Apprenticeship Learning Based on Inconsistent Demonstrations

Gakuto Masuyama[1] and Kazunori Umeda[1]

***Abstract*— Apprenticeship learning based on inconsistent demonstrations is presented in this paper. We address a problem where given demonstrations are not directly applicable to reward function estimation due to the non-stationarity of an environment or the difference between the dynamics of a robot and a demonstrator. A basic idea of the proposed method is to use a subset of the trajectories sampled from the baseline policy as training data for inverse reinforcement learning. All consistent sample trajectories and inconsistent demonstrations are abstracted by an affine transformation invariant feature. Using the feature, the importance of each sample trajectory is estimated. Rating the sample trajectories based on importance, the training data for inverse reinforcement learning are identified. The validity of our approach is verified through simulation in two scenarios: inconsistency caused by variation of an environment and performance of a robot.**

I. INTRODUCTION

Reinforcement Learning (RL) [1] is a promising approach to developing an autonomous and adaptive robot control system. In the RL framework, a robot is forced to interact with an environment by trial and error, and the robot receives a reward signal as a result of each decision. The robot learns the optimal control policy for a given problem by maximizing expectation of the accumulated reward signal. Large studies have demonstrated the effectiveness of RL for the robot control problem [2], [3], [4].

As is the nature of autonomous learning through exploration, performance of the RL algorithms is vitally dependent on the reward function. However, designing the reward function is prone to be time consuming for system designers, especially in cases where the problem involves complex tasks and dynamics from the environment. To address that problem, Inverse Reinforcement Learning (IRL) and Apprenticeship Learning (AL) have been developed [5], [6], [7]. IRL is a framework for estimating the underlying reward function given an expert's demonstrations, and AL recovers policy via the procedures of IRL and RL.

Most IRL and AL algorithms assume that the experts and a robot make their decisions in the same Markov Decision Process (MDP). However, when the assumption cannot hold, demonstrations observed from the expert might be *inconsistent* for the robot to solve the given problem. In this paper, an AL algorithm is proposed to enable the agent [1] to obtain a suitable reward function and policy in current MDP from potentially inconsistent demonstrations. Here we consider inconsistencies in the sense that the demonstrations are helpful but intractable for the agent's estimation of its reward function in a direct IRL approach [2]. Note that inconsistency is a rather different problem than imperfect or poor demonstration, which has been discussed in IRL/AL literature in that the MDP itself is shared by the expert and the agent.

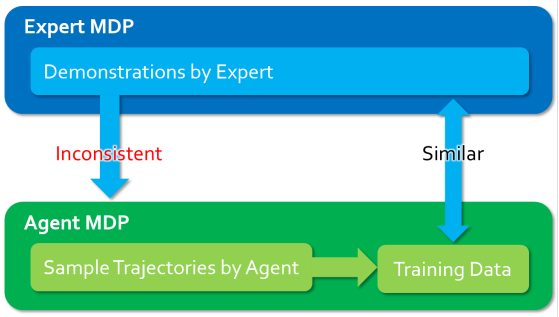


Fig. 1: Basic idea of proposed AL. Trajectories demonstrated in Expert MDP are inconsistent with Agent MDP. Measuring similarity between the demonstrated and sampled trajectories, the agent identifies some of the sampled trajectories as training data for IRL.

We call supposed MDP by an expert and an agent as *Expert MDP* and *Agent MDP*, respectively. The basic idea of our AL is quite simple as is shown in Fig. 1. Direct AL procedure using inconsistent demonstration might lead the agent to unsuitable policy because the policy of the expert could be infeasible in Agent MDP. On the other hand, it is at least guaranteed that trajectories sampled by an arbitrary *baseline policy* of the agent are feasible. Therefore, we use a subset of the *sample trajectories* as demonstrations in Agent MDP. In this paper, when we refer to a sample trajectory, we are indicating a trajectory potentially acceptable as a demonstration in Agent MDP. Demonstrations of the expert are utilized to identify a suitable sample trajectory. The similarities between the sample trajectories and the demonstrations are measured by an abstracted time series feature. Then a subset of the sample trajectories is accepted as training data for IRL based on the importance estimation technique. Simulation experiments demonstrate that the proposed method enables the agent to obtain better policy than that obtained by a direct AL procedure.

[1]Authors are with the Department of Precision Mechanics, Faculty of Science and Engineering, Chuo University, 1-13-27 Kasuga, Bunkyo-ku, Tokyo 113-8551, Japan. {masuyama, umeda} at mech.chuo-u.ac.jp

[1]Although our eventual purpose is to create a control policy for a robot, the range of the application of the proposed method is not necessarily limited to the robotic control problem. Therefore, we refer to agents rather than to robots, hereafter, unless context requires.

[2]Inconsistencies in demonstrations are supposed to be caused by variations of an environment after observation and differences in dynamics, for instance.

II. PRELIMINARIES

In RL problems, it is often assumed that the problem satisfies the MDP, which is represented by a tuple $(S, A, T, R, d_0, \gamma)$, where S is a set of states; A is a set of actions; $T(s, a, s') = Pr(s_{t+1} = s' \mid s_t = s, a_t = a)$ is a transition probability from a state $s \in S$ to a next state $s' \in S$ under action a; R is the reward function; d_0 is an initial state distribution; and $\gamma \in [0, 1)$ is a discount rate. A stochastic policy $\pi(s, a) : S \times A \to [0, 1]$ gives a probability of selecting action a in state s. The objective of RL is to obtain optimal policy π^*, which maximizes expected return $E[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \mid d_0, \pi, T]$, i.e., value function $V^\pi(s)$.

IRL is different from RL in that the underlying reward function for a given task is not provided. Therefore, the MDP without reward function MDP\R is assumed. The reward function is assumed to be represented as $R(s, a) = \sum_{i=1}^{k} \theta_i f_i(s, a)$, where f_i and θ_i are the ith element of feature vector $\boldsymbol{f} \in \mathbb{R}^k$ and parameter vector $\boldsymbol{\theta} \in \mathbb{R}^k$, respectively. The expected feature (count) under policy π is denoted by $f_i^\pi = E[\sum_{t=0}^{\infty} \gamma^t f_i(s_t, a_t) \mid d_0, \pi, T]$. This notation simplifies the value function $V^\pi(s) = \sum_{i=1}^{k} \theta_i f_i^\pi(s, a)$.

III. SIMILARITY MEASURE OF TRAJECTORIES

IRL algorithms often match the feature count for the policy of an expert and that of an agent. In [8], a reward function is estimated under a constraint:

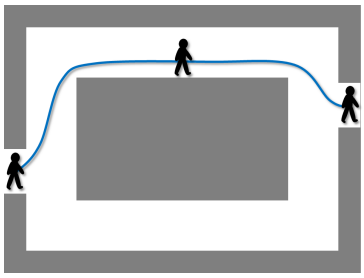
$$|\sum_{\tau \in \mathcal{T}} Pr(\tau) f_i^\tau - \hat{f}_i| \leq \epsilon_i, \tag{1}$$

where f_i^τ is a discounted feature count along a trajectory $\tau = \{(s_0, a_0), \cdots, (s_n, a_n)\}$, $\hat{f}_i$ is an empirical expectation of feature observed from demonstrated trajectories, $\mathcal{T}$ is a set of possible trajectories, and ϵ_i is a threshold. Such a constraint indicates that the estimated reward function makes the agent mimic the behavior of the expert as exactly as possible. However, direct imitation might not be feasible for the agent if the demonstrations are inconsistent with the current situation of the agent. The constraint of matching features in (1) might not work out when we must surmise the inconsistency.

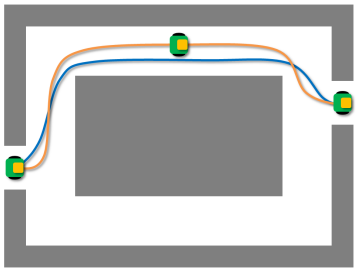
This paper addresses the problem by selecting training data from sample trajectories, which are obtained by the agent's exploration. Although the sample trajectories should be consistent with the Agent MDP, obviously, most of sample trajectories would not be relevant as training data for reward function estimation if we cannot suppose potent prior information [3]. Demonstrations are then used as a reference for selecting relevant data for the Agent MDP from the consistent sample trajectories. Demonstrations by an expert is inconsistent with an Agent MDP, but it can be supposed to be (nearly) optimal in an Expert MDP. Hence, demonstrations can be a key to measuring the relevance of the sampled trajectories. Each sample trajectory is evaluated as to its relevance to the problem based on "similarity" between demonstrations and sample trajectories. The feasibility of the above scheme then depends on the evaluation measure of the similarity between the obtained trajectories and demonstrations.

One simple approach is to use a spatial distance such as the Euclidean distance between demonstrations and the sample trajectories. If there is a sample trajectory showing a small distance to one of the demonstrations, then the trajectory possibly has high relevance and consistency with the Agent MDP. This would be the case if the Agent MDP and the

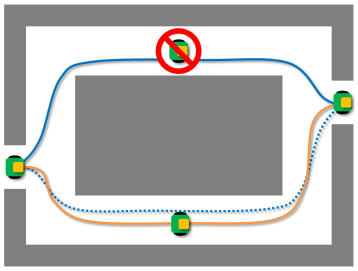
[3]The sample trajectories are obtained in the Agent MDP whereas demonstrations are obtained in the Expert MDP. The sample trajectories are feasible for the agent because they are sampled by the agent itself. However, there is no guarantee that the sample trajectories are relevant to the appropriate reward function for a given problem.



(a) A given demonstration



(b) A sample trajectory of a robot: *spatially* near the demonstration



(c) A sample trajectory of a robot: *geometrically* near the demonstration

Fig. 2: A conceptual example of the distance measure between a demonstration and sample trajectories. (a) A demonstrated trajectory. (b) A sample trajectory spatially near the demonstration. The relevance of the sample trajectory can be evaluated using common distance measure. (c) The robot is prohibited from entering the upper corridor, but no demonstrations are observed in the lower corridor. The demonstration can be available in the lower corridor if the distance measure permits the agent to rotate the demonstration.

Expert MDP have roughly the same properties. A conceptual example is illustrated in Fig. 2. The trajectory of a human from an entrance to an exit is demonstrated (blue line in Fig. 2a). In Fig. 2b, an orange-colored trajectory represents a relevant sample trajectory of a mobile robot. The mobile robot requires wider turns than does the human demonstrator due to its nonholonomic constraints and requirements for safety. However, it seems that feasible and reasonable sample trajectories possibly exist spatially near the demonstration. Therefore, it would be plausible to evaluate the sample trajectories based on common distance measure from the demonstrations. On the other hand, in Fig. 2c, the robot is prohibited from going into the upper corridor; however, no demonstrations are observed in the lower corridor. In this case, the demonstration is inconsistent with the Agent MDP; however, it still can have a certain relevance. Rotating the demonstration by 180 degrees (dotted line) does not cause it to lose consistency and relevance. A relevant sample trajectory in Fig. 2c resembles the rotated demonstrations; hence, a rotation invariant measure (generalization) would be helpful in the situation.

Fig. 2 is just an example, but there are similar situations in robotic control problems. Therefore, such a generalization, including but not limited to rotations, is important for finding consistent and relevant trajectories. In this paper, Affine Invariant Feature (AIF) [9] is introduced as an abstracted feature of a trajectory.

Let $\boldsymbol{\xi} \in \mathbb{R}^d$ be a feature vector (*e.g.*, position and velocity of a robot), and let $\boldsymbol{\Xi}^\tau_{0:n} = [\boldsymbol{\xi}^\tau_0, \cdots, \boldsymbol{\xi}^\tau_t, \cdots, \boldsymbol{\xi}^\tau_n]$ be a time series of $\boldsymbol{\xi}$ along trajectory τ. For any affine transformation for $\boldsymbol{\xi}_t$

$$\bar{\boldsymbol{\xi}}_t = \boldsymbol{A}\boldsymbol{\xi}_t + \boldsymbol{c}, \tag{2}$$

AIF M for $\boldsymbol{\Xi}^\tau_{0:n}$ satisfies $M(\boldsymbol{\Xi}^\tau_{0:n}) = M(\bar{\boldsymbol{\Xi}}^\tau_{0:n})$, where $\bar{\boldsymbol{\Xi}}^\tau_{0:n} = [\bar{\boldsymbol{\xi}}^\tau_0, \cdots, \bar{\boldsymbol{\xi}}^\tau_t, \cdots, \bar{\boldsymbol{\xi}}^\tau_n]$. An actual functional form of M is

$$M(\boldsymbol{\Xi}^\tau_{0:n}) = \sqrt{(\boldsymbol{\mu}_{\tau_a} - \boldsymbol{\mu}_{\tau_b})^T (\boldsymbol{\Sigma}_{\tau_a} + \boldsymbol{\Sigma}_{\tau_b})^{-1} (\boldsymbol{\mu}_{\tau_a} - \boldsymbol{\mu}_{\tau_b})}, \tag{3}$$

where $\boldsymbol{\mu}_{\tau_a}$ and $\boldsymbol{\Sigma}_{\boldsymbol{\tau_a}}$ are the mean and covariance matrix of $\boldsymbol{\Xi}_{\tau_a}$, respectively. $\boldsymbol{\Xi}_{\tau_a} := \boldsymbol{\Xi}^\tau_{0:t_s}$ is an arbitrary subsequence of $\boldsymbol{\Xi}^\tau_{0:n}$, then

$$\boldsymbol{\mu}_{\tau_a} = \frac{1}{t_s + 1} \sum_{t=0}^{t_s} \boldsymbol{\xi}_t, \tag{4}$$

$$\boldsymbol{\Sigma}_{\tau_a} = \frac{1}{t_s + 1} \sum_{t=0}^{t_s} (\boldsymbol{\xi}_t - \boldsymbol{\mu}_{\tau_a})(\boldsymbol{\xi}_t - \boldsymbol{\mu}_{\tau_a})^T. \tag{5}$$

$\boldsymbol{\mu}_{\tau_b}$ and $\boldsymbol{\Sigma}_{\tau_b}$ are defined for subsequence $\boldsymbol{\Xi}_{\tau_b} := \boldsymbol{\Xi}^\tau_{t_s+1:n}$ in the same manner as with τ_a.

AIF was originally developed as a speaker invariant feature in speech recognition research. It is well known that differences in vocal tract length and recording equipment can be approximately modeled by affine transformation for cepstral vectors. Thus, affine transformation invariant features offer a measure of inherent structure in particular speech languages without the normalization of massive data. In the context of this paper, we assume that AIF can be an invariant feature of trajectory, which enables generalization of the trajectory without losing consistency for the MDP. AIF represents the geometric structure of an arbitrary trajectory. Hence, AIF can be seen as an abstracted feature of time series derived from observation ruled by each decision of the expert/agent. Robot control problems are (implicitly) constrained by the dynamics, external environments, and tasks, as is the case with the demonstrator. Therefore, a subset of sample trajectories might possibly have AIFs similar to that of trajectories relevant to a given task [4].

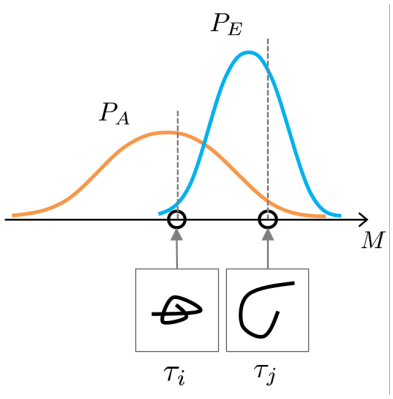


Fig. 3: Densities of the AIF obtained from the policy of the expert and the baseline policy of the agent. The blue- and orange-colored plots represent the probability density of M under the expert policy in the Expert MDP and the baseline policy in the Agent MDP, respectively. τ_i and τ_j are sample trajectories. The AIF of τ_i is frequently observed in the agent but rarely observed in the expert. The AIF of τ_j is frequently observed in the expert and rarely observed in the agent.

IV. PICKING OUT SAMPLE TRAJECTORIES VIA IMPORTANCE ESTIMATION

AIF is introduced in the previous section to measure the similarity of trajectories. Using AIFs of every demonstration and sample trajectory, training data for IRL is selected from a set of sample trajectories. A naive approach is likely to accept a sample trajectory with an AIF close to that of the demonstrations. Any sample trajectory can then be accepted if there exists at least one demonstration showing a similar AIF. However, each demonstration should not have the same importance in implementing a task of the expert. It is natural to suppose that a frequently observed demonstration is essential to implementing the given task. Besides, it is not realistic to assume that the observation and policy of the expert are perfect and deterministic.

Importance is a criterion for deciding which sample trajectory should be accepted as training data for IRL. Fig. 3 gives the perspective of importance. The density of demonstrations

[4]When given tasks are complicated, using AIFs might not be justified. Complicated tasks would require complex trajectories for experts and robots. In such cases, affine transformation invariance would be an abstraction too powerful. Therefore, in this paper, we focus on relatively simple tasks that can be solved with a few chains of motor primitives (skills).

with respect to the AIF (P_E) is right-sided, as compared with the density of the sample trajectories (P_A) in the figure. A sample trajectory τ_i has its AIF $M(\tau_i)$ near the peak of P_A; it is at the skirt of P_E at the same time. On the other hand, the AIF of τ_j is near the peak of P_E, and P_A has a relatively small value at $M(\tau_j)$. Sample trajectory τ_j should have higher importance than τ_i because τ_j has the feature similar to the fundamental trajectories of the expert. Contrary to τ_j, τ_i has lower importance, although it is often observed by the agent.

In general, it should be natural to assume that a baseline policy is not particularly informative about the true optimal policy in the Agent MDP. Therefore, most of the sample trajectory would be irrelevant for IRL, and they would take the AIF into neighborhood of the peak of P_A. On the other hand, in the Expert MDP, trajectories in the neighborhood of the peak of P_E are frequently observed, even if the expert demonstrates iteratively in different initial conditions. Such a set of trajectories must be fundamental for implementing a given task for the expert, as for the agent when the trajectories are abstracted using AIFs. The assumption "frequently observed trajectories in the sense of AIFs are important" is analogic with *feature matching* in the IRL/AL framework.

Along with above discussion, the importance of each sample trajectory τ_i is defined as

$$w(M(\tau_i)) := \frac{P_E(M(\tau_i))}{P_A(M(\tau_i))}, \tag{6}$$

where $w(M(\tau_i))$ is an importance of τ_i. The density of AIFs with respect to demonstrations and sample trajectories are denoted by $P_E(M)$ and $P_A(M)$, respectively. Let us suppose that the agent is given n_E demonstrations $\{\tau_i^E\}_{i=1}^{n_E}$ and n_A sample trajectories $\{\tau_i^A\}_{i=1}^{n_A}$. If densities P_E and P_A are specified, the importance w can be estimated. However, it is rather difficult to estimate density probability in general. Therefore, we introduce the Kullback-Leibler Importance Estimation Procedure (KLIEP) [10] to directly estimate the importance.

KLIEP estimates w instead of estimating each density. Using KLIEP, KL divergence from demonstration density P_E to its estimate $\hat{P}_E(M) = \hat{w}(M)P_A(M)$ is minimized, where $\hat{w}$ denotes the importance estimate:

$$KL(P_E(M)||\hat{P}_E(M)) = \int_{\mathcal{M}} P_E(M) \log \frac{P_E(M)}{\hat{w}(M)P_A(M)} dM. \tag{7}$$

$\mathcal{M}$ is a set of AIFs. The importance estimate $\hat{w}$ is represented by a linear combination of a kernel function. In this paper, the Gaussian distribution has been chosen as the kernel.

Some of the sample trajectories are labeled as training data for IRL based on their estimated importance. Then the reward function is estimated using the IRL algorithm; finally, the policy is obtained by implementing RL under the estimated reward function. The entire procedure of the proposed method is shown in Fig. 4. Relative Entropy Inverse Reinforcement Learning (REIRL) has been chosen as the IRL algorithm. REIRL is suited for our method because it is trajectory based, model free, and does not require recursive computation of policy during reward estimation.

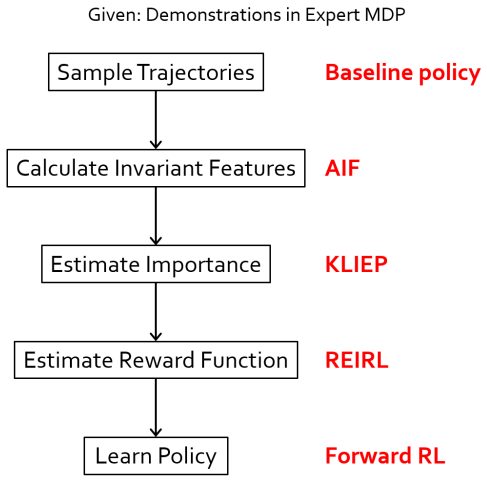


Fig. 4: Schematic procedure of the proposed method. The agent first samples trajectories using an arbitrary baseline policy. AIFs are calculated for each trajectory. The importance of each AIF is then estimated by KLIEP. Training data is picked out from the sample trajectories based on their importance. The reward function is estimated using the selected training data; finally, the agent learns the policy for the estimated reward function.

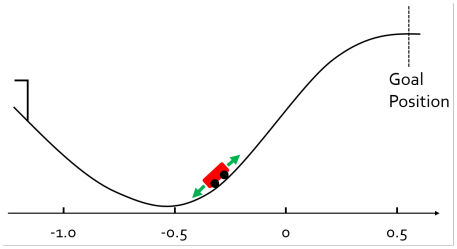


Fig. 5: The mountain car problem. The task is to find policy that enables the car to climb the hill and reach the goal position. The car's motor is underpowered to reach the destination if it keeps accelerating to the right side. Therefore, the car must ascend to a certain height of the left hill to utilize gravity. For more details about the mountain car problem, see [11].

V. SIMULATION

Simulation experiments were conducted to verify the validity of the procedure proposed in Fig. 4. We tested the performance via the mountain car problem (Fig. 5) in two different scenarios. One environmental difference pertains to where the car attempts to climb the hill. The other pertains to the difference in the maximum power of the car. We compared the proposed method with naive REIRL, i.e., REIRL using demonstrations, to estimate the reward function. For each condition below, simulations were implemented 30 times.

TABLE I: Policy losses and success rates at different decay rates. The mean and standard deviation are shown.

	PL (mean)	PL (sd)	SR
Proposed	38.2	21.1	0.57
REIRL	44.3	21.9	0.43

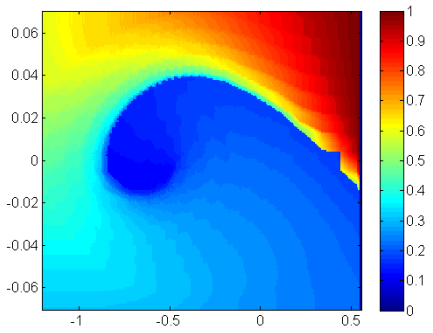
The proposed method and REIRL were evaluated based on *Policy Loss* (PL) $L(R,\pi) = \|\boldsymbol{V}^*(R) - \boldsymbol{V}^{\pi}(R)\|$ [12] and the Success Rate (SR) of accomplishing task in the Agent MDP, where $\boldsymbol{V}^*(R)$ denotes the vectorized optimal value function in the Agent MDP, and $\boldsymbol{V}^{\pi}(R)$ denotes the vectorized value function under estimated policy π. We wish to obtain $\boldsymbol{V}^*(R)$ and its optimal policy; therefore, the small PL is desirable. We applied value iteration [11] in forward RL step. Note that we assume that the expert does not know $\boldsymbol{V}^*(R)$. The assumption is remarkably different from general IRL and AL settings. SR was obtained in the Agent MDP using the optimal policy for each estimated reward function, while the initial state was chosen in the range $[-1, 0]$ and $[-0.02, -0.02]$ of position and velocity. For each policy, the averaged success rate was obtained from 30 trials.

The true reward function gave $+1$ if the position was 0.55 and 0 otherwise. Two variables, the position and velocity of the car, were discretized into 200 grids. Therefore, the number of total possible states is 40000. Each element of feature vector $\boldsymbol{f} \in \{0,1\}^{40000}$ takes 1 iff the car is in a state that corresponds to the element. An initial state of each trajectory was chosen from a uniform distribution. The number of demonstrations was 30, and that of sample trajectories was 100. The sample trajectories were ranked by their estimated importance, and the top 30 trajectories were used as training data of the proposed method. The length of each demonstration and sample trajectory was 60 steps. We determined $t_s = 30$ as a natural choice for segmenting 60-step trajectories into two distributions. The discount rate was 0.99. There were three available actions: acceleration to the left, acceleration to the right, and coasting.

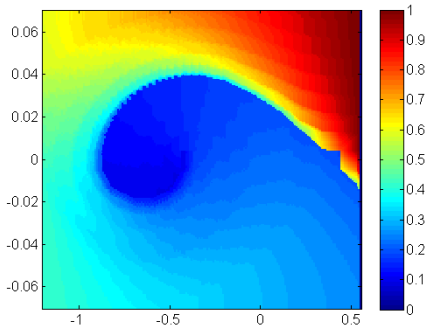
A. Difference in the Decay Rate of Velocity

Different decay rates for the velocity of the car were set in this experiment. In the Expert MDP, the decay rate for velocity was 0.001 everywhere. On the other hand, the decay rate was 0.01 in the Agent MDP if the position was equal to or less than -0.5. The optimal value function in each MDP is shown in Fig. 6. It shows that the difference in the decay rate visibly distorts the optimal path in the state space, i.e., demonstrations were inconsistent with the Agent MDP.

Table I represents the PL and the SR of the proposed method and REIRL. The proposed method improved both the PL and the SR. As compared with the mean of the PL, the standard deviation of the PL seems to result in a relatively large value. This is due to the fact that, occasionally, there were substantially wrong estimates of the reward function. Such failures were caused by the uneven



(a) Optimal value function in Expert MDP



(b) Optimal value function in Agent MDP

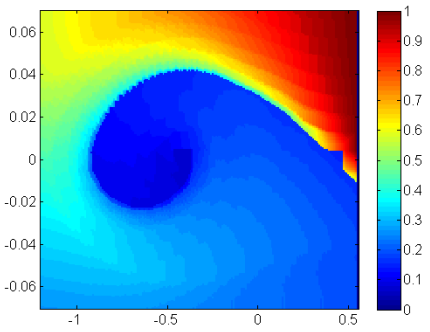
Fig. 6: Optimal value function in (a) the Expert MDP and (b) the Agent MDP. The horizontal axis represents velocity, and vertical axis represents the position of the car. The color bar represents the state value. The decay rate in the Agent MDP was greater than that of the Expert MDP in positions equal to or less than -0.5.

provision of demonstrations or sample trajectories. We did not assume any prior information about the baseline policy; hence, there were occasionally no sample trajectories arriving at the goal state. The relationship between the baseline policy and performance is discussed in V-C.

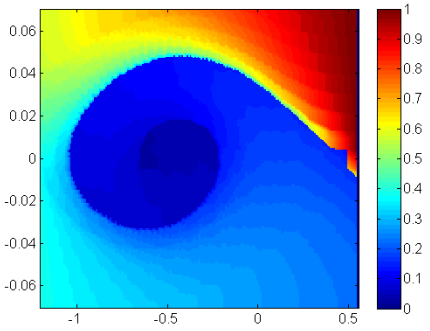
B. Difference in the Power of the Car

We tested the performance of the proposed method with respect to the power of the car. Because the power of the car is lowered, the car is required to climb the left hill higher to take advantage of the force of gravity. As a result, the number of back-and-forth movements of the car increases, and the optimal policy in the Agent MDP differs remarkably from that in the Expert MDP. The power of the car in the Agent MDP was multiplied by coefficients 0.9, 0.8, 0.7, and 0.6. Fig. 7 depicts optimal value functions in the Agent MDP, where the coefficients for the power of the car are 0.8 and 0.6. When the coefficient equals 1.0, the optimal value function is the same as that in Fig. 6a.

The PL and SR results are shown in Table II and Table III, respectively. As compared with REIRL, the proposed method reduced the performance decrement, which is associated with the decrease in power. This tendency is particularly



(a) Coefficient for the power of the car: 0.8



(b) Coefficient for the power of the car: 0.6

Fig. 7: Optimal value function for coefficients (a) 0.8 and (b) 0.6. The horizontal axis represents velocity, and the vertical axis represents the position of the car. The color bar represents the state value.

outstanding at coefficient 0.6, where it was almost impossible for REIRL to provide appropriate policy for the agent. This result indicates the contribution of the abstraction using AIFs and importance-based selection of training data.

C. Discussion

It can be seen that certain improvement is attained by the proposed method. However, it would be difficult to achieve the same level of performance with the usual AL scenario, at the present. We consider that the most fundamental issue for further improvement is the baseline policy. In this paper, the baseline policy is given by a uniform distribution. In the mountain car problem, the car was affected by limited power, the force of gravity, and velocity decay. These conditions resulted in characteristic trajectories, even though the baseline policy is completely random. Therefore, we could find the characteristic trajectories using AIFs and obtained relevant training data. However, there were also trials with no relevant sample trajectories to estimate the goal-directed reward function. The quality of the baseline policy would directly affect performance, especially in high dimensional problem spaces, where available samples are sparse.

VI. CONCLUSIONS

This paper addressed the problem of estimating the reward function based on demonstrations that are potentially inconsistent with current MDP. We have presented the AL method, which selects training data out of trajectories sampled from arbitrary baseline policy. Using AIFs, each trajectory is abstracted and compared with demonstrations. Then the training data for IRL is selected based on importance estimation procedures. Simulation experiments have demonstrated that the proposed method can improve robustness to the inconsistency between the expert and the agent.

We are considering integrating the proposed method with skill-driven intrinsically motivated RL [13] and Nonparametric Bayesian IRL [14] to design appropriate baseline policy.

TABLE II: The policy loss for each coefficient. The top row denotes the coefficients for the power of the car.

	1.0	0.9	0.8	0.7	0.6
Proposed (mean)	28.1	29.7	40.5	35.7	25.9
Proposed (sd)	10.4	16.2	20.8	20.1	15.1
REIRL (mean)	30.6	29.4	43.9	44.6	31.9
REIRL (sd)	14.6	14.9	21.0	21.0	15.8

TABLE III: Success rate for each coefficient.

	1.0	0.9	0.8	0.7	0.6
Proposed	0.90	0.83	0.52	0.47	**0.53**
REIRL	0.75	0.73	0.38	0.36	**0.07**

REFERENCES

[1] M. Wiering, M. van Otterlo, Reinforcement learning: State of the art, Springer-Verlag, 2012.
[2] J. Peters, S. Schaal, Natural actor-critic, Journal of Neurocomputing, vol.71, no.7, pp.1180-1190, 2008.
[3] E. Theodorou, J. Buchli, S. Schaal, A generalized path integral control approach to reinforcement learning, Journal of Machine Learning Research, vol.11, pp.3137-3181, 2010.
[4] J. Kober, J.A. Bagnell, J. Peters, Reinforcement learning in robotics: A survey, The Int. Journal of Robotics Research, vol.32, no.11, pp.1238-1274, 2013.
[5] A. Ng, S. Russell, Algorithms for inverse reinforcement learning, Proc. of the 17th Int. Conf. on Machine Learning, pp.663-670, 2000.
[6] P. Abbeel, A. Ng, Apprenticeship learning via inverse reinforcement learning, Proc. of the 21st ACM Int. Conf. on Machine Learning, 2004.
[7] N. Ratliff, J. Bagnell, M. Zinkevich, Maximum margin planning, Proc. of the 23rd Int. Conf. on Machine Learning, pp.729-736, 2006.
[8] A. Boularias, J. Kober, J. Peters, Relative entropy inverse reinforcement learning, Proc. of the 14th Int. Conf. on Artificial Intelligence and Statistics, pp.182-189, 2011.
[9] Y. Qiao, M. Suzuki, N. Minematsu, Affine invariant features and their application to speech recognition, Proc. of the 34th IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, pp.4629-4632, 2009.
[10] M. Sugiyama, S. Nakajima, H. Kashima, P. Buenau, M. Kawanabe, Direct importance estimation with model selection and its application to covariate shift adaptation, Advances in Neural Information Processing Systems, pp.1433-1440, 2008.
[11] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, Cambridge, MA, MIT Press, 1998.
[12] D. Ramachandran, E. Amir, Bayesian inverse reinforcement learning, Proc. of the 20th Int. Joint Conf. on Artificial Intelligence, pp.2586-2591, 2007.
[13] G. Masuyama, A. Yamashita, H. Asama, Selective exploration exploiting skills in hierarchical reinforcement learning framework, Proc. of the 26th IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp.692-697, 2013.
[14] J. Choi, K. Kim, Nonparametric Bayesian inverse reinforcement learning for multiple reward functions, Advances in Neural Information Processing Systems, pp.305-313, 2012.