

# Planning with Critical Decision Points: Robots that Influence Humans to Infer Their Strategy

Debasmita Ghose\*, Michal Lewkowicz\*, David Dong, Andy Cheng, Tran Doan, Emma Adams, Marynel Vázquez and Brian Scassellati

**Abstract**—To enable sophisticated interactions between humans and robots in a shared environment, robots must infer the intentions and strategies of their human counterparts. This inference can provide a competitive edge to the robot or enhance human-robot collaboration by reducing the necessity for explicit communication about task decisions. In this work, we identify specific states within the shared environment, which we refer to as *Critical Decision Points*, where the actions of a human would be especially indicative of their high-level strategy. A robot can significantly reduce uncertainty regarding the human’s strategy by observing actions at these points. To demonstrate the practical value of Critical Decision Points, we propose a Receding Horizon Planning (RHP) approach for the robot to influence the movement of a human opponent in a competitive game of hide-and-seek in a partially observable setting. The human plays as the hider and the robot plays as the seeker. We show that the seeker can influence the hider to move towards Critical Decision Points, and this can facilitate a more accurate estimation of the hider’s strategy. In turn, this helps the seeker catch the hider faster than estimating the hider’s strategy whenever the hider is visible or when the seeker only optimizes for minimizing its distance to the hider.

## I. INTRODUCTION

In an interactive task between a human and a robot, it is beneficial for the robot to reason about the human’s high-level behavior strategy. For instance, if the human and the robot are competing, inferring the human’s strategy can give the robot an advantage over the human competitor. During human-robot collaboration, if the robot could infer a human partner’s strategy by observing their actions, precise communication between the human and the robot would not be as critical, helping make the collaboration seamless.

Prior work has typically represented a human’s strategy by a policy that maps the state of the world to the human’s actions [14] and investigated ways for robots to infer this strategy in fully observable settings [13], [18]. Typically, the robot uses passive observations of the human’s actions to solve this inference problem (e.g., [6], [9]), putting more emphasis on how humans may model the robot in the interactive setup than on how robots can leverage information about the environment to better infer the human’s strategy. In scenarios with limited observability among the interactants, though, we suspect that more explicit reasoning about human strategies in relation to the environment could facilitate better reasoning about the human’s behavior.

In this work, we hypothesize that robots should reason about *Critical Decision Points* when trying to infer a human’s

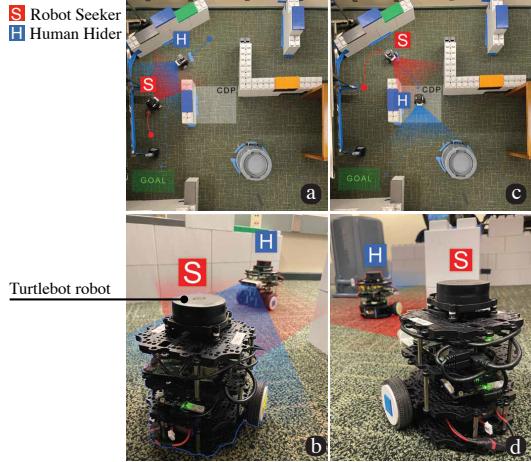


Fig. 1. The robot seeker (S) influences the human hider (H) towards a Critical Decision Point (CDP) in a hide-and-seek game: a), b) The human-controlled hider sees the robot seeker on its way to its goal while following its high-level strategy. c) The robot seeker actively influences the human-controlled hider to escape towards a CDP (gray area). d) When the hider is at the CDP, the robot seeker optimizes for not being seen by the hider. So, the hider continues to navigate toward its goal. The robot seeker sees the hider take action toward its goal and identifies its strategy.

strategy in an interactive task. These Critical Decision Points are states in the physical environment where human actions are particularly revealing about their high-level strategy.

The distribution of Critical Decision Points depends on the complexity of the shared environment and the possible human strategies for the task of interest. In some cases, most human strategies may result in different actions for many states of the environment, making most of the states Critical Decision Points. In turn, one would expect passive observation of the human to reveal their strategy quickly. In other cases, human strategies may result in the same action across many states, leading to few Critical Decision Points.

Our work investigates the value of Critical Decision Points when they are sparsely located across a partially-observable environment, as these are challenging situations for inferring a human’s strategy. Our main hypothesis is that in these cases, an active approach in which the robot influences the human toward Critical Decision Points is particularly beneficial. Thus, we propose a method for the robot to do this based on Receding Horizon Planning (RHP) [10] and demonstrate it in a partially-observable setting through a hide-and-seek game between a human and a robot (Fig. 1). In this game, the seeker robot tries to catch a human-controlled hider by estimating their strategy.

\*Authors Contributed Equally

The authors are with the Department of Computer Science, Yale University, New Haven, CT, USA

Our experiments in simulation and our demonstration in the real-world show that when the robot seeker influences the hider toward Critical Decision Points, the seeker can catch the hider faster than when the robot tries to estimate the human’s strategy while randomly exploring the map or minimizing the distance between itself and the hider. Overall, our findings suggest that robots can leverage the properties of a shared environment in interactive human-robot tasks in order to estimate a human’s strategy under partial observability.

## II. RELATED WORK

**Robots Influencing Humans.** In recent years, there has been a growing interest in enabling robots to select actions that influence human behaviors [22] in both collaborative and competitive scenarios. A common approach is to explicitly model the human’s reward [1] and then optimize for robot actions that, at least in the near future, steer the human towards states where they are more likely to act in a desired manner [16]. More recently, high-level strategies of humans have been inferred from repeated interactions using unsupervised learning, with the inferred strategies being used to influence their behavior [14], [23]. Prior work has also leveraged offline reinforcement learning to train robots to influence opponent behavior more optimally [8], [22]. Building upon this line of research, we frame the problem of influencing the human toward Critical Decision Points as a Receding Horizon Planning problem [10]. This enables an observing robot to select actions that drive a human toward Critical Decision Points, where their actions are more revealing about their strategy than in other states of the environment.

**Opponent Modeling.** While we believe that Critical Decision Points can be useful in both collaborative and competitive settings, our work demonstrates their value in competitive settings specifically. In competitive settings, opponent modeling is typically key. That is, agents build a model of an opponent’s behavior using observations of their actions [11], [18], [25]. Over the years, opponent modeling has been extensively studied for applications such as playing stochastic games between agents [20], negotiations between agents [12] and for cooperative tasks [21], [24]. For human-robot interaction tasks, prior work uses Theory of Mind [13], [15], [20] to recursively reason about the human’s actions based on the robot’s behavior. Some methods estimate a human’s reward model [2], [3] and then use the reward model to estimate optimal robot behavior via reinforcement learning techniques. Building on these ideas, we use a Bayesian approach to recursively estimate the likelihood of a human’s strategy in a competitive setting.

## III. METHOD

**Problem Definition.** In this work, we are concerned with enabling a robot to identify a human’s high-level strategy in a shared environment. We assume that a human strategy is represented by a policy,  $\pi_i$ , which maps states to actions. Also, the robot has access to a policy bank  $\mathcal{P}$  with a set of reasonable policies  $\pi_1, \pi_2, \dots, \pi_n$  that the human may

follow to achieve their objective(s) in the environment. We represent the full state of the environment,  $s = (s_w, s_h, s_r)$ , as a combination of the world state  $s_w$ , which contains information about the physical properties of the environment in a discrete representation (ex. occupancy map), the state of the human  $s_h$ , and the state of the robot  $s_r$ . We assume that the robot always has access to  $s_w$  and  $s_r$ , but can only reason about  $s_h$  through observations of the human’s behavior.

Given the above setup, the robot’s goal is to identify the best strategy from the policy bank that most accurately represents the human’s observed behavior in a partially observable environment. The robot keeps track of a belief, or a probability distribution, over the policies in  $\mathcal{P}$ , the maximum of which is the policy the robot believes the human is most likely following, denoted as  $\hat{\pi}_h$ . Another important assumption we make is that the human’s actual strategy,  $\pi_h$ , remains constant during an interaction.

**Critical Decision Points.** The main insight of our work is that there are states in the environment in which observing a human take an action can reveal important information about their high-level strategy. We refer to those states as Critical Decision Points ( $S_{crit}$ ), for which the divergence in the human’s potential actions (as per the policy bank  $\mathcal{P}$ ) is maximized. Mathematically, let the environment state  $s_w$  be represented by a grid, with each grid cell having semantic information about that specific location in the world. For example, if the grid is 2-dimensional, then each cell can contain occupancy information. If the grid is higher-dimensional, then more or alternative semantic features about the environment can be included in the representation. Then, for each possible grid cell  $s_w^{(i,j)}$  in the environment state, we compute ( $S_{crit}$ ) as:

$$S_{crit} = \{s_w^{(i,j)} \mid \text{strategy\_div}(\mathcal{P}, s^{(i,j)}) > \theta\}, \quad (1)$$

where  $s^{(i,j)}$  represents the full state of the world with the human agent being located at the grid cell  $s_w^{(i,j)}$ ,  $\text{strategy\_div}(\mathcal{P}, s^{(i,j)})$  measures the divergence in the strategies of the policy bank  $\mathcal{P}$ , and  $\theta$  is a threshold for the divergence. Depending on the task,  $\text{strategy\_div}()$  can be calculated as the count of the number of actions differing at a given state or quantified as some similarity measure of the next states. The central hypothesis of our work is that at Critical Decision Points, the human’s actions will be more revealing about their strategy to an observing robot than other states in the environment, as shown in Fig. 2.

**Influencing the Human Towards Critical Decision Points.** We develop a method for the robot to drive a human towards Critical Decision Points, such that the robot can reduce the entropy of its belief over the human’s current strategy, given the set of reasonable human policies in the policy bank  $\mathcal{P}$ .

We frame the above problem as a discrete-time planning problem, which we solve using a Receding Horizon Planning (RHP) approach [10]. RHP takes inspiration from Model Predictive Control [5] but concerns discrete decision-making rather than continuous control. Generally, to perform RHP, an agent iteratively solves a planning problem over a receding

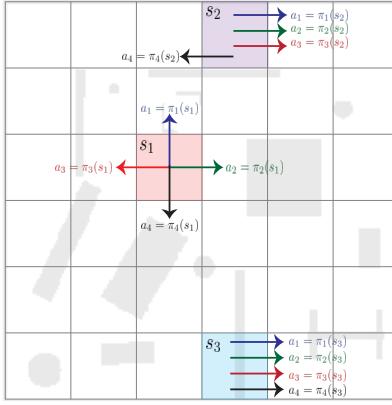


Fig. 2. **Critical Decision Point Computation.** For every policy of the human in the policy bank, we compute what states in the environment produce the most divergent actions. For instance, if there were four reasonable policies the human could be following, at  $s_1$ , all four of them produce different actions, making  $s_1$  a Critical Decision Point (as compared to  $s_2$  and  $s_3$  where most policies predict the same action).

horizon, considering possible future actions by expanding a search tree. After planning on a given time step, the first action leading to the best possible outcome is executed, the horizon window moves forward, and planning is performed again for the next possible futures.

Alg. 1 summarizes our approach for a robot to influence a human towards Critical Decision Points. At a given time  $t$ , the robot rolls out all possible future actions that can be taken by the human per the policy bank  $\mathcal{P}$  along with its own responses to the human's possible actions. These steps are repeated to grow the search tree to a certain horizon  $\mathcal{D}$ . For every leaf node in the tree that results from the human acting according to a policy  $\pi_i$ , the robot computes a cost  $\mathcal{C}_{\pi_i}(s^t)$  that describes how good that specific future is based on 1) the distance between the projected human position at time  $t+\mathcal{D}$  and the closest Critical Decision Point ( $s_{crit}$ ) and 2) the satisfaction of other task-related objectives:

$$\mathcal{C}_{\pi_i}(s^t) = (\lambda_1 \cdot dist(s_h^{t+\mathcal{D}}, s_{crit}) + \lambda_2 \cdot C_{task}) \frac{1}{bel(\pi_i^t)} \quad (2)$$

where  $dist(s_h^{t+\mathcal{D}}, s_{crit})$  is the shortest path between the human and the closest Critical Decision Point  $\mathcal{D}$  steps in the future when the human acts according to  $\pi_i$  and the robot takes corresponding actions,  $C_{task}$  is a task-specific cost function, and  $\lambda_1, \lambda_2$  are weights to balance the two objectives. The cost  $\mathcal{C}$  is scaled by the likelihood of the human policy  $bel(\pi_i^t)$ , such that less likely policies result in higher cost. We compute  $bel(\pi_i^t)$  recursively over time with a Bayesian update, similar to a Bayes Filter [17] with an identity transition function as we assume that the human does not change the actual underlying strategy  $\pi_h$  during an interaction:

$$\begin{aligned} bel(\pi_i^t) &= P(\pi_i^t | s^1 \dots s^t) \propto P(s^t | \pi_i) bel(\pi_i^{t-1}) \\ \implies bel(\pi_i^t) &= P(s^t | \pi_i) bel(\pi_i^{t-1}) \end{aligned} \quad (3)$$

where  $bel(\pi_i^{t-1})$  is the prior belief at time  $t-1$ , and the probability of a state given a human policy,  $P(s^t | \pi_i)$ , is

---

**Algorithm 1:** Influencing Humans to Critical Decision Points using Receding Horizon Planning (RHP)

---

**Input :** Policy bank  $\mathcal{P} = \{\pi_1, \pi_2, \dots, \pi_n\}$ , Full state of the environment  $s = (s_w, s_h, s_r)$ , Weights  $\lambda_1, \lambda_2$ , Action space for robot  $\mathcal{A}_r$ , Task-specific cost function  $C_{task}$ , Planning Horizon  $\mathcal{D}$

**Output :** Actions taken by the robot that influence the opponent  $a_r$

```

1 Initialize  $bel(\pi_i)$  over policy bank  $\mathcal{P}$  with uniform distribution
2 Identify Critical Decision Points  $S_{crit}$  for each environment cell in  $s_w$  based on divergence  $\theta$  as in 1
3  $S_r = [s_r^{init}]$ 
4  $a_r^1 = a_r^{init}$ 
5 for  $t = 1, \dots, T$  do
6   Execute  $a_r^t$  while human executes  $a_h^t$ 
7   Observe  $s_h^t$ 
8   for  $\pi_i \in \mathcal{P}$  do
9      $bel(\pi_i^t) = P(\pi_i^t | s^1 \dots s^t) \propto P(s^t | \pi_i) bel(\pi_i^{t-1})$ 
10    for  $d = 0, \dots, \mathcal{D}$  do
11       $t' = t + d$ 
12       $a_h^{t'} = \pi_i(s_h^{t'})$ 
13       $s_h^{t'+1} = execute\_action(s_h^{t'}, a_h^{t'})$ 
14       $S_r^{t'+1} = []$ 
15      for  $a_r^{t'} \in \mathcal{A}_r$  do
16        for  $s_r^{t'} \in S_r^{t'}$  do
17           $s_r^{t'+1} = execute\_action(s_r^{t'}, a_r^{t'})$ 
18           $S_r^{t'+1}.add(s_r^{t'+1})$ 
19        end for
20      end for
21       $S_r^{t'} \leftarrow S_r^{t'+1}$ 
22    end for
23     $\mathcal{C}_{\pi_i}(s^t) = (\lambda_1 \cdot dist(s_h^{t+\mathcal{D}}, s_{crit}) + \lambda_2 \cdot C_{task}) \cdot \frac{1}{bel(\pi_i^t)}$ 
24  end for
25  Choose  $a_r^*$  as the robot action that responds to the predicted human policy  $\hat{\pi}_h$  at time  $t$ , such that  $\hat{\pi}_h = \arg \min_{\pi_i \in \mathcal{P}} \mathcal{C}_{\pi_i}(s^t)$ .
26   $a_r^t = a_r^*$ 
27 end for

```

---

computed empirically based on state visitation frequencies of the human when they are following the given policy.

Finally, the robot selects its best response action  $a_r^*$  at time  $t$  as the action that minimizes the cost function  $\mathcal{C}_{\pi_i}(s^t)$  for all possible  $\pi_i \in \mathcal{P}$  over the horizon  $\mathcal{D}$ . This allows it to influence the human toward Critical Decision Points, because the cost is dependent on where the human ends up being after  $\mathcal{D}$  steps in the future.

## IV. EVALUATION

### A. Task and Experimental Setup

**Task:** We use the game of hide and seek as our experimental setup. The game warrants the construction of arbitrarily complex environments, requires the agents to take a large variety of environment-dependent strategies, and requires that agents not communicate about their strategies.

We developed the hide-and-seek task in a photo-realistic simulation and the real world. In both settings, we embodied the agents with Turtlebot3 robots such that there is no

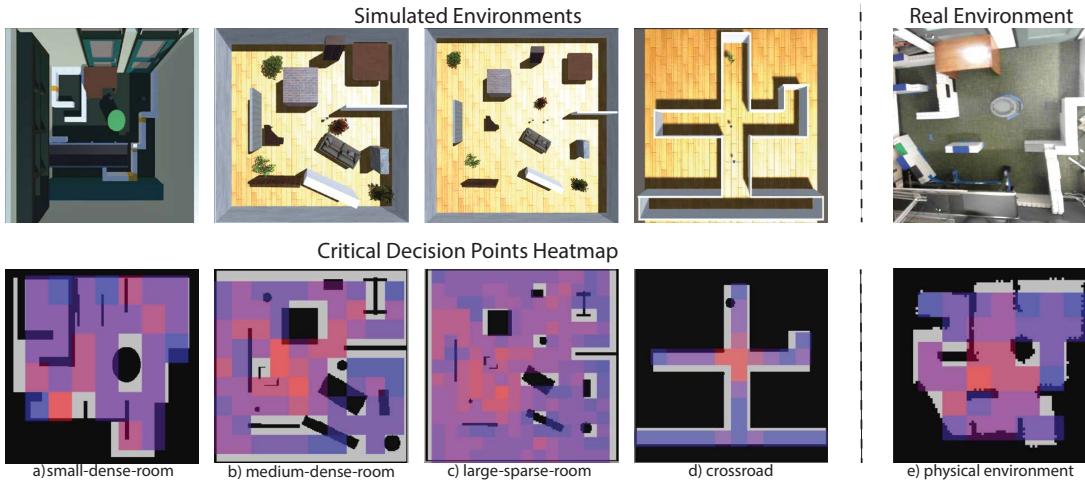


Fig. 3. (Top) Photo-realistic Simulated and real-world environments for the hide-and-seek task. (Bottom) Heatmaps of Critical Decision Points for corresponding environments. States closer to red denote the most Critical Decision Points and states closer to blue denote the least critical points. Note: Even though the figure shows Critical Decision Points for every state, typically, it is sufficient to compute them in a small region around the robot.

advantage due to different agent morphology in the game. The robots have a forward-facing camera, giving access to a first-person view of the environment. The placement of objects in the environment and agents' access to visual information through their cameras makes the hide-and-seek game partially observable for both agents.

The photo-realistic simulation environments were created by modifying the SEAN 2.0 simulator [19] to support multiple robots. In simulation, an autonomous agent assumes the role of the seeker, while a simulated human controls the hider. For real-world experiments, we setup a lab environment similar to one of the simulated worlds. One of the researchers controlled the hider robot in this case, while the seeker robot made decisions autonomously.

**Environments:** We constructed four environments in simulation, as shown in the top half of Fig. 3. The first environment is a **small-dense-room** with several objects densely placed in a small room. The second environment is a **medium-dense-room** with objects spread across the map. The third is a **large-sparse-room**, which is sparse and has the same objects as **medium-dense-room** but wider open spaces. The last one is the **crossroad** environment, with four hallways emanating from a single crossroad and distinct objects or environmental geometries at the end of the hallways. The laboratory environment for physical experiments is similar to the **small-dense-room**.

**Policy Bank:** We model the various strategies of the human-controlled hider using behavior trees [4]. Each behavior tree takes the hider to various hiding locations within a given map, determined by the presence of particular objects in those locations or locations with different environmental occlusions. If the hider were to spot the seeker while navigating to a pre-defined location, the behavior tree would enable the hider to perform a fixed evasive maneuver to escape the seeker's field of view. Each behavior tree was executed multiple times with different starting locations on the map for both the hider and the seeker to collect a dataset of states and actions for each behavior of the hider. To construct the

hider's policy bank, each policy was trained using the state-action trajectories in the corresponding dataset with a Long Short-Term Memory network [7] and behavioral cloning [1].

### B. Implementation Details

The map of the shared environment is discretized into a  $n \times n$  grid, and each map cell is represented as a node in an undirected graph. Each node in the environment graph contains information about the center position of the cell, the objects present in the cell, and a visibility score of that cell, computed from every other cell in the map using ray casting in a raster grid. The state space  $S_H$  of the hider agent is 6-dimensional, containing  $(x_h, y_h, \theta_h, \hat{x}_s, \hat{y}_s, \hat{\theta}_s)$ , where  $x_h, y_h$  denote the position of the hider,  $\theta_h$  is their heading,  $\hat{x}_s, \hat{y}_s$  is the estimated position of the seeker by the hider, and  $\hat{\theta}_s$  is the estimated heading of the seeker by the hider. To obtain the perceived position and heading of the hider, we first train a real-time object detection model to predict a bounding box and a heading estimate for a given robot when it is visible in the first-person view of another robot. Then, based on the robot's known physical size, the predicted bounding box in the image, and the camera parameters, we use a pinhole camera model to estimate the relative node position of the hider w.r.t. the seeker. The action space of the seeker consists of transitions to each adjacent cell from an inhabited cell.

To compute Critical Decision Points, for every node in an environment we predict the actions that the hider would take per a given policy,  $\pi_i \in \mathcal{P}$ , assuming that the hider cannot see the seeker. This results in a map of Critical Decision Points, as shown in the bottom half of Fig. 3. The map shows that the environments and the policy bank created for the hide-and-seek task have afforded the existence of sparsely located Critical Decision Points.

The seeker jointly optimizes to drive the hider toward the closest Critical Decision Point (with  $\theta \geq 4$  in Eq. 1) in a fixed radius around the hider's current position while minimizing its distance to the hider ( $C_{task}$  in eq. 2). Additionally, the

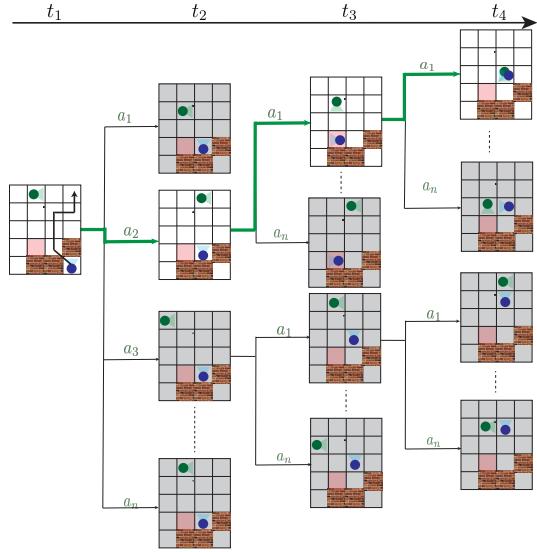


Fig. 4. **Example RHP tree rollout for influencing the robot:** The green and the blue agent are playing a hide-and-seek game, where the green agent (the seeker) is trying to influence the blue agent (the hider) to a Critical Decision Point (the red cell) using RHP to infer its policy and therefore catch it faster. The brick walls represent obstacles. The green agent imagines all possible future actions, assuming the blue agent follows a fixed policy, and selects the most optimal action in the next timestep to lead to the most favorable outcome at horizon  $D = t_4$ . The black arrow at  $t_1$  represents the underlying strategy. Both agents' fields of view are restricted to all the cells in the direction they face. At  $t_2$ , when the blue agent detects the green agent, it abandons its underlying strategy at  $t_3$  to perform an evasive maneuver, then resumes its original path once out of the green's field of view at  $t_4$ . This allows the green agent to observe the blue agent's underlying policy at a critical decision point at  $t_3$  without being observed, and ultimately catch it in  $t_4$ . Note that the green arrow denotes the optimal next action for the green agent. After rolling out all possible futures, the green agent determines its optimal sequence of actions to be  $a_2 \rightarrow a_1 \rightarrow a_1$  (highlighted in white). Since the green agent performs RHP, it takes the first action  $a_2$  from the sequence and constructs the tree from the next state.

seeker optimizes for staying out of the hider's field of view when it approaches a Critical Decision Point using RHP.

Whenever the hider is visible, the seeker expands a tree of all possible future actions it could take in response to a hider following a given policy up to a certain horizon ( $D = 5$ ). Concurrently, the seeker performs a Bayesian update on the belief of the hider's policies, given observations of the hider, to compute the likelihood of each policy being executed from the policy bank. Finally, this belief is used to evaluate the cost function for each branch in the tree. Fig. 4 shows an example of tree expansion. Note that if the hider has been seen in the past 4 ( $D - 1$ ) timesteps and is not currently visible, the seeker uses the stored action queue from the RHP tree in anticipation of spotting the hider in the future if its belief over the policy currently being followed by the hider is correct. Once no more stored actions are in the queue, the autonomous seeker reverts to a default exploration strategy where the seeker chooses an object on the map according to the policy it is following and navigates toward it.

### C. Experiments

We evaluate our method in simulation and demonstrate our approach in a real-world environment.

**Simulation:** We measure the effectiveness of our approach by comparing it with two baselines:

- 1) **Baseline 1:** The seeker randomly explores the environment and performs a Bayesian update of the belief over the hider's policies whenever it is visible (per eq. 3).
- 2) **Baseline 2:** The seeker uses RHP to optimize for minimizing its distance between itself and the estimated position of the hider whenever it is visible while computing the Bayesian update (per eq. 3).
- 3) **Our Method:** The seeker selects actions by expanding the RHP tree to drive the hider towards the closest Critical Decision Point per eq. (2), and estimates their strategy using the Bayesian update (per eq. 3).

For our simulated experiments, we assume that a simulated human controls the hider and follows one of the policies from the policy bank during an interaction episode. Depending on the environment, the hider's policy bank contains between 6 and 8 policies. We run each experiment for 20 episodes or trials, each of which ends when the seeker lands on the same node as the hider (indicating it has caught the hider) or after 100 time steps. To compare our method's performance with the baselines, we compute the average number of time steps needed for the seeker to catch the hider in each environment aggregated along all ground truth policies.

**Real World:** We demonstrate the applicability of our method in the real-world environment (shown in Fig. 1). We used Simultaneous Localization and Mapping to generate a map of the physical environment. Like the simulation environments, we discretized the map to create an undirected graph with the states as nodes and valid actions from the states as edges. Then, we developed the policy bank of the hider using the behavior trees and generating policies with behavioral cloning. One of the researchers controlled the hider robot through velocity commands using a PlayStation controller while following a given policy from the policy bank. The seeker robot selected actions optimized for driving the opponent toward the Critical Decision Points.

### D. Results

Fig. 5 shows that our method outperformed the baselines across all four simulation environments. When the seeker influenced the hider toward Critical Decision Points, it caught the hider faster than when it explored the map (Baseline 1) or minimized its distance to the hider (Baseline 2).

Qualitatively, we observed that with Baseline 1, when the seeker randomly traversed the map and updated its belief of the hider's policy with each sighting, the seeker could not accurately estimate the hider's strategy. In Baseline 2, when the seeker used RHP to minimize its distance to the perceived position of the hider, it frequently entered the hider's field of view, making the hider trigger a fixed evasive maneuver. The seeker's estimate of the hider's strategy then became inaccurate. With our method, the seeker entered the hider's field of view to trigger an evasion and then exited it to observe the hider at the Critical Decision Point continue to its original goal. It then selected actions to intercept the hider quickly once it learned the hider's strategy.

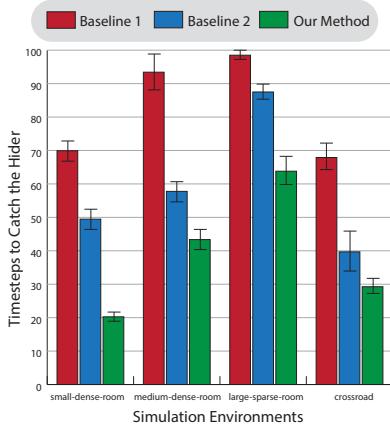


Fig. 5. Mean and Standard error of timesteps taken by the seeker to catch the hider for each experiment aggregated across all hider policies for a given environment and number of trials (lower is better)

We demonstrated our method on the real robot across five trials each for six policies, as illustrated in Fig. 1. A researcher controlled the hider robot, with access to only the map of the environment and the first-person view from the robot’s camera. Our method took the seeker  $27 \pm 3$  timesteps to catch the hider. Baselines 1 and 2 took the seeker  $39 \pm 2$  timesteps and  $24 \pm 1$  timesteps longer than our approach to catch the hider, respectively. The supplementary video shows an example trial of the real-world demonstration.

## V. DISCUSSION

In this work, we developed a novel method for a robot to identify a human’s strategy with limited observability of human actions. Assuming the robot has access to reasonable policies a human might follow, the robot examines the actions these policies predict at all states in the environment. Then, by observing humans at the states most policies diverge, which we call Critical Decision Points, the robot can more quickly predict which of the strategies the human is using in the environment. We demonstrated our approach in a game of hide-and-seek in simulation and the real world, assuming that the human’s strategy remained unchanged throughout a single game. Our results suggested that influencing a human-controlled hider toward Critical Decision Points can help an autonomous robot infer the human’s strategy faster in the game. In the future, we aim to relax the assumption that the human’s strategy is constant and sampled from a fixed policy bank.

## ACKNOWLEDGMENT

This work was funded by the National Science Foundation (NSF) under grants No. 1955653 and 2106690 and the Office of Naval Research (ONR) under grant No. N00014-24-1-2124. The authors thank Drazen Brzicic, Shasvat Desai, Nathan Tsoi, Sasha Lew and Cameron Adams for their assistance and feedback to improve this work.

## REFERENCES

- [1] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004.
- [2] T. Bonjour, M. Haliem, A. Alsalem, S. Thomas, H. Li, V. Aggarwal, M. Kejriwal, and B. Bhargava. Decision making in monopoly using a hybrid deep reinforcement learning approach. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(6), 2022.
- [3] T. V. Bui, T. Mai, and T. H. Nguyen. Imitating opponent to win: Adversarial policy imitation learning in two-player competitive games. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, pages 1285–1293, 2023.
- [4] M. Colledanchise and P. Ögren. *Behavior trees in robotics and AI: An introduction*. CRC Press, 2018.
- [5] C. E. Garcia, D. M. Prett, and M. Morari. Model predictive control: Theory and practice—a survey. *Automatica*, 25(3):335–348, 1989.
- [6] D. Ghose, M. A. Lewkowicz, K. Gezahegn, J. Lee, T. Adamson, M. Vázquez, and B. Scassellati. Tailoring visual object representations to human requirements: A case study with a recycling robot. In *Conference on Robot Learning*, pages 583–593. PMLR, 2023.
- [7] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [8] J. Hong, S. Levine, and A. Dragan. Learning to influence human behavior with offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.
- [9] H. Karnan, G. Warnell, X. Xiao, and P. Stone. Voila: Visual-observation-only imitation learning for autonomous navigation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2497–2503. IEEE, 2022.
- [10] X. Ma and D. A. Castanon. Receding horizon planning for dubins traveling salesman problems. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 5453–5458. IEEE, 2006.
- [11] S. Nashed and S. Zilberstein. A survey of opponent modeling in adversarial domains. *Journal of Artificial Intelligence Research*, 73:277–327, 2022.
- [12] Z. Nazari, G. M. Lucas, and J. Gratch. Opponent modeling for virtual human negotiators. In *Intelligent Virtual Agents: 15th International Conference, IVA 2015, Delft, The Netherlands, August 26–28, 2015, Proceedings 15*, pages 39–49. Springer, 2015.
- [13] I. Oguntola, J. Campbell, S. Stepputis, and K. Sycara. Theory of mind as intrinsic motivation for multi-agent reinforcement learning. *arXiv preprint arXiv:2307.01158*, 2023.
- [14] S. Parekh and D. P. Losey. Learning latent representations to co-adapt to humans. *Autonomous Robots*, pages 1–26, 2023.
- [15] N. Rabinowitz, F. Perbet, F. Song, C. Zhang, S. A. Eslami, and M. Botvinick. Machine theory of mind. In *International conference on machine learning*, pages 4218–4227. PMLR, 2018.
- [16] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan. Planning for autonomous cars that leverage effects on human actions. In *Robotics: Science and systems*, volume 2. Ann Arbor, MI, USA, 2016.
- [17] S. Thrun, W. Burgard, and D. Fox. Probabilistic robotics. *Kybernetes*, 35(7/8):1299–1300, 2006.
- [18] Y. Tian, K.-R. Kladny, Q. Wang, Z. Huang, and O. Fink. Multi-agent actor-critic with time dynamical opponent model. *Neurocomputing*, 517:165–172, 2023.
- [19] N. Tsoi, A. Xiang, P. Yu, S. S. Sohn, G. Schwartz, S. Ramesh, M. Hussein, A. W. Gupta, M. Kapadia, and M. Vázquez. Sean 2.0: Formalizing and generating social situations for robot navigation. *IEEE Robotics and Automation Letters*, 7(4):11047–11054, 2022.
- [20] F. B. Von Der Osten, M. Kirley, and T. Miller. The minds of many: Opponent modeling in a stochastic game. In *IJCAI*, 2017.
- [21] W. Z. Wang, M. Beliaev, E. Biyik, D. A. Lazar, R. Pedarsani, and D. Sadigh. Emergent prosociality in multi-agent games through gifting. *arXiv preprint arXiv:2105.06593*, 2021.
- [22] W. Z. Wang, A. Shih, A. Xie, and D. Sadigh. Influencing towards stable multi-agent interactions. In A. Faust, D. Hsu, and G. Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 1132–1143. PMLR, 08–11 Nov 2022.
- [23] A. Xie, D. Losey, R. Tolsma, C. Finn, and D. Sadigh. Learning latent representations to influence multi-agent interaction. In *Conference on robot learning*, pages 575–588. PMLR, 2021.
- [24] J. Yang, A. Li, M. Farajtabar, P. Sunehag, E. Hughes, and H. Zha. Learning to incentivize other learning agents. *Advances in Neural Information Processing Systems*, 33:15208–15219, 2020.
- [25] X. Yu, J. Jiang, W. Zhang, H. Jiang, and Z. Lu. Model-based opponent modeling. *Advances in Neural Information Processing Systems*, 35:28208–28221, 2022.