Efficient Inverse Reinforcement Learning using Adaptive State-Graphs.

Billy Okal
Social Robotics Lab
University of Freiburg
D-79110 Freiburg im Breisgau
Email: okal@cs.uni-freiburg.de

Hugo Gilbert
Sorbonne Universités, UPMC Paris 06
CNRS, UMR 7606, LIP6
F-75005, Paris
Email: hugo.gilbert@lip6.fr

Kai O. Arras
Social Robotics Lab
University of Freiburg
D-79110 Freiburg im Breisgau
Email: arras@cs.uni-freiburg.de

***Abstract*—Inverse Reinforcement Learning (IRL) provides a powerful mechanism for learning complex behaviors from demonstration by rationalizing such demonstrations. Unfortunately its applicability has been largely hindered by lack of powerful representations that can take advantage of various task affordances while still admitting scalability. Inspired by the success of sampling based approaches in classical motion planning, we use adaptive state graphs to model the underlying Markov decision process (MDP) allowing us to further incorporate task specific constraints efficiently. We then develop a new Bayesian IRL (BIRL) algorithm to learn behaviors using sampled trajectories over the adaptive state graph. We demonstrate the effectiveness of this approach in the task of learning socially compliant robot navigation policies.**

I. Introduction

In many complex tasks required of interactive robots the objective representing the behavior is difficult to specify concretely. Such tasks include socially compliant robot navigation in which the robot is expected to follow implicit rules like not splitting groups, not getting too close to people or alternatively get close to people to serve them drinks etc. This is largely due to typical conflicting objectives like generating short paths while not splitting groups of people. It is however, considerably easier to demonstrate such behaviors and therefore learning from demonstration e.g. via IRL has become the common approach to succeed in enabling robots to perform and generalize such tasks.

Most existing IRL approaches are however plagued with difficulty in handling large state spaces efficiently, mainly due to poor grid-based discretization of the underlying MDP. Such discretization also do not allow incorporation of task specific constraints like kinodynamic and holonomic constraints common in navigation tasks. The resulting learned behaviors typically then include policies that may not be realizable on real robots. Sampling based approaches in motion planning like Probabilistic Roadmaps (PRMs) by Kavraki et al. [3] however has been able to deal with large state spaces well even though their underlying model is a discrete graph.

In the past Guestrin and Ormoneit [2] applied these planning techniques to Reinforcement Learning (RL) by combining local controllers into global solutions, but their approach crucially depends on the availability of the environment dynamics which is uncommon in most IRL tasks. Additionally Neumann et al. [4] extended the previous approach to handle cases with unknown rewards and absence of accurate environment dynamics. They proposed an online RL algorithm that uses feedback from the environment to adapt the state graph representing the underlying MDP while handling the exploration-exploitation trade-off using heuristics. We build upon this work to develop our IRL learning scheme by adding features over local controllers and by modifying BIRL likelihood function to learn reward posterior using sampled trajectories.

II. Approach

We model a robot's decision making task as an MDP $M = \langle \mathcal{S}, \mathcal{A}, T, r, \gamma \rangle$ with states and action spaces $\mathcal{S}$ and $\mathcal{A}$ respectively. $T(s, a, s') = p(s'|s, a)$ is the transition function read as the probability of going to state s' by performing action a in state s. $\gamma \in [0, 1)$ is a discount factor while $r(s, a)$ is the reward obtained by performing action a in state s. A policy $\pi : \mathcal{S} \longmapsto \mathcal{A}$ specifies the action to take at a given state. The goal of the robot is to find the policy π that maximizes its accumulated rewards.

A. Adaptive State-Graph MDP Representation

We build an adaptive graph over the states of the MDP where the nodes are samples from the state space and actions are edges of the graph as described in Neumann et al. [4]. Concretely, the graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ is used to represent the MDP so that $\mathcal{S} \coloneqq \mathcal{V}$ and $\mathcal{A} \coloneqq \mathcal{E}$. The actions/edges correspond to arbitrary local controllers. The reward $r(s, a)$ therefore includes the cost of the local controller a. This permits inclusion of additional task specific costs directly into the reward function via reward basis features without additional complex state space modeling, (e.g. use kinodynamic motion primitives). Trajectories are thus a sequence of edges in the graph from a start to a goal node and the optimal policy then describes which edges to pick at every node. The graph is iteratively built by online exploration that takes into account values of nodes as well as sample concentration to focus adding of new nodes in relevant areas as opposed to naïve uniform sampling.

The graph is initialized using a set of uniformly sampled starting states. In our case, we initialize using the centers of Voronoi cells covering the environment using the human poses.

The graph expansion is guided by an *exploration score*. This is a weighted sum of two terms, a term accounting for the *quality* of the best path visiting node s and a *concentration* term proportional to the number of existing nodes $\mathcal{N}(s)$ within some fixed radius. At each iteration a set of nodes $\mathcal{N}$ from the graph is selected for expansion. From this set, the nodes which are part of best trajectories from start to goal (in terms of maximum rewards) are separated into $\mathcal{N}_b$. A refined expansion set $\mathcal{N}_e$ is chosen to be either $\mathcal{N}_b$ or $\mathcal{N} \setminus \mathcal{N}_b$ with some fixed probability p_b. A node $v \in \mathcal{V}$ from $\mathcal{N}_e$ is then picked with probability proportional to it's exploration score and expanded.

Finally new nodes are sampled uniformly around v, then added to the graph with probabilities proportional to their exploration scores. The values of these new sampled nodes are quickly estimated using Gaussian process regression (see Rasmussen and Williams [7]) using nodes in some neighborhood. The graph MDP is iteratively solved using policy iteration which converges in few steps as only few nodes are added at each iteration ensuring a quick improvement of the best trajectories.

B. Bayesian IRL using Adaptive State-Graph (BIRL-G)

Given a set of M expert demonstration trajectories $\Xi_E = \{\xi_i\}$, where each ξ_i is a set of state-action pairs, we want to recover the reward function. As in Ramachandran and Amir [6] we assume rewards to be iid and are a linear combination of features $\boldsymbol{w}^T \boldsymbol{f}$. We seek the reward posterior given by (1).

$$p(r \,|\, \Xi_E, \Xi_G) \;\propto\; p(\Xi_E, \Xi_G \,|\, r) p(r) \tag{1}$$

where Ξ_G is the set of trajectories generated using the learned reward function. The BIRL approach of Ramachandran and Amir [6] uses PolicyWalk which includes solving an MDP at each iteration to get the policy – which is costly. We use an iterative procedure inspired by Ng and Russell [5]. In our procedure, at each iteration, we only need to estimate the Q function of few nodes using the new reward. This Q function can be estimated by Monte Carlo methods or by just computing the Q function of the best path. We therefore develop a data likelihood that directly works on sampled trajectories as shown in (2). Given a set of starting states $\mathcal{S}_s$,

$$p(\Xi_E, \Xi_G \,|\, r) \;=\; \prod_{s \in \mathcal{S}_s} \frac{\exp(\beta Q_E^{\pi}(s))}{\exp(\beta Q_E^{\pi}(s)) + \sum_{i=1}^{n} \exp(\beta Q_i^{\pi}(s))} \tag{2}$$

where $Q_E^{\pi}(s) \coloneqq \sum_{(s,a) \in \xi} Q^{\pi}(s, a \,;\, r)$ for expert trajectories and $Q_i^{\pi}(s)$ defined analogously for the trajectories generated using the reward at the i^{th} iteration of the algorithm, n is the iteration and β is the expert optimality parameter. This likelihood function is similar to the one of Ramachandran and Amir [6] when each trajectory is interpreted as an MDP action. The reward posterior mean can then be estimated using Markov Chain Monte Carlo (MCMC) techniques. In our case we compute the MAP estimate using a gradient scheme similar to Choi and Kim [1]. We skip the equations for gradient updates here due to space limitations.

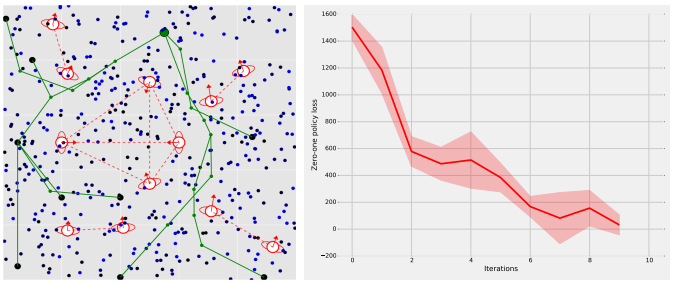

Fig. 1. **Left:** Paths generated using learned rewards showing socially compliant navigation. Colors: green→goal, black→starts, blue→samples (light means recent) using 460 samples in $10m^2$ room. Dotted red lines indicate social links between people (shown with ellipses). **Right:** Policy loss.

We conduct experiments using simple reward features based on distances to people and relations links between them to learn behaviors such as avoid splitting groups, give enough space around persons. We assume that poses of people and the relations between them (represented as lines) are available, and learn policies for moving among the people. See Fig. 1 (left) for resulting paths learned and the algorithm progress.

III. Conclusions and Future Work

We have presented a method for applying IRL algorithms in large state spaces while incorporating important task constraints via an adaptive state graph. We also presented a simple Bayesian IRL algorithm that takes advantage of this representation. We have conducted initial experiments to demonstrate that indeed our approach is able to learn complex behaviors. In the future, we intend to conduct more experiments as well as discuss the key theoretical issues that arise to the sampling and to give guarantees on the learning process.

Acknowledgments

This work has been partly supported by the European Commission under contract number FP7-ICT-600877 (SPENCER).

References

[1] Jaedeug Choi and Kee-Eung Kim. Map inference for bayesian inverse reinforcement learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

[2] Carlos Guestrin and Dirk Ormoneit. Robust combination of local controllers. In *Int. Conf. on Uncertainty in Artificial Intelligence (UAI)*, Seattle, USA, 2001.

[3] Lydia E Kavraki, Petr Svestka, J-C Latombe, and Mark H Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *Robotics and Automation, IEEE Transactions on*, 12(4):566–580, 1996.

[4] Gerhard Neumann, Michael Pfeiffer, and Wolfgang Maass. Efficient continuous-time reinforcement learning with adaptive state graphs. In *European Conference on Machine Learning (ECML)*. Warsaw, Poland, 2007.

[5] Andrew Y Ng and Stuart J Russell. Algorithms for inverse reinforcement learning. In *Int. Conf. on Machine Learning (ICML)*, Haifa, Israel, 2000.

[6] Deepak Ramachandran and Eyal Amir. Bayesian inverse reinforcement learning. In *Int. Joint Conf. on Artificial Intelligence (IJCAI)*, Hyderabad, India, 2007.

[7] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.