Online Inverse Reinforcement Learning with Learned Observation Model

Anonymous Author(s)
Affiliation
Address
email

Abstract: With the motivation of extending incremental/online inverse reinforcement learning (I2RL) to real world robotics applications with noisy observations as well as unknown observation model, we introduce a new method (RI2RL-MEOM) that approximates the observation model in order to best estimate the noise-free ground truth underlying the observations. It learns a maximum entropy distribution over the observation features governing the perception process, and then uses the inferred observation model to learn the reward function. Experimental evaluation is performed in two robotics tasks: (1) post-harvest vegetable sorting with Sawyer arm based on human demonstration, and (2) breaching a perimeter patrol with 2 Turtlebots. Our experiments reveal that RI2RL-MEOM learns a more accurate policy compared to (a) a state-of-the-art IRL method that does not directly learn an observation model, and (b) a custom baseline that learns a less sophisticated observation model. Furthermore, we show that RI2RL-MEOM admits formal guarantees of monotonic convergence and sample complexity bound.

Keywords: Observation model, Inverse reinforcement learning, Maximum entropy

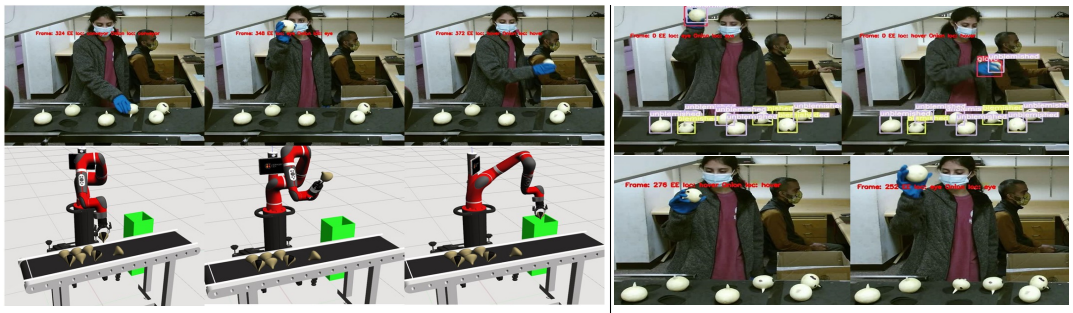
1 Introduction

Inverse reinforcement learning (IRL) methods learn the specification for a task that best explains the observed behavior of an expert executing that task [1, 2], in the form of a reward function for a Markov decision process. Most of the literature in IRL for robotic applications assumes that the learner's perception is noise-free, and that the learning occurs offline. However, real world sensing is often noisy and agents may need to learn incrementally because not all the training data is available in one batch. Motivated by the challenging goal of bringing robotic automation to post-harvest processing lines for vegetables, we consider examples of noisy perception by a robot learner observing a human sorting blemished onions out of a collection of onions. To learn this complicated manipulation task from its observations, the learner needs to unambiguously understand aspects of the task such as whether an onion is blemished or not, how the different locations of the expert's hand help realize the task (e.g., hand near table, hand near eyes of the sorter), etc.; see Fig. 1(a). As the robot's sensing is imperfect, it may mistake a blemished onion for an unblemished one or vice versa. When the expert places a blemished onion in a bin (for bad onions), the blemish may not be perceived correctly by the learner's camera. This introduces an error in the perceived state because the learner incorrectly sees the human placing an unblemished onion in a destination meant for bad onions (see Fig. 1 b/Top). Another example of imperfect perception is shown in Fig. 1 b/Bottom. Although the expert is holding an onion just above the table, the learner may incorrectly perceive it to be in the inspection (or eye) region. This ambiguity occurs because these two regions are in close proximity.

Shahryari and Doshi [3] address the problem of offline IRL under noisy observation using a method called *Robust IRL*, but they assume that the learner knows the observation (or sensor) model. This assumption may not hold in real world applications because the observation model is typically a function of obscure variables including inaccuracies in sensing hardware and computer vision. *Real world robotic applications need a method that can learn the unknown observation model under noisy input and use that model to learn a reward function.*

When a learner passively watches the expert perform the task, the observations arrive as a continuous stream of data over an extended period of time. However, methods such as Robust IRL require all

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.



(a) (b)

Figure 1: **(a/Top)** A human demonstrator picks an onion, inspects it closely to check if it is blemished, and places it in the bin on finding it to be blemished. **(a/Bottom)** The learner (Sawyer) robot uses the learned policy to imitate the observed behavior. **(b/Top)** A blemished onion detected as an unblemished onion. **(b/Bottom Left)**: an onion held in the hover region; **(b/Bottom Right)**: an onion held at the eye (or inspection) region.

observations to be available together as a batch. *Current IRL applications need a method that can learn online from noisy input without a'priori knowledge of the sensor model*. To meet the challenge, we develop a method that generalizes the batch Robust IRL method to the incremental setting, but now with an unknown observation model. We call this method Robust Incremental IRL with Maximum Entropy Estimation of Observation Model (RI2RL-MEOM), which is also an instantiation of the recent I2RL framework [4]. We formulate it as a sequence of incremental learning sessions where each session is a two-step process – maximum entropy learning of the observation model of the learner, followed by learning the reward function of the expert by utilizing the learned observation model. Fig. 2 gives a brief overview.

Using noisy training data of a human engaged in sorting onions, we evaluate the performance of RI2RL-MEOM on a Gazebo simulation of the onion sorting domain in which a Sawyer robot processes the observations incrementally and eventually imitates the task of sorting the onions (see Fig. 1(a)). We show that it generates a significantly improved sorting performance compared to a state-of-the-art end-to-end IRL baseline that accepts raw images as input. As this and other off-the-shelf IRL methods do not directly learn an observation model, we also evaluate the comparative advantage of RI2RL-MEOM using a custom baseline that learns a less sophisticated observation model. Similar results are observed in a second task of a Turtlebot learning to breach a patrol by another. Furthermore, we formally show that RI2RL-MEOM improves monotonically in data likelihood, and admits a probabilistic bound for sample complexity.

2 Related Work

Popular GAN based algorithms e.g., GAIL, belong to the class of imitation learning rather than IRL, therefore as our state-of-the-art baseline we choose GAN based Adversarial IRL [5] (or AIRL) that learns a reward function directly from raw observations. It trains a discriminator to classify expert data from the output of a generator (approximation of learned policy) and trains the generator to confuse the discriminator, the process resulting in a reward function being learned as the discriminator. Observation noise is not directly addressed by state-of-the-art literature, except in [3] which estimates the ground truth underlying the noisy observations, but does not address cases where the observation model is unknown, or the data is only available incrementally. The need for learning an observation model for IRL entails an approximation of a stochastic mapping that directly impacts the learning of the expert's reward function. To our knowledge, IRL practitioners have not contributed much in this direction. Among other relevant IRL methods are those that learn the stochastic transition dynamics, either by using importance sampling [6] or by decomposing each transition into its component features and learning from feature aggregates [7]. Adapting the former method for unknown observation models will need a preexisting near-perfect base observation model which appears unrealistic. We take inspiration from the latter method. Analogously to [7], we decompose an observation into its component observation features, and then the learner generalizes that information. Fig. 2 shows examples of such features that collectively model whether the expert's behavior is perceived correctly.

3 Background

An IRL agent aims to find a reward function under which the observed behavior of the expert is optimal, with the dynamics modeled as an incomplete MDP $\langle S, A, T \rangle$ [8, 2]. Abbeel and Ng [9]

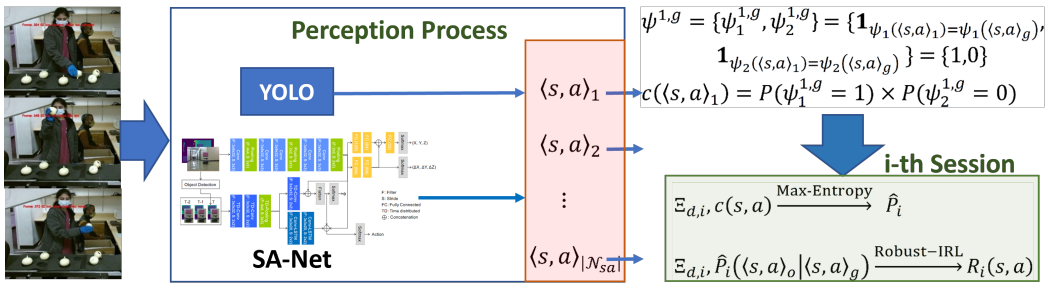


Figure 2: An overview of RI2RL-MEOM. On the left is an image-stream containing the latent ground truth, $\langle s,a\rangle_g$, from the expert's behavior. As the perception process corrupts the ground truth, the input to the learning algorithm gets noisy. This input is an array of corrupted observations and corresponding (classification) scores, $c(\langle s,a\rangle)$, that measure possible match with unknown $\langle s,a\rangle_g$. In this example, the perception process is decomposed into observation features, ψ_1: <did onion move with the hand of sorter?> and ψ_2: <is blemish present on the considered onion?>. See Secs. 4, 5 for details.

propose modeling the reward function as a linear combination of K binary features, ϕ_k, each of which maps a state and an action to a value in $\{0,1\}$. The linearised reward function is $R_{\boldsymbol{\theta}}(s,a) = \boldsymbol{\theta}^T\phi(s,a) = \sum_{k=1}^{K}\theta_k \cdot \phi_k(s,a)$, where θ_k are the *feature weights* in vector $\boldsymbol{\theta}$. The learner's task is to find a $\boldsymbol{\theta}$ that makes the observed behavior optimal in the completed MDP $\langle S, A, R_{\boldsymbol{\theta}}, T\rangle$.

Let Ξ be the space of all possible ground truth trajectories (i.e., sequence of (s,a) pairs) executable by the expert, and Ξ_d be the set of trajectories observed by the learner. The cumulative value of any feature ϕ_k is given by a function $f_k : \Xi \to \mathbb{R}$ defined as $f_k(\xi) = \sum_{\langle s,a\rangle_t \in \xi} \gamma^t \phi_k(\langle s,a\rangle_t)$. Ziebart et al. [10] solve IRL by finding the distribution over all trajectories that exhibits the maximum entropy and is constrained to match the observed feature expectations.

$$\max_{\Delta}\left(-\sum_{\xi\in\Xi} P(\xi)\log P(\xi)\right) \textbf{ subject to } \sum_{\xi\in\Xi} P(\xi) = 1, \textbf{ and } E_{\Xi}[\phi_k] = \hat{\phi}_k \forall k = 1\ldots K \quad (1)$$

Here, Δ is the space of all distributions over the set Ξ, and $E_{\Xi}[\phi_k] = \sum_{\xi\in\Xi} P(\xi)\, f_k(\xi)$. The learner empirically estimates the true feature expectation as $\hat{\phi}_k = \frac{1}{|\Xi_d|}\sum_{\xi\in\Xi_d} f_k(\xi)$.

3.1 Robust IRL with a Known Observation Model

Let $\iota_{sa} = \{\langle s,a\rangle_i \,|i \in \mathcal{N}_{sa} = \{1,2\ldots|S|\times|A|\}\}$ be an enumerated space for all possible (s,a) pairs. The observation model can be defined as a mapping that takes an observed state-action pair and a demonstrated (ground truth) state-action pair as input and gives the likelihood of observing the former pair given the latter [1]. That is, $O_b : \iota_{sa} \times \iota_{sa} \to [0,1]$ where

$$O_b(\langle s,a\rangle_o, \langle s,a\rangle_g) = P(\langle s,a\rangle_o \,|\, \langle s,a\rangle_g), g, o \in \mathcal{N}_{sa}, \quad (2)$$

the RHS being the probability of the learner observing $\langle s,a\rangle_o$ when the expert actually demonstrated $\langle s,a\rangle_g$. Extending this observation model to trajectories, for a ground truth trajectory $\xi \in \Xi$, the probability of observing trajectory $\xi' \in \Xi_d$ is

$$P(\xi'|\xi) = \prod_{\langle s,a\rangle_o \in \xi', \langle s,a\rangle_g \in \xi} P(\langle s,a\rangle_o \,|\, \langle s,a\rangle_g) \quad (3)$$

The batch learning method of Robust IRL [3] uses the aforesaid observation model to learn the reward function under noisy observation. In the process, it estimates the kth feature expectation as

$$\hat{\phi}_{\boldsymbol{\theta},k} = \frac{1}{|\Xi_d|}\sum_{\xi'\in\Xi_d}\sum_{\xi\in\Xi} P(\xi|\xi';\boldsymbol{\theta}) f_k(\xi) \quad (4)$$

where $P(\xi|\xi';\boldsymbol{\theta}) = \eta P(\xi'|\xi)P(\xi;\boldsymbol{\theta})$ and $P(\xi;\boldsymbol{\theta})$ is the probability of the generation of a trajectory ξ using a policy computed with $\boldsymbol{\theta}$. Then the program in (1) evolves to

$$\max_{\Delta}\left(-\sum_{\xi'\in\Xi_d,\xi\in\Xi} P(\xi',\xi)\log P(\xi',\xi)\right) \textbf{ subject to } \sum_{\xi'\in\Xi_d,\xi\in\Xi} P(\xi',\xi) = 1 \textbf{ and } E_{\Xi}[\phi_k] = \hat{\phi}_{\boldsymbol{\theta},k} \;\; \forall k \quad (5)$$

where $E_{\Xi}[\phi_k] = \sum_{\xi\in\Xi} P(\xi',\xi)\, f_k(\xi)$. We extend this optimization problem to the case of unknown observation model and online learning in Sections 4, 5 respectively.

[1] Although we use a state-action pair as an observation, the concept of robust-IRL is generalizable to any form of observation [3].

3.2 Framework on I2RL

Arora et al. [4] formulate the online/incremental IRL problem as a sequence of sessions with R_i as the ith session's estimate of the expert's reward function.

DEFN 3.1 (SESSION). *A session of I2RL, $\zeta_i(MDP_{/R}, \Xi_{d,i}, \alpha_{1:i-1}, R_{i-1})$, $i \in \mathbb{N}^+$, takes as input the expert's MDP (w/o R), the current (i^{th}) demonstration, $\Xi_{d,i} \in 2^{\Xi}$, the sufficient statistic summarizing previous $i-1$ sessions ($\alpha_{1:i-1}$), and the reward function estimated previously. It yields a revised estimate of the expert's reward function, R_i.*

DEFN 3.2 (I2RL). *Incremental IRL is a sequence of learning sessions $\{\zeta_1(MDP_{/R}, \Xi_{d,1}, \emptyset, R_0), \zeta_2(MDP_{/R}, \Xi_{d,2}, \alpha_{1:1}, R_1), \zeta_3 \ldots, \}$, which continue indefinitely or until a stopping criterion assessing convergence is met.*

As a measure of how well the learned reward function explains the observed data, we use $LL(R_i|\Xi_{d,1:i})$ as the *log likelihood of the accumulated demonstration received up to i^{th} session*.

4 Learning the Observation Model

The main idea is to approximate the unknown observation model using a feature set shared among observations, and then fit it to observations via max-entropy optimization. Let Ψ be this set of m observation features. Each feature, $\psi_j \in \Psi$ ($j \in \{1, 2, \ldots m\}$) is a predicate that takes as input a state-action pair and tells us whether a specific aspect in the expert's behavior is true or not for that input pair. Let $\psi^{o,g}$ be the set of $|\Psi| = m$ indicators, each computed by comparing the value of a feature on ground truth pair $\langle s, a\rangle_g$ and on observation $\langle s, a\rangle_o$

$$\psi_j^{o,g} = \mathbb{1}_{\psi_j(\langle s,a\rangle_o)=\psi_j(\langle s,a\rangle_g)}, \psi_j \in \Psi, j \in \{1, 2, \ldots, m\}. \tag{6}$$

Fig. 2 shows an example of such observation features ψ_j and the corresponding set $\psi^{o,g}$. We assume that the learner knows the set of features and the features are independent. For each $\langle s, a\rangle_o \in \Xi_d$, $\langle s, a\rangle_g \in \Xi$, we approximate the observation probability (Eq. 2) to be the product of (unknown) likelihoods of the values taken by the indicators $\psi_j^{o,g}$

$$O_b(\langle s, a\rangle_o, \langle s, a\rangle_g) \approx \prod_{\psi_j^{o,g} \in \psi^{o,g}} P(\psi_j^{o,g}) \tag{7}$$

Even though the learner knows the features, it cannot see the noise-free ground truth (s, a) pairs, and the (s, a) pairs not included in the training data. Computing a *complete* observation model requires a mechanism to infer a distribution over observation features *for all state-action pairs, observed or not*.

Learning the Maximum-Entropy Observation Model: We deploy YOLO [11] and SA-Net [12] to convert raw videos of human sorting onions to (noisy) confidence scores over ι_{sa}, as the function $c(\langle s, a\rangle)$ (Fig. 2). The likelihood for an observation $\langle s, a\rangle_o$ given *any* ground truth $\langle s, a\rangle_g$ is empirically estimated as a running average of the SA-Net confidence scores c from observed trajectories as

$$\hat{O}_o(\langle s, a\rangle_o, \langle s, a\rangle_g) = \frac{\sum_{\xi' \in \Xi_d} \sum_{\langle s,a\rangle \in \xi'} c(\langle s, a\rangle) \mathbb{1}_{\langle s,a\rangle_o = \langle s,a\rangle}}{\sum_{\xi' \in \Xi_d} \sum_{\langle s,a\rangle \in \xi'} \mathbb{1}_{\langle s,a\rangle_o = \langle s,a\rangle}}, \forall g \in \mathcal{N}_{sa}. \tag{8}$$

Utilizing the principle of maximum entropy (a form of least commitment), the learner infers a unique distribution over the observation features Ψ which has the maximum entropy, while satisfying the constraints that the learned likelihoods of the observations (Eq. 7) match the corresponding empirical estimates of true likelihoods (Eq. 8). Since the ground truth is not available to the learner, it considers all possible state action pairs $\langle s, a\rangle_{\overline{g}}$ that could have been in the input, where $\overline{g} \equiv$ *potential* ground truth from the expert. We simplify the notation here by using ψ and $\bar{\psi}$ for cases $\psi_j^{o,\overline{g}} = 1$ and $\psi_j^{o,\overline{g}} = 0$, and by using $\psi \in \psi^{o,\overline{g}}$ for $\psi_j^{o,\overline{g}} \in \psi^{o,\overline{g}}$.

$$\max_{\Delta_\psi} -\sum\nolimits_{\psi \in \psi^{o,\overline{g}}} \left(P(\psi) \log P(\psi) + P(\bar{\psi}) \log P(\bar{\psi})\right)$$

$$\textbf{subject to} \prod_{\psi \in \psi^{o,\overline{g}}} P(\psi) = \hat{O}_o \forall \langle s, a\rangle_o \in \Xi_d, \forall \langle s, a\rangle_{\overline{g}} \in \Xi \textbf{ and } P(\psi) + P(\bar{\psi}) = 1 \; \forall \psi \in \psi^{o,\overline{g}}$$

The Lagrangian relaxation of the above problem is:

$$\mathcal{L}(\Psi, \mathbf{v}, \boldsymbol{\lambda}) = -\left(\sum\nolimits_{\psi \in \psi^{o,\overline{g}}} P(\psi)\log P(\psi) + P(\bar{\psi})\log P(\bar{\psi})\right)$$
$$+ \sum_{\langle s,a\rangle_{\overline{g}} \in \Xi} \sum_{\langle s,a\rangle_{o} \in \Xi_d} v_{o,g}\left(\prod_{\psi \in \psi^{o,\overline{g}}} P(\psi) - \hat{O}_o\right) + \sum_{\psi \in \psi^{o,\overline{g}}} \lambda_\psi \left(\left(P(\psi) + P(\bar{\psi})\right) - 1\right) \quad (9)$$

where $\mathbf{v}, \boldsymbol{\lambda}$ are vectors of multipliers $v_{o,g}$ and λ_ψ.

Let $\hat{P}(\psi)$ denote the learned maximum-entropy distribution over features. As multiple distinct combinations of $\langle s,a\rangle_{\overline{g}}$ and observation $\langle s,a\rangle_o$ may map to the same values of $\psi_j^{o,g}$, we need not distinguish between these combinations, and may obtain the same probability value for all of these combinations due to the common feature mappings in $\hat{O}_o$ (similar to the problem solved in Bard [13]). The critical points of a Lagrangian occur at saddle points rather than a local maxima or minima [14], and most numerical optimization solvers are not designed to find the saddle points. Therefore, we modify the Lagrangian objective to make critical points occur at local extrema, $\mathcal{L}' = \sqrt{\left(\frac{\partial \mathcal{L}}{\partial P(\psi)}\right)^2 + \left(\frac{\partial \mathcal{L}}{\partial v_{o,g}}\right)^2 + \left(\frac{\partial \mathcal{L}}{\partial \lambda_\psi}\right)^2}$, and use limited memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) [15, 16] unconstrained optimization solver to learn $\hat{P}$. Using $\hat{P}$ and Eq. 7 and Eq. 2, we can derive the likelihoods of the observations. After approximating a complete observation model, the observation probability for trajectory ξ' can be computed using Eq. 3.

5 Robust Incremental IRL

To extend this process to an online setting, the learner uses $\hat{P}_i(\psi)$ learned at the beginning of session i to estimate the feature expectations

$$\hat{\phi}^{i}_{\boldsymbol{\theta}^i,k} \triangleq \frac{1}{|\Xi_{d,i}|} \sum_{\xi' \in \Xi_{d,i}} \sum_{\xi \in \Xi} \hat{P}_i(\xi|\xi';\boldsymbol{\theta}) f_k(\xi) = \frac{1}{|\Xi_{d,i}|} \sum_{\xi' \in \Xi_{d,i}} \sum_{\xi \in \Xi} \eta \hat{P}_i(\xi'|\xi) P(\xi;\boldsymbol{\theta}) f_k(\xi) \quad (10)$$

Let $\Xi_{d,1:i}$ be the cumulative demonstration up to the ith session, and $\hat{\phi}^{1:i}_{\boldsymbol{\theta}^i,k}$ be the corresponding cumulative feature expectations. The learner updates the latter from session to session as

$$\hat{\phi}^{1:i}_{\boldsymbol{\theta}^i,k} \triangleq \frac{1}{|\Xi_{d,1:i}|} \sum_{\xi' \in \Xi_{d,1:i}} \sum_{\xi \in \Xi} \hat{P}_{1:i}(\xi|\xi';\boldsymbol{\theta}) f_k(\xi) = \frac{1}{|\Xi_{d,1:i}|}\Bigg(\sum_{\xi' \in \Xi_{d,1:i-1}} \sum_{\xi \in \Xi} \hat{P}_{1:i-1}(\xi|\xi';\boldsymbol{\theta}) f_k(\xi) +$$
$$\sum_{\xi' \in \Xi_{d,i}} \sum_{\xi \in \Xi} \hat{P}_i(\xi|\xi';\boldsymbol{\theta}) f_k(\xi)\Bigg) = \frac{1}{|\Xi_{d,1:i-1}| + |\Xi_{d,i}|}\left(|\Xi_{d,1:i-1}|\, \hat{\phi}^{1:i-1}_{\boldsymbol{\theta}^{i-1},k} + |\Xi_{d,i}|\, \hat{\phi}^{i}_{\boldsymbol{\theta}^i,k}\right). \quad (11)$$

5.1 The RI2RL-MEOM Method

Following Def. 3.1, a session of RI2RL-MEOM is $\zeta_i(MDP_{/R}, \Xi_{d,i}, \alpha_{1:i-1}, \boldsymbol{\theta}^{i-1})$ where sufficient statistic $\alpha_{1:i-1} = (|\Xi_{d,1:i-1}|, \hat{P}_{1:i-1}(\psi), \hat{\phi}^{1:i-1}_{\boldsymbol{\theta}^{i-1}})$. For demonstration $\Xi_{d,i}$ of the current session i, the scores $c(\langle s,a\rangle)$ for input (s,a) pairs are available from the perception mechanism. The process starts by computing $\hat{O}_o$ using the scores in Eq. 8. The resulting $\hat{O}_o$ is plugged into Eq. 9 to get $\nabla\mathcal{L}'$, using which $\hat{P}_{1:i}(\psi)$ is updated by L-BFGS gradient ascent starting with the initial value of $\hat{P}_{1:i-1}(\psi)$. The learned $\hat{P}_{1:i}$ are then used to compute the updated observation model probabilities $P(\langle s,a\rangle_o \,|\, \langle s,a\rangle_g)$ (Eqs. 7, 2). This extends to a corresponding update for full trajectories, $\hat{P}_{1:i}(\xi'|\xi,\boldsymbol{\theta})$ (Eq. 3). Solving Eq. 10, the learner uses MCMC to sample ground truth trajectories from $\hat{P}_{1:i}(\xi|\xi',\boldsymbol{\theta})$ for computing the feature expectations $\hat{\phi}^{i}_{\boldsymbol{\theta}^i}$, followed by computation of $\hat{\phi}^{1:i}_{\boldsymbol{\theta}^i}$ via incremental update (Eq. 11). Finally, the latter is used for the main IRL optimization (Eq. 5) via L-BFGS gradient descent to learn the reward weights ($\boldsymbol{\theta}^i$) and corresponding policy π_i. Due to space constraint, we show the detailed algorithm in the Appendix.

5.2 Convergence Properties of RI2RL-MEOM

While most of the results in this section are similar to corresponding results in [4], Lemma 2 is the main original result of this paper since it incorporates the additional uncertainty due to noisy

observations and unknown observation model. Due to space constraint, we present all proofs in the Appendix. The demonstration log-likelihood for session i is given by $LL(\boldsymbol{\theta}^i|\Xi_{d,i}, \alpha_{1:i-1}, \boldsymbol{\theta}^{i-1})$.

LEMMA 1 (MONOTONICITY). *The demonstration likelihood increases monotonically with each new session,* $LL(\boldsymbol{\theta}^i|\Xi_{d,i}, \alpha_{1:i-1}, \boldsymbol{\theta}^{i-1}) - LL(\boldsymbol{\theta}^{i-1}|\Xi_{d,i-1}, \alpha_{1:i-2}, \boldsymbol{\theta}^{i-2}) \geqslant 0$*, when* $|\Xi_{d,1:i-1}| \gg |\Xi_{d,i}|$.

Notice that feature expectations estimated under noisy perception with estimated observation model, $\hat{\phi}^{1:i}_{\boldsymbol{\theta}^i,k}$ from Eq. 11, is an approximation of the expectations for the observation model being accurately known (Eqs. 10, 11 with true P, instead of $\hat{P}$). $\hat{\phi}^{1:i}_{\boldsymbol{\theta}^i,k}$ is computed by sampling the hidden ground truth ξ from $P(\xi|\xi', \boldsymbol{\theta}^{i-1})$. This, in turn, is an approximation of the expectations estimated under noise-free perception, $\hat{\phi}^{1:i}_k$. As in [4], we assume that the latter estimation using n_s samples obeys Hoeffding bounds, i.e., $P(|\hat{\phi}^{1:i}_k - \hat{\phi}^{1:i}_{\boldsymbol{\theta}^i,k}| \leq \varepsilon_s) \geq 1 - \delta_s$, where $\delta_s = 2K\exp(-2n_s\varepsilon_s^2)$. Additionally for this paper, we assume that our observation model estimation with n_o samples also obeys Hoeffding bounds as $P(\max_{\langle s,a\rangle_o} |O^* - \hat{O}| \leq \varepsilon_o) \geq 1 - \delta_o/K$, where $O^*_{o,g}$ is the true observation model (of which $\hat{O}_o$ is an estimate), and $\delta_o = 2K|\Psi|\exp(-2n_o\varepsilon_o^2)$. Note that the $\max$ operator in this assumption is only for *observed* $\langle s, a\rangle_o$, not all possible $\langle s, a\rangle_o$, but we assume that all features in Ψ are observed in the very first session. Then the following lemma holds.

LEMMA 2 (CONSTRAINT BOUND). *Under the assumptions stated above, the following holds with probability at least* $\max(0, 1 - \delta_r)$*:* $\left|(1-\gamma)\big(E_\Xi[\phi_k] - \hat{\phi}^{1:i}_{\boldsymbol{\theta}^i,k}\big)\right|_1 \leqslant \varepsilon_r, \forall k \in \{1, 2 \ldots K\}$*, where* $\delta_r = \delta + \delta_s + \delta_o$*,* $\varepsilon_r = \varepsilon + \varepsilon_s + L|\Psi|\varepsilon_o$*, L is the maximum length of any trajectory, and* ε, δ *are as defined in Theorem 1 in [4].*

Lemma 2 allows us to probabilistically bound the error in the log likelihood for RI2RL-MEOM, leading to the main result below specifying the confidence in its convergence.

THEOREM 1 (CONFIDENCE). *Let* ε_r, δ_r *be as defined in Lemma 2, and* $\boldsymbol{\theta}^i$ *be the solution of session* i *for RI2RL-MEOM. Then* $LL(\boldsymbol{\theta}_E|\Xi_{d,1:i}) - LL(\boldsymbol{\theta}^i|\Xi_{d,i}, \alpha_{1:i-1}, \boldsymbol{\theta}^{i-1}) \leq \frac{2K\varepsilon_r}{(1-\gamma)}$*, with confidence at least* $\max(0, 1 - \delta_r)$*, where* $\boldsymbol{\theta}_E$ *are the true weights of the expert.*

6 Experiments

Our primary domain for the evaluation of RI2RL-MEOM is the onion sorting domain (Fig 1). The expert aims to identify blemished onions from a collection of onions on a table, and drop blemished onions in a bin while allowing others to pass. A state of the task is composed of the following features: *onion location* and *end-effector location* [on table, picked up in region of hover, at eye level (under inspection), or near the bin for blemished-onion], and *quality prediction* [blemished, unblemished, or unknown]. The domain has 100 states. The actions involve focusing attention on a new onion on the table, picking it up, bringing the grasped onion closer to eye and inspecting [2] it, placing it in the bin, and placing it back on the table. We utilize the following 9 predicates as the binary features $\phi(s, a)$.

- *ClaimNewOnion*(s,a): action a in state s considers a new onion on table;
- *PickUnknown*(s,a): action a picks an onion with unknown prediction;
- *AvoidNoOp*(s,a): the action a changes the state;
- *InspectNewOnion*(s,a): onion is inspected for the first time and a prediction is made for it;
- *GoodOnTable*(s,a): considered onion is predicted to be unblemished and is placed on the table;
- *BlemishedNotOnTable*(s,a): onion is predicted to be blemished and is not placed on the table;
- *GoodNotInBin*(s,a): onion is predicted to be unblemished and is not placed in the bin;
- *BlemishedInBin*(s,a): onion is predicted to be blemished and is placed in the bin;
- *PickBlemished*(s,a): onion with prediction blemished is picked;

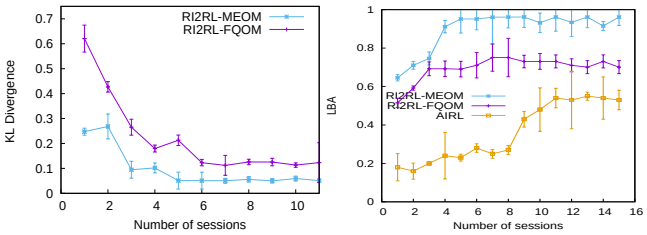
We utilize the following 8 binary predicates [3] ψ_j as observation features for learning the observation model. Each indicator $\psi^{o,g}_j$ takes the value 1 iff the predicate is T/F for both $\langle s, a\rangle_g$ and $\langle s, a\rangle_o$.

[2] During simulation, a distribution assigning a reasonable mass to the correct quality prediction is used.

[3] We used Pearson product-moment correlation (0.03 with p-value 0.04, averaged over all pairs of features from demonstration) to confirm that these observation features are sufficiently independent.

- *BlemishedOnion* considered onion is blemished;
- *MoveWithHand* onion moves with the hand;
- *StartFromConv* onion was on the table before action;
- *LeavingAtEye* onion leaves atEye location;
- *OnionToBin* onion moves to the bin;
- *HandToBin* hand moves to the bin;
- *OnionToTable* onion moves to the table;
- *HandToTable* hand moves to the table;

Baselines: We use AIRL [5] as a state-of-the-art baseline to compare with RI2RL-MEOM. AIRL is trivially incremental because the learned weights of the discriminator and generator networks are a sufficient statistic. To our knowledge, no off-the-shelf IRL method infers an observation model. Therefore, we create a custom baseline using the frequentist approach, Robust I2RL with Frequentist Estimation of Observation Model or RI2RL-FQOM. RI2RL-MEOM solves an optimization problem (Sec. 4) to learn $\hat{P}_i$, but RI2RL-FQOM assumes that the ground truth is the same as the observation and learns $\hat{P}_i$ as the running average of normalized frequencies of observation features activated in the demonstration.



	TP,FP,FN,TN	P%,R%
Expert	7,0,12,5	100.00, 58.33
RI2RL-MEOM	8,1,11,4	88.89, 66.67
RI2RL-FQOM	8,3,9,4	72.73, 66.67
AIRL	5,4,8,7	55.56, 41.67

Figure 3: (Left) Divergence between $P(\psi)$ and $\hat{P}(\psi)$ for **simulated dataset**. (Middle) LBA of the learned policy using **real world dataset**. (Right) Column labels TP $\equiv$ true positive (# blemished onions in bin), FP $\equiv$ false positive (# good onions in bin), TN $\equiv$ true negatives (# good onions remaining on table), and FN $\equiv$ false negatives (# blemished onions remaining on table). P and R denote precision and recall in %, respectively.

Implementation Setup: We use videos of human sorting onions (Fig. 1) as training data (demonstration). The perception pipeline considers only the state with highest prediction score ($c(\langle s, a\rangle)$ value) as the current state in a trajectory $\xi' \in \Xi_d$. We provide the sequence of perceived noisy trajectory and the scores as input to I2RL sessions of all methods. We also simulate the domain in the 3D Gazebo 7, with the robotic arm Sawyer functioning as the learner.

Performance Results: As we do not know the distribution $P(\psi)$ corresponding to the true observation model, we compare the accuracy of learned $\hat{P}(\psi)$ with a *simulated* dataset from a manually designed true distribution $P(\psi)$. We introduce noise as $P(\psi_j(\langle s, a\rangle_o) \neq \psi_j(\langle s, a\rangle_g) = 0.3$ for $j =$ *BlemishedOnion, OnionToBin, OnionToTable*, and 0 for other features. Fig. 3(left) shows how the KL divergence between the distributions $P(\psi)$ and $\hat{P}_i(\psi)$ changes with session i. Each data point is averaged over 50 runs, and each session has 2 trajectories of size 3 as input. The divergence for RI2RL-MEOM is significantly lower than that for RI2RL-FQOM throughout the sessions, indicating more accurate $\hat{P}_i$ for the former. After the first few sessions, there is little improvement in KLD for either method. As AIRL does not directly learn an observation model, we cannot measure KLD for it.

We also measure *learned behavior accuracy* (LBA=proportion of states in which the action prescribed by the learned policy match the actual action of the expert). Fig. 3(middle) shows the average LBA for *real world dataset* as the number of sessions increase, for all three methods. LBA for AIRL improves initially until session 11, but flattens out considerably lower than other two methods (maximum 48%). On average, AIRL is also slower ($\sim$ 8 minutes) to converge as compared to RI2RL-MEOM (96.69 seconds). We attribute these results to two reasons: a) data hungry nature of GAN based methods, b) unlike RI2RL, AIRL is not specifically designed to accommodate noisy observations. Furthermore, RI2RL-FQOM's LBA flattens out around 23% lower than RI2RL-MEOM. We believe the reason is that the maximum-entropy optimization in RI2RL-MEOM generalizes the learned information across (s, a) pairs not present in the demonstration, but the frequentist baseline fails to generalize. Eventually, RI2RL-MEOM reaches LBA of 96% whereas the latter stays around 70%. As RI2RL-MEOM solves an additional optimization problem for learning $\hat{P}$, it takes longer to converge (96.69 seconds, on average) compared to RI2RL-FQOM (85.99 seconds).

To evaluate the sorting performance, we spawn a collection of onions for a fixed time period, and the learner (Sawyer) uses the learned policy for sorting as many onions as possible before they disappear. For object tracking and onion classification, we use YOLO with a simulated camera in Gazebo. For

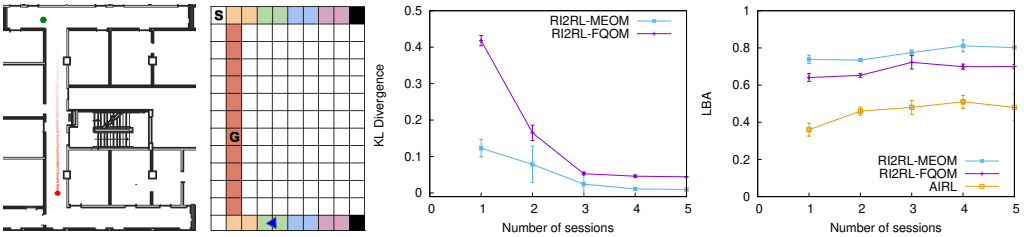


Figure 4: (Left) Patroller (red) navigates hallways [17], and learner (green) wants to pass undetected. (Left-middle) The color shaded regions (long hallway, turning points and three small divisions in small hallways on both sides) are the five regions defining movement and turn-around features. S and G are start and goal locations for the learner. (Right-middle) Divergence between $P(\psi)$ and $\hat{P}(\psi)$, and (Right) LBA of the learned policies.

each RI2RL method, we execute the policy learned in the final session, for six moving collections with four onions each - two blemished and two good. Fig. 3(right) shows the average precision and recall of the expert and three learning methods. Due to time-consuming inspection, the expert has a high precision but low recall. The precision for RI2RL-MEOM is closer to the expert's than the two baselines, due to a lower FP. AIRL performs worse in both precision and recall having higher FP and TN. This is consistent with its low LBA, indicating its reduced ability to replicate the expert behavior in a noisy setting. Among two RI2RL methods, as the steps after learning the observation model are exactly the same, we attribute the better precision of RI2RL-MEOM to more accurate $\hat{P}$.

To establish the generality of RI2RL-MEOM, we evaluate it in a second domain–a simplified version of perimeter patrol domain from [17]. It involves two robots (Turtlebots in Gazebo), one patroller continuously navigates some hallway in cyclic trajectories; the other is a learner tasked with reaching a goal location without being spotted by the patroller (Fig. 4). The learner must solve its own distinct decision-making problem (modeled as another MDP), which is dependent on correctly predicting the patrol. The latter can be estimated from inferring patroller's preferences given that the learner knows their dynamics. The state of patroller in the MDP has three dimensions $\langle x, y, \theta \rangle$, which gives the x and y coordinates of the cell decomposition of the corridors and hallways, and θ is the orientation of the patroller. The domain has 124 states. The patroller can execute one of four possible actions: move-forward, turn left or turn right 90 degrees, and stop. The reward function utilizes the following 6 binary state-action feature functions, $\phi(s, a)$, which divide the grid broadly into five regions as shown in Fig. 4 (left-middle): *HasMoved*$(s, a) = 1$ iff a in s makes the patroller change its grid cell; *Turn1–Turn5*$(s, a) = 1$ iff a in s makes the patroller turn (left or right) in the hallway regions orange, yellow, green, blue and magenta, respectively. A weight vector $\boldsymbol{\theta}_E$ for these features such as $\langle .57, 0, 0, 0, .43, 0 \rangle$ makes the patroller constantly execute cyclic trajectory.

The observation feature set Ψ for this domain contain the following 4 binary predicates:

- *MoveForward* patroller is moving forward;
- *y is 0* patroller location has $y = 0$;
- *TurnLeft* patroller is turning left;
- *TurnRight* patroller is turning right;

We introduce noise as $P(\psi_j(\langle s, a \rangle_o) \neq \psi_j(\langle s, a \rangle_g) = 0.3$ for $j =$ *MoveForward, TurnRight*, and 0 for other features. Average of pairwise feature correlation from the patroller's demonstration is -0.14 (p-value 0.06), indicating that the features are reasonably independent. The two figures (right) in Fig. 4 show the performance of both RI2RL methods in perimeter patrol domain. As in onion-sorting, KLD and LBA for MEOM is better than FQOM and AIRL. These results in 2 unrelated tasks indicate that the proposed method is sufficiently general for broader applicability.

7 Conclusion

We have introduced an online IRL method that accommodates noisy observations and infers the unknown observation model by maximum entropy optimization. Not only does this method fill a gap in the prior work on online IRL but it also makes online learning more pragmatic, as evaluated in two robotics tasks of post-harvest onion sorting and breaching a patrol.

Limitations: The current set up restricts the observation features to binary functions. The assumption of feature independence, and the key approximation of Eq. 7, may not be strictly valid, and may have contributed to RI2RL-MEOM's failure to learn an *accurate* observation model (KLD=0). To meet the assumption that all ψ are observed in the first session, it may need to contain more demonstrations.

References

[1] S. Arora and P. Doshi. A Survey of Inverse Reinforcement Learning: Challenges, Methods and Progress. *Artificial Intelligence*, 297:103500, 2021. doi:https://doi.org/10.1016/j.artint.2021.103500.

[2] A. Ng and S. Russell. Algorithms for inverse reinforcement learning. In *Seventeenth International Conference on Machine Learning*, pages 663–670, 2000.

[3] S. Shahryari and P. Doshi. Inverse Reinforcement Learning Under Noisy Observations. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '17, Richland, SC, 2017. International Foundation for Autonomous Agents and Multiagent Systems.

[4] S. Arora, P. Doshi, and B. Banerjee. I2RL: Online inverse reinforcement learning under occlusion. *Autonomous Agents and Multi-Agent Systems*, 35(1):4, Nov 2020. ISSN 1573-7454.

[5] J. Fu, K. Luo, and S. Levine. Learning Robust Rewards with Adverserial Inverse Reinforcement Learning. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, 2018. URL https://openreview.net/forum?id=rkHywl-A-.

[6] A. Boularias, J. Kober, and J. Peters. Relative entropy inverse reinforcement learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, pages 182–189, 2011.

[7] K. Bogert and P. Doshi. Toward Estimating Others' Transition Models Under Occlusion for Multi-robot IRL. In *24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1867–1873, 2015.

[8] S. Russell. Learning agents for uncertain environments (extended abstract). In *Eleventh Annual Conference on Computational Learning Theory*, pages 101–103, 1998.

[9] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Twenty-first International Conference on Machine Learning (ICML)*, pages 1–8, 2004.

[10] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *23rd National Conference on Artificial Intelligence - Volume 3*, pages 1433–1438, 2008.

[11] G. Jocher, A. Stoken, J. Borovec, C. Stan, L. Changyu, A. Hogan, L. Diaconu, Marc, Doug, F. Ingham, Frederik, Guilhen, Hatovix, J. Poznanski, J. Fang, L. Yu, M. Wang, N. Gupta, O. Akhtar, P. Dvoracek, and P. Rai. ultralytics/yolov5: v3.1, Oct. 2020.

[12] N. Soans, E. Asali, Y. Hong, and P. Doshi. SA-Net: Robust State-Action Recognition for Learning from Observations. In *2020 IEEE International Conference on Robotics and Automation, ICRA*, pages 2153–2159, 2020.

[13] Y. Bard. Estimation of state probabilities using the maximum entropy principle. *IBM Journal of Research and Development*, 24(5):563–569, 1980.

[14] G. R. Walsh. *Methods of optimization*. Wiley, London, 1975. ISBN 0471919225.

[15] D. C. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

[16] J. Nocedal and N. Okazaki. libLBFGS: A library of Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) . https://github.com/chokkan/liblbfgs, 2013.

[17] K. Bogert and P. Doshi. Multi-robot Inverse Reinforcement Learning Under Occlusion with Interactions. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS '14, pages 173–180, 2014.