Reward Learning with Intractable Normalizing Functions

Joshua Hoegerman and Dylan P. Losey

Abstract— **Robots can learn to imitate humans by inferring what the human is optimizing for. One common framework for this is Bayesian reward learning, where the robot treats the human's demonstrations and corrections as observations of their underlying reward function. Unfortunately, this inference is *doubly-intractable*: the robot must reason over all the trajectories the person could have provided and all the rewards the person could have in mind. Prior work uses existing robotic tools to approximate this normalizer. In this paper, we group previous approaches into three fundamental classes and analyze the theoretical pros and cons of their approach. We then leverage recent research from the statistics community to introduce *Double MH reward learning*, a Monte Carlo method for asymptotically learning the human's reward in continuous spaces. We extend Double MH to conditionally independent settings (where each human correction is viewed as completely separate) and conditionally dependent environments (where the human's current correction may build on previous inputs). Across simulations and user studies, our proposed approach infers the human's reward parameters more accurately than the alternate approximations when learning from either *demonstrations* or *corrections*. See videos here: https://youtu.be/EkmT3o5K5ko**

Index Terms— **Intention Recognition, Learning from Demonstration, Probabilistic Inference**

I. Introduction

Consider a robot arm learning from demonstrations (see Figure 1). The human guides the robot through example trajectories and the robot tries to infer the human's objective based on their demonstrations. For instance, here the robot arm should learn to slide the box across the table.

State-of-the-art research often tackles this *reward learning* problem using *Bayesian inference* [1]–[3]. The trajectories provided by the human are observations of their latent objective (i.e., their reward function), and the robot recovers a distribution over the rewards by applying Bayes' theorem. Put intuitively: the robot infers rewards under which the human's demonstrations are approximately optimal.

Unfortunately, reward learning in continuous spaces is *doubly intractable*. The robot must normalize across the space of trajectories (i.e., what other demonstrations could the human have provided?), and over the space of rewards (i.e., what else could the human be optimizing for?). Today's robots recognize that they cannot compute these normalizers exactly and so they make a variety of *approximations* [4]–[15].

Inaccurate approximations of the normalizing function can lead to incorrect inference: instead of learning what the human

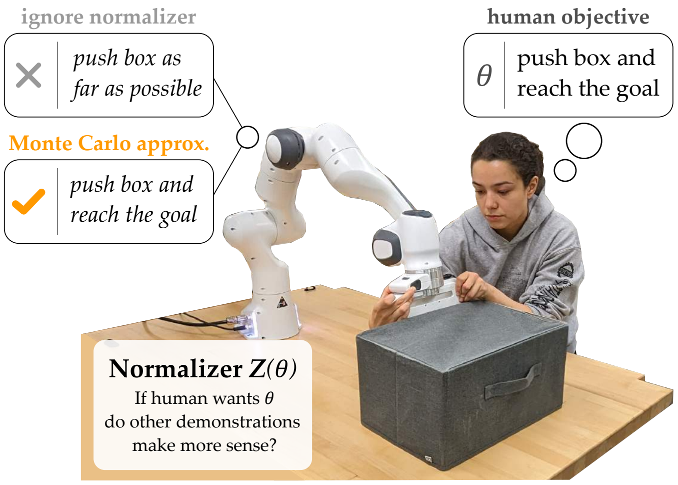
ignore normalizer
push box as far as possible
Monte Carlo approx.
push box and reach the goal
human objective
θ
push box and reach the goal
Normalizer $Z(\theta)$
If human wants θ do other demonstrations make more sense?

Fig. 1. To infer the user's reward, (i.e., objective) robots must compare the human's actual demonstration to other demonstrations the human *could* have provided. Computing this normalizer makes Bayesian reward learning doubly-intractable and ignoring the normalizer entirely can lead to incorrect robot learning. We propose a Monte Carlo method for a more accurate approximation.

meant, the robot learns to perform different and potentially undesirable tasks. Refer back to Figure 1: here ignoring the normalizer can cause the robot to knock the box off the table.

In this paper, we focus on inferring the human's reward from demonstrations or corrections. Our insight is that:

We can enable asymptotic reward learning by leveraging novel Monte Carlo methods from the statistics community.

Our resulting framework applies to problems where the robot has a predictive model of the environment. Given a sequence of independent or interconnected human demonstrations, we enable robots to accurately learn the human's reward despite doubly intractable normalizing functions. Returning to Figure 1: our robot learns to push the box correctly using the same amount of data as the baseline.

Overall, we make the following contributions:

Comparing Approximations. We theoretically and experimentally analyze three existing classes of methods for approximating the normalizer during Bayesian reward learning.

Introducing Double Metropolis-Hastings Sampling. We apply novel statistics research to develop a Double MH algorithm for Bayesian reward learning in continuous spaces.

Learning from Demonstrations and Corrections. Across simulations and user studies, we show that Double MH results in more accurate reward learning from both conditionally independent (e.g., separate) demonstrations and conditionally dependent (e.g., interconnected) corrections.

This work is supported in part by NSF Grant #2222468.
The authors are with the Collaborative Robotics Lab (Collab), Dept. of Mechanical Engineering, Virginia Tech, Blacksburg, VA 24061.
Corresponding Author's Email : jhoegerm@vt.edu

II. Related Work

IRL. Our problem is an instance of Inverse Reinforcement Learning (IRL) where the robot tries to recover the reward that the human wants the robot to optimize for [16]. Related work formalizes this as *Bayesian inference* [1]–[3]. Wherein, given a prior distribution and the human's inputs (e.g., state-action pairs [2] or trajectories [1] provided by the human), the robot arm infers a posterior over the space of possible rewards. In order to connect the human's inputs to rewards, today's robots model the human as a noisily rational teacher [3], [17], [18]. Unfortunately, the human model becomes *intractable* in continuous spaces; without this accurate human model, the robot cannot correctly infer the human's reward.

As we will demonstrate in Section III, the key challenge is the *normalizing function* of the human model. Within this normalizing function, the robot integrates over the space of all possible human inputs (e.g., their actions or trajectories). This normalizing function makes reward learning doubly intractable: we can use standard Markov chain Monte Carlo (MCMC) methods to eliminate a part of this problem [2], [4], [5], but the normalizing function still remains. Existing IRL algorithms have developed different approaches for dealing with the normalizer. Some works may not explicitly account for the normalizing function [4]–[6], and there are settings where *ignoring* this normalizer is reasonable. Others use *sampling* to approximate the normalizer: this includes sampling uniformly from the trajectory space [7], [8], sampling trajectories close to the human's inputs [9], or using importance sampling to convert between the robot's trajectory distribution and a uniform distribution over trajectories [10], [11]. Finally, related works have also used the *maximum* trajectory reward in place of the normalizing function [12]. This substitution can be justified using Laplace's approximation [14], [15].

To summarize, prior work on Bayesian IRL uses ignore, sampling, and maximum approaches to estimate the normalizing function and infer the human's reward. In this paper, we theoretically and experimentally compare these different approaches to understand their relative advantages.

Approximate Bayesian Inference. Within the robotics community, we have developed a variety of techniques for learning with an intractable normalizing function. But what about work *outside* of robotics? Statistics research has recently proposed several Markov chain Monte Carlo (MCMC) algorithms for Bayesian inference in the presence of intractable normalizing functions [19]–[22]. These approaches modify techniques such as Metropolis-Hastings sampling to obtain computationally efficient and accurate algorithms for inferring *doubly intractable* posterior distributions. In this paper, we apply recent breakthroughs from the statistics community to propose a new method for Bayesian reward learning.

III. Problem Formulation

We consider settings where a robot arm is using Bayesian inference to learn from human examples. The human teacher knows what task the robot should perform. More specifically, the human has in mind a reward function that the robot should optimize, and the robot is trying to infer this reward from the human's data. The human might provide complete examples of their desired behavior (i.e., demonstrations), or they could just modify snippets of the robot's existing motion (i.e., corrections). We recognize that humans are not perfect teachers: when showing the robot how to carry a glass of water, the human may not have enough time or ability to meticulously orchestrate every joint. The robot views the human as a *noisily rational* agent that approximately maximizes their reward.

Task and Reward. This problem is an instance of a Markov decision process (MDP) where the robot does not know the reward function. Let the MDP be a tuple $M = \langle \mathcal{S}, \mathcal{A}, f, r, T \rangle$ where $s \in \mathcal{S}$ is the system state and $a \in \mathcal{A}$ is the robot's action. For example, s could be the arm's joint position and the pose of the cup, and a could be the robot's joint velocity. At timestep t, the system transitions to a new state according to the deterministic dynamics $s^{t+1} = f(s^t, a^t)$. The task ends after a total of T timesteps. Let $\xi = (s^0, \dots s^T)$ be the robot's *trajectory*, i.e., the sequence of visited states across T timesteps, and let Ξ be the set of possible trajectories.

During every timestep, the robot receives a scalar reward $r(s)$. Remember that the robot *does not* know the desired reward function. Without loss of generality, we will write the reward as $r(s, \theta)$, where vector $\theta \in \mathbb{R}^d$ captures the aspects of the reward function that the robot does not know. For example, in related works [1], [16] the reward function is often a linear combination of features such that $r(s, \theta) = \theta \cdot \phi(s)$. Here $\phi : \mathcal{S} \to \mathbb{R}^d$ is a d-length feature vector that captures task-relevant aspects of the state (e.g., the distance from the table, the orientation of the cup), and θ determines the relative importance of these features. Across an entire trajectory, the robot's cumulative reward is: $R(\xi, \theta) = \sum_{s \in \xi} r(s, \theta)$. If the reward function is a linear combination of features, this simplifies to: $R(\xi, \theta) = \theta \cdot \sum_{s \in \xi} \phi(s) = \theta \cdot \Phi(\xi)$.

Human Data. The robot is attempting to learn the reward (and more precisely, the unknown parameters θ) from human examples. Let $\mathcal{D} = \{\xi_1, \dots \xi_K\}$ be a dataset of K trajectories provided by the human expert. Our approach is not tied to any specific way of gathering this dataset. The trajectories could be collected *offline* as the human kinesthetically guides the robot through task demonstrations [6], [9], [10]. Although, we could also add improved trajectories *online* as the human corrects the robot arm [7], [12], [23]. In either case, we aggregate the human's trajectories into the dataset $\mathcal{D}$.

For simplicity, we now assume that the human teacher only provides a *single* trajectory, i.e., $\mathcal{D} = \xi$. In Section V we will extend our analysis to a dataset of K trajectories.

Bayesian Inference. The robot infers the parameters θ from the human's trajectory ξ. Let $P(\theta \mid \xi)$ denote the probability that the human is optimizing for reward parameters θ given the input trajectory ξ. Applying Bayes' theorem:

$$P(\theta \mid \xi) \propto P(\xi \mid \theta) \cdot P(\theta) \tag{1}$$

where $P(\theta)$ is the prior and $P(\xi \mid \theta)$ is the likelihood function. Intuitively, $P(\xi \mid \theta)$ expresses how likely it is (from the robot's perspective) that the human provides trajectory ξ given the human is optimizing for reward parameters θ.

Human Model. The likelihood function $P(\xi \mid \theta)$ is a human model: it tries to capture how the human teacher maps their hidden objective to an example trajectory. Prior work in behavioral economics [18], cognitive science [17], and reward learning [3] suggests that humans are *noisily rational* agents. These humans are not perfect: instead of always choosing the best possible trajectory, noisily rational humans are exponentially more likely to select behaviors with higher rewards. Under the noisily rational model:

$$P(\xi \mid \theta)=\frac{\exp\big(\beta\cdot R(\xi,\theta)\big)}{\int_{\Xi}\ \exp\big(\beta\cdot R(\xi',\theta)\big)\ d\xi'} \tag{2}$$

Where $\beta\in[0,\infty)$ is a hyperparameter set by the designer. As $\beta\rightarrow 0$ each trajectory becomes equally likely and the robot treats the human as a random agent. At the other extreme, as $\beta\rightarrow\infty$ the human is only likely to input optimal trajectories and the robot views the human as perfectly rational.

Normalizing Function. The numerator of Equation (2) is straightforward: we simply substitute ξ and θ into the reward function and evaluate. But once we find $\exp\big(\beta\cdot R(\xi,\theta)\big)$, how good (i.e., how likely) is that trajectory? We need a sense of scale to understand the relative reward for ξ as compared to the alternatives — perhaps there is another trajectory ξ' that achieves a much higher reward. This is where the denominator of Equation (2) comes in. The denominator is a *normalizing function* that integrates over the continuous space of possible trajectories $\xi'\in\Xi$ given reward parameters θ. We refer to the normalizing function as $Z(\theta)$:

$$Z(\theta)=\int_{\Xi}\ \exp\big(\beta\cdot R(\xi',\theta)\big)\ d\xi' \tag{3}$$

The normalizing function $Z(\theta)$ serves to calibrate our human model. Importantly, $Z(\theta)$ can be different for different values of θ. We will show examples of how Z changes (or does not change) as a function of θ in the following sections.

Summary. To infer the human's reward through Bayesian inference we need to find $P(\xi\mid\theta)$ in Equation (1). But to get $P(\xi\mid\theta)$ we first must be able to solve Equation (3), and this normalizing function is *intractable* when Ξ is a continuous manifold [10]. This leads to our core problem: how should robots approximate (or avoid) the normalizing function $Z(\theta)$ when learning from human data?

IV. Approximating the Normalizer

In this section, we analyze three state-of-the-art approaches for dealing with the normalizing function in Bayesian inference. In Section IV-A we prove that robots can completely ignore the normalizing function if Z does not depend on θ; we also identify the necessary conditions for this special case when the reward is a linear combination of features. Moving beyond this special case, we next explore two methods for approximating $Z(\theta)$. In Section IV-B we discuss a sampling approach and in Section IV-C we use the maximum value in place of the normalizer. We prove that the maximum approach will match or outperform the sampling approach as $\beta\rightarrow\infty$ in our noisily rational human model.

Working Example. We first introduce a simplified example to illustrate the analysis throughout this section. Consider a robot arm that is learning how to carry a cup. The robot can hold the cup at any angle between 0 radians (horizontal) and $\pi/2$ radians (vertical). The human teacher inputs a trajectory $\xi=s$ where they specify a single orientation of the cup. The reward is: $r(s,\theta)=-5\cos(\theta)\cdot(s+1)-\sin(\theta)\cdot(\pi/2-s)$. Based on the human's input ξ, the robot is trying to determine whether (a) $\theta=0$ and the robot should hold the cup horizontally at angle $s=0$, or whether (b) $\theta=\pi/2$ and the robot should hold the cup vertically at angle $\pi/2$. Let the robot have a uniform prior over these two possible reward parameters. Applying Equations (1)-(3), the robot's belief that $\theta=0$ is:

$$P(0\mid\xi)=\frac{\exp\big(\beta R(\xi,0)\big)}{\exp\big(\beta R(\xi,0)\big)+\frac{Z(0)}{Z(\pi/2)}\cdot\exp\big(\beta R(\xi,\frac{\pi}{2})\big)} \tag{4}$$

In this simple example, numerical integration is possible and we can exactly find $Z(0)$ and $Z(\pi/2)$. Plugging these exact values into Equation (4) gives us the *ideal* belief. Put another way, this is what the robot *should* learn. We will compare this ideal result to approaches that approximate the normalizer.

A. Ignoring the Normalizing Function

In some settings, it may be reasonable to ignore the normalizing function altogether. Methods such as [4]–[6] use MCMC sampling to cancel out the partition function $P(\xi)$, but they do not explicitly account for the normalizing function within $P(\xi\mid\theta)$. In our working example for instance, these approaches may omit the Z terms from Equation (4).

Proposition 1. *We can ignore the normalizing function when Z does not depend on θ.*

Proof. Consider a problem setting where Z is a constant, i.e., where $Z(\theta_i)=Z(\theta_j)$ for any choice of θ_i and θ_j. When we substitute Equation (2) back into Equation (1) to infer the reward, both the numerator and denominator are multiplied by Z and this normalizing constant cancels out:

$$P(\theta\mid\xi)=\frac{Z}{Z}\left(\frac{\exp\big(\beta\cdot R(\xi',\theta)\big)\cdot P(\theta)}{\int_{\Theta}\ \exp\big(\beta\cdot R(\xi,\theta')\big)\cdot P(\theta')\ d\theta'}\right) \tag{5}$$

Looking specifically at the working example in Equation (4), if $Z(0)=Z(\pi/2)$ then the Z terms cancel. □

So far our analysis shows that we can ignore Z without any loss in performance *if* the normalizer is a constant. But when is this the case? To answer this question we focus on a common framework where the reward function is a linear combination of features, $R(\xi,\theta)=\theta\cdot\Phi(\xi)$. We find that:

Proposition 2. *If $R(\xi,\theta)=\theta\cdot\Phi(\xi)$ and $\theta\in\mathbb{R}^d$ is a d-dimensional unit vector, Z does not depend on θ* if and only if *the feature space $\Phi(\Xi)$ is a sphere centered at zero with radius $\sigma\geq 0$, i.e., $\Phi(\Xi)=\{v\in\mathbb{R}^d:\|v\|=\sigma\}$.*

Proof. Let θ_i and θ_j be two arbitrary unit vectors. Consider Equation (2) with $\beta\in[0,\infty)$. For the integrals $Z(\theta_i)$ and $Z(\theta_j)$ to be equal, for every $\xi\in\Xi$ there must exist some $\xi'\in\Xi$ such that: $\theta_i\cdot\Phi(\xi)=\theta_j\cdot\Phi(\xi')$. This is only satisfied when $\Phi(\Xi)$ is proportional to a unit sphere. □

In practice, it is challenging to ensure the feature space is a sphere. Not only do we need the average of each individual feature to be zero, but the combination of features must always have the same radius. Consider Figure 1 where the human

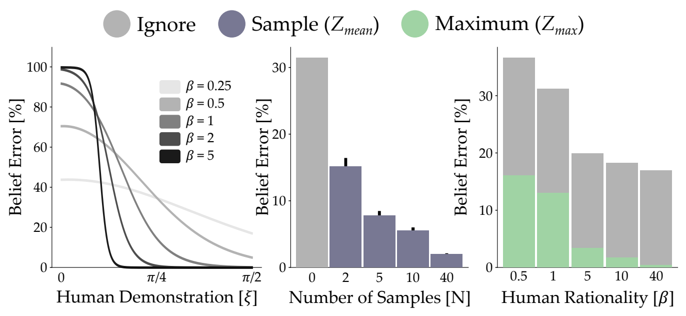

Fig. 2. Approximating the normalizer in our working example. *Belief Error* is the difference between evaluating Equation (4) with the exact $Z(\theta)$ and Equation (4) with approximations for $Z(\theta)$. The lower error indicates the robot learned the correct belief. (Left) The human provides demonstrations ξ of carrying the cup at different angles. If we **ignore** the normalizer, as $\beta \to \infty$ the robot always learns to carry the cup vertically, even when the human wants the opposite. (Middle) Here we left $\beta = 1.0$. As the number of **samples** for Z_{mean} increases, the belief error converges to zero. (Right) As $\beta \to \infty$ the error with the **maximum** approach converges to zero.

is teaching the robot arm: along the human's trajectory they might minimize the robot's height and orientation, resulting in a feature vector $\Phi(\xi)$ where each element is close to zero. Alternatively, the human might move the robot far from the table while changing the orientation, leading to a $\Phi(\xi)$ where each element is close to one. The magnitude of these two feature vectors is different — and thus we *cannot* apply Proposition 2 and ignore the normalizer.

In Propositions 1 and 2 we have identified a special case where the robot does not need to evaluate Z. However, for common settings where Z *is* a function of θ, ignoring the normalizing function results in errors in the robot's learning. See our working example in Figure 2 where we plot the error between Equation (4) with and without $Z(\theta)$.

B. Approximating the Normalizer with Sampling

Instead of ignoring the normalizing function, next we will estimate $Z(\theta)$. One approach is to approximate the integral in Equation (3) through sampling [7]–[11], [13]. Let $\{\xi_1, \dots, \xi_N\}$ be N trajectories sampled uniformly at random from the trajectory space Ξ. We use these samples to approximate the mean value of $\exp\big(\beta \cdot R(\xi, \theta)\big)$ as shown below:

$$Z_{mean}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \exp\big(\beta \cdot R(\xi_i, \theta)\big) \tag{6}$$

Applying the law of the unconscious statistician, this estimate noisily converges to the actual mean as the number of ξ samples increases. It may seem unintuitive at first that we are estimating the mean and not $Z(\theta)$. However, $Z(\theta)$ is equal to this mean multiplied by a volumetric constant that does not depend on θ; because the constant cancels out during Bayesian inference, we only need the mean.

We test this sampling approach on our working example in Figure 2. Compared to a robot that ignores the normalizing function, attempting to estimate $Z(\theta)$ leads to more accurate learning. But we do recognize that the sampling approach comes with an assumption: specifically, we now assume that the robot knows the space of possible trajectories Ξ.

C. Approximating the Normalizer as the Maximum

Other state-of-the-art algorithms use a maximum value in place of the normalizing function [12], [14], [15]. These approaches find the maximum of the numerator in Equation (2), and then treat this maximum as the denominator:

$$Z_{max}(\theta) = \max_{\xi \in \Xi} \exp\big(\beta \cdot R(\xi, \theta)\big) \tag{7}$$

Intuitively, scaling by Z_{max} makes sense because it ensures that the resulting $P(\theta \mid \xi)$ is always less than or equal to one. Recall that for the sampling approach Z_{mean} converges as the number of ξ samples increases. Interestingly, we find an analogous convergence for the maximum approach:

Proposition 3. *The error between Bayesian inference with Z_{max} and an ideal robot that uses Z converges to zero as the human becomes increasingly optimal, i.e., as $\beta \to \infty$.*

Proof. For any given θ, let there be a single trajectory ξ^* that maximizes the human's reward such that $R(\xi^*, \theta) > R(\xi, \theta)$ for all $\xi \in \Xi \setminus \xi^*$. Use numerical integration to estimate Z:

$$Z(\theta) \doteq C\left(Z_{max}(\theta) + \sum_{i=1}^{N} \exp\big(\beta \cdot R(\xi_i, \theta)\big)\right) \tag{8}$$

where C is a volumetric constant that cancels out in Bayesian inference, and $\{\xi_1, \dots, \xi_N\}$ are N non-optimal trajectories sampled uniformly at random from $\Xi \setminus \xi^*$. Taking the limit as $\beta \to \infty$, and remembering that $R(\xi^*, \theta) > R(\xi, \theta)$, we have that $Z_{max}(\theta)$ dominates the remaining terms. Accordingly, as $\beta \to \infty$ Equation (8) converges to $C \cdot Z_{max}(\theta)$, and the difference between a robot that learns using $Z(\theta)$ and a robot that learns using Z_{max} converges to zero. □

We apply Proposition 3 to our working example in Figure 2. As expected, using Z_{max} as the normalizing function becomes increasingly accurate as $\beta \to \infty$. But now that we have two different methods for approximating the normalizer, we are left with a decision: when should designers use Z_{mean} and when should designers use Z_{max}? The answer to this question varies as the problem setting and reward function change. However, we do find a general trend:

Proposition 4. *As $\beta \to \infty$ in the human model, robots learn an equal or more accurate estimate of $P(\theta \mid \xi)$ using Bayesian inference with Z_{max} instead of Z_{mean}.*

Proof. From Proposition 3 we already know that Z_{max} converges to ideal performance as $\beta \to \infty$. It only remains to evaluate the performance of Z_{mean}. Recall from Equation (6) that Z_{mean} samples N trajectories from space Ξ. Because N is a finite number, there will be cases when the robot does not sample the optimal trajectory ξ^*. Compare the numerical integration in Equation (8) to the sampled mean in Equation (6). If the robot does not sample ξ^* in Equation (6), then as $\beta \to \infty$ Equation (6) is not necessarily proportional to Equation (8), where Z is dominated by the exponentiated reward of ξ^*. Because Z_{mean} is not necessarily proportional to Z, a robot that learns using Z_{mean} may not match the performance of an ideal robot that learns with Z. □

We demonstrate Proposition 4 in Figure 3. For *lower values* of β we find that approximating the normalizer with *sampling*

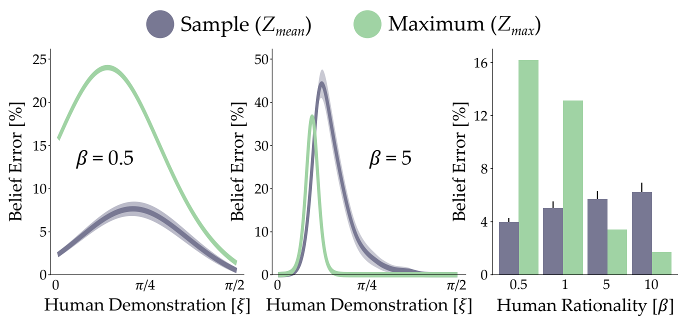

Fig. 3. Comparing **sampling** and **maximum** approaches in our working example. Remember that β from Equation (2) captures how close-to-optimal the human is. (Left) Error when $\beta = 0.5$. (Middle) Error when $\beta = 5$. (Right) For lower values of β we find that the sampling approach is more accurate. However, as $\beta \to \infty$ the maximum approach leads to less error. For **sampling** we perform 100 separate runs where Z_{mean} samples $N = 10$ trajectories each run; the shaded region and bars show standard error.

outperforms the maximum approach. By contrast, for *higher values* of β using the *maximum* as the normalizing function results in more accurate learning. The exact trade-off point is problem-specific, but this general trend holds across our theoretical analysis and experimental results.

V. Scaling up with Metropolis-Hastings Sampling

In Section IV, we analyzed the normalizing function when the human only provides a single trajectory, i.e., when $\mathcal{D} = \xi$. In this section, we scale up to general cases where the robot is learning from a dataset $\mathcal{D}$ of K trajectories. As we scale up, we recognize that the space of rewards Θ is *continuous*. In our working example we assumed that the human wanted the robot to hold the cup either horizontally or vertically; but, more generally, the human may want the robot to hold the cup at any angle. To enable Bayesian inference in continuous reward spaces, we turn to *Metropolis-Hastings (MH) sampling* [24]. We first formulate conditionally independent and dependent reward learning from multiple trajectories (Section V-A) and then combine approaches for approximating the normalizer with MH sampling (Section V-B). In Section V-C we introduce the *Double MH algorithm* for Bayesian reward learning.

A. Learning from Multiple Trajectories

Let $\mathcal{D} = \{\xi_1, \dots, \xi_K\}$ be a dataset of K trajectories input by the human teacher. The probability that the human has in mind reward θ given dataset $\mathcal{D}$ is:

$$P(\theta \mid \mathcal{D}) \propto P(\mathcal{D} \mid \theta) \cdot P(\theta) \tag{9}$$

Similar to Equation (1), here $P(\theta)$ is the prior over the space of rewards and $P(\mathcal{D} \mid \theta)$ is the likelihood the human inputs $\mathcal{D}$ given their reward is parameterized by θ.

Conditionally Independent. If the human selects each trajectory separately then the human's inputs are conditionally *independent*. For instance, consider a human that repeatedly demonstrates a task to the robot: each demonstration depends on their reward θ, but the human does not reason over ξ_i when selecting ξ_j [1], [3], [6]. When the trajectories are conditionally independent Equation (9) reduces to:

$$P(\theta \mid \mathcal{D}) \propto P(\theta) \cdot \prod_{i=1}^{K} P(\xi_i \mid \theta) \tag{10}$$

Plugging in our human model from Equation (2), we reach:

$$P(\theta \mid \mathcal{D}) \propto \frac{\exp\left(\beta \cdot \sum_{i=1}^{K} R(\xi_i, \theta)\right) \cdot P(\theta)}{Z(\theta)^K} \tag{11}$$

where $R(\xi, \theta) = \sum_{s \in \xi} r(s, \theta)$ is the total reward along input ξ and $Z(\theta)$ is the normalizing function from Equation (3).

Conditionally Dependent. Alternatively, if the human provides multiple interconnected trajectories the human's inputs are conditionally *dependent*. For example, imagine a human that iteratively improves the robot's motion by making small corrections: the human's input ξ_j will depend on the human's reward but also on the distance between ξ_j and the previous trajectory ξ_i [12]. In this case, we cannot simplify Equation (9). Instead, we define an augmented human model:

$$P(\mathcal{D} \mid \theta) = \frac{\exp\left(\beta \cdot \mathbf{R}(\mathcal{D}, \theta)\right)}{\int_{\mathbb{D}} \exp\left(\beta \cdot \mathbf{R}(\mathcal{D}', \theta)\right) \, d\mathcal{D}'} \tag{12}$$

where $\mathbf{R}$ is the total reward over dataset $\mathcal{D}$. Going back to our example of a human that makes small improvements, $\mathbf{R}$ could be [12]: $\mathbf{R}(\mathcal{D}, \theta) = \sum_{i=2}^{K} R(\xi_i, \theta) - \|\xi_i - \xi_{i-1}\|^2$. Looking at the denominator of Equation (12), for conditionally dependent trajectories we need to normalize over the entire dataset $\mathcal{D}$ rather than the individual inputs ξ:

$$\mathbf{Z}(\theta) = \int_{\mathbb{D}} \exp\left(\beta \cdot \mathbf{R}(\mathcal{D}', \theta)\right) \, d\mathcal{D}' \tag{13}$$

Here $\mathbb{D}$ is the space of possible datasets $\mathcal{D}$. To find $P(\theta \mid \mathcal{D})$ and perform Bayesian inference we plug Equation (12) with normalizing function $\mathbf{Z}(\theta)$ back into Equation (9).

B. Metropolis-Hastings Sampling

Regardless of whether the human's inputs are conditionally independent or conditionally dependent, we want to use dataset $\mathcal{D}$ to infer the reward parameters θ. This leads us back to the posterior distribution $P(\theta \mid \mathcal{D})$. To evaluate $P(\theta \mid \mathcal{D})$ in Equation (9) we have to deal with *another* normalizer; specifically, the denominator $P(\mathcal{D}) = \int_{\Theta} P(\mathcal{D} \mid \theta) \cdot P(\theta) d\theta$. When Θ is a discrete space (e.g., in our working example where the human wants the cup either horizontal or vertical) we can compute this denominator and find the probability of each $\theta \in \Theta$. But when Θ is continuous, Bayesian inference becomes *doubly intractable* and we cannot typically find closed-form expressions for $P(\theta \mid \mathcal{D})$. Instead, the robot learner uses the MH algorithm to *sample* values of θ from the non-normalized posterior, i.e., $\theta \sim P(\,\cdot \mid \mathcal{D})$.

MH Algorithm. We combine MH sampling with methods for approximating the normalizer in Algorithm 1. At each iteration, we propose a new reward parameter θ'. The robot then compares the probability of θ' with the probability of θ, and accepts θ' with probability $\min\{1, P(\theta' \mid \mathcal{D})/P(\theta \mid \mathcal{D})\}$. Similar to our analysis in Section IV, any terms that do not depend on θ cancel out when we divide the posteriors. Each different approach for approximating the normalizer uses a different method for selecting Z or $\mathbf{Z}$.

- *Ignore:* Set $Z(\theta) = 1$
- *Sampling:* Approximate $Z(\theta)$ using Equation (6)
- *Maximum:* Approximate $Z(\theta)$ using Equation (7)

Algorithm 1 Bayesian Reward Learning with Normalizer Approximation

```
1: $\theta \leftarrow$ sample from $P(\theta)$
2: for each iteration do
3:     $\theta' \leftarrow$ sample from $\Theta$ near $\theta$
4:     Conditionally Independent:
5:     $\frac{P(\theta'|\mathcal{D})}{P(\theta|\mathcal{D})} \leftarrow \frac{\exp\left(\beta \cdot \sum_{i=1}^{K} R(\xi_i,\theta')\right) \cdot Z(\theta)^K \cdot P(\theta')}{\exp\left(\beta \cdot \sum_{i=1}^{K} R(\xi_i,\theta)\right) \cdot Z(\theta')^K \cdot P(\theta)}$
6:     Conditionally Dependent:
7:     $\frac{P(\theta'|\mathcal{D})}{P(\theta|\mathcal{D})} \leftarrow \frac{\exp\left(\beta \cdot \mathbf{R}(\mathcal{D},\theta')\right) \cdot \mathbf{Z}(\theta) \cdot P(\theta')}{\exp\left(\beta \cdot \mathbf{R}(\mathcal{D},\theta)\right) \cdot \mathbf{Z}(\theta') \cdot P(\theta)}$
8:     if $P(\theta' \mid \mathcal{D})/P(\theta \mid \mathcal{D}) > \alpha \sim \mathcal{U}[0,1]$ then $\theta \leftarrow \theta'$
9: Return $\theta$
```

Algorithm 2 Bayesian Reward Learning with Double MH

```
1: $\theta \leftarrow$ sample from $P(\theta)$
2: for each iteration do
3:     $\theta' \leftarrow$ sample from $\Theta$ near $\theta$
4:     $\xi' \sim \mathcal{T}(\mathcal{D}, \theta')$          ▷ inner sampler in Algorithm 3
5:     $\frac{P(\theta'|\mathcal{D})}{P(\theta|\mathcal{D})} \leftarrow \frac{\exp \beta \cdot \sum_{i=1}^{K} \left(R(\xi_i,\theta') - R(\xi',\theta')\right) \cdot P(\theta')}{\exp \beta \cdot \sum_{i=1}^{K} \left(R(\xi_i,\theta) - R(\xi',\theta)\right) \cdot P(\theta)}$
6:     if $P(\theta' \mid \mathcal{D})/P(\theta \mid \mathcal{D}) > \alpha \sim \mathcal{U}[0,1]$ then $\theta \leftarrow \theta'$
7: Return $\theta$
```

These same equations extend to $\mathbf{Z}$, but now the robot samples from the space of datasets $\mathbb{D}$ instead of trajectories Ξ.

C. Reward Learning with Double MH Sampling

In addition to the ignore, sample, and maximum methods from Section IV, we can now introduce one final approach for approximating the normalizer. Standard MH approaches divide $P(\theta' \mid \mathcal{D})$ by $P(\theta \mid \mathcal{D})$ so that any terms that do not depend on θ are cancelled out. Here we take this concept one step further through *double* MH sampling [19]. At a high level, the double MH algorithm introduces an auxiliary variable such that, when we divide the posteriors, $Z(\theta) \cdot Z(\theta')$ appears in both the numerator and denominator, enabling us for the first time to cancel out the normalizing function. This shifts our problem: instead of approximating $Z(\theta)$, we need a method for generating the auxiliary variable.

Double MH Algorithm. We outline Double MH sampling for Bayesian reward learning in Algorithms 2 (outer sampler) and 3 (inner sampler). For clarity we focus on *conditionally independent* trajectories; it is straightforward to modify this pseudocode for the *conditionally dependent* case. At each iteration, the outer sampler proposes a new θ'. The inner sampler then inputs this θ' and generates a trajectory ξ' from the distribution $P(\xi \mid \theta')$. The new trajectory — which is sampled, and does not come from human demonstrations — is the auxiliary variable. We leverage this auxiliary variable to cancel out the normalizing functions and avoid computing $Z(\theta)$. Specifically, the robot accepts θ' with probability:

$$\min\left\{1, \frac{P(\theta') \cdot \left(\prod_{i=1}^{K} P(\xi_i \mid \theta')P(\xi' \mid \theta)\right)}{P(\theta) \cdot \left(\prod_{i=1}^{K} P(\xi_i \mid \theta)P(\xi' \mid \theta')\right)}\right\} \tag{14}$$

Algorithm 3 Inner Sampler for Double MH

```
1: Input dataset $\mathcal{D}$ and reward parameter $\theta$
2: $\xi \leftarrow$ sample trajectory from $\mathcal{D}$
3: for each iteration do
4:     $\xi' \leftarrow$ sample from $\Xi$ near $\xi$
5:     if $e^{\beta\left(R(\xi',\theta) - R(\xi,\theta)\right)} > \alpha \sim \mathcal{U}[0,1]$ then $\xi \leftarrow \xi'$
6: Return $\xi$
```

Substituting in our human model and normalizing function:

$$\min\left\{1, \frac{P(\theta')Z(\theta)^K Z(\theta')^K \cdot e^{\beta \sum_{i=1}^{K}\left(R(\xi_i,\theta') + R(\xi',\theta)\right)}}{P(\theta)Z(\theta')^K Z(\theta)^K \cdot e^{\beta \sum_{i=1}^{K}\left(R(\xi_i,\theta) + R(\xi',\theta')\right)}}\right\}$$

Hence, the normalizing functions cancel out and we are left with the acceptance rule in Algorithm 2. We note that this Double MH approach also extends to learning rewards from state-action pairs (instead of trajectories) if we replace $R(\xi, \theta)$ with the Q-function (i.e., the cost-to-go).

Parameters. In our approach there are three main parameters for the designer to tune: a) the number of iterations in Algorithm 2, b) the number of iterations in Algorithm 3, and c) the rationality constant β. Increasing the number of outer and inner samples increases the expected accuracy of the learned θ [19], [22], but also leads to longer run-times[1]. For example, when tested on our working example from Section IV, Double MH took 20% longer to complete the same number of MCMC iterations as sampling or maximization baselines.

VI. Simulations

Here we compare approaches for approximating the normalizer and performing Bayesian reward learning in controlled environments. We simulate noisily rational humans who provide multiple, conditionally independent demonstrations. The space of rewards Θ is continuous, and we attempt to infer the simulated human's $\theta \in \Theta$ based on their demonstrations.

Independent Variables. We vary the robot's learning method across the algorithms introduced in Section V. This includes naïve robots that **Ignore** the normalizer, robots that approximate the normalizer using **Sampling** or **Maximum**, and our proposed **Double MH** approach. We also vary the simulated human's rationality β at two levels: noisy humans ($\beta = 5$) and consistent humans ($\beta = 25$). These values of β were identified through a preliminary round of simulations: below $\beta = 5$ humans acted almost completely randomly, and above $\beta = 25$ the humans converged to always select the optimal ξ.

Dependent Variable. Each human samples their true reward θ uniformly at random. We report the *Error* between the actual θ and $\hat{\theta}$, the mean of the robot's estimate: *Error* $= \|\theta - \hat{\theta}\|$.

Environments. We performed simulations across three dynamic physics-based environments where each environment had two features (see 4, 5, and 6). In *Push* the human's reward traded off between the distance the robot pushed the box and the length the robot travelled. In *Close*, the human's reward

[1]We provide our code, environments, and additional results in this repository. This includes an example of learning from state-action pairs.

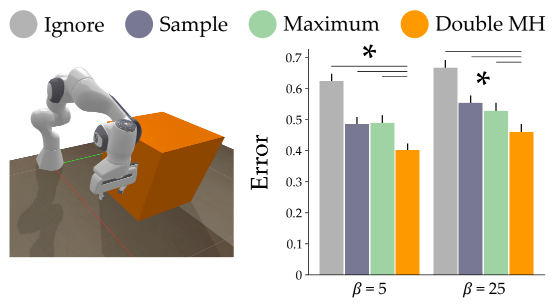

Fig. 4. Results from the *Push* simulation. (Left) The reward depends on the distance the box is moved and the distance the end-effector travels. (Right) Error in the learned θ across 100 simulated humans. Error bars show standard error, and an $*$ denotes statistical significance ($p < .05$).

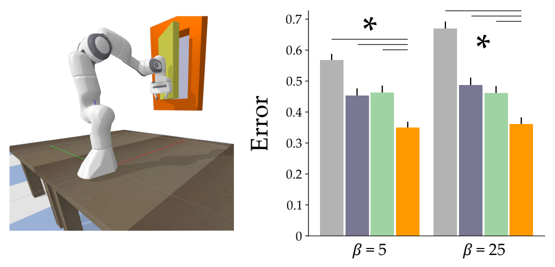

Fig. 5. Results from the *Close* simulation. (Left) The reward depends on the angle of the door and the robot's height from the table.

traded off between pushing the door closed (i.e., the angle of the door) and keeping the robot's end-effector close to the table (i.e., the robot's height). Finally, in *Pour*, the robot needed to pour coffee at a specific position, and the features were the distance travelled and holding the cup upright. In each environment, the robot had an accurate predictive model of the world and could simulate the outcomes of each trajectory.

Procedure. Each simulated human chose a θ vector uniformly at random. The human then generated $K = 3$ demonstrations of their desired motion so that $\mathcal{D} = \{\xi_1, \xi_2, \xi_3\}$. These demonstrations were sampled from our noisily rational model in Equation (2) and were conditionally independent (i.e., ξ_2 did not depend on ξ_1). The robot then observed $\mathcal{D}$ and used its Bayesian reward learning approach to get a mean estimate of θ. For each environment, we repeated this procedure across 100 simulated humans and reported the average results.

Results. Our results for *Push*, *Close*, and *Pour* are shown in Figures 4, 5, and 6. To analyze these results we first performed separate repeated measures ANOVAs on each environment and found that the normalizer approximation had a statistically significant effect. Post-hoc t-tests revealed that **Ignore** learned the *least accurate* estimate (i.e., had the most error) across the board. At the other end of the spectrum, **Double MH** resulted in the *most accurate* estimate for each environment and rationality β. Consider Figure 4 with $\beta = 5$ for instance: here t-tests show that **Double MH** has significantly lower error than **Ignore** ($t(99) = 7.8$, $p < .001$), **Sample** ($t(99) = 3.0$, $p < .05$), and **Maximum** ($t(99) = 3.7$, $p < .001$). Overall, our simulation results in these three physics-based tasks suggest that (a) ignoring the normalizer altogether leads to inaccurate inference, and (b) using Double MH sampling to approximate the normalizer outperforms existing approximation methods.

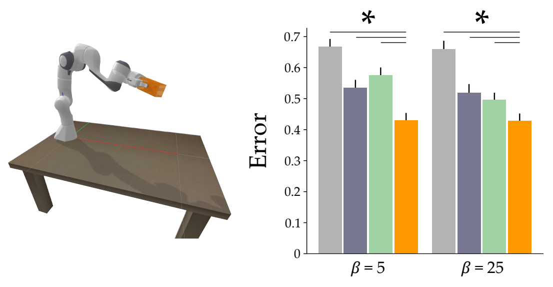

Fig. 6. Results from the *Pour* simulation. (Left) The reward depends on the orientation of the cup and the length of the robot's trajectory in joint space.

VII. User Study

We compared our proposed approach to existing approximations when learning rewards from actual users. In each task, the robot started with an initial trajectory and users physically *corrected* the robot arm to better align its motion with their objective. To standardize these results, we first displayed the desired trajectory that the human should teach to the robot (i.e., we specified the user's reward parameters θ). The participant's corrections were then used to infer an estimate of θ, and we compared what the robot learned to the objective that the human was trying to teach the robot.

Independent Variables. For this study, we varied the robot's learning along two factors: approximation type and conditional dependence. The robot used the **Ignore**, **Sample**, **Maximum**, and **Double MH** algorithms. We emphasize that **Ignore** [4]–[6], **Sample** [7]–[9], and **Maximum** [12], [14] come from prior work. We also compared *Conditionally Independent* and *Conditionally Dependent* versions of these algorithms. Recall from Section V-A that — when the robot treats the human's inputs as conditionally dependent — it recognizes that the human's current correction could build upon their prior corrections. Given that the human's corrections are sequential and interconnected, we anticipated that conditionally dependent learning would result in more accurate inference. To sample conditionally dependent corrections $\mathcal{D}'$ in Equation (13), we gave the robot an initial trajectory and then applied uniformly distributed perturbations to the waypoints along that trajectory.

Dependent Measures. We recorded each participant's corrections and applied Bayesian reward learning to infer their objective θ. As in Section VI, we compared the *Error* between the θ given to users and the robot's learned estimate $\hat{\theta}$: *Error* $= \|\theta - \hat{\theta}\|$. We also computed the *Regret* between the ideal trajectory ξ (i.e., the trajectory we showed to participants which optimizes for θ) and the learned trajectory $\hat{\xi}$ (i.e., the optimal trajectory under the robot's estimate $\hat{\theta}$).

$$Regret = R(\xi, \theta) - R(\hat{\xi}, \theta), \qquad \hat{\xi} = \arg\max_{\xi \in \Xi} R(\xi, \hat{\theta}) \tag{15}$$

Lower *Regret* means the robot has learned the correct behavior.

Experimental Setup. Users taught the robot three tasks (see Figure 7). One of these tasks (*Push*) was consistent with the Simulations in Section VI, and we introduced two new tasks to test the generality of our approach. In *Press* the robot traded off between pressing a button, reaching a goal, and avoiding an obstacle. In *Reach*, the robot tries to offer coffee to the user and then moves away after the coffee is delivered. Here *Press*

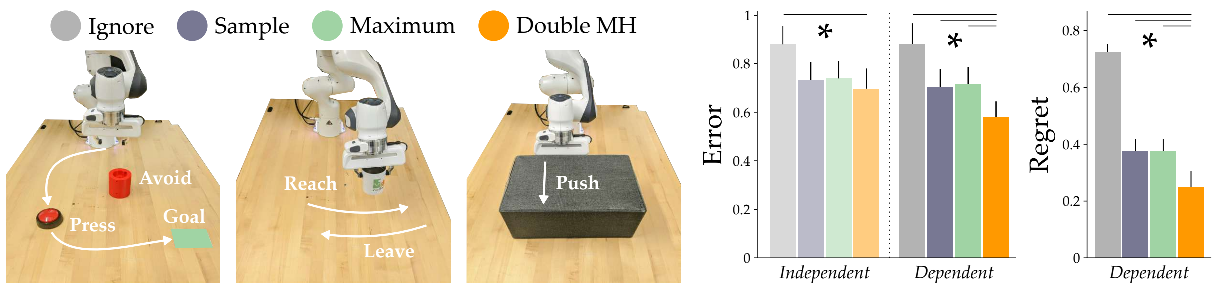

Fig. 7. Results from our user study in Section VII. (Left) The *Press*, *Reach*, and *Push* tasks. In each task, the robot moved along an initial trajectory, and users physically corrected the robot to teach it their desired behavior. (Right) *Error* and *Regret* averaged across the 10 users and three tasks. Lower *Regret* indicates that the robot's learned trajectory was better aligned with the desired behavior. Error bars show standard error, and an $*$ denotes $p < .001$.

had four features and *Push* and *Reach* each had three features. For each task, the human provided three separate sets of corrections for three different values of θ. Users first watched the robot demonstrate the ideal motion (i.e., the trajectory that optimized θ) then gave a sequence of corrections to convey that θ to the robot.

Participants. A total of 10 participants from the Virginia Tech community took part in this study (2 female, ages 27 ± 6.4 years). Eight of the ten users had interacted with robots before, and the other two users had no prior experience in robotics. Users provided written consent under IRB#20-755.

Results. Our results averaged across these 10 users and three tasks are summarized in Figure 7.

We first analyzed the effects of treating the human's corrections as conditionally independent or dependent. A repeated measures ANOVA revealed that conditionally dependent learning led to lower *Error* across the board: $F(1, 29) = 10.1$, $p < .001$. This result matched our intuition: it appeared that users often tried to fix something in their current correction based on what went wrong in the previous correction.

We next focused on the type of normalizer. Looking specifically at conditionally dependent learning, post-hoc analysis showed that **Double MH** resulted in lower *Error* and *Regret* as compared to each state-of-the-art alternative ($p < .001$). This suggests that — not only does **Double MH** lead to a more accurate estimate of the human's reward — but that estimate also results in robot trajectories that better match the human's desired behavior. See videos of our user study and the learned behaviors here: `https://youtu.be/EkmT3o5K5ko`

VIII. Conclusion

Our work is a step towards robot learners that infer the human's objective from demonstrations and corrections. In this paper, we explored the doubly-intractable nature of Bayesian reward learning, where the robot must reason over all possible trajectories and rewards. We grouped existing robotic approximations into three classes and theoretically derived their relative strengths and weaknesses. We then introduced a new Monte Carlo approximation method from the statistics community. Overall, our simulations and user studies suggest that this Double MH approach more accurately infers the human's objective, and is versatile enough to learn from independent demonstrations or interconnected corrections.

References

[1] H. J. Jeon, S. Milli, and A. Dragan, "Reward-rational (implicit) choice: A unifying formalism for reward learning," *NeurIPS*, 2020.

[2] D. Ramachandran and E. Amir, "Bayesian inverse reinforcement learning," in *IJCAI*, vol. 7, 2007, pp. 2586–2591.

[3] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *AAAI*, 2008.

[4] Y. Cui and S. Niekum, "Active reward learning from critiques," in *IEEE International Conference on Robotics and Automation*, 2018.

[5] D. S. Brown, Y. Cui, and S. Niekum, "Risk-aware active inverse reinforcement learning," in *Conference on Robot Learning*, 2018.

[6] E. Bıyık, D. P. Losey, M. Palan, N. C. Landolfi, G. Shevchuk, and D. Sadigh, "Learning reward functions from diverse sources of human feedback: Optimally integrating demonstrations and preferences," *The International Journal of Robotics Research*, vol. 41, pp. 45–67, 2022.

[7] A. Bobu, A. Bajcsy, J. F. Fisac, S. Deglurkar, and A. D. Dragan, "Quantifying hypothesis space misspecification in learning from human–robot demonstrations and physical corrections," *IEEE Transactions on Robotics*, vol. 36, no. 3, pp. 835–854, 2020.

[8] A. Jonnavittula and D. P. Losey, "I know what you meant: Learning human objectives by (under) estimating their choice set," in *IEEE International Conference on Robotics and Automation*, 2021.

[9] M. Kalakrishnan, P. Pastor, L. Righetti, and S. Schaal, "Learning objective functions for manipulation," in *IEEE International Conference on Robotics and Automation*, 2013, pp. 1331–1336.

[10] C. Finn, S. Levine, and P. Abbeel, "Guided cost learning: Deep inverse optimal control via policy optimization," in *ICML*, 2016.

[11] A. Boularias, J. Kober, and J. Peters, "Relative entropy inverse reinforcement learning," in *AISTATS*, 2011, pp. 182–189.

[12] M. Li, A. Canberk, D. P. Losey, and D. Sadigh, "Learning human objectives from sequences of physical corrections," in *IEEE International Conference on Robotics and Automation*, 2021, pp. 2877–2883.

[13] D. Hadfield-Menell, S. Milli, P. Abbeel, S. J. Russell, and A. Dragan, "Inverse reward design," *NeurIPS*, 2017.

[14] S. Levine and V. Koltun, "Continuous inverse optimal control with locally optimal examples," in *ICML*, 2012, pp. 475–482.

[15] A. D. Dragan and S. S. Srinivasa, "A policy-blending formalism for shared control," *The International Journal of Robotics Research*, vol. 32, no. 7, pp. 790–805, 2013.

[16] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, "An algorithmic perspective on imitation learning," *Foundations and Trends in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.

[17] C. L. Baker, R. Saxe, and J. B. Tenenbaum, "Action understanding as inverse planning," *Cognition*, vol. 113, no. 3, pp. 329–349, 2009.

[18] R. D. Luce, *Individual Choice Behavior: A Theoretical Analysis*. Courier Corporation, 2012.

[19] F. Liang, "A double Metropolis–Hastings sampler for spatial models with intractable normalizing constants," *Journal of Statistical Computation and Simulation*, vol. 80, no. 9, pp. 1007–1022, 2010.

[20] A.-M. Lyne, M. Girolami, Y. Atchadé, H. Strathmann, and D. Simpson, "On Russian roulette estimates for Bayesian inference with doubly-intractable likelihoods," *Statistical Science*, vol. 30, pp. 443–467, 2015.

[21] P. Alquier, N. Friel, R. Everitt, and A. Boland, "Noisy Monte Carlo: Convergence of Markov chains with approximate transition kernels," *Statistics and Computing*, vol. 26, no. 1-2, pp. 29–47, 2016.

[22] J. Park and M. Haran, "Bayesian inference in the presence of intractable normalizing functions," *Journal of the American Statistical Association*, vol. 113, no. 523, pp. 1372–1390, 2018.

[23] D. P. Losey, A. Bajcsy, M. K. O'Malley, and A. D. Dragan, "Physical interaction as communication: Learning robot objectives online from human corrections," *IJRR*, vol. 41, no. 1, pp. 20–44, 2022.

[24] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Pearson Education, Inc., 2022.