PROWL-2: FIDELITY-GATED EXPLORATION FOR MULTI-AGENT WORLD-MODEL DUAL CURRICULA

Anonymous authors
Paper under double-blind review

ABSTRACT

World models promise sample-efficient multi-agent reinforcement learning by letting policies learn from imagined interaction. Imagination, however, introduces a fundamental ambiguity: a trajectory with high learning signal may expose a genuine weakness of the task-solving policy, or may simply reflect inaccurate world-model predictions. A curriculum that prioritises such experience without accounting for model fidelity can therefore reinforce world-model errors rather than improve the agents. We introduce PROWL-2, a fidelity-gated dual-curriculum framework that maintains one curriculum for training the task agents and a second for discovering and repairing world-model failures. PROWL-2 scores imagined experience along two axes—learning potential and world-model fidelity measured against the real environment—and routes it accordingly: reliable, informative rollouts are promoted to the agent curriculum, while unreliable ones are sent for world-model repair. A dedicated developer policy actively searches for further model failures, and repaired regions are re-evaluated before re-entering agent training, closing the loop between world-model improvement and policy learning. On cooperative tasks in StarCraft Multi-Agent Challenge v2 (SMACv2) and the Multi-Agent Quadruped Environment, spanning discrete combat and continuous robotic collaboration, PROWL-2 matches or outperforms strong model-based baselines and its own repair-only and ungated ablations in every setting at matched task-policy interaction, with the largest gains on the hardest tasks. Our results show that learning inside world models requires deciding not only which imagined experience is useful but which is trustworthy and that making this distinction explicit turns world-model failures into a source of policy improvement rather than a cause of policy error.

1 INTRODUCTION

Multi-agent reinforcement learning (MARL) has made substantial progress in cooperative decision making, driven largely by model-free methods (Nguyen et al., 2020; Lowe et al., 2020; Foerster et al., 2018; Rashid et al., 2018; Yu et al., 2022). Coordination, however, is expensive to learn: partial observability forces each agent to act on incomplete information, and inter-agent dependencies make the joint state-action space grow rapidly with the number of agents, so model-free algorithms typically require extensive environment interaction before coordinated behavior emerges (Ning & Xie, 2024; Huh & Mohapatra, 2024). The difficulty compounds when agents must stay robust across varied task instances, as in the StarCraft Multi-Agent Challenge v2 (SMACv2) (Samvelyan et al., 2019; Ellis et al., 2023) and the Multi-Agent Quadruped Environment (MQE) (Xiong et al., 2024). World models address the interaction cost by learning the environment's dynamics and letting policies train on imagined trajectories rather than real ones (Ha & Schmidhuber, 2018; Hafner et al., 2020; Robine et al., 2023; Zhang et al., 2023; Micheli et al., 2023; Hafner et al., 2024; Alonso et al., 2024; Hafner et al., 2025), and a growing family of multi-agent world models has brought these sample-efficiency gains to cooperative MARL (Egorov & Shpilman, 2022; Zhang et al., 2024; 2025; Toledo & Prorok, 2024; Zhao et al., 2026). Once trajectories can be generated on demand, however, the question becomes which of them are worth learning from.

Curriculum methods select experience by learning potential, typically some form of value or temporal-difference (TD) error, on the premise that experience the agent predicts poorly is experience it has something to learn from (Schaul et al., 2016; Jiang et al., 2021; Güzel et al., 2025).

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.

Inside a world model this premise breaks, because a high-error trajectory is ambiguous: it may expose a real weakness of the current policy, or it may be a low-fidelity rollout whose predicted dynamics the agent cannot value correctly because they never occur in the environment; compounding world-model error is a known failure mode of learning in imagination (Janner et al., 2019; Buckman et al., 2018). Recent methods address the two sides of this problem separately, and both for a single agent: Imagined Autocurricula (IMAC) prioritizes imagined experience by agent-side learning potential but assumes a fixed world model during policy learning (Güzel et al., 2025), while PROWL-1 (Prioritized Regret-driven Optimization for World-model Learning; Güzel et al., 2026) actively discovers and repairs world-model failures but maintains no curriculum over what the policy trains on. Neither couples the two, although they depend on each other: world-model errors matter most where the agent trains, and training is reliable only where the world model is accurate.

We introduce PROWL-2, a fidelity-gated dual-curriculum framework for cooperative multi-agent learning that couples both learning problems into one continual process. The agent curriculum prioritizes where the policy practices in imagination, whereas the developer explores the real environment for where the world model fails. A fidelity gate then decides which of that exploration and the agent's own practice may feed into policy training. Unlike their single-agent counterparts, both curricula operate on the joint problem: start states index joint action–observation histories, the developer acts for all agents jointly, and fidelity is measured on per-agent, coordination-relevant quantities (Section 4.2). Our key contributions are as follows:

- We introduce PROWL-2, a framework that trains the task-solving agents and the world model with two coupled curricula: one chooses where the agents train in imagination, and the other repairs the world model on failures found by a developer policy as well as the agents' own curriculum.
- We propose a fidelity gate that separates informative imagined experience from world-model errors: it checks the most promising imagined experience against real trajectories, withholds what the world model fails to predict, sends it for repair, and returns it to training once fixed.
- Across SMACv2 and MQE, using an actor–critic and a planning backbone, we find that the curricula are not additive: repair alone yields only modest improvements and the ungated curriculum falls below the backbone's performance in all nine SMACv2 scenarios, whereas adding the fidelity gate turns the same curriculum into the largest single gain, reaching two to four times the backbone's success rate on the hardest MQE tasks.

2 Related Work

Multi-agent model-based reinforcement learning. Cooperative MARL is commonly studied under centralized training with decentralized execution (Oliehoek & Amato, 2016; Sunehag et al., 2017), where model-free methods such as QMIX and MAPPO provide strong baselines (Rashid et al., 2018; Yu et al., 2022) at the cost of substantial environment interaction. Model-based MARL reduces this cost by learning approximate dynamics and training policies on generated experience. MAMBA extends Dreamer-style (Hafner et al., 2020) latent models with cross-agent attention (Egorov & Shpilman, 2022); CoDreamer adds a graph communication module (Toledo & Prorok, 2024); MARIE pairs decentralized local dynamics with centralized representation aggregation in a Transformer world model (Micheli et al., 2023; Zhang et al., 2024); DIMA models the joint transition through a diffusion-inspired denoising process (Zhang et al., 2025); and SeqWM decomposes joint dynamics into sequential agent-wise models for multi-robot cooperation (Zhao et al., 2026). Model-based MARL has also been surveyed (Wang et al., 2022), and MAZero brings MuZero-style planning to the multi-agent setting (Liu et al., 2024). These methods address how multi-agent dynamics should be represented. PROWL-2 addresses which of the experience an imperfect, improving world model generates should train the policy; it changes the experience distribution and leaves the world model and the MARL algorithm unchanged.

Curriculum learning and adaptive experience. Curriculum learning improves sample efficiency by controlling what the learner experiences rather than only how it is optimized (Bengio et al., 2009). Prioritized level replay (PLR) preferentially replays levels with high learning potential (Schaul et al., 2016; Jiang et al., 2021), and PAIRED and ACCEL extend this to unsupervised environment design

(UED), generating challenges near the frontier of an agent's ability (Dennis et al., 2020; Parker-Holder et al., 2022), and MAESTRO extends it to multi-agent settings (Samvelyan et al., 2023). These methods select or generate levels that are then executed in the true environment, so the learning signal they prioritize is never an artifact of the simulator producing it, although the learning-potential proxies themselves are imperfect (Jiang et al., 2022; Rutherford et al., 2024). Imagination removes that guarantee. IMAC applies PLR-style prioritization to imagined experience but holds the world model fixed during policy learning (Güzel et al., 2025); PROWL-1 instead uses an adversarial developer policy to discover learnable model failures for continual repair, but maintains no curriculum over what the task policy trains on (Güzel et al., 2026). Fidelity in PROWL-2 is always realized prediction error against a recorded continuation, which distinguishes it from ensemble-disagreement estimates of uncertainty (Chua et al., 2018; Kurutach et al., 2018; Pathak et al., 2019); disagreement may still serve as a search heuristic for the developer, but never as the criterion that admits experience to policy training. Selective trust in a learned model has been studied in single-agent model-based reinforcement learning through short or adaptive rollouts (Pan et al., 2020; Janner et al., 2019), uncertainty-weighted targets (Buckman et al., 2018; Feinberg et al., 2018), selective planning (Abbas et al., 2020), and uncertainty penalties in offline MBRL (Yu et al., 2020; Kidambi et al., 2020). These methods trust the model less where it seems unreliable. PROWL-2 instead measures reliability against real trajectories, withholds what fails from training, and repairs the model there.

3 PRELIMINARIES

Cooperative MARL and world models. We consider a fully cooperative decentralized partially observable Markov decision process (Dec-POMDP) (Oliehoek & Amato, 2016) $\mathcal{M} = (\mathcal{N}, \mathcal{S}, \mathcal{A}, P, R, \Omega, O, \gamma)$ with agents $\mathcal{N} = \{1, \dots, n\}$, states $s_t \in \mathcal{S}$, joint action space $\mathcal{A} = \prod_i \mathcal{A}^i$, transition distribution $P(s_{t+1} \mid s_t, \mathbf{a}_t)$, shared team reward $r_t = R(s_t, \mathbf{a}_t)$, observation spaces $\Omega = \prod_i \Omega^i$ with observation functions $O = \{O^i\}_{i=1}^n$, and discount factor $\gamma \in [0, 1)$. Each agent receives a local observation $o_t^i = O^i(s_t)$ and acts through a decentralized policy $a_t^i \sim \pi^i(\cdot \mid h_t^i)$ conditioned on its action–observation history, with the joint policy $\pi = \{\pi^i\}_{i=1}^n$ maximizing $J(\pi) = \mathbb{E}_{\pi,P}[\sum_t \gamma^t R(s_t, \mathbf{a}_t)]$. Model-based MARL adds a learned world model $M_\theta(h_t, \mathbf{a}_t) \to (\hat{\mathbf{o}}_{t+1}, \hat{r}_{t+1}, \hat{d}_{t+1})$, where $h_t = \{h_t^i\}_{i=1}^n$ is the joint history and $\hat{d}_{t+1}$ denotes continuation; applied autoregressively, it generates imagined rollouts of horizon T on which an actor–critic policy with critic V_ψ is optimized without environment interaction (Zhang et al., 2024).

Prioritizing imagined experience. IMAC ports PLR's prioritized replay (Jiang et al., 2021) into a learned world model (Güzel et al., 2025): a start state $\ell_i = (e_i, t_i)$ indexes timestep t_i of recorded episode e_i, from which the current world model and policy generate an imagined rollout. We write $\mathcal{B}_G$ for the buffer of such keys. Letting $\hat{\delta}_t = \hat{r}_t + \gamma \hat{d}_t V_\psi(\hat{z}_{t+1}) - V_\psi(\hat{z}_t)$ be the imagined temporal-difference residual, with $\hat{z}_t$ the world model's latent state at imagined step t, the learning potential of ℓ_i is its positive value loss over the rollout,

$$s_G(\ell_i) = \frac{1}{T} \sum_{t=0}^{T-1} \sum_{k=t}^{T-1} (\gamma\lambda)^{k-t} \, [\hat{\delta}_k]_+, \qquad [x]_+ = \max(0, x), \tag{1}$$

where $\lambda \in [0, 1]$ is the generalized-advantage parameter. The score emphasizes start states where imagined returns exceed what the critic currently predicts; the residuals are recovered from the generalized-advantage estimates the actor–critic update already computes (Schulman et al., 2015). Whether such a rollout can be trusted, however, cannot be determined from inside imagination. Section 4 grounds that decision in the environment and couples it to active world-model repair.

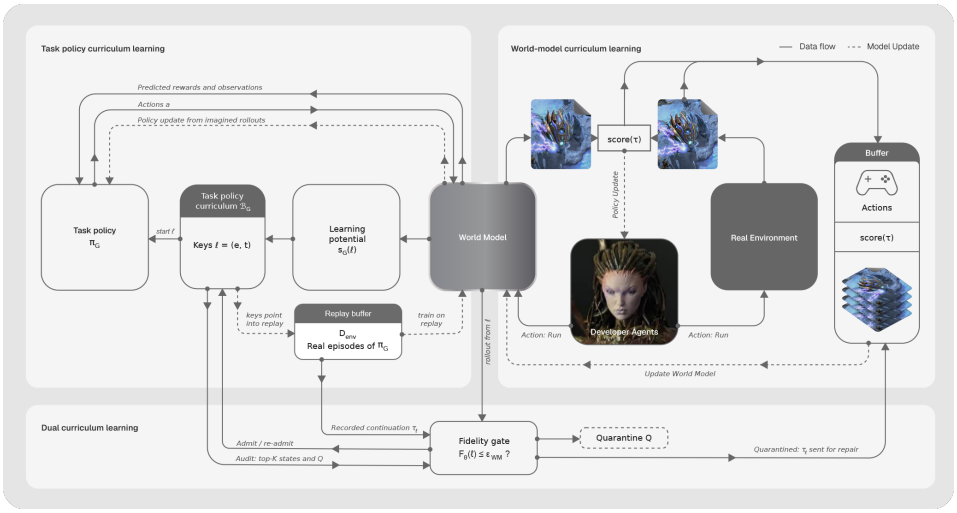


Figure 1: **Overview of PROWL-2**, with the three parts described in Section 4: task policy curriculum learning (top left), world-model curriculum learning (top right) and dual curriculum learning (bottom).

4 PROWL-2: Fidelity-Gated Dual Curricula

PROWL-2 involves two policies and a world model. The *task policy* π_G is the multi-agent policy we want to train; it learns inside the world model M_θ from imagined rollouts. The *developer policy* π_D never acts for the task; it searches the real environment for states on which M_θ fails. The world model M_θ is itself a learner: it keeps training on real experience as in its backbone, and is additionally repaired on the failures the developer policy explores and those the task policy meets during training. PROWL-2 organises their training in three parts, shown in Figure 1.

Task policy curriculum learning (Section 4.1) decides *where the task policy trains*, by ranking the start states of its imagined rollouts in a buffer $\mathcal{B}_G$ by learning potential. **World-model curriculum learning** (Section 4.2) decides *where the world model is repaired*, by collecting the real trajectory segments it mispredicts in a pool $\mathcal{B}_{\mathrm{WM}}$. **Dual curriculum learning** (Section 4.3) couples the two through a *fidelity gate*, which tests the task policy's start states against the real environment, withholds those the world model mispredicts, and sends them for repair.

The task policy therefore trains only where learning potential is high and the world model's rollouts are high-fidelity, and the world model is repaired where the task policy needs it. As repair makes the world model reliable on harder start states, those states return to training, so the curriculum can grow harder as the world model improves. PROWL-2 is *world-model agnostic*: it changes what the world model and the task policy train on, not their architectures or objectives. We instantiate it on two backbones of different kinds, the actor–critic learner MARIE (Zhang et al., 2024) on SMACv2 and the planning learner SeqWM (Zhao et al., 2026) on MQE. Appendix D gives pseudocode for PROWL-2, and Appendix E lists all hyperparameters.

4.1 Task Policy Curriculum Learning

Task policy curriculum learning builds on IMAC (Güzel et al., 2025), which prioritises the start states of imagined rollouts for a single agent, and extends it to the multi-agent setting: a start state $\ell = (e, t)$ in the task policy curriculum buffer $\mathcal{B}_G$ indexes the joint action–observation history of all agents at step t of a recorded episode, and imagination rolls all agents forward together from it.

Learning potential. On SMACv2, where the task policy is an actor–critic trained in imagination, we define a start state's learning potential by IMAC's positive value loss $s_G(\ell)$ (Eq. 1), averaged over agents. For MQE, where the task policy is a planner whose twin critic Q_1, Q_2 trains on real transitions, we measure it on real replay instead, as the magnitude of the critic's temporal-difference residual $\delta_Q = y_0 - Q_1(z(o_0), a_0)$ on a recorded transition (o_0, a_0), where $z(o_0)$ is the encoded

latent state and y_0 the critic's multi-step temporal-difference target from the recorded rewards and the target critic's value of the recorded successor.

Curriculum buffer sampling. We update a start state's score every time it is replayed. On SMACv2, with probability ρ we draw a batch of start states from $\mathcal{B}_G$ by PLR's rank-based prioritisation with staleness (Jiang et al., 2021), and otherwise uniformly from replay; on MQE, we draw a fraction ρ of each critic and actor batch in proportion to $|\delta_Q|$. In both cases we exclude start states in quarantine, whose fidelity error exceeds the gate threshold (Section 4.3). Policy learning depends on how faithful the world model is where it trains: on SMACv2 every update comes from imagined rollouts, and on MQE the planner chooses every action from the world model's rollouts. Without a check against real data, a state the world model mispredicts can look like one with high learning potential and steer the task policy towards the world model's own errors, as in IMAC, which trusts its world model everywhere; the quarantine removes this weakness.

4.2 World-Model Curriculum Learning

To repair the world model where it fails, we build the world-model curriculum on PROWL-1 (Güzel et al., 2026), which trains a developer policy to find the failures of a single-agent world model and repairs the world model on them. We extend it to the multi-agent setting: the developer policy acts for all agents jointly, and we score failures on per-agent, coordination-relevant quantities rather than a single reconstruction loss. In contrast to PROWL-1, the developer policy starts from a random initialisation, instead of a copy of the task policy. This leads to a higher task-policy win rate as we show in our ablation (Appendix C).

Developer policy. The developer policy π_D runs in its own instance of the real environment and never acts for the task. As in PROWL-1, on SMACv2 we train π_D with PPO (Schulman et al., 2017) to maximise

$$J_D(\pi_D) = \mathbb{E}_{\tau\sim\pi_D}[R(\tau)] - c_{\mathrm{KL}}\,\mathbb{E}_{s\sim\pi_D}[D_{\mathrm{KL}}(\pi_D(\cdot \mid s)\,\|\,\pi_{\mathrm{ref}}(\cdot \mid s))]\,, \tag{2}$$

where $R(\tau)$ is the reward of a collected segment τ, π_{ref} a frozen copy of the developer policy at its random initialisation, D_{KL} the Kullback–Leibler divergence between their action distributions at state s, and c_{KL} its weight. The KL term keeps the developer policy close to its random initialisation while it learns to favour states where the world model fails, and prevents the entropy collapse observed without it (Appendix C). Unlike regret-based adversaries in environment design (Dennis et al., 2020; Parker-Holder et al., 2022), which seek failures of the task policy, the developer policy seeks world model failures. On MQE, the developer is the task planner itself, run in its own environment instance with a bonus $\beta_{\mathrm{acq}}|Q_1 - Q_2|$ for twin-critic disagreement that steers it towards states where the world model is likely to fail. The bonus only affects where the developer collects data; which experience the task policy trains on is still decided by the fidelity gate, never by disagreement.

Reward and failure score. To reward only failures the world model can learn from, we score each segment before and after the repair update that follows it:

$$R(\tau) = s_{\mathrm{fail}}^{\mathrm{before}}(\tau) + \beta_{\mathrm{LP}}\big(s_{\mathrm{fail}}^{\mathrm{before}}(\tau) - s_{\mathrm{fail}}^{\mathrm{after}}(\tau)\big), \tag{3}$$

where β_{LP} weights a learning-progress term (Oudeyer et al., 2007; Portelas et al., 2019) that steers the search away from errors the world model cannot learn. The *failure score* s_{fail} measures how badly M_θ predicts a recorded real segment $\tau = (o_{0:H}, a_{0:H-1}, r_{0:H-1}, m_{0:H-1})$ of observations, joint actions, team rewards and available-action masks. We roll M_θ forward from the segment's start under the recorded actions, and compare each prediction with the recorded continuation. Both follow the same actions from the same state, so any divergence is the world model's error, and no new interaction is needed. Because low prediction loss does not guarantee good control (Lambert et al., 2020; Grimm et al., 2020), we choose the errors $E_k(\tau)$ for their relevance to control. On SMACv2 we use $k \in \mathcal{C} = \{\mathrm{death}, \mathrm{mask}, r, \mathrm{obs}\}$ (ally death timing read from each unit's predicted health, available-action-mask disagreement, reward and observation error), and on MQE per-agent latent error at horizons 1–3 and reward error (Appendix A). We standardise each error against its running statistics, which puts the errors on a common scale, and average them with equal weights $w_k = 1/|\mathcal{C}|$:

$$s_{\mathrm{fail}}(\tau) = \sum_{k\in\mathcal{C}} w_k\,\frac{E_k(\tau) - \bar{\mu}_k}{\bar{\sigma}_k}, \tag{4}$$

where $\bar{\mu}_k$ and $\bar{\sigma}_k$ are exponential moving averages of the mean and standard deviation of E_k over recent developer and matched random-policy episodes, which keep the reward informative as errors shrink. On MQE, where the developer policy is not trained, we compute the statistics over the segments currently in $\mathcal{B}_{\mathrm{WM}}$.

World-model repair mechanism. The world-model curriculum buffer $\mathcal{B}_{\mathrm{WM}}$ holds recorded real segments of two kinds: (i) those collected by the developer policy and (ii) the continuations of high-learning-potential start states from the dual curriculum (Section 4.3) whose world-model rollouts diverged. Each entry j has a priority p_j: the failure score for developer segments, and the start state's learning potential for curriculum segments, so repair focuses on the states the task policy is waiting to use. The world model trains on a blend of ordinary replay $\mathcal{D}_{\mathrm{env}}$ and the pool:

$$\mathcal{D}_{\mathrm{WM}} \sim (1-\rho_{\mathrm{WM}})\,\mathcal{D}_{\mathrm{env}} + \rho_{\mathrm{WM}}\,\mathcal{B}_{\mathrm{WM}}, \tag{5}$$

with repair fraction ρ_{WM} and pool entries sampled $\propto \exp(p_j)$; the objective is unchanged, and repair updates run after each developer cycle. Developer entries lose priority over time and are evicted once their priority falls below a fixed bar; when the pool is full, the lowest-priority entry is evicted. Whereas PROWL-1 repairs the world model only where the developer policy finds errors, PROWL-2 also repairs it on the high-learning-potential trajectories of the task policy curriculum.

4.3 Dual Curriculum Learning

Dual curriculum learning, the novel component of PROWL-2, combines world-model repair with task-policy training. It couples the two curricula through a *fidelity gate*, which decides from which start states of the task policy curriculum the world model produces high-fidelity rollouts. Each start state $\ell = (e, t)$ in $\mathcal{B}_G$ points into the replay buffer $\mathcal{D}_{\mathrm{env}}$, so its real continuation τ_ℓ is already stored and can be evaluated at no extra cost, leaving the task policy's stochastic imagined training unchanged. The gate rolls M_θ forward from step t under the recorded actions, as in Section 4.2, and measures the errors $E_k(\tau_\ell)$ against the recorded continuation. We then standardise them against fixed reference statistics, the mean μ_k^{ref} and standard deviation σ_k^{ref} of E_k on random replay of the backbone, computed before PROWL-2 starts and never updated, and average them with the same equal weights $w_k = 1/|\mathcal{C}|$:

$$F_\theta(\ell) = \sum_{k\in\mathcal{C}} w_k \,\frac{E_k(\tau_\ell) - \mu_k^{\mathrm{ref}}}{\sigma_k^{\mathrm{ref}}}. \tag{6}$$

We call $F_\theta(\ell)$ the *fidelity error* of ℓ: larger values indicate that the world model's rollout from ℓ departs further from the recorded continuation.

Fidelity gate threshold. A start state is quarantined when its fidelity error exceeds a threshold ϵ_{WM}. On SMACv2 we use $\epsilon_{\mathrm{WM}} = 0.5$, which lies at the 72nd–83rd percentile of the backbone's fidelity error on its reference data, so the gate quarantines a start state when the world model predicts it worse than it predicts 72–83% of the backbone's replay (Appendix B.1). The threshold is fixed on the reference scale rather than set relative to each audit: a percentile of each audited batch would quarantine a fixed fraction regardless of how accurately the world model predicts it, whereas a fixed threshold quarantines fewer states as the world model improves. We set it once before training and use it in every SMACv2 scenario and method without tuning.

Gating mechanism. Every N_{gate} environment transitions, the gate audits two groups of start states. The first comprises the K highest-priority states in $\mathcal{B}_G$, which the curriculum samples most often and where world-model errors are therefore most harmful. Restricting the audit to them also bounds its cost. The second comprises all states in the quarantine set Q, which we re-audit to determine whether repair has restored their fidelity. Each audited state ℓ is admitted to training if $F_\theta(\ell) \le \epsilon_{\mathrm{WM}}$ and quarantined in Q otherwise; we exclude quarantined states from sampling until re-admission.

Exchange between the curricula. The gate moves start states between the two curricula in both directions. *From $\mathcal{B}_G$ to $\mathcal{B}_{\mathrm{WM}}$:* a quarantined state keeps its entry in $\mathcal{B}_G$ but is no longer sampled for training, and its recorded continuation τ_ℓ is added to $\mathcal{B}_{\mathrm{WM}}$ for repair. This breaks the feedback loop in which a world-model error inflates a state's learning potential and the task policy continues to train on the error. *From $\mathcal{B}_{\mathrm{WM}}$ to $\mathcal{B}_G$:* we re-evaluate quarantined states at each subsequent audit. After n_{pass} consecutive passes, a state's learning potential is recomputed under the repaired world model, its segment leaves $\mathcal{B}_{\mathrm{WM}}$, and it returns to training; after n_{fail} failed audits, it is removed from both curricula.

5 EXPERIMENTS

5.1 SMACv2

We evaluate PROWL-2 on nine procedurally varying SMACv2 scenarios: 5v5, 10v10, and 10v11 combat for each of *Terran*, *Protoss*, and *Zerg*, where 10v10 scales coordination and 10v11 adds a numerical disadvantage. We compare against MARIE (Zhang et al., 2024), the model-based learner PROWL-2 is built on, and five model-free baselines: QMIX (Rashid et al., 2018), QPLEX (Wang et al., 2021), MAPPO (Yu et al., 2022), HAPPO, and HATRPO (Kuba et al., 2022). All methods train the task policy π_G on 1M real environment steps with identical task-policy update counts.

In Repair, Dual and PROWL-2 (Table 1), the world model also performs 25 repair updates per developer cycle on batches mixing $\mathcal{D}_{\text{env}}$ and $\mathcal{B}_{\text{WM}}$ (Eq. 5), about 6% more world-model gradient steps than MARIE, and the developer policy is trained with its own PPO updates. Filling $\mathcal{B}_{\text{WM}}$ requires the developer π_D to interact with a separate environment instance, a cost that is substantial: counting the matched random-policy rollouts used to normalise its reward, the developer adds between one and two times the task-policy interaction on SMACv2 (roughly 8% on MQE), with compute in Appendix G. Repair carries this same cost and gains little (Table 1), so neither the additional interaction nor the repair updates are the origin of PROWL-2's gain. Each seed trains the backbone alone until 100K steps on 5v5 and 200K on 10v10 and 10v11, after the knee of its world-model loss curve (Appendix F). PROWL-2 and its ablations branch from this shared checkpoint, where the developer policy is initialised, so all arms of a seed are identical up to it (dashed lines in Figure 2). Evaluation runs 10 stochastic episodes every 500 steps, identically across methods and not charged to the budget; reported win rates average the final 200 evaluation points per seed, mean ± standard error over three seeds. Hyperparameters are in Appendix E.

Table 1: **Performance and ablation on SMACv2.** Evaluation win rate (%, ↑) across nine procedurally varying combat scenarios, mean ± standard error over three seeds. Model-free baselines on the left; model-based methods sharing the MARIE backbone on the right, adding components cumulatively: Repair adds active failure discovery and continual world-model repair, Dual adds the imagined agent curriculum without a fidelity gate, and PROWL-2 adds the gate.

		Model-free					Model-based (MARIE backbone)			
Race	Combat	QMIX	QPLEX	MAPPO	HAPPO	HATRPO	MARIE	Repair	Dual	**PROWL-2**
Terran	5v5	18.9±2.6	21.2±0.8	20.8±3.1	17.0±0.1	10.4±0.6	45.0±1.8	49.6±1.5	36.6±0.4	**52.9±0.9**
	10v10	8.6±2.0	8.3±1.8	11.6±1.9	1.8±0.3	0.1±0.1	21.2±1.2	21.9±1.7	19.4±1.8	**27.9±2.3**
	10v11	3.3±1.0	3.2±1.4	2.6±0.8	0.9±0.0	0.0±0.0	5.6±0.1	6.1±0.9	5.3±0.7	**10.7±1.2**
Protoss	5v5	14.1±1.3	8.4±0.5	16.1±4.9	7.1±1.2	4.3±1.1	54.1±0.2	55.8±2.8	47.1±1.1	**60.2±0.6**
	10v10	2.5±0.2	4.0±1.7	5.1±0.2	1.6±0.1	0.3±0.3	27.9±2.3	30.4±1.0	20.3±1.9	**34.8±1.1**
	10v11	0.3±0.3	0.1±0.1	0.4±0.3	0.4±0.3	0.0±0.0	6.0±0.8	6.4±1.2	4.9±0.8	**8.9±0.6**
Zerg	5v5	11.1±0.7	7.8±2.2	10.4±2.1	12.6±1.3	8.3±0.6	27.5±0.6	27.7±0.3	24.4±0.7	**28.7±0.3**
	10v10	3.9±0.7	3.6±1.6	5.0±1.0	3.4±0.3	0.4±0.3	12.2±1.3	13.7±2.0	11.8±1.4	**15.8±1.1**
	10v11	2.6±0.7	2.6±0.6	1.3±0.6	0.9±0.7	0.0±0.0	5.6±0.4	5.9±0.4	5.2±0.2	**6.7±0.4**

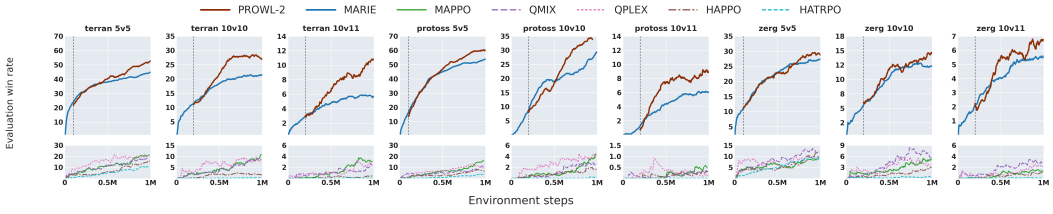


Figure 2: **Learning across nine SMACv2 combat scenarios.** Panels run left to right through *Terran*, *Protoss* and *Zerg*, each at 5v5, 10v10 and 10v11 combat. Curves report the mean evaluation win rate across three seeds. Upper panels: PROWL-2 versus its MARIE backbone; lower panels: model-free baselines. Vertical dashed lines mark the branch point, where the developer is initialised (Section 5.1).

Comparison with baselines. PROWL-2 achieves the highest mean win rate in all nine settings (Table 1, Figure 2); since the backbone architecture and objective are unchanged, the gap isolates the effect of the curriculum and repair mechanisms. In relative terms the gap is larger with ten agents than with five in every race: the relative gain over the backbone is 4–18% at 5v5 and 20–91% at 10v10 and 10v11.

Ablations. The ablations in Table 1 add one component at a time. Repair alone exceeds the backbone in all nine settings, but in seven within one standard error; Dual falls below both the backbone and Repair in all nine, reflecting the feedback loop of Section 4.3; and the gate turns the same curriculum into the largest gain, with PROWL-2's increment over Repair exceeding Repair's over the backbone in eight of nine settings. The appendix covers the remaining ablations: gate controls that quarantine the same number of states at random or by world-model uncertainty (Appendix B.3); the threshold and what the gate audits during training (Appendices B.1 and B.2); the learning-potential score, the developer's initialisation and its KL anchor (Appendix C); whether flagged failures are repairable (Appendix A.2); and the choice of branch point (Appendix F).

5.2 Multi-Agent Quadruped Environment

We next evaluate PROWL-2 on embodied cooperative control in the Multi-Agent Quadruped Environment (MQE), using SeqWM (Zhao et al., 2026) as the world-model backbone (Section 4). We compare against SeqWM and four established MARL baselines: HASAC (Liu et al., 2023), MARIE (Zhang et al., 2024), MAPPO (Yu et al., 2022), and MAT (Wen et al., 2022). Rather than selecting only tasks on which the backbone struggles, we consider two matched difficulty pairs, Gate-2/Gate-3 and Shepherd-Easy/Shepherd-Hard. All methods train for 1.5M environment steps, with the developer's interaction counted in this budget; the measurement protocol is given in Appendix E.

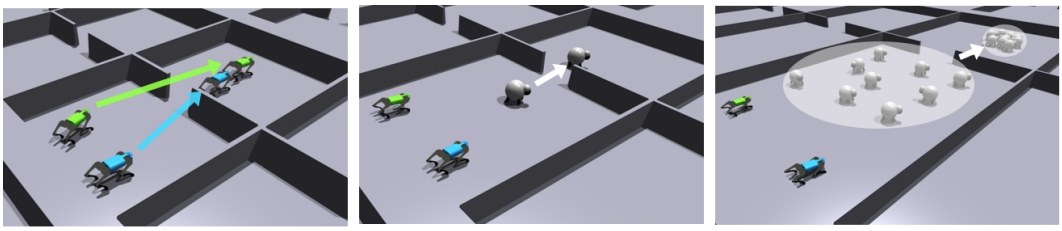


Figure 3: **MQE tasks.** Gate-n: n quadrupeds pass through a narrow gate as quickly as possible without collision. Shepherd: two quadrupeds herd one (Easy) or nine (Hard) sheep into a target region. Renders from Xiong et al. (2024).

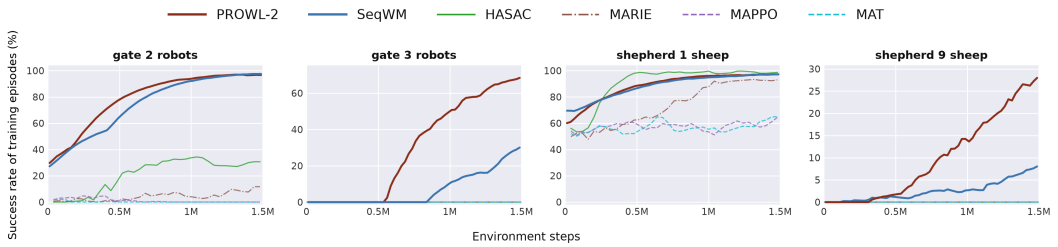


Figure 4: **Learning curves on MQE.** Success rate of training episodes over 1.5M environment steps on Gate-2, Gate-3, Shepherd-Easy, and Shepherd-Hard; mean over three seeds.

Table 2: **Success rate of training episodes on MQE** (%, ↑) at 1.5M environment steps, mean ± 95% confidence interval over three seeds. Bold marks the best mean in each row and any mean within one point of it. Repair and Dual are the ablations of Table 1 instantiated on the SeqWM backbone.

	Baselines				SeqWM backbone			
Task	HASAC	MARIE	MAPPO	MAT	SeqWM	Repair	Dual	**PROWL-2**
Gate-2	29.3 ± 20.8	10.3 ± 1.7	0.0 ± 0.0	0.0 ± 0.0	$\mathbf{99.1 \pm 1.8}$	$\mathbf{99.0 \pm 0.7}$	$\mathbf{99.1 \pm 0.6}$	$\mathbf{99.2 \pm 0.8}$
Gate-3	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	29.8 ± 2.5	34.6 ± 1.1	12.3 ± 2.3	$\mathbf{70.4 \pm 2.5}$
Shepherd-Easy	$\mathbf{99.7 \pm 0.3}$	92.7 ± 0.5	64.4 ± 3.7	64.1 ± 3.4	95.2 ± 2.2	$\mathbf{99.1 \pm 0.7}$	$\mathbf{98.9 \pm 0.9}$	$\mathbf{99.2 \pm 1.1}$
Shepherd-Hard	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	7.3 ± 2.0	10.3 ± 2.1	7.7 ± 2.1	$\mathbf{28.2 \pm 0.9}$

Comparison with baselines. Figure 4 and Table 2 show two effects. On Gate-2 and Shepherd-Easy, PROWL-2 reaches 99.2%, level with the other SeqWM-based methods on Gate-2 and within half a point of HASAC's 99.7% on Shepherd-Easy, and it reaches its final level sooner on Gate-2; on Shepherd-Easy the model-free HASAC learns fastest and all strong methods end within a few points. The harder variants separate the methods: on Gate-3 PROWL-2 begins to succeed at roughly 0.55M steps and reaches 70.4%, whereas SeqWM begins near 0.85M and reaches 29.8%; on Shepherd-Hard PROWL-2 reaches 28.2% against 7.3%, with every other baseline at zero on both. Both world-model learners are still improving at 1.5M, so these are gaps within a fixed budget, and they are largest where the backbone alone makes the least progress.

Ablations. The ablation columns of Table 2 show the same structure as on SMACv2, more sharply. Repair alone adds only a few points over the backbone; the ungated curriculum falls well below it on Gate-3 and is inert on Shepherd-Hard; and the gate supplies almost the entire gain, taking the same curriculum from 12.3% to 70.4% on Gate-3.

6 LIMITATIONS

On SMACv2 our 1M-step budget is five times MARIE's own, so the comparison covers both learning speed and final performance. On MQE our 1.5M budget is below SeqWM's 4M, a compute constraint, so it measures sample efficiency; both world-model learners are still improving on the hardest tasks, and whether the gap persists asymptotically is open. Fidelity targets are chosen per domain, a design step that a learned fidelity measure could remove, but with that choice adapted the same gate and repair loop transferred across discrete and continuous control, actor–critic and planning backbones, and teams of two to ten agents. We did not evaluate PROWL-2 in single-agent settings; the framework applies unchanged with $n = 1$, but whether the gains carry over is untested. Finally, the developer's real-environment search adds one to two times the task-policy interaction on SMACv2; on MQE, where it is charged to the task budget, PROWL-2 still leads.

7 CONCLUSION

We introduced PROWL-2, a fidelity-gated dual-curriculum framework for multi-agent learning inside world models. PROWL-2 separates learning signal from world-model error by coupling an agent-learning curriculum with continual world-model repair through an environment-grounded fidelity gate, so that reliable high-learning-potential experience trains the policy while unreliable experience repairs the world model and, once repaired, returns to the curriculum. Across SMACv2 and MQE this separation improves on the underlying world-model learners in every SMACv2 scenario and on the harder MQE tasks, and matches them where all strong methods saturate. The ablations show why the two curricula cannot simply be combined: repair alone yields little, an ungated imagined curriculum degrades performance below the backbone, and the gate converts that same curriculum into the largest single gain. Being agnostic to the world-model architecture and the MARL learner beneath it, PROWL-2 improves world-model agent training on both an actor–critic and a planning backbone, across discrete and continuous environments. Natural next steps are evaluating PROWL-2 on coordination among humanoid robots, whose contact-rich dynamics make world-model errors more frequent, and on long-horizon multi-agent games, where the fidelity gate would need audits that extend with the imagination horizon.

AI DISCLOSURE STATEMENT

In this work, we have not used generative AI tools for any of the tasks that require disclosure (generating data, formulating claims). However, we did use generative LLMs for tasks with recommended disclosure, specifically to edit the research paper to improve grammar, wording, and overall readability. All AI-assisted edits were manually reviewed by the authors to ensure the original meaning and scientific accuracy were preserved. We take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Algorithm 1 and the text that follows it (Appendix D) give PROWL-2 as implemented on both backbones. Appendix E lists every hyperparameter with its source, including the environment settings and evaluation protocol; Appendix A defines the fidelity targets and the construction of the reference statistics; and Appendix G reports hardware and compute. All arms of a seed branch from a shared backbone checkpoint, and training and evaluation environment seeds are disjoint.

REFERENCES

Zaheer Abbas, Samuel Sokota, Erin J. Talvitie, and Martha White. Selective dyna-style planning under limited model capacity. ICML'20. JMLR.org, 2020.

Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in Atari. In *Advances in Neural Information Processing Systems*, 2024.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pp. 41–48, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553380. URL https://doi.org/10.1145/1553374.1553380.

Jacob Buckman, Danijar Hafner, George Tucker, Eugene Brevdo, and Honglak Lee. Sample-efficient reinforcement learning with stochastic ensemble value expansion. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/f02208a057804ee16ac72ff4d3cec53b-Paper.pdf.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models, 2018. URL https://arxiv.org/abs/1805.12114.

Michael Dennis, Natasha Jaques, Eugene Vinitsky, Alexandre Bayen, Stuart Russell, Andrew Critch, and Sergey Levine. Emergent complexity and zero-shot transfer via unsupervised environment design. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Vladimir Egorov and Aleksei Shpilman. Scalable multi-agent model-based reinforcement learning, 2022. URL https://arxiv.org/abs/2205.15023.

Benjamin Ellis, Jonathan Cook, Skander Moalla, Mikayel Samvelyan, Mingfei Sun, Anuj Mahajan, Jakob N. Foerster, and Shimon Whiteson. SMACv2: An improved benchmark for cooperative multi-agent reinforcement learning, 2023. URL https://arxiv.org/abs/2212.07489.

Vladimir Feinberg, Alvin Wan, Ion Stoica, Michael I. Jordan, Joseph E. Gonzalez, and Sergey Levine. Model-based value estimation for efficient model-free reinforcement learning, 2018. URL https://arxiv.org/abs/1803.00101.

Jakob N. Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the Thirty-Second AAAI*

Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence, AAAI'18/IAAI'18/EAAI'18. AAAI Press, 2018. ISBN 978-1-57735-800-8.

Christopher Grimm, Andre Barreto, Satinder Singh, and David Silver. The value equivalence principle for model-based reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 5541–5552. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/3bb585ea00014b0e3ebe4c6dd165a358-Paper.pdf.

Ahmet H. Güzel, Matthew Thomas Jackson, Jarek Luca Liesen, Tim Rocktäschel, Jakob Nicolaus Foerster, Ilija Bogunovic, and Jack Parker-Holder. Imagined autocurricula. In *Advances in Neural Information Processing Systems*, 2025.

Ahmet H. Güzel, Jenny Seidenschwarz, Benjamin Graham, Jonathan Sadeghi, Jeffrey Hawke, Jack Parker-Holder, and Ilija Bogunovic. PROWL: Prioritized regret-driven optimization for world model learning, 2026. URL https://arxiv.org/abs/2605.18803.

David Ha and Jürgen Schmidhuber. World models. 2018. doi: 10.5281/ZENODO.1207631. URL https://zenodo.org/record/1207631.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination, 2020. URL https://arxiv.org/abs/1912.01603.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models, 2024. URL https://arxiv.org/abs/2301.04104.

Danijar Hafner, Wilson Yan, and Timothy Lillicrap. Training agents inside of scalable world models, 2025. URL https://arxiv.org/abs/2509.24527.

Dom Huh and Prasant Mohapatra. Multi-agent reinforcement learning: A comprehensive survey, 2024. URL https://arxiv.org/abs/2312.10256.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/5faf461eff3099671ad63c6f3f094f7f-Paper.pdf.

Minqi Jiang, Edward Grefenstette, and Tim Rocktäschel. Prioritized level replay. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4940–4950. PMLR, 2021. URL https://proceedings.mlr.press/v139/jiang21b.html.

Minqi Jiang, Michael Dennis, Jack Parker-Holder, Andrei Lupu, Heinrich Küttler, Edward Grefenstette, Tim Rocktäschel, and Jakob Foerster. Grounding aleatoric uncertainty for unsupervised environment design, 2022. URL https://arxiv.org/abs/2207.05219.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 21810–21823. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/f7efa4f864ae9b88d43527f4b14f750f-Paper.pdf.

Jakub Grudzien Kuba, Ruiqing Chen, Muning Wen, Ying Wen, Fanglei Sun, Jun Wang, and Yaodong Yang. Trust region policy optimisation in multi-agent reinforcement learning, 2022. URL https://arxiv.org/abs/2109.11251.

Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and P. Abbeel. Model-ensemble trust-region policy optimization. *ArXiv*, abs/1802.10592, 2018. URL https://api.semanticscholar.org/CorpusID:3536221.

Nathan Lambert, Brandon Amos, Omry Yadan, and Roberto Calandra. Objective mismatch in model-based reinforcement learning. In Alexandre M. Bayen, Ali Jadbabaie, George Pappas, Pablo A. Parrilo, Benjamin Recht, Claire Tomlin, and Melanie Zeilinger (eds.), *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, volume 120 of *Proceedings of Machine Learning Research*, pp. 761–770. PMLR, 10–11 Jun 2020. URL https://proceedings.mlr.press/v120/lambert20a.html.

Jiarong Liu, Yifan Zhong, Siyi Hu, Haobo Fu, Qiang Fu, Xiao-Jun Chang, and Yaodong Yang. Maximum entropy heterogeneous-agent reinforcement learning. In *International Conference on Learning Representations*, 2023. URL https://api.semanticscholar.org/CorpusID:259203104.

Qihan Liu, Jianing Ye, Xiaoteng Ma, Jun Yang, Bin Liang, and Chongjie Zhang. Efficient multi-agent reinforcement learning by planning, 2024. URL https://arxiv.org/abs/2405.11778.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments, 2020. URL https://arxiv.org/abs/1706.02275.

Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models, 2023. URL https://arxiv.org/abs/2209.00588.

Thanh Thi Nguyen, Ngoc Duy Nguyen, and Saeid Nahavandi. Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications. *IEEE Transactions on Cybernetics*, 50(9):3826–3839, 2020. ISSN 2168-2275. doi: 10.1109/tcyb.2020.2977374. URL http://dx.doi.org/10.1109/TCYB.2020.2977374.

Zepeng Ning and Lihua Xie. A survey on multi-agent reinforcement learning and its application. *Journal of Automation and Intelligence*, 3(2):73–91, 2024. ISSN 2949-8554. doi: https://doi.org/10.1016/j.jai.2024.02.003. URL https://www.sciencedirect.com/science/article/pii/S2949855424000042.

Frans A. Oliehoek and Christopher Amato. *The Decentralized POMDP Framework*, pp. 11–32. Springer International Publishing, Cham, 2016. ISBN 978-3-319-28929-8. doi: 10.1007/978-3-319-28929-8_2. URL https://doi.org/10.1007/978-3-319-28929-8_2.

Pierre-Yves Oudeyer, Frdric Kaplan, and Verena V. Hafner. Intrinsic motivation systems for autonomous mental development. *IEEE Transactions on Evolutionary Computation*, 11(2):265–286, 2007. doi: 10.1109/TEVC.2006.890271.

Feiyang Pan, Jia He, Dandan Tu, and Qing He. Trust the model when it is confident: Masked model-based actor-critic, 2020. URL https://arxiv.org/abs/2010.04893.

Jack Parker-Holder, Minqi Jiang, Michael D Dennis, Mikayel Samvelyan, Jakob Nicolaus Foerster, Edward Grefenstette, and Tim Rocktäschel. Evolving curricula with regret-based environment design. *arXiv preprint arXiv:2203.01302*, 2022.

Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement, 2019. URL https://arxiv.org/abs/1906.04161.

Rémy Portelas, Cédric Colas, Katja Hofmann, and Pierre-Yves Oudeyer. Teacher algorithms for curriculum learning of deep rl in continuously parameterized environments, 2019. URL https://arxiv.org/abs/1910.07224.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning, 2018. URL https://arxiv.org/abs/1803.11485.

Jan Robine, Marc Höftmann, Tobias Uelwer, and Stefan Harmeling. Transformer-based world models are happy with 100k interactions, 2023. URL https://arxiv.org/abs/2303.07109.

Alexander Rutherford, Michael Beukman, Timon Willi, Bruno Lacerda, Nick Hawes, and Jakob Foerster. No regrets: Investigating and improving regret approximations for curriculum discovery, 2024. URL https://arxiv.org/abs/2408.15099.

Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philip H. S. Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge, 2019. URL https://arxiv.org/abs/1902.04043.

Mikayel Samvelyan, Akbir Khan, Michael Dennis, Minqi Jiang, Jack Parker-Holder, Jakob Foerster, Roberta Raileanu, and Tim Rocktäschel. Maestro: Open-ended environment design for multi-agent reinforcement learning, 2023. URL https://arxiv.org/abs/2303.03376.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay, 2016. URL https://arxiv.org/abs/1511.05952.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

John D. Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. *CoRR*, abs/1506.02438, 2015. URL https://api.semanticscholar.org/CorpusID:3075448.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning, 2017. URL https://arxiv.org/abs/1706.05296.

Edan Toledo and Amanda Prorok. CoDreamer: Communication-based decentralised world models, 2024. URL https://arxiv.org/abs/2406.13600.

Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. Qplex: Duplex dueling multi-agent q-learning, 2021. URL https://arxiv.org/abs/2008.01062.

Xihuai Wang, Zhicheng Zhang, and Weinan Zhang. Model-based multi-agent reinforcement learning: Recent progress and prospects, 2022. URL https://arxiv.org/abs/2203.10603.

Muning Wen, Jakub Grudzien Kuba, Runji Lin, Weinan Zhang, Ying Wen, Jun Wang, and Yaodong Yang. Multi-agent reinforcement learning is a sequence modeling problem. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Ziyan Xiong, Bo Chen, Shiyu Huang, Wei-Wei Tu, Zhaofeng He, and Yang Gao. Mqe: Unleashing the power of interaction with multi-agent quadruped environment. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5918–5924, 2024. doi: 10.1109/IROS58592.2024.10801682.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of PPO in cooperative, multi-agent games, 2022. URL https://arxiv.org/abs/2103.01955.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization, 2020. URL https://arxiv.org/abs/2005.13239.

Weipu Zhang, Gang Wang, Jian Sun, Yetian Yuan, and Gao Huang. Storm: Efficient stochastic transformer based world models for reinforcement learning, 2023. URL https://arxiv.org/abs/2310.09615.

Yang Zhang, Chenjia Bai, Bin Zhao, Junchi Yan, Xiu Li, and Xuelong Li. Decentralized transformers with centralized aggregation are sample-efficient multi-agent world models. *arXiv preprint arXiv:2406.15836*, 2024.

Yang Zhang, Xinran Li, Jianing Ye, Shuang Qiu, Delin Qu, Xiu Li, Chongjie Zhang, and Chenjia Bai. Revisiting multi-agent world modeling from a diffusion-inspired perspective, 2025. URL https://arxiv.org/abs/2505.20922.

Zijie Zhao, Honglei Guo, Shengqian Chen, Kaixuan Xu, Bo Jiang, Yuanheng Zhu, and Dongbin Zhao. Empowering multi-robot cooperation via sequential world models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=IvUM6UwYCJ.

A WORLD-MODEL FIDELITY SCORE

This appendix defines the fidelity targets (§A.1), shows that the failures they flag are repairable (§A.2), and gives the MQE targets (§A.3).

A.1 TARGET DEFINITIONS

A recorded segment of length H is replayed through M_θ open-loop under its recorded joint actions, conditioned on the stacked observation context at its start ($t_0 = t_i + 4$ for a frame stack of 5), and each prediction is compared with the recorded continuation. Hatted quantities are predictions.

Ally death timing. Let $D(\tau)$ be the allies that die within the segment. A unit is predicted dead when its predicted own-health channel falls to 0.05 or below, and truly dead when its recorded health reaches 0. With $t_i^\star$ and $\hat{t}_i^\star$ the true and first predicted death steps of ally i,

$$E_{\text{death}}(\tau) = \frac{1}{\max(1, |D(\tau)|)} \sum_{i \in D(\tau)} c_i, \qquad c_i = \begin{cases} |\hat{t}_i^\star - t_i^\star|, & \text{if death is predicted,} \\ H, & \text{otherwise.} \end{cases} \tag{7}$$

Death is read from the health channel rather than the episode-level termination signal, which preserves unit identity and timing, and is averaged over dying allies so that the score reflects per-death fidelity rather than the number of casualties. The own-health channel is located from the observation layout of each race (6 own features for Terran and Zerg, 7 for Protoss).

Action mask. E_{mask} is the sum of two rates over $t = 1, \ldots, H$: the fraction of (agent, action) entries whose rounded predicted availability differs from the recorded mask, and the fraction of agents whose predicted mask is structurally invalid—no action available for a living agent, the no-op set or the stop action cleared, or an attack action available on an enemy the predicted observation marks as not attackable.

Reward and observation. $E_r = \frac{1}{H} \sum_{t=1}^{H} |\hat{r}_t - r_t|$, with $\hat{r}_t$ the agent-averaged predicted reward and r_t the team reward; E_{obs} is the mean absolute error over all observation entries of all agents, for $t = 0, \ldots, H$.

Horizons. The reference statistics of Eq. 6 are computed at $H = 14$. Each audited batch is rolled for the smallest $\min(14, \text{steps left})$ among its segments, between 8 and 14 steps, because a recorded successor is required at every rolled step; the developer scores its own episodes over their full length (at most 25 steps).

A.2 ARE FLAGGED FAILURES REPAIRABLE?

A developer rewarded for error is useful only if the errors it finds can be repaired. We bucket replay by failure signature, fine-tune a fresh copy of the base world model θ_0 for a fixed number of steps on each bucket's training split ($\theta_0 \to \theta_b$), and measure the reduction in death error on the bucket's held-out split,

$$\Phi_b = E_{\text{death}}^{\theta_0}(\text{HO}_b) - E_{\text{death}}^{\theta_b}(\text{HO}_b). \tag{8}$$

Targeted adaptation on the missed-death bucket reduces held-out death error by $\Phi = 4.64$ steps per death, matching its training-split reduction, so the gain generalises rather than memorises; the same adaptation on the low-error clean bucket worsens it ($\Phi = -2.81$). Missed deaths are therefore a coverage gap that targeted training closes, not a representational limit of the model, which is the premise behind repair and behind the learning-progress term of the developer reward (Eq. 3). This analysis is offline, on Zerg 5v5 replay, and separate from the reported runs.

A.3 MQE TARGETS

On MQE the world model is latent, so fidelity is measured in its latent space. For each agent i and horizon $h \in \{1, 2, 3\}$ the latent error is the mean squared difference between the open-loop predicted latent and the encoding of the recorded observation h steps ahead, and the reward error

is the absolute difference between the decoded predicted reward and the recorded team reward, averaged over agents. Segments are valid only up to their first terminal. Each error is z-scored and the composite is the mean over agents and horizons of the latent z-score plus the reward z-score. The gate reads the one-step ($h = 1$) composite against the 95th percentile of a trailing window of uniformly sampled replay (Appendix D).

B FIDELITY-GATE ANALYSIS

This appendix examines the gate on SMACv2: what its threshold means (§B.1), what it sees when it audits the curriculum (§B.2), and whether its benefit comes from choosing states by fidelity or from quarantining states at all (§B.3).

B.1 THE THRESHOLD

The threshold $\epsilon_{\mathrm{WM}} = 0.5$ is the absolute counterpart of an upper-tail rule: rather than flagging a fixed share of each audited batch, it flags states whose error exceeds a fixed point in the upper tail of the backbone's own error distribution. It was fixed on the scale of Eq. 6 before any reported run, shared across scenarios, and never tuned on task performance. Table 3 scores each scenario's reference segments, random replay states of the backbone run scored by its world model, with the gate's composite: 0.5 lies at the 72nd–83rd percentile in every scenario, so the gate quarantines a start state when the world model is less reliable there than on 72–83% of the states it was trained on. Because the composite averages four reference-standardised z-scores, its standard deviation is below 1, so 0.5 is best read as a percentile, not as half a standard deviation.

Table 3: **What the threshold means per scenario.** Percentile of $\epsilon_{\mathrm{WM}} = 0.5$ and share of reference segments above it, on each scenario's reference data; training rejection, the share of newly audited start states quarantined, pooled over all audits and seeds of the reported arm; and the ratio of the two.

Scenario	Percentile of 0.5	Reference > 0.5	Training rejection	Ratio
Terran 5v5	83.2	16.8%	4.6%	0.3
Protoss 5v5	82.8	17.2%	6.6%	0.4
Zerg 5v5	78.5	21.5%	7.1%	0.3
Terran 10v10	74.5	25.5%	49.1%	1.9
Protoss 10v10	72.1	27.9%	34.4%	1.2
Zerg 10v10	74.4	25.6%	53.3%	2.1
Terran 10v11	75.5	24.5%	60.4%	2.5
Protoss 10v11	75.7	24.3%	52.0%	2.1
Zerg 10v11	77.5	22.5%	61.4%	2.7

Training-time rejection. The last two columns report the share of newly audited start states the gate quarantined during training, and its ratio to the reference tail. At 5v5 the gate rejects less than the reference tail (0.3–$0.4\times$), consistent with a world model that has improved since the reference statistics were fitted. At 10v10 and 10v11 it rejects 1.2–2.7 times the reference tail: the curriculum's highest-priority start states are disproportionately ones the world model predicts poorly, and these are also the geometries where PROWL-2's relative gain over the backbone is largest (Table 1). Because the threshold is absolute, these rates are measured rather than imposed. The reference and audited populations differ, so the ratio is descriptive rather than a controlled comparison.

B.2 WHAT THE GATE SEES

Figure 5 shows every start state audited at a single gate tick at three points in training, for Terran 5v5 and Terran 10v11; points beyond rank 64 are quarantined states under re-audit, which keep their stored priority. Failing states occur throughout the audited ranks, including the highest-ranked state, and across audits the correlation between stored priority and fidelity error is weak and inconsistent in sign (r from -0.42 to $+0.07$ over eight logged seeds at 5v5, 10v10 and 10v11, with $|r| \leq 0.11$ in seven), so priority cannot stand in for the audit. The share of audited states above ϵ_{WM} falls

over training, from 11–12% to 2% on Terran 5v5 and from 93% to 69% on Terran 10v11, where most audited states are quarantined states under re-audit; because the threshold is fixed, this decline reflects repair. Resolved quarantines end in re-admission in 62–96% of cases at 5v5 and 10v10 and in 34–67% at Terran 10v11, where rejection is highest.

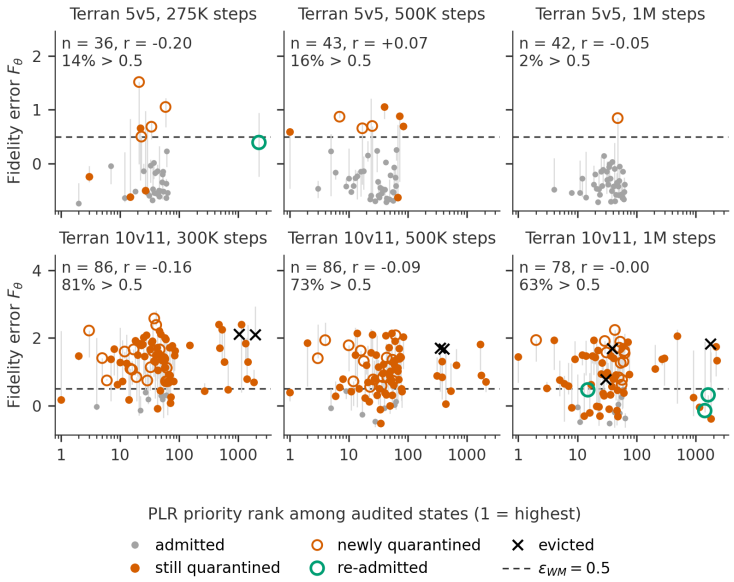


Figure 5: **Audited start states at one gate tick**, at three points in training (Terran 5v5 and Terran 10v11, seed 1 of the reported runs; F_θ recomputed offline from the checkpoint at each step). Horizontal axis: priority rank in $\mathcal{B}_G$ (1 = highest, log scale); vertical axis: fidelity error F_θ (Eq. 6); dashed line: $\epsilon_{\text{WM}} = 0.5$. Markers give the verdict at that tick: admitted, newly quarantined, still quarantined (quarantined earlier, whether it passed or failed this audit), re-admitted after a second consecutive pass, or evicted after a sixth failure. n is the number of audited states, which can fall below 64 because states with fewer than 8 recorded steps left are not audited; "> 0.5" is the share of audited states, new and re-audited, above ϵ_{WM}.

B.3 Gate controls

The gate could help because it quarantines the right start states, or merely because it quarantines some. To separate the two, we replace its verdict on Terran 5v5 while keeping everything else identical (Figure 6). At each audit the fidelity gate still computes F_θ for every audited state, which fixes the number k of states it would quarantine; each control changes only which k states are quarantined. The *random gate* draws them uniformly at random, with the same split between newly audited and already quarantined states as the fidelity gate. The *pooled random gate* draws them uniformly from all audited states together, so most of its failures fall on new states, and quarantined states are re-admitted quickly and rarely evicted. The *uncertainty gate* takes the states with the highest predictive entropy of the world model's next observation tokens, averaged over agents, tokens and the gate's four rollouts, without reference to the recorded continuation; it keeps the fidelity gate's split. In every control, quarantined states are sent for repair, and re-admission and eviction follow the same counts of passes and failures as in PROWL-2, applied to the control's own verdicts.

Until about 240K steps the random gate tracks PROWL-2; afterwards all three controls fall back towards Dual, while PROWL-2 keeps improving. Over the last 100K steps, the uncertainty gate recovers about a quarter of the gain of PROWL-2 over Dual, and the two random gates recover none, ending slightly below Dual. Most of the gate's benefit therefore comes from choosing states by their measured fidelity; quarantining states at random does not help.

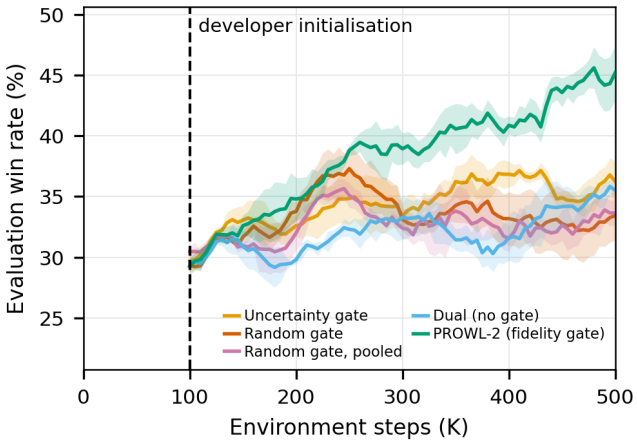


Figure 6: **Gate controls on Terran 5v5.** All arms branch from the same checkpoint at 100K steps, where the developer is initialised (dashed line). Dual has no gate; the other arms differ only in which states the gate quarantines. Mean over three seeds with ± 1 s.e. bands.

C Design ablations

Learning-potential score. Replacing the positive-TD score (Eq. 1) with the mean absolute generalised advantage, the L1 value loss of PLR (Jiang et al., 2021), changes only the ordering and eviction of start states. On Zerg 5v5 in the ungated Dual arm, both scores fall 5–7 points below Repair and are indistinguishable from each other (paired difference -1.8 ± 0.5 points, $p = 0.07$; three replicates of one seed under the default score span 4.0 points), so the harm of the ungated curriculum is not specific to the score (Table 4).

Table 4: **Learning-potential score.** SMACv2 Zerg 5v5, Dual arm (no gate), $\rho = 0.5$, branched at 100K; win rate (%) over the last 200 evaluations up to 500K steps, mean $\pm$ s.e.

Score	Seeds	Win rate
$\max(0, \delta)$ (Eq. 1, default)	3	20.2 ± 0.9
$\lvert\hat{A}\rvert$	3	18.4 ± 1.3
Repair (no curriculum)	3	25.1 ± 0.7

Developer initialisation. In the reported runs, the developer policy π_D and its reference π_{ref} start from the same random initialisation at the branch point. Cloning both from the task actor, as in PROWL-1, instead lowers the win rate of PROWL-2 on Terran 5v5 from about 45% to about 40% at 500K steps (Figure 7). The gap opens after about 220K steps and persists to the end of the run.

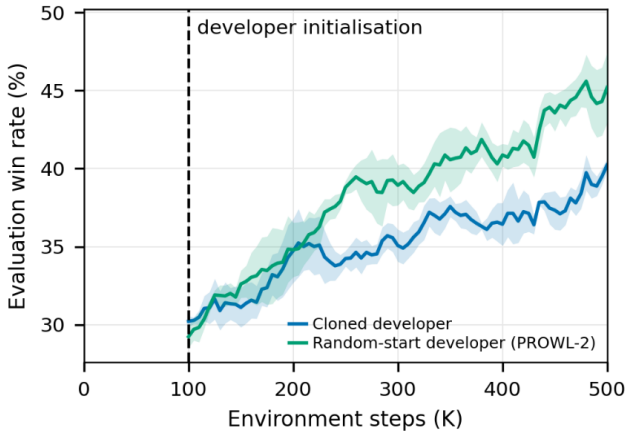


Figure 7: **Developer initialisation on Terran 5v5.** PROWL-2 with a randomly initialised developer, as in the reported runs, against a developer cloned from the task actor at the branch point. Mean over 3 seeds with ± 1 s.e. bands.

Developer KL anchor. Varying c_{KL} on Terran 5v5 in the Repair arm, the anchor controls divergence as intended ($\mathrm{KL}(\pi_D \,\|\, \pi_{\mathrm{ref}})$ from 0.04 to 0.21 to 3.9 nats) and prevents the entropy collapse seen without it. Removing it does not improve discovery: the developer's best episode in a cycle beats the frozen reference's in 30% of cycles at $c_{\mathrm{KL}} = 0$, against 51% at $c_{\mathrm{KL}} = 1.0$. Win rates differ by at most about two points and are not monotone in c_{KL} (Table 5).

Table 5: **Developer KL anchor.** SMACv2 Terran 5v5, Repair arm, developer and π_{ref} randomly initialised at 100K as in the reported runs, run to 500K, three seeds; win rate (%) over all evaluations after branching, mean $\pm$ s.e.; diagnostics are seed means over the last 20% of developer cycles.

c_{KL}	Win rate	$\mathrm{KL}(\pi_D \,\|\, \pi_{\mathrm{ref}})$	Entropy	Early term.
1.0	33.0 ± 0.2	0.044	0.234	0.017
0.1	30.9 ± 0.7	0.214	0.235	0.015
0.0	32.0 ± 0.2	3.90	0.134	0.018

D ALGORITHM

Algorithm 1 gives PROWL-2 on SMACv2, with the symbols of Section 4 and Table 6. Each run branches from a checkpoint of the backbone trained without PROWL-2 (Section 5.1), at which $\mathcal{B}_G$, $\mathcal{B}_{\mathrm{WM}}$ and Q are empty.

Algorithm 1: PROWL-2

Input : world model M_θ; task policy π_G; replay $\mathcal{D}_{\text{env}}$; reference statistics $(\mu^{\text{ref}}, \sigma^{\text{ref}})$

Initialise π_D randomly and freeze a copy as π_{ref}; $\mathcal{B}_G, \mathcal{B}_{\text{WM}}, Q \leftarrow \emptyset$;
while *task-policy steps* $<$ *budget* **do**
  Collect experience with π_G; add it to $\mathcal{D}_{\text{env}}$; update M_θ on $\mathcal{D}_{\text{env}}$ as in the backbone;
  With probability ρ draw start states from $\mathcal{B}_G \setminus Q$, otherwise from $\mathcal{D}_{\text{env}}$; imagine T steps in M_θ; update π_G;
  Score each start state by s_G (Eq. 1) and store it in $\mathcal{B}_G$;
  if *every* N_{gate} *steps* **then** `GateTick()`;
  if *every* N_{dev} *steps* **then** `DeveloperCycle()`;

procedure `GateTick()`
  $A \leftarrow$ the K highest-priority keys of $\mathcal{B}_G$, together with all of Q;
  foreach $\ell \in A$ **do**
    Roll M_θ open-loop from ℓ under the recorded actions; compute $F_\theta(\ell)$ (Eq. 6);
    if $F_\theta(\ell) > \epsilon_{\text{WM}}$ **then**
      if $\ell \notin Q$ **then** add ℓ to Q and its recorded segment to $\mathcal{B}_{\text{WM}}$ **else** after n_{fail} failed audits, evict ℓ from $\mathcal{B}_G$, Q and $\mathcal{B}_{\text{WM}}$;
    else
      if $\ell \in Q$ **then** after n_{pass} consecutive passes, re-score $s_G(\ell)$ and remove ℓ from Q and $\mathcal{B}_{\text{WM}}$;

procedure `DeveloperCycle()`
  In a separate environment instance, collect episodes with π_D and matched random-policy episodes for the normaliser of Eq. 4;
  Score each developer episode by s_{fail} (Eq. 4) and add it to $\mathcal{B}_{\text{WM}}$;
  Repair: update M_θ on batches mixing $\mathcal{D}_{\text{env}}$ and $\mathcal{B}_{\text{WM}}$ at rate ρ_{WM} (Eq. 5);
  Re-score the developer episodes; update π_D on $R(\tau)$ (Eq. 3) under the KL anchor of Eq. 2;
  Add developer segments whose score after repair is below b_G to $\mathcal{B}_G$; decay and evict stale entries of $\mathcal{B}_{\text{WM}}$;

On MQE the same loop runs synchronously on SeqWM with the substitutions of Section 4. Priority is the critic's realised TD residual $|\delta_Q|$ over critic and actor replay rows rather than s_G over imagined start states. The developer is the task planner run in a separate environment instance with the acquisition bonus $\beta_{\text{acq}}|Q_1 - Q_2|$; it does not update the actor and fills $\mathcal{B}_{\text{WM}}$ with 3-step windows. The gate scores replay rows against the 95th percentile of a trailing reference window, quarantines failing rows, excluding them from prioritised replay and adding them to $\mathcal{B}_{\text{WM}}$ for repair, and releases them once they pass; no developer segments enter the curriculum.

E HYPERPARAMETERS

Backbones. MARIE (release `b2a60f6`) runs at its released settings. The SeqWM release (`a983b7c`) ships a configuration only for Bi-DexHands, so on MQE SeqWM uses the hyperparameters reported by Zhao et al. (2026), including a planning horizon of 3 and 20-step returns, except for discount $\gamma = 0.99$ (release: 0.96) and 2,000 random warm-up steps without world-model pre-training (release: 50K steps with 2,500 pre-training updates).

Environments. SMACv2 uses the `10gen` scenarios on the 32×32 map with an episode limit of 200, continuing episodes, Zerg banelings allowed, and three engine fixes (enemy-observation clearing, agent-order shuffle, deterministic capability sampling), so numbers are not directly comparable to results on the released SMACv2. MQE episodes last 752 steps (Gate-2, Shepherd) or 1,002 steps (Gate-3), with a gate-width observation fix and bootstrapped time-limit truncations.

Budgets and evaluation. SMACv2 arms branch at 100K (5v5) or 200K (10v10, 10v11) steps and train to 1M task-policy steps in total; developer interaction is not charged to this budget. Evaluation runs 10 stochastic episodes every 500 steps on held-out environment seeds, and tables average the last 200 evaluations. MQE arms branch from a shared 8K-step checkpoint and are reported to 1.5M environment steps with developer interaction counted (8%). MQE results report the success rate of training episodes, with the planner's exploration noise on, logged every 200 environment steps (50 learner iterations) over the last 20 episodes; an episode succeeds only if all robots pass the gate, or

all sheep reach the target region. Curves report the mean over three seeds. All results use three seeds (1–3); on SMACv2, training and evaluation environment seeds are disjoint.

Table 6: **PROWL-2 hyperparameters**, added on top of the backbone settings above. Src: B = backbone world model (MARIE, Zhang et al., 2024; SeqWM, Zhao et al., 2026), I = IMAC (Güzel et al., 2025), P = PROWL-1 (Güzel et al., 2026), O = this work. X / Y gives the sources of the SMACv2 and MQE values.

	Symbol	SMACv2	MQE	Src
Agent curriculum				
Learning potential	–	s_G (Eq. 1)	$\lvert\delta_Q\rvert$	I / O
Imagination horizon	T	15	–	B
Curriculum size	$\lvert\mathcal{B}_G\rvert$	2,500	all replay rows	I / O
Rank temperature	β	0.1	–	I
Staleness coefficient	–	0.1	–	I
Replay rate	ρ	0.5	0.5	I
Fidelity gate				
Threshold	ϵ_{WM}	0.5	95th percentile	O
Target weights	w_k	0.25 each	equal	O
Reference data	–	1,200 segments	4,000-row window	O
Interval	N_{gate}	5,000 transitions	every update	O
Audited states	K	64 + quarantined	replay rows	O
Audit horizon	–	8–14 steps	1 step	O
Passes to re-admit	n_{pass}	2	1	O
Failures to evict	n_{fail}	6	–	O
Developer				
Initialisation of π_D, π_{ref}	–	random	task planner (not trained)	O
Interval	N_{dev}	200 task steps	4 of 50 iterations	O
Episodes per cycle	–	8 (+ 8 random)	–	O
Episode / window length	–	25 steps	3-step windows	O
KL coefficient	c_{KL}	1.0	–	O
Learning-progress weight	β_{LP}	0.5	–	P
Acquisition weight	β_{acq}	–	4.66	O
EMA coefficient	–	0.99	–	O
PPO learning rate	–	3×10^{-4}	–	P
World-model repair				
Repair fraction	ρ_{WM}	0.25	0.25	P
Repair updates per developer cycle	–	25 (batch 30)	–	O
Pool capacity	$\lvert\mathcal{B}_{\mathrm{WM}}\rvert$	5,000	5,000	O
Staleness decay	–	0.01	–	O
Eviction bar	–	-2	–	O
Developer $\rightarrow \mathcal{B}_G$ bar	b_G	-0.5	–	O

F BRANCH POINTS

Each run trains the backbone alone until a branch point, at which the developer starts and from which PROWL-2 and its ablations continue: 100K environment steps on SMACv2 5v5, 200K on 10v10 and 10v11, and 8K on MQE. Every branch point lies after the knee of the backbone's world-model loss, where its steep initial decrease ends (Figures 8–10). Before the knee, the world model's errors are large on most states, so a failure search or a fidelity audit would not single out specific ones.

On SMACv2, the total loss reaches its minimum at about 40K steps on 5v5 and later, up to about 90K, on 10v10 and 10v11 (Figure 8). It then rises again before declining. The loss is evaluated on each run's own replay buffer, whose content changes as the task policy improves, so this rise does not by itself indicate a less accurate world model. Figure 9 splits the loss into its terms. The observation, termination and available-action terms fall by one to three orders of magnitude and reach their knee before the branch point. The later rise of the total loss comes mainly from the observation term, which dominates it. The reward term falls on Zerg and changes little on Terran and Protoss. On MQE, the loss falls steeply until about 4K–6K steps and then declines slowly (Figure 10).

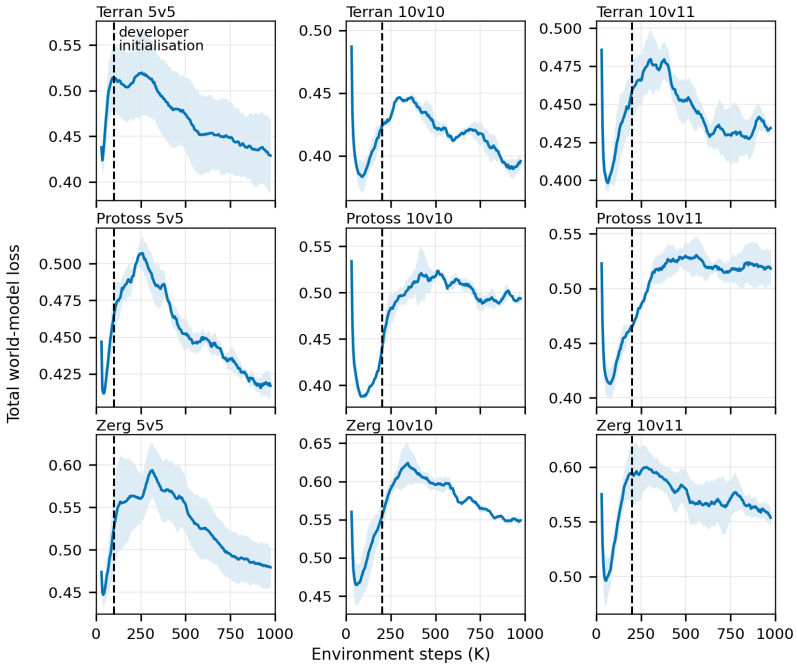


Figure 8: **Total world-model loss of MARIE on SMACv2.** Mean over seeds with ± 1 s.e. bands. Dashed lines mark the developer initialisation points. All arms are identical before the branch, so only MARIE is shown.

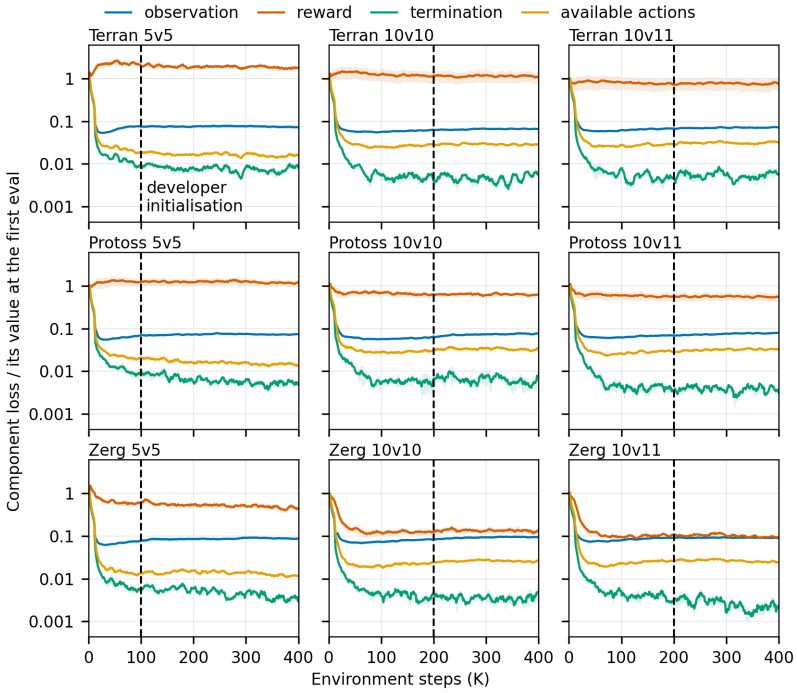


Figure 9: **World-model loss terms of MARIE on SMACv2**, each divided by its value at the first evaluation (log scale); mean over seeds with ± 1 s.e. bands. Dashed lines mark the developer initialisation points.

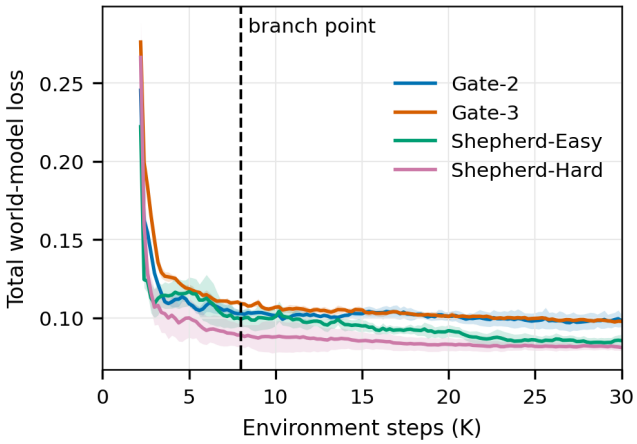


Figure 10: **Total world-model loss of SeqWM on MQE** over the first 30K environment steps, mean over three seeds with ± 1 s.e. bands. Curves start after the 2K-step random warm-up. The dashed line marks the branch point.

G COMPUTE

SMACv2 experiments ran on NVIDIA B200 GPUs and MQE experiments on NVIDIA RTX 6000 PRO Blackwell GPUs, one GPU per run. Table 7 gives the approximate cost of one run. The difference between the backbone and the PROWL variants is the developer; the fidelity gate itself adds under 1% on SMACv2 and about 1% on MQE (Dual vs. PROWL-2). The reported SMACv2 runs used about $17{,}000$ GPU-hours and the reported MQE runs about 750.

Table 7: Approximate GPU-hours per run: SMACv2 at 1M steps (B200), MQE at 1.5M steps (RTX 6000).

Method	SMACv2	MQE
Backbone (MARIE / SeqWM)	≈ 100	≈ 13
Repair	≈ 200	≈ 18
Dual	≈ 200	≈ 21
PROWL-2	≈ 200	≈ 21