Toward Estimating Others' Transition Models Under Occlusion for Multi-Robot IRL

Kenneth Bogert and Prashant Doshi
THINC Lab, Department of Computer Science
University of Georgia, Athens, GA 30602
{kbogert,pdoshi}@uga.edu

Abstract

Multi-robot inverse reinforcement learning (mIRL) is broadly useful for learning, from observations, the behaviors of multiple robots executing fixed trajectories and interacting with each other. In this paper, we relax a crucial assumption in IRL to make it better suited for wider robotic applications: we allow the transition functions of other robots to be stochastic and do not assume that the transition *error* probabilities are known to the learner. Challenged by occlusion where large portions of others' state spaces are fully hidden, we present a new approach that maps stochastic transitions to distributions over features. Then, the underconstrained problem is solved using nonlinear optimization that *maximizes entropy* to learn the transition function of each robot from occluded observations. Our methods represent significant and first steps toward making mIRL pragmatic.

1 Introduction

We seek to learn the individual and joint behaviors of multiple robots executing fixed trajectories in a common space from observations. This is useful in various applications including a robot penetrating simple perimeter patrols; joining an ad hoc robot team [MacAlpine *et al.*, 2014]; and regulating traffic [Natarajan *et al.*, 2010]. Two characteristics make the learning challenging: the multiple observed robots situated in a common space may interact and their trajectories could be partially occluded from the learner. We study this problem in the context of an application setting involving two mobile robots executing simple cyclic trajectories for perimeter patrolling. Both robots' patrolling motions are disturbed when they approach each other in narrow corridors leading to an interaction. A subject robot observes them from a *hidden* vantage point that affords partial observability of their trajectories only. It's task is to penetrate the patrols and reach a goal location without being spotted. Thus, its eventual actions do not impact the other robots.

Inverse reinforcement learning (IRL) [Russell, 1998; Ng and Russell, 2000] is well suited as a starting point here because the task is to learn the preferences of passively-observed experts from their state-action trajectories. Previously, Bogert and Doshi [2014] models each observed robot in the setting as guided by a policy from a Markov decision process (MDP) and utilizes IRL generalized for occlusion. However, the interactions between the patrollers must be modeled as well. As these are sparse and scattered, the robots are modeled as playing a game at each point of interaction. Consequently, this method labeled mIRL*+Int generalizes IRL – so far limited to single-expert contexts – to multiple experts exhibiting sparse interactions and whose trajectories are partially occluded from the learner.

Key model assumptions of popular IRL methods are that the expert's stochastic transition function is completely known to the learner as in apprenticeship learning [Abbeel and Ng, 2004] and in Bayesian IRL [Ramachandran, 2007]. Alternately, the transition function is effectively deterministic and thus is easily approximated from the observed trajectories as in entropy maximization [Ziebart and Maas, 2008] with the assumption that transition randomness has a limited effect on the final behavior. The prior knowledge requirement is often difficult to satisfy in practice, for example, in scenarios that are not cooperative such as the patrolling application. Alternately, the supposed impotency of transition errors is a strong assumption in the context of robots.

Motivated by these substantive limitations of existing IRL methods, this paper makes the following contributions. We partially relax IRL's prior knowledge requirements and tread a middle path: we limit to those settings where a mobile robot's stochastic transition function may be viewed as composed of a deterministic core perturbed by transition error probabilities that make it stochastic. Given a state-action pair, the learner knows the intended next state of each expert. However, the transition error probabilities are unknown. Of course, the learner may learn the complete transition functions using supervised learning if it observes the experts fully and long enough. But, partial occlusion and a finite observation time motivate prudence and sophisticated methods.

Challenged by occlusion, we present mIRL$^*_{/T}$+Int a novel method that is based on the key insight that different transitions share underlying component features, and features that are associated with observed state-action pairs may transfer information to transitions in occluded portions. Subsequently, mIRL$^*_{/T}$+Int maps each state-action pair to a subset of features, which themselves are not perfectly predictable. In robots, these features could represent the physical components involved in the action, e.g., wheel rotations. Thus, the probability of success of an action in a state resulting in the intended next state is the joint probability of success of all features involved in that action. Our task reduces to finding the probability of success of each feature from observations that, importantly, do not inform each feature but instead pertain to *feature aggregates*.

2 Background: Multi-Robot IRL

IRL [Russell, 1998] seeks to find the most likely policy, π_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.* This assumption is strong and is seldom met in noncooperative and other contexts.

As the space of possible reward functions is very large, the function is commonly expressed as a linear combination of $K > 0$ *feature functions*, $R_I(s, a) = \sum_K \theta_k \cdot \phi_k(s, a)$, 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 policy. The k^{th} feature expectation for a learned policy, π_I, is, $\sum_s \mu_{\pi_I}(s) \cdot \phi_k(s, \pi_I(s))$. Here, $\mu_{\pi_I}(s)$ is the visitation frequency: the number of times state, s, is visited on using policy, π_I. The expectations are compared with those of the expert's from its observed trajectory, $\hat{\phi}_k = \sum_{s,a \in traj} \phi_k(s, a)$. Let Π_I be the space of expert's policies of size $|A_I|^{|S|}$. IRL seeks to learn a policy, $\pi_I^* \in \Pi_I$ that minimizes the difference between the two expectations.

Maximum entropy method Multiple policies may match the observed feature expectations equally well. In order to resolve this ill-posed problem, the principle of *maximum entropy* is useful [Gzyl, 1995]. It maintains a distribution over the policies constrained to match the observed feature expectations while being noncommittal to any policy. Mathematically, we define the problem as a nonlinear optimization:

$$\begin{aligned}
&\max_{\Delta} \left(-\sum_{\pi \in \Pi_I} Pr(\pi_I) \, log Pr(\pi_I) \right) \\
&\textbf{subject to} \quad \sum_{\pi_I \in \Pi_I} Pr(\pi_I) = 1 \\
&\sum_{\pi_I \in \Pi_I} Pr(\pi_I) \sum_{s \in S} \mu_{\pi_I}(s) \phi_k(s, \pi_I(s)) = \hat{\phi}_k \qquad \forall k
\end{aligned} \tag{1}$$

Here, Δ is the space of all distributions, $Pr(\Pi_I)$. We may apply Lagrangian relaxation bringing both the constraints into the objective function and solving the dual approximately.

Previous applications of IRL such as learning from demonstrations [Argall *et al.*, 2009] and apprenticeship learning [Abbeel and Ng, 2004] focus on a single expert agent and assume the entire state and action spaces are observable. Bogert and Doshi [2014] generalizes it to contexts shared by multiple interacting robots and whose trajectories are *partially occluded*, and label the method as mIRL*+Int.

Let $Obs(S) \subseteq S$ be the subset of all states of the robot I that are observable. Policy π_I is then obtained by limiting the optimization in Eq. 1 to over the observable state space only because we are unable to compute the observed feature expectations, $\hat{\phi}_k$, for the occluded states. While feature expectations from the observed states could be statistically projected to the occluded states, this approach may not account for any disturbances in the trajectory.

In the context of $N \geq 2$ observed robots, mIRL*+Int ascribes a smaller, individual MDP to each robot. Focusing on contexts where interactions are sparse or scattered, actions at the interacting states are modeled separately and override those prescribed by the individual MDPs at the interacting states. Subsequently, a nonlinear program that combines the one in Eq.1 for each mobile robot and accounting for the occlusion is used for learning the policies of the robots. In particular, the second constraint of Eq. 1 is limited to observed states only, $Obs(S)$. As the Lagrangian gradient is undefined for the unobservable states, a numerical method that does not use the gradient function such as Nelder-Mead's simplex [1965] is more suitable when some states are occluded.

mIRL*+Int models the robots as playing a *game* at interacting states. Bogert and Doshi [2014] shows an example interaction game that models the coordination needed when two patrollers approach each other in a narrow corridor. Robots perform the joint action corresponding to a Nash equilibrium of the game. Computing the state-visitation frequency, μ_{π_n}, in Eq. 1 bifurcates into using the equilibrium profile if the state involves an interaction, otherwise the attributed policy is utilized. The challenges are that the interacting state may be occluded and multiple equilibria may exist. mIRL*+Int addresses these by empirically computing the state-visitation frequency by projecting the current policies, and allowing for candidate equilibria with the maximum likelihood.

3 Learning Others' Occluded Transitions

Contributing toward making IRL robust for real-world applications, we take the significant step of partially relaxing the assumption in mIRL*+Int that the stochastic transition function of each observed robot's MDP is fully known.

3.1 Transitions and Error Model

Let ψ: $S \times A \rightarrow S$ map an observed robot's transition from state, s, given action a to a particular next state, s'. The function, ψ, gives the intended outcome of each action from each state. We may view this as a *deterministic* transition function.

Of course, actions may not always generate their intended outcomes leading to small errors in the corresponding transitions. Furthermore, parts of the robot's trajectory may be occluded from the subject robot and the robot may be guided by a policy. Both these factors make it unlikely that the learning robot will observe every action in every state enough times to reliably compute the full transition function. *Therefore, we focus on learning the probability of transitioning to the intended state given a state-action pair for an observed robot I,* $T_I(s, a, \psi(s, a))$. The remaining probability mass, $1 - T_I(s, a, \psi(s, a))$, could be distributed uniformly among the states that are the intended outcomes of other actions given the state, or wholly assigned to a default error state.

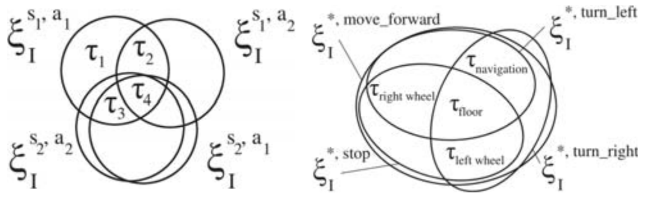
This approach requires that ψ is available to the learner (but not the probability with which $\psi(s, a)$ results). Alternately, the learner is able to construct an accurate ψ. Yet, this is a significant relaxation of the requirement by existing IRL methods that the full stochastic transition function including all probabilities is known. In domains involving mobile robots where actions are movement, intended next states may be determined accurately. Indeed, we follow this approach in our experimentation. Nevertheless, we are investigating ways of ascertaining ψ from observations given related effort [Ziebart and Maas, 2008] but which are cognizant of motion errors.

3.2 Mapping from Transition to Feature Subset

In order to learn robustly under occlusion, our approach is based on the following key observation:

Observation If transition probabilities are a function of underlying component outcome probabilities, then the observed trajectory may inform associated component probabilities. Subsequently, if some of these components are shared with transitions in occluded states, then information is transferred that facilitates obtaining occluded transition probabilities.

Motivated by this, we begin by mapping each state-action to a subset of lower-level transition features. Let $\xi_I^{s,a} = \{\tau_1, \ldots, \tau_k\}$ be the subset of independent features mapped to a state-action pair, $\langle s, a\rangle$, where each feature, $\tau \in \mathcal{T}_I$, is a binary random variable whose states are τ and $\bar{\tau}$. Here, $\mathcal{T}_I$ the set of all transition feature variables for robot I, is known – these are obtained from observing the type of robot I – and $\bigcup_{(s,a)} \xi_I^{s,a} = \mathcal{T}_I$. Figure 1 illustrates the above observation.



$\xi_I^{s_1, a_1}$
$\xi_I^{s_1, a_2}$
τ_1
τ_2
τ_3
τ_4
$\xi_I^{s_2, a_2}$
$\xi_I^{s_2, a_1}$
$\xi_I^{*, \text{move\_forward}}$
$\xi_I^{*, \text{turn\_left}}$
$\tau_{\text{navigation}}$
$\tau_{\text{right wheel}}$
τ_{floor}
$\tau_{\text{left wheel}}$
$\xi_I^{*, \text{stop}}$
$\xi_I^{*, \text{turn\_right}}$

Figure 1: Venn diagrams showing intersections between sets $\xi_I^{s_1,a_1}$, $\xi_I^{s_1,a_2}$, $\xi_I^{s_2,a_1}$ and $\xi_I^{s_1,a_2}$, and for a mobile robot. Observations informing transition features that are shared with other transitions facilitate a transfer of information to possibly occluded transitions.

Subsequently, define for a transition, $\langle s, a, \psi(s, a)\rangle$,

$$T_I(s, a, \psi(s, a)) \approx \prod_{\tau \in \xi_I^{s,a}} Pr(\tau) \tag{2}$$

The relationship between the feature random variables is ideally represented using a Bayesian network. As computing the joint of a Bayesian network takes exponential time in general and there are many transitions, mutually independent features offer tractability; this provides guidance on how to select features. Equation 2 is loosely analogous to using feature functions in the reward function with the difference that T_I is probabilistic due to the features being random variables.

To illustrate, consider the move_forward action of a robot on a smooth surface in a grid, which is enabled by its components such as all its wheels rotating with the same velocity. A *differential drive* robot with 2 wheels could have the move_forward action from a state involving a smooth surface mapped to two features: left wheel rotating with some velocity on a smooth surface and right wheel rotating with the same velocity on the surface. Then, $T_I(s, \text{move\_forward}, \psi(s, \text{move\_forward}))$=$Pr$(left_wheel rotating on smooth surface) $\times$ Pr(right_wheel rotating on smooth surface). A damaged wheel decreases this probability.

3.3 Observed Probabilities

Equation 2 casts the problem of inversely learning the transition function as the problem of learning the distributions of the state-action features. *However, the challenge is that we may not be able to pinpoint the performance of the various features in the observed trajectory; rather we obtain* **aggregated** *empirical distributions.* An observed trajectory of length T is a sequence of state-action pairs, $\{\langle s, a\rangle^0, \langle s, a\rangle^1, \ldots, \langle s, \emptyset\rangle^T\}$, where $\emptyset$ is the null action. From this, we obtain the probabilities of transitioning to the intended state given the previous state and action, denoted by $q^{\psi(s,a)}$, as simply the proportion of times the intended state is observed as the next state in the trajectory:

$$q_I^{\psi(s,a)} = \frac{\sum_{t=0:\langle s^t,a^t\rangle=\langle s,a\rangle}^{T-1} \delta(s^{t+1}, \psi(s^t, a^t))}{\sum_{t=0:\langle s^t,a^t\rangle=\langle s,a\rangle}^{T-1} 1}$$

where $\delta(\cdot,\cdot)$ is the indicator function that is equal to 1 when its two arguments are equal, otherwise 0.

Consider a simple trajectory, $\{\langle s_1, \text{move\_forward}\rangle, \langle s_2, \text{turn\_around}\rangle, \langle s_2, \text{move\_forward}\rangle, \langle s_1, \text{turn\_around}\rangle, \langle s_1, \text{move\_forward}\rangle, \langle s_3, \emptyset\rangle\}$, if s_2 is the intended state from s_1 on performing action, move_forward, then $q_I^{\psi(s_1, \text{move\_forward})} = 1/2$. Of course, a trajectory observed over a longer time leads to more accurate probabilities.

Notice that the probability, $q_I^{\psi(s,a)}$, obtained from an observed trajectory is equivalent to $T_I(s, a, \psi(s, a))$. Consequently, Eq. 2 relates the observed probability as obtained above to the transition features,

$$\prod_{\tau \in \xi_I^{s,a}} Pr(\tau) = q_I^{\psi(s,a)} \tag{3}$$

While $\xi_I^{s,a}$ tells us which features are assigned to each state-action and Eq. 3 constrains the feature distributions, *we arrive at an ill-posed problem where there could be many possible feature distributions satisfying the observed transition probabilities that serve as aggregates.*

3.4 Underconstrained Optimization

One way to make progress in an underconstrained problem is to again utilize the principle of maximum entropy optimization [Gzyl, 1995] because it makes the least assumptions beyond the problem formulation. In this context, mIRL$^*_{/T}$+Int maximizes the sum total entropy of all feature distributions. Constraints for this nonlinear optimization problem are given by Eq. 3 for each state-action pair present in the observed trajectory. The novel nonlinear optimization problem for finding the transition feature distributions of N robots is:

$$\max_{\Delta_1,\ldots,\Delta_N} \left(-\sum_{n=1}^{N} \sum_{\tau \in \mathcal{T}_n} Pr(\tau) log\, Pr(\tau) + Pr(\bar{\tau}) log\, Pr(\bar{\tau})\right)$$

subject to

$$\begin{array}{ll} \sum_{\tau \in \xi_n^{s,a}} log\, Pr(\tau) = log\, q_n^{\psi(s,a)} & \forall \langle s, a\rangle \in Obs(S) \times Obs(A) \\ & n = 1 \ldots N \\ Pr(\tau) + Pr(\bar{\tau}) = 1 & \forall \tau \in \mathcal{T}_n, n = 1 \ldots N \end{array} \tag{4}$$

The Lagrangian relaxation of the above optimization problem introduces vectors of Lagrangian multipliers, $\mathbf{v}$ and $\boldsymbol{\lambda}$, and the relaxed objective function is:
$$\mathcal{L}(\mathcal{T}_1, \ldots, \mathcal{T}_N, \mathbf{v}, \boldsymbol{\lambda}) = \Big(-\sum_{n=1}^{N} \sum_{\tau \in \mathcal{T}_n} Pr(\tau) log\ Pr(\tau) + Pr(\bar{\tau}) log\ Pr(\bar{\tau})) + \sum_{n=1}^{N} \sum_{j=1}^{|Obs(S)||Obs(A)|} v_{n,j} \Big(\Big(\sum_{\tau \in \xi_n^{s,a}} log\ Pr(\tau)\Big) - log\ q_n^{\psi(s,a)}\Big) + \sum_{n=1}^{N} \sum_{\tau \in \mathcal{T}_n} \lambda_{n,i} ((Pr(\tau) + Pr(\bar{\tau})) - 1)$$
Vector $\mathbf{v}$ has as many elements as the number of state-action pairs and observed robots. In practice, multiple state-action pairs for a robot may map to the same set of feature variables. In other words, $\xi_n^{s,a}$ for these state-action pairs is the same. Thus, we need not distinguish between these state-action pairs above, and we may obtain a single observed probability for the feature subset by summing $q_n^{\psi(s,a)}$ for the individual $\langle s, a \rangle$ pairs.

To properly handle saddle points in the Lagrangian function we take the sum of squares of the partial derivatives of $\mathcal{L}$ to arrive at the final objective function, $\mathcal{L}'$. Next, we minimize $\mathcal{L}'$ by using the penalty approach with Broyden-Fletcher-Goldfarb-Shanno (BFGS) [Byrd *et al.*, 1995] unconstrained gradient descent technique.

Occlusion Previously unseen actions could have been performed in the occluded portions of other robot's trajectory. Nevertheless, these actions map to feature variables in $\mathcal{T}_I$. *As some of the features in $\mathcal{T}_I$ are factors in observed actions, we may obtain (partially) informed transition distributions for the unseen actions as well under the maximum entropy principle.* This observation highlights the advantage of considering feature variables. This projection into the unobservable portion of the state and action space is justified by the maximum entropy method: over all possible distributions taking into account all available information the one with the maximum entropy is expected to be the least wrong.

3.5 Estimating Full Transition Function

mIRL$^*_{/T}$+Int solves the nonlinear optimization to obtain feature distributions that maximize the total entropy. Equation 2 then allows us to compute $T_I(s, a, \psi(s, a))$ for each state-action pair. In order to estimate the complete transition function, we need the probability of reaching unintended states due to action errors as well. The mass $1 - T_I(s, a, \psi(s, a))$ could be uniformly distributed among the intended states that would result due to performing actions other than a from s. Subsequently, the full transition function of other robot, I, is obtained as:

$$T_I(s^t, a^t, s^{t+1}) = \delta(s^{t+1}, \psi(s^t, a^t)) \prod_{\tau \in \xi_I^{s^t, a^t}} Pr(\tau) + \sum_{a' \in A_I : a' \neq a^t} \delta(s^{t+1}, \psi(s^t, a')) \frac{1 - \prod_{\tau \in \xi_I^{s^t, a^t}} Pr(\tau)}{|A_I| - 1} \tag{5}$$

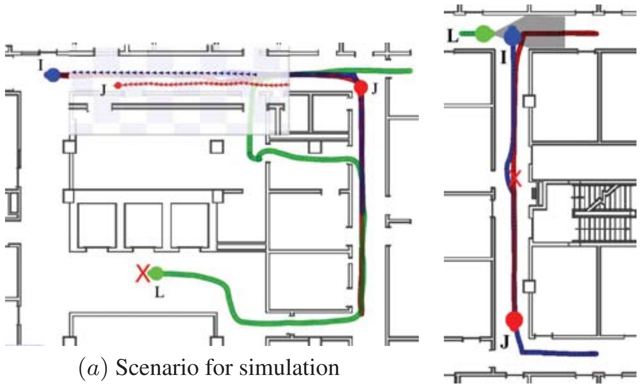
where $\delta(\cdot, \cdot)$ is an indicator function. In Eq 5, if state s^{t+1} is the intended next state due to action a^t from state s^t, then the probability is $T_I(s^t, a^t, \psi(s^t, a^t))$, otherwise the probability is due to erroneously reaching s^{t+1} by performing one or more other actions, which is $\frac{1 - T_I(s^t, a^t, \psi(s^t, a^t))}{|A_I| - 1}$. Alternately, we may allocate the entire mass, $1 - T_I(s, a, \psi(s, a))$, to a default (error) state. The latter may well be the state that results from stopping because a mobile robot often stops and relocalizes itself after errant motion, for example.

3.6 Iterative Optimization

Given the estimated transition functions of others, mIRL$^*_{/T}$+Int next inversely learns their reward functions as briefly described in Section 2. mIRL$^*_{/T}$+Int thus collapses to mIRL*+Int, learning reward weights $\boldsymbol{\theta}$ and obtaining policies of the observed robots. Correctly identifying and labeling the state-action pairs in robot I's observed trajectory is crucial to the performance of IRL in general, and particularly under occlusion. mIRL$^*_{/T}$+Int iteratively improves on the initial identification, which is often inaccurate due to perception errors, by utilizing the learned policies of the observed robots. Specifically, in the next iteration, for each observed state of a robot I, we replace the perceived action by that prescribed by the policy or the equilibrium behavior in case of an interaction at the state. mIRL$^*_{/T}$+Int then utilizes these relabeled trajectories to learn new transition probabilities for the observed robots. This procedure iterates until the labels for the observed state-action pairs fixate.

4 Performance Evaluation

We evaluate mIRL$^*_{/T}$+Int in an application domain introduced by Bogert and Doshi [2014] involving mobile robots, I and J, patrolling hallways using cyclical trajectories as shown in Fig. 2. Learning robot, L, starts at a hidden vantage point with a limited field of view and must autonomously reach a goal state located in the hallways undetected by the patrollers. Thus, L's eventual actions do not impact I and J.



(a) Scenario for simulation

(b) For physical runs

Figure 2: Hallways of a building patrolled by I (in blue) and J (in red) with the start location of L inside a room looking out of an open door. The goal locations are marked with an 'X'. Area visible to L in the physical runs is shown in gray. This corresponds to just 14% of the state space with the rest occluded.

The learner, L, models both the patrollers as being guided by policies output from their respective MDPs, whose states are the cell decompositions of the space, $\langle x, y \rangle$, the discretized orientation of the robot, φ, and a time step dimension, t. The actions of the patrollers and the subject robot allow each to move forward one cell, stop, turn right or left while moving. As the actions all involve motion, the *intended* next states, ψ, of each patroller are readily determined. $\mathsf{mIRL}^*_{/T}$+Int allows for nondeterministic transition functions that model the noise in physical robot motion. Furthermore, a wheel of one of the patrollers is artificially damaged in our experiments. *A robust test of the performance of $\mathsf{mIRL}^*_{/T}$+Int is whether L can identify the patroller and which of its wheels is damaged.* The learner's transition function models the probability of any of its *own* action failing at 2.5%. Each patroller's reward function is modeled as a linear combination of weighted feature functions, as we mentioned previously. L's reward function is positive for the goal state regardless of the time step. Penalties for detections by a patroller get specified at run time after the patrollers have been observed and their policies learned. We may then project their trajectories forward in time and space thereby predicting its location at each time step. States at which the location of L is within visible distance of the patroller with both at the same time step are given a negative reward.

L utilizes the following independent binary feature random variables as part of $\mathcal{T}_I$ and $\mathcal{T}_J$:

1. *Rotate left wheel at specified speed*, used at all states and for all actions except stop;
2. *Rotate right wheel at specified speed*, used at all states and for all actions except stop;
3. *Navigation ability* that models the robot's localization and plan following capabilities in the absence of motion errors, used at all states and for all actions except stop.
4. *Floor slip*, used for all states and actions

$R(s,a)$ consists of the following binary feature functions as I and J's trajectories are simple cycles: *has moved*, *in upper hallway*, *in lower hallway*, and *turn around at state, s*. The true vector of weights, $\boldsymbol{\theta}$, in the patrollers' reward functions then rewards features that result in cyclic patrols.

4.1 Algorithm and Baselines

The learner utilizes the following algorithm to determine when it should start moving toward the goal: It observes the patrollers for a predetermined amount of time and uses the observed trajectories to first learn their transition functions, followed by using them in performing mIRL^*+Int for learning their reward functions under occlusion. The MDPs of the patrollers are then solved and the policies are obtained, which are projected forward in time and space. In the value function of L's MDP, future time steps (in the state) exhibiting a positive value at the starting location of L are indicative of finding a path that leads to the goal state without being detected by a patroller. If found, L executes the resulting optimal policy to move toward the goal. Of course, none of the states may obtain a positive value. If so, L continues to observe and project the future positions of the patrollers until a solution is found or the waiting time elapses resulting in failure.

A dynamic Bayesian network (DBN) [Charniak, 1991] provides a graphical way for modeling a robot's transition function. While several approaches exist for learning the network parameters from data, a popular method for learning from data with missing values – due to the occlusion – is expectation-maximization (EM) [Dempster *et al.*, 1977]. In this method, labeled as DBN_{EM}, robots I and J are each modeled using a two-step state-action-state DBN whose parameters are learned from the observed portions of the trajectories as data using EM.

Other baselines allow for a comprehensive analyses. The first knows the reward functions of patrollers including how they interact, with L acting accordingly. This approach either learns the transition function, labeled as Known $\mathsf{R}_{/T}$ and acts as an *upper bound*, or sets the transition function, Known R. Next comparator is Random, in which L ignores all observations, waits for a pre-specified observation duration, and then chooses a random time to start moving to the goal.

4.2 Experiments

Each robot in our simulations and physical experiments is a **TurtleBot** equipped with a Kinect, which provides a camera and an infrared ranging sensor. The bases include an iRobot Create or a Kobuki. Each robot also has a laptop running ROS Hydro on Ubuntu 12.04. A robot identifies another by detecting its unique color signature using CMVision's blob finder. ROS's default actuator and sensor models for the TurtleBot and the default local motion planner in move_base are used for navigation. Each robot localizes in a predefined map using the adaptive Monte Carlo localization available in ROS. The virtual simulations are performed in Stage. L is spotted if it is roughly within 6 cells of a patroller and the patroller faces it. We vary the starting locations of the patrollers across runs.

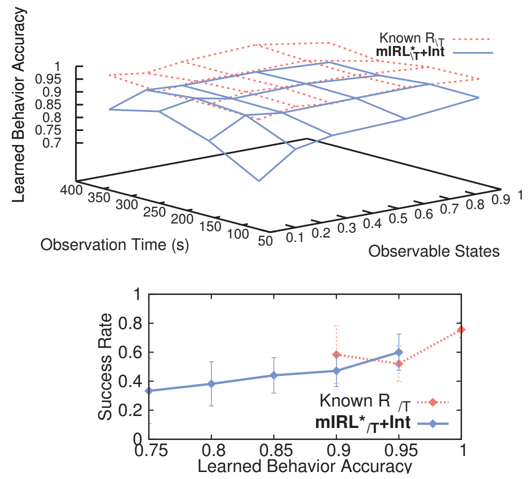
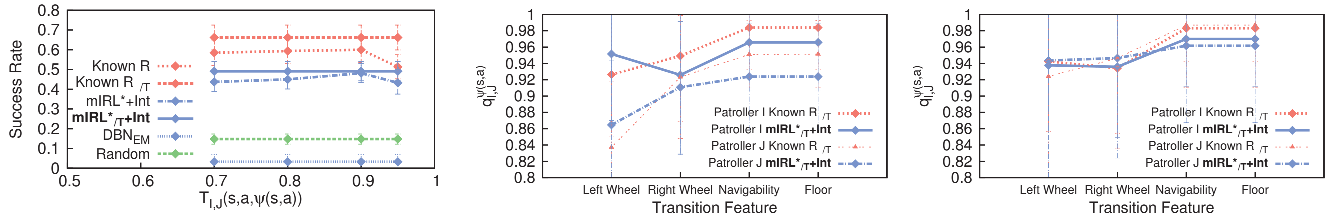


Figure 3: (**top**) Learned behavior accuracy of $\mathsf{mIRL}^*_{/T}$+Int and Known $\mathsf{R}_{/T}$ for different occlusion rates and observing times. (**bottom**) Improving accuracy of learned behavior correlates almost linearly with success rate. Vertical bars denote one std. dev.

Simulations

We first study the comparative impact of $\mathsf{mIRL}^*_{/T}$+Int on L's *success rate* in simulation. This is the proportion of runs in which L reaches the goal state unspotted by a patroller. This metric comprehensively measures the performance of all aspects of the approach and has practical implications. It is indicative of whether the observation in Section 3.2 is valid, the



(a) (b) (c)

Figure 4: (a) Comparative success rates of L for various methods that either learn T_I and T_J or fix it arbitrarily. True transition probabilities of the patrollers are not known. (b) Transition feature probabilities that correctly identify that the left wheel of J is partially damaged as indicated by its comparatively low success probability. (c) Transition feature probabilities when both patrollers are operating properly.

accuracy of the learned transition function and patroller policy under occlusion, L's simulated prediction accuracy, and the ability of L to traverse a route within an allotted time. Another key metric is the *learned behavior accuracy*, which is the proportion of all states at which the actions prescribed by the inversely learned policy of the patroller coincide with their actual actions. This metric permits focus on the learning in mIRL$^*_{/T}$+Int.

We begin by evaluating the learned behavior accuracy of mIRL$^*_{/T}$+Int as a function of the degree of observability and observing time, in Fig. 3. The degree is the proportion of all (x, y) cells in the state space that are visible to L; its complement gives a measure of the occlusion. Known R$_{/T}$ provides an artificial upper bound as we show. Each data point is the average of 200 simulated runs. Expectedly, the accuracy of mIRL$^*_{/T}$+Int improves with both observability and time. As observability improves, the optimization has more information and relies less on maximizing the entropy of the distributions. Furthermore, behavior accuracy correlates positively with success rate that reaches up to **60%** for mIRL$^*_{/T}$+Int.

We now consider the scenario where J's left wheel is slightly damaged unknown to L. The partially damaged patroller is slowed down, and L's overall success rate should improve. However, the latter depends on whether the damaged patroller is correctly identified by L, and whether the damaged wheel is "seen" in the feature probabilities. Figure 4(a) answers this important question: What is the benefit of learning the transition distributions of the patrollers? Observe that mIRL$^*_{/T}$+Int leads to a success rate that is greater than assuming arbitrary transition error distributions for both I and J (and then using mIRL*+Int), and is equal when a success probability of 0.9 is assumed. This difference is amplified on comparing Known R$_{/T}$ with Known R. Furthermore, DBN$_{EM}$ results in a poor success rate that is below Random. High occlusion to the extent that some actions are never seen poses a difficult challenge for the EM. The percentage of all runs during which the DBN$_{EM}$ could not find any state with a positive value (and therefore did not even attempt a penetration) is 100% under occlusion, which falls to about 90% when there is no occlusion. In comparison, this timed-out failure rate is under 4% for the other techniques regardless of occlusion showing that transition features that are shared between actions facilitate robust learning.

In Fig. 4(b), we show learned transition feature probabilities for both patrollers, $q_I^{\psi(s,a)}$ and $q_J^{\psi(s,a)}$, averaged over all state-action pairs. mIRL$^*_{/T}$+Int correctly learns patroller J's feature probabilities – comprising left and right wheel rotation, navigability and floor state – that are all lower than those of I. Importantly, the left wheel's success probability is significantly lower than that of the right wheel (Student's 2-tailed t-test, $p \ll 0.001$) – whose probability coincides with that of a properly working wheel – thereby correctly identifying that left wheel is damaged. This partial failure negatively impacts other features such as navigability.

Next, we consider the scenario where none of the patrollers' wheels are damaged. Figure 4(c) reports the learned transition feature probabilities for both patrollers where the error is small and coincide for both patrollers, as we may expect. These learned probabilities are close to the upper bound provided by Known R$_{/T}$. The latter is relevant as the shown feature probabilities are obtained from a subsequent iteration after an initial optimization.

Physical robots

We evaluate mIRL$^*_{/T}$+Int using physical robots in a smaller hallway environment (Fig. 2(b)). From its vantage point, L can observe approximately just **14%** of the state space corresponding to the lower observability setting in the previous simulations. The TurtleBots detect each other using color blob finders which are calibrated to match the detection range of the simulated robots. A successful run occurs when L reaches its goal without being detected. Failed runs may be caused by: I or J detecting L or 25 minutes have passed.

Method	**Success rate**	
	***J*'s left wheel damaged**	**No damaged wheels**
mIRL$^*_{/T}$+Int	*0.60*	*0.50*
mIRL*+Int	0.50	0.40
Random	0.40	0.20

Table 1: L's success rates using various methods over 10 *physical runs* each. L suffers from very high occlusion of patrollers.

We experiment with mIRL$^*_{/T}$+Int, mIRL*+Int fixing a transition success rate of 0.9, and Random in both scenarios: when J's left wheel is artificially damaged thereby slowing it down and creating an uneven trajectory, and when the patrollers are operating properly. We report the success rates of the physical runs in Table 1. Each data point is the average of 10 runs in the real environment. Note the success rate increase when J's wheel is damaged due to more opportunities to reach the goal if J's motion can be correctly predicted. A demonstration of the physical runs may be viewed at `https://youtu.be/X0HymCtjYh0`.

5 Concluding Remarks

Few other approaches investigate relaxing prior knowledge requirements of IRL. Boularias et al. [2011] 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. In contrast, we explicitly first learn the transition function under occlusion making this a first semi-model based method. We demonstrated its use by a robot with a limited field of view tasked with penetrating simple patrols by two other robots – it showed a success rate of 1 in 2 or more. As future work, we seek to continue relaxing the prior knowledge requirements that IRL places on its learners.

We focused on settings where interactions between robots are sparse and scattered. This allows observed robots to be individually modeled as tractable MDPs. Consequently, the transition probabilities of the multiple observed robots are assumed to be conditionally independent of each other; we may assume locality. Of course, the problem becomes a joint decentralized MDP [Goldman and Zilberstein, 2008] when interactions are pervasive and extended, and whose solution is usually highly intractable. Locality is relaxed by allowing joint actions in T_I and ξ_I thereby mapping state and joint actions to features.

Finally, Bard [1980] used maximum entropy to estimate state probabilities when event probabilities, which are aggregation of states, are known. This is challenging because there are fewer event probabilities than the number of unknown state probabilities. $\mathsf{mIRL}^*_{/T}$+Int builds on Bard's approach with several additional challenges. These include our use of features that are random variables, trajectories that may contain several events (aggregate features), occlusion, and its novel extension toward learning transition probabilities.

Acknowledgments

This research is partially supported by NSF CAREER grant IIS-0845036 and by ONR grant N000141310870. Discussions with Brian Ziebart and feedback from seminar participants at the University of Illinois at Chicago and the University of Michigan at Ann Arbor are gratefully acknowledged.

References

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

[Argall *et al.*, 2009] 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.

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

[Bogert and Doshi, 2014] 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.

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

[Byrd *et al.*, 1995] R. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

[Charniak, 1991] Eugene Charniak. Bayesian networks without tears. *AI Magazine*, Winter:51–63, 1991.

[Dempster *et al.*, 1977] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Roy. Statist. Soc. Ser. B*, 39(1):1–38, 1977.

[Goldman and Zilberstein, 2008] Claudia Goldman and Shlomo Zilberstein. Communication-based decomposition mechanisms for decentralized MDPs. *Journal of Artificial Intelligence Research*, 32:169–202, 2008.

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

[MacAlpine *et al.*, 2014] P. MacAlpine, Katie Genter, Samuel Barrett, and Peter Stone. The RoboCup 2013 drop-in player challenges: A testbed for ad hoc teamwork. In *Autonomous Agents and Multi-Agent Systems Conference*, pages 1461–1462, 2014.

[Natarajan *et al.*, 2010] S. Natarajan, G. Kunapuli, K. Judah, P. Tadepalli, K. Kersting, and J. Shavlik. Multi-agent inverse reinforcement learning. In *International Conference on Machine Learning and Applications (ICMLA)*, pages 395–400, 2010.

[Nelder and Mead, 1965] J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer*, 7(4):308–313, 1965.

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

[Ramachandran, 2007] Deepak Ramachandran. Bayesian inverse reinforcement learning. In *International Joint Conference on AI (IJCAI)*, pages 2586–2591, 2007.

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

[Ziebart and Maas, 2008] BD Ziebart and Andrew Maas. Maximum entropy inverse reinforcement learning. In *Twenty-Second Conference on Artificial Intelligence (AAAI)*, pages 1433–1438, 2008.