Including Uncertainty when Learning from Human Corrections

Dylan P. Losey
Rice University
dlosey@rice.edu

Marcia K. O'Malley
Rice University
omalleym@rice.edu

Abstract: It is difficult for humans to efficiently teach robots how to correctly perform a task. One intuitive solution is for the robot to iteratively learn the human's preferences from corrections, where the human improves the robot's current behavior at each iteration. When learning from corrections, we argue that while the robot should estimate the most likely human preferences, it should also *know what it does not know*, and integrate this uncertainty as it makes decisions. We advance the state-of-the-art by introducing a Kalman filter for learning from corrections: this approach obtains the uncertainty of the estimated human preferences. Next, we demonstrate how the estimate uncertainty can be leveraged for *active learning* and *risk-sensitive* deployment. Our results indicate that obtaining and leveraging uncertainty leads to faster learning from human corrections.

Keywords: human-robot interaction (HRI), inverse reinforcement learning (IRL)

1 Introduction

While robots can be pre-programmed by an expert designer to execute a wide range of behaviors, each robot user has different preferences for how their robot should behave. Recent work has focused on learning the human end-user's preferences from *corrections*: here the robot shows the human how it has been pre-programmed to perform the task, and the human corrects the robot's behavior to suit their personal preferences. Importantly, these corrections do not need to be perfect; instead, a human correction is simply a *noisy improvement* of the robot's current behavior.

Consider a robotic manipulator carrying a cup of coffee for its human end-user. This robot knows to avoid obstacles, but is not sure about the human's preferences: e.g., should the robot carry coffee over a laptop, across a table, or avoid both regions? When learning from corrections, the robot shows the human its current estimate of the optimal trajectory. The human then corrects this trajectory—using physical human-robot interaction or a virtual interface—and, for example, pushes the robot farther away from the laptop. The robot learns iteratively (i.e., online), and updates its understanding of the human's preferences after each correction. See Fig. 1 for an overview of this process.

Human corrections indicate which preferences are more probable. For instance, if the human pushes the robot away from the laptop, then the robot should infer that preferences which result in the robot avoiding the laptop are more likely. Within the state-of-the-art, robots estimate the *most likely* human preferences given the human's corrections [1, 2]. However, these robots miss out on the *uncertainty* of their estimate: i.e., in practice, the robot may not understand the human's preferences with much confidence. Our insight is that—because human corrections imply a probability distribution over their preferences—*the robot should not only estimate the most likely human preferences from these corrections, but should also recognize which estimates it is not confident about*.

Let us return to our working example, where a user wants the robot to avoid carrying coffee over their laptop. Because the human pushed the robot away from their laptop—and the table is nearby—the robot learns to avoid both the laptop and table. If the robot only estimates the most likely human preferences, it will avoid carrying coffee over the table. But the human may actually want the robot to move over the table! A robot which knows the uncertainty of its estimate is *confident* that it should avoid the laptop, but *unsure* whether it should avoid the table. We can leverage this uncertainty to elicit informative corrections (that teach the robot about the human's table preference) or for risk-sensitive deployment (that avoids the table entirely) when more corrections are unavailable.

Contributions. First, we show how iterative inverse reinforcement learning can be performed using a *Kalman filter*, where human corrections are noisy observations. This approach extends the state-of-the-art to now track the uncertainty over the estimated preferences. Next, we leverage uncertainty within our setting to *actively learn* from human corrections, so that the robot can elicit corrections

2nd Conference on Robot Learning (CoRL 2018), Zürich, Switzerland.

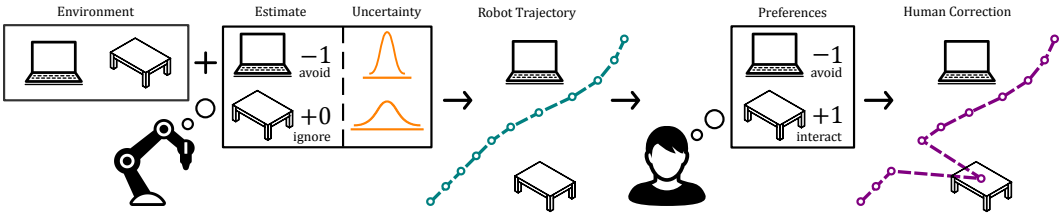

Figure 1: Iterative learning from human corrections. Given an environment and the current estimated preferences, the robot selects a trajectory (teal). The human then observes this trajectory, and provides a correction to better match their true preferences (purple). Traditionally, the robot uses the correction to update its estimate at the next iteration. We propose that the robot should also obtain the uncertainty over this estimate (orange).

that will reduce uncertainty. After the learning is completed, and the human stops providing corrections, we also describe how uncertainty can be leveraged for *risk-sensitive deployment*, i.e., to avoid interacting with preferences about which the robot is most uncertain.

2 Related Work

The problem we are considering is based on learning from human corrections, an instance of inverse reinforcement learning (IRL). Our solution will also build upon active learning approaches, where the robot reasons over uncertainty during IRL. Here we briefly overview both fields.

Inverse reinforcement learning. Also known as inverse optimal control, IRL attempts to recover the human's preferences from demonstrations which are optimal [3, 4, 5]. In practice, however, it is challenging for humans to provide optimal demonstrations: consider an end-user trying to guide the motion of a multi degree-of-freedom (DoF) robotic manipulator [6]. One solution is presented by probabilistic IRL approaches [7, 8], which assume that the human is noisily optimal, and learn a distribution over the space of possible human preferences.

Alternatively, the robot can learn from corrections. At each iteration the robot maximizes its current estimate of the human's preferences, and the human responds by slightly improving, or correcting, the robot's behavior. Here the human's corrections do not need to be noisily optimal. Shivaswamy and Joachims [2] model learning from corrections as Coactive Learning, and derive iterative IRL algorithms which are similar to [1]. In particular, the Preference Perceptron from [2] has been applied to robotic manipulators by [9, 10, 11]. While these works learn a maximum a posteriori estimate of the human's objective, we note that they do not maintain the uncertainty over this estimate.

Active learning. Other works explicitly reason over uncertainty while learning from the human. For instance, active learning reduces uncertainty by enabling the robot to choose informative queries, which are then answered by the expert human [12]. Active learning has previously been applied to improve IRL in [13, 14, 15]. Most relevant to our work is recent research by Cui and Niekum [15], where the robot proposes a trajectory to the human, and the human segments this trajectory into "good" and "bad" portions. The robot updates its understanding of the human's preferences based on this segmentation; moreover, to increase its learning rate, the robot actively generates trajectories which are expected to result in user critiques that best reduce uncertainty. Our research is similar to [15], but here we focus on learning from human corrections rather than user segmentation.

3 Background

Within this section we derive the Preference Perceptron, the current state-of-the-art approach when learning from human corrections [1, 2, 9]. We note that the Preference Perceptron is equivalent to online Maximum Margin Planning without any loss function.

Notation. Consider a robot with state $x \in \mathcal{X}$, action $a \in \mathcal{A}$, and dynamics f. These dynamics define the probability distribution over the robot's next state given its current state and action: i.e., $f(x^{i+1}|x^i, a^i)$, where i denotes the current timestep. The task ends after $T \in \mathbb{Z}^+$ timesteps.

Trajectory and Environment. Let us define the robot's trajectory $\xi \in \Xi$ as the sequence of robot states x, such that $\xi = x^{0:T}$. Although this trajectory describes the robot's behavior, it does not tell us about the world in which the robot is acting. Accordingly, let E denote the robot's environment; we can equivalently think of E as the "world description" [16], or as the "context" [2]. We include the prior distribution over the robot's start state x^0 within the environment E.

Reward. The human end-user has in mind a reward function, R, which determines the utility of the robot following trajectory ξ in environment E. Like previous IRL works [1, 3, 5, 7], we will assume that R is a linear combination of features $\phi(\xi, E) \in [0, 1]^k$ weighted by a parameter vector $\theta \in \mathbb{R}^k$:

$$R(\xi, E) = \theta \cdot \sum_{i=0}^{T} \phi(x^i, E) = \theta \cdot \phi(\xi, E) \tag{1}$$

The features ϕ are known by both the human and the robot. Given θ, we have described an instance of a Markov decision process [17] that can be solved to find the optimal robot policy. In practice, however, the true reward parameter θ is *known only by the human*. In other words, the choice of θ is user-specific: θ encodes the human's *preferences* over the robot's trajectory, and varies from one end-user to another [11]. Hence, the robot must learn θ from the current end-user.

Corrections. The robot learns about the human's true preferences θ from the human's corrections. At each iteration t, the robot observes an environment E^t and chooses a trajectory ξ^t. The human end-user observes both E^t and ξ^t, and corrects the robot's trajectory to ξ_h^t. We assume that the human's corrected trajectory has higher reward than the robot's original trajectory:

$$R(\xi_h^t, E^t) > R(\xi^t, E^t) \tag{2}$$

$$\theta \cdot \phi(\xi_h^t, E^t) > \theta \cdot \phi(\xi^t, E^t) \tag{3}$$

where we have applied the reward function (1). Intuitively, here we are claiming that the human sees the robot's behavior, and then modifies that behavior so that the robot's actions better align with their preferences. Notice that the human does not have to correct the robot to the optimal trajectory—i.e., provide a noisily optimal demonstration—but only needs to slightly *improve* the robot's trajectory.

Preference Perceptron. Given that the human's correction returns an improved trajectory, the current state-of-the-art robot learns with the Preference Perceptron. Let $\hat{\theta}$ be the robot's estimate of the human's true preferences θ. At each iteration t, the robot updates $\hat{\theta}$ to maximize the margin between the estimated rewards associated with ξ^t and ξ_h^t such that $\hat{\theta} \cdot \phi(\xi^t, E^t) < \hat{\theta} \cdot \phi(\xi_h^t, E^t)$. Put another way, the robot minimizes the following cost function:

$$J\left(\hat{\theta}\right) = \hat{\theta} \cdot \left[\phi(\xi^t, E^t) - \phi(\xi_h^t, E^t)\right] \tag{4}$$

Since J is differentiable with respect to $\hat{\theta}$, we leverage online gradient descent [18] to get:

$$\hat{\theta}^{t+1} = \hat{\theta}^t + \alpha^t \left[\phi(\xi_h^t, E^t) - \phi(\xi^t, E^t)\right] \tag{5}$$

where $\alpha > 0$ is the *learning rate*. This update rule (5) is referred to as the Preference Perceptron. Intuitively, a robot leveraging (5) learns by comparing feature counts between the corrected and original trajectories: features that the human has increased are weighted more highly, while features that the human has decreased are weighted less highly. Prior work has demonstrated that (5) is also the robot's *maximum a posteriori* (MAP) estimate of the human's true preferences θ [9].

Optimal Trajectory. Given $\hat{\theta}^t$ and E^t, the robot can identify an optimal trajectory that maximizes its current estimate of the human's reward. We obtain this trajectory by solving:

$$\xi^t = \arg\max_{\xi \in \Xi} \ \hat{\theta}^t \cdot \phi(\xi, E^t) \tag{6}$$

For robotic manipulators, a trajectory optimizer such as [19, 20] can be leveraged to solve (6).

Summary. The robot observes an environment E^t at each iteration t. Based on E^t and the robot's current estimate of θ, the robot solves (6) for its trajectory ξ^t. The human then corrects the robot's trajectory, and provides an improved trajectory ξ_h^t. Finally, the robot updates its estimate of θ using (5), and the process repeats at the next iteration. We can alternatively think of ξ^t as the label which the robot assigns to the input E^t, while ξ_h^t is an improved label provided by the human's correction.

Uncertainty. Although (5) provides a maximum a posteriori estimate of θ, this Preference Perceptron does not obtain a probability distribution over θ. Thus, when the robot learns using (5), we do not know the uncertainty of our estimate $\hat{\theta}$. Instead, a robot using the Preference Perceptron falsely assumes that it completely understands the human's preferences after each correction.

4 Kalman Filter for Inverse Reinforcement Learning

Our first contribution is to recognize that (5)—the standard IRL update rule for learning from human corrections—can be rewritten as a Kalman filter [21]. We argue that the key advantage to using a Kalman filter is that it not only provides an iterative estimate similar to (5), but it also *obtains the uncertainty over this estimate*. Here we explain how to apply a Kalman filter for iterative IRL.

Transition Model. We model the human's preferences θ^t as constant between iterations, like previous IRL works [1, 3, 5, 7]. Then, we can write the transition function:

$$\theta^t = \theta^{t-1} + m^t \qquad m^t \sim \mathcal{N}(0, M^t) \tag{7}$$

where m^t is the *process noise* at iteration t. We assume that m^t is drawn from a zero-mean Gaussian distribution with covariance M^t. Introducing this process noise enables the robot to capture how the imperfect human may unintentionally alter their preferences between iterations, i.e., the human may not know exactly what they want: (7) implies that the end-user's preferences are *noisily constant*.

Observation Model. The robot learns about the end-user's preferences θ^t by observing the human's corrected trajectory ξ_h^t, and—more specifically—by observing the features ϕ along that corrected trajectory. Accordingly, the robot has an observation model:

$$\phi(\xi_h^t, E^t) = \phi\big(\xi_*(\theta^t, E^t), E^t\big) + n^t \qquad n^t \sim \mathcal{N}(0, N^t) \tag{8}$$

where ξ_* is the true correction the human intends to provide, and n^t is the *observation noise* at iteration t. We again assume that n^t is drawn from a zero-mean Gaussian distribution with covariance N^t. Here observation noise indicates that actual end-users are unable to provide exactly the feature counts that they have in mind: e.g., it is challenging to perfectly guide a multi-DoF manipulator [6].

Biased Feedback. Although (8) assumes that the observation noise is unbiased, this simplification may not always be true in practice. We therefore perform simulations where the human's feedback is biased in Section 6 to support our Kalman filter approach for cases where n^t is not Gaussian noise.

Intended Correction. In (8), ξ_* is the correction the human intends to provide given that their current preferences are θ^t and the environment is E^t. Many choices of ξ_* are possible. To be consistent with previous works, we here assume that the human intends to give the following correction:

$$\xi_*(\theta^t, E^t) = \arg\max_{\xi \in \Xi} \ \theta^t \cdot \phi(\xi, E^t) \tag{9}$$

Recalling (6), notice that ξ_* is now the *optimal trajectory* that maximizes R. We recognize that modeling the human as intending to provide the optimal trajectory (9) and then incorporating Gaussian observation noise over the feature counts (8) *is analogous to noisy optimal demonstrations* [7, 8].

IRL as a Dynamical System. Together (7)–(9) express iterative IRL as a dynamical system, where the human's preferences are noisily constant, and the human demonstrates approximately optimal feature counts as a function of the their hidden preferences. We want to estimate these preferences.

Extended Kalman Filter. Given the transition model (7) and observation model (8), we can leverage a Kalman filter to obtain an optimal estimate of the human's preferences θ^t [21]. To be more precise, since the observation model is here nonlinear, we apply an *extended Kalman filter* (EKF). This EKF linearizes the observation model around the current estimated preferences, and then acts as a standard Kalman filter. We point out that recent developments, such as the *unscented Kalman filter* (UKF) [22], may outperform an EKF [23]. For simplicity of exposition—as well as the insight it provides—we here present the EKF, while noting that we implement the UKF in our simulations.

Preference Estimate and Covariance. Let $\hat{\theta}^t$ be the *mean estimate* of the human's preferences, and let P^t be the *covariance* (i.e., uncertainty) *of this estimate*. Here we list the steps to update the estimate and covariance matrix via an EKF. First, we use a Taylor series expansion to linearize the observation model (8) around the current estimate, and reach the observation Jacobian $H \in \mathbb{R}^{k \times k}$:

$$H^t = \frac{\partial \phi}{\partial \xi_*} \cdot \frac{\partial \xi_*}{\partial \theta}\bigg|_{\hat{\theta}^t} \tag{10}$$

Intuitively, H tells us how the intended feature counts will vary as the human's preferences change. We expect the performance of our EKF to improve when (10) is approximately linear. Applying H, we can now write the EKF update rule for iterative IRL:

$$\hat{\theta}^{t+1} = \hat{\theta}^t + K^t\big[\phi(\xi_h^t, E^t) - \phi(\xi_*(\hat{\theta}^t, E^t), E^t)\big] \tag{11}$$

Because ξ_* within (11) is the optimal trajectory given $\hat{\theta}^t$ and E^t, we see that $\xi_* = \xi^t$ from (6). Therefore, we substitute $\xi_* = \xi^t$ into (11) to finally derive our *proposed update rule*[1]:

$$\hat{\theta}^{t+1} = \hat{\theta}^t + K^t\big[\phi(\xi_h^t, E^t) - \phi(\xi^t, E^t)\big] \tag{12}$$

[1]Read $\hat{\theta}^{t+1}$ as the mean estimate of θ at iteration t, given the observed feature counts after t iterations.

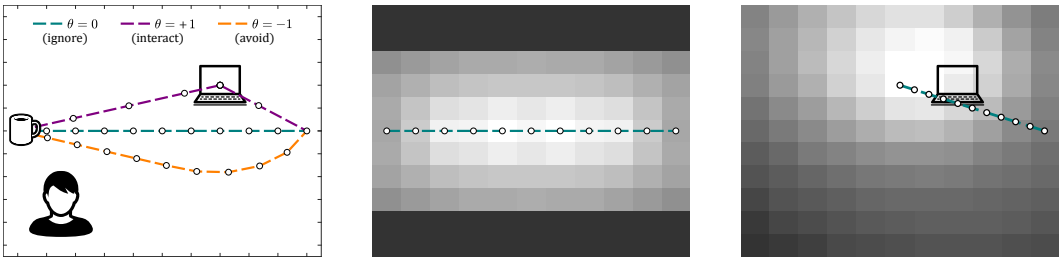

Figure 2: Selecting an environment to minimize uncertainty. **Left**: The robot is carrying a cup of coffee to a goal location, while a human stands nearby. The robot does not know whether it should carry the coffee over the user's laptop (three possible preferences are given). **Middle**: The robot's current trajectory ξ is shown. We consider where best to place the laptop to *minimize the robot's covariance* P (prior to the human's correction). Lighter grid cells indicate laptop positions that will minimize P. **Right**: Alternatively, we could fix the laptop location (as shown), and then vary the robot's start location. Lighter grid cells indicate start locations that will best minimize the robot's covariance. We used a UKF [22] and TrajOpt [20] to perform these simulations.

Comparing this proposed update rule (12) to the Preference Perceptron (5), we have straightforwardly replaced the *learning rate* $\alpha > 0$ with the *Kalman gain matrix* $K \in \mathbb{R}^{k \times k}$:

$$K^t = (P^t + M^t)(H^t)^T \left[H^t(P^t + M^t)(H^t)^T + N^t\right]^{-1} \tag{13}$$

Since we are now using a Kalman filter, however, we additionally obtain the *covariance matrix of the estimate*, $P \in \mathbb{R}^{k \times k}$, which is updated according to:

$$P^{t+1} = (I - K^t H^t)(P^t + M^t) \tag{14}$$

Summary. After formulating iterative IRL as a dynamical system with state θ, we derived a Kalman filter estimate of the human's preferences. This approach has *extended* the Preference Perceptron (5) by adding the Kalman gain matrix, K from (13), and the covariance matrix, P from (14). Our resulting update rule (12) is proportional to the Preference Perceptron (5), but now we have additionally obtained the covariance (i.e., the uncertainty) of the estimated human preferences (14).

5 Leveraging Uncertainty when Learning from Corrections

We showed that the Preference Perceptron can be extended to include uncertainty via a Kalman filter: but how should we leverage this uncertainty? In this section, we explore how the covariance of the robot's estimate, P, can be used to *actively learn* from human corrections, and then *safely deploy* a resultant trajectory. We consider examples consistent with previous applications of learning from corrections [9, 10, 11] where: (a) the robot can select virtual environments to elicit more informative human corrections, and (b) the robot deploys with risk-averse or risk-sensitive behavior.

Minimizing Covariance. One reasonable goal for a robot that is learning the human's preferences θ is to *minimize its uncertainty over those preferences*. When uncertainty is high, the robot is unsure about how it should behave, and when uncertainty is low, the robot is confident that it understands the human's preferences. Within our Kalman filter approach, the robot should therefore elicit corrections that minimize the covariance matrix P. Eliciting these corrections is an instance of active learning.

Active Learning. The robot can elicit corrections that reduce uncertainty by altering aspects of the environment $E \in \mathcal{E}$ in which those corrections are provided. For instance, the robot might change its start state. Alternatively—because the robot is learning from corrections—we can use simulated (i.e., virtual) environments for learning [11]. We here perform simulations for both settings: some where only the start state can be changed, and others where the virtual environment can be altered.

Greedy Start and/or Environment Selection. At each iteration t, the robot greedily minimizes its uncertainty *regardless of the human's actual correction* by selecting $E^t \in \mathcal{E}$:

$$E^t = \arg\min_{E \in \mathcal{E}} \|P^{t+1}\|_F = \arg\min_{E \in \mathcal{E}} \|(I - K^t H^t)(P^t + M^t)\|_F \tag{15}$$

In the above, $\|\cdot\|_F$ is the Frobenius norm (although other norms can be used). Note that H depends on E from (10), and so K also depends on E from (13). As pointed out by [24], we can evaluate the uncertainty P^{t+1} in advance—i.e., before the human provides a correction—and hence we can solve (15) to select an environment *without knowing what correction the human will actually give*.

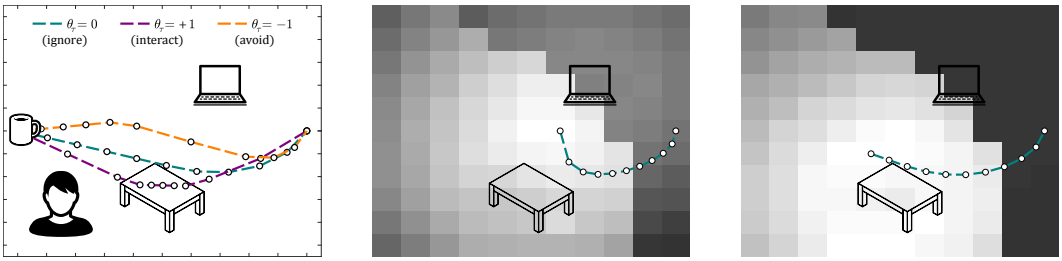

Figure 3: Selecting an environment to minimize uncertainty with multiple features. **Left**: The task is the same as Fig 2, but now with a table which the human might prefer for the robot to move across. While the robot has previously learned to avoid the laptop, it does not know θ_τ, the human's true preference for the table. **Middle**: Assuming that the laptop and table have a fixed position, the robot searches for the best start location. Here the initial covariance is the same for both features. **Right**: Next, we increase the initial covariance over $\hat{\theta}_\tau$. This is meant to emulate situations where *one feature is well understood*, while the robot is *uncertain about another feature*. As before, lighter grid cells will minimize the robot's uncertainty, and we used a UKF with TrajOpt.

Intuition. Fig. 2 demonstrates how we can leverage greedy environment selection. Inspecting these results, we see that environments where small changes in θ lead to large changes in ϕ better reduce uncertainty; put another way, we generally want to *maximize* H. For example, consider the middle simulation in Fig. 2. When the laptop is too far away from the robot's current optimal trajectory ξ, local corrections do not alter the feature counts, and so the robot cannot learn from this environment.

Multiple Features. We next use (15) to select informative environments when the robot is uncertain about multiple features; here the robot must trade-off between learning different features (see Fig. 3). We find that—if the covariance over each feature is equal—interacting with all features is optimal. On the other hand, when the robot has greater uncertainty over a specific feature, the greedy robot favors environments that elicit corrections on that feature. In Fig. 3, the robot chooses a start location *between* the laptop and table when the initial uncertainty is equal, but biases its starting location *towards the table* when it has greater uncertainty about the table feature. A robot using (15) will select environments where the current robot trajectory *interacts with the most uncertain features*.

Risk-Sensitive Deployment. After the robot has learned from the human's corrections and is deployed to perform the task (without human feedback), we can leverage the covariance matrix P^t to select *safer* robotic behavior. Recall that the robot's trajectory optimizes (6) based on the estimated preferences $\hat{\theta}^t$. Planning only with $\hat{\theta}^t$ fails to account for the covariance over this estimate: the robot might be confident about some learned preferences, but unsure about others. Hence, we will use a *risk-averse* trajectory planning approach similar to [25]. First, we generate a set of preferences Γ^t:

$$\Gamma_0^t = \hat{\theta}^t; \quad \Gamma_i = \hat{\theta}^t + \left(\sqrt{P^t}\right)_i, \;\; i = 1, \dots, k; \quad \Gamma_i = \hat{\theta}^t - \left(\sqrt{P^t}\right)_{i-k}, \;\; i = k+1, \dots, 2k \quad (16)$$

Here $\left(\sqrt{P^t}\right)_i$ is the i-th column of the matrix square root of P^t. The robot now has $2k+1$ estimates of θ, where Γ_0^t is the Kalman filter estimate, and $\Gamma_{1:2k}^t$ are one standard deviation away (as defined by the current covariance P^t). Our risk-averse robot optimizes the worst-case reward over Γ^t:

$$\xi^t = \arg\max_{\xi \in \Xi} \left\{ \min_{\gamma \in \Gamma^t} \; \gamma \cdot \phi(\xi, E^t) \right\} \quad (17)$$

By extension, a *risk-seeking* robot optimizes the best-case reward over Γ^t, i.e., uses $\max$ in (17), and a *risk-neutral* robot simply optimizes with the mean estimate Γ_0^t, which reduces to (6).

Simplification. In practice, solving (17) for multi-DoF robotic manipulators moving in continuous spaces is challenging. One particular concern is local minima, which naturally occur during trajectory optimization [20, 26]; this problem is now compounded in (17) by a nested optimization. To make risk-sensitive planning more tractable, we will *reverse* the order of optimization:

$$\gamma^t = \arg\min_{\gamma \in \Gamma^t} \left\{ \max_{\xi \in \Xi} \; \gamma \cdot \phi(\xi, E^t) \right\} \quad (18)$$

In the above, we first find the best possible reward for each preference in Γ^t, and then choose the worst-case preference γ^t. Finally, we use $\gamma^t = \hat{\theta}^t$ in (6), and obtain the risk-adverse trajectory. We demonstrate the results of risk-sensitive deployment using this simplification in Fig. 4.

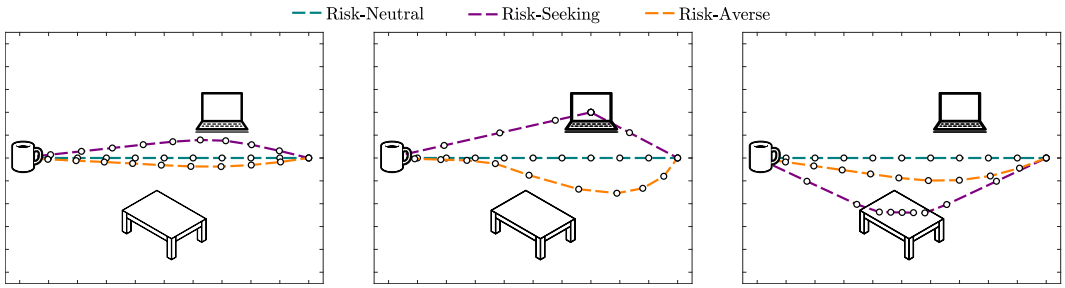

Figure 4: Risk-sensitive deployment based on uncertainty. The robot is performing the same task as in Fig. 3, but now without human supervision. **Left**: If the robot has little uncertainty, risk-sensitive planning is almost the same as risk-neutral planning. **Middle**: When we increase the uncertainty, the robot attempts to *decrease* feature counts (risk-averse) or *increase* feature counts (risk-seeking) as compared to the learned preferences (risk-neutral). **Right**: Previously, the uncertainty over both the table and laptop features was equivalent. Here the robot is confident about the laptop feature, but uncertain as to whether it should avoid crossing the table. We used (16) and (18) together with TrajOpt to perform these simulations in a continuous state space.

Summary. We leveraged the robot's uncertainty during both *learning* and *deployment*. We demonstrated how the robot could actively learn by adjusting its start state (or, more generally, the environment) to elicit corrections from the human that reduce the robot's uncertainty (15). We intuitively found that this active learning caused the robot to interact with the features about which it was most unsure (see Figs. 2 and 3). Next, when the robot is deployed—i.e., no more human corrections are provided—we showed how the robot could exploit uncertainty during risk-sensitive planning. We described a simplified approach for risk-sensitive planning in (16) and (18) that is tractable for continuous state spaces. Risk-averse planning using this approach resulted in robots that avoided interacting with preferences that were not clearly understood (see Fig. 4, risk-averse case).

Thus, when the robot knows what it does not know, *the robot explores preferences with high uncertainty while learning from human corrections* (learning), *and then avoids preferences with high uncertainty after the human stops providing corrections* (deployment).

6 Learning Simulations

In order to support our proposed Kalman filter approach (Section 4) and demonstrate its active learning application (Section 5), we here perform user simulations where the robot iteratively learns from human corrections. We compare robots that learn with the *Preference Perceptron (PP)*, robots that learn with our *Kalman Filter (KF)* approach, and robots that leverage this Kalman filter for *Active Learning (AL)*. The simulated human end-user provides a correction at each iteration: importantly, these imperfect corrections are biased, and violate our Gaussian observation noise assumption from (8). We hypothesize that—even though the simulated human behavior does not match our Kalman filter assumptions—robots that obtain and reason over uncertainty (KF and AL) will learn from human corrections faster than the state-of-the-art (PP).

Setup. The robotic manipulator is carrying a cup of coffee, and is unsure whether the user would prefer for the robot to move over a laptop and/or across a table (see Fig. 3). At each iteration t, the robot observes an environment E^t and executes the optimal trajectory given that environment and $\hat{\theta}^t$, its current estimate of the user's preferences. There are 48 possible environments $E \in \mathcal{E}$: these environments have different start states, laptop locations, and table locations. The KF and PP robots are given environments *uniformly at random*, while the AL robot *greedily selects* E^t using (15).

Biased Users. The simulated human end-user observes E^t and corrects the robot's trajectory ξ^t at each iteration. This realistic user does not provide optimal or noisily optimal corrections; instead, the human's corrections are biased improvements. More specifically, the simulated human corrects the robot's trajectory, ξ^t, by moving *one waypoint* from ξ^t towards the equivalent waypoint along their intended trajectory, ξ^t_*. The human corrects only the waypoint with the largest error. We emphasize that the simulated human therefore violates our Gaussian noise assumption from (8), and tests the robustness of our Kalman filter approach within a realistic learning from corrections setting [11].

Implementation. The PP and KF robots were simulated 100 times to obtain their expected performance across randomly selected environments; since the AL robot deterministically selects environments, it was only simulated once. To ensure that the learning rate α for PP was consistent with the Kalman gain K for KF and AL, we set α^t as the expected mean value of the matrix diagonal of K^t across all KF simulations. Like before, we used TrajOpt [20] to obtain the optimal robot trajectory ξ^t, and we used an unscented Kalman filter (UFK) [22] for the KF and AL robots.

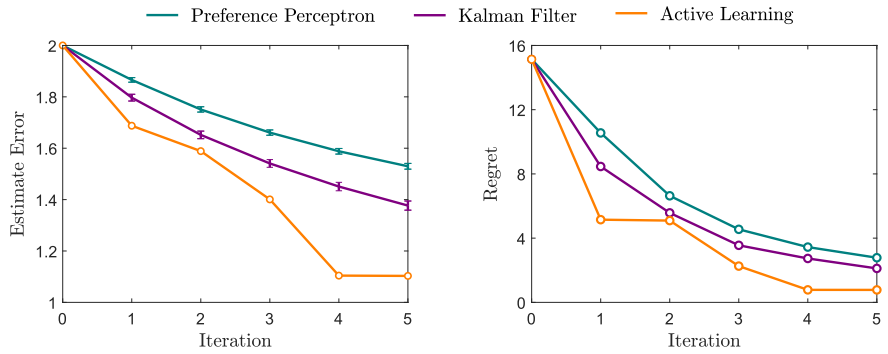

Figure 5: Comparing PP, KF, and AL when iteratively learning from corrections. In PP, the robot learns a maxmium a posteriori estimate. In KF, the robot also obtains the uncertainty of this estimate. In AL, we use the uncertainty to elicit more informative corrections. Here the true preferences are $\theta = (+1, -1)$, and the robot's initial estimate is $\hat{\theta}^0 = (0, 0)$. The initial covariance is equal over both features, such that $P^0 = I$. Error bars show standard error of the mean. We found *Regret* using the expected values of $\hat{\theta}^t$ for PP and KF.

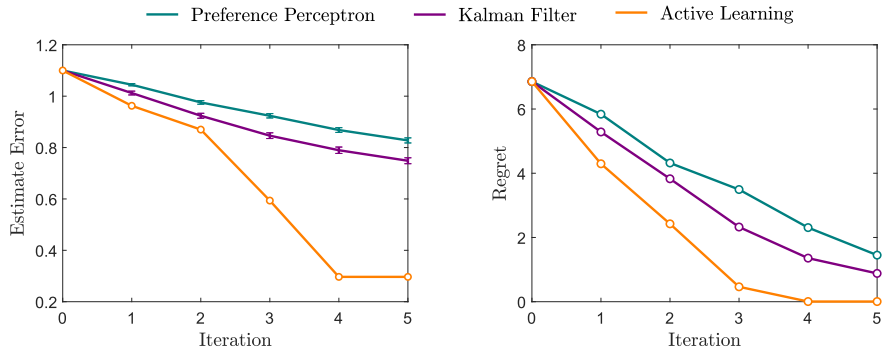

Figure 6: Comparing PP, KF, and AL when iteratively learning from human corrections. Unlike Fig. 5, here the robot starts with an accurate estimate of θ_c, the human's true preference for the coffee, but a poor estimate of θ_τ, the human's true preference for the table. The true preferences are still $(\theta_c, \theta_\tau) = (+1, -1)$, but the robot's initial estimate is $\hat{\theta}^0 = (+0.9, 0)$. The robot's initial covariance over the table weight is higher than the initial covariance over the coffee weight, such that $P^0 = \text{diag}(10^{-2}, 1)$.

Results. Our results are summarized in Figs. 5 and 6. Within these plots, *Estimate Error* refers to the difference between the true preferences, θ, and the robot's estimate, $\hat{\theta}$. We calculated this metric with the L_1 norm: $\|\hat{\theta}^t - \theta\|_1$. *Regret* captures the difference between the reward the robot would receive if it knew θ, and the reward the robot actually receives using its estimate $\hat{\theta}$. *Regret* is found by comparing ξ_*^t, the optimal trajectory given θ, and ξ_t, the optimal trajectory given $\hat{\theta}^t$. Recalling (1), *Regret* equals: $\theta \cdot \phi(\xi_*^t, E^t) - \theta \cdot \phi(\xi^t, E^t)$. We summed *Regret* across all environments $E \in \mathcal{E}$.

Discussion. Based on our results, KF and AL outperformed the state-of-the-art PP in terms of both *Estimate Error* and *Regret*. From Fig. 5, we found that KF and AL led to faster learning than PP when the robot's initial uncertainty over the human's preferences was uniform. In Fig. 6, we found that AL was especially advantageous when some aspects of the human's preferences were initially well understood, but the robot was uncertain about others. These results also demonstrate that our Kalman filter approach is effective even when the human corrections are biased.

7 Conclusion

When learning from human corrections, we argue that the robot should recognize which preferences are well understood, and which user preferences remain uncertain. We therefore proposed a Kalman filter approach to iterative IRL, which extends the state-of-the-art by obtaining the uncertainty over the learned human preferences. We then demonstrated how the robot can leverage (a) active learning to reduce this uncertainty when learning from human corrections, and (b) risk-sensitive deployment to avoid uncertain preferences after the human stops providing corrections. Our user simulations showed that the proposed approach results in faster learning than the current state-of-the-art, even for cases where the biased human corrections do not match our Kalman filter assumptions.

Acknowledgments

We would like to thank the reviewers for their thoughtful and encouraging advice. We would also like to thank Scott Niekum for reaching out and bringing related work to our attention.

This project was funded in part by the NSF GRFP-1450681. The authors are both members of the Mechatronics and Haptic Interfaces (MAHI) Laboratory, Department of Mechanical Engineering, Rice University, Houston, TX 77005.

References

[1] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich. Maximum margin planning. In *Proc. International Conference on Machine Learning (ICML)*, pages 729–736, 2006.

[2] P. Shivaswamy and T. Joachims. Coactive learning. *Journal of Artificial Intelligence Research*, 53:1–40, 2015.

[3] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proc. International Conference on Machine Learning (ICML)*, 2004.

[4] A. Y. Ng and S. J. Russell. Algorithms for inverse reinforcement learning. In *Proc. International Conference on Machine Learning (ICML)*, pages 663–670, 2000.

[5] 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*, 7(1-2):1–179, 2018.

[6] B. Akgun, M. Cakmak, K. Jiang, and A. L. Thomaz. Keyframe-based learning from demonstration. *International Journal of Social Robotics*, 4(4):343–355, 2012.

[7] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *Proc. Association for the Advancement of Artificial Intelligence (AAAI)*, volume 8, pages 1433–1438, 2008.

[8] D. Ramachandran and E. Amir. Bayesian inverse reinforcement learning. *Urbana*, 51(61801): 1–4, 2007.

[9] A. Bajcsy, D. P. Losey, M. K. O'Malley, and A. D. Dragan. Learning robot objectives from physical human interaction. In *Prof. Conference on Robot Learning (CoRL)*, pages 217–226, 2017.

[10] A. Bajcsy, D. P. Losey, M. K. O'Malley, and A. D. Dragan. Learning from physical human corrections, one feature at a time. In *Proc. ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 141–149, 2018.

[11] A. Jain, S. Sharma, T. Joachims, and A. Saxena. Learning preferences for manipulation tasks from online coactive feedback. *The International Journal of Robotics Research*, 34(10):1296–1313, 2015.

[12] B. Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

[13] M. Lopes, F. Melo, and L. Montesano. Active learning for reward estimation in inverse reinforcement learning. In *Proc. Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, pages 31–46, 2009.

[14] R. Cohn, E. Durfee, and S. Singh. Comparing action-query strategies in semi-autonomous agents. In *Proc. International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1287–1288, 2011.

[15] Y. Cui and S. Niekum. Active reward learning from critiques. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[16] S. H. Huang, D. Held, P. Abbeel, and A. D. Dragan. Enabling robots to communicate their objectives. In *Proc. Robotics: Science and Systems (RSS)*, 2017.

[17] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2014.

[18] L. Bottou. Online learning and stochastic approximations. In *On-line Learning in Neural Networks*, volume 17, pages 9–42, 1998.

[19] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, 2011.

[20] 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. *The International Journal of Robotics Research*, 33(9):1251–1270, 2014.

[21] H. M. Choset. *Principles of Robot Motion: Theory, Algorithms, and Implementation*. MIT press, 2005.

[22] E. A. Wan and R. Van Der Merwe. The unscented Kalman filter for nonlinear estimation. In *Proc. Adaptive Systems for Signal Processing, Communications, and Control Symposium (AS-SPCC)*, pages 153–158, 2000.

[23] R. Kandepu, B. Foss, and L. Imsland. Applying the unscented kalman filter for nonlinear state estimation. *Journal of Process Control*, 18(7-8):753–768, 2008.

[24] J. Van Den Berg, P. Abbeel, and K. Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. *The International Journal of Robotics Research*, 30(7):895–913, 2011.

[25] D. Hadfield-Menell, S. Milli, P. Abbeel, S. J. Russell, and A. Dragan. Inverse reward design. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 6768–6777, 2017.

[26] J. Pan, Z. Chen, and P. Abbeel. Predicting initialization effectiveness for trajectory optimization. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages 5183–5190, 2014.