A Study of Global and Episodic Bonuses for Exploration in Contextual MDPs

Mikael Henaff [1] **Minqi Jiang** [1 2] **Roberta Raileanu** [1]

Abstract

Exploration in environments which differ across episodes has received increasing attention in recent years. Current methods use some combination of *global novelty bonuses*, computed using the agent's entire training experience, and *episodic novelty bonuses*, computed using only experience from the current episode. However, the use of these two types of bonuses has been ad-hoc and poorly understood. In this work, we shed light on the behavior of these two types of bonuses through controlled experiments on easily interpretable tasks as well as challenging pixel-based settings. We find that the two types of bonuses succeed in different settings, with episodic bonuses being most effective when there is little shared structure across episodes and global bonuses being effective when more structure is shared. We develop a conceptual framework which makes this notion of shared structure precise by considering the variance of the value function across contexts, and which provides a unifying explanation of our empirical results. We furthermore find that combining the two bonuses can lead to more robust performance across different degrees of shared structure, and investigate different algorithmic choices for defining and combining global and episodic bonuses based on function approximation. This results in an algorithm which sets a new state of the art across 16 tasks from the MiniHack suite used in prior work, and also performs robustly on Habitat and Montezuma's Revenge.

[1]Meta AI Research [2]University College, London. Correspondence to: Mikael Henaff <mikaelhenaff@meta.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

1. Introduction

Balancing exploration and exploitation is a long-standing challenge in reinforcement learning (RL). A large body of research has studied this problem within the Markov Decision Process (MDP) framework (Sutton & Barto, 2018), both from a theoretical standpoint (Kearns & Singh, 2002; Brafman & Tennenholtz, 2002; Agarwal et al., 2020) and an empirical one. This has led to practical exploration algorithms such as pseudocounts (Bellemare et al., 2016b), intrinsic curiosity modules (Pathak et al., 2017a) and random network distillation (Burda et al., 2019b), yielding impressive results on hard exploration problems like Montezuma's Revenge and PitFall (Bellemare et al., 2012).

More recently, there has been increasing interest in algorithms which move beyond the MDP framework. The standard MDP framework assumes that the agent is initialized in the same environment at each episode (we will refer to these MDPs as *singleton* MDPs). However, several studies have found that agents trained in singleton MDPs exhibit poor generalization, and that even minor changes to the environment can cause substantial degradation in agent performance (Justesen et al., 2018; Zhang et al., 2018a;b; Packer et al., 2018; Farebrother et al., 2018; Cobbe et al., 2019; Song et al., 2020; Kirk et al., 2021a). This has motivated the use of *contextual* MDPs (CMDPs) (Hallak et al., 2015), where different episodes correspond to different environments which nevertheless share structure. Examples of CMDPs include procedurally-generated environments (Chevalier-Boisvert et al., 2018; Samvelyan et al., 2021; Küttler et al., 2020; Juliani et al., 2019; Cobbe et al., 2020; Beattie et al., 2016; Hafner, 2021; Petrenko et al., 2021) or embodied AI tasks where the agent must generalize across different physical spaces (Savva et al., 2019; Shen et al., 2020; Gan et al., 2020; Xiang et al., 2020).

While exploration is well-studied in the singleton MDP case, it becomes more nuanced when dealing with CMDPs. For singleton MDPs, a common and successful strategy consists of defining an exploration bonus which is added to the reward function being optimized. This exploration bonus typically represents how novel the current state is, where novelty is computed with respect to the entirety of the agent's experience across all episodes. However, it is unclear to what extent this strategy is applicable in the CMDP setting—if

two environments corresponding to different episodes are very different, we might not want the experience gathered in one to affect the novelty of a state observed in the other. For example, if an agent is faced with procedurally generated maps with random start and goal locations, exploring the top-left corner of one map does not necessarily mean it should not visit the top-left corner of a different map, since their contents may be different.

An alternative to using global bonuses is to use episodic ones. Episodic bonuses define novelty with respect to the experience gathered in the current episode alone, rather than across all episodes. Recently, several works (Stanton & Clune, 2018; Raileanu & Rocktäschel, 2020; Flet-Berliac et al., 2021; Zhang et al., 2021b; Henaff et al., 2022; Wang et al., 2023) have used episodic bonuses, with Henaff et al. (2022) and Wang et al. (2023) showing that this is an essential ingredient for solving many sparse reward CMDPs. However, as we will show here, an episodic bonus alone may not be optimal if there is considerable shared structure across different episodes in the CMDP.

In this work, we study the strengths and weaknesses of global and episodic novelty bonuses for exploration in CMDPs, and investigate ways to mitigate their limitations. First, through a series of easily interpretable examples, we show that *global bonuses, which are commonly used in singleton MDPs, can be poorly suited for CMDPs that share little structure across episodes; however, episodic bonuses, which are commonly used in CMDPs, can also fail in cases where knowledge transfer across episodes is crucial.* We develop a conceptual framework which makes this notion of shared structure precise by considering the variance of the value function in representation space across contexts, providing a unifying explanation of our empirical results. Second, we show that by multiplicatively combining episodic and global bonuses, we are able to get more robust performance on both contextual MDPs that share little structure across episodes and singleton MDPs that are identical across episodes. We furthermore validate our findings in two challenging pixel-based settings, Habitat (Savva et al., 2019) and Montezuma's Revenge (Bellemare et al., 2012), demonstrating that the tradeoffs between bonus types and advantages of the combined bonus apply there as well. Third, motivated by these observations, we comprehensively evaluate different combinations of episodic and global bonuses which do not rely on counts, as well as strategies for integrating them, on a wide array of tasks from the MiniHack suite (Samvelyan et al., 2021). Our investigations yield an algorithm which combines the elliptical episodic bonus of Henaff et al. (2022) and the RND global bonus of Burda et al. (2019b), and sets a new state of the art across 16 tasks from the MiniHack environment, solving the majority of them. Our code is available at: www.github.com/facebookresearch/e3b.

2. Background

2.1. Contextual MDPs

We consider a contextual Markov Decision Process (CMDP) defined by $(\mathcal{S}, \mathcal{A}, \mathcal{C}, P, r, \mu_C, \mu_S)$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{C}$ is the context space, P is the transition function, μ_S is the initial state distribution conditioned on the context, and μ_C is the context distribution. At each episode, we first sample a context $c \sim \mu_C$ and an initial state $s_0 \sim \mu_S(\cdot|c)$. At each step t in the episode, the next state is then sampled according to $s_{t+1} \sim P(\cdot|s_t, a_t, c)$ and the reward is given by $r_t = r(s_t, a_t)$. Let d_π^c represent the distribution over states induced by following policy π with context c. The goal is to learn a policy which maximizes the expected return, averaged across contexts:

$$R = \mathbb{E}_{c\sim\mu_C, s\sim d_\pi^c, a\sim\pi(\cdot|s)}[r(s,a)]$$

Examples of CMDPs include procedurally-generated environments, such as ProcGen (Cobbe et al., 2020), MiniGrid (Chevalier-Boisvert et al., 2018), NetHack (Küttler et al., 2020), or MiniHack (Samvelyan et al., 2021), where each context c corresponds to the random seed used to generate the environment. In this case, the number of contexts $|\mathcal{C}|$ is effectively infinite and we will slightly abuse notation by writing $|\mathcal{C}| = \infty$. Other examples include embodied AI environments (Savva et al., 2019; Szot et al., 2021; Gan et al., 2020; Shen et al., 2020; Xiang et al., 2020), where the agent is placed in different simulated houses and must navigate to a location or find an object. In this setting, each context $c \in \mathcal{C}$ represents a house identifier and the number of houses $|\mathcal{C}|$ is typically between 20 and 1000. For an in-depth review of the literature on CMDPs and generalization in RL, see (Kirk et al., 2021b). Singleton MDPs are a special case of contextual MDPs with $|\mathcal{C}| = 1$.

2.2. Exploration Bonuses

At a high level, exploration bonuses operate by estimating the novelty of a given state, and assign a high bonus if the state is novel according to some measure. The exploration bonus is then combined with the extrinsic reward provided by the environment, and the result is optimized using RL. More precisely, the reward function optimized by the agent is given by $\bar{r}(s,a) = r(s,a) + \alpha \cdot b(s,a)$, where $r(s,a)$ is the extrinsic reward, $b(s,a)$ is the exploration bonus, and α is a parameter governing the balance between exploration and exploitation. Some bonuses do not depend on a or additionally depend the next state s', which will be clear from the context. To account for the variations in scale across different environments and times during training, the exploration bonus is sometimes divided by a running estimate of its standard deviation (Burda et al., 2019b).

In tabular domains with a small number of discrete states, a common choice is to use the inverse counts: $b(s) =$

$1/\sqrt{N(s)}$ (Strehl & Littman, 2006), where $N(s)$ is the number of times state s has been encountered by the agent. However, in most settings of interest the number of states is large or infinite, and many states will not be seen more than once, rendering this bonus ineffective. This has motivated alternative approaches using function approximation. The methods below have proven successful on sparse reward singleton MDPs (RND) and/or sparse reward CMDPs (RIDE, AGAC, NovelD and E3B).

Random Network Distillation (RND) (Burda et al., 2019b) randomly initializes a neural network $\bar{f} : \mathcal{S} \to \mathbb{R}^k$, and then trains a second neural network f with the same architecture to predict the outputs of $\bar{f}$ on states encountered by the agent. The exploration bonus associated with a given state s is given by the mean squared error:

$$b_{\text{RND}}(s_t) = \|f(s_t) - \bar{f}(s_t)\|_2^2 \tag{1}$$

The intuition is that for states similar to ones previously encountered by the agent, the error will be low, whereas it will be high for very different states. RND has performed well on hard singleton MDPs and is a commonly used component of other exploration algorithms.

Novelty Difference (NovelD) (Zhang et al., 2021b) uses the difference between RND bonuses at two consecutive time steps, regulated by an episodic count-based bonus. Specifically, its bonus is:

$$b_{\text{NovelD}}(s_t, a, s_{t+1}) = \Big[b_{\text{RND}}(s_{t+1}) - c \cdot b_{\text{RND}}(s_t)\Big]_+ \cdot \mathbb{I}[N_e(s_{t+1}) = 1] \tag{2}$$

Here b_{RND} represents the RND bonus defined above, and $N_e(s)$ represents the number of times s has been encountered within the current episode. The first term is a *global novelty bonus*, which measures novelty with respect to cross-episode experience, whereas the second term is an *episodic novelty bonus*, which measures novelty with respect to experience within the current episode only.

Adversarially Guided Actor-Critic (AGAC) (Flet-Berliac et al., 2021) also combines global and episodic novelty bonuses. Its bonus is defined by:

$$b_{\text{AGAC}}(s_t) = D_{\text{KL}}(\pi(\cdot|s_t)\|\pi_{\text{adv}}(\cdot|s_t)) + \beta \frac{1}{\sqrt{N_e(s_t)}} \tag{3}$$

where π_{adv} is a policy trained to mimic the behavior policy π (usually with a smaller learning rate). The motivation is that this will encourage the policy to adopt different behaviors as it tries to remain different from the adversary. The second term is an episodic bonus based on $N_e(s)$, the number of times the state s has been encountered within the current episode.

Rewarding Impact-Driven Exploration (RIDE) (Raileanu & Rocktäschel, 2020) uses an episodic novelty bonus which is the product of two terms: a count-based reward and the difference between two consecutive state embeddings:

$$b_{\text{RIDE}}(s_t) = \frac{1}{\sqrt{N_e(s_t)}} \|\phi(s_{t+1}) - \phi(s_t)\|_2 \tag{4}$$

Here the ϕ embedding is learned using a combination of inverse and forward dynamics models. The motivation for the second term in the bonus is to reward the agent for taking actions which cause significant changes in the environment. RIDE does not use a global novelty bonus.

Exploration via Elliptical Episodic Bonuses (E3B) (Henaff et al., 2022) also uses an episodic novelty bonus only, and is motivated by the following observation: while the count-based episodic bonuses used in NovelD, RIDE and AGAC are essential for good performance, they do not scale to complex environments where each state is rarely seen more than once. E3B uses a feature extractor ϕ learned using an inverse dynamics model, and defines the episodic bonus as follows:

$$b_{\text{E3B}}(s_t) = \phi(s_t)^\top \Big[\sum_{i=t_0}^{t-1} \phi(s_i)\phi(s_i)^\top + \lambda I\Big]^{-1} \phi(s_t) \tag{5}$$

Here t_0 denotes the start of the current episode. This can be seen as a generalization of an episodic count-based bonus to continuous state spaces, by noting that it reduces to inverse episodic counts if ϕ is a one-hot encoding of the state.

3. When are Global and Episodic Novelty Bonuses Useful?

Although RIDE, NovelD, AGAC and E3B all use different combinations of episodic and global novelty bonuses, their use in CMDPs has been largely heuristic. The RIDE and NovelD papers simply state that the episodic bonus is included to prevent the agent from going back and forth between a sequence of states within the same episode. Furthermore, the global novelty bonuses are justified using the singleton MDP case, but it is unclear to what extent these justifications carry over to the CMDP case. Therefore, a closer investigation of when episodic and global novelty bonuses are useful in CMDPs is required. All experiment details for this section are included in Appendix E.1.

3.1. Advantages of Episodic Bonuses

We begin by providing an example of CMDPs where global novelty bonuses fail and episodic bonuses succeed. Consider the procedurally-generated MiniHack environment

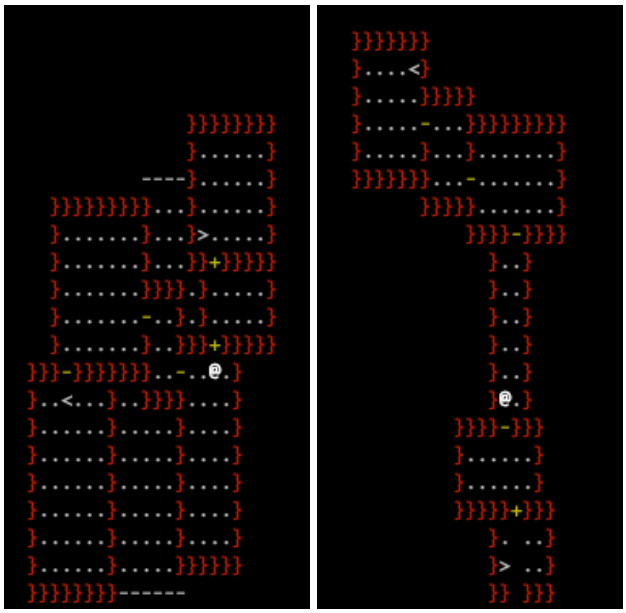


Figure 1. Two different contexts of the `MultiRoom-N6-Lava` environment. Legend: `@`: agent, `<`: start, `>`: goal, `}`: lava

shown in Figure 1. Here, each episode corresponds to a different map where the agent must navigate from the starting location to the goal. The agent only receives reward if it reaches the goal, and the episode terminates if it touches the walls which are made of lava. Because of this, random exploration has a very small chance of reaching the goal before the episode ends, and exploration bonuses are needed.

We ask the question: are global or episodic novelty bonuses more appropriate here? For simplicity, we consider bonuses based on counts of (x, y) locations, which have been commonly used in prior work (Flet-Berliac et al., 2021; Samvelyan et al., 2021; Zhang et al., 2021b) to avoid the issue of each state being unique:

$$b_{\text{global}}(s) = \frac{1}{\sqrt{N(\psi(s))}}, \quad b_{\text{episodic}}(s) = \mathbb{I}[N_e(\psi(s)) = 1]^1 \tag{6}$$

Here N represents counts over all the agent's experience, and N_e represents counts within the current episode only, while ψ is a feature extractor which extracts the (x, y) coordinates of the agent from the state. In general, methods which do not require handcrafted features are preferable, and we focus on them later on in this section and in Section 4. However, this simple bonus facilitates interpretability, which is the present focus.

[1]We also tried $\frac{1}{\sqrt{N_e(\psi(s))}}$, but it performed worse.

Environment	$\lvert\mathcal{C}\rvert$	ψ	Global	Episodic
`MultiRoom`	1	P	0.99 ± 0.00	0.83 ± 0.23
`MultiRoom`	3	P	0.59 ± 0.32	0.92 ± 0.13
`MultiRoom`	5	P	0.23 ± 0.39	0.98 ± 0.02
`MultiRoom`	10	P	0.02 ± 0.06	0.78 ± 0.17
`MultiRoom`	∞	P	0.00 ± 0.00	0.87 ± 0.10
`Corridors`	1	P	0.96 ± 0.03	0.10 ± 0.68
`KeyRoom`	∞	M	0.97 ± 0.00	0.89 ± 0.01
`MultiRoom`	∞	M	0.99 ± 0.01	0.59 ± 0.49

Table 1. Reward for global and episodic bonuses for different CMDPs, averaged across 5 seeds. Performance is close to 0 for all environments if no bonus is used. Here $\lvert\mathcal{C}\rvert$ denotes the number of different contexts/maps which are sampled from at each episode. The ψ column indicates which feature encodings are used (P for positions, M for messages).

We train agents using the global and episodic bonuses in equation 6 over different numbers of contexts $\lvert\mathcal{C}\rvert$ on the `MultiRoom` environment shown in Figure 1. The number of contexts represents the number of distinct maps, and one of them is chosen at random at the start of each episode. Results are shown in the top section of Table 1. The agent using the global bonus consistently obtains near-perfect performance for the singleton MDP setting where $\lvert\mathcal{C}\rvert = 1$, but performance steadily degrades as the number of contexts increases. In contrast, when using the episodic bonus, performance remains high as the number of contexts increases, even when $\lvert\mathcal{C}\rvert = \infty$ (no two maps are repeated during training). We observed similar trends on two other MiniHack tasks (see Appendix G.1). In section 3.3, we provide a framework which explains why the global bonus fails and the episodic bonus is preferable here.

3.2. Advantages of Global Bonuses

We next provide an example where the episodic bonus fails but the global bonus succeeds. Consider a singleton MDP with M corridors which can be crossed in T steps, with a single one containing reward at the end (shown in Figure 2). If the episode length is T, then any policy which reaches the end of any of the M corridors will get equivalent episodic bonus, and hence the chance of success will be $1/M$. On the other hand, a global bonus will solve the task: after sufficiently visiting one of the corridors, the global bonus there will become depleted and the agent will move on to another one, eventually visiting the corridor with the reward.

We illustrate this argument using a singleton version of the `MiniHack-Corridors-R5` environment (shown in Figure 3), where the agent must explore different corridors to find its way to the exit. This is similar to the example in Figure 2 in the sense that the agent will likely need to explore multiple dead ends before finding the goal. Table 1 (middle section) shows results for agents trained with the

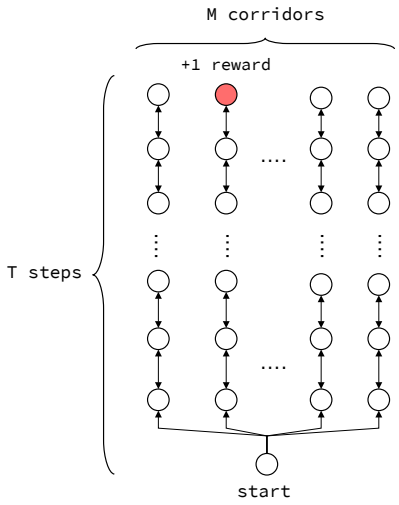


Figure 2. Simple example where episodic bonus fails.

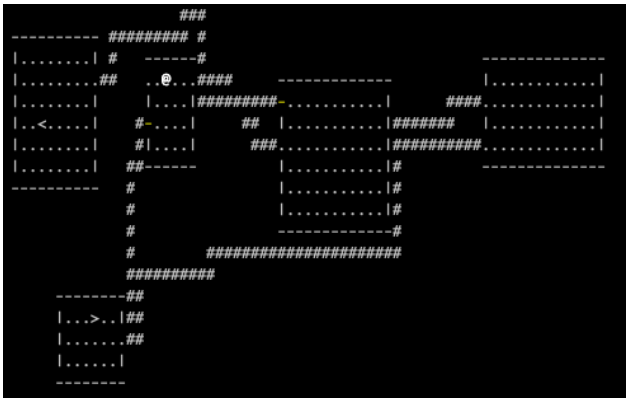


Figure 3. Example map for `MiniHack-Corridors-R5` enviroment. `@` indicates agent, `#` corridors connecting rooms, `<` start location and `>` goal.

episodic and global bonus on the `Corridors` environment. In contrast to the previous example, but consistent with our argument above, the global bonus succeeds across all seeds whereas the episodic bonus produces inconsistent performance across seeds, leading to poor performance overall.

Are global bonuses only useful in the special case of singleton MDPs? We next show that this is not the case, and that they can also be useful in general CMDPs with large $|\mathcal{C}|$. We consider the `KeyRoom` environment, illustrated in Figure 4. In this environment, the agent must pick up a key and use it to open a door to a small room to reach the exit. Here different contexts correspond to different placements of the agent, key, room, door and exit. We define the ψ feature extractor in equation 6 to extract the message rather than the (x, y) coordinates (using coordinates does not solve the task for either bonus). We also evaluate both global and episodic bonuses on the `MultiRoom` environment where

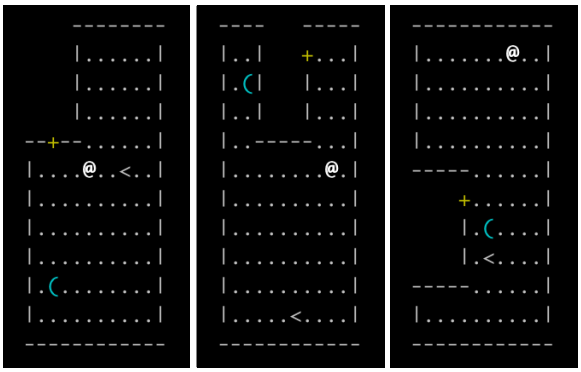


Figure 4. Three example maps for `MiniHack-KeyRoom-S10` enviroment. `@` indicates agent, `(` key, `+` door, `<` start location and `>` goal. Each observation additionally includes a message such as "You see here a key of Master Thievery" or "It's a wall".

ψ extracts messages rather than positions. Results in Table 1 (bottom section) show that the global bonus solves both environments, even though $|\mathcal{C}| = \infty$ in both cases.

3.3. A Framework for Understanding Global and Episodic Novelty Bonuses

We now develop a framework for better understanding when and why global and episodic bonuses are effective, and which explains our results so far. Let $\psi : \mathcal{S} \to Z$ be a feature extractor mapping states to a space Z where novelty bonuses will be computed. This mapping could be hard-coded (as in the examples above) or learned, and states in $\mathcal{S}$ could potentially be high dimensional. In Appendix H.1, we show how a number of existing exploration algorithms can be instantiated using this framework, including tabular count-based algorithms, deep RL algorithms that use position or message counts, kernel-based algorithms with global elliptical bonuses, as well as the E3B algorithm which uses a learned ψ and an episodic bonus. Now consider the function $V^\star_{\psi,c} : Z \to [R_{min}, R_{max}]$, defined by:

$$V^\star_{\psi,c}(z) = \inf_{s \in \psi^{-1}(z)} V^\star(s) \tag{7}$$

Here $V^\star$ denotes the optimal value function, R_{min} and R_{max} denote the minimum and maximum possible return, and $\psi^{-1}(z) = \{s \in \mathcal{S}_c : \psi(s) = z\}$, where $\mathcal{S}_c$ is the set of states reachable by the agent in context c. The function $V^\star_{\psi,c}$ can be thought of as a value function over Z corresponding to context c. Note that the infimum ensures that high-value regions in Z correspond to high-value regions in $\mathcal{S}$. We additionally assume that ψ is defined such that there is some subset of Z for which $V^\star_{\psi,c}(z) \approx R_{max}$. This assumption is necessary to rule out pathological cases such as ψ mapping every state to the same point, and holds for all the examples we consider here.

Example 1: Consider the `MultiRoom` environment in Figure 1 with positional encodings, i.e. $\psi(s) = (x, y)$, where (x, y) is the location of the agent. Then Z is a 2D lattice the size of the map. The value function $V^\star_{\psi,c}$ will be high centered at the goal and propagate outwards. Note that since the goal changes location for each map, $V^\star_{\psi,c}$ varies significantly across contexts. We verify this empirically and provide visualizations in Figure 10 of Appendix H.2.

Example 2: Consider the `KeyRoom` environment shown in Figure 4 with message encodings, i.e. $\psi(s)$ returns the message associated with state s. Here Z is the set of all possible messages. The function $V^\star_{\psi,c}(z)$ will then be high for messages indicating that the door has been opened or that the agent has found the key (such as "You see here a Key of Master Thievery") and low for other messages ("the door is locked"), regardless of the context. Therefore, $V^\star_{\psi,c}$ varies little across contexts. See Figure 11 in Appendix H.2 for visualizations.

Example 3: Consider the `MultiRoom` environment with message encodings. Here Z is the set of all possible messages, and $V^\star_{\psi,c}$ will be high for messages indicating that doors have been opened (such as "The door opens!"), since this indicates the agent has moved to a new room and is thus closer to the goal. Conversely, $V^\star_{\psi,c}$ will be low for other messages (like the blank message " ") which do not indicate progress towards the goal. As in the previous example, which messages have high or low values of $V^\star_{\psi,c}$ will not depend much on the context c, hence $V^\star_{\psi,c}$ changes little across contexts. See Figure 12 in Appendix H.2 for visualizations.

Example 4: Consider any singleton MDP, such as the `Corridors` example from the previous section. Trivially, since the context is always identical, $V^\star_{\psi,c}$ does not change across contexts regardless of ψ.

We now argue that global bonuses will fail when $V^\star_{\psi,c}$ changes significantly across different contexts, and succeed when it changes little. To see this, note that a global bonus will induce a sequence of policies $\pi_1, \pi_2, \pi_3, ...$ which progressively visit different parts of the Z space. If $V^\star_{\psi,c}$ varies little across contexts c, then eventually some policy π_j will visit a part of the Z space which has high value across *all* contexts c. Since high value regions in Z correspond to high value regions in $\mathcal{S}$, this means the policy obtains high return across all contexts. On the other hand, if $V^\star_{\psi,c}$ varies significantly across contexts, it is more likely that a part of the Z space which was previously visited by policy π_i will have high value for some context c which is sampled later on during training. In this case, the agent will no longer visit this region since the global bonus has been exhausted there, thus missing a high-value region in $\mathcal{S}$ as well.

In contrast to the global bonus, the episodic bonus favors policies which try to cover the *entire* Z space *within each episode*. This is a harder task, and may in fact be impossible if the time horizon is short (see the counterexample in Figure 2). However, if the agent is able to cover the entire Z space within each episode, then they will always visit high-value regions in Z (and thus in $\mathcal{S}$), even if these regions change from one episode to the next—thus avoiding the limitation of the global bonus described above.

This framework provides a consistent explanation for our results so far: recall that the global bonuses succeed in examples 2, 3, 4 (where $V^\star_{\psi,c}$ varies little) and fail in example 1, where $V^\star_{\psi,c}$ varies a lot and the episodic bonus succeeds. Note that since $V^\star$ and ψ both appear in the definition of $V^\star_{\psi,c}$ in equation 7, the relative advantage of the global vs. episodic bonuses will depend both on the structure of the CMDP, *and* the feature extractor ψ used to compute the novelty bonus. This framework may also serve to guide practitioners: if sufficient knowledge of the CMDP and ψ is available to estimate how much the $V^\star_{\psi,c}$ function will vary across contexts, this can inform whether to use the global bonus (if it varies little) or the episodic bonus (if it varies a lot). In Appendix H.3 we further discuss how the variation of $V^\star_{\psi,c}$ across episodes can be made precise, and illustrate how it relates to the performance of the global bonus empirically.

3.4. Combining Global and Episodic Bonuses

Our framework described in Section 3.3 can provide guidance regarding which bonus to use, when knowledge of the CMDP and feature extractor are available. However, for complex CMDPs or learned feature extractors, it may be difficult to predict how much the $V^\star_{\psi,c}$ function will change across contexts. This motivates the investigation of bonuses which perform robustly across a wide range of CMDPs with differing degrees of shared structure.

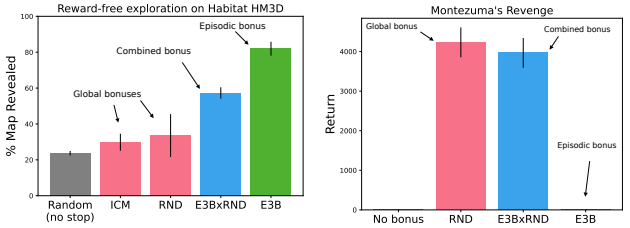
We next investigate a simple strategy whereby we combine global and episodic bonuses via multiplication, which we hypothesize would lead to more robust performance across different regimes compared to either bonus alone. The resulting combined bonus is given by:

$$b_{\text{combined}}(s_t) = \mathbb{I}[N_e(\psi(s_t)) = 1] \cdot \frac{1}{\sqrt{N(\psi(s_t))}} \quad (8)$$

This is motivated by the following observations. First, let us consider the MDP in Figure 2: note that following any of the corridors will maximize the episodic bonus by providing an episodic bonus of 1 at each step. The total combined bonus in equation 8 is then equal to the global bonus, and optimizing the global bonus causes the agent to visit each of the corridors until it reaches the one with the reward, solving the MDP.

Environment	$\mathcal{C}$	ψ	Combined Bonus
`MultiRoom`	∞	P	0.83 ± 0.04
`Corridors`	1	P	0.91 ± 0.01
`KeyRoom`	∞	M	0.99 ± 0.00
`MultiRoom`	∞	M	0.97 ± 0.00

Table 2. Performance of combined bonus, averaged across 5 seeds. $|\mathcal{C}|$ denotes number of contexts, P position encodings and M message encodings for ψ.



(a) Habitat, $|\mathcal{C}| = 1000$ (b) Montezuma, $|\mathcal{C}| = 1$

Figure 5. Comparison of global, episodic and combined bonuses on Habitat and Montezuma's Revenge. Errors bars correspond to standard deviation across 3 seeds.

Now let us consider the `MultiRoom` environment with position encodings. If the agent is initialized roughly uniformly throughout the map, the global bonus will decay roughly uniformly across regions over time. This means that the bonus in equation 8 will be roughly equal to the episodic bonus (scaled by a constant), which we know is effective. Finally, in `KeyRoom` both the episodic and global bonuses will assign high novelty to messages associated with picking up the key, which aligns with the optimal policy, suggesting that their product will also be effective.

Results for all environments are shown in Table 2 (we use the same ψ feature extractor as in previous experiments for each environment). The combined bonus obtains good performance on all environments, suggesting that it retains the advantages of both the global and episodic bonus here.

3.5. Scaling to Pixel-Based Settings

We next test whether the tradeoffs we have observed between global and episodic bonuses, as well as the advantages of the combined bonus, also apply in high-dimensional, pixel-based settings. As an example of a pixel-based CMDP with little shared structure across environments, we use Habitat (Savva et al., 2019), a photorealistic simulator of indoor environments. Habitat is conceptually similar to the `MultiRoom` environment in the sense that at each episode, the agent finds itself in a different indoor space consisting of connected rooms. However, the maps in Habitat are considerably more complex and the observations are pixel-based. Here we compare global and episodic bonuses based on function approximation, since counts are not meaningful with high-dimensional images. We use the reward-free exploration setup described in Henaff et al. (2022), with results shown in Figure 5(a). We see that, similarly to `MultiRoom` with position encodings, the global bonuses (ICM and RND) perform poorly whereas the episodic bonus (E3B) performs well. See Appendix E.4 for experiment details.

We perform a second set of experiments using the Atari environment Montezuma's Revenge (Bellemare et al., 2012), a notoriously difficult hard exploration game. This is a singleton MDP where structure is completely shared across contexts, hence our previous results suggest that global bonuses are preferable to episodic ones here. We again compare E3B to RND, with results shown in Figure 5(b). Consistent with our previous results, we see that the global bonus (RND) performs well, whereas the episodic bonus (E3B) performs poorly. This provides evidence that the tradeoffs we have identified apply more broadly. See Appendix E.5 for experiment details.

For both environments, we also tested a combined bonus obtained by multiplying the episodic bonus from E3B with the global bonus from RND. This approach nearly matches RND's performance on Montezuma's Revenge, and improves upon the global bonus' performance on Habitat, although it does not match the performance of the episodic bonus. The combined bonus thus provides more robust performance across scenarios here than an episodic or global bonus alone, although it does not always match the optimal bonus on each task.

4. Design Choices for Episodic and Global Novelty Bonuses

The previous section has shown that global and episodic bonuses succeed in different types of CMDPs, and that combining them via multiplication can yield a bonus which is more robustly effective across tasks. However, in order to facilitate interpretability we used count-based bonuses for our MiniHack experiments, which do not scale to complex environments unless task-specific prior knowledge is used (e.g. knowing how to extract (x, y) positions or messages). In this section, we perform a thorough a study of global and episodic bonus designs based on function approximation, which do not require such prior knowledge, across a wide range of tasks from the MiniHack suite (Samvelyan et al., 2021).

4.1. Experimental Setup

As our experimental testbed, we use 16 procedurally-generated tasks from the MiniHack suite (Samvelyan et al., 2021) used in prior work (Henaff et al., 2022). The MiniHack tasks are designed to precisely evaluate different ca-

pabilities of a given agent, such as navigation, planning or the ability to use objects. Furthermore, many of the MiniHack tasks involve sparse rewards and complex observations which include irrelevant information. For evaluation, we follow the protocol suggested by (Agarwal et al., 2021) and report the mean, median and interquartile mean (IQM) together with 95% confidence intervals using stratified bootstrapping. We use 5 random seeds for each of the 16 tasks. Our full experimental details can be found in Appendix E.1.

4.2. Results

We now investigate combining different global novelty bonuses from AGAC, RND and NovelD with the elliptical episodic bonus. We use E3B's elliptical bonus as our episodic bonus instead of a count-based one, since prior work has shown that count-based bonuses either fail in complex environments, or are highly dependent on task-specific feature extractors (Henaff et al., 2022). In contrast, the elliptical bonus has been shown to work well across a wide range of environments without requiring task-specific prior knowledge. We also experimented with the KNN-based episodic bonus from the NGU agent (Badia et al., 2020), but found that it worked poorly (see Appendix G.3 for more details).

Two questions we aim to answer are: i) which global bonus (if any) gives the most improvements when combined with E3B's episodic bonus, and ii) which strategy is best for combining the two bonuses. To answer this, we consider all possible combinations of E3B's episodic bonus with the global bonuses from RND, NovelD and AGAC, combined either via addition or multiplication. The exact bonuses for each algorithm are detailed in Appendix F. We compare to E3B as a baseline since it was previously shown to outperform other methods such as IMPALA, RND, ICM, RIDE and NovelD (Henaff et al., 2022).

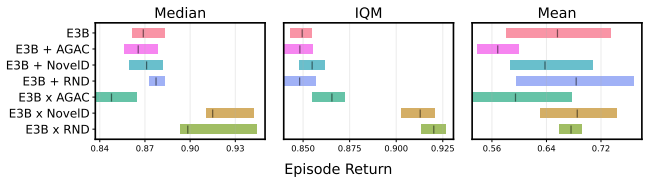


Figure 6. Aggregate performance on 16 MiniHack tasks. Bars indicate 95% confidence intervals computed using stratified bootstrapping over 5 seeds.

Results are shown in Figure 6. First, we see that additively combining any of the global bonuses with the elliptical episodic bonus does not provide a meaningful improvement over E3B for any metric. However, multiplicatively combining E3B with either RND or NovelD bonuses produces a large and statistically significant improvement in both median and IQM performance over E3B (the more robust metrics according to (Agarwal et al., 2021)). This establishes a new state-of-the-art on MiniHack.

One explanation for the superior performance of the multiplicative combination over the additive one is that the scale of the global bonus decreases significantly throughout training whereas the scale of the episodic bonus does not, since it is reset each episode. Because of this, if we combine the two bonuses via addition, the combined bonus will become increasingly dominated by the episodic bonus. However, if we are combining the two multiplicatively, the global bonus will still have an effect regardless of its scale. See Appendix G.2 for additional results and discussion.

5. Related Work

Exploration in singleton MDPs is a well-studied problem in RL (Sutton & Barto, 2018; Schmidhuber, 1991; Oudeyer et al., 2007; Oudeyer & Kaplan, 2009). Many theoretical works exist which propose provably efficient algorithms for tabular or linear MDPs (Kearns & Singh, 2002; Brafman & Tennenholtz, 2002; Strehl & Littman, 2006; Jin et al., 2020; Cai et al., 2020; Agarwal et al., 2020; Kolter & Ng, 2009; Fruit & Lazaric, 2017; Fruit et al., 2018a;b; Tarbouriech et al., 2020). A number of methods which combine deep RL agents with exploration bonuses have also been proposed for general MDPs (Stadie et al., 2015; Achiam & Sastry, 2017). These include model-free methods such as RND (Burda et al., 2019b), ICM (Pathak et al., 2017a) and pseudocounts (Bellemare et al., 2016b; Strehl & Littman, 2008; Bellemare et al., 2016a; Ostrovski et al., 2017; Martin et al., 2017; Tang et al., 2017; Machado et al., 2020), as well as model-based approaches (Shyam et al., 2019; Henaff, 2019; Sekar et al., 2020; Zhang et al., 2021a; Manek & Kolter, 2021). However, these are all designed for the singleton MDP setting and use some form of global bonus which, as we show in Section 3, is not always appropriate to the more general CMDP setting we consider here. We also note the work of (Stanton & Clune, 2018), which showed that episodic bonuses can aid exploration in singleton MDPs.

More recently, RIDE (Raileanu & Rocktäschel, 2020), AGAC (Flet-Berliac et al., 2021), NovelD (Zhang et al., 2021b), its variants (Mu et al., 2022b), and others (Parisi et al., 2021; Campero et al., 2020; Zhang et al., 2021a; Seurin et al., 2021; Fickinger et al., 2021; Tam et al., 2022; Jo et al., 2022; Ramesh et al., 2022) have begun to tackle exploration in procedurally-generated MDPs, a type of CMDP commonly used in empirical research. These methods use combinations of global bonuses designed for singleton MDPs and count-based episodic bonuses. The recent works of Henaff et al. (2022) and Wang et al. (2023) highlighted the practical importance of episodic bonuses, with Henaff et al. (2022) proposing the elliptical episodic bonus as a solution to the limitations of count-based episodic bonuses,

but they did not include a global bonus. Compared to these prior works, our work makes two main contributions. First, whereas previous works justified using global bonuses in CMDPs by appealing to intuitions from singleton MDPs, and provided little justification for using episodic bonuses aside from their empirical performance, we provide clear justifications for the use of each bonus in different types of CMDPs and identify tradeoffs. In particular, we experimentally examine the behavior of each bonus type across different representative settings, and provide a new framework for understanding each one's effect on exploration. This may additionally guide practitioners in choosing an appropriate bonus for the problem at hand. Second, whereas previous works have investigated different combinations of global and episodic bonuses in isolation, there has not been a systematic comparison of bonuses and combination strategies, which we perform in Section 4. This investigation results in a new algorithmic combination which outperforms the previously proposed ones.

6. Conclusion

In this work, we have shed light on the tradeoffs between global and episodic exploration bonuses in CMDPs through experiments in both easily interpretable gridworlds and challenging pixel-based settings, and by developing a new framework which provides a unifying explanation of our empirical results. In particular, we find that the effectiveness of each bonus depends on the degree of shared structure between value functions in feature space across different contexts. Episodic bonuses tend to be more effective when there is little shared structure across contexts, whereas global bonuses tend to succeed when more structure is shared. We further show that multiplicatively combining global and episodic bonuses can lead to more robust performance across different settings than either bonus alone. Finally, we perform a thorough investigation of design choices for global and episodic bonuses, which leads to an algorithm that sets a new state of the art on a wide range of tasks from the MiniHack suite. This work opens up several avenues for future research. Formally quantifying the tradeoffs between global and episodic bonuses through sample complexity bounds presents itself as an intriguing theoretical question. Another promising direction is to investigate algorithms which more effectively combine the different bonus types. While our multiplicative bonus provides a first step in this direction, it is still limited in the sense that it does not always match the performance of the best bonus type on each individual task. More broadly, we hypothesize that agents in rich and ever-changing environments such as NetHack and Minecraft will require both local and global exploration, in order to acquire information at different timescales—how to best achieve this remains an open question.

References

Achiam, J. and Sastry, S. Surprise-based intrinsic motivation for deep reinforcement learning. *arXiv preprint arXiv:1703.01732*, 2017.

Agarwal, A., Henaff, M., Kakade, S., and Sun, W. Pc-pg: Policy cover directed exploration for provable policy gradient learning. *Advances in neural information processing systems*, (33), 2020.

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A., and Bellemare, M. G. Deep reinforcement learning at the edge of the statistical precipice. *Advances in Neural Information Processing Systems*, 2021.

Badia, A. P., Sprechmann, P., Vitvitskyi, A., Guo, Z. D., Piot, B., Kapturowski, S., Tieleman, O., Arjovsky, M., Pritzel, A., Bolt, A., and Blundell, C. Never give up: Learning directed exploration strategies. *ICLR*, 2020.

Beattie, C., Leibo, J. Z., Teplyashin, D., Ward, T., Wainwright, M., Küttler, H., Lefrancq, A., Green, S., Valdés, V., Sadik, A., et al. Deepmind lab. *arXiv preprint arXiv:1612.03801*, 2016.

Bellemare, M., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47, 07 2012. doi: 10.1613/jair.3912.

Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016a.

Bellemare, M. G., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. Unifying count-based exploration and intrinsic motivation. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pp. 1479–1487, Red Hook, NY, USA, 2016b. Curran Associates Inc. ISBN 9781510838819.

Brafman, R. I. and Tennenholtz, M. R-MAX - A general polynomial time algorithm for near-optimal reinforcement learning. *J. Mach. Learn. Res.*, 3:213–231, 2002.

Burda, Y., Edwards, H., Pathak, D., Storkey, A., Darrell, T., and Efros, A. A. Large-scale study of curiosity-driven learning. In *ICLR*, 2019a.

Burda, Y., Edwards, H., Storkey, A., and Klimov, O. Exploration by random network distillation. In *International Conference on Learning Representations*, 2019b.

Cai, Q., Yang, Z., Jin, C., and Wang, Z. Provably efficient exploration in policy optimization. In III, H. D. and Singh,

A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1283–1294. PMLR, 13–18 Jul 2020.

Campero, A., Raileanu, R., Küttler, H., Tenenbaum, J. B., Rocktäschel, T., and Grefenstette, E. Learning with amigo: Adversarially motivated intrinsic goals. *arXiv preprint arXiv:2006.12122*, 2020.

Chevalier-Boisvert, M., Willems, L., and Pal, S. Minimalistic gridworld environment for openai gym. https://github.com/maximecb/gym-minigrid, 2018.

Clevert, D., Unterthiner, T., and Hochreiter, S. Fast and accurate deep network learning by exponential linear units (elus). In Bengio, Y. and LeCun, Y. (eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

Cobbe, K., Klimov, O., Hesse, C., Kim, T., and Schulman, J. Quantifying generalization in reinforcement learning. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1282–1289. PMLR, 09–15 Jun 2019.

Cobbe, K., Hesse, C., Hilton, J., and Schulman, J. Leveraging procedural generation to benchmark reinforcement learning. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 2048–2056. PMLR, 13–18 Jul 2020.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., Legg, S., and Kavukcuoglu, K. IMPALA: scalable distributed deep-rl with importance weighted actor-learner architectures. In Dy, J. G. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1406–1415. PMLR, 2018.

Farebrother, J., Machado, M. C., and Bowling, M. Generalization and regularization in DQN. *CoRR*, abs/1810.00123, 2018.

Fickinger, A., Jaques, N., Parajuli, S., Chang, M., Rhinehart, N., Berseth, G., Russell, S., and Levine, S. Explore and control with adversarial surprise. *arXiv preprint arXiv:2107.07394*, 2021.

Flet-Berliac, Y., Ferret, J., Pietquin, O., Preux, P., and Geist, M. Adversarially guided actor-critic. *CoRR*, abs/2102.04376, 2021.

Fruit, R. and Lazaric, A. Exploration-exploitation in mdps with options. In *Artificial intelligence and statistics*, pp. 576–584. PMLR, 2017.

Fruit, R., Pirotta, M., and Lazaric, A. Near optimal exploration-exploitation in non-communicating markov decision processes. *Advances in Neural Information Processing Systems*, 31, 2018a.

Fruit, R., Pirotta, M., Lazaric, A., and Ortner, R. Efficient bias-span-constrained exploration-exploitation in reinforcement learning. In *International Conference on Machine Learning*, pp. 1578–1586. PMLR, 2018b.

Gan, C., Schwartz, J., Alter, S., Schrimpf, M., Traer, J., De Freitas, J., Kubilius, J., Bhandwaldar, A., Haber, N., Sano, M., et al. Threedworld: A platform for interactive multi-modal physical simulation. *arXiv preprint arXiv:2007.04954*, 2020.

Hafner, D. Benchmarking the spectrum of agent capabilities. *arXiv preprint arXiv:2109.06780*, 2021.

Hallak, A., Di Castro, D., and Mannor, S. Contextual markov decision processes. *arXiv preprint arXiv:1502.02259*, 2015.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

Henaff, M. Explicit explore-exploit algorithms in continuous state spaces. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Henaff, M., Raileanu, R., Jiang, M., and Rocktäschel, T. Exploration via elliptical episodic bonuses. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Jin, C., Allen-Zhu, Z., Bubeck, S., and Jordan, M. I. Is q-learning provably efficient? In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Jin, C., Yang, Z., Wang, Z., and Jordan, M. I. Provably efficient reinforcement learning with linear function approximation. In Abernethy, J. and Agarwal, S. (eds.), *Proceedings of Thirty Third Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pp. 2137–2143. PMLR, 09–12 Jul 2020.

Jo, D., Kim, S., Nam, D. W., Kwon, T., Rho, S., Kim, J., and Lee, D. Leco: Learnable episodic count for task-specific intrinsic reward. *arXiv preprint arXiv:2210.05409*, 2022.

Juliani, A., Khalifa, A., Berges, V., Harper, J., Henry, H., Crespi, A., Togelius, J., and Lange, D. Obstacle tower: A generalization challenge in vision, control, and planning. *CoRR*, abs/1902.01378, 2019.

Justesen, N., Torrado, R. R., Bontrager, P., Khalifa, A., Togelius, J., and Risi, S. Procedural level generation improves generality of deep reinforcement learning. *CoRR*, abs/1806.10729, 2018.

Kearns, M. and Singh, S. Near-optimal reinforcement learning in polynomial time. In *Machine Learning*, pp. 209–232. Morgan Kaufmann, 2002.

Kirk, R., Zhang, A., Grefenstette, E., and Rocktäschel, T. A survey of generalisation in deep reinforcement learning. *CoRR*, abs/2111.09794, 2021a.

Kirk, R., Zhang, A., Grefenstette, E., and Rocktäschel, T. A survey of generalisation in deep reinforcement learning. *CoRR*, abs/2111.09794, 2021b.

Kolter, J. Z. and Ng, A. Y. Near-bayesian exploration in polynomial time. In *Proceedings of the 26th annual international conference on machine learning*, pp. 513–520, 2009.

Küttler, H., Nardelli, N., Miller, A. H., Raileanu, R., Selvatici, M., Grefenstette, E., and Rocktäschel, T. The nethack learning environment. *CoRR*, abs/2006.13760, 2020.

Machado, M. C., Bellemare, M. G., and Bowling, M. Count-based exploration with the successor representation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5125–5133, 2020.

Manek, G. and Kolter, J. Z. Model-based reinforcement learning with ensembled model-value expansion. 2021.

Martin, J., Sasikumar, S. N., Everitt, T., and Hutter, M. Count-based exploration in feature space for reinforcement learning. *arXiv preprint arXiv:1706.08090*, 2017.

Mu, J., Zhong, V., Raileanu, R., Jiang, M., Goodman, N., Rocktäschel, T., and Grefenstette, E. Improving intrinsic exploration with language abstractions. *arXiv preprint arXiv:2202.08938*, 2022a.

Mu, J., Zhong, V., Raileanu, R., Jiang, M., Goodman, N. D., Rocktäschel, T., and Grefenstette, E. Improving intrinsic exploration with language abstractions. *CoRR*, abs/2202.08938, 2022b.

Ostrovski, G., Bellemare, M. G., Oord, A., and Munos, R. Count-based exploration with neural density models. In *International conference on machine learning*, pp. 2721–2730. PMLR, 2017.

Oudeyer, P.-Y. and Kaplan, F. What is intrinsic motivation? a typology of computational approaches. *Frontiers in neurorobotics*, 1:6, 2009.

Oudeyer, P.-Y., Kaplan, F., and Hafner, V. V. Intrinsic motivation systems for autonomous mental development. *IEEE transactions on evolutionary computation*, 11(2): 265–286, 2007.

Packer, C., Gao, K., Kos, J., Krähenbühl, P., Koltun, V., and Song, D. Assessing generalization in deep reinforcement learning. *CoRR*, abs/1810.12282, 2018.

Parisi, S., Dean, V., Pathak, D., and Gupta, A. Interesting object, curious agent: Learning task-agnostic exploration. *Advances in Neural Information Processing Systems*, 34: 20516–20530, 2021.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. *CoRR*, abs/1705.05363, 2017a.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pp. 2778–2787. PMLR, 2017b.

Petrenko, A., Wijmans, E., Shacklett, B., and Koltun, V. Megaverse: Simulating embodied agents at one million experiences per second. In *International Conference on Machine Learning*, pp. 8556–8566. PMLR, 2021.

Raileanu, R. and Rocktäschel, T. Ride: Rewarding impact-driven exploration for procedurally-generated environments. In *International Conference on Learning Representations*, 2020.

Ramakrishnan, S. K., Gokaslan, A., Wijmans, E., Maksymets, O., Clegg, A., Turner, J. M., Undersander, E., Galuba, W., Westbury, A., Chang, A. X., Savva, M., Zhao, Y., and Batra, D. Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

Ramesh, A., Kirsch, L., van Steenkiste, S., and Schmidhuber, J. Exploring through random curiosity with general value functions. *arXiv preprint arXiv:2211.10282*, 2022.

Samvelyan, M., Kirk, R., Kurin, V., Parker-Holder, J., Jiang, M., Hambro, E., Petroni, F., Küttler, H., Grefenstette, E., and Rocktäschel, T. Minihack the planet: A sandbox

for open-ended reinforcement learning research. *CoRR*, abs/2109.13202, 2021.

Savva, M., Kadian, A., Maksymets, O., Zhao, Y., Wijmans, E., Jain, B., Straub, J., Liu, J., Koltun, V., Malik, J., Parikh, D., and Batra, D. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Schmidhuber, J. A possibility for implementing curiosity and boredom in model-building neural controllers. In *Proc. of the international conference on simulation of adaptive behavior: From animals to animats*, pp. 222–227, 1991.

Sekar, R., Rybkin, O., Daniilidis, K., Abbeel, P., Hafner, D., and Pathak, D. Planning to explore via self-supervised world models. In *ICML*, 2020.

Seurin, M., Strub, F., Preux, P., and Pietquin, O. Don't do what doesn't matter: Intrinsic motivation with action usefulness. *arXiv preprint arXiv:2105.09992*, 2021.

Shen, B., Xia, F., Li, C., Martín-Martín, R., Fan, L., Wang, G., Pérez-D'Arpino, C., Buch, S., Srivastava, S., Tchapmi, L., et al. igibson 1.0: A simulation environment for interactive tasks in large realistic scenes. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7520–7527. IEEE, 2020.

Shyam, P., Jaśkowski, W., and Gomez, F. Model-based active exploration. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5779–5788. PMLR, 09–15 Jun 2019.

Song, X., Jiang, Y., Tu, S., Du, Y., and Neyshabur, B. Observational overfitting in reinforcement learning. In *International Conference on Learning Representations*, 2020.

Stadie, B. C., Levine, S., and Abbeel, P. Incentivizing exploration in reinforcement learning with deep predictive models. *arXiv preprint arXiv:1507.00814*, 2015.

Stanton, C. and Clune, J. Deep curiosity search: Intra-life exploration improves performance on challenging deep reinforcement learning problems. *CoRR*, abs/1806.00553, 2018.

Strehl, A. L. and Littman, M. L. An analysis of model-based interval estimation for markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

Strehl, E. L. and Littman, M. L. An analysis of model-based interval estimation for markov decision processes. *Journal of Computer and System Sciences*, 2006.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Szot, A., Clegg, A., Undersander, E., Wijmans, E., Zhao, Y., Turner, J., Maestre, N., Mukadam, M., Chaplot, D., Maksymets, O., Gokaslan, A., Vondrus, V., Dharur, S., Meier, F., Galuba, W., Chang, A., Kira, Z., Koltun, V., Malik, J., Savva, M., and Batra, D. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Tam, A. C., Rabinowitz, N. C., Lampinen, A. K., Roy, N. A., Chan, S. C., Strouse, D., Wang, J. X., Banino, A., and Hill, F. Semantic exploration from language abstractions and pretrained representations. *arXiv preprint arXiv:2204.05080*, 2022.

Tang, H., Houthooft, R., Foote, D., Stooke, A., Xi Chen, O., Duan, Y., Schulman, J., DeTurck, F., and Abbeel, P. # exploration: A study of count-based exploration for deep reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

Tarbouriech, J., Garcelon, E., Valko, M., Pirotta, M., and Lazaric, A. No-regret exploration in goal-oriented reinforcement learning. In *International Conference on Machine Learning*, pp. 9428–9437. PMLR, 2020.

Wang, K., Zhou, K., Kang, B., Feng, J., and YAN, S. Revisiting intrinsic reward for exploration in procedurally generated environments. In *The Eleventh International Conference on Learning Representations*, 2023.

Wijmans, E., Kadian, A., Morcos, A. S., Lee, S., Essa, I., Parikh, D., Savva, M., and Batra, D. Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. In *ICLR*, 2020.

Xiang, F., Qin, Y., Mo, K., Xia, Y., Zhu, H., Liu, F., Liu, M., Jiang, H., Yuan, Y., Wang, H., et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11097–11107, 2020.

Zhang, A., Ballas, N., and Pineau, J. A dissection of overfitting and generalization in continuous reinforcement learning. *arXiv preprint arXiv:1806.07937*, 2018a.

Zhang, C., Vinyals, O., Munos, R., and Bengio, S. A study on overfitting in deep reinforcement learning. *CoRR*, abs/1804.06893, 2018b.

Zhang, T., Rashidinejad, P., Jiao, J., Tian, Y., Gonzalez, J. E., and Russell, S. Made: Exploration via maximizing deviation from explored regions. *Advances in Neural Information Processing Systems*, 34:9663–9680, 2021a.

Zhang, T., Xu, H., Wang, X., Wu, Y., Keutzer, K., Gonzalez, J. E., and Tian, Y. Noveld: A simple yet effective exploration criterion. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021b.

A. Broader Impact Statement

This work makes progress towards better understanding and designing methods which can efficiently explore contextual MDPs, a very broad framework with applications in video games, virtual reality, autonomous driving, robotics and healthcare. Efficient exploration typically reduces sample complexity, which make real-world application more feasible. Like any RL algorithm, our approach aims to facilitate discovering a policy that maximizes some user-specified reward. Depending on the reward function, executing such a policy could have positive or negative consequences.

B. Limitations

The main technical limitations of this work are twofold. First, although the multiplicative bonus is more robust than the global or episodic bonus in terms of aggregate performance across tasks, it does not always match the best-performing bonus on individual tasks. This is evidenced by our results on Habitat, where the combined bonus still performs worse than the episodic one. We view the multiplicative bonus as a first step towards a method that works across all regimes, but not as a definitive solution. Furthermore, we only consider simple combination strategies for the two bonuses like addition/multiplication, and we do not consider adaptively combining the two based on environment interaction, which is left for future work. Concerning potential negative societal impacts, exploration methods in general can potentially cause harm if deployed in the real world without appropriate safety measures since they seek out novel states, possibly leading to unpredicted behavior.

C. Reproducibility Statement

Our code can be found at: https://github.com/facebookresearch/e3b. Experiment details can be found in Appendix E.

D. Additional Related Work

Exploration in singleton MDPs is a well-studied problem in RL (Sutton & Barto, 2018; Schmidhuber, 1991; Oudeyer et al., 2007; Oudeyer & Kaplan, 2009). Many theoretical works exist which propose provably efficient algorithms for tabular or linear MDPs (Kearns & Singh, 2002; Brafman & Tennenholtz, 2002; Strehl & Littman, 2006; Jin et al., 2020; Cai et al., 2020; Agarwal et al., 2020; Kolter & Ng, 2009; Fruit & Lazaric, 2017; Fruit et al., 2018a;b; Tarbouriech et al., 2020). A number of methods which combine deep RL agents with exploration bonuses have also been proposed for general MDPs (Stadie et al., 2015; Achiam & Sastry, 2017). These include model-free methods such as RND (Burda et al., 2019b), ICM (Pathak et al., 2017a) and pseudocounts (Bellemare et al., 2016b; Strehl & Littman, 2008; Bellemare et al., 2016a; Ostrovski et al., 2017; Martin et al., 2017; Tang et al., 2017; Machado et al., 2020), as well as model-based approaches (Shyam et al., 2019; Henaff, 2019; Sekar et al., 2020; Zhang et al., 2021a; Manek & Kolter, 2021). However, these are all designed for the singleton MDP setting and use some form of global bonus which, as we show in Section 3, is not always appropriate to the more general CMDP setting we consider here. We also note the work of (Stanton & Clune, 2018) which used episodic bonuses for singleton MDPs.

E. Experiment Details

E.1. MiniHack

We used the same architectures and hyperparameters for the experiments with count-based bonuses in Section 3 and with function approximation in Section 4.

E.1.1. Architecture Details

We follow the policy network architecture described in (Samvelyan et al., 2021). The policy network has four trunks: i) a 5-layer convolutional trunk which maps the full symbol image (of size 79×21) to a hidden representation, ii) a second 5-layer convolutional trunk which maps a 9×9 crop centered at the agent to a hidden representation, iii) an MLP trunk which maps the stats vector to a hidden representation, and iv) a 1-D convolutional trunk with interleaved max-pooling layers, followed by a fully-connected network which maps the message to a hidden representation. The hidden representations are then concatenated together, passed through a 2-layer fully-connected network followed by an LSTM (Hochreiter &

Schmidhuber, 1997) layer. The output of the LSTM layer is then passed to linear layers which produce action probabilities and a value function estimate.

The convolutional trunks i) and ii) have the following hyperparameters: 5 layers, filter size 3, symbol embedding dimension 64, stride 1, filter number 16 at each layer except the last, which is 8, and ELU non-linearities (Clevert et al., 2016). The MLP trunk iii) has 2 hidden layers of 64 hidden units each with ReLU non-linearities. The trunk iv) for processing messages has 6 convolutional layers, each with 64 input and output feature maps. The first two have kernel size 7 and the rest have kernel size 3. All have stride 1 and there are max-pooling layers (kernel size 3, stride 3) after the 1st, 2nd and 6th convolutional layers. The last two layers are fully-connected and have 128 hidden units and ReLU non-linearities.

For E3B, we used the same architecture as the policy encoder for the feature embedding ϕ, except we removed the last layers mapping the hidden representation to the actions and value estimate. The inverse dynamics model is a single-layer fully-connected network with 256 hidden units, mapping two concatenated ϕ outputs to a softmax distribution over actions.

For RND and NovelD, we also used the same architecture as above for the target and predictor networks. For AGAC, we used the same network as the policy for the adversary.

E.2. RL Hyperparameters

For all algorithms we use IMPALA (Espeholt et al., 2018) as our base policy optimizer. Hyperparameters which are common to all methods are shown in Table 3. All algorithms were trained for 50 million environment steps. We did not anneal learning rates for any of the methods during training.

Hyperparameters specific to the E3B, RND, NovelD and AGAC components are shown in Tables 4, 5 and 6. For all algorithms using an exploration bonus, we used a rolling normalization of the intrinsic reward similar to that proposed in the RND paper (Burda et al., 2019b). Specifically, we maintained a running standard deviation σ of the intrinsic rewards and divided the intrinsic rewards by σ before feeding them to the policy optimizer. For E3B and NovelD, we set the hyperparameters to the values reported in (Henaff et al., 2022). For AGAC, we set the adversary learning rate to be $0.3\times$ the policy learning rate as was done in the official code release. We used the same coefficient for the adversary loss (0.00004)–we also experimented with higher values of the adversary loss, but these performed less well.

Table 3. Common IMPALA Hyperparameters for MiniHack

Learning Rate	0.0001
RMSProp smoothing constant	0.99
RMSProp momentum	0
RMSProp ϵ	10^{-5}
Unroll Length	80
Number of buffers	80
Number of learner threads	4
Number of actor threads	256
Max gradient norm	40
Entropy Cost	0.005
Baseline Cost	0.5
Discounting Factor	0.99

Table 4. E3B Hyperparameters

Ridge λ	0.1
Intrinsic Reward Normalization	True
Intrinsic Reward Coefficient	1.0

For the experiments in Section 4.2, we tuned the β hyperparameter over the range $\{1, 10, 100, 1000, 10000, 100000\}$. In initial experiments we noticed that the episodic bonus was several orders of magnitude smaller than the episodic bonus, hence we used a high range of values for the β hyperparameter to bring the global bonus to a similar range.

Table 5. RND Hyperparameters

Predictor Learning Rate	0.0001
Intrinsic Reward Normalization	True
Intrinsic Reward Coefficient	1.0

Table 6. NovelD Hyperparameters

Predictor Learning Rate	0.0001
Scaling Factor c	0.1
Intrinsic Reward Normalization	True
Intrinsic Reward Coefficient	1.0

Table 7. AGAC Hyperparameters

Adversary Learning Rate	0.00003
Adversary loss term	0.00004
Intrinsic Reward Normalization	True
Intrinsic Reward Coefficient	1.0

E.3. Task Details

We used the following set of 16 MiniHack tasks: `'MiniHack-MultiRoom-N4-Locked-v0'`, `'MiniHack-MultiRoom-N6-Lava-v0'`, `'MiniHack-MultiRoom-N6-Lava-OpenDoor-v0'`, `'MiniHack-MultiRoom-N6-LavaMonsters-v0'`, `'MiniHack-MultiRoom-N10-v0'`, `'MiniHack-MultiRoom-N10-OpenDoor-v0'`, `'MiniHack-MultiRoom-N10-Lava-v0'`, `'MiniHack-MultiRoom-N10-Lava-OpenDoor-v0'`, `'MiniHack-LavaCrossingS19N13-v0'`, `'MiniHack-LavaCrossingS19N17-v0'`, `'MiniHack-Labyrinth-Big-v0'`, `'MiniHack-Levitate-Potion-Restricted-v0'`, `'MiniHack-Levitate-Boots-Restricted-v0'`, `'MiniHack-Freeze-Horn-Restricted-v0'`, `'MiniHack-Freeze-Wand-Restricted-v0'`, `'MiniHack-Freeze-Random-Restricted-v0'`.

Note that the `-Restricted-` versions of the tasks have restricted action spaces, as described in (Henaff et al., 2022).

E.4. Habitat

E.4.1. Environment Details

We used the HM3D (Ramakrishnan et al., 2021) dataset, which consists of 1000 high-quality renderings of indoor scenes. Observations consist of 4 modalities: an RGB and depth image (shown in Figure 7a), GPS coordinates and the compass heading. The action space consists of 4 actions: $\mathcal{A} = \{$`stop_episode`, `move_forward (0.25m)`, `turn_left (10°)`, `turn_right (10°)`$\}$. The dataset scenes are split into 800/100/100 train/validation/test splits. Since the test split is not publicly available, we evaluate all models on the validation split. Each scene corresponds to a different context $c \in \mathcal{C}$ in the CMDP framework.

To measure exploration coverage, we compute the area revealed by the agent's line of site using the function provided by the Habitat codebase [2], which uses a modified version of Bresenham's line cover algorithm. We define the exploration coverage to be:

$$\text{coverage} = \frac{\text{revealed area}}{\text{total area}}$$

See Figure 7b) for an illustration. For the results in Figure 5(a), we evaluated exploration performance for each algorithm by measuring its coverage on 100 episodes using scenes from the validation set (which were not used for training).

(a) (b)

Figure 7. a) Visual observations in Habitat b) Exploration is measured as the proportion of the environment revealed by the agent's line of sight over the course of the episode.

[2] https://github.com/facebookresearch/habitat-lab/blob/main/habitat/utils/visualizations/fog_of_war.py

E.4.2. ARCHITECTURE DETAILS

For all Habitat experiments we used the same policy network as in (Wijmans et al., 2020), which includes a ResNet50 visual encoder (He et al., 2015) and a 2-layer LSTM (Hochreiter & Schmidhuber, 1997) policy. In addition to RGB and Depth images, the agent also receives GPS coordinates and compass orientation, represented by 3 scalars total, which are fed into the policy. See the official code release at https://github.com/facebookresearch/habitat-lab/tree/main/habitat_baselines for full details.

For exploration algorithms which use inverse dynamics models (and ICM), we set the architecture of the encoder ϕ to be identical to that of the policy network, except that the last layer mapping hidden units to actions is removed. The inverse dynamics model was a single layer MLP with 256 hidden units and ReLU non-linearities.

For exploration algorithms which use random network distillation (RND and NovelD), we set the architecture of the random network to be identical to that of the policy network.

E.4.3. RL HYPERPARAMETERS

The DD-PPO hyperparameters which are common to all the algorithms are listed in Table 8. The hyperparameters which are specific to each algorithm are listed in Table 9, 10, 6. For NovelD's count-based bonus, hashing the full image was too slow to be practical, so we subsampled images by a factor of 1000 used that for the count-based bonus, along with the GPS coordinates and compass direction.

Table 8. Common PPO/DD-PPO Hyperparameters for Habitat

Clipping	0.2
PPO epochs	2
Number of minibatches	2
Value loss coefficient	0.5
Entropy coefficient	0.00005
Learning rate	0.00025
ϵ	10^{-5}
Max gradient norm	0.2
Rollout steps	128
Use GAE	True
γ	0.99
τ	0.95
Use linear clip decay	False
Use linear LR decay	False
Use normalized advantage	False
Hidden size	512
DD-PPO Sync fraction	0.6

E.4.4. COMPUTE DETAILS

Each job was run for 225 million steps, which took approximately 3 days on 32 GPUs with 10 CPU threads.

Table 9. Hyperparameters for E3B on Habitat

Hyperparameter	Values considered	Final Value
Ridge regularizer λ	$\{0.1\}$	0.1
Intrinsic reward coefficient β	$\{1.0, 0.1, 0.01, 0.001, 0.0001$	0.1
Inverse Dynamics Model updates per PPO epoch	3	3

Table 10. Hyperparameters for RND on Habitat

Hyperparameter	Values considered	Final Value
Intrinsic reward coefficient β	$\{1.0, 0.1, 0.01, 0.001, 0.0001$	0.1
Predictor Model updates per PPO epoch	3	3

Table 11. Hyperparameters for ICM on Habitat

Hyperparameter	Values considered	Final Value
Intrinsic reward coefficient β	$\{1.0, 0.1, 0.01, 0.001, 0.0001$	0.1
Forward Dynamics Model loss coefficient	$\{1.0\}$	1.0

E.5. Experiment Details for Montezuma's Revenge

We used the PyTorch RND implementation from https://github.com/jcwleo/random-network-distillation-pytorch as our base codebase, which reimplements Burda et al. (2019b). The policy network consists of 3 convolutional layers with $32/64/64$ feature maps followed by an MLP with 3 hidden layers with $256/448/448$ hidden units respectively and ReLU non-linearities. The RND predictor network consists of 3 convolutional layers with $32/64/64$ feature maps and Leaky ReLUs followed by a 2-layer MLP with 512 hidden units and standard ReLUs. The target network is the same as the predictor network, except it only has a single hidden layer MLP following the convolutional layers.

For E3B we used the same network architecture as for the predictor network for the ψ encoder. We experimented with both learning the ψ encoder using an inverse dynamics model and using a fixed random network, which has been shown to work well in certain cases (Burda et al., 2019a), and found that using a fixed random network worked best. We tuned the ridge regularization for the covariance matrix over the range $\{0.01, 0.1, 1.0\}$ and kept 1.0 as the final value.

Table 12. Hyperparameters for PPO+RND on Montezuma's Revenge

Hyperparameter	Value
Max Step Per Episode	4500
Extrinsic Reward Coefficient	2
Intrinsic Reward Coefficient	1.
Learning Rate	1e-4
Num. Env	128
Rollout length	128
γ	0.999
Intrinsic γ	0.99
λ	0.95
StableEps	1e-8
Frame Stack	4
Image Height	84
Image Width	84
UseGAE	True
Gradient Clipping Norm	0.5
Entropy	0.001
Epoch	4
MiniBatch	4
PPOEps	0.1
ActionProb	0.25
UpdateProportion	0.25
ObsNormStep	50

F. Algorithm Details for MiniHack Experiments

The bonuses for each algorithm we consider are detailed below:

$$
\begin{aligned}
b_{\text{E3B}\times\text{AGAC}}(s_t) &= \left[\phi(s_t)^\top C_{t-1}^{-1}\phi(s_t)\right] \cdot D_{\text{KL}}(\pi(\cdot|s_t)\|\pi_{\text{adv}}(\cdot|s_t)) \\
b_{\text{E3B}\times\text{RND}}(s_t) &= \left[\phi(s_t)^\top C_{t-1}^{-1}\phi(s_t)\right] \cdot \|f(s_t) - \bar{f}(s_t)\|_2^2 \\
b_{\text{E3B}\times\text{NovelD}}(s_t) &= \left[\phi(s_t)^\top C_{t-1}^{-1}\phi(s_t)\right] \cdot \left[\|f(s_{t+1}) - \bar{f}(s_{t+1})\|_2^2 - c\|f(s_t) - \bar{f}(s_t)\|_2^2\right]_+ \\
b_{\text{E3B}+\text{AGAC}}(s_t) &= \left[\phi(s_t)^\top C_{t-1}^{-1}\phi(s_t)\right] + \beta D_{\text{KL}}(\pi(\cdot|s_t)\|\pi_{\text{adv}}(\cdot|s_t)) \\
b_{\text{E3B}+\text{RND}}(s_t) &= \left[\phi(s_t)^\top C_{t-1}^{-1}\phi(s_t)\right] + \beta\|f(s_t) - \bar{f}(s_t)\|_2^2 \\
b_{\text{E3B}+\text{NovelD}}(s_t) &= \left[\phi(s_t)^\top C_{t-1}^{-1}\phi(s_t)\right] + \beta\left[\|f(s_{t+1}) - \bar{f}(s_{t+1})\|_2^2 - c\|f(s_t) - \bar{f}(s_t)\|_2^2\right]_+
\end{aligned}
$$

Here ϕ is learned online using an inverse dynamics model. The algorithms above include all possible combinations of global bonuses (second term) with the elliptical bonus (first term), and combining the two by multiplication or by taking a weighted sum. For the algorithms which take a weighted sum, we tuned the β term on a subset of tasks, and report the best value on all 16 tasks.

G. Additional Experiment Results

G.1. Additional MiniHack Results

In Table 13 we report results for two additional MiniHack variants of the `MultiRoom` environment. The first features locked doors which the agent must kick down in order to move to the next room and eventually reach the exit. The second features moving monsters which the agent must fight or avoid while navigating towards the exit. In both cases, we see similar phenomena as in Section 3.1: the global bonus exhibits a severe performance drop when going from $|\mathcal{C}| = 1$ to $|\mathcal{C}| = \infty$, whereas the episodic bonus' performance is largely unchanged.

Environment	$\lvert\mathcal{C}\rvert$	ψ	Global	Episodic
`MultiRoom-N4-Locked-v0`	1	P	0.722 ± 0.446	0.948 ± 0.01
`MultiRoom-N4-Locked-v0`	∞	P	-0.230 ± 0.039	0.925 ± 0.01
`MultiRoom-N6-LavaMonsters-v0`	1	P	0.994 ± 0.001	0.76 ± 0.38
`MultiRoom-N6-LavaMonsters-v0`	∞	P	0.00 ± 0.00	0.75 ± 0.21

Table 13. Reward for global and episodic bonuses for different CMDPs, averaged across 5 seeds. Here $|\mathcal{C}|$ denotes the number of different contexts/maps which are sampled from at each episode. The ψ column indicates which feature encodings are used (P for positions, M for messages).

G.2. Evolution of Global and Episodic bonuses

In this section we show the evolution of the global and episodic bonus terms for the E3B×NovelD algorithm for four of the MiniHack tasks (see Figure 8). The first row displays the inverse dynamics model loss used for learning the ϕ embedding in E3B's episodic bonus, the second row shows the E3B episodic bonus itself, the third shows the NovelD global bonus, and the fourth shows the true extrinsic reward provided by the environment.

First, note that the global bonus spans a much larger range of values than the episodic bonus does, initially starting at a high value, exhibiting a first rapid decay, and then further decaying at a slower rate. In contrast, the episodic bonus spans a more limited range, and during most of the training it has higher magnitude than the global bonus. For the episodic bonus, we first see an initial decrease in magnitude, which is likely due to the ϕ features stabilizing (note that this coincides with the stabilization of the inverse dynamics loss used for learning ϕ). After this, the episodic bonus increases, indicating that the agent's policy is learning to maximize the episodic bonus. The episodic bonus then decreases again once the extrinsic

reward starts to increase, indicating that the agent has found the true environment reward and is switching to exploitation rather than exploration.

We hypothesize that additively combining the episodic and global bonuses does not offer much benefit over the episodic bonus alone because the magnitude of the global bonus decays significantly over time. Recall that the global bonus is computed using *all* the agent's experience, so it eventually becomes exhausted. Since the episodic bonus is reset each episode, it does not become exhausted the same way, as evidenced by the fact that it *increases* once the feature encoder ϕ has stabilized. If we add the two together, eventually the contribution of the global bonus will be small compared to the contribution of the episodic bonus. However, if we combine the two multiplicatively the global bonus will still have an effect regardless of its scale.

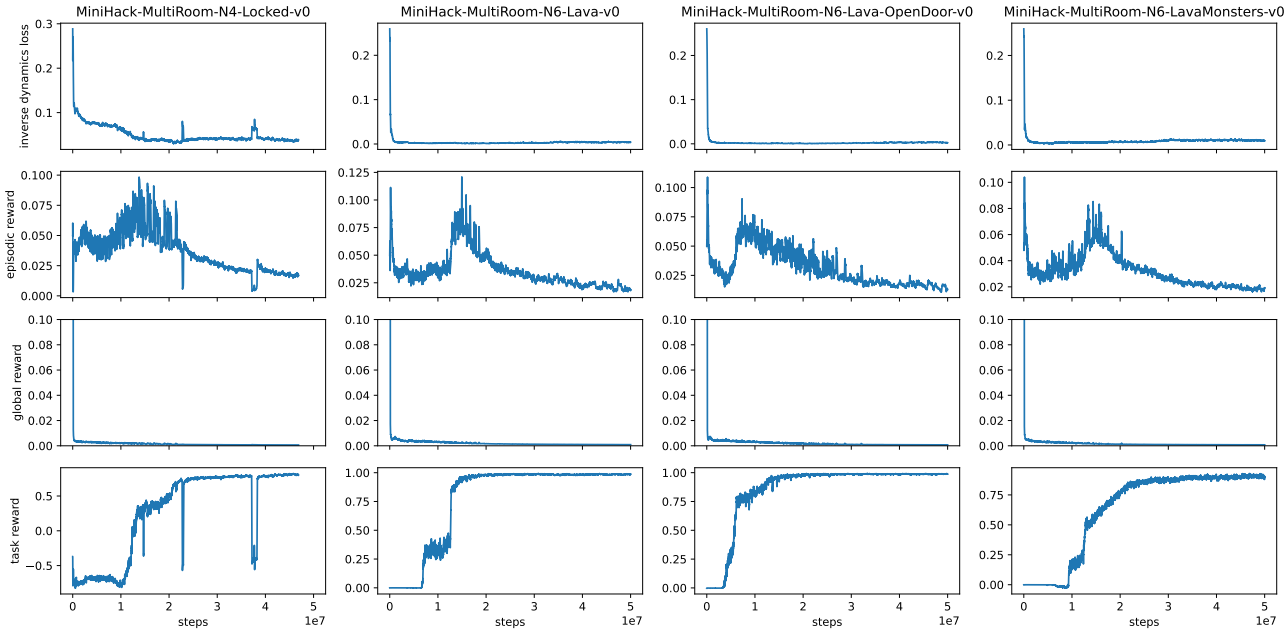


Figure 8. Evolution of inverse dynamics loss, episodic bonus, global bonus and extrinsic reward throughout training for E3B×NovelD.

G.3. Results with NGU episodic bonus

In this section we report results for the KNN-based episodic bonus from the Never-Give-Up (NGU) agent proposed in (Badia et al., 2020). Like the E3B bonus, the NGU bonus operates in the embedding space induced by an inverse dynamics model. A key difference is that it is based on the euclidean norm between the current state's embedding and its neareset neighbors within the episode, whereas E3B's bonus is computed using the metric induced by the episodic covariance matrix.

Results for a combination of NGU's episodic bonus with NovelD's global bonus are shown in Figure 9. Although this approach performs better than not including an exploration bonus (which gives average return of 0), it performs considerably worse than the variants which use E3B. Note that our results are consistent with (Wang et al., 2023), who also report poor results for NGU's episodic bonus on MiniGrid.

One possible explanation may be that since NGU's KNN-based bonus uses the euclidean norm, if one dimension has higher scale than others it may dominate and reduce the effect of other more informative features. Therefore, this bonus may be more sensitive to spurious features in embedding space. On the other hand, E3B automatically adjusts the scale of each feature by normalizing by the inverse covariance matrix, and may therefore be more robust.

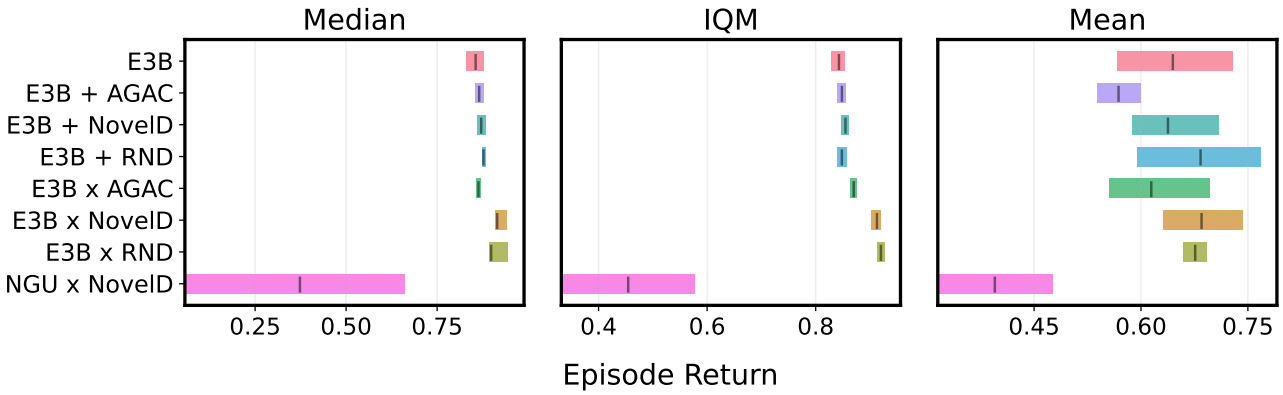


Figure 9. Performance for all methods including using NGU's episodic bonus.

H. Additional Discussion

In this section we provide additional discussion and examples for the framework described in Section 3.3.

H.1. Examples of ψ functions and Z spaces

1. **Tabular Algorithms:** In classical tabular algorithms with a UCB-style bonus (Kearns & Singh, 2002; Brafman & Tennenholtz, 2002; Strehl & Littman, 2006; Jin et al., 2018), ψ is simply the identity and $Z = \mathcal{S}$. The bonus is of the form $b(s) = 1/\sqrt{N(\psi(s))} = 1/\sqrt{N(s)}$.

2. **Deep RL Algorithms with position-based counts:** A number of recent exploration algorithms have used inverse counts of (x, y) locations to drive exploration in gridworlds, for example RIDE (Raileanu & Rocktäschel, 2020; Samvelyan et al., 2021), AGAC (Flet-Berliac et al., 2021) and NovelD (Zhang et al., 2021b; Henaff et al., 2022). We use this approach for our experiments on `MiniHack-MultiRoom` and `MiniHack-Corridor` in Section 3. In this case, $\psi(s_t) = (x_t, y_t)$, the coordinates of the agent at time t, and the bonus is defined as $b(s) = 1/\sqrt{N(\psi(s_t))}$. The space Z is a 2D lattice the size of the map, i.e.

$$Z = \{(i, j) : 1 \leq i \leq W, 1 \leq j \leq H\}$$

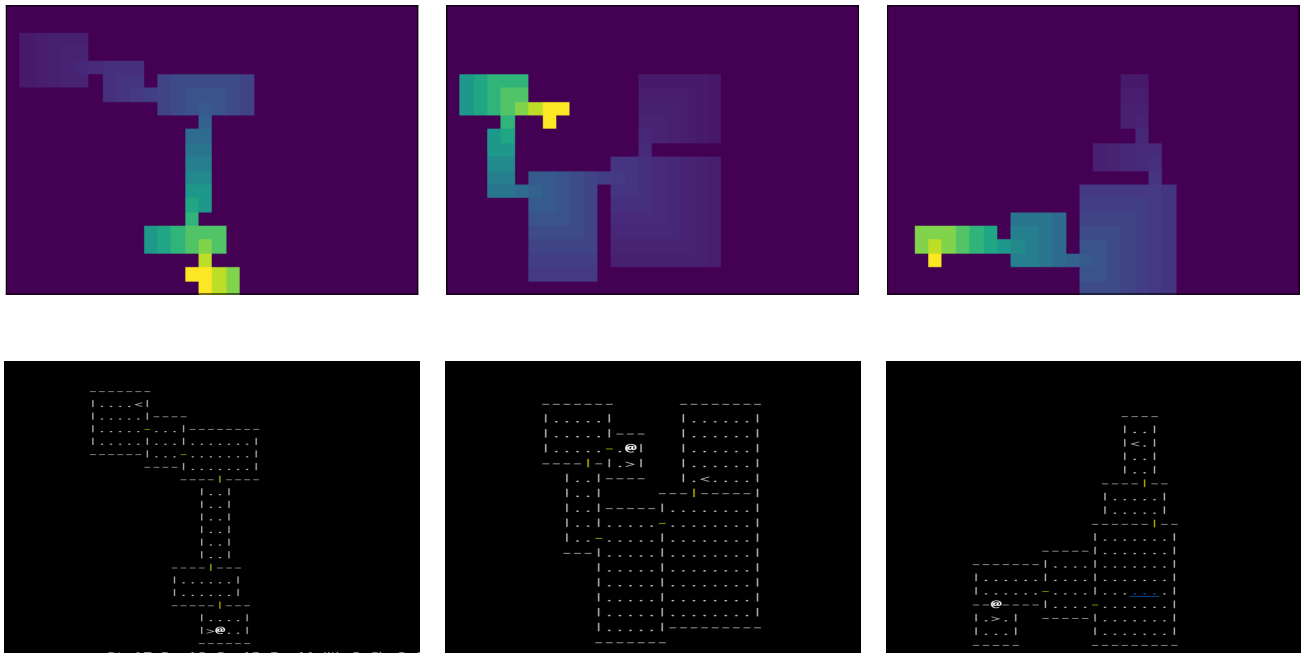
   where W, H correspond to the maximum width and height of the map across all contexts.

3. **Deep RL Algorithms with message-based counts:** Some recent methods have used inverse counts based on textual messages (Mu et al., 2022a; Henaff et al., 2022) for environments where these are available, such as MiniGrid or MiniHack. We use this approach for the `MiniHack-KeyRoom` environment shown in Figure 4. In this case, ψ extracts the message from the state and the bonus is computed as $b(s) = 1/\sqrt{N(\psi(s))}$. The space Z then consists of all possible messages for the given environment. For example, for the `MiniHack-KeyRoom` environment, we have

```
Z = {"You can't move diagonally out of an intact doorway.",
     "Be careful!  New moon tonight.",
     " ",
     "It's a wall.",
     "You see here a key named The Master Key of Thievery.",
     "h - a key named The Master Key of Thievery.",
     "The stairs are solidly fixed to the floor.",
     "It won't come off the hinges.",
     "You can't move diagonally into an intact doorway.",
     "This door is locked.",
     "Never mind.",
     "There is nothing here to pick up."
    }
```

4. **Elliptical Global Bonuses with kernel functions** The examples above have used count-based bonuses in feature space. However, our framework also captures algorithms which use exploration bonuses other than counts, such as elliptical bonuses. For example, the work of (Agarwal et al., 2020) uses an elliptical bonus in the space induced by an RBF kernel. In this case, ψ is the mapping to kernel space, $Z = \mathbb{R}^n$ (where n is the number of points used to compute the RBF kernel) and the bonus is given by $b(s_t) = \psi(s_t)^\top C_{t-1}^{-1} \psi(s_t)$. Here $C_{t-1} = \sum_{i=1}^{t-1} \psi(s_i)\psi(s_i)^\top$ is the (unnormalized) covariance matrix computed using all the agent's experience.

5. **Elliptical Episodic Bonuses with learned embedding** The recent E3B algorithm (Henaff et al., 2022) can be described within our framework as well. Here $\psi : \mathcal{S} \to Z$ is learned using an inverse dynamics model, and $Z = \mathbb{R}^k$ is the embedding space. The bonus is given by $b(s) = \psi(s_t)^\top C_t^{-1} \psi(s_t)$, where $C_{t-1} = \sum_{i=t_0}^{t-1} \psi(s_i)\psi(s_i)^\top$ is the episodic covariance matrix (with t_0 denoting the start of the current episode). Here ψ is learned using an inverse dynamics model (Pathak et al., 2017b).



(a) $c = 1$ (b) $c = 2$ (c) $c = 3$

Figure 10. Top row: Value maps $V^\star_{\psi,c}$ for 3 different contexts of the `MultiRoom` environment. For each $z = (x, y)$ location on the map, we display $V^\star_{\psi,c}(z) = \gamma^{k(x,y)}$, where $k(x, y)$ is the shortest path from the (x, y) location to the goal and $\gamma = 0.9$. Bottom row shows corresponding maps. Note that $V^\star_{\psi,c}$ changes significantly with different values of c.

H.2. Visualizations of $V^\star_{\psi,c}$ functions

We now provide visualizations of the $V^\star_{\psi,c}$ functions defined in Examples 1 and 2 of Section 3.3. Figure 10 shows the functions $V^\star_{\psi,c}$ for three different contexts c in Example 1, corresponding to different maps, with $\psi : \mathcal{S} \to Z$ given by $\psi(s_t) = (x_t, y_t)$. Note that here Z is a 2D lattice consisting of all possible (x, y) positions on the map. For each $z = (x, y)$, we have $V^\star_{\psi,c}(z) = \gamma^{k(x,y)}$, where $k(x, y)$ denotes the shortest path to the goal computed using graph search. We use $\gamma = 0.9$ for these visualizations. The function $V^\star_{\psi,c}$ changes significantly for different contexts (maps) c, since the geometry of the map and the goal location change from one context to the other.

Next, in Figure 11 we visualize the $V^\star_{\psi,c}$ function for the `MiniHack-KeyRoom` environment (shown in Figure 4), with $\psi : \mathcal{S} \to Z$ extracting messages from states. In this case Z consists of the set of all possible messages which can be seen in this environment. We show $V^\star_{\psi,c}$ for 3 different contexts c. Unlike in the previous example, here $V^\star_{\psi,c}$ has similar shape across all contexts. Messages which can only occur once the door has opened, such as `"It won't come off the hinges"` and `"You can't move diagonally out of an intact doorway"`, have the highest value. Messages corresponding to the agent visiting or picking up the key, which is necessary for opening the door, have medium value (e.g. `"You see here a key names The Master Key of Thievery"`, `"h - a key named The Master Key of Thievery"`. The remaining messages, such as `" "`, `"This door is locked"`, `"It's a wall"`, do not indicate that the agent has made progress on the task and have the lowest value.

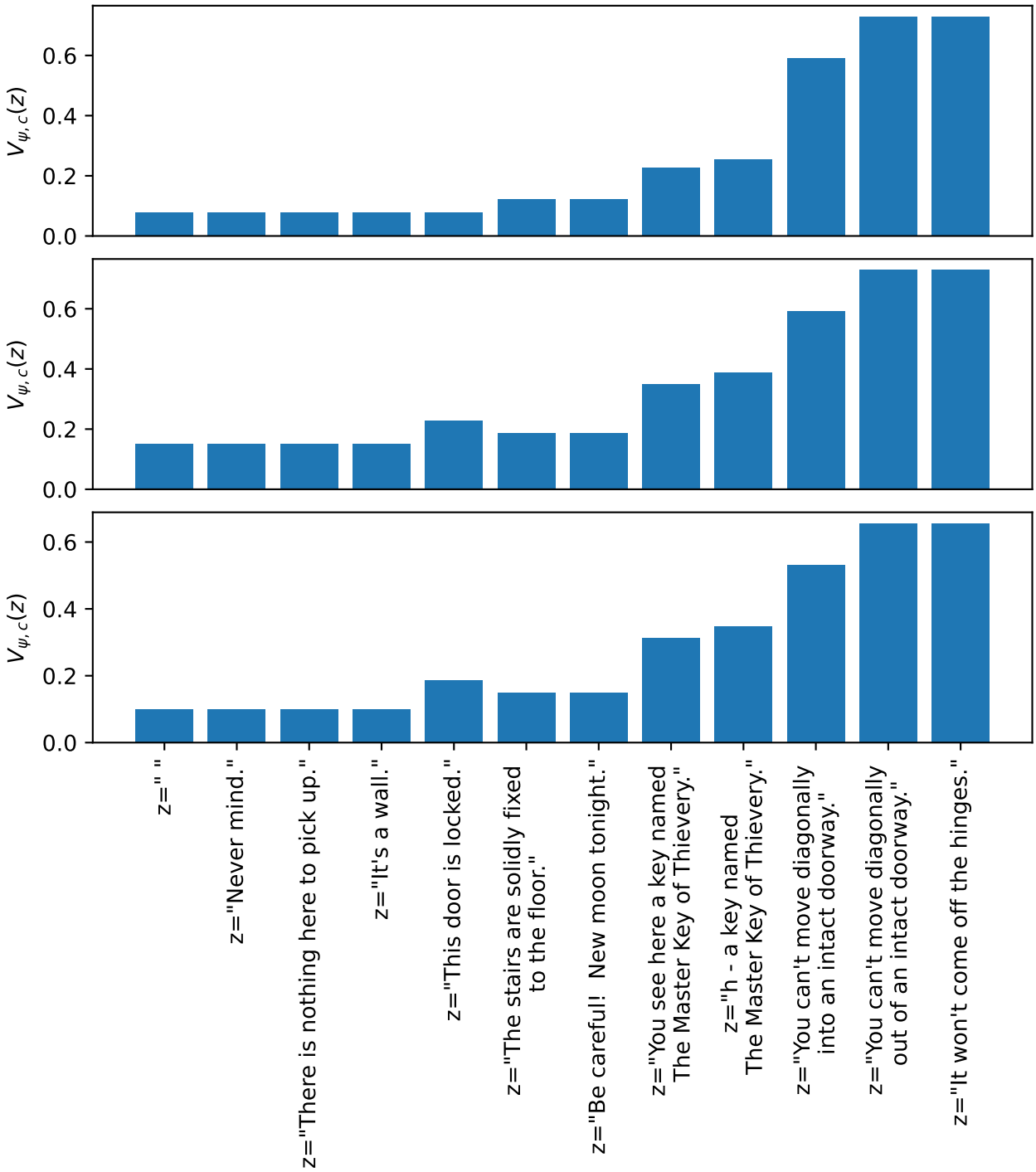


Figure 11. Visualization of value maps $V^{\star}_{\psi,c}$ for 3 different contexts for the `KeyRoom` environment where ψ encodes messages. Here $V^{\star}_{\psi,c}$ changes little with different contexts.

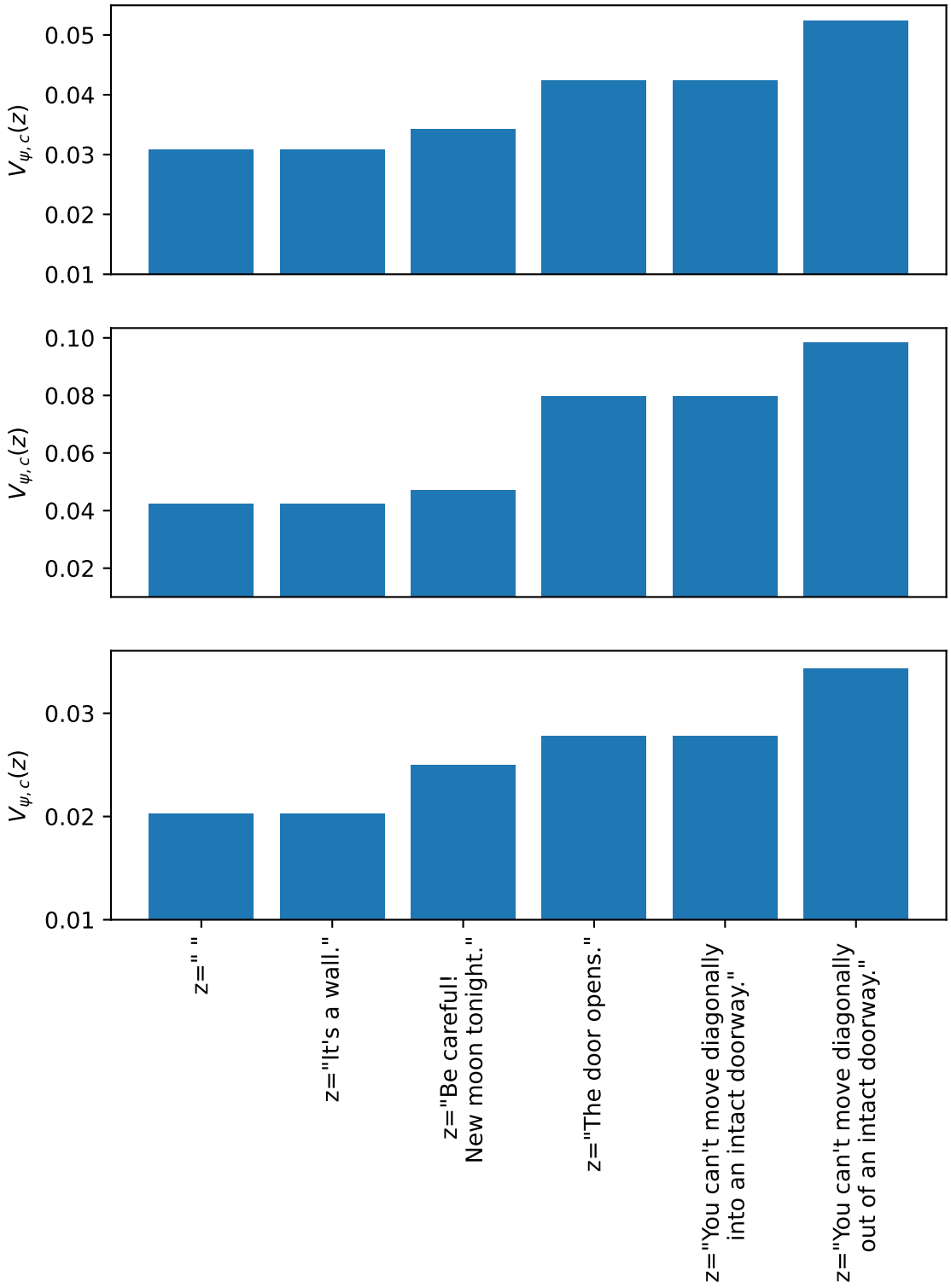


Figure 12. Visualization of value maps $V^{\star}_{\psi,c}$ for 3 different contexts for the `MultiRoom` environment where ψ encodes messages. Here $V^{\star}_{\psi,c}$ changes little with different contexts.

Example	Environment	ψ	Average Cosine Similarity
1	MultiRoom	positions	0.198
2	KeyRoom	messages	0.991
3	MultiRoom	messages	0.951

Table 14. Average cosine similarity for different examples from Section 3.3.

H.3. Quantifying the Variation of $V^\star_{\psi,c}$ across episodes

One way in which the variation of $V^\star_{\psi,c}$ across contexts can be made precise is to take the average cosine similarity between $V^\star_{\psi,c}$ and $V^\star_{\psi,c'}$ for randomly sampled contexts c, c'. Note that this is well-defined if Z is finite or infinite—if Z is infinite, we replace dot products between vectors by inner products between functions:

$$\text{average cosine similarity} = \mathbb{E}_{c,c' \sim \mu_C}\left[\frac{\langle V^\star_{\psi,c}, V^\star_{\psi,c'}\rangle}{\|V^\star_{\psi,c}\| \cdot \|V^\star_{\psi,c'}\|}\right] \tag{9}$$

Average cosine similarities for all three examples in Section 3.3 are given in Table 14. This is consistent with what we observe in the visualizations in Appendix H.2, where the $V^\star_{\psi,c}$ for Example 1 appear very different across contexts whereas those for Examples 2 and 3 appear very similar. It is also consistent with our experimental results where the episodic bonus succeeds for Example 1 and the global bonus succeeds for Examples 2 and 3.

It is difficult to say in general at what exact rate the global performance will degrade with decreasing average cosine similarity of $V^\star_{\psi,c}$ across contexts. However, to get some idea we can check this empirically for our `MultiRoom` experiments with position encodings and different $|\mathcal{C}|$ from Table 1. In Table 15 below we compute the average cosine similarity of $V^\star_{\psi,c}$ across contexts and compare this to the global bonus performance. Again, we see that the global bonus performs worse in settings with low average cosine similarity.

Environment	ψ	$\lvert\mathcal{C}\rvert$	Average Cosine Similarity	Global Bonus Performance
MultiRoom	position	1	1.000	0.99
MultiRoom	position	3	0.465	0.59
MultiRoom	position	5	0.358	0.23
MultiRoom	position	10	0.278	0.02
MultiRoom	position	∞	0.198	0.00

Table 15. Average cosine similarity vs. global bonus performance on MultiRoom for different numbers of contexts.