Explicable Reward Design for Reinforcement Learning Agents

Rati Devidze[1] rdevidze@mpi-sws.org
Goran Radanovic[1] gradanovic@mpi-sws.org
Parameswaran Kamalaruban[2] kparameswaran@turing.ac.uk
Adish Singla[1] adishs@mpi-sws.org

[1]Max Planck Institute for Software Systems (MPI-SWS), Saarbrucken, Germany
[2]The Alan Turing Institute, London, UK

Abstract

We study the design of *explicable* reward functions for a reinforcement learning agent while guaranteeing that an optimal policy induced by the function belongs to a set of target policies. By being explicable, we seek to capture two properties: (a) *informativeness* so that the rewards speed up the agent's convergence, and (b) *sparseness* as a proxy for ease of interpretability of the rewards. The key challenge is that higher informativeness typically requires dense rewards for many learning tasks, and existing techniques do not allow one to balance these two properties appropriately. In this paper, we investigate the problem from the perspective of discrete optimization and introduce a novel framework, EXPRD, to design explicable reward functions. EXPRD builds upon an informativeness criterion that captures the (sub-)optimality of target policies at different time horizons in terms of actions taken from any given starting state. We provide a mathematical analysis of EXPRD, and show its connections to existing reward design techniques, including potential-based reward shaping. Experimental results on two navigation tasks demonstrate the effectiveness of EXPRD in designing explicable reward functions.

1 Introduction

A reward function plays the central role during the learning/training process of a reinforcement learning (RL) agent. Given a "task" the agent is expected to perform (i.e., the desired learning outcome), there are typically many different reward specifications under which an optimal policy has the same performance guarantees on the task. This freedom in choosing the reward function, in turn, leads to the fundamental question of reward design: *What are different criteria that one should consider in designing a reward function for the agent, apart from the agent's final output policy?* [1–3].

One of the important criteria is *informativeness*, capturing that the rewards should speed up the agent's convergence [1–6]. For instance, a major challenge faced by an RL agent is because of delayed rewards during training; in the worst-case, the agent's convergence is slowed down exponentially w.r.t. the time horizon of delay [7]. In this case, we seek to design a new reward function that reduces this time horizon of delay while guaranteeing that any optimal policy induced by the designed function is also optimal under the original reward function [3]. The classical technique of potential-based reward shaping (when applied with appropriate state potentials) indeed allows us to reduce this time horizon of delay to 1; see [3, 8] and Section 2. With 1, it means that globally optimal actions for any state are also myopically optimal, thereby making the agent's learning process trivial.

While informativeness is an important criterion, it is not the only criterion to consider when designing rewards for many practical applications. Another natural criterion to consider is *sparseness* as a proxy for ease of interpretability of the rewards. There are several practical settings where sparseness and interpretability of rewards are important, as discussed next. The first motivating application is when

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

rewards are designed for human learners who are learning to perform sequential tasks, for instance, in pedagogical applications such as educational games [9], virtual reality-based training simulators [10, 11], and solving open-ended problems (e.g., block-based visual programming [12]). In this context, tasks can be challenging for novice learners and a teacher agent can assist these learners by designing explicable rewards associated with these tasks. The second motivating application is when rewards are designed for complex compositional tasks in the robotics domain that involve reward specifications in terms of logic, automata, or subgoals [13, 14]—these specifications induce a form of sparsity structure on the underlying reward function. The third motivating application is related to defense against reward-poisoning attacks in RL (see [15–19]) by designing structured and sparse reward functions that are easy to debug/verify. Beyond these practical settings, many naturally occurring reward functions in real-life tasks are inherently sparse and interpretable, further motivating the need to distill these properties in the automated reward design process. The key challenge is that higher informativeness typically requires dense rewards for many learning tasks – for instance, the above-mentioned potential-based shaped rewards that achieve a time horizon of 1 would require most of the states be associated with some real-valued reward (see Sections 2 and 4). To this end, an important research question that we seek to address is: *How to balance these two criteria of informativeness and sparseness in the reward design process while guaranteeing an optimality criterion on policies induced by the reward function?*

In this paper, we formalize the problem of designing *explicable* reward functions, focusing on the criteria of informativeness and sparseness. We investigate this problem from an expert/teacher's point of view who has full domain knowledge (in this case, an original reward function along with optimal policies induced by the original function), and seeks to design a new reward function for the agent—see Figure 1 and further discussion in Section 5 on expert-driven vs. agent-driven reward design. We tackle the problem from the perspective of discrete optimization and introduce a novel framework, ExpRD, to design reward functions. ExpRD allows us to appropriately balance informativeness and sparseness while guaranteeing that an optimal policy induced by the function belongs to a set of target policies. ExpRD builds upon an informativeness criterion that captures the (sub-)optimality of target policies at different time horizons from any given starting state. Our main contributions are:[1]

I. We formulate the problem of explicable reward functions to balance the two important criteria of informativeness and sparseness in the reward design process. (Sections 2 and 3.1)

II. We propose a novel optimization framework, ExpRD, to design reward functions. As part of this framework, we introduce a new criterion capturing informativeness of reward functions that is amenable to optimization techniques and is of independent interest. (Sections 3.2 and 3.3)

III. We provide a detailed mathematical analysis of ExpRD and show its connections to popular techniques, including potential-based reward shaping. (Sections 3.3 and 3.4)

IV. We provide a practical extension to apply our framework to large state spaces. We perform extensive experiments on two navigation tasks to demonstrate the effectiveness of ExpRD in designing explicable reward functions. (Sections 3.5 and 4)

2 Problem Setup

Environment. An environment is defined as a Markov Decision Process (MDP) $M := (\mathcal{S}, \mathcal{A}, T, \gamma, R)$, where the set of states and actions are denoted by $\mathcal{S}$ and $\mathcal{A}$ respectively. $T : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ captures the state transition dynamics, i.e., $T(s' \mid s, a)$ denotes the probability of landing in state s' by taking action a from state s. Here, γ is the discounting factor. The underlying reward function is given by $R : \mathcal{S} \times \mathcal{A} \rightarrow [-R_{\max}, R_{\max}]$, for some $R_{\max} > 0$. We interchangeably represent the reward function by a vector $R \in \mathbb{R}^{|\mathcal{S}| \cdot |\mathcal{A}|}$, whose $(s\,|\mathcal{A}| + a)$-th entry is given by $R\,(s, a)$. We define the support of R as $\mathrm{supp}(R) := \{s : s \in \mathcal{S}, R\,(s, a) \neq 0 \text{ for some } a \in \mathcal{A}\}$, and the ℓ_0-norm of R as $\|R\|_0 := |\mathrm{supp}(R)|$.

Preliminaries and definitions. We denote a stochastic policy $\pi : \mathcal{S} \rightarrow \Delta\,(\mathcal{A})$ as a mapping from a state to a probability distribution over actions, and a deterministic policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ as a mapping from a state to an action. For any policy π, the state value function V^{π}_{∞} and the action value function Q^{π}_{∞} in the MDP M are defined as follows respectively: $V^{\pi}_{\infty}\,(s) = \mathbb{E}\,[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | s_0 = s, T, \pi]$ and $Q^{\pi}_{\infty}\,(s, a) = \mathbb{E}\,[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s, a_0 = a, T, \pi]$. Further, the optimal value functions are given by $V^{*}_{\infty}\,(s) = \sup_{\pi} V^{\pi}_{\infty}\,(s)$ and $Q^{*}_{\infty}\,(s, a) = \sup_{\pi} Q^{\pi}_{\infty}\,(s, a)$. There always exists a

[1] Github repo: https://github.com/adishs/neurips2021_explicable-reward-design_code.

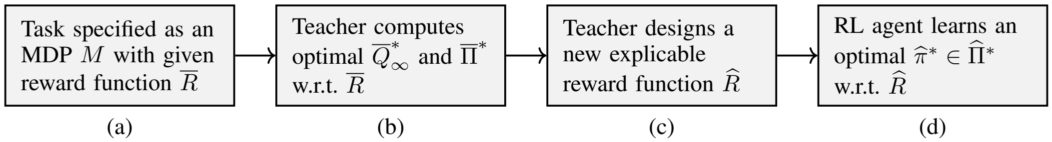


Figure 1: Illustration of the explicable reward design problem in terms of a task specified through MDP M, an RL agent whose objective is to perform this task, and a teacher/expert whose objective is to help this RL agent. **(a)** MDP M with a given reward function $\overline{R}$ specifying the task the RL agent is expected to perform; **(b)** The teacher computes optimal action value function $\overline{Q}^*_\infty$ along with the set of optimal policies $\overline{\Pi}^*$ w.r.t. $\overline{R}$; **(c)** The teacher designs a new explicable reward function $\widehat{R}$ for the RL agent; **(d)** The RL agent trains using the designed reward $\widehat{R}$ and outputs a policy $\widehat{\pi}^*$ from the set of optimal policies $\widehat{\Pi}^*$ w.r.t. $\widehat{R}$. Our framework designs an explicable reward function $\widehat{R}$ with three properties: *invariance*, *informativeness*, and *sparseness*; see main text for formal definitions of these properties.

deterministic stationary policy π that achieves the optimal value function simultaneously for all $s \in \mathcal{S}$ [7, 20], and we denote all such deterministic optimal policies by the set $\Pi^* := \{\pi : \mathcal{S} \to \mathcal{A} \text{ s.t. } V^\pi_\infty(s) = V^*_\infty(s), \forall s \in \mathcal{S}\}$. From here onwards, we focus on deterministic policies unless stated otherwise. For any π and R, we define the following quantities that capture the ∞-step (global) optimality gap and the 0-step (myopic) optimality gap of action a at state s, respectively:

$$\delta^\pi_\infty(s,a) := Q^\pi_\infty(s,\pi(s)) - Q^\pi_\infty(s,a), \text{ and } \delta^\pi_0(s,a) := Q^\pi_0(s,\pi(s)) - Q^\pi_0(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A},$$

where $Q^\pi_0(s,a) = R(s,a)$ is the 0-step action value function of policy π. The $\delta^\pi_\infty(s,a)$ values are same for all $\pi \in \Pi^*$, and we denote it by $\delta^*_\infty(s,a) = V^*_\infty(s) - Q^*_\infty(s,a)$; however, this is not the case with $\delta^\pi_0(s,a)$ values in general. For any state $s \in \mathcal{S}$ and a set of policies Π, we define $\Pi_s := \{a : a = \pi(s), \pi \in \Pi\}$. Then, we have that $\delta^*_\infty(s,a) = 0, \forall s \in \mathcal{S}, a \in \Pi^*_s$.

Explicable reward design. Figure 1 presents an illustration of the explicable reward design problem that we formalize below. A task is specified as an MDP M with a given goal-based reward function $\overline{R}$ where $\overline{R}$ has non-zero rewards only on goal states $\mathcal{G} \subseteq \mathcal{S}$, i.e., $\overline{R}(s,a) = 0, \forall s \in \mathcal{S} \backslash \mathcal{G}, a \in \mathcal{A}$. Many naturally occurring tasks (see Section 1 for motivating applications) are goal-based and challenging for learning an optimal policy when the state space $\mathcal{S}$ is very large. In this paper, we study the following explicable reward design problem from an expert/teacher's point of view: Given $\overline{R}$ and the corresponding optimal policy set $\overline{\Pi}^*$ w.r.t. $\overline{R}$ as the input, the teacher designs a new reward function $\widehat{R}$ with criteria of *informativeness* and *sparseness* while guaranteeing an *invariance* requirement (these properties are formalized in Section 3). Informally, the invariance requirement is that any optimal policy learned using the new reward $\widehat{R}$ belongs to the optimal policy set $\overline{\Pi}^*$ induced by $\overline{R}$.[2]

Typical techniques for reward design and issues. Given a set of important states (subgoals) in the environment, one could design a handcrafted reward function $\widehat{R}_{\text{CRAFT}}$ by assigning non-zero reward values only to these states. Even though this simple approach produces a reward function with a specified sparsity level, it often fails to satisfy the invariance requirement. In particular, there are some well-known "reward bugs" that can arise in this approach and mislead the agent into learning sub-optimal policies (see [2, 3]). In the seminal work [3], the authors introduced the potential-based reward shaping (PBRS) method to alleviate this issue. The reward function produced by the PBRS method with optimal value function $\overline{V}^*_\infty$ under $\overline{R}$ as the potential function is defined as follows:

$$\widehat{R}_{\text{PBRS}}(s,a) \;:=\; \overline{R}(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s' \mid s,a) \cdot \overline{V}^*_\infty(s') - \overline{V}^*_\infty(s). \tag{1}$$

The set of optimal policies $\widehat{\Pi}^*$ induced by $\widehat{R}_{\text{PBRS}}$ is exactly equal to the set of optimal policies $\overline{\Pi}^*$ induced by $\overline{R}$ since $\widehat{\delta}^\pi_\infty(s,a) = \overline{\delta}^*_\infty(s,a)$ for all $\pi \in \overline{\Pi}^*$ [3]. In addition, for any state $s \in \mathcal{S}$, globally optimal actions $\overline{\Pi}^*_s \subseteq \mathcal{A}$ under $\overline{R}$ are also myopically optimal under $\widehat{R}_{\text{PBRS}}$ since $\widehat{\delta}^\pi_0(s,a) = \overline{\delta}^*_\infty(s,a)$ for all $\pi \in \overline{\Pi}^*$ [3, 8] – this leads to a dramatic speed-up in the learning process. However, the potential-based reward shaping produces dense reward function which is less interpretable (see Section 4).

[2]In the rest of the paper, the quantities defined corresponding to $R := \overline{R}$ are denoted by an *overline*, e.g., the optimal policy set by $\overline{\Pi}^*$ and the ∞-step optimality gaps by $\overline{\delta}^*_\infty$; the quantities defined corresponding to $R := \widehat{R}$ are denoted by a *widehat*, e.g., the optimal policy set by $\widehat{\Pi}^*$.

3 Our Reward Design Framework ExpRD

In Sections 3.1, 3.2, and 3.3, we propose an optimization formulation and a greedy solution for the explicable reward design problem. In Section 3.4, we provide a theoretical analysis of our greedy solution. In Section 3.5, we provide a practical extension to apply our framework to large state spaces.

3.1 Discrete Optimization Formulation

Given $\overline{R}$ and the corresponding optimal policy set $\overline{\Pi}^*$, we systematically develop a discrete optimization framework (ExpRD) to design an explicable reward function $\widehat{R}$ (see Figure 1).

Sparseness, informativeness, and invariance. The sparseness of the reward function $\widehat{R}$ is captured by $\mathrm{supp}(\widehat{R})$. In Section 3.2, we formalize an informativeness criterion $I(\widehat{R})$ of $\widehat{R}$ that captures how hard/easy it is to learn an optimal behavior induced by $\widehat{R}$. We explicitly enforce the invariance requirement (see Section 2) for the new reward $\widehat{R}$ by choosing a set of candidate policies $\Pi^\dagger \subseteq \overline{\Pi}^*$, and satisfying the following (Bellman-optimality) conditions:

$$Q^{\pi^\dagger}_\infty(s,a) \;=\; \widehat{R}(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s'|s,a) \cdot Q^{\pi^\dagger}_\infty(s', \pi^\dagger(s')), \quad \forall a \in \mathcal{A}, s \in \mathcal{S}, \pi^\dagger \in \Pi^\dagger \quad \text{(C.1)}$$

$$Q^{\pi^\dagger}_\infty(s,\pi^\dagger(s)) \;\geq\; Q^{\pi^\dagger}_\infty(s,a) + \overline{\delta}^*_\infty(s), \quad \forall a \in \mathcal{A}\backslash\overline{\Pi}^*_s, s \in \mathcal{S}, \pi^\dagger \in \Pi^\dagger, \quad \text{(C.2)}$$

where $\overline{\delta}^*_\infty(s) := \min_{a \in \mathcal{A}\backslash\overline{\Pi}^*_s} \overline{\delta}^*_\infty(s,a), \forall s \in \mathcal{S}$.[3] The above conditions guarantee that any optimal policy induced by $\widehat{R}$ is also optimal under $\overline{R}$, i.e., $\Pi^\dagger \subseteq \widehat{\Pi}^* \subseteq \overline{\Pi}^*$. Here, the set $\Pi^\dagger \subseteq \overline{\Pi}^*$ is used to reduce the number of constraints. Note that for the potential-based shaped reward $\widehat{R}_{\text{PBRS}}$, we have $\widehat{\Pi}^* = \overline{\Pi}^*$.

Maximizing informativeness for a given set of important states. When a domain expert provides us a set of important states (subgoals) in the environment [21–24], we want to use this set in a principled way to design a reward $\widehat{R}$, while avoiding the "reward bugs" that can arise from hand-crafted rewards $\widehat{R}_{\text{Craft}}$. To this end, for any given set of subgoals $\mathcal{Z} \subseteq \mathcal{S}\backslash\mathcal{G}$, we optimize the informativeness criterion $I(R)$ while satisfying the invariance requirement:

$$\begin{aligned} g(\mathcal{Z}) \;:=\; & \max_{R:\mathrm{supp}(R) \subseteq \mathcal{Z} \cup \mathcal{G}} \quad I(R) \\ & \quad\text{subject to} \quad \text{conditions (C.1)} - \text{(C.2) with } \widehat{R} \text{ replaced by } R \text{ hold} \\ & \qquad\qquad\qquad |R(s,a)| \;\leq\; R_{\max}, \; \forall s \in \mathcal{S}, a \in \mathcal{A}. \end{aligned} \quad \text{(P1)}$$

Let $R^{(\mathcal{Z})}$ denote the R that maximizes $g(\mathcal{Z})$. Let $\mathcal{R} \subseteq \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$ be a constraint set on R that captures only the conditions (C.1) – (C.2) and the $R_{\max}$ bound.

Jointly finding subgoals along with maximizing informativeness. Based on (P1), we propose the following discrete optimization formulation that allows us to select a set of important states (of size B) and design a reward function that maximizes informativeness automatically:

$$\max_{\mathcal{Z}:\mathcal{Z} \subseteq \mathcal{S}\backslash\mathcal{G}, |\mathcal{Z}| \leq B} g(\mathcal{Z}). \quad \text{(P2)}$$

We can incorporate prior knowledge about the quality of subgoals using a set function $D : 2^{\mathcal{S}} \to \mathbb{R}$ (we assume D to be a submodular function [25]). Finally, the full ExpRD formulation is given by:

$$\max_{\mathcal{Z}:\mathcal{Z} \subseteq \mathcal{S}\backslash\mathcal{G}, |\mathcal{Z}| \leq B} g(\mathcal{Z}) + \lambda \cdot D(\mathcal{Z} \cup \mathcal{G}), \;\; \text{for some } \lambda \geq 0. \quad \text{(P3)}$$

We study the problems (P1), (P2), and (P3) in the following subsections.

[3] Note that the true action values $\overline{Q}^*_\infty$ are used in the conditions (C.1) – (C.2) to obtain the terms $\overline{\delta}^*_\infty(s,a)$, $\mathcal{A}\backslash\overline{\Pi}^*_s$, and $\Pi^\dagger$. However, when we only have an approximate estimate of $\overline{Q}^*_\infty$, we can adapt (C.1) – (C.2) appropriately with approximate versions of $\overline{\delta}^*_\infty(s,a)$, $\mathcal{A}\backslash\overline{\Pi}^*_s$, and $\Pi^\dagger$.

3.2 Informativeness Criterion

Understanding the informativeness of a reward function is an important problem, and several works have investigated it [4, 5, 26–28]. Our goal is to define an informativeness criterion that is amenable to optimization techniques. As noted in Section 2, for any policy $\pi \in \overline{\Pi}^*$, 0-step and ∞-step optimality gaps induced by $\widehat{R}_{\mathrm{PBRS}}$ are all equal to ∞-step optimality gaps induced by $\overline{R}$, i.e., $\widehat{\delta}_0^\pi(s,a) = \widehat{\delta}_\infty^\pi(s,a) = \overline{\delta}_\infty^*(s,a)$. For any reward function R, one could ask how much these two quantities could differ, and even consider the intermediate cases between 0-step and ∞-step optimality. Inspired by the h-step optimality notions studied in [4, 26], we define the h-step action value function of any policy π as $Q_h^\pi(s,a) = \mathbb{E}\left[\sum_{t=0}^h \gamma^t R(s_t,a_t) | s_0 = s, a_0 = a, T, \pi\right]$, and it satisfies the following recursive relationship: $Q_h^\pi(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s'|s,a) \cdot Q_{h-1}^\pi(s', \pi(s'))$.

Let $\mathcal{H}$ be a set of horizons for which we want to maximize informativeness. For any policy π and reward function R, we define the following quantity that captures the h-step optimality gap of action a at state s: $\delta_h^\pi(s,a) := Q_h^\pi(s,\pi(s)) - Q_h^\pi(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}, h \in \mathcal{H}$. Later, in the proof of Proposition 2, we show that $\delta_h^\pi(s,a)$ is linear in R, i.e., $\delta_h^\pi(s,a) = \langle w_{h;(s,a)}, R \rangle$ for some vector $w_{h;(s,a)} \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$. Interestingly, the following proposition states that, for any policy $\pi \in \overline{\Pi}^*$ and any h, the h-step optimality gap induced by $\widehat{R}_{\mathrm{PBRS}}$ given in (1) is equal to the ∞-step optimality gap induced by $\overline{R}$:

Proposition 1. *The goal-based reward function $\overline{R}$, and the potential-based shaped reward function $\widehat{R}_{\mathrm{PBRS}}$ given in (1) satisfy the following: $\widehat{\delta}_h^\pi(s,a) = \overline{\delta}_\infty^*(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}, \pi \in \overline{\Pi}^*, h \in \mathcal{H}$.*

Let $\ell : \mathbb{R} \to \mathbb{R}$ be a monotonically non-decreasing concave function. Then, based on the h-step optimality gaps, we define the informativeness criterion of the reward R as follows:

$$I_\ell(R) \ := \ \sum_{\pi^\dagger \in \Pi^\dagger} \sum_{h \in \mathcal{H}} \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A} \setminus \overline{\Pi}_s^*} \ell(\delta_h^{\pi^\dagger}(s,a)).$$

From here onwards, we let I be I_ℓ in the problem (P1). As an example for ℓ, we consider the negated hinge loss given by $\ell_{\mathrm{hg}}(\delta(s,a)) := -\max(0, \overline{\delta}_\infty^*(s,a) - \delta(s,a))$. By Proposition 1, we have that $I_{\ell_{\mathrm{hg}}}(\widehat{R}_{\mathrm{PBRS}}) = 0$, and $I_{\ell_{\mathrm{hg}}}(R) \leq 0$ for any other R, i.e., $\widehat{R}_{\mathrm{PBRS}}$ achieves the maximum value of $I_{\ell_{\mathrm{hg}}}$.

3.3 Iterative Greedy Algorithm

First, we show that the problem (P1) can be efficiently solved using the standard concave optimization methods to find $R^{(\mathcal{Z})}$ for any given $\mathcal{Z} \subseteq \mathcal{S} \backslash \mathcal{G}$:

Proposition 2. *For any given $\mathcal{Z} \subseteq \mathcal{S} \backslash \mathcal{G}$, the problem (P1) is a concave optimization problem in $R \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$ with linear constraints. Further, the feasible set of the problem (P1) is non-empty.*

Then, inspired by the Forward Stepwise Selection method from [29], we propose an iterative greedy solution (see Algorithm 1) to solve the problems (P2) and (P3). To compute the incremental gain at each step, we would need to solve the concave optimization problem (P1) for different values of $\mathcal{Z}$. The problem (P1) has $|\mathcal{S}| \cdot |\mathcal{A}|$ optimization variables and $\mathcal{O}(|\mathcal{S}| \cdot |\mathcal{A}| \cdot |\Pi^\dagger| \cdot |\mathcal{H}|)$ constraints.

Algorithm 1 Iterative Greedy Algorithm for ExpRD

1: **Input:** MDP $\overline{M} := (\mathcal{S}, \mathcal{A}, T, \gamma, \overline{R})$, $\overline{\delta}_\infty^*(s,a)$ values, sets $\overline{\Pi}^*, \overline{\Pi}^\dagger, \mathcal{G}, \mathcal{H}$, sparsity budget B
2: **Initialize:** $\mathcal{Z}_0 \leftarrow \emptyset$
3: **for** $k = 1, 2, \ldots, B$ **do**
4: $\quad z_k \leftarrow \arg\max_{z \in \mathcal{S} \backslash \mathcal{Z}_{k-1}} g(\mathcal{Z}_{k-1} \cup \{z\}) + \lambda \cdot D(\mathcal{Z}_{k-1} \cup \mathcal{G} \cup \{z\}) - g(\mathcal{Z}_{k-1}) - \lambda \cdot D(\mathcal{Z}_{k-1} \cup \mathcal{G})$
5: $\quad \mathcal{Z}_k \leftarrow \mathcal{Z}_{k-1} \cup \{z_k\}$
6: **Output:** $\mathcal{Z}_B$ and the corresponding optimal reward function $R^{(\mathcal{Z}_B)}$.

3.4 Theoretical Analysis

Here, we provide guarantees for the solution returned by our Algorithm 1. Below, we give an overview of the main technical ideas, and leave a detailed discussion along with proofs in the

Appendix. For some $\mu \geq 0$, let $I_\ell^{\text{reg}}(R) := I_\ell(R) - \mu \|R\|_2^2$ be the regularized informativeness criterion. We define a normalized set function $f : 2^{\mathcal{S}} \to \mathbb{R}$ as follows:

$$f(\mathcal{Z}) = \max_{R:\text{supp}(R) \subseteq \mathcal{Z} \cup \mathcal{G}, R \in \mathcal{R}} (I_\ell^{\text{reg}}(R) - I_\ell^{\text{reg}}(R^{(\emptyset)})) + \lambda \cdot (D(\mathcal{Z} \cup \mathcal{G}) - D(\mathcal{G})), \tag{2}$$

where $R^{(\emptyset)} = \arg\max_{R:\text{supp}(R) \subseteq \mathcal{G}, R \in \mathcal{R}} I_\ell^{\text{reg}}(R)$. Note that the regularized variant (I_ℓ replaced by I_ℓ^{reg}) of the optimization problem (P3) is equivalent to $\max_{\mathcal{Z}: \mathcal{Z} \subseteq \mathcal{S} \setminus \mathcal{G}, |\mathcal{Z}| \leq B} f(\mathcal{Z})$. For a given sparsity budget B, let $\mathcal{Z}_B^{\text{Greedy}}$ be the set selected by our Algorithm 1 and $\mathcal{Z}_B^{\text{OPT}}$ be the optimal set that maximizes the regularized variant of problem (P3). The corresponding f values of these sets are denoted by f_B^{Greedy} and f_B^{OPT} respectively; in the following, we are interested in comparing these two values. The problem (P3) is closely related to the subset selection problem studied in [29] with a twist of an additional constraint set $\mathcal{R}$ (see the discussion after (P1)), making the theoretical analysis more challenging. Inspired by the analysis in [29], we need to prove a weak form of submodularity [25, 30] for f (since D is already a submodular function, we need to prove this for the case when $\lambda = 0$). To this end, we require the regularized informativeness criterion I_ℓ^{reg} to satisfy certain structural assumptions. First, we define the restricted strongly concavity and restricted smoothness notions of a function that are used in our analysis.

Definition 1 (Restricted Strong Concavity, Restricted Smoothness [31])**.** *A function $\mathcal{L} : \mathbb{R}^{|\mathcal{S}| \cdot |\mathcal{A}|} \to \mathbb{R}$ is said to be restricted strong concave with parameter m_Ω and restricted smooth with parameter M_Ω on a domain $\Omega \subset \mathbb{R}^{|\mathcal{S}| \cdot |\mathcal{A}|} \times \mathbb{R}^{|\mathcal{S}| \cdot |\mathcal{A}|}$ if for all $(x, y) \in \Omega$:*

$$-\frac{m_\Omega}{2} \|y - x\|_2^2 \;\geq\; \mathcal{L}(y) - \mathcal{L}(x) - \langle \nabla \mathcal{L}(x), y - x \rangle \;\geq\; -\frac{M_\Omega}{2} \|y - x\|_2^2 .$$

For any integer k, we define the following two sets: $\Omega_k := \{(x, y) : \|x\|_0 \leq k, \|y\|_0 \leq k, \|x - y\|_0 \leq k, x, y \in \mathcal{R}\}$, and $\tilde{\Omega}_k := \{(x, y) : \|x\|_0 \leq k, \|y\|_0 \leq k, \|x - y\|_0 \leq 1, x, y \in \mathcal{R}\}$. Let $m_k := m_{\Omega_k}$ and $M_k := M_{\Omega_k}$ (similarly we define $\tilde{m}_k$ and $\tilde{M}_k$).

When there is no $R \in \mathcal{R}$ constraint in (2), the following assumption on the regularized informativeness criterion is sufficient to prove the weak submodularity of f [29]:

Assumption 1. *The regularized informativeness criterion I_ℓ^{reg} is $m_{2B+|\mathcal{G}|}$-restricted strongly concave and $M_{2B+|\mathcal{G}|}$-restricted smooth on $\Omega_{2B+|\mathcal{G}|}$.*

However, due to the additional $R \in \mathcal{R}$ constraint, we need to enforce further requirements on I_ℓ^{reg} formally captured in Assumption 2 provided in the Appendix; here, we discuss these requirements informally. Let $\mathcal{Z}$ be any set such that $\mathcal{Z} \subseteq \mathcal{S} \setminus \mathcal{G}$, and $\nabla I_\ell^{\text{reg}}(R^{(\mathcal{Z})})$ be the gradient of the regularized informativeness criterion at the optimal reward $R^{(\mathcal{Z})}$. Then, we need to ensure the following: (i) the ℓ_2-norm of the projection of $\nabla I_\ell^{\text{reg}}(R^{(\mathcal{Z})})$ on $(\mathcal{Z} \cup \mathcal{G})$ is upper-bounded, captured by $d_{\max}^{\text{opt}}$; (ii) the ℓ_2-norm of the projection of $\nabla I_\ell^{\text{reg}}(R^{(\mathcal{Z})})$ on any $j \in \mathcal{S} \setminus (\mathcal{Z} \cup \mathcal{G})$ is lower-bounded, captured by $d_{\min}^{\text{non}}$; and (iii) the components of the optimal reward $R^{(\mathcal{Z})}$ outside $(\mathcal{Z} \cup \mathcal{G})$ do not lie in the boundary of $\mathcal{R}$, captured by κ. Then, by using Assumption 1 and Assumption 2 (see Appendix), we prove the weak submodularity of f. Finally, by applying Theorem 3 from [29], we obtain the following theorem:

Theorem 1. *Let I_ℓ^{reg} satisfies Assumption 1 and Assumption 2 requirements. Then, we have $f_B^{\text{Greedy}} \geq (1 - e^{-\gamma}) f_B^{\text{OPT}}$, where $\gamma = \frac{\kappa \cdot m_{2B+|\mathcal{G}|}}{M_{2B+|\mathcal{G}|}} \cdot \frac{(d_{\min}^{\text{non}})^2}{(d_{\max}^{\text{opt}})^2 + (d_{\min}^{\text{non}})^2}$.*

We provide Assumption 2 and a detailed proof of the theorem in the Appendix.

3.5 Extension to Large State Spaces using State Abstractions

This section presents an extension of our ExpRD framework that is scalable to large state spaces by leveraging the techniques from state abstraction literature [32–34]. We use an abstraction $\phi : \mathcal{S} \to \mathcal{X}_\phi$, which is a mapping from high-dimensional state space $\mathcal{S}$ to a low-dimensional latent space $\mathcal{X}_\phi$. Let $\phi^{-1}(x) := \{s \in \mathcal{S} : \phi(s) = x\}, \forall x \in \mathcal{X}_\phi$, and $\overline{M} := (\mathcal{S}, \mathcal{A}, T, \gamma, \overline{R})$. We propose the following pipeline:

1. By using $\overline{M}$ and ϕ, we construct an abstract MDP $\overline{M}_\phi = (\mathcal{X}_\phi, \mathcal{A}, T_\phi, \gamma, \overline{R}_\phi)$ as follows, $\forall x, x' \in \mathcal{X}_\phi, a \in \mathcal{A}$: $T_\phi(x'|x, a) = \frac{1}{|\phi^{-1}(x)|} \sum_{s \in \phi^{-1}(x)} \sum_{s' \in \phi^{-1}(x')} T(s'|s, a)$, and $\overline{R}_\phi(x, a) = \frac{1}{|\phi^{-1}(x)|} \sum_{s \in \phi^{-1}(x)} \overline{R}(s, a)$. We compute the set of optimal policies $\overline{\Pi}_\phi^*$ for the MDP $\overline{M}_\phi$.

2. We run our ExpRD framework on $\overline{M}_\phi$ with $\Pi^\dagger = \overline{\Pi}^*_\phi$, and the resulting reward is denoted $\widehat{R}_\phi$.
3. We define the reward function $\widehat{R}$ on the state space $\mathcal{S}$ as follows: $\widehat{R}(s,a) = \widehat{R}_\phi(\phi(s),a)$.

By assuming certain structural conditions on ϕ formalized in the Appendix, we can show that any optimal policy induced by the above reward $\widehat{R}$ acts nearly optimal w.r.t. $\overline{R}$. This pipeline can be extended to continuous state space as well, similar to [34–36]. We provide more details in the Appendix.

4 Experimental Evaluation

In this section, we evaluate ExpRD on two environments: RoomsNavEnv (Section 4.1) and LineKeyNavEnv (Section 4.2). RoomsNavEnv corresponds to a navigation task in a grid-world where the agent has to learn a policy to quickly reach the goal location in one of four rooms, starting from an initial location. Even though this environment has a small state space, it provides a very rich and an intuitive problem setting to validate different reward design techniques, and variants of RoomsNavEnv have been used extensively in the literature [14, 21, 22, 37–40]. LineKeyNavEnv corresponds to a navigation task in a one-dimensional space where the agent has to first pick the key and then reach the goal. The agent's location in this environment is represented as a point on a line segment. Given the large state space representation, it is computationally challenging to apply the reward design technique from Section 3.3 and we use the state abstraction-based extension of our framework from Section 3.5. This environment is inspired by variants of navigation tasks in the literature where an agent needs to perform sub-tasks [3, 41]. We give an overview of main results here, and provide a more detailed description of the setup and additional results in the Appendix.

4.1 Evaluation on RoomsNavEnv

RoomsNavEnv (Figure 2). We represent the environment as an MDP with $\mathcal{S}$ states each corresponding to cells in the grid-world indicating the agent's current location (shown as "blue-circle"). Goal (shown as "green-star") is located at the top-right corner cell. The agent can take four actions given by $\mathcal{A} := \{\text{“up”}, \text{“left”}, \text{“down”}, \text{“right”}\}$. An action takes the agent to the neighbouring cell represented by the direction of the action; however, if there is a wall (shown as "brown-segment"), the agent stays at the current location. Furthermore, when an agent takes an action $a \in \mathcal{A}$, there is p_{rand} probability that an action $a' \in \mathcal{A} \setminus \{a\}$ will be executed instead of a. In addition to these walls, there are a few terminal walls (shown as "thick-red-segment") that terminates the episode—at the bottom-left corner cell, "left" and "down" actions terminate; at the top-right corner cell, "right" action terminates. The agent gets a reward of $R_{\max}$ after it has navigated to the goal and then takes a "right" action (i.e., only one state-action pair has a reward); note that this action also terminates the episode. The reward is 0 for all other state-action pairs and there is a discount factor γ. This MDP has $|\mathcal{S}| = 49$ and $|\mathcal{A}| = 4$; we set $p_{\text{rand}} = 0.1$, $R_{\max} = 10$, and $\gamma = 0.95$ in our evaluation.

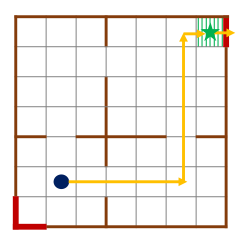
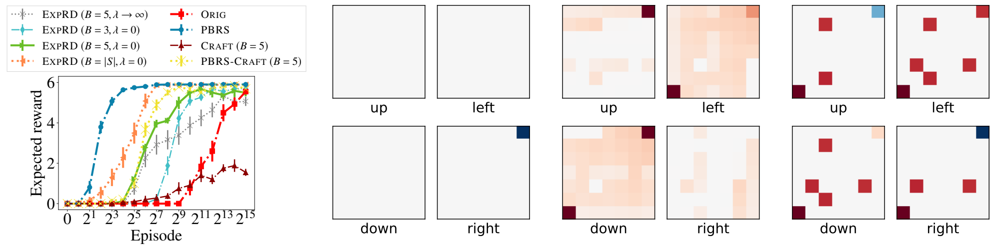


Figure 2: RoomsNavEnv

Techniques evaluated. We consider the following baselines: (i) $\widehat{R}_{\text{Orig}} := \overline{R}$, which simply represents default reward function, (ii) $\widehat{R}_{\text{PBRS}}$ obtained via the PBRS technique with the optimal value function $\overline{V}^*_\infty$ w.r.t. $\overline{R}$ (see Section 2), (iii) $\widehat{R}_{\text{Craft}}$ that we design manually (see Section 2 and description below), and (iv) $\widehat{R}_{\text{PBRS-Craft}(B=5)}$ obtained via the PBRS technique with the optimal value function w.r.t. $\widehat{R}_{\text{Craft}}$ instead of $\overline{V}^*_\infty$ [42].[4] To design $\widehat{R}_{\text{Craft}}$, we first hand-crafted a set function D that assigns scores to the states in the MDP, e.g., the scores are higher for the four entry points in the rooms. In general, one could learn such D automatically using the techniques from [21–24]—see full details about D in the Appendix. Then, for a fixed budget B, we pick the top B states according to the scoring by D and assign a reward of $+1$ for optimal actions and -1 for others. For the evaluation, we use $B = 5$ and denote the function as $\widehat{R}_{\text{Craft}(B=5)}$. Note that apart from B states, $\widehat{R}_{\text{Craft}(B=5)}$ also has a reward assigned for the goal state taken from $\overline{R}$.

[4]The reward shaping method in [42] is based on the PBRS technique and leads to dense reward functions. However, their method is more practical as it does not require solving the original task w.r.t. $\overline{R}$.



(a) Convergence (b) $\widehat{R}_{\text{ORIG}}$ (c) $\widehat{R}_{\text{PBRS}}$ (d) $\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$

Figure 3: Results for ROOMSNAVENV. **(a)** shows convergence in performance of the agent w.r.t. training episodes. Here, performance is measured as the expected reward per episode computed using $\overline{R}$; note that the x-axis is exponential in scale. **(b-d)** visualize the designed reward functions $\widehat{R}_{\text{ORIG}}$, $\widehat{R}_{\text{PBRS}}$, and $\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$. These plots illustrate reward values for all combinations of $\mathcal{S} \times \mathcal{A}$ shown as four 7×7 grids corresponding to different actions. Blue color represents positive reward, red color represents negative reward, and the magnitude of the reward is indicated by color intensity. As an example, consider "right" action grid for $\widehat{R}_{\text{ORIG}}$ in **(b)** where the dark blue color in the corner indicates the goal. To increase the color contrast, we clipped rewards in the range $[-4, +4]$ for this visualization even though the designed rewards are in the range $[-10, +10]$. See Section 4.1 for details.

The reward functions $\widehat{R}_{\text{EXPRD}}$ designed by our EXPRD framework are parameterized by budget B and hyperparameter λ. For λ, we consider two extreme settings: (a) $\lambda = 0$ where the problem (P3) reduces to (P2), and (b) $\lambda \to \infty$ where the problem (P3) reduces to (P1) corresponding to the reward design with subgoals pre-selected by the function D. We use the same function D that we used for $\widehat{R}_{\text{CRAFT}}$ above. For budget B, we consider values from $\{3, 5, |S|\}$. In particular, we evaluate the following reward functions: $\widehat{R}_{\text{EXPRD}(B=5,\lambda\to\infty)}$, $\widehat{R}_{\text{EXPRD}(B=3,\lambda=0)}$, $\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$, and $\widehat{R}_{\text{EXPRD}(B=|S|,\lambda=0)}$. For the evaluation in this section, we use the following parameter choices for EXPRD: $\mathcal{H} = \{1, 4, 8, 16, 32\}$, ℓ is the negated hinge loss ℓ_{hg}, and $\Pi^\dagger$ contains only one policy from $\overline{\Pi}^*$.

Results. We use standard Q-learning method for the agent with a learning rate 0.5 and exploration factor 0.1 [7]. During training, the agent receives rewards based on $\widehat{R}$, however, is evaluated based on $\overline{R}$. A training episode ends when the maximum steps (set to 50) is reached or an agent's action terminates the episode. All the results are reported as average over 40 runs and convergence plots show mean with standard error bars. The convergence behavior in Figure 3a demonstrates the effectiveness of the reward functions designed by our EXPRD framework.[5] Note that $\widehat{R}_{\text{CRAFT}(B=5)}$ leads to sub-optimal behavior due to "reward bugs" (see Section 2), whereas $\widehat{R}_{\text{EXPRD}(B=5,\lambda\to\infty)}$ fixes this issue using the same set of subgoals. EXPRD leads to good performance even without domain knowledge (i.e., when $\lambda = 0$), e.g., the performance corresponding to $\widehat{R}_{\text{EXPRD}(B=3,\lambda=0)}$ is comparable to that of $\widehat{R}_{\text{EXPRD}(B=5,\lambda\to\infty)}$. The visualizations of $\widehat{R}_{\text{ORIG}}$, $\widehat{R}_{\text{PBRS}}$, and $\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$ in Figures 3b, 3c, and 3d highlight the trade-offs in terms of sparseness and interpretability of the reward functions. The reward function $\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$ designed by our EXPRD framework provides a good balance in terms of convergence performance while maintaining high sparseness. Additional visualizations and results are provided in the Appendix.

4.2 Evaluation on LINEKEYNAVENV

LINEKEYNAVENV (Figure 4). We represent the environment as an MDP with $\mathcal{S}$ states corresponding to the agent's status comprising of the current location (shown as "blue-circle" and is a point x in $[0, 1]$) and a binary flag whether the agent has acquired a key (shown as "cyan-bolt"). Goal (shown as "green-star") is available in locations on the segment $[0.9, 1]$, and the key is available in locations on the segment $[0.1, 0.2]$. The agent can take three actions given by $\mathcal{A} :=$ {"left", "right", , "pick"}. "pick" action does not change the agent's location, however, when

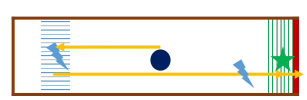
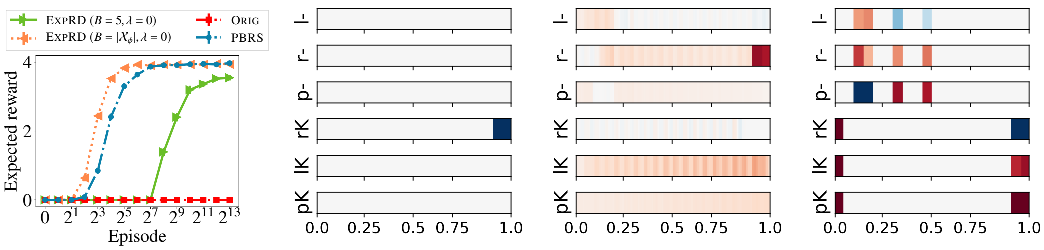


Figure 4: LINEKEYNAVENV

[5] As we discussed in Sections 1 and 2, $\widehat{R}_{\text{PBRS}}$ designed using $\overline{V}^*_\infty$ makes the agent's learning process trivial.



(a) Convergence (b) $\widehat{R}_{\text{ORIG}}$ (c) $\widehat{R}_{\text{PBRS}}$ (d) $\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$

Figure 5: Results for LINEKEYNAVENV. **(a)** shows convergence in performance of the agent w.r.t. training episodes. Here, performance is measured as the expected reward per episode computed using $\overline{R}$. **(b-d)** visualize the designed reward functions $\widehat{R}_{\text{ORIG}}$, $\widehat{R}_{\text{PBRS}}$, and $\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$. These plots illustrate reward values for all combination of triplets, i.e., agent's location on the segment $[0.0, 1.0]$ (shown as horizontal bar), agent's status whether it has acquired key or not (indicated as 'K' or '-'), and three actions (indicated as 'l' for "left", 'r' for "right", 'p' for "pick"). We use a color representation similar to Figure 3, and we clipped rewards in the range $[-3, +3]$ to increase the color contrast for this visualization. As an example, consider 'rK' bar for $\widehat{R}_{\text{ORIG}}$ in **(b)** where the dark blue color on the segment $[0.9, 1]$ indicate the locations with goal. See Section 4.2 for details.

executed in locations with availability of the key, the agent acquires the key; if agent already had a key, the action does not affect the status. A move action of "left" or "right" takes the agent from the current location in the direction of move with the dynamics of the final location captured by two hyperparameters $(\Delta_{a,1}, \Delta_{a,2})$; for instance, with current location x and action "left", the new location x$'$ is sampled uniformly among locations from $(\text{x} - \Delta_{a,1} - \Delta_{a,2})$ to $(\text{x} - \Delta_{a,1} + \Delta_{a,2})$. Similar to ROOMSNAVENV, the agent's move action is not applied if the new location crosses the wall, and there is p_{rand} probability of a random action. The agent gets a reward of R_{max} after it has navigated to the goal locations after acquiring the key and then takes a "right" action; note that this action also terminates the episode. The reward is 0 elsewhere and there is a discount factor γ. We set $p_{\text{rand}} = 0.1$, $R_{\text{max}} = 10$, $\gamma = 0.95$, $\Delta_{a,1} = 0.075$ and $\Delta_{a,2} = 0.01$.

Techniques evaluated. The baseline $\widehat{R}_{\text{ORIG}} := \overline{R}$ represents the default reward function. We evaluate the variants of $\widehat{R}_{\text{PBRS}}$ and $\widehat{R}_{\text{EXPRD}}$ using an abstraction. For a given hyperparameter $\alpha \in (0, 1)$, the set of possible locations X are obtained by α-level discretization of the line segment from 0.0 to 1.0, leading to a $^1/_\alpha$ set of locations. For the abstraction ϕ associated with this discretization [43], the abstract MDP $\overline{M}_\phi$ (see Section 3.5) has $|\mathcal{X}_\phi| = {}^2/_\alpha$ and $|\mathcal{A}| = 3$. We use $\alpha = 0.05$. We compute the optimal state value function in the abstract MDP $\overline{M}_\phi$, lift it to the original state space via ϕ, and use the lifted value function as the potential to design $\widehat{R}_{\text{PBRS}}$ [35]. We follow the pipeline in Section 3.5 to design $\widehat{R}_{\text{EXPRD}}$ – in the subroutine, we run EXPRD on $\overline{M}_\phi$ for a budget $B = 5$ and a full budget $B = |\mathcal{X}_\phi|$; we set $\lambda = 0$. For other parameters ($\mathcal{H}$, ℓ, and $\Pi^\dagger$), we use the same choices as in Section 4.1.

Results. The agent uses Q-learning method in the original MDP $\overline{M}$ by using a fine-grained discretization of the state space; rest of the method's parameters are same as in Section 4.1. All the results are reported as average over 40 runs and convergence plots show mean with standard error bars. Figure 5a demonstrates that all three designed reward functions—$\widehat{R}_{\text{PBRS}}$, $\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$, $\widehat{R}_{\text{EXPRD}(B=|\mathcal{X}_\phi|,\lambda=0)}$—substantially improves the convergence, whereas the agent is not able to learn under $\widehat{R}_{\text{ORIG}}$. Based on the visualizations in Figures 5b, 5c, and 5d, $\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$ provides a good balance between convergence and sparseness. Interestingly, $\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$ assigned a high positive reward for the "pick" action when the agent is in the locations with key (see 'p-' bar in Figure 5d).

5 Related Work

Potential-based reward shaping. Introduced in the seminal work of [3], potential-based reward shaping is one of the most well-studied reward design technique (see [8, 14, 37, 38, 40, 44, 45, 45–48]). As we discussed in Section 2, the shaped reward function $\widehat{R}_{\text{PBRS}}$ is obtained by modifying $\overline{R}$ using a state-dependent potential function. The technique preserves a strong invariance property: a policy π is optimal under $\widehat{R}_{\text{PBRS}}$ *iff* it is optimal under $\overline{R}$. Furthermore, when using the optimal value-function $\overline{V}^*_\infty$ under $\overline{R}$ as the potential function, the shaped rewards achieve the maximum possible informativeness as per the notion we use in EXPRD. To balance informativeness and sparseness, our framework EXPRD can be seen as a relaxation of the potential-based shaping in the following ways: (i) EXPRD provides a guarantee on preserving a weaker invariance property whereby an optimal policy under $\widehat{R}_{\text{ExpRD}}$ is also optimal under $\overline{R}$; (ii) EXPRD finds $\widehat{R}_{\text{ExpRD}}$ that maximizes informativeness under hard constraints of preserving this weaker policy-invariant property and a given spareness-level.

Optimization-based techniques for reward design. Beyond potential-based shaping, we can formulate reward design as an optimization problem [15–19]. In particular, optimization-based techniques for reward design are popularly used in data poisoning attacks where an attacker's goal is to minimally perturb the original reward function to force the agent into executing a target policy chosen by the attacker [17–19]. Our EXPRD framework builds on the optimization framework of [17–19]. The key novelty of EXPRD is that we optimize for informativeness of the reward function under a sparseness constraint, which makes our problem formulation much more challenging.

Agent-driven reward design. An important categorization of reward design techniques is based on who is designing the rewards and what domain knowledge is available. Agent-driven reward design techniques involve a reinforcement learning method where an agent self-designs its own rewards during the training process, with the objective of improving the exploration and speeding up the convergence [6, 49–53]. These agent-driven techniques use a wide-variety of ideas such as designing intrinsic rewards based on exploration bonus [49, 50, 54], designing rewards using some additional domain knowledge [51], and using credit assignment to create intermediate rewards [6, 52].

Expert-driven reward design. In contrast to agent-driven techniques, we have expert-driven reward design techniques where an expert/teacher with full domain knowledge can design a reward function for the agent [1, 3, 14–19, 48]. Our EXPRD framework falls into the category of teacher-driven reward design. The above-mentioned techniques of potential-based reward shaping and optimization-based techniques can be seen as expert-driven reward design techniques; however, the distinction between expert-driven and agent-driven techniques can be blurry at times when one uses an expert-driven technique with minimal domain knowledge (e.g., when using approximate potentials [3]).

Reward automatas, landmark-based rewards, and subgoal discovery. Our EXPRD framework is also connected to techniques that specify rewards using higher-level abstract representations of the environment including symbolic automata and landmarks [13, 14, 37, 40, 55, 56]. In recent works [13, 14, 55, 56], potential-based reward shaping technique has been used with automata-based rewards to design interpretable and informative rewards. While similar in the overall objective, our work is technically quite different and our proposed optimization framework to reward design can be seen as complementary to these works. Another relevant line of work focuses on automatic discovery of subgoals in the environment [21–24] – these works are complementary and useful as subroutines in our framework by providing a prior knowledge about which states are important for assigning rewards.

6 Conclusions

We developed a novel optimization framework, EXPRD, to design explicable reward functions in which we can appropriately balance informativeness and sparseness in the reward design process. As part of the framework, we introduced a new criterion capturing informativeness of reward functions that is of independent interest. The mathematical analysis of EXPRD shows connections of our framework to the popular reward-design techniques, and provides theoretical underpinnings of expert-driven explicable reward design. Importantly, EXPRD allows one to go beyond using a potential function for principled reward design, and provides a general recipe for developing an optimization-based reward design framework with different structural constraints. We also provided a practical extension to apply our framework in environments with large state spaces via state abstractions.

To make our framework more scalable, we plan to investigate alternate formulations of the reward design problem that avoids enumerating all the constraints explicitly (see Section 3). There are several promising directions for future work, including but not limited to the following: (a) using a combination of our optimization-based reward design technique with automata-driven rewards as well as other structured rewards, (b) extending our framework for agent-driven reward design, (c) applying our framework in a transfer setting using techniques from [42, 57], and (d) investigating the usage of our informativeness criterion for discovering subgoals.

References

[1] Maja J. Mataric. Reward Functions for Accelerated Learning. In *ICML*, pages 181–189, 1994.

[2] Jette Randløv and Preben Alstrøm. Learning to Drive a Bicycle Using Reinforcement Learning and Shaping. In *ICML*, pages 463–471, 1998.

[3] Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. In *ICML*, pages 278–287, 1999.

[4] Adam Laud and Gerald DeJong. The Influence of Reward on the Speed of Reinforcement Learning: An Analysis of Shaping. In *ICML*, pages 440–447, 2003.

[5] Falcon Z. Dai and Matthew R. Walter. Maximum Expected Hitting Cost of a Markov Decision Process and Informativeness of Rewards. In *NeurIPS*, pages 7677–7685, 2019.

[6] Jose A. Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. RUDDER: Return Decomposition for Delayed Rewards. In *NeurIPS*, pages 13544–13555, 2019.

[7] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018.

[8] Haosheng Zou, Tongzheng Ren, Dong Yan, Hang Su, and Jun Zhu. Reward Shaping via Meta-Learning. *CoRR*, abs/1901.09330, 2019.

[9] Eleanor O'Rourke, Kyla Haimovitz, Christy Ballweber, Carol S. Dweck, and Zoran Popovic. Brain Points: A Growth Mindset Incentive Structure Boosts Persistence in an Educational Game. In *CHI*, pages 3339–3348, 2014.

[10] VirtaMed. Virtamed: Simulators for Medical Training and Education. https://www.virtamed.com/en/.

[11] Virtual Driver Interactive. https://www.driverinteractive.com/.

[12] John H. Maloney, Kylie Peppler, Yasmin Kafai, Mitchel Resnick, and Natalie Rusk. Programming by choice: Urban youth learning programming with Scratch. In *SIGCSE*, pages 367–371, 2008.

[13] Rodrigo Toro Icarte, Toryn Q. Klassen, Richard Anthony Valenzano, and Sheila A. McIlraith. Reward Machines: Exploiting Reward Function Structure in Reinforcement Learning. *CoRR*, abs/2010.03950, 2020.

[14] Yuqian Jiang, Suda Bharadwaj, Bo Wu, Rishi Shah, Ufuk Topcu, and Peter Stone. Temporal-Logic-Based Reward Shaping for Continuing Reinforcement Learning Tasks. In *AAAI*, 2021.

[15] Haoqi Zhang and David C. Parkes. Value-Based Policy Teaching with Active Indirect Elicitation. In *AAAI*, pages 208–214, 2008.

[16] Haoqi Zhang, David C. Parkes, and Yiling Chen. Policy Teaching through Reward Function Learning. In *EC*, pages 295–304, 2009.

[17] Yuzhe Ma, Xuezhou Zhang, Wen Sun, and Jerry Zhu. Policy Poisoning in Batch Reinforcement Learning and Control. In *NeurIPS*, pages 14543–14553, 2019.

[18] Amin Rakhsha, Goran Radanovic, Rati Devidze, Xiaojin Zhu, and Adish Singla. Policy Teaching via Environment Poisoning: Training-time Adversarial Attacks against Reinforcement Learning. In *ICML*, volume 119, pages 7974–7984, 2020.

[19] Amin Rakhsha, Goran Radanovic, Rati Devidze, Xiaojin Zhu, and Adish Singla. Policy Teaching in Reinforcement Learning via Environment Poisoning Attacks. *Journal of Machine Learning Research*, 22(210):1–45, 2021.

[20] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1st edition, 1994.

[21] Amy McGovern and Andrew G. Barto. Automatic Discovery of Subgoals in Reinforcement Learning using Diverse Density. In *ICML*, pages 361–368, 2001.

[22] Özgür Simsek, Alicia P. Wolfe, and Andrew G. Barto. Identifying Useful Subgoals in Reinforcement Learning by Local Graph Partitioning. In *ICML*, volume 119, pages 816–823, 2005.

[23] Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic Goal Generation for Reinforcement Learning Agents. In *ICML*, volume 80, pages 1514–1523, 2018.

[24] Sujoy Paul, Jeroen van Baar, and Amit K. Roy-Chowdhury. Learning from Trajectories via Subgoal Discovery. In *NeurIPS*, pages 8409–8419, 2019.

[25] Andreas Krause and Daniel Golovin. Submodular Function Maximization. *Tractability*, 3:71–104, 2014.

[26] Michael J. Kearns, Yishay Mansour, and Andrew Y. Ng. A Sparse Sampling Algorithm for Near-Optimal Planning in Large Markov Decision Processes. *Machine Learning*, 49(2-3):193–208, 2002.

[27] Hiroki Furuta, Tatsuya Matsushima, Tadashi Kozuno, Yutaka Matsuo, Sergey Levine, Ofir Nachum, and Shixiang Shane Gu. Policy Information Capacity: Information-Theoretic Measure for Task Complexity in Deep Reinforcement Learning. In *ICML*, pages 3541–3552, 2021.

[28] Adam Gleave, Michael Dennis, Shane Legg, Stuart Russell, and Jan Leike. Quantifying Differences in Reward Functions. In *ICLR*, 2021.

[29] Ethan R Elenberg, Rajiv Khanna, Alexandros G Dimakis, and Sahand Negahban. Restricted Strong Convexity Implies Weak Submodularity. *Annals of Statistics*, 46(6B):3539–3568, 2018.

[30] Abhimanyu Das and David Kempe. Submodular meets spectral: greedy algorithms for subset selection, sparse approximation and dictionary selection. In *ICML*, pages 1057–1064, 2011.

[31] Sahand N Negahban, Pradeep Ravikumar, Martin J Wainwright, Bin Yu, et al. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. *Statistical science*, 27(4):538–557, 2012.

[32] Robert Givan, Thomas Dean, and Matthew Greig. Equivalence notions and model minimization in markov decision processes. *Artificial Intelligence*, 147(1-2):163–223, 2003.

[33] Lihong Li, Thomas J Walsh, and Michael L Littman. Towards a unified theory of state abstraction for mdps. *ISAIM*, 4:5, 2006.

[34] David Abel, David Hershkowitz, and Michael Littman. Near Optimal Behavior via Approximate State Abstraction. In *ICML*, pages 2915–2923, 2016.

[35] Bhaskara Marthi. Automatic shaping and decomposition of reward functions. In *ICML*, pages 601–608, 2007.

[36] Parameswaran Kamalaruban, Rati Devidze, Volkan Cevher, and Adish Singla. Environment Shaping in Reinforcement Learning using State Abstraction. *CoRR*, abs/2006.13160, 2020.

[37] Marek Grzes and Daniel Kudenko. Plan-based Reward Shaping for Reinforcement Learning. In *International IEEE Conference on Intelligent Systems*, volume 2, pages 10–22, 2008.

[38] John Asmuth, Michael L. Littman, and Robert Zinkov. Potential-based Shaping in Model-based Reinforcement Learning. In *AAAI*, pages 604–609, 2008.

[39] Michael R. James and Satinder P. Singh. Sarsalandmark: An Algorithm for Learning in POMDPs with Landmarks. In *AAMAS*, pages 585–591, 2009.

[40] Alper Demir, Erkin Çilden, and Faruk Polat. Landmark Based Reward Shaping in Reinforcement Learning with Hidden States. In *AAMAS*, pages 1922–1924, 2019.

[41] Roberta Raileanu, Emily Denton, Arthur Szlam, and Rob Fergus. Modeling Others using Oneself in Multi-Agent Reinforcement Learning. In *ICML*, volume 80, pages 4254–4263, 2018.

[42] Anna Harutyunyan, Sam Devlin, Peter Vrancx, and Ann Nowé. Expressing Arbitrary Reward Functions as Potential-Based Advice. In *AAAI*, pages 2652–2658, 2015.

[43] John Burden and Daniel Kudenko. Uniform state abstraction for reinforcement learning. *arXiv preprint arXiv:2004.02919*, 2020.

[44] Eric Wiewiora. Potential-Based Shaping and Q-Value Initialization are Equivalent. *Journal of Artificial Intelligence Research*, 19:205–208, 2003.

[45] Eric Wiewiora, Garrison W. Cottrell, and Charles Elkan. Principled Methods for Advising Reinforcement Learning Agents. In *ICML*, pages 792–799, 2003.

[46] Sam Devlin and Daniel Kudenko. Dynamic Potential-based Reward Shaping. In *AAMAS*, pages 433–440, 2012.

[47] Marek Grzes. Reward Shaping in Episodic Reinforcement Learning. In *AAMAS*, pages 565–573, 2017.

[48] Prasoon Goyal, Scott Niekum, and Raymond J. Mooney. Using natural language for reward shaping in reinforcement learning. In *IJCAI*, pages 2385–2391, 2019.

[49] Andrew G. Barto. Intrinsic Motivation and Reinforcement Learning. In *Intrinsically Motivated Learning in Natural and Artificial Systems*, pages 17–47. 2013.

[50] Tejas D. Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation. In *NeurIPS*, pages 3675–3683, 2016.

[51] Alexander Trott, Stephan Zheng, Caiming Xiong, and Richard Socher. Keeping Your Distance: Solving Sparse Reward Tasks Using Self-Balancing Shaped Rewards. In *NeurIPS*, pages 10376–10386, 2019.

[52] Johan Ferret, Raphaël Marinier, Matthieu Geist, and Olivier Pietquin. Self-Attentional Credit Assignment for Transfer in Reinforcement Learning. In *IJCAI*, pages 2655–2661, 2020.

[53] Jonathan Sorg, Satinder P. Singh, and Richard L. Lewis. Reward Design via Online Gradient Ascent. In *NeurIPS*, pages 2190–2198, 2010.

[54] Xuezhou Zhang, Yuzhe Ma, and Adish Singla. Task-Agnostic Exploration in Reinforcement Learning. In *NeurIPS*, 2020.

[55] Alberto Camacho, Oscar Chen, Scott Sanner, and Sheila A McIlraith. Decision-Making with Non-Markovian Rewards: From LTL to Automata-based Reward Shaping. In *Proceedings of the Multi-disciplinary Conference on Reinforcement Learning and Decision Making (RLDM)*, pages 279–283, 2017.

[56] Kishor Jothimurugan, Rajeev Alur, and Osbert Bastani. A Composable Specification Language for Reinforcement Learning Tasks. In *NeurIPS*, pages 13021–13030, 2019.

[57] Tim Brys, Anna Harutyunyan, Matthew E Taylor, and Ann Nowé. Policy Transfer using Reward Shaping. In *AAMAS*, pages 181–188, 2015.

[58] Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun. Reinforcement Learning: Theory and Algorithms, 2019.

[59] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[60] Francis Bach et al. Learning with submodular functions: A convex optimization perspective. *Foundations and Trends® in Machine Learning*, 6(2-3):145–373, 2013.

[61] Satinder P Singh and Richard C Yee. An upper bound on the loss from approximate optimal-value functions. *Machine Learning*, 16(3):227–233, 1994.

[62] Ronald J. Williams. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning*, 8:229–256, 1992.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] The paper is organized exactly according to the contributions listed at the end of the introduction section.
   (b) Did you describe the limitations of your work? [Yes] We discussed the scalability related limitations of our ExpRD framework in Sections 3.3 and 6. We also outlined a future plan to address these limitations.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] As stated in Section 6, this work primarily presents the theoretical underpinnings of reward design in reinforcement learning. As such in the present form there are no direct negative societal impacts of our work. However, given the importance of reward design in RL, one needs to be cautious in practical applications of these techniques in future.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] We confirm that our paper conforms with the ethics review guidelines.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] For example, Assumption 1 and Assumption 2 (see Appendix) gather the structural assumptions that the informativeness criterion needs to satisfy.
   (b) Did you include complete proofs of all theoretical results? [Yes] Complete proofs of all theoretical results are included in the Appendix of the supplementary material.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code and instructions are included as a URL.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Training details are presented in the Appendix of the supplementary material and are also present in the code.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Error bars are included in all the result graphs, as can be seen in Figures 3a and 5a.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] The details are provided in the Appendix of the supplementary material.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A List of Appendices

In this section, we give a brief description of the content provided in the appendices of the paper.

- Appendix B provides proofs for Propositions 1 and 2. (Sections 3.2 and 3.3)
- Appendix C provides additional details and proofs for the theoretical analysis. (Section 3.4)
- Appendix D provides additional details and proofs for using state abstractions. (Section 3.5)
- Appendix E provides additional results for ROOMSNAVENV. (Section 4.1)
- Appendix F provides additional results for LINEKEYNAVENV. (Section 4.2)

B Proofs for Propositions 1 and 2 (Sections 3.2 and 3.3)

B.1 Proof of Proposition 1

Proposition 1. *The goal-based reward function $\overline{R}$, and the potential-based shaped reward function $\widehat{R}_{\text{PBRS}}$ given in (1) satisfy the following: $\widehat{\delta}_h^{\pi}(s,a) = \overline{\delta}_{\infty}^{*}(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}, \pi \in \overline{\Pi}^*, h \in \mathcal{H}$.*

Proof. Consider any optimal policy $\pi \in \overline{\Pi}^*$, $s \in \mathcal{S}$, $a \in \mathcal{A}$, and $h \in \mathcal{H}$. The ∞-step optimality gap induced by $\overline{R}$ is $\overline{\delta}_{\infty}^{*}(s,a) = \overline{V}_{\infty}^{*}(s) - \overline{Q}_{\infty}^{*}(s,a)$, and the h-step optimality gap induced by $\widehat{R}_{\text{PBRS}}$ is $\widehat{\delta}_h^{\pi}(s,a) = \widehat{Q}_h^{\pi}(s,\pi(s)) - \widehat{Q}_h^{\pi}(s,a)$. In the following, we express the two terms of $\widehat{\delta}_h^{\pi}$ in terms of $\overline{V}_{\infty}^{*}$ and $\overline{Q}_{\infty}^{*}$.

The term $\widehat{Q}_h^{\pi}(s,\pi(s))$ for any $\pi \in \overline{\Pi}^*$. We show that $\widehat{Q}_h^{\pi}(s,\pi(s)) = 0$ for any non-negative integer h by using mathematical induction. First ($h = 0$ case), we consider the 0-step optimal action value function:

$$\begin{aligned}
\widehat{Q}_0^{\pi}(s,\pi(s)) &= \widehat{R}_{\text{PBRS}}(s,\pi(s)) \\
&= \overline{R}(s,\pi(s)) + \gamma \sum_{s' \in \mathcal{S}} T(s' \mid s,\pi(s))\, \overline{V}_{\infty}^{*}(s') - \overline{V}_{\infty}^{*}(s) \\
&= \overline{Q}_{\infty}^{*}(s,\pi(s)) - \overline{V}_{\infty}^{*}(s) \\
&= \overline{V}_{\infty}^{*}(s) - \overline{V}_{\infty}^{*}(s) \\
&= 0.
\end{aligned}$$

Now assume that $\widehat{Q}_{h-1}^{\pi}(s,\pi(s)) = 0$. Then, consider the h-step optimal action value function:

$$\begin{aligned}
\widehat{Q}_h^{\pi}(s,\pi(s)) &= \widehat{R}_{\text{PBRS}}(s,\pi(s)) + \gamma \sum_{s' \in \mathcal{S}} T(s' \mid s,\pi(s))\, \widehat{Q}_{h-1}^{\pi}(s',\pi(s)) \\
&= \widehat{R}_{\text{PBRS}}(s,\pi(s)) + 0 \\
&= 0.
\end{aligned}$$

Thus, by mathematical induction, we have that $\widehat{Q}_h^{\pi}(s,\pi(s)) = 0$ for any non-negative integer h.

The term $\widehat{Q}_h^{\pi}(s,a)$ for any $a \in \mathcal{A}$. Consider the h-step optimal action value function:

$$\begin{aligned}
\widehat{Q}_h^{\pi}(s,a) &= \widehat{R}_{\text{PBRS}}(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s' \mid s,a)\, \widehat{Q}_{h-1}^{\pi}(s',\pi(s')) \\
&= \widehat{R}_{\text{PBRS}}(s,a) + 0 \\
&= \overline{R}(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s' \mid s,a)\, \overline{V}_{\infty}^{*}(s') - \overline{V}_{\infty}^{*}(s) \\
&= \overline{Q}_{\infty}^{*}(s,a) - \overline{V}_{\infty}^{*}(s).
\end{aligned}$$

Finally, by combining these two terms, we get:

$$\widehat{\delta}_h^{\pi}(s,a) = \widehat{Q}_h^{\pi}(s,\pi(s)) - \widehat{Q}_h^{\pi}(s,a) = \overline{V}_{\infty}^{*}(s) - \overline{Q}_{\infty}^{*}(s,a) = \overline{\delta}_{\infty}^{*}(s,a).$$

□

B.2 Proof of Proposition 2

Proposition 2. *For any given $\mathcal{Z} \subseteq \mathcal{S}\backslash\mathcal{G}$, the problem (P1) is a concave optimization problem in $R \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$ with linear constraints. Further, the feasible set of the problem (P1) is non-empty.*

Proof. We write the problem (P1) explicitly as follows:

$$\max_{R} \sum_{\pi^\dagger \in \Pi^\dagger} \sum_{h \in \mathcal{H}} \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A} \backslash \overline{\Pi}^*_s} \ell(\delta_h^{\pi^\dagger}(s,a)) \tag{3}$$

subject to $R\left(s,a\right) = 0,\ \forall s \in \mathcal{S}\backslash \left\{\mathcal{Z} \cup \mathcal{G}\right\}, a \in \mathcal{A}$ (4)

$$Q_\infty^{\pi^\dagger}(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s'|s,a) Q_\infty^{\pi^\dagger}(s', \pi^\dagger(s')),\ \forall s \in \mathcal{S}, a \in \mathcal{A}, \pi^\dagger \in \overline{\Pi}^\dagger \tag{5}$$

$$Q_\infty^{\pi^\dagger}(s,\pi^\dagger(s)) \geq Q_\infty^{\pi^\dagger}(s,a) + \overline{\delta}_\infty^*(s),\ \forall s \in \mathcal{S}, a \in \mathcal{A}\backslash\overline{\Pi}_s^*, \pi^\dagger \in \overline{\Pi}^\dagger \tag{6}$$

$$Q_0^{\pi^\dagger}(s,a) = R(s,a),\ \forall s \in \mathcal{S}, a \in \mathcal{A}, \pi^\dagger \in \overline{\Pi}^\dagger \tag{7}$$

$$Q_h^{\pi^\dagger}(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s'|s,a) Q_{h-1}^{\pi^\dagger}(s', \pi(s')),\ \forall s \in \mathcal{S}, a \in \mathcal{A}, h \in \mathcal{H}, \pi^\dagger \in \overline{\Pi}^\dagger \tag{8}$$

$$\delta_h^{\pi^\dagger}(s,a) = Q_h^{\pi}(s,\pi^\dagger(s)) - Q_h^{\pi^\dagger}(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}, h \in \mathcal{H}, \pi^\dagger \in \overline{\Pi}^\dagger \tag{9}$$

$$\left|R\left(s,a\right)\right| \leq R_{\max},\ \forall s \in \mathcal{S}, a \in \mathcal{A} \tag{10}$$

In the following, we show that the above problem is a concave optimization problem (the objective is concave and the constraints are linear) by writing it in the matrix form as follows:

$$\begin{aligned} &\max_{R \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}} \sum_{\pi^\dagger \in \Pi^\dagger} \sum_{h \in \mathcal{H}} \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A} \backslash \overline{\Pi}^*_s} \ell\left(\left\langle w_{h;(s,a)}^{\pi^\dagger}, R\right\rangle\right) \\ &\text{subject to } A \cdot R \succeq b, \end{aligned}$$

for some vectors $w_{h;(s,a)}^{\pi^\dagger}, b \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$, and some matrix $A \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|\times|\mathcal{S}|\cdot|\mathcal{A}|}$.

Notation. We mainly follow the notation from [58]. Given a deterministic policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$, we define the transition matrix $T_\pi \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|\times|\mathcal{S}|\cdot|\mathcal{A}|}$ induced by π as follows:

$$[T_\pi]_{(s,a),(s',a')} := \begin{cases} T(s'|s,a), & \text{if } a' = \pi(s') \\ 0, & \text{otherwise.} \end{cases}$$

Also, for any $s \in \mathcal{S}$, we define $\mathrm{Id}_\pi\left(s\right) \in \mathbb{R}^{|\mathcal{A}|\times|\mathcal{A}|}$ as follows:

$$\left[\mathrm{Id}_\pi\left(s\right)\right]_{:,a} := \begin{cases} 1, & \text{if } a = \pi(s) \\ 0, & \text{otherwise.} \end{cases}$$

Then, we define $\mathrm{Id}_\pi \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|\times|\mathcal{S}|\cdot|\mathcal{A}|}$ as a block diagonal matrix with block size of $|\mathcal{A}| \times |\mathcal{A}|$, and $\mathrm{Id}_\pi\left(s\right)$ as the s^{th} diagonal block, $\forall s \in \mathcal{S}$. We define the diagonal matrix $L_{\overline{\Pi}^*} \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|\times|\mathcal{S}|\cdot|\mathcal{A}|}$, whose $(s,a)^{\text{th}}$ diagonal entry is given by:

$$[L_{\overline{\Pi}^*}]_{(s,a),(s,a)} := \begin{cases} 0, & \text{if } a \in \overline{\Pi}^*_s \\ 1, & \text{otherwise.} \end{cases}$$

We define the diagonal matrix $L_{\mathcal{Z}} \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|\times|\mathcal{S}|\cdot|\mathcal{A}|}$, whose $(s,a)^{\text{th}}$ diagonal entry is given by:

$$[L_{\mathcal{Z}}]_{(s,a),(s,a)} := \begin{cases} 0, & \text{if } s \in \mathcal{Z} \\ 1, & \text{otherwise.} \end{cases}$$

Let $e_i \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$ be a vector having 1 only in the i^{th} entry, and 0 elsewhere. Let $\overline{\delta}_\infty^* \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$ be a vector such that its $(s,a)^{\text{th}}$ entry is given by $\left[\overline{\delta}_\infty^*\right]_{(s,a)} = \overline{\delta}_\infty^*\left(s\right), \forall a \in \mathcal{A}$. Let $\mathbf{1} \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|}$ be a vector of all ones. Let $\mathrm{Id} \in \mathbb{R}^{|\mathcal{S}|\cdot|\mathcal{A}|\times|\mathcal{S}|\cdot|\mathcal{A}|}$ be the identity matrix.

Bound constraint. The bound constraint in Eq. (10) can be written as follows:

$$R_{\max} \cdot \mathbf{1} \ \succeq \ R \ \succeq \ -R_{\max} \cdot \mathbf{1}.$$

The above is linear inequality in R.

Sparsity constraint. The sparsity constraint in Eq. (4) can be written as follows:

$$L_{\mathcal{Z}} R \ = \ 0.$$

The above is linear equality in R.

Global optimality constraints. The recursive form of the action value function $Q_\infty^\pi(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s' \mid s,a) Q_\infty^\pi(s', \pi(s'))$ can be written in the matrix form as follows:

$$Q_\infty^\pi \ = \ R + \gamma T_\pi Q_\infty^\pi \quad \Longrightarrow \quad Q_\infty^\pi \ = \ \left(\mathrm{Id} - \gamma T_\pi\right)^{-1} R.$$

Then, the global optimality constraints in Eq. (6) can be written as follows, for all $\pi^\dagger \in \overline{\Pi}^\dagger$:

$$\left(\mathrm{Id}_{\pi^\dagger} - \mathrm{Id}\right) Q_\infty^{\pi^\dagger} \ \succeq \ L_{\overline{\Pi}^*} \overline{\delta}_\infty^* \quad \Longrightarrow \quad \left(\mathrm{Id}_{\pi^\dagger} - \mathrm{Id}\right) \left(\mathrm{Id} - \gamma T_{\pi^\dagger}\right)^{-1} R \ \succeq \ L_{\overline{\Pi}^*} \overline{\delta}_\infty^*.$$

The above is linear inequality in R.

Information $I_\ell(R)$ is concave in R. For $h = 0$, $Q_0^\pi(s,a) = R\left(s,a\right)$ can be written as follows:

$$Q_0^\pi \ = \ R.$$

For $h = 1$, $Q_1^\pi(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s' \mid s,a) Q_0^\pi(s', \pi(s'))$ can be written as follows:

$$Q_1^\pi \ = \ R + \gamma T_\pi Q_0^\pi \ = \ \left(\mathrm{Id} + \gamma T_\pi\right) R.$$

For $h = 2$, $Q_2^\pi(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s' \mid s,a) Q_1^\pi(s', \pi(s'))$ can be written as follows:

$$Q_2^\pi \ = \ R + \gamma T_\pi Q_1^\pi \ = \ \left(\mathrm{Id} + \gamma T_\pi + \gamma^2 T_\pi T_\pi\right) R.$$

For any h, $Q_h^\pi(s,a) = R(s,a) + \gamma \sum_{s' \in \mathcal{S}} T(s' \mid s,a) Q_{h-1}^\pi(s', \pi(s'))$ can be written as follows:

$$Q_h^\pi \ = \ \left(\mathrm{Id} + \gamma T_\pi + \gamma^2 T_\pi^{(2)} + \cdots + \gamma^h T_\pi^{(h)}\right) R,$$

where $T_\pi^{(h)} = \underbrace{T_\pi T_\pi \cdots T_\pi}_{h-\text{times}}$. Then, we can write $\delta_h^\pi(s,a) = Q_h^\pi(s, \pi(s)) - Q_h^\pi(s,a)$ as follows:

$$\delta_h^\pi(s,a) \ = \ \left\langle \left(\mathrm{Id}_\pi - \mathrm{Id}\right) \left(\mathrm{Id} + \gamma T_\pi + \gamma^2 T_\pi^{(2)} + \cdots + \gamma^h T_\pi^{(h)}\right) R, e_{(s,a)} \right\rangle,$$

i.e., $\delta_h^\pi(s,a)$ is linear in R for every $s \in \mathcal{S}$, and $a \in \mathcal{A}$. From the above equation, one can easily show that $\delta_h^\pi(s,a) = \left\langle w_{h;(s,a)}^{\pi^\dagger}, R \right\rangle$, where $w_{h;(s,a)}^{\pi^\dagger} := \rho_{h;(s,\pi^\dagger(s))}^{\pi^\dagger} - \rho_{h;(s,a)}^{\pi^\dagger}$. Since $\ell : \mathbb{R} \to \mathbb{R}$ is monotonically non-decreasing concave function, we have that $\ell \circ \delta_h^\pi(s,a)$ is concave [59]. From the fact that the sum of concave functions is concave, $I_\ell(R)$ is concave in R.

In summary, for the problem (P1), the objective is concave and the constraints are of linear form ($A \cdot R \succeq b$). Thus, (P1) is a concave optimization problem.

Feasibility. One can easily verify that the original reward function $\overline{R}$ satisfies all the constraints in (4)-(10) of the sparse reward shaping formulation for any $\mathcal{Z}$, i.e., $\overline{R}$ is a feasible solution. Furthermore, when $\mathcal{Z} = \mathcal{S} \backslash \mathcal{G}$, the potential-based shaped reward function $\widehat{R}_{\text{PBRS}}$ given in (1) satisfies all the constraints in (4)-(10) of the sparse reward shaping formulation. $\square$

C Additional Details and Proofs for Theoretical Analysis (Section 3.4)

First, we define the submodularity and weak submodularity notions of a normalized set function, which are used in the proof of Theorem 1.

Definition 2 (Submodularity [60]). *Let $g : 2^{\mathcal{V}} \rightarrow \mathbb{R}$ be a normalized set function ($g(\emptyset) = 0$). g is submodular if for all $\mathcal{W} \subseteq \mathcal{V}$ and $j, k \in \mathcal{V} \backslash \mathcal{W}$:*

$$g(\mathcal{W} \cup \{k\}) - g(\mathcal{W}) \;\geq\; g(\mathcal{W} \cup \{j,k\}) - g(\mathcal{W} \cup \{j\}).$$

Definition 3 (Weak Submodularity [30]). *Let $\mathcal{Y}, \mathcal{X} \subset \mathcal{V}$ be two disjoint sets, and $g : 2^{\mathcal{V}} \rightarrow \mathbb{R}$ be a normalized set function. The submodularity ratio of $\mathcal{X}$ with respect to $\mathcal{Y}$ is given by*

$$\gamma_{\mathcal{X},\mathcal{Y}} \;:=\; \frac{\sum_{j \in \mathcal{Y}} \left(g\left(\mathcal{X} \cup \{j\}\right) - g\left(\mathcal{X}\right)\right)}{g\left(\mathcal{X} \cup \mathcal{Y}\right) - g\left(\mathcal{X}\right)}. \tag{11}$$

The submodularity ratio of a set $\mathcal{W}$ with respect to an integer k is given by

$$\gamma_{\mathcal{W},k} \;:=\; \min_{\mathcal{X},\mathcal{Y}: \mathcal{X} \cap \mathcal{Y} = \emptyset, \mathcal{X} \subseteq \mathcal{W}, |\mathcal{Y}| \leq k} \gamma_{\mathcal{X},\mathcal{Y}}.$$

Let $\gamma > 0$. We call a function γ-weakly submodular at a set $\mathcal{W}$ and an integer k if $\gamma_{\mathcal{W},k} \geq \gamma$.

A set function $g : 2^{\mathcal{V}} \rightarrow \mathbb{R}$ is called monotone if and only if $g(\mathcal{X}) \leq g(\mathcal{Y})$ for all $\mathcal{X} \subseteq \mathcal{Y}$.

For any $x \in \mathbb{R}^{|\mathcal{S}| \cdot |\mathcal{A}|}$ and $\mathcal{U} \subseteq \mathcal{S}$, $x_{\mathcal{U}}$ is defined as $x_{\mathcal{U}}\left(j,a\right) = x\left(j,a\right), \forall a \in \mathcal{A}$ when $j \in \mathcal{U}$, and $x_{\mathcal{U}}\left(j,a\right) = 0, \forall a \in \mathcal{A}$ otherwise. For any $j \in \mathcal{S}$, $e_j \in \mathbb{R}^{|\mathcal{S}| \cdot |\mathcal{A}|}$ is defined as $e_j\left(j',a\right) = 1, \forall a \in \mathcal{A}$ when $j' = j$, and $e_j\left(j',a\right) = 0, \forall a \in \mathcal{A}$ otherwise. The following assumption captures the additional requirements on the regularized informativeness criterion I_ℓ^{reg}:

Assumption 2. *Let $\mathcal{Z}$ be any set such that $\mathcal{Z} \subseteq \mathcal{S} \backslash \mathcal{G}$. The regularized informativeness criterion I_ℓ^{reg} satisfies the following:*

- $\left\| \nabla I_\ell^{\text{reg}}(R^{(\mathcal{Z})})_{(\mathcal{Z} \cup \mathcal{G})} \right\|_2 \leq d_{\max}^{\text{opt}}$,
- $\left\| \nabla I_\ell^{\text{reg}}(R^{(\mathcal{Z})})_j \right\|_2 \geq d_{\min}^{\text{non}}, \forall j \in \mathcal{S} \backslash (\mathcal{Z} \cup \mathcal{G})$,
- $\left\| \nabla I_\ell^{\text{reg}}(R^{(\mathcal{Z})})_j \right\|_\infty \leq d_{\max}^{\text{non}}, \forall j \in \mathcal{S} \backslash (\mathcal{Z} \cup \mathcal{G})$, *and*
- $\exists \kappa \leq 1$ *such that* $\forall j \in \mathcal{S} \backslash (\mathcal{Z} \cup \mathcal{G}) : R^{(\mathcal{Z})} \pm \kappa \cdot \frac{d_{\max}^{\text{non}}}{\tilde{M}_{|\mathcal{Z}|+|\mathcal{G}|+1}} \cdot e_j \in \mathcal{R}$.

C.1 Proof of Theorem 1

Let $\mathcal{Z} \subseteq \mathcal{S} \backslash \mathcal{G}$. Consider the set function $f : 2^{\mathcal{S}} \rightarrow \mathbb{R}_+$ defined in (2):

$$f(\mathcal{Z}) \;=\; \max_{R: \text{supp}(R) \subseteq \mathcal{Z} \cup \mathcal{G}, R \in \mathcal{R}} \left(I_\ell^{\text{reg}}(R) - I_\ell^{\text{reg}}(R^{(\emptyset)})\right) + \lambda \cdot \left(D(\mathcal{Z} \cup \mathcal{G}) - D(\mathcal{G})\right),$$

where $R^{(\emptyset)} = \arg\max_{R: \text{supp}(R) \subseteq \mathcal{G}, R \in \mathcal{R}} I_\ell^{\text{reg}}(R)$. Note that f is a normalized, monotone set function. For a given sparsity budget B, let $\mathcal{Z}_B^{\text{Greedy}}$ be the set selected by our Algorithm 1, and $\mathcal{Z}_B^{\text{OPT}}$ be the optimal set that maximizes the regularized variant of problem (P3). The corresponding f values of these sets are denoted by f_B^{Greedy} and f_B^{OPT} respectively.

Theorem 1. *Let I_ℓ^{reg} satisfies Assumption 1 and Assumption 2 requirements. Then, we have $f_B^{\text{Greedy}} \geq \left(1 - e^{-\gamma}\right) f_B^{\text{OPT}}$, where $\gamma = \frac{\kappa \cdot m_{2B+|\mathcal{G}|}}{M_{2B+|\mathcal{G}|}} \cdot \frac{\left(d_{\min}^{\text{non}}\right)^2}{\left(d_{\max}^{\text{opt}}\right)^2 + \left(d_{\min}^{\text{non}}\right)^2}$.*

Proof. If f is γ-weakly submodular at the set $\mathcal{Z}_B$ and the integer B (i.e., $\gamma_{\mathcal{Z}_B,B} \geq \gamma$), then, using Theorem 3 from [29] (which holds for any normalized, monotone, γ-weakly submodular function), we can complete the proof of Theorem 1:

$$f^{\text{Greedy}} \;\geq\; \left(1 - e^{-\gamma_{\mathcal{Z}_B^{\text{Greedy}},B}}\right) f^{\text{OPT}} \;\geq\; \left(1 - e^{-\gamma}\right) f^{\text{OPT}}.$$

Thus, it remains to prove the weak submodularity of f. Let f_0 denote f with $\lambda = 0$, and define $\bar{D}(\mathcal{Z}) := D(\mathcal{Z} \cup \mathcal{G}) - D(\mathcal{G})$. Note that $\bar{D}$ is a normalized, monotone, submodular function. Then, the submodularity ratio of f with general λ is bounded as follows:

$$\begin{aligned}\gamma_{\mathcal{X},\mathcal{Y}} \;&=\; \frac{\sum_{j\in\mathcal{Y}}\left(f_0(\mathcal{X}\cup\{j\}) - f_0(\mathcal{X})\right) + \lambda\sum_{j\in\mathcal{Y}}\left(\bar{D}(\mathcal{X}\cup\{j\}) - \bar{D}(\mathcal{X})\right)}{f_0(\mathcal{X}\cup\mathcal{Y}) - f_0(\mathcal{X}) + \lambda\left(\bar{D}(\mathcal{X}\cup\mathcal{Y}) - \bar{D}(\mathcal{X})\right)} \\ &\geq\; \min\left(\frac{\sum_{j\in\mathcal{Y}}\left(f_0(\mathcal{X}\cup\{j\}) - f_0(\mathcal{X})\right)}{f_0(\mathcal{X}\cup\mathcal{Y}) - f_0(\mathcal{X})}, 1\right),\end{aligned}$$

where the inequality is due to the fact that the submodularity ratio of $\bar{D}$ is ≥ 1 [29]. If the submodularity ratio of f_0 is ≥ 1, then $\gamma_{\mathcal{X},\mathcal{Y}} \geq 1$. This would lead to the following bound:

$$f^{\text{Greedy}} \;\geq\; \left(1 - \frac{1}{e}\right) f^{\text{OPT}}.$$

If the submodularity ratio of f_0 is ≤ 1 (this would be the case in general; thus, we consider this case in the theorem), then the submodularity ratio $\gamma_{\mathcal{X},\mathcal{Y}}$ of f with general λ is lower bounded by the submodularity ratio of f_0. By applying Lemma 1 with $\left(\mathcal{Z}_B^{\text{Greedy}}, B\right)$, we have that (since $\left|\mathcal{Z}_B^{\text{Greedy}}\right| = B$):

$$\gamma_{\mathcal{Z}_B^{\text{Greedy}},B} \;\geq\; \frac{\kappa \cdot m_{2B+|\mathcal{G}|}}{M_{2B+|\mathcal{G}|}} \cdot \frac{\left(d_{\min}^{\text{non}}\right)^2}{\left(d_{\max}^{\text{opt}}\right)^2 + \left(d_{\min}^{\text{non}}\right)^2} \;=:\; \gamma.$$

This completes the proof. $\square$

The following lemma provides a lower bound on the submodularity ratio $\gamma_{\mathcal{Z},k}$ of f_0 (for any $\mathcal{Z}$ s.t. $|\mathcal{Z}| \leq B$, and $k \leq B$):

Lemma 1. *Let the regularized informativeness criterion I_ℓ^{reg} satisfies the Assumption 1 and 2. Then, for any set $\mathcal{Z}$ s.t. $\mathcal{Z} \subseteq \mathcal{S}\backslash\mathcal{G}$, $|\mathcal{Z}| \leq B$, and $k \leq B$, the submodularity ratio $\gamma_{\mathcal{Z},k}$ of f_0 is lower bounded by*

$$\gamma_{\mathcal{Z},k} \;\geq\; \frac{\kappa \cdot m_{|\mathcal{Z}|+|\mathcal{G}|+k}}{M_{|\mathcal{Z}|+|\mathcal{G}|+k}} \cdot \frac{\left(d_{\min}^{\text{non}}\right)^2}{\left(d_{\max}^{\text{opt}}\right)^2 + \left(d_{\min}^{\text{non}}\right)^2}.$$

Proof. Since I_ℓ^{reg} is $m_{2B+|\mathcal{G}|}$-restricted strongly concave and $M_{2B+|\mathcal{G}|}$-restricted smooth on $\Omega_{2B+|\mathcal{G}|}$, we have that I_ℓ^{reg} is $m_{|\mathcal{Z}|+|\mathcal{G}|+k}$-restricted strongly concave and $M_{|\mathcal{Z}|+|\mathcal{G}|+k}$-restricted smooth on $\Omega_{|\mathcal{Z}|+|\mathcal{G}|+k}$ for any $\mathcal{Z}$ s.t. $|\mathcal{Z}| \leq B$, and $k \leq B$. In addition I_ℓ^{reg} is $\tilde{M}_{|\mathcal{Z}|+|\mathcal{G}|+1}$-restricted smooth on $\tilde{\Omega}_{|\mathcal{Z}|+|\mathcal{G}|+1}$ since $\Omega_{|\mathcal{Z}|+|\mathcal{G}|+k} \supseteq \tilde{\Omega}_{|\mathcal{Z}|+|\mathcal{G}|+k} \supseteq \tilde{\Omega}_{|\mathcal{Z}|+|\mathcal{G}|+1}$ (and $M_{|\mathcal{Z}|+|\mathcal{G}|+k} \geq \tilde{M}_{|\mathcal{Z}|+|\mathcal{G}|+k} \geq \tilde{M}_{|\mathcal{Z}|+|\mathcal{G}|+1}$).

Consider the two sets $\mathcal{X}, \mathcal{Y}$ such that $(\mathcal{X} \cup \mathcal{G}) \cap \mathcal{Y} = \emptyset$, $\mathcal{X} \subseteq \mathcal{Z}$, and $|\mathcal{Y}| \leq k$. We proceed by upper bounding the denominator and lower bounding the numerator of Eq. (11). Let $\overline{k} = |\mathcal{X}| + |\mathcal{G}| + k$. First, we apply Definition 1 with $x = R^{(\mathcal{X})}$ and $y = R^{(\mathcal{X}\cup\mathcal{Y})}$ (note that $(x, y) \in \Omega_{\overline{k}}$):

$$\frac{m_{\overline{k}}}{2}\left\|R^{(\mathcal{X}\cup\mathcal{Y})} - R^{(\mathcal{X})}\right\|_2^2 \;\leq\; I_\ell^{\text{reg}}(R^{(\mathcal{X})}) - I_\ell^{\text{reg}}(R^{(\mathcal{X}\cup\mathcal{Y})}) + \left\langle \nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})}), R^{(\mathcal{X}\cup\mathcal{Y})} - R^{(\mathcal{X})}\right\rangle.$$

Rearranging and noting that I_ℓ^{reg} is monotone for increasing supports:

$$\begin{aligned}0 \;\leq\; I_\ell^{\text{reg}}(R^{(\mathcal{X}\cup\mathcal{Y})}) - I_\ell^{\text{reg}}(R^{(\mathcal{X})}) \;&\leq\; \left\langle \nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})}), R^{(\mathcal{X}\cup\mathcal{Y})} - R^{(\mathcal{X})}\right\rangle - \frac{m_{\overline{k}}}{2}\left\|R^{(\mathcal{X}\cup\mathcal{Y})} - R^{(\mathcal{X})}\right\|_2^2 \\ &\leq\; \max_{v : v_{(\mathcal{X}\cup\mathcal{Y}\cup\mathcal{G})^c} = 0} \left\langle \nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})}), v - R^{(\mathcal{X})}\right\rangle - \frac{m_{\overline{k}}}{2}\left\|v - R^{(\mathcal{X})}\right\|_2^2.\end{aligned}$$

Setting $v = R^{(\mathcal{X})} + \frac{1}{m_{\overline{k}}}\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_{\mathcal{X}\cup\mathcal{Y}\cup\mathcal{G}}$ that achieves the maximum above, we have

$$0 \;\leq\; I_\ell^{\text{reg}}(R^{(\mathcal{X}\cup\mathcal{Y})}) - I_\ell^{\text{reg}}(R^{(\mathcal{X})}) \;\leq\; \frac{1}{2m_{\overline{k}}}\left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_{\mathcal{X}\cup\mathcal{Y}\cup\mathcal{G}}\right\|_2^2$$

$$= \frac{1}{2m_{\overline{k}}}\left(\left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_{\mathcal{X}\cup\mathcal{G}}\right\|_2^2 + \left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_{\mathcal{Y}}\right\|_2^2\right),$$

where the last equality is due to $(\mathcal{X}\cup\mathcal{G})\cap\mathcal{Y}=\emptyset$.

Next, consider a single state $j\in\mathcal{Y}$. The function I_ℓ^{reg} at $R^{(\mathcal{X}\cup\{j\})}$ is larger than the function at any other R on the same support. In particular, $I_\ell^{\text{reg}}\left(R^{(\mathcal{X}\cup\{j\})}\right) \geq I_\ell^{\text{reg}}(y_j)$, where $y_j := R^{(\mathcal{X})} + \frac{\kappa}{\tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+1}}\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_j$. Noting that $\left(x=R^{(\mathcal{X})}, y=y_j\right) \in \tilde{\Omega}_{|\mathcal{X}|+|\mathcal{G}|+1}$ and applying Definition 1:

$$\begin{aligned}
& I_\ell^{\text{reg}}(R^{(\mathcal{X}\cup\{j\})}) - I_\ell^{\text{reg}}(R^{(\mathcal{X})}) \\
\geq\ & I_\ell^{\text{reg}}\left(R^{(\mathcal{X})} + \frac{\kappa}{\tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+1}}\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_j\right) - I_\ell^{\text{reg}}(R^{(\mathcal{X})}) \\
\geq\ & \left\langle \nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})}), \frac{\kappa}{\tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+1}}\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_j\right\rangle - \frac{\tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+1}}{2}\left\|\frac{\kappa}{\tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+1}}\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_j\right\|_2^2 \\
=\ & \frac{\kappa}{\tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+1}}\left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_j\right\|_2^2 - \frac{\kappa^2}{2\tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+1}}\left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_j\right\|_2^2 \\
\geq\ & \frac{\kappa}{2\tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+1}}\left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_j\right\|_2^2.
\end{aligned}$$

Summing over all $j\in\mathcal{Y}$:

$$\begin{aligned}
\sum_{j\in\mathcal{Y}}\left[I_\ell^{\text{reg}}(R^{(\mathcal{X}\cup\{j\})}) - I_\ell^{\text{reg}}(R^{(\mathcal{X})})\right] &\geq \frac{\kappa}{2\tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+1}}\sum_{j\in\mathcal{Y}}\left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_j\right\|_2^2 \\
&= \frac{\kappa}{2\tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+1}}\left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_{\mathcal{Y}}\right\|_2^2.
\end{aligned}$$

Then, we have:

$$\begin{aligned}
\gamma_{\mathcal{X},\mathcal{Y}} &\geq \frac{\kappa\cdot m_{|\mathcal{X}|+|\mathcal{G}|+k}}{\tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+1}}\cdot\frac{\left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_{\mathcal{Y}}\right\|_2^2}{\left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_{\mathcal{X}\cup\mathcal{G}}\right\|_2^2 + \left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_{\mathcal{Y}}\right\|_2^2} \\
&= \frac{\kappa\cdot m_{|\mathcal{X}|+|\mathcal{G}|+k}}{\tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+1}}\cdot\frac{1}{\frac{\left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_{\mathcal{X}\cup\mathcal{G}}\right\|_2^2}{\left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_{\mathcal{Y}}\right\|_2^2}+1} \\
&\overset{(i)}{\geq} \frac{\kappa\cdot m_{|\mathcal{X}|+|\mathcal{G}|+k}}{\tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+1}}\cdot\frac{1}{\frac{\left(d_{\max}^{\text{opt}}\right)^2}{\left(d_{\min}^{\text{non}}\right)^2}+1} \\
&\overset{(ii)}{\geq} \frac{\kappa\cdot m_{|\mathcal{Z}|+|\mathcal{G}|+k}}{M_{|\mathcal{Z}|+|\mathcal{G}|+k}}\cdot\frac{1}{\frac{\left(d_{\max}^{\text{opt}}\right)^2}{\left(d_{\min}^{\text{non}}\right)^2}+1},
\end{aligned}$$

where (i) is due to $\left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_{\mathcal{X}\cup\mathcal{G}}\right\|_2^2 \leq \left(d_{\max}^{\text{opt}}\right)^2$ and $\left\|\nabla I_\ell^{\text{reg}}(R^{(\mathcal{X})})_{\mathcal{Y}}\right\|_2^2 \geq |\mathcal{Y}|\left(d_{\min}^{\text{non}}\right)^2 \geq \left(d_{\min}^{\text{non}}\right)^2$ (see Assumption 2); and (ii) is due to $m_{|\mathcal{X}|+|\mathcal{G}|+k} \geq m_{|\mathcal{Z}|+|\mathcal{G}|+k}$ and $M_{|\mathcal{Z}|+|\mathcal{G}|+k} \geq \tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+k} \geq \tilde{M}_{|\mathcal{X}|+|\mathcal{G}|+1}$ (note that $1\leq|\mathcal{Y}|\leq k$ and $1\leq|\mathcal{X}|\leq|\mathcal{Z}|$). $\square$

D Additional Details and Proofs for using State Abstractions (Section 3.5)

We present an extension of our EXPRD framework that is scalable to large state spaces by leveraging the techniques from state abstraction literature [32–34]. We use an abstraction $\phi : \mathcal{S} \to \mathcal{X}_\phi$, which is a mapping from high-dimensional state-space $\mathcal{S}$ to a low-dimensional latent space $\mathcal{X}_\phi$. Let $\phi^{-1}(x) := \{s \in \mathcal{S} : \phi(s) = x\}, \forall x \in \mathcal{X}_\phi$. We propose the following pipeline (called EXPRD-ABS):

1. By using the original MDP $\overline{M} = \left(\mathcal{S}, \mathcal{A}, T, \gamma, P_0, \overline{R}\right)$ and the abstraction ϕ, we construct an abstract MDP $\overline{M}_\phi = \left(\mathcal{X}_\phi, \mathcal{A}, T_\phi, \gamma, P_0, \overline{R}_\phi\right)$ as follows, $\forall x, x' \in \mathcal{X}_\phi, a \in \mathcal{A}$: $T_\phi(x'|x,a) = \frac{1}{|\phi^{-1}(x)|} \sum_{s \in \phi^{-1}(x)} \sum_{s' \in \phi^{-1}(x')} T(s'|s,a)$, and $\overline{R}_\phi(x,a) = \frac{1}{|\phi^{-1}(x)|} \sum_{s \in \phi^{-1}(x)} \overline{R}(s,a)$. We compute the set of optimal policies $\overline{\Pi}^*_\phi$ for the MDP $\overline{M}_\phi$.
2. We run our EXPRD framework on $\overline{M}_\phi$ with $\Pi^\dagger = \overline{\Pi}^*_\phi$, and the resulting reward is denoted $\widehat{R}_\phi$. The corresponding MDP is denoted by $\widehat{M}_\phi = \left(\mathcal{X}_\phi, \mathcal{A}, T_\phi, \gamma, P_0, \widehat{R}_\phi\right)$.
3. We define the reward function $\widehat{R}$ on the state space $\mathcal{S}$ as follows: $\widehat{R}(s,a) = \widehat{R}_\phi(\phi(s),a)$. The corresponding MDP is denoted by $\widehat{M} = \left(\mathcal{S}, \mathcal{A}, T, \gamma, P_0, \widehat{R}\right)$.

In summary, the EXPRD-ABS pipeline is given by: $\overline{M} \to \overline{M}_\phi \to \widehat{M}_\phi \to \widehat{M}$.

Define $\epsilon_\phi := \min_{x \in \mathcal{X}_\phi} \min_{a \in \mathcal{A} \setminus \overline{\Pi}^*_{\phi,x}} \overline{\delta}^*_{\phi,\infty}(x,a)$, where $\overline{\delta}^*_{\phi,\infty}$ is the ∞-step optimality gap in the abstract MDP $\overline{M}_\phi = \left(\mathcal{X}_\phi, \mathcal{A}, T_\phi, \gamma, P_0, \overline{R}_\phi\right)$. For our analysis, we require the abstraction $\phi : \mathcal{S} \to \mathcal{X}_\phi$ to satisfy the following conditions:

- ϕ is $(\epsilon_{\overline{R}}, \epsilon_T)$-approximate model irrelevant abstraction [34] for the MDP $\overline{M} = \left(\mathcal{S}, \mathcal{A}, T, \gamma, P_0, \overline{R}\right)$, i.e., $\forall s_1, s_2 \in \mathcal{S}$ where $\phi(s_1) = \phi(s_2)$, we have, $\forall a \in \mathcal{A}$: $\left|\overline{R}(s_1,a) - \overline{R}(s_2,a)\right| \leq \epsilon_{\overline{R}}$, and $\sum_{x' \in \mathcal{X}_\phi} \left|\sum_{s' \in \phi^{-1}(x')} \left(T(s'|s_1,a) - T(s'|s_2,a)\right)\right| \leq \epsilon_T$.
- The change in the transition dynamics T during the compression-decompression process using the abstraction ϕ is very small, i.e., $\max_{s,a} \sum_{s'} \left|T(s'|s,a) - \frac{T_\phi(\phi(s')|\phi(s),a)}{|\phi^{-1}(\phi(s'))|}\right| \leq \frac{(1-\gamma)^2 \epsilon_\phi}{2\gamma R_{\max}}$.

The following theorem shows that any optimal policy induced by the reward $\widehat{R}$ resulting from the EXPRD-ABS pipeline acts nearly optimal w.r.t. $\overline{R}$:

Theorem 2. *Let $\phi : \mathcal{S} \to \mathcal{X}_\phi$ satisfy the conditions discussed above. The original reward function $\overline{R}$, and the reward function $\widehat{R}$ output by the* EXPRD-ABS *pipeline satisfy the following:* $\max_s \left|\overline{V}^*_\infty(s) - \overline{V}^\pi_\infty(s)\right| \leq \frac{2\epsilon_{\overline{R}}}{(1-\gamma)^2} + \frac{\gamma \cdot \epsilon_T \cdot R_{\max}}{2(1-\gamma)^3}, \forall \pi \in \widehat{\Pi}^*$*, i.e., any optimal policy induced by $\widehat{R}$ acts nearly optimal w.r.t. $\overline{R}$.*

Proof. Given an abstract policy $\pi : \mathcal{X}_\phi \to \mathcal{A}$ acting on $\mathcal{X}_\phi$, we define the lifted policy $[\pi]_{\uparrow M} : \mathcal{S} \to \mathcal{A}$ as $[\pi]_{\uparrow M}(s) := \pi(\phi(s)), \forall s \in \mathcal{S}$. Similarly, given a set of policies $\Pi = \{\pi : \mathcal{X}_\phi \to \mathcal{A}\}$, we define $[\Pi]_{\uparrow M} := \{[\pi]_{\uparrow M} : \pi \in \Pi\}$. We define an auxiliary MDP $\widetilde{M} = \left(\mathcal{S}, \mathcal{A}, \widetilde{T}, \gamma, P_0, \widetilde{R}\right)$, where $\widetilde{R}(s,a) = \widehat{R}_\phi(\phi(s),a)$, and $\widetilde{T}(s'|s,a) = \frac{T_\phi(\phi(s')|\phi(s),a)}{|\phi^{-1}(\phi(s'))|}$.

Step $\overline{M} \to \overline{M}_\phi$. Since ϕ is $(\epsilon_{\overline{R}}, \epsilon_T)$-approximate model irrelevant abstraction, we have the following (see [34]):

$$\left|\overline{Q}^*_\infty(s,a) - \overline{Q}^*_{\phi,\infty}(\phi(s),a)\right| \leq \frac{\epsilon_{\overline{R}}}{1-\gamma} + \frac{\gamma \cdot \epsilon_T \cdot R_{\max}}{2(1-\gamma)^2}, \quad \forall s \in \mathcal{S}, a \in \mathcal{A},$$

where $\overline{Q}^*_{\phi,\infty}$ is the optimal action value function of the MDP $\overline{M}_\phi$. Then, for any $\pi \in \left[\overline{\Pi}^*_\phi\right]_{\uparrow \overline{M}}$, we have the following (see [61]):

$$\max_s \left|\overline{V}^*_\infty(s) - \overline{V}^\pi_\infty(s)\right| \leq \frac{2}{1-\gamma} \cdot \max_{s,a} \left|\overline{Q}^*_\infty(s,a) - \overline{Q}^*_{\phi,\infty}(\phi(s),a)\right| \leq \frac{2\epsilon_{\overline{R}}}{(1-\gamma)^2} + \frac{\gamma \cdot \epsilon_T \cdot R_{\max}}{2(1-\gamma)^3},$$

i.e., any optimal policy of $\overline{M}_\phi$, when lifted to $\mathcal{S}$, acts as a near-optimal policy in $\overline{M}$.

Step $\overline{M}_\phi \rightarrow \widehat{M}_\phi$. In the step 2 of our ExpRD-Abs pipeline, we set $\Pi^\dagger = \overline{\Pi}^*_\phi$. Our ExpRD framework ensures that any optimal policy for $\widehat{M}_\phi$ is also optimal in $\overline{M}_\phi$, i.e., $\widehat{\Pi}^*_\phi \subseteq \overline{\Pi}^*_\phi$. In addition, since $\Pi^\dagger = \overline{\Pi}^*_\phi$ and $\Pi^\dagger \subseteq \widehat{\Pi}^*_\phi$, we have that $\widehat{\Pi}^*_\phi = \overline{\Pi}^*_\phi$.

Step $\widehat{M}_\phi \rightarrow \widetilde{M}$. By the definition of $\widetilde{M}$, ϕ is a model irrelevant abstraction for $\widetilde{M}$. Thus, we have the following (see [34]):

$$\widetilde{Q}^*_\infty(s, a) \;=\; \widehat{Q}^*_{\phi,\infty}(\phi(s), a), \quad \forall s \in \mathcal{S}, a \in \mathcal{A}. \tag{12}$$

From the above equation, note that $\widetilde{\Pi}^* = \left[\widehat{\Pi}^*_\phi\right]_{\uparrow\widetilde{M}}$. Finally, we have that, for any $\pi \in \widetilde{\Pi}^*$:

$$\max_s \left|\overline{V}^*_\infty(s) - \overline{V}^\pi_\infty(s)\right| \;\leq\; \frac{2\epsilon_{\overline{R}}}{(1-\gamma)^2} + \frac{\gamma \cdot \epsilon_T \cdot R_{\max}}{2(1-\gamma)^3},$$

i.e., any optimal policy of $\widetilde{M}$ acts as a near-optimal policy in the original MDP $\overline{M}$.

Optimality in $\widetilde{M}$. Our ExpRD framework guarantees the following:

$$\widehat{Q}^{\pi^\dagger}_{\phi,\infty}(x, \pi^\dagger(x)) \;\geq\; \widehat{Q}^{\pi^\dagger}_{\phi,\infty}(x, a) + \epsilon_\phi, \quad \forall x \in \mathcal{X}_\phi, a \in \mathcal{A}\backslash\overline{\Pi}^*_{\phi,x}, \pi^\dagger \in \Pi^\dagger,$$

which can be rewritten as follows:

$$\widehat{Q}^*_{\phi,\infty}(\phi(s), \pi^\dagger(\phi(s))) \;\geq\; \widehat{Q}^*_{\phi,\infty}(\phi(s), a) + \epsilon_\phi, \quad \forall s \in \mathcal{S}, a \in \mathcal{A}\backslash\overline{\Pi}^*_{\phi,\phi(s)}, \pi^\dagger \in \Pi^\dagger.$$

From the above inequality and using (12), we have the following:

$$\widetilde{Q}^*_\infty(s, [\pi^\dagger]_{\uparrow\widetilde{M}}(s)) \;\geq\; \widetilde{Q}^*_\infty(s, a) + \epsilon_\phi, \quad \forall s \in \mathcal{S}, a \in \mathcal{A}\backslash\left[\overline{\Pi}^*_\phi\right]_{\uparrow\widetilde{M},s}, [\pi^\dagger]_{\uparrow\widetilde{M}} \in [\Pi^\dagger]_{\uparrow\widetilde{M}},$$

which can be rewritten as follows:

$$\widetilde{Q}^*_\infty(s, \pi^*(s)) \;\geq\; \widetilde{Q}^*_\infty(s, a) + \epsilon_\phi, \quad \forall s \in \mathcal{S}, a \in \mathcal{A}\backslash\widetilde{\Pi}^*_s, \pi^* \in \widetilde{\Pi}^*.$$

From the above inequality, for any deterministic policy $\pi \notin \widetilde{\Pi}^*$, we have (at least on one state $s \in \mathcal{S}$):

$$\widetilde{V}^*_\infty(s) \;=\; \widetilde{Q}^*_\infty(s, \pi^*(s)) \;\geq\; \widetilde{Q}^*_\infty(s, \pi(s)) + \epsilon_\phi \;\geq\; \widetilde{Q}^\pi_\infty(s, \pi(s)) + \epsilon_\phi \;=\; \widetilde{V}^\pi_\infty(s) + \epsilon_\phi,$$

i.e., $\max_s \left|\widetilde{V}^*_\infty(s) - \widetilde{V}^\pi_\infty(s)\right| \geq \epsilon_\phi$.

Comparison $\widehat{M}$ vs. $\widetilde{M}$. Now, we show that any deterministic optimal policy in $\widehat{M}$ is also optimal in $\widetilde{M}$, i.e., $\widehat{\Pi}^* \subseteq \widetilde{\Pi}^*$. Let $\max_{s,a} \left\|T(\cdot|s,a) - \widetilde{T}(\cdot|s,a)\right\|_1 = \beta_T$. Then, for any $\widehat{\pi} \in \widehat{\Pi}^*$ and $s \in \mathcal{S}$, we have:

$$\left|\widetilde{V}^*_\infty(s) - \widetilde{V}^{\widehat{\pi}}_\infty(s)\right| \;\leq\; \left|\widetilde{V}^*_\infty(s) - \widehat{V}^{\widehat{\pi}}_\infty(s)\right| + \left|\widehat{V}^{\widehat{\pi}}_\infty(s) - \widetilde{V}^{\widehat{\pi}}_\infty(s)\right| \;\leq\; \frac{2\gamma\beta_T R_{\max}}{(1-\gamma)^2} \;<\; \epsilon_\phi,$$

where the second last inequality is due to Lemma 3 and Lemma 4 from [36]. Then, from the optimality in $\widetilde{M}$, it must me the case that $\widehat{\pi} \in \widetilde{\Pi}^*$.

Finally, for any $\pi \in \widehat{\Pi}^*$, we have:

$$\max_s \left|\overline{V}^*_\infty(s) - \overline{V}^\pi_\infty(s)\right| \;\leq\; \frac{2\epsilon_{\overline{R}}}{(1-\gamma)^2} + \frac{\gamma \cdot \epsilon_T \cdot R_{\max}}{2(1-\gamma)^3},$$

i.e., any optimal policy of $\widehat{M}$ acts as a near-optimal policy in the original MDP $\overline{M}$. $\square$

E Additional Details and Results for RoomsNavEnv (Section 4.1)

In this appendix, we expand on Section 4.1 and provide a more detailed description of the setup as well as additional results. Full implementation of our techniques is available in a Github repo as mentioned in Footnote 1.

Recall that the MDP for RoomsNavEnv has $|\mathcal{S}| = 49$ states corresponding to cells in the grid-world and four actions given by $\mathcal{A} := \{$"up", "left", "down", "right"$\}$. To refer to a specific state, we will use an enumeration scheme where the bottom-left cell is $s = 0$; the cell numbers increase going from left to right and bottom to top. With this convention, the top-right cell with the goal is $s = 48$, and four "gates" (cells that need to be crossed to go across rooms when navigating to the goal) correspond to states $\{9, 15, 19, 37\}$. In this MDP, we have one goal state $s = 48$, i.e., the set $\mathcal{G}$ in the problem (P3) is $\{48\}$. Furthermore, the original reward function has $\overline{R}(48, \text{"right"}) = R_{\text{max}}$ and is 0 elsewhere.

Additional details for the techniques evaluated. Below, we describe different reward design techniques along with hyperparameters that are evaluated in this section. More concretely, we have:

(i) $\widehat{R}_{\text{Orig}}$ simply represents the default reward function $\overline{R}$.

(ii) $\widehat{R}_{\text{PBRS}}$ is obtained via the PBRS technique based on Eq. 1, see Section 2.

(iii) $\widehat{R}_{\text{Craft}(B)}$ is designed manually based on the ideas discussed in Section 2. For selecting the states that we will assign non-zero rewards, we first develop a set function D as described below after this list. Then, for a fixed budget B, we pick a set of top $B + |\mathcal{G}|$ states that maximize the value of the set function D. Then, we assign rewards to these picked states as follows: (a) for the B states excluding $|\mathcal{G}|$ goal states, we assign a reward of $+1$ for one of the optimal action and -1 for others; (b) for $|\mathcal{G}|$ goal states, we assign the same rewards as $\overline{R}$. For the evaluation, we use $B = 5$ and denote the function as $\widehat{R}_{\text{Craft}(B=5)}$.

(iv) $\widehat{R}_{\text{PBRS-Craft}(B=5)}$ is obtained via the reward shaping technique from [42]. First, we compute the optimal state value function $\widehat{V}^*_\infty$ w.r.t. $\widehat{R}_{\text{Craft}(B=5)}$ designed above, i.e., we need to solve the task with the reward function $\widehat{R}_{\text{Craft}(B=5)}$. Then, we obtain the reward function $\widehat{R}_{\text{PBRS-Craft}(B=5)}$ using the PBRS technique based on Eq. 1 with the value function $\widehat{V}^*_\infty$ instead of the optimal value function $\overline{V}^*_\infty$ w.r.t. $\overline{R}$.

(v) $\widehat{R}_{\text{ExpRD}(B,\lambda\to\infty)}$ is the reward function designed by our ExpRD framework for a budget B and an extreme setting of $\lambda \to \infty$. For this setting, the problem (P3) reduces to (P1) corresponding to the reward design with subgoals pre-selected by the function D—we use the same function D that we used for $\widehat{R}_{\text{Craft}}$ above. For the evaluation, we use $B = 5$ and denote the designed reward function as $\widehat{R}_{\text{ExpRD}(B=5,\lambda\to\infty)}$. As discussed in Section 3, the budget B here refers to the additional number of states that are allowed to be in $\text{supp}(R)$ along with the goal states $\mathcal{G}$ (see (P3)). Apart from hyperparameters B and λ, ExpRD requires a choice of $\Pi^\dagger$, $\mathcal{H}$, and $I(R)$ – we discuss that below after this list.

(vi) $\widehat{R}_{\text{ExpRD}(B,\lambda=0)}$ is the reward function designed by our ExpRD framework for a budget B and an important setting of $\lambda = 0$ where the problem (P3) reduces to (P2) corresponding to fully automated reward design without using any prior knowledge about the importance of states. For budget B, we consider values from $\{3, 5, |S|\}$ and denote the designed reward functions as $\widehat{R}_{\text{ExpRD}(B=3,\lambda=0)}$, $\widehat{R}_{\text{ExpRD}(B=5,\lambda=0)}$, and $\widehat{R}_{\text{ExpRD}(B=|S|,\lambda=0)}$. As stated above, the budget B here refers to the additional number of states that are allowed to be in $\text{supp}(R)$ along with the goal states $\mathcal{G}$; the choice of $\Pi^\dagger$, $\mathcal{H}$, and $I(R)$ is discussed below.

Here we describe the set function D used for computing $\widehat{R}_{\text{Craft}(B=5)}$ and $\widehat{R}_{\text{ExpRD}(B=5,\lambda\to\infty)}$. For the set function D, we used a simple modular function given by $D(\mathcal{Z}) := \sum_{s\in\mathcal{Z}} w_s$ where w_s is a weight/score assigned to a state s capturing its importance in terms of reward design. We used the following weights: $w_s = 2$ for $s = 48$ (the goal state); $w_s = 1$ for $s = 9$, $s = 15$, $s = 19$, and $s = 37$ (the four "gates"); $w_s = 0.5$ for $s = 8$, $s = 11$, $s = 29$, and $s = 32$ (centers of the four rooms); and $w_s = 0.1$ otherwise. Even though this function is simple, it captures the prior knowledge one expects to intuitively apply in practice. In general, one could learn such D automatically using the techniques from [21–24].

Reward $\widehat{R}$	Sparseness $\lvert\text{supp}(\widehat{R})\rvert$	Invariance property Eq. 13	Eq. 14	Informativeness $I(\widehat{R})$	Convergence: #Episodes to % value 25%	75%	95%
$\widehat{R}_{\text{ORIG}}$	1	0.0009	0.0009	-0.1557	1,688	6,752	20,570
$\widehat{R}_{\text{PBRS}}$	49	0.0009	0.0009	0.0000	3	5	15
$\widehat{R}_{\text{CRAFT}(B=5)}$	6	-4.8366	-0.1645	-0.1122	1010	∞	∞
$\widehat{R}_{\text{PBRS-CRAFT}(B=5)}$	49	0.0009	0.0009	-0.0797	35	79	146
$\widehat{R}_{\text{EXPRD}(B=5,\lambda\to\infty)}$	6	0.0000	0.0010	-0.1070	49	773	14,252
$\widehat{R}_{\text{EXPRD}(B=3,\lambda=0)}$	4	0.0000	0.0009	-0.0842	177	474	1,514
$\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$	6	0.0000	0.0009	-0.0709	37	280	822
$\widehat{R}_{\text{EXPRD}(B=\lvert S\rvert,\lambda=0)}$	49	0.0000	1.5147	0.0000	9	48	90

Figure 6: Results for ROOMSNAVENV. The designed reward functions are evaluated w.r.t. criteria of sparseness, invariance, informativeness, and convergence. Here, the invariance property is captured through two different notions stated in Eq. 13 and Eq. 14 (a negative value represents a violation in the invariance property). Convergence is measured w.r.t the number of episodes needed to get a specific % of the total expected reward, and are based on the convergence results in Figure 3a.

Apart from B and λ, EXPRD requires us to specify $\Pi^\dagger$, $\mathcal{H}$, and $I(R)$. For the results reported in Figures 3 and 6, we use the following parameter choices for EXPRD: $\mathcal{H} = \{1, 4, 8, 16, 32\}$, $I(R)$ is given by Eq. 15, and the set $\Pi^\dagger$ contains only one policy from $\overline{\Pi}^*$. Later in this section, we also consider variations of $\mathcal{H}$ and $I(R)$, and report additional results in Figures 9 and 10.

Results w.r.t. different criteria. Next, we evaluate the above-mentioned designed reward functions w.r.t. criteria of sparseness, invariance, informativeness, and convergence. Sparseness is measured by $\lvert\text{supp}(\widehat{R})\rvert$, and informativeness is measured by $I(\widehat{R})$ that is used in the optimization problem (P3). Convergence is measured w.r.t the number of episodes needed to get a specific % of the total expected reward, and is based on the convergence results in Figure 3a by taking various horizontal slices of the convergence plot. To measure the invariance property, we consider two different notions stated below:

$$\min_{\widehat{\pi}^* \in \widehat{\Pi}^*} \min_{s \in \mathcal{S}} \Big(\overline{Q}^*_\infty(s, \widehat{\pi}^*(s)) - \overline{Q}^*_\infty(s, \overline{\pi}^*(s)) \Big) \text{ for any } \overline{\pi}^* \in \overline{\Pi}^* \tag{13}$$

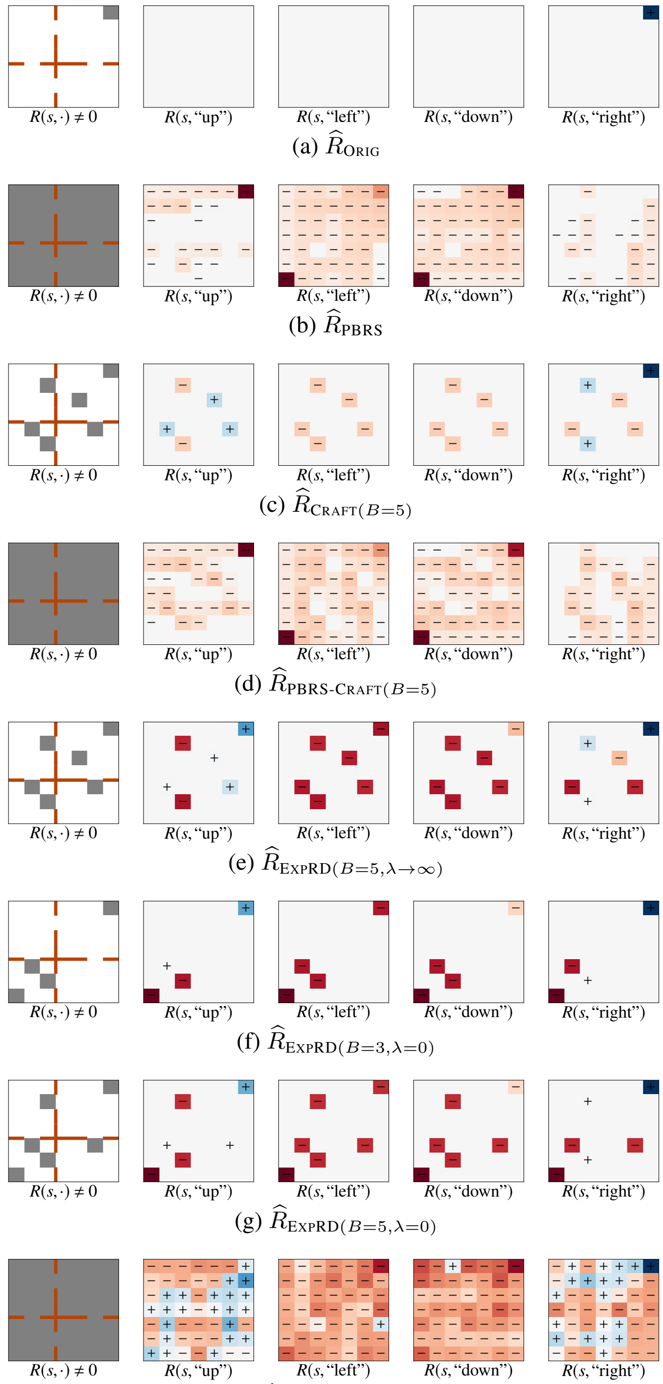
$$\min_{\pi \in \Pi^\dagger} \min_{s \in \mathcal{S}} \min_{a \in \mathcal{A} \setminus \overline{\Pi}^*_s} \Big(\widehat{Q}^\pi_\infty(s, \pi(s)) - \widehat{Q}^\pi_\infty(s, a) \Big) \tag{14}$$

The notion in Eq. (13) looks at one of the optimal policy $\widehat{\pi}^*$ w.r.t. $\widehat{R}$, and compares the gap in Q action values w.r.t. $\overline{R}$ – this quantity should be zero to ensure that none of the optimal policies w.r.t. $\widehat{R}$ is suboptimal w.r.t. $\overline{R}$. The notion in (14) is closer to the invariance constraint that we incorporate in the optimization problem of EXPRD – this quantity should be non-negative to ensure that none of the optimal policies w.r.t. $\widehat{R}$ is suboptimal w.r.t. $\overline{R}$.

In Figure 6, we compare the designed reward functions w.r.t. these different criteria. In the "Sparseness" column, the quantity $\lvert\text{supp}(\widehat{R})\rvert$ is $B + 1$ for $\widehat{R}_{\text{CRAFT}(B=5)}$, $\widehat{R}_{\text{EXPRD}(B=5,\lambda\to\infty)}$, $\widehat{R}_{\text{EXPRD}(B=3,\lambda=0)}$, and $\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$ as the goal states $\mathcal{G}$ are included in the design. In the "Invariance property" columns, we see that $\widehat{R}_{\text{CRAFT}(B=5)}$ fails to satisfy the invariance property highlighting the well-known "reward bugs" that can arise in this approach and mislead the agent into learning suboptimal policies (see Section 2 and [2, 3]); this issue is further emphasized in the "Convergence" columns for $\widehat{R}_{\text{CRAFT}(B=5)}$, highlighting that the agent is stuck with a suboptimal policy.

The last three columns related to "Convergence" highlight that the informativeness criteria we use in the optimization problem is a useful indicator about the agent's convergence when learning from designed reward functions. Furthermore, EXPRD can provide an effective trade-off in sparseness and informativeness while maintaining invariance property and speed up the agent's convergence. Even for small budgets of $B = 3$ or $B = 5$, the reward functions $\widehat{R}_{\text{EXPRD}(3,\lambda=0)}$ and $\widehat{R}_{\text{EXPRD}(5,\lambda=0)}$ lead to substantial speedups in the agent's convergence in contrast to the original reward function $\overline{R}$. Figures 7f and 7g further highlights that the states picked by EXPRD are important – the Algorithm 1 automatically picked the "gates" in the design process.

Visualizations of the designed reward functions. Figure 7 below shows a visualization of the eight different designed reward functions – this visualization is a variant of the visualization shown in Figure 3, where only three reward functions were shown.



(a) $\widehat{R}_{\text{ORIG}}$

(b) $\widehat{R}_{\text{PBRS}}$

(c) $\widehat{R}_{\text{CRAFT}(B=5)}$

(d) $\widehat{R}_{\text{PBRS-CRAFT}(B=5)}$

(e) $\widehat{R}_{\text{ExpRD}(B=5,\lambda\to\infty)}$

(f) $\widehat{R}_{\text{ExpRD}(B=3,\lambda=0)}$

(g) $\widehat{R}_{\text{ExpRD}(B=5,\lambda=0)}$

(h) $\widehat{R}_{\text{ExpRD}(B=|S|,\lambda=0)}$

Figure 7: Results for ROOMSNAVENV. These plots show visualization of different designed reward functions discussed in Figure 6 – this visualization is a variant of the visualization shown in Figure 3 where only three reward functions were shown. For each of the reward functions, the first plot titled $R(s, .) \neq 0$ shows which states have a non-zero reward assigned to at least one action and are marked with Gray color. The next four plots titled $R(s, \text{“up”})$, $R(s, \text{“left”})$, $R(s, \text{“down”})$, $R(s, \text{“right”})$ show rewards assigned to each state/action: here, a negative reward is shown in Red color with sign “$-$”, a positive reward is shown in Blue color with sign “$+$” and zero reward is shown in white. The magnitude of the reward is indicated by Red or Blue color intensity (see color representation in Figure 3).

Results w.r.t. variations in $I(R)$. For the results reported in Figures 3 and 9a, we fix $\mathcal{H} = \{1, 4, 8, 16, 32\}$, the set $\Pi^\dagger$ contains only one policy from $\overline{\Pi}^*$, and we use the following functional form for $I(R)$ corresponding to the negated hinge loss:

$$I_1(R) := \frac{1}{|\Pi^\dagger| \cdot |\mathcal{H}| \cdot |\mathcal{S}|} \cdot \sum_{\pi^\dagger \in \Pi^\dagger} \sum_{h \in \mathcal{H}} \sum_{s \in \mathcal{S}} \max_{a \in \mathcal{A} \setminus \overline{\Pi}^*_s} \Big(-\max(0, \overline{\delta}^*_\infty(s) - \delta^{\pi^\dagger}_h(s,a)) \Big) \tag{15}$$

Here, we perform additional experiments to understand the effect of variations in $I(R)$ on the reward functions designed by ExpRD. In Figures 9b, 9c, and 9d, we consider the following different functional forms of $I(R)$ corresponding to the negated hinge loss, respectively:

$$I_2(R) := \frac{1}{|\Pi^\dagger| \cdot |\mathcal{H}| \cdot |\mathcal{S}|} \cdot \sum_{\pi^\dagger \in \Pi^\dagger} \sum_{h \in \mathcal{H}} \sum_{s \in \mathcal{S}} \max_{a \in \mathcal{A} \setminus \overline{\Pi}^*_s} \Big(-\max(0, \overline{\delta}^*_\infty(s,a) - \delta^{\pi^\dagger}_h(s,a)) \Big) \tag{16}$$

$$I_3(R) := \frac{1}{|\Pi^\dagger| \cdot |\mathcal{H}| \cdot |\mathcal{S}|} \cdot \sum_{\pi^\dagger \in \Pi^\dagger} \sum_{h \in \mathcal{H}} \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A} \setminus \overline{\Pi}^*_s} \Big(-\max(0, \overline{\delta}^*_\infty(s) - \delta^{\pi^\dagger}_h(s,a)) \Big) \tag{17}$$

$$I_4(R) := \frac{1}{|\Pi^\dagger| \cdot |\mathcal{H}| \cdot |\mathcal{S}|} \cdot \sum_{\pi^\dagger \in \Pi^\dagger} \sum_{h \in \mathcal{H}} \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A} \setminus \overline{\Pi}^*_s} \Big(-\max(0, \overline{\delta}^*_\infty(s,a) - \delta^{\pi^\dagger}_h(s,a)) \Big) \tag{18}$$

Finally, in Figures 9e and 9f, we use the following different functional forms of $I(R)$ corresponding to the linear and negated exponential functions (instead of negated hinge loss), respectively:

$$I_5(R) := \frac{1}{|\Pi^\dagger| \cdot |\mathcal{H}| \cdot |\mathcal{S}|} \cdot \sum_{\pi^\dagger \in \Pi^\dagger} \sum_{h \in \mathcal{H}} \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A} \setminus \overline{\Pi}^*_s} \Big(-(\overline{\delta}^*_\infty(s,a) - \delta^{\pi^\dagger}_h(s,a)) \Big) \tag{19}$$

$$I_6(R) := \frac{1}{|\Pi^\dagger| \cdot |\mathcal{H}| \cdot |\mathcal{S}|} \cdot \sum_{\pi^\dagger \in \Pi^\dagger} \sum_{h \in \mathcal{H}} \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A} \setminus \overline{\Pi}^*_s} \Big(-\exp(\overline{\delta}^*_\infty(s,a) - \delta^{\pi^\dagger}_h(s,a)) \Big) \tag{20}$$

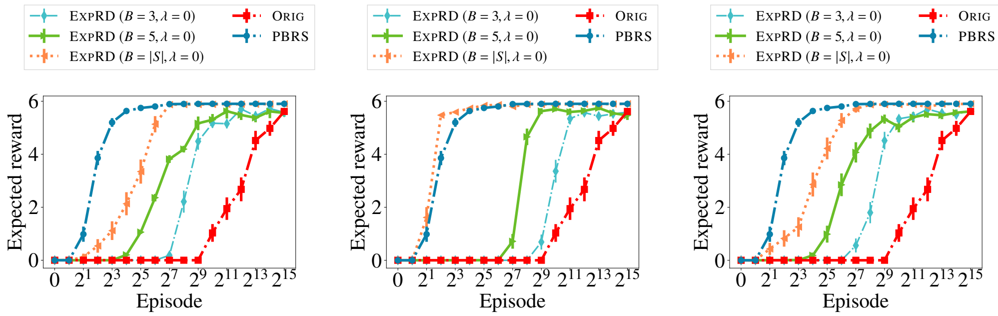
Additionally, we report results by varying the choice of the set $\mathcal{H}$. More concretely, in Figure 10, we fix the functional form of $I(R)$ as given Eq. 15, the set $\Pi^\dagger$ is same as above, and we vary the set $\mathcal{H}$ as follows: $\{1, 4, 8, 16, 32\}$, $\{1, 2, \ldots, 19, 20\}$, and $\{10, 11, \ldots, 19, 20\}$. Note that the value 20 corresponds to $\frac{1}{1-\gamma}$.

All the results in this section are reported as an average over 40 runs and convergence plots show mean with standard error bars. Overall, the convergence behavior in Figures 9 and 10 suggests that the reward functions designed by our ExpRD framework are effective under different functional forms of $I(R)$ and different choices of the set $\mathcal{H}$.

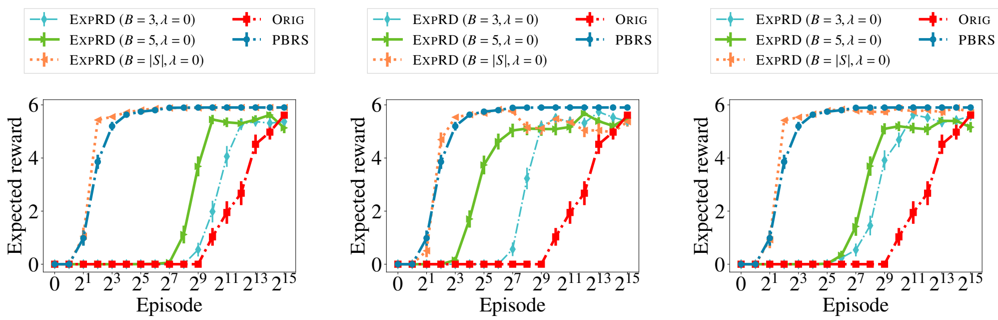
Run times for a varying number of states and actions. Here, we report the run times for solving an instance of the optimization problem (P1) when set $\mathcal{Z}$ is fixed. In order to easily vary the number of states $|\mathcal{S}|$ as well as the number of actions $|\mathcal{A}|$, we consider a simple chain navigation environment where an agent can take "left" or "right" actions to navigate across the states (think of this as a one-dimensional variant of RoomsNavEnv). To increase $|\mathcal{A}|$ beyond size 2, we added dummy actions which keep the agent's location unchanged. For reporting the run times, we consider $|\Pi^\dagger| = 1$, $\mathcal{H} = \{1, 4, 8, 16, 32\}$, and vary $|\mathcal{S}|$ as well as $|\mathcal{A}|$. These run times are reported when solving the formulation of the optimization problem in terms of matrices as shown in Section 2. Numbers are reported in seconds and are based on an average of 5 runs for each setting. These run times are obtained by running the computation on a laptop machine with 2.3 GHz Quad-Core Intel Core i5 processor and 16 GB RAM. Overall, these run times are of the same order as that of solving an optimization problem instance in environment poisoning attacks reported in the literature (see [19] and Section 5).

$|\mathcal{A}|$ \ $|\mathcal{S}|$	25	50	75	100	125	150	175	200
2	0.42s	0.91s	1.63s	2.35s	3.22s	4.34s	6.42s	7.62s
5	1.11s	3.04s	6.73s	13.48s	26.89s	51.52s	102.22s	335.38s

Figure 8: Run times for solving an instance of the optimization problem (P1) as we vary $|\mathcal{S}|$ and $|\mathcal{A}|$.

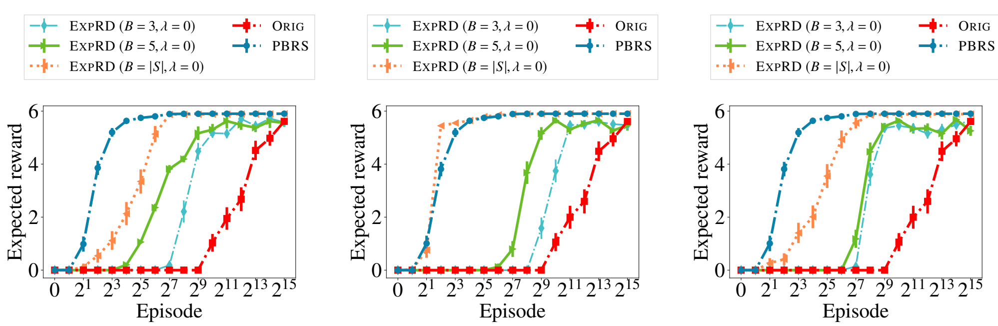


(a) $I(R)$ from Eq. 15 (b) $I(R)$ from Eq. 16 (c) $I(R)$ from Eq. 17



(d) $I(R)$ from Eq. 18 (e) $I(R)$ from Eq. 19 (f) $I(R)$ from Eq. 20

Figure 9: Results for ROOMSNAVENV. The plots show convergence in performance of the agent w.r.t. training episodes. Here, performance is measured as the expected reward per episode computed using $\overline{R}$; note that the x-axis is exponential in scale. As the parameter choices for EXPRD, we use $\mathcal{H} = \{1, 4, 8, 16, 32\}$ and the set $\Pi^\dagger$ contains only one policy from $\overline{\Pi}^*$. Each plot is obtained for a different functional form of $I(R)$.



(a) $\mathcal{H} = \{1, 4, 8, 16, 32\}$ (b) $\mathcal{H} = \{1, 2, \ldots, 19, 20\}$ (c) $\mathcal{H} = \{10, 11, \ldots, 19, 20\}$

Figure 10: Results for ROOMSNAVENV. The plots show convergence in performance of the agent w.r.t. training episodes. Here, performance is measured as the expected reward per episode computed using $\overline{R}$; note that the x-axis is exponential in scale. As the parameter choices for EXPRD, we use $I(R)$ from Eq. 15 and the set $\Pi^\dagger$ contains only one policy from $\overline{\Pi}^*$. Each plot is obtained for a different choice of $\mathcal{H}$. Note that Figure 10a is same as Figure 9a.

F Additional Details and Results for LINEKEYNAVENV (Section 4.2)

In this appendix, we expand on Section 4.2 and provide a more detailed description of the setup as well as additional results. Full implementation of our techniques is available in a Github repo as mentioned in Footnote 1.

Additional details for the techniques evaluated. Below, we describe different reward design techniques along with hyperparameters that are evaluated in this section. More concretely, we have:

(i) $\widehat{R}_{\text{ORIG}}$ simply represents the default reward function $\overline{R}$.

(ii) $\widehat{R}_{\text{PBRS}}$ is obtained via the PBRS technique based on Eq. 1 and using an abstraction (see Section 3.5, [35]). We first define an abstraction $\phi : \mathcal{S} \rightarrow \mathcal{X}_\phi$ as described below after this list. Based on this abstraction ϕ, we construct an abstract MDP $\overline{M}_\phi$ using the original MDP $\overline{M}$, and compute the optimal state value function $\overline{V}^*_{\phi,\infty}$ in the abstract MDP $\overline{M}_\phi$. Finally, we lift $\overline{V}^*_{\phi,\infty}$ to the original state space $\mathcal{S}$ (see Appendix D), and use the lifted value function as the potential function for the PBRS.

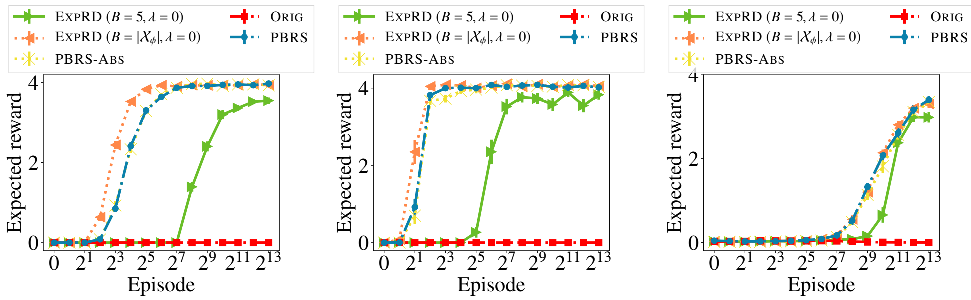
(iii) $\widehat{R}_{\text{PBRS-ABS}}$ is a variant of $\widehat{R}_{\text{PBRS}}$. Similar to $\widehat{R}_{\text{PBRS}}$, we compute the optimal state value function $\overline{V}^*_{\phi,\infty}$ in the abstract MDP $\overline{M}_\phi$. We use this value function as the potential function for the PBRS to design $\widehat{R}_{\text{PBRS},\phi}$ in the MDP $\overline{M}_\phi$. Finally, we lift $\widehat{R}_{\text{PBRS},\phi}$ to the original state space $\mathcal{S}$ (see Appendix D). Note that $\widehat{R}_{\text{PBRS-ABS}}$ is not guaranteed to satisfy the invariance property of $\widehat{R}_{\text{PBRS}}$.

(iv) $\widehat{R}_{\text{EXPRD}(B,\lambda=0)}$ is the reward function designed by our pipeline in Section 3.5 that relies on our EXPRD framework and an abstraction. We use the same abstraction $\phi : \mathcal{S} \rightarrow \mathcal{X}_\phi$ for all the techniques and is described below after this list. In the subroutine, we run EXPRD on $\overline{M}_\phi$ for a budget $B = 5$ and a full budget $B = |\mathcal{X}_\phi|$; we set $\lambda = 0$. We denote the designed reward functions as $\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$ and $\widehat{R}_{\text{EXPRD}(B=|\mathcal{X}_\phi|,\lambda=0)}$. Similar to Figure 9a, we fix $\mathcal{H} = \{1, 4, 8, 16, 32\}$, and we use the functional form given in Eq. 15 for $I(R)$.

Here, we describe the abstraction ϕ used for computing $\widehat{R}_{\text{PBRS}}$, $\widehat{R}_{\text{PBRS-ABS}}$, and $\widehat{R}_{\text{EXPRD}(B,\lambda=0)}$. Recall the description of the original MDP $\overline{M}$ from Section 4.2 – the state corresponds to the agent's status comprising of the current location (a point x in $[0, 1]$) and a binary flag whether the agent has acquired a key. For a given hyperparameter $\alpha \in (0, 1)$, we obtain a finite set of locations X by α-level discretization of the line segment $[0, 1]$, leading to a $1/\alpha$ number of locations. For the abstraction ϕ associated with this discretization, the abstract MDP $\overline{M}_\phi$ has $|\mathcal{X}_\phi| = 2/\alpha$ corresponding to $1/\alpha$ locations and a binary flag for the key. We use $\alpha = 0.05$ in the experiments.

Results for Q-learning agent with 0.01-level location discretization. For the results reported in the main paper (Figure 5a) and in Figure 11a, the agent uses Q-learning method in a discretized version of the original MDP $\overline{M}$ with a 0.01-level discretization of the location (i.e., the number of states in the agent's discretized MDP is 200). The rest of the method's parameters are same as in Section 4.1, i.e., we use standard Q-learning method for the agent with a learning rate 0.5 and exploration factor 0.1 [7]. During training, the agent receives rewards based on $\widehat{R}$, however, is evaluated based on $\overline{R}$. A training episode ends when the maximum steps (set to 50) is reached or an agent's action terminates the episode. For this agent, the convergence results are reported in Figure 11a as an average over 40 runs. These results demonstrate that all four designed reward functions—$\widehat{R}_{\text{PBRS}}$, $\widehat{R}_{\text{PBRS-ABS}}$, $\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$, $\widehat{R}_{\text{EXPRD}(B=|\mathcal{X}_\phi|,\lambda=0)}$—substantially improves the convergence, whereas the agent is not able to learn under $\widehat{R}_{\text{ORIG}}$.

Results for Q-learning agent with 0.005-level location discretization. Here, we demonstrate that our abstraction based pipeline in Section 3.5 is robust to the state representation used by the agent. In particular, for the results reported in Figure 11b, the agent uses a discretized version of the original MDP $\overline{M}$ with a 0.005-level discretization of the location. As in the setting above, the agent uses Q-learning method in this discretized version of the original MDP $\overline{M}$. Similar to Figure 11a, Figure 11b demonstrates that the performance associated with all four designed reward functions—$\widehat{R}_{\text{PBRS}}$, $\widehat{R}_{\text{PBRS-ABS}}$, $\widehat{R}_{\text{EXPRD}(B=5,\lambda=0)}$, $\widehat{R}_{\text{EXPRD}(B=|\mathcal{X}_\phi|,\lambda=0)}$—substantially improves the convergence in contrast to $\widehat{R}_{\text{ORIG}}$.



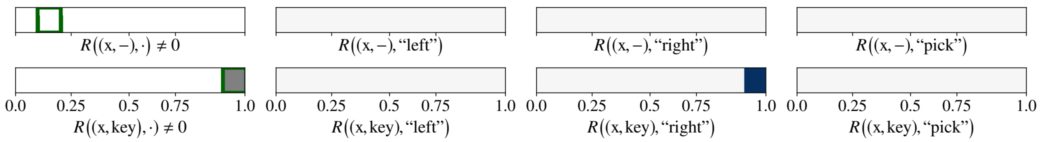
(a) Q-learning agent (0.01) (b) Q-learning agent (0.005) (c) REINFORCE agent

Figure 11: Results for LINEKEYNAVENV. These plots show convergence in performance of the agent w.r.t. training episodes. Here, performance is measured as the expected reward per episode computed using $\overline{R}$. **(a)** shows convergence for a Q-learning agent who uses a 0.01-level discretization of the location. **(b)** shows convergence for a Q-learning agent who uses a 0.005-level discretization of the location. **(c)** shows convergence for an agent who uses REINFORCE learning method with continuous representation of the location. All these agents receive rewards using the designed reward functions shown in Figure 12.

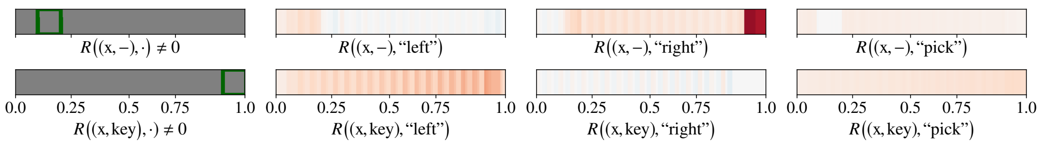
Results for REINFORCE agent with continuous location representation. For the results reported in Figure 11c, the agent uses the REINFORCE policy gradient method (see [7, 62]) in the original MDP $\overline{M}$ with continuous representation of the location. We use a neural network to learn the policy, which takes a continuous value in $[0, 1]$ (the location) and a binary flag (whether the agent has acquired a key) as the input representing a state s. The neural network has a hidden layer with 256 nodes. Given a state s (the input to the network), the policy network outputs three scores for three different actions. Then, applying softmax operation over these three scores gives the policy's action distribution. We use the REINFORCE method with a learning rate 0.0005. The gradient update happens at the end of each episode. In contrast to the maximum episode length of 50 used by Q-learning agents, we set this to 150 for the REINFORCE agent.

Figure 11c shows convergence results for this agent as an average over 20 runs; for each individual run, we additionally applied a moving-window average over a window size of 100 episodes. With neural representation for states, the policy invariance might not hold anymore. However, Figure 11c demonstrates that all four designed reward functions—$\widehat{R}_{\text{PBRS}}$, $\widehat{R}_{\text{PBRS-Abs}}$, $\widehat{R}_{\text{ExpRD}(B=5,\lambda=0)}$, $\widehat{R}_{\text{ExpRD}(B=|\mathcal{X}_\phi|,\lambda=0)}$—substantially improves the convergence (slightly weaker compared to Figures 11a and 11b), whereas the agent is not able to learn under $\widehat{R}_{\text{Orig}}$. This observation highlights our pipeline in Section 3.5 as a promising approach for reward design in high-dimensional settings. As future work, we plan to (both theoretically and empirically) investigate the effectiveness of the reward functions designed by our EXPRD framework or its adaptions in accelerating the learning process in high-dimensional settings for policy gradient methods.

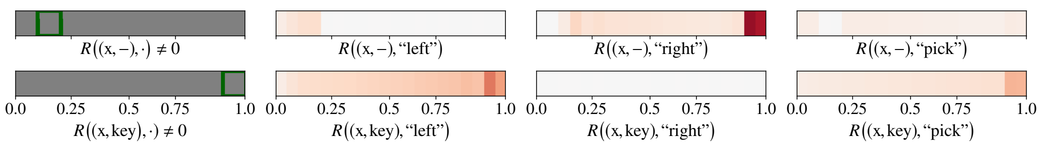
Visualizations of the designed reward functions. Figure 12 shows visualization of the five different designed reward functions discussed above – this visualization is a variant of the visualization shown in Figure 5 where only three reward functions were shown. This visualization provides important insights into the reward functions designed by EXPRD. Interestingly, $\widehat{R}_{\text{ExpRD}(B=5,\lambda=0)}$ assigned a high positive reward for the "pick" action when the agent is in the locations with key (see $R((\text{x}, -), \text{"pick"})$ bar in Figure 12d).



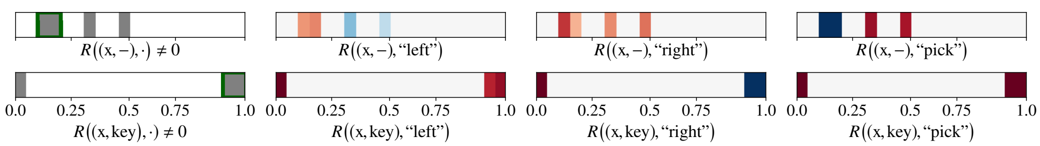
(a) $\widehat{R}_{\text{ORIG}}$



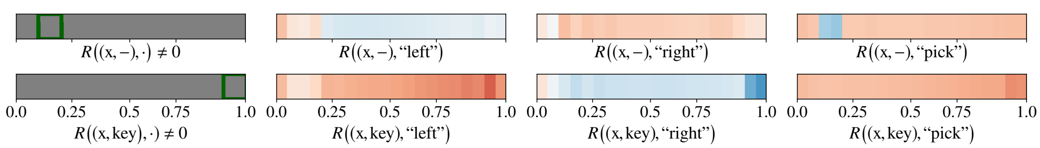
(b) $\widehat{R}_{\text{PBRS}}$



(c) $\widehat{R}_{\text{PBRS-ABS}}$



(d) $\widehat{R}_{\text{ExpRD}(B=5,\lambda=0)}$



(e) $\widehat{R}_{\text{ExpRD}(B=|\mathcal{X}_\phi|,\lambda=0)}$

Figure 12: Results for LINEKEYNAVENV. These plots show visualization of the five different designed reward functions discussed above – this visualization is a variant of the visualization shown in Figure 5 where only three reward functions were shown. For each of the reward functions, we show a total of 8 horizontal bars. Denoting a state as tuple $(\text{x}, -)$ (i.e., location x when the key has not been picked) or (x, key) (i.e., location x when the key has been picked), these 8 horizontal bars have the following interpretation. The two bars, titled $R((\text{x}, -), \cdot) \neq 0$ and $R((\text{x}, \text{key}), \cdot) \neq 0$, indicate states in Gray color for which a non-zero reward is assigned to at least one action; in these two bars, we have further highlighted the segment $[0.9, 1]$ with the goal, and the segment $[0.1, 0.2]$ with the key. The remaining six bars, titled $R((\text{x}, -), \text{“left”})$, $R((\text{x}, -), \text{“right”})$, $R((\text{x}, -), \text{“pick”})$, $R((\text{x}, \text{key}), \text{“left”})$, $R((\text{x}, \text{key}), \text{“right”})$, and $R((\text{x}, \text{key}), \text{“pick”})$, show rewards assigned to each state/action: here, a negative reward is shown in Red color, a positive reward is shown in Blue color, and zero reward is shown in white. The magnitude of the reward is indicated by Red or Blue color intensity and we use the same color representation as in Figure 5.