Journal of Zhejiang University-SCIENCE C (Computers & Electronics)
ISSN 1869-1951 (Print); ISSN 1869-196X (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Convergence analysis of an incremental approach to online inverse reinforcement learning*

Zhuo-jun JIN†, Hui QIAN†‡, Shen-yi CHEN, Miao-liang ZHU
(*School of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China*)
†E-mail: {jinzhuojun, qianhui}@zju.edu.cn

Received Jan. 9, 2010; Revision accepted Apr. 6, 2010; Crosschecked Dec. 6, 2010

Abstract: Interest in inverse reinforcement learning (IRL) has recently increased, that is, interest in the problem of recovering the reward function underlying a Markov decision process (MDP) given the dynamics of the system and the behavior of an expert. This paper deals with an incremental approach to online IRL. First, the convergence property of the incremental method for the IRL problem was investigated, and the bounds of both the mistake number during the learning process and regret were provided by using a detailed proof. Then an online algorithm based on incremental error correcting was derived to deal with the IRL problem. The key idea is to add an increment to the current reward estimate each time an action mismatch occurs. This leads to an estimate that approaches a target optimal value. The proposed method was tested in a driving simulation experiment and found to be able to efficiently recover an adequate reward function.

Key words: Incremental approach, Reward recovering, Online learning, Inverse reinforcement learning, Markov decision process
doi:10.1631/jzus.C1010010 **Document code:** A **CLC number:** TP181

1 Introduction

The inverse reinforcement learning (IRL) problem was first proposed by Russell (1998). The problem relates, in the context of Markov decision processes (MDPs), to recovering a reward function that an agent is optimizing, given a model of the environment. It is especially useful in situations where the reward itself is of interest, for example, when mimicking or evaluating a person's driving style. Knowledge of the reward function allows the agent to generalize a new policy even when the environment changes.

Numerous approaches have been proposed to solve the IRL problem. Ng and Russell (2000) used, as a theoretical basis, the maximum margin principle, which involves making the demonstration policy look significantly better than any alternative. In this way, the reward learning problem can be examined as an optimization problem. Abbeel and Ng (2004) proposed practical algorithms to resolve this optimization problem. Other methods have also been investigated to recover an adequate reward (Ratliff *et al.*, 2006; Neu and Szepesvari, 2007; Ramachandran and Amir, 2007; Syed and Schapire, 2008; Syed *et al.*, 2008; Ziebart *et al.*, 2008). Due to their offline nature, these methods are inefficient and have a time consuming batch-learning course. Lopes *et al.* (2009) applied active learning within the context of IRL to reduce the requirement of samples from the expert. Active learning uses a Bayesian approach to identify an appropriate reward function in both batch and online settings. These approaches concentrate on cases where the agent is able to query for demonstrations at specific states as needed.

In this paper, we first assume that the learner has access to demonstrations by an expert who is trying to maximize a latent reward function, which can be expressed as a linear combination of several known features. We then provide a convergence analysis of

‡ Corresponding author
* Project (No. 90820306) supported by the National Natural Science Foundation of China

the general incremental solution in an IRL background, and then propose an incremental IRL algorithm. Through an empirical experiment, we demonstrate that our algorithm usually terminates within a few steps and produces a learned policy that has a style highly similar to the expert's and produces a reward function that leads to that policy.

2 Notation and problem formulation

A finite-state MDP is a tuple (S, A, P, γ, R), where S is a finite set of N states, $A=\{a_1, a_2, \ldots, a_k\}$ is a set of k actions, P denotes the state transition function, $P_a(s)$ denotes the state transition probabilities upon taking action a in state s, $\gamma\in[0, 1]$ refers to the discount factor, and R is the reward function, bounded in absolute value by $R_{\max}$. Throughout this paper, the reward function is written as $R(s)$ instead of $R(s,a)$ for simplicity. This is valid in situations where the goal is related only with the state, such as in a path planning problem.

A series of observations of the expert's behavior, $O=\{(s_1,a_1), (s_2,a_2), \ldots, (s_T,a_T)\}$, which means that in the state s_i the expert takes action a_i at the time step i. A policy is defined as a map π: $S\rightarrow A$.

As for the expert, there are two basic assumptions: (1) the expert is attempting to maximize the total accumulated reward according to a latent reward function, which we will call the 'reward optimality condition', and (2) the expert always performs the optimal action.

For the solution to the problem, two classical statements concerning MDP are required, Bellman equations and Bellman optimality (Sutton and Barto, 1998).

Statement 1 (Bellman equations) Given an MDP $M=(S, A, P, \gamma, R)$ and a policy π: $S\rightarrow A$, for all $s\in S$ and $a\in A$, V^π and Q^π satisfy

$$V^{\pi}(s)=R(s)+\gamma\sum_{s'}P_{s\pi}(s')V^{\pi}(s'), \tag{1}$$

$$Q^{\pi}(s,a)=R(s)+\gamma\sum_{s'}P_{sa}(s')V^{\pi}(s'). \tag{2}$$

Statement 2 (Bellman optimality) Given an MDP $M=(S, A, P, \gamma, R)$ and a policy π: $S\rightarrow A$, π is an optimal policy for M if and only if, for all $s\in S$,

$$\pi(s)\in\arg\max_{a\in A}Q^{\pi}(s,a). \tag{3}$$

Within a loose coupling state space, in order to obtain better generalization ability, state space is often represented as a linear combination of features. As suggested by Ng and Russell (2000), in this work it is assumed that

$$R(s)=\sum_{i=1}^{d}\omega_i\varphi_i(s),$$

where $\varphi_1, \varphi_2, \ldots, \varphi_d$ are fixed basis reward functions, each corresponding to one particular feature, and $\boldsymbol{\omega}=[\omega_1, \omega_2, \ldots, \omega_d]$ is the feature weight or feature coefficient among these basis reward functions. Under this assumption, the following relationship between the value function and feature expectation exists: $V=\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}$, where $\mu_i=E\left[\sum_{t=1}^{\infty}\gamma^t\varphi_i(s_t)\,|\,\pi\right]$ is called the feature expectation.

Based on the maximum margin principle, which states that the feature weight vector should maximize the difference of the return between the demonstration and the other alternatives, the IRL problem can be stated as an optimization problem in the form

$$\begin{aligned}&\max\left(V_R(\pi^*)-E_\pi\left[V_R(\pi)\right]\right)\\ &\text{s.t.}\qquad \|R\|_2=1,\end{aligned} \tag{4}$$

where π^* is the expert's policy and the expectation is over all other policies π. Intuitively, the goal is to make the margin between the optimal policy and the others as large as possible.

3 Incremental inverse reinforcement learning algorithm

This section provides the derivation of the sufficient and necessary conditions of the optimal action from the Bellman equation. Then, a variant of the original optimization form is provided. Furthermore, the analysis and proof of the convergence property of the incremental algorithm in the context of the IRL problem are emphasized. This is followed by a detailed description of the incremental IRL algorithm.

For simplicity, Eq. (1) can be rewritten in vector form:

$$\boldsymbol{V}^{\pi}=\boldsymbol{R}+\gamma\boldsymbol{P}^{\pi}\boldsymbol{V}^{\pi}.$$

Simple manipulation gives

$$\boldsymbol{V}^{\pi}=(\boldsymbol{I}-\gamma\boldsymbol{P})^{-1}\boldsymbol{R}.$$

As suggested by Ng and Russell (2000), suppose in the state s_1 the expert takes action a^* as the optimal choice. Then the objective function in Eq. (4) has the following equivalent expressions:

$$\begin{aligned}&\boldsymbol{P}_{a^*}^{\mathrm{T}}(s_1)\boldsymbol{V}^{\pi}-\boldsymbol{P}_{a}^{\mathrm{T}}(s_1)\boldsymbol{V}^{\pi}\\&=\boldsymbol{P}_{a^*}^{\mathrm{T}}(s_1)(\boldsymbol{I}-\gamma\boldsymbol{P})^{-1}\boldsymbol{R}-\boldsymbol{P}_{a}^{\mathrm{T}}(s_1)(\boldsymbol{I}-\gamma\boldsymbol{P})^{-1}\boldsymbol{R}\\&=(\boldsymbol{P}_{a^*}(s_1)-\boldsymbol{P}_{a}(s_1))^{\mathrm{T}}(\boldsymbol{I}-\gamma\boldsymbol{P})^{-1}\boldsymbol{R},\end{aligned}\tag{5}$$

where $a\in A/a^*$. Thus, after a series of the demonstration $O=\{(s_1,a_1), (s_2,a_2), \ldots, (s_T,a_T)\}$, based on the maximum margin principle, we can write the optimization problem as

$$\max\left(\sum_{i=0}^{T}\sum_{a\in A/a^*}(\boldsymbol{P}_{a^*}(s_i)-\boldsymbol{P}_{a}(s_i))^{\mathrm{T}}(\boldsymbol{I}-\gamma\boldsymbol{P}_{a^*})^{-1}\boldsymbol{R}\right)$$
$$\text{s.t.}\quad \|\boldsymbol{R}\|_2=1.$$

For conciseness, the demonstration instance at each step is denoted by

$$\boldsymbol{v}_i^{\mathrm{T}}=(\boldsymbol{P}_{a^*}(s_i)-\boldsymbol{P}_{a}(s_i))^{\mathrm{T}}(\boldsymbol{I}-\gamma\boldsymbol{P}_{a^*})^{-1}$$

for $a\in A/a^*$. Note that the $\boldsymbol{v}_i^{\mathrm{T}}$ implicitly contains multiple vectors, with each corresponding to one non-optimal action when the non-optimal action is not unique. $\boldsymbol{c}=\sum_{i=0}^{T}\boldsymbol{v}_i$ is called the accumulated demonstration instance.

Suppose at step i the expert is in s_i and is changed to s_{i+1} after performing optimal action a^*, so that all other actions are regarded as non-optimal actions by default. Since statement a^* is optimal and equivalent to $Q^*(s_i,a^*)\geq Q^*(s_i,a)$, $\forall a\in A/a^*$, it is natural to define the loss function as

$$g(\boldsymbol{R},\boldsymbol{v}_i)=\begin{cases}-\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{R}, & \text{if } i\in M,\\ 0, & \text{if } i\notin M,\end{cases}$$

where $M=\{i\text{: } \boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{R}_i<0\}$, $\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{R}=Q^*(s_i,a^*)-Q^*(s_i,a)$. At this stage, the problem statement takes the more condensed form of

$$\inf_{\|\boldsymbol{R}\|_2=1}-\sum_{i=0}^{T}\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{R}=\inf_{\|\boldsymbol{R}\|_2=1}-\boldsymbol{c}^{\mathrm{T}}\boldsymbol{R},\tag{6}$$

which actually means the same as the optimization problem described above. The problem with such a form is commonly regarded as the binary classification problem (BCP), which can be solved in sequential fashion by using the quasi-additive online classification algorithm (QOCA). The convergence analysis of the incremental IRL algorithm, which is an extension of the QOCA in the IRL problem, will be discussed in the following.

3.1 Convergence analysis

Without loss of generality, a classic incremental algorithm shall take the following form. Within each step of the learning process, the agent first makes a prediction of an action and then observes the correct action. If they are the same then nothing happens; otherwise, an adjusting weight is added to the current estimate to produce a new estimate. The above steps are repeated until a stop condition is reached. The key update rule can be formalized as $\boldsymbol{R}_{i+1}:=\boldsymbol{R}_i+\alpha\delta_i\boldsymbol{v}_i$. This subsection will place emphasis on the convergence and performance analysis of these methods.

Subsequent discussion is based on the following lemma:

Lemma 1 Given an MDP/R, suppose the optimal action in state s_i is a^* while a is any other non-optimal action. Let $\boldsymbol{A}=(\boldsymbol{I}-v\boldsymbol{P}_{a^*})^{-1}$. Then for each pair $\boldsymbol{v}_i$, $\boldsymbol{v}_j$, we have

$$\sup\left|\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{v}_j\right|=4\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}).$$

Proof First, for each s_i,

$$\begin{aligned}\|\boldsymbol{P}_{a^*}(s_i)-\boldsymbol{P}_{a}(s_i)\|_2&\leq\|\boldsymbol{P}_{a^*}(s_i)\|_2+\|\boldsymbol{P}_{a}(s_i)\|_2\\&\leq\|\boldsymbol{P}_{a^*}(s_i)\|_1+\|\boldsymbol{P}_{a}(s_i)\|_1=2,\end{aligned}\tag{7}$$

which starts with a triangular inequality of the norm, and is based on the fact that for the arbitrary distribution $\boldsymbol{p}\in\{\boldsymbol{p}\text{: } \boldsymbol{p}^{\mathrm{T}}\boldsymbol{1}=1,\ \boldsymbol{p}\geq\boldsymbol{0}\}$, $\|\boldsymbol{p}\|_2\leq\|\boldsymbol{p}\|_1=1$ always holds true.

For simplicity, let $\boldsymbol{x}_i=\boldsymbol{P}_{a^*}(s_i)-\boldsymbol{P}_{a}(s_i)$. Then,

$$\begin{aligned}\left|\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{v}_j\right| &\equiv \left|\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{x}_j\right| \le \left\|\boldsymbol{A}^{\mathrm{T}}\boldsymbol{x}_i\right\|_2 \left\|\boldsymbol{A}^{\mathrm{T}}\boldsymbol{x}_j\right\|_2 \\ &\le \left\|\boldsymbol{x}_i\right\|_2 \left\|\boldsymbol{A}^{\mathrm{T}}\right\|_2^2 \left\|\boldsymbol{x}_j\right\|_2 = 4\left\|\boldsymbol{A}^{\mathrm{T}}\right\|_2^2 \\ &= 4\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}),\end{aligned} \tag{8}$$

where we used, sequentially, (1) the definition of $\boldsymbol{v}_i^{\mathrm{T}}$, (2) the Cauchy-Schwarz inequality, (3) the property of the matrix norm, (4) inequality (7), and (5) the definition of the matrix norm, where $\lambda_{\max}(\boldsymbol{A})$ denotes the maximum eigenvalue of matrix $\boldsymbol{A}$. Proof completed.

Further analysis together with the above lemma gives rise to the following convergence theorem, the first part of which is based on Kivinen (2003):

Theorem 1 Given an MDP/R and an example sequence $O=\{(s_1,a_1), (s_2,a_2), \ldots, (s_T,a_T)\}$, assuming that $\|\boldsymbol{v}_i\|_2\le R$ and $\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{R}^*>l$ hold for any i. Supposing there exists at least one $\boldsymbol{R}^*$ satisfying $\|\boldsymbol{R}^*-\boldsymbol{R}_0\|_2\le U$, then for the incremental IRL algorithm one can establish:

(1) This algorithm will make at most

$$|M| \le U^2R^2/l^2$$

mistakes in the example sequence.

(2) This algorithm has a regret bound of

$$\mathrm{regret}(\boldsymbol{R}^*,T) \le 2\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})|M|^2,$$

where $\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})$ is the maximum eigenvalue of $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$.

Proof (1) First, we define

$$\delta_i = \begin{cases}1, & \boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{R}_i \ge 0, \\ 0, & \text{otherwise.}\end{cases}$$

Recall that in an incremental algorithm each update in the estimate can be formalized as $\boldsymbol{R}_{i+1}:=\boldsymbol{R}_i+\alpha\delta_i\boldsymbol{v}_i$, where α is the learning rate and $\boldsymbol{v}_i$ is the demonstration instance.

If a mistake occurs at step i and the estimate is updated, $\boldsymbol{R}_{i+1}$ is guaranteed to approach $\boldsymbol{R}^*$ instead of $\boldsymbol{R}_i$. This relationship can be expressed as

$$\begin{aligned}&\left\|\boldsymbol{R}^*-\boldsymbol{R}_i\right\|_2^2 - \left\|\boldsymbol{R}^*-\boldsymbol{R}_{i+1}\right\|_2^2 \\ &= \left\|\boldsymbol{R}^*-\boldsymbol{R}_i\right\|_2^2 - \left\|\boldsymbol{R}^*-\boldsymbol{R}_i-\alpha\delta_i\boldsymbol{v}_i\right\|_2^2 \\ &= 2\alpha\delta_i\boldsymbol{v}_i^{\mathrm{T}}(\boldsymbol{R}^*-\boldsymbol{R}_i) - \alpha\delta_i\left\|\boldsymbol{v}_i\right\|_2^2 \\ &\ge \delta_i(2\alpha l - \alpha^2R^2),\end{aligned} \tag{9}$$

where we have made use of $\delta_i^2=\delta_i$ in line 3, $\|\boldsymbol{v}_i\|_2\le R$, $\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{R}^*>l$, and $\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{R}_i<0$ in line 4.

Setting the derivative of $\delta_i(2\alpha l-\alpha^2R^2)$ with respect to α to zero and then rearranging, the solution $\alpha=l/R^2$ is obtained. Substituting it into Eq. (9) gives

$$\left\|\boldsymbol{R}^*-\boldsymbol{R}_i\right\|_2^2 - \left\|\boldsymbol{R}^*-\boldsymbol{R}_{i+1}\right\|_2^2 \ge \delta_i\, l^2/R^2.$$

By summing this over $i=0, 1, \ldots, T$ and eliminating the intermediate items, we obtain

$$\begin{aligned}&\left\|\boldsymbol{R}^*-\boldsymbol{R}_0\right\|_2^2 - \left\|\boldsymbol{R}^*-\boldsymbol{R}_{T+1}\right\|_2^2 \\ &= \sum_{i=0}^{T}\left\|\boldsymbol{R}^*-\boldsymbol{R}_i\right\|_2^2 - \left\|\boldsymbol{R}^*-\boldsymbol{R}_{i+1}\right\|_2^2 \\ &\ge \sum_{i=1}^{T}\delta_i l^2/R^2.\end{aligned}$$

Noting that $\left\|\boldsymbol{R}^*-\boldsymbol{R}_{T+1}\right\|_2^2 \ge 0$ and the assumption $\left\|\boldsymbol{R}^*-\boldsymbol{R}_0\right\|_2^2 \le U^2$, we conclude that

$$\sum_{i=1}^{T}\delta_i \le U^2R^2/l^2.$$

Considering the fact $M=\sum_{i=0}^{T}\delta_i$, the result is obtained. The proof is completed.

It is noteworthy that l denotes the minimum of the margin between value functions of the expert's policy and the alternatives when the optimal reward function is given, and that U is the distance between the optimal solution and the initial estimate. The bound of the mistakes can be represented in terms of l and U only when they are well defined.

(2) Noting that $g(\boldsymbol{R}^*,\boldsymbol{v}_i)=0$ holds for any i, we obtain

$$\begin{aligned}\mathrm{regret}(\boldsymbol{R}^*,T) &\equiv \sum_{i=0}^{T}g(\boldsymbol{R}_i,\boldsymbol{v}_i) - \sum_{i=0}^{T}g(\boldsymbol{R}^*,\boldsymbol{v}_i) \\ &= -\sum_{i=0,i\in M}^{T}\sum_{j<i,j\in M}\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{v}_j \le \sum_{i=0,i\in M}^{T}\sum_{j<i,j\in M}\left|\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{v}_j\right| \\ &\le \sup\left\{\left|\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{v}_j\right|\right\}|M|^2/2 = 2\lambda_{\max}(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})|M|^2,\end{aligned}$$

where we sequentially use (1) the definition, (2) $g(\boldsymbol{R}^*,\boldsymbol{v}_i)=0$ for any i, (3) inequality $\sum_i\boldsymbol{x}_i\le|\sum_i\boldsymbol{x}_i|\le\sum_i|\boldsymbol{x}_i|$,

(4) summation expansion, and (5) Lemma 1. This completes the proof.

3.2 Algorithm description

This subsection describes the incremental algorithm used to solve the IRL problem.

A detailed description of the incremental IRL algorithm is provided in Algorithm 1, and the policy iteration algorithm listed in Algorithm 2 is the same RL algorithm as the one presented by Sutton and Barto (1998). Note that R and $\boldsymbol{R}$ both denote the reward function, the former in function form, while the latter in vector form.

Algorithm 1 Incremental IRL algorithm

Parameters: Learning rate $\alpha\in(0, 1]$, model MDP(S, A, P, γ).

```
1   Initialize $\boldsymbol{R}_0 := 1/\sqrt{|S|}$;
2   For $i$:=0, 1, …, $T-1$
3     $\pi$:=policyIteration(MDP, $\boldsymbol{R}_i$);
4     Observe ($s_i$, $S_a^*$) and choose $a^*\in S$;
5     If $\pi(s_i)\in S_a^*$
6       $\boldsymbol{R}_{i+1}:=\boldsymbol{R}_i$;
7     Else
8       $\boldsymbol{v}_i := (\boldsymbol{P}_{a^*}(s_i)-\boldsymbol{P}_{\pi(s_i)}(s_i))^{\mathrm{T}}(\boldsymbol{I}-\gamma\boldsymbol{P}_{a^*})^{-1}$;
9       $\boldsymbol{R}_{i+1}:=\boldsymbol{R}_i+\alpha\boldsymbol{v}_i$;
10    End
11  End
12  $\boldsymbol{R}_T:=\boldsymbol{R}_T/\|\boldsymbol{R}_T\|_2$;
13  Return $\boldsymbol{R}_T$.
```

Algorithm 2 Policy iteration algorithm

Parameters: Model MDP(S, A, P, γ), reward function R.

```
1   Initialize $V$:=0, $\pi$:=0;
2   Repeat
3     $\Delta$:=0;
4     For each $s\in S$
5       $v:=V(s)$;
6       $V^{\pi}(s):=R(s)+\gamma\sum_{s'}P_{s\pi}(s')V^{\pi}(s')$;
7       $\Delta:=\max(\Delta, |v-V(s)|)$;
8     End
9   Until $\Delta\leq\varepsilon$
10  policy_stable!=1;
11  For each $s\in S$
12    $b:=\pi(s)$;
13    $\pi(s)=\mathrm{argmax}_{a\in A}(R(s)+\gamma\sum_{s'}P_{sa}(s')V^{\pi}(s'))$;
14    If $b!=\pi(s)$
15      policy_stable!=1;
16      If policy_stable=1
17        Return $\pi$;
18      End
19    Else
20      Go to line 2;
21    End
22  End
```

The reward estimate was initialized, confirming a uniform distribution over the entire state space. At each step, the agent obtains the optimal action $\pi(s_i)$ based on the current reward estimate $\boldsymbol{R}_i$ via the policy iteration method, and then compares this action with the observed expert's action. Note that the expert gives a set of the optimal actions S_a^* when there is more than one optimal action. We can then decide whether a reward update is necessary by checking whether the agent's choice belongs to the optimal policy set S_a^*, as performed in line 5 of Algorithm 1. If the action returned by the policy iteration algorithm is already optimal, then the reward estimate needs no adjustment. Otherwise, an increment should be added to the current estimate to improve the objective function:

$$(\boldsymbol{P}_{a^*}(s_i)-\boldsymbol{P}_{\pi(s_i)}(s_i))^{\mathrm{T}}(\boldsymbol{I}-\gamma\boldsymbol{P}_{a^*})^{-1}\boldsymbol{R}_{i+1}$$
$$\geq(\boldsymbol{P}_{a^*}(s_i)-\boldsymbol{P}_{\pi(s_i)}(s_i))^{\mathrm{T}}(\boldsymbol{I}-\gamma\boldsymbol{P}_{a^*})^{-1}\boldsymbol{R}_i.$$

For conveying an intuitive meaning, Fig. 1 gives an example illustrating what the reward estimate update could look like geometrically. To simplify this discussion, only two states are presented.

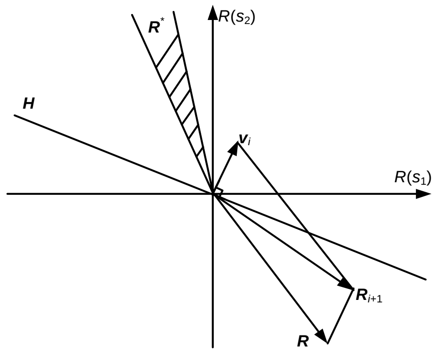

Fig. 1 Schematic of the reward estimate update in the incremental inverse reinforcement learning (IRL) algorithm within a single step, considering the case of only two states (the extension to more states is trivial)

The assumption is that there exists an optimal $\boldsymbol{R}^*$, which is denoted by the shaded area. $\boldsymbol{H}$ refers to a hyper-plane that cuts the reward function space into two half spaces, one of which meets the inequality $\boldsymbol{v}_i^{\mathrm{T}}\boldsymbol{R}^*\geq0$ and the other of which does not. As can be seen in Fig. 1, each time a mismatch occurs, the current estimate $\boldsymbol{R}_i$ is updated to $\boldsymbol{R}_{i+1}$, which approaches $\boldsymbol{R}^*$. Hence, this algorithm improves the current reward estimate after each adjustment.

4 Experiments

In this section we present the empirical results of using the proposed algorithm in a driving simulation. Apprenticeship learning by using the projection method (Abbeel and Ng, 2004) is also considered for comparison.

The experimental interface is shown schematically in Fig. 2. There are five lanes, including two off-road lanes, marked as 0, 1, 2, 3, and 4. The main car, colored black, is driving at a fixed speed, which is faster than all of the other cars. The other cars are driving in the left, middle, and right lanes. This MDP has five different actions {lane0, lane1, lane2, lane3, lane4}, three of which cause the car to steer smoothly to one of the other lanes, while the other two cause it to drive off of, but parallel to, the road, on either the left or the right side. To describe a state, the algorithm at least needs to know which lane the main car is in as well as the distance to the closest car in each lane. This distance is uniformly discretized into ten grids; the distance to the closest car in a lane without other cars is always counted as the greatest distance. Therefore the total state number is 5×10×10×10=5000.

This simulation ran at 10 Hz and the driving style was observed from a single trajectory of 1200 samples.

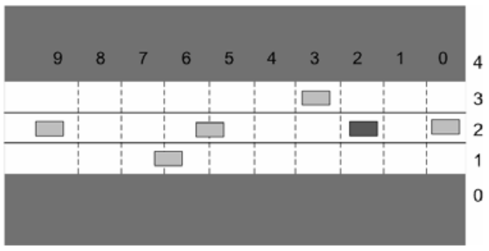

Fig. 2 Highway driving simulation interface

A variety of different driving styles were demonstrated to see if the algorithm could mimic the same style in every scene. Eight driving styles (Table 1) are taken into account in this experiment.

The learned policy was compared with the expert's policy by comparing the state-action pairs. All states not visited in the demonstration were ignored. The policy similarity of the learned policy, as a measure of policy discrepancy, equals the percentage of the number of the matching state-action pairs.

The policy similarity when using the incremental IRL algorithm and apprenticeship learning was examined (Fig. 3), tested within the same period of time (here 1200 steps). From Fig. 3, in most cases the incremental IRL algorithm achieved equivalent or better learning quality than its counterpart.

The fact that the incremental IRL algorithm is inferior to apprenticeship learning in a few complex scenarios can be attributed to any of several reasons. On the one hand, the policy optimal condition is rarely satisfied for many problems of practical interest. Without an existing exact optimal solution, the sequential method definitely leads to intermittent oscillations. On the other hand, policy similarity is a

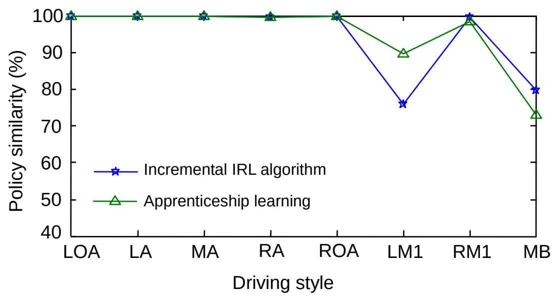

Fig. 3 Plot of the policy similarity of both the apprenticeship learning and incremental inverse reinforcement learning (IRL) algorithms versus various driving styles (tested within the same period of time, 1200 steps)

Table 1 Driving styles in the driving simulation

Style name	Abbreviation	Style description
Left lane always	LA	Always drive in the left lane, regardless of collisions
Middle lane always	MA	Always drive in the middle lane, regardless of collisions
Right lane always	RA	Always drive in the right lane, regardless of collisions
Left off-road always	LOA	Always drive in the left off-road lane
Right off-road always	ROA	Always drive in the right off-road lane
Left lane mainly	LM1	Drive mainly in the left lane, and drive to the left off-road lane to avoid collisions at a close distance, then back to the left lane
Right lane mainly	RM1	Drive mainly in the right lane, and drive to the right off-road lane to avoid collisions at a close distance, then back to the right lane
Middle lane bad	MB	Drive mainly in the middle lane and hit as many other cars as possible

subjective measurement and therefore cannot be used to depict exactly to what degree the two policies are distinct.

Fig. 4 shows that in some simple cases, such as when the main car remains in a certain lane, the incremental IRL algorithm converges rapidly and opts to select the optimal action after only a few steps. In other cases it takes more time for the algorithm to obtain the best policy. The disturbance while learning the LM1 and MB styles probably stems from a non-optimal demonstration.

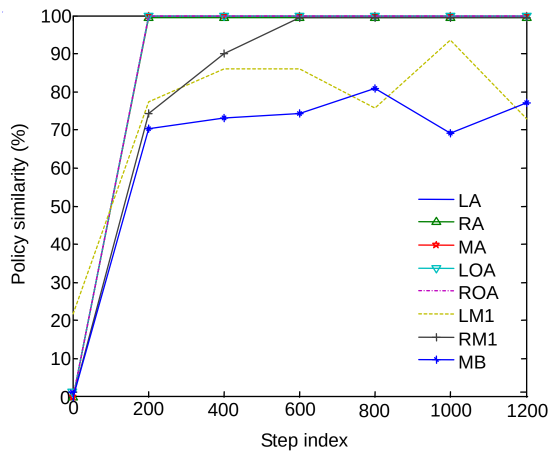

Fig. 4 Convergence rate of the incremental inverse reinforcement learning (IRL) algorithm with the eight driving styles

5 Conclusions

Inverse reinforcement learning, as a class of approaches to finding an optimal policy when the reward function is unavailable, has been successfully applied in several domains (Abbeel *et al*., 2007; 2008; Ratliff *et al*., 2007; Kolter *et al*., 2008). Nevertheless, there remain open questions if this method is to be used in more comprehensive applications.

In summary, in this paper, an innovative online approach to the IRL problem is proposed. There are several advantages of our technique over previous work: (1) The approach does not employ a strategy of matching the feature count between an observed policy and a learner's policy, which might lead to ambiguity due to the fact that many policies can generate the same feature counts. In contrast, the intentions of the expert are taken into account in our method. Note that the feature count and the function approximation mentioned above differ in the functionality. The former is used as the goal of the learning while the latter is used to represent the reward function as a linear combination of several components. (2) This approach avoids solving the quadratic problem, which is generally reckoned as a costly operation. (3) It also enjoys the benefit of the ability to handle data that arrive singly. On the other hand, this algorithm suffers from some disadvantages: (1) Its performance might fluctuate when encountering a non-optimal demonstration. It is well known that human demonstration suffers from non-optimality due to delayed responses and inaccurate judgments. (2) The algorithm's performance greatly depends on the learning features, which are, for now, manually designated.

One of the primary assumptions is that the reward function is expressible as a linear function of some known fixed features. Hence, it remains an important problem to develop methods for learning reward functions that might be non-linear functions of the features. On the other hand, the learning feature is expected to be constructed or selected from date instead of being manually adjusted (Chen *et al*., 2010). Furthermore, it might be possible to substitute the temporal difference method for the dynamic programming method, when solving the Bellman equation to strengthen the online characterization. Nevertheless, this might give rise to a challenging problem of performing value iteration with a variant reward function.

References

Abbeel, P.Y., Ng, A.Y., 2004. Apprenticeship Learning via Inverse Reinforcement Learning. 21st Int. Conf. on Machine Learning, p.1-8. [doi:10.1145/1015330.1015430]

Abbeel, P.Y., Coates, A., Quigley, M.Y., Ng, A., 2007. An Application of Reinforcement Learning to Aerobatic Helicopter Flight. Advances in Neural Information Processing Systems. MIT Press, Cambridge, McCallum, p.76-84.

Abbeel, P.Y., Dolgov, D., Ng, A.Y., Thrun, S., 2008. Apprenticeship Learning for Motion Planning with Application to Parking Lot Navigation. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, p.1083-1090.

Chen, S.Y., Qian, H., Fan, J., Jin, Z.J., Zhu, M.L., 2010. Modified reward function on abstract features in inverse reinforcement learning. *J. Zhejiang Univ.-Sci. C (Comput. & Electron.)*, **11**(9):718-723. [doi:10.1631/jzus.C0910486]

Kivinen, J., 2003. Online learning of linear classifiers. *Adv. Lect. Mach. Learn.*, **26**(1):235-258. [doi:10.1007/3-540-36434-X_8]

Kolter, J.Z., Abbeel, P.Y., Ng, A., 2008. Hierarchical Apprenticeship Learning with Application to Quadruped Locomotion. Advances in Neural Information Processing Systems. MIT Press, Cambridge, UK, p.769-776.

Lopes, M., Melo, F., Montesano, L., 2009. Active learning for reward estimation in inverse reinforcement learning. *LNCS*, **5782**:31-46. [doi:10.1007/978-3-642-04174-7_3]

Neu, G., Szepesvari, C., 2007. Apprenticeship Learning Using Inverse Reinforcement Learning and Gradient Methods. 23rd Conf. on Uncertainty in Artificial Intelligence, p.295-302.

Ng, A., Russell, S., 2000. Algorithms for Inverse Reinforcement Learning. 17th Int. Conf. on Machine Learning, p.663-670.

Ramachandran, D., Amir, E., 2007. Bayesian Inverse Reinforcement Learning. 20th Int. Joint Conf. on Artifical Intelligence, p.2586-2591.

Ratliff, D.N., Bagnell, J.A., Zinkevich, M., 2006. Maximum Margin Planning. 23rd Int. Conf. on Machine Learning, p.729-736. [doi:10.1145/1143844.1143936]

Ratliff, D.N., Bagnell, J.A., Srinivasa, S.S., 2007. Imitation Learning for Locomotion and Manipulation. 7th IEEE-RAS Int. Conf. on Humanoid Robots, p.392-397. [doi:10.1109/ICHR.2007.4813899]

Russell, S., 1998. Learning Agents for Uncertain Environments. 11th Annual Conf. on Computational Learning Theory, p.101-103. [doi:10.1145/279943.279964]

Sutton, R.S., Barto, A.G., 1998. Reinforcement Learning: an Introduction. MIT Press, USA, p.51-86.

Syed, U., Schapire, R.E., 2008. A Game-Theoretic Approach to Apprenticeship Learning. Advances in Neural Information Processing Systems. MIT Press, Cambridge, MA, p.1449-1456.

Syed, U., Bowling, M., Schapire, R.E., 2008. Apprenticeship Learning Using Linear Programming. 25th Int. Conf. on Machine Learning, p.1032-1039. [doi:10.1145/1390156.1390286]

Ziebart, B.D., Maas, A., Bagnell, J.A., Dey, A.K., 2008. Maximum Entropy Inverse Reinforcement Learning. 23rd National Conf. on Artificial Intelligence, p.1433-1438.