Applied Intelligence (2021) 51:1086–1107
https://doi.org/10.1007/s10489-020-01849-3

Fast and slow curiosity for high-level exploration in reinforcement learning

Nicolas Bougie[1,2] · Ryutaro Ichise[1,2]

Published online: 16 September 2020

Abstract

Deep reinforcement learning (DRL) algorithms rely on carefully designed environment rewards that are extrinsic to the agent. However, in many real-world scenarios rewards are sparse or delayed, motivating the need for discovering efficient exploration strategies. While intrinsically motivated agents hold promise of better local exploration, solving problems that require coordinated decisions over long-time horizons remains an open problem. We postulate that to discover such strategies, a DRL agent should be able to combine local and high-level exploration behaviors. To this end, we introduce the concept of fast and slow curiosity that aims to incentivize long-time horizon exploration. Our method decomposes the curiosity bonus into a fast reward that deals with local exploration and a slow reward that encourages global exploration. We formulate this bonus as the error in an agent's ability to reconstruct the observations given their contexts. We further propose to dynamically weight local and high-level strategies by measuring state diversity. We evaluate our method on a variety of benchmark environments, including Minigrid, Super Mario Bros, and Atari games. Experimental results show that our agent outperforms prior approaches in most tasks in terms of exploration efficiency and mean scores.

Keywords Reinforcement learning · Exploration · Autonomous exploration · Curiosity in reinforcement learning

1 Introduction

In recent years, deep reinforcement learning (DRL) has achieved many accomplishments in a wide range of application domains, such as game playing [40, 51], robot control [35], and autonomous vehicles [33]. DRL algorithms rely on maximizing the cumulative rewards that are provided by the environment. However, most DRL algorithms rely on well-designed and dense rewards to guide the behavior of the agent. Hand-crafting such reward functions is a challenging engineering problem. In order to deploy DRL to real-world settings wherein rewards are often sparse or poorly defined, DRL agents will have to discover efficient exploration strategies. Multiple heuristics such as entropy regularization [41] were introduced but did not yield significant improvements in sparse reward tasks.

Several works attempt to tackle this challenge by providing a new intrinsic exploration bonus (i.e. curiosity) to the agent. For example, count-based exploration [55] keeps visit counts for states and favors the exploration of states rarely visited. Another class of methods relies on predicting dynamics of the environment [2]. For instance, ICM [44] predicts the feature representation of the next state based on the current state and the action taken by the agent. Nevertheless, maximizing the prediction error tends to attract the agent to stochastic transitions, where the consequences of actions are hardly predictable [11]. This issue has motivated several recent works [11, 48]. Despite their ability to deal with *local exploration* - exploring the consequences of short-term decisions (e.g. how to pass an obstacle or whether to interact with a particular object), global exploration remains an open problem. *Global exploration*, also referred to as "deep exploration," considers the future usefulness of decisions within some temporally-extended horizon. One reason is that intrinsic rewards were shown to vanish quickly [48] with additional visitations, letting the agent without long-term incentive. Therefore, these methods generally struggle

✉ Nicolas Bougie
nicolas-bougie@nii.ac.jp

Ryutaro Ichise
ichise@nii.ac.jp

[1] National Institute of Informatics, Tokyo, Japan

[2] The Graduate University for Advanced Studies, Sokendai, Tokyo, Japan

in tasks wherein a large enough intrinsic reward should be given to the agent to discover long-time horizon strategies and avoid local optima.

Some recent works have suggested that what makes human learners so efficient at learning is the brain's computational capacities to act over multiple spatial-temporal scales [9, 22], that is, humans employ fast and slow forms of learning. One natural question that arises is: how these principles relate to exploration and how can we replicate them to improve exploration. Inspired by this observation, we present Fast and Slow intrinsic curiosity (FaSo) that can deal with high-level exploration, the combination of fast and slow dominant exploration phases. We postulate that to efficiently explore its environment, the agent should combine a short-time and long-time horizon strategy. To do so, the proposed method decomposes the curiosity bonus into a fast and a slow intrinsic reward. Fast rewards deal with local exploration by rapidly adapting to evaluate the novelty of states. In contrast with fast rewards, slow rewards change slowly and remain large to explicitly encourage the exploration of distant or hard-to-reach regions of the state space (global exploration). We formulate intrinsic rewards as the reconstruction errors of the observations given their *contexts*. Namely, a *reconstructor* network takes an observation with missing or corrupted regions, the observation's context (Fig. 1), and attempts to reconstruct the original image. We show that having two types of reconstruction tasks (i.e. fast model and slow model) can lead to different exploration behaviors characterized by different time horizons. A key difference between these models lies in their context creation strategies. Moreover, using noisy inputs forces the model to capture meaningful visual features and salient environmental dynamics at different scales (e.g. character-level or word-level). We further propose an adaptive scaling technique to modulate fast and slow rewards by measuring state diversity.

In summary, our main contribution is a high-level exploration mechanism relying on fast and slow rewards, which can scale to problems with complex visual observations and that is applicable to most on-policy algorithms. We propose as curiosity reward the reconstruction errors of the observations given their contexts using two different architectures of deep neural network inspired by auto-encoders. Furthermore, we trade-off local with global exploration strategies by estimating state diversity. We evaluate the performance of our method in a variety of sparse reward environments, including Minigrid, Super Mario Bros, and Atari games. We demonstrate that FaSo is preferable to the previous intrinsic reward methods in terms of exploration efficiency, especially in environments featuring sparse rewards. Experimental results also show that high-level exploration is crucial when the environment contains deceptive rewards such as in Pitfall (Atari).

2 Background

Our method builds on top of on-policy approaches for reinforcement learning, and unsupervised image representation learning for defining an exploration bonus. We briefly introduce them in this section.

2.1 Reinforcement learning

Reinforcement learning [56] consists of an agent learning a policy π by interacting with an environment. At each time-step the agent receives an observation s_t and chooses an action a_t. The agent gets a feedback from the environment called reward, r_t. Given this reward and the current observation, the agent can update its policy to improve the future expected rewards. Given a discount factor γ, the future discounted rewards, called return $\mathcal{R}_t$, is defined as $\mathcal{R}_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$, where T is the time-step at which the epoch terminates. The agent learns to select the action with the maximum return $\mathcal{R}_t$ achievable for a given observation [57]. We can define the action value $Q^{\pi}(s, a)$ at a time t as the expected reward for selecting an action a for a given state s_t and following a policy π: $Q(s, a) =$

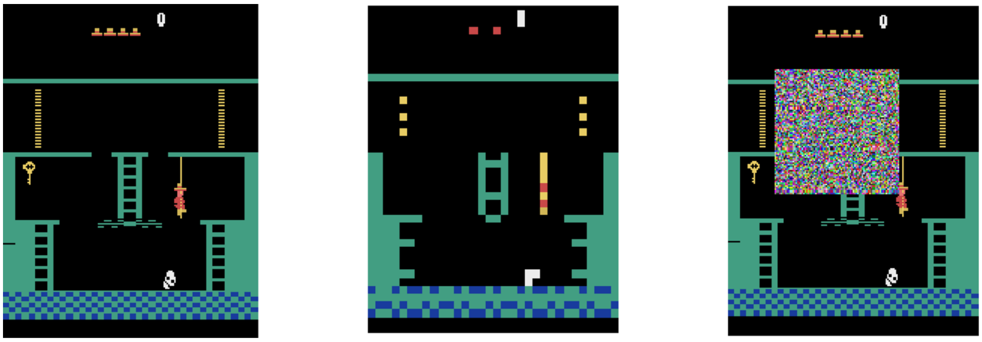

(a) Original Frame (b) Downsample Context (c) Noisy Context

Fig. 1 Examples of contexts created from one frame (left column) of the Montezuma's Revenge environment. Middle column: downsample context. Right column: noisy context

$\mathbb{E}[R_t \mid s_t = s, a]$. The policy being learned is called the target policy, and the policy used to generate exploration behavior is called the behavior policy.

2.2 On-policy methods

We now review two state-of-the-art on-policy algorithms which were used as the learning method in this work. In contrast with off-policy methods such as DQN [40] that decouple the behavior and target policies, on-policy methods update their value functions based on the trajectories generated following the current policy. In the setting of curiosity-driven learning, on policy-methods (e.g. PPO) were shown to be robust learning methods that require little hyperparameter tuning [12]. It is also desirable to guide exploration behaviors based on intrinsic motivation.

A2C is a synchronous variant of A3C [41] that takes advantage of parallel learning to efficiently learn. It consists in a critic that estimates the value function to criticize the actions made by the actor, and an actor that learns a policy $\pi(a|s, \theta)$ by minimizing the following loss function:

$$\mathcal{L}_{actor}(\theta) = -\mathbb{E}_{s,a\sim\pi}\left[\mathcal{R}_t - V_\pi(s) + \beta H(\pi(.|s, \theta))\right] \quad (1)$$

where $H(\pi(.|s, \theta))$ is the entropy to encourage exploration and avoid convergence toward a sub-optimal policy, β controls the importance of exploration during training, and $\mathcal{R}$ is an estimation of the return. The value function $V_\pi(s)$ represents the excepted return for a state s following the policy π: $V_\pi(s) = \mathbb{E}_{a\sim\pi(a|s)}\left[\mathcal{R}_t|s_t = s\right]$

We now present Proximal Policy Optimization (PPO), a specific technique for optimizing policies. PPO introduces a penalty which controls the change of the policy at each iteration to reduce oscillating behaviors. The objective function becomes:

$$\mathcal{L}_{CLIP}(\theta) = \mathbb{E}_t\left[\min(\varrho_t(\theta)A_t, \text{clip}(\varrho_t(\theta), 1-\epsilon, 1+\epsilon)A_t)\right] \quad (2)$$

where ϱ_t is the probability ratio, $\varrho_t = \pi_\theta(a|s)/\pi_{\theta_{old}}(a|s)$, and ϵ is a hyperparameter. The ratio ϱ_t is clipped to fall between $(1-\epsilon)$ and $(1+\epsilon)$. A_t represents the advantage function $A_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t)$ - the benefits of taking an action compared to the others, in a state.

2.3 Unsupervised image representation learning

Our method combines reinforcement learning with unsupervised representation learning to detect novelty. Here, we briefly introduce deep unsupervised image representation learning methods. An autoencoder [5] consists in an encoder network learning how to compress the input data x into an encoded representation $\phi(x)$, and then the original image is reconstructed $\hat{x}$ by a decoder network. By using a low-dimensional middle layer (i.e. bottleneck layer), the model is forced to extract relevant features. The parameters of the neural network are trained to optimize a loss function L_{AE} where the first term $\mathcal{L}$ penalizes the reconstruction error of the target given the input and the second term prevents overfitting, such as L1 regularization:

$$\mathcal{L}_{AE} = \mathcal{L}(x, \hat{x}) + \lambda \sum_i |a_i^{(h)}| \quad (3)$$

where $a_i^{(h)}$ represents the activation values in layer h for observation i, and λ is a scaling parameter.

In contrast, variational autoencoders (VAEs) [31] are generative models used to learn latent representation of high dimensional data such as images. The input image is passed through an encoder network q_ϕ which outputs the parameters μ and σ of a multivariate Gaussian distribution. A latent vector is sampled and the decoder network p_ψ decodes it into the original state space. The parameters ϕ and ψ of the encoder and decoder are jointly optimized to maximize:

$$\mathcal{L}(\psi, \phi; s^{(i)}) = \beta D_{KL}(q_\phi(z|s^{(i)})||p(z)) - \mathbb{E}_{q_\phi(z|s^{(i)})}\left[\log p_\psi(s^{(i)}|z)\right] \quad (4)$$

where the first term is a regularizer, the Kullback-Leibler divergence between the encoder distribution $q_\phi(z|s^{(i)})$ and $p(z)$. $p(z)$ is some prior specified as a standard normal distribution $p(z) = \mathcal{N}(0, 1)$. The second term is the expected negative log-likelihood - the reconstruction loss.

One application of encoder-decoder models is image denoising. For example, sparse denoising autoencoders [17] use a sparsity regularization during training. A recent follow-up of this work proposes a simple sparsification method that sparsifies the latent representation [15]. Another solution is to divide the training dataset into several subtasks and conquer each subtask using a neural network [24]. Another strategy is to exploit a serie of convolutional layers and deconvolutional layers and add skip connections between corresponding convolutional and deconvolutional layers [38]. The proposed reconstruction-based curiosity can be viewed as a form of image denoising. However, we do not intend a perfect image reconstruction as the above-mentioned approaches, but instead, we aim to measure the reconstruction error. In order to meet real-time constraints, the proposed network architectures (see Section 4.2) use "shallow" architectures and connect all the locations together within a feature map. We further introduce different losses to capture small details.

3 Related work

Our work is also related to encouraging exploration in reinforcement learning. Most approaches can be grouped into three classes: count-based strategies, goal conditioned learning, and curiosity-driven exploration. This section provides a comprehensive comparison with those researches.

One line of work is to keep visit counts for states to favor exploration of rarely visited states [7, 37, 55]. To apply count-based exploration to continuous state spaces, a solution [43] is to train an observation density model to supply counts. Another solution [58] maps states to hash codes and counts state visitations with a hash table. In order to reduce the size of the count table, a method clusters states and keeps visit counts of clusters instead of the original states [1]. A prior work [39] introduces a count-based optimistic algorithm by estimating the uncertainty associated with each state. However, one can expect these methods to be less effective when some *valuable* states require more attention - more visits. In this setting an agent can visit a less frequently visited state many times, even though the value in this state is already estimated. On the other hand, the proposed curiosity formulation is based on the agent's understanding of the world. Hence, states easy to reconstruct will be less novel than states featuring sophisticated visual patterns.

Goal conditioned learning [28] motivates an agent to explore by constructing a goal-conditioned policy, and then optimize the rewards with respect to a goal distribution. While intrinsically motivated agents can easily get trapped in local optima, goal conditioned methodologies aim to maximally cover a behavioral goal space, acting as a "global" exploration strategy. For instance, universal value function approximators (UVFA) [49] construct a set of optimal value functions by a single function approximator that can generalize over both states and goals. The recent work on hindsight experience replay (HER) [4] forms an implicit curriculum by using visited states as a target. However, selecting relevant goals is not easy. A class of work [6] and its recent follow-up [18, 42], proposed to generate increasingly difficult goals to provide additional feed-back during exploration. Rather than operating directly on observations, the approach [45] learns an embedding for the goal space using unsupervised learning and then chooses the goals from that space. Another line of work [19] improves exploration by focusing on goals that provide maximal learning progress. The recent work [47], Skew-fit, proposes an exploration objective that maximizes state diversity. In particular, they use the density in a VAE latent space to model state diversity. The key idea is to learn a maximum-entropy goal distribution to match the weighted empirical distribution, where the rare states receive larger weights. In contrast, we assume a fixed goal and train a VAE to reconstruct the original image given a noisy input. Our method also works in the latent space to estimate state diversity, but state diversity is used to adjust the scaling factors of intrinsic rewards. In a similar fashion, option discovery methods [3, 27, 36] use options as sub-goals in order to introduce high-level supervision during training. For instance, FeUdal network [60] employs a manager and low-level policy. The manager policy guides the low-level policy by sending sub-goal signals that are not explicitly defined.

This work belongs to the category of techniques that consider curiosity as a drive to explore. Some methods [44, 53] rely on predicting environment dynamics using an inverse or forward dynamic model. Another class of approaches uses prediction errors in the feature space as measure of the importance of states [32]. For instance, RND [11] predicts the output of a randomly initialized neural network on the current state. In contrast, our work introduces the idea of reward decomposition to achieve flexible exploration behaviors. Another crucial difference here is that each reward stream is independently calculated, by taking advantage of context-based features. A recent solution [64] aims to leverage motion features in the observations. They utilize the errors from a forward and backward optical flow estimation to asses the novelty of observations. In [10], the authors formulate curiosity as the ability of the agent to achieve a set of goals. By doing so, they incentivize the exploration of hard-to-learn states. Introducing a new term in the loss function that attemps to maximize state diversity was used to deal with local exploration [25]. Learning without extrinsic rewards has also been studied and is referred to as *novelty search* [34] - an evolutionary algorithm designed to escape from local optima by defining selection pressure in terms of behavior. The novelty of an event is estimated as the distance of the current behavior features to the *k-nearest* neighbors in a memory of behavior features. Episodic curiosity through reachability [48] addresses the "noisy TV" problem by considering the number of time-steps between two states as curiosity measure. Exploration bonus can also be based on maximizing information gain about the agent's knowledge of the environment [26]. As a step toward addressing this problem, an algorithm [20] guides exploration by maximizing an entropy objective. In this work, our formulation of curiosity is a deterministic problem which not only deals with local exploration but aims to introduce high-level exploration in DRL.

4 Fast and slow exploration

The main objective of the proposed fast and slow driven exploration (FaSo) method is to encourage high-level exploration. We focus on tasks where extrinsic rewards r_t^e are sparse; zero for most of the time steps t. In addition to this reward, our method produces an intrinsic curiosity-based bonus r_t^i that can be decomposed into a fast reward r_t^{fast} and a slow reward r_t^{slow}. The fast reward r_t^{fast} deals with local exploration by rapidly changing over time based on the novelty of the current observation. The slow reward r_t^{slow} acts as a global exploration bonus to explicitly encourage deep exploration behaviors which would move the agent toward regions of the state-action space that are novel or hard to reach. r_t^{fast} and r_t^{slow} are assigned by using the idea of image reconstruction error given an observation's *contexts*. Formally, we consider an agent interacting with an environment; at each time step t the agent performs an action and receives an augmented reward:

$$r_t = r_t^e + r_t^i = r_t^e + \left[\alpha r_t^{fast} + \beta r_t^{slow}\right] \tag{5}$$

where α and β are hyperparameters to weight the importance of both rewards. The policy $\pi(s_t; \theta_P)$ is represented by a deep neural network with parameters θ_P. Its parameters are optimized to maximize the excepted sum of these two rewards:

$$\max_{\theta_P} \mathbb{E}_{\pi(s_t;\theta_P)}\left[\sum_t r_t\right] \tag{6}$$

In the following section, we describe in detail the key components. First, we present how an intrinsic reward signal is generated based on reconstruction error given observation's context. Second, we introduce two neural network models to reconstruct noisy images. Finally, we formalize how fast and slow reconstruction-based rewards are combined to achieve high-level exploration.

4.1 Reconstruction-based curiosity

We propose the concept of reconstruction-based curiosity as our novelty measurement scheme (Fig. 2). At every step, the module takes the *context* of the current observation s^* as its input, and reconstructs the original image s. In this work, the *context* of an observation refers to a version of it with one or more noisy, or corrupted regions. The pixels can be recovered by propagating local features and capturing the overall structure of the context. The discrepancy between the reconstructed image and the actual image then serves as the intrinsic reward.

We found that reconstructing an observation given its context is more robust to random perturbations or small changes in the environment, and helps to capture important visual features (see Section 5.1.2). Moreover, our formulation bypasses the need of predicting the next observation given the current observation and the agent's action [44], which tends to attract the agent to stochastic transitions to maximize the prediction error [12]. In contrast with such stochastic approaches, we propose a curiosity measure where the reconstruction is a deterministic problem.

4.1.1 Context creation

We introduce two methods to artificially extract the *context* of an observation that we present below.

- Downsample Context: The original image of size $w \times w$ is downscaled to a smaller image of size $\frac{w}{K} \times \frac{w}{K}$ using a nearest-neighbor interpolation [21] and then upscaled to the original size, introducing small artifacts. The hyperparameter K controls the amount of artifacts. As an intuition, it can be viewed as a simplified version of the image.
- Noisy Context: The observation is augmented with a region of white noise which makes $K \times K$ pixels. The white noise region position changes for each observation to improve robustness. Theoretically, randomly choosing the position would be a good choice, however, in practice this solution is less than ideal in some cases (e.g. the region falls on the background). Empirically, we found that a random position within a radius $K/2$ from the center of the frame works well as a robust substitute.

An example is shown in Fig. 1. When using downsample context creation, the reconstructor network attempts to reconstruct the detailed content of the original frame (Fig. 1a) given its blurry / corrupted context (Fig. 1b). When using noisy context creation, the reconstructor network aims to infer the original pixel values of a large noisy region (Fig. 1a) based on the context of the surrounding pixels (Fig. 1c).

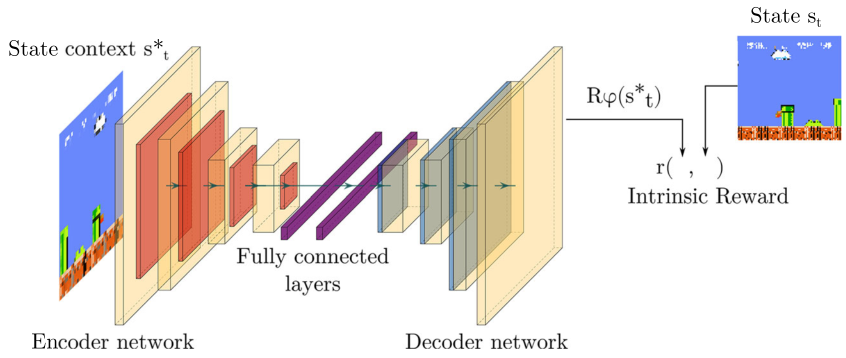


Fig. 2 The reconstructor R_θ architecture. The image's context is passed through an encoder network and the features all around the feature map are connected using fully connected layers. The reconstruction-based curiosity reward is calculated based on the discrepancy between the original image and the reconstructed image

4.1.2 Reward calculation

In this section we formulate the procedure to calculate reconstruction-based intrinsic reward. This involves the prediction error of a *reconstructor* network trained to reconstruct an observation given as input the observation's context.

Formally, let s_t be the original observation at time t and s_t^* its context. The reconstructor network, $R_\theta : s^* \mapsto s$, takes the context of the observation and reconstructs the original image s_t. We denote the reconstructed image as $\hat{s}_t$. The parametrization of R_θ is discussed further in the next section. The reconstruction process can be formulated as:

$$\hat{s}_t = R_\theta(s_t^*) \tag{7}$$

This reconstruction will have some errors that can be measured using a distance function such as the euclidean distance. However, empirically we found that the euclidean distance is ineffective to distinguish noisy data (such as contexts) from the original images. Instead, we propose to embrace SSIM metric [62], a popular technique used in the field of computer vision for comparing the structural similarity of two images. Our novelty measure is based on the single-scale SSIM metric that compares corresponding pixels and their neighborhoods in two images with three metrics: luminance, contrast, and structure. The reconstruction error $e(s_t, \hat{s}_t)$ can therefore be expressed via:

$$e(s_t, \hat{s}_t) = 1.0 - \left[\frac{1}{P} \sum_{i=1}^{P} L(s_t^i, \hat{s}_t^i)\Gamma(s_t^i, \hat{s}_t^i)S(s_t^i, \hat{s}_t^i) \right] \tag{8}$$

where s_t^i and $\hat{s}_t^i$ are the i^{th} sliding patch over s_t and $\hat{s}_t$, respectively, and P is the number of patchs per image. The luminance (L), contrast (Γ), and structure (S) can be computed as follows:

$$L(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \quad \Gamma(x, y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2} \quad S(x, y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \tag{9}$$

where μ_x, μ_y, σ_x, σ_y denote the mean pixel intensity and the standard deviations of pixel intensity of two image patchs x and y. Following this work [62], we use a square neighborhood of 5×5 pixels resulting in patches of size 11×11 pixels. σ_{xy} is the sample correlation coefficient between corresponding pixels in the two patches and C_1, C_2, C_3 are small constants for numerical stability.

When using reconstruction error as intrinsic reward, the reward function is non-stationary and its scale can fluctuate greatly between different points in time. In order to keep a consistent scale of the intrinsic rewards, it is useful to normalize them. This can be achieved by dividing the intrinsic rewards by a running estimate of the standard deviations of the sum of discounted intrinsic rewards $\mathcal{R}_{ib}$ [12]. We can now assign a curiosity bonus r^{ib} as:

$$r^{ib}(s_t) = \left[\frac{e(s_t, R_\theta(s_t^*))}{\sigma(\mathcal{R}_{ib})} \right] \tag{10}$$

This approach is motivated by the idea that, as our ability to model the dynamics and salient features of a particular state improves, the agent has a better understanding and hence the intrinsic motivation is lower. We show in Section 4.3 how the reconstruction-based curiosity method can be used for estimating fast and slow rewards.

4.2 Image reconstruction architecture

In this section we consider the task of reconstructing an observation given its context. We propose two different neural network architectures inspired by autoencoders (denoted by Cb-AE and Cb-VAE) to learn $\hat{s} = R_\theta(s^*)$. They both consist of a number of convolutional and deconvolutional layers for reconstructing the original image.

4.2.1 Context-based autoencoder

Given the context of an observation, the most direct way to reconstruct the original observation is to train an autoencoder (AE). However, autoencoder architectures cannot propagate information from one part of the feature map to another. This is because the encoder network directly feeds feature map to the decoder network and convolutional layers never connect all the locations together within a feature map. To meet this need, we introduce multiple fully connected layers between the encoder network and the decoder network (Cb-AE)(Fig. 2) like done in [13]. By connecting the features altogether, we can except to enable the propagation of information for all locations across the image and hence improve image reconstruction. As an intuition, when a pixel is missing or corrupted, only using the pixel's neighborhoods only enables to capture local geometric features, whereas our architecture (Cb-AE) can capture the general appearance structure of images. Therefore, pixels are reconstructed to make the overall prediction look more real. Please note that unlike autoencoders, the input image is different from the target image that alleviates the need to have a small middle layer.

During training, we further propose a novel context-based loss function $\mathcal{L}_{cb}$ that captures the overall structure of the observations. The objective is to favor the reconstruction of regions that are noisy or corrupted, and hence require more attention. Given the context s^* of a state s, the reconstructor network R_θ generates an output $R_\theta(s^*)$ and is trained to minimize the following loss function:

$$\mathcal{L}_{cb}(s, s^*) = \|(s - R_\theta(s^*)) \odot (1 - N)\|_2 \tag{11}$$

where N is a mask with values between 0 and 1 of the same size as the reconstructed image. The mask is automatically generated based on the type of context. When using the *downsample* context, all the pixels are corrupted and hence the mask is filled with 0. When the image is filled with white noise (i.e. *noisy* context), the mask is filled with +1 for the pixels not corrupted and 0 for the corresponding pixels within the noisy area.

4.2.2 Context-based variational autoencoder

We propose context-based variational autoencoder (Cb-VAE) inspired by variational autoencoders (VAEs). VAEs are known to give representations with disentangled factors [23] due to gaussian priors on the latent variables. They have been shown to be more scalable to large datasets and to capture better representations than AEs. Cb-VAE resembles a standard autoencoder, using an encoder network q_ϕ that outputs a latent variable z given the image's context s^*, and then a decoder network p_ψ that maps z to the original image s. Its parameters are optimized by maximizing a variational lower bound on the likelihood function:

$$\mathcal{L}(\psi, \phi; s, s^*) = \tau D_{KL}(q_\phi(z|s^*)||p(z)) - \mathbb{E}_{q_\phi(z|s)}\left[\log p_\psi(s|z)\right] \quad (12)$$

where the first term is a regularizer, the Kullback-Leibler divergence between the encoder distribution $q_\phi(z|s^*)$ and $p(z)$, and the second term is equivalent to a l_2 loss. However, VAEs suffer from the effect of l_2 loss and therefore are prone to generate blurry images. We observe that it can be written as:

$$\mathcal{L}(\psi, \phi; s, s^*) = \tau D_{KL}(q_\phi(z|s^*)||p(z)) frac12\sigma^2 \sum_{i=1}^{N}(s^i - f_\psi(z^{(i)}))^2 \quad (13)$$

where $f_\psi(z^{(i)})$ is computed by the decoder network. Thus, we can rewrite this loss function to integrate any differential loss $g(s, \hat{s})$ that better captures small details and generates more sharp images (see Section 5.1.3). The loss function that Cb-VAE optimizes then becomes:

$$\mathcal{L}(\psi, \phi; s, s^*) = \tau D_{KL}(q_\phi(z|s^*)||p(z)) + \rho \cdot \mathbb{E}_{q_\phi(z|s)}\left[g(s, \hat{s})\right] \quad (14)$$

where τ and ρ are scalars to weight the two components of the loss function, and $\hat{s} = f_\psi(z)$. Empirically, we found that minimizing the loss related to the sum of structural similarity scores works well as a robust substitute to l_2 [52]:

$$g(s, \hat{s}) = -\sum_i \text{MS-SSIM}(s_i, \hat{s}_i) \quad (15)$$

where i is an index over scales and MS-SSIM is the multiscale SSIM [61]. While l_2 ignores the intricate characteristics of the human visual system, MS-SSIM is sensitive to changes in local structures and performs well on real-world images with different scales. For training the MS-SSIM loss, we use 5 scales.

Similarly to Cb-AE, we further modified the original architecture by adding multiple fully connected layers to connect features altogether.

4.3 Fast and slow rewards

Algorithm 1 Fast and Slow intrinsic curiosity (FaSo).

```
Given:
  –  an on-policy RL algorithm $\pi_{\theta_P}$                                  ▷ PPO, A2C
  –  a replay buffer $R$
  –  a fast context buffer $\Omega_f$ and a slow context buffer $\Omega_s$
  –  a fast model $R_{\theta_f}$ and a slow $R_{\theta_s}$ reconstructor model
  –  a context creation strategy                                    ▷ Noisy, Downsample
  –  context creation parameters $K_{fast}$ and $K_{slow}$
Initialize $\pi_{\theta_P}$, $R_{\theta_f}$ and $R_{\theta_s}$                  ▷ initialize neural networks
Initialize $R = \{\}$, $\Omega_f = \{\}$, and $\Omega_s = \{\}$
Initialize $\alpha = 0.5$ and $\beta = 0.5$
for m=0,...,M do
    for t=0,...,H-1 do
        Get action $a_t = \pi(s_t|\theta_P)$
        Execute $a_t$ and observe next state $s_{t+1}$
        Create fast $s^*_f$ and slow contexts $s^*_s$ given the current observation
        Calculate intrinsic reward $r^i_t = \alpha\left[\frac{e(s_t, R_{\theta_f}(s^*_f))}{\sigma(\mathcal{R}^f_{ib})}\right] + \beta\left[\frac{e(s_t, R_{\theta_s}(s^*_s))}{\sigma(\mathcal{R}^s_{ib})}\right]$
        Store transition $R = R \cup (s_t, a_t, s_{t+1}, r^e_t, r^i_t)$
        Store $s^*_f$ in $\Omega_f$ and $s^*_s$ in $\Omega_s$
        Update reward normalization parameters using $i_t$
    Normalize the intrinsic rewards contained in $R$
    Update the RL model on $R$
    Compute state diversity on $\Omega_f$, and $\Omega_s$
    Assign $\alpha$ and $\beta$ based on state diversity progress at time m and m+1
    Periodically fine-tune $R_{\theta_f}$
    Periodically fine-tune $R_{\theta_s}$
```

In this section we provide an algorithm built upon reconstruction-based curiosity to deal with high-level exploration. Although a reconstruction-based curiosity bonus can improve local exploration such as how to interact with a particular object or avoid a collision; global exploration is beyond the reach of a single curiosity-based reward. We found that the agent may get stuck in local optima or not receive enough intrinsic reward to promote long-time horizons strategies. This can happen after the novelty of a state has vanished, the agent is not encouraged to visit it again, regardless of its importance on a long-time horizon. To build intuition we consider a task wherein a robot has to reach a target location behind a door. There are multiple levers but only the combination of two of them can unlock this door. To use the correct levers, the agent must give up immediate rewards associated with easy-to-reach levers and explore more distant areas. Solving such a task requires a long-term exploration bonus to escape from sub-optimal policies by compensating the loss of easy immediate extrinsic reward (i.e. keep the key) and incentivizing useful decisions on a long-time horizon. One solution to the vanishing-curiosity issue could be to decrease the learning rate of the reconstructor network, however, it tends to make the agent's learning slow as the curiosity continuously encourages the revisit of familiar states (not all regions need to be revisited).

This paper introduces a different approach where the intrinsic bonus r_t^i is the combination of two distinct reconstruction-based rewards:

$$r_t^i = \alpha r_t^{fast} + \beta r_t^{slow} \tag{16}$$

where r_t^{fast} deals with local exploration and r_t^{slow} is the slow reconstruction-based reward that aims to push the agent to explore long-time horizon strategies. In detail, the fast context-based model quickly learns to reconstruct observations to assess the short-term novelty of each state. On the other hand, r_t^{slow} changes slowly and remains large to encourage deep exploration. The scalars α and β weight the fast and slow reconstruction-based rewards.

We now explain how r^{fast} and r^{slow} are calculated. They are estimated by two distinct reconstruction-based curiosity models which reconstruct the original observation s given its fast s_f^* and slow s_s^* contexts respectively, $R_{\theta_f} : s_f^* \mapsto s$ and $R_{\theta_s} : s_s^* \mapsto s$ (see Fig. 3). They are parameterized by a set of trainable parameters θ_f and θ_s. The key difference is how to generate frame contexts, s_f^* and s_s^*, to achieve exploration behaviors with different ranges of time horizons. Let K_{fast} the parameter used to create the fast contexts, we typically used $K_{fast} \times 2$ or $K_{fast} \times 4$ to create the slow contexts - slow contexts are more corrupted. Slow reward acts as a global exploration mechanism in two ways. 1) As the reconstruction task becomes more challenging when large regions of images are noisy or corrupted (i.e. slow contexts), slow reward remains large even with further state visitations. 2) The reconstruction error is maximized only in states that induce a significant shift in the state distribution (e.g. visiting a new room), driving the agent to seek out such richer and novel regions (i.e. global exploration). Please note that since slow contexts have a more blurry representation of states, thus, the reconstructor network will not see fine-grained local differences and will drive the agent to further explore novel regions. In contrast, contexts of fast rewards are nearly unique which entails that slightly deviating from previous policies - visiting novel states, or revisiting surprising states is sufficient to significantly increase the reconstruction errors; encouraging to locally explore the environment. Therefore, such a reward enables fast learning by encouraging local exploration but swiftly downmodulates states that become more familiar across episodes.

Thus, the overall intrinsic reward provided by FaSo is calculated as:

$$r_t^i = \alpha \left[\frac{e(s_t, R_{\theta_f}(s_f^*))}{\sigma(\mathcal{R}_{ib}^f)} \right] + \beta \left[\frac{e(s_t, R_{\theta_s}(s_s^*))}{\sigma(\mathcal{R}_{ib}^s)} \right] \tag{17}$$

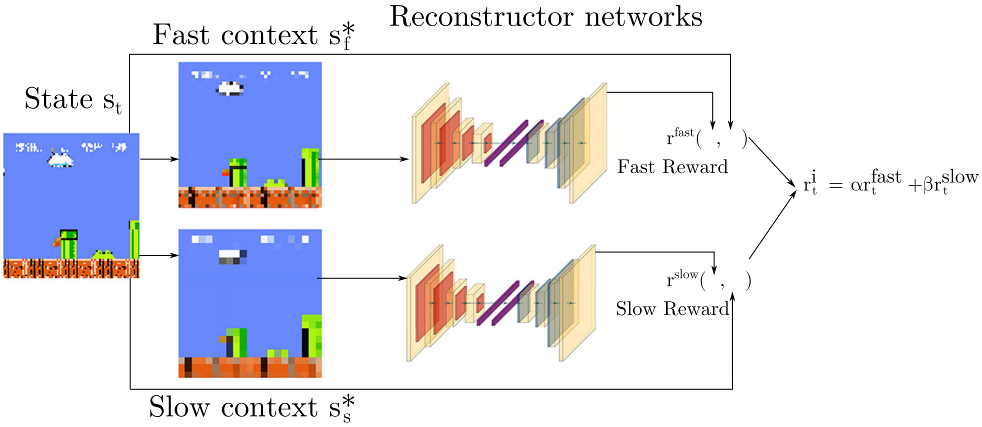


Fig. 3 Fast and slow exploration model architecture

where e is the reconstruction error defined in (8). Algorithm 1 provides an outline of the basic training loop.

One problem is how to select α and β to modulate local and global exploration strategies. We found that a fixed value can be difficult to tune and not ideal. Instead, we propose an adaptive solution to weight the fast and slow intrinsic rewards based on the idea of *state diversity*. At the end of each epoch α and β are selected. We compared two methodologies depending of the choice of the reconstructor network architecture:

1. **Context-based autoencoder (Cb-AE):** state diversity can be estimated by maintaining an episodic memory of the last experienced contexts, and measuring diversity progress between the memory and the memory augmented with the new observation's contexts. Let $\Omega = \{s_0^*, s_1^*, ..., s_N^*\}$ an episodic memory of the last contexts. We can estimate pairwise dissimilarities $\overline{d}$ among the contexts as follows:

$$\overline{d} = \frac{1}{|\Omega|(|\Omega|-1)} \sum_{i=0}^{N} \sum_{\substack{j=0 \\ i \neq j}}^{N} ||s_i^* - s_j^*||_2 \tag{18}$$

State diversity progress can be measured as the difference between pairwise dissimilarities at time $t+1$ and t.

$$d = \frac{\overline{d}_{t+1} - \overline{d}_t}{\sigma(D)} \tag{19}$$

where $\sigma(D)$ is an estimation of the standard deviations of the state diversity progresses. We then clip d to be between 0 and 1 and the new experience are added to the memory bank. When the capacity is exceeded a random element is substituted in memory with the current element. The scaling factors α and β are independently calculated based on their dedicated episodic memory and their value is equal to respectively d_{fast} and d_{slow}.

2. **Context-based variational autoencoder (Cb-VAE):** we indirectly estimate the state diversity by measuring the quality of the Cb-VAE for encoding the latent representations in the data. We assume that an observation's context s^* is generated by a random latent process z. Learning progress can be measured by measuring the distance between the posterior distribution $p(z|s^*)$ after experiencing new observation' contexts and the prior $p(z)$. Several measures such as KL divergence can be used. However, computing the posterior distribution $p(z|s^*)$ is often intractable. Instead, we propose to use Cb-VAE to model the approximate posterior $q_\phi(z|s^*)$.

Let $\Omega = \{s_0^*, s_1^*, ..., s_N^*\}$ a set of observation's contexts, we define the state diversity for the fast and slow models as:

$$d = 1.0 - \left[\frac{1}{N} \sum_{i=0}^{N} D_{KL}(q_\phi(z|s^{*(i)})||p(z)) \right] \tag{20}$$

To ensure state-diversity measure to adapt over time, Ω is filled with contexts experienced during the last episodes and contexts collected by executing a random policy. The diversity progress is independently measured for fast and slow rewards, $\alpha = d_{fast}$, $\beta = d_{slow}$ respectively; and we clip the values to be within [0.1,1.0]. Since state-diversity greatly decreases when more states become familiar, it is useful to normalize α and β by diving the state diversity by a running estimate of the standard deviations of the state diversity. By doing so, the agent seeks out to explore states as diverse as possible - this increases the distance between $p(z|s^*)$ and $p(z)$ in average, to receive large intrinsic rewards induced by high values of α and β.

5 Experiments

In this section, we present the experimental results on a variety of environments including Minigrid, Super Mario Bros, and Atari games (Fig. 4). In Minigrid, we consider three types of hard exploration tasks: Door & Key, KeyCorridor, and Multiroom. We first present an ablation analysis of FaSo. Second, we test FaSo trained on a fixed and randomly generated mazes (Multiroom). Third, we evaluate the performance of our model on Super Mario Bros in the absence of any extrinsic reward signal. Fourth, we measure the performance of FaSo on dense reward environments. Finally, we compare the proposed algorithm with the previous curiosity-based approaches on five Atari

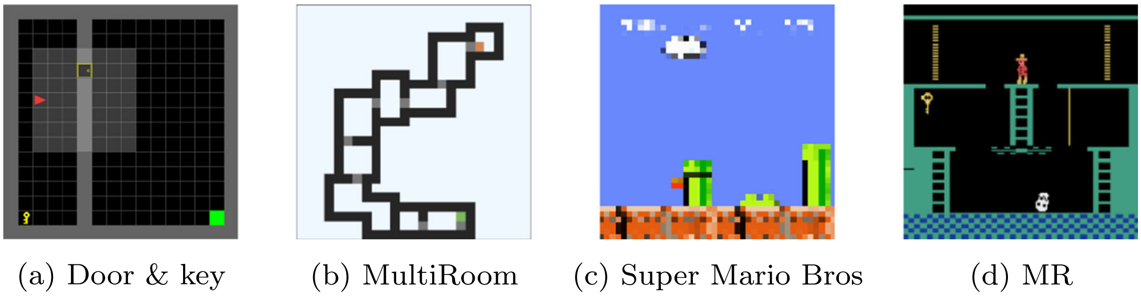

(a) Door & key (b) MultiRoom (c) Super Mario Bros (d) MR

Fig. 4 Frames from Door & Key, MultiRoom, Super Mario Bros, and Montezuma's Revenge (MR)

games, Door & Key, and Super Mario Bros, combining intrinsic rewards with sparse or deceptive extrinsic rewards.

Implementation Details. In all the experiments the observations are given in the form of images. The RGB images are converted to 84×84 grayscale images. The input given to the policy network consists of the current observation concatenated with the previous three frames. We set the end of an episode to when the game ends. As our policy learning method, we use PPO with similar hyperparameters as in the original implementation of RND [11], but 32 learners. The output of the last convolutional layer is fed into a fully connected layer with 256 units. It is followed by two separate fully connected layers of size 448, used to predict the value function of each reward component (extrinsic and intrinsic advantage).

In order to find the hyperparameters for our method, we ran grid searches over the context creation techniques as well as K_{fast} and K_{slow}, which control the degree of fast and slow contexts. On Minigrid, we select Cb-AE as the reconstructor network architecture and *downsample* contexts created with $K_{fast} = 2$ and $K_{slow} = 4$. On Super Mario Bros, we train a Cb-VAE to reconstruct the observations given their *downsample* contexts ($K_{fast} = 2$, $K_{slow} = 5$). For Atari games, we compare our method trained with *noisy* contexts ($K_{fast} = 24$, $K_{slow} = 46$) with baselines. We set the coefficients $\tau = 0.5$ and $\rho = 0.5$. The value of the memory size, N, is 80 for FaSo(Cb-AE), and 200 for FaSo(Cb-VAE). We estimate the sum of discounted intrinsic rewards using a discount factor of 0.99 on rollouts of length 128. $\sigma(\mathcal{R}_{ib})$ (10) is defined as the running average of the standard deviations of these discounted intrinsic rewards. The architectures of Cb-AE and Cb-VAE consist of a sequence of four convolutional layers with 32 filters each, stride: 2,1,1, kernel size of 3 × 3, and padding 1. We apply a rectifier non-linearity after each convolutional layer. The output of the last convolutional layer is passed to a serie of two fully connected layers of size 256. The last layers are the corresponding decoding layers. For online training of the reconstructor networks, we store the experience and make 5 epochs of training every 16K time steps. We found that retraining the slow reconstructor network R_{θ_s} once every 48K steps is sufficient. Note that we decouple the training speed of the reconstructor models to ensure that the models achieve the desired behaviors (fast learning or slow learning of the reconstruction task). We observed that decoupling the training speed is only useful in visually simple tasks such as Door & Key. Training is carried out with a fixed learning rate of 0.0002 using the Adam optimizer [30], with a batch size of 256.

Environments We conduct experiments on several sparse reward environments:

– Minigrid: In Minigrid [14], the world is a partially observable grid. Each tile in the grid contains nothing or one object: ball, box, door, wall, or door. An observation consists of the visible cells surrounding the agent. The agent can choose among 7 possible actions: turn left, turn right, move forward, pick up an object, drop the object being carried, open a door and complete the task. The Door & Key task consists of two rooms connected by a door. The agent has to pick up the key in order to unlock the door and then get to the goal. In KeyCorridor, the task is similar to Door & Key but there are multiple rooms and multiple doors. In Multiroom, the agent has to open a serie of doors to reach the final goal. Solving such sparse tasks is challenging since the object locations are randomized and the agent only receives a positive reward +1 when it reaches the final goal. These tasks also require sequential decision making (e.g. saving the key to open a distant door) to reach the final goal.
– Super Mario Bros: We consider the first level of the infamous Nintendo game Super Mario Bros [29]. Initially, this game is played using a joystick which requires pressing simultaneously multiple buttons. In our implementation, each combination of buttons is mapped to a unique action resulting in 12 possible actions. Solving this game requires complex exploration strategies and involves sparse rewards (+10 when a flag is reached, -10 when the agent is killed, and 0 otherwise). Other challenges stem from the extremely long delay before reaching the final goal.
– Atari: We conduct experiments in the Arcade Learning Environment [8]. We selected five hard exploration games. They might also contain deceptive rewards such as in Pitfall. In these games, the agent has to navigate in complex environments, explore labyrinths, avoid enemies, pass obstacles, or collect objects. In Montezuma's Revenge the player explores rooms filled with enemies, obstacles, traps, in an underground labyrinth. In pitfall, in order to recover 32 treasures, the player must maneuver through numerous hazards, including pits, quicksand, and rolling logs. In gravitar and seaquest, the agent must shoot enemies to survive. Private eye is an investigation game in which the agent must search the city for a specific clue to the crime and for the object stolen in the crime. They are known for their challenging dynamics, complex visual patterns, and sparse extrinsic rewards.

5.1 Ablation study

We have conducted ablation studies for all the three sets of tasks (Door & key 16×16, Super Mario Bros sparse, Montezuma's Revenge) to investigate: (1) the impact of a fast/slow reward decomposition, (2) the effect of the choice of the context creation method, (3) the influence of MS-SSIM (Eq 15) on our method, (4) the effect of using adaptive scaling factors as part of the intrinsic reward, and (5) the performance of FaSo on “noisy-TV” tasks.

5.1.1 Fast and slow reward decomposition

As described in Section 4, our method decomposes the curiosity reward into a *fast* reward and a *slow* reward. To isolate how much each reward contributes to our method, we show in Table 1 the performance of FaSo trained only with *fast* rewards (‘PPO+Fa(Cb-AE)’,‘PPO+Fa(Cb-VAE)’) and only with *slow* rewards (‘PPO+So(Cb-AE)’,‘PPO+So(Cb-VAE)’). In Table 1 we see that methods trained only with *fast* rewards tend to learn faster during early training, but are outperformed later by models guided using *slow* rewards. However, we found that only a slow reward model tends to provide similar rewards for the observations of a same region of the state space, resulting in an incomplete exploration. We also observe that PPO+FaSo achieves the highest performance which indicates that it takes advantage of the two reward streams to discover more efficient exploration strategies.

The advantages of a fast and slow reward decomposition become more noticeable in environments with sparse rewards, or harder exploration such as Montezuma’s revenge. Similarly to RND [11], our agents trained using a single intrinsic reward stream cannot explore all the rooms on Montezuma’s revenge. In the first stage, the agent has to pick keys and open two doors. Without long-time exploration, baseline models open the first easy doors to receive the associated rewards and therefore fail (i.e. local optima). On the other hand, *slow rewards* compensate for the loss of immediate reward (no door are open) to let the agent try more global exploration behaviors (e.g. save the keys for later). In Pitfall, similar behaviors can be observed: *slow* rewards encourage the agent to travel through tunnels, useful to later collect treasures (positive rewards.) Since tunnels are hard to explore and contain a lot of objects resulting in negative/deceptive rewards, baselines tend to stay in the jungle but cannot finish the level. *Slow rewards* balance negative rewards during early exploration to incentivize strategies that may result in larger rewards on a long-time horizon scale.

5.1.2 Choice of the context creation method

To see the potential benefits of using observations’ contexts rather original frames, we explore the performance of FaSo(Cb-VAE) with contexts created using the following

Table 1 Ablative performance comparisons on Door & key 16×16, Super Mario Bros sparse, and Montezuma’s Revenge

	Door & key 16×16			Super Mario Bros sparse			Montezuma’s Revenge		
Method	2M	6M	15M	2M	6M	10M	50M	250M	500M
PPO+Fa(Cb-AE)	0.03±0.08	0.88±0.15	0.92±0.05	0.30±0.08	0.87±0.07	0.89±0.03	4,523±252	8,537±189	8,785±341
PPO+So(Cb-AE)	0.02±0.05	0.65±0.21	0.93±0.06	0.21±0.05	0.82±0.06	0.81±0.05	2,876±345	6,123±512	8,823±472
PPO+Fa(Cb-VAE)	0.02±0.10	0.75±0.18	0.86±0.08	0.35±0.09	0.90±0.03	0.88±0.05	5,524±157	9,762±212	9,340±464
PPO+So(Cb-VAE)	0.01±0.06	0.61±0.18	0.98±0.04	0.18±0.07	0.79±0.07	0.74±0.04	3,921±415	8,311±475	9,025±378
PPO+FaSo(downsample)	0.02±0.07	0.79±0.17	0.96±0.04	0.24±0.03	0.83±0.06	0.93±0.05	3.245±165	6.628±200	8.878±622
PPO+FaSo(noisy)	0.03±0.08	0.76±0.15	0.93±0.03	0.27±0.06	0.80±0.07	0.91±0.06	3,278±180	6,897±225	9,651±442
PPO+FaSo(original)	0.03±0.05	0.55±0.12	0.78±0.07	0.12±0.03	0.66±0.10	0.57±0.08	2,128±450	3,287±511	4,425±709
PPO+FaSo(random)	0.04±0.10	0.64±0.23	0.85±0.09	0.11±0.08	0.57±0.09	0.76±0.08	2,876±237	5,450±487	6,973±904
PPO+FaSo(noisy)	0.03±0.08	0.76±0.15	0.93±0.03	0.27±0.06	0.80±0.07	0.91±0.06	3,278±180	6,897±225	9,651±442
PPO+FaSo(fixed)	0.03±0.04	0.71±0.10	0.90±0.05	0.22±0.05	0.73±0.07	0.85±0.05	3,043±340	5,394±412	8,036±667
PPO+(Cb-AE/$\mathcal{L}_{cb}$)	0.02±0.07	0.79±0.17	0.96±0.06	0.24±0.03	0.83±0.06	0.93±0.05	3,278±180	6,897±225	9,651±442
PPO+FaSo(Cb-VAE/MS-SSIM)	0.01±0.09	0.64±0.15	0.91±0.08	0.30±0.04	0.96±0.05	0.97±0.04	5,325±208	9,363±345	11,466±584
FaSo+Cb-AE(MSE)	0.01±0.12	0.66±0.13	0.92±0.06	0.26±0.06	0.64±0.08	0.54±0.13	2,748±165	3,846±303	5,025±687
FaSo+Cb-VAE(MSE)	0.02±0.07	0.75±0.16	0.97±0.05	0.25±0.04	0.68±0.06	0.87±0.06	3,142±415	7,424±402	6,560±457
PPO+FaSo(Cb-AE)	0.02±0.07	0.79±0.17	0.96±0.06	0.24±0.03	0.83±0.06	0.93±0.05	3,278±180	6,897±225	9,651±442
PPO+FaSo(Cb-VAE)	0.01±0.08	0.64±0.15	0.91±0.04	0.30±0.04	0.96±0.05	0.97±0.04	5,325±208	9,363±345	11,466±584

Averages over 10 trials are reported at different timesteps (in millions M). We report scores (mean±std) for each component of the proposed method (line 1-4), different choices of context creation method (line 5-7), different strategies to choose the position of noisy contexts (line 8-10), various predictor network loss functions (line 11-14), and the overall method (line 15-16 strategies: downsample context, noisy context, original context (i.e. the context is the original observation, $s^* = s$) (Table 1). The agents trained with the original context method perform poorly. This behavior is expected since fast contexts and slow contexts are identical. It might also be related to the fact that PPO+FaSo(original) is more affected by small changes in the environment. However, for the other agents, using a context creation method (downsample or noisy) greatly improves the performance.

When using noisy contexts, another question that arises is how to choose the noise position. We experimentally evaluate different strategies for choosing the position of the white noise region. The choice of the position has a relative importance on the intrinsic bonus and the state diversity. For example, if the noisy region falls on the background or on relevant regions of the images, the reconstruction task may become easier or more difficult.

So far, the location is randomly chosen around the center of the frame (see Section 4.1.1). Apart from it we consider the following strategies:

- full random: the location is randomly chosen within the frame without any constraint.
- fixed: the white noise region is fixed and located in the center of the observation.

We report the performance (mean±std) of the proposed method trained with noisy contexts where the noise region is: randomly selected (random), within a radius from the center $K/2$ (noisy), and fixed (fixed). As shown in Table 1 (line 8-10), FaSo(noisy) learns considerably faster and better than the models trained with the other strategies. We also observe that FaSo(random) is more prone to outlier and more unstable than FaSo(noisy). On the other hand, reconstructing the image from fixed noisy contexts tends to not capture the novelty of states where small changes are located in the center of it.

5.1.3 Predictor network loss function comparison

FaSo relies on two reconstructor networks. One legitimate question is to study the impact of the choice of loss function on the performance. To answer this question, we keep other components the same and only change the loss function during training. As shown in Table 1, the agents trained using 'FaSo+Cb-VAE(MS-SSIM)'and

Table 2 Final mean score (± std) of our method with various scaling strategies of rewards on Atari games and average success rate (± std) on Door & Key 16×16 and Super Mario Bros

	Maximum Mean Score (at convergence)					Success rate	
Method	Montezuma's Revenge	Private Eye	Gravitar	Pitfall	Seaquest	Door & Key	Super Mario Bros
FaSo (Cb-AE) / N = 50	9,711±587	11,807±754	3,812±325	234±24	3,120±270	0.95 ±0.07	0.92 ± 0.04
FaSo (Cb-AE) / N = 80	9,651±442	13,423±775	3,656±280	247±28	4,989±311	0.96 ±0.06	0.93 ± 0.06
FaSo (Cb-AE) / N = 100	7,245±371	14,008±637	3,658±254	15±8	5,244±327	0.94±0.06	0.95 ± 0.07
FaSo (Cb-AE) / N = 150	6,388±356	13,996±588	3,125±266	-5±1	4,584±386	0.90±0.07	0.74 ± 0.12
FaSo (Cb-AE) / N = 200	6,461±425	11,652±752	2,718±303	2±2	4,256±297	0.84 ± 0.13	0.68 ± 0.11
[5pt] FaSo (Cb-VAE) / N = 50	8,625±440	13,325±862	3,001±444	70±7	4,125 ± 645	0.72 ± 0.16	0.71 ± 0.12
FaSo (Cb-VAE) / N = 80	9,121±512	13,311±743	3,257±396	65±10	4,659 ± 587	0.77 ± 0.13	0.77 ± 0.10
FaSo (Cb-VAE) / N = 100	10,998±436	15,010±754	3,389±352	71±6	5,027 ± 463	0.82 ± 0.11	0.92 ± 0.08
FaSo (Cb-VAE) / N = 150	11,895±487	15,994±712	3,715±318	180±12	4,826 ± 327	0.90 ± 0.06	0.97 ± 0.05
FaSo (Cb-VAE) / N = 200	11,466±584	16,135±688	3,431±325	189±17	5,123±251	0.91 ± 0.04	0.97 ± 0.04
FaSo (Cb-AE) / (Schedule 1)	5,430 ± 462	11,927 ± 711	2,734 ± 523	-1 ± 4	3,871 ± 401	0.90 ± 0.08	0.71 ± 0.09
FaSo (Cb-VAE) / (Schedule 1)	8,046 ± 613	13,780 ± 499	2,508 ± 682	-3 ± 3	4,024 ± 455	0.86 ± 0.08	0.66 ± 0.08
FaSo (Cb-AE) / (Schedule 2)	5,712 ± 587	12,489 ± 539	2,115 ± 713	10 ± 6	3,915 ± 473	0.80 ± 0.07	0.72± 0.13
FaSo (Cb-VAE) / (Schedule 2)	7,369 ± 632	12,647 ± 500	2,828 ± 654	26 ± 12	4,687 ± 601	0.86 ± 0.11	0.69 ± 0.10
FaSo (Cb-AE) / ($\alpha = 0.5$, $\beta = 0.5$)	6,251±398	11,684±542	3,213±412	12±2	4,182±221	0.85 ± 0.06	0.75 ± 0.06
FaSo (Cb-VAE) / ($\alpha = 0.5$, $\beta = 0.5$)	8,750±467	12,459±493	3,312±338	85±6	4,701±186	0.82 ± 0.04	0.76 ± 0.08
FaSo (Cb-AE) / ($\alpha = 0.8$, $\beta = 0.2$)	7,465 ± 363	11,414 ± 522	3512 ± 564	15 ± 7	4,106 ± 328	0.82 ± 0.05	0.78 ± 0.04
FaSo (Cb-VAE) / ($\alpha = 0.8$, $\beta = 0.2$)	8,868 ± 484	11,988 ± 440	2532 ± 380	38 ± 8	4,789 ± 323	0.83 ± 0.06	0.81 ± 0.07
FaSo (Cb-AE) / ($\alpha = 0.2$, $\beta = 0.8$)	5,139 ± 526	8,234 ± 619	2234 ± 389	1 ± 2	3,401 ± 511	0.62 ± 0.15	0.65 ± 0.09
FaSo (Cb-VAE) / ($\alpha = 0.2$, $\beta = 0.8$)	5,340 ± 438	9,030 ± 782	2646 ± 401	-1 ± 3	3,876 ± 499	0.64 ± 0.10	0.70 ± 0.12

Averages over 10 runs are shown after 500M steps (Atari), 15M steps (Door & Key), and 10M steps (Super Mario Bros)

'FaSo+Cb-AE($\mathcal{L}_{cb}$)'outperform 'FaSo+Cb-VAE(MSE)'and 'FaSo+Cb-AE(MSE)', respectively. MSE loss successfully handles visually simple tasks such as on Door & key, but gives unsatisfactory performance in more complex domains such as Atari games. We conjecture that small details cannot be captured by the MSE loss and therefore MS-SSIM loss is more suitable for reconstructing complex frames.

5.1.4 Adaptive scaling of rewards

We aim to understand the effect of using adaptive scaling of rewards. We compare the performance of FaSo(Cb-AE) and FaSo(Cb-VAE) trained with fixed scaling of rewards ($\alpha = 0.5$, $\beta = 0.5$),($\alpha = 0.8$, $\beta = 0.2$),($\alpha = 0.2$, $\beta = 0.8$) and adaptive scaling factors. Adaptive scaling factors are obtained with N varying between 50 and 200. We also evaluate two schedule strategies that switch between fast ($\alpha = 0.8$, $\beta = 0.2$) and slow ($\alpha = 0.2$, $\beta = 0.8$) regimes. We report the performance of Faso trained with a slow (Schedule 1) and a frequent switch in regimes (Schedule 2). Schedule 1 switches in regimes every 10 epochs and Schedule 2 every 3 epochs.

As shown in Table 2, methods using adaptive factors perform significantly better in most of the tasks. On Gravitar, there is only little difference between the variants although the full model worked slightly better on average. On Montezuma's revenge, the effect of adaptive weights is more clear; the agent can discover more rooms and therefore achieves a higher score. We further observe that FaSo(Cb-AE), $N = 80$, and FaSo(Cb-VAE), $N = 200$, perform well in most of the games. Similar trends can be observed on Door & Key and Super Mario Bros as well. This experiment leads us to the conclusion that FaSo is reasonably robust to the choice of N. In Private Eye and Door & Key, we found that a frequent switch in the regimes improves the performance compared to fixed scaling factors. However, these parameters can be difficult to tune in the absence of domain knowledge. Under this lens, having an adaptive scaling of rewards appears to be the best solution to trade-off local and global exploration strategies.

To further examine the importance of state-diversity automatic schedule, we plot the evolution of α and β across learning on Montezuma's Revenge. The results are plotted in Fig. 5. They show that α maintains a relatively stable value. Intuitively, exploring the state surrounding the agent enables sufficient state diversity. This is because their fast context is nearly unique, which entails that the diversity progress does not sharply decrease across many episodes. On the other hand, β tends to produce spikes only in states that significantly drift away from the known states (e.g. a new room, a new type of obstacle). Overall, we can observe a frequent switch in regimes of fast and slow dominant phases and that "fast exploration" phases tend to be longer.

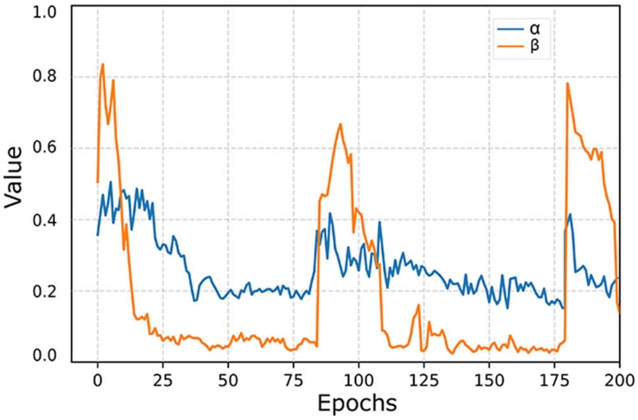


Fig. 5 Evolution of α and β across learning on the Montezuma's Revenge environment. We use Cb-VAE and downsample contexts to compute the coefficient values

Rather than fixed α and β, we argue that a switch in regimes is closer to how humans explore and learn.

5.1.5 Randomized environments

As pointed out by many authors [11, 48], agents that maximize the "surprise" - inability to predict the future, tend to suffer from the *TV noise* problem. For example, let us consider a curiosity formulation where the agent predicts the next observation given the current observation and agent's action. An agent maximizing this prediction error may seek out stochasticity (e.g. randomized transitions, high-frequency images) in the environment to maximize the error.

We now evaluate our method trained on randomized environments. We create versions of the Montezuma's Revenge environment with added sources of stochasticity. We test several settings:

- "Original": the original environment.
- "Noise": if the agent selects the action *jump*, a noise pattern (32×32) is displayed on the lower right of the observation - TV screen. The noise is sampled from [0,255] independently for each pixel.
- "Noise Action": if the agent selects the action *jump*, with a probability $\varrho \in \{0.05, 0.10\}$, the action performed by the agent is uniformly sampled among the possible actions.

In almost all cases, the performance of all methods deteriorates due to the stochasticity (Fig. 3). Nevertheless, our method is reasonably robust to randomized transitions (i.e. noise action $\varrho = 0.05$ and noise action $\varrho = 0.10$). We observed that ICM gets stuck in local optima - the ICM agent frequently uses the action *jump* to maximize the prediction error. On the other hand, our formulation does not rely on the agent's action and consequently is more robust to stochastic transitions. The scores for PPO+FaSo

Table 3 Average reward in the randomized-TV versions of Montezuma's Revenge (mean±std)

	Maximum Mean Score (at convergence)			
Method	Original	Noise	Noise Action $\varrho = 0.05$	Noise Action $\varrho = 0.10$
RND [11]	8,152±653	3,642±902	6,224±647	5,824±733
PPO+EC [48]	8,025±770	**4,008±823**	7,160±845	6,860±862
PPO+ICM [44]	329±118	125±106	78±40	56±74
PPO+FaSo (Cb-AE)	9,651±442	3,854±779	8,734±611	7,487±884
PPO+FaSo (Cb-VAE)	**11,466±584**	3,708±806	**9,965±599**	**8,609±740**

Results are average over 25 random seeds after 10M timesteps of training without seed tuning

Bold values indicate the best performing method

(Cb-AE) and PPO+FaSo (Cb-VAE) are significantly higher compared to the baselines as indicated by paired t-tests at 95% confidence level ($p < 0.002$).

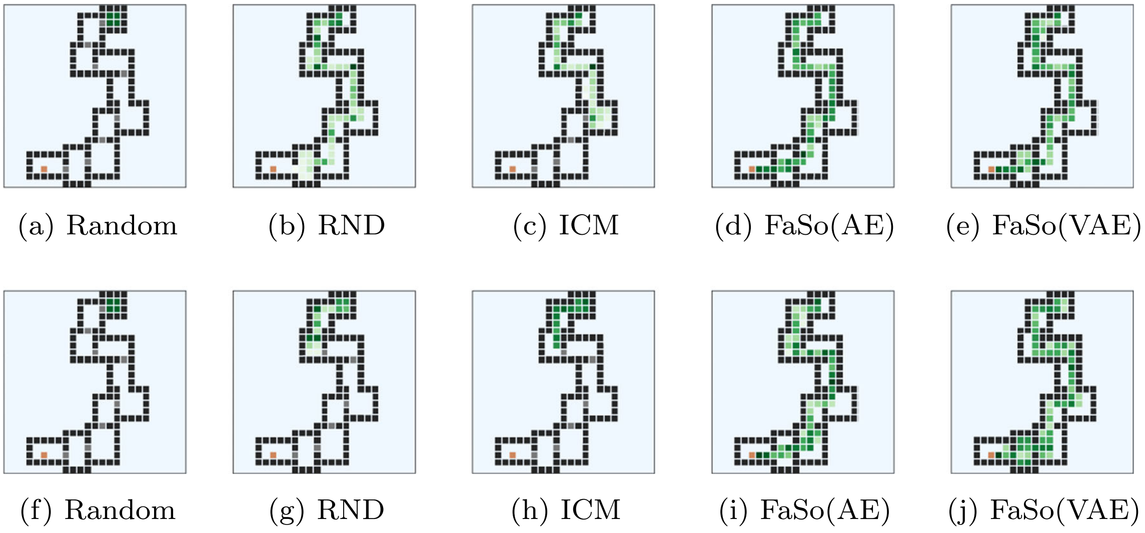
When adding visual noise to the environment, the performance of FaSo appears to deteriorate more. It is quite likely that visiting a state with a noise pattern produces constantly reward for such an area. That said, further analysis found that slow rewards may be large enough to escape from local optima by incentivizing the agent to try other actions. This observation motivated the use of formulations that quantify the relative improvement of the reconstruction, rather than its absolute error, but we leave it to future work to explore this direction further.

5.2 Fixed versus randomly generated environments

In this experiment we aim to investigate the ability of our agent to learn from randomly generated environments and generalize to unseen views or appearances. We use MultiRoom tasks from the Minigrid [14] domain. We compare PPO+FaSo(AE) and PPO+FaSo(VAE) with three baselines: random agent, RND [11], and ICM [44].

Figure 6 shows state visitation heatmaps on fixed (top row) and randomly generated (bottom row) mazes. We found that a random agent can only explore the first room. We also observe that ICM gets trapped in local optima in both scenarios. However, our approach discovers a large number of rooms when trained from fixed or randomly generated mazes. When trained on randomly generated mazes, existing methods are exploring much less efficiently, resulting in a poor state coverage.

To further evaluate the robustness of FaSo to random perturbations, we report in Table 4 the average success rate of agents trained on more environments (fixed and randomly generated) from the Minigrid domain. In all tasks, we observe that the proposed methods considerably outperforms the baselines approaches. The results further suggest that FaSo enables better generalization across the environments and is less distracted by small details that change from on environment to another.


(a) Random (b) RND (c) ICM (d) FaSo(AE) (e) FaSo(VAE)

(f) Random (g) RND (h) ICM (i) FaSo(AE) (j) FaSo(VAE)

Fig. 6 State visitation heatmaps averaged over 10 runs for different models: random, RND, ICM, FaSo(Cb-AE), and FaSo(Cb-VAE). We trained the models for 40m frames on a fixed maze (top row) and on randomly generated mazes (bottom row) in MultiRoomN10

Table 4 Average success rate on fixed and randomly generated tasks from the Minigrid domain. The results are averaged over 100 runs after 40 millions training steps

	Fixed			Random		
Method	MultiRoomN10	Door&Key 16×16	KeyCorridorS6R3	MultiRoomN10	Door&Key 16×16	KeyCorridorS6R3
RND	51±1.1	97±0.6	62±0.7	0±3.7	92±4.7	30±7.1
ICM	18±0.3	65±0.8	23±1.2	0±2.1	3±1.2	21±5.6
PPO+FaSo(Cb-AE)	**91±0.8**	98±0.5	88±0.9	87±2.9	**97±3.5**	81±3.5
PPO+FaSo(Cb-VAE)	89±0.6	**99±0.2**	**94±1.2**	**88±3.8**	96±2.9	**90±3.6**

Bold values indicate the best performing method

5.3 No extrinsic reward

For testing the good exploration coverage of our method, we trained our agent on Super Mario Bros without any reward from the environment. Our agent only receives a curiosity-based signal to reinforce its policy. As can be seen in Fig. 7, in order to remain curious the agent is pushed to explore distant regions of the state space, which entails that its coverage increases over time. It highlights that in the absence of extrinsic rewards, FaSo provides enough indirect supervision exploration signal for learning useful behaviors. We found a statistically significant difference between PPO+FaSo(Cb-VAE) and PPO+FaSo(Cb-AE) after 2.7M steps (paired t-test, $p < 0.05$).

5.4 Dense reward

A desirable property of the proposed method is to avoid hurting performance in tasks where rewards are dense and well-defined. We evaluate this scenario in MultiRoomN10, Door&Key 16×16, and KeyCorridorS6R3. In those tasks, the agent has to collect keys, open doors, and reach a target position. In the sparse setting, the agent is only provided a sparse terminal reward of +1 if it finds the target and 0 otherwise. In the dense setting, the agent is rewarded for collecting keys (+0.3) and opening doors (+0.3), as well as reaching the goal (+1). The results show (Table 5) that our method does not significantly deteriorate performance in dense reward tasks (paired t-test p>0.05), with the exception of FaSo(Cb-AE) on MultiroomN10 (p=0.0034). Even though PPO+FaSo(Cb-AE) and PPO+FaSo(Cb-VAE) perform slightly worse in the dense setting, they still greatly improve performance as compared to plain PPO.

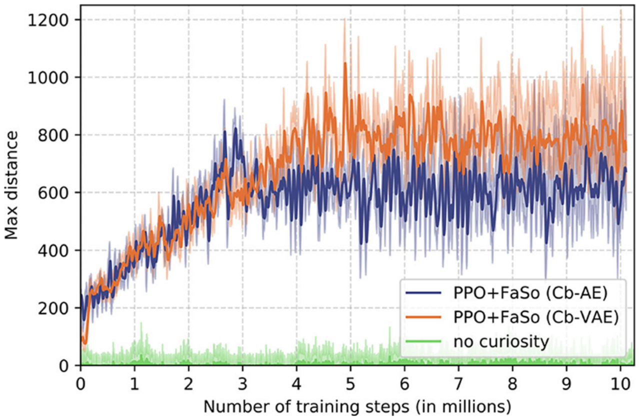


Fig. 7 Maximum distance achieved with no extrinsic reward on Super Mario Bros. We report average distance over 10 seeds. Darker line represents mean and shaded area represents standard error

5.5 Exploration with sparse extrinsic rewards

We now report experimental results in three domains including Minigrid, Super Mario Bros, and Atari games, characterized by sparse rewards. It aims to investigate how useful our proposed method is for hard exploration tasks on which recent advanced exploration methods mainly focused.

5.5.1 Minigrid

We performed a set of two experiments on Door & Key to evaluate the overall performance of our algorithm. First, we verify if our model (PPO+FaSo) achieves better performance than traditional RL methods (DQN [40], PPO [50], A2C [41]). Second, we compare it against state-of-the-art curiosity-based learners for different degree of sparsity (i.e. size of board). We evaluated our model against RND [11], PPO+EC [48], A3C+ICM [44], and, PPO+ICM [44] that were shown to perform well in sparse reward environments.

We present in Fig. 8 the evolution of the extrinsic reward achieved by the agents. The results of each run are averaged to provide a mean curve in each figure, and the standard error is used to make the shaded region surrounding each curve. In such sparse tasks, the learning curve shows that our model always outperforms RL baselines. Moreover, only our method scales with the size of the environment and is significantly faster in term of convergence speed. The performance gap is more pronounced in levels hard to learn (i.e. size > 5) and a significant difference was found

Table 5 Average success rate on tasks from the Minigrid domain with dense and sparse settings

	Sparse			Dense		
Method	MultiRoomN10	Door&Key 16×16	KeyCorridorS6R3	MultiRoomN10	Door&Key 16×16	KeyCorridorS6R3
PPO	0.3±4.3	0.0±2.1	0.0±1.1	22±3.1	63±1.8	16± 1.5
PPO+FaSo(Cb-AE)	87±2.9	**97±3.5**	81±3.5	74±1.1	**93±1.3**	77± 1.9
PPO+FaSo(Cb-VAE)	**88±3.8**	96±2.9	**90±3.6**	**83±1.6**	94±2.9	**86±1.8**

The results are averaged over 100 runs after 40 millions training steps

Bold values indicate the best performing method

between our method and every other baselines (paired t-test at 95% confidence, p<0.001). In Door & Key 5×5, the difference between FaSo and PPO is, however, not statistically significant (t-test p>0.05).

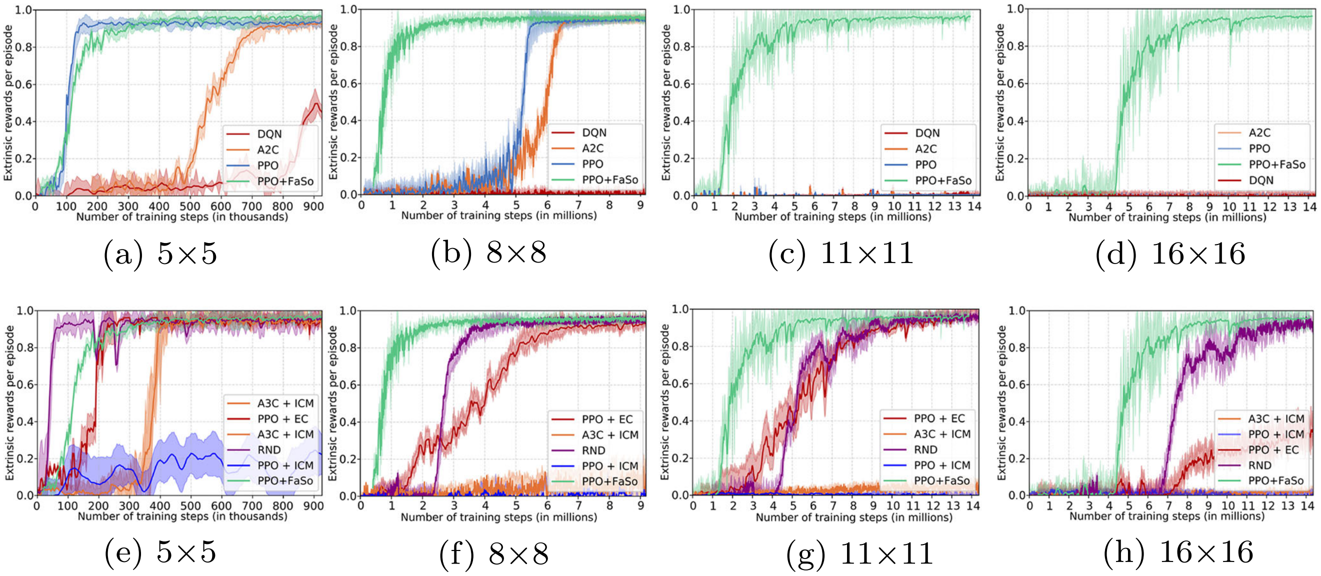
We further observed that baseline methods will exhaust their curiosity quickly after experiencing unexpected events such as after *picking the key*, and therefore struggle to reach the final goal. On the other hand, we found that when using a slow curiosity reward, intrinsic reward remains large enough to encourage long-time horizon exploration strategies such as *opening the door after picking the key*, which significantly improves performance.

5.5.2 Super Mario Bros

Next, we apply our method to the Super Mario Bros environment [29]. As a baseline, we compare our model to the A3C, A2C+ICM and PPO+EC algorithms. For a fair comparison with the state-of-the-art approach [44], we combine FaSo with A2C, from the open-source implementation [16]. We use the same hyperparameters as in the work [44]. Figure 9 shows the normalized average reward (over 10 runs) obtained for each method. The main result is that A2C+FaSo obtains a near perfect score in a smaller number of epochs than any other method. The proposed method can significantly accelerate learning compared to the state-of-the-art ICM algorithm (t-test p=0.023, t=2.26).

5.5.3 Atari games

We also evaluate the proposed curiosity method on five difficult exploration Atari 2600 games from the Arcade Learning Environment (ALE) [8]: Montezuma's Revenge, Private Eye, Gravitar, Pitfall, and Seaquest. In the selected games, training an agent with a poor exploration strategy often results in a suboptimal policy. We compare our method to the performance of A2C and PPO without intrinsic



(a) 5×5 (b) 8×8 (c) 11×11 (d) 16×16

(e) 5×5 (f) 8×8 (g) 11×11 (h) 16×16

Fig. 8 Comparison of PPO+FaSo with baselines with no curiosity (top row) and agents augmented with an exploration bonus (bottom row) in Minigrid Door & Key. The hardness of the exploration task (i.e. sparsity) is gradually increased from left to right. Results are averaged over 10 random seeds. No seed tuning is performed. The shaded area shows the standard errors of 10 runs

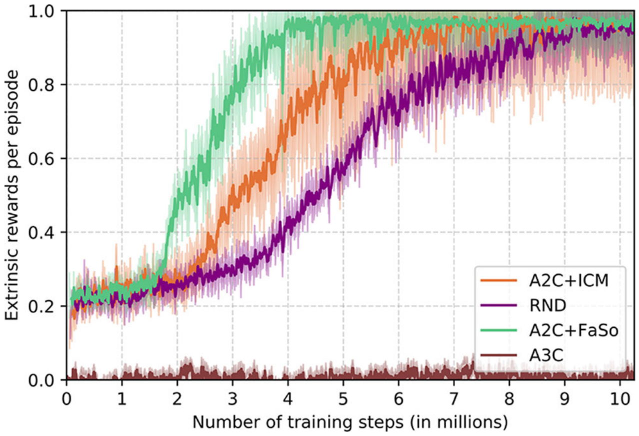


Fig. 9 Average task reward obtained in the Super Mario Bros environment with sparse reward setting. We run every method with a repeat of 10 and show all runs. Mean and standard error of the mean over trials are plotted

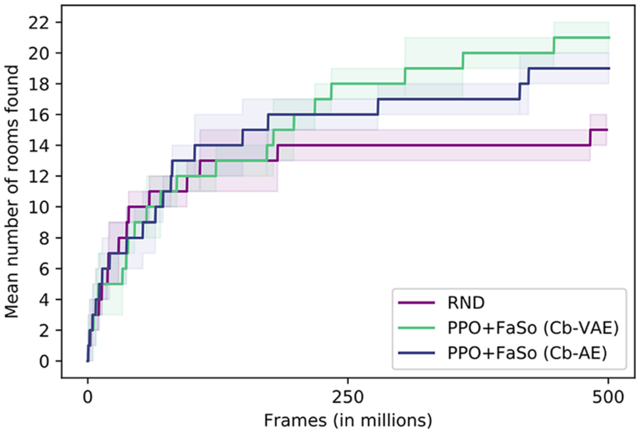


Fig. 10 Average number of rooms (± std-error) found during the training phase on Montezuma's Revenge. We run every algorithm with a repeat of 10

reward. The results are shown in Table 6. We consider the mean final reward of 10 training runs with the same set of hyperparameters. It is observed that both baselines obtained a score close to zero and could not solve most of the tasks.

We further compare PPO+FaSo against various methods using different exploration strategies. To evaluate the significance of the scores, a paired t-test was conducted to compare the received average total reward in the proposed method and the best-performing methods on the five Atari 2600 games. A significant difference was found between our agents and every other methods ($p<0.001$), except on Gravitar where no significant difference was found between between RND and PPO+FaSo(Cb-AE) ($p=0.073$). As presented in Table 6, on Montezuma's Revenge, Seaquest and Private Eye our model outperforms other approaches that mainly deal with local exploration. It suggests that high-level exploration is crucial for exploring in complex environments. For instance, on Montezuma's Revenge, FaSo(Cb-VAE) exceeds state of the art performance. It might be related to the very fact that slow rewards are large enough to motivate the agent to discover and visit new rooms. As a result, our agent explores a larger number of rooms as compared to RND (Fig. 10). In Pitfall, many interactions yield negative rewards that dissuade baselines from exploring efficiently the environment. We found that while the baselines focus on short-term rewards, they tend to converge prematurely to sub-optimal policies. In this task, extremely long-term exploration bonus is required in order for the agent to compensate deceptive extrinsic rewards and discover alternate policies. To the best of our knowledge, this is the first approach without use of expert knowledge that achieves a positive score on Pitfall.

6 Discussion

Our work takes a step toward achieving high-level exploration in DRL. We have constructed a mechanism based

Table 6 Final mean score of our method and baselines on Atari games. We report the results achieved over total 500M timesteps of training, averaged over 10 seeds

	Maximum Mean Score (at convergence)				
Method	Montezuma's Revenge	Private Eye	Gravitar	Pitfall	Seaquest
A2C [41]	15±20	572±136	2,758±185	-17±2	1,613±244
PPO [50]	2,487±942	103±56	3,438±412	-31±5	1,548±341
RND [11]	8,152±653	8,666±1051	**3,906±246**	-3±1	3,179±378
PPO+EC [48]	8,025±770	9,244±634	3,521±246	-12±1	4,650±358
PPO+ICM [44]	329±118	485±71	3,447±242	-15±2	2,165±223
DeepCS [54]	3,500	1,105	881	-186	3,343
Average Human [63]	4,753	69,571	3351	6,464	20,182
PPO+FaSo (Cb-AE)	9,651±442	13,423±775	3,656±280	**247±28**	4,989±311
PPO+FaSo (Cb-VAE)	**11,466±584**	**16,135±688**	3,431±325	189±17	**5,123±251**

on reward decomposition and showed that the method can help exploration in challenging sparse-reward environments. These experiments demonstrate the effectiveness of this approach by achieving significant improvements on notoriously difficult tasks such as Pitfall or Montezuma's Revenge. They suggest that two streams of intrinsic rewards can greatly improve exploration efficiency.

A key element of the current method includes the use of reconstruction-based curiosity, which relies on a reconstructor network. In order to provide an intuition about what this approach does, we study the effectiveness of Cb-VAE to reconstruct observations given noisy inputs. To shed a light on the question above, we visually examine the evolution of errors in reconstruction during exploration. Figure 11 shows examples of reconstructed images with a learned model (i.e. Cb-VAE, *downsample* contexts $K = 2$) of the Super Mario Bros game after 10 and 30 training epochs. We observe that over time the quality of the reconstructed images improves. The general appearance can be quickly captured but small details require a larger number of epochs. In further analysis, we found that the reconstruction errors of a state decrease as more similar states are encountered. Nevertheless, complex or novel states require more visits to be accurately reconstructed, as expected. It suggests that this method can be used to evaluate novelty.

Although a single stream of reconstruction-based rewards is a good source of endogenous motivation, it may suffer from the above-mentioned issues. We found that generating two distinct reward streams can help to overcome these challenges. In order to better understand how the architecture works, we further provide a visual analysis of fast and slow rewards. To do so, we evaluate the general trend of each reward stream (r_t^{fast} and r_t^{slow}) over the first episode on Montezuma's revenge. We also show frames at each reward pike. As can be seen in Fig. 12, both rewards decrease in the number of state visitations, which entails that they can be used to assess novelty. However, a key difference is that fast rewards quickly adapt to measure local state novelty. As a result, the large pikes correspond to important novel events such as collecting an object (5,12), losing a life (8), or overcoming an obstacle (6,9,13). On the other hand, slow rewards pikes mostly correspond to novel events with large state changes, such as discovering a new room (1,4,7,11). It demonstrates that slow rewards are less impacted by minor changes in the environment; and push the agent to seek out novel regions. Please note that small pikes are linked to rare events but already encountered. Therefore, by combining these two rewards, we can expect to discover flexible exploration strategies.

The above experiments and our ablation studies show that the choice of reconstructor architecture (Cb-AE or Cb-VAE) has a limited impact on the performance. However, when facing a novel task, we found Cb-VAE generally better at reconstructing noisy observations. Cb-VAE should be preferred over Cb-AE in environments with complex visual patterns or small details. As a general rule to choose K_{fast} and K_{slow}, *slow contexts* should be more noisy or corrupted than *fast contexts*, to incentive long-term exploration (i.e. $K_{slow} > K_{fast}$). An interesting outcome of the experiments is that increasing K_slow (e.g. $K_{slow} = 5$) helps to overcome very sparse rewards such as in Atari games. We can except $2 \leq K_{fast} \leq 3, 4 \leq K_{slow} \leq 6$ (downsample context) and

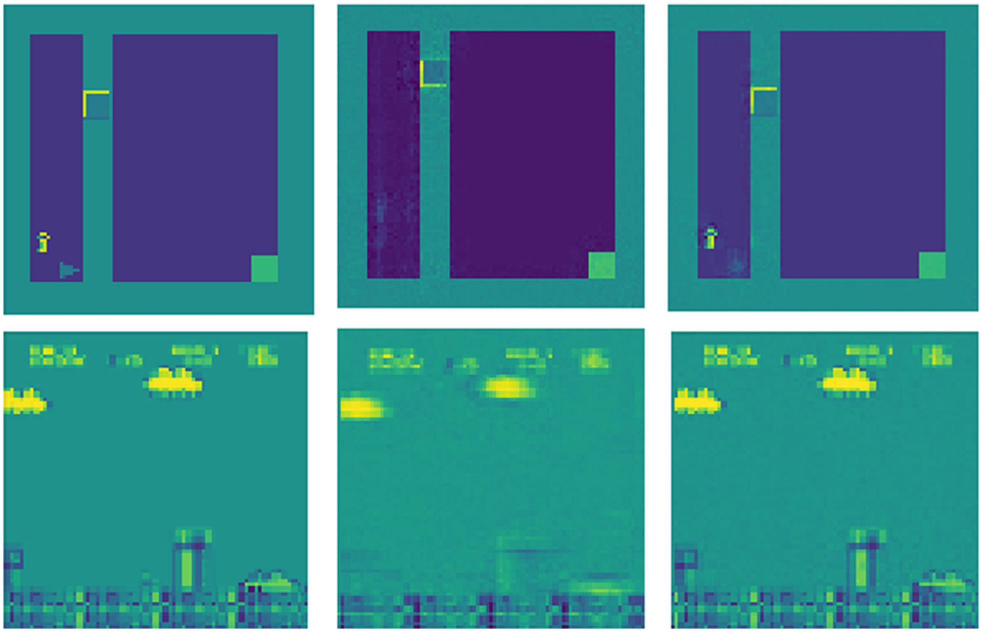


Fig. 11 Examples of reconstruction results for Door & Key (top row) and Super Mario Bros (bottom row). Left column: ground truth. Reconstruction errors are large after 10 epochs (middle column) but only small errors can be noticed after 30 epochs of training (right column)

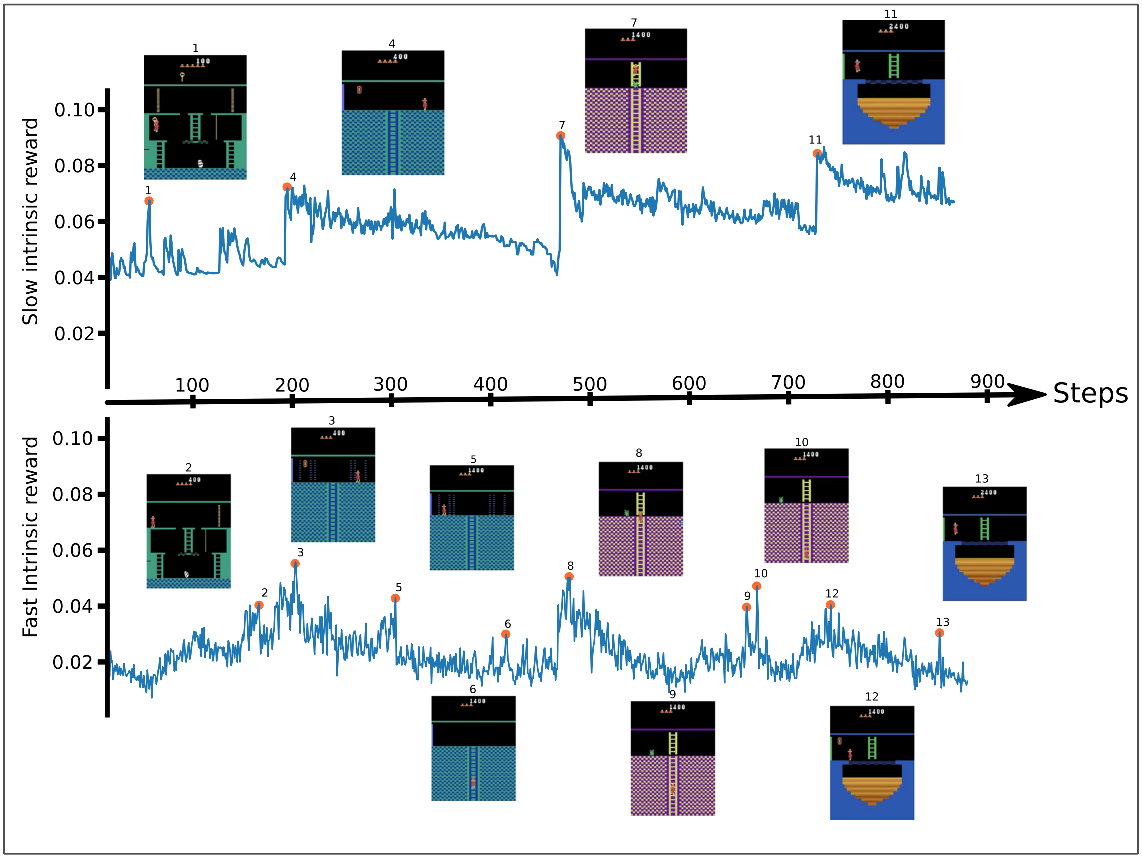


Fig. 12 Fast and slow intrinsic rewards over an episode of Montezuma's Revenge

$16 \leq K_{fast} \leq 32, 32 \leq K_{slow} \leq 64$ (noisy context) to perform well in most environments.

We demonstrate the effectiveness of our approach by achieving significant improvements on notoriously hard exploration games. That being said, we acknowledge that our approach has certain limitations. A key element to our curiosity formulation is to reconstruct images, which entails that images easy to reconstruct (e.g. uniform background) will be less novel than complex images (i.e. high-frequency images). In real-world settings where images tend to contain fine-grained details, the agent may get stuck in such states. Ideally, we would want curiosity to adapt based on the agent's reconstruction progress but we leave it to future work to explore this direction further.

So far, ablation analysis such as in Fig. 12 confirmed the above-mentioned intuitions but lack theoretical understanding. In future work, we aim to improve theoretical understanding of our approach, more specifically, how denoising of images relates to the fast and slow rewards and how fast and slow components contribute to exploration.

Finally, when running experiments, we needed to choose hyperparameters such as context creation K, network architectures, or loss functions. Even though we achieved improvements on most tasks with fixed hyperparameters, the games vary in how long is the delay between action and reward, and how visual complexity affects curiosity. One way to overcome the need of parameter tuning is to incorporate adaptive context creation K based off the magnitude of the reconstruction error. Another solution is to construct relevant high-level features from raw sensory data that lead to a more *unified* input representation for the networks (i.e. state abstraction).

7 Conclusion

We have proposed a novel approach for combining local and global exploration that relies on the concept of curiosity-driven by context reconstruction. Our method can be effectively combined with any on-policy RL algorithm without any prior knowledge of the environment or the tasks being performed. Further benefits stem from efficiently adjusting the scaling of fast curiosity rewards and slow curiosity rewards, based on the idea of maximizing state diversity. This mechanism enables the agent to receive enough intrinsic reward to try global exploration strategies and escape from local optima induced by poorly-defined extrinsic rewards. We demonstrated the effectiveness of our approach and compared it against several baselines on Minigrid, Super Mario Bros, and Atari. A promising research direction is to integrate multiple levels of exploration that could benefit in exploration efficiency. In the future, we also hope to extend our method to deal with other kinds of states such as robot sensors [46, 59]. Another intriguing direction for future work is to take advantage of the trained autoencoder to improve the initialization of the policy network.

References

1. Abel D, Agarwal A, Diaz F, Krishnamurthy A, Schapire RE (2016) Exploratory gradient boosting for reinforcement learning in complex domains. ICML Workshop on Abstraction in Reinforcement Learning
2. Achiam J, Sastry S (2017) Surprise-based intrinsic motivation for deep reinforcement learning. arXiv:170301732
3. Achiam J, Edwards H, Amodei D, Abbeel P (2018) Variational option discovery algorithms. arXiv:180710299
4. Andrychowicz M, Wolski F, Ray A, Schneider J, Fong R, Welinder P, McGrew B, Tobin J, Abbeel OP, Zaremba W (2017) Hindsight experience replay. In: Proceedings of advances in neural information processing systems, pp 5048–5058
5. Baldi P (2012) Autoencoders, unsupervised learning, and deep architectures. In: Proceedings of International conference on machine learning workshop on unsupervised and transfer learning, pp 37–49
6. Baranes A, Oudeyer PY (2013) Active learning of inverse models with intrinsically motivated goal exploration in robots. Robot. Auton. Syst. 61(1):49–73
7. Bellemare M, Srinivasan S, Ostrovski G, Schaul T, Saxton D, Munos R (2016) Unifying count-based exploration and intrinsic motivation. In: Proceedings of advances in neural information processing systems, pp 1471–1479
8. Bellemare MG, Naddaf Y, Veness J, Bowling M (2013) The arcade learning environment: An evaluation platform for general agents. J. Artif. Intell. Res. 47:253–279
9. Botvinick M, Ritter S, Wang JX, Kurth-Nelson Z, Blundell C, Hassabis D (2019) Reinforcement learning, fast and slow. Trends in cognitive sciences
10. Bougie N, Ichise R (2019) Skill-based curiosity for intrinsically motivated reinforcement learning. Mach Learn 109:493–512
11. Burda Y, Edwards H, Storkey A, Klimov O (2018) Exploration by random network distillation. arXiv:181012894
12. Burda Y, Edwards H, Pathak D, Storkey A, Darrell T, Efros AA (2019) Large-scale study of curiosity-driven learning. In: Proceedings of the The International Conference on Learning Representations
13. Burgess CP, Higgins I, Pal A, Matthey L, Watters N, Desjardins G, Lerchner A (2018) Understanding disentangling in *beta*-vae. arXiv:180403599
14. Chevalier-Boisvert M, Willems L, Pal S (2018) Minimalistic gridworld environment for openai gym. https://github.com/maximecb/gym-minigrid
15. Cho K (2013) Simple sparsification improves sparse denoising autoencoders in denoising highly corrupted images. In: International conference on machine learning, pp 432–440
16. Dhariwal P, Hesse C, Klimov O, Nichol A, Plappert M, Radford A, Schulman J, Sidor S, Wu Y, Zhokhov P (2017) Openai baselines. https://github.com/openai/baselines
17. Elad M, Aharon M (2006) Image denoising via sparse and redundant representations over learned dictionaries. IEEE Trans. Image Process. 15(12):3736–3745
18. Florensa C, Held D, Geng X, Abbeel P (2018) Automatic goal generation for reinforcement learning agents. In: Proceedings of the International Conference on Machine Learning
19. Forestier S, Mollard Y, Oudeyer PY (2017) Intrinsically motivated goal exploration processes with automatic curriculum learning. arXiv:170802190
20. Haarnoja T, Zhou A, Abbeel P, Levine S (2018) Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. Machine Learning Research
21. Han D (2013) Comparison of commonly used image interpolation methods. In: Proceedings of the international conference on computer science and electronics engineering
22. Hassabis D, Kumaran D, Summerfield C, Botvinick M (2017) Neuroscience-inspired artificial intelligence. Neuron 95(2):245–258
23. Higgins I, Matthey L, Glorot X, Pal A, Uria B, Blundell C, Mohamed S, Lerchner A (2016) Early visual concept learning with unsupervised deep learning. arXiv:160605579
24. Hong I, Hwang Y, Kim D (2019) Efficient deep learning of image denoising using patch complexity local divide and deep conquer. Pattern Recogn. 96:106945
25. Hong ZW, Shann TY, Su SY, Chang YH, Fu TJ, Lee CY (2018) Diversity-driven exploration strategy for deep reinforcement learning. In: Proceedings of Advances in neural information processing systems
26. Houthooft R, Chen X, Chen X, Duan Y, Schulman J, De Turck F, Abbeel P (2016) Vime: Variational information maximizing exploration. In: Proceedings of advances in neural information processing systems, pp 1109–1117
27. Jinnai Y, Park JW, Abel D, Konidaris G (2019) Discovering options for exploration by minimizing cover time. In: Proceedings of the International Conference on Machine Learning
28. Kaelbling LP (1993) Learning to achieve goals. In: Proceedings of the International Joint Conferences on Artificial Intelligence, pp 1094–1098
29. Kauten C (2018) Super Mario Bros for OpenAI Gym. https://github.com/Kautenja/gym-super-mario-bros
30. Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. arXiv:14126980
31. Kingma DP, Welling M (2014) Auto-encoding variational Bayes. In: Proceedings of the international conference on learning representations
32. Klyubin AS, Polani D, Nehaniv CL (2005) Empowerment: A universal agent-centric measure of control. In: IEEE Congress on Evolutionary Computation, vol 1, pp 128–135
33. Kuderer M, Gulati S, Burgard W (2015) Learning driving styles for autonomous vehicles from demonstration. In: IEEE International Conference on Robotics and Automation, pp 2641–2646

34. Lehman J, Stanley KO (2011) Abandoning objectives: Evolution through the search for novelty alone. Evolutionary computation, 189–223
35. Lillicrap TP, Hunt JJ, Pritzel A, Heess N, Erez T, Tassa Y, Silver D, Wierstra D (2016) Continuous control with deep reinforcement learning. In: Proceedings of international conference on learning representations
36. Machado MC, Bellemare MG, Bowling M (2017) A laplacian framework for option discovery in reinforcement learning. In: Proceedings of the International Conference on Machine Learning, pp 2295–2304
37. Machado MC, Bellemare MG, Bowling M (2018) Count-based exploration with the successor representation. arXiv:180711622
38. Mao XJ, Shen C, Yang YB (2016) Image restoration using convolutional auto-encoders with symmetric skip connections. arXiv:160608921
39. Martin J, Sasikumar SN, Everitt T, Hutter M (2017) Count-based exploration in feature space for reinforcement learning. In: Proceedings of the International Joint Conference on Artificial Intelligence
40. Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G, et al. (2015) Human-level control through deep reinforcement learning. Nature 518(7540):529
41. Mnih V, Badia AP, Mirza M, Graves A, Lillicrap T, Harley T, Silver D, Kavukcuoglu K (2016) Asynchronous methods for deep reinforcement learning. In: Proceedings of international conference on machine learning, pp 1928–1937
42. Nair AV, Pong V, Dalal M, Bahl S, Lin S, Levine S (2018) Visual reinforcement learning with imagined goals. In: Proceedings of advances in neural information processing systems, pp 9191–9200
43. Ostrovski G, Bellemare MG, van den Oord A, Munos R (2017) Count-based exploration with neural density models. In: Proceedings of the international conference on machine learning, pp 2721–2730
44. Pathak D, Agrawal P, Efros AA, Darrell T (2017) Curiosity-driven exploration by self-supervised prediction: In Proceedings of the international conference on international conference on machine learning
45. Pere A, Forestier S, Sigaud O, Oudeyer PY (2018) Unsupervised learning of goal spaces for intrinsically motivated goal exploration. In Proceedings of the international conference on learning representations
46. Plappert M, Andrychowicz M, Ray A, McGrew B, Baker B, Powell G, Schneider J, Tobin J, Chociej M, Welinder P, et al. (2018) Multi-goal reinforcement learning: Challenging robotics environments and request for research. arXiv:180209464
47. Pong VH, Dalal M, Lin S, Nair A, Bahk S, Levine S (2019) Skew-fit: State-covering self-supervised reinforcement learning. arXiv:190303698
48. Savinov N, Raichuk A, Marinier R, Vincent D, Pollefeys M, Lillicrap T, Gelly S (2019) Episodic curiosity through reachability. In: Proceedings of the international conference on learning representations
49. Schaul T, Horgan D, Gregor K, Silver D (2015) Universal value function approximators. In: Proceedings of the International conference on machine learning
50. Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O (2017) Proximal policy optimization algorithms. arXiv:170706347
51. Silver D, Huang A, Maddison CJ, Guez A, Sifre L, Van Den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, et al. (2016) Mastering the game of go with deep neural networks and tree search. Nature 529(7587):484
52. Snell J, Ridgeway K, Liao R, Roads BD, Mozer MC, Zemel RS (2017) Learning to generate images with perceptual similarity metrics. In: IEEE International Conference on Image Processing (ICIP), vol 2017. IEEE, pp 4277–4281
53. Stadie BC, Levine S, Abbeel P (2015) Incentivizing exploration in reinforcement learning with deep predictive models
54. Stanton C, Clune J (2019) Deep curiosity search: Intra-life exploration improves performance on challenging deep reinforcement learning problems. In: Proceedings of the international conference on international conference on machine learning
55. Strehl AL, Littman ML (2008) An analysis of model-based interval estimation for Markov decision processes. J. Comput. Syst. Sci. 74(8):1309–1331
56. Sutton RS (1988) Learning to predict by the methods of temporal differences. Machine Learning 3(1):9–44
57. Sutton RS, Barto AG (1998) Reinforcement learning: an introduction. MIT Press, Cambridge
58. Tang H, Houthooft R, Foote D, Stooke A, Chen OX, Duan Y, Schulman J, DeTurck F, Abbeel P (2017) # exploration: A study of count-based exploration for deep reinforcement learning. In: Proceedings of Advances in neural information processing systems, pp 2753–2762
59. Todorov E, Erez T, Tassa Y (2012) Mujoco: A physics engine for model-based control. In: IEEE/RSJ international conference on intelligent robots and systems, pp 5026–5033
60. Vezhnevets AS, Osindero S, Schaul T, Heess N, Jaderberg M, Silver D, Kavukcuoglu K (2017) Feudal networks for hierarchical reinforcement learning. In: Proceedings of the international conference on machine learning, pp 3540–3549
61. Wang Z, Simoncelli EP, Bovik AC (2003) Multiscale structural similarity for image quality assessment. In: Proceedings of the Conference on Signals, Systems & Computers, vol 2. IEEE, pp 1398–1402
62. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP, et al. (2004) Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing 13(4):600–612
63. Wang Z, Schaul T, Hessel M, Van Hasselt H, Lanctot M, De Freitas N (2016) Dueling network architectures for deep reinforcement learning
64. Yang HK, Chiang PH, Hong MF, Lee CY (2019) Exploration via flow-based intrinsic rewards. arXiv:190510071

Nicolas Bougie graduated from the University of Paris Sud, France in 2017. He studied machine learning and artificial intelligence. He is currently a PhD student at the National Institute of Informatics in Japan and Sokendai University. His research area covers reinforcement learning, deep learning and advanced decision-making in cooperation with humans.

Ryutaro Ichise received his Ph.D. degree in computer science from Tokyo Institute of Technology, Tokyo, Japan, in 2000. From 2001 to 2002, he was a visiting scholar at Stanford University. He is currently an associate professor in Principles of Informatics Research Division at the National Institute of Informatics in Japan. His research interests include machine learning, semantic web, and data mining.