In *Proceedings of the 35th AAAI Conference on Artificial Intelligence* (AAAI 2021),
A Virtual Conference, February 2021

Temporal-Logic-Based Reward Shaping for Continuing Reinforcement Learning Tasks

Yuqian Jiang[1], **Suda Bharadwaj**[2], **Bo Wu**[2], **Rishi Shah**[1,3], **Ufuk Topcu**[2], **Peter Stone**[1,4]
[1] Department of Computer Science, The University of Texas at Austin
[2] Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin
[3] Amazon (work done outside Amazon) [4] Sony AI
{jiangyuqian, suda.b}@utexas.edu, bowu86@gmail.com, rishihahs@gmail.com, utopcu@utexas.edu, pstone@cs.utexas.edu

Abstract

In continuing tasks, average-reward reinforcement learning may be a more appropriate problem formulation than the more common discounted reward formulation. As usual, learning an optimal policy in this setting typically requires a large amount of training experiences. Reward shaping is a common approach for incorporating domain knowledge into reinforcement learning in order to speed up convergence to an optimal policy. However, to the best of our knowledge, the theoretical properties of reward shaping have thus far only been established in the discounted setting. This paper presents the first reward shaping framework for average-reward learning and proves that, under standard assumptions, the optimal policy under the original reward function can be recovered. In order to avoid the need for manual construction of the shaping function, we introduce a method for utilizing domain knowledge expressed as a temporal logic formula. The formula is automatically translated to a shaping function that provides additional reward throughout the learning process. We evaluate the proposed method on three continuing tasks. In all cases, shaping speeds up the average-reward learning rate without any reduction in the performance of the learned policy compared to relevant baselines.

1 Introduction

Reinforcement learning (RL) is a popular method for autonomous agents to learn optimal behavior through repeated interactions with the environment. Most RL algorithms aim to optimize the total *discounted reward* received by the learner. However, in cases involving infinite-horizon or *continuing tasks*, a discount factor can often lead to undesirable behaviors since the agent sacrifices long-term benefits for short-term gains (Mahadevan 1996). Hence, the natural quantity to optimize for many continuing tasks is the *average reward*. However, reinforcement learning with an average reward objective has received very little attention in the literature despite its importance in artificial intelligence (Wan, Naik, and Sutton 2020).

Many robotics applications for RL have delayed or sparse rewards, slowing down learning significantly due to long stretches of possibly uninformative exploration (Mahadevan 1996). *Reward shaping* (Gullapalli and Barto 1992; Mataric 1994) is a common way to inject domain knowledge to guide exploration, but requires care so as not to change the underlying RL problem. (Randløv and Alstrøm 1998) added an intuitive signal to the reward of an agent learning to ride a bicycle, but had the unintended effect of causing the agent to ride in circles. Ng et al. (1999) considered additional rewards that are expressed as the difference of a *potential function* between the agent's current state and its next state and showed that this so-called *potential-based reward shaping* (PBRS) does not affect the optimal policy. Wiewiora et al. (2003) further extended PBRS to include shaping functions of actions as well as states and Devlin & Kudenko (2012) allow time-varying shaping functions. To the best of our knowledge, all of the work in reward shaping concentrates on the discounted-reward setting, and often for Q-learning in particular. However, the reinforcement learning problem under discounted reward objectives is fundamentally incompatible with continuing learning tasks (Naik et al. 2019) and requires a major modification to handle average reward objectives. In this paper, we develop a reward shaping framework for *average-reward* RL.

In general, manually constructing a potential function is difficult since it requires detailed knowledge of the task and the reward structure (Grześ and Kudenko 2010). To address this challenge, previous works provide methods of learning shaping functions online (Grześ and Kudenko 2010) or offline (Marthi 2007). In contrast, our focus is on incorporating side-information or domain knowledge into the shaping function. For example, a user may wish to advise a robot tasked with cleaning to spend more time in one room compared to another. While this statement is intuitive for humans, it can be non-trivial to translate such a statement to an appropriate shaping function. We introduce a method to synthesize the shaping function directly from a temporal logic formula representing the domain knowledge. Temporal logic is a structured natural-like language designed to reason about temporal information, which makes it a suitable choice for non-expert users to encode knowledge for continuing tasks, rather than directly interacting with a mathematical function. Furthermore, manually constructing a shaping function requires taking into account the relative magnitudes of rewards. In contrast, our proposed method allows for a qualitative input of domain knowledge and does not depend on knowledge of the original reward values and the underly-

Copyright © 2021, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

ing RL algorithm.

We test the proposed framework in three continuing learning tasks: continual area sweeping (Ahmadi and Stone 2005; Shah et al. 2020), control of a cart pole in OpenAI gym (Brockman et al. 2016), and motion-planning in a grid world (Mahadevan 1996). We compare with two benchmarks: the baseline differential Q-learning without reward shaping; and shielding, which is a learning method where the learning agent never violates the given temporal logic formula (Alshiekh et al. 2018). The proposed framework improves learning performance from the baseline in terms of sample efficiency. We also show that when the provided domain knowledge is inaccurate, the proposed method still learns the optimal policy and is faster than the baseline differential Q-learning. At the same time, shielding fails to learn the optimal policy as the learner is forced to strictly follow the inaccurate domain knowledge.

Our contributions are as follows: 1) we develop a reward shaping method for average-reward RL, 2) we remove the need for potential function engineering by allowing domain knowledge to be specified as a temporal logic formula, 3) we compare the approach to baseline differential Q-learning without reward shaping as well as shielding to illustrate the advantage of our method.

2 Related Work

The goal of this paper is to provide a framework to facilitate high-level interactions with RL algorithms for users without significant RL expertise. Specifically, we are interested in providing a framework to allow users to provide advice to help increase the rate of learning. As such, we focus on providing guarantees of optimality regardless of the quality of advice provided.

There are several existing areas of research that develop techniques to learn from human knowledge. Learning from demonstration is the problem of learning a policy from examples of a teacher performing a task (Argall et al. 2009; Hussein et al. 2017). Knowledge from a human trainer can be transferred through real-time feedback to improve learning performance (Thomaz and Breazeal 2006; Knox and Stone 2009; Warnell et al. 2018). These techniques usually optimize some combination of inferred human rewards and task rewards. This work focuses on optimizing natural environmental rewards regardless of the quality of knowledge. For continuing tasks, demonstrations or real-time feedback data can be difficult to collect. Our approach takes a temporal-logic specification as input and does not require the human to perform the task or observe the learner during training.

Since we focus on obtaining advice from users without RL expertise, we use temporal logic to sidestep manual reward engineering. In recent years, there has been growing interest in using temporal logic in RL. For example, Icarte et al. proposed using LTL formulas (Toro Icarte et al. 2018b) and later reward machines (Toro Icarte et al. 2018a) to specify reward functions directly. In contrast, our work focuses on using LTL formulas to shape a given reward function from the environment to improve the rate of learning. Shaping has also been used to improve reward machines so that the output reward functions are easier to learn for (Camacho et al. 2018, 2019), but it only exploits the values of different reward machine states. Consider a simple specification "always follow the human." The shaping approach introduced by Camacho et al. (2019) would only differentiate states based on whether the human is visible now. In contrast, our synthesis of the potential function encourages actions that keep the human visible in the future, assuming knowledge of the transition graph.

Some other approaches use temporal logic formulas as constraints that must be satisfied to guarantee safety during learning (Alshiekh et al. 2018; Jansen et al. 2018). To that end, the actions violating the safety formula are overwritten and never taken by the learning agent. In our setting, these actions are not inherently unsafe but simply "not recommended" based on the provided domain knowledge. Not allowing the agent to take these actions can be overly restrictive, and in cases where the provided domain knowledge is not exactly correct, can lead to sub-optimal policies.

3 Preliminaries

In this section, we introduce the definitions and concepts of average-reward reinforcement learning, reward shaping, and temporal logic used throughout the paper.

3.1 Average-Reward Reinforcement Learning

A Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, s_I, \mathcal{A}, R, P)$ is a tuple consisting of a finite set $\mathcal{S}$ of states, an initial state $s_I \in \mathcal{S}$, a finite set $\mathcal{A}$ of actions , reward function $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$, and a probabilistic transition function $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ that assigns a probability distribution over successor states given a state and an action. For a stationary policy $\pi : \mathcal{S} \to \mathcal{A}$, the expected *average reward* is defined as

$$\rho^{\pi}_{\mathcal{M}} := \liminf_{n\to\infty} \frac{1}{n}\mathbb{E}\left[\sum_{k=0}^{n-1} R(s_k, \pi(s_k), s_{k+1})\right]. \quad (1)$$

We are interested in reinforcement learning (RL) problems where the objective is to maximize an average reward over time. More specifically, given an MDP $\mathcal{M}$, where the transition function P and/or reward function R are *unknown* a priori, the objective is to learn an optimal policy π^* such that $\rho^{\pi^*}_{\mathcal{M}} \geq \rho^{\pi}_{\mathcal{M}}$ for any stationary policy π.

Given a policy π, the Q-function $Q^{\pi}_{\mathcal{M}}(s, a)$ is the differential value of taking action a in state s and thereafter following π, and defined as

$$Q^{\pi}_{\mathcal{M}}(s, a) := \mathbb{E}\left[\sum_{k=0}^{\infty} R(s_k, a_k, s_{k+1}) - \rho^{\pi}_{\mathcal{M}} | s_0 = s, a_0 = a\right]. \quad (2)$$

The optimal differential Q-function $Q^*_{\mathcal{M}}$ satisfies the Bellman equation (Sutton, Barto et al. 1998):

$$Q^*_{\mathcal{M}}(s, a) = \mathbb{E}[R(s, a, s') + \max_{a'\in\mathcal{A}}(Q^*_{\mathcal{M}}(s', a'))|P(s, a, s') > 0] - \rho^*_{\mathcal{M}}, \forall s \in \mathcal{S}, a \in \mathcal{A}, \quad (3)$$

where $\rho^*_{\mathcal{M}} = \rho^{\pi^*}_{\mathcal{M}}$, the expected average reward of executing the optimal policy π^*. From $Q^*_{\mathcal{M}}$, the optimal policy π^* can be determined by $\pi^*(s) = \arg\max_{a\in\mathcal{A}}(Q^*_{\mathcal{M}}(s,a))$.

R-learning (Schwartz 1993; Mahadevan 1996) is a classical approach for average-reward RL but lacks theoretical properties. Differential Q-learning (Wan, Naik, and Sutton 2020) is a recent provably convergent approach for learning $Q^*_{\mathcal{M}}$ and $\rho^*_{\mathcal{M}}$, where the standard action-value update step in Q-learning is replaced with one derived from (3):

$$\delta_t \leftarrow r_t + \max_{a_{t+1}} Q_t(s_{t+1}, a_{t+1}) - \rho_t(s_t) - Q_t(s_t, a_t),$$
$$Q_{t+1}(s_t, a_t) \leftarrow Q_t(s_t, a_t) + \alpha_t \delta_t,$$
$$\rho_{t+1}(s_t) \leftarrow \rho_t(s_t) + \eta\alpha_t\delta_t,$$

where δ_t is the temporal-difference (TD) error at time step t, α is the learning rate and η is a positive constant.

3.2 Potential-Based Reward Shaping and Potential-Based Advice

Potential-based reward shaping (PBRS) augments the reward function R by a shaping reward function F, where $F(s,s') = \gamma\Phi(s') - \Phi(s)$, $\Phi : \mathcal{S} \to \mathbb{R}$ is a *potential function*, and $0 < \gamma < 1$ is the discount factor. The resulting MDP is $\mathcal{M}' = (\mathcal{S}, s_I, \mathcal{A}, R', P)$ with $R' = R + F$. It has been proven, for the discounted and episodic settings, that $\mathcal{M}$ and $\mathcal{M}'$ have the same optimal policy (Ng, Harada, and Russell 1999). The goal is to learn the optimal policy in $\mathcal{M}'$ with improved sample efficiency compared to learning in $\mathcal{M}$. Potential-based look-ahead advice (Wiewiora 2003) further extends PBRS such that the potential function also depends on actions: $F(s,a,s',a') = \gamma\Phi(s',a') - \Phi(s,a)$, where a' is chosen by the current policy.

3.3 Temporal Logic

We first define some basic notations. The set of finite sequences w over a set Σ is denoted Σ^*, and the set of infinite sequences is denoted Σ^ω.

Given an MDP $\mathcal{M} = (\mathcal{S}, s_I, \mathcal{A}, R, P)$ and a policy π, a *path* $\sigma = s_0 s_1 s_2 \cdots \in \mathcal{S}^\omega$ is a sequence of states with $s_0 = s_I$ and $P(s_i, \pi(s_i), s_{i+1}) > 0$ for $i \geq 0$. Given a set AP of atomic propositions (Boolean variables), we introduce the *labelling function* $L : \mathcal{S} \to 2^{\mathsf{AP}}$. For a path $\sigma = s_0 s_1 s_2 \cdots \in \mathcal{S}^\omega$, its corresponding label sequence is $w = w_0 w_1 \cdots \in \left(2^{\mathsf{AP}}\right)^\omega$, where $w_i = L(s_i)$.

A linear temporal logic (LTL) formula φ constrains all finite and infinite label sequences $w \in \left(2^{\mathsf{AP}}\right)^\omega \cup \left(2^{\mathsf{AP}}\right)^*$ corresponding to paths in $\mathcal{M}$. For the full semantics of LTL, we refer the reader to (Baier and Katoen 2008). We translate the LTL formula φ to an equivalent *deterministic finite automaton* (DFA) (Kupferman and Vardi 2001). A DFA is a tuple $\mathcal{T}^\varphi = (Q, q_I, 2^{\mathsf{AP}}, \delta, H)$ where Q is a finite set of states, $q_I \in Q$ is the initial state, 2^{AP} is a finite input alphabet, $\delta : Q \times 2^{\mathsf{AP}} \to Q$ is a deterministic transition function and $H \subseteq Q$ is a set of *accepting* states. A *run* $\overline{q}$ of $\mathcal{T}^\varphi$ over label sequence w is an infinite sequence $\overline{q} = q_0 q_1 \cdots \in Q^\omega$ where $q_0 = q_I$ and $q_{i+1} = \delta(q_i, w_i)$. The run is *accepted* by $\mathcal{T}^\varphi$ if $q_i \in H$ for all $i \geq 0$. A label sequence w satisfies φ iff the induced run is accepted by $\mathcal{T}^\varphi$. Only a subclass of LTL formulas can be translated to an equivalent DFA with acceptance condition as presented above. We informally refer to such formulas as *safety formulas* and the translated DFA as a safety automaton (Kupferman and Vardi 2001).

Example 1. *Consider a cleaning robot operating in the environment with a human as shown in Figure 1a, which we model as an MDP on a discrete gridworld shown in Figure 1b. Trash can appear in any state in the grid with a given probability, and the robot is rewarded when it finds trash. Suppose we wish to provide the domain knowledge "the presence of humans tends to increase the probability of trash appearance". We provide this knowledge in the form of advice as an LTL formula "Always human visible" written as* $\Box$ *human_visible. We use this formula to guide the agent during learning to more quickly associate human presence with the appearance of trash. The robot has line-of-sight with a range of 5. We have* $\mathsf{AP} = \{human_visible\}$*, and labelling function* L *that labels all states in visible range as* $\{human_visible\}$ *and all other states as* $\{\neg\ human_visible\}$*. The corresponding DFA is shown in Figure 1c.*

In the above example, trash appears in states with a higher probability if the state is occupied by a human. However, the given formula φ to always keep the human visible, if never violated by the robot, will likely result in the robot learning suboptimal behavior for the house cleaning task since trash could accumulate in other areas as well. In the next sections, instead of relying on LTL formulas to constrain the RL agent's behavior, we discuss how to use a given LTL formula to *guide* the learning of an optimal policy that may violate the formula.

4 Reward Shaping for Average-Reward RL

The main contribution of this paper is a novel approach to provide shaping rewards based on an LTL formula to guide a reinforcement learning agent in the average-reward setting while allowing the optimal policy to be learned even if it violates the given formula. We first define a potential-based shaping function, and prove that the optimal policy can be recovered given any potential function (see supplementary material for the proof).

Theorem 1. *Let* $F : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ *be a shaping function of the form*

$$F(s,a,s') = \Phi\left(s', \arg\max_{a'}(Q^*_{\mathcal{M}}(s',a'))\right) - \Phi(s,a), \quad (4)$$

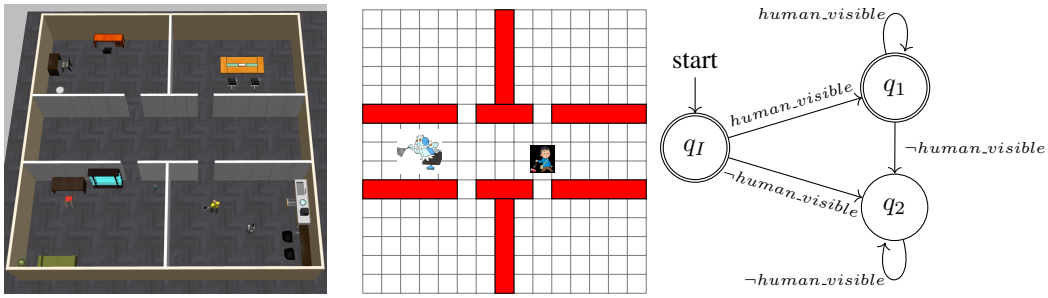
where $\Phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ *is a real-valued function and* $Q^*_{\mathcal{M}}$ *is the optimal average-reward Q-function satisfying (3). Define* $\hat{Q}_{\mathcal{M}'} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ *as*

$$\hat{Q}_{\mathcal{M}'}(s,a) := Q^*_{\mathcal{M}}(s,a) - \Phi(s,a). \quad (5)$$

Then $\hat{Q}_{\mathcal{M}'}$ *is the solution to the following modified Bellman equation in* $\mathcal{M}' = (\mathcal{S}, s_I, A, R', P)$ *with* $R' = R + F$*:*

$$\hat{Q}_{\mathcal{M}'}(s,a) = \mathbb{E}[R'(s,a,s') + \hat{Q}_{\mathcal{M}'}(s',a^*)] - \rho^{\pi^*}_{\mathcal{M}'}, \quad (6)$$

where $a^* = \arg\max_{a'}(\hat{Q}_{\mathcal{M}'}(s',a') + \Phi(s',a'))$*, and* $\rho^{\pi^*}_{\mathcal{M}'}$ *is the expected average reward of the optimal policy in* $\mathcal{M}$*,* $\pi^*_{\mathcal{M}}$*, executed in* $\mathcal{M}'$*.*



(a) Gazebo environment for cleaning robot example.

(b) Grid world representation of (a).

(c) DFA for $\varphi = \Box\ human_visible$.

Figure 1: A service robot example where the robot is rewarded for detecting trash that can appear at any state in the gridworld. Both robot and human can move in any of the cardinal directions.

In MDPs with finite state and action spaces, $\hat{Q}_{\mathcal{M}'}$ is the unique solution to (6) because it forms a fully-determined linear system. After learning $\hat{Q}_{\mathcal{M}'}$ with the shaping rewards in $\mathcal{M}'$, it follows from (5) that the optimal policy π^* can be recovered as:

$$\pi^*(s) = \arg\max_{a \in \mathcal{A}} (\hat{Q}_{\mathcal{M}'}(s,a) + \Phi(s,a)).$$

It is important to note that our goal is not to learn the optimal policy of $\mathcal{M}'$. Rather, we learn a value function $\hat{Q}_{\mathcal{M}'}$ from which it is possible to recover the optimal policy of $\mathcal{M}$. Note that $\hat{Q}_{\mathcal{M}'}$ is not the same as $Q^*_{\mathcal{M}'}$ where $\hat{Q}_{\mathcal{M}'}$ satisfies (6) but $Q^*_{\mathcal{M}'}$ satisfies the following Bellman optimality equation:

$$Q^*_{\mathcal{M}'}(s,a) = \mathbb{E}[R(s,a,s') + \max_{a' \in \mathcal{A}} (Q^*_{\mathcal{M}'}(s',a'))] - \rho^*_{\mathcal{M}'}.$$

In practice, the shaping rewards in (4) require knowledge of $Q^*_{\mathcal{M}}(s',a')$, which can be bootstrapped by $\hat{Q}_{\mathcal{M}'}(s',a') + \Phi(s',a')$ as its estimate. When shaping is applied to differential Q-learning, action selection is also based on $\hat{Q}_{\mathcal{M}'}(s,a) + \Phi(s,a)$ rather than $\hat{Q}_{\mathcal{M}'}(s,a)$.

5 Potential Function Synthesis

We now present an algorithm for constructing a potential function Φ from a given temporal logic formula φ. Informally, the reward shaping function captures the information in the formula φ by *penalizing* the learning agent for visiting the states from which a violation of φ *can occur with a non-zero probability*. We illustrate this concept with an example.

Example 2. *Consider again the example shown in Figure 1. In order to not violate the formula $\varphi = \Box\ human_visible$, the robot must "plan ahead" to ensure that no matter what the human does in the future, the robot will still not lose visibility of them. Figure 1b, shows a case where the current state does not cause a violation of φ as the robot can still see the human. However, there is a non-zero probability that the human will move south into a room, and the robot will not be able to maintain visibility and hence will violate φ.*

Informally, we compute the set of state-action pairs from which the probability of violating φ is 0. We refer to this set $\mathcal{W}$ as the *almost-sure winning region*. Some knowledge of the dynamics, i.e., the graph structure of $\mathcal{M}$ (not necessarily the transition probabilities), is necessary to "plan ahead" and compute $\mathcal{W}$. To this end, we define a *product MDP* $\overline{\mathcal{M}}_\varphi = \mathcal{M} \times \mathcal{T}^\varphi$. Given an MDP $\mathcal{M} = (\mathcal{S}, s_I, \mathcal{A}, R, P)$, a DFA $\mathcal{T}^\varphi = (Q, q_I, 2^{\mathsf{AP}}, \delta, H)$, and a labelling function L a product MDP is defined as $\overline{\mathcal{M}}_\varphi = \mathcal{M} \times \mathcal{T}^\varphi = \left(V, v_I, \mathcal{A}, \Delta, \overline{H}\right)$ where $V = S \times Q$ is the joint set of states, $v_I = (s_I, Q_I) = (s_I, \delta(q_I, L(s_I)))$ is the initial state, $\Delta : V \times \Sigma \times V \to [0,1]$ is the probability transition function such that $\Delta((s,q),a,(s',q')) = P(s,a,s')$, if $\delta(q, L(s')) = q'$, and 0, otherwise, and $\overline{H} = (\mathcal{S} \times H) \subseteq V$ is the set of accepting states.

In order to compute the almost-sure winning region $\mathcal{W} \subseteq \mathcal{S} \times \mathcal{A}$ in the MDP $\mathcal{M}$, we first compute the set $\mathcal{W}_0^{min}(\overline{H}) \subseteq V \times \mathcal{A}$ of states and corresponding actions that have a minimum probability of 0 of leaving $\overline{H}$. This set can be computed using graph-based methods in $\mathcal{O}(V \times \mathcal{A})$ (Baier and Katoen 2008). An algorithm for computing the set as well as ways to improve scalability for very large state spaces are provided in the supplementary material. We can then extract $\mathcal{W}$ from $\mathcal{W}_0^{min}(\overline{H})$ by defining $\mathcal{W} := \{(s,a) \in \mathcal{S} \times \mathcal{A} \mid (s,q,a) \in \mathcal{W}_0^{min}(\overline{H})\}$.

Given $\mathcal{W}$, we construct Φ as:

$$\Phi(s,a) = \begin{cases} C & (s,a) \in \mathcal{W} \\ d(s,a) & (s,a) \notin \mathcal{W} \end{cases} \tag{7}$$

where $C \in \mathbb{R}$ is an arbitrary constant, and $d : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is any function such that $d(s,a) < C$ for all s, a. This construction incentivizes the agent to follow the advice by ensuring that state-action pairs in the almost-sure winning region are assigned higher potential values.

Example 3. *Continuing the cleaning robot example, we set $C = 1$ and $d(s,a)$ as the negative of the distance between s and the closest state in $\mathcal{W}$. The results are detailed in Section 6.1.*

The hyperparameters are formulated generally to allow encoding helpful knowledge about the domain if it is available (such as the Manhattan distance in a grid world), but they can be set arbitrarily as long as $d(s,a) < C$. The results are not sensitive to the precise values, as the shaping

function takes the relative difference between the potential values.

Remark 1. *We note that while the method presented in this section can be applied for any safety formula, the synthesized potential function will not, in general, be Markovian. Since the goal of this paper is to help the agent learn an optimal policy with respect to a Markovian reward function $R(s, a)$, it suffices to consider Markovian potential functions. Restricting φ to a subclass of safety formulas called invariant formulas will guarantee Φ is Markovian (Baier and Katoen 2008).*

6 Experiments

We evaluate our temporal-logic-based reward shaping framework in three average-reward RL benchmarks: continual area sweeping, cart pole, and continuing grid world. In all scenarios, we compare against the standard differential Q-learning approach that we refer to as the *baseline* and the same differential Q-learning approach where actions violating the given temporal logic formula are disallowed using the method introduced by Alshiekh et al. (2018). We refer to the latter as *shielding*, and we refer the framework presented in this paper as *shaping*. Furthermore, we study cases when the optimal learned policy must violate the given formula - in such situations, we refer to the provided formula as *imperfect advice*.

We show that 1) our reward shaping framework speeds up baseline average-reward RL, 2) shaping cannot outperform shielding with perfect advice, but 3) when the advice is imperfect, shaping still learns the optimal policy faster than the baseline, whereas shielding fails to learn the optimal policy. All source code is available as supplementary material.

6.1 Continual Area Sweeping

The problem of a robot sweeping an area repeatedly and non-uniformly for some task has been formalized as *continual area sweeping*, where the goal is to maximize the average reward per unit time without assuming the distribution of rewards (Shah et al. 2020). We run the set of experiments on the environment in Figure 1b, where the robot has a maximum speed of 3, i.e. it can move to any cell within a Manhattan distance of 3. We study three cases. First, the robot operates without a human and is given the formula $\Box\ kitchen$ where the kitchen is the bottom right room of Figure 1b. Second, a human is also present, and the robot is given the formula $\Box\ human_visible$ as described in Examples 1-3. Third, a human is present and there is trash appearing in the corridor, enabling us to study how these approaches handle complex formulae such as conjunctions. The second and the third cases are examples where it is difficult to hand-craft a function, since the winning region is non-trivial, but the advice can be captured with simple and intuitive LTL formulas. In all three scenarios, we compare our framework against the DQN-based deep average-reward RL approach introduced by Shah et al. (2020), with their R-learning update replaced by the recent differential Q-learning approach.

Always kitchen In this scenario, the kitchen has the most cleaning needs, and the given formula is to always stay in the kitchen. State-action pair $(s, a) \in \mathcal{W}$ if $L(s) = \{kitchen\}$ and $L(s') = \{kitchen\}$ for all s' such that $P(s, a, s') > 0$. The potential function Φ is constructed as Equation 7, where $C = 1$ and $d(s, a)$ is the negative of the minimal distance between s and the kitchen and plus 1 if a gets closer to the kitchen. For the shielding approach, (s, a) is allowed only if $(s, a) \in \mathcal{W}$ or a will decrease the distance to the kitchen from s.

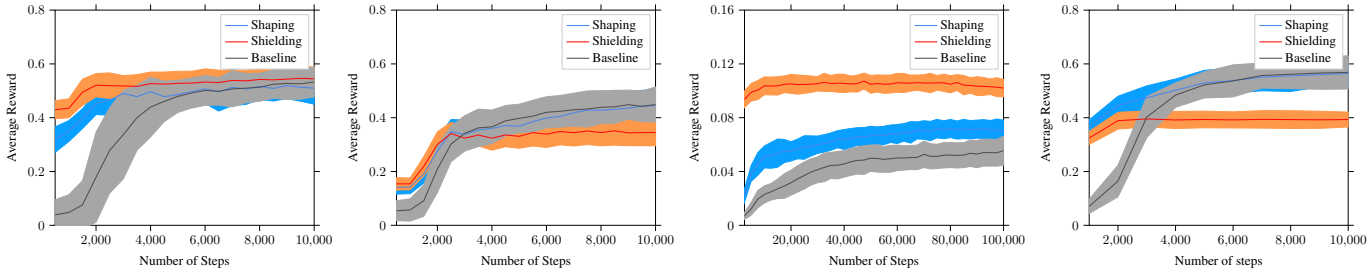
Experiments are conducted in the following two scenarios: a) cleaning is required only in the kitchen and nowhere else, and b) cleaning can be required in the kitchen and some states that are randomly selected from the right half of the corridor. Each cell that might require cleaning is assigned a frequency between $1/20$ and $1/10$ so that the robot has to learn an efficient sweeping strategy among those cells. At each time step, there is also a 0.2 probability that a dirty cell no longer needs cleaning (such as trash gets picked up by people, the wet floor dries as time passes).

Figure 2a reports the learning curves when the formula is perfect, i.e., trash only appears in the kitchen. While shielding performs well from the beginning, shaping quickly catches up, and both learn significantly faster than the baseline. Figure 2b reports the results in the second case where there are unexpected rewards outside of the specified region. Shielding gets a head-start by blocking actions away from the region, but it fails to discover the unexpected reward outside the region. Shaping performs better than the baseline at the beginning and converges to a better policy that sweeps in the corridor too.

Always keep human visible Consider the scenario described in Examples 1-3. At every time step, there is a 0.2 probability that the current position of the human needs cleaning. There is also a 0.2 probability that a dirty cell becomes clean by itself with every step. The human moves randomly between the corridor and the top left room and has a speed of 1 cell per step. The state space includes the current location of the human if the human is visible. Φ is constructed as described in Example 3. When the human is invisible to the robot, the winning region is unknown, and $d(s, a)$ is set to -6.

Figure 2c shows the results when the human is the only source of trash appearance. Shielding outperforms shaping and the baseline by forcing the agent to always follow the human. The learning curves of shaping and baseline methods do not converge to the optimal average reward because softmax action selection is used throughout learning. The exploratory actions can lose sight of the human, which is very costly in this task.

Always keep human visible and always corridor Suppose each cell in the corridor is assigned a frequency between $1/20$ and $1/10$ to require cleaning besides the cell that the human occupies at each step. In this case, the best strategy is to always follow the human in the corridor and keep sweeping the corridor when the human goes into the top left



(a) Trash only appears in the kitchen.

(b) Trash can occur outside the kitchen as well.

(c) Trash only appears where the human has been.

(d) Trash can appear in other states as well.

Figure 2: Learning rate comparison for shaping, shielding, and the baseline in the two continual area sweeping tasks - $\Box$ *kitchen* (top row) and $\Box$ *human_visible* (bottom row). In (a) and (b), the average reward of every 500 steps is plotted, averaged by 100 runs. (c) plots the average reward every 2500 steps, and (d) plots the average reward every 1000 steps, averaged by 100 runs. Only (c) shows the average reward for 100,000 steps (while the rest show 10,000 steps) because the reward is much sparser when trash only appears with human presence. The shaded areas represent one standard deviation from the mean.

room. Shielding cannot use the conjunction of always keeping human visible and always staying in the corridor because there are no allowed actions when the human leaves the corridor. Shaping can easily combine both pieces of advice by adding a potential function based on distance to corridor to the potential function in Example 3. Figure 2d shows that the shielding method fails to learn the optimal policy if it is forced to strictly follow the human and misses the reward of staying the corridor; the shaping method is able to leverage both specifications even though they sometimes contradict each other, and outperforms both shielding and the baseline.

6.2 Cart Pole

Cart pole is a widely studied classic control problem in the reinforcement learning literature. The agent is a cart with a pole attached to a revolute joint. Situated on a horizontal track, the agent must keep its pole balanced upright by applying a force to the right or the left. Each state is composed of the position x, the linear velocity $\dot{x}$, the pole angle θ, and the angular velocity $\dot{\theta}$. At each step, the agent receives reward of 1 if $x \in [-2.4, 2.4]$ and $\theta \in [-\pi/15, \pi/15]$. Otherwise, the episode terminates. We modify the cart pole environment in OpenAI Gym (Brockman et al. 2016) to a continuing task by removing the termination conditions. As shown in Figure 3a, we allow the cart to move anywhere on the x-axis and the pole to be at any angle, and restrict the max linear velocity by $|\dot{x}| \leq 1$. Other aspects of the motion model are kept the same, and reward is given under the same conditions. Therefore, the agent has to learn a policy that keeps the cart position in the scoring range (i.e., $x \in [-2.4, 2.4]$) and swings the pole up when it falls down.

We compare our shaping framework with the same baseline deep differential Q-learning and shielding. The specification is to keep the cart position in the scoring range. We test the methods in the case of not knowing the full dynamics model, and approximate the winning region by predicting the next x position as $x_{t+1} = x_t + \dot{x}_t \Delta t$ where Δt is the time between steps. The shield blocks the action in the direction of x_{t+1} if x_{t+1} is not in the scoring range. $\Phi(s, a)$ is constructed with $C = 1$ and $d(s, a)$ as the minimal distance between the scoring range and x_{t+1} (and multiplied by a constant[1]).

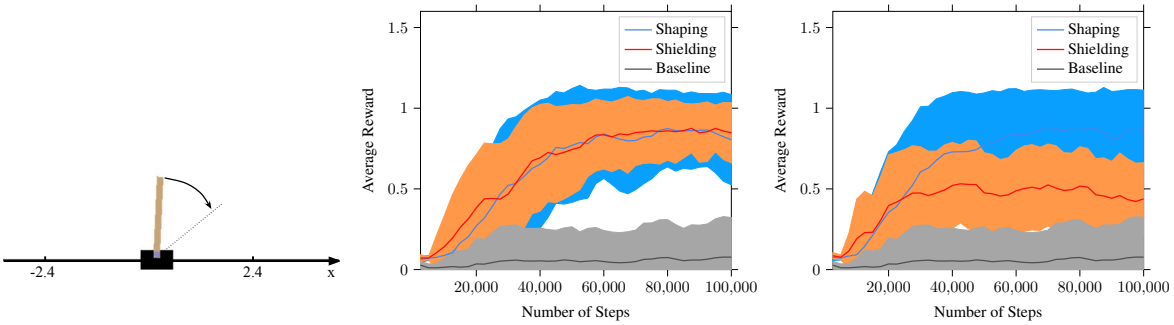
Figure 3b plots the learning performance when the accurate scoring range $[-2.4, 2, 4]$ is given to shaping and shielding. Figure 3c shows the results when an inaccurate scoring range $[-2, 2]$ is used. Because the cart can leave the scoring range and never reset, the baseline gets extremely sparse reward and fails to solve the problem in a reasonable amount of time in most trials. Shaping and shielding are both able to significantly improve the learning performance, but when the knowledge is inaccurate, shielding over-restricts action choices and achieves worse average reward than shaping. The results also show that shaping is robust to only knowing approximate dynamics.

6.3 Grid World

We further evaluate our method in a continuing grid world environment. The purpose of this experiment is to include a common benchmark that has been studied in the context of both average-reward RL (Mahadevan 1996) and reward shaping (Ng, Harada, and Russell 1999), and showcase that our temporal-logic-based approach performs as well as the hand-crafted shaping function. Figure 4a shows the 6*6 grid with a +100 reward in the green cell. The agent can move to a neighboring cell in one of the four cardinal directions deterministically at every step, and gets "transported" to a random cell when it reaches the green cell and when it fails to reach the green cell after 100 steps.

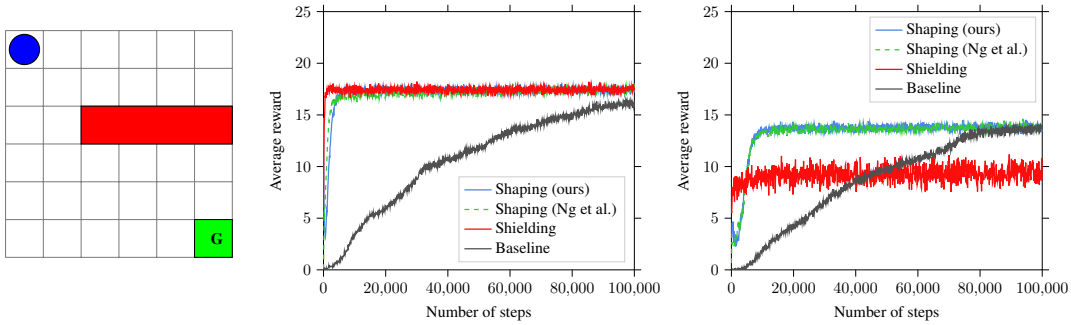
The following four methods are compared: our temporal-logic-based reward shaping, hand-crafted reward shaping, shielding, and the baseline. Shielding blocks actions up and left and only allows actions down and right. For temporal-logic-based shaping, the winning region consists of all states, but only with the actions down and right. The poten-

[1]It is intuitive to penalize the agent for going out of the scoring range by scaling $d(s, a)$. We tried scaling factors of 1, 10, and 100, and saw the most improvements with 100. Overall, no fine tuning was required for the hyperparameters C and $d(s, a)$.



(a) Continuing cart pole setup. (b) Perfect knowledge. (c) Unexpected cart range.

Figure 3: Learning rate comparison between shaping, shielding, and the baseline in the continuing cart pole task



(a) Grid world setup. (b) Perfect knowledge. (c) Unexpected wall.

Figure 4: Grid world experiments comparing the performance of temporal-logic-based reward shaping (ours), hand-crafted reward shaping (Ng et al.), shielding, and baseline. The average reward of every 100 steps is plotted, averaged by 100 runs.

tial function Φ is constructed with $C = 1$ and $d(s, a) = -1$. The hand-crafted potential function is introduced by Ng et al. (1999), where $\Phi(s)$ is defined as the negative of the distance from s to the green cell. The correctness of the advice is varied by adding the red wall from (2, 2) to (2, 5) as shown in Figure 4a. When the agent is directly above the wall, it cannot reach the green cell by only moving down or right.

Figure 4 compares the learning performance of each method in the two conditions. The observations are consistent with the results reported in the other benchmarks. Shielding outperforms other methods when the advice is perfect, i.e., there is no wall, and fails to reach the maximum average reward when the wall is added. Our temporal-logic-based reward shaping method performs similarly to the hand-crafted shaping function while allowing the reward shaping to be automatically constructed. Both shaping methods still outperform the baseline differential Q-learning when the advice is imperfect.

7 Conclusion

We provide a reward shaping framework to construct shaping functions from a temporal logic specification to improve performance of average-reward RL. We prove and empirically show that our shaping framework speeds up the learning rate, and allows the optimal policy to be learned when the provided advice is imperfect.

Since this is the first work to extend reward shaping to the average-reward setting (to the best of our knowledge), we focus on providing theoretical guarantees and intuitive examples rather than evaluations in a large suite of benchmarks. Given that average-reward RL remains under-studied and thus lacks a wide range of benchmarks, this paper also contributes the Continual Area Sweeping task and our continuing version of Cart Pole as new possible benchmarks.

This paper focuses on the (not uncommon) case when the user has some knowledge of the dynamics. In robotics applications, knowledge of the underlying graph is a reasonable assumption as the high-level dynamic capabilities of the robot are generally known even if there is low-level variation which is captured by the detailed (unknown) transition probabilities. The proposed approach is also robust to the user only knowing approximate dynamics, similar to imperfect advice. One direction for future research is extending the work to apply when the dynamics or models are fully unknown.

Another interesting direction for future work that this paper opens up is studying *adversarial* advice - i.e., formulas provided to actively hinder the learning process. We are also interested in expanding the learning problem to include both non-Markovian reward structures and shaping functions.

Acknowledgement

This work was partially supported by ONR N00014-20-1-2115, ARO W911NF-20-1-0140.

This work has partially taken place in the Learning Agents Research Group (LARG) at UT Austin. LARG research is supported in part by NSF (CPS-1739964, IIS-1724157, NRI-1925082), ONR (N00014-18-2243), FLI (RFP2-000), ARO (W911NF-19-2-0333), DARPA, Lockheed Martin, GM, and Bosch. Peter Stone serves as the Executive Director of Sony AI America and receives financial compensation for this work. The terms of this arrangement have been reviewed and approved by the University of Texas at Austin in accordance with its policy on objectivity in research.

We would like to thank all reviewers of this paper for their constructive feedback.

Ethics Statement

The guiding philosophy behind the work in this paper is that the barrier to entry for reinforcement learning is too high in terms of domain expertise and data availability. We believe this is particularly exacerbated in robotics applications as robotics has become integrated in various research fields, many of which involve no computer scientists and data collection is costly. RL as a tool has huge potential to unlock the use of robotics for research in many fields, but often requires a significant amount of tuning, training, and expertise to take advantage of properly. Our goal is to make this paper be the first of a series of papers aimed at facilitating high-level interaction with RL algorithms without requiring significant RL expertise and improving the sampling efficiency.

Since the work at its current stage is still theoretical, there is little direct negative societal consequences. However, this work, as is the case for most robotics-related research, contributes to the increased adoption of robots and autonomous systems in society. In particular, RL allows autonomous systems to achieve complex tasks that they were previously not capable of. In this paper, we use cleaning robots as the motivating case study throughout the paper. Autonomous systems being able to reliably complete complex tasks has the potential to displace the service industries, surveillance, and search and rescue, where the tasks are repetitive or even dangerous.

While we acknowledge the potential job loss that can occur from maturation of such technology, we believe the unlocking the potential of RL to non-experts can lead to the acceleration of research in many fields.

References

Ahmadi, M.; and Stone, P. 2005. Continuous area sweeping: A task definition and initial approach. In *ICAR'05. Proceedings., 12th International Conference on Advanced Robotics, 2005.*, 316–323. IEEE.

Alshiekh, M.; Bloem, R.; Ehlers, R.; Könighofer, B.; Niekum, S.; and Topcu, U. 2018. Safe reinforcement learning via shielding. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Argall, B. D.; Chernova, S.; Veloso, M.; and Browning, B. 2009. A survey of robot learning from demonstration. *Robotics and autonomous systems* 57(5): 469–483.

Baier, C.; and Katoen, J.-P. 2008. *Principles of model checking*. MIT press.

Brockman, G.; Cheung, V.; Pettersson, L.; Schneider, J.; Schulman, J.; Tang, J.; and Zaremba, W. 2016. OpenAI Gym. *arXiv e-prints* arXiv:1606.01540.

Camacho, A.; Chen, O.; Sanner, S.; and McIlraith, S. A. 2018. Non-Markovian rewards expressed in LTL: Guiding search via reward shaping (extended version). In *GoalsRL, a workshop collocated with ICML/IJCAI/AAMAS*.

Camacho, A.; Icarte, R. T.; Klassen, T. Q.; Valenzano, R. A.; and McIlraith, S. A. 2019. LTL and Beyond: Formal Languages for Reward Function Specification in Reinforcement Learning. In *IJCAI*, volume 19, 6065–6073.

Devlin, S. M.; and Kudenko, D. 2012. Dynamic potential-based reward shaping. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, 433–440. IFAAMAS.

Grześ, M.; and Kudenko, D. 2010. Online learning of shaping rewards in reinforcement learning. *Neural Networks* 23(4): 541–550.

Gullapalli, V.; and Barto, A. G. 1992. Shaping as a method for accelerating reinforcement learning. In *Proceedings of the 1992 IEEE International Symposium on Intelligent Control*, 554–559.

Hussein, A.; Gaber, M. M.; Elyan, E.; and Jayne, C. 2017. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)* 50(2): 1–35.

Jansen, N.; Könighofer, B.; Junges, S.; and Bloem, R. 2018. Shielded Decision-Making in MDPs. *CoRR* abs/1807.06096. URL http://arxiv.org/abs/1807.06096.

Knox, W. B.; and Stone, P. 2009. Interactively Shaping Agents via Human Reinforcement: The TAMER Framework. In *The Fifth International Conference on Knowledge Capture*.

Kupferman, O.; and Vardi, M. Y. 2001. Model checking of safety properties. *Formal Methods in System Design* 19(3): 291–314.

Mahadevan, S. 1996. Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine learning* 22(1-3): 159–195.

Marthi, B. 2007. Automatic Shaping and Decomposition of Reward Functions. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, 601–608. New York, NY, USA: Association for Computing Machinery. ISBN 9781595937933. doi:10.1145/1273496.1273572. URL https://doi.org/10.1145/1273496.1273572.

Mataric, M. J. 1994. Reward functions for accelerated learning. In *Machine Learning Proceedings 1994*, 181–189. Elsevier.

Naik, A.; Shariff, R.; Yasui, N.; and Sutton, R. S. 2019. Discounted reinforcement learning is not an optimization problem. *arXiv preprint arXiv:1910.02140* .

Ng, A. Y.; Harada, D.; and Russell, S. J. 1999. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, 278–287. Morgan Kaufmann Publishers Inc.

Randløv, J.; and Alstrøm, P. 1998. Learning to Drive a Bicycle Using Reinforcement Learning and Shaping. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, 463–471. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN 1558605568.

Schwartz, A. 1993. A reinforcement learning method for maximizing undiscounted rewards. In *Proceedings of the Tenth International Conference on International Conference on Machine Learning*, 298–305. Morgan Kaufmann Publishers Inc.

Shah, R.; Jiang, Y.; Hart, J.; and Stone, P. 2020. Deep R-Learning for Continual Area Sweeping. *arXiv e-prints* arXiv:2006.00589.

Sutton, R. S.; Barto, A. G.; et al. 1998. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge.

Thomaz, A. L.; and Breazeal, C. 2006. Reinforcement Learning with Human Teachers: Evidence of Feedback and Guidance with Implications for Learning Performance. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, AAAI'06, 1000–1005. AAAI Press. ISBN 9781577352815.

Toro Icarte, R.; Klassen, T.; Valenzano, R.; and McIlraith, S. 2018a. Using reward machines for high-level task specification and decomposition in reinforcement learning. In *International Conference on Machine Learning*, 2107–2116.

Toro Icarte, R.; Klassen, T. Q.; Valenzano, R.; and McIlraith, S. A. 2018b. Teaching multiple tasks to an RL agent using LTL. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, 452–461. International Foundation for Autonomous Agents and Multiagent Systems.

Wan, Y.; Naik, A.; and Sutton, R. S. 2020. Learning and Planning in Average-Reward Markov Decision Processes. *arXiv e-prints* arXiv:2006.16318.

Warnell, G.; Waytowich, N.; Lawhern, V.; and Stone, P. 2018. Deep TAMER: Interactive agent shaping in high-dimensional state spaces. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

Wiewiora, E. 2003. Potential-based shaping and Q-value initialization are equivalent. *Journal of Artificial Intelligence Research* 19: 205–208.

Wiewiora, E.; Cottrell, G. W.; and Elkan, C. 2003. Principled methods for advising reinforcement learning agents. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, 792–799.