Accepted for publication in IEEE International Conference on Robotics and Automation (ICRA), May 2022

Human-Guided Motion Planning in Partially Observable Environments

Carlos Quintero-Peña*, Constantinos Chamzas*, Zhanyi Sun, Vaibhav Unhelkar and Lydia E. Kavraki

***Abstract*— Motion planning is a core problem in robotics, with a range of existing methods aimed to address its diverse set of challenges. However, most existing methods rely on complete knowledge of the robot environment; an assumption that seldom holds true due to inherent limitations of robot perception. To enable tractable motion planning for high-DOF robots under partial observability, we introduce BLIND, an algorithm that leverages human guidance. BLIND utilizes inverse reinforcement learning to derive motion-level guidance from human critiques. The algorithm overcomes the computational challenge of reward learning for high-DOF robots by projecting the robot's continuous configuration space to a motion-planner-guided discrete task model. The learned reward is in turn used as guidance to generate robot motion using a novel motion planner. We demonstrate BLIND using the Fetch robot and perform two simulation experiments with partial observability. Our experiments demonstrate that, despite the challenge of partial observability and high dimensionality, BLIND is capable of generating safe robot motion and outperforms baselines on metrics of teaching efficiency, success rate, and path quality.**

I. Introduction

Planning robot motion is a long-standing problem in robotics [1]. Most existing solutions require complete knowledge of the robot environment. At the same time, despite tremendous advances, robot perception remains susceptible to occlusions, limited field of view, and measurement noise. For robust deployment of robots in homes, hospitals, and other unstructured domains, techniques for addressing the challenge of motion planning with partial information are essential. As a representative example of this challenge, consider the robot shown in Fig. 1. It is tasked with retrieving the green object placed inside the partially observable blue box. When planning its motion from a starting configuration outside the box to the green object, planners that do not reason about partial observability are likely to return an infeasible trajectory (e.g., Fig. 1b).

As detailed in Sec. II, multiple solutions exist to plan robot motion in this and similar partially observable settings. Broadly, these solutions either assume nominal shapes of occluded objects are known a priori [3], [4] or reason over the combinatorial set of possible object locations [5], [6]. In practice, however, prior knowledge of object shapes may not be available and, due to the curse of dimensionality, reasoning over the combinatorial set of unobservable objects can become computationally expensive. Moreover, most existing approaches do not guarantee collision-free trajectories and, thus, can lead to unsafe robot behavior.

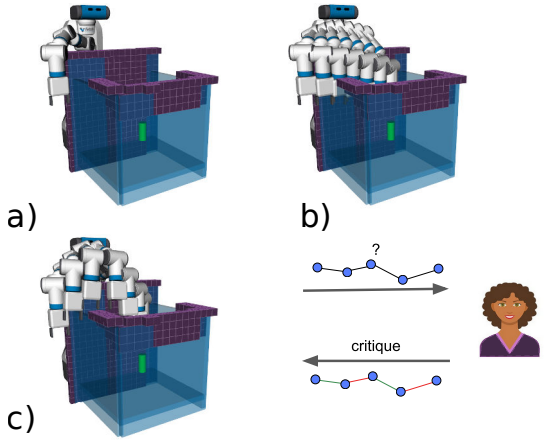

Fig. 1. The Fetch robot is tasked with picking up the green object inside the blue box. Due to sensing limitations, the robot can only obtain a partial view of the box. The robot can see the opaque voxels but not the transparent faces. **a)** The robot starts with its arm outside the box. **b)** Due to partial observability, a motion planner (such as [2]) produces a trajectory that does not collide with the opaque voxels but does so with the unobservable portion the box. **c)** Our approach, BLIND, is capable of producing a safe trajectory despite partial observability, by incorporating guidance from the human user.

Informed by these challenges, we explore an alternate insight, namely, *learning from human guidance*. Our choice is motivated by two reasons: humans can help augment robot perception and, importantly, prevent the execution of unsafe motion. To realize this insight, we introduce a human-in-the-loop algorithm for motion planning. Our algorithm, **B**ayesian **L**earning **IN** the **D**ark (BLIND), combines inverse reinforcement learning (IRL) with motion planning in a novel way to efficiently plan motion of high degree-of-freedom (DOF) robots in partially observable environments (Fig. 1c).

BLIND builds upon recent research on robot learning from human teachers [7]–[9]; however, in contrast to prior art, it focuses on human-guided learning for *high-DOF robots* and enables efficient *human-guided motion planning under partial observability* of the physical environment. To ensure that human end-users can effectively provide guidance, BLIND learns from *critiques of robot motion*–a segmentation of trajectory snippets into good and bad. This is in contrast to alternate forms of human guidance, such as demonstrations and corrections, where explicit examples of trajectories are provided to the robot–a task known to be challenging for humans when working alongside high-DOF robots [7], [10].

Our core contribution is to incorporate these inputs in a computationally tractable and sample efficient manner for the challenging problem of high-DOF motion planning under

*Authors contributed equally and are ordered randomly.

All authors are affiliated with the Department of Computer Science, Rice University, Houston TX, USA {carlosq, chamzas, zs19, vaibhav.unhelkar, kavraki}@rice.edu. This work was supported in part by NSF 2008720, NSF-GRFP 1842494, NSF 1718478 and Rice University Funds.

partial observability. BLIND achieves this by performing reward learning on an algorithmically-generated graph and utilizing this learned reward for high-DOF motion planning using a novel human-guided variant of TrajOpt [2].

Through extensive simulation evaluations and example robot demonstrations detailed in Sec. VI, we verify BLIND's capability to find safe robot trajectories in partially observable environments by utilizing human guidance. We observe that BLIND achieves a higher success rate and requires fewer user queries than an extension of prior art [9] to high-DOF robots, thereby emphasizing the need for novel approaches for incorporating human guidance to solve high-dimensional robotic problems. Our ablation studies reinforce the importance of BLIND's key design decisions for incorporating human guidance while reasoning about a high-DOF configuration space. Encouraged by these results, we believe that BLIND will provide users an effective way to convey their preferences and serve as a building block to incorporate novel forms of human guidance for high-DOF robot learning.

II. Related Work

Before describing our solution, we review research related to our problem setting and solution approach.

A. Motion Planning in Partial Observable Environments

In the real world, robots often need to reason under partial observability. Partially observable Markov decision processes (POMDPs) provide a principled approach to generate robot behavior in these settings [5]. While POMDPs are computationally challenging to solve, continued advances in POMDP solvers (such as [11]) have made them exceedingly tractable and attractive for robotics. However, POMDP solvers largely cater to discrete state and action spaces and, thus, do not readily extend to motion planning for high-DOF robots.

For tractable planning in continuous spaces, specialized methods that assume measurement noise to be of a specific functional form (e.g., Gaussian) have been developed [12]. Related to these works, robust extensions of sampling- and optimization- based motion planners have been developed to compute paths robust to sensing uncertainties. These methods, too, either assume specific models of measurement noise [13], [14] or require access to samples from the noise distribution [4]. Critically, these methods require a priori knowledge of the geometry of partially observable objects; an assumption that might not hold in practice.

Closer to our work are methods that perform planning in partially observable environments through specialized information gathering actions, namely, contact information [15], [16]. In these approaches, planning and execution are interleaved to find valid paths based on contact feedback. However, these methods assume that the robot can make contact with the environment. For safety-critical applications, unintended robot contact can be unsafe or highly undesirable (e.g., colliding with a human or breaking a glass). In contrast, our method leverages another form of information gathering action – guidance from the human end-user – and, thus, avoids the need to execute potentially unsafe trajectories.

B. Learning from Human Guidance

Approaches for robot learning from human guidance have also received significant attention in the last two decades, with Learning from Demonstrations (LfD) being the predominant paradigm [7], [8], [17]. While LfD-based approaches have been used to teach motion skills to robots, providing demonstrations for high-DOF robots can be difficult for end-users [7]. To reduce the burden of providing optimal demonstrations, more recent approaches have explored alternative forms of human guidance such as preferences, correction, and critiques [9], [10], [18]–[24]. Among these, we focus on critiques.

Argall et al. were the first to explore the use of critiques to learn robot behavior [25]. Using a pre-programmed or learned robot reward, their approach first generates examples of robot behaviors. These behaviors are then critiqued by a human as either good or bad and the critiques are then used to refine the robot reward and behavior. More recently and closest to our approach, [9] provide a Bayesian extension of this framework. However, [9], [25] consider problems with discrete state and action spaces and do not focus on motion planning in continuous spaces. Our approach builds upon these frameworks and, in contrast, enables safe high-DOF motion planning by utilizing human critiques.

Our approach also complements existing reward learning approaches from human guidance (demonstrations, corrections, or preferences) that focus on discrete problem spaces [10], [20], [26]. A common assumption in these related techniques, which abstract away continuous robot motion, is that reward learned in a discrete problem space (e.g., grid world in the end-effector space) using human guidance can be readily translated to execute continuous robot motion. Our experiments demonstrate that this translation is, in fact, non-trivial and that it can be effectively achieved using BLIND. Lastly, human guidance has also been explored in the context of motion planning as heuristics for search-base or sampling-based planning [27]–[29]. However, these methods do not consider partial observability or incorporate robot learning.

III. Preliminaries

A. Optimization-based Motion Planning: TrajOpt

Optimization-based motion planners compute locally optimal trajectories by optimizing a cost function over the trajectory while ensuring collision-free motions [2], [30]. In TrajOpt [2], for instance, collision avoidance is achieved by keeping a positive signed distance between robot links and obstacles in the workspace. Other behaviors such as joint limits, dynamics, and constraints can be incorporated as additional terms in the following optimization formulation:

$$\begin{aligned} \underset{q}{\text{minimize}} \quad & f(q), \\ \text{subject to} \quad & g_i(q) \leq 0,\ i = 1, \ldots, m, \\ & h_i(q) = 0,\ i = 1, \ldots, n. \end{aligned}$$

Here, f usually encourages smoothness; g_i can be used for joint limits and collision avoidance; and, h_i can be used to enforce end-effector poses or to comply with robot dynamics.

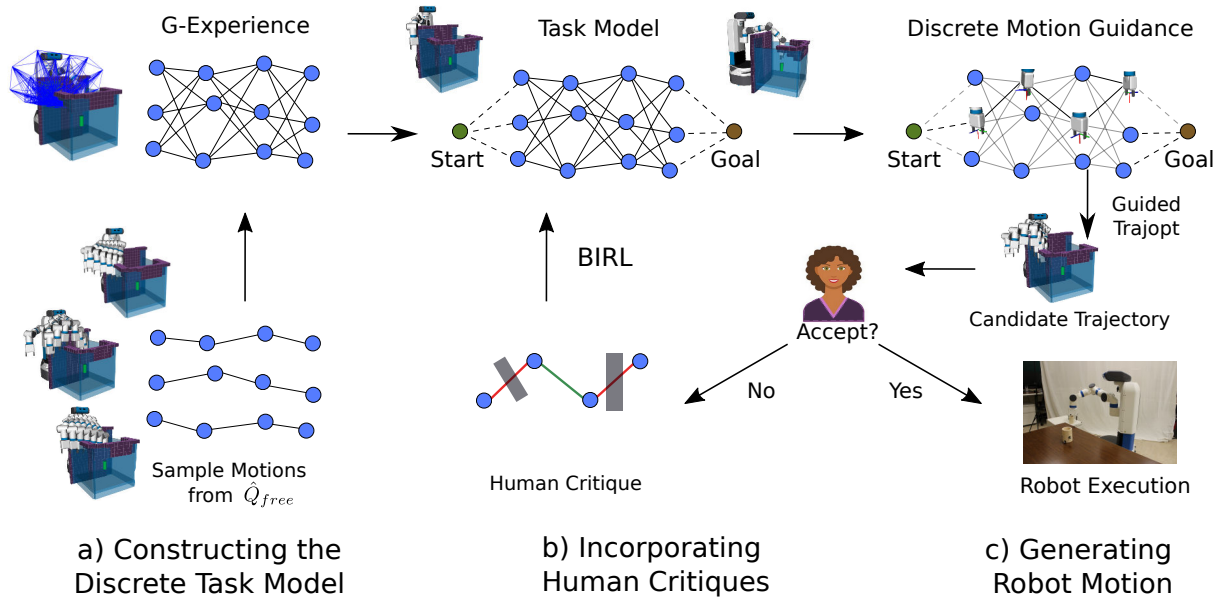

Fig. 2. Schematic of the BLIND algorithm. **a)** BLIND begins by constructing an experience graph, $\mathcal{G}_{exp}$, informed by sample motions generated using the robot's incomplete model of the environment $\hat{\mathcal{Q}}_{free}$. **b)** Next, it creates a discrete task model (an MDP) by connecting the start and goal configurations to $\mathcal{G}_{exp}$ and initializing a reward over the graph edges. The reward, which aims to encode safe regions in the environment, is updated in subsequent iterations using human critiques and BIRL. **c)** The task model is used to arrive at discrete motion guidance, which is used as an input for Guided TRAJOPT to generate a candidate trajectory. The trajectory is shown to the human, who either critiques it or accepts it for safe execution.

This formulation results in a non-convex optimization problem that can be solved using sequential convex optimization, where each non-convex term is linearized around a nominal trajectory and a locally convex version of the problem is solved at every iteration [2], [31].

B. Bayesian Inverse Reinforcement Learning

Bayesian inverse reinforcement learning (BIRL) [32] provides a Bayesian approach to recover the reward [33] of an expert agent given demonstrations of its behavior. The expert agent is assumed to operate in a Markov Decision Process (MDP) $M = (S, A, T, R)$ according to a reward function R, where S is the set of states, A, the set of actions, and T is the transition function. BIRL models the learned reward function as a probability distribution. Given demonstrations $O_{\mathcal{X}} = \{(s_1, a_1), \dots (s_k, a_k)\}$ from the expert, BIRL samples values from the posterior probability of the reward function:

$$P_{r_{\mathcal{X}}}(R \mid O_{\mathcal{X}}) = \frac{1}{Z'} \exp\left(\alpha \sum_i Q^*(s_i, a_i, R)\right) P_R(R)$$

where α quantifies the optimality of the expert, Z' is a normalizing constant, $Q^*(s_i, a_i, R)$ is the optimal Q-function of the expert's optimal policy for reward function R and $P_R(R)$ is the prior reward distribution. To approximate this posterior, BIRL utilizes a Markov Chain Monte Carlo algorithm called *PolicyWalk* that converges from the prior distribution to the correct solution in polynomial time.

IV. Problem Statement

We consider a robot with d controllable joints acting in workspace $\mathcal{W}$. The robot can be described in terms of a vector $q \in \mathcal{Q} \subset \mathbb{R}^d$, where $\mathcal{Q}$ is its configuration space. The subset of all robot configurations that are not in collision is denoted as $\mathcal{Q}_{free} \subseteq \mathcal{Q}$. A valid trajectory $\pi : [0, T] \rightarrow \mathcal{Q}_{free}$ is a time-parameterized function that assigns collision-free configurations to timesteps, where T is the duration of the trajectory. The motion planning problem $(q_0, q_T, \mathcal{Q}_{free})$ is defined as that of finding a valid trajectory π that connects the start $q_0 = \pi(0)$ with the goal $q_T = \pi(T)$. In this paper, we consider partial observability of the obstacles in $\mathcal{W}$, which means that the robot has an incorrect estimate of the free configuration space $\hat{\mathcal{Q}}_{free} \neq \mathcal{Q}_{free}$. Additionally, we assume that a human, co-located with the robot, has full observability of $\mathcal{W}$ and can provide critiques when candidate trajectories are presented as shown in Fig. 1c).

V. Technical Approach: BLIND

We provide a human-in-the-loop algorithm termed BLIND to address the problem described in Sec. IV. Fig. 2 provides an overview of the algorithm, which is formalized in Alg. 1. BLIND maintains two representations – the robot's estimate of the free configuration space $\hat{\mathcal{Q}}_{free}$ and a discrete task model $\mathcal{G}$– one continuous and the other discrete. $\hat{\mathcal{Q}}_{free}$ is obtained using sensors and only considers the partially-observable obstacles, while $\mathcal{G}$ is constructed using the procedure described in Sec. V-A. To identify safe regions in the partially observable environment, BLIND iteratively learns a reward function using human critiques. The algorithm overcomes the computational challenge of reward learning in the continuous configuration space by leveraging the discrete task model and a low-dimensional feature representation based on workspace information (Sec. V-B). In each iteration, the task model is used to find the current optimal path in the discrete space, which a novel "guided motion planner" uses as guidance

Algorithm 1: BLIND

input : start, goal, $\hat{\mathcal{Q}}_{free}$, N_q, $\mathcal{G}_{exp}$
output : Collision-free Trajectory $\mathcal{T}$ or $\emptyset$ when fail

1 **if** `isempty` $(\mathcal{G}_{exp})$ **then**
2 $\quad \mathcal{G}_{exp} \leftarrow$ `CreateGexp` (start, goal, $\hat{\mathcal{Q}}_{free}$)
3 $\mathcal{G} \leftarrow$ `CreateTaskModel` (start, goal, $\mathcal{G}_{exp}$) ;
4 **for** i = 1,...,N_q **do**
5 $\quad P \leftarrow$ `GuidanceSearch` $(\mathcal{G})$;
6 $\quad \mathcal{T} \leftarrow$`GuidedTrajOpt` (start, goal, $\hat{\mathcal{Q}}_{free}$, P);
7 $\quad$ **if** `HumanAccepts` $(\mathcal{T})$ **then**
8 $\quad\quad$ Return $\mathcal{T}$;
9 $\quad$ **else**
10 $\quad\quad (C^+, C^-) \leftarrow$`HumanCritique` $(\mathcal{T})$;
11 $\quad\quad \mathcal{G} \leftarrow$BIRL $(\mathcal{G}, (C^+, C^-))$;
12 Return $\emptyset$;

to compute high-DOF trajectories in $\mathcal{Q}$ (Sec. V-C). This translation of human critiques to guide motion via the task model and the guided motion planner is central to BLIND.

Algorithm Overview: As shown in Alg. 1, along with the problem parameters (start, goal, $\hat{\mathcal{Q}}_{free}$), BLIND also assumes a budget of human queries N_q and (optionally) a task model $\mathcal{G}_{exp}$ that comes from previous experiences of the robot in similar tasks. Given a new motion planning problem, BLIND starts by creating a new task model $\mathcal{G}$ (line 3). It then computes a path P from the start to goal using the task model (line 5), which is then used by the guided motion planner to find a trajectory that respects constraints in P (line 6). The trajectory $\mathcal{T}$ returned by the planner is presented to the human as a candidate solution that she can accept or reject (line 7). In case she does *not* accept, BLIND asks the human for a detailed critique (line 10), i.e., the human is asked to segment $\mathcal{T}$ into sets of good (C^+) and bad (C^-) transitions according to her preferences. The critiques are used to recover the expert's reward function using BIRL (line 11). The process (lines 3-10) repeats until a generated trajectory is accepted or the budget of human queries N_q is exhausted. We now describe the key elements of BLIND in more detail.

A. Constructing the Discrete Task Model

While our goal is to plan motion of high-DOF robots using human expertise, learning motion or rewards directly in the continuous configuration space $\hat{\mathcal{Q}}_{free}$ can be computationally challenging. Thus, BLIND utilizes a discrete representation of the problem $\mathcal{G}$, which provides a tractable state space to perform human-guided reward learning. Additionally, through the variable $\mathcal{G}_{exp}$, the task model allows for reuse of past robot experiences while solving tasks in a partially observable environment and helps accelerate planning in new problems.

Formally, the task model $\mathcal{G}$ is a discrete MDP, whose states correspond to time-parameterized waypoints in the partially observable environment. A naive approach to generate $\mathcal{G}$ is to uniformly discretize the robot workspace (e.g., as a grid world); however, as we show in experiments, this can lead to inferior performance. Instead, BLIND generates the state space by leveraging motion planning. In particular, the state space is defined by building on a previous task model $\mathcal{G}_{exp}$, derived while solving previous instances of BLIND in $\mathcal{W}$.

If $\mathcal{G}_{exp}$ is unavailable, its state space is constructed in line 2 by first generating M motion planning problems $\{(\text{start}_j, \text{goal}_j, \hat{\mathcal{Q}}_{free}) | j = 1 : M\}$; then, solving these problems using existing motion planners to obtain M trajectories $\mathcal{T}_{1:M}$; and, finally, discretizing the trajectories into time-parameterized waypoints to arrive at the set of states (nodes) of $\mathcal{G}_{exp}$. We highlight that the trajectories $\mathcal{T}_{1:M}$ are computed using partial information of the environment $\hat{\mathcal{Q}}_{free}$ and motion planners (such as [2]) that do not consider partial observability. As such, they may be infeasible in the environment $\mathcal{Q}_{free}$ and are not executed by the robot.

As shown in Fig. 2 (left), the action space and transition function of $\mathcal{G}_{exp}$ are represented using a directed graph, whose nodes correspond to the states of $\mathcal{G}_{exp}$ and edges correspond to possible transitions. The transitions are modeled as deterministic. The out-edges of each node are constructed by finding nearest time-parameterized neighbor of the waypoint in each of the M trajectories. Mathematically, consider trajectories π_A and π_B with time-parameterized waypoints $(t_{Ai}, \pi_A(t_{Ai}))$ and $(t_{Bj}, \pi_B(t_{Bj}))$, respectively. An out-edge from $\pi_A(t_{Ai})$ to $\pi_B(t_{Bj})$ is created if

$$t_{Bj} = \arg \min_{t_{Bk} \in \mathcal{T}_B, t_{Bk} - t_{Ai} \geq 0} t_{Bk} - t_{Ai}$$

where $\mathcal{T}_B$ is the set of times of the discretized trajectory π_B. Finally, as depicted in Fig. 2 (center, top), BLIND creates the task model $\mathcal{G}$ by adding the start and goal of the motion planning problem to $\mathcal{G}_{exp}$. The reward function of the task model $\mathcal{G}$ is an unknown, which BLIND learns from human guidance as described next.

B. Learning from Human Critiques

As the robot has to plan under partial observability, it relies on the human to learn about (un)safe regions of its environment. To perform this learning in a sample efficient and generalizable manner, BLIND utilizes IRL in conjunction with the discrete task model $\mathcal{G}$. The unknown reward function of the task model is used to quantify safety and human preference over robot motion, and is learned from critiques. Mathematically, the reward is represented as a linear combination of state-dependent features [34]. In the experiments, we use features that depend on robot's end-effector position, as they provide meaningful information for the tasks at hand. However, we stress that BLIND is flexible to the choice of features, which can be selected based on the domain.

To obtain human critiques, BLIND first computes a candidate trajectory (see Sec. V-C), which is then presented to a human (see Fig. 2, center). If the human finds the trajectory desirable, she can accept it as is. Otherwise, the human is asked to decide whether each pair (s_i, a_i) belongs to the good (C^+) or the bad (C^-) segments according to her notion of safety. In Fig. 2, paths critiqued as good and bad are shown in green and red, respectively. In BLIND, the human provides only the critiques as inputs, which contrasts with the standard IRL setting where full demonstrations of (s_i, a_i)-pairs are required.

To perform reward learning given critiques, inspired by [9], we maintain a belief over the reward function and utilize BIRL [32]. BIRL requires a likelihood model, which in our case depends on the probability of a segment receiving a good/bad critique. The probability p_i of segment (s_i, a_i) belonging to the set C^+ of good segments is modeled as $p_i \propto \exp(\alpha Q(s_i, a_i, R))$, where $Q(s_i, a_i, R)$ is the optimal Q-value and α is a hyper-parameter. Similarly, the probability q_i of a segment belonging to C^- is modeled as $q_i = 1 - p_i$. The likelihood of the overall critique is given as:

$$\Pr(C^+, C^- \mid R) = \prod_{(s_i,a_i)\in C^+} p_i \prod_{(s_i,a_i)\in C^-} q_i$$

Given the likelihood model and critiques, BLIND learns the reward function using the *PolicyWalk* algorithm of [32].

C. Generating Human-guided Robot Motion

Having learned the reward using the discrete task model, BLIND needs a method to use this learning for continuous motion planning. To do so, in line 5, BLIND computes the optimal path P in $\mathcal{G}$ using the Bellman-Ford algorithm [35]. A key insight in BLIND is that P can be used as a motion-level guidance to produce a feasible trajectory from start to goal that passes through the set of end-effector poses given by P. To accomplish this, we propose a guided variant of TrajOpt [2] with additional end-effector pose constraints extracted from the guidance P. Formulation (1) shows "Guided TrajOpt":

$$\begin{aligned}
\underset{q_0,\ldots,q_T}{\text{minimize}} \quad & \sum_{t=0}^{T-1} \|q_{t+1} - q_t\|^2 && \text{(1a)}\\
\text{subject to} \quad & (q_0, q_T) = (q_{St}, q_G), && \text{(1b)}\\
& \text{sd}(A_{it}, O_j) \geq d_s, \ \forall i, j, t && \text{(1c)}\\
& F_k^{-1}\text{FK}(q_\tau) = 0, \ \forall (k, \tau) \in P && \text{(1d)}
\end{aligned}$$

where the variables $q_t \in \mathcal{Q}$ are waypoints ($t = 0 : T$) in configuration space; q_{St}, q_G are given start and goal configurations, respectively, sd( ) is the signed distance between the i-th robot link at timestep t, A_{it}, and the j-th obstacle O_j in the observed workspace and d_s is a safe distance. In (1d), F_k denotes the k-th target pose in P that needs to be enforced at timestep τ and $\text{FK}(q_\tau)$ is the pose of the end-effector at q_τ.

VI. Experiments

We evaluate BLIND through simulations of Fetch (using [36]), a high-DOF robot, performing manipulation tasks in partially observable environments. Additionally, we also qualitatively demonstrate BLIND on a physical manipulation task as shown in the accompanying video.

A. Experiment Scenarios

We investigate the performance of our algorithm in two tasks referred to as Box and Stove. The Box environment was initially introduced in our preliminary workshop paper [37]. Both tasks are designed to be challenging and realistic for motion planning in the sense that they consider high-DOF robotic manipulation with non-trivial workspace obstacles

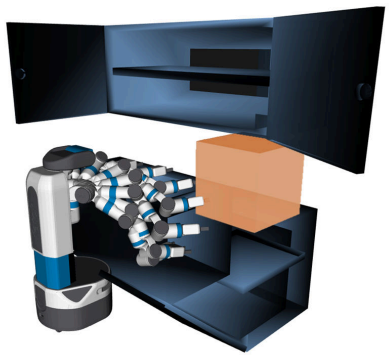

Fig. 3. Stove task: the robot needs to move an object from the dishwasher to the sink, while avoiding the potentially dangerous hot region above the stove (orange cuboid). Partial visibility here comes into play since the robot cannot sense temperature.

represented as OctoMaps [38] and meshes. Additionally, a significant part of the workspace can not be observed by the robot. To the best of our knowledge, competing methods can not scale to such problems and those that learn in end-effector space can not trivially transfer the learned policies to compute full joint-space trajectories.The Box task (Fig. 1) assumes that the workspace information is captured by an on-board depth camera that creates an occupancy grid as an OctoMap. The robot is tasked with grasping the green object inside the box, while starting from outside. However, due to the robot's location with respect to the box, most of the box's geometry is occluded. In the Stove task (Fig. 3), the robot needs to manipulate an object from the dishwasher to the sink. This scenario uses meshes and the full geometry is available at planning time; however, the robot can not sense the heat of the stove.

In all cases, we are interested in finding trajectories that are safe using partial workspace information and the human critiques. To create the task model, we follow the procedure described in Sec. V-A using $M = 7$ and 10 for the respective domains. We model the reward function $R(s_i, a_i)$ as a linear combination of features [34]. As features, we use the spatial coordinates of the robot's end-effector (for both domains) and whether a segment lies in (un)observable regions of the workspace (for the Box). To test the methods, we created 20 variations of the two tasks. We vary the position of the box and that of the kitchen by $\pm$10cm along the X and Y axes. For the Box task, we additionally vary the box's orientation relative to the robot base by ± 15 deg. For the Stove, the dishwasher and cupboard doors are randomly sampled between completely open and closed. These environments were generated with MotionBenchMaker [39].

Given a new problem, all methods propose a trajectory and query the human for approval. If the human approves the trajectory, it is executed. If the human does not approve, she provides critiques for all trajectory snippets, and a new trajectory is proposed until approval. For the simulation experiments, we used an exact collision checker with the full workspace geometry instead of a human. For the demonstration on the real robot, one of the authors performed the role of human operator. We leave experiments with a varied

TABLE I
RESULTS: EFFECT OF TASK MODEL AND MOTION PLANNER

	Teaching effort		Success rate		Path length (rad)	
	Box	Stove	Box	Stove	Box	Stove
Guided TrajOpt						
Graph (**BLIND**)	**2.57**	**6.86**	**0.90**	**0.72**	**18.17**	**16.55**
Grid	5.26	8.19	0.76	0.65	18.63	18.13
Cartesian Planner						
Graph	8.52	9.36	0.72	0.64	18.36	25.28
Grid ([9]-ext.)	6.85	12.21	0.70	0.55	19.38	20.01

TABLE II
RESULTS: EFFECT OF HUMAN GUIDANCE

	Teaching effort		Success rate		Path length (rad)	
	Box	Stove	Box	Stove	Box	Stove
BLIND	**2.57**	**6.86**	**0.90**	**0.72**	**18.17**	**16.55**
PBLIND	4.16	9.30	0.84	0.60	18.99	17.45
RBLIND	15.23	18.85	0.43	0.21	19.00	26.90
TRAJOPT	n/a	n/a	0.46	0.22	12.23	12.34

set of human operators and a usability study of our interface for future work. We make 5 independent runs of each of the 20 problems. To test the methods, we evaluate the teaching effort (i.e., the number of candidate trajectories shown to the human to achieve approval), the success rate, and the path length of the resulting safe trajectory in configuration space. The problem instance is considered a failure if it requires the human to critique more than 20 trajectories.

B. Baselines

We compare BLIND with several baselines. We use the TRAJOPT algorithm, initialized with a random trajectory, as a baseline to investigate how a motion planner that does not reason about partial observability behaves on the described tasks. We also show the result of applying an extension of the method proposed in [9] to the two described tasks. Similar to BLIND, the method in [9] utilizes BIRL and human critiques; however, it performs reward learning on a task model defined using a predefined 2-dimensional grid and does not compute joint-space trajectories. To alleviate this limitation, we create an extension that constructs a 3-dimensional grid in the workspace to perform reward learning (i.e., grid-based task model) and compute full joint-space trajectories from the guidance path P using a Cartesian planner, i.e., by solving collision-aware inverse kinematic problems to move the robot from one waypoint to the next. Inspired by this idea, we also compare against two additional variations of this extension: 1) one that uses our guided planner and the grid-based task model (in contrast, to the graph in BLIND) and 2) another one that uses the Cartesian planner (in contrast, to the guided planner in BLIND) on our graph-based task model.

C. Results

Table I shows the performance of BLIND and the baselines for the two tasks, averaged across the 100 problem instances. Each row corresponds to a different baseline, where we specify first the motion planner used (i.e., Guided TrajOpt vs Cartesian Planner) and then the type of representation used for the task model (Graph vs Grid). For example, BLIND corresponds to the first combination Guided TrajOpt and Graph while the extension of [9] corresponds to the last combination, Cartesian Planner and Grid.

Similar trends are observed for the two tasks, Stove being the more challenging scenario. We observe that BLIND requires much less teaching effort than the extension of [9] and higher success rate, which shows the advantage of our approach relative to prior art for high-DOF motion planning under partial observability. BLIND also outperforms the extension that performs reward learning on a grid-based task model (i.e., Guided TrajOpt with Grid), which strongly highlights the importance of guiding the creation of task model using motion planning. Further, even within the grid-based task model, the variant using Guided TrajOpt outperforms the extension of [9], highlighting the ability of guided planner to better accommodate task-level guidance. Finally, we note that BLIND also attains lower path length of the resulting trajectories compared to the baselines, especially for the Stove task. In summary, these set of experiments demonstrate the need of specialized representations and techniques to perform human-guided learning for high-DOF robots and BLIND as a solution that fulfills this need.

We also investigate the effect of the reward learning process by performing an ablation study over variations of BLIND. Instead of using BIRL to update the weights of the graph, we use two simple approaches: RBLIND where the weights of the graph are randomly generated at every iteration and PBLIND where the weights of the bad segments are penalized by a constant value. The results are shown in Table II. It is noteworthy that RBLIND performs poorly compared to BLIND and PBLIND in both teaching effort and success rate, which demonstrates the advantage of learning over random behavior. In particular, the success rate of RBLIND is comparable to that of TRAJOPT, highlighting the importance of not only the human guidance but also how it is incorporated. Finally, BLIND also outperforms PBLIND, showing the benefits of the reward learning compared to simpler approaches.

VII. CONCLUDING REMARKS

We presented BLIND, a new method for motion planning in partially observable environments by efficiently leveraging human guidance. BLIND combines learning from human critiques with motion planning to enable computation of continuous joint-space trajectories for high-DOF robots in partially observable environments, while keeping the problem tractable. Experiments show the benefits of BLIND when compared to existing methods and variations, highlighting the importance of techniques that jointly consider partial observability, high-DOF continuous spaces, and human guidance. In all cases, using both our guided planner and task model showed improved results in terms of teaching efficiency, success rate, and path length of the resulting trajectory. Future work includes performing evaluation with human users as well as exploring other modes of human guidance (such as preferences and corrections) for high-DOF robot learning.

REFERENCES

[1] H. M. Choset, S. Hutchinson, K. M. Lynch, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementation*. MIT Press, 2005.

[2] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *Int. J. of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[3] C. Dawson, A. Jasour, A. Hofmann, and B. Williams, "Provably safe trajectory optimization in the presence of uncertain convex obstacles," in *IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, pp. 6237–6244, 2020.

[4] C. Quintero-Peña, A. Kyrillidis, and L. E. Kavraki, "Robust optimization-based motion planning for high-DOF robots under sensing uncertainty," in *IEEE Int. Conf. Robot. Autom.*, 2021.

[5] J. van den Berg, S. Patil, and R. Alterovitz, "Motion planning under uncertainty using iterative local optimization in belief space," *Int. J. of Robotics Research*, vol. 31, no. 11, pp. 1263–1278, 2012.

[6] W. Sun, J. Van den Berg, and R. Alterovitz, "Stochastic extended LQR for optimization-based motion planning under uncertainty," *IEEE Transactions on Automation Science and Engineering*, vol. 13, pp. 437–447, Apr. 2016.

[7] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.

[8] S. Chernova and A. L. Thomaz, "Robot learning from human teachers," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 8, no. 3, pp. 1–121, 2014.

[9] Y. Cui and S. Niekum, "Active reward learning from critiques," in *IEEE Int. Conf. Robot. Autom.*, pp. 6907–6914, 2018.

[10] M. Palan, G. Shevchuk, N. Charles Landolfi, and D. Sadigh, "Learning reward functions by integrating human demonstrations and preferences," in *Robotics: Science and Syst.*, 2019.

[11] A. Somani, N. Ye, D. Hsu, and W. S. Lee, "DESPOT: Online POMDP planning with regularization," in *Int. Conf. on Neural Information Processing Systems*, vol. 26, 2013.

[12] J. Van Den Berg, P. Abbeel, and K. Goldberg, "LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information," *Int. J. of Robotics Research*, vol. 30, pp. 895–913, June 2011.

[13] B. Luders, M. Kothari, and J. How, "Chance constrained RRT for probabilistic robustness to environmental uncertainty," in *AIAA guidance, navigation, and control conference*, p. 8160, 2010.

[14] B. Axelrod, L. P. Kaelbling, and T. Lozano-Pérez, "Provably safe robot navigation with obstacle uncertainty," *Int. J. of Robotics Research*, vol. 37, no. 13-14, pp. 1760–1774, 2018.

[15] B. Saund and D. Berenson, "Motion planning for manipulators in unknown environments with contact sensing uncertainty," in *Int. Symp. on Experim. Robotics*, pp. 461–474, Nov. 2018.

[16] B. Saund, S. Choudhury, S. Srinivasa, and D. Berenson, "The blindfolded robot: A bayesian approach to planning with contact feedback," in *Int. Symp. on Robotics Research*, Oct. 2019.

[17] S. Schaal, "Learning from demonstration," *Int. Conf. on Neural Information Processing Systems*, pp. 1040–1046, 1996.

[18] P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," in *Int. Conf. on Neural Information Processing Systems*, pp. 4302–4310, 2017.

[19] D. Brown, R. Coleman, R. Srinivasan, and S. Niekum, "Safe imitation learning via fast bayesian reward inference from preferences," in *Int. Conf. on Machine Learning*, pp. 1165–1177, 2020.

[20] D. P. Losey and M. K. O'Malley, "Including uncertainty when learning from human corrections," in *Conf. on Robot Learning*, pp. 123–132, 2018.

[21] G. Gonzalez and J. Wachs, "ICONS: Imitation CONStraints for robot collaboration," in *IEEE Int. Conf. on Robot & Human Interactive Comm.*, pp. 147–154, IEEE, 2021.

[22] N. Wilde, D. Kulić, and S. L. Smith, "Learning user preferences from corrections on state lattices," in *IEEE Int. Conf. Robot. Autom.*, pp. 4913–4919, 2020.

[23] M. Li, A. Canberk, D. Losey, and D. Sadigh, "Learning human objectives from sequences of physical corrections," in *IEEE Int. Conf. Robot. Autom.*, pp. 2877–2883, 2021.

[24] A. Jain, S. Sharma, T. Joachims, and A. Saxena, "Learning preferences for manipulation tasks from online coactive feedback," *Int. J. of Robotics Research*, vol. 34, no. 10, pp. 1296–1313, 2015.

[25] B. Argall, B. Browning, and M. Veloso, "Learning by demonstration with critique from a human teacher," in *IEEE-ACM Int. Conf. on Human-Robot Interaction*, pp. 57–64, IEEE, 2007.

[26] A. Singh, L. Yang, C. Finn, and S. Levine, "End-to-end robotic reinforcement learning without reward engineering," in *Robotics: Science and Syst.*, 2019.

[27] J. Denny, J. Colbert, H. Qin, and N. M. Amato, "On the theory of user-guided planning," in *IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, pp. 4794–4801, 2016.

[28] N. García, R. Suárez, and J. Rosell, "HG-RRT*: Human-guided optimal random trees for motion planning," in *Conference on Emerging Technologies & Factory Automation (ETFA)*, pp. 1–7, IEEE, 2015.

[29] F. Islam, O. Salzman, and M. Likhachev, "Online, interactive user guidance for high-dimensional, constrained motion planning," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pp. 4921–4928, 2018.

[30] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "CHOMP: Covariant hamiltonian optimization for motion planning," *Int. J. of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[31] R. Bonalli, A. Cauligi, A. Bylard, , and M. Pavone, "GuSTO: guaranteed sequential trajectory optimization via sequential convex programming," in *IEEE Int. Conf. Robot. Autom.*, pp. 6741–6747, 2019.

[32] D. Ramachandran and E. Amir, "Bayesian inverse reinforcement learning," in *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, (San Francisco, CA, USA), pp. 2586–2591, Morgan Kaufmann Publishers Inc., 2007.

[33] S. Russell, "Learning agents for uncertain environments (extended abstract)," in *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pp. 101–103, ACM Press, 1998.

[34] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Int. Conf. on Machine Learning*, p. 1, 2004.

[35] Richard Bellman, "On a routing problem," *Quart. Appl. Math.*, pp. 87–90, 1958.

[36] Z. Kingston and L. E. Kavraki, "Robowflex: Robot motion planning with MoveIt made easy," *arXiv preprint arXiv:2103.12826*, 2021.

[37] C. Quintero-Peña, C. Chamzas, V. Unhelkar, and L. E. Kavraki, "Motion planning via bayesian learning in the dark," in *ICRA 2021: Workshop on Machine Learning for Motion Planning*, June 2021.

[38] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Autonomous Robots*, 2013.

[39] C. Chamzas, C. Quintero-Peña, Z. Kingston, A. Orthey, D. Rakita, M. Gleicher, M. Toussaint, and L. E. Kavraki, "MotionBenchMaker: A tool to generate and benchmark motion planning datasets," *IEEE Robot. Autom. Letters*, vol. 7, p. 882–889, Apr. 2022.