Robotics: Science and Systems 2025
Los Angeles, CA, USA, June 21–25, 2025

Online Competitive Information Gathering for Partially Observable Trajectory Games

Mel Krusniak*, Hang Xu*, Parker Palermo*, and Forrest Laine*
* Vanderbilt University, Nashville, TN 37212
Email: mel.krusniak@vanderbilt.edu

***Abstract*—Game-theoretic agents must make plans that optimally gather information about their opponents. These problems are modeled by partially observable stochastic games (POSGs), but planning in fully continuous POSGs is intractable without heavy offline computation or assumptions on the order of belief maintained by each player. We formulate a finite history/horizon refinement of POSGs which admits competitive information gathering behavior in trajectory space, and through a series of approximations, we present an online method for computing rational trajectory plans in these games which leverages particle-based estimations of the joint state space and performs stochastic gradient play. We also provide the necessary adjustments required to deploy this method on individual agents. The method is tested in continuous pursuit-evasion and warehouse-pickup scenarios (alongside extensions to $N > 2$ players and to more complex environments with visual and physical obstacles), demonstrating evidence of active information gathering and outperforming passive competitors.**

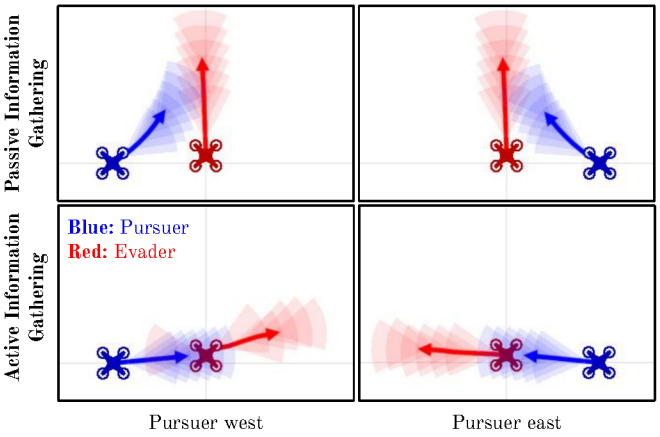

Fig. 1. Illustration of information gathering modalities in a pursuit-evasion game "TAG". Cones indicate field of view at each planned step. The pursuer's initial position ("east" / "west") is random and unknown to the evader.

I. INTRODUCTION

In this work, we consider rational online trajectory planning in non-cooperative, imperfect-information settings. While approximate solutions can be found to some partially observable multi-robot problems through e.g. reinforcement learning, modeling the information interactions of noncooperative robots in realistic spaces during execution remains difficult. For instance, robots in adversarial scenarios can leverage the sensing limitations of their adversaries to mislead or misdirect. These interactions are critical to intelligent behavior: intelligent agents seek out and use information, and anticipate their counterparts doing the same.

Observation interactions such as this are often modeled with either *extensive form games* or *partially observable stochastic games*, two closely related formulations [28]. These game-theoretic formulations allow for provably correct behavior with respect to adversarial information gathering and deception. Using these formulations to solve discrete parlor games like poker [24] is well studied, but at present, it remains an open challenge to do so live in continuous spaces as required in robotics without extensive offline computation and training. Our work addresses this case, presenting a method to plan information-aware strategies in the manner of model predictive game play (or MPGP — an extension of model predictive control to game-theoretic multi-robot environments) [7]. We contend that this work is complementary to and compatible with offline methods [4], and even when online methods are resource-intensive, they allow us to consider what is required to compute solutions from scratch. Our method is not real-time, though we demonstrate that real-time performance is within reach.

As a concrete example, consider a pursuit-evasion scenario between two free-moving UAVs with mounted cameras. Each UAV observes its opponent's location if its opponent is within its field of view (assumed to face the direction of motion), and observes random noise otherwise. The pursuer starts with equal probability in one of two locations — unknown to the evader, which starts between them. In an ideal, information-aware trajectory plan, the evader briefly turns to check one location, then executes the remainder of the plan based on its observation: a desirable behavior known as *active information gathering*. (In the alternative, *passive information gathering*, plans are generated without considering potential future observations.) Figure 1 visualizes the distinction in the UAV pursuit-evasion scenario, where each column shows a possible ground truth pursuer location.

Standard MPGP considers only perfect information games, so this work develops a variant which uses imperfect information games to permit active information gathering. We present a simple, live planner for these scenarios which permits competition through information as well as in trajectory space using a particle representation of the joint distribution of observation histories and states.

A. Technical Contributions

This work presents the following technical contributions:

- A novel enhancement of model-predictive game play to handle imperfect information games, addressing POSGs over a finite history and horizon;
- Description of a simple online solver method for this setup, with additions to permit execution in more realistic robotic planning contexts;
- Evidence that this method achieves active information gathering in pursuit-evasion and warehouse-pickup robotics scenarios, alongside extensions to N-player games and more realistic setups with visual and physical obstacles; and
- Experimental data regarding timing and equilibrium agreement using our method, as necessary considerations for practical game-theoretic multi-agent robotics.

II. Background

Our work is influenced by several overlapping formulations for imperfect information rational planning.

- *Partially observable stochastic games* (POSGs) are used in modern training of competitive agents.
- *Extensive form games* (EFGs), closely related to POSGs, are foundational for no-regret algorithms regarding agent interactions but are difficult to apply to large problems.
- *Multi-agent belief space planning* adapts belief-based control theory methods to N-player environments.
- *Interactive POMDPs* are applicable to some specific multi-agent cases with limitations on information modeling.

In this section, we address these major influences.

A. POSGs and EFGs

Partially observable stochastic games are N-player extensions of POMDPs [1], [28]. As an extension of planning in POMDPs, planning in POSGs is NP-hard [26], and most POSGs cannot be tractably solved. Nevertheless, modern advances in multi-agent reinforcement learning (MARL) have yielded approaches to finding strong strategies in POSGs, particularly in imperfect information parlor/video games like Stratego [17], Starcraft [25], and Diplomacy [2]. POSGs are closely related to *extensive form games* (EFGs), which provide a treelike interpretation of planning under imperfect information. Both model similar problems; the exact distinction made varies [12] [4].

Many important multi-agent RL-based methods are informed by POSGs and/or EFGs, including neural fictitious self-play [10], deep counterfactual regret minimization [6], and variations on policy space response oracles [14] [16]. In these methods, deep neural networks extend from tabular to continuous contexts and handle large state and action spaces at the cost of expensive offline training. Rather than solving such games in their entirety in this manner, we take an online view, seeking rational behavior in settings where pre-computation is not an option.

B. Belief Space Planning and Model Predictive Game Play

Given the context of online planning, we also consider control-theoretic work. For instance, belief space planning is an important concept in single-player planning which creates plans in state distributions (belief space) rather than in state space [19]. These methods are online, using minimal if any deep learning. Unfortunately, while formulating POMDPs into belief MDPs for this purpose is common, there is no widespread corresponding concept of a "belief POSG." Attributing beliefs to other players (as in the theory of mind [21]) forms a belief hierarchy: players have beliefs over the state, beliefs over other players' beliefs, beliefs over those beliefs, and so on. Working around this issue imposes limitations. Some methods (e.g., Laine et al. [13]) attribute immediate intentions in a planning framework but not formal beliefs. Schwarting et al. [22] presents an LQR-style controller for multiplayer belief-space planning but only considers first-order beliefs, which limits the potential competitive behavior in the information space.

Our work relies on a paradigm called *model predictive game play* (sometimes *game planning*), or MPGP [7] [27]. Rather than solving a trajectory optimization problem for each horizon as in typical model predictive control, in MPGP, players solve a trajectory equilibrium problem (that is, a game). Existing MPGP approaches are not designed for imperfect information games, though progress has been made by explicitly rewarding information gathering — for instance, Sadigh et al. [20] do so by rewarding observations that inform a model of human behavior. (Here, we reward agents for seeking information in terms of improvement on the underlying non-informational task rather than introducing a separate reward.)

C. Alternative Multi-Agent POMDPs

In competitive settings, *interactive POMDPs* (I-POMDPs) [8] model uncertainty interactions with arbitrarily hierarchical beliefs. This yields nuanced strategies but causes a large belief space that is computationally difficult to handle. Methods to address this include particle-filter approaches (somewhat akin to the approach used in our work) and Monte Carlo tree search variants [9] [23]. However, these methods were tested only on simple, discrete-space games. Furthermore, I-POMDP-based approaches are fundamentally limited by reasoning at a finite order. Higher-order reasoning (e.g., reasoning about other players' beliefs about one's own belief) is made intractable by an exponential dependency of belief space on belief order.

D. Summary of Background

Existing learning-based methods on frameworks like POSGs and EFGs address partially observable settings, but do so via offline training (and sometimes only in discrete spaces). More specific formulations restrict competition over information. Meanwhile, belief-space planning and model-predictive game play address online planning, but are either partially observable or multiplayer; never both. We situate our work between these to consider online game-theoretic planning under imperfect information in full.

In the remainder of this work we use POSGs as our descriptive framework, which suitably express imperfect information while permitting intuitive model-predictive game play.

III. Formulation

A. Framework

Consider a POSG $\mathcal{G}$, as defined as a tuple with the following elements:

- $\mathcal{N} := 1..N$, the set of players;
- $\mathcal{X}^{(i)}$, the set of states for player i (s.t. $x_t^{(i)} \in \mathcal{X}^{(i)}$);
- $\mathcal{A}^{(i)}$, the set of actions for player i (s.t. $a_t^{(i)} \in \mathcal{A}^{(i)}$);
- $\mathcal{Z}^{(i)}$, the set of observations for player i (s.t. $z_t^{(i)} \in \mathcal{Z}^{(i)}$);
- $T^{(i)}(x_t^{(i)}|x_{t-1}, a_{t-1})$, the state transition for player i;
- $O^{(i)}(z_t^{(i)}|x_t)$, the observation model for player i;
- $r^{(i)}(x_t)$, the (simplified) reward function for player i; and
- $X_0(x_0)$, the joint prior distribution over players' states. (This distributes over initial configurations of the game for all players. These initial states may be correlated between players.)

For notational simplicity we assume x is factored by player, though this need not be the case. We index over both players and time: $x^{(i)}$ denotes player i's states for all timesteps, x_t denotes *all* players' states at timestep t, and x denotes the joint state at all timesteps (and likewise for actions a and observations z). To maintain parity with extensive form games, we speak in terms of cumulative costs, and define them based on instant rewards as $c^{(i)}(x) := -\sum_{t=1}^{\infty} r^{(i)}(x_t)$.

We make the following high-level assumptions on all POSGs that we consider:

Assumption 1 (*Complete POSG and common knowledge*). $\mathcal{G}$ is fully defined; we do not consider incomplete games or model-free methods. All costs and transitions are public.

Assumption 2 (*Differentiability*). $T^{(i)}$, $O^{(i)}$, and $r^{(i)}$ are differentiable in all variables. This is required to use gradient play to seek equilibria at the planning level.

Assumption 3 (*Deterministic Strategies*). While mixed strategies can elicit competitive improvement in the types of games we consider [18], this improvement is often minor. For simplicity, we only consider pure strategies in this work.

B. Imperfect Information MPGP

We encounter two major necessities when adapting POSGs to MPGP:

1) **Active information gathering**: Rather than planning single trajectories, players find maps from possible future observation histories to trajectories. (This follows from the concept of information sets in extensive form games; observation histories in POSGs uniquely and perfectly index information sets. It also corresponds to a shift from open- to closed-loop Nash equilibria [3].)
2) **Belief modeling**: Players must model account for the past and future observations of *other* players, and plan accordingly, without invoking the belief hierarchy.

To address the former, we consider finite past and future of lengths T_{past} and T_{future} respectively, and optimize mappings from observation histories $z_{[t]}^{(i)} := [z_{t-T_{\text{past}}+1}^{(i)}, ..., z_t^{(i)}]$ to actions $a_t^{(i)}$. (We represent these mappings as policies π with parameters θ.) To address the latter, we require players to track the (fully unconditioned) joint distribution of observation histories and states $q_{\bar{t}}(x_{\bar{t}}, z_{[\bar{t}]}))$ from X_0 to the planning time $\bar{t}$, a move which we motivate in the section that follows.

The resulting game is

$$\left\{ \underset{\theta^{(i)}}{\operatorname{argmin}} \int_{\substack{x \in \mathcal{X}^{1+T_{\text{future}}} \\ z \in \mathcal{Z}^{T_{\text{past}}+1+T_{\text{future}}}}} c^{(i)}(x) p(x, z|x_{\bar{t}}, z_{[\bar{t}]}) q_{\bar{t}}(x_{\bar{t}}, z_{[\bar{t}]}) \right\}_i \tag{1}$$

where

$$p(x, z|x_{\bar{t}}, z_{[\bar{t}]}) = \prod_{t=\bar{t}}^{\bar{t}+T_{\text{future}}} T(x_{t+1}|x_t, \pi_\theta(z_{[t]}))\, O(z_t|x_t) \tag{2}$$

and $i \in 1..N$. **This game is the core of our approach**; each player $k \in 1..N$ will simultaneously solve all N of the optimization problems in (1) once per timestep to generate a plan (and generate plans for all other players as an important byproduct). There is no reference to the planning player k in this equilibrium problem — in theory all players solve identical problems, though we introduce some mechanisms for recovery if this fails to be the case in sections IV-B and IV-C.

C. Motivating Opponent Information Tracking

How is it that all players solve an identical game? In theory, a player does not know its opponents' observation histories, and it plays against opponents which do not know its own. To explain how Eq. 1 is formulated under this condition, we might imagine an optimization problem in which all players (as modeled by the planner) play in expectation over possible states and opponent observations, conditioned on their own observations. This leads to a version of q which can be forward-updated via Bayesian filter. Since observations are private, each player is represented by an infinite number of optimization units in the equilibrium problem solved by player k during planning, each with a unique possible observation history. We describe each optimization unit in this game as

$$\left\{ \underset{\theta^{(i,\hat{z}^{(i)})}}{\operatorname{argmin}} \int_{\substack{x \in \mathcal{X}^{1+T_{\text{future}}} \\ z \in \mathcal{Z}^{1+T_{\text{future}}}}} c^{(i)}(x) p(x, z|x_{\bar{t}}, \hat{z}^{(i)}) q_{\bar{t}}(x_{\bar{t}}, \hat{z}^{(\neg i)}|\hat{z}^{(i)}) \right\} \tag{3}$$

for *every* possible observation history at planning time held by each player, $\hat{z}^{(i)} \in (\mathcal{Z}^i)^{T_{\text{past}}}$. (This includes the planning player k — even though $\bar{z}^{(i)}$ is known by that player for $i = k$, the opponents modeled during planning must still treat it as unknown.)

Actualizing this setup would require agents to maintain the prior over both states and observation histories, $q_{\bar{t}}(x_{\bar{t}}, \hat{z}^{(\neg i)}|\hat{z}^{(i)})$, for every possible observation history $\hat{z}^{(i)}$ on which it might be conditioned. Of course, this is not possible, as there are infinitely many possible observation histories. Maintaining the fully joint probability over states and observation histories (which embeds $\hat{z}$) works around this. We use $q_{\bar{t}}^{(i)}(x_{\bar{t}}, \hat{z}^{(\neg i)}|\hat{z}^{(i)}) = q_{\bar{t}}^{(i)}(x_{\bar{t}}, \hat{z})/p(\hat{z}^{(i)})$, and because

player i selects unique actions for all $\hat{z}^{(i)}$, any scaling factor of the form $f(\hat{z}^{(i)}) > 0$ in player i's objective does not affect the policy parameters at equilibrium. As such, given $p(\bar{z}^{(i)}) > 0$, we may omit the marginal probability of $\hat{z}^{(i)}$ and consider $q_{\bar{t}}(x_{\bar{t}}, z_{[\bar{t}]})$, forming Eq. 1.

IV. Approach

Broadly, our approach maintains and updates $q_{\bar{t}}^{(i)}(x, z)$ using a particle representation, estimates the integral via Monte Carlo sampling, and solves the game with gradient play. Then, each player acts in accordance with model predictive game play, using their equilibrium policy with their true real-world observation history as input and advancing the true state of the world by only the first step of each plan. Players record a new real-world observation and the process is repeated.

The following sections describe this approach in detail.

A. Equilibrium Computation

We first turn to Nash equilibrium planning to solve the game presented in Eq. 1 given $q_{\bar{t}}^{(i)}(x_{\bar{t}}, z_{[\bar{t}]})$.

For each player, we evaluate the objective via ordinary Monte Carlo sampling, drawing K_{batch} particles $(x_{\bar{t}}, z_{[\bar{t}]})$ from $q^{(i)}$ and then rolling out the game for T_{future} steps on each particle to sample the remaining timesteps of x and z, generating actions using each players' policy. Using automatic differentiation, the cost gradient with respect to the policy parameters $\nabla_{\theta^{(i)}}[c^{(i)}(x)\ p(x, z)]$ is calculated, and applied via gradient descent. Gradient steps are performed until convergence for all players. (All components of the game are differentiable per our Assumption 2). There are some conditions under which gradient play does not converge to a Nash equilibrium [15]; however, empirically, we find it converges reliably enough for online planning, as we will discuss in section VI-B.

As we will discuss, each player's $q_{\bar{t}}^{(i)}$ is implemented with a particle representation consisting of potential states X, potential finite observation histories Z, and weights w. With this representation, Algorithms 1 and 2 describe the equilibrium planning procedure in detail, with the former implementing the basic Monte Carlo objective estimate and the latter implementing gradient play over particles. `opt` refers to any gradient-based optimization procedure on γ.

Algorithm 1: `obj` (expected cost over particles)

Input : Particles (X, Z, w); parameters θ; player i
Output: Expected finite-horizon cost for i across particles

for $k \in 1..K_{batch}$ **do**
  $x_{\bar{t},k}, z_{[\bar{t}],k} \sim (X,\ Z)$ w.p. w
  $c \leftarrow 0$
  while $t < \bar{t} + T_{future}$ **do**
    $z_{t+1} \sim \mathcal{Z}$ w.p. $O^{(i)}(\cdot|x_t)$
    $a_{t_1} \leftarrow \pi_\theta(o_{[t+1]})$
    $x_{t+1} \sim \mathcal{X}$ w.p. $T^{(i)}(\cdot|x_t, a_{t+1})$
    $c \leftarrow c + r^{(i)}(x_{t+1})$
  end
end
return c

Algorithm 2: `calc_eq` (gradient play over particles)

Input : Particles (X, Z, w); initialization θ, tolerance ϵ
Output: Parameters in equilibrium

$c_0^{(i)} \leftarrow \infty\ \ \forall i \in 1..N$
$j \leftarrow 1$
repeat
  for $i \in 1..N$ **do**
    $\theta^{(i)} \leftarrow \texttt{opt}(\theta^{(i)}, \nabla_{\theta^{(i)}}[\texttt{obj}\ ((X,\ Z,\ \text{w}),\ \theta,\ i)])$
    $c_j^{(i)} \leftarrow \texttt{obj}\ ((X,\ Z,\ \text{w}),\ \theta,\ i)$
    $\delta^{(i)} \leftarrow c_j^{(i)} - c_{j-1}^{(i)}$
  end
  $j \leftarrow j + 1$
until $\delta^{(i)} < \epsilon\ \ \forall i \in 1..N$
return θ

B. Distribution Update

The planning component previously discussed requires knowledge of $q_{\bar{t}}^{(i)}$, the distribution over game histories at the current time according to player i, necessitating a suitable update rule. Intuitively, $q_{\bar{t}}^{(i)}$ is the joint distribution of state/observation histories at time $\bar{t}$ assuming that all players have behaved rationally up to time $\bar{t}$. Since agents find equilibrium policies at each time step, $q_{\bar{t}}^{(i)}$ can be updated in an open-loop manner using only π, completely ignoring real-world observations. This is possible because π maps all possible observation histories to an action, not just $\bar{z}_{[\bar{t}]}$. By maintaining $q_{\bar{t}}^{(i)}$ as the joint distribution over histories of observations and states, players need not track a belief hierarchy over each others' beliefs: this joint distribution is the same for all players, and players' actions are determined entirely by the observations they receive, to which they respond optimally.

To that end, we approximate $q_{\bar{t}}^{(i)}$ with $K_{\text{all}} \gg K_{\text{batch}}$ particles (X, Z, w), with $X_k = x_{\bar{t},k}$ and $Z_k = z_{[\bar{t}],k}$ for $k \in 1..K_{\text{all}}$, each particle with weight w_k. After planning, each particle is stepped forward with the equilibrium policies and receives a sampled observation for each player according to its joint state. Particle weights are not affected, nor is there any conditioning on $\bar{z}_{\bar{t}}$.

We use this unconditioned approach because any approach conditioning $q_{\bar{t}}^{(i)}$ with $\bar{z}_{\bar{t}}$ causes an asymmetry between agents, which results in planning against opponents that are not representative of the real world (and therefore a suboptimal plan). However, if any player diverges from the equilibrium, or players find different equilibria, particles are updated with unrepresentative equilibrium policies and $q_{\bar{t}}^{(i)}$ may no longer match the true state. Furthermore, with limited particles, it is possible that the planning player's true observations $\bar{z}^{(i)}$ may not be represented jointly with the true state in any particle, causing the policy for $\bar{z}^{(i)}$ to optimize poorly during gradient play. As such, there is a fundamental trade-off between usage

of true observations to model the world state efficiently, and correctness in the corresponding model of the opponent.

Typically, a small amount of closed-loop behavior is desirable, even at the cost of inaccurate opponent representations. To accomplish this, we condition a small proportion of particles γ with true observations, reweighting them accordingly. γ can be interpreted as modulating the closed-loop / opponent-modeling trade-off.

Algorithm 3 implements this distribution update.

Algorithm 3: `update_particles` (update rule for $q^{(i)}$)

Input : Particles (X, Z, w), true observation $\bar{z}_{\bar{t}}^{(i)}$, policy parameters θ, probability γ
Output: Updated particles (X', Z', w')
for $k \in 1..K_{all}$ **do**
  $z'_{\bar{t}} \sim \mathcal{Z}$ w.p. $O(\cdot|X_k)$
  $w'_k \leftarrow w_k$
  with probability γ
    $z'_{\bar{t}} \leftarrow [\bar{z}_{\bar{t}}^{(i)}, z_{\bar{t}}'^{(\neg i)}]$
    $w'_k \leftarrow w_k \cdot O(\bar{z}_{\bar{t}}|X_k)$
  $Z'_k \leftarrow [Z_k ..., z'_{\bar{t}}]_{[\bar{t}]}$
  $X'_k \sim \mathcal{X}$ w.p. $T(\cdot|X_k, \pi(z'; \theta))$
end
return (X', Z', w')

C. Equilibrium Selection

Our method can be deployed in a "shared brain" setting, in which case it calculates an equilibrium joint policy once across all players, or in a "separate brain" setting, in which case all players do so independently. In the former case, it is assumed that $q_{\bar{t}}^{(i)} = q_{\bar{t}}^{(j)}$ for all $i, j \in 1..N$. However, in the latter, more realistic case where agents plan individually, this is not generally true. Agents may find different local equilibrium joint policies, potentially modeling opponent behavior incorrectly and causing $q_{\bar{t}}^{(i)}$ and $q_{\bar{t}}^{(j)}$ to diverge for $j \neq i$.

Figure 2 visualizes this failure case in a pursuit-evasion scenario (for more detail on the pursuit-evasion game used here, see Section V). The left and right plots visualize the pursuer (blue) and evader (red) distributions respectively. In this case, the pursuer finds one deterministic-strategy equilibrium in which the evader turns right at the boundary, and the evader finds a different equilibrium in which it turns left. The discrepancy primarily harms the pursuer, causing it to pursue an inaccurate model of the evader.

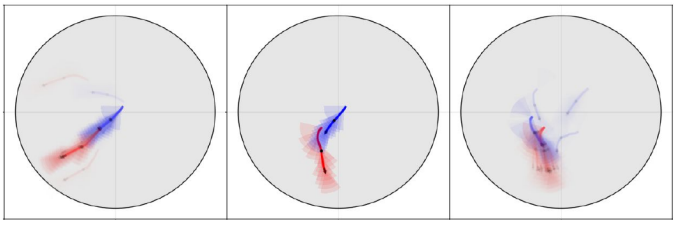

Fig. 2. Visualization of an equilibrium selection failure in a pursuit-evasion game. Opacity mapped to particle weight.
(Left): Pursuer $q^{(i)}$. (Middle): True state. (Right): Evader $q^{(i)}$.

Fortunately, as long as one or more particles behave consistently with an opponent's true policy and $\gamma > 0$, the correct $D^{(i)}$ can generally be recovered as conditional updates deweight particles that are inconsistent with observations of the opponent's true state evolution in favor of those that are. To promote the existence of particles following different possible equilibria selections, an agent can complete the equilibrium planning step N_{eq} times. Each of these candidate policies is applied to a subset of particles during the distribution update step, and the first one is arbitrarily chosen to provide the real-world action.

Alg. 4 fully describes our proposed method in the separate-brain setting. In Alg. 4, `partition` creates N_{eq} roughly-equal index sets of the particles, and `act` performs an action in the real world; the remaining functions are as given in Alg.s 2 and 3. Note that Alg. 4 is not always guaranteed to converge for all POSGs — in particular, those which do not have pure strategy equilibria (violating Assumption 3). However, we find its empirical performance compelling, as we demonstrate over the following two sections.

Algorithm 4: Active information planning

for $k \in 1..K_{all}$ **do**
  $X_k \sim X_0$
  $Z_k \leftarrow \vec{0}_{T_{\text{past}} \times |\mathcal{O}|}$
end
$\mathcal{P} \leftarrow$ `partition`$(1..K_{all}, N_{\text{eq}})$
$\bar{z}^{(i)} \leftarrow \vec{0}_{T_{\text{past}} \times |\mathcal{O}^{(i)}|}$

for $\bar{t} \in 1..\infty$ **do**
  for $p \in 1..N_{eq}$ **do**
    $\theta_p \leftarrow$ `calc_eq`$((X, Z, w), \theta_p)$
  end
  $\bar{z}_{\bar{t}}^{(i)} \leftarrow$ `act`$(\pi_{\theta_1^{(i)}}^{(i)}(\bar{z}_{[\bar{t}]}))$
  for $p \in 1..N_{eq}$ **do**
    $P \leftarrow \mathcal{P}_p$
    $(X_P, Z_P, w_P) \leftarrow$
     `update_particles`$((X_P, Z_P, w_P), \bar{z}_{\bar{t}}^{(i)}, \theta_p)$
  end
end

D. Implementation

We implemented this approach in the Julia programming language [5]. Policies are represented with feedforward neural networks and optimized with the Adam optimizer. We used Flux.jl [11] to implement these networks and perform automatic differentiation.

V. Experimental Setup

To test our method, we considered three variants of a UAV pursuit-evasion scenario as well as a warehouse-pickup scenario, all in continuous space. We describe these scenarios in this section.

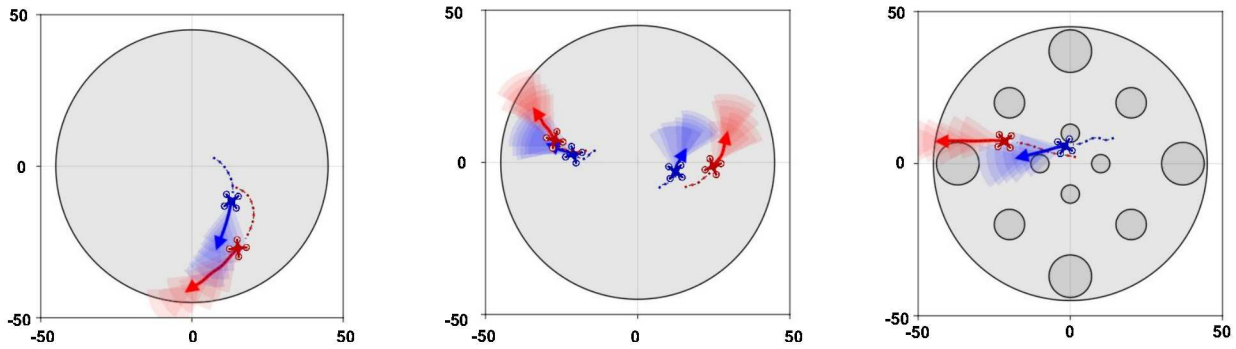

Fig. 3. Trajectories in the three simulated UAV tag environments. Pursuer is blue; evader is red. Plans are solid lines and histories are dotted. (Left) Field of view tag. (Center) Tag chain. (Right) Hide and seek.

A. *Pursuit-Evasion (UAV Tag)*

Pursuit-evasion is a self-explanatory scenario in two dimensions comparable to the childhood game of "tag." We simulated two UAVs equipped with fixed field-of-view cameras, notated "pursuer" and "evader".

Control. Agents control their own velocity x_{vel}, and their position x_{pos} is simulated through double-integrator dynamics for $T_{\text{future}} = T_{\text{past}} = 7$ timesteps. The evader has a slightly greater maximum velocity than the pursuer.

Costs. The pursuer (arbitrarily indexed $i = 1$) seeks to minimize $||x_{\text{pos}}^{(1)} - x_{\text{pos}}^{(2)}||_2$ while the evader seeks to maximize it at every timestep. Agents are independently penalized for departures from a circular area; otherwise, this is a zero-sum game.

Observations. Both players receive perfect observations of their own state. Observations of opponent position are limited to a field of view in the UAV's direction of motion. Specifically, observations of the opponent are drawn from a trimmed multivariate Gaussian around the opponent location with $\mu = x_{\text{pos}}^{(\neg i)}$ and

$$\sigma^2 = \sigma_{\text{base}}^2 + \begin{cases} C_{\text{scale}} \cdot (|\theta| - \frac{f}{2}) & |\theta| \geq \frac{f}{2} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where θ is the bearing between the agent's heading (taken from their velocity) and their opponent, and f is the agent's field of view, both in radians. Observations are trimmed to be within the play area. Parameters σ_{base}^2 and C_{scale} adjust the minimum variance and the variance per radian outside the field of view, respectively.

Initialization. Initial states are normally distributed around the origin. There is no process noise; uncertainty is a result of initial uncertainty or observation noise (as a simplification, rather than a limitation of the method).

Variant: Tag chain. In this variant, there are any (even) number of players: $\frac{N}{2}$ pursuers labeled $P_1..P_{\frac{N}{2}}$ pursue evaders $E_1..E_{\frac{N}{2}}$. Pursuer P_i pursues evader E_i, but evader E_i evades pursuer P_{i+1} (and evader $E_{\frac{N}{2}}$ evades pursuer P_1). This results in a cycle of partial pursuit-evasion games. Agents receive observations of all other agents in their field of view. We consider this variant to demonstrate extensibility to N-player, general-sum scenarios. We use $N = 4$, and for simplicity, we consider chain tag in the "shared brain" setting only, assuming $q_t^{(i)} = q_t^{(j)}$.

Variant: Hide and seek. In this variant, the simulated play area includes circular obstacles that block UAVs' fields of view. Agents receive high-variance observations for obstructed opponents in the same way as those outside the field of view. A penalty is applied for colliding with these obstacles in the same manner as the exterior boundary. We consider this variant to showcase a complex environment which may incentivize nuanced hiding and detection behavior.

B. *Warehouse Pickup*

We also consider one non-pursuit-evasion example. Imagine a warehouse which uses an autonomous robot ("P1") to load goods. P1 simply proceeds greedily towards the nearest task location. At a later time, an additional robot ("P2"), created by a different manufacturer, is introduced. Unlike P1, P2 is designed for multi-robot environments and does not proceed greedily towards task location, instead reasoning about other agents in a decentralized manner. P2 is connected to a simple station which broadcasts the locations of both robots.

We seek a plan for P2 that works around P1's inability to cooperate. However, there is an additional wrinkle — locations broadcast by the station may be noisy. This inaccuracy is modeled as the sum of two components: inaccuracy caused by potentially stale data about a distant P1, and inaccuracy caused by channel noise between P2 and the station. We use P1's and P2's distance from the station, respectively, as proxies for the amount of noise incurred by these effects.

Control. The warehouse is modeled by the $[0, 1]^2$ space. Control is the same as in pursuit-evasion examples. P2 has a higher maximum velocity than P1.

Costs. Both players are penalized by their distance from obstacles, with additive penalty $-\exp(-\beta||x_{\text{pos}}^{(i)} - \tau_j||_2^2)$ for each target j. P2 is additionally penalized for proximity to P1, with additive penalty $-\alpha \exp(-\beta||x_{\text{pos}}^{(2)} - x_{\text{pos}}^{(1)}||_2^2)$. α and β control the weighting between P2's objectives and the scale used to determine proximity; we use $\alpha = 4.0$ and $\beta = 20$.

Observations. P1 only observes its own position perfectly. P2 observes its own position perfectly and P1's position

imperfectly. Its observations of P1 are exact when both players are at the station (notated $s := [0.5, 1.0]$) and receive additive noise $\sim \mathcal{N}(0, \sigma^2)$ where $\sigma = \eta_1 ||x_{\text{pos}}^{(1)} - s||_2 + \eta_2 ||x_{\text{pos}}^{(2)} - s||_2$. η controls the strength of each noise component; here, we use $\eta_1 = \eta_2 = 4$.

Initialization. Both robots are randomly initialized anywhere in the warehouse. We use two task locations, which are also random.

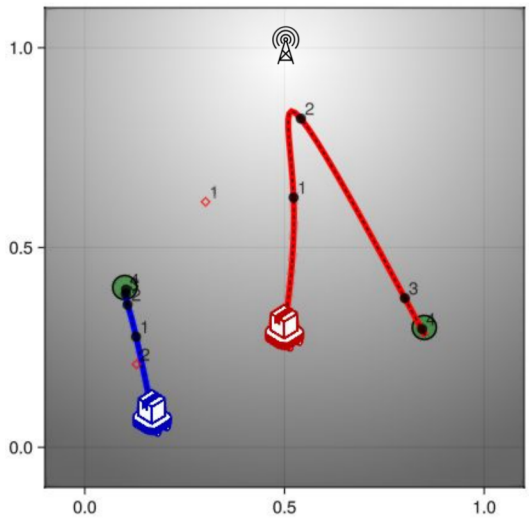

Fig. 4. Active information gathering behavior in a 4-step warehouse pickup environment. Background gradient indicates observation accuracy. Diamonds indicate P2's observations (observations 3 and 4 are out of bounds). Green circles indicate task locations.

Fig. 4 visualizes correct behavior in the warehouse pickup scenario (as generated by Alg. 2). P2 (red) first travels to the station to localize P1, then selects the task location further from P1 to prevent a collision.

VI. Results

A. Improvement from Active Information Gathering

As a baseline, we used an passive-gathering version of our solver which implements a variant of Eq. 1 where π maps only from past observations. We ran each of the four possible configurations — passive or active solver for each player — on twenty independent trials for each scenario (seventy for the warehouse example). Each lasted twenty timesteps. We used $T_{\text{future}} = T_{\text{past}} = 6$, $\gamma = 0.1$, $K_{\text{all}} = 1000$, and $K_{\text{batch}} = 10$, except where noted.

For simplicity of analysis, we assume equilibrium selection succeeds and players find the same equilibrium — we explicitly consider equilibrium selection in the next section. We terminate Alg. 2 after 100 gradient iterations (rather than allowing convergence) for a more level comparison between active and passive information gathering, as the active version typically takes more iterations to converge.

The competitive behavior of the agents for each active/passive configuration and each scenario is quantified in Table I, listing the average costs (in distance units) achieved by each player over all trials with standard error. We do not include boundary penalties in the reported cost. Both pursuer/P1 (blue) and evader/P2 (red) costs are shown in each cell. We note the following scenario-specific details: (1) TAG and HIDE&SEEK are effectively zero-sum, thus the apparent repetition. (2) In TAGCHAIN, mean costs for each team (evaders and pursuers) are used rather than individual costs. (3) In the WAREHOUSE example, P1 has no imperfect observations and therefore there is no distinction between active and passive modes.

In all cases except one, our method for active information gathering elicits a reduction in cost. In Table I, blue (pursuer) costs are lower in the left column, which corresponds to that agent using active information gathering, and likewise red (evader/P2) costs are lower in the second row of each scenario. Figures 1 and 4 show the desirable, active plans in TAG and WAREHOUSE, where in both cases an information-gathering trajectory clearly outperforms a greedy plan. Several of these differences (particularly in TAG and WAREHOUSE) are statistically significant with $p < 0.05$.

The single exception to improvement is the evader in HIDE&SEEK, which exhibits a cost increase under our method. We reason that optimal evader strategy in this game is too complex to be captured with only T_{past} past observations, and therefore modeling information in any given 6-step horizon is a poor strategic decision which unnecessarily complicates the optimization procedure.

		Passive pursuer(s)	**Active pursuer(s)**
TAG	**Passive evader**	12.58 ± 1.44 -12.58 ± 1.44	9.97 ± 1.19 -9.97 ± 1.19
	Active evader	13.97 ± 1.92 -13.97 ± 1.92	12.98 ± 1.18 -12.98 ± 1.18
TAGCHAIN	**Passive evaders**	17.68 ± 1.39 -25.91 ± 1.54	16.79 ± 1.43 -25.12 ± 1.57
	Active evaders	15.79 ± 1.23 -27.48 ± 1.39	15.71 ± 1.26 -27.29 ± 1.41
HIDE&SEEK	**Passive evader**	19.62 ± 1.33 -19.62 ± 1.33	17.93 ± 1.34 -17.93 ± 1.34
	Active evader	18.91 ± 1.25 -18.91 ± 1.25	16.75 ± 1.34 -16.75 ± 1.34
		Passive P1	**Active P1**
WAREHOUSE	**Passive P2**	-5.12 ± 0.26 -0.02 ± 0.34	n/a
	Active P2	-5.21 ± 0.26 -1.76 ± 0.54	n/a

TABLE I
COSTS IN PASSIVE/ACTIVE CONFIGURATIONS, PER SCENARIO

B. Timing and Convergence

While this method is not yet real-time, we do report some preliminary timing results.

Table II gives the average wall time *per gradient step*, per player, for each scenario. We average over 20 trials,

otherwise using the same setup as in the previous experiments. (Note that the particle distribution updates take a negligible amount of time in comparison). Based on these results, we believe real-time execution is within reach, particularly as code optimization, GPU support, better handling of matrix sparsity, and improved parallelization all remain as potential improvements.

Scenario	Time (seconds)
TAG	0.085 ± 0.036
TAGCHAIN	0.222 ± 0.078
HIDE&SEEK	0.117 ± 0.030
WAREHOUSE	0.062 ± 0.039

TABLE II
OPTIMIZATION TIMES PER GRADIENT STEP, PER PLAYER

As noted previously, Alg. 4 is not guaranteed to converge in all scenarios — for instance, some configurations of pursuit-evasion have mixed strategy solutions which cannot be found using pure gradient play. However, empirically, we note that the algorithm converges well when pure strategy solutions are available. Figure 5 shows the cost convergence when optimizing the first step of WAREHOUSE over 600 gradient iterations, taken across ten trials, against a randomly placed P1. While there is still inherent noise due to the method of stochastic gradient play, the method largely converges, and as policy parameters from the previous timestep are reused as initial parameters, subsequent steps benefit from past optimization (motivating our choice of 100 gradient steps in our experiments).

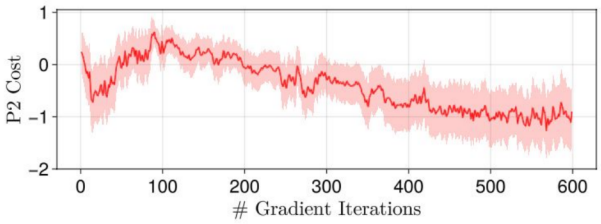

Fig. 5. P2's cost as the first step of WAREHOUSE is solved. Ribbon indicates standard error.

We also examined empirical scaling with respect to the method parameters. Table III shows scaling behavior w.r.t T_{future} over ten trials. The relation between T_{future} and time-to-converge is theoretically exponential: the number of possible world configurations, and ergo particles that must be processed, explodes with time. However, information gathering early in the plan limits potential future configurations (and reduces the future cost). (On the other hand, we found little relation between T_{past} and time-to-converge, which we believe to be specific to the relatively atemporal games we analyzed.)

T_{future}	Cost	Wall Time (s)
1	0.604 ± 0.21	4.411 ± 0.29
2	0.338 ± 0.11	7.907 ± 1.35
3	0.253 ± 0.09	7.254 ± 1.03
4	0.023 ± 0.10	11.281 ± 1.33
5	-0.075 ± 0.11	12.009 ± 1.85
6	-0.197 ± 0.10	16.415 ± 2.91
7	-0.162 ± 0.13	19.509 ± 2.67

TABLE III

Finally, we report scaling with respect to the number of particles per batch K_{batch}. (K_{all} has little effect on convergence, as gradient play, the most expensive component, only occurs over the batch). Insufficient K_{batch} results in fast convergence to an information-unaware minimum, as indicated in Table IV.

K_{batch}	P2 Cost	Wall Time (s)
2	0.326 ± 0.02	4.86 ± 0.29
6	0.308 ± 0.02	5.692 ± 1.11
15	0.227 ± 0.03	7.199 ± 1.18
39	0.206 ± 0.05	7.673 ± 0.84
100	0.067 ± 0.04	11.791 ± 1.5
251	0.052 ± 0.04	14.462 ± 1.79
630	0.001 ± 0.07	32.497 ± 4.65

TABLE IV

C. Equilibrium Agreement

In general, POSGs can admit arbitrarily many equilibria in deterministic strategies, and determining which equilibrium an opponent is playing is a fundamental difficulty of agent modeling. Section IV-C details our approach to minimizing this issue. In this section we analyze how well this approach performs.

We tested ordinary field of view tag with different settings of N_{eq} for each player, playing the game for 100 steps but otherwise keeping the same setup as Section VI-A. Figure 6 shows distance between the pursuer and evader for each configuration of N_{eq} (blue configurations favor the pursuer; red favor the evader). Performance improves for both players with more equilibrium solves. Games with insufficient N_{eq} favor the evader, possibly because pursuer equilibrium policies (which converge on the evader) are more similar than evader equilibrium policies (which diverge).

To confirm that this improvement is actually caused by better opponent modeling, we tracked the approximate expected negative log likelihood (surprisal) of the true opponent position for various settings of N_{eq} for each agent, using a Gaussian approximation of $q_{\bar{t}}^{(i)}(\bar{x}_{\bar{t}}^{(\neg i)})$ and taking $N_{\text{eq}} = 4$ for the opponent. Higher values indicate the opponent true position is not captured by $q_{\bar{t}}^{(i)}$. Figure 7 shows the results, indicating that a greater number of solves decreases uncertainty about the opponent position. The pursuer tends to receive more informative observations, improving its estimates independently of N_{eq}; this causes the discrepancy between pursuer and evader seen in Figure 7. Note that trials for each setting of N_{eq} are seeded by sample number.

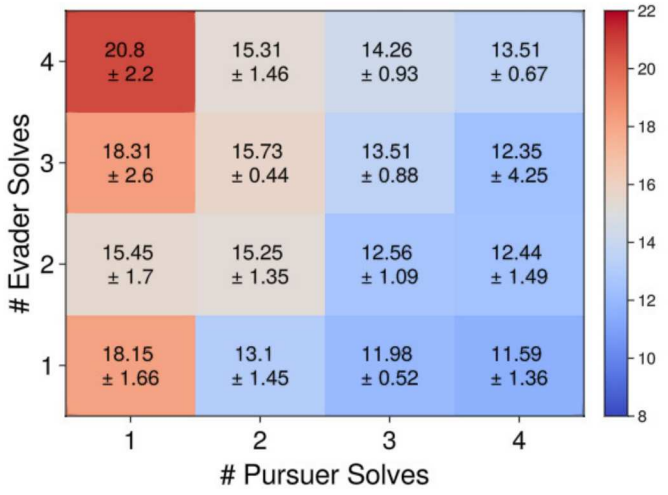

Fig. 6. Distance between players in field of view tag for different configurations of N_{eq} Red configurations favor the evader; blue favor the pursuer.

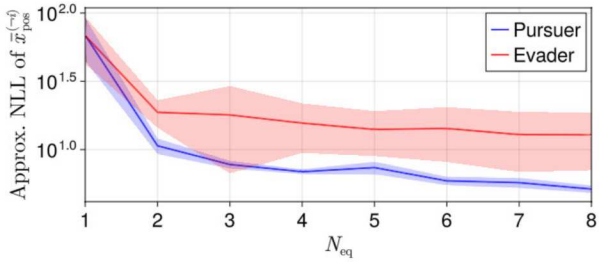

Fig. 7. Negative log likelihood of opponent position for settings of N_{eq}. Logarithmic scale. Ribbons indicate standard error.

VII. Conclusion

A. Summary

In this work we discussed rational information gathering in the context of model predictive game planning. We formulated the active planning problem in a finite-history/horizon setting, and described a basic method for incorporating potential future observations into the planning procedure using gradient play and a number of approximations and additions to make the method suitable for independent planning agents. Using that method, we showed that active information gathering is indeed competitively preferable for players in variants of a pursuit-evasion game, and provided additional analysis on the parameters of the solution method.

B. Limitations

As noted, we implemented our approach with no attention given to fast planning. In reality, code and hardware optimizations make this prospect more likely, making this approach "online" in both the analytical and practical sense.

More fundamentally, we have assumed a finite history of observations is suitably informative, and as we discussed regarding the "hide-and-seek" example, performance suffers when this is not the case. There are many potential approaches to handling the curse of history, and applying any of them in a way that avoids the need for recursively hierarchical beliefs is a promising research direction.

The method is also subject to all assumptions made in Section III-A. In particular, nondeterministic strategies can be permitted with a minor formulation change, and the ability to deliberately, directly introduce uncertainty into the state is quite consequential in settings permitting information gathering (and also potentially permits theoretical convergence guarantees). Combining imperfect information methods with mixed-strategy approaches like that in [18] may be a source of further improvement in competitive play.

References

[1] Stefano V Albrecht, Filippos Christianos, and Lukas Schäfer. *Multi-agent reinforcement learning: Foundations and modern approaches*. MIT Press, 2024.

[2] Anton Bakhtin, David J Wu, Adam Lerer, Jonathan Gray, Athul Paul Jacob, Gabriele Farina, Alexander H Miller, and Noam Brown. Mastering the game of no-press diplomacy via human-regularized reinforcement learning and planning. *arXiv preprint arXiv:2210.05492*, 2022.

[3] Tamer Başar and Georges Zaccour. *Handbook of dynamic game theory*. Springer, 2018.

[4] Tyler Becker and Zachary Sunberg. Bridging the Gap between Partially Observable Stochastic Games and Sparse POMDP Methods, May 2024. URL http://arxiv.org/abs/2405.18703. arXiv:2405.18703 [cs].

[5] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

[6] Noam Brown, Adam Lerer, Sam Gross, and Tuomas Sandholm. Deep counterfactual regret minimization. In *International conference on machine learning*, pages 793–802. PMLR, 2019.

[7] Simon Le Cleac'h, Mac Schwager, and Zachary Manchester. Algames: A fast solver for constrained dynamic games. *arXiv preprint arXiv:1910.09713*, 2019.

[8] Piotr J Gmytrasiewicz and Prashant Doshi. Interactive pomdps: Properties and preliminary results. In *International Conference on Autonomous Agents: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-*, volume 3, pages 1374–1375, 2004.

[9] Yanlin Han and Piotr Gmytrasiewicz. Learning others' intentional models in multi-agent settings using interactive pomdps. *Advances in Neural Information Processing Systems*, 31, 2018.

[10] Johannes Heinrich and David Silver. Deep reinforcement learning from self-play in imperfect-information games. *arXiv preprint arXiv:1603.01121*, 2016.

[11] Mike Innes. Flux: Elegant machine learning with julia. *Journal of Open Source Software*, 3(25):602, 2018.

[12] Vojtěch Kovařík, Martin Schmid, Neil Burch, Michael Bowling, and Viliam Lisý. Rethinking formal models of partially observable multiagent decision making. *Artificial Intelligence*, 303:103645, 2022.

[13] Forrest Laine, David Fridovich-Keil, Chih-Yuan Chiu, and Claire Tomlin. Multi-hypothesis interactions in

game-theoretic motion planning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8016–8023. IEEE, 2021.

[14] Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Pérolat, David Silver, and Thore Graepel. A unified game-theoretic approach to multiagent reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

[15] Eric Mazumdar, Lillian J Ratliff, and S Shankar Sastry. On gradient-based learning in continuous games. *SIAM Journal on Mathematics of Data Science*, 2(1):103–131, 2020.

[16] Stephen McAleer, JB Lanier, Kevin A Wang, Pierre Baldi, and Roy Fox. XDO: A Double Oracle Algorithm for Extensive-Form Games. In *Advances in Neural Information Processing Systems*, volume 34, pages 23128–23139. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/hash/c2e06e9a80370952f6ec5463c77cbace-Abstract.html.

[17] Julien Perolat, Bart De Vylder, Daniel Hennes, Eugene Tarassov, Florian Strub, Vincent de Boer, Paul Muller, Jerome T Connor, Neil Burch, Thomas Anthony, et al. Mastering the game of stratego with model-free multiagent reinforcement learning. *Science*, 378(6623):990–996, 2022.

[18] Lasse Peters, David Fridovich-Keil, Laura Ferranti, Cyrill Stachniss, Javier Alonso-Mora, and Forrest Laine. Learning mixed strategies in trajectory games. In *18th Robotics: Science and Systems, RSS 2022*. MIT Press Journals, 2022.

[19] Robert Platt Jr, Russ Tedrake, Leslie Pack Kaelbling, and Tomas Lozano-Perez. Belief space planning assuming maximum likelihood observations. In *Robotics: Science and systems*, volume 2, 2010.

[20] Dorsa Sadigh, S Shankar Sastry, Sanjit A Seshia, and Anca Dragan. Information gathering actions over human internal state. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 66–73. IEEE, 2016.

[21] Brian Scassellati. Theory of mind for a humanoid robot. *Autonomous Robots*, 12:13–24, 2002.

[22] Wilko Schwarting, Alyssa Pierson, Sertac Karaman, and Daniela Rus. Stochastic dynamic games in belief space. *IEEE Transactions on Robotics*, 37(6):2157–2172, 2021.

[23] Jonathon Schwartz, Ruijia Zhou, and Hanna Kurniawati. Online planning for interactive-pomdps using nested monte carlo tree search. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8770–8777. IEEE, 2022.

[24] Oskari Tammelin, Neil Burch, Michael Johanson, and Michael Bowling. Solving heads-up limit texas hold'em. In *Twenty-fourth international joint conference on artificial intelligence*, 2015.

[25] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575(7782):350–354, 2019.

[26] Nikos Vlassis, Michael L Littman, and David Barber. On the computational complexity of stochastic controller optimization in pomdps. *ACM Transactions on Computation Theory (TOCT)*, 4(4):1–8, 2012.

[27] Zijian Wang, Riccardo Spica, and Mac Schwager. Game theoretic motion planning for multi-robot racing. In *Distributed Autonomous Robotic Systems: The 14th International Symposium*, pages 225–238. Springer, 2019.

[28] Yaodong Yang and Jun Wang. An overview of multiagent reinforcement learning from game theoretical perspective. *arXiv preprint arXiv:2011.00583*, 2020.