurs when updating the action-value function $Q^i_{\mathbf{ss}}$ in (25). Evaluating the product $g_{\mathbf{s}}^\top V^i_{\mathbf{bb}} g_{\mathbf{s}}$ requires the multiplication of matrices with dimensions $\mathcal{O}(n_{\mathbf{x}}^2) \times \mathcal{O}(n_{\mathbf{x}}^2)$, resulting in $\mathcal{O}(n_{\mathbf{x}}^6) = \mathcal{O}(N^6 n_{\mathbf{x}}^{i,6})$ complexity. This operation has to be completed for each of the N agents, such that the complexity increases to $\mathcal{O}(N^7 n_{\mathbf{x}}^{i,6})$. The term $W_{\mathbf{s}}^{(j),\top} V^i_{\mathbf{bb}} W_{\mathbf{s}}^{(j)}$ in (25) must be computed $n_{\mathbf{x}}$ times, but can be evaluated in $\mathcal{O}(n_{\mathbf{x}}^5)$, since W only contains $n_{\mathbf{x}}$ nonzero elements. See (18) for the definition of W. We solve the quadratic game at each stage by finding the inverse of the $n_{\mathbf{u}} \times n_{\mathbf{u}}$ matrix $\hat{Q}_{\mathbf{uu}}$ in (40), which has complexity $\mathcal{O}(n_{\mathbf{x}}^3)$. Therefore, $\mathcal{O}(N^7 n_{\mathbf{x}}^{i,6})$ remains the dominant runtime complexity.

Next, we investigate the complexity of evaluating derivatives, Hessians, and Jacobians. The cost Hessian $c^i_{\mathbf{ss}}$ only contains $\mathcal{O}(n_{\mathbf{x}}^4)$ elements. Automatic differentiation through source code transformation yields $\mathcal{O}(1)$ complexity for each element, such that the cost Hessian term has no significant impact on the overall runtime complexity. The EKF belief dynamics can be evaluated in $\mathcal{O}(n_{\mathbf{x}}^3)$, such that linearizing the belief dynamics to obtain $W_{\mathbf{s}}$ and $g_{\mathbf{s}}$, both with $\mathcal{O}(n_{\mathbf{x}}^2)$ entries, results in $\mathcal{O}(n_{\mathbf{x}}^5)$.

Thus, we find the dominant runtime complexity in a single backward step is $\mathcal{O}(N^7 n_{\mathbf{x}}^{i,6})$, and a final runtime complexity of $\mathcal{O}(lN^7 n_{\mathbf{x}}^{i,6})$ in a full iteration of Algorithm 1.

IV. Case Studies

We demonstrate the performance and flexibility of our algorithm in three case studies that combine the information-seeking behavior with our game-theoretic formulation. These case studies examine how the agents interact in the game, gain information, and use the information gain to improve their control policies. We choose these illustrative examples due to their variations in agent interactions and demonstration of broader capabilities. Each of the case studies employs a different dynamics and observation model as well as distinct objectives for the agents. We find the Nash equilibrium to each of these games through Algorithm 1.

A. Active Surveillance

In this case study, Agents 1 and 2 are in an environment with variable lighting conditions. Agent 1 is tasked with observing Agent 2, but the quality of the observations depends on the available lighting at the location in the environment. Agent 2 has no goal but is assigned the objective of maintaining a constant velocity while avoiding Agent 1. In the provided examples, the agents do not directly exchange information. Instead, they perceive themselves and the other agent only through observations. At the time of initialization, neither agent has perfect information of the other but only a noisy state estimate defining the initial belief $\mathbf{b}_0$. Using our approach, we show that Agent 1 can successfully herd Agent 2 into the lighted region to achieve its surveillance objective, which would not be possible without incorporating the BSP into the dynamic game. Figs. 1 and 2 show the planned trajectories in two environments. Our case study goes beyond the commonly studied multirobot herding problem [41]–[44] which has the goal of herding agents into a specified location. In contrast, our goal is to reduce uncertainty in the final state of another agent, which happens to coincide with pushing the other agent into the light. Algorithms commonly applied to the herding problem are not applicable here as they do not reason about the uncertainty of other agents.

The state of both car-like robots $\mathbf{x}^{(i)} = [x^{(i)}, y^{(i)}, \theta^{(i)}, v^{(i)}]$ consists of their position (x, y), orientation θ, and speed v. The control inputs $\mathbf{u}^{(i)} = [u^{(i)}_{\text{acc},k}, u^{(i)}_{\text{steer},k}]$ are acceleration $u_{\text{acc},k}$

and steering wheel angle $u_{\text{steer},k}$. The deterministic continuous dynamics of both agents are given by

$$\dot{\mathbf{x}}_k^{(i)} = \left[v_k^{(i)} \cos\theta_k^{(i)}, \tilde{} v_k^{(i)} \sin\theta_k^{(i)}, u_{\text{acc},k}^{(i)}, \frac{v_k^{(i)}}{L\tan(u_{\text{steer},k}^{(i)})} \right]^\top$$

where L is the length of the robots. The discrete time dynamics are defined by

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{m}_k) = \mathbf{x}_k + \dot{\mathbf{x}}_k \tau + M(\mathbf{u}_k) \cdot \mathbf{m}_k$$

with timestep τ. $M(\mathbf{u}_k)$ scales the motion noise $\mathbf{m}_k$ proportional to the control input $\mathbf{u}_k$, such that uncertainty increases if excessive controls are executed. We encode the agent's objective and goals in this game by defining the current and terminal costs for Agent 1 and Agent 2 as

$$\begin{aligned} c_k^{(1)}(\mathbf{b}_k, \mathbf{u}_k) &= \mathbf{u}_k^{(1),\top} R \mathbf{u}_k^{(1)} \\ c_l^{(1)}(\mathbf{b}_l) &= \det(\Sigma_{x,y,l}^{(2)}) \\ c_k^{(2)}(\mathbf{b}_k, \mathbf{u}_k) &= \mathbf{u}_k^{(2),\top} R \mathbf{u}_k^{(2)} + a_1 (v_k^{(2)} - v_{k,\text{des}}^{(2)})^2 + a_2 c_{\text{coll}}(\mathbf{x}_k) \\ c_l^{(2)}(\mathbf{b}_l) &= a_1 (v_l^{(2)} - v_{l,\text{des}}^{(2)})^2 + a_2 c_{\text{coll}}(\mathbf{x}_l). \end{aligned}$$

Agent 1's overall objective is to lower the uncertainty about the position of Agent 2 at the end of the planning horizon, encoded by $c_l^{(1)}(\mathbf{b}_l)$. The term $\det(\Sigma_{x,y}^{(2)})$ is equivalent to the area of the 1σ-threshold ellipse of Agent 2 and representative of the location uncertainty of Agent 2 at the end of the planning horizon. Note that both agents penalize control effort by $\mathbf{u}_k^{(i),\top} R \mathbf{u}_k^{(i)}$, and Agent 2 has additional objectives for maintaining a desired velocity v_{des} and avoiding collisions via an exponential barrier $c_{\text{coll}}(\mathbf{x}_k) = \exp(-d(\mathbf{x}_k))$. Here $d(\mathbf{x}_k)$ is the expected Euclidean distance until collision between the two agents, taking their outline into account.

We restrict the robots' sensing abilities to only include noisy position measurements. The observation model varies across the environment based on the available light at a particular location

$$\mathbf{z}_k^{(i)} = h(\mathbf{x}_k^{(i)}, \mathbf{n}_k^{(i)}) = [x_k^{(i)}, y_k^{(i)}]^T + N(\mathbf{x}_k^{(i)}) \cdot \mathbf{n}_k^{(i)}$$

where the matrix $N(\mathbf{x}_k^{(i)})$ scales the measurement noise based on the current position (x, y) in the map. We show the nominal trajectories and the associated beliefs of the solution computed using Algorithm 1 in Figs. 1 and 2. In both cases Agent 1 (blue) is able to force Agent 2 into the light to successfully reduce uncertainty. The emergent behavior would not have been possible without BSP, reasoning about another agent's uncertainty, and without the dynamic game, estimating how their own actions influence another agent's actions. We show the resulting behavior without BSP and without any reasoning about Agent 2's uncertainty in the inset of Fig. 2.

B. Guide Dog for Blind Agent

In this scenario, Agent 2 guides Agent 1 toward a goal location while choosing a path that reduces the uncertainty in Agent 1's position. The game is won if Agent 1 knows it reaches the goal location with a low uncertainty about its state. However, Agent 1 does not have the ability to navigate itself. We refer to Agent 1 as the "blind" agent. Following this analogy, Agent 2 acts as the "guide dog" for the blind agent. The guide dog can gather information about its own state and the blind agent's state by passing through light sources in the environment which reduces uncertainty. The agents are tethered together, which we model with spring dynamics and refer to the tether as the "leash." If the guide takes the blind agent on the direct path to the goal, the guide would not have sufficient information to know it brought the blind agent to the goal location. Under our approach, the guide dog detours to key areas to reduce the blind agent's uncertainty. We use the analogy of a guide dog leading a blind agent to create an intuitive visual for the reader, however, this system is relevant to many other robotic applications.

We model the system dynamics as two masses on a surface with friction connected by a spring tether. The states of the blind agent and the guide dog are $\mathbf{x}^{(i)} = [\mathbf{r}^{(i)}, \mathbf{v}^{(i)}]$ the 2D position $\mathbf{r}$ and velocities $\mathbf{v}$. The inputs $\mathbf{u}^{(i)} = F^{(i)}$ are their respective force vectors. The blind agent and guide dog have masses $c_{\text{mass,h}}$ and $c_{\text{mass,d}}$, respectively, and are bound to friction coefficients $c_{\text{fric,h}}$ and $c_{\text{fric,d}}$. The accelerations are

$$\begin{aligned} \mathbf{a}^{(1)} &= 1/c_{\text{mass,h}} (\mathbf{u}^{(1)} - f_{\text{spring}}(\Delta\mathbf{r}) - c_{\text{fric,h}} \mathbf{v}^{(1)}) \\ \mathbf{a}^{(2)} &= 1/c_{\text{mass,d}} (\mathbf{u}^{(2)} + f_{\text{spring}}(\Delta\mathbf{r}) - c_{\text{fric,d}} \mathbf{v}^{(2)}) \end{aligned}$$

and influenced by the spring force

$$f_{\text{spring}}(\Delta r) = \frac{\Delta\mathbf{r}}{||\Delta\mathbf{r}||} c_{\text{spring}} \max(||\Delta\mathbf{r}|| - c_{\text{leash}}, 0)$$

which is dependent on the distance vector $\Delta(\mathbf{r}) = [\mathbf{r}^{(1)} - \mathbf{r}^{(2)}]$. The dog's leash is flexible with spring constant c_{spring} and has length c_{leash}, such that it only generates a spring force if extended beyond c_{leash} and is otherwise slack. The deterministic continuous dynamics are $\dot{\mathbf{x}}^{(i)} = [\mathbf{v}^{(i),\top}, \mathbf{a}^{(i),\top}]^\top$, and the discrete time dynamics

$$f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{m}_k) = \mathbf{x}_k + \dot{\mathbf{x}}_k \tau + M(\mathbf{u}_k) \cdot \mathbf{m}_k$$

for timestep τ and where $M(\mathbf{u}_k)$ scales the motion noise proportional to the inputs $\mathbf{u}_k$.

We use the cost functions to encode the behaviors and objects of each agent. Similar to the previous case study, minimizing $\det(\Sigma_{\mathbf{r},l}^{(1)})$ reduces the uncertainty at the end of the planning horizon. We define

$$\begin{aligned} c_k^{(1)}(\mathbf{b}_k, \mathbf{u}_k) &= \mathbf{u}_k^{(1),\top} R \mathbf{u}_k^{(1)} + c_{\text{acc,h}} \mathbf{a}_k^{(1),\top} \mathbf{a}_k^{(1)} \\ c_l^{(1)}(\mathbf{b}_l) &= 0 \\ c_k^{(2)}(\mathbf{b}_k, \mathbf{u}_k) &= \mathbf{u}_k^{(2),\top} R \mathbf{u}_k^{(2)} \\ c_l^{(2)}(\mathbf{b}_l) &= \det(\Sigma_{\mathbf{r},l}^{(1)}) + ||\mathbf{r}_l^{(1)} - \mathbf{r}_{\text{goal}}||^2. \end{aligned}$$

Here, the term $||\mathbf{r}_l^{(1)} - \mathbf{r}_{\text{goal}}||^2$ drives the guide dog to relocate the blind agent to the goal. We reduce the control efforts of each agent by $\mathbf{u}_k^{(i),\top} R \mathbf{u}_k^{(i)}$, and the blind agent has the additional objective of reducing accelerations with $c_{\text{acc,h}} \mathbf{a}_k^{(1),\top} \mathbf{a}_k^{(1)}$. We use a noisy observation model

$$\mathbf{z}_k^{(i)} = h(\mathbf{x}_k^{(i)}, \mathbf{n}_k^{(i)}) = \mathbf{x}_k^{(i)} + N(\mathbf{x}_k^{(i)}) \cdot \mathbf{n}_k^{(i)} \tag{50}$$

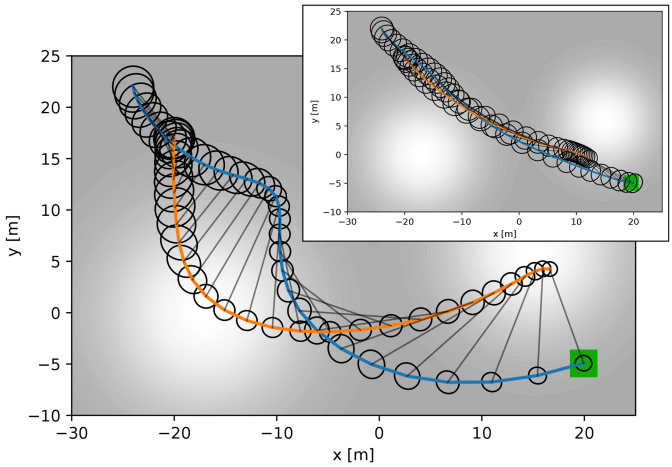

Fig. 3. Guide dog (orange) with leash (black line) guides the blind agent (blue) toward the goal location (green). While doing so it passes by both light sources to reduce the uncertainty of the blind person's position at the goal location. The top right inset shows the case, where the guide dog is indifferent about the blind person's uncertainty.

where the matrix $N(\mathbf{x}_k^{(i)})$ scales the measurement noise based on the environment shown in Fig. 3.

The resulting behavior is shown in Fig. 3: The guide dog (orange) guides the blind agent (blue) from its initial position to the blind person's goal location (green) while reducing the uncertainty of the blind agent's final state by planning a slight detour through the light sources instead of directly toward the goal location. The guide does so while also taking the complex interaction originating from the blind person's forces on the tether into account. The inset on Fig. 3 is the path taken by the dog with no optimization over the blind agent's uncertainty. While it takes a direct path to the goal, the final uncertainty of the blind agent is large.

C. Autonomous Racing

Finally, we demonstrate our approach in competitive racing, a common problem in dynamic games. By incorporating BSP into the dynamic game formulation, we show a significant increase in racing performance. This allows the agents to reduce uncertainty and decrease chance constraints. Thus, maneuvers like overtaking on tight road segments become possible.

In all racing runs each agent maintains a separate instance of Algorithm 1. This means that each agent separately computes their own optimal control actions, the predictions of other respective agents, and their own Nash equilibrium. No other additional information, such as state estimates, beliefs, policies, or initializations are shared among agents. Since each agent executes a separate instance of Algorithm 1, Assumption 2 may not be accurate, i.e., the belief computed by agent j over agent i may only inaccurately resemble the belief of agent i over itself. Nonetheless, we will show that despite a first-order belief assumption, the presented approach yields superior performance to all other baselines.

Each agent's state $\mathbf{x}^{(i)} = [x^{(i)}, y^{(i)}, \theta^{(i)}, v^{(i)}]$ and controls $\mathbf{u}^{(i)} = [u^{(i)}_{\text{acc},k}, u^{(i)}_{\text{steer},k}]$ are the same as in the active surveillance

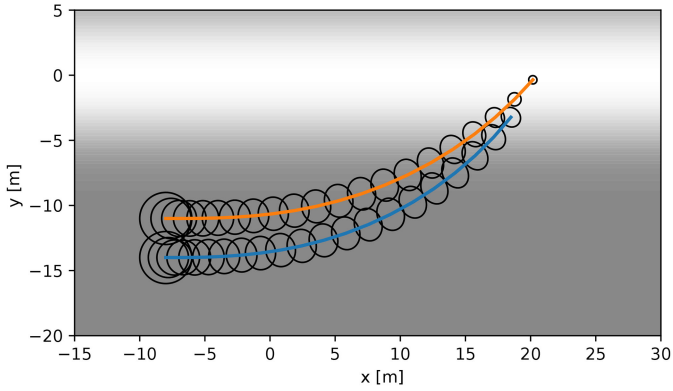

Fig. 4. Agent 1 (blue) is pushing Agent 2 (orange) into the light to reduce the uncertainty over Agent 2 at the end of the planning horizon. Uncertainties are visualized by covariance ellipses. Both agents are initialized with positive velocity in the x-direction.

experiment but the different deterministic continuous dynamics are of the from

$$\dot{\mathbf{x}}_k^{(i)} = \Big[v_k^{(i)} \cos(\theta_k^{(i)}), v_k^{(i)} \sin(\theta_k^{(i)}) u^{(i)}_{\text{acc},k} \\ -c_{\text{drag,i}} v_k^{(i)} - c_{\text{slip,i}} (\dot{\theta}^{(i)})^2, \dot{\theta}^{(i)} \Big]^\top$$

with yaw rate $\dot{\theta}^{(i)} = v_k^{(i)} / L \tan(u^{(i)}_{\text{steer},k})$, and drag- $c_{\text{drag,i}}$ and slip coefficient $c_{\text{slip,i}}$. The stochastic discrete time dynamics

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, \mathbf{m}_k) = \mathbf{x}_k + \dot{\mathbf{x}}_k \tau + M(\mathbf{b}_k, \mathbf{u}_k) \cdot \mathbf{m}_k$$

are subject to noise scaled by $M(\mathbf{b}_k, \mathbf{u}_k)$ proportional to the control input $\mathbf{u}_k$ as well as the squared yaw rate $(\dot{\theta}^{(i)})^2$ of each agent i separately. The observation model

$$\mathbf{z}_k^{(i)} = h(\mathbf{x}_k^{(i)}, \mathbf{n}_k^{(i)}) = \mathbf{x}_k^{(i)} + N(\mathbf{x}_k^{(i)}) \cdot \mathbf{n}_k^{(i)}$$

is subject to noise scaled by $N(\mathbf{x}_k^{(i)})$, depending on the position on the race track map. As shown in Fig. 4, we indicate zones of low measurement noise as red. It may be beneficial for agents to plan to drive through these low-measurement noise regions to increase information gain and to reduce uncertainty.

Each agent's goal is to maximize progress along the race track while staying on the track and not colliding with other agents. We define the progress along the track for any point $\mathbf{p} = (x, y)$ as the arc-length progress $r(\mathbf{p})$ of the closest point on the centerline. Likewise, we define $d(\mathbf{p})$ as the distance of the closest point on the track to $\mathbf{p}$. We visualize both the distance transform as well as the progress transform of the race track shown in Figs. 4 and 5. For competitive racing, each agent tries to maximize the relative progress over other agents $r(\mathbf{p}^{(i)}) - r(\mathbf{p}^{(\neg i)})$. Consequently, agents will engage in competitive blocking and cutting behavior. We design the current and terminal costs of each agent as

$$c_k^{(i)}(\mathbf{b}_k, \mathbf{u}_k) = \mathbf{u}_k^{(i),\top} R \mathbf{u}_k^{(i)} + c^{(i)}_{\text{track}}(\mathbf{b}_k) + c^{(i)}_{\text{coll}}(\mathbf{b}_k)$$

$$c_l^{(i)}(\mathbf{b}_l) = -r(p_l^{(i)}) + r(p_l^{(\neg i)})$$

penalizing control effort by R, while $c^{(i)}_{\text{track}}(\mathbf{b}_k)$ and $c^{(i)}_{\text{coll}}(\mathbf{b}_k)$ keep the agent on the track and out of collision. We achieve this by finding the upper bound of the 2σ positional uncertainty $\Sigma^{(i)}_{x,y}$

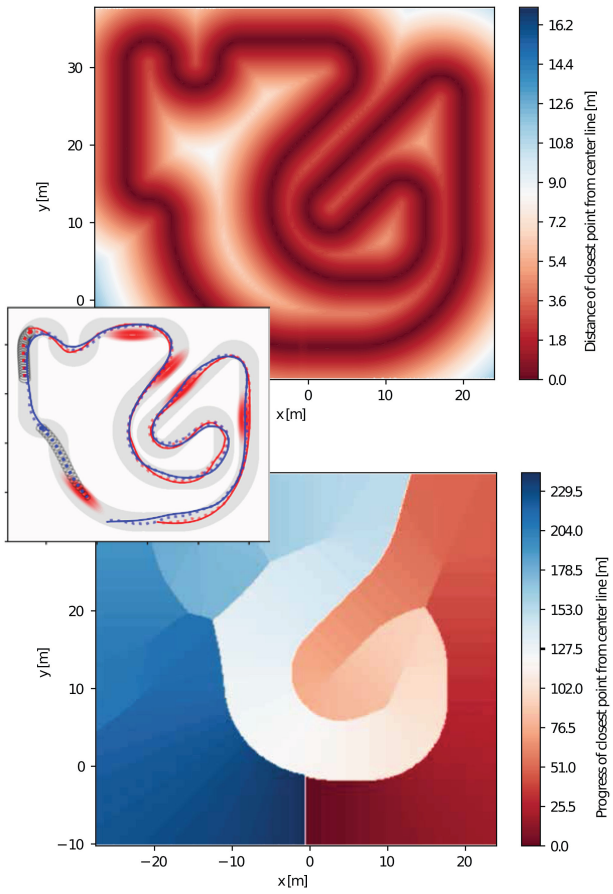

Fig. 5. *Top: (Distance Transform)* Map of the distances to the closest point on the center line $d(\mathbf{p})$ of the race track shown in Fig. 4. *Bottom: (Progress Transform)* Map of the progress $r(\mathbf{p})$ along the race track of the closest point on the center line.

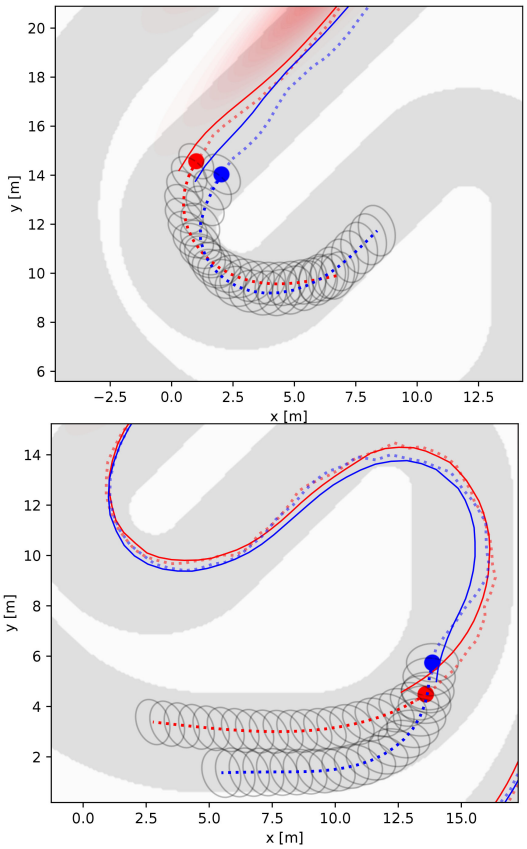

Fig. 6. *Top:* Blue agent cuts in front of the red agent, forcing the red agent to break. As a result, the blue agent can remain in front of the red agent at the end of the turn. *Bottom:* The red agent blocks the blue agent's overtaking maneuver forcing the blue agent to stay behind and take a wider line in the upcoming right turn. Significant amount of noise is simulated visualized by the deviation of the true trajectory (solid lines) and the predicted mean of the belief (dashed lines).

as $\alpha = 2\sqrt{\max(\text{eig}(\Sigma_{x,y}^{(i)}))}$. We can then formulate a chance collision constraint with other agents (limiting $||\mathbf{p}^{(i)} - \mathbf{p}^{(j)}||$) and the boundary of the race track (limiting $d(\mathbf{p})$) by restricting positions in the α vicinity. Finally, to arrive at $c_{\text{track}}^{(i)}(\mathbf{b}_k)$ and $c_{\text{coll}}^{(i)}(\mathbf{b}_k)$ we convert the constraints to soft constraints, penalizing constraint violation exponentially strong, as suggested in [28]. Additionally, we also limit control inputs $\mathbf{u}_k$ by soft constraints.

1) Competitive Racing: In our racing simulation, each car executes the current commanded control computed by their own separate instance of Algorithm 1. The environment's dynamics are propagated forward subject to significant amount of noise. Subsequently, a noisy observation is generated to simulate measurement uncertainty and the current belief is updated by an EKF step. Each agent runs an individual and independent EKF and maintains their own separate belief over themselves and others. Each agent receives noisy measurements with noise drawn independently from other agents. Agents do not share any information, such as policies, measurements, initializations, state estimates, or beliefs, during online operation. To test robustness, we simulate substantial amounts of noise, such that the belief $\mathbf{b}$ may significantly deviate from the true state of the system $\mathbf{x}$, shown in Figs. 4 and 6.

We encourage interaction by starting one agent with lower drag coefficient (and therefore higher speed) behind another *slower* agent. The *faster* agent will eventually catch up to the previous agent and initiate an overtaking maneuver. The better the interactions that are predicted and integrated into planning, the more successful overtaking maneuvers will occur.

The algorithm described in this article is able to synthesize competitive emergent behavior such as blocking of other vehicles and cutting in front of others, illustrated in Fig. 6. Additionally, although tight racing lines cut corners very closely, the chance constraints are successful in prohibiting collisions under the presence of motion and observation noise.

2) Benefits of Dynamic Game Planning: We compare the performance of dynamic game (DG) planning to conventional methods such as model predictive control (MPC). Both DG and MPC agents plan in belief space. The MPC agent has the exact same cost structure, but observes the other agents' executed actions and predicts agents to continue with the same action. The MPC baseline therefore predicts agents to not react to changes of their own actions and cannot leverage the effects of their own actions on other agents. The MPC is capable of

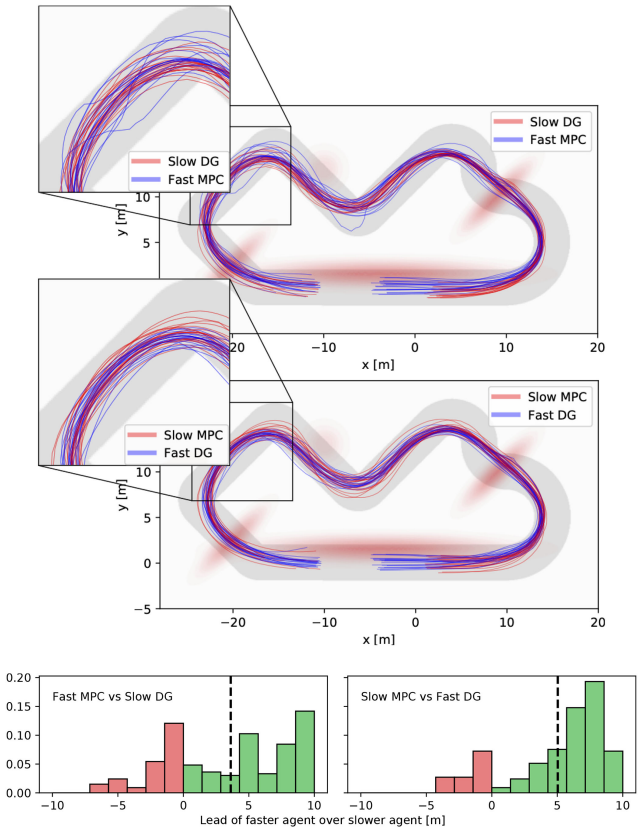

Fig. 7. *Top:* Traces of agents comparing MPC and DG. In both cases the MPC method moves away from the ideal racing line more often due to failed overtaking attempts. It cannot foresee its influence on the DG agent's actions and thus, is less efficient. It is also not able to take advantage of estimating its implicit control over the other agent like the DG agent. The agents start from random initialization locations around the origin. *Bottom:* Histograms of the Δarc-length lead of the faster agent over the slower agent. Green indicates that the faster agent won the race against the agent starting in the lead, whereas red indicates the opposite. In comparison, the DG method won more races than the MPC method and had a higher average lead.

TABLE II
RACING PERFORMANCE: DG VERSUS MPC AND BSP VERSUS NON-BSP

Competition Pair	Fraction of Fast winning
Fast DG BSP vs Slow MPC BSP	**82%**
Fast MPC BSP vs Slow DG BSP	64%
Fast DG BSP vs Slow DG non-BSP	**77%**
Fast DG non-BSP vs Slow DG BSP	63%
Fast DG BSP vs Slow DG BSP	**67%**

TABLE III
RACING PERFORMANCE: WINNING RATIO

Competition Pair	Win ratio
DG BSP vs MPC BSP	**1.44:1**
DG BSP vs DG non-BSP	**1.33:1**

synthesizing competitive racing trajectories, shown in Fig. 7, which are identical to the DG trajectories when no other agents are present. The performance of the DG planning distinguishes itself when interactions occur.

We display the results of 200 runs in Fig. 7, Tables II and III. The DG method wins t44% more races relative to the MPC baseline and has a larger lead on average. These results clearly

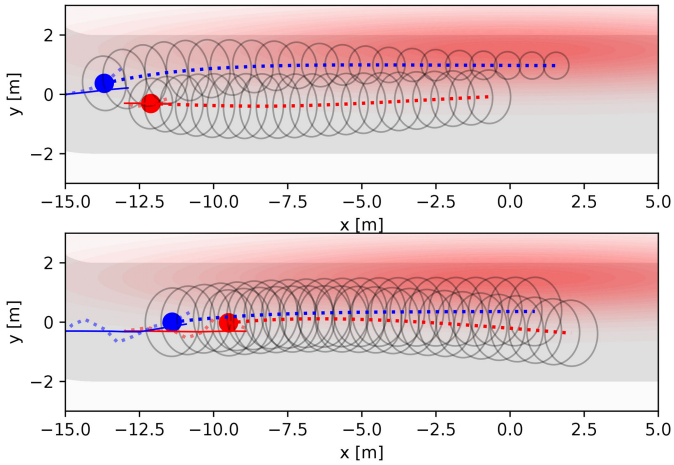

Fig. 8. *Top:* The blue agent overtakes the red agent by decreasing the uncertainty through the low-noise region and reducing the chance constraint (ellipses). *Bottom:* The blue agent has the same uncertainty over the planning horizon and fails to overtake since the chance constraints remain large.

illustrate the competitive advantage of our game-theoretic algorithm from leveraging how others react to one's own actions when planning.

3) Benefits of Belief-Space Planning: We also compare the performance of DG planning with and without BSP. In the non-BSP case the current uncertainty Σ_0 of the belief $\mathbf{b}_0$ is held constant over the planning horizon and is not influenced by expected measurements. Note that the current belief is still updated online by an EKF for both agents. Results are reported in Fig. 9 and Table II. The BSP variant wins 33% more races, has a larger average lead, and the fewest number of collisions. The non-BSP method collides nearly 10 times more often and exhibits behavior inappropriate for observed uncertainty levels. For example, agents are too conservative because low-noise regions are not considered in the planning phase, or too aggressive when entering sharp turns since additional motion noise due to breaking and steering are not accounted for.

Fig. 8 gives an intuitive explanation for the competitive advantage of planning in belief space. Without information gain, the follower will never be able to overtake due to the large chance constraint. Whereas with information gain, the chance constraint shrinks while moving through a low-noise zone, allowing the blue agent to overtake the leading agent. As shown in Fig. 9, the BSP agent can adapt their trajectories to account for increased noise due to strong actuation, i.e., braking and steering, and gaining information in low-noise regions.

Finally, we compare the performance when both agents use DG BSP, Table II, and see that the faster agent wins 67% of the races. Since the performance gain of the faster over the slower agent is smaller than in the previous two comparisons, we assume that the slower agent improves their blocking behavior more than the faster agent improves their ability to overtake. In scenarios where high uncertainty causes the chance constraints to occupy large parts of the track's width, the slower agent can often block the faster agent by proceeding in the middle of the road.

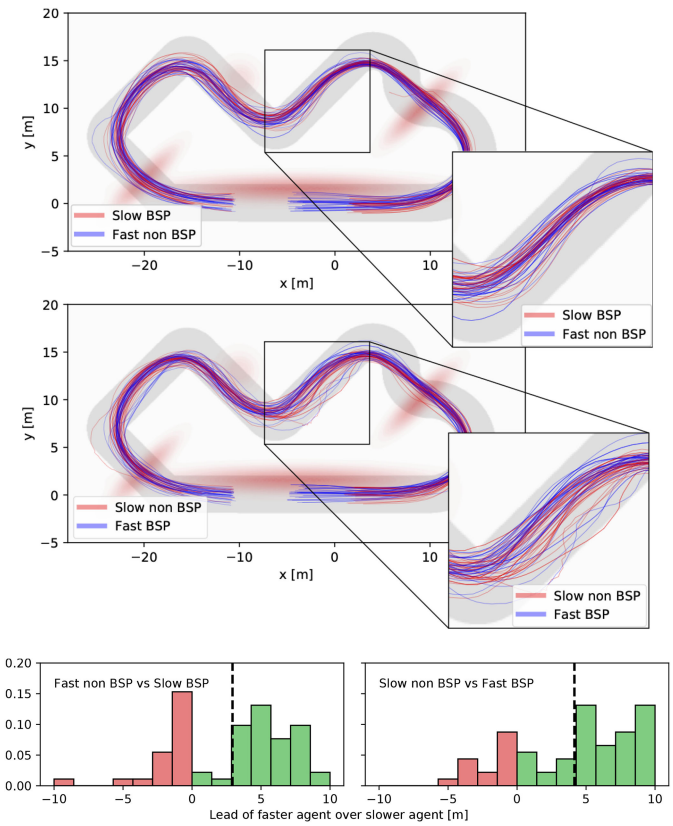

Fig. 9. *Top:* Traces of agents comparing BSP and non-BSP. In both cases the non-BSP method shows more unsafe behavior, leaving the track several times and nearly colliding with the other agent. The BSP agent attempts more aggressive overtaking maneuvers due to the lower uncertainty estimate over itself and the other agent, as shown in the cutout. The agents start from random initialization locations around the origin. *Bottom:* Histograms of the Δarc-length lead of the faster agent over the slower agent. Here, the BSP method won more races than the non-BSP method and had a higher average lead.

TABLE IV
AVERAGE COMPUTATION TIME

Experiment	Per iteration	Until convergence
Active surveillance	9.3 ms	371.3 ms
Guide dog	11.2 ms	474.9 ms
Racing	5.8 ms	110.5 ms

D. Real-Time Implementation Details

We implement our solver in the CasADi [45] framework leveraging autodifferentiation by source code transformation, automatic problem specific compute graph generation, C-code generation, and sparse operations. Exploiting sparsity is highly important to allow for real-time performance since the belief space, encompassing the mean state and the upper triangle of the covariance matrix can make respective Jacobian and Hessian matrices very large. The average compute times on a Ryzen 7 1700X 3.4 GHz are reported in Table IV. Algorithm 1 was run until convergence starting from a cold start for all experiments, i.e., the initial control trajectory $\mathbf{u}$ consists of all zeros. Nonetheless, it is also possible to run the algorithm sequentially by hot starting the optimization with the previous solution. This is a common practice in related optimization techniques for controls such as sequential quadratic programming [46] and allows to run Algorithm 1 at 100–200 Hz. In these cases it is often enough to run only very few iterations to update the previous solution.

V. CONCLUSION

In this article, we propose a formulation for integrating BSP into dynamic games, and present a real-time algorithm for solving the local Nash equilibria of these dynamic games in belief space. We demonstrate its performance of combining game-theoretic planning and information gathering with three case studies: Active surveillance, guiding blind agents, and racing with autonomous vehicles.

While game-theoretic planning models the interaction and dependency among agents, it does not address the quality of information available to the agent for decision-making. Incorporating BSP in dynamic games allows for new capabilities not possible with other approaches, essential in house service robots or interacting with human agents in traffic. Reasoning about another agent's uncertainty and simultaneously leveraging the effect of own actions on other agents' actions resulted in complex emergent behavior such as indirectly pushing and guiding others through regions of light, without the use of any form of direct communication.

In competitive use cases such as racing, emergent behavior consists of cutting, blocking, forcing others to break hard with the goal of increasing their uncertainty and slowing them down in turns, as well as the exploitation of high-information-gain zones for overtaking. In particular, game-theoretic BSP significantly increased performance in dynamic racing when benchmarked against state-of-the-art planning methods. Game-theoretic BSP wins 44% more races when competing with a nongame-theoretic baseline with BSP and 34% more races than a game-theoretic baseline without BSP.

In this work we limit ourselves to first-order beliefs to avoid the explosion in parameters for recursive beliefs over beliefs. Nonetheless, even in cases where a first-order belief assumption is a simplification of the true belief dynamics, such as racing, we see improved performance to baselines that do not take the belief over other agents into account. In future work we intend to develop extensions beyond first-order belief spaces.

We efficiently solve for Nash equilibria in belief space and achieve real-time performance, operating our algorithm at more than 100 Hz. Efficiently solving a quadratic game at each stage of the recursive backward pass of a belief-space variant of iLQG resulted in an algorithm with runtime complexity $\mathcal{O}(lN^7 n_{\mathbf{x}}^{i,6})$. Linear complexity in the planning horizon allowed for online deployment, in comparison to point-based POMDP algorithms with exponential complexity. While our algorithm also achieved polynomial runtime complexity in the number of agents N, future work will investigate lowering the complexity further.

ACKNOWLEDGMENT

This work was supported in part by NSF under Grant 1723943, in part by the Office of Naval Research (ONR) under Grant N00014-18-1-2830, and in part by the Toyota Research Institute (TRI). TRI has provided funds to assist the authors with their

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

REFERENCES

[1] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *Proc. 23rd AAAI Conf. Artif. Intell.*, 2008, vol. 8, pp. 1433–1438.

[2] D. Sadigh, N. Landolfi, S. S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for cars that coordinate with people: Leveraging effects on human actions for planning and active information gathering over human internal state," *Auton. Robots*, vol. 42, no. 7, pp. 1405–1426, 2018.

[3] M. Kuderer, S. Gulati, and W. Burgard, "Learning driving styles for autonomous vehicles from demonstration," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 2641–2646.

[4] W. Schwarting, A. Pierson, S. Karaman, and D. Rus, "Social behavior for autonomous vehicles," *Proc. Nat. Acad. Sci.*, vol. 116, no. 50, pp. 24972–24978, 2019.

[5] H. Kretzschmar, M. Kuderer, and W. Burgard, "Learning to predict trajectories of cooperatively navigating agents," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 4015–4020.

[6] 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 Trans. Control Syst. Technol.*, vol. 26, no. 5, pp. 1782–1797, Sep. 2018.

[7] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and decision-making for autonomous vehicles," *Annu. Rev. Control, Robot., Auton. Syst.*, vol. 1, pp. 187–210, 2018.

[8] J. R. Marden and J. S. Shamma, "Game theory and control," *Annu. Rev. Control, Robot., Auton. Syst.*, vol. 1, pp 105–134, 2018.

[9] 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 *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 2403–2410.

[10] A. Liniger and J. Lygeros, "A non-cooperative game approach to autonomous racing," 2017, *arXiv:1712.03913*.

[11] R. Spica, D. Falanga, E. Cristofalo, E. Montijano, D. Scaramuzza, and M. Schwager, "A real-time game theoretic planner for autonomous two-player drone racing," in *Proc. Robot. Sci. Syst.*, 2018, Art. no. 1801.

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

[13] Z. Wang, R. Spica, and M. Schwager, "Game theoretic motion planning for multi-robot racing," in *Proc. Int. Symp. Distrib. Auton. Robot. Syst.*, 2019, pp. 225–238.

[14] Z. Wang, T. Taubner, and M. Schwager, "Multi-agent sensitivity enhanced iterative best response: A real-time game theoretic planner for drone racing in 3D environments," *Robot. Auton. Syst.*, vol. 125, 2020, Art. no. 103410.

[15] D. Fridovich-Keil, E. Ratner, L. Peters, A. D. Dragan, and C. J. Tomlin, "Efficient iterative linear-quadratic approximations for nonlinear multi-player general-sum differential games," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 1475–1481.

[16] S. L. Cleac'h, M. Schwager, and Z. Manchester, "ALGAMES: A fast solver for constrained dynamic games," 2019, *arXiv:1910.09713*.

[17] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artif. Intell.*, vol. 101, no. 1/2, pp. 99–134, 1998.

[18] H. Bai, D. Hsu, and W. S. Lee, "Integrated perception and planning in the continuous space: A POMDP approach," *Int. J. Robot. Res.*, vol. 33, no. 9, pp. 1288–1302, 2014.

[19] H. Kurniawati, D. Hsu, and W. S. Lee, "SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces," in *Robotics: Science and Systems.* vol. 2008. Cambridge, MA, USA: MIT Press, 2008. [Online]. Available: http://www.roboticsproceedings.org/rss04/p9.html

[20] J. Pineau *et al.*, "Point-based value iteration: An anytime algorithm for POMDPs," in *Proc. Int. Joint Conf. Artif. Intell.*, vol. 3, 2003, pp. 1025–1032.

[21] E. A. Hansen, D. S. Bernstein, and S. Zilberstein, "Dynamic programming for partially observable stochastic games," *Proc. 19th Nat. Conf. Artif. Intell. Assoc. Adv.*, vol. 4, 2004, pp. 709–715.

[22] A. Bry and N. Roy, "Rapidly-exploring random belief trees for motion planning under uncertainty," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2011, pp. 723–730.

[23] S. Patil, J. Van DenBerg, and R.Alterovitz, "Estimating probability of collision for safe motion planning under Gaussian motion and sensing uncertainty," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 3238–3244.

[24] S. Prentice and N. Roy, "The belief roadmap: Efficient planning in belief space by factoring the covariance," *Int. J. Robot. Res.*, vol. 28, no. 11/12, pp. 1448–1465, 2009.

[25] A. Lee *et al.*, "Sigma hulls for Gaussian belief space planning for imprecise articulated robots amid obstacles," in *Proc. Int. Conf. Intell. Robots Syst.*, 2013, pp. 5660–5667.

[26] S. Patil, G. Kahn, M. Laskey, J. Schulman, K. Goldberg, and P. Abbeel, "Scaling up Gaussian belief space planning through covariance-free trajectory optimization and automatic differentiation," in *Proc. Algorithmic Foundations Robot. XI*, 2015, pp. 515–533.

[27] R. Platt, R. Tedrake, L. Kaelbling, and T. Lozano-Perez, "Belief space planning assuming maximum likelihood observations," in *Proc. Robot. Sci. Syst. Conf.*, 2010. [Online]. Available: http://www.roboticsproceedings.org/rss06/p37.html

[28] J. Van DenS. BergPatil, and R.Alterovitz, "Motion planning under uncertainty using iterative local optimization in belief space," *Int. J. Robot. Res.*, vol. 31, no. 11, pp. 1263–1278, 2012.

[29] W. Sun, J. van denBerg, and R.Alterovitz, "Stochastic extended LQR for optimization-based motion planning under uncertainty," *IEEE Trans. Automat. Sci. Eng.*, vol. 13, no. 2, pp. 437–447, Apr. 2016.

[30] B. Scassellati, "Theory of mind for a humanoid robot," *Auton. Robots*, vol. 12, no. 1, pp. 13–24, 2002.

[31] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA, USA: MIT Press, 2005.

[32] T. Basar and G. J. Olsder, *Dynamic Noncooperative Game Theory*. Philadelphia, PA, USA: SIAM, vol. 23, 1999.

[33] M. J. Osborne and A. Rubinstein, *A Course in Game Theory*. Cambridge, MA, USA: MIT Press, 1994.

[34] E. Wan, "Sigma-point filters: An overview with applications to integrated navigation and vision assisted control," in *Proc. Nonlinear Statistical Signal Process. Workshop*, 2006, pp. 201–202.

[35] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proc. IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[36] E. Todorov and W. Li, "A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems," in *Proc. IEEE Amer. Control Conf.*, 2005, pp. 300–306.

[37] W. Li and E. Todorov, "Iterative linear quadratic regulator design for nonlinear biological movement systems." in *Proc. 1st Int. Conf. Inform. Control, Automat. Robot.*, no. 1, 2004, pp. 222–229.

[38] D. H. Jacobson and D. Q. Mayne, *Differential Dynamic Programming*. New York, NY, USA: Elsevier, 1970.

[39] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quart. Appl. Math.*, vol. 2, no. 2, pp. 164–168, 1944.

[40] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *Proc. Int. Conf. Intell. Robots Syst.*, 2012, pp. 4906–4913.

[41] J.-M. Lien, S. Rodriguez, J. Malric, and N. Amato, "Shepherding behaviors with multiple shepherds," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2005, pp. 3402–3407.

[42] A. Pierson and M. Schwager, "Bio-inspired non-cooperative multi-robot herding," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 1843–1849.

[43] A. Pierson and M. Schwager, "Controlling noncooperative herds with robotic herders," *IEEE Trans. Robot.*, vol. 34, no. 2, pp. 517–525, Apr. 2018.

[44] D. Strömbom *et al.*, "Solving the shepherding problem: Heuristics for herding autonomous, interacting agents," *J. Roy. Soc. Interface*, vol. 11, no. 100, 2014, Art. no. 20140719.

[45] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi - a software framework for nonlinear optimization and optimal control," *Math. Program. Comput.*, vol. 11, pp. 1–36, 2019.

[46] J. Nocedal and S. Wright, *Numerical Optimization*. New York, NY, USA: Springer, 2006.

Wilko Schwarting received the B.Sc. and M.Sc. degrees in robotics, systems, and control from ETH Zurich, Zürich, Switzerland, in 2014 and 2016, respectively, and the Ph.D. degree in electrical engineering and computer science from the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 2021 while at the Computer Science and Artificial Intelligence Laboratory (CSAIL).

His research interests include interactions in mixed human–robot environments, learning complex behaviors that enable high-level reasoning, and planning under uncertainty arising from perception and prediction while incorporating others' beliefs.

Alyssa Pierson (Member, IEEE) received the B.S. degree in engineering from Harvey Mudd College, Claremont, CA, USA in 2010, and the M.S. and Ph.D degrees in mechanical engineering from Boston University, Boston, MA, USA in 2016 and 2017, respectively.

She is currently an Assistant Professor with the Department of Mechanical Engineering, Boston University (BU). Prior to joining BU in 2021, she was a Research Scientist with the Computer Science and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology (MIT), Cambridge, MA, USA. Her research interests include modeling cooperation and trust in multiagent systems, distributed control, and socially compliant autonomous system design.

Dr. Pierson was the recipient of the Clare Boothe Luce Fellowship in 2012.

Sertac Karaman (Member, IEEE) received the B.S. degrees in mechanical engineering and computer engineering from the Istanbul Technical University, Istanbul, Turkey, in 2007, the S.M. degree in mechanical engineering and the Ph.D. degree in electrical engineering and computer science from the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 2009 and 2012, respectively.

He is currently an Associate Professor of Aeronautics and Astronautics with MIT. His research interests include the broad areas of robotics and control theory. In particular, he is focusing on the applications of probability theory, stochastic processes,stochastic geometry, formal methods,and optimization for the design and analysis of high-performance cyber-physical systems. The application areas of his research include driverless cars, unmanned aerial vehicles, distributed aerial surveillance systems, air traffic control, certification and verification of control systems software, and many others.

Dr. Karaman was the recipient of the IEEE Robotics and Automation Society Early Career Award, in 2017, the Office of Naval Research Young Investigator Award, in 2017, the Army Research Office Young Investigator Award, in 2015, the National Science Foundation Faculty Career Development (CAREER) Award, in 2014, the AIAA Wright Brothers Graduate Award, in 2012, and the NVIDIA Fellowship, in 2011.

Daniela Rus (Fellow, IEEE) received the Ph.D. degree in computer science from Cornell University, Ithaca, NY, USA, in 1993.

She is the Andrew (1956) and Erna Viterbi Professor of Electrical Engineering and Computer Science and the Director of the Computer Science and Artificial Intelligence Laboratory with the Massachusetts Institute of Technology (MIT), cambridge, MA, USA. Prior to joining MIT, she was a Professor with the Department of Computer Science, Dartmouth College, Hanover, NH, USA. Her research interests include robotics, mobile computing, and big data. The key focus of her research is to develop the science of networked/distributed/collaborative robotics.

Dr. Rus is a Class of 2002 MacArthur Fellow, a Fellow of ACM and AAAI, and a member of the National Academy of Engineering.