Reward Shaping by Demonstration

Halit Bener Suay
Worcester Polytechnic Institute
benersuay@wpi.edu

Tim Brys
Vrije Universiteit Brussel
timbrys@vub.ac.be

Matthew E. Taylor
Washington State University
taylorm@eecs.wsu.edu

Sonia Chernova
Worcester Polytechnic Institute
soniac@wpi.edu

Abstract

Potential-based reward shaping is a theoretically sound way of incorporating prior knowledge in a reinforcement learning setting. While providing flexibility for choosing the potential function, under certain conditions this method guarantees the convergence of the final policy, regardless of the properties of the potential function. However, this flexibility of choice may cause confusion when making a design decision for a specific domain, as the number of possible candidates for a potential function can be overwhelming. Moreover, the potential function either can be manually designed, to bias the behavior of the learner, or can be recovered from prior knowledge, e.g. from human demonstrations. In this paper we investigate the efficacy of two different methods of using a potential function recovered from human demonstrations. Our first approach uses a mixture of Gaussian distributions generated by samples collected during demonstrations (Gaussian-Shaping), and the second approach uses a reward function recovered from demonstrations with Relative Entropy Inverse Reinforcement Learning (RE-IRL-Shaping). We present our findings in Cart-Pole, Mountain Car, and Puddle World domains. Our results show that Gaussian-Shaping can provide an efficient reward heuristic, accelerating learning through its ability to capture local information, and RE-IRL-Shaping can be more resilient to bad demonstrations. We report a brief analysis of our findings and we aim to provide a future reference for reinforcement learning agent designers who consider using reward shaping by human demonstrations.

Keywords: Reinforcement Learning, Inverse Reinforcement Learning,
Reward Shaping, Learning from Demonstration.

Acknowledgements

Halit Bener Suay's work is supported by the Office of Naval Research award N000141410795.

Tim Brys is funded by a PhD scholarship of the Research Foundation Flanders (FWO).

1 Introduction

Incorporating prior knowledge in Reinforcement Learning (RL) in a domain-independent way is an open research question. While there is not a single, domain-generic, *best* way, of incorporating prior knowledge in RL, some options include: using a model of the environment dynamics, having an initial policy, a value function, a reward function, or initial state values [1, 2].

In this paper we explore two new options for incorporating human demonstrations into the reward signal of an RL agent [3]. The first approach we present, *Gaussian-Shaping* [4], takes advantage of the demonstrations locally within the state space and allows demonstrations of different lengths in order to generate multiple Gaussian distributions for reward shaping with a potential function based on a similarity metric (i.e., the distance to the closest demonstrated state-action pair). The second approach, *Relative Entropy Inverse Reinforcement Learning-Shaping* (RE-IRL-Shaping), is based on feature matching and it uses a set of demonstrated trajectories in order to infer a potential function that generalizes over the entire state space for shaping. Feature matching is one of the main approaches to solve the inverse reinforcement learning (IRL) problem. In this approach an IRL algorithm tries to find a set of parameters for a reward function, in order to maximize the reward for experiences with matching feature counts to a demonstration set. RE-IRL algorithm is a model-free algorithm that finds reward feature weights by minimizing the relative entropy between a set of demonstrated features, and features along trajectories that are sampled from a demonstrator's internal policy modeled as a probability distribution. A common property of both of our approaches is that the potential function serves just as a heuristic reward input in the RL setting. An important implication is that when learning just a reward heuristic, a learner does not need to assume that the demonstrator is an expert, or that the outcome of the inferred data is optimal by any metric. This is because the environment reward acts as a consistent, true reward signal that the agent can rely on. Therefore as long as the demonstrations are not deliberately malicious to hinder learning, we can drop the requirement that the demonstrations originate from an expert demonstrator. Note that, unlike our work, most state-of-the-art learning from demonstration algorithms do make the assumption that the demonstrator is an expert for learning a task [5]. We investigate the learning performance of our agents in three commonly used simulated domains: Cart-Pole, Mountain Car, and Puddle World. We aim to help future RL solver designers make more informed design decisions through comparative analysis of the presented methods.

To give an insight into our motivation, we first look at how prior work has incorporated human input in the RL setting. Knox and Stone introduced the TAMER framework where an RL agent receives feedback only from a human in order to model the internal reward signal of the human. Although agents in the TAMER framework do incorporate human input in the RL setting, the framework omits the environment signal altogether, which contrasts with our approach [6].

In a follow up work, Knox and Stone designed a comparative based on analysis of eight different techniques for combining the modeled human reward signal with environment reward. Approaches they introduced varied from using the modeled human reward as a potential function for shaping the environment reward, to simply adding an extra action choice that maximizes the modeled human reward [7]. The general finding of the study is that the combination of a modeled human reward function with a RL agent outperforms the RL agent, and the agent that maximizes the modeled human reward alone. The general idea of using multiple reward functions is similar to our approach in spirit, however our approach is not interactive. Instead we first learn reward functions based on a demonstration data-set. Moreover instead of comparing different techniques for *merging the environment reward* with the human reward signal, we analyze *the use of two different shaping rewards*.

Griffith et al. use simple human feedback such as *right* or *wrong* for shaping a policy in interactive reinforcement learning setting [8]. The approach they present interprets the human feedback as a comment on the optimality of actions. In our work we shape the reward function, and instead of treating the human input as the ground truth, both of our approaches use demonstrations to recover a reward heuristic.

To the best of our knowledge, our choice of potential functions for reward shaping, both using a mixture of Gaussians, and RE-IRL has not been proposed, or studied before. Next, we talk about the two different potential functions we used for reward shaping, the functions' effects on the performance curve of a Q-Learner agent, and our interpretation of the results.

2 Reward Shaping by Demonstration

In this paper we focus on recording the trajectory (s_t, a_t) of an agent throughout multiple episodes of teaching, and using the recorded trajectories for inferring a shaping function. When available, if we rely on the environment reward as a consistent source of reward, we can drop the assumption that the human input has to be optimal. We can shape the standard reward function using a potential function generated based on demonstrations. Even though the demonstrations should still be non-malicious (i.e. not target toward purposefully confusing the learner), reward shaping supplies an agent with a biased, and hopefully more informative reward input. Our baseline agent is a Q-Learning agent, and in both of our methods, the only addition we make is for the shaping of the reward function of a standard Q-Learner. Watkins defines the standard update for a Q-function [3] as

$$Q(s,a) \leftarrow Q(s,a) + \alpha(R(s,a,s') + \gamma \arg\max_{a'} Q(s',a') - Q(s,a)) \tag{1}$$

In our methods, instead of using just a reward function $R(s,a)$ we add a shaping reward $F(s,a,s',a')$ and use a shaped reward $R' = R+F$. We compute the shaping reward F based on a potential function $\Phi(s,a)$ [9, 10]. Next we present how we infer the potential function $\Phi(s,a)$, and the shaping reward $F(s,a,s',a')$ from human demonstrations. We would like to highlight that neither of our techniques require the model of the environment, but only human interaction with the system for collecting demonstration data.

2.1 Using Local Information with Gaussian Distributions

Here we introduce our first method for inferring a potential function Φ using demonstration data, based on sample (i.e. state) similarity given a discrete action a. We show i) how we decide what constitutes a *similar sample*, ii) how we pick the most similar sample within a demonstration set, and once we do, iii) how we use the similarity as a heuristic for reward shaping. We use the following set of equations:

$$\Sigma = \sigma I \tag{2}$$

$$g(s, s^d, \Sigma) = e^{(-\frac{1}{2}(s-s^d)^T \Sigma^{-1}(s-s^d))} \tag{3}$$

$$\Phi(s,a) = \max_{(s^d,a)} g(s, s^d, \Sigma) \tag{4}$$

$$F(s,a,s',a') = \gamma\Phi(s',a') - \Phi(s,a) \tag{5}$$

where, s is the state that is being experienced by the agent at a given time, s^d is one of the states demonstrated which we iterate through, I is an identity matrix, and Σ is the covariance matrix that defines the reach of influence of demonstrated state-action pairs. Σ is domain specific and we tune σ manually, however it is possible to learn metrics, such as Σ, autonomously for RL agents [11]. The intuition for tuning Σ is to set it to larger values for higher dimensional state-spaces. Eq. 3 is a multivariate Gaussian distribution, which outputs the value of a similarity metric that varies between 0 and 1 depending on the distance between the current state and the closest demonstrated sample within the set of demonstrations. Eq. 4 is the potential function for reward shaping, based on which we compute our shaping reward using Eq. 5 [10].

The idea behind our algorithm is simple: once we decide the spread of demonstrated samples and tune σ (Eq. 2), we iterate through demonstrated samples using Eq. 3, and assign the maximum similarity value as the value of the potential for the state (Eq. 4). We compute the similarity metric both for the current state s' and the previous state s of the agent, and compute the difference using Eq. 5, where γ is the discount factor. For a more detailed description and analysis of this approach we refer the readers to Brys et al. [4].

Eq. 3 is the reason why we call this approach *local*. Given an action and a set of demonstrations, encounters with states that are similar to demonstrated states result in high potential for reward shaping. If the agent transitions from a state s that is not similar to any of the demonstrated states, into a state s' that is very similar to one of the demonstrated states, Eq. 5 returns a positive shaping reward indicating that the agent took an action in a *good* direction. In the opposite case, Eq. 5 returns a negative shaping reward.

Even though using sample similarity in this fashion helps to guide the agent toward demonstrated states, this approach imposes a constraint on the quality of the demonstrations themselves. Since currently we do not incorporate the quality of a given sample within our algorithm (i.e. good demonstration vs. bad demonstration), similarity to all demonstrated states is considered to be equally good. That is, if the agent has been demonstrated an action in a state similar to its current state, this can only mean that the action is valuable in that current state. Consequently, if the set of demonstrations includes undesired data points (intentional or unintentional), when the agent experiences states similar to these *undesired* data points, it is possible that the resulting shaping reward will be positive, which would degrade the agent's learning performance. We recognize the importance of automated demonstration quality assessment. One idea for assessing the demonstration quality is to analyze the statistics of the features which may give an idea about the consistency of the data, however we leave this idea for future work, since it deserves to be the main focus of a separate analysis.

2.2 Using RE-IRL As a Heuristic for Reward Shaping

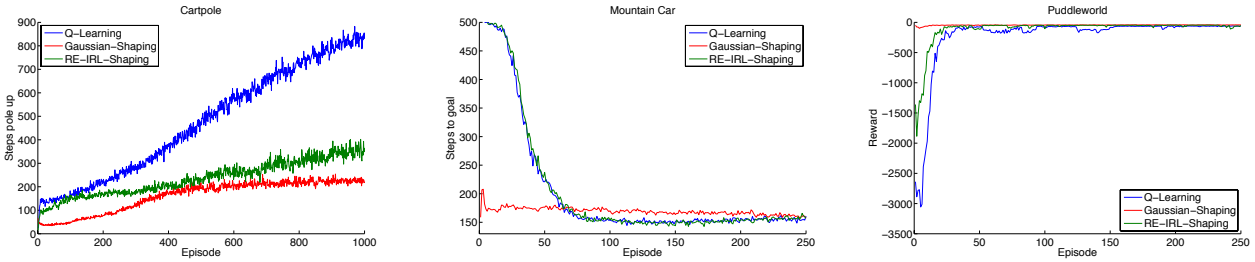
Our second method for inferring a potential function Φ from demonstration data is based on Relative Entropy Inverse Reinforcement Learning (RE-IRL), a model-free inverse reinforcement learning technique introduced by Boularias et al. [12]. The input for RE-IRL is a set of demonstrations which are trajectories of equal length, and the output is a set of reward feature weights ω required to compute a reward value for a given state s. We show i) how we compute a reward using a linear combination of learned feature weights, and ii) how we use the reward as the potential function for reward shaping. The set of equations we use for our approach are:

$$R_{IRL}(s,a) = \sum_{i=0}^{n} \omega_i f_i(s) \tag{6}$$

$$\Phi(s) = R_{IRL}(s,a) \tag{7}$$

$$F(s,a,s',a') = \gamma\Phi(s') - \Phi(s) \tag{8}$$

where, f_i is ith reward feature, ω_i is ith feature weight, s' is the state vector for the current state of the agent, and s is the previous state vector.



(a) (b) (c)

Figure 1: Learning performance results in (a) Cart-Pole, (b) Mountain Car, and (c) Puddle World domains.

Coeff.	Cart-Pole	Mountain Car	Puddle World
α	$\frac{0.25}{16}$	$\frac{0.1}{14}$	$\frac{0.2}{10}$
γ	1.0	1.0	1.0
ϵ	0.05	0.1	0.05
λ	0.25	0.95	0.95
σ	0.2	0.1	0.1

(a)

variable	ω_i
$cart_x$	0.4245
$cart_{\dot{x}}$	0.56805
$pole_\theta$	0.33986
$pole_{\dot{\theta}}$	0.61774

(b)

variable	ω_i
x	-0.999
$\dot{x}$	0.001

(c)

variable	ω_i
x	0.76
y	0.6497

(d)

Table 1: Learning coefficients we used for our experiments (a). Reward feature weights in (b) Cart-Pole, (c) Mountain Car, and (d) Puddle World domains. x and y are position variables; $\dot{x}$ is velocity; θ is angle and $\dot{\theta}$ is angular velocity.

Our approach has two parts: first we learn a reward function by recovering reward feature weights based on a set of demonstrations, and then we use the learned reward function as a potential function for reward shaping. In order to obtain reward feature weights, RE-IRL computes the average feature values (for each feature) over a set of demonstrated trajectories, and uses importance sampling for estimating feature weights in order to iteratively drive a gradient to zero. The gradient is simply the difference between the average (demonstrated) feature values and sampled feature values. For further details of the RE-IRL we refer our readers to [12]. Instead of using the learned function as a reward function (Eq. 6), we define a potential function (Eq. 7) for shaping reward (Eq. 8). This algorithmic setup incorporates a function we learn from human demonstrations into autonomous learning as a shaping reward (or heuristic).

The weights we recover for each reward feature indicate how each feature value affects the reward output. For instance a feature can positively, or negatively dominate others in terms of reward output and understanding each feature's contribution in the reward output can be informative. Moreover RE-IRL computes feature weights using the entire trajectory space and set of demonstrations. As a consequence of average feature matching, this approach is robust against small changes in demonstrations. This *global* use of the demonstration data is in contrast with the previous approach we introduced. If the agent transitions into a state where, the linear combination of the reward features is higher than a previous state, this transition yields a positive shaping reward.

Although we take advantage of the demonstration data on average, in this approach we are limited with the linear combination of the reward features for computing the potential function. This is an important factor to consider when choosing which reward features to use. Even though the reward features can be arbitrarily engineered to design an ad-hoc solution depending on the domain, here for simplicity and compatibility with our previous approach, we chose to use the state variables as reward feature in each domain. We acknowledge that as a future work we need a thorough investigation for choosing the set of reward features.

2.3 Experiments

We tested our approach in three domains using 10 demonstrated trajectories for each domain. The length of demonstrated trajectories respectively were 156, 182, and 39 respectively for Cart-Pole, Mountain Car, and Puddle World domains. In order to keep the demonstration lengths equal, for shorter demonstrations, we appended the final state as an absorbing terminal state at the end of the demonstration. The demonstrations we recorded begin at the initial state and end at the goal (for Mountain Car and Puddle World) *or* the failure state (for Cart-Pole) of the agent. The demonstrator is the first author who has experience with all three domains. The demonstrator made the demonstrations for reaching the goal as quickly as possible in Mountain Car and Puddle World, and for keeping the pole in balance as long as possible for the Cart-Pole domain. Table 1a shows the RL coefficients we used for our experiments.

Table 1b - 1d list the feature weights we use for RE-IRL-Shaping, which we obtained from our set of demonstrations. We can see that in Cart-Pole domain, positive values of state variables result in a positive potential. Although these values are based on demonstrations, one can argue that encouraging the agent toward positive states is not optimal in Cart-Pole. RE-IRL-Shaping performed better than our Gaussian-Shaping algorithm (Fig. 1a). We conclude that this difference

originates from the way our two techniques take advantage of the demonstrations. However, in Cart-Pole the standard Q-Learning agent outperformed both of our approaches.

For RE-IRL-Shaping agent, in the Mountain Car, the position of the car is negatively weighted, and even though positive velocities contribute positively, the effect of the velocity of the car is highly dampened in comparison with the position of the car. This is because the demonstration data includes left and right swings of the car until the car reaches the goal state. The initial state in the domain is $s_0 = [-0.5, 0.0]$, and the shaping reward is positive as long as the car goes to the left. This is the only way for the car to speed up fast enough, so it can climb the hill to reach its goal. This set of weights provides an informative heuristic for the RE-IRL-Shaping agent however its performance is not any better than the standard Q-Learner.

In the Puddle World domain, weights for both state variables (x and y position) yield a high potential as the RE-IRL-Shaping agent goes to the right and up in the world. This aligns well with the purpose of the agent since the goal is on the upper right corner of the world, and we can see the positive effect of this potential function on the learning performance of RE-IRL-Shaping agent especially within the first 30 episodes. This algorithm outperforms the standard Q-Learner, even though Gaussian-Shaping algorithm shows the best overall learning performance.

As shown in Fig. 1b and 1c, the Gaussian-Shaping agent outperforms the standard Q-Learning agent, and RE-IRL-Shaping agent in Mountain Car and Puddle World domains. In both domains, the agents start from an initial *non-goal* state, and the common purpose is to keep moving until reaching a goal state. We use demonstration data not only to help guide the agents toward desired states, but also to initialize the Q-function using Eq. 4, with $Q_0(s, a) = \Phi(s, a)$ as described by Wiewiora et al. [10]. The results show a remarkable boost in the learning performance of Gaussian-Shaping agent in these two domains, starting from the very first episode.

On the other hand, in the Cart-Pole, agents start in a desired equilibrium state, and the purpose is to try to stay close to that initial state. This fundamental difference in the purpose of the task, the level of difficulty of the task for a demonstrator, and the failure (or borderline failure) states that are saved in demonstrations, affect the learning performance of the Gaussian-Shaping agent (Fig. 1a). The performance of the agent suffers from samples that are close to failure states, which highlights the importance of the content of the demonstrations.

3 Conclusion

With this work we present two different potential functions for reward shaping: a function based on similarity of the samples to demonstrated states, and a function learned with inverse reinforcement learning. Both potential functions enable us incorporate prior knowledge in RL. Our analysis in three standard RL domains shows that our sample similarity approach speeds up learning remarkably. However, it is sensitive to states that may be less optimal that are included in the demonstration data-set. On the other hand, even though with the current choice of reward features, our inverse reinforcement learning approach is less effective in improving the learning performance, when the majority of the demonstrations are in agreement, this approach is comparatively more robust against small disturbances in the demonstration data. We leave addressing the noise sensitivity, and reward feature selection issues for future work.

References

[1] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*. Cambridge, MA, USA: MIT Press, 1st ed., 1998.

[2] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *IJRR*, 2013.

[3] C. J. C. H. Watkins, *Learning from delayed rewards*. PhD thesis, University of Cambridge England, 1989.

[4] T. Brys, A. Harutyunyan, H. B. Suay, S. Chernova, M. E. Taylor, and A. Nowé, "Reinforcement learning from demonstration through shaping," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2015.

[5] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, pp. 469–483, May 2009.

[6] W. B. Knox and P. Stone, "Interactively shaping agents via human reinforcement: The tamer framework," in *Proceedings of the fifth international conference on Knowledge capture*, pp. 9–16, ACM, 2009.

[7] W. B. Knox and P. Stone, "Combining manual feedback with subsequent mdp reward signals for reinforcement learning," in *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, 2010.

[8] S. Griffith, K. Subramarian, J. Scholz, C. Isbell, and A. Thomaz, "Policy Shaping: Integrating Human Feedback with Reinforcement Learning," in *Advances in Neural Information Processing Systems*, pp. 2625–2633, 2013.

[9] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *In Proceedings of the Sixteenth International Conference on Machine Learning*, pp. 278–287, Morgan Kaufmann, 1999.

[10] E. Wiewiora, G. W. Cottrell, and C. Elkan, "Principled methods for advising reinforcement learning agents," in *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA*, pp. 792–799, 2003.

[11] M. E. Taylor, B. Kulis, and F. Sha, "Metric learning for reinforcement learning agents," in *The 10th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '11, (Richland, SC), pp. 777–784, International Foundation for Autonomous Agents and Multiagent Systems, 2011.

[12] A. Boularias, J. Kober, and J. Peters, "Relative entropy inverse reinforcement learning," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, pp. 182–189, 2011.