# Relative Entropy Inverse Reinforcement Learning

**Abdeslam Boularias** **Jens Kober** **Jan Peters**
Max-Planck Institute for Intelligent Systems
72076 Tübingen, Germany
{abdeslam.boularias,jens.kober,jan.peters}@tuebingen.mpg.de

## Abstract

We consider the problem of imitation learning where the examples, demonstrated by an expert, cover only a small part of a large state space. Inverse Reinforcement Learning (IRL) provides an efficient tool for generalizing the demonstration, based on the assumption that the expert is optimally acting in a Markov Decision Process (MDP). Most of the past work on IRL requires that a (near)-optimal policy can be computed for different reward functions. However, this requirement can hardly be satisfied in systems with a large, or continuous, state space. In this paper, we propose a model-free IRL algorithm, where the relative entropy between the empirical distribution of the state-action trajectories under a baseline policy and their distribution under the learned policy is minimized by stochastic gradient descent. We compare this new approach to well-known IRL algorithms using learned MDP models. Empirical results on simulated car racing, gridworld and ball-in-a-cup problems show that our approach is able to learn good policies from a small number of demonstrations.

## 1 Introduction

Modern robots are designed to perform complicated planning and control tasks, such as manipulating objects, navigating in outdoor environments, and driving in urban areas. Unfortunately, manually programming these tasks is generically an expensive as well as time-intensive process. An easier form of instruction is a key bottleneck in robotics. Markov Decision Processes (MDPs) provide efficient mathematical tools to handle such tasks with a little help from an expert. Here, the expert can define the task by simply specifying an informative reward function. However, in many problems, even the specification of a reward function is not always straightforward. In fact, it is frequently easier to demonstrate examples of a desired behavior than to define a reward function (Ng and Russell, 2000).

Learning policies from demonstrated examples, also known as imitation learning, is a technique that has been used to learn many tasks in robotics (Schaal, 1999). One can generally distinguish between direct and indirect imitation approaches (Ratliff et al., 2009). In direct methods, the robot learns a function that maps state features into actions by using a supervised learning technique (Atkeson and Schaal, 1997). Despite the remarkable success of many systems built on this paradigm (Pomerleau, 1989), direct methods are generally suited for learning reactive policies, where the optimal action in a given state depends on local features, without taking future states into account.

To overcome this drawback, Abbeel and Ng (2004) introduced a new indirect imitation learning approach known as apprenticeship learning. The aim of apprenticeship learning is to recover a reward function under which the expert's policy is optimal, rather than to directly mimic the actions of the expert. The learned reward function is then used to find an optimal policy. The key idea behind apprenticeship learning is that the reward function is the most succinct hypothesis for explaining a behavior. Consequently, an observed behavior can be better generalized when the reward function is known. The process of recovering a reward function from a demonstration is known as Inverse Reinforcement Learning (IRL).

Many IRL methods are based on the strong assumption that an MDP model of the system is either given *a priori* or can be accurately learned from the demonstrated trajectories. For instance, (Abbeel and Ng, 2004) minimizes the worst-case loss in the value of

Appearing in Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS) 2011, Fort Lauderdale, FL, USA. Volume 15 of JMLR: W&CP 15. Copyright 2011 by the authors.

the learned policy compared to the expert's one. The proposed algorithm proceeds iteratively by finding the optimal policy of an MDP at each iteration. Similarly, the Maximum Margin Planning (MMP) algorithm, proposed by (Ratliff et al., 2006), consists in minimizing a cost function by a subgradient descent, where an MDP problem is solved at each step. The same assumption is considered in the Bayesian IRL approach (Ramachandran and Amir, 2007; Lopes et al., 2009), the natural gradient approach (Neu and Szepesvari, 2007), and the game-theoretic approach (Syed and Schapire, 2008). Along these lines, both the LPAL algorithm (Syed et al., 2008) and the Maximum Entropy algorithm (Ziebart et al., 2008, 2010) used an MDP model for calculating a probability distribution on the state-actions. A notable exception is the work of Abbeel et al. (2010) where a locally linear dynamics model of a helicopter was learned from demonstration.

However, most of these methods only need access to a subroutine for finding an optimal policy, which could be a model-free RL algorithm. Nevertheless, RL algorithms usually require a large number of time-steps before converging to a near-optimal policy, which is not efficient since the subroutine is called several times.

In this paper, we build on the Maximum Entropy framework (Ziebart et al., 2008, 2010) and introduce a novel model-free algorithm for Inverse Reinforcement Learning. The proposed algorithm is based on minimizing the relative entropy (KL divergence) between the empirical distribution of the state-action trajectories under a baseline policy and the distribution of the trajectories under a policy that matches the reward feature counts of the demonstration. We will show that this divergence can be minimized with a stochastic gradient descent that can be empirically estimated without requiring an MDP model. Simulation results show an improvement over model-based approaches in the quality of the learned reward functions.

## 2 Preliminaries

Formally, a Markov Decision Process (MDP) is a tuple $(\mathcal{S}, \mathcal{A}, T, R, d_0, \gamma)$, where $\mathcal{S}$ is a set of states and $\mathcal{A}$ is a set of actions. $T$ is a transition function with $T(s,a,s') = P(s_{t+1} = s' | s_t = s, a_t = a)$ for $s, s' \in \mathcal{S}, a \in A$, and $R$ is a reward function where $R(s,a)$ is the reward given for executing action $a$ in state $s$. The initial state distribution is denoted by $d_0$, and $\gamma$ is a discount factor. A Markov Decision Process without a reward function is denoted by MDP\R. We assume that the reward function $R$ is given by a linear combination of $k$ feature vectors $f_i$ with weights $\theta_i$ such that $\forall (s,a) \in \mathcal{S} \times \mathcal{A} : R(s,a) = \sum_{i=1}^{k} \theta_i f_i(s,a)$. A deterministic policy $\pi$ is a function that returns an action $\pi(s)$ for each state $s$. A stochastic policy $\pi$ is a probability distribution on the action to be executed in each state, defined as $\pi(s,a) = P(a_t = a | s_t = s)$. The expected return $J(\pi)$ of a policy $\pi$ is the expected sum of rewards that will be received if policy $\pi$ will be followed, i.e., $J(\pi) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | d_0, \pi, T]$. An optimal policy $\pi$ is one satisfying $\pi = \arg\max_\pi J(\pi)$. The expectation (or count) of a feature $f_i$ for a policy $\pi$ is defined as $f_i^\pi = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t f_i(s_t, a_t) | d_0, \pi, T]$. Using this definition, the expected return of a policy $\pi$ can be written as a linear function of the feature expectations $J(\pi) = \sum_{i=1}^{k} \theta_i f_i^\pi(s,a)$. One can also define the discounted sum of a feature $f_i$ along a trajectory $\tau = s_1 a_1, \dots s_H a_H$ as $f_i^\tau = \sum_t \gamma^t f_i(s_t, a_t)$. Therefore, the expected return of a policy $\pi$ can be written as $J(\pi) = \sum_{\tau \in \mathcal{T}} P(\tau | \pi, T) \sum_{i=1}^{k} \theta_i f_i^\tau$, where $\mathcal{T}$ is the set of trajectories.

## 3 Inverse Reinforcement Learning

In this section, we will quickly review the foundations of IRL, and subsequently, review Maximum Entropy IRL, which is the most related approach to ours.

### 3.1 Overview

The aim of apprenticeship learning is to find a policy $\pi$ that is at least as good as a policy $\pi^E$ demonstrated by an expert, i.e., $J(\pi) \geq J(\pi^E)$. However, the expected returns of $\pi$ and $\pi^E$ cannot be directly compared, unless a reward function is provided. As a solution to this problem, Ng and Russell (2000) proposed to first learn a reward function, assuming that the expert is optimal, and then use it to recover the expert's generalized policy. However, the problem of learning a reward function given an optimal policy is ill-posed (Abbeel and Ng, 2004). In fact, a large class of reward functions, including all constant functions for instance, may lead to the same optimal policy. Most of the IRL literature has focused on solving this particular problem. Examples of the proposed solutions include incorporating prior information on the reward function, minimizing the margin $\|J(\pi) - J(\pi^E)\|$, or maximizing the entropy of the distribution on state-actions under the learned policy $\pi$. This latter approach is known as Maximum Entropy IRL (Ziebart et al., 2008) and will be described in the following section.

### 3.2 Maximum Entropy IRL

The principle of maximum entropy states that the policy that best represents the demonstrated behavior is the one with the highest entropy, subject to the constraint of matching the reward value of the demonstrated actions. This latter constraint can be satisfied by ensuring that the feature counts of the learned policy match with those of the demonstration, i.e.,

$$\forall i \in \{1, \dots, k\} : \sum_{\tau \in \mathcal{T}} P(\tau|\pi, T) f_i^\tau = \hat{f}_i \tag{1}$$

where $\hat{f}_i$ denotes the empirical expectation of feature $i$ calculated from the demonstration. The approach of Ziebart et al. (2008) consists of finding the parameters $\theta$ of a policy $\pi$ that maximizes the entropy of the distribution on the trajectories subject to constraint (1). Solving this problem leads to maximizing the likelihood of the demonstrated trajectories under the following distribution

$$Pr(\tau|\theta, T) \propto d_0(s_1) \exp\left(\sum_{i=1}^{k} \theta_i f_i^\tau\right) \prod_{t=1}^{H} T(s_t, a_t, s_{t+1})$$

where $\tau = s_1 a_1, \dots s_H a_H$.

Note that this distribution was suggested as an approximation of a more complex one derived by using the principle of maximum entropy. Unfortunately, the likelihood function of the demonstrations cannot be calculated unless the transition function $T$ is known. To solve this problem, we propose a new method inspired by the Relative Entropy Policy Search (REPS) approach (Peters et al., 2010). We minimize the relative entropy between an arbitrary distribution on the trajectories and the empirical distribution under a baseline policy. We also bound the difference between the feature counts of the learned policy and those of the demonstration. Finally, we show how to efficiently approximate the corresponding gradient by using Importance Sampling.

## 4 Relative Entropy IRL

In this section, we propose a new approach that is based on REPS (Peters et al., 2010) and Generalized Maximum Entropy methods (Dudik and Schapire, 2006).

### 4.1 Problem Statement

We consider trajectories of a fixed horizon $H$, and denote by $\mathcal{T}$ the set of such trajectories. Let $P$ be a probability distribution on the trajectories of $\mathcal{T}$. Let $Q$ be the distribution on the trajectories of $\mathcal{T}$ under a baseline policy and the transition matrices $T^a$. Maximum Entropy IRL can be reformulated as the problem of minimizing the relative entropy between $P$ and $Q$,

$$\min_P \sum_{\tau \in \mathcal{T}} P(\tau) \ln \frac{P(\tau)}{Q(\tau)}, \tag{2}$$

subject to the following constraints

$$\forall i \in \{1, \dots, k\} : |\sum_{\tau \in \mathcal{T}} P(\tau) f_i^\tau - \hat{f}_i| \le \epsilon_i, \tag{3}$$

$$\sum_{\tau \in \mathcal{T}} P(\tau) = 1, \tag{4}$$

$$\forall \tau \in \mathcal{T} : P(\tau) \ge 0, \tag{5}$$

The thresholds $\epsilon_i$ can be calculated by using Hoeffding's bound. Given $n$ sampled trajectories, and a confidence probability $\delta$, we set

$$\epsilon_i = \sqrt{\frac{-\ln(1-\delta)}{2n}} \frac{\gamma^{H+1} - 1}{\gamma - 1} \Big( \max_{s,a} f_i(s,a) - \min_{s,a} f_i(s,a) \Big)$$

The parameter $\delta$ is an upper bound on the probability that the difference between the feature counts given the distribution $P$ and the true feature counts of the expert's policy is larger than $2\epsilon$.

Notice that the optimality of the expert's policy is not required. Moreover, the difference between the expected return of the expert and that of the policy extracted from the distribution $P$ is bounded by $\sum_i \epsilon_i |\theta_i|$, where $\{\theta_i\}$ are the reward weights.

### 4.2 Derivation of the Solution

The Lagrangian of this problem is given by (Dudik and Schapire, 2006)

$$L(P, \theta, \eta) = \sum_{\tau \in \mathcal{T}} P(\tau) \ln \frac{P(\tau)}{Q(\tau)} - \sum_{i=1}^{k} \theta_i \Big( \sum_{\tau \in \mathcal{T}} P(\tau) f_i^\tau - \hat{f}_i \Big) - \sum_{i=1}^{k} |\theta_i| \epsilon_i + \eta \Big( \sum_{\tau \in \mathcal{T}} P(\tau) - 1 \Big).$$

Due to the KKT conditions, we have

$$\begin{aligned} \partial_{P(\tau)} L(P, \theta, \eta) &= \ln\left(P(\tau)/Q(\tau)\right) - \sum_{i=1}^{k} \theta_i f_i^\tau + \eta + 1 \\ &= 0. \end{aligned}$$

Then

$$P(\tau) = Q(\tau) \exp\left( \sum_{i=1}^{k} \theta_i f_i^\tau - \eta - 1 \right).$$

Since $\sum_{\tau \in \mathcal{T}} P(\tau) = 1$, the normalization constant is determined by

$$\exp(\eta + 1) = \sum_{\tau \in \mathcal{T}} Q(\tau) \exp\left( \sum_{i=1}^{k} \theta_i f_i^\tau \right) \overset{def}{=} Z(\theta).$$

Therefore

$$P(\tau|\theta) = \frac{1}{Z(\theta)} Q(\tau) \exp\left( \sum_{i=1}^{k} \theta_i f_i^\tau \right) \tag{6}$$

The dual function resulting from this step is

$$g(\theta) = \sum_{i=1}^{k} \theta_i \hat{f}_i - \ln Z(\theta) - \sum_{i=1}^{k} |\theta_i| \epsilon_i.$$

The dual problem consists in maximizing $g(\theta)$, where $\theta \in \mathbb{R}^k$. The function $g$ is concave and differentiable everywhere except for $\theta_i = 0$, hence, it can be maximized by using a subgradient ascent. The subgradient is given by

$$\frac{\partial}{\partial \theta_i} g(\theta) = \hat{f}_i - \sum_{\tau \in \mathcal{T}} P(\tau|\theta) f_i^\tau - \alpha_i \epsilon_i \quad (7)$$

with $\alpha_i = 1$ if $\theta_i \geq 0$ and $\alpha_i = -1$ otherwise. The subgradient $\partial_{\theta_i} g(\theta)$ cannot be obtained analytically unless the transition function $T$, used for calculating $Q$ and $P$, is known. However, the lack of knowledge of $T$ is essential in many problems and the reason for the quest for model-free methods. In the remainder of this section, we present a simple method for empirically estimating the gradient. This method is based on sampling trajectories by following an arbitrary policy $\pi$, and then using Importance Sampling for approximating the gradient.

### 4.3 Gradient Estimation with Importance Sampling

The function $Q$ can be decomposed as $Q(\tau) = D(\tau)U(\tau)$ where $D(\tau) = d_0(s_1) \prod_{t=1}^{H} T(s_t, a_t, s_{t+1})$ is the joint probability of the state transitions in $\tau$, for $\tau = s_1 a_1, \ldots s_H a_H$, and $U(\tau)$ is the joint probability of the actions conditioned on the states in $\tau$. Therefore, Equation (6) becomes

$$P(\tau|\theta) = \frac{D(\tau)U(\tau) \exp\left(\sum_{i=1}^{k} \theta_i f_i^\tau\right)}{\sum_{\tau \in \mathcal{T}} D(\tau)U(\tau) \exp\left(\sum_{i=1}^{k} \theta_i f_i^\tau\right)}.$$

The term $\sum_{\tau \in \mathcal{T}} P(\tau|\theta) f_i^\tau$ in Equation (7) can be approximated given a set $\mathcal{T}_N^\pi$ of $N$ trajectories sampled by executing a given policy $\pi$ by using Importance Sampling. Thus, we can determine the sample-based gradient

$$\begin{aligned}
\frac{\hat{\partial g}}{\partial \theta_i}(\theta) &= \hat{f}_i - \frac{1}{N} \sum_{\tau \in \mathcal{T}_N^\pi} \frac{P(\tau|\theta)}{D(\tau)\pi(\tau)} f_i^\tau - \alpha_i \epsilon_i \\
&= \hat{f}_i - \frac{1}{N} \frac{\sum_{\tau \in \mathcal{T}_N^\pi} \frac{D(\tau)U(\tau) \exp\left(\sum_{j=1}^{k} \theta_j f_j^\tau\right)}{D(\tau)\pi(\tau)} f_i^\tau}{\sum_{\tau \in \mathcal{T}} D(\tau)U(\tau) \exp\left(\sum_{j=1}^{k} \theta_j f_j^\tau\right)} - \alpha_i \epsilon_i \\
&= \hat{f}_i - \frac{\frac{1}{N} \sum_{\tau \in \mathcal{T}_N^\pi} \frac{D(\tau)U(\tau) \exp\left(\sum_{j=1}^{k} \theta_j f_j^\tau\right)}{D(\tau)\pi(\tau)} f_i^\tau}{\frac{1}{N} \sum_{\tau \in \mathcal{T}_N^\pi} \frac{D(\tau)U(\tau) \exp\left(\sum_{j=1}^{k} \theta_j f_j^\tau\right)}{D(\tau)\pi(\tau)}} - \alpha_i \epsilon_i \\
&= \hat{f}_i - \frac{\sum_{\tau \in \mathcal{T}_N^\pi} \frac{U(\tau)}{\pi(\tau)} \exp\left(\sum_{j=1}^{k} \theta_j f_j^\tau\right) f_i^\tau}{\sum_{\tau \in \mathcal{T}_N^\pi} \frac{U(\tau)}{\pi(\tau)} \exp\left(\sum_{j=1}^{k} \theta_j f_j^\tau\right)} - \alpha_i \epsilon_i, \quad (8)
\end{aligned}$$

where $\pi(\tau) = \prod_{t=1}^{H} Pr(a_t|s_t)$, for $\tau = s_1 a_1, \ldots s_H a_H$.

## 5 Experiments

To validate our approach, we experimented on three benchmark problems. The first domain is a racetrack, the second one is a gridworld and the last benchmark corresponds to a toy known as the ball-in-a-cup problem. While the racetrack and gridworld problems are not meant to be challenging tasks, they allow us to compare our approach to other methods of generalizing the demonstrations. The first approach that we compare to is the model-based Maximum Entropy, where a transition matrix is learned from uniformly sampled trajectories. The second approach corresponds to a naive model-free adaptation of Maximum Margin Planning (MMP). At each step of the subgradient descent in MMP, a near-optimal policy is found by using the reinforcement learning algorithm SARSA, which requires a large number of additional sampled trajectories. We also compare these IRL methods to a simple classification algorithm where the action in a given state is selected by performing a majority vote on the $k$-nearest neighbor states where the expert's action is known. For each state, the distance $k$ is gradually increased until at least one state that appeared in the demonstration is encountered. The distance between two states corresponds to the shortest path between them with a positive probability.

The performance of different IRL methods can be compared by learning the optimal policies corresponding to the learned reward functions, and comparing the expected returns of these policies. However, such an approach would be biased by the algorithm and the parameters used for learning the policies. Therefore, we will use the accurate transition functions for finding the optimal policies corresponding to the learned reward functions.

### 5.1 Racetrack

We implemented a simplified car race simulator, the corresponding racetrack is shown in Figure 1. The states correspond to the position of the vehicle in the racetrack and its velocity. We considered two discretized velocities, low and high, in each direction of the vertical and horizontal axis, in addition to a zero velocity in each axis, leading to a total of 25 possible combinations of velocities and 5100 states. The controller can accelerate or decelerate in each axis, or do nothing. The controller cannot however combine a horizontal and a vertical action, the number of actions then is 5. When the velocity is low, acceleration/deceleration actions succeed with probability 0.9, and fail with probability 0.1, leaving the velocity

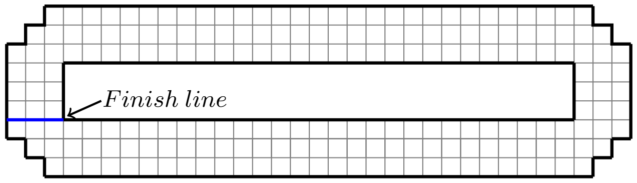


Figure 1: Configuration of the racetrack

unchanged. The success probability falls down to 0.2 when the velocity is high, making the vehicle harder to control. When the vehicle tries to move off the track, it remains in the same position and its velocity falls down to zero. The controller receives a reward of 0 for each step except for off-roads, where it receives $-1$, and for reaching the finish line, where the reward is 5. A discount factor of 0.99 is used in order to favor shorter trajectories. The vehicle starts from a random position on the start line, and the length of each demonstration trajectory is 40. The results are averaged over $10^3$ independent trials of length 50. There is a binary reward feature corresponding to the finish line and one for driving off-track, in addition to a feature with value 1 in all the states.

Figure 2 shows the average reward per time-step, the average number of steps required for reaching the finish line, and the average frequency of driving off the track. The results are a function of the number of sampled trajectories. For the IRL algorithms, there are only 10 demonstrations provided by an expert, the additional samples are those used for learning the transition function or the stochastic gradient (Equation 8, with a uniform sampling policy $\pi$), or for learning a policy in the case of MMP. In this latter case, the number of trajectories corresponds to the number of trials used by SARSA for each step of the subgradient descent. For $k$-NN, all the trajectories are provided by an expert.

In this experiment, both the model-based Maximum Entropy and the model-free Relative Entropy approaches learned a reward function close to the expert's one. Consequently, the policies found by using the corresponding reward functions achieve nearly optimal performances. In fact, the model-based algorithm was provided with the list of possible next states for each state and action, and the only unknown parameter was the success probability. Moreover, since the race always starts from the same line, a small number of sampled trajectories is sufficient for learning an accurate model of the dynamics.

We also notice the underperformance of the naive model-free MMP, caused by the large number of samples required for finding optimal policies by reinforcement. Finally, we remark that $k$-NN converges to an optimal policy after a 100 demonstrations. This result cannot be directly compared to the other methods, where only 10 trajectories correspond to demonstrations.

## 5.2 Gridworld

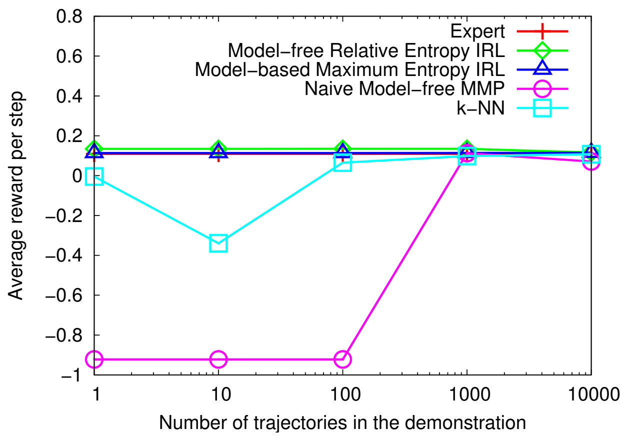
We consider a $50 \times 50$ gridworld. The state corresponds to the location of the agent on the grid. The agent has four actions for moving in one of the directions of the compass. The actions succeed with probability 0.7, a failure results in a uniform random transition to one of the adjacent states. A reward of 1 is given for reaching the goal state, located on the upper-right corner. For the remaining states, the reward function was randomly set to 0 with probability $2/3$ and to $-1$ with probability $1/3$. The initial state is sampled from a uniform distribution on the states. The discount factor is set to 0.99. We used only 10 demonstration trajectories for the IRL methods, and a variable number of demonstrations for $k$-NN. The duration of each trajectory is 100 time-steps, and the results are averaged over $10^3$ independent trials.

Figure 2(d) shows the average reward per time-step of the policies found by using different learning approaches. We notice that the average return of the model-based Maximum Entropy method is zero. In fact, the high reward associated to the goal state was not learned by this method. This was mainly caused by the uniform distribution on the initial state of each trajectory, which resulted in an inaccurate learned model. The reward function learned by the model-free approach was similar to the expert's one after only 10 uniformly sampled trajectories were used to estimate the stochastic gradient.

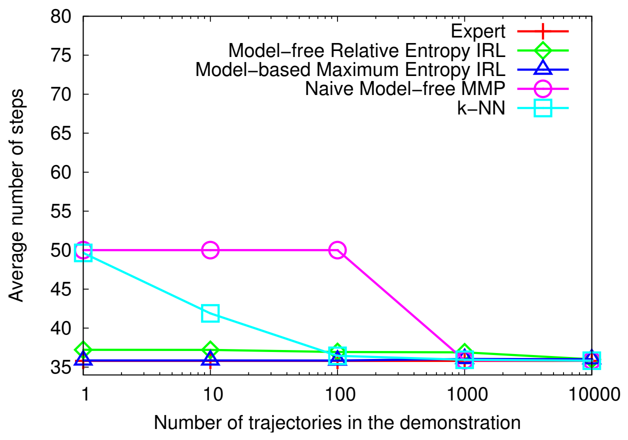
## 5.3 Ball-in-a-cup

As a final evaluation, we employ the model-free Relative Entropy method to recover the reward of the children's motor game ball-in-a-cup. The toy consists of a small cup and ball attached to its bottom by a string (see Figure 3). Initially, the ball is hanging below the cup and the goal of the game is to toss the ball into the cup by moving the cup. The state space consists of the Cartesian positions and velocities of the ball and the cup. The actions correspond to the Cartesian accelerations of the cup. Both the state and the action space are continuous. The dynamics of the system cannot be accurately described by a model. Therefore, model-based approaches will not be considered in this experiment.

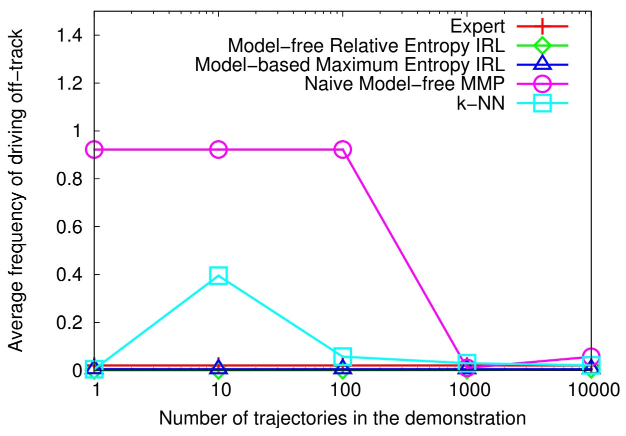
We recorded a total of 17 movements of the ball and the cup in a motion capture setup. These recordings



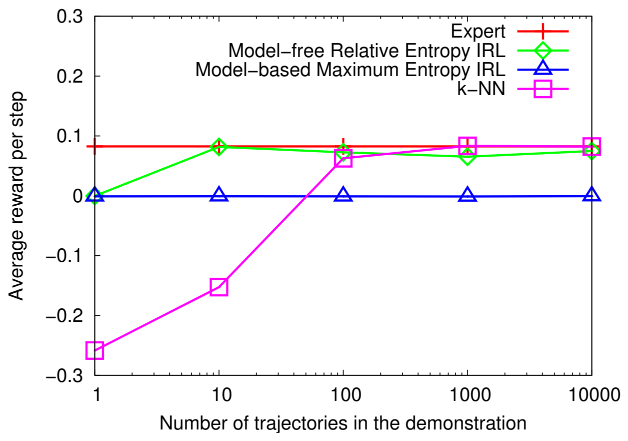
(a) Average reward in the racetrack



(b) Average number of time-steps per round



(c) Average frequency of driving off-track



(d) Average reward in the gridworld

Figure 2: Racetrack and gridworld results

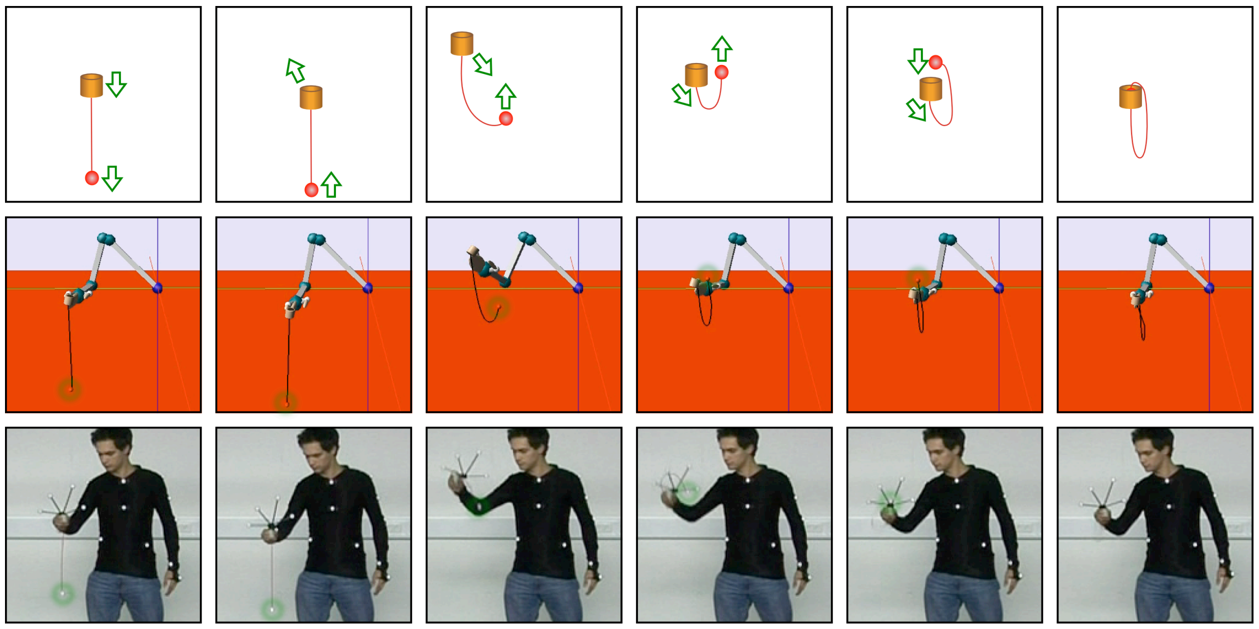


Figure 3: This figure shows schematic drawings of the Ball-in-a-Cup motion, the final learned robot motion as well as a motion-captured human motion. The green arrows show the directions of the momentary movements.

were filtered in order to make them consistent with our simulator. The inverse reinforcement learning algorithm is provided 10 of the 17 trials as expert demonstrations. These expert demonstrations result in successful trials, i.e., the ball is caught in the cup. As global policy search in this high dimensional continuous space is infeasible, the search algorithm employed to test the recovered reward functions relies on local optimization in the vicinity of an initialization. The remaining seven recorded movements were employed to initialize the policy search during the testing. We perturbed these trajectories slightly ensuring that the ball is not directly caught in the cup. As the “true” reward function is unknown, we use the proxy reward of ‘ball caught with the cup’ instead for evaluating the success of the learned trajectories. The reward features for this task consist of the continuous values of the absolute and relative positions and velocities of the ball and the cup as well as their squared values, the distance between the ball and the cup, the angle between the ball and the cup. Additionally, the binary feature “the ball is in the cup” is provided. These 29 features are additionally localized in time using 10 equispaced Gaussian weighting functions, resulting in a total feature count of 290.

Figure 4 illustrates the number of samples and the corresponding success rates of both the model-free Relative Entropy and the model-free MMP approaches for the ball-in-a-cup task. The results are averaged over three runs with the error bars indicating the standard deviation. The policy found by employing the learned reward function converged to a success rate of 100% after using only 1000 sampled trajectories in the case of the proposed method. A larger number of sampled trajectories is needed by the model-free variant of MMP.

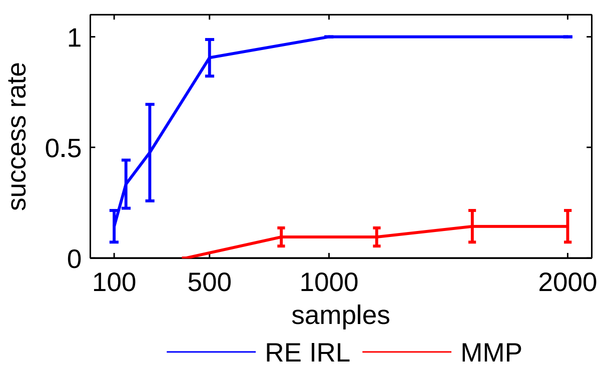


Figure 4: Ball-in-a-cup results. RE IRL refers to the model-free Relative Entropy IRL algorithm and MMP refers to the model-free MMP algorithm.

## 6 Conclusion

Apprenticeship Learning via Inverse Reinforcement Learning (IRL) provides an elegant solution to the problem of generalization in imitation learning. This approach consists in first learning a reward function that explains the observed behavior, and then using it for generalization. A strong assumption considered in IRL-based algorithms is that the dynamics model of the underlying MDP is known, or it can be learned from sampled trajectories.

In this paper, we showed that using inaccurate models, learned from a small number of samples, may lead to learning reward functions that are completely different from the true ones. Inspired by the work on Generalized Maximum Entropy and Relative Entropy Policy Search, we proposed a novel model-free IRL algorithm where the cost function is minimized with a stochastic gradient descent. Empirical results on simulated problems show that our algorithm is able to learn good policies from a small number of samples.

As a future work, we mainly plan to experiment on more difficult problems, and to explore other techniques of stochastic optimization.

## References

Abbeel, P., Coates, A., and Ng, A. Y. (2010). Autonomous helicopter aerobatics through apprenticeship learning. *International Journal of Robotics Research (IJRR)*, 29(13).

Abbeel, P. and Ng, A. Y. (2004). Apprenticeship Learning via Inverse Reinforcement Learning. In *Proceedings of the Twenty-first International Conference on Machine Learning (ICML'04)*, pages 1–8.

Atkeson, C. and Schaal, S. (1997). Robot Learning From Demonstration. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML'97)*.

Dudik, M. and Schapire, R. (2006). Maximum entropy distribution estimation with generalized regularization. In *Proceedings of the 19th Annual Conference on Learning Theory (COLT'06)*, pages 123–138.

Lopes, M., Melo, F., and Montesano, L. (2009). Active Learning for Reward Estimation in Inverse Reinforcement Learning. In *European Conference on Machine Learning (ECML'09)*, pages 31–46.

Neu, G. and Szepesvari, C. (2007). Apprenticeship Learning using Inverse Reinforcement Learning and Gradient Methods. In *Conference on Uncertainty in Artificial Intelligence (UAI'07)*, pages 295–302.

Ng, A. and Russell, S. (2000). Algorithms for Inverse Reinforcement Learning. In *Proceedings of the*

*Seventeenth International Conference on Machine Learning (ICML'00)*, pages 663–670.

Peters, J., Mulling, K., and Altun, Y. (2010). Relative Entropy Policy Search. In *Proceedings of the Twenty-Fourth National Conference on Articial Intelligence (AAAI'10)*.

Pomerleau, D. (1989). ALVINN: An Autonomous Land Vehicle in a Neural Network. In *Neural Information Processing Systems (NIPS'89)*, pages 769–776.

Ramachandran, D. and Amir, E. (2007). Bayesian Inverse Reinforcement Learning. In *Proceedings of The twentieth International Joint Conference on Artificial Intelligence (IJCAI'07)*, pages 2586–2591.

Ratliff, N., Bagnell, J., and Zinkevich, M. (2006). Maximum Margin Planning. In *Proceedings of the Twenty-third International Conference on Machine Learning (ICML'06)*, pages 729–736.

Ratliff, N., Silver, D., and Bagnell, A. (2009). Learning to Search: Functional Gradient Techniques for Imitation Learning. *Autonomous Robots*, 27(1):25–53.

Schaal, S. (1999). Is Imitation Learning the Route to Humanoid Robots? *Trends in Cognitive Sciences*, 3(6):233–242.

Syed, U., Bowling, M., and Schapire, R. E. (2008). Apprenticeship Learning using Linear Programming. In *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)*, pages 1032–1039.

Syed, U. and Schapire, R. (2008). A Game-Theoretic Approach to Apprenticeship Learning. In *Advances in Neural Information Processing Systems 20 (NIPS'08)*, pages 1449–1456.

Ziebart, B., Bagnell, A., and Dey, A. (2010). Modeling Interaction via the Principle of Maximum Causal Entropy. In *Proceedings of the Twenty-seventh International Conference on Machine Learning (ICML'10)*, pages 1255–1262.

Ziebart, B., Maas, A., Bagnell, A., and Dey, A. (2008). Maximum Entropy Inverse Reinforcement Learning. In *Proceedings of The Twenty-third AAAI Conference on Artificial Intelligence (AAAI'08)*, pages 1433–1438.