POSTERIOR GRADIENT ESTIMATION CHANGES THE CAPABILITIES OF INTRINSICALLY DRIVEN AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Changing only posterior gradient estimation in world models with discrete latent states can lead intrinsically driven agents to acquire markedly different capabilities. We investigate this effect by changing how gradients are estimated through posterior samples across intrinsic-drive methods. Experiments across methods and tasks reveal substantial changes in performance, with both the magnitude and direction depending on the drive and task. Policies with similar aggregate scores can also acquire markedly different skill profiles. These findings show that posterior gradient estimation affects both performance and the skills acquired under intrinsic rewards.

1 INTRODUCTION

Intrinsic drives support autonomous exploration and behavior learning, reducing agents' reliance on task rewards. When task rewards are sparse or absent, agents can obtain exploration signals from their own prediction errors (Pathak et al., 2017), or construct intrinsic rewards from interaction history (Henaff et al., 2022) or internal needs (Keramati & Gutkin, 2014). Prior work shows that curiosity rewards alone can produce useful behavior across games (Burda et al., 2019a), and that knowledge acquired through autonomous exploration can support subsequent task adaptation (Sekar et al., 2020). These advances highlight the value of intrinsic drives for autonomous learning and the importance of understanding what agents learn from them.

World models support this autonomous learning by predicting action consequences. Agents can learn policies in imagined trajectories generated by the model (Hafner et al., 2020) and use intrinsic returns to evaluate actions before executing them. For example, Plan2Explore (P2E) selects exploratory behavior using ensemble disagreement about future states (Sekar et al., 2020). Studies that integrate different intrinsic drives into a shared world model further demonstrate this framework's role in unsupervised pretraining and task adaptation (Rajeswar et al., 2023). In these methods, intrinsic rewards work together with model predictions to guide actions. Understanding the resulting capabilities therefore also requires examining how the model learns the state representations and predictions used to evaluate actions.

In world models with discrete latent states, the model infers a latent-state distribution from current observations and history, then samples from it for subsequent predictions. Propagating prediction-loss gradients through these discrete posterior samples requires a gradient estimator, because ordinary differentiation through discrete sampling does not provide useful training gradients (Bengio et al., 2013). DreamerV3 uses this representation and trains its posterior network with straight-through estimation (Hafner et al., 2025). Different straight-through estimators can preserve the same forward sample under fixed parameters and random numbers while returning different gradients to the network (Paulus et al., 2021). Even with unchanged reward computation and model architecture, posterior gradient estimation may therefore alter the predictions the model learns, affecting policy choices and the experiences subsequently collected.

Prior estimator research examines gradient bias and variance (Mohamed et al., 2020) and compares learning outcomes in generative modeling and other tasks (Paulus et al., 2021; Fan et al., 2022). Related comparisons in reinforcement learning also examine how discrete-action gradient estimators affect task performance (Tilbury et al., 2023). However, there has been limited systematic study of how posterior gradient estimation in world models affects the capabilities of agents trained with different intrinsic drives. Methods evaluate experience using different signals. Random Network

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Distillation (RND) measures novelty through image-feature prediction error (Burda et al., 2019b), P2E evaluates ensemble disagreement about future states (Sekar et al., 2020), and homeostatic reinforcement learning rewards the relief of bodily needs (Keramati & Gutkin, 2014). These methods may require different state information and future predictions, so they need not respond alike to the same change in posterior gradient estimation. We study how posterior gradient estimation affects their performance and acquired skills, and examine the role of the correspondence between actual samples and backward surrogates.

To investigate this question, we keep each method's reward computation, model architecture, and forward sampling rule fixed within a DreamerV3 backbone. We compare Straight-Through (ST; Paulus et al., 2021), Straight-Through Gumbel-Softmax (ST-GS; Jang et al., 2017), and Gapped Straight-Through (GST; Fan et al., 2022). These existing estimators provide different posterior backward rules, allowing us to examine how capabilities change under the same reward design. Our experiments combine cross-task performance comparisons with evaluation of all Crafter achievements. An independent-backward-noise control further tests the correspondence between the surrogate and the actual posterior sample.

Posterior gradient estimation substantially affects the performance of intrinsically driven agents, with effects that vary in magnitude and direction across methods and tasks and alter their relative performance. It also affects which behaviors agents learn. Even with the same intrinsic reward and similar aggregate scores, different posterior gradient estimators can lead policies to acquire markedly different skill profiles. Our main contributions are:

- We show that posterior gradient estimation substantially affects the performance of intrinsically driven agents, with effects that vary in magnitude and direction across methods and tasks.
- We find that posterior gradient estimation can produce large changes in skill success rates and different skill profiles despite similar aggregate scores.
- We identify the correspondence between forward sampling and backward gradient computation as an important factor affecting the learning performance of intrinsically driven agents.

2 Related Work

2.1 Intrinsic Drives and World Models

Intrinsic-drive learning depends on both reward construction and its implementation. Feature representations affect prediction-error-driven learning (Burda et al., 2019a), while policy optimization affects disagreement-driven exploration (Pathak et al., 2019). RLeXplore identifies implementation and optimization choices that influence intrinsic-reward methods, including normalization and representation learning (Yuan et al., 2025). Recent MiniGrid studies examine how intrinsic rewards change exploration under different observations (Kayal et al., 2025) and how adding them changes behavior alongside task returns (Villalobos-Arias et al., 2025). Together, these studies motivate evaluating the learning process and behavior, beyond specifying a reward formula.

World-model-based research uses predictions of future experience to guide intrinsically driven behavior. Dreamer learns latent dynamics and reward predictions and trains policies in imagination (Hafner et al., 2020), with DreamerV3 extending this framework across visual control tasks (Hafner et al., 2025). Plan2Explore uses ensemble disagreement to guide policy learning in imagination and data collection (Sekar et al., 2020), while Information is Power learns information-seeking control through latent visitation entropy (Rhinehart et al., 2021). Work using RND examines how exploration policies and dynamics models contribute to subsequent transfer (Walker et al., 2023), and comparisons within a shared Dreamer backbone examine pretraining and adaptation across intrinsic drives (Rajeswar et al., 2023). These studies connect intrinsic rewards, model predictions, and behavior learning. Later work examines interactions between latent learning and exploration (Mantiuk et al., 2025) and extends reward sources through change-based rewards (Ferrao & Cunha, 2025) and SENSEI's combination of semantic guidance and model uncertainty (Sancaktar et al., 2025).

Research on model training also shows that prediction targets and their optimization affect control performance. HarmonyDream balances observation and reward modeling losses (Ma et al., 2024),

and DreamSmooth changes reward-prediction targets through temporal smoothing (Lee et al., 2024). Following this connection between model predictions and behavior, we examine gradient estimation at discrete posterior nodes. We keep each method's reward computation and model architecture fixed and test how the performance and skills acquired under different intrinsic drives change.

2.2 Straight-through Gradient Estimation for Discrete Latent States

Straight-through estimators train networks containing discrete variables through backward surrogates (Bengio et al., 2013). Gumbel-Softmax (Jang et al., 2017) and Concrete (Maddison et al., 2017) provide continuous relaxations of discrete sampling. DreamerV2 (Hafner et al., 2021) and DreamerV3 (Hafner et al., 2025) use straight-through estimation to train discrete latent states, making gradient estimation an implementation choice in world-model learning.

Estimator research studies bias, variance, and computational cost (Mohamed et al., 2020). Gumbel-Rao uses conditional averaging to reduce variance (Paulus et al., 2021), and the zero-temperature Gumbel-Rao (ZGR) estimator has a computationally simple form (Shekhovtsov, 2023). GST constructs a gapped backward surrogate (Fan et al., 2022), while ReinMax improves gradient approximation (Liu et al., 2023) and later work incorporates variance reduction (Wang & Bui, 2026). Recent analyses characterize bias–variance trade-offs within estimator families (Hooper & Shekhovtsov, 2026). Decoupled ST separates forward and backward temperatures (Shah et al., 2026), and ReDGE derives categorical gradient estimators from a diffusion-based relaxation (Gourevitch et al., 2026). These studies distinguish the sampling rule from the derivative used for learning, providing a basis for controlled posterior-gradient interventions.

Task-level studies test how these choices affect learning. GST has been evaluated on variational autoencoders with discrete latent variables and ListOps (Fan et al., 2022). Comparisons in Multi-Agent Deep Deterministic Policy Gradient (MADDPG) examine discrete action estimators (Tilbury et al., 2023), while communication-learning experiments examine estimators for discrete messages (Vanneste et al., 2026). Our study applies existing estimators to world-model observation posteriors and tests their effects across intrinsic drives. Its focus is the performance and skills acquired under different backward rules.

3 Study Framework

3.1 Prediction Losses and the Observation Posterior

Fig. 1a outlines how reward and return targets contribute to world-model learning through prediction losses and marks the posterior-gradient intervention. Our DreamerV3 backbone learns latent states from observation–action histories and trains a policy on imagined trajectories generated by the model (Hafner et al., 2025). Intrinsic rewards define the returns used for policy optimization and also contribute to latent-state learning through reward or value prediction losses. A reward describes feedback from one transition, whereas a return aggregates future rewards with discounting. We follow the backward path from the prediction loss to the posterior sample, then examine how estimators propagate this gradient to the posterior network.

DreamerV3 uses a recurrent state-space model (RSSM; Hafner et al., 2019; 2025) to represent environmental states and learn how they change with actions. Let o_t denote the observation and a_t the action. The latent state consists of a continuous deterministic state d_t and a discrete stochastic state z_t. The deterministic state summarizes previous states and actions. Together with the current observation, it determines a distribution over z_t, called the observation posterior and denoted by q_θ. We use DreamerV3's discrete stochastic representation, in which z_t comprises multiple categorical variables.

$$d_t = f_\theta(d_{t-1}, z_{t-1}, a_{t-1}), \qquad q_\theta(z_t^{(m)} \mid d_t, o_t) = \mathrm{Cat}(\mathrm{softmax}(\ell_t^{(m)})). \tag{1}$$

Here, θ denotes model parameters, m indexes a group of discrete latent variables, $\ell_t^{(m)}$ contains its posterior logits, and Cat denotes a categorical distribution. The groups are conditionally independent given d_t and o_t and jointly form z_t. The resulting latent state supports observation reconstruction, dynamics learning, and reward and value prediction.

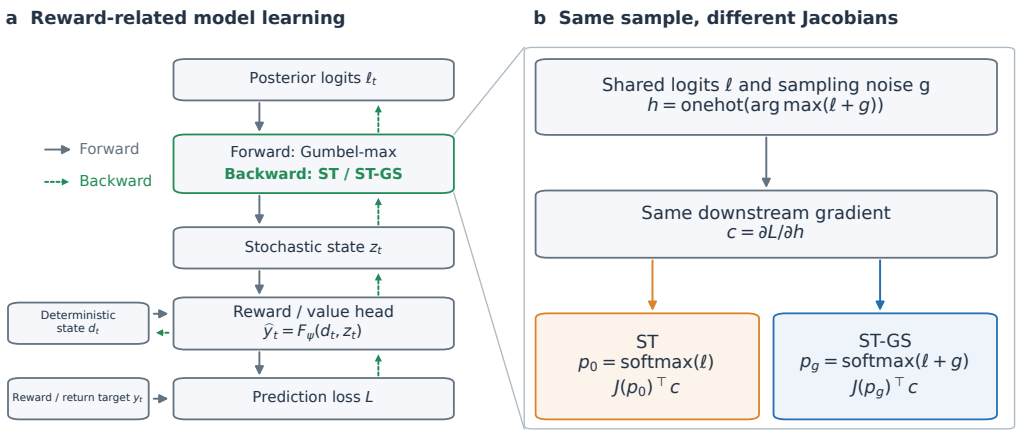

Figure 1: Prediction losses associated with intrinsic rewards and posterior gradient estimation. (a) Reward and return targets contribute to DreamerV3 model updates through prediction losses. (b) ST and ST-GS illustrate the backward rules for one group of discrete latent variables in (a). With fixed parameters and random numbers, they share the forward sample but use different surrogates to propagate a given downstream gradient to the posterior logits. GST acts at the same node; Section 3.2 defines its surrogate.

Reward and value heads read the latent state and predict immediate rewards and expected future returns, respectively. Let one such head be F_ψ, with parameters ψ and output $\widehat{y}_t = F_\psi(d_t, z_t)$. Its target y_t is the corresponding reward or return estimate. The prediction loss $L(\widehat{y}_t, y_t)$ measures the difference between output and target. Treating y_t as constant, we denote the loss gradient with respect to the mth stochastic group by c, using column vectors for gradients:

$$c = \frac{\partial L(\widehat{y}_t, y_t)}{\partial z_t^{(m)}} = \left(\frac{\partial F_\psi(d_t, z_t)}{\partial z_t^{(m)}}\right)^\top \frac{\partial L}{\partial \widehat{y}_t}. \tag{2}$$

The head derivative describes how the latent state affects the prediction, and the loss derivative describes how the prediction affects the loss. Together, they determine the gradient returned to the latent state. The next section compares how estimators propagate this gradient through discrete sampling. Fig. 1 emphasizes reward-related prediction paths, but the intervention also affects other gradients that pass through posterior samples, including reconstruction gradients. Appendix D.1 gives the complete learning paths.

3.2 Controlled Comparison of Posterior Gradient Estimators

Fig. 1b illustrates how ST and ST-GS propagate gradients through the same discrete sample using different backward surrogates. We use three existing estimators, ST, ST-GS, and GST, whose surrogates depend on the original categorical probabilities, the same sampling noise, and the sampled category, respectively. With fixed parameters, inputs, and sampling random numbers, they produce the same forward sample but use different surrogate derivatives. Our experiments keep reward computation and model architecture fixed within each method to test how this difference in backward computation affects acquired capabilities.

Discrete category selection does not provide useful training gradients through ordinary differentiation. Straight-through estimators preserve the discrete forward sample and use the derivative of a continuous function in the backward pass (Bengio et al., 2013; Jang et al., 2017). We sample the posterior with Gumbel-max (Jang et al., 2017). Omitting time and group indices, let h denote the one-hot sample $z_t^{(m)}$ and i index categories within the group:

$$g_i = -\log[-\log u_i], \quad u_i \sim \mathrm{Uniform}(0,1), \qquad h = \mathrm{one_hot}\left(\arg\max_i(\ell_i + g_i)\right). \tag{3}$$

Independent Gumbel noise for each category makes h follow the categorical distribution defined by the logits. All three estimators use this forward sampling rule. Following Paulus et al. (2021),

we use ST for the straight-through estimator with categorical probabilities as its backward surrogate, and ST-GS for Straight-Through Gumbel-Softmax. ST has the gradient form used for discrete stochastic states in DreamerV2 (Hafner et al., 2021) and our DreamerV3 implementation (Hafner et al., 2025). ST-GS keeps the discrete forward sample and uses the derivative of a continuous relaxation constructed from the same Gumbel noise (Jang et al., 2017). Let sg denote stop-gradient. Both surrogates use softmax temperature 1:

$$\begin{aligned} p_0 &= \mathrm{softmax}(\ell), & z_{\mathrm{ST}} &= p_0 + \mathrm{sg}(h - p_0), \\ p_g &= \mathrm{softmax}(\ell + \mathrm{sg}(g)), & z_{\text{ST-GS}} &= p_g + \mathrm{sg}(h - p_g). \end{aligned} \tag{4}$$

Both forward outputs equal the one-hot state h. Their backward passes differentiate p_0 and p_g with respect to the logits. The former excludes sampling noise, while the latter includes the noise used to draw the sample. Suppose A has the highest probability in the original distribution, but B is sampled. Both estimators send B to the model. ST backpropagates through p_0, where A still has the highest probability, whereas ST-GS backpropagates through p_g, where the sampling noise makes B most probable. Even with the same downstream gradient evaluated at sample B, the gradients of individual logits can differ in magnitude and even sign.

Using the prediction loss from Section 3.1, let $c = \partial L/\partial h$ be the gradient returned to the sample and $J(p) = \mathrm{diag}(p) - pp^\top$ the softmax Jacobian. For the same logits, forward sample, and c, the estimated loss gradients with respect to the logits are

$$\begin{aligned} \widehat{\nabla_\ell L}_{\mathrm{ST}} &= J(p_0)^\top c, \\ \widehat{\nabla_\ell L}_{\text{ST-GS}} &= J(p_g)^\top c. \end{aligned} \tag{5}$$

In Eq. (5), c is the gradient propagated from the prediction loss through the prediction head to the sampled state. ST uses the derivative associated with the original probabilities; ST-GS uses that associated with probabilities containing the current sampling noise. Appendix A.1 expands these expressions by category.

GST (Gapped Straight-Through) preserves the same forward sample h and constructs a gapped backward surrogate from the selected category (Fan et al., 2022). Let k be the sampled category, $M = \max_j \ell_j$ the largest original logit, and $\kappa > 0$ the gap. The adjustment for each category is

$$\Delta_i = \begin{cases} M - \ell_i, & i = k, \\ -\max(0, \ell_i + \kappa - M), & i \neq k. \end{cases} \tag{6}$$

This raises the selected category's surrogate logit to M and caps the other surrogate logits at $M - \kappa$, giving the selected category a margin of at least κ. Applying stop-gradient to the adjustment yields

$$p_{\mathrm{GST}} = \mathrm{softmax}(\ell + \mathrm{sg}(\Delta)), \qquad z_{\mathrm{GST}} = p_{\mathrm{GST}} + \mathrm{sg}(h - p_{\mathrm{GST}}). \tag{7}$$

We use gap $\kappa = 1$ and softmax temperature 1. For example, with original logits $(2, 1, 0)$ and sampled category B, GST adjusts the surrogate logits to $(1, 2, 0)$. The forward pass still outputs B, while the backward pass uses the adjusted softmax derivative to propagate the downstream gradient to the original logits. Given the original logits and selected category, the surrogate is fixed; it does not require the full noise vector that caused B to be selected. GST thus provides another sample-dependent backward rule for testing how surrogate construction affects acquired capabilities. The distinction concerns how the surrogate uses sampling information. The full ST gradient also contains the downstream gradient from the actual sample.

The three-estimator comparison tests responses to posterior gradient rules. An additional independent-backward-noise control tests the correspondence between the surrogate and the actual sample. It generates h as before but replaces g in the ST-GS surrogate with independent, identically distributed noise g'. At fixed logits, the forward-sample distribution and backward-surrogate distribution are each unchanged, but the surrogate no longer uses the random numbers that produced the sample. Appendix A.2 analyzes the local gradient effect of this change.

4 Experimental Setup

Environments. Our main experiments use Crafter, a two-dimensional survival environment in which agents gather resources, craft tools, and manage survival needs. Its 22 achievements track

Table 1: Final Crafter scores under three posterior gradient estimators.

Method	ST	ST-GS	GST
HD	7.05 $\pm$ 1.04	29.51 $\pm$ 2.51	32.20 $\pm$ 2.97
RND	20.02 $\pm$ 1.68	24.57 $\pm$ 2.10	24.40 $\pm$ 1.20
P2E	21.90 $\pm$ 2.53	22.68 $\pm$ 2.08	31.21 $\pm$ 3.45
E3B	18.49 $\pm$ 0.33	20.56 $\pm$ 3.83	22.76 $\pm$ 3.24
ICM	13.72 $\pm$ 5.13	21.68 $\pm$ 0.10	20.61 $\pm$ 1.63
Task Reward	70.63 $\pm$ 1.94	73.24 $\pm$ 0.95	69.98 $\pm$ 0.87

different skills (Hafner, 2022), allowing us to examine both aggregate performance and skill profiles. Cross-task experiments use Ms. Pac-Man and Seaquest in the Arcade Learning Environment (ALE; Bellemare et al., 2013), with Walker Walk from the DeepMind Control Suite as an additional control task (Tassa et al., 2018).

Backbone and implementation. All intrinsic-drive methods use DreamerV3 as a shared model-based reinforcement learning backbone (Hafner et al., 2025). Crafter experiments use approximately 25M parameters and 50M environment steps. Atari and Walker use smaller model configurations from the corresponding domains. All experiments are run with five independent training seeds. The implementation uses JAX, with training on NVIDIA GeForce RTX 4090 GPUs. Within each method and task, only the posterior gradient estimator changes. Appendix D lists the training configurations.

Intrinsic-drive methods. Crafter experiments cover RND (Burda et al., 2019b), P2E (Sekar et al., 2020), Exploration via Elliptical Episodic Bonuses (E3B; Henaff et al., 2022), Intrinsic Curiosity Module (ICM; Pathak et al., 2017), and a homeostatic drive-based method (HD). E3B computes exploration rewards from within-episode state visits, while ICM uses action-conditioned feature-prediction error. HD follows the homeostatic reinforcement learning principle of rewarding need reduction (Keramati & Gutkin, 2014). In Crafter, it sums the squared deficits in food, water, energy, and health relative to their target levels to define homeostatic drive, and uses its reduction as reward. Each method preserves the key structure of its reward definition while being adapted to the shared backbone. Appendix D.2 gives the implementations.

Comparisons and evaluation. We compare ST, ST-GS, and GST for all five intrinsic-drive methods in Crafter, measuring aggregate scores and skill success rates, and conduct the independent-backward-noise control. In Atari and Walker Walk, we compare ST and ST-GS for RND and P2E. The Atari experiments also include GST. Intrinsic-drive methods train only on their own intrinsic rewards, with a separate task-reward control in Crafter. We evaluate frozen final policies on 256 environment instances per task, initialized with the same evaluation seeds across models. Crafter results report the official score and success rates for all 22 achievements (Hafner, 2022); Atari and Walker results report native returns. We compute each metric separately for each training run and then average across runs with equal weights. Appendix D.4 defines pairing and standard deviations, and Appendix B provides the full results.

5 Results and Analysis

5.1 Method-dependent Performance Changes

Table 1 summarizes Crafter scores under the three posterior gradient estimators, showing different responses across intrinsic-drive methods. HD scores poorly with ST and improves substantially with ST-GS and GST. RND and ICM also improve, but by smaller amounts. P2E and RND have similar mean scores under ST and ST-GS, but P2E scores substantially higher under GST. RND performs similarly under ST-GS and GST, ICM scores higher under ST-GS, and E3B scores higher under GST. Posterior gradient estimation therefore affects capability development differently across drives and can change their relative performance.

HD computes rewards from a fixed function of physiological state, while RND and ICM generate rewards with image modules separate from the world model. Their performance changes therefore arise even when reward computation does not directly use the posterior gradient estimator. The agent must still learn to predict and use these rewards through the world model. Posterior gradient rules

Table 2: Native returns of final policies in Atari and Walker Walk.

Task	Intrinsic drive	ST	ST-GS
Ms. Pac-Man	RND	2580.49 $\pm$ 188.06	2025.34 $\pm$ 285.39
Ms. Pac-Man	P2E	1033.93 $\pm$ 106.29	1592.97 $\pm$ 194.85
Seaquest	RND	618.09 $\pm$ 26.94	737.02 $\pm$ 152.06
Seaquest	P2E	4080.86 $\pm$ 1727.72	7366.22 $\pm$ 3760.32
Walker Walk	RND	42.78 $\pm$ 12.00	74.60 $\pm$ 22.82
Walker Walk	P2E	244.19 $\pm$ 70.46	155.77 $\pm$ 42.85

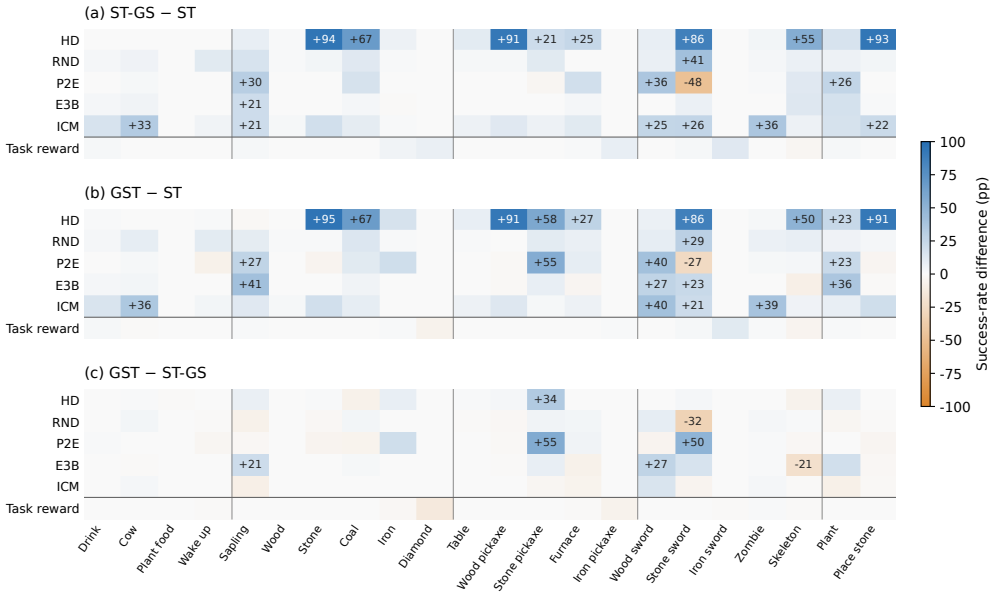

Figure 2: Skill changes under three posterior gradient estimators in Crafter. The panels show success-rate differences for ST-GS−ST, GST−ST, and GST−ST-GS. Values are annotated when the absolute difference is at least 20 percentage points.

can affect this process through latent-state learning and the experiences collected by the resulting policy.

Table 2 compares final-policy returns in Atari and Walker Walk, showing that the direction of the effect also varies across tasks. In Ms. Pac-Man, RND performs better under ST, whereas P2E performs better under ST-GS. The relative performance of ST and ST-GS is reversed for both methods in Walker Walk. In Seaquest, both methods achieve higher mean returns under ST-GS, with a larger improvement and greater variation across training runs for P2E. Reward type alone thus does not determine sensitivity. The effect also needs to be understood in relation to the state information, behavioral demands, and learning configuration of each task.

5.2 Effects on Skill Profiles

Crafter's 22 achievements record resource gathering, tool crafting, and survival skills, providing a more detailed view of the capabilities affected by posterior gradient estimation. Fig. 2 shows both joint improvements and simultaneous gains and losses across skills. ST-GS and GST substantially improve stone collection, basic tool crafting, and stone-sword crafting for HD. ICM improves across eating, resource gathering, crafting, and combat. For P2E, switching from ST to ST-GS increases wood-sword crafting and planting success but reduces stone-sword crafting. The same switch increases stone-sword success for RND, showing that skill changes also depend on the intrinsic drive.

The absolute success rates in Fig. 3 show effects on both acquiring basic tools and developing subsequent skills. Under ST, HD often crafts a wood sword but rarely a wood pickaxe. ST-GS increases

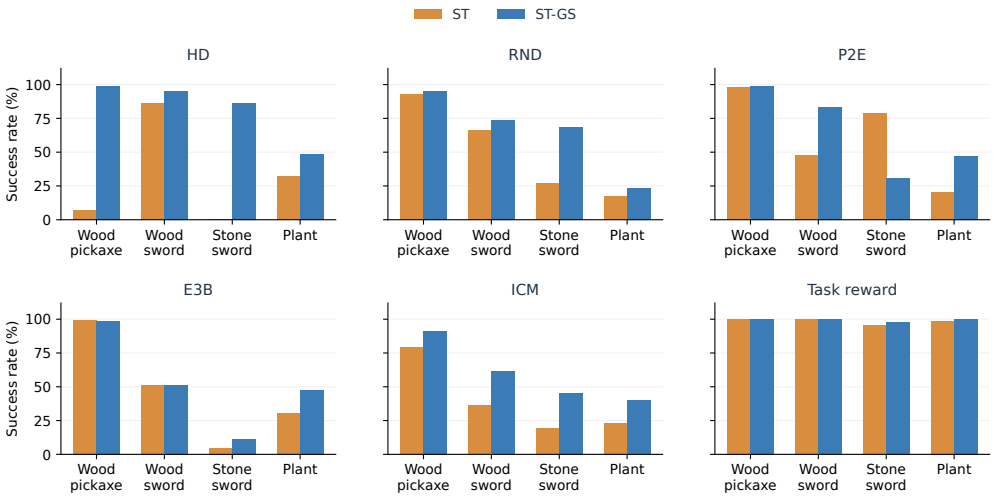

Figure 3: Success rates for the same skills across methods in Crafter.

both wood-pickaxe and stone-sword success. RND and P2E already achieve high wood-pickaxe success under both estimators, yet their stone-sword success changes in opposite directions. Thus, similar success in acquiring a basic tool does not imply similar success in developing subsequent skills.

Different estimator pairs can produce similar aggregate scores with different skill profiles. P2E has similar ST and ST-GS scores, but stone-sword success is approximately 79% and 31%, respectively, while wood-sword success is approximately 48% and 84% (Fig. 3). RND also has similar ST-GS and GST scores, despite a roughly 32-percentage-point reduction in stone-sword success under GST (Table 1 and Fig. 2c). Crafter geometrically aggregates skill success rates, so gains and losses contribute in opposite directions to the score; Appendix B.7 gives the full decomposition. Similar scores can therefore conceal substantial, offsetting changes in skill success rates. With intrinsic-reward computation fixed, posterior gradient estimation can change both overall performance and which skills agents acquire.

5.3 Correspondence between Forward Samples and Backward Surrogates

We test how the correspondence between the backward surrogate and the actual posterior sample affects intrinsic-drive learning. ST-GS uses the same Gumbel noise to generate the discrete sample and the surrogate, so the sampled category is also the most probable category in the surrogate. The same noise therefore determines both the sample and the derivative used to backpropagate its loss. To test this link, we keep forward sampling unchanged and construct the backward surrogate with a separate, identically distributed Gumbel draw in Crafter. If B is sampled, for example, the independent surrogate may assign its highest probability to A while still processing the loss gradient returned from B. At fixed logits, this control preserves the forward-sample distribution, backward-surrogate distribution, and temperature, changing only their correspondence. Fig. 4 compares final performance under these two settings across methods.

Independent backward noise reduces HD's score from 31.0 to 6.5 and RND's from 26.8 to 16.6. P2E, ICM, and task-reward training show smaller decreases, whereas E3B improves from 17.8 to 23.9 (Fig. 4). As Section 3.2 shows, the loss gradient returns from the sampled latent state and reaches the posterior logits through the surrogate derivative. Even when the derivative has the same marginal distribution at fixed logits, its product with the sample-dependent downstream gradient can have different statistics. Appendix A.2 decomposes the local expected gradient. These results identify the correspondence between forward samples and backward surrogates as a factor in learning performance, with different effects across intrinsic drives.

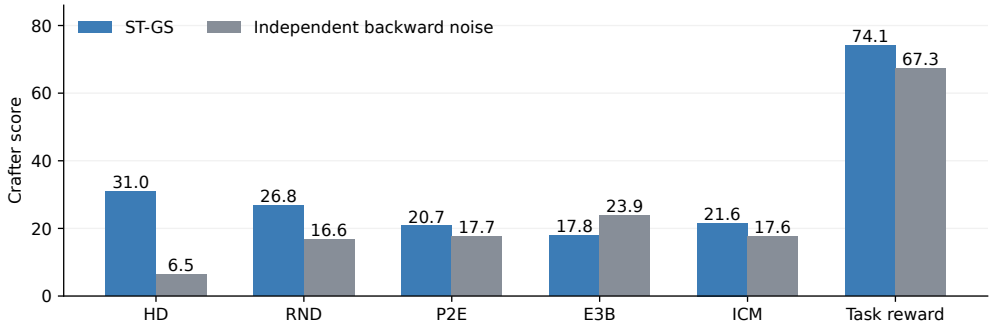

Figure 4: Effects of changing the correspondence between backward noise and forward samples in Crafter. Final Crafter scores for ST-GS and the independent-backward-noise control.

6 Discussion and Limitations

6.1 Possible Sources of Differential Sensitivity

Differences in sensitivity may relate to how reward-related prediction targets shape representation learning. HarmonyDream shows that the balance between observation and reward modeling affects control performance (Ma et al., 2024). In our setting, different intrinsic drives provide different reward and return targets. Posterior gradient estimation may affect how well the model learns to predict these targets, leading to representation differences that influence action selection.

Another possible pathway is feedback between exploration and model learning. Plan2Explore uses a model to guide exploration and updates the model with new experience (Sekar et al., 2020). RND rewards also change as its predictor learns from collected data (Burda et al., 2019b). These mechanisms could allow posterior-gradient effects to extend from predictions to data collection and subsequent learning. The open question is whether this feedback amplifies the differences observed here, and whether fixed rewards and rewards that evolve with experience respond differently.

6.2 Capability Evaluation and Scope

Achieving higher task scores and maximizing expected intrinsic return are distinct goals. The original RND study observes that agents trained only on intrinsic rewards need not maintain high task returns once task-related interactions become repetitive (Burda et al., 2019b). Within each method, the effects of posterior gradient estimation on expected intrinsic return and acquired capabilities should be examined together.

These explanations require further testing. The available evidence does not separate the contributions of representation learning, prediction errors, and interaction feedback. Cross-task tests mainly cover RND and P2E, while detailed skill analysis focuses on Crafter, limiting the scope of the conclusions. Further work should prioritize identifying which decision-relevant information is affected by posterior gradients and how these changes accumulate into capability differences through interaction.

7 Conclusion

We find that changing only posterior gradient estimation in world models with discrete latent states can lead intrinsically driven agents to acquire markedly different capabilities. The magnitude and direction of performance changes depend on the drive and task. Skill changes include both joint improvements and simultaneous gains and losses across skills. The independent-backward-noise control further shows that changing the correspondence between the surrogate and the actual posterior sample affects learning, with effects that also vary across methods. Understanding these effects on intrinsically driven behavior requires evaluating both aggregate performance and individual skills.

AI USE STATEMENT

Generative AI tools assisted with literature review, code implementation, and manuscript editing.

REPRODUCIBILITY STATEMENT

Section 4 describes the experimental setup. Appendix D provides method implementations, training configurations, and evaluation procedures. Appendix C details the protocols for local prediction tests.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Deep Reinforcement Learning at the Edge of the Statistical Precipice. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 29304–29320, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/f514cec81cb148559cf475e7426eed5e-Abstract.html.

Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The Arcade Learning Environment: An Evaluation Platform for General Agents. *Journal of Artificial Intelligence Research (JAIR)*, 47:253–279, 2013. URL https://doi.org/10.1613/jair.3912.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation. arXiv:1308.3432, 2013. URL https://arxiv.org/abs/1308.3432.

Glen Berseth, Daniel Geng, Coline Devin, Nicholas Rhinehart, Chelsea Finn, Dinesh Jayaraman, and Sergey Levine. SMiRL: Surprise Minimizing Reinforcement Learning in Unstable Environments. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=cPZOyoDloxl.

Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A. Efros. Large-Scale Study of Curiosity-Driven Learning. In *International Conference on Learning Representations (ICLR)*, 2019a. URL https://openreview.net/forum?id=rJNwDjAqYX.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by Random Network Distillation. In *International Conference on Learning Representations (ICLR)*, 2019b. URL https://openreview.net/forum?id=H1lJJnR5Ym.

Ting-Han Fan, Ta-Chung Chi, Alexander I. Rudnicky, and Peter J. Ramadge. Training Discrete Deep Generative Models via Gapped Straight-Through Estimator. In *International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pp. 6059–6073. PMLR, 2022. URL https://proceedings.mlr.press/v162/fan22a.html.

Jeremias Lino Ferrao and Rafael F. Cunha. World Model Agents with Change-Based Intrinsic Motivation. In *Proceedings of the Northern Lights Deep Learning Conference (NLDL)*, volume 265 of *Proceedings of Machine Learning Research*, pp. 66–74. PMLR, 2025. URL https://proceedings.mlr.press/v265/ferrao25a.html.

Samson Gourevitch, Alain Oliviero Durmus, Eric Moulines, Jimmy Olsson, and Yazid Janati. Categorical Reparameterization with Denoising Diffusion Models. In *International Conference on Machine Learning (ICML)*, 2026. URL https://icml.cc/virtual/2026/poster/61885.

Danijar Hafner. Benchmarking the Spectrum of Agent Capabilities. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=1W0z96MFEoH.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning Latent Dynamics for Planning from Pixels. In *International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2555–2565. PMLR, 2019. URL https://proceedings.mlr.press/v97/hafner19a.html.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to Control: Learning Behaviors by Latent Imagination. In *International Conference on Learning Representations (ICLR)*, 2020. URL https://openreview.net/forum?id=S1lOTC4tDS.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with Discrete World Models. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=0oabwyZbOu.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, 2025. URL https://www.nature.com/articles/s41586-025-08744-2.

Mikael Henaff, Roberta Raileanu, Minqi Jiang, and Tim Rocktäschel. Exploration via Elliptical Episodic Bonuses. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 37631–37646, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/f4f79698d48bdc1a6dec20583724182b-Abstract-Conference.html.

James Hooper and Alexander Shekhovtsov. Generalized and Optimal Straight-Through Estimators. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 300 of *Proceedings of Machine Learning Research*, pp. 3187–3195. PMLR, 2026. URL https://proceedings.mlr.press/v300/hooper26a.html.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical Reparameterization with Gumbel-Softmax. In *International Conference on Learning Representations (ICLR)*, 2017. URL https://openreview.net/forum?id=rkE3y85ee.

Aya Kayal, Eduardo Pignatelli, and Laura Toni. The impact of intrinsic rewards on exploration in Reinforcement Learning. *Neural Computing and Applications*, 37(21):16269–16303, 2025. URL https://link.springer.com/article/10.1007/s00521-025-11340-0.

Mehdi Keramati and Boris Gutkin. Homeostatic reinforcement learning for integrating reward collection and physiological stability. *eLife*, 3:e04811, 2014. URL https://elifesciences.org/articles/04811.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014. URL https://arxiv.org/abs/1312.6114.

Vint Lee, Pieter Abbeel, and Youngwoon Lee. DreamSmooth: Improving Model-based Reinforcement Learning via Reward Smoothing. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/6f7d90b1198fec96defd80b5ebd5bc81-Abstract-Conference.html.

Liyuan Liu, Chengyu Dong, Xiaodong Liu, Bin Yu, and Jianfeng Gao. Bridging Discrete and Backpropagation: Straight-Through and Beyond. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 12291–12311, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/28b5dfc51e5ae12d84fb7c6172a00df4-Abstract-Conference.html.

Haoyu Ma, Jialong Wu, Ningya Feng, Chenjun Xiao, Dong Li, Jianye Hao, Jianmin Wang, and Mingsheng Long. HarmonyDream: Task Harmonization Inside World Models. In *International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 33983–34007. PMLR, 2024. URL https://proceedings.mlr.press/v235/ma24o.html.

Marlos C. Machado, Marc G. Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the Arcade Learning Environment: Evaluation Protocols and Open Problems for General Agents. *Journal of Artificial Intelligence Research (JAIR)*, 61:523–562, 2018. URL https://doi.org/10.1613/jair.5699.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. In *International Conference on Learning Representations (ICLR)*, 2017. URL https://openreview.net/forum?id=S1jE5L5gl.

Fryderyk Mantiuk, Hanqi Zhou, and Charley M. Wu. From Curiosity to Competence: How World Models Interact with the Dynamics of Exploration. In *Proceedings of the Annual Conference of the Cognitive Science Society (CogSci)*, pp. 2398–2405, 2025. URL https://escholarship.org/content/qt0f04713z/qt0f04713z.pdf.

Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte Carlo Gradient Estimation in Machine Learning. *Journal of Machine Learning Research (JMLR)*, 21(132):1–62, 2020. URL https://jmlr.org/papers/v21/19-346.html.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven Exploration by Self-supervised Prediction. In *International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2778–2787. PMLR, 2017. URL https://proceedings.mlr.press/v70/pathak17a.html.

Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-Supervised Exploration via Disagreement. In *International Conference on Machine Learning (ICML)*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5062–5071. PMLR, 2019. URL https://proceedings.mlr.press/v97/pathak19a.html.

Max B. Paulus, Chris J. Maddison, and Andreas Krause. Rao-Blackwellizing the Straight-Through Gumbel-Softmax Gradient Estimator. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=Mk6PZtgAgfq.

Sai Rajeswar, Pietro Mazzaglia, Tim Verbelen, Alexandre Piché, Bart Dhoedt, Aaron Courville, and Alexandre Lacoste. Mastering the Unsupervised Reinforcement Learning Benchmark from Pixels. In *International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pp. 28598–28617. PMLR, 2023. URL https://proceedings.mlr.press/v202/rajeswar23a.html.

Nicholas Rhinehart, Jenny Wang, Glen Berseth, John Co-Reyes, Danijar Hafner, Chelsea Finn, and Sergey Levine. Information is Power: Intrinsic Control via Information Capture. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 10745–10758, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/59112692262234e3fad47fa8eabf03a4-Abstract.html.

Cansu Sancaktar, Christian Gumbsch, Andrii Zadaianchuk, Pavel Kolev, and Georg Martius. SENSEI: Semantic Exploration Guided by Foundation Models to Learn Versatile World Models. In *International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pp. 52745–52777. PMLR, 2025. URL https://proceedings.mlr.press/v267/sancaktar25a.html.

Aidan Scannell, Mohammadreza Nakhaei, Kalle Kujanpää, Yi Zhao, Kevin Sebastian Luck, Arno Solin, and Joni Pajarinen. Discrete Codebook World Models for Continuous Control. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/1dd85d064697ec1a258cd2b8755b8c6d-Abstract-Conference.html.

Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to Explore via Self-Supervised World Models. In *International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8583–8592. PMLR, 2020. URL https://proceedings.mlr.press/v119/sekar20a.html.

Rushi Shah, Mingyuan Yan, Michael Curtis Mozer, and Dianbo Liu. Improving Discrete Optimisation Via Decoupled Straight-Through Estimator. arXiv:2410.13331v2, 2026. URL https://arxiv.org/abs/2410.13331v2. Revised February 2026; first submitted 2024.

Alexander Shekhovtsov. Cold Analysis of Rao-Blackwellized Straight-Through Gumbel-Softmax Gradient Estimator. In *International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pp. 30931–30955. PMLR, 2023. URL https://proceedings.mlr.press/v202/shekhovtsov23a.html.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. DeepMind Control Suite. arXiv:1801.00690, 2018. URL https://arxiv.org/abs/1801.00690.

Callum Rhys Tilbury, Filippos Christianos, and Stefano V. Albrecht. Revisiting the Gumbel-Softmax in MADDPG. In *Proceedings of the Adaptive and Learning Agents Workshop (ALA) at AAMAS*, 2023. URL https://alaworkshop2023.github.io/papers/ALA2023_paper_23.pdf.

Astrid Vanneste, Simon Vanneste, Tom De Schepper, Siegfried Mercelis, Peter Hellinckx, and Kevin Mets. An in-depth analysis of discretization methods for communication learning using backpropagation with multi-agent reinforcement learning. *Neural Computing and Applications*, 38(10):404, 2026. URL https://link.springer.com/article/10.1007/s00521-025-11782-6.

Jose M. G. Vilar and Leonor Saiz. Propensity Straight-Through Gradients for Discrete Stochastic Systems. arXiv:2608.25631, 2026. URL https://arxiv.org/abs/2608.25631.

Leonardo Villalobos-Arias, Grant Forbes, Jianxun Wang, David L. Roberts, and Arnav Jhala. Minding Motivation: The Effect of Intrinsic Motivation on Agent Behaviors. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, 21(1): 347–356, 2025. URL https://ojs.aaai.org/index.php/AIIDE/article/view/36838.

Jacob C. Walker, Eszter Vértes, Yazhe Li, Gabriel Dulac-Arnold, Ankesh Anand, Theophane Weber, and Jessica B. Hamrick. Investigating the Role of Model-Based Learning in Exploration and Transfer. In *International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pp. 35368–35383. PMLR, 2023. URL https://proceedings.mlr.press/v202/walker23a.html.

Daniel Wang and Thang D. Bui. Beyond ReinMax: Low-Variance Gradient Estimators for Discrete Latent Variables. *Transactions on Machine Learning Research (TMLR)*, 2026. URL https://openreview.net/forum?id=crlvtnsyIT.

Mingqi Yuan, Roger Creus Castanyer, Bo Li, Xin Jin, Wenjun Zeng, and Glen Berseth. RLeXplore: Accelerating Research in Intrinsically-Motivated Reinforcement Learning. *Transactions on Machine Learning Research (TMLR)*, 2025. URL https://openreview.net/forum?id=B9BHjTN4z6.

A Posterior Gradient Interventions and Derivations

A.1 Surrogate Gradient Differences at a Single Node

Fig. A1 illustrates how surrogate probabilities and gradients differ for the same forward selection in a three-category example. We expand the surrogate gradients from Section 3.2 by category below.

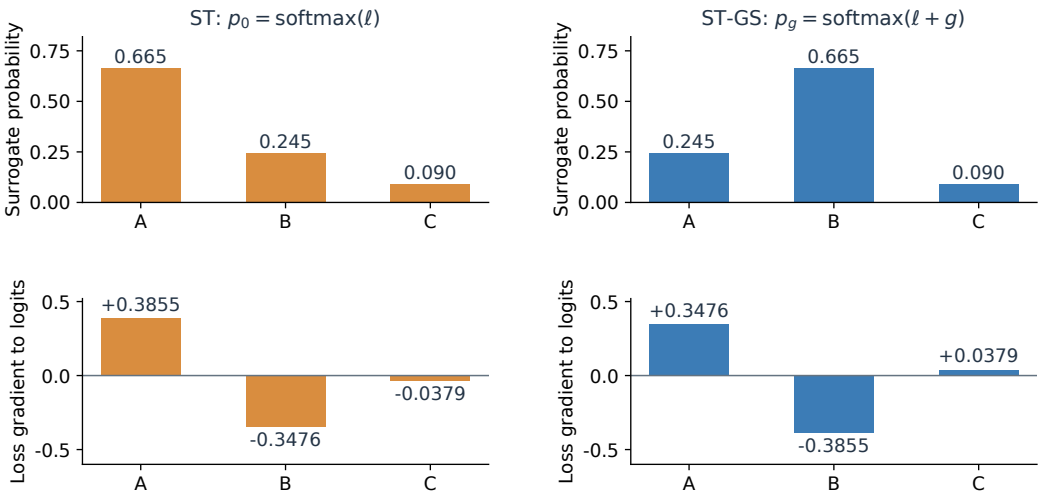

Figure A1: Different estimators can change gradient signs despite the same discrete selection. For one latent group with categories A, B, and C, the logits are $(2, 1, 0)$, the noise is $(-1, 1, 0)$, and the temperature is 1. Both estimators select B. The top row shows surrogate probability vectors; the bottom row shows loss gradients with respect to the logits for a fixed incoming gradient $(1, -1, 0)$. The gradient for category C changes sign.

For the current categorical group, let p be the surrogate probability vector and c_j the jth component of the downstream gradient c, namely the derivative of the loss with respect to that component of the one-hot state. Indices i, j, k refer to categories within the group, and $\mathbf{1}\{i = k\}$ equals 1 if the indices match and 0 otherwise. The softmax derivative gives

$$\begin{aligned} \frac{\partial p_i}{\partial \ell_k} &= p_i\big(\mathbf{1}\{i = k\} - p_k\big), \\ \big[J(p)^\top c\big]_k &= \sum_i c_i p_i\big(\mathbf{1}\{i = k\} - p_k\big) = p_k\left(c_k - \sum_j p_j c_j\right). \end{aligned} \tag{8}$$

ST uses $p = p_0$, and ST-GS uses $p = p_g$. The loss gradient with respect to logit k depends on its probability weight and the difference between c_k and the probability-weighted average $\sum_j p_j c_j$. Different probability vectors can therefore change gradient magnitudes and signs even for the same incoming c. This comparison holds the local inputs and downstream gradient fixed for one categorical group in Eq. (8). In a full RSSM, later sampling nodes also participate in backpropagation, so changing their estimators can change the c reaching the current node. Propagating local differences to model parameters additionally involves network Jacobians, loss paths acting directly on logits, and the optimizer.

A.2 Sampling Correspondence and Expected Gradients

Fix the logits and other context at one sampling node, change only its backward estimator, and hold the remaining backward rules fixed. Write the downstream gradient from Section 3.2 as $c(h)$. Both estimators share this function, which varies with the actual sample h. The ST-GS surrogate derivative and $c(h)$ use information from the same sample, and their dependence affects the expected

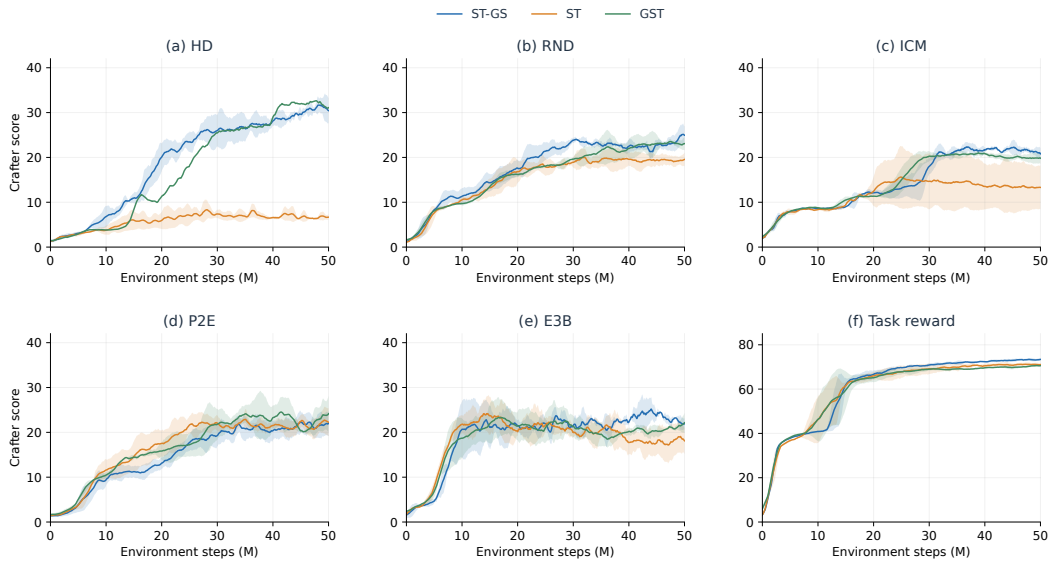

Figure B1: Learning under three posterior gradient estimators. Panels (a–f) show mean learning curves over 0–50M steps for HD, RND, ICM, P2E, E3B, and task-reward training. Shading indicates the sample standard deviation across smoothed training curves.

gradient. Let $\bar{J} = \mathbb{E}[J(p_g)]$ and $\bar{c} = \mathbb{E}[c(h)]$. Then

$$\mathbb{E}[\widehat{\nabla_\ell L}_{\text{ST-GS}}] - \mathbb{E}[\widehat{\nabla_\ell L}_{\text{ST}}] = (\bar{J} - J(p_0))^\top \bar{c} + \mathbb{E}\left[(J(p_g) - \bar{J})^\top (c(h) - \bar{c})\right]. \tag{9}$$

The first term in Eq. (9) comes from the difference in mean local Jacobians; the second captures the dependence between the local Jacobian and downstream gradient. If the backward pass uses separate noise g' with the same distribution and temperature, independent of the random sources producing h and $c(h)$, the second term vanishes under this fixed single-node setting. The expected gradient difference from ST is then given by the first term. Full recurrent-network training also includes downstream gradient changes caused by other nodes. Rao–Blackwellization instead averages auxiliary noise conditional on a fixed one-hot state, preserving the original estimator's expectation and bias while reducing variance (Paulus et al., 2021; Shekhovtsov, 2023).

The distinction between local and trajectory-level derivatives also appears in discrete stochastic simulation. Vilar & Saiz (2026) derive an exact one-step conditional-mean sensitivity for a class of continuous-time Markov chains, but show that composing these sensitivities need not recover the exact multistep derivative. Their result concerns a different stochastic system and does not establish gradient accuracy for an RSSM.

B Full Results and Additional Controls

B.1 Learning Curves

Fig. B1 shows the full learning curves for the five intrinsic-drive methods and the task-reward control.

B.2 Additional GST Comparisons in Atari

Table B1 extends the GST comparison to Atari. In Ms. Pac-Man, GST performs below ST and ST-GS for RND and between them for P2E. In Seaquest, GST is close to ST for RND and outperforms both alternatives for P2E. Adjusting the surrogate around the sampled category does not produce consistent gains across tasks. Its effect depends on learning under the particular task and drive.

Table B1: Mean native returns of final policies under three estimators in Atari.

Task	Method	ST	ST-GS	GST
Ms. Pac-Man	RND	2773.48	2199.53	2081.76
Ms. Pac-Man	P2E	1027.77	1426.29	1195.31
Seaquest	RND	647.19	604.77	639.61
Seaquest	P2E	3207.58	3410.70	4748.16

B.3 Additional ZGR Comparisons

We apply the zero-temperature limit of the Gumbel-Rao estimator, ZGR (Shekhovtsov, 2023), at the same observation-posterior node. Table B2 reports RND and task-reward results as an additional test of posterior gradient rules. ZGR does not outperform ST-GS in either configuration. Its bias–variance advantages in prior work do not directly translate into higher final scores here.

Table B2: Final Crafter scores for ZGR and other estimators.

Method	ST	ST-GS	GST	ZGR
RND	18.15	26.80	23.56	19.11
Task Reward	69.18	74.07	70.54	64.53

B.4 Other Tested Methods

Table B3 reports additional results for Surprise Minimizing Reinforcement Learning (SMiRL) (Berseth et al., 2021). All four estimators yield low Crafter achievement scores. These scores measure achievement completion rather than the maintenance of familiar states, so they do not establish that SMiRL is insensitive to posterior gradient estimation.

Table B3: Final Crafter scores for SMiRL.

Estimator	ST	ST-GS	GST	Independent backward noise
Score	4.23	6.43	4.35	3.83

B.5 Comparison with Gaussian Latent States

Latent-state design is a separate modeling choice. Discrete codebook world models have also been compared with continuous representations in state-based control (Scannell et al., 2025). To test whether continuous latent states offer a direct architectural alternative, we follow the Gaussian latent-state modeling approach of early Dreamer (Hafner et al., 2020) and use a 32-dimensional Gaussian stochastic variable in the current backbone. Reparameterized sampling and pathwise gradients avoid straight-through estimation for discrete latent states (Kingma & Welling, 2014). Table B4 reports final-policy evaluation after 50M training steps, using the same 256-episode protocol as the other Crafter results.

Table B4: Final Crafter scores with Gaussian stochastic latent states.

Method	Task Reward	HD	RND	P2E
Score	70.33	12.95	13.18	5.54

The Gaussian model achieves a high score with task rewards, but HD, RND, and P2E remain below their discrete ST-GS counterparts (Table 1). Avoiding discrete straight-through estimation does not directly improve the performance of intrinsically driven agents under this configuration and budget. Switching to Gaussian states also changes the latent distribution, stochastic representation dimension, Kullback–Leibler (KL) divergence term, and P2E prediction target. This comparison therefore

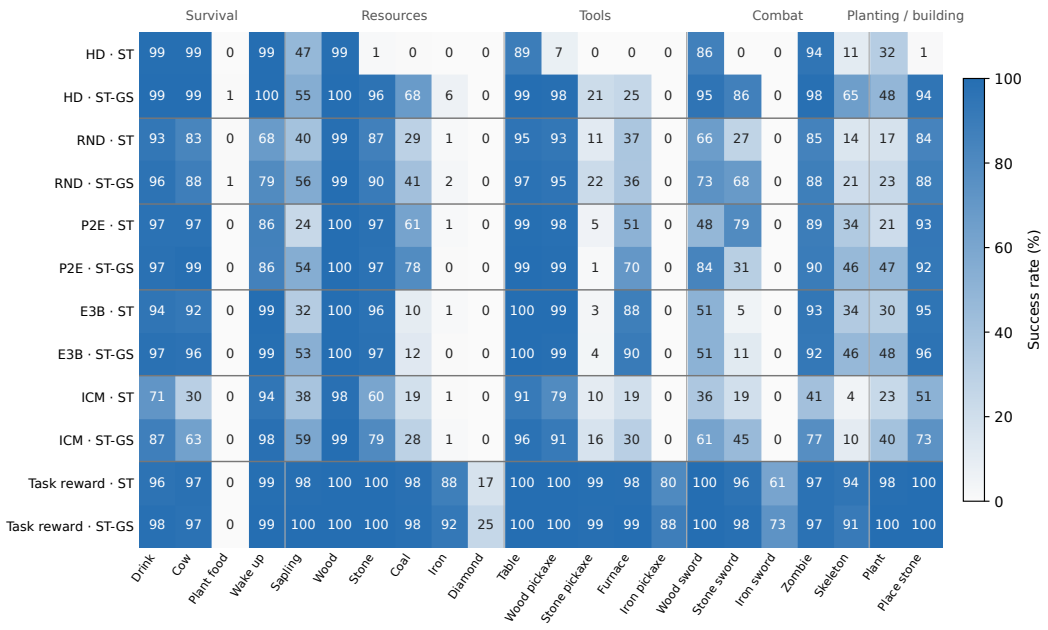

Figure B2: Success rates for all 22 Crafter achievements under ST and ST-GS. Each method has two rows, ST and ST-GS, reporting mean success rates (%).

tests an architectural alternative as a whole. It neither isolates the effect of straight-through estimation nor determines the performance of continuous models under other configurations or longer training.

B.6 Full Skill Results

Fig. B2 shows absolute success rates for all skills, and Table B5 reports changes from ST to ST-GS with variation across training runs. Together, they show the performance levels underlying skill differences and the consistency of those differences across runs.

Skill changes have different structures across methods. HD mainly improves in stone collection and tool crafting, whereas P2E shows large changes in both directions. P2E's stone- and wood-sword differences also vary substantially across training runs. Mean differences therefore describe aggregate skill changes without implying that an estimator necessarily suppresses a particular skill. The comparison distinguishes joint improvements from simultaneous gains and losses; it does not establish competition between skills or treat every nonzero difference as statistically significant.

B.7 Skill Contributions to the Aggregate Score

Crafter geometrically aggregates the success rates of 22 achievements (Hafner, 2022). Let p_j be the success rate of skill j on a 0–100 scale and S the official score. For a matched training pair, define skill j's contribution to the ST-GS versus ST difference as d_j:

$$\log(1+S) = \frac{1}{22}\sum_{j=1}^{22}\log(1+p_j),$$

$$d_j = \frac{\log(1+p_j^{\text{ST-GS}}) - \log(1+p_j^{\text{ST}})}{22}, \tag{10}$$

$$\log(1+S_{\text{ST-GS}}) - \log(1+S_{\text{ST}}) = \sum_{j=1}^{22} d_j.$$

Table B5: Full skill differences for five intrinsic drives and task-reward training. Entries are mean ST-GS−ST success-rate differences ± sample standard deviation across training runs, in percentage points.

Skill	HD	RND	P2E	E3B	ICM	Task Reward
Collect coal	+67.45 ± 6.07	+11.72 ± 10.27	+17.06 ± 2.22	+2.54 ± 6.91	+9.77 ± 3.87	+0.52 ± 1.19
Collect diamond	+0.00 ± 0.00	+0.00 ± 0.00	+0.00 ± 0.00	+0.00 ± 0.00	+0.00 ± 0.00	+7.55 ± 11.93
Drink water	+0.39 ± 0.39	+2.73 ± 1.79	+0.26 ± 2.74	+2.93 ± 0.28	+15.62 ± 4.97	+2.21 ± 0.45
Collect iron	+5.86 ± 4.74	+0.91 ± 2.22	−0.78 ± 1.41	−0.98 ± 1.38	+0.78 ± 0.55	+4.56 ± 5.67
Collect sapling	+7.94 ± 15.28	+16.02 ± 3.77	+30.47 ± 24.08	+20.70 ± 35.36	+20.90 ± 1.38	+1.56 ± 1.41
Collect stone	+94.40 ± 1.80	+3.52 ± 1.03	−0.52 ± 0.60	+0.39 ± 0.00	+19.34 ± 1.93	+0.13 ± 0.23
Collect wood	+1.43 ± 0.98	+0.91 ± 0.90	+0.26 ± 0.45	+0.00 ± 0.00	+1.56 ± 1.66	+0.00 ± 0.00
Defeat skeleton	+54.56 ± 4.63	+6.38 ± 9.68	+12.37 ± 12.32	+12.70 ± 6.91	+5.86 ± 6.08	−3.52 ± 1.03
Defeat zombie	+3.78 ± 2.48	+2.73 ± 5.17	+0.78 ± 1.56	−0.59 ± 0.83	+36.13 ± 59.94	+0.00 ± 1.03
Eat cow	+0.39 ± 1.03	+4.95 ± 7.86	+1.95 ± 2.95	+4.10 ± 0.83	+33.20 ± 33.70	−0.65 ± 1.76
Eat plant	+0.39 ± 0.39	+0.26 ± 0.60	+0.00 ± 0.39	+0.20 ± 0.28	+0.00 ± 0.00	+0.00 ± 0.00
Make iron pickaxe	+0.00 ± 0.00	+0.00 ± 0.00	+0.00 ± 0.00	+0.00 ± 0.00	+0.00 ± 0.00	+8.20 ± 8.07
Make iron sword	+0.13 ± 0.23	+0.00 ± 0.00	+0.00 ± 0.00	+0.00 ± 0.00	+0.00 ± 0.00	+12.24 ± 8.79
Make stone pickaxe	+20.96 ± 11.76	+11.07 ± 9.81	−3.91 ± 6.20	+0.39 ± 2.76	+5.86 ± 0.55	−0.26 ± 1.80
Make stone sword	+86.46 ± 3.16	+41.15 ± 35.69	−48.44 ± 48.71	+6.64 ± 3.31	+25.98 ± 23.48	+2.21 ± 0.60
Make wood pickaxe	+91.41 ± 4.75	+1.69 ± 2.77	+0.52 ± 1.37	−0.78 ± 0.55	+12.11 ± 9.94	+0.00 ± 0.00
Make wood sword	+8.46 ± 3.18	+7.29 ± 13.92	+35.94 ± 44.48	+0.20 ± 0.83	+25.20 ± 31.76	−0.26 ± 0.45
Place furnace	+24.74 ± 2.93	−0.26 ± 7.62	+19.14 ± 37.21	+2.73 ± 4.97	+11.33 ± 12.15	+0.78 ± 0.68
Place plant	+16.28 ± 13.09	+5.99 ± 5.69	+26.30 ± 23.83	+17.38 ± 31.21	+16.41 ± 3.87	+1.56 ± 1.41
Place stone	+93.36 ± 1.70	+3.26 ± 2.66	−0.26 ± 2.74	+1.37 ± 0.28	+22.07 ± 3.04	+0.13 ± 0.23
Place table	+10.42 ± 5.25	+1.69 ± 2.26	+0.26 ± 1.85	−0.20 ± 0.28	+5.47 ± 5.52	+0.00 ± 0.00
Wake up	+0.39 ± 0.78	+11.33 ± 9.97	−0.39 ± 7.68	−0.39 ± 0.55	+3.91 ± 3.31	+0.26 ± 0.81

Table B6: Skill contributions to final score differences. Positive contributions, negative contributions, and net changes are computed on the log-score scale and then averaged across matched training pairs.

Method	Positive contribution	Negative contribution	Net change
HD	+1.3390	−0.0029	+1.3362
RND	+0.2224	−0.0263	+0.1961
P2E	+0.1940	−0.1589	+0.0351
E3B	+0.1336	−0.0405	+0.0931
ICM	+0.4655	−0.0018	+0.4637
Task Reward	+0.0424	−0.0063	+0.0361

Summing positive and negative values of d_j separately gives the positive and negative contributions in Table B6; their sum is the net change. This decomposition describes how individual skills contribute to the aggregate score difference.

P2E's positive and negative contributions largely cancel, whereas HD and ICM are dominated by positive contributions. P2E's similar aggregate scores thus reflect offsetting skill gains and losses, not an absence of substantial skill changes. This decomposition explains how skill profiles contribute to the aggregate score.

C Local Return-Prediction Tests

RND uses a reward head to predict image-based rewards during imagination, whereas P2E computes disagreement directly on latent states. These two ways of using rewards let us examine how posterior gradient estimation affects local return predictions. These tests do not cover HD or GST and do not provide a complete explanation of performance differences across methods.

C.1 Action-Return Rankings under Different Prediction Conditions

Fig. C1 compares model-predicted rankings of candidate actions with rankings from real-environment continuations. Spearman correlation measures the accuracy of their relative ordering.

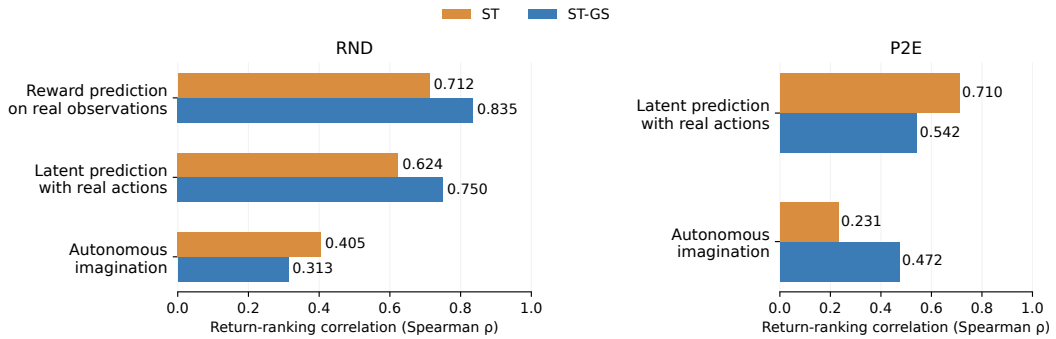

Figure C1: Action-return rankings under different prediction conditions for RND and P2E in Crafter. Bars show mean Spearman correlations over the same histories. For RND, the first row infers posterior states from real observations, predicts rewards, and accumulates them along real trajectories. The second predicts latent states and rewards along recorded real actions, without future observations, using real stopping conditions. The third imagines subsequent states and actions. P2E computes latent-state disagreement directly and uses the latter two conditions. All returns have a 15-step horizon without a terminal value estimate.

We use 14 histories with crafting opportunities. The agent is near a workbench, has enough resources to craft the target tool, and does not yet hold a stone sword. Selection uses only real environmental states, not model predictions or policy outputs. P2E ST-GS and ST supply 6 and 5 histories, respectively; RND ST-GS and ST supply 2 and 1. The predefined main and held-out groups contain 7 histories each, with no RND ST history in the held-out group. All four models are trained to 45M steps and process the same complete observation–action histories from episode reset. At each history endpoint, we clone the environment and evaluate 17 candidate first actions.

After each candidate first action, subsequent actions are sampled from the frozen policy for at most 15 transitions. The reward on the death transition is included, with no further stepping or reset. We repeat each continuation four times using fixed, paired latent-state and action-sampling noise; these two noise sources are independent. Environment continuations share an initial random-number state, although later random-number calls can depend on actions.

Let r_k be the intrinsic reward at transition k, $\mathrm{done}_k \in \{0, 1\}$ indicate true termination, and γ be the discount factor. The finite-horizon return is

$$R_H(a) = \sum_{k=0}^{H-1} \left[\prod_{j=0}^{k-1} \gamma(1 - \mathrm{done}_j) \right] r_k. \tag{11}$$

The empty product is 1, so the terminal action's reward is retained. P2E computes disagreement among prediction heads from the current posterior state and action. RND computes prediction error on the real successor image using its frozen reward module. Neither return uses achievement logs. The world model, reward modules, and normalization statistics remain fixed throughout the continuations.

Real-environment continuations provide reference returns. Two model-prediction paths start from the same history and candidate first action. One imagines subsequent actions and states; the other receives the recorded real actions and stopping conditions and predicts latent states without future images. A further comparison replaces only the stopping condition to distinguish this change from replacing both actions and stopping conditions. All paths use the same history-encoding settings and paired inference noise. For each history, returns are averaged over four continuations before computing Spearman correlation across the 17 candidate first actions. Correlations are then averaged equally across the 14 histories, using average ranks for tied returns.

For RND, return predictions using real observations have ranking correlations of 0.712 and 0.835 under ST and ST-GS, respectively. With real actions supplied, they are 0.624 and 0.750; under autonomous imagination, they are 0.405 and 0.313 (Fig. C1). For P2E, the relative ranking accuracy of ST and ST-GS also reverses between prediction along recorded actions and autonomous imagina-

Table C1: Action-return rankings after updates from the same initialization and data in Crafter. Initialization identifies the original model. Values are mean Spearman correlations on fixed histories. The difference is ST-GS minus ST after 50 updates.

Method	Initialization	Before updates	ST, 50 updates	ST-GS, 50 updates	Difference
RND	ST	0.623	0.663	0.601	−0.061
RND	ST-GS	0.748	0.706	0.705	−0.001
P2E	ST	0.717	0.608	0.641	+0.033
P2E	ST-GS	0.553	0.459	0.447	−0.012

tion. On these histories, the prediction advantage along real action trajectories does not carry over to action evaluation in autonomous imagination. The conditions change access to future observations, actions, or stopping conditions, so they do not isolate which component causes the ranking changes.

C.2 Short Updates from the Same Initialization and Data

Table C1 examines how these rankings change under local learning updates by comparing ST and ST-GS from the same model initialization and training data. For each of RND and P2E, we start from both ST and ST-GS models trained to 45M steps. Each checkpoint initializes two branches that differ only in the posterior gradient estimator. The branches share initial parameters, main-optimizer state, data order, and random numbers. Their initial predictions and first forward losses match. Training uses the same saved trajectories, and evaluation uses 14 crafting-opportunity histories from worlds separate from the training worlds.

Updates retain the main losses and freeze the independent intrinsic-reward modules. They use two continuous 64-step data streams, resetting recurrent state at actual episode starts. After 0, 1, 10, and 50 updates, we re-encode evaluation histories and predict 15-step returns along recorded real actions and stopping conditions, without future images or a terminal value estimate. Reference returns remain those of the pre-update model's real continuations, keeping the evaluation criterion fixed across updates. Although P2E's disagreement networks are frozen, their input representation changes with the RSSM, so the reward mapping may change. Keeping the reference returns fixed lets us assess how the updates change agreement with the original action rankings. We average four continuations per history, compute Spearman correlation over 17 first actions, and then average equally across histories.

Starting from the RND ST checkpoint, ST updates yield a higher ranking correlation; from the RND ST-GS checkpoint, the two updates are similar. For P2E, which estimator yields higher ranking accuracy depends on the starting checkpoint, with no consistent ST-GS ranking advantage. These local updates do not establish a consistent link between short-horizon return rankings and long-term performance advantages. Since the test covers only mature models, selected histories, and 50 updates on fixed data, it does not rule out a role for improved predictions over longer training.

D Implementation and Statistics

D.1 Model Configuration and Gradient Paths

RSSM categorical probabilities are mixed with a uniform distribution using a unimix coefficient of 0.01, then converted to log probabilities for sampling (Hafner et al., 2025). The symbol $\ell_t^{(m)}$ in Section 3.1 denotes these logits for group m; ℓ_t in Fig. 1a denotes all groups. The policy also uses unimix 0.01. These settings, continuation prediction, the action sampler, the imagination prior, reward timing, and gradient paths are fixed within each method's paired experiments. Implementation checks compare posterior logits, discrete samples, and forward losses under identical parameters, inputs, recurrent states, and random numbers, and check backward gradients and the first update.

Prediction heads and gradient paths. Reward and value heads output parameters of their predictive distributions and use the backbone's distributional losses. Loss forms and weights are fixed within each method's paired comparisons. Reward prediction retains gradients to its inputs (`reward_grad=True`). Value learning on replay states is enabled (`repval_loss=True`) and

also retains input gradients (`repval_grad=True`), with loss weight 0.3. Both losses can update the encoder and RSSM through their prediction heads, while gradients are stopped at the supervision targets. Policy updates use action log probabilities and stop-gradient advantage estimates. Policy and value losses on imagined states do not propagate gradients to input features (`ac_grads=False`).

Value learning on replay states. Following DreamerV3, imagined sequences and posterior states from real replay share a value head. For replay states, training rewards and imagined return estimates define a λ-return target (Hafner et al., 2025). Let b_t be the estimated return from imagination starting at replay state t, G_t the replay value target, and T the end of the sampled replay segment. Within an episode,

$$G_t = r_{t+1} + \gamma\big[(1-\lambda)b_{t+1} + \lambda G_{t+1}\big], \qquad G_T = b_T. \tag{12}$$

Here, r_{t+1} is the training reward recorded on the real transition or recomputed by the current intrinsic-reward module, γ is the discount factor, and λ controls the mixture of one-step bootstrapping and recursive returns. Table D1 gives their values. We do not bootstrap at terminal states. At a nonterminal episode truncation, recursion stops across the episode boundary and uses the imagined return estimate for that state. The value head fits stopped, normalized return targets and retains the backbone's slow-value regularizer, whose target is also stopped. The slow-value network is a moving-average copy of the value head.

D.2 Intrinsic and Task Rewards

RND. An image-based reward module uses the prediction error of a trainable predictor against a fixed random target network (Burda et al., 2019b). Normalized errors supervise Dreamer's reward head on replay data, and policy learning in imagination uses its predictions. A similar adaptation appears in Rajeswar et al. (2023).

P2E. The original Plan2Explore implementation uses Gaussian stochastic latent states, five prediction heads, and the next image encoding as the prediction target (Sekar et al., 2020). We use the authors' later disagreement module in DreamerV2; the corresponding migration is described in their code repository. This module contains ten heads that predict the next stochastic latent state from the current state and action. The disagreement reward is the standard deviation across heads for each feature, averaged over features.

Gradients are stopped at the inputs and targets used to train the auxiliary prediction heads. Disagreement rewards define imagined returns and value targets on replay states; the ordinary reward head fits zero task rewards. We first perform the main update using the current disagreement model, then update the auxiliary prediction heads. This ordering is adapted to our DreamerV3 update procedure and is identical in the ST and ST-GS conditions.

E3B and ICM. E3B maintains coverage statistics over image representations and resets them each episode (Henaff et al., 2022). ICM uses consecutive images and actions with separate inverse and forward prediction modules (Pathak et al., 2017). Both retain the key structure of their reward

Table D1: Crafter model and training configuration.

Setting	Value
Environment	Crafter 1.8.3
Backbone	DreamerV3, approximately 25M parameters (size25m)
Stochastic latent state	32 groups $\times$ 24 categories
Deterministic state	3072 dimensions
Imagination horizon	15
Discount factor	$\gamma = 1 - 1/333 \approx 0.997$
Return mixing coefficient	$\lambda = 0.95$
Precision	bfloat16 (BF16)
Learning rate	8×10^{-5}
Training ratio (configuration value)	32
Batch (sequences $\times$ length)	16×64
Replay capacity	1M
Parallel environments	8
Run budget / analysis endpoint	50.1M / 50M environment steps

definitions. In the Dreamer adaptation, their image-based rewards supervise the reward head, whose predictions are used in imagination. Neither method receives an additional physiological vector.

Homeostatic drive (HD). In Crafter, HD defines homeostatic drive from deficits in food, water, energy, and health relative to target levels, and rewards its reduction (Keramati & Gutkin, 2014). Let $v_{t,i}$ denote each physiological variable, all with maximum value 9. The normalized state $x_{t,i}$, deficit $\delta_{t,i}$, homeostatic drive D_t, and reward for the transition from t to $t+1$ are

$$\begin{aligned} x_{t,i} &= \mathrm{clip}(v_{t,i}/9, 0, 1), \qquad \delta_{t,i} = \max(1 - x_{t,i}, 0), \\ D_t &= \sum_{i=1}^{4} \delta_{t,i}^2, \qquad\qquad r_{t+1}^{\mathrm{HD}} = D_t - D_{t+1}. \end{aligned} \tag{13}$$

All weights and targets equal 1. Squared deficits are summed without an exponential moving average or dead zone. The reward scale is 1, without additional clipping, shifting, or environment rewards. Terminal rewards use the actual physiological state; terminal drive is not set to zero, and no extra death penalty is added. Reset initializes D_0 and returns zero reward without a cross-episode difference. Normalization clipping is used only for reward computation and does not change the environment state or rendering. The agent observes a 64×64 RGB image with the original status bar, without an additional physiological vector. The world model predicts this scalar reward for policy learning in imagination, without a dedicated physiological prediction head or policy.

Achievement rewards. Crafter's original reward includes achievement rewards and health-change shaping (Hafner, 2022). Our achievement-only control gives a reward of 1 for the first completion of each achievement in an episode. It directly matches the scored skills and excludes health-change shaping.

D.3 Cross-task Configurations

Atari and Walker use DreamerV3 configurations adapted to their image inputs and action spaces. Within each task and method, ST/ST-GS pairs keep all training and environment settings fixed except the posterior gradient estimator.

Atari uses the full action set, sticky actions with probability 0.25 (Machado et al., 2018), 0–30 random no-op (NOOP) actions at reset, and an episode limit of 108000 nominal frames. Life loss does not independently terminate an episode. Walker uses image observations only. Native rewards in both domains are used only for evaluation, not intrinsic-drive training. Atari steps are converted to nominal frames using action repeat 4. The count includes resets and is not exactly the number of frames executed by ALE. Table 2 and the cross-task tables in the appendix all use final-policy evaluation without task-reward adaptation.

Table D2: Configurations across tasks.

Setting	Crafter	Both Atari tasks	Walker Walk
Image input	64×64 RGB	96×96 grayscale	64×64 RGB
Model size	size25m	size12m	size12m
RSSM deterministic / stochastic dimensions	3072 / 32 groups × 24 classes	2048 / 32 groups × 16 classes	2048 / 32 groups × 16 classes
Imagination horizon / precision	15 / BF16	15 / BF16	15 / BF16
Batch	16×64	16×64	16×64
Learning rate	8e−5	4e−5	4e−5
Parallel environments	8	16	16
Replay capacity	1M	200k	200k
Training ratio (configuration value)	32	32	256
Action repeat	1	4	1
Analysis endpoint	50M	50M internal steps	5M internal steps

Reward-generation modules retain their cross-domain configurations. RND's image predictor is not reduced with the backbone size. P2E uses ten prediction networks with four layers of 400 units each. The number of categorical latent classes changes with backbone size, and the P2E target dimension changes accordingly.

D.4 Final Evaluation and Statistical Procedures

Final-policy evaluation uses the same 256 environment-initialization seeds within each task, with statistics computed over complete episodes. Crafter and Atari use 50M checkpoints, and Walker uses 5M checkpoints. Model parameters remain frozen, actions are sampled from the policy, and policy memory and episodic statistics reset each episode. Time-truncated episodes are included.

Crafter episodes end at death or a 10000-step limit. For each model, achievement success rates and the Crafter score are first computed over 256 episodes, then averaged equally across independent training runs. Atari and Walker native returns are aggregated in the same order, using the termination and time limits in Appendix D.3. All final-performance and skill tables follow this protocol. A Crafter achievement counts as successful if completed at least once during an episode. Learning curves use scores from the latest 100 completed training episodes, aligned by environment steps. Fig. B1 applies a centered moving average to each run, using a 1M-step window for ST and ST-GS and a 2M-step window for GST, before computing the mean and sample standard deviation across runs. At the boundaries, smoothing uses the available portion of the window. Curves with one training run have no across-run variability band. Smoothing affects only visualization, not final evaluation. Local prediction tests use the frozen-model, short-horizon protocols in Appendix C.

Scores, success rates, and paired differences are computed within each training run before equal-weight aggregation, preventing unequal episode counts from changing run weights. Unless otherwise stated, shaded regions and "$\pm$" denote sample standard deviations across training runs. Walker in Table 2 is an exception: it reports the standard deviation of returns over 256 evaluation worlds for a fixed model, using variance denominator 255 without division by the square root of the episode count. This measures return dispersion, not the standard error of the mean or training stability. Evaluation intervals for fixed checkpoints likewise capture only evaluation-sampling uncertainty (Agarwal et al., 2021).