Beyond Surprise: Improving Exploration Through Surprise Novelty

Hung Le
Applied AI Institute, Deakin University
Geelong, Australia
thai.le@deakin.edu.au

Kien Do
Applied AI Institute, Deakin University
Geelong, Australia
k.do@deakin.edu.au

Dung Nguyen
Applied AI Institute, Deakin University
Geelong, Australia
dung.nguyen@deakin.edu.au

Svetha Venkatesh
Applied AI Institute, Deakin University
Geelong, Australia
svetha.venkatesh@deakin.edu.au

ABSTRACT

We present a new computing model for intrinsic rewards in reinforcement learning that addresses the limitations of existing surprise-driven explorations. The reward is the *novelty of the surprise* rather than the surprise norm. We estimate the surprise novelty as retrieval errors of a memory network wherein the memory stores and reconstructs surprises. Our surprise memory (SM) augments the capability of surprise-based intrinsic motivators, maintaining the agent's interest in exciting exploration while reducing unwanted attraction to unpredictable or noisy observations. Our experiments demonstrate that the SM combined with various surprise predictors exhibits efficient exploring behaviors and significantly boosts the final performance in sparse reward environments, including Noisy-TV, navigation and challenging Atari games.

KEYWORDS

Reinforcement Learning; Exploration; Intrinsic Motivation; Memory

ACM Reference Format:
Hung Le, Kien Do, Dung Nguyen, and Svetha Venkatesh. 2024. Beyond Surprise: Improving Exploration Through Surprise Novelty. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 17 pages.

1 INTRODUCTION

What motivates agents to explore? Successfully answering this question would enable agents to learn efficiently in formidable tasks. Random explorations such as ϵ-greedy are inefficient in high dimensional cases, failing to learn despite training for hundreds of million steps in sparse reward games [5]. Alternative approaches propose to use intrinsic motivation to aid exploration by adding bonuses to the environment's rewards [5, 33]. The intrinsic reward is often proportional to the novelty of the visiting state: it is high if the state is novel (e.g. different from the past ones [2, 3]) or less frequently visited [5, 35].

Another view of intrinsic motivation is from surprise, which refers to the result of the experience being unexpected, and is determined by the discrepancy between the expectation (from the agent's prediction) and observed reality [4, 31]. Technically, surprise is the difference between prediction and observation representation vectors. The norm of the residual (i.e. prediction error) is used as the intrinsic reward. Here, we will use the terms "surprise" and "surprise norm" to refer to the residual vector and its norm, respectively. Recent works have estimated surprise with various predictive models such as dynamics [33], episodic reachability [29] and inverse dynamics [25]; and achieved significant improvements with surprise norm [7]. However, surprise-based agents tend to be overly curious about noisy or unpredictable observations [16, 30]. For example, consider an agent watching a television screen showing white noise (noisy-TV problem). The TV is boring, yet the agent cannot predict the screen's content and will be attracted to the TV due to its high surprise norm. This distraction or "fake surprise" is common in partially observable Markov Decision Process (POMDP), including navigation tasks and Atari games [8]. Many works have addressed this issue by relying on the learning progress [1, 30] or random network distillation (RND) [8]. However, the former is computationally expensive, and the latter requires many samples to perform well.

This paper overcomes the "fake surprise" issue by using *surprise novelty* - a new concept that measures the uniqueness of surprise. To identify surprise novelty, the agent needs to compare the current surprise with surprises in past encounters. One way to do this is to equip the agent with some kind of associative memory, which we implement as an autoencoder whose task is to reconstruct a query surprise. The lower the reconstruction error, the lower the surprise novelty. A further mechanism is needed to deal with the rapid changes in surprise structure within an episode. As an example, if the agent meets the same surprise at two time steps, its surprise novelty should decline, and with a simple autoencoder this will not happen. To remedy this, we add an episodic memory, which stores intra-episode surprises. Given the current surprise, this memory can retrieve similar "surprises" presented earlier in the episode through an attention mechanism. These surprises act as a context added to the query to help the autoencoder better recognize whether the query surprise has been encountered in the episode or not. The

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

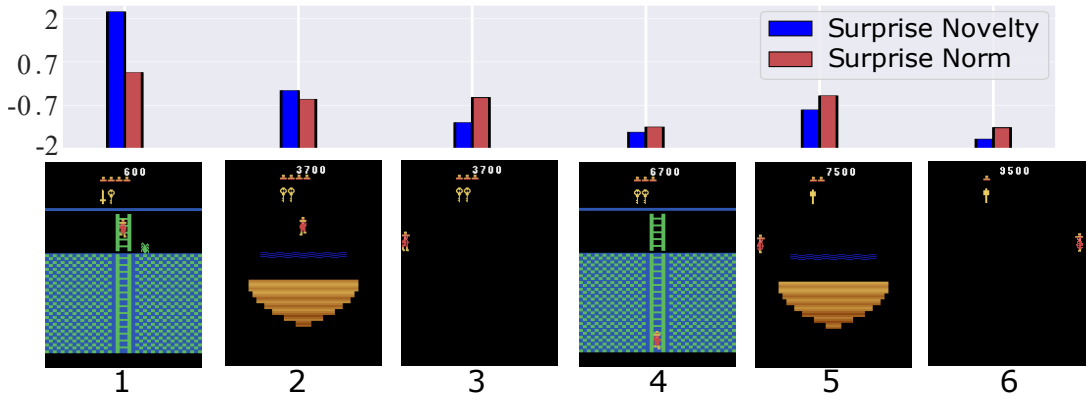

Figure 1: Montezuma Revenge: *surprise novelty* better reflects the originality of the environment than *surprise norm*. While surprise norm can be significant even for dull events such as those in the dark room due to unpredictability, surprise novelty tends to be less (3^{rd} and 6^{th} image). On the other hand, surprise novelty can be higher in truly vivid states on the first visit to the ladder and island rooms (1^{st} and 2^{nd} image) and reduced on the second visit (4^{th} and 5^{th} image). Here, surprise novelty and surprise norm are quantified and averaged over steps in each room.

error between the query and the autoencoder's output is defined as *surprise novelty*, to which the intrinsic reward is set proportionally.

We argue that using surprise novelty as an intrinsic reward is better than surprise norm. As in POMDPs, surprise norms can be very large since the agent cannot predict its environment perfectly, yet there may exist patterns of prediction failure. If the agent can remember these patterns, it will not feel surprised when similar prediction errors appear regardless of the surprise norms. An important emergent property of this architecture is that when random observations are presented (e.g., white noise in the noisy-TV problem), the autoencoder can act as an identity transformation operator, thus effectively "passing the noise through" to reconstruct it with low error. We conjecture that the autoencoder is able to do this with the surprise rather than the observation as the surprise space has lower variance, and we show this in our paper. Since our memory system works on the surprise level, we need to adopt current intrinsic motivation methods to generate surprises. The surprise generator (SG) can be of any kind based on predictive models mentioned earlier and is jointly trained with the memory to optimize its own loss function. To train the surprise memory (SM), we optimize the memory's parameters to minimize the reconstruction error.

Our contribution is two-fold:

- We propose a new concept of surprise novelty for intrinsic motivation. We argue that it reflects better the environment originality than surprise norm (see motivating graphics Fig. 1).
- We design a novel memory system, named Surprise Memory (SM) that consists of an autoencoder associative memory and an attention-based episodic memory. Our two-memory system estimates surprise novelty within and across episodes.

In our experiments, the SM helps RND [8] perform well in our challenging noisy-TV problem while RND alone performs poorly. Not only with RND, we consistently demonstrate significant performance gain when coupling three different SGs with our SM in sparse-reward tasks. Finally, in hard exploration Atari games, we boost the scores of 2 strong SGs, resulting in better performance under the low-sample regime.

2 METHODS

2.1 Surprise Novelty

Surprise is the difference between expectation and observation [11]. If a surprise repeats, it is no longer a surprise. Based on this intuition, we hypothesize that surprises can be characterized by their novelties, and an agent's curiosity is driven by the surprise novelty rather than the surprising magnitude. Moreover, surprise novelty should be robust against noises: it is small even for random observations. For example, watching a random-channel TV can always be full of surprises as we cannot expect which channel will appear next. However, the agent should soon find it boring since the surprise of random noises reoccurs repeatedly, and the channels are entirely unpredictable.

We propose using a memory-augmented neural network (MANN) to measure surprise novelty. The memory remembers past surprise patterns, and if a surprise can be retrieved from the memory, it is not novel, and the intrinsic motivation should be small. The memory can also be viewed as a reconstruction network. The network can pass its inputs through for random, pattern-free surprises, making them retrievable. Surprise novelty has an interesting property: if some event is unsurprising (the expectation-reality residual is $\overrightarrow{0}$), its surprise ($\overrightarrow{0}$ with norm 0) is always perfectly retrievable (surprise novelty is 0). In other words, low surprise norm means low surprise novelty. On the contrary, high surprise norm can have little surprise novelty as long as the surprise can be retrieved from the memory either through associative recall or pass-through mechanism. Another property is that the variance of surprise is generally lower than that of observation (state), potentially making the learning on surprise space easier. This property is formally stated as follows.

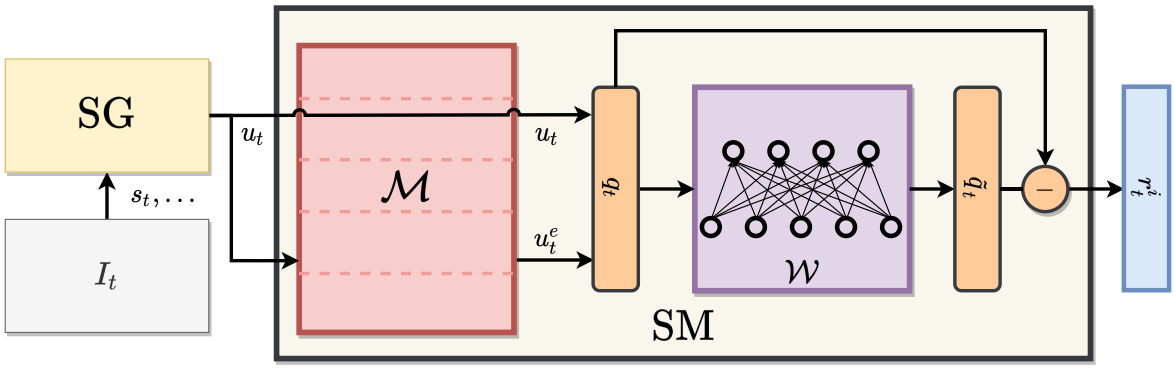

Figure 2: Surprise Generator+Surprise Memory (SG+SM). The SG takes input I_t from the environment to estimate the surprise u_t at state s_t. The SM consists of two modules: an episodic memory ($\mathcal{M}$) and an autoencoder network ($\mathcal{W}$). $\mathcal{M}$ is slot-based, storing past surprises within the episode. At timestep t, given surprise u_t, $\mathcal{M}$ retrieves read-out u_t^e to form a query surprise $q_t = [u_t^e, u_t]$ to $\mathcal{W}$. $\mathcal{W}$ reconstructs the query and takes the reconstruction error (surprise novelty) as the intrinsic reward r_t^i.

Proposition 1. *Let X and U be random variables representing the observation and surprise at the same timestep, respectively. Under an imperfect SG, the following inequality holds:*

$$\forall i: \ \left(\sigma_i^X\right)^2 \geq \left(\sigma_i^U\right)^2$$

where $\left(\sigma_i^X\right)^2$ and $\left(\sigma_i^U\right)^2$ denote the i-th diagonal elements of $\mathrm{var}(X)$ and $\mathrm{var}(U)$, respectively.

Proof. See Appendix E. □

2.2 Surprise Generator

Since our MANN requires surprises for its operation, it is built upon a prediction model, which will be referred to as Surprise Generators (SG). In this paper, we adopt many well-known SGs (e.g. RND [8] and ICM [25]) to predict the observation, compute the surprise u_t for every step in the environment. The surprise norm is the Euclidean distance between the expectation and the reality:

$$\|u_t\| = \|SG(I_t) - O_t\| \tag{1}$$

where $u_t \in \mathbb{R}^n$ is the surprise vector of size n, I_t the input of the SG at step t of the episode, $SG(I_t)$ and O_t the SG's prediction and the observation target, respectively. The input I_t is specific to the SG architecture choice, which can be the current (s_t) or previous state, action (s_{t-1}, a_t). The observation target O_t is usually a transformation (can be identical or random) of the current state s_t, which serves as the target for the SG's prediction. The SG is usually trained to minimize:

$$\mathcal{L}_{SG} = \mathbb{E}_t\left[\|u_t\|\right] \tag{2}$$

Here, predictable observations have minor prediction errors or little surprise. One issue is that a great surprise norm can be simply due to noisy or distractive observations. Next, we propose a remedy for this problem.

2.3 Surprise Memory

The surprise generated by the SG is stored and processed by a memory network dubbed Surprise Memory (SM). It consists of an episodic memory $\mathcal{M}$ and an autoencoder network $\mathcal{W}$, jointly optimized to reconstruct any surprise. At each timestep, the SM receives a surprise u_t from the SG module and reads content u_t^e from the memory $\mathcal{M}$. $\{u_t^e, u_t\}$ forms a surprise query q_t to $\mathcal{W}$ to retrieve the reconstructed $\tilde{q}_t$. This reconstruction will be used to estimate the novelty of surprises forming intrinsic rewards r_t^i. Fig. 2 summarizes the operations of the components of our proposed method. Our 2 memory design effectively recovers surprise novelty by handling intra and inter-episode surprise patterns thanks to $\mathcal{M}$ and $\mathcal{W}$, respectively. $\mathcal{M}$ can quickly adapt and recall surprises that occur within an episode. $\mathcal{W}$ is slower and focuses more on consistent surprise patterns across episodes during training.

Here the query q_t can be directly set to the surprise u_t. However, this ignores the rapid change in surprise within an episode. Without $\mathcal{M}$, when the SG and $\mathcal{W}$ are fixed (during interaction with environments), their outputs u_t and $\tilde{q}_t$ stay the same for the same input I_t. Hence, the intrinsic reward r_t^i also stays the same. It is undesirable since when the agent observes the same input at different timesteps (e.g., $I_1 = I_2$), we expect its curiosity should decrease in the second visit ($r_1^i < r_2^i$). Therefore, we design SM with $\mathcal{M}$ to fix this issue.

The episodic memory $\mathcal{M}$ stores representations of surprises that the agent encounters during an episode. For simplicity, $\mathcal{M}$ is implemented as a first-in-first-out queue whose size is fixed as N. Notably, the content of $\mathcal{M}$ is wiped out at the end of each episode. Its information is limited to a single episode. $\mathcal{M}$ can be viewed as a matrix: $\mathcal{M} \in \mathbb{R}^{N \times d}$, where d is the size of the memory slot. We denote $\mathcal{M}(j)$ as the j-th row in the memory, corresponding to the surprise u_{t-j}. To retrieve from $\mathcal{M}$ a read-out u_t^e that is close to u_t, we perform content-based attention [14] to compute the attention weight as $w_t(j) = \frac{(u_t Q)\mathcal{M}(j)^\top}{\|(u_t Q)\|\|\mathcal{M}(j)\|}$. The read-out from $\mathcal{M}$ is then $u_t^e = w_t \mathcal{M} V \in \mathbb{R}^n$. Here, $Q \in \mathbb{R}^{n \times d}$ and $V \in \mathbb{R}^{d \times n}$ are learnable weights mapping between the surprise and the memory space. To force the read-out close to u_t, we minimize:

$$\mathcal{L}_{\mathcal{M}} = \mathbb{E}_t\left[\left\|u_t^e - u_t\right\|\right] \tag{3}$$

The read-out and the SG's surprise form the query surprise to $\mathcal{W}$: $q_t = \left[u_t^e, u_t\right] \in \mathbb{R}^{2n}$. $\mathcal{M}$ stores intra-episode surprises to assist the autoencoder in preventing the agent from exploring "fake surprise" within the episode. Our episodic memory formulation is unlike prior KNN-based episodic memory [3] because our system learns to enforce the memory retrieval error to be small. This is critical when the representations stored in $\mathcal{M}$ can not be easily discriminated if we only rely on unsupervised distance-based metrics. More importantly, under our formulation, the retrieval error can still be small even when the stored items in the memory differ from the query. This helps detect the "fake surprise" when the query surprise seems unlike those in the memory if considered individually, yet can be approximated as weighted sum of the store surprises.

In general, since we optimize the parameters to reconstruct u_t using past surprises in the episode, if the agent visits a state whose surprise is predictable from those in $\mathcal{M}$, $\left\|u_t^e - u_t\right\|$ should be small. Hence, the read-out context u_t^e contains no extra information than u_t and reconstructing q_t from $\mathcal{W}$ becomes easier as it is equivalent to reconstructing u_t. In contrast, visiting diverse states leads to a more novel read-out u_t^e and makes it more challenging to reconstruct q_t, generally leading to higher intrinsic reward.

The autoencoder network $\mathcal{W}$ can be viewed as an associative memory of surprises that persist across episodes. At timestep t in any episode during training, $\mathcal{W}$ is queried with q_t to produce a reconstructed memory $\tilde{q}_t$. The surprise novelty is determined as:

$$r_t^i = \|\tilde{q}_t - q_t\| \tag{4}$$

which is the norm of the surprise residual $\tilde{q}_t - q_t$. It will be normalized and added to the external reward as an intrinsic reward bonus. The details of computing and using normalized intrinsic rewards can be found in Appendix C.

We implement $\mathcal{W}$ as a feed-forward neural network that learns to reconstruct its own inputs. The query surprise is encoded to the weights of the network via backpropagation as we minimize the reconstruction loss below:

$$\mathcal{L}_{\mathcal{W}} = \mathbb{E}_t\left[r_t^i\right] = \mathbb{E}_t\left[\|\mathcal{W}(q_t) - q_t\|\right] \tag{5}$$

Here, $\tilde{q}_t = \mathcal{W}(q_t)$. Intuitively, it is easier to retrieve non-novel surprises experienced many times in past episodes. Thus, the intrinsic reward is lower for states that leads to these familiar surprises. On the contrary, rare surprises are harder to retrieve, which results in high reconstruction errors and intrinsic rewards. We note that autoencoder (AE) has been shown to be equivalent to an associative memory that supports memory encoding and retrieval through attractor dynamics [27]. Unlike slot-based memories, AE has a fixed memory capacity, compresses information and learns data representations. We could store the surprise in a slot-based memory across episodes, but the size of this memory would be autonomous, and the data would be stored redundantly. Hence, the quality of the stored surprise will reduce as more and more observations come in. On the other hand, AE can efficiently compress surprises to latent representations and hold them to its neural weights, and the surprise retrieval is optimized. Besides, AE can learn to switch between 2 mechanisms: pass-through and pattern retrieval, to optimally achieve its objective. We cannot do that with slot-based memory. Readers can refer to Appendix A to see the architecture details and how $\mathcal{W}$ can be interpreted as implementing associative memory.

Algorithm 1 Intrinsic rewards computing via SG+SM framework.

Require: u_t, and our surprise memory SM consisting of a slot-based memory $\mathcal{M}$, parameters Q, V, and a neural network $\mathcal{W}$

Compute $\mathcal{L}_{SG} = \|u_t\|$
Query $\mathcal{M}$ with u_t, retrieve $u_t^e = w_t \mathcal{M} V$ where w_t is the attention weight
Compute $\mathcal{L}_{\mathcal{M}} = \|u_t^e - u_t.detach()\|$
Query $\mathcal{W}$ with $q_t = [u_t^e, u_t]$, retrieve $\tilde{q}_t = \mathcal{W}(q_t)$
Compute intrinsic reward $r_t^i = L_{\mathcal{W}} = \|\tilde{q}_t - q_t.detach()\|$
return $\mathcal{L}_{SG}, \mathcal{L}_{\mathcal{M}}, L_{\mathcal{W}}$

Algorithm 2 Jointly training SG+SM and the policy.

Require: buffer, policy π_θ, surprise-based predictor SG, and our surprise memory SM consisting of a slot-based memory $\mathcal{M}$, parameters Q, V, and a neural network $\mathcal{W}$

Initialize π_θ, SG, Q, $\mathcal{W}$
for $iteration = 1, 2, \ldots$ **do**
  for $t = 1, 2, \ldots T$ **do**
    Execute policy π_θ to collect s_t, a_t, r_t, forming input $I_t = s_t, \ldots$ and target O_t
    Compute surprise $u_t = SG(I_t) - O_t.detach()$ (Eq. 1)
    Compute intrinsic reward r_t^i using Algo. 1
    Compute final reward $r_t \leftarrow r_t + \beta r_t^i / r_t^{std}$
    Add $(I_t, O_t, s_{t-1}, s_t, a_t, r_t)$ to buffer
    Add $u_t Q$ to $\mathcal{M}$
    if done episode **then** clear $\mathcal{M}$
  end for
  for $k = 1, 2, .., K$ **do**
    Sample I_t, O_t from buffer
    Compute surprise $u_t = SG(I_t) - O_t.detach()$ (Eq. 1)
    Compute $\mathcal{L}_{SG}$, $\mathcal{L}_{\mathcal{M}}$, $L_{\mathcal{W}}$ using Algo. 1
    Update SG, Q and $\mathcal{W}$ by minimizing the loss $\mathcal{L} = \mathcal{L}_{SG} + \mathcal{L}_{\mathcal{M}} + \mathcal{L}_{\mathcal{W}}$
    Update π_θ with sample (s_{t-1}, s_t, a_t, r_t) from buffer using backbone algorithms
  end for
end for

The whole system SG+SM is trained end-to-end by minimizing the following loss: $\mathcal{L} = \mathcal{L}_{SG} + \mathcal{L}_{\mathcal{M}} + \mathcal{L}_{\mathcal{W}}$. Here, we block the gradients from $\mathcal{L}_{\mathcal{W}}$ backpropagated to the parameters of SG to avoid trivial reconstructions of q_t. The pseudocode of our algorithm is presented in Algo. 1 and 2. We note that vector notations in the algorithm are row vectors. For simplicity, the algorithm assumes 1 actor. In practice, our algorithm works with multiple actors and mini-batch training. See Appendix C for explaination of β and r^{std}.

3 EXPERIMENTAL RESULTS

3.1 Noisy-TV: Robustness against Noisy Observations

We use Noisy-TV, an environment designed to fool exploration methods [8, 29], to confirm that our method can generate intrinsic

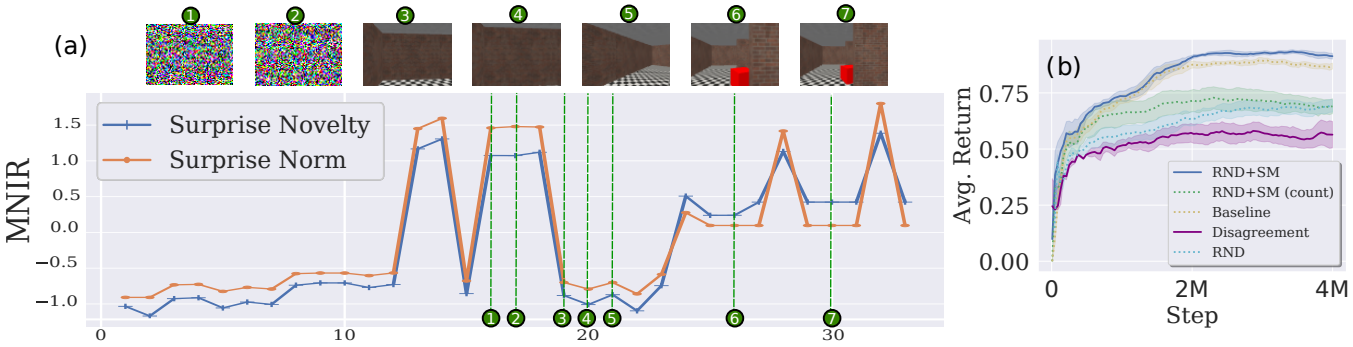

Figure 3: Noisy-TV: (a) mean-normalized intrinsic reward (MNIR) produced by RND and RND+SM at 7 selected steps in an episode. (b) Average task return (mean±std. over 5 runs) over 4 million training steps.

rewards that (1) are more robust to noises and (2) can discriminate rare and common observations through surprise novelty. We simulate this problem by employing a 3D maze environment with a random map structure. The TV is not fixed in specific locations in the maze to make it more challenging. Instead, the agent "brings" the TV with it and can choose to watch TV anytime. Hence, there are three basic actions (turn left, right, and move forward) plus an action: watch TV. When taking this action, the agent will see a white noise image sampled from standard normal distribution and thus, the number of TV channels can be considered infinity. The agent's state is an image of its viewport, and its goal is to search for a red box randomly placed in the maze (+1 reward if the agent reaches the goal).

The main baseline is RND [8], a simple yet strong SG that is claimed to obviate the stochastic problems of Noisy-TV. Our SG+SM model uses RND as the SG, so we name it RND+SM. The source code can be accessed on our GitHub repository at https://github.com/thaihungle/SM. Since our model and the baseline share the same RND architecture, the difference in performance must be attributed to our SM. We also include a disagreement-based method (Disagreement) [26], which leverages the variance of multiple dynamic predictors as the intrinsic reward. Finally, as a reference, we use PPO without intrinsic reward as the vanilla Baseline, which is not affected by the noisy TV traps.

The result demonstrates that the proposed SM outperforms other intrinsic motivators by a significant margin, as shown in Fig. 3 (b). Notably, SM improves RND performance by around 25 %, showcasing the impact of surprise novelty. The result also shows that RND+SM outperforms the vanilla Baseline. Although the improvement is moderate (0.9 vs 0.85), the result is remarkable since the Noisy-TV is designed to fool intrinsic motivation methods and among all, only RND+SM can outperform the vanilla Baseline.

Fig. 3 (a) illustrates the mean-normalized intrinsic rewards (MNIR)[1] measured at different states in our Noisy-TV environment. The first two states are noises, the following three states are common walls, and the last two are ones where the agent sees the box. The MNIR bars show that both models are attracted mainly by the noisy TV, resulting in the highest MNIRs. However, our model with SM suffers less from noisy TV distractions since its MNIR is lower than RND's. We speculate that SM is able to partially reconstruct the white-noise surprise via the pass-through mechanism, making the normalized surprise novelty generally smaller than the normalized surprise norm in this case. That mechanism is enhanced in SM with surprise reconstruction (see Appendix D.1 for explanation).

On the other hand, when observing red box, RND+SM shows higher MNIR than RND. The difference between MNIR for common and rare states is also more prominent in RND+SM than in RND because RND prediction is not perfect even for common observations, creating relatively significant surprise norms for seeing walls. The SM fixes that issue by remembering surprise patterns and successfully retrieving them, producing much smaller surprise novelty compared to those of rare events like seeing red box. Consequently, the agent with SM outperforms the other by a massive margin in task rewards (Fig. 3 (b)).

As we visualize the number of watching TV actions and the value of the intrinsic reward by RND+SM and RND over training time, we realize that RND+SM helps the agent take fewer watching actions and thus, collect smaller amounts of intrinsic rewards compared to RND. See more in Appendix D.1.

3.2 MiniGrid: Compatibility with Different Surprise Generators

We show the versatility of our framework SG+SM by applying SM to 4 SG backbones: RND [8], ICM [25], NGU [3] and autoencoder-AE (see Appendix D.2 for implementation details). We test the models on three tasks from MiniGrid environments: Key-Door (KD), Dynamic-Obstacles (DO) and Lava-Crossing (LC) [10]. If the agent reaches the goal in the tasks, it receives a +1 reward. Otherwise, it can be punished with negative rewards if it collides with obstacles or takes too much time to finish the task. These environments are not as stochastic as the Noisy-TV but they still contain other types of distraction. For example, in KD, the agent can be attracted to irrelevant actions such as going around to drop and pick the key. In DO, instead of going to the destination, the agent may chase obstacle balls flying around the map. In LC the agent can commit unsafe actions like going near lava areas, which are different from typical paths. In any case, due to reward sparsity, intrinsic motivation is beneficial. However, surprise alone may not be enough to guide an

[1]See Appendix C for more information on this metric.

Task	w/o intrinsic reward	RND		ICM		NGU		AE	
		w/o SM	w/ SM	w/o SM	w/ SM	w/o SM	w/ SM	w/o SM	w/ SM
KD	0.0±0.0	48.3±26	*79.3±4*	5.9±5	4.7±3	64.4±3	*83.4±4*	1.4±1	***91.2±6***
DO	-27.0±0.7	-13.6±8	***70.8±11***	-27.7±2	*43.6±16*	-23.9±3	*48.6±28*	-5.1±2	*67.5±13*
LC	78.0±1.7	25.0±35	*71.1±5*	56.2±40	***84.6±1***	42.2±40	*69.5±5*	29.0±6/	*70.9±2*

Table 1: MiniGrid: test performance after 10 million training steps. The numbers are average task return×100 over 128 episodes (mean±std. over 5 runs). Bold denotes the best results on each task. Italic denotes that SG+SM is better than SG regarding Cohen effect size less than 0.5.

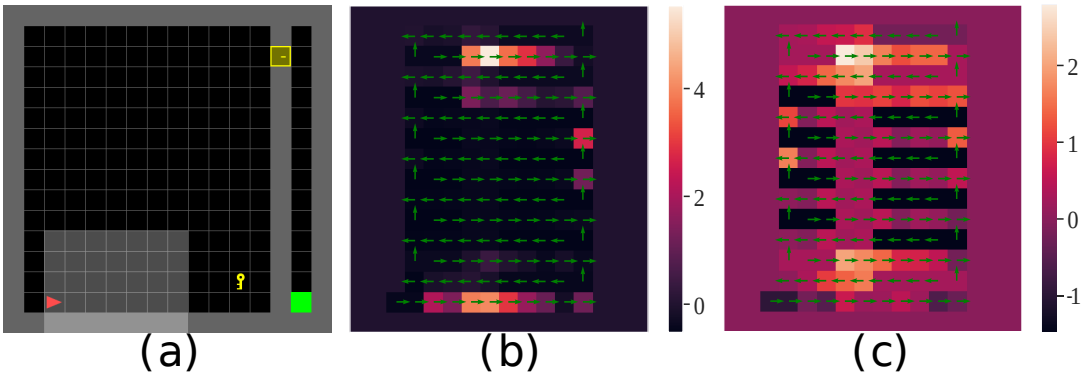

Figure 4: Key-Door: (a) Example map in Key-Door where the light window is the agent's view window (state). MNIR produced for each cell in a manually created trajectory for RND+SM (b) and RND (c). The green arrows denote the agent's direction at each location. The brighter the cell, the higher MNIR assigned to the corresponding state.

efficient exploration since the observation can be too complicated for SG to minimize its prediction error. Thus, the agent quickly feels surprised, even in unimportant states.

Table 1 shows the average returns of the models for three tasks. The Baseline is the PPO backbone trained without intrinsic reward. RND, ICM, NGU and AE are SGs providing the PPO with surprise-norm rewards while our method SG+SM uses surprise-novelty rewards. The results demonstrate that models with SM often outperform SG significantly and always contain the best performers. Notably, in the LC task, SGs hinder the performance of the Baseline because the agents are attracted to dangerous vivid states, which are hard to predict but cause the agent's death. The SM models avoid this issue and outperform the Baseline for the case of ICM+SM. Compared to AE, which computes intrinsic reward based on the novelty of the state, AE+SM shows a much higher average score in all tasks. That manifests the importance of modeling the novelty of surprise instead of states.

Regarding the compatibility of our SM with different SGs, we realize that if the SG is strong, SG+SM tends to have better performance. For example, in LC task, ICM is the best SG, resulting ICM+SM being the best performer. The exception is the AE SG in KD task. We speculate that AE generates intrinsic rewards as the reconstruction error, which can be more sensitive to the state representation of specific tasks. In KD, the viewport of the agent is almost empty. The state representations look similar most of the time, leading to similar AE's reconstruction errors (surprise norm), even for special events (pick the key). Therefore, AE performance is just slightly above the no-IR baseline. The good thing is that other noisy states like throwing the key also receive low IR. When equipped with SM, AE+SM differentiates the reconstruction residual vectors, which can vary even when they have similar norms (Appendix Fig. 8). Therefore, AE+SM can assign high IR to special events while still providing low IR to noisy states, which is optimal in this case.

To analyze the difference between the SG+SM and SG's MNIR structure, we visualize the MNIR for each cell in the map of Key-Door in Figs. 4 (b) and (c). We create a synthetic trajectory that scans through all the cells in the big room on the left and, at each cell, uses RND+SM and RND models to compute the corresponding surprise-norm and surprise-novelty MNIRs, respectively. As shown in Fig. 4 (b), RND+SM selectively identifies truly surprising events, where only a few cells have high surprise-novelty MNIR. Here, we can visually detect three important events that receive the most MNIR: seeing the key (bottom row), seeing the door side (in the middle of the rightmost column) and approaching the front of the door (the second and fourth rows). Other less important cells are assigned very low MNIR. On the contrary, RND often gives high surprise-norm MNIR to cells around important ones, which creates a noisy MNIR map as in Fig. 4 (c). As a result, RND's performance is better than Baseline, yet far from that of RND+SM. Another analysis of how surprise novelty discriminates against surprises with similar norms is given in Appendix Fig. 8.

3.3 Atari: Sample-efficient Benchmark

We adopt the sample-efficiency Atari benchmark [17] on six hard exploration games where the training budget is only 50

Task	EMI♠	LWM♠	RND♠	NGU♢	LWM♢	LWM+SM♢	RND♢	RND+SM♢
Freeway	**33.8**	30.8	33.3	2.6	31.1	31.6	22.2	22.2
Frostbite	7002	8409	2227	1751	8598	***10258***	2628	*5073*
Venture	646	998	707	790	985	***1381***	1081	*1119*
Gravitar	558	1376	546	839	1,242	***1693***	739	*987*
Solaris	**2688**	1268	2051	1103	1839	*2065*	2206	*2420*
Montezuma	387	2276	377	2062	2192	2269	2475	***5187***
Norm. Mean	61.4	80.6	42.2	31.2	80.5	***97.0***	50.7	*74.8*
Norm. Median	34.9	60.8	32.7	33.1	66.5	*83.7*	58.3	***84.6***

Table 2: Atari: average return over 128 episodes after 50 million training frames (mean over 5 runs). ♠ is from a prior work [12]. ♢ is our run. The last two rows are mean and median human normalized scores. Bold denotes best results. Italic denotes that SG+SM is significantly better than SG regarding Cohen effect size less than 0.5.

million frames. We use our SM to augment 2 SGs: RND [8] and LWM [12]. Unlike RND, LWM uses a recurrent world model and forward dynamics to generate surprises. Details of the SGs, training and evaluation are in Appendix D.3. We run the SG and SG+SM in the same codebase and setting. Table 2 reports our and representative results from prior works, showing SM-augmented models outperform their SG counterparts in all games (same codebase). In Frostbite and Montezuma Revenge, RND+SM's score is almost twice as many as that of RND. For LWM+SM, games such as Gravitar and Venture observe more than 40% improvement. Overall, LWM+SM and RND+SM achieve the best mean and median human normalized score, improving 16% and 22% w.r.t the best SGs, respectively. Notably, RND+SM shows significant improvement for the notorious Montezuma Revenge. We also test the episodic memory baseline NGU to verify whether episodic memory on the state level is good enough in such a challenging benchmark. The result shows that NGU does not perform well within 50 million training frames, resulting in human normalized scores much lower than our LWM+SM and RND+SM.

We also verify the benefit of the SM in the long run for Montezuma Revenge and Frostbite. As shown in Fig. 5 (a,b), RND+SM still significantly outperforms RND after 200 million training frames, achieving average scores of 10,000 and 9,000, respectively. The result demonstrates the scalability of our proposed method. When using RND and RND+SM to compute the average MNIR in several rooms in Montezuma Revenge (Fig. 1), we realize that SM makes MNIR higher for surprising events in rooms with complex structures while depressing the MNIR of fake surprises in dark rooms. Here, even in the dark room, the movement of agents (human or spider) is hard to predict, leading to a high average MNIR. On the contrary, the average MNIR of surprise novelty is reduced if the prediction error can be recalled from the memory.

Finally, measuring the running time of the models, we notice little computing overhead caused by our SM. On our Nvidia A100 GPUs, LWM and LWM+SM's average time for one 50M training are 11h 38m and 12h 10m, respectively. For one 200M training, RND and RND+SM's average times are 26h 24m and 28h 1m, respectively. These correspond to only 7% more training time while the performance gap is significant (4000 scores).

3.4 Ablation Study

Role of Memories Here, we use Minigrid's Dynamic-Obstacle task to study the role of $\mathcal{M}$ and $\mathcal{W}$ in the SM (built upon RND as the SG). Disabling $\mathcal{W}$, we directly use $\|q_t\| = \left\|\left[u_t^e, u_t\right]\right\|$ as the intrinsic reward, and name this version: SM (no $\mathcal{W}$). To ablate the effect of $\mathcal{M}$, we remove u_t^e from q_t and only use $q_t = u_t$ as the query to $\mathcal{W}$, forming the version: SM (no $\mathcal{M}$). We also consider different episodic memory capacities and slot sizes N-d= $\{32-4, 128-16, 1024-64\}$. As N and d increase, the short-term context expands and more past surprise information is considered in the attention. In theory, a big $\mathcal{M}$ is helpful to capture long-term and more accurate context for constructing the surprise query.

Fig. 5 (c) depicts the performance curves of the methods after 10 million training steps. SM (no $\mathcal{W}$) and SM (no $\mathcal{M}$) show weak signs of learning, confirming the necessity of both modules in this task. Increasing N-d from $32-4$ to $1024-64$ improves the final performance. However, $1024-64$ is not significantly better than $128-16$, perhaps because it is unlikely to have similar surprises that are more than 128 steps apart. Thus, a larger attention span does not provide a benefit. As a result, we keep using $N = 128$ and $d = 16$ in all other experiments for faster computing. We also verify the necessity of $\mathcal{M}$ and $\mathcal{W}$ in Montezuma Revenge, Frostbite, Venture and illustrate how $\mathcal{M}$ generates lower MNIR when 2 similar event occurs in the same episode in Key-Door (see Appendix D.4).

No Task Reward In this experiment, we remove task rewards and merely evaluate the agent's ability to explore using intrinsic rewards. The task is to navigate 3D rooms and get a +1 reward for picking an object [9]. The state is the agent's image view, and there is no noise. Without task rewards, it is crucial to maintain the agent's interest in the unique events of seeing the objects. In this partially observable environment, surprise-prediction methods may struggle to explore even without noise due to a lack of information for good predictions, leading to usually high prediction errors. For this testbed, we evaluate random exploration agent (Baseline), RND and RND+SM in 2 settings: 1 room with three objects (easy), and 4 rooms with one object (hard).

To see the difference among the models, we compare the cumulative task rewards over 100 million steps (see Appendix D.4 for details). RND is even worse than Baseline in the easy setting because predicting causes high biases (intrinsic rewards) towards the unpredictable, hindering exploration if the map is simple. In

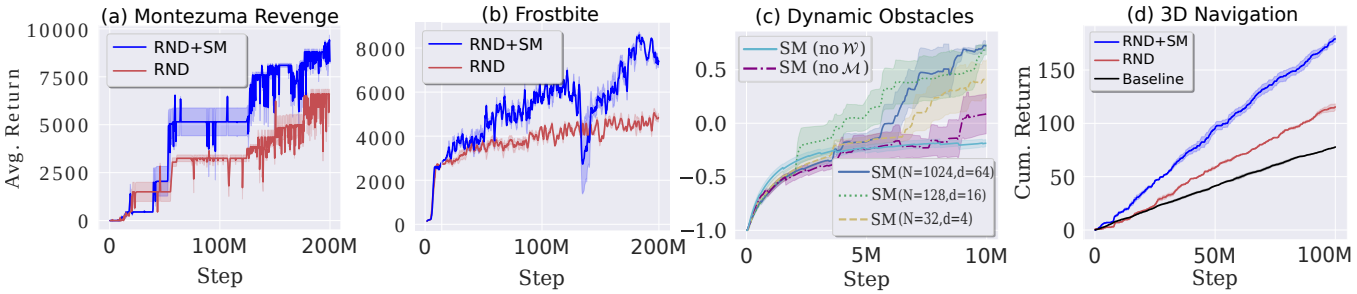

Figure 5: (a,b) Atari long runs over 200 million frames: average return over 128 episodes. (c) Ablation study on SM's components. (d) MiniWorld exploration without task reward: Cumulative task returns over 100 million training steps for the hard setting. The learning curves are mean±std. over 5 runs.

contrast, RND+SM uses surprise novelty, generally showing smaller intrinsic rewards (see Appendix Fig. 12 (right)). Consequently, our method consistently demonstrates significant improvements over other baselines (see Fig. 5 (d) for the hard setting).

4 RELATED WORKS

Intrinsic motivation approaches usually give the agent reward bonuses for visiting novel states to encourage exploration. The bonus is proportional to the mismatch between the predicted and reality, also known as surprise [31]. One kind of predictive model is the dynamics model, wherein the surprise is the error of the models as predicting the next state given the current state and action [1, 33]. One critical problem of these approaches is the unwanted bias towards transitions where the prediction target is a stochastic function of the inputs, commonly found in partially observable environments. Recent works focus on improving the features of the predictor's input by adopting representation learning mechanisms such as inverse dynamics [25], variational autoencoder, random/pixel features [7], or whitening transform [12]. Although better representations may improve the reward bonus, they cannot completely solve the problem of stochastic dynamics and thus, fail in extreme cases such as the noisy-TV problem [8].

Besides dynamics prediction, several works propose to predict other quantities as functions of the current state by using autoencoder [24], episodic memory [29], and random network [8]. Burda et al. (2018) claimed that using a deterministic random target network is beneficial in overcoming stochasticity issues. Other methods combine this idea with episodic memory and other techniques, achieving good results in large-scale experiments [2, 3]. From an information theory perspective, the notation of surprise can be linked to information gain or uncertainty, and predictive models can be treated as parameterized distributions [1, 15, 34]. Furthermore, to prevent the agent from unpredictable observations, the reward bonus can be measured by the progress of the model's prediction [1, 23, 30] or disagreement through multiple dynamic models [26, 32]. However, these methods are complicated and hard to scale, requiring heavy computing. A different angle to handle stochastic observations during exploration is surprsie minimization [6, 28]. In this direction, the agents get bigger rewards for seeing more familiar states. Such a strategy is somewhat opposite to our approach and suitable for unstable environments where the randomness occurs separately from the agents' actions.

These earlier works use surprise as an incentive for exploration and differ from our principle that utilizes surprise novelty. Also, our work augments these existing works with a surprise memory module and can be used as a generic plug-in improvement for surprise-based models. We note that our memory formulation differs from the memory-based novelty concept using episodic memory [3], momentum memory [13], or counting [5, 35] because our memory operates on the surprise level, not the state level. Moreover, our utilization of memory to store surprise novelty represents a novel approach compared to other memory-based RL methods that typically use memory for storing trajectories [20], policies [19, 22], and hyperparameters [18]. In our work, exploration is discouraged not only in frequently visited states but also in states whose surprises can be reconstructed using SM. Our work provides a more general and learnable novelty detection mechanism, which is more flexible than counting lookup table.

5 DISCUSSION

This paper presents Surprise Generator-Surprise Memory (SG+SM) framework to compute surprise novelty as an intrinsic motivation for the reinforcement learning agent. Exploring with surprise novelty is beneficial when there are repeated patterns of surprises or random observations. In Noisy-TV, our SG+SM can harness the agent's tendency to visit noisy states such as watching random TV channels while encouraging it to explore rare events with distinctive surprises. We empirically show that our SM can supplement four surprise-based SGs to achieve more rewards in fewer training steps in three grid-world environments. In 3D navigation without external reward, our method significantly outperforms the baselines. On two strong SGs, our SM also achieve superior results in hard-exploration Atari games within 50 million training frames. Even in the long run, our method maintains a clear performance gap from the baselines, as shown in Montezuma Revenge and Frostbite.

ACKNOWLEDGMENTS

This research was partially funded by the Australian Government through the Australian Research Council (ARC). Prof Venkatesh is the recipient of an ARC Australian Laureate Fellowship (FL170100006).

REFERENCES

[1] Joshua Achiam and Shankar Sastry. 2017. Surprise-based intrinsic motivation for deep reinforcement learning. *arXiv preprint arXiv:1703.01732* (2017).

[2] Adrià Puigdomènech Badia, Bilal Piot, Steven Kapturowski, Pablo Sprechmann, Alex Vitvitskyi, Zhaohan Daniel Guo, and Charles Blundell. 2020. Agent57: Outperforming the atari human benchmark. In *International Conference on Machine Learning*. PMLR, 507–517.

[3] Adrià Puigdomènech Badia, Pablo Sprechmann, Alex Vitvitskyi, Daniel Guo, Bilal Piot, Steven Kapturowski, Olivier Tieleman, Martin Arjovsky, Alexander Pritzel, Andrew Bolt, et al. 2019. Never Give Up: Learning Directed Exploration Strategies. In *International Conference on Learning Representations*.

[4] Andrew Barto, Marco Mirolli, and Gianluca Baldassarre. 2013. Novelty or surprise? *Frontiers in psychology* (2013), 907.

[5] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. 2016. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems* 29 (2016).

[6] Glen Berseth, Daniel Geng, Coline Manon Devin, Nicholas Rhinehart, Chelsea Finn, Dinesh Jayaraman, and Sergey Levine. 2020. SMiRL: Surprise Minimizing Reinforcement Learning in Unstable Environments. In *International Conference on Learning Representations*.

[7] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A Efros. 2018. Large-Scale Study of Curiosity-Driven Learning. In *International Conference on Learning Representations*.

[8] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. 2018. Exploration by random network distillation. In *International Conference on Learning Representations*.

[9] Maxime Chevalier-Boisvert. 2018. gym-miniworld environment for OpenAI Gym. https://github.com/maximecb/gym-miniworld.

[10] Maxime Chevalier-Boisvert, Lucas Willems, and Suman Pal. 2018. Minimalistic Gridworld Environment for OpenAI Gym. https://github.com/maximecb/gym-minigrid.

[11] Paul Ed Ekman and Richard J Davidson. 1994. *The nature of emotion: Fundamental questions*. Oxford University Press.

[12] Aleksandr Ermolov and Nicu Sebe. 2020. Latent world models for intrinsically motivated exploration. *Advances in Neural Information Processing Systems* 33 (2020), 5565–5575.

[13] Zheng Fang, Biao Zhao, and Guizhong Liu. 2022. Image Augmentation Based Momentum Memory Intrinsic Reward for Sparse Reward Visual Scenes. *arXiv preprint arXiv:2205.09448* (2022).

[14] Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401* (2014).

[15] Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. 2016. Vime: Variational information maximizing exploration. *Advances in neural information processing systems* 29 (2016).

[16] Laurent Itti and Pierre Baldi. 2005. Bayesian surprise attracts human attention. *Advances in neural information processing systems* 18 (2005).

[17] Hyoungseok Kim, Jaekyeom Kim, Yeonwoo Jeong, Sergey Levine, and Hyun Oh Song. 2019. EMI: Exploration with Mutual Information. In *International Conference on Machine Learning*. PMLR, 3360–3369.

[18] Hung Le, Majid Abdolshah, Thommen K George, Kien Do, Dung Nguyen, and Svetha Venkatesh. 2022. Episodic policy gradient training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 7317–7325.

[19] Hung Le, Thommen Karimpanal George, Majid Abdolshah, Dung Nguyen, Kien Do, Sunil Gupta, and Svetha Venkatesh. 2022. Learning to Constrain Policy Optimization with Virtual Trust Region. *Advances in Neural Information Processing Systems* 35 (2022), 12775–12786.

[20] Hung Le, Thommen Karimpanal George, Majid Abdolshah, Truyen Tran, and Svetha Venkatesh. 2021. Model-based episodic memory induces dynamic hybrid controls. *Advances in Neural Information Processing Systems* 34 (2021), 30313–30325.

[21] Hung Le, Truyen Tran, and Svetha Venkatesh. 2019. Learning to remember more with less memorization. *arXiv preprint arXiv:1901.01347* (2019).

[22] Hung Le and Svetha Venkatesh. 2022. Neurocoder: General-purpose computation using stored neural programs. In *International Conference on Machine Learning*. PMLR, 12204–12221.

[23] Manuel Lopes, Tobias Lang, Marc Toussaint, and Pierre-Yves Oudeyer. 2012. Exploration in model-based reinforcement learning by empirically estimating learning progress. *Advances in neural information processing systems* 25 (2012).

[24] Mikkel Sannes Nylend. 2017. *Data Efficient Deep Reinforcement Learning through Model-Based Intrinsic Motivation*. Master's thesis. NTNU.

[25] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. 2017. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*. PMLR, 2778–2787.

[26] Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. 2019. Self-supervised exploration via disagreement. In *International conference on machine learning*. PMLR, 5062–5071.

[27] Adityanarayanan Radhakrishnan, Mikhail Belkin, and Caroline Uhler. 2020. Overparameterized neural networks implement associative memory. *Proceedings of the National Academy of Sciences* 117, 44 (2020), 27162–27170.

[28] Nicholas Rhinehart, Jenny Wang, Glen Berseth, John D Co-Reyes, Danijar Hafner, Chelsea Finn, and Sergey Levine. 2021. Intrinsic Control of Variational Beliefs in Dynamic Partially-Observed Visual Environments. In *ICML 2021 Workshop on Unsupervised Reinforcement Learning*. https://openreview.net/forum?id=gb5rUScIY5x

[29] Nikolay Savinov, Anton Raichuk, Damien Vincent, Raphael Marinier, Marc Pollefeys, Timothy Lillicrap, and Sylvain Gelly. 2018. Episodic Curiosity through Reachability. In *International Conference on Learning Representations*.

[30] Jürgen Schmidhuber. 1991. Curious model-building control systems. In *Proc. international joint conference on neural networks*. 1458–1463.

[31] Jürgen Schmidhuber. 2010. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE transactions on autonomous mental development* 2, 3 (2010), 230–247.

[32] Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. 2020. Planning to explore via self-supervised world models. In *International Conference on Machine Learning*. PMLR, 8583–8592.

[33] Bradly C Stadie, Sergey Levine, and Pieter Abbeel. 2015. Incentivizing Exploration In Reinforcement Learning With Deep Predictive Models. *arXiv e-prints* (2015), arXiv–1507.

[34] Susanne Still and Doina Precup. 2012. An information-theoretic approach to curiosity-driven reinforcement learning. *Theory in Biosciences* 131, 3 (2012), 139–148.

[35] Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. 2017. # exploration: A study of count-based exploration for deep reinforcement learning. *Advances in neural information processing systems* 30 (2017).