Autonomous Agents and Multi-Agent Systems (2021) 35:4
https://doi.org/10.1007/s10458-020-09485-4

I2RL: online inverse reinforcement learning under occlusion

Saurabh Arora[1] · Prashant Doshi[1] · Bikramjit Banerjee[2]

Accepted: 22 October 2020 / Published online: 5 November 2020

Abstract
Inverse reinforcement learning (IRL) is the problem of learning the preferences of an agent from observing its behavior on a task. It inverts RL which focuses on learning an agent's behavior on a task based on the reward signals received. IRL is witnessing sustained attention due to promising applications in robotics, computer games, and finance, as well as in other sectors. Methods for IRL have, for the most part, focused on batch settings where the observed agent's behavioral data has already been collected. However, the related problem of online IRL—where observations are incrementally accrued, yet the real-time demands of the application often prohibit a full rerun of an IRL method—has received significantly less attention. We introduce the first formal framework for online IRL, called incremental IRL (I2RL), which can serve as a common ground for online IRL methods. We demonstrate the usefulness of this framework by casting existing online IRL techniques into this framework. Importantly, we present a new method that advances maximum entropy IRL with hidden variables to the online setting. Our analysis shows that the new method has monotonically improving performance with more demonstration data as well as probabilistically bounded error, both under full and partial observability. Simulated and physical robot experiments in a multi-robot patrolling application situated in varied-sized worlds, which involves learning under high levels of occlusion, show a significantly improved performance of I2RL as compared to both batch IRL and an online imitation learning method.

Keywords Robot learning · Online learning · Robotics · Reinforcement learning · Inverse reinforcement learning

✉ Prashant Doshi
pdoshi@uga.edu

Saurabh Arora
sa08751@uga.edu

Bikramjit Banerjee
Bikramjit.Banerjee@usm.edu

[1] THINC Lab, Department of Computer Science, 415 Boyd GSRC, University of Georgia, Athens, GA 30602, Georgia

[2] School of Computing Sciences and Computer Engineering, University of Southern Mississippi, 118 College Drive, Hattiesburg, MS 39406, USA

1 Introduction

Inverse reinforcement learning (IRL) [26, 32] refers to both the problem and method of observing an agent as it performs a task to learn the agent's preferences or its reward function guiding its actions on a task. While reinforcement learning aims to learn a behavior that optimizes the rewards received, methods for IRL infer a reward function that explains an input behavior; thus, IRL inverts RL. The inverse learning approach is appealing to many researchers and practitioners especially because it broadens the scope of machine learning to those applications in which a reward function can not be easily specified and in which an understanding of the observed task is needed. IRL is useful in controlled environments where learning takes place via observing others. For example IRL shows promise toward applications in robot learning through demonstrations by human teachers [4], penetrating multirobot patrols [11], imitation learning [27], and ad hoc collaboration in sorting tasks [34].

Most IRL methods operate on large batches of observations, and, in a single pass, they estimate the expert's reward function [1, 8, 14, 15, 28]. These methods may be satisfactory for the applications mentioned above where the performed task is first observed for some time and the gathered data is utilized toward IRL. However, more recent applications of IRL impose a different set of requirements. These require the learner to continuously observe and repeatedly update its estimate of the learned reward function of the observed agent in order to facilitate the learner's ongoing activities. Consider, for example, the task of forecasting a person's goals in an everyday setting from observing her ongoing activities using a body camera – this enables the use of assistive robots in households [31], or a robotic learner that is continuously monitoring activities from a given location and learning the patrol pattern for reaching a specific goal as quickly as possible without being detected [9]. Both of these applications include streaming observations and could be improved with interleaving, observing, and learning the preferences of the expert. As such, there is a need to generalize IRL toward online learning with data provided in mini-batches.

In this article, we first present a formal framework to facilitate new methods for online IRL and to conceptually compare them. The framework, labeled as incremental IRL (I2RL), establishes the key components of this problem and rigorously defines the notion of an incremental variant of IRL. Furthermore, I2RL offers candidate stopping criteria that a potential online IRL method could utilize as well as a regret-based metric for measuring the performance quality of the method. To demonstrate its usefulness, we model three existing methods that perform online IRL [20, 21, 31] using the components of this framework. This provides initial evidence that the framework is sufficient for modeling online approaches.

Our primary contribution is a new method that advances the recent progress in maximum entropy IRL from a partially hidden demonstration of data [12] by generalizing it to an online setting in the context of the I2RL framework. We establish key theoretical properties of this new method, which we call *online latent maximum entropy* IRL (LME-I2RL). Specifically, we show that the demonstration data likelihood increases monotonically for this method as more of the demonstration is seen. Consequently, the method shows probabilistic convergence within a desired error of the feature expectations of the learned policy (from those of the expert's true policy) in both fully observed and partially occluded contexts. Finally, we prove that with a high confidence the method exhibits no-regret learning asymptotically as the number of sessions increases. In other words, the average loss across multiple sessions approaches zero with an increasing number of sessions, which is a desirable property for online learning.

We conducted experiments on a previously introduced robotic application of IRL [9] to comprehensively evaluate the performance of LME-I2RL. The domain involves the use of IRL toward learning the reward functions of one or two independent mobile robots continuously patrolling corridors of varying configurations. A third robot observing the patrollers from a vantage point that has a limited view is tasked with penetrating the patrol to reach a goal location undetected. It uses the learned reward functions and known motion models to predict the patrolling trajectories and to identify, if possible, a path to the goal location without being spotted.

The performance of LME-I2RL is compared with the previous batch LME to evaluate the advantage of online IRL in the above mentioned domain in both simulations and physical experiments. Our evaluation shows that the incremental method learns the patrolling behaviors equally well as compared to the batch method but faster, which leads to a higher success rate. It suffers from far fewer timeouts as compared to the batch method, where timeouts result due to failures in completing the inverse learning and forward planning within a reasonable time limit. Furthermore, we develop an online variant of a well-known imitation learning method based on generative adversarial networks [20]. LME-I2RL learns behavior that is significantly more accurate as compared to this method for imitation learning.

In our simulation experiments, we vary the degree of occlusion, the number of states in the domain, as well as the number of patrollers. Our experiments with physical TurtleBots in two environments confirm the performance obtained in simulation with the learner capable of observing just about 30% and as less as 18% of the patrollers' trajectories, respectively. As such, the framework and the method presented in this article may be viewed as essential first steps toward the emerging problem of online learning of reward function.

The remainder of this article is structured as follows. In the following section, we briefly review the concepts that are key to understanding IRL and the known method of maximum entropy IRL. We also discuss its generalization to IRL with hidden variables. In Sect. 3, we lay out the framework of incremental IRL (I2RL) by defining the components of online IRL followed by a discussion of existing online approaches in the context of I2RL. Next, we present the details of the LME-I2RL method and the theoretical convergence properties of this method. In Sect. 5, we assess the performance advantages of I2RL on both simulated and physical robotic domains, and analyze the convergence properties and experiments. After discussing related work in the prevalent literature, we conclude this article in Sect. 7 and introduce some options for future research in this area. The Appendix gives the full proofs of the lemmas and theorems stated in Sect. 4.

Some portions of this article have been published previously in a conference paper [6]. This article significantly expands on the conference paper in the following ways: (1) We formally introduce regret in the framework of incremental IRL in Sect. 3, (2) we add complete proofs of all convergence results in the Appendix, including the new result of no-regret learning; and (3) in Sect. 5.4, we offer evidence toward the scalability of our method of online learning under occlusion using a new physical robot domain that features a larger state space and significantly greater occlusion (greater than 18%) for the learner. (4) Finally, we have altogether significantly expanded the exposition in nearly all sections of the article.

2 Background on IRL under occlusion

Informally, IRL connotes both the problem and method by which an agent learns goals and preferences of another agent that explain the latter's observed behavior [5, 32]. IRL is appealing because the learned preferences are amenable for transfer from the observed agent to the subject robot (with minor adjustments) – as Russell [32] strongly argues – because both robot and agent will operate in the same environment. The transferred preferences may be utilized to perform the same task or, as in our domain, they are integrated for use in a decision making and planning framework. We briefly review the key concepts of IRL in the next subsection, followed by a leading technique for IRL in Sect. 2.2. Finally, we review this technique's generalization toward occlusion in Sect. 2.3.

2.1 Inverse RL

The observed agent is often considered an "expert" in the performed task. To model the observed agent denoted E, it is assumed that E is executing an optimal policy based on a standard MDP defined as $\langle S_E, A_E, T_E, R_E \rangle$. The learning agent L is assumed to exhibit perfect knowledge of the MDP parameters except of the reward function. Therefore, the learner's task is to infer a reward function that best explains the observed behavior of the expert under these assumptions. A *policy* is a function mapping each state to an action. It can be deterministic, $\pi : S \rightarrow A$ or stochastic $\pi : S \rightarrow \mathsf{Prob}(A)$. For a policy π, value function $V^\pi : S \rightarrow \mathbb{R}$ gives the value of a state s as the long-term expected cumulative reward obtained from the state by following π. The value of a policy π from some initial state s_0 is,

$$V^\pi(s_0) = E\left[\sum_{t=0}^{\infty} \gamma^t R_E(s_t, \pi(s_t)) | \pi, s_0\right]. \tag{1}$$

The problem of IRL is generally ill-posed because for any given behavior there are infinitely-many reward functions which may explain the behavior. The under constrained nature of this problem is exacerbated as less data is observed due to occlusion. Ng and Russell [26] initially approached the problem with linear programming inferring a reward function that maximizes the difference between the value of the expert's optimal policy and the next best policy under the assumption that the expert's complete policy is available. Abbeel and Ng [1] relaxed this assumption using an algorithm in which the expert, E, provides a *demonstration* of the task performance instead of its policy. (Demonstrations may be seen as composed of simulations of the expert's optimal policy.) To model the reward function, a linear combination of K binary features is used, ϕ_k: $S_E \times A_E \rightarrow [0, 1]$, $k \in \{1, 2 \ldots K\}$. Each feature maps a state from the set of states, S_E, and an action from the expert's set of actions, A_E, to a value in $\{0,1\}$. Note that non-binary features can always be converted into binary features although there will be more of them. Choosing appropriate feature functions is important, and, if these features are not known to the learner, they can be learned from the data thereby diminishing the need for feature engineering [25].

The reward function for the expert, E, is then defined as $R_E(s,a) = \boldsymbol{\theta}^T \phi(s,a) = \sum_{k=1}^{K} \theta_k \cdot \phi_k(s,a)$, where θ_k are the *weights* in vector $\boldsymbol{\theta}$; let $\mathscr{R} = \mathbb{R}^{|S_E \times A_E|}$ be the continuous space of reward functions. The learner task is simplified to one of completing the reward function by finding an appropriate vector of weights so that the demonstrated behavior is optimal. Let $\mathbb{N}^+$ be a bounded set of natural numbers. It is pertinent to formally define a demonstration here.

Definition 1 *(Set of fixed-length trajectories)* The set of all trajectories of finite length T from an MDP attributed to the expert E is defined as, $\mathbb{X}^T = \{X | X = (\langle s,a\rangle_1, \langle s,a\rangle_2, \ldots, \langle s,a\rangle_T), T \in \mathbb{N}^+\}, \forall s \in S_E, \forall a \in A_E\}$.

Then, the set of *all* trajectories is $\mathbb{X} = \mathbb{X}^1 \cup \mathbb{X}^2 \cup \ldots \cup \mathbb{X}^{|\mathbb{N}^+|}$. A demonstration is some finite set of trajectories of varying lengths, $\mathscr{X} = \{X^T | X^T \in \mathbb{X}^T, T \in \mathbb{N}^+\}$, and it includes the empty set.[1] Subsequently, we may define the set of demonstrations.

Definition 2 *(Set of demonstrations)* The set of demonstrations is the set of all subsets of the set of trajectories of varying lengths. Therefore, it is the power set, $2^{\mathbb{X}} = 2^{\mathbb{X}^1 \cup \mathbb{X}^2 \cup \ldots \cup \mathbb{X}^{|\mathbb{N}^+|}}$.

Per the definitions above, IRL ascribes an MDP without the reward function to the expert, and thus, consists of estimating the reward function, $\hat{R}_E \in \mathscr{R}$, to provide the best explanation of the learner's observations $\mathscr{X} \in 2^{\mathbb{X}}$. Therefore, we can view IRL as a function: $\zeta(MDP_{/R_E}, \mathscr{X}) = \hat{R}_E$. This concise formulation of IRL will be utilized later in the article.

Abbeel and Ng [1] define feature expectations as the discounted sum of feature values computed over the trajectories in a demonstration,

$$\phi_k^{\pi_E} = E_{\mathscr{X}}[\phi_k] = E\left[\sum_{t=0}^{T} \gamma^t \phi_k(s_t, a_t) | s \in S_E, a = \pi_E(s) \in A_E\right] \tag{2}$$

For a linear reward function with a fixed set of weights, feature expectations provide a way to obtain the expected value of a policy: $V^\pi(s_0) = \sum_{i=1}^{k} \theta_i \cdot E_{\mathbb{X}}[\phi]$. Therefore, matching the feature expectations of the learned behavior with those for demonstrated behavior is equivalent to matching the expected values of the learned policy and the expert's policy.

As the expert's policy is unavailable to the learner, the vector of weights is found by comparing the feature expectations from the expert's policy to feature counts empirically estimated from the observed trajectories [38]. The expert's feature expectations are estimated using the discounted average of feature values for all observed trajectories, $\hat{\phi}_k = \frac{1}{|X|} \sum_{X \in \mathscr{X}} \sum_{\langle s,a\rangle_t \in X} \gamma^t \phi_k(\langle s,a\rangle_t)$, where X is a trajectory in the set of all observed trajectories, $\mathscr{X}$, and $\gamma \in (0,1)$ is a discount factor. The learner completes the expert's MDP using the learned reward function and may solve it to obtain π_E. The difference $\hat{\phi} - \phi^{\pi_E}$ provides a gradient with respect to the reward weights for a numerical solver.

[1] Repeated trajectories in a demonstration can usually be excluded for many methods without affecting the learning.

2.2 Maximum entropy IRL

While expected to be useful in some contexts, the max-margin approach of Abeel and Ng [1] introduces a bias into the learned reward function. While biases help guide the search in ill-posed problems, they may preclude other meaningful solutions. Consequently, this motivates methods that make the least assumptions. In this regard, Ziebart et al. [38] finds the distribution over all trajectories that exhibits the maximum entropy and is constrained to match the observed feature expectations. The following nonlinear program gives this distribution.

$$\begin{array}{l} \max\limits_{\Delta} \left(-\sum_{X\in\mathbb{X}} P(X)\ log\ P(X)\right) \\ \text{subject to } \sum_{X\in\mathbb{X}} P(X) = 1 \\ E_{\mathbb{X}}[\phi_k] = \hat{\phi}_k \qquad \forall k \end{array} \tag{3}$$

Here, Δ is the space of all distributions over the set $\mathbb{X}$ of all trajectories, and $E_{\mathbb{X}}[\phi_k] = \sum_{X\in\mathbb{X}} P(X) \sum_{\langle s,a\rangle_t \in X} \gamma^t \phi_k(\langle s,a\rangle_t)$. The problem reduces to finding $\boldsymbol{\theta}$, which parameterizes the exponential distribution that exhibits the highest likelihood:

$$P(X;\boldsymbol{\theta}) \propto e^{\sum_{\langle s,a\rangle\in X} \theta^T \phi(s,a)}. \tag{4}$$

As the distribution $P(\cdot)$ is parameterized by learned weights $\boldsymbol{\theta}$, $E_{\mathbb{X}}[\phi_k]$ represents the feature expectations $\phi_k^{\pi_E}$. Notice from Eq. 4 that the chances of an expert agent following a trajectory is proportional to the cumulative reward incurred along that path. The benefit of this approach is that distribution $P(X)$ makes no further assumptions beyond those which are needed to match the constraints of (3) and is maximally noncommittal to any one trajectory. As such, it is most generalizable by being the least wrong most often of all alternative distributions. Since its introduction, there have been numerous extensions and applications of maximum entropy IRL [2, 13, 37, 39]. A disadvantage is that it becomes intractable for long trajectories because the set of trajectories grows exponentially with time steps. In this regard, another formulation defines the maximum entropy distribution over policies [14], the size of which is also large but fixed.

2.3 IRL under occlusion

Our motivating application involves a subject robot that must observe other mobile robots from a fixed vantage point. Its local sensors allow it a limited observation area; within this area, it can observe the other robots fully, outside this area it cannot observe at all. Previous methods [9, 10] denote this special case of partial observability where certain states are either fully observable or fully hidden as *occlusion*. Subsequently, the trajectories gathered by the learner exhibit missing data associated with time steps where the expert robot is in one of the occluded states. The empirical feature expectation of the expert $\hat{\phi}_k$ will thus exclude the occluded states (and actions in those states).

Bogert and Doshi [9], while maximizing entropy over policies [14], limited the calculation of feature expectations for policies to observable states only. To ensure that the feature expectation constraint of IRL methods accounts for the missing data, a recent approach [11, 12] improves on this method by taking an expectation over the missing data conditioned on the observations. Completing the missing data in this way allows the use of all states in the constraint and with it the Lagrangian dual's gradient as well. The nonlinear program in (3) is modified to account for the hidden data and its expectation.

Let Y be the observed portion of a trajectory, Z is one way of completing the hidden portions of this trajectory, and $X = Y \cup Z$. Treating Z as a latent variable and taking its expectation gives a new definition for the expert's empirical feature expectations:

$$\hat{\phi}^{Z|Y}_{\theta,k} \triangleq \frac{1}{|\mathscr{Y}|} \sum_{Y \in \mathscr{Y}} \sum_{Z \in \mathbb{Z}} P(Z|Y;\theta) \sum\nolimits_{t=1}^{T} \gamma^t \phi_k(\langle s,a \rangle_t) \tag{5}$$

where $\langle s,a \rangle_t \in Y \cup Z$, $\mathscr{Y}$ is the set of all observed Y, $\mathbb{Z}$ is the set of all possible hidden Z that can complete a trajectory. Note the underlying assumption that the learner is aware that a segment of the expert's trajectory is occluded from its view. The length of this segment is variable and need not be pre-determined. The program in (3) is modified by replacing $\hat{\phi}_k$ with $\hat{\phi}^{Z|Y}_{\theta,k}$, as we show below. Notice that in the case of no occlusion $\mathbb{Z}$ is empty and $\mathscr{X} = \mathscr{Y}$. Therefore $\hat{\phi}^{Z|Y}_{\theta,k} = \hat{\phi}_k$ and this method reduces to (3). Thus, this method generalizes the previous maximum entropy IRL method.

$$\begin{aligned} &\max_{\Delta} \left(-\sum\nolimits_{X \in \mathbb{X}} P(X)\, log\, P(X)\right) \\ &\text{subject to} \quad \sum\nolimits_{X \in \mathbb{X}} P(X) = 1 \\ &E_{\mathbb{X}}[\phi_k] = \hat{\phi}^{Z|Y}_{\theta,k} \qquad \forall k \end{aligned} \tag{6}$$

However, the program in (6) becomes nonconvex due to the presence of $P(Z|Y)$. As such, finding its optima by Lagrangian relaxation is not trivial. Wang et al. [36] suggests a log linear approximation to cast the problem of finding the parameters of distribution (feature weights) as the likelihood maximization that can be solved within the schema of expectation-maximization [16]. An application of this approach to the problem of IRL under occlusion yields the following two steps (with more details in [12]):

E-step This step involves calculating Eq. 5 to arrive at $\hat{\phi}^{Z|Y,(t)}_{\theta,k}$, a conditional expectation of the K feature functions using the parameter $\theta^{(t)}$ from the previous iteration. We may initialize the parameter vector randomly.

M-step In this step, the modified program (6) is optimized by utilizing $\hat{\phi}^{Z|Y,(t)}_{\theta,k}$ from the E-step above as the expert's constant feature expectations in order to obtain $\theta^{(t+1)}$. Now, optimizing the relaxed Lagrangian becomes easier. Normalized exponentiated gradient descent [24, 33], a version of gradient descent in which the learned parameter is scaled using the exponent of the gradient, solves the program. This variant of the gradient descent exhibits improved worst-case loss bounds than the standard gradient descent and often yields faster convergence.

As EM may converge to local minima, this process is repeated with random initial θ and the solution with the maximum entropy is chosen as the final one.

3 Incremental IRL (I2RL)

We present a framework for incremental IRL, labeled I2RL, in order to realize IRL in online settings. We identify and rigorously define the key concepts integral to online inverse learning such as sessions, stopping criteria, incremental learning, and loss. These provide a common foundation for researchers interested in developing new techniques for online IRL, which facilitates comparing between them conceptually as well as developing their theoretical properties.

3.1 Framework

To establish the definition of incremental IRL, we must first define a *session* of I2RL. Let $\hat{R}_E^0$ be an initial estimate of the expert's reward function. Our definitions reference trajectories and demonstrations, which were previously defined in Sect. 2.1.

Definition 3 *(Session)* A session of I2RL, $\zeta_i(MDP_{/R_E}, \mathscr{X}_i, \hat{R}_E^{i-1})$, $i > 0$ and $i \in \mathbb{N}$, takes as input the expert's MDP sans the reward function, the current (i^{th}) demonstration, $\mathscr{X}_i \in 2^{\mathbb{X}}$, and the reward function estimated previously. It yields a revised estimate of the expert's reward function, $\hat{R}_E^i$.

Note that we may replace the reward function estimates with some parameter sufficiently representing it (e.g., $\boldsymbol{\theta}$). Also, for expedience in formal analysis, we may impose the assumption that the trajectories in a session $\mathscr{X}_i$ are i.i.d. as the trajectories in the previous session.[2] When the trajectories in $\mathscr{X}_i$ are i.i.d., the demonstrations $\mathscr{X}_i, i \in \{1, 2, \ldots\}$ are also i.i.d. This assumption enables deriving probabilistic convergence bounds by making it possible to apply Hoeffding's inequality, as we demonstrate later in this article.

We can let the sessions run indefinitely. Alternately, we may establish some stopping criteria for the incremental learning, which then offer a basis to automatically terminate the sessions once the criterion is satisfied. Let $LL(\hat{R}_E^i | \mathscr{X}_{1:i})$ be the log likelihood of the demonstrations received up to the i^{th} session given the current estimate of the expert's reward function. We may view this likelihood as a measure of how well the learned reward function explains the observed data. In the context of I2RL, the log likelihood must be computed without storing data from previous sessions. Here onwards, $\widehat{\mathscr{X}}$ denotes a sufficient statistic that replaces *all input trajectories from previous sessions* in the computation of log likelihood.

Definition 4 *(Stopping criterion #1)* Terminate the sessions of I2RL when $|LL(\hat{R}_E^i | \mathscr{X}_i, \widehat{\mathscr{X}}) - LL(\hat{R}_E^{i-1} | \mathscr{X}_{i-1}, \widehat{\mathscr{X}})| \leqslant \rho$, where ρ is a very small positive number.

Definition 4 reflects the fact that additional sessions are not improving the learning performance significantly. On the other hand, a more effective stopping criterion is possible if we know the expert's true policy. We utilize the *inverse learning error* [15] in this criterion, which gives the loss of value if learner uses the learned policy on the task instead of the expert's: $ILE(\pi_E^*, \pi_E) = ||V^{\pi_E^*} - V^{\pi_E}||_1$. Here, $V^{\pi_E^*}$ is the optimal value function of E's MDP and V^{π_E} is the value function due to utilizing the policy π_E (obtained from solving the MDP with the learned reward function) in E's true MDP. The norm is needed as the value functions are vectors over the states. Notice that when the learned reward function results in an optimal policy identical to E's true policy, $\pi_E^* = \pi_E$, ILE will be zero; it increases monotonically as the two policies increasingly diverge in value. Instead of using an absolute difference, our experiments use a normalized difference, $\overline{ILE}(\pi_E^*, \pi_E) = \frac{||V^{\pi_E^*} - V^{\pi_E}||_1}{||V^{\pi_E^*}||_1}$.

[2] This assumption holds when each session starts from the same state and the trajectories are produced by the expert's fixed policy. In case of occlusion, even though inferring the hidden portion Z of a trajectory $X \in \mathscr{X}_i$, is influenced by the visible portion, Y, this does not make the trajectories necessarily dependent on each other.

Let π_E^i be the optimal policy obtained from solving the expert's MDP with the reward function $\hat{R}_E^i$ learned in session i. Another stopping criterion is then defined as given below.

Definition 5 *(Stopping criterion #2)* Terminate the sessions of I2RL when $\overline{ILE}(\pi_E^*, \pi_E^{i-1}) - \overline{ILE}(\pi_E^*, \pi_E^i) \leqslant \rho$, where ρ is a very small positive error and is given.

Obviously, prior knowledge of the expert's policy is not common in practice. Therefore, we view this criterion as being more useful during the formative evaluation of I2RL methods.

Utilizing Definitions 3, 4, and 5, we formally define I2RL next.

Definition 6 (I2RL) Incremental IRL is a sequence of learning sessions $\{\zeta_1(MDP_{/R_E}, \mathscr{X}_1, \hat{R}_E^0), \zeta_2(MDP_{/R_E}, \mathscr{X}_2, \hat{R}_E^1), \zeta_3(MDP_{/R_E}, \mathscr{X}_3, \hat{R}_E^2), \ldots, \}$, which continue infinitely or until a stopping criterion assessing convergence is met (criterion #1 or #2 depending on which one is chosen a'priori).

The previous reward function estimate in each session may be replaced with some parameter(s) sufficiently representing it.

Regret is a common performance measure for online learning methods. It is often defined based on an expected value of cumulative loss or gain. For example, in the context of online learning in the multi-arm bandit problem [7], external regret is defined as the difference between the expected value of total loss from method M, and the total loss from the best decision in hindsight.

$$Regret_T(M) = E_{\{s_t|t\in\{1,2,\ldots T\}\}}\left[\Sigma_{t=1}^T l(s_t, a_t^M) - \min_{a\in A} \Sigma_{t=1}^T l(s_t, a)\right]$$

where $l(\cdot,\cdot)$ is a loss function, s_t is the situation at round t, and a_t^M is the decision that method M makes in situation s_t. If the distribution over $\{s_t|t \in \{1,2,\ldots T\}\}$ is unknown, it may be estimated as

$$Regret_T(M) = \frac{1}{T}\left[\Sigma_{t=1}^T l(s_t, a_t^M) - \min_{a\in A} \Sigma_{t=1}^T l(s_t, a)\right]. \tag{7}$$

We may model the sessions of I2RL as the iterations of an adversarial iterative game with the learner as a player, and the IRL hypotheses $\{\hat{R}_E^i\}$ as the decisions made by the player. Note that the best possible hypothesis for $\hat{R}_E^i$ is the true reward function R_E of the expert. Thus, on the one hand the true minimum loss achievable relative to R_E is 0; on the other hand the loss from decision $\hat{R}_E^i$ can be measured in terms of the log likelihood loss $\left(LL(R_E|\mathscr{X}_i, \hat{\mathscr{X}}) - LL(\hat{R}_E^{i-1}|\mathscr{X}_{i-1}, \widehat{\mathscr{X}'})\right)$. Plugging these choices into Eq. 7, we get the following definition of average regret:

Definition 7 *(Average Regret in* I2RL*)* With $\left(LL(R_E|\mathscr{X}_i, \hat{\mathscr{X}}) - LL(\hat{R}_E^{i-1}|\mathscr{X}_{i-1}, \widehat{\mathscr{X}'})\right)$ as the loss incurred in session i by the learning algorithm M_{I2RL}, average regret after T sessions is given by:

$$Regret_T(M_{\text{I2RL}}) = \frac{1}{T}\Sigma_{i=1}^T\left(LL(R_E|\mathscr{X}_i, \hat{\mathscr{X}}) - LL(\hat{R}_E^{i-1}|\mathscr{X}_{i-1}, \widehat{\mathscr{X}'})\right).$$

Table 1 A conceptual comparison of the existing online methods cast in the I2RL framework along with the new method presented in this article

Method	Session definition	Session output	Stopping criterion
LP-I2RL	$(MDP_{/R_E}, \mathscr{X}_i, \hat{R}_E^{i-1})$	$\hat{R}_E^i$	None (until no more data)
DARKO	$(MDP_{/R_E}, \mathscr{X}_i, \boldsymbol{\theta}^{i-1})$	$\boldsymbol{\theta}^i$	None (until no more data)
Online-GAIL	$(MDP_{/R_E}, \mathscr{X}_i, D_{\boldsymbol{\theta}^{i-1}}, G_{\eta^{i-1}})$	$(D_{\boldsymbol{\theta}^i}, G_{\eta^i})$	None (until no more data)
LME-I2RL	$(MDP_{/R_E}, \mathscr{Y}_i, \lvert\mathscr{Y}_{1:i-1}\rvert, \hat{\phi}_{\boldsymbol{\theta}^{i-1}}^{Z\lvert Y,1:i-1}, \boldsymbol{\theta}^{i-1})$	$(\boldsymbol{\theta}^i, \lvert\mathscr{Y}_{1:i}\rvert, \hat{\phi}_{\boldsymbol{\theta}^i}^{Z\lvert Y,1:i})$	Likelihood not improved (Definition 4)

While somewhat straightforward, these rigorous definitions for I2RL allow us to conceptually situate the few existing online IRL techniques, and to introduce online IRL with hidden variables, as we see next.

3.2 Existing methods in I2RL

One of our objectives in developing I2RL is to facilitate a portfolio of online methods under the framework of I2RL each with its own appealing properties. This will enable online IRL in various applications. We may easily present the method by Jin et al. [21] within the framework of I2RL. A session of this method $\zeta_i(MDP_{/R_E}, \mathscr{X}_i, \hat{R}_E^{i-1})$ is realized as follows: Each $\mathscr{X}_i$ is a single state-action pair $\langle s, a\rangle$ and initial reward function $\hat{R}_E^0 = \frac{1}{\sqrt{|S_E|}}$. For $i > 0$, $\hat{R}_E^i = \hat{R}_E^{i-1} + \alpha \cdot v_i$, where v_i is the difference in the expected value of the observed action a at state s and the (predicted) optimal action obtained by solving the MDP with the reward function $\hat{R}_E^{i-1}$, and α is the learning rate. While no explicit stopping criterion is specified, the incremental method terminates when it runs out of observed state-action pairs.

Another I2RL method is the DARKO algorithm, used for first person activity forecasting [31]. Casting the method to the framework of I2RL, a session of this method is $\zeta_i(MDP_{/R_E}, \mathscr{X}_i, \boldsymbol{\theta}^{i-1})$, which yields $\boldsymbol{\theta}^i$. When the person wearing a camera stops the current activity, the stoppage is perceived as reaching a goal state in MDP. Input demonstration for the session, $\mathscr{X}_i$, comprises all the activity trajectories observed since the end of the previous goal until the next goal is reached. The session IRL finds the reward weights $\boldsymbol{\theta}^i$ that minimize the margin $E_{\mathbb{X}}[\phi|\pi_E^*] - \hat{\phi}$ using gradient descent. Here, the expert's policy π_E^* is obtained by using soft value iteration for solving the complete MDP that includes a reward function estimate obtained using previous weights $\boldsymbol{\theta}^{i-1}$. No stopping criterion is utilized for the online learning, thereby emphasizing its continuity.

GAIL [20] is a state-of-the-art offline policy learning method cast in the schema of generative adversarial networks. In GAIL, learning happens by training the generator G_η representing the learned parameterized policy π_η while the discriminator $D_{\boldsymbol{\theta}}$ represents the learned reward function.

We modified GAIL to be online and cast it under the I2RL framework. A session of online GAIL is $\zeta_i(MDP_{/R_E}, \mathscr{X}_i, D_{\boldsymbol{\theta}^{i-1}}, G_{\eta^{i-1}})$, with $(D_{\boldsymbol{\theta}^i}, G_{\eta^i})$ as outputs of inference. Each session involves two steps: a gradient ascent on $\boldsymbol{\theta}$, which increases the entropy-regularized occupancy-measure loss w.r.t. $D_{\boldsymbol{\theta}}$, and a TRPO step on η, which decreases the loss w.r.t. G_η. As there is no stopping criteria specified, the learning continues until the input demonstrations stop.

Table 1 provides a quick conceptual comparison between the three online IRL methods discussed in this subsection as well as the new method, which we present next.

4 Incremental latent MaxEnt

We present a new method for online IRL under the I2RL framework, which modifies the latent maximum entropy (LME) optimization reviewed in the Background section. It offers the capability to perform online IRL in contexts where portions of the observed trajectory may be occluded.

For differentiation, we refer to the original method as the *batch* version. Recall the k^{th} feature expectation of the expert computed in Eq. 5 as part of the E-step. $\hat{\phi}^{Z|Y,i}_{\theta^i,k}$ is the expectation of k^{th} feature for the demonstration obtained in i^{th} session, $\hat{\phi}^{Z|Y,1:i}_{\theta^i,k}$ is the expectation computed for all demonstrations obtained until the i^{th} session, we may rewrite Eq. 5 for feature k as:

$$\begin{aligned}
\hat{\phi}^{Z|Y,1:i}_{\theta^i,k} &\triangleq \frac{1}{|\mathscr{Y}_{1:i}|}\sum_{Y\in\mathscr{Y}_{1:i}}\sum_{Z\in\mathbb{Z}} P(Z|Y;\boldsymbol{\theta})\sum_{t=1}^{T(i)}\gamma^t\phi_k(\langle s,a\rangle_t)\\
&= \frac{1}{|\mathscr{Y}_{1:i}|}\sum_{Y\in(\mathscr{Y}_{1:i-1}\cup\mathscr{Y}_i)}\sum_{Z\in\mathbb{Z}} P(Z|Y;\boldsymbol{\theta})\sum_{t=1}^{T(i)}\gamma^t\phi_k(\langle s,a\rangle_t)\\
&= \frac{1}{|\mathscr{Y}_{1:i}|}\Bigg(\sum_{Y\in\mathscr{Y}_{1:i-1}}\sum_{Z\in\mathbb{Z}} P(Z|Y;\boldsymbol{\theta})\sum_{t=1}^{T(i)}\gamma^t\phi_k(\langle s,a\rangle_t)+\\
&\sum_{Y\in\mathscr{Y}_i}\sum_{Z\in\mathbb{Z}} P(Z|Y;\boldsymbol{\theta}^i)\sum_{t=1}^{T(i)}\gamma^t\phi_k(\langle s,a\rangle_t)\Bigg)\\
&= \frac{1}{|\mathscr{Y}_{1:i-1}|+|\mathscr{Y}_i|}\left(|\mathscr{Y}_{1:i-1}|\,\hat{\phi}^{Z|Y,1:i-1}_{\theta^{i-1},k}+|\mathscr{Y}_i|\,\hat{\phi}^{Z|Y,i}_{\theta^i,k}\right)\\
&\text{(Using Eq. 5 and } |\mathscr{Y}_{1:i}|=|\mathscr{Y}_{1:i-1}|+|\mathscr{Y}_i|)\\
&= \frac{|\mathscr{Y}_{1:i-1}|}{|\mathscr{Y}_{1:i-1}|+|\mathscr{Y}_i|}\,\hat{\phi}^{Z|Y,1:i-1}_{\theta,k}+\frac{|\mathscr{Y}_i|}{|\mathscr{Y}_{1:i-1}|+|\mathscr{Y}_i|}\,\hat{\phi}^{Z|Y,i}_{\theta,k}
\end{aligned}\tag{8}$$

A session of incremental LME takes as input the expert's MDP sans the reward function, the current session's trajectories, the number of trajectories observed until the previous session, the expert's empirical feature expectation and reward weights from previous session. More concisely, each session is denoted by $\zeta_i(MDP_{/R_E},\mathscr{Y}_i,|\mathscr{Y}_{1:i-1}|,\hat{\phi}^{Z|Y,1:i-1}_{\theta^{i-1}},\boldsymbol{\theta}^{i-1})$. The sufficient statistic $\hat{X}$ for the session comprises $(|\mathscr{Y}_{1:i-1}|,\hat{\phi}^{Z|Y,1:i-1}_{\theta^{i-1}})$. In each session, the feature expectations using that session's observed trajectories are computed, and the output feature expectations $\hat{\phi}^{Z|Y,1:i}_{\theta^i,k}$ are obtained by including these as shown above in Eq. 8, which is then used in the M-step. The equation shows how computing sufficient statistic replaces the need for storing the data input in previous sessions. Of course, each session may involve several iterations of the E- and M-steps until the converged reward weights $\boldsymbol{\theta}^i$ are obtained thereby giving the corresponding reward function estimate. We refer to this method as LME I2RL.

Wang et al. [35] shows that if the distribution over the trajectories in (6) is log linear, then the reward function that maximizes the entropy of the distribution over trajectories also maximizes the log likelihood of the observed portions of the trajectories. Given this linkage with log likelihood, the stopping criterion #1 as given in Definition 4 can be utilized. As shown in Algorithm 1, the sessions will terminate when, $|LL(\boldsymbol{\theta}^i|\mathscr{Y}_i, |\mathscr{Y}_{1:i-1}|, \hat{\phi}^{Z|Y,1:i-1}_{\boldsymbol{\theta}^{i-1}}, \boldsymbol{\theta}^{i-1}) - LL(\boldsymbol{\theta}^{i-1}|\mathscr{Y}_{i-1}, |\mathscr{Y}_{1:i-2}|, \hat{\phi}^{Z|Y,1:i-2}_{\boldsymbol{\theta}^{i-2}}, \boldsymbol{\theta}^{i-2})| \leq \rho$, where $\boldsymbol{\theta}^i$ fully parameterizes the reward function estimate for the i^{th} session and ρ is a given acceptable difference.

Algorithm 1 Algorithm INCREMENTAL-LME($MDP_{/R_E}$,ϕ,ρ)

```
1: $i \leftarrow 1$; $\mathscr{Y}_{1:i-1} \leftarrow \emptyset$
2: $\hat{\phi}^{Z|Y,1:i-1}_{\boldsymbol{\theta}^{i-1},k} \leftarrow 0$; $[\boldsymbol{\theta}^0]_k \sim \text{uniform}(0,1)$
3: while $|LL(\mathscr{Y}_i, |\mathscr{Y}_{i-1}|, \hat{\phi}^{Z|Y,1:i-1}_{\boldsymbol{\theta}^{i-1}}, \boldsymbol{\theta}^i) - LL(\mathscr{Y}_{i-1}, |\mathscr{Y}_{i-2}|, \hat{\phi}^{Z|Y,1:i-2}_{\boldsymbol{\theta}^{i-2}}, \boldsymbol{\theta}^{i-1})| > \rho$ do
4:     /* session $\zeta_i(M_{/R_E}, \mathscr{Y}_i, |\mathscr{Y}_{1:i-1}|, \hat{\phi}^{Z|Y,1:i-1}_{\boldsymbol{\theta}^{i-1}}, \boldsymbol{\theta}^{i-1})$ */
5:     repeat
6:         /* E-step */
7:         Use MCMC to sample trajectories from $P((Y,Z)|\boldsymbol{\theta}^{i-1})$, and compute $\hat{\phi}^{Z|Y,i}_{\boldsymbol{\theta}^i}$ for sampled trajectories.
8:         /* Updating feature expectations using sufficient statistic. */
9:         Use Equation 8 to compute $\hat{\phi}^{Z|Y,1:i}_{\boldsymbol{\theta}^i,k}$ for all $k$.
10:        $|\mathscr{Y}_{1:i}| \leftarrow |\mathscr{Y}_{1:i-1}| + |\mathscr{Y}_i|$
11:        /* M-step */
12:        $\boldsymbol{\theta}_0 \leftarrow \boldsymbol{\theta}^{i-1}, t \leftarrow 1$
13:        repeat
14:            Compute $\pi^*_{E,(t-1)}$ using $\boldsymbol{\theta}_{(t-1)}$ and $E_{\mathbb{X}}[\phi_k]$ using trajectories sampled from $\pi^*_{E,(t-1)}$.
15:            $z \leftarrow \hat{\phi}^{Z|Y,1:i}_{\boldsymbol{\theta}^i} - E_{\mathbb{X}}[\phi]$   {gradient}
16:            $\boldsymbol{\theta}_{t,k} \leftarrow \frac{\boldsymbol{\theta}_{(t-1),k}\exp(-\eta z_{(t-1),k})}{\sum_{i=1}^k \boldsymbol{\theta}_{(t-1),k}\exp(-\eta z_{(t-1),k})}$
17:            $t \leftarrow t+1$
18:        until $|z| \leq \varepsilon/(1-\gamma)$
19:    until gradient of likelihood $\approx 0$
20:    Compute $\hat{\pi}_i$ using learned reward $\boldsymbol{\theta}^i \leftarrow \boldsymbol{\theta}_t$.
21:    $i \leftarrow i+1$
```

4.1 Convergence bounds

LME I2RL admits some significant convergence guarantees with a confidence of meeting the specified error on the demonstration likelihood. To establish the guarantees of LME I2RL, we first focus on the full observability setting. For a desired relaxed bound $\varepsilon/(1-\gamma)$ on the feature matching constraint (see line 18 in Algorithm 1) for session i, the confidence is bounded as follows:

Theorem 1 (Confidence for ME I2RL) *Given $\mathscr{X}_{1:i}$ as the (fully observed) demonstration until session i, $\boldsymbol{\theta}_E \in [0,1]^K$ is the expert's weights, and $\boldsymbol{\theta}^i$ is the converged weight vector for session i for ME I2RL, we have,*

$$LL(\boldsymbol{\theta}_E|\mathscr{X}_{1:i}) - LL(\boldsymbol{\theta}^i|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\phi}^{1:i-1}, \boldsymbol{\theta}^{i-1}) \leqslant \frac{2K\varepsilon}{(1-\gamma)}$$

with probability at least $\max(0, 1-\delta)$, *where* $\delta = 2K\exp(-2|\mathscr{X}_{1:i}|\varepsilon^2)$.

The proof of this theorem is given in the Appendix.

Note that sufficient statistic $\hat{X}$ for the full-observability scenario is $(|\mathscr{X}_{i-1}|, \hat{\phi}^{1:i-1})$. Theorem 1 also holds for the online method by Rhinehart et al. [31] because it uses incremental (full-observability) maximum entropy IRL. As the method implements online learning without an incremental update of feature expectations of the expert, we set $\hat{\phi}^{1:i} = \hat{\phi}^{i}$; an absence of sufficient statistic means that $|\mathscr{X}_{i-1}| = 0$, and set $\hat{\phi}_k^{1:i-1} = 0, \forall k$ in Theorem 1. This demonstrates the benefit of Theorem 1 to relevant methods.

Relaxing the full observability assumption, the following lemma, whose proof is available in the Appendix, proves that LME I2RL converges monotonically.

Lemma 1 (Monotonicity) *LME I2RL increases the demonstration likelihood monotonically with each new session,* $LL(\theta^i|\mathscr{Y}_i, |\mathscr{Y}_{1:i-1}|, \hat{\phi}_{\theta^{i-1}}^{Z|Y,1:i-1}, \theta^{i-1}) - LL(\theta^{i-1}|\mathscr{Y}_{i-1}, |\mathscr{Y}_{1:i-2}|, \hat{\phi}_{\theta^{i-2}}^{Z|Y,1:i-2}, \theta^{i-2}) \geqslant 0$, *when* $|\mathscr{Y}_{1:i-1}| \gg |\mathscr{Y}_i|$.

While Lemma 1 suggests that the log likelihood of the demonstration can only improve from session to session after learner has accumulated a significant amount of observations, a stronger result illuminates the confidence with which LME I2RL approaches, over a sequence of sessions, the log likelihood of the expert's true weights θ_E. As a step toward such a result, we first consider the error in approximating the feature expectations of the unobserved portions of the data, accumulated from the first to the current session of I2RL. Notice that $\hat{\phi}_{\theta^i,k}^{Z|Y,1:i}$ given by Eq. 8 is an approximation of the full-observability expectation $\hat{\phi}_k^{1:i}$, computed by sampling the hidden Z from $P(Z|Y,\theta^{i-1})$ [12]. The following lemma, whose proof is given in Appendix, relates the error due to this sampling-based approximation, i.e., $\left|\hat{\phi}_k^{1:i} - \hat{\phi}_{\theta^i,k}^{Z|Y,1:i}\right|$, to the difference between feature expectations for learned policy and that estimated for the expert's true policy.

Lemma 2 *(Constraint Bounds for LME* I2RL*) Suppose* $\mathscr{X}_{1:i}$ *has portions of trajectories in* $\mathbb{Z}_{1:i} = \{Z|(Y,Z) \in \mathscr{X}_{1:i}\}$ *occluded from the learner. Let* ε_s *be a given bound on the error* $|\hat{\phi}_k^{1:i} - \hat{\phi}_{\theta^i,k}^{Z|Y,1:i}|_1, k \in \{1, 2 \ldots K\}$ *after* n_s *samples for approximation. Then, with probability at least* $\max(0, 1-(\delta+\delta_s))$, *the following holds*:

$$\left|(1-\gamma)\left(E_{\mathbb{X}}[\phi_k] - \hat{\phi}_{\theta^i,k}^{Z|Y,1:i}\right)\right|_1 \leqslant \varepsilon + \varepsilon_s, k \in \{1, 2 \ldots K\}$$

where ε, δ *are as defined in Theorem* 1, *and* $\delta_s = 2K\exp(-2n_s\varepsilon_s^2)$.

LME I2RL computes θ^i by an optimization process using the result $\phi^{Z|Y,i}$ of the E step (sampling of occluded data) of current session along with other inputs (feature expectations and θ computed from previous session) which, in turn, depend on the sampling process in previous sessions. Theorem 1 and Lemma 2 allow us to probabilistically bound the error in log likelihood for LME I2RL:

Theorem 2 *(Confidence for LME* I2RL*) Let* $\mathscr{Y}_{1:i} = \{Y|(Y,Z) \in \mathscr{X}_{1:i}\}$ *be the observed portions of the demonstration until session i.* ε *and* ε_s *are inputs as defined in Lemma* 2, *and* θ^i *is the solution of session i for LME I2RL. Then*

$$LL(\theta_E|\mathscr{Y}_{1:i}) - LL(\theta^i|\mathscr{Y}_i, |\mathscr{Y}_{i-1}|, \hat{\phi}_{\theta^{i-1}}^{Z|Y,1:i-1}, \theta^{i-1}) \leq \frac{4K\varepsilon_l}{(1-\gamma)}$$

with confidence at least $\max(0, 1-\delta_l)$, *where* $\varepsilon_l = \frac{\varepsilon+\varepsilon_s}{2}$, *and* $\delta_l = \delta + \delta_s$.

The proof of this theorem is given in Appendix.

Given $\varepsilon, \varepsilon_s, N$ and the total number of input partial-trajectories, $|\mathscr{Y}_{1:i}|$, Theorem 2 gives the confidence $1-\delta_l$ for I2RL under occlusion. Equivalently, the number of observed trajectories $|\mathscr{Y}_{1:i}|$ can be derived using desired error bounds and confidence. As a boundary case of LME I2RL, if learner ignores occluded data (no sampling or $n_s = 0$ for E-step), the confidence for convergence becomes zero because δ_s becomes larger than 1.

A desirable feature is for an online learning algorithm to be *no-regret*, i.e., where the average regret vanishes in the limit. In the context of ME I2RL, we follow our definition of average regret (Definition 7) up to session $i = T$ as

$$Regret_T(M_{ME\text{I2RL}}) = \frac{1}{T}\sum_{i=1}^{T} LL(\boldsymbol{\theta}_E|\mathscr{X}_{1:i}) - \frac{1}{T}\sum_{i=1}^{T} LL(\boldsymbol{\theta}^i|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\boldsymbol{\phi}}^{1:i-1}, \boldsymbol{\theta}^{i-1})$$

This is the regret that I2RL experiences in hindsight, in terms of log-likelihood loss, for returning $\boldsymbol{\theta}^i$ instead of $\boldsymbol{\theta}_E$ after session i, $i = 1, \ldots, T$, averaged over the T sessions. Theorem 1 implies that with a high confidence, the above expression for average regret is constant bounded, specifically by $2K\epsilon/(1-\gamma)$. However, by setting a diminishing (variable) threshold in line 18 of Algorithm 1, the total regret can be made to grow slower than T, so that the average regret vanishes in the limit (as $T \to \infty$). One such choice (by no means unique) for a variable ε for line 18 of Alg. 1 is

$$\varepsilon_i/(1-\gamma) = c/i \tag{9}$$

where c is some constant and i is the session index. This choice ensures that with a high confidence, $Regret_T(M_{ME\text{I2RL}})$ is $O(\log T)/T$, which indeed vanishes in the limit. We formalize this intuition for the fully observable setting (i.e., in the context of Theorem 1; a similar result follows in the context of Theorem 2 as well) in the following theorem.

Theorem 3 (No Regret Learning) *There exist choices for a variable threshold bound on the feature matching constraint,* ε_i *as a function of session i, such that with probability at least* $\max(0, \prod_{i=1}^{i=T}(1-\delta_i))$, *where* $\delta_i = 2K\exp(-2|\mathscr{X}_{1:i}|\varepsilon_i^2)$,

$$Regret_T(M_{ME\text{I2RL}}) = o(T)/T,$$

thus approaching 0 *as* $T \to \infty$.

Note that the above theorem assumes that line 18 of Alg. 1 exits in *every* session. If this does not occur in some session, for instance if the algorithm gets stuck in a local optima which becomes increasingly likely as the threshold in Eq. 9 tightens, then the theorem does not apply. The proof of this theorem is given in the Appendix.

In contrast with Theorem 1 (and Theorem 2 in the partially observable case), Theorem 3 assesses the *asymptotic behavior* of the log likelihood loss when accumulated over multiple sessions. This takes a long term view of the learner's performance, and the theorem itself demands a gradually improving performance by this measure, while Theorem 1 merely improves the confidence (δ) on the learner's performance with accumulating sessions, without similarly demanding an improved performance (e.g., by reducing the log likelihood loss). The price for demanding greater long term accuracy is the diminishing schedule of ϵ, which may become increasingly harder to meet.

5 Experiments

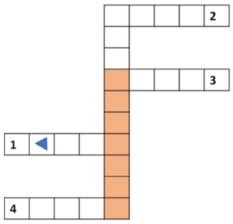
We evaluate the performance of I2RL on three instances of the previously introduced patrolling domain [9], where an observing robot is tasked with penetrating a perimeter patrol on a grid undetected. In the following sections, we first explain the domain and the particular instances used in our experiments. In Sect. 5.2, we show how this task is modeled as an MDP. Next, we evaluate the performance of LME I2RL on experiments in the domain both in simulation and on physical robots. Section 5.3 compares LME I2RL with the batch method in simulations of two instances of the domain. None of the existing online IRL methods discussed in Sect. 3.2 are easily amenable to learning under occlusion, except for GAIL which we include in our comparisons. For example, Jin et al. [21] utilize a single state-action pair in each session. If this state-action pair is occluded, there is no input and the reward function is not updated. After simulations, in Sect. 5.4 we show a similar comparison on physical robots on two instances of the domain, one of which is significantly larger than the other.

5.1 Domain: observing and penetrating cyclic patrols

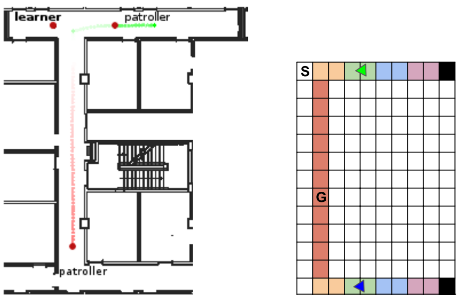
Bogert and Doshi [9] introduced the domain of robot patrolling for evaluating IRL under occlusion and simulated it in ROS-based Player Stage [18]. It involves a robotic learner observing patrollers from a vantage point with a limited view continuously navigate a hallway in cyclic trajectories. The learner is tasked with reaching a goal location without being spotted by any of the patrollers in a given amount of time. Each patroller can see up to three grid cells in front. This domain differs from the usual applications of IRL toward imitation learning. In particular, the learner must solve its own distinct decision-making problem (modeled as another MDP), which is dependent on correctly predicting the patrols. The latter can be estimated from inferring each patroller's preferences given that the learner knows their dynamics.

We evaluate the performance of LME I2RL on three instances of the patrolling domain. The first instance, shown in Fig. 1a, involves a single patroller covering fully four hallways protruding from a common corridor in a loop. The learner is able to see a portion of just the corridor (shown shaded) from its vantage point leading to about 70% of the patroller's trajectory being occluded from its view. We utilize this first instance to evaluate the accuracy of learning only. Figure 1b shows a second instance of the patrolling domain, also utilized by Bogert and Doshi previously. The map for this instance pertains to a portion of the fifth floor of the Boyd building on the University of Georgia campus. Two patrollers execute, independently for the most part, a cyclic trajectory with the learner able to view just 32% of the trajectory from its vantage point in the physical instance. The patrollers engage in coordinated motion when they pass by each other to avoid collisions. The learner is tasked with reaching the cell location marked 'G' without being spotted by any of the patrollers in a given amount of time. Consequently, this instance requires the learner to utilize the learned behaviors of the patrollers in its own planning problem and execute its plan.

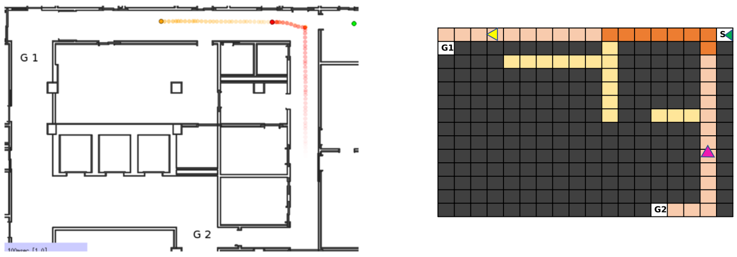
The final instance of the domain, shown in Fig. 1c, involves two patrollers executing cyclic trajectories in a significantly larger space (also on Boyd's fifth floor) with the learner located in a room and viewing outside. Both patrollers and the learner can also enter two rooms whose entrances lie in the two hallways. The learner is able to view just 18% of the



(a) A single patroller denoted by the triangle moves clockwise to the ends of four hallways in the numbered order, and only the shaded area is visible to the learner.



(b) Two patrollers on the right navigate the hallways in an environment, which simulates that of a real building [9]. 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.



(c) A larger instance of the multi-robot patrolling domain. The two patrollers are navigating the hallway while the learner penetrates the patrol after learning has finished. The patrollers and their corresponding trails are shown in yellow and light pink, the learner is shown in green. The adjacent grid indicates navigable cells of hallways in orange and those of rooms in yellow. We show the two goal locations in the grid. S is the start location, G1 and G2 are the two goal locations. The area visible to the learner is shown in dark orange at the top right region of the grid.

Fig. 1 We use three differently-sized instances of the patrolling domain [9] for evaluating the performance of LME I2RL. The learner is unaware of where each patroller turns around, their speed, and navigation capabilities

patrolling trajectory from its vantage point, and is tasked with reaching one of two possible goal locations without being detected by any patroller. Consequently, this instance uses a map that is significantly larger than the previous ones, potentially more complex trajectories and planning, and significantly greater occlusion.

5.2 Model setup

LME I2RL, as with most other IRL methods, ascribes an MDP sans the reward function to model the expert's task behavior. In all instances of the domain, each patroller's behavior is modeled using an MDP. The state of each 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. Each patroller executes one of four possible actions: move-forward, turn left or turn right 90 degrees, and stop. The motion model of the patroller, modeled as the transition function in the MDP, is stochastic. The MDP for instance 1 of the domain has 192 states, instance 2 has 124 states while the MDP for the third instance has 184 states.

All reward functions are sufficiently modeled as a weighted combination of pre-determined feature functions where the weights are unknown. The reward function of the single patroller in Fig. 1a utilizes *four* feature functions. Each function activates when the patroller reaches the end of the previously numbered hallway and remains activated until the end of the target hallway is reached, thereby enabling clockwise patrolling. To continue moving out of the hallways and in the vertical corridor (not a part of any hallway), the state in this domain instance additionally includes the recently visited hallway. A hallway is deemed to have been visited when the navigating agent has reached its end. The four binary feature functions are:

- *SwitchToHallway1(s,a)* returns 1 when action a in state s makes the patroller move from hallway 4 to hallway 1 in s', otherwise 0;
- *SwitchToHallway2(s,a)* returns 1 when action a in state s makes the patroller move from hallway 1 to 2;
- *SwitchToHallway3(s,a)* returns 1 when action a in state s makes the patroller move from hallway 2 to 3; and
- *SwitchToHallway4(s,a)* returns 1 when action a in state s makes the patroller move from hallway 3 to 4.

If equal weights are given to each of the above features, the MDP policy then guides the patroller to cycle through the four hallways in a clockwise manner.

The reward function for each patroller in the second instance of the domain utilizes *six* binary state-action feature functions, which divide the grid broadly into five regions as shown in Fig. 1b.

- *HasMoved(s,a)* returns 1 if action a at state s makes the patroller change its grid cell, 0 otherwise;
- *Turn1(s,a)* returns 1 if action a in state s makes the patroller turn (left or right) in the region of the hallway shaded orange in Fig 1b;
- *Turn2(s,a)* returns 1 if action a in state s makes the patroller turn in the region of the hallway shaded yellow in Fig. 1b;
- *Turn3(s,a)* returns 1 if action a in state s makes the patroller turn in the region of the hallway shaded green in Fig. 1b;
- *Turn4(s,a)* returns 1 if action a in state s makes the patroller turn in the region of the hallway shaded blue in Fig. 1b;
- *Turn5(s,a)* returns 1 if action a in state s makes the patroller turn in the region of the hallway shaded magenta in Fig. 1b.

A weight vector $\boldsymbol{\theta}_E$ for these features such as $\langle .57, 0, 0, 0, .43, 0\rangle$ makes each of the two patrollers constantly execute cyclic trajectory that involves turning around in the region shaded blue of the top and bottom hallway.

For the third instance of the domain (Fig. 1c), the reward function is composed of the following *five* feature functions.

- *HasMoved(s,a)* returns 1 if action a at state s makes the patroller change its grid cell, 0 otherwise;
- *InRoom(s,a)* returns 1 if the patroller in state s is inside a room, 0 otherwise;

- *Turn(s,a)* returns 1 if action a in state s makes the patroller turn in a hallway;
- *EnterRoom(s,a)* returns 1 if action a at state s makes the patroller enter a room from one of the hallways;
- *LeaveRoom(s,a)* returns 1 if action a at state s makes the patroller enter a hallway from one of the rooms.

A weight vector of $\langle 1, -1, .1, 0, 0\rangle$ gives the highest preference to constantly moving, some preference to turning around in the hallways, and the least preference to being in a room. No reward is given for entering or leaving the rooms. As a result, the patrollers keep patrolling the hallways and turn around just before the hallways end.

For the second and third instances, after learning the reward function, the learner computes the policies for both patrollers. LME I2RL can utilize both finite- and infinite-horizon look aheads in the MDPs. It uses the policies and recently observed locations to predict the future locations of the two patrollers. The learner's own MDP has the same state space as the patroller's and additionally includes a discrete timestep as a fourth dimension of its state. As the patrollers are constantly moving and the learner incurs a penalty for being spotted, the timestep allows the learner to represent the map with the patrollers' current location included. The learner solves its MDP over a finite horizon to obtain a policy that guides its actions to reach the goal location. On seeing both patrollers, it waits until a state with a positive value occurs before moving.

5.3 Performance evaluation in simulation

As the learner's vantage point limits its observability, the patrolling domain requires IRL but under occlusion. To establish the benefit of I2RL for LME, we compare the performances of both its *batch* and *incremental* variants. Efficacy of the methods is compared using the following *metrics*:

- *Learned behavior accuracy (LBA)* is the proportion of all states at which the actions prescribed by the inversely learned policies of both patrollers coincide with their actual actions;
- *Inverse learning error (ILE)*, as previously defined in Sect. 3.1, is the value loss due to using the learned policy in the expert's MDP; and
- *Success rate* is the percentage of runs where the learner reaches the goal state undetected by the patrollers, which is the culmination of learning the patrollers' trajectories accurately, planning, and navigating to the goal location.

Note that when the learned behavior accuracy is high, the ILE is expected to be low. However, as MDPs admit multiple optimal policies, a low ILE need not translate into a high behavior accuracy. As such, these two metrics are not strictly correlated.

We report the LBA, ILE, and the computation time for the learning process (learning duration in seconds) of the inverse learning for both batch and incremental LME. The same data was input to both methods in order to achieve a fair comparison. Each data point is the mean of 100 trials for a fixed degree of observability and a fixed number of trajectories in a demonstration $\mathscr{X}$. While the entire demonstration was given as input to the batch variant, the $\mathscr{X}_i$ for each session of I2RL had one trajectory composed of five state-action pairs. As sessions arbitrarily segment the trajectory of state-actions pairs, the sessions may not be

Table 2 Number of trajectories at most needed for ϵ_l convergence in the second patrolling domain ($K = 6, \gamma = 0.99$) with confidence $1 - \delta_l = 1 - (\delta + \delta_s) = 1 - (0.1 + 0.1) = 0.8$

ϵ_l (ϵ, ϵ_s)	$\lvert\mathscr{Y}_{1:i}\rvert$
0.125 (0.2, 0.05)	60
0.075 (0.1, 0.05)	**239**
0.05 (0.05, 0.05)	957
0.045 (0.04, 0.05)	1496

We use $\epsilon_s = 0.075$ for both 30% and 70% observability

Table 3 Confidence of convergence increases with more trajectories (from more sessions) with $\epsilon_l = 0.075$

	max
$\lvert\mathscr{Y}_{1:i}\rvert$	$(0, 1 - \delta_l)$
115	0
135	0.19
200	0.78
375	0.99

i.i.d. The incremental learning stops when there are no more trajectories remaining to be processed.

If the sessions are i.i.d., Theorem 2 allows us to derive an upper bound on the number of state-action pairs needed across all sessions to meet the given log likelihood error, which signals convergence. Table 2 shows this relation between the acceptable error ϵ_l, which is a function of ϵ and ϵ_s, and the number of trajectories for a 80% confidence level. In our simulations, we choose $\epsilon = 0.05$ and $\epsilon_s = 0.05$, which yields $\epsilon_l = \frac{\epsilon+\epsilon_s}{2} = 0.075$. Setting $\epsilon_s = 0.05$ yields the maximum number of MCMC samples required in each E-step as $N = -\frac{1}{2\epsilon_s^2} \ln \frac{\delta_s}{2K} = 957$. For a ϵ_l of 0.075 for our experiments, Table 2 shows that at most 239 trajectories would be needed. Table 3 shows that, for the chosen value of ϵ_l, the confidence of convergence increases with more sessions as we should expect.

In the single-patroller instance, LME I2RL shows a learning accuracy that remains close to that of batch LME with the accuracy increasing as the number of observed trajectories increase (Fig. 2). Importantly, LME I2RL processes the trajectories and achieves the eventual learning accuracy in significantly less time. Furthermore, it shows slow growth in its cumulative learning duration as we provide more trajectories.

Next, we report the LBA, ILE, and learning durations for the second domain instance involving two patrollers. As these experiments are simulations, we may vary the learner's observability, and Fig. 3a shows the results under a 30% degree of observability while Fig. 3b is for 70% degree of observability. To better understand the differentiations in performance, we introduce a third variant that implements each session as, $\zeta_i(MDP_{/R_E}, \mathscr{Y}_i, |\mathscr{Y}_{i:i-1}|, \hat{\phi}^{Z|Y,1:i-1}_{\theta^i})$. Notice that this incremental variant does not utilize the previous session's reward weights; instead, it initializes them randomly in each session. We label it as LME I2RL *with random weights*.

We empirically verify that convergence is indeed achieved within 239 sessions (each having one trajectory). As the size of demonstration increases, both batch and incremental variants exhibit a similar quality of learning although initially the incremental performs slightly worse. Due to the higher standard deviations exhibited by the incremental variant with random weights, we find the performances of the two incremental variants to be

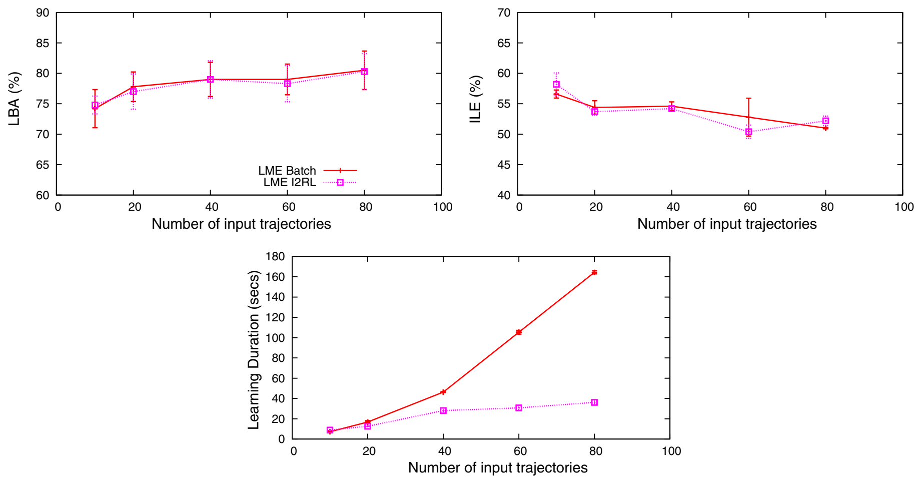
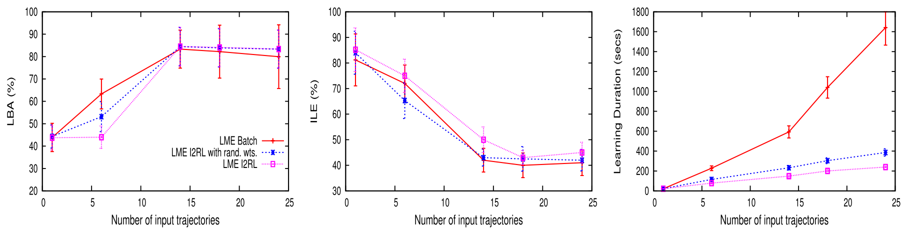


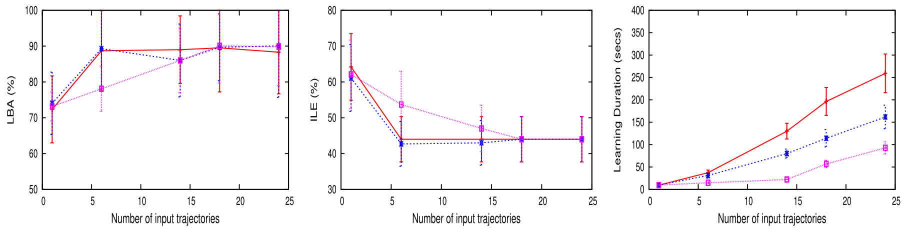
Fig. 2 Performances of batch and incremental LME in the first domain instance of Fig. 1a. All simulations were run on a Ubuntu 14 LTS system with an Intel i5 2.8 GHz CPU core and 8GB RAM. The error bars denote one standard deviation

similar despite slight differences in the means. More importantly, LME I2RL achieves these learning accuracies in significantly less time compared to the batch method, with the speed up ratio increasing to four as $|\mathscr{X}|$ grows. On the other hand, the batch method generally fails to converge in the total time taken by the incremental variant. Between the two degrees of observability, less observability exhibits a longer learning duration because of the need for increased inference that is time consuming. Notice that a random initialization of weights in each session, performed in LME I2RL with random weights, leads to higher learning durations as expected. The video of a simulation is available at https://youtu.be/B3wA6z111ws.

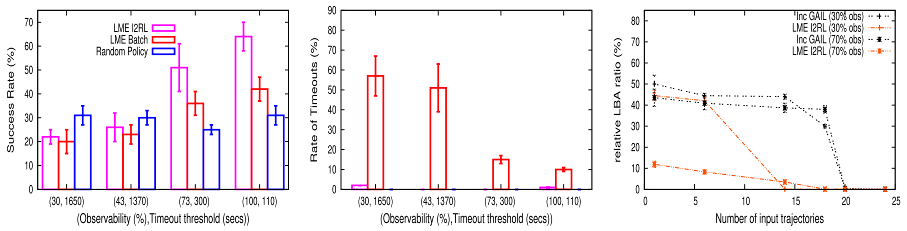
Is there a benefit due to the reduced learning time? Figure 3c shows the success rates of the learner when each of the three methods are utilized for IRL. LME I2RL begins to demonstrate comparatively better success rates under 30% observability, which further improves when the observability is at 70%, as we should expect. While the batch LME's success rate does not exceed 40%, the incremental variant succeeds in reaching the goal location undetected in about 65% of the runs under full observability (the last data point). A deeper analysis in order to understand these differences in success rates between batch and the incremental generalization of LME reveals that batch LME suffers from a large percentage of *timeouts* while incremental LME suffers from very few timeouts. A timeout occurs when IRL fails to converge to a reward estimate in a reasonable amount of time for each run. We set the threshold for a timeout as the total time taken for perception of trajectories, learning, and two rounds of patrolling averaged over many trials. This gives both batch IRL and I2RL at least two chances for penetrating the patrol. Notice that as the perception and learning times increase with the size of input data, so does the corresponding timeout threshold. The batch method shows a small 10% timeout rate for the full observability case which increases to more than a 50% timeout rate for low observability; whereas, the rate for incremental method stays below 4% throughout. LME with low observability requires more time due to the larger portion of the trajectory being hidden, which requires sampling a larger trajectory for computing the expectation. The longer learning durations



(a) Learned behavior accuracy, ILE, and learning duration under a 30% degree of observability.



(b) Learned behavior accuracy, ILE, and learning duration under a 70% degree of observability.



(c) Success rates and timeouts under 30%, 70%, and full observability. Performance when the learner employs a random policy is shown as well. This baseline method does not engage in IRL and picks a random set of reward weights for computing the patroller's policy. Rightmost chart shows the relative difference computed as (LBA for full observability - LBA under occlusion)/(LBA for full observability) for both 30% and 70% observability. We expect this ratio to reduce as the observability increases.

Fig. 3 Various metrics for comparing the performances of batch and incremental LME on the second instance of the patrolling domain (Fig. 1b)

of the batch and I2RL methods in comparison to the no-learning random policy leads to more time outs, which negatively impacts their success rates for the low observability settings. However, performance of the random policy is significantly worse than the I2RL methods for better observability. Of course, other factors play secondary roles in the success rate as well.

We compared the performance of LME I2RL with that of an online GAIL. We experimented with various simulation settings for GAIL and eventually settled on one that seemed most appropriate for our domain (500 iterations of TRPO with an adversary-batch-size of 1000, two hidden-layer [64×8] network for both generator and adversary, 5 epochs for adversary, and batch-size 150 for generator). We obtained a maximum LBA of 52% for the fully observable simulations (note that fully observable trajectories still may not yield all possible state-action pairs). As this absolute performance was rather low, we analyzed the relative impact of occlusion in our scenario on the performance of GAIL. The rightmost chart in Fig. 3c shows that while both LME I2RL and online GAIL demonstrate the same relative difference initially, GAIL requires significantly more trajectories before it

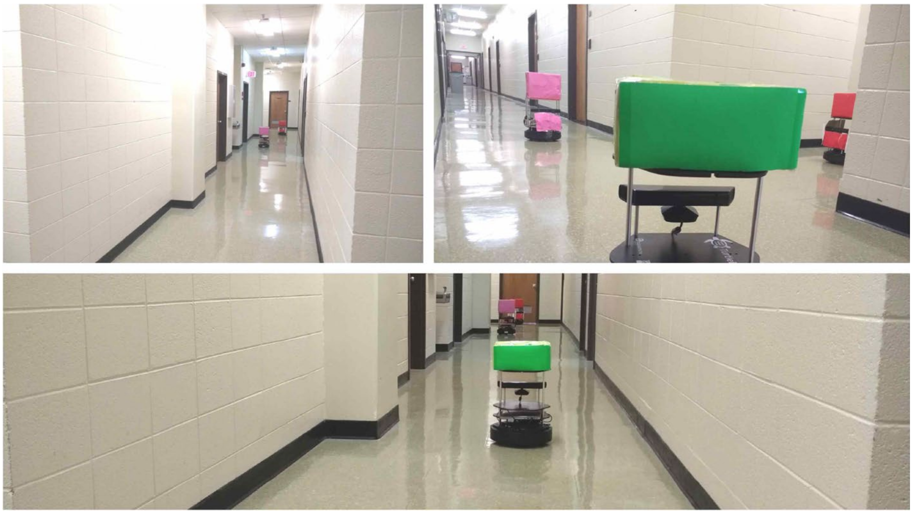


Fig. 4 **(top-left)** Patrollers (in pink and red) in the longer hallway of instance 2 of the domain. **(top-right)** Learner's (green) perspective as it observes the patrollers from its vantage point. **(bottom)** Learner penetrating the patrol to reach the goal undetected (first door on its right). The learner perceives the location and the orientation of each patroller by using the depth and the bounding box for the color blob detected via CMVision.

catches up with its full-observability performance, for both the 30% and 70% observability cases.[3] As such, online GAIL appears to be more impacted by occlusion than LME I2RL.

5.4 Performance evaluation on physical robots

Physical TurtleBots were then engaged in the perimeter patrol experiment on the *second* and *third* instances of the domain both of which exhibit less than 30% observability. The TurtleBots acting as both the patrollers and the learner are each equipped with a Kinect-360 RGB-D camera and an ASUS laptop. Differently colored boxes are placed on top of each as markers (see Fig. 4). The learner uses the ROS stacks for TurtleBot and and CMVision to perceive the centroid, the width, and the height of the colored boxes on the two patrollers to track them. Utilizing this data about the patrollers, the dimensions of the known floor map, and the learner's own position retrieved via Monte Carlo localization, the learner tracks the state (x,y, orientation) from its observations of each of the two patrollers. We utilize aggregated observations over a duration of 2 seconds to recognize the state. The aggregation involved taking the average of the x and y locations and the mode for the orientation. Having recognized the states in this way, the action of a patroller is inferred by using the states before and after its motion for 2 seconds. The computation of the timeout threshold is same as that for simulations.

For each data point in the physical experiments, we conducted 50 trials with the number of trajectories and the number of state-action pairs per trajectory being the

[3] As more trajectory data is provided to GAIL, the accuracy of the expert's estimated occupancy measure for the occluded state-action pairs improves. This helps GAIL in achieving its objective of minimizing the regularized cost.

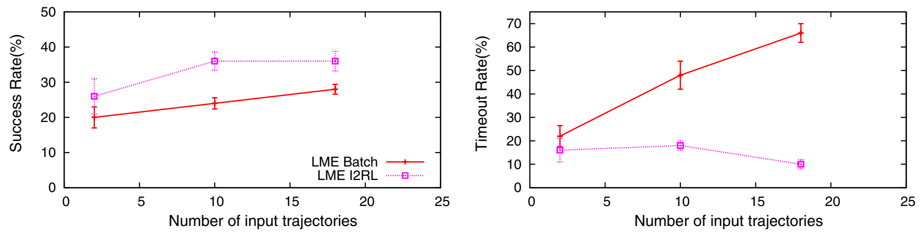


Fig. 5 The success and timeout rates achieved in the physical experiments for the second instance of the domain. The learner has less than 30% observability

same as those in simulations. As the degree of observability cannot be changed in our physical setup, we varied the number of input trajectories in order to observe the change in success rate and timeout rate. Figure 5 shows the results of a comparison between the performances of batch LME and LME I2RL deployed on the TurtleBot. We do not include GAIL in these experiments because of its poor performance in the simulations (as is evident from Fig. 3c) and the fact that the observability in these experiments is lower than 30%, which will clearly further degrade its performance. Despite the low observability, the success rate for LME I2RL is consistently higher than that for batch LME (reaching close to 40%), thereby showing consistency with the results in simulations. The timeout rate while higher than those in simulations remains much lower for LME I2RL compared to its batch counterpart.

We also ran physical robot experiments on the larger third instance of the patrolling domain whose map is shown in Fig. 1c. Snapshots of the location of the attacker, the patrollers, and the hallways navigated by the patrollers are shown in Fig. 6. A video of a run in this instance is available here: https://youtu.be/xTmXUI5P76g. Note that the theoretical convergence analysis performed previously in Sect. 5.3 for the second domain instance may also apply to the third instance because the number of MCMC samples in the E-step remain the same except that the number of features K is 5 for the third domain. Following the same sample complexity calculations as before, LME I2RL must converge to $\varepsilon_l = 0.075$ with probability $(1 - \delta_l) = 0.8$ in about 230 input trajectories. Indeed, it achieves convergence within this prescribed size of the input data.

Despite very low observability and a larger state space, I2RL has a success rate going up to 35% as shown in Fig. 7, which is 10% higher than that for the batch method. Interestingly, that success rate is reached by observing about 22 trajectories only, i.e. 110 state-action pairs. The timeout rate for I2RL stays below 10% while that of batch LME crosses 40%. Clearly, faster convergence helps the attacker to succeed more often. Observe that for the initial data points, despite both methods exhibiting a similar LBA, the success rate of I2RL is increasing but that of batch LME stays constant followed by a slight increase. This occurs in part because I2RL shows less timeouts for those data points and also learns patroller behaviors whose accuracy improves with more data.

To summarize, experiments conducted on multiple configurations of the patrolling domain in both simulation and on physical robots demonstrate the significant improvement in task performance brought about by online IRL. This is predominantly due to the incremental learning that leverages the learned outcomes from previous sessions while keeping the learning problem in each session small.

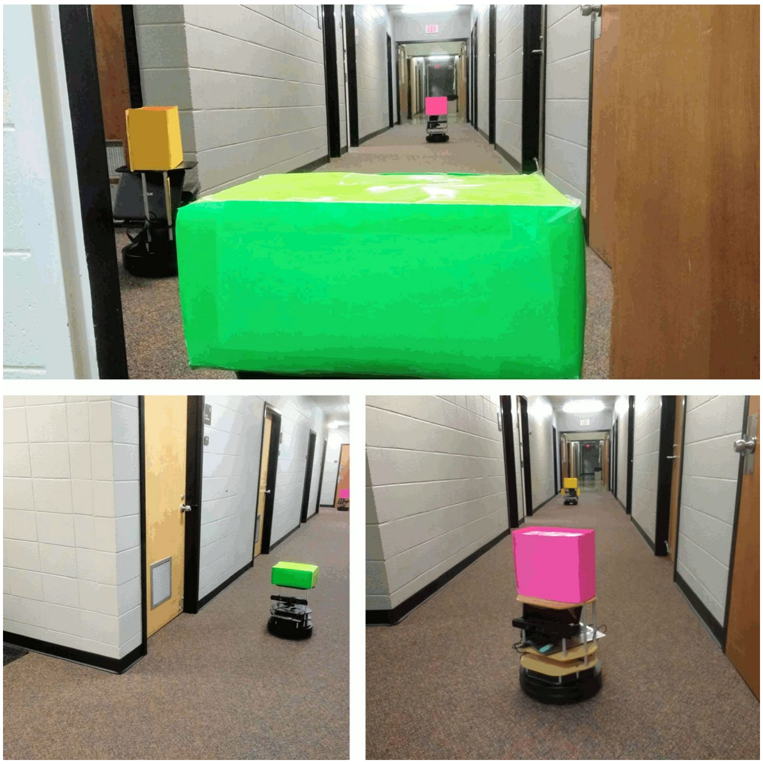


Fig. 6 **(top)** The learner (green) observes the two patroller (pink and yellow). **(bottom-right)** Patrollers navigating the hallway at the top of the map. **(bottom-left)** The learner penetrates the patrol moving through the hallway on the right side of the map (reaching the goal marked G2 in the grid)

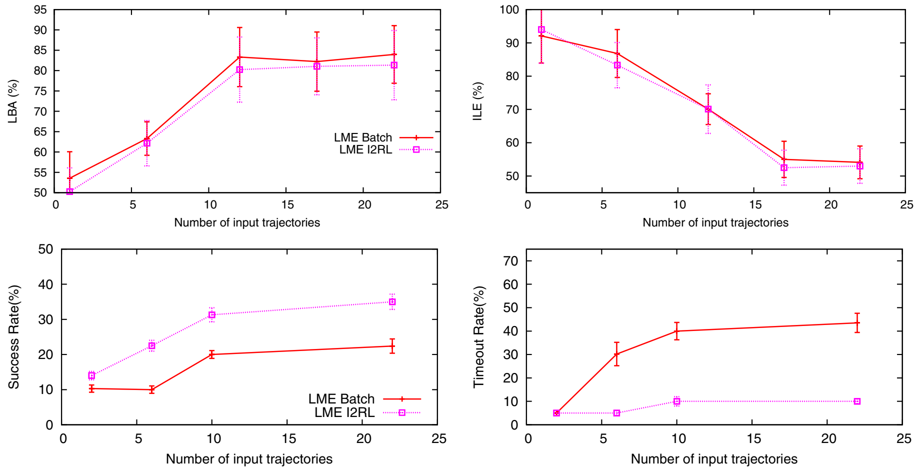


Fig. 7 Comparative performance of batch LME and LME I2RL in the third instance of the patrolling domain using physical robots. We assumed knowledge of the patrollers' true polices in order to obtain the LBA and ILE metrics but note that such information is typically unavailable to the learner in practice

6 Related work

As the field of online IRL is relatively new, the research literature is in the nascent stages of progress. An early method for online IRL [21] modifies Ng and Russell's linear program [26]. It takes as input a single trajectory (instead of the policy) and replaces the linear program with an incremental update of the reward function. In particular, for each new observation, the learner updates the reward weights every time the observed action differs from the predicted action of the expert. Apart from the algorithm for this method, the authors provide error bounds as well.

Activity forecasting is one of the newer applications of IRL. Researchers have focused on learning human behavior models from observations, and predicting latent goals [23]. A recent method called DARKO [31] performs online IRL for activity forecasting. By tracking short term location goals of a person wearing a camera, DARKO uses IRL to learn a reward function based on the types of locations and objects in the environment. Based on the inferred model, it predicts a subset of possible future goals from a given finite set of goals. DARKO builds the states, actions, and the transitions during execution. However, occlusion of some states and actions would lead to an inaccurate model thereby severely deteriorating its IRL performance. Indeed, its not immediately clear how the online model building can be generalized to account for occlusion. Herman et al. [19] present a solution to the problem of (online) learning socially acceptable navigation behavior. By using an input demonstration *labeled with acceptability*, this method incrementally adapts a learned reward function according to recent changes in the navigation behavior of the observed humans. As observed behavior can continuously change, there is no concept of a stopping criterion for the method. While the approach assumes that the trajectories are fully observed, the method is classic MaxEnt, which can be replaced with incremental LME to generalize this technique to occlusion.

There are some learning approaches in imitation learning that can be modified and adapted to the context of online IRL. One of these is our online version of GAIL [20], which we used as a baseline in our experiments. GAIL uses the generative adversarial network architecture for learning the policy from observations. It aims to minimize a regularized loss computed as a function of the difference between the estimated occupancy measure of the expert's policy and the occupancy measure for learned policy. The former occupancy measure is analogous to the distribution of data input to GAN and the latter occupancy measure is data distribution generated by a generator G_η. The batch algorithm of maximum margin planning (MMP) [30] is a method for imitation learning, evaluated on path planning domains. In that paper, Ratliff et al. also provide an extension of MMP to online MMP, and prove that the algorithm admits a sub-linear regret bound. A separate paper [29] presents an application of the technique to the problem of online structured prediction, but focused primarily on classification domains. Other relevant methods admit online learning given deliberate teaching from a human [3, 22], but we focus on the class of methods with *passive observations* by the learner without any intentional teaching on the part of the demonstrator.

7 Concluding remarks

This article contributes to the nascent problem of online IRL by offering a formal framework, I2RL, to help analyze the class of methods for online IRL. I2RL facilitates a conceptual comparison of various online IRL techniques, and facilitates establishing the theoretical properties of online methods. In particular, it provides a common ground for the definition of sessions, stopping criteria, and an evaluation metric among researchers interested in developing techniques for online IRL.

We presented a new method within the I2RL framework that generalizes recent advances in maximum entropy IRL to online settings. Casting this method to the context of I2RL allowed us to establish key theoretical properties of maximum entropy I2RL and LME I2RL under full and partial observability by ensuring the desired monotonic progress in learning toward convergence with a given confidence. With more training trajectories or better observability $|\mathscr{Y}_{1:i}|$, likelihood loss $LL(\theta_E|\mathscr{Y}_{1:i}) - LL(\theta^i|\mathscr{Y}_i)$ for I2RL gets smaller and the learned weights θ^i get closer to the true weights or their equivalent. The confidence of convergence $1 - \delta$ increases with more training (greater $|\mathscr{Y}_{1:i}|$), more room for error (higher ϵ and ϵ_s), and less features (lower K). For LME, one way to look at the theoretical results is that Lemma 2 uses ϵ and ϵ_s to bound the absolute value of the gradient $(\hat{\phi} - E_X[\phi])$ used in a likelihood-maximization process. And, bounding the gradient also restricts the difference between the value of a learned policy and the value of the expert's policy. Exploiting that information from Lemma 2, Theorem 2 bounds the probability of error in maximization. To complete the formal analysis of our online method, we introduced the notion of regret for I2RL and we proved that the average regret for LME I2RL approaches zero as the number of learning sessions increase. However, one limitation of our theoretical results is that the no-regret learning property of LME I2RL is limited to fully-observable settings only.

Our comprehensive experiments show that the new I2RL method is better than the state-of-the-art batch method in time-limited domains; it generally reproduces the batch method's accuracy but in significantly less time. In particular, we showed that given the practical constraints on computation time exhibited by an online IRL application, the new method is able to solve the problem with a higher success rate. This IRL generalization also suffers less from occlusion than a popular imitation learning method that directly learns the policy or behavior. We showed that LME I2RL continues to perform better than batch IRL on a larger state space and under lower observability. This offers evidence that the I2RL method is scalable to more complex domains.

While the method presented in this article has shown advances in the area of online IRL there remain multiple avenues for future research. One path is to focus on understanding how methods in the I2RL framework can learn without prior knowledge of the dynamics of experts. Another avenue is to explore the usage of LME I2RL in other applications. Notice that our empirical results are limited to the domain of learning the patrolling trajectories of multiple robots by observing them. Additional evaluations in other domains will help toward demonstrating further benefit and robustness of I2RL. For example, Rhinehart et. al.'s domain of activity forecasting. Yet another possibility for further research is to extend the proposed approach to the case of online learning of multiple reward functions.

Acknowledgements We thank Kenneth Bogert for insightful discussions, and the anonymous reviewers for helpful comments and suggestions. This work was supported in part by a research contract with the Toyota Research Institute of North America (TRI-NA), and by National Science Foundation grants IIS-1830421 (to PD) and IIS-1526813 (to BB).

Appendix

$\mathbb{X}$ is the space of all possible trajectories. The expected value of any feature $\phi_k \in [0,1], k \in \{1,2\ldots K\}$ for trajectory X is given by function $f_i : \mathbb{X} \to \mathbb{R}$ defined as $f_k(X) = \sum_{\langle s,a\rangle_t \in X} \gamma^t \phi_k(\langle s,a\rangle_t)$. Although a trajectory in a non-terminating MDP can be infinitely long, we derive range of f_k first for bounded-length trajectories and extend it later by applying infinity limit. Let T_{max} be the maximum length of any trajectory, $0 \le |X| \le T_{max}$.

Then,

$$\sum_{t=0}^{0} \gamma^t 0 \le \sum_{\langle s,a\rangle_t \in X} \gamma^t \phi_k(\langle s,a\rangle_t) \le \sum_{t=0}^{T_{max}} \gamma^t$$
$$0 \le f_k(X) \le (1-\gamma^{T_{max}})/(1-\gamma)$$

Applying limit $T_{max} \to \infty$ gives us

$$0 \le f_k(X) \le \frac{1}{1-\gamma} \tag{10}$$

Extending the definition to all k features, we introduce function $f : \mathbb{X} \to \mathbb{R}^k$ as $f(X) = \sum_{\langle s,a\rangle_t \in X} \gamma^t \phi(\langle s,a\rangle_t)$.

Note that learned feature expectations can be expressed in terms of f_k as

$$E_{\mathbb{X}}[\phi_k] \triangleq \sum_{X \in \mathbb{X}} Pr(X) f_k(X),\ k = 1\ldots K$$

The sessions for latent and full-observation MAXENT IRL updates estimated feature expectations of expert as follows.

$$\begin{aligned}\hat{\phi}^{Z|Y,1:i}_{\theta^i,k} &\triangleq \frac{1}{|\mathscr{Y}_{1:i}|} \sum_{Y \in \mathscr{Y}_{1:i}} \sum_{Z \in \mathbb{Z}} Pr(Z|Y;\theta) \\ &\sum_{\langle s,a\rangle_t \in Y \cup Z} \gamma^t \phi_k(\langle s,a\rangle_t) \\ &= \frac{|\mathscr{Y}_{1:i-1}|}{|\mathscr{Y}_{1:i-1}| + |\mathscr{Y}_i|} \hat{\phi}^{Z|Y,1:i-1}_{\theta^{i-1},k} + \frac{|\mathscr{Y}_i|}{|\mathscr{Y}_{1:i-1}| + |\mathscr{Y}_i|} \hat{\phi}^{Z|Y,i}_{\theta^i,k}\end{aligned} \tag{11}$$

$$\begin{aligned}\hat{\phi}^{1:i}_k &\triangleq \frac{1}{|\mathscr{Y}_{1:i}|} \sum_{Y \in \mathscr{Y}_{1:i}} \sum_{Z \in \mathbb{Z}} Pr(Z|Y;\theta) \sum_{\langle s,a\rangle_t \in Y \cup Z} \gamma^t \phi_k(\langle s,a\rangle_t) \\ &= \frac{|\mathscr{Y}_{1:i-1}|}{|\mathscr{Y}_{1:i-1}| + |\mathscr{Y}_i|} \hat{\phi}^{1:i-1}_k + \frac{|\mathscr{Y}_i|}{|\mathscr{Y}_{1:i-1}| + |\mathscr{Y}_i|} \hat{\phi}^{i}_k\end{aligned} \tag{12}$$

From definitions of feature-expectations and Eq. 10, $E_{\mathbb{X}}[\phi_k], \hat{\phi}^{1:i}_k, \hat{\phi}^{Z|Y,1:i}_{\theta^i,k} \in \left[0, \frac{1}{(1-\gamma)}\right]$.

We give below the proofs of the theorems and lemmas stated in the main exposition of the article.

Proof of Theorem 1 We use the notation:

$$E_{\mathbb{X}}[\phi_k] \triangleq \sum_{X \in \mathbb{X}} Pr(X) \sum_{\langle s,a\rangle \in X} \phi_k(s,a),\ k = 1\ldots K$$

By allowing a relaxation in the constraints of maximum entropy estimation problem, [17] derived sample complexity bounds for the problem.

$$\begin{aligned}
&\max_{\Delta} \left(-\textstyle\sum_{X\in\mathbb{X}} Pr(X)\; log\; Pr(X)\right)\\
&\text{subject to} \quad \textstyle\sum_{X\in\mathbb{X}} Pr(X) = 1\\
&\left|E_{\mathbb{X}}[\phi_k] - \hat{\phi}_k^{1:i}\right| \le \beta_k^{full} \qquad \forall k \in \{1\ldots K\}
\end{aligned} \tag{13}$$

Here $\beta^{full} \in \mathbb{R}^K$ is a vector of upper bounds on the differences between $E_{\mathbb{X}}[\phi_k]$ and $\hat{\phi}_k^{1:i}$.

Following proofs by Dudik et al., relaxed constraints maximum entropy IRL problem is same as $\min_\theta(-\sum_{X\in\mathscr{X}_{1:i}} \tilde{Pr}(X)\;\; \log\;\; Pr(X|\theta) + \sum_k \;\beta_k^{full}|\theta_k|) \; = \min_\theta(-LL(\theta|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\phi}^{1:i-1}, \theta^{i-1}) + \sum_k \beta_k^{full}|\theta_k|) = \min_\theta NLL_{\beta^{full}}(\theta|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\phi}^{1:i-1}, \theta^{i-1})$ (say).

The proof here is partially inspired from Corollary 1 in [17]. Let $\beta_k^{full} = \beta_c^{full} = \varepsilon/(1-\gamma)$ for all $k \in \{1\ldots K\}$, where β_c^{full} is constant because ε is fixed input. For normalized exponentiated gradient descent used here for computing maximum, $\sum_1^K |\theta_k| = 1$. Then, $NLL_{\beta^{full}}(\theta|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\phi}^{1:i-1}, \theta^{i-1}) = (-LL(\theta|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\phi}^{1:i-1}, \theta^{i-1}) + \beta_c^{full}\sum_1^k|\theta_k|) = (-LL(\theta|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\phi}^{1:i-1}, \theta^{i-1}) + \beta_c^{full})$. Assume that θ^i minimizes $NLL_{\beta^{full}}(\theta|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\phi}^{1:i-1}, \theta^{i-1})$, a solution maximizing $LL(\theta|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\phi}^{1:i-1}, \theta^{i-1})$.

Since $E_{\mathbb{X}}[\phi_k] \in \left[0, \frac{1}{(1-\gamma)}\right]$, we get $(1-\gamma)E_{\mathbb{X}}[\phi_k] \in [0,1]$. We multiply the relaxed constraint with $(1-\gamma)$ and define the negation of constraint as following event: $A : \left|(1-\gamma)E_{\mathbb{X}}[\phi_k] - (1-\gamma)\hat{\phi}_k^{1:i}\right| > (1-\gamma)\beta_c^{full} = \varepsilon \quad$ for some $k \in \{1\ldots K\}$. A can be decomposed into following feature specific events

$$A_k : (1-\gamma)\left|E_{\mathbb{X}}[\phi_k] - \hat{\phi}_k^{1:i}\right| > \varepsilon,$$

where $k \in \{1, 2\ldots K\}$. We divide this constraint on absolute value further in two signed events:

$$\begin{aligned}
(A_k)_1 &: (1-\gamma)E_{\mathbb{X}}[\phi_k] - (1-\gamma)\hat{\phi}_k^{1:i} > \varepsilon\\
(A_k)_2 &: -(1-\gamma)E_{\mathbb{X}}[\phi_k] + (1-\gamma)\hat{\phi}_k^{1:i} > \varepsilon
\end{aligned}$$

Then event A is same as logical disjunction $(A_1)_1 \vee (A_1)_2 \vee (A_2)_1 \ldots$.

Applying Hoeffding's inequality, the upper bound of probability of each signed event is given by: $P((A_k)_1) \le \exp(-2\varepsilon^2|\mathscr{X}_{1:i}|) = \frac{\delta}{2K}(say), P((A_k)_2) \le \frac{\delta}{2K}$.

Applying aforesaid bounds to events for each of the K features, we get $2K$ events with exactly same upper bound $\frac{\delta}{2K}$ on their respective probabilities. We use Fretchet's inequality to derive an upper bound for the disjunction:

$$P(A) = P((A_1)_1 \vee (A_1)_2 \vee (A_2)_1 \ldots) \le \min(1, P((A_1)_1) + P((A_1)_2) + P((A_2)_1)\ldots)$$

As each of the probabilities in RHS are bounded from above by $\frac{\delta}{2K}$, their sum is bounded as:

$$P(A) \le \min\left(1, \sum_1^{2K} \frac{\delta}{2K}\right) = \min(1, \delta)$$

Reverting to the negation of A, the probability that $\left|(1-\gamma)E_{\mathbb{X}}[\phi_k] - (1-\gamma)\hat{\phi}_k^{1:i}\right| \le \varepsilon \quad \forall k \in \{1\ldots K\}$ is at least $1 - \min(1,\delta) = \max(0, 1-\delta)$.

To keep reward value bounded, IRL assumes $||\boldsymbol{\theta}^*||_1 \leq 1$ for all $\boldsymbol{\theta}^*$. Using the assumption and Theorem 1 in [17], we get error bound:

For every $\boldsymbol{\theta}^* \in [0,1]^K$, $NLL_{\beta full}(\boldsymbol{\theta}^i|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\boldsymbol{\phi}}^{1:i-1}, \boldsymbol{\theta}^{i-1}) \; - NLL_{\beta full}(\boldsymbol{\theta}^*|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\boldsymbol{\phi}}^{1:i-1}, \boldsymbol{\theta}^{i-1}) \leq 2\sum_1^K \beta_c^{full} = 2K\ \beta_c^{full} = \frac{2K\epsilon}{(1-\gamma)}$ with probability at least $\max(0, 1-\delta)$, where $\delta = \ 2K \exp\big(-2\,\epsilon^2|\mathscr{X}_{1:i}|\big)$.

We modify the bound in the form of positive log-likelihood of expert's policy, by using relation $NLL_{\beta full}(\boldsymbol{\theta}^*|\mathscr{X}_{1:i}) = (-LL(\boldsymbol{\theta}^*|\mathscr{X}_{1:i}) + \sum_1^K \beta_k^{full}|\theta_k|)$ and $\boldsymbol{\theta}^* = \boldsymbol{\theta}_E$.

Then, with $\mathscr{X}_{1:i}$ as input, with probability at least $\max(0, 1-\delta)$,

$$\begin{aligned}&NLL_{\beta full}(\boldsymbol{\theta}^i|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\boldsymbol{\phi}}^{1:i-1}, \boldsymbol{\theta}^{i-1}) - NLL_{\beta full}(\boldsymbol{\theta}_E|\mathscr{X}_{1:i})\\ &\quad = LL(\boldsymbol{\theta}_E|\mathscr{X}_{1:i}) - LL(\boldsymbol{\theta}^i|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\boldsymbol{\phi}}^{1:i-1}, \boldsymbol{\theta}^{i-1}) \leq \frac{2K\epsilon}{(1-\gamma)}.\end{aligned}$$

□

Proof of Lemma 1 Log-likelihood of demonstrated behavior can be split as

$$\begin{aligned}LL(\boldsymbol{\theta}^i|\mathscr{Y}_{1:i}) &= LL(\boldsymbol{\theta}^i|\mathscr{Y}_i, |\mathscr{Y}_{i-1}|, \hat{\boldsymbol{\phi}}_{\boldsymbol{\theta}^{i-1}}^{Z|Y,1:i-1}, \boldsymbol{\theta}^{i-1})\\ &= \sum_{Y\in\mathscr{Y}_{1:i}} \tilde{Pr}(Y)\ log\ Pr(Y;\boldsymbol{\theta})\\ &= \sum_{Y\in\mathscr{Y}_{1:i}} \tilde{Pr}(Y) \sum_{Z\in\mathbb{Z}} Pr(Z|Y;\boldsymbol{\theta}^i)\ log\ Pr(Y,Z;\boldsymbol{\theta})\\ &\quad + \left(-\sum_{Y\in\mathscr{Y}_{1:i}} \tilde{Pr}(Y) \sum_{Z\in\mathbb{Z}} Pr(Z|Y;\boldsymbol{\theta}^i)\ log\ Pr(Z|Y;\boldsymbol{\theta})\right)\\ &= Q(\mathscr{Y}_i, |\mathscr{Y}_{i-1}|, \hat{\boldsymbol{\phi}}_{\boldsymbol{\theta}^{i-1}}^{Z|Y,1:i-1}, \boldsymbol{\theta}, \boldsymbol{\theta}^{i-1}) + C(\mathscr{Y}_{1:i}, \boldsymbol{\theta}, \boldsymbol{\theta}^i)\end{aligned}$$

Here $\tilde{Pr}$ is distribution of trajectories in observed training data ($\sum\limits_{X\in\mathscr{X}} \tilde{Pr}(X)[\cdot]$ and $\frac{1}{|\mathscr{X}|}\sum\limits_{X\in\mathscr{X}}[\cdot]$ can be used interchangeably). EM method maximizes the log-likelihood by maximizing only Q value over $\boldsymbol{\theta}$; and $\boldsymbol{\theta} = \boldsymbol{\theta}^i$ maximizes $Q(\mathscr{Y}_i, |\mathscr{Y}_{i-1}|, \hat{\boldsymbol{\phi}}_{\boldsymbol{\theta}^{i-1}}^{Z|Y,1:i-1}, \boldsymbol{\theta}, \boldsymbol{\theta}^{i-1})$ ([36]). After all the EM iterations for current session i, the final Q value is $Q(\mathscr{Y}_i, |\mathscr{Y}_{i-1}|, \hat{\boldsymbol{\phi}}_{\boldsymbol{\theta}^{i-1}}^{Z|Y,1:i-1}, \boldsymbol{\theta}^i, \boldsymbol{\theta}^i)$. Therefore, the difference in the likelihoods achieved by weights learned in consecutive sessions can be expressed as a difference in Q values. Note that LME IRL learns reward weights by inferring the maximum entropy distribution $Pr(X;\boldsymbol{\theta}) = \frac{\exp(\sum_k \theta_k f_k(X))}{\Omega_{\boldsymbol{\theta}}^X}$, where $\Omega_{\boldsymbol{\theta}}^X = \sum_{X\in\mathbb{X}} \exp(\sum_k \theta_k f_k(X))$ and $X = (Y, Z)$. Expand Q value as $Q(\mathscr{Y}_i, |\mathscr{Y}_{i-1}|, \hat{\boldsymbol{\phi}}_{\boldsymbol{\theta}^{i-1}}^{Z|Y,1:i-1}, \boldsymbol{\theta}^i, \boldsymbol{\theta}^i) = \sum\limits_{Y\in\mathscr{Y}_{1:i}} \tilde{Pr}(Y) \sum\limits_{Z\in\mathbb{Z}} Pr(Z|\ Y;\boldsymbol{\theta}^i)\ \log\left(\frac{\exp(\sum_k \theta_k^i f_k((Y,Z)))}{\Omega_{\boldsymbol{\theta}^i}^{(Y,Z)}}\right) = \sum_k \theta_k^i \cdot \sum\limits_{Y\in\mathscr{Y}_{1:i}} \tilde{Pr}(Y) \sum\limits_{Z\in\mathbb{Z}} Pr(Z|\ Y;\boldsymbol{\theta}^i) f_k((Y,Z)) - \log\ \Omega_{\boldsymbol{\theta}^i}^{(Y,Z)} = \sum_k \theta_k^i \cdot \hat{\boldsymbol{\phi}}_{\boldsymbol{\theta}^i,k}^{Z|Y,1:i} - \log\ \Omega_{\boldsymbol{\theta}^i}^{(Y,Z)}$.

Therefore the improvement in log likelihood over session i is

$$
\begin{aligned}
&LL(\theta^i|\mathscr{Y}_i, |\mathscr{Y}_{i-1}|, \hat{\phi}^{Z|Y,1:i-1}_{\theta^{i-1}}, \theta^{i-1}) - LL(\theta^{i-1}|\mathscr{Y}_{i-1}, |\\
&\mathscr{Y}_{i-2}|, \hat{\phi}^{Z|Y,1:i-2}_{\theta^{i-2}}, \theta^{i-2})\\
&\quad = Q(\mathscr{Y}_i, |\mathscr{Y}_{i-1}|, \hat{\phi}^{Z|Y,1:i-1}_{\theta^{i-1}}, \theta^i, \theta^i) - Q(\mathscr{Y}_{i-1}, |\\
&\mathscr{Y}_{i-2}|, \hat{\phi}^{Z|Y,1:i-2}_{\theta^{i-2}}, \theta^{i-1}, \theta^{i-1})\\
&\quad = \sum_k \theta^i_k \hat{\phi}^{Z|Y,1:i}_{\theta^i,k} - \log\ \Omega^{(Y,Z)}_{\theta^i} - \sum_k \theta^{i-1}_k \hat{\phi}^{Z|Y,1:i-1}_{\theta^{i-1},k} +\\
&\log\ \Omega^{(Y,Z)}_{\theta^{i-1}}\\
&\quad = \log \frac{\Omega^{(Y,Z)}_{\theta^{i-1}}}{\Omega^{(Y,Z)}_{\theta^i}} + \sum_k \left(\theta^i_k \frac{|\mathscr{Y}_{1:i-1}|}{|\mathscr{Y}_i| + |\mathscr{Y}_{1:i-1}|} - \theta^{i-1}_k \right) \hat{\phi}^{Z|Y,1:i-1}_{\theta^{i-1},k}\\
&\qquad + \sum_k \left(\theta^i_k \frac{1}{|\mathscr{Y}_i| + |\mathscr{Y}_{1:i-1}|} \hat{\phi}^{Z|Y,i}_{\theta^i,k} \right)\\
&\qquad\quad \text{(substitute } \hat{\phi}^{Z|Y,1:i}_{\theta^i,k} \text{ using Eq.8 and simplifying)}
\end{aligned}
$$

The final expression is minimized only for $\theta^i = \theta^{i-1}$ when $|\mathscr{Y}_{1:i-1}| \gg |\mathscr{Y}_i|$, i.e., when a significant amount of training data has been accumulated. The expression is also concave in parameter θ^i. Therefore, $LL(\theta^i|\mathscr{Y}_i, |\mathscr{Y}_{i-1}|, \hat{\phi}^{Z|Y,1:i-1}_{\theta^{i-1}}, \theta^{i-1}) - LL(\theta^{i-1}|\mathscr{Y}_{i-1}, |\mathscr{Y}_{i-2}|, \hat{\phi}^{Z|Y,1:i-2}_{\theta^{i-2}}, \theta^{i-2}) \geq 0$ for consecutive sessions thereafter. Hence, the LME I2RL is proved to converge over sequence of sessions, yielding a feasible log-linear solution to latent-MAXENT and corresponding weights solving IRL. □

Proof of Lemma 2 We define the event $A_k : (1-\gamma)\left|E_{\mathbb{X}}[\phi_k] - \hat{\phi}^{1:i}_k\right| > \epsilon, k \in \{1, 2 \ldots K\}$.

Applying Hoeffding's inequality for A_k, we get $P(A_k) \leq 2\exp(-2\epsilon^2|\mathscr{X}_{1:i}|) \leq \frac{\delta}{K}$ for any $k \in \{1, 2 \ldots K\}$, and for the same ϵ, δ as in Theorem 1. Similarly, for partial observation, given ϵ_s as the bound on the error in sampling based approximation of $\hat{\phi}^{1:i}_l$ as $\hat{\phi}^{Z|Y,1:i}_{\theta^i,l}$, and n_s samples, let us define the event

$$B_l : (1-\gamma)\left|\hat{\phi}^{1:i}_l - \hat{\phi}^{Z|Y,1:i}_{\theta^i,l}\right| > \epsilon_s, l \in \{1, 2 \ldots K\}.$$

Similar to procedure for $P(A_k)$, applying Hoeffding bound gives us $P(B_l) < \frac{\delta_s}{K}, \delta_s = 2K\exp(-2(\epsilon_s)^2 n_s)$.

Applying Fretchets inequality over both sets A and B of events gives us:

$$P\big((\cup_k A_k) \vee (\cup_l B_l)\big) < \min\left(1, \sum_{k=1}^{K} \frac{\delta}{K} + \sum_{l=1}^{K} \frac{\delta_s}{K}\right) = \min(1, \delta + \delta_s).$$

That is, $P\big(\exists k, l, s.t. A_k \vee B_l\big) < \min(1, \delta + \delta_s)$. Taking complement, $P\Big(\forall k, l, \overline{A}_k \wedge \overline{B}_l\Big) \geq \max(0, 1 - \delta - \delta_s)$. But $\forall k, l, \overline{A}_k \wedge \overline{B}_l$ implies that $\forall k$:

$$(1-\gamma)(\left|E_{\mathbb{X}}[\phi_k] - \hat{\phi}^{1:i}_k\right| + \left|\hat{\phi}^{1:i}_k - \hat{\phi}^{Z|Y,1:i}_{\theta^i,k}\right|) \leq \epsilon + \epsilon_s$$

Calling $(\epsilon + \epsilon_s) = 2\epsilon_l$, and $(\delta + \delta_s) = \delta_l$ we get

$$P\Big(\forall k, (1-\gamma)\Big(\Big|E_{\mathbb{X}}[\phi_k] - \hat{\phi}_k^{1:i}\Big| + \Big|\hat{\phi}_k^{1:i} - \hat{\phi}_{\theta^i,k}^{Z|Y,1:i}\Big|\Big) \\ \leq 2\varepsilon_l\Big) \geq \max(0, 1-\delta_l).$$

Using inequality $\Big|E_{\mathbb{X}}[\phi_k] - \hat{\phi}_{\theta^i,k}^{Z|Y,1:i}\Big| \leq \big|E_{\mathbb{X}}[\phi_k] - \hat{\phi}_k^{1:i}\big| + \big|\hat{\phi}_k^{1:i} - \hat{\phi}_{\theta^i,k}^{Z|Y,1:i}\big|$, we get:

$$P\Bigg(\forall k, (1-\gamma)\Big(\Big|E_{\mathbb{X}}[\phi_k] - \hat{\phi}_{\theta^i,k}^{Z|Y,1:i}\Big|\Big) \leq 2\varepsilon_l\Bigg) \geq \max(0, 1-\delta_l).$$

□

Proof of Theorem 2 Latent maximum entropy IRL problem is equivalent to $\max_\theta \sum_{Y\in\mathscr{Y}_{1:i}} \tilde{Pr}(Y) \log Pr(Y|\theta)$ (Sect. 3.3, [12]) or $\max_\theta LL(\theta^i|\mathscr{Y}_i, |\mathscr{Y}_{i-1}|, \hat{\phi}_{\theta^{i-1}}^{Z|Y,1:i-1}, \theta^{i-1})$.

Relaxed constraint latent maximum entropy IRL is:

$$\begin{aligned} &\max_{\Delta} \left(-\textstyle\sum_{X\in\mathbb{X}} Pr(X)\ log\ Pr(X)\right) \\ &\text{subject to} \quad \textstyle\sum_{X\in\mathbb{X}} Pr(X) = 1 \\ &\Big|E_{\mathbb{X}}[\phi_k] - \hat{\phi}_{\theta^i,k}^{Z|Y,1:i}\Big| \leq \beta_k \qquad \forall k \in \{1 \dots K\} \end{aligned} \tag{14}$$

Here $\beta \in \mathbb{R}^K$ is an estimate of vector of upper bounds on the differences between $E_{\mathbb{X}}[\phi_k]$ and $\hat{\phi}_{\theta^i,k}^{Z|Y,1:i}$.

The form of relaxed latent maximum entropy problem and the likelihood for that problem is no different than those for relaxed maximum entropy. Starting from results in Lemma 2, assuming $\beta_k = \beta_c = 2\varepsilon_l/(1-\gamma)$ for all $k \in \{1 \dots K\}$ and using steps similar to the proof of Theorem 1, we get $LL(\theta_E|\mathscr{Y}_{1:i}) - LL(\theta^i|\mathscr{Y}_i, |\mathscr{Y}_{i-1}|, \hat{\phi}_{\theta^{i-1}}^{Z|Y,1:i-1}, \theta^{i-1}) \leq \frac{4K\varepsilon_l}{(1-\gamma)}$ with probbility at least $\max(0, 1-\delta_l)$. □

Proof of Theorem 3 The log-loss after ith session is $LL(\theta_E|\mathscr{X}_{1:i}) - LL(\theta^i|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\phi}^{1:i-1}, \theta^{i-1})$. Let the regret after $i = T$ be $\frac{1}{T}\sum_{i=1}^T LL(\theta_E|\mathscr{X}_{1:i}) - \frac{1}{T}\sum_{i=1}^T LL(\theta^i|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\phi}^{1:i-1}, \theta^{i-1})$.

According to Theorem 1, $LL(\theta_E|\mathscr{X}_{1:i}) - LL(\theta^i|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\phi}^{1:i-1}, \theta^{i-1}) \leq \beta \cdot \varepsilon_l$ with probbility at least $\max(0, 1-\delta)$, where $\beta = \frac{2K}{(1-\gamma)}$. As ε is user specified, let $\varepsilon = \frac{c}{i}$. Then, the inequality becomes $LL(\theta_E|\mathscr{X}_{1:i}) - LL(\theta^i|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\phi}^{1:i-1}, \theta^{i-1}) \leq \beta\frac{c}{i}$. Summing the result over $i \in \{1, 2, \dots T\}$ and dividing by T, we get

$$\frac{1}{T}\sum_{i=1}^T LL(\theta_E|\mathscr{X}_{1:i}) - \frac{1}{T}\sum_{i=1}^T LL(\theta^i|\mathscr{X}_i, |\mathscr{X}_{i-1}|, \hat{\phi}^{1:i-1}, \theta^{i-1}) \leq \beta\frac{1}{T}\sum_{i=1}^T \frac{c}{i}$$

The RHS above is bounded as $\beta\frac{1}{T}\sum_{i=1}^T \frac{c}{i} \leq \beta\frac{1}{T}c\log T = \beta c\frac{\log T}{T}$. As $T \to \infty$, $\beta c\frac{\log T}{T} \to 0$. Therefore, as sessions progress, regret is guaranteed to vanish. □

References

1. Abbeel, P., & Ng, A.Y. (2004). Apprenticeship learning via inverse reinforcement learning. In Twenty-first international conference on machine learning (ICML), pp. 1–8.
2. Aghasadeghi, N., & Bretl, T. (2011). Maximum entropy inverse reinforcement learning in continuous state spaces with path integrals. In: 2011 IEEE/RSJ International conference on intelligent robots and systems, pp. 1561–1566.
3. Amin, K., Jiang, N., & Singh, S. (2017). Repeated inverse reinforcement learning. In Advances in neural information processing systems, pp. 1815–1824.
4. Argall, B. D., Chernova, S., Veloso, M., & Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, *57*(5), 469–483.
5. Arora, S., & Doshi, P. (2018). A survey of inverse reinforcement learning: Challenges, methods and progress. CoRR arXiv:1806.06877
6. Arora, S., Doshi, P., & Banerjee, B. (2019). Online inverse reinforcement learning under occlusion. In: Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, pp. 1170–1178. International Foundation for Autonomous Agents and Multiagent Systems
7. Auer, P., Cesa-Bianchi, N., Freund, Y., & Schapire, R.E. (2000). Gambling in a rigged casino: The adversarial multi-armed bandit problem. Electronic Colloquium on Computational Complexity (ECCC) **7**(68).
8. Babes-Vroman, M., Marivate, V., Subramanian, K., & Littman, M. (2011). Apprenticeship learning about multiple intentions. In 28th International conference on machine learning (ICML), pp. 897–904.
9. Bogert, K., & Doshi, P. (2014). 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, pp. 173–180.
10. Bogert, K., & Doshi, P. (2015). Toward estimating others' transition models under occlusion for multi-robot irl. In 24th International joint conference on artificial intelligence (IJCAI), pp. 1867–1873.
11. Bogert, K., & Doshi, P. (2017). Scaling expectation-maximization for inverse reinforcement learning to multiple robots under occlusion. In Proceedings of the 16th conference on autonomous agents and multiagent systems, AAMAS '17, pp. 522–529.
12. Bogert, K., Lin, J.F.S., Doshi, P., & Kulic, D. (2016). Expectation-maximization for inverse reinforcement learning with hidden data. In 2016 International conference on autonomous agents and multiagent systems, pp. 1034–1042.
13. Boularias, A., Kober, J., & Peters, J. (2011). 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, pp. 182–189
14. Boularias, A., Krömer, O., & Peters, J. (2012). Structured apprenticeship learning. *European Conference on Machine Learning and Knowledge Discovery in Databases, Part*, *II*, 227–242.
15. Choi, J., & Kim, K. E. (2011). Inverse reinforcement learning in partially observable environments. *J. Mach. Learn. Res.*, *12*, 691–730.
16. Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B (Methodological)*, *39*, 1–38.
17. Dudík, M., Phillips, S. J., & Schapire, R. E. (2004). Performance guarantees for regularized maximum entropy density estimation. In J. Shawe-Taylor & Y. Singer (Eds.), *Learning Theory* (pp. 472–486). Berlin Heidelberg: Springer.
18. Gerkey, B., Vaughan, R.T., & Howard, A. (2003). The player/stage project: Tools for multi-robot and distributed sensor systems. In Proceedings of the 11th international conference on advanced robotics, vol. 1.
19. Herman, M., Fischer, V., Gindele, T., & Burgard, W. (2015). Inverse reinforcement learning of behavioral models for online-adapting navigation strategies. In 2015 IEEE international conference on robotics and automation (ICRA), pp. 3215–3222. IEEE.
20. Ho, J., & Ermon, S. (2016). Generative adversarial imitation learning. *Advances in Neural Information Processing Systems (NIPS)*, *29*, 4565–4573.
21. Jun Jin, Z., Qian, H., Yi Chen, S., & Liang Zhu, M. (2010). Convergence analysis of an incremental approach to online inverse reinforcement learning. *Journal of Zhejiang University-Science C*, *12*(1), 17–24.
22. Kamalaruban, P., Devidze, R., Cevher, V., & Singla, A. (2019). Interactive teaching algorithms for inverse reinforcement learning. arXiv preprint arXiv:1905.11867.

23. Kitani, K.M., Ziebart, B.D., Bagnell, J.A., & Hebert, M. (2012). Activity forecasting. In 12th European conference on computer vision - Volume Part IV, pp. 201–214.
24. Kivinen, J., & Warmuth, M. K. (1997). Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, *132*(1), 1–63.
25. Levine, S., Popović, Z., & Koltun, V. (2010). Feature construction for inverse reinforcement learning. In Proceedings of the 23rd international conference on neural information processing systems, NIPS'10, pp. 1342–1350. Curran Associates Inc., USA
26. Ng, A., & Russell, S. (2000). Algorithms for inverse reinforcement learning. In Seventeenth international conference on machine learning, pp. 663–670.
27. Osa, T., Pajarinen, J., Neumann, G., Bagnell, J.A., Abbeel, P., & Peters, J. (2018). An algorithmic perspective on imitation learning. Foundations and Trends® in Robotics **7**(2), 1–179.
28. Ramachandran, D., & Amir, E. (2007). Bayesian inverse reinforcement learning. In 20th international joint conference on artifical intelligence (IJCAI), pp. 2586–2591.
29. Ratliff, N., Bagnell, J., & Zinkevich, M. (2007). (online) subgradient methods for structured prediction. *Journal of Machine Learning Research - Proceedings Track*, *2*, 380–387.
30. Ratliff, N.D., Bagnell, J.A., & Zinkevich, M.A. (2006). Maximum margin planning. In 23rd international conference on machine learning, pp. 729–736.
31. Rhinehart, N., & Kitani, K.M. (2017). First-person activity forecasting with online inverse reinforcement learning. In International conference on computer vision (ICCV).
32. Russell, S. (1998). Learning agents for uncertain environments (extended abstract). In Eleventh annual conference on computational learning theory, pp. 101–103.
33. Steinhardt, J., & Liang, P. (2014). Adaptivity and optimism: An improved exponentiated gradient algorithm. In 31st International conference on machine learning, pp. 1593–1601.
34. Trivedi, M., & Doshi, P. (2018). Inverse learning of robot behavior for collaborative planning. In 2018 IEEE/RSJ international conference on intelligent robots and systems (IROS), pp. 1–9.
35. Wang, S., Rosenfeld, R., Zhao, Y., & Schuurmans, D. (2002). The Latent Maximum Entropy Principle. In IEEE international symposium on information theory, pp. 131–131.
36. Wang, S., & Schuurmans Yunxin Zhao, D. (2012). The Latent Maximum Entropy Principle. ACM Transactions on Knowledge Discovery from Data **6**(8).
37. Wulfmeier, M., & Posner, I. (2015). Maximum Entropy Deep Inverse Reinforcement Learning. arXiv preprint.
38. Ziebart, B.D., Maas, A., Bagnell, J.A., & Dey, A.K. (2008). Maximum entropy inverse reinforcement learning. In 23rd national conference on artificial intelligence - Volume 3, pp. 1433–1438.
39. Ziebart, B.D., Ratliff, N., Gallagher, G., Mertz, C., Peterson, K., Bagnell, J.A., Hebert, M., Dey, A.K., & Srinivasa, S. (2009). Planning-based prediction for pedestrians. In: Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'09, pp. 3931–3936. IEEE Press, Piscataway, NJ, USA.