Expectation-Maximization for Inverse Reinforcement Learning with Hidden Data

Kenneth Bogert
THINC Lab, Dept. of
Computer Science
University of Georgia
Athens, GA 30602
kbogert@uga.edu

Jonathan Feng-Shun Lin
Dept. of Electrical and
Computer Engg.
University of Waterloo
Waterloo, ON N2L 3G1
jf2lin@uwaterloo.ca

Prashant Doshi
THINC Lab, Dept. of
Computer Science
University of Georgia
Athens, GA 30602
pdoshi@cs.uga.edu

Dana Kulic
Dept. of Electrical and
Computer Engg.
University of Waterloo
Waterloo, ON N2L 3G1
dana.kulic@uwaterloo.ca

ABSTRACT

We consider the problem of performing inverse reinforcement learning when the trajectory of the agent being observed is partially occluded from view. Motivated by robotic scenarios in which limited sensor data is available to a learner, we treat the missing information as hidden variables and present an algorithm based on expectation-maximization to solve the non-linear, non-convex problem. Previous work in this area simply removed the occluded portions from consideration when computing feature expectations; in contrast our technique takes expectations over the missing values, enabling learning even in the presence of dynamic occlusion. We evaluate our new algorithm in a simulated reconnaissance scenario in which the visible portion of the state space varies. Finally, we show our approach enables apprenticeship learning by observing a human performing a sorting task in spite of key information missing from observations.

1. INTRODUCTION

Inverse reinforcement learning (IRL) [1] offers a way to learn skilled behavior by passively observing an expert. It finds applications as a methodology for robot learning from demonstrations and in imitation learning [2]. It approaches the task as that of learning the expert's preferences from observations given that the expert's set of capabilities and any associated non-determinism are known. The underlying premise is that the preference function is more easily transferable from the expert to the learner than other artifacts of the expert's behavior.

Technically, this problem is usually modeled by ascribing a Markov decision process (MDP) to the expert whose solution guides the expert's actions. While the dynamics of the expert are presumed to be available, the learner seeks to learn the reward function of this MDP from observing trajectories composed of state-action pairs of the expert [3]. As this maximum likelihood problem is under constrained, a popular method involves finding the distribution over possible expert behaviors that conform to the observed trajectories and which maximizes entropy [4, 5].

To motivate this paper, consider a line robot who must learn to sort fruit by observing a human expert. The particular task here involves sorting distinguishably ripe fruit from fruit that may not be ripe as yet. Immersing the expert in a motion capture system coupled with a video camera provides trajectory and image data that is informative about which type of fruit is placed in which bin. However, the coupled system does not reveal that the ripe fruit must be handled very gently during sorting and other fruit may not need special care. While force sensors could help here these tend to be inaccurate and imprecise. Nevertheless, it seems possible to infer the *latent* variable related to handling from the fact that the sorted fruit is generally not damaged and the time consumed in sorting types of fruit.

In the above example, a component (or factor) of the expert's action is unobserved in the collected trajectories. Knowing this component is essential in order to inversely learn the correct preferences of the expert from its behavior and complete the task successfully. Toward this end, this paper makes the following contributions:

1. We generalize IRL to operate in the context of data containing hidden factors. Specifically, we present a novel generalization of Ziebart et al.'s [4] maximum entropy optimization that conditions on hidden variables. Because the corresponding nonlinear program is not convex, the optimization is approximated to become convex.

2. Expectation-maximization (EM) [6] is an iterative scheme that is specifically well suited for optimization given hidden variables. Wang et al. [7] present an application of EM toward solving the latent-variable maximum entropy approximation. We adapt the EM to operate in the context of IRL where observed trajectories have missing action factors and address associated challenges.

3. The inverse learning is evaluated on data with missing factors collected from human demonstrations of a ball-sorting task. Humans sorted two types of colored balls (as substitutes for fruit) while immersed in a motion capture system coupled with a video camera; no force senors were utilized. We show that our approach allowed the learner – a robotic arm – to correctly learn to sort the balls while realizing that a certain type of ball needed

Appears in: *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016),*
J. Thangarajah, K. Tuyls, C. Jonker, S. Marsella (eds.),
May 9–13, 2016, Singapore.

to be handled gently in order to complete the task successfully, analogously to the human expert.

The generalized IRL that operates with possibly hidden data is well suited for a variety of other tasks where portions of the data may be occluded from the learner. This paper's contributions represent significant steps toward making IRL ready for real-world applications.

Rest of this paper is organized as follows. We review IRL and an entropy optimization method for such learning in Section 2. Hidden variables are introduced in Section 3 and a generalization of maximal entropy IRL in the context of hidden data is presented. We evaluate the performance of the generalized IRL on two robotic problem domains in Section 4 and discuss related work in Section 5. Concluding remarks and future work is outlined in Section 6.

2. BACKGROUND

We briefly review IRL and a solution method that is based on maximum entropy [8].

2.1 Inverse Reinforcement Learning

IRL [3] seeks to find the most likely reward function, R_I, that an expert I is executing. Current IRL methods typically apply in the presence of a single expert where the expert has solved a Markov decision process (MDP). Furthermore, they assume that this MDP excluding the reward function is known to the learner. The expert executes its policy multiple times and each (fully) observed trajectory of arbitrary length T is a sequence of state-action pairs, $Y = (\langle s,a\rangle^0, \langle s,a\rangle^1, \ldots, \langle s,\varnothing\rangle^T)$, where $\varnothing$ is the null action and Y belongs to the set of observed trajectories $\mathcal{Y}$. Let $\mathbb{Y}$ be the finite set of all trajectories of length T; $\mathcal{Y} \subseteq \mathbb{Y}$.

As the space of possible reward functions is very large, the function is commonly expressed as a linear combination of $K > 0$ *feature functions*, $R_I(s,a) \triangleq \sum_{k=1}^{K} \theta_k \; \phi_k(s,a)$, where θ_k are the weights, and ϕ: $S \times A_I \rightarrow \{0,1\}$, is a feature function. It maps a state from the set of states, S, and an action from the set of I's actions, A_I, to 0 or 1. IRL algorithms use feature expectations to evaluate the quality of the learned reward function. The k^{th} feature expectation for a learned reward function R_I is, $\sum_{Y\in\mathbb{Y}} Pr(Y)$ $\sum_{\langle s,a\rangle\in Y} \phi_k(s,a)$. Alternately, the k^{th} feature expectation may be formulated as, $\sum_{\pi_I\in\Pi_I} Pr(\pi_I) \sum_{s\in S} \mu_{\pi_I}(s)\; \phi_k(s,\pi_I(s))$. Here, $\mu_{\pi_I}(s)$ is the state visitation frequency: the number of times state s is visited on using deterministic policy π_I from the set of policies Π_I. The expectations are compared with those of the expert's from its observed trajectory, $\hat{\phi}_k \triangleq \sum\limits_{Y\in\mathcal{Y}} \tilde{Pr}(Y) \sum\limits_{\langle s,a\rangle\in Y} \phi_k(s,a)$, where $\tilde{Pr}(Y)$ is the empirical probability of Y. For example, if all trajectories in $\mathcal{Y}$ are observed just once then we may obtain, $\hat{\phi}_k \triangleq \frac{1}{|\mathcal{Y}|} \sum\limits_{Y\in\mathcal{Y}} \sum\limits_{\langle s,a\rangle\in Y} \phi_k(s,a)$.

2.2 Maximum Entropy IRL

Multiple policies may induce matches with the observed feature expectations equally well. In order to resolve this ill-posed problem, the principle of maximum entropy is useful [8]. While multiple formulations of the IRL problem exist that involve maximizing entropy [4, 5], we focus on the approach by Ziebart *et al.* [4] in this paper. Hence, we briefly review this problem formulation.

The approach maintains a distribution over all trajectories constrained to match the observed feature expectations while being maximally noncommittal to any one trajectory. Mathematically, the problem is formulated as a nonlinear optimization:

$$\max_{\Delta} \left(-\textstyle\sum_{Y\in\mathbb{Y}} Pr(Y)\; log\; Pr(Y) \right)$$

$$\textbf{subject to} \;\; \textstyle\sum_{Y\in\mathbb{Y}} Pr(Y) = 1 \qquad (1)$$

$$\textstyle\sum_{Y\in\mathbb{Y}} Pr(Y) \; \sum_{\langle s,a\rangle\in Y} \phi_k(s,a) = \hat{\phi}_k \qquad \forall k$$

Here, Δ is the space of all distributions $Pr(Y)$ and $\hat{\phi}_k$ is as defined previously in Section 2.1.

We may apply Lagrangian relaxation bringing both the constraints into the objective function and then solving the dual. The relaxed objective function becomes,

$$\mathcal{L}(Pr,\boldsymbol{\theta},\eta) = -\textstyle\sum_{Y\in\mathbb{Y}} Pr(Y)\; logPr(Y) + \sum_k \theta_k$$
$$\left(\textstyle\sum_{Y\in\mathbb{Y}} Pr(Y) \; \sum_{\langle s,a\rangle\in Y} \phi_k(s,a) - \hat{\phi}_k\right) + \eta$$
$$\left(\textstyle\sum_{Y\in\mathbb{Y}} Pr(Y) - 1\right) \qquad (2)$$

As Eq. 2 is convex for a deterministic MDP, taking the derivative with respect to $Pr(Y)$ and setting it to zero gives us the optimum:

$$\frac{\partial\mathcal{L}}{\partial Pr(Y)} = -log\; Pr(Y) - 1 + \sum_k \theta_k \sum_{\langle s,a\rangle\in Y} \phi_k(s,a) + \eta = 0$$

$$Pr(Y) = \frac{e^{\sum_k \theta_k \sum_{\langle s,a\rangle\in Y} \phi_k(s,a)}}{\Xi(\boldsymbol{\theta})} \qquad (3)$$

where $\Xi(\boldsymbol{\theta})$ is the normalization constant $e^{-\eta+1}$; as such η may easily be obtained. Plugging Eq. 3 back into the Lagrangian (Eq. 2), we arrive at the dual $\mathcal{L}^{\text{dual}}(\boldsymbol{\theta})$ which is concave. We have,

$$\mathcal{L}^{\text{dual}}(\boldsymbol{\theta}) = log\; \Xi(\boldsymbol{\theta}) - \textstyle\sum_k \theta_k \hat{\phi}_k \qquad (4)$$

The dual above may be optimized to obtain $\boldsymbol{\theta}^*$ using the exponentiated gradient descent [9]. The gradient,

$$\nabla\mathcal{L}^{\text{dual}}(\boldsymbol{\theta}) \; = \; \textstyle\sum_{Y\in\mathbb{Y}} Pr(Y) \displaystyle\sum_{\langle s,a\rangle\in Y} \phi_k(s,a) - \hat{\phi}_k$$

involves summing over the set of all trajectories, which may be very large. Ziebart *et al.* [4] suggest an efficient approach that calculates the expected state visitation frequency and action distribution instead. The feature expectations may be obtained easily given these values.

3. MAXENT IRL WITH HIDDEN DATA

We consider situations where portions of the expert's trajectory encompassing some states, parts of actions or both are hidden from the learner's view. A simple modification to the maximum entropy IRL reviewed in the previous section is to simply ignore occluded portions of the trajectories when calculating feature expectations. However, the gradient for those features that activate in the occluded states only or due to occluded actions become zero. As a result, this method may not learn weights for many features, which motivates a principled way to managing hidden data.

We begin by providing a general formulation of maximum entropy IRL with hidden data followed by applying it to our specific case where components of actions cannot be observed by the learner.

3.1 General Formulation

Consider again the expert's observed trajectory of length T, $Y = (\langle s,a\rangle^0, \langle s,a\rangle^1, \ldots, \langle s,\varnothing\rangle^T)$. Let Y be the observed portion of the full trajectory X and denote the occluded portion as Z; in other words, $X = (Y \cup Z)$. Z could be a few more state-action pairs that were performed in a portion of the state space that is occluded from the learner. It could also be the component of the expert's action that is hidden from the learner's view.

Of course, we may simply ignore the occluded data [10]. However, we hypothesize that a revised formulation of the maximum entropy nonlinear program in (1), which allows for an expectation of Z given Y provides a tighter solution. In other words, the resulting distribution may possess lesser entropy because the observed Y informs an expectation of Z effectively allowing more data.

Recall $\hat{\phi}_k$ in Section 2.1 whose analogous definition is $\sum_{X \in \mathbb{X}} \tilde{Pr}(X) \times \sum_{\langle s,a \rangle \in X} \phi_k(s,a)$. We may rewrite this as follows $\sum_{Y \in \mathcal{Y}} \sum_{Z \in \mathbb{Z}} \tilde{Pr}(Y,Z) \times \sum_{\langle s,a \rangle \in X} \phi_k(s,a)$. However, as Z is hidden we may treat it as a latent variable and decompose the joint $\tilde{Pr}(Y,Z)$ as,

$$\hat{\phi}_k^{Z|Y} \triangleq \sum_{Y \in \mathcal{Y}} \tilde{Pr}(Y) \sum_{Z \in \mathbb{Z}} Pr(Z|Y) \sum_{\langle s,a \rangle \in Y \cup Z} \phi_k(s,a) \qquad (5)$$

The maximum entropy program of (1) then changes to accommodate the hidden data as given below,

$$\begin{aligned} &\max_{\Delta} \left(-\textstyle\sum_{X \in \mathbb{X}} Pr(X)\ log\ Pr(X)\right) \\ &\textbf{subject to} \quad \textstyle\sum_{X \in \mathbb{X}} Pr(X) = 1 \\ &\textstyle\sum_{X \in \mathbb{X}} Pr(X) \sum_{\langle s,a \rangle \in X} \phi_k(s,a) = \hat{\phi}_k^{Z|Y} \qquad \forall k \end{aligned} \qquad (6)$$

Here, Δ is the space of all distributions $Pr(X)$ and $\hat{\phi}_k^{Z|Y}$ is as defined in Eq. 5.

The key difference between the program of (1) and its generalized version above is in the computation of the observed feature expectations, $\hat{\phi}_k$, in the primary constraint. The original program simply sums the value of the k^{th} feature function across each $\langle s,a \rangle$ pair in the observed trajectory. However, as some $\langle s,a \rangle$ pairs may be missing from the trajectory, we obtain an expectation of the hidden state-action pairs given the observed portion, $Pr(Z|Y)$. Prior probability $\tilde{Pr}(Y)$ empirically approximates the true prior; we may use the fraction of times Y appears in $\mathcal{Y}$. Each pair in the complete data $Y \cup Z$ is then utilized to find the observed feature function counts.

The revised maximum entropy optimization above may be seen as an application of Wang *et al.* [7]'s generalization of the maximum entropy program to allow considerations of incomplete data to the context of inverse reinforcement learning. This gives us the first principled generalization of IRL to contexts involving hidden data that does not simply ignore it. Notice that if Z is empty indicating that all data is observed, then $X = Y$ and (6) reduces to the maximum entropy program of (1). Thus, the above generalizes the original maximum entropy program.

3.2 Hidden Actions

Motivated by the application of designing a robotic fruit sorter, we consider the setting where the expert's action consists of $N > 1$ components: $A_I = A_1 \times A_2 \times \ldots \times A_N$. A subset of these components, $M \leq N$ are unobserved in its trajectory. Subsequently, we may rewrite the set of actions as: $A_I = A_1 \times A_2 \times \ldots \times A_M \times \ldots \times A_N$. Let $\hat{A}_I = A_1 \times A_2 \times \ldots \times A_M$.

Given the above, Y reduces from the sequence of observed state-action pairs to the observed state-action pairs where the actions do not include the hidden component. Formally, Y = $\{\langle s, a_{/\hat{a}} \rangle^0, \langle s, a_{/\hat{a}} \rangle^1, \ldots, \langle s, \varnothing \rangle^T\}$, and $Z = X/Y$. The maximum entropy program of (6) is applied to accommodate this hidden data.

Due to the presence of the conditional probability in the Lagrangian $\mathcal{L}(Pr(X), \boldsymbol{\theta}, \eta)$, the relaxed objective function is not convex. Its partial derivative w.r.t. $Pr(X)$ shown below may not yield a closed-form solution:

$$\frac{\partial \mathcal{L}}{\partial Pr(X)} = -log\ Pr(X) - 1 + \sum_{k=1}^{K} \theta_k \sum_{\langle s,a \rangle \in X} \phi_k(s,a) + \sum_{k=1}^{K} \theta_k \left(\sum_{Y \in \mathcal{Y}} \tilde{Pr}(Y) \frac{\sum_{Z' \in \mathbb{Z}} \left[\sum_{\langle s,a \rangle \in X'} \phi_k(s,a) - \sum_{\langle s,a \rangle \in X} \phi_k(s,a) \right] Pr(X')}{Pr(Y)^2} \right) + \eta$$

where $X' = (Y \cup Z')$ and Z' is a hidden trajectory in $\mathbb{Z}$. Wang *et al.* suggest approximating the above partial derivative with the following simpler form:

$$\frac{\partial \mathcal{L}}{\partial Pr(X)} \approx -log\ Pr(X) - 1 + \sum_{k=1}^{K} \theta_k \sum_{\langle s,a \rangle \in X} \phi_k(s,a) + \eta$$

Setting the above to 0 and solving for $Pr(X)$ yields an optimum for $Pr(X)$ that is log linear:

$$Pr(X)^* \approx \frac{e^{\sum_k \theta_k \sum_{\langle s,a \rangle \in X} \phi_k(s,a)}}{\Xi(\boldsymbol{\theta})} \qquad (7)$$

Now that we have the (approximately) optimal value of $Pr(X)$, we seek to find the maximizing value of Lagrangian parameter vector $\boldsymbol{\theta}$. In order to find this, Eq. 7 is substituted back in $\mathcal{L}(Pr, \boldsymbol{\theta}, \eta)$ and we minimize the resulting dual to obtain the following:

$$\mathcal{L}^{\text{dual}}(\boldsymbol{\theta}) \approx log\ \Xi(\boldsymbol{\theta}) - \sum_{k=1}^{K} \theta_k \sum_{Y \in \mathcal{Y}} \tilde{Pr}(Y) \sum_{Z \in \mathbb{Z}} Pr(Z|Y) \times \sum_{\langle s,a \rangle \in Y \cup Z} \phi_k(s,a) \qquad (8)$$

3.3 Expectation-Maximization

One way to obtain the maximizing value of parameter vector $\boldsymbol{\theta}$ is to again seek to perform the exponentiated gradient descent. However, Eq. 8 may not offer a closed-form gradient due to the presence of $Pr(Z|Y)$. Alternatively, Wang *et al.* offer an EM-based iterative scheme, which when adapted to our maximum entropy IRL converges to a fixed point yielding $\boldsymbol{\theta}$ that maximizes Eq. 8.

We seek to maximize the likelihood of Lagrangian parameters $\boldsymbol{\theta}$, where the log likelihood is defined as $LL(\boldsymbol{\theta}|\mathcal{Y}) = log\ Pr(\mathcal{Y}; \boldsymbol{\theta})$. This becomes,

$$\begin{aligned} LL(\boldsymbol{\theta}|\mathcal{Y}) &= log \prod_{Y \in \mathcal{Y}} Pr(Y;\boldsymbol{\theta})^{\tilde{Pr}(Y)} = \sum_{Y \in \mathcal{Y}} \tilde{Pr}(Y)\ log\ Pr(Y;\boldsymbol{\theta}) \\ &= \sum_{Y \in \mathcal{Y}} \tilde{Pr}(Y)\ log\ Pr(Y;\boldsymbol{\theta}) \sum_{Z \in \mathbb{Z}} Pr(Z|Y;\boldsymbol{\theta}) \\ &= \sum_{Y \in \mathcal{Y}} \tilde{Pr}(Y) \sum_{Z \in \mathbb{Z}} Pr(Z|Y;\boldsymbol{\theta})\ log\ Pr(Y;\boldsymbol{\theta}) \end{aligned}$$

Rewriting $Pr(Y;\boldsymbol{\theta}) = \frac{Pr(Y,Z;\boldsymbol{\theta})}{Pr(Z|Y;\boldsymbol{\theta})}$ in the above equation we get,

$$
\begin{aligned}
LL(\theta|\mathcal{Y}) &= \sum_{Y\in\mathcal{Y}} \tilde{P}r(Y) \sum_{Z\in\mathbb{Z}} Pr(Z|Y;\boldsymbol{\theta})\, log\, \frac{Pr(Y,Z;\boldsymbol{\theta})}{Pr(Z|Y;\boldsymbol{\theta})} \\
&= \sum_{Y\in\mathcal{Y}} \tilde{P}r(Y) \sum_{Z\in\mathbb{Z}} Pr(Z|Y;\boldsymbol{\theta})\, (log\, Pr(Y,Z;\boldsymbol{\theta}) \\
&\quad -log\, Pr(Z|Y;\boldsymbol{\theta}))
\end{aligned}
$$

The above likelihood may be iteratively improved until convergence (possibly to a local optima) by casting it in an EM scheme. Specifically, the likelihood may be rewritten as, $Q(\boldsymbol{\theta},\boldsymbol{\theta}^{(t)})+C(\boldsymbol{\theta},\boldsymbol{\theta}^{(t)})$ where,

$$
Q(\boldsymbol{\theta},\boldsymbol{\theta}^{(t)}) = \sum_{Y\in\mathcal{Y}} \tilde{P}r(Y) \sum_{Z\in\mathbb{Z}} Pr(Z|Y;\boldsymbol{\theta}^{(t)})\, log\, Pr(Y,Z;\boldsymbol{\theta}) \tag{9}
$$

$$
C(\boldsymbol{\theta},\boldsymbol{\theta}^{(t)}) = -\sum_{Y\in\mathcal{Y}} \tilde{P}r(Y) \sum_{Z\in\mathbb{Z}} Pr(Z|Y;\boldsymbol{\theta}^{(t)})\, log\, Pr(Z|Y;\boldsymbol{\theta})
$$

Notice that in Eq. 9, we may replace $Pr(Y,Z;\boldsymbol{\theta})$ with $Pr(X;\boldsymbol{\theta})$, and we may now substitute in Eq. 7. This gives us,

$$
\begin{aligned}
Q(\boldsymbol{\theta},\boldsymbol{\theta}^{(t)}) &= \sum_{Y\in\mathcal{Y}} \tilde{P}r(Y) \sum_{Z\in\mathbb{Z}} Pr(Z|Y;\boldsymbol{\theta}^{(t)}) \Bigg(\sum_k \theta_k \\
&\quad \times \sum_{\langle s,a\rangle\in X} \phi_k(s,a) - log\, \Xi(\boldsymbol{\theta})\Bigg) \\
&= -\Bigg(log\, \Xi(\boldsymbol{\theta}) - \sum_k \theta_k \sum_{Y\in\mathcal{Y}} \tilde{P}r(Y) \sum_{Z\in\mathbb{Z}} Pr(Z|Y; \\
&\quad \boldsymbol{\theta}^{(t)}) \sum_{\langle s,a\rangle\in X} \phi_k(s,a)\Bigg)
\end{aligned}
$$

Notice that the Q function above is almost the negative of the dual Lagrangian presented in Eq. 8 with the only difference being $\boldsymbol{\theta}^{(t)}$. On maximizing Q to convergence, we have $\boldsymbol{\theta} = \boldsymbol{\theta}^{(t)}$, due to which we conclude that maximizing the Q function minimizes the dual Lagrangian function. Therefore, approximately solving the maximum entropy IRL of (6) for $\boldsymbol{\theta}$ is equivalent to maximizing the Q function; Theorem 1 states this formally.

THEOREM 1. *Under the assumption that the distribution of the complete trajectories is log linear, the reward function that maximizes the log likelihood of the observed incomplete trajectories also maximizes the entropy of the distribution constrained by the conditional expectation of the feature functions.*

This gives us another path to a feasible solution to the generalized IRL in (6) subject to the constraints and the log-linear model for $Pr(X)^*$. However, due to the nonconvexity of the Lagrangian this solution may not necessarily be the optimal one.

Using this insight, we may now split the primary constraint in (6) into two portions and solve each separately.

3.3.1 E-step

The E-step involves obtaining a conditional expectation of the K feature functions using the parameter $\boldsymbol{\theta}^{(t)}$ from the previous iteration. We may initialize the parameter vector randomly. For all $k = 1 \ldots K$ we obtain,

$$
\hat{\phi}_k^{Z|_Y,(t)} = \sum_{Y\in\mathcal{Y}} \tilde{P}r(Y) \sum_{Z\in\mathbb{Z}} Pr(Z|Y;\boldsymbol{\theta}^{(t)}) \sum_{\langle s,a\rangle\in Y\cup Z} \phi_k(s,a)
$$

3.3.2 M-step

This involves solving a simpler version of the constrained maximum entropy program of (6) by utilizing $\hat{\phi}_k^{*,(t)}$ from the E-step above and the log linear model for $Pr(X)$ to obtain $\boldsymbol{\theta}$.

$$
\begin{aligned}
&\max_{\Delta} \Big(-\sum_{X\in\mathbb{X}} Pr(X)\, log\, Pr(X)\Big) \\
&\textbf{subject to} \quad \sum_{X\in\mathbb{X}} Pr(X) = 1 \\
&\sum_{X\in\mathbb{X}} Pr(X) \sum_{\langle s,a\rangle\in X} \phi_k(s,a) = \hat{\phi}_k^{Z|_Y,(t)} \qquad \forall k
\end{aligned}
\tag{10}
$$

We point out that solving the nonlinear program above is facilitated by the fact that $\hat{\phi}_k^{Z|_Y,(t)}$ is already available from the E-step and therefore may be treated as a constant. Consequently, optimizing the relaxed Lagrangian for (10) becomes straightforward. The iterative EM converges when $\boldsymbol{\theta}$ obtained from the M-step is identical to the previous $\boldsymbol{\theta}^{(t)}$.

Due to the presence of local optima, we perform restarts with different initial values for $\boldsymbol{\theta}$ and treat the run that achieved the highest entropy as the optimal one.

4. PERFORMANCE EVALUATION

In this section, we describe the method for evaluating the performance of the EM based IRL presented previously. We introduce suitable baselines for comparison and report on our experiments.

4.1 Problem Domains and Features

We introduce two problem domains to evaluate the performance of the EM based generalized IRL with hidden data.

UAV reconnaissance with ground cover Our first domain is a simulation-only reconnaissance application in which an aerial drone L tracks a fugitive I moving repeatedly through a grid of sectors to a safe sector. The drone is tasked with learning the policy of the fugitive from observing multiple runs over time each of which differs because the fugitive starts at a random location and exhibits a stochastic transition function. An important challenge is that some of the sectors are forested, which provides cover for the fugitive. Unfortunately, the fugitive's action is completely occluded from L in the forested sectors. We illustrate this domain in Fig. 1.

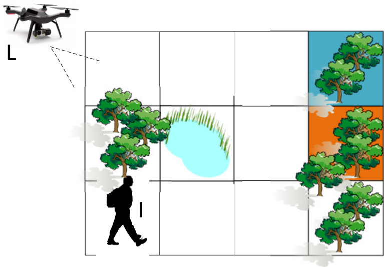


Figure 1: An overhead drone is tracking a fugitive moving through a 3 × 4 grid of sectors. The fugitive's movement is partially occluded as it passes through the forest cover. The sectors shaded blue and orange are safe and penalty sectors respectively.

Learner L models the fugitive I as following the same policy output from its MDP across the runs. The states, actions and transition function of the MDP are known to L; this is the standard assumption in IRL although recent work seeks to partially relax this assumption [11]. The states of the MDP are the sector coordinates (x, y) and I has four actions that move it in each of the four cardinal directions. I's transition function is modeled as I moving in the intended direction successfully with a probability of 0.9 and the remaining probability mass is distributed to moving in the other three directions. The reward function of the fugitive R_I is modeled as a linear combination of weighted feature functions. R_I is composed of the following three binary feature functions:

1. **Cost of movement** is the penalty for taking an action. This encourages I to reach the safe sector quickly. It is used at all states and actions;
2. **Goal state reached** is activated if I reaches the sector located at (4,3);
3. **Penalty state reached** is activated if I reaches the sector located at (4, 2) in the theater.

Learning from human demonstration of ball sorting The second domain involves a learning robot L equipped with Trossen Robotics' PhantomX Pincher arm that is given data on humans simulating a fruit sorting task. Humans are immersed in a Motion Analysis, 8 Eagle camera system and are tasked with sorting 24 colored balls into two bins without deforming the balls. Motion capture markers are placed along one of their arms and the wrist and coordinate data is recorded in the Cortex software. A synchronized video camera records the performance as well. L must successfully perform the same task.

Four types of balls – 6 each – are present. These are laboratory substitutes for ripe and regular fruit of two varieties. Balls made of soft playable clay represent ripe fruit and must be handled gently. Ping-pong balls and other solid balls can be handled regularly.

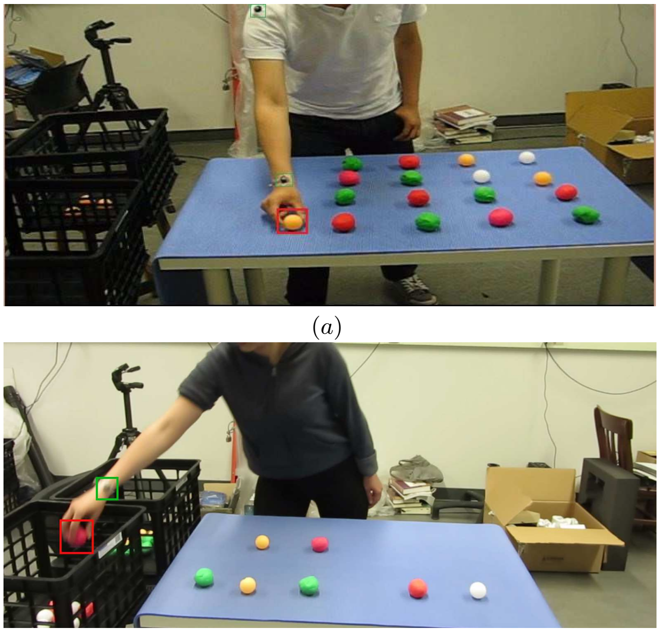
Despite the presence of the motion capture and video camera system, a key factor in performing the task successfully remains uncaptured. This is the amount of force that is applied while grabbing the balls and placing them in a bin. In particular, the soft clay ball (substitute for ripe fruit) must be grabbed gently to avoid damaging it while the hard ball (regular fruit) is grabbed with default force. We show frames of two human subjects sorting the balls in Fig. 2.

Learner L models human I's sorting behavior as guided by a policy that is output from a MDP whose states, actions and transition function are known to L. States of the MDP are combinations of the following four factors: ball_position = { on table, in center, bin 1, bin 2}, ball_type = {1,2,3,4}, holding_ball = {true, false}, and time_for_prev_action which is the time duration of the last action discretized into 12 segments: {less than 0.3s, 0.3s, 0.4s, . . . , 1.3s, greater than 1.3s}.

Human I performs one of the following seven actions:

- *Grab hard* denotes the hand grabbing a ball with default force and sets the holding_ball factor to true;
- *Grab soft* denotes the hand grabbing a ball gently and sets holding_ball to true.

  On average, grabbing a ball gently is slower than grabbing it by default. Both grab actions are applicable for states where the ball position is on the table only;
- *Move To center* denotes the hand moving to the center position while holding a ball. This action applies for states where the hand has grabbed the ball that is on the table;
- *Move to bin 1* is the hand moving to bin 1 while holding the ball;



(b)

Figure 2: Video captures from our experiment showing a subject performing the ball sorting task. Notice that there are four types of balls in all - soft clay balls of two colors and hard ping-pong balls of two colors. Balls are sorted into two bins based on their color. (a) Subject picks a hard ball, and (b) Subject places a soft ball in one of the bins. Positions of the red bounding box correspond to the states of the MDP.

- *Move to bin 2* is the hand holding the ball moving to bin 2;

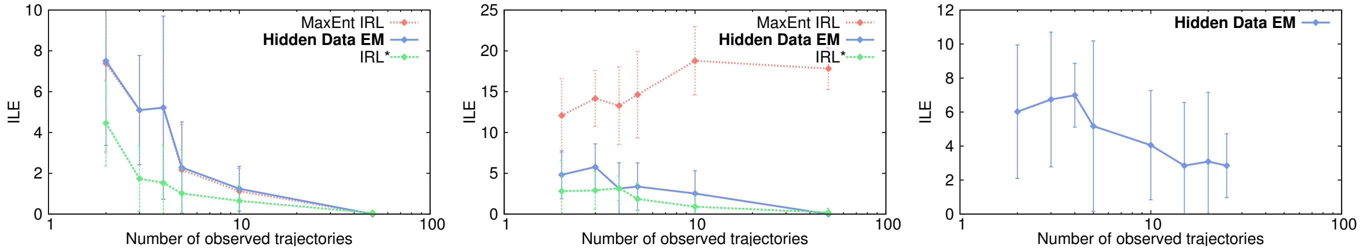
  These actions that move the ball to the bins are applicable when the ball is held and its position is in the center;
- *Release hard* places the ball with default force in a bin and sets the factor holding_ball to false, and on average is faster than grab soft; only for states in the center position
- *Release soft* places the ball gently in a bin and sets holding_ball to false.

  On average, placing a ball gently in a bin is slower than placing it with regular force. Both release actions are applicable when the ball is held and its position is in the center.

The transition function models each action as having its intended effect with a probability of 0.9 with the remaining probability mass distributed to the intended states of other actions applicable at the current state. For the state variable time_for_prev_action, the remaining mass is distributed to the other values.

Finally, I's reward function is modeled to have the following feature functions:

- **Release ball type X in bin 1** leads to 4 feature functions each of which is activated when a ball of that type is placed in bin 1;
- **Release ball type X in bin 2**, 4 features each of which is activated when a ball of that type is placed in bin 2;
- **Handle ball type X gently** also leads to 4 features each of which is activated when the ball of the corresponding type is grabbed and moved gently.



(a) (b) (c)

Figure 3: Performance evaluation of all three methods on the UAV reconnaissance simulation. (a) **ILE for all methods in the fully observable setting. Notice that Hidden Data EM's performance coincides with that of MaxEnt IRL as we may expect.** (b) **ILE for the setting where the fugitive's trajectories are partially occluded.** (c) **High levels of occlusion which changes randomly between trajectories. Lower ILE is better. Vertical bars denote the standard deviation.**

In total, the reward function is composed of 12 weighted feature functions.

Parameters of the robot's MDP are similar to those of the human's except that the arm took the same time to move the ball from the table to the bin whether it is grasping the ball gently or not. However, we noticed that the arm exhibited a greater tendency to drop the ball when grasping it gently. Therefore, we modified the transition function to reflect this property of the robotic arm. Specifically, actions with the default (hard) grasp have a 0.89 probability of being successful and 0.11 probability of dropping the ball. On the other hand, movement actions involving a gentle grasp have an increased 0.17 probability of dropping the ball while moving it to the bin (values were determined empirically prior to the experiment). Finally, the reward function features are the same as for the human.

4.2 Metrics and Baselines

In order to evaluate the performance of a method for IRL, it is tempting to compare the learned reward function directly to the actual reward function of I, R_I, if available. However, the learned reward function depends on the quality and quantity of the observations, and such comparisons ignore the fact that reward functions different from I's actual one may still result in the same optimal policy as the one utilized by I; these reward functions should be considered equivalent. Consequently, we adopt the metric used by Choi and Kim [12], which compares between the value functions of policies obtained from optimally solving I's MDP and from solving the MDP that is formed by using the learned reward function. In other words, let R_I^L be the reward function learned by L, π_I^* be the optimal policy of I and π_I^L be the policy obtained by optimally solving the MDP whose reward function is now R_I^L. Then, we use the following *inverse learning error (ILE)* metric:

$$\mathsf{ILE} = ||V^{\pi_I^*} - V^{\pi_I^L}||_1$$

where $V^{\pi_I^*}$ is the optimal value function of I's MDP and $V^{\pi_I^L}$ is the value function due to utilizing the policy π_I^L on I's MDP. In the event that the learned reward function results in a policy identical to I's actual optimal policy, $\pi_I^* = \pi_I^L$, ILE will be 0; ILE monotonically increases as the two diverge. Furthermore, use of the difference in values also allows the error to grow proportionately to the value of the states where the learned policy diverges thereby placing emphasis on higher-valued states.

When the true reward function is unavailable we must utilize another metric to show successful learning. As our fruit sorting domain involves learning from humans with unknown reward functions we evaluate the learned rewards by utilizing them to guide a real robot in an apprenticeship learning procedure. Specifically, the learned rewards are transfered to the robot's MDP, which is then solved optimally to obtain the policy executed by the robot. This allows us to evaluate the performance of our new method by evaluating the performance of the robot in its assigned task.

To provide a comparative baseline, we evaluate the EM based IRL against *two* existing methods. First is a simple modification of Ziebart *et al.*'s IRL that was reviewed in Section 2.2. Specifically, occluded portions of the trajectories are simply ignored when calculating feature expectations. Thus the gradient for those features that activate in the occluded states or due to occluded actions become zero. As a result, this algorithm may not learn weights for many features. We label this algorithm as MaxEnt IRL.

A second baseline is to use a single-expert version of the method presented by Bogert and Doshi [10] that manages occlusion. While this algorithm also ignores feature expectations due to occluded states, it does not use the gradient and as such may find the correct weights for most features. However, this method requires the set of occluded states and actions to be fixed across all trajectories. As such, it cannot be applied to scenarios where this changes between observations. This method is labeled as IRL*.

Finally, we note that a possible approach in the context of occluded actions when actions are deterministic or when the transition stochasticity is very low is to simply fill in the blanks with the most likely action that produces the observed state transition in the trajectory. However, this approach becomes implausible in the context of actions with moderate or high nondeterminism or when two or more actions have similar transition probabilities; the latter case presents itself in both our domains. As such, this approach lacks robustness and a principled way of learning under occlusion is needed.

4.3 Experiments

We begin by comparing the performance of EM based IRL, labeled as Hidden Data EM, with the performances of MaxEnt IRL and IRL* on simulations in the UAV reconnaissance domain. Each method received up to 50 trajectories where a trajectory involved movement by the fugitive for 5 steps. At least one path exists from any start sector that reaches the safe sector within 5 steps. The fugitive started at different locations and its state-action pairs constitute the trajectory. The UAV can observe neither the state nor the action of the fugitive when it is passing through sectors covered by forest. Hidden Data EM used 10 restarts with random initial feature weights and the result with the maximum entropy is chosen as output.

Figure $3(a)$ reports the ILE as UAV L observes an increasing number of trajectories in the absence of any occlusion. The data points shown are the means of 100 runs. Here, we expect the per-

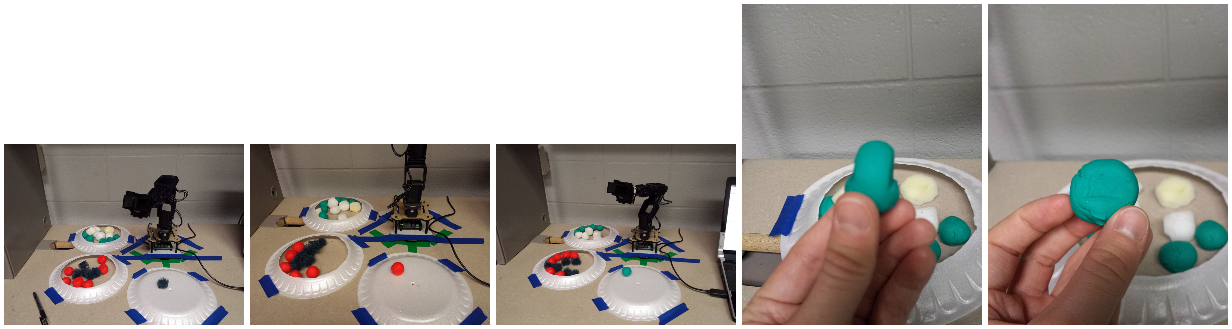


Figure 4: A series of snapshots of the Phantom Pincher arm sorting the balls. We introduce balls one at a time due to lack of sufficient space for several balls and the arm being small. The two bins with balls are visible on the left side of the arm. We show the deformed clay ball due to mishandling and the non-deformed clay ball on the right.

formance of Hidden Data EM to be identical to the performance of MaxEnt IRL because the former collapses into the latter when the set Z is empty. Indeed, the performances of the two methods coincide. IRL* performs slightly better than the other two methods in this test – this may be attributed to its policy-oriented approach making better use of available data.

Next, we compare all three methods under fixed occlusion (Fig. 3(b)) and when the occlusion varies between trajectories (Fig. 3(c)). The latter may occur if the fugitive is tracked in different regions of the theater that exhibit differing forest cover. Notice that MaxEnt IRL performs much worse now due to occlusion and its ILE remains significantly above 0 whereas the ILE of both IRL* and Hidden Data EM eventually reaches zero. The difference between Hidden Data EM and MaxEnt IRL is now statistically significant (paired Student's t-test, $p < 0.001$). For randomly varying levels of occlusion, neither MaxEnt IRL nor IRL* are applicable as both require the occluded state space to be the same across all trajectories. However, the EM-based IRL completes each trajectory with expected information and uses it in the inverse learning thereby showing a robust performance even for this pathological case.

Our second experiment involved sorting 24 balls of four types into two bins using a Phantom Pincher arm. [1] Each of the three methods learned a reward function from a single run (24 balls sorted) of the ball sorting task performed by a human subject. As we mentioned previously, the type of grasp employed by the human (gentle or default grasp) is hidden from the learner's view. Additionally, we utilized a control method that used the MaxEnt IRL on trajectory data in which the type of grasp was made explicit. Hence, there was no occlusion; we refer to this method as Occlusion-free control and it provides an upper bound performance for all other methods. For a deeper analysis of the learned reward function, we show the weights of the 12 feature functions learned by all four methods in the Appendix.

Table 1 shows the performance of all four methods on the task of sorting the 24 balls into two bins. Unable to infer the occluded action, MaxEnt IRL utilized the default grasp while sorting the balls thereby deforming 11 of the 24 balls. Nevertheless, it did sort most of the balls correctly. IRL* correctly inferred that one type of clay ball should be grasped gently but failed to arrive at a similar conclusion for the second set of soft balls and incorrectly found that the hard balls should also receive gentle handling. As a result, it damaged 5 and dropped 3. On the other hand, Hidden Data EM correctly learns a reward function – in 4 EM iterations – that has the arm grasp gently all the balls of clay with the two colors. As a result, none of the balls were damaged while one ball was dropped during the sorting. All balls were correctly sorted into the two bins by color. This strong performance by Hidden Data EM matches that of the Occlusion-free control method.

Method	Ball correctly sorted	Balls dropped	Balls damaged
MaxEnt IRL	23	1	11
IRL*	21	3	5
Hidden Data EM	23	1	0
Occlusion-free control	23	1	0

Table 1: Performance of the various methods on the Phantom Pincher arm on the ball sorting task. Notice that the EM-based IRL did not damage any balls and dropped just one ball while sorting. It's performance is similar to our control method that provides an upper bound.

We show a series of snapshots in Fig. 4 of an example run of sorting the balls by the robotic arm. The right-most two snapshots compare a deformed and nondeformed clay ball for reference. Notice that most of the balls are correctly sorted at the end. A video of the sort is available for viewing at `https://www.youtube.com/channel/UC0IivL24ibPm18FJwdN3ITw`.

In summary, our generalized formulation of maximum entropy IRL presents a plausible method for inversely learning from data that suffers from occlusion. Experiments on two domains provide convincing evidence that the method can operate in the context of varying amount and type of occlusion – of state and components of action. The EM converges quickly to a reward function that accurately reflects that of the expert.

5. RELATED WORK

Ng and Russell [3] introduced IRL as a problem involving a single subject agent learning from a single expert modeled as a MDP. Inverse optimal control [13] generalizes IRL by allowing for control frameworks other than MDPs as well. Previous improvements to IRL include modeling the reward function as a linear combination of features [14] and learning with noisy feature functions [15].

In this paper, we relax the key assumption that the entire trajectory of the expert is fully observable, which has significant implications toward making IRL more pragmatic. Bogert and Doshi [10] also investigates maximum entropy IRL under occlusion. However, key differences exist in its method and treatment of occlusion. It seeks to maximize the entropy over all possible policies as introduced by

[1] We replaced ping-pong balls with smaller balls because the former were overly large for the Phantom's small gripper.

Ball sorting task												
Method	ϕ_1	ϕ_2	ϕ_3	ϕ_4	ϕ_5	ϕ_6	ϕ_7	ϕ_8	ϕ_9	ϕ_{10}	ϕ_{11}	ϕ_{12}
MaxEnt IRL	0.2213	0.2456	0	0	0	0	0.27	0.259	0	0	0	0
IRL*	0.8331	1.697	-0.4020	-0.44	0.25	0.26	1.08	1.07	1.08	1.06	0.61	-0.78
Hidden Data EM	0.1622	0.1033	0	0	0	0	0.1879	0.1647	0	0.37	0	0.0109
Occlusion-free control	0.0002	0.0001	0	0	0	0	0.0002	0.00006	0	0.5	0	0.4991

Table 2: Feature weights learned by all four methods from ball sorting data. ϕ_1 - ϕ_4: reward for sorting ball types 1-4 into bin 1, ϕ_5 - ϕ_8: reward for sorting ball types 1-4 into bin 2, and ϕ_9 - ϕ_{12}: reward for handling ball types 1-4 gently.

Method	Ball type hard and black	Ball type soft and red	Ball type hard and white	Ball type soft and green
MaxEnt IRL	To Bin 1 firmly	To Bin 1 firmly	To Bin 2 firmly	To Bin 2 firmly
IRL*	To Bin 1 gently	To Bin 1 gently	To Bin 2 gently	To Bin 2 firmly
Hidden Data EM	To Bin 1 firmly	To Bin 1 gently	To Bin 2 firmly	To Bin 2 gently
Occlusion-free control	To Bin 1 firmly	To Bin 1 gently	To Bin 2 firmly	To Bin 2 gently

Table 3: Optimal policies learned by all four methods from the ball sorting data with hidden variables. While the state consists of multiple variables, we show the robot's action map for the main state variable of ball type here. Underlined actions are erroneous.

Boularias et al. [15] rather than over trajectories. More importantly, it limits the optimization to the observed portions of the trajectories only. Consequently, the Lagrangian gradient becomes undefined and the method resorts to performing optimization using a technique that does not utilize the gradient. In comparison, we seek to infer expected data for portions of the trajectory that are occluded and the use of EM allows us to continue using the Lagrangian gradient for descent as shown by Wang et al. [7].

Few approaches investigate relaxing other knowledge requirements of IRL. Boularias et al. [16] propose model-free IRL with a single expert, learning the reward function by minimizing the relative entropy between distributions over trajectories generated by a baseline and target policies. Importance sampling is used for computing the relative entropy and the reward function reflects greedy maximization at each state in the absence of a transition function. In contrast, Bogert and Doshi [11] present a method that explicitly first learns the transition function under occlusion, making it a semi-model based method, before moving to IRL.

6. CONCLUDING REMARKS

Motivated by real-world applications in robotics, we focused on inverse reinforcement learning with partially occluded trajectories by treating the missing data as hidden variables. We developed an expectation-maximization based method to solve this problem and examined its performance in a UAV reconnaissance domain. Our approach improves on existing methods, which simply ignored the missing information, by enabling learning in the presence of occlusion that may even be dynamic; this is critical for real-world scenarios where the learner or environment is dynamic. Additionally, we demonstrated the value of our approach in an apprenticeship learning from demonstration application in which a critical component of the task being demonstrated is naturally hidden from the learner. In future work we will continue relaxing the data requirements of IRL, such as focusing on noisy-data scenarios where the learner observes the subject's state or action with sensor noise; this is common with robots.

Acknowledgments

This research is supported in part by a grant from NSF IIS-0845036 and a grant from ONR N000141310870. This paper benefited from conversations with Brian Ziebart.

APPENDIX

All four methods utilized the data collected from the experiment that involved humans sorting the balls. Recall that I's reward function contained 12 feature functions as described in Section 4.1. We show the weights learned for these 12 feature functions by all four methods in Table 2. Weights of features that should be minimized are ϕ_3 to ϕ_6, ϕ_9 and $\phi_1 1$. All methods give low weights to these. Among the rest, ϕ_{10} and ϕ_{12} are most important and the reference method Occlusion-free control gives high weight to these.

In Table 3 we also show broadly the policies obtained from the MDPs with the reward functions learned by all four methods. It provides key insight into the observed performances of the various methods.

REFERENCES

[1] Stuart Russell. Learning agents for uncertain environments (extended abstract). In *Conference on Learning Theory (COLT)*, pages 101–103, 1998.

[2] Brenna D. Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, May 2009.

[3] Andrew Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In *International Conference on Machine Learning (ICML)*, pages 663–670, 2000.

[4] Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *Twenty-Second Conference on Artificial Intelligence (AAAI)*, pages 1433–1438, 2008.

[5] Krishnamurthy Dvijotham and Emanuel Todorov. Inverse optimal control with linearly-solvable MDPs. In *International Conference on Machine Learning (ICML)*, pages 335–342, 2010.

[6] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Ser. B*, 39(1):1–38, 1977.

[7] Shaojun Wang, Dale Schuurmans, and Yunxin Zhao. The latent maximum entropy principle. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(2):8:1–8:42, 2012.

[8] Henryk Gzyl. *The Method of Maximum Entropy*. World Scientific, 1995.

[9] Jyrki Kivinen and Manfred K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–63, 1997.

[10] Kenneth Bogert and Prashant Doshi. Multirobot inverse reinforcement learning with interactions under occlusion. In *Autonomous Agents and Multi-Agent Systems Conference (AAMAS)*, pages 173–180, 2014.

[11] Kenneth Bogert and Prashant Doshi. Toward estimating others' transition models under occlusion for multi-robot irl. In *24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1867–1873, 2015.

[12] J Choi and Kee-eung Kim. Inverse reinforcement learning in partially observable environments. *Machine Learning Research*, 12:691–730, 2011.

[13] RW Obermayer and Frederick A Muckler. *On the inverse optimal control problem in manual control systems*, volume 208. NASA, 1965.

[14] Pieter Abbeel and AY Ng. Apprenticeship learning via inverse reinforcement learning. In *International Conference on Machine Learning (ICML)*, page 1, 2004.

[15] Abdeslam Boularias, O Krömer, and J Peters. Structured apprenticeship learning. *Machine Learning and Knowledge Discovery in Databases*, pages 227–242, 2012.

[16] A Boularias, Jens Kober, and J Peters. Relative entropy inverse reinforcement learning. In *International Conference on AI and Statistics (AISTATS)*, pages 182–189, 2011.