TRUST BEFORE USE: WHEN DO WORLD-MODEL QUALITY METRICS PREDICT CONTROL?

Anonymous authors
Paper under double-blind review

ABSTRACT

World models are being scaled to billions of parameters and fine-tuned on robot data in the hope of unlocking new skills, yet it remains unclear which properties of a world model make it useful for downstream control. Practitioners typically judge world models by prediction loss or video fidelity, and learn whether a model is useful only after training and evaluating a policy, a slow and costly loop. We ask when world-model quality metrics predict control. We find that prediction loss and video fidelity can mislead: a world model with lower prediction loss can yield a worse policy, and a world-action model with far worse video fidelity can control without a detectable loss in success. We study the expected signal-to-noise ratio (ESNR) of policy gradients computed through a world model, which measures the gradient signal the model offers to policy extraction and can be computed during training without a trained policy or environment interaction. Across policy-extraction methods and world-model architectures on DMControl and Meta-World, ESNR tracks downstream performance more closely than prediction loss, and it tracks the planning performance of large pretrained world models at a fraction of the evaluation cost. On a 6B-parameter world-action model trained by behavior cloning, a gradient signal-to-noise ratio of the action loss orders the four model variants exactly as closed-loop success does, whereas LPIPS and SSIM rank the most successful variant last. Visualizations and code will be available at https://trust-before-use.github.io.

1 INTRODUCTION

Large pretrained world models have delivered promising results for robot control (Assran et al. (2025), NVIDIA et al. (2025)). Presently, the most common training recipe involves pretraining a world model on a large corpus of unstructured data, which gives the model basic understanding of world dynamics, and then finetuning it on a smaller state action dataset to embed knowledge of how action influences state. With an increased interest in improving the efficacy of such models without expending valuable resources, many works have sought to understand what makes a better world model from the perspective of representation (Nair et al. (2022), Xiao et al. (2022), Assran et al. (2025))

However, the only way of evaluating the quality of such models with respect to downstream policy performance is to finetune it until convergence and then execute the policy in simulation or a real environment, often requiring days or weeks of training and evaluation. Despite being the de facto procedure in evaluating world models, this costly approach limits the rapid testing of alternative architectures and subsequent policy extraction methods. We use *policy extraction* to mean algorithms that utilize world model to drive executable control policy, e.g. online planning (CEM/MPPI) or policy-gradient methods (zeroth-/first-order).

Towards this end, we propose Expected Signal-to-Noise Ratio (ESNR) as a training-time metric to identify the *potential* downstream performance of world models. ESNR requires neither a trained policy nor environment rollouts, drastically reducing the wall-clock time and compute requirements relative to standard robot evaluation. In this paper, we highlight key empirical properties of ESNR: (1) **Policy extraction selection.** ESNR can guide the choice of policy extraction method for a given world model. (2) **Architecture ranking.** ESNR discriminates between world model architectures, providing a proxy ranking for expected policy performance. (3) **Scaling.** ESNR remains predictive

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.

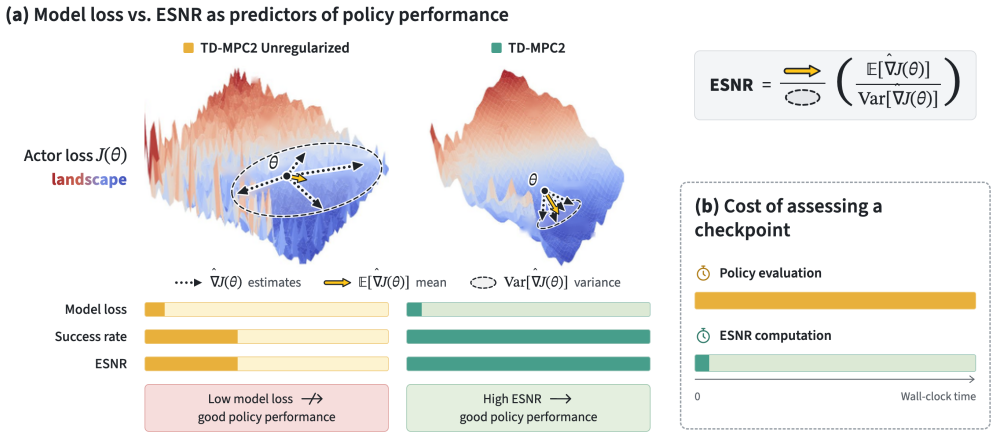


Figure 1: Overview. We find that different world model architectures can reach similar loss values under the same loss function, yet these values do not predict downstream policy performance. Instead, the Expected Signal-to-Noise Ratio (ESNR) of actor gradients $\nabla J(\theta)$ correlates strongly with final performance. ESNR can be computed quickly during training, orders of magnitude faster than full model training or policy evaluation, and captures the smoothness of the optimization landscape, providing an efficient surrogate for policy learnability.

of planning performance for large pretrained world models, at a fraction of the evaluation cost. (4) **World-action models.** For a world-action model trained by behavior cloning, where no return is available, the signal-to-noise ratio of the action-loss gradient orders the model variants as closed-loop success does, while LPIPS and SSIM rank the most successful variant last. This allows researchers to avoid excruciatingly long cycle times and rapidly iterate across different model architectures and converge to the best one.

Through this paper, we build concrete evidence of ESNR efficacy as a *downstream policy performance metric* on a variety of vision-based world model architectures, both traditional and recent SoTA large models. Additionally, we study the most common policy extraction methods, Zeroth-order policy gradients, First-order policy gradients, online planning, on a variety of continuous control tasks. Finally, we scale our experiments to **4** pre-trained world models representations – *ResNet, Dino, R3M* encoder world models (He et al. (2015), Nair et al. (2022), Zhou et al. (2025)) and *VJEPA2* (Assran et al. (2025)) – demonstrating ESNR's practical utility when designing large foundation world models *in the wild*. We further study a 6B-parameter world-action model trained by behavior cloning, for which there is no return to differentiate; there, we use the model's own training objective, the action loss, as a surrogate objective and compute the signal-to-noise ratio of its gradient.

2 Related Work

World models for robotics have advanced rapidly, building on a long line of model-based control and representation learning. Early works like PILCO showed that learned dynamics can yield highly sample-effcient policy learning on real robots (Deisenroth & Rasmussen (2011), while PETS popularized probabilistic ensembles with MPC for robust control (Chua et al. (2018)). In parallel, the "World Models" framework of Ha & Schmidhuber (2018) leveraged the generative capabilties of such models and demonstrated the idea of learning policies "in imagination", leading to latent-dynamics works like PlaNet and Dreamer, which optimize policies entirely in latent space by predicting state reward/value and training the policy in an RL paradigm (Hafner et al. (2019), Hafner et al. (2020)). Complementarily, control-centric methods like TD-MPC combined learned latent dynamics models with online trajectory optimization at test time (Hansen et al. (2022)).

Recent work scales this vision: Dino-WM combines a pretrained DinoV2 encoder with a forward dynamics predictor and performs online planning via a goal-reaching (Zhou et al. (2025)). Large foundation world models like VJEPA2 (Assran et al. (2025)) and Cosmos (NVIDIA et al. (2025)) scale the model and dataset size further to learn more general representations. Orthogonal to *how* models are used, a parallel line studies *what* representations help control: PVR vs. training-from-

scratch comparisons Hansen et al. (2023b) and PVR within MBRL Schneider et al. (2025), along with innovations in general-purpose encoders such as MVP, V-JEPA, and R3M (Xiao et al. (2022), Bardes et al. (2024), R3M (Nair et al. (2022)). Despite this progress, the model properties that actually enable downstream policy learning, beyond raw prediction quality, remains underexplored, motivating our focus on *policy awareness*.

There have been hints of works in this direction. Zhang et al. (2023) analyze policy gradients and link failure modes to exploding gradient variance arising from a lack of objective landscape regularization. SimbaV2 introduces regularization strategies to help improve policy performance in an RL setting, emphasizing the importance of regularization for policy gradient methods (Lee et al. (2025)). Parmas et al. (2023) utilized the exptected signal-to-noise ratio (ESNR) to assess the quality of gradient estimators, and PWM used ESNR to show that regularized world models yield more reliable first-order policy gradients, guiding hyperparameter choices (Georgiev et al. (2025)). However, prior work has not established a single, practically useful metric for predicting downstream policy performance across heterogeneous world model architectures and policy extraction methods. In this work, we provide evidence that the policy-gradient ESNR of a pretrained world model can serve as such a metric. This focus on mechanistic properties that predict downstream success echoes broader trends in ML as a whole. For example, there exists a line of work relating generalization to loss-landscape geometry and shape (Keskar et al. (2017), Foret et al. (2021)), and scaling-law analyses that connect model and training-time properties to task performance (Kaplan et al. (2020), Hoffmann et al. (2022), Alabdulmohsin et al. (2022)). Our contribution brings a similar lens to world models for robotics: using ESNR to *anticipate* policy learnability before policy training.

3 Background

3.1 World Models and Policy Extraction

A world model is a learned surrogate T_M for the environment's transition kernel T_e, typically an encoder $z_t = E_\phi(s_t)$, latent dynamics $z_{t+1} = F_\phi(z_t, a_t)$, an optional decoder $\hat{s}_{t+1} = D_\phi(z_{t+1})$, and reward or value heads. *Reconstruction-based* models (e.g., DreamerV3) predict the next observation and *reconstruction-less* models (e.g., TD-MPC2) the next latent state:

$$\mathcal{L}_{\text{rec}} = \mathbb{E}_{\mathcal{D}}\Big[\sum_{t=0}^{H} \ell\big(D(\hat{z}_{t+1}), s_{t+1}\big) + \ell_{\text{task}}\Big], \quad \mathcal{L}_{\text{no rec}} = \mathbb{E}_{\mathcal{D}}\Big[\sum_{t=0}^{H} \ell\big(\hat{z}_{t+1}, E(s_{t+1})\big) + \ell_{\text{task}}\Big], \tag{1}$$

where $\hat{z}_{t+1} = F(E(s_t), a_t)$, $\mathcal{D}$ is a dataset of sub-trajectories, ℓ is a distance such as the MSE, and ℓ_{task} collects task terms such as reward prediction. We call either objective the *prediction loss* $\mathcal{L}_{\text{pred}}$.

Policy extraction maximizes the model return $J_M(\theta) = \mathbb{E}_{\pi_\theta, M}[\sum_t \gamma^t r(z_t, a_t)]$ over a policy $\pi_\theta(a_t \mid z_t)$ or an action sequence $a_{0:H}$. Gradient methods average N single-sample estimates $\hat{g}_n$ of one of two kinds (Sutton et al., 1999; Heess et al., 2015): zeroth-order (ZoG, REINFORCE), $\hat{g}^{[0]} = R(\tau)\nabla_\theta \log p_\theta(\tau)$, and first-order (FoG, pathwise), $\hat{g}^{[1]} = \nabla_\theta R(\tau_\theta(\xi))$, which differentiates through the model with reparameterized noise ξ; $R(\tau)$ is the discounted return of an imagined rollout τ. Online planners such as CEM and MPPI (Rubinstein & Kroese, 2004; Williams et al., 2015) refit a Gaussian proposal over $a_{0:H}$ to high-return samples (CEM moves its mean to the mean of the elite samples); we treat this update as a pseudo-gradient step $a_{0:H} \leftarrow a_{0:H} + \alpha\, \hat{g}^{[\text{CEM}]}$ (Algorithm 3).

3.2 Model Error Versus Gradient Quality

Why can a lower prediction loss yield a worse policy? Let V_e^π and V_M^π be the value functions of a policy π in the environment and in the model, $J_e(\pi)$ and $J_M(\pi)$ their means over initial states, and let rewards lie in $[0, R_{\max}]$. For brevity the model uses the true reward (a reward error ϵ_r adds $\epsilon_r/(1-\gamma)$).

Lemma 3.1 (Simulation lemma; Kearns & Singh, 2002; follows from Agarwal et al., 2022, Lemma 2.2)**.** *Let $\epsilon_{\text{TV}} = \max_{s,a} \text{TV}\big(T_e(\cdot \mid s,a), T_M(\cdot \mid s,a)\big)$, where $\text{TV}(P,Q) = \sup_A |P(A) - Q(A)|$. For every fixed policy π, $\|V_e^\pi - V_M^\pi\|_\infty \le \gamma R_{\max}\, \epsilon_{\text{TV}}/(1-\gamma)^2$, hence so is $|J_e(\pi) - J_M(\pi)|$. If instead $W_1\big(T_e(\cdot \mid s,a), T_M(\cdot \mid s,a)\big) \le \Delta$, the reward is K_R-Lipschitz in the state and T_e or T_M is $\bar{K}$-Lipschitz with $\gamma\bar{K} < 1$, then $\|V_e^\pi - V_M^\pi\|_\infty \le \gamma K_R \Delta / \big((1-\gamma)(1-\gamma\bar{K})\big)$ (Asadi et al., 2018, Thm. 2).*

The TV bound scales with the value range $R_{\max}/(1-\gamma)$ (the ℓ_1 form of Agarwal et al. (2022) is twice as large); in the W_1 form, the constant involves the Lipschitz constant of V^π. Either way the lemma bounds the *value* of a *fixed* policy, not its gradient. Applied to an optimal policy $\pi^\star \in \arg\max_\pi J_e(\pi)$ and to the policy $\hat{\pi}$ extracted from M, it gives, with the maximum in the optimization-gap term ranging over all policies,

$$J_e(\pi^\star) - J_e(\hat{\pi}) \;\le\; \underbrace{\frac{2\gamma R_{\max}\,\epsilon_{\mathrm{TV}}}{(1-\gamma)^2}}_{\text{model error}} + \underbrace{\max_\pi J_M(\pi) - J_M(\hat{\pi})}_{\text{optimization gap in } M}. \tag{2}$$

Prediction loss is at best a loose proxy for the first term, and loosely: it averages a surrogate error over the data, whereas ϵ_{TV} is a worst case. The second term, how well policy extraction optimizes J_M, is not controlled by model error, and the two can disagree, a form of the objective mismatch of Lambert et al. (2020). ESNR attempts to bridge this gap: it measures the quality of the gradient signal that the model provides to policy extraction, which the model-error term does not capture.

3.3 Methods

We evaluate the proposed method with representative methods for each paradigm. We select DreamerV3 (reconstruction-based) (Hafner et al. (2020)), TD-MPC2 (reconstruction-less) (Hansen et al. (2023a)), and PWM (Georgiev et al. (2025)) (recontruction-less + First order Gradients), three proven algorithms in their respective training strategies. We also add one additional variant of TD-MPC2, substituting LayerNorm and ReLU activation with Mish (Misra (2020)) and SimNorm (Lavoie et al. (2022)), to study how decreasing levels of world-model regularization affect downstream policy performance and SNR.

Table 1: Overview of world models that we consider.

Model / Encoder	π-extraction
RL World Models	
• TD-MPC2	Online planning
• TD-MPC2 *basic*	Online planning
• DreamerV3	Zeroth-order gradients
• PWM	First-order gradients
Large Pretrained World Models	
• ResNet18	Online planning
• R3M	Online planning
• DINO	Online planning
• VJEPA2	Online planning
World-Action Models	
• Fast-WAM, S=5 (a)	Action expert (BC)
• Fast-WAM, S=1 (b)	Action expert (BC)
• Fast-WAM, S=10 (c)	Action expert (BC)
• Fast-WAM, blur (f)	Action expert (BC)

These world models are designed for the RL setting, where policy extraction is based on the presence of rewards. We then scale up our experiments to large-scale world models with encoders pre-trained on large corpus of data: ResNet, R3M, DinoV2, VJEPA-2 world models ((Russakovsky et al. (2015)), Nair et al. (2022), Oquab et al. (2023), Assran et al. (2025)). Finally, we study a world-action model, Fast-WAM (Yuan et al., 2026), whose action expert is trained by behavior cloning jointly with a video generator; we train four variants that differ only in the video training objective (Section 5). We present an overview of all methods and their respective policy extraction method in Table 1.

4 ESNR as a World-Model Quality Metric

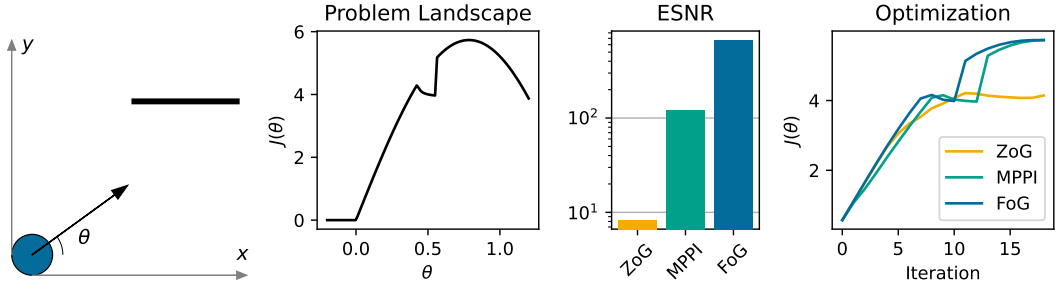


Figure 2: Toy ESNR example. In a ball-shooting task with a wall-induced discontinuity, ESNR over $\theta \in [-0.2, 1.2]$ ranks gradient estimators and predicts optimization speed: higher ESNR $\Rightarrow$ faster ascent.

For a world model to be useful for robot control, it must be (1) accurate and (2) induce a good optimization landscape for learning a policy, the two terms of Eq. 2. The former has been well studied in the video prediction model community where it is common to use PSNR, SSIM and LPIPS (Wang et al. (2004), Zhang et al. (2018)). In this work, we focus on the latter, which prediction loss does not measure.

ESNR was introduced by Parmas et al. (2023) to compare policy-gradient estimators; we use it, computed on action-sequence gradients, as a diagnostic of the world model that produces those gradients.

$$ESNR = \mathbb{E}_{o\sim\mathcal{O}}\left[\frac{\mathbb{E}_{a\sim\pi_\theta(\cdot|o)}[\hat{\nabla}^{[*]}_\theta J(\theta)]^2}{\mathbb{V}_{a\sim\pi_\theta(\cdot|o)}[\hat{\nabla}^{[*]}_\theta J(\theta)]}\right] \tag{3}$$

If $\hat{g}_1$ and $\hat{g}_2$ are unbiased stochastic gradient estimators of $\nabla_\theta J(\theta)$ and $\text{Var}(\hat{g_1}) < \text{Var}(\hat{g_2})$, then optimization with $\hat{g_1}$ is expected to converge faster. However, many modern methods such as DreamerV3 introduce gradient estimator bias via world-model surrogates as a learning signal for policy learning (Hafner et al. (2023)). ESNR overcomes this by instead computing the ratio between the signal (size of gradients) to noise (variance of gradients). In general, given two models have similar inductive bias, the one with a higher ESNR will produce *better* gradients. We build intuition of ESNR with a pedagogical task: a projectile is thrown forward with the goal of maximizing distance traveled in the presence of a object, which introduces discontinuities in the problem landscape (Figure 2). Starting from $\theta = 0$, we maximize $J(\theta)$ by gradient ascent and find that gradient estimators with higher ESNRs converge faster.

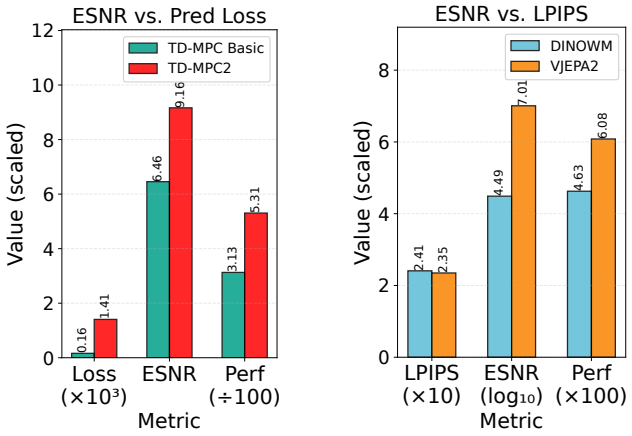
We propose using **ESNR as a test-time metric** to *predict* downstream policy performance. ESNR is computable during world model pretraining amd requires no trained policy or environment interaction. We believe this property is important to scaling up to world models with billions of parameters: an *a priori* metric enables rapid iteration across different world model architectures, allowing practitioners to identify high-performing designs more quickly. In Code 1 (Appendix B) we provide a reference implementation that can be added to offline world model training. In this implementation, we compute the ESNR with respect to the action trajectory gradients rather than policy parameter gradients. This allows us to compare ESNR across parameteric policy extraction methods (FoG, ZoG) and non-parameteric policy extraction methods (online planning).

5 Experiments

In this section, we identify that prediction loss, which is prevalently used in the community to gauge whether a world model is ready for robot deployment, can be misleading. We posit that ESNR can serve as a viable alternative: we study ESNR's ability to (i) identify the best policy learning method (ii) rank world model architectures, and (iii) test whether ESNR remains predictive as we scale up to large pretrained world models; (iv) we then ask which model-quality metrics track control in a large world-action model trained with a behavior-cloning objective. For experiments 1–3, we utilize the same ESNR definition outlined in Code 1. Experiments 1 and 2 share the following setup; experiments 3 and 4 describe their own. We evaluate the RL methods (refer to Table 1) on 12 tasks spanning 2 domains: DMControl (Tassa et al. (2018)) and Meta-World (Yu et al. (2020)). Combined, they comprise of a variety of object manipulation and locomotion tasks. To train our models, we use visual observations from both suites, and partial-propioceptive data in MetaWorld. Learning a policy is heavily dependent on imagined rollouts of the world model, which can qualitatively change based on the data distribution. Therefore, we use an offline training regime for these experiments: we first collect a fixed dataset for every task using TD-MPC2; we then pre-train *all* world models on this fixed dataset; lastly, we train policies or perform online planning exclusively with the pre-trained world model. This last stage can be termed *policy extraction*. Similar to recent large world-model such as Assran et al. (2025), the only interaction with the true environment is during evaluation which is subsequently how we assess performance on all experiments.

Why is prediction loss not enough?

Prediction loss alone can be misleading as a proxy for policy performance, especially when the world-model training objective is not aligned with policy learning. Our results in Figure 3 illustrate



Setting	Metric	Corr. w/ perf.
RL	ESNR	+0.57
	Pred. loss	−0.52
Planning	ESNR	+1.00
	LPIPS	−0.33

Figure 3: ESNR vs. baselines and correlation summary. Left: ESNR vs. prediction loss (left) and ESNR vs. LPIPS (right) across tasks. **Right:** Spearman Correlation of ESNR and standard model-quality metrics with downstream performance, split into *RL* (policy-learning tasks) and *Planning* (planning with a frozen world model).

this: • TD-MPC2 *basic* attains lower prediction loss than • TD-MPC2, yet yields worse downstream performance on all MetaWorld tasks. • TD-MPC2 *basic* uses weaker regularization than • TD-MPC2 and consequently induces a sharper, more irregular policy objective landscape. We can parallel this observation to the ceiling-bounce toy problem in Figure 7, where faithfully modeling all discontinuities does not necessarily aid optimization. In both cases, prediction loss is agnostic to how learnable the resulting policy objective is, whereas ESNR correctly ranks the two world models in line with their policy performance. We further evaluate ESNR on larger world models (• DINO-WM and • VJEPA2) across 3 tabletop manipulation tasks (see further experiment details in Appendix C). A standard approach is to assess world-model quality using LPIPS between predicted and ground-truth representations. However, our results in Figure 3 show that LPIPS fails to reliably distinguish models that yield strong downstream policy performances, whereas ESNR identifies the world model that produces the best policy. We attribute this to ESNR being, by design, a policy-centric metric: it not only measures how well a model predicts its representations, but also how conducive those representations are to strong downstream policy performance.

1. **Can ESNR be used to determine which policy extraction method is best?**

Our results indicate that the ESNR strongly correlates with the episode returns achieved by different policy extraction methods. For this experiment, we fix a trained TD-MPC2 world model and evaluate three policy extraction paradigms: • MPC with Cross-Entropy Method • FoG and • ZoG. For each method, we extract the policy at a TD-MPC2 world model pretrained at 200K steps; Appendix D discusses when during training ESNR can be read consistently. Figure 4 and Table 2 show that ESNR strongly correlates with downstream performance. Further aggregating all task performances reveals a global correlation between ESNR and policy performance. FoG methods utilize ground truth gradients from a differentiable objective, so it is unsurprising that • FoG yields higher ESNR and returns than • ZoGs: the gradients will have lower variance and greater expected norm when the objective is well defined. The surprising result is that CEM consistently attains both the highest ESNR and policy performance. However, the fundamental tradeoff here is that learning-based approaches incur high upfront training cost but have low test-time cost, while online planning has low upfront cost but high test-time cost that scales with planning horizon and number of sampled trajectories.

Experiment	Spearman
$\leftrightarrow \pi$ methods	0.79
$\leftrightarrow$ WM archs	0.80

Table 2: ESNR–performance correlations across axes. Spearman correlations between ESNR and downstream performance when varying π-extraction methods and world model architectures.

2. **Can ESNR be used to compare the downstream performance of world model architectures?**

To assess ESNR's predictive power across different world model architectures, we train three world model architectures (• TD-MPC2 *basic* • TD-MPC2 • DreamerV3) for 200K gradient updates (see

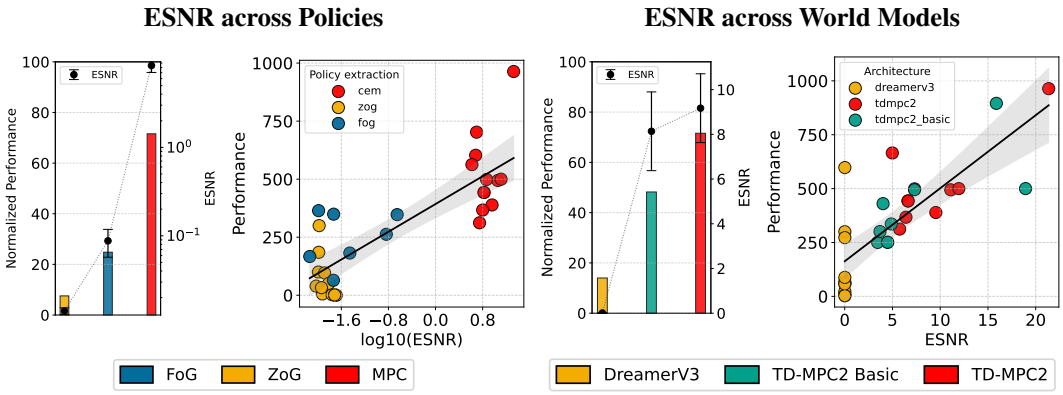
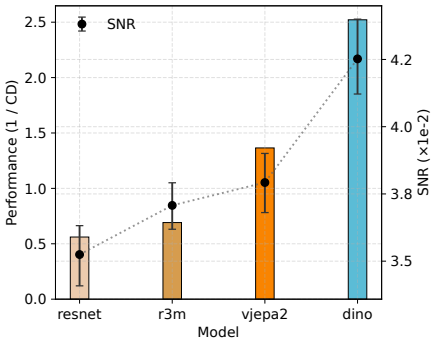


Figure 4: ESNR and policy-extraction performance. Left: bar and scatter plots show the ESNR–performance relationship across policy-extraction methods. **Right:** the same relationship across world-model architectures. ESNR error bars are 95% bootstrap CIs; bar-chart performance is normalized and averaged over all tasks.



Method	ESNR time (hrs)	Eval time (hrs)
ResNet	0.003	0.70
R3M	0.003	2.21
DinoWM	0.027	0.72
V-JEPA2	0.358	4.58

Figure 5: ESNR & Performance of Large Pretrained Models. Left: Performance $1/\mathrm{CD}$ (lower CD is better) of different large pretrained models. **Right**: Wall time to evaluate ESNR vs. evaluation.

Appendix D) and evaluate policies as prescribed by each method. On average, ESNR aligns with the episode returns across architectures (see Figure 4, Table 2): • DreamerV3 underperforms on most tasks and shows the lowest ESNR. • TD-MPC2 architecture, which features stronger regularization compared to • TD-MPC2 *basic*, achieves higher returns and correspondibly higher ESNR. The world model substantially shapes the optimization landscape seen by the policy (Figure 1). Prior work linked objective landscape properties with RL performance (Lee & Yoon (2025), Xing et al. (2025)). We extend this understanding with a policy-centric lens: ESNR quantifies the effective signal avaliable to the policy under a given world model. Our experiments also underscore the importance of objective landscapes induced by world models: the performance gap between • TD-MPC2 and • TD-MPC2 *basic* (similar models but varying in regularization) suggests that the world model, not the policy learning method, is often the primary bottleneck.

3. **Does ESNR still serve as a reliable indicator in large vision-based world models with billions of parameters?**

VJEPA2 Assran et al. (2025) and Cosmos NVIDIA et al. (2025) underlined the drastically slow action execution times required to perform online planning. Without good evaluation metrics, it will be difficult to scale and converge to the right architecture. We probe ESNR capabilities in predicting the performance of such large models. We finetune 4 pretrained world models (• ResNet • R3M • DINO-WM • VJEPA2) on an offline dataset comprising of observation-action transitions. We then perform online planning with all models using MPC with CEM optimization process on the SoftGym rope-manipulation task, in which a 7-DoF arm must move a deformable rope into a goal configuration (task and planner details in Appendix C). Each algorithm is allowed to step in the environment for 50 timesteps and we test across 10 different seeds. The performance is measured as the inverse of Chamfer Distance between the current state and goal state, where we take the lowest Chamfer

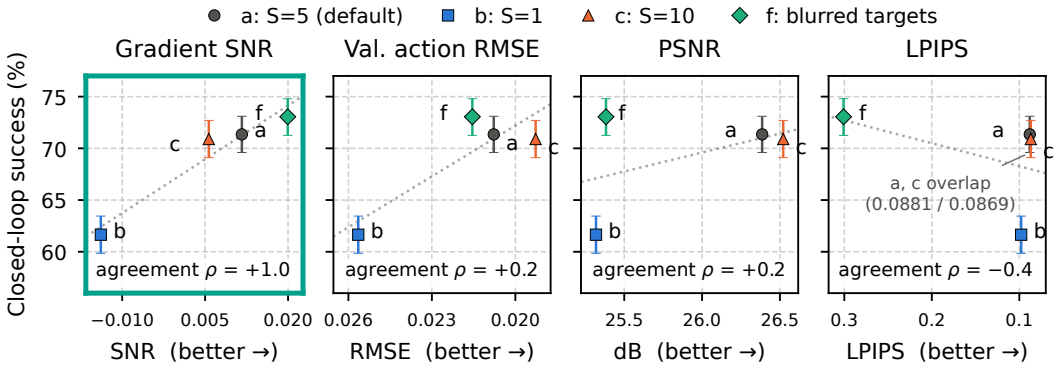


Figure 6: Fast-WAM metrics vs. closed-loop success at the final checkpoint (step 2,780). Four models that differ only in the video training objective (see legend). Success over 2,000 episodes per model with 95% bootstrap CIs; better metric values lie to the right; lines are least-squares fits. ρ: sign-adjusted Spearman correlation with success. Green frame: the gradient SNR is the only metric that orders all four models as control does.

Distance achieved in the entire environment rollout as our performance metric. Figure 5 indicates that the policy performance and ESNR are highly correlated. Furthermore, the table in Figure 5 highlights the demanding walltime requirements of inferencing and rolling out these models in simulator. On the other hand, we can calculate ESNR in a fraction of the time.

4. **Which model-quality metrics track control in a world-action model?**

World-action models (WAMs) pair a video generator with an action head and learn both from demonstrations. We ask whether the quality metrics a practitioner could compute on such a model agree with its closed-loop performance. We fine-tune Fast-WAM (Yuan et al., 2026), a 6.0B-parameter model that pairs the pretrained Wan2.2 5B video diffusion transformer (Team Wan, 2025) with a 1B-parameter action expert, on 5,500 demonstration episodes from 10 RoboTwin 2.0 tasks (Chen et al., 2025). We train four variants that differ only in how the video branch is trained. The video loss is a flow-matching loss whose noise level is sampled with a timestep shift S (Esser et al., 2024); larger S places more weight on high-noise, coarse denoising and less on fine detail. Variant • a uses the default $S = 5$, • b uses $S = 1$, • c uses $S = 10$, and • f keeps $S = 5$ but trains on Gaussian-blurred ($\sigma = 2$ px) future frames. The action head is trained with the same objective in all four variants, and all four share initialization, data, seed, a 2,780-step schedule and an identical inference procedure; we compare them at the final step (details in Appendix E). We measure control as closed-loop success over 2,000 simulated episodes per variant (10 tasks, 200 environment seeds each).

We compare the following metrics at this step (Figure 6): video fidelity of the eight predicted frames on 110 clips from 55 held-out episodes (PSNR, SSIM and LPIPS); validation action RMSE between predicted and demonstrated action chunks on 64 held-out clips, which Allshire et al. (2026) identify as the strongest predictor of success for behavior cloning; and an action-loss gradient SNR. ESNR itself (Eq. 3) is not defined for WAM-style models: they predict no reward or return to differentiate, and in Fast-WAM the video branch does not condition on actions. We therefore compute its closest analogue that is available during training, the signal-to-noise ratio $\|\mathbb{E}[g]\|^2 / \operatorname{tr}\Sigma$ of the action-loss gradient g: the squared norm of the mean gradient divided by its total variance across held-out clips, estimated from a training-time probe on 8 held-out clips following McCandlish et al. (2018). Both action-side metrics rely on the demonstrated actions of held-out episodes, whereas the video metrics require only held-out frames.

The action-loss gradient SNR ranks the four variants in the same order as their success rates ($\rho = +1.0$). Validation action RMSE and PSNR reach $+0.2$, while LPIPS and SSIM (-0.4) rank the blur variant f last, although its success is not detectably lower than the default's (73.1% vs. 71.4%). Video fidelity can therefore mislead: blurring the video targets makes the model look far worse on LPIPS and SSIM without a detectable loss in control. In video fidelity the variants rank c > a > b > f on LPIPS and SSIM, whereas their success rates rank f > a > c > b: the variant with the best-looking predictions is third in success, and the one with the worst is first. To our knowledge, this is one of the first controlled observations of objective mismatch (Lambert et al., 2020) in a large world-action

model: better video prediction does not necessarily yield better actions. For behavior cloning, the metrics that rely on held-out demonstrations perform best: both the action-loss gradient SNR and validation action RMSE place the degraded $S = 1$ variant last without flagging the blur variant. The gradient SNR also ranks the blur variant first, in line with its small lead in success, whereas validation action RMSE ranks it third.

6 Conclusion

We introduced the Expected Signal-to-Noise Ratio (ESNR) of policy gradients as a training-time metric to anticipate the downstream policy performance of world models. Unlike traditional evaluation procedures that require fully training and deploying a policy, ESNR can be computed directly during world model training with minimal overhead. Across a wide range of experiments, we showed that ESNR provides actionable insights: it signals when a world model is sufficiently pretrained to support policy learning, distinguishes between model architectures with different inductive biases, and identifies which policy extraction method—zeroth-order, first-order, or online planning—is best suited for a given model. Extending our analysis to large pretrained vision-based world models, we found that ESNR remains a reliable proxy for policy quality even when standard evaluation is prohibitively expensive.

Limitations and Future Work:

While ESNR provides a powerful and efficient proxy for downstream policy performance, it must be used with care. ESNR is not meaningful when the underlying world model is fundamentally inaccurate, as biased gradients can yield deceptively high ESNR values. This limits its utility in the very early stages of training or when modeling assumptions are severely violated. Furthermore, we have not yet evaluated ESNR extensively on large-scale world model tasks with billions of parameters. Future work should explore applying ESNR in broader large-model settings, as well as investigating complementary metrics that account for model bias in addition to gradient variance.

AI use statement

We used large language models (LLMs) as assistants to polish the writing, to search and summarize related literature, and to support parts of research execution, including drafting analysis and plotting code and generating synthetic data, as declared on the submission form. The authors designed the study and reviewed and verified all results, code, and text, and take full responsibility for the content of this paper.

Reproducibility Statement

We are committed to ensuring the reproducibility of our results. To this end, we have open-sourced the full codebase used to conduct our experiments, as well as the datasets used for training. We also release all relevant pretrained model checkpoints to facilitate verification of our results and to enable further research. Detailed hyperparameter settings, compute resources, and random seeds are documented in the accompanying code release.

REFERENCES

Alekh Agarwal, Nan Jiang, Sham M. Kakade, and Wen Sun. Reinforcement learning: Theory and algorithms, 2022. URL https://rltheorybook.github.io/. Working draft, January 31, 2022.

Ibrahim Alabdulmohsin, Behnam Neyshabur, and Xiaohua Zhai. Revisiting neural scaling laws in language and vision, 2022. URL https://arxiv.org/abs/2209.06640.

Arthur Allshire, Himanshu Gaurav Singh, Ritvik Singh, Adam Rashid, Hongsuk Choi, David McAllister, Justin Yu, Yiyuan Chen, Huang Huang, Pieter Abbeel, Xi Chen, Rocky Duan, Phillip Isola, Jitendra Malik, Fred Shentu, Guanya Shi, Philipp Wu, and Angjoo Kanazawa. Scalable behavior cloning with open data, training, and evaluation. *arXiv preprint arXiv:2606.27375*, 2026.

Kavosh Asadi, Dipendra Misra, and Michael Littman. Lipschitz continuity in model-based reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 264–273. PMLR, 2018.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba, Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning, 2025. URL https://arxiv.org/abs/2506.09985.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video, 2024. URL https://arxiv.org/abs/2404.08471.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan-ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, and Yao Mu. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

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.

Marc P Deisenroth and Carl E Rasmussen. Pilco: A model-based and data-efficient approach to policy search, Jun 2011. URL http://hdl.handle.net/10044/1/11585.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning*, pp. 12606–12633. PMLR, 2024.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization, 2021. URL https://arxiv.org/abs/2010.01412.

Ignat Georgiev, Varun Giridhar, Nicklas Hansen, and Animesh Garg. Pwm: Policy learning with multi-task world models, 2025. URL https://arxiv.org/abs/2407.02466.

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

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels, 2019. URL https://arxiv.org/abs/1811.04551.

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. *arXiv preprint arXiv:2301.04104*, 2023.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control, 2022. URL https://arxiv.org/abs/2203.04955.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023a.

Nicklas Hansen, Zhecheng Yuan, Yanjie Ze, Tongzhou Mu, Aravind Rajeswaran, Hao Su, Huazhe Xu, and Xiaolong Wang. On pre-training for visuo-motor control: Revisiting a learning-from-scratch baseline, 2023b. URL https://arxiv.org/abs/2212.05749.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

Nicolas Heess, Greg Wayne, David Silver, Timothy Lillicrap, Yuval Tassa, and Tom Erez. Learning continuous control policies by stochastic value gradients, 2015. URL https://arxiv.org/abs/1510.09142.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022. URL https://arxiv.org/abs/2203.15556.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2-3):209–232, 2002. doi: 10.1023/A:1017984413808.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima, 2017. URL https://arxiv.org/abs/1609.04836.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Nathan Lambert, Brandon Amos, Omry Yadan, and Roberto Calandra. Objective mismatch in model-based reinforcement learning. In *Learning for Dynamics and Control*, pp. 761–770. PMLR, 2020.

Samuel Lavoie, Christos Tsirigotis, Max Schwarzer, Ankit Vani, Michael Noukhovitch, Kenji Kawaguchi, and Aaron Courville. Simplicial embeddings in self-supervised learning and downstream classification, 2022. URL https://arxiv.org/abs/2204.00616.

Hojoon Lee, Youngdo Lee, Takuma Seno, Donghu Kim, Peter Stone, and Jaegul Choo. Hyperspherical normalization for scalable deep reinforcement learning, 2025. URL https://arxiv.org/abs/2502.15280.

Hyun Kyu Lee and Sung Whan Yoon. Flat reward in policy parameter space implies robust reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=4OaO3GjP7k.

Sam McCandlish, Jared Kaplan, Dario Amodei, and OpenAI Dota Team. An empirical model of large-batch training. *arXiv preprint arXiv:1812.06162*, 2018.

Diganta Misra. Mish: A self regularized non-monotonic activation function, 2020. URL https://arxiv.org/abs/1908.08681.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation, 2022. URL https://arxiv.org/abs/2203.12601.

NVIDIA, :, Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, Daniel Dworakowski, Jiaojiao Fan, Michele Fenzi, Francesco Ferroni, Sanja Fidler, Dieter Fox, Songwei Ge, Yunhao Ge, Jinwei Gu, Siddharth Gururani, Ethan He, Jiahui Huang, Jacob Huffman, Pooya Jannaty, Jingyi Jin, Seung Wook Kim, Gergely Klár, Grace Lam, Shiyi Lan, Laura Leal-Taixe, Anqi Li, Zhaoshuo Li, Chen-Hsuan Lin, Tsung-Yi Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Arsalan Mousavian, Seungjun Nah, Sriharsha Niverty, David Page, Despoina Paschalidou, Zeeshan Patel, Lindsey Pavao, Morteza Ramezanali, Fitsum Reda, Xiaowei Ren, Vasanth Rao Naik Sabavat, Ed Schmerling, Stella Shi, Bartosz Stefaniak, Shitao Tang, Lyne Tchapmi, Przemek Tredak, Wei-Cheng Tseng, Jibin Varghese, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Xinyue Wei, Jay Zhangjie Wu, Jiashu Xu, Wei Yang, Lin Yen-Chen, Xiaohui Zeng, Yu Zeng, Jing Zhang, Qinsheng Zhang, Yuxuan Zhang, Qingqing Zhao, and Artur Zolkowski. Cosmos world foundation model platform for physical ai, 2025. URL https://arxiv.org/abs/2501.03575.

Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.

Paavo Parmas, Takuma Seno, and Yuma Aoki. Model-based reinforcement learning with scalable composite policy gradient estimators. In *International Conference on Machine Learning*, pp. 27346–27377. PMLR, 2023.

Reuven Y. Rubinstein and Dirk P. Kroese. *The Cross Entropy Method: A Unified Approach To Combinatorial Optimization, Monte-carlo Simulation (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2004. ISBN 038721240X.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge, 2015. URL https://arxiv.org/abs/1409.0575.

Moritz Schneider, Robert Krug, Narunas Vaskevicius, Luigi Palmieri, and Joschka Boedecker. The surprising ineffectiveness of pre-trained visual representations for model-based reinforcement learning, 2025. URL https://arxiv.org/abs/2411.10175.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In S. Solla, T. Leen, and K. Müller (eds.), *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999. URL https://proceedings.neurips.cc/paper_files/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf.

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, 2018. URL https://arxiv.org/abs/1801.00690.

Team Wan. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. doi: 10.1109/TIP.2003.819861.

Grady Williams, Andrew Aldrich, and Evangelos Theodorou. Model predictive path integral control using covariance variable importance sampling, 2015. URL https://arxiv.org/abs/1509.01149.

Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control, 2022. URL https://arxiv.org/abs/2203.06173.

Eliot Xing, Vernon Luk, and Jean Oh. Stabilizing reinforcement learning in differentiable multiphysics simulation, 2025. URL https://arxiv.org/abs/2412.12089.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pp. 1094–1100. PMLR, 2020.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-WAM: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric, 2018. URL https://arxiv.org/abs/1801.03924.

Shenao Zhang, Boyi Liu, Zhaoran Wang, and Tuo Zhao. Model-based reparameterization policy gradient methods: Theory and practical algorithms, 2023. URL https://arxiv.org/abs/2310.19927.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. Dino-wm: World models on pre-trained visual features enable zero-shot planning, 2025. URL https://arxiv.org/abs/2411.04983.

A TOY PROBLEM DETAILS

This section provides more details on the ceiling-bounce toy example used to showcase ESNR and its effects on stochastic optimization in Section 4. We constructed a simple problem of a point mass (ball) being launched from the origin $(0, 0)$ with velocity v at initial angle θ. The goal is to maximize the horizontal distance traveled before the projectile lands on the ground ($y = 0$). The dynamics without contact are given by

$$x(t) = v\cos(\theta)t \qquad\qquad y(t) = v\sin(\theta)t + \tfrac{1}{2}gt^2,$$

where g is the gravitational acceleration. A rigid horizontal ceiling is placed at height y_s and spans the interval $[x_1, x_2]$. If the trajectory passes entirely below the ceiling, the ball follows standard ballistic motion and lands after

$$t_{\text{ground}} = \frac{-2v\sin(\theta)}{g},$$

yielding a final horizontal distance $J(\theta) = v\cos(\theta)\, t_{\text{ground}}$.

If the projectile intersects the ceiling before reaching the ground, a contact event occurs. The contact time t_c is given by the smallest positive solution of

$$\tfrac{1}{2}gt_c^2 + v\sin(\theta)\, t_c - y_s = 0,$$

with the additional requirement that $x(t_c) \in [x_1, x_2]$ and $t_c < t_{\text{ground}}$. At impact, the vertical velocity is updated according to a restitution coefficient $e \in [0, 1]$,

$$v_y^+ = -\,e\big(v\sin(\theta) + gt_c\big),$$

while the horizontal velocity remains unchanged. The ball then continues from (x_c, y_s) with velocities $(v\cos(\theta), v_y^+)$ until it hits the ground. The post-contact flight time τ_g is obtained by solving

$$y_s + v_y^+\tau_g + \tfrac{1}{2}g\tau_g^2 = 0,$$

and the total horizontal distance in the bounce case is

$$J(\theta) = x_c + v\cos(\theta)\,\tau_g.$$

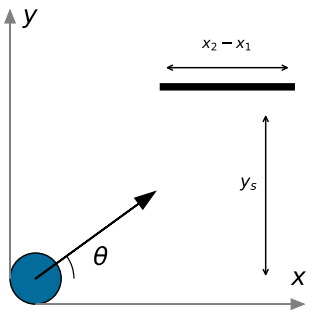


Figure 7: Pedagogical ceiling-bounce toy problem visualized.

This toy task produces a discontinuous objective landscape. Small variations in θ can switch contact on or off, leading to sharp changes in $J(\theta)$. This property makes the ceiling-bounce example a challenging benchmark for stochastic gradient estimators, which must handle both the variance and discontinuity of gradient signals. Problem parameters we used were $g = -9.81$, $x_1 = 1.0$, x_2=2.5, $y_s = 0.5$, $e = 0.5$.

For the ESNR estimation and optimization process, we used standard pytorch autograd with $N = 64$ samples and noise $\sigma = 0.03$ for the action sampling. We optimized the problem with each gradient estimator type for 20 iterations using learning rate $\alpha = 0.05$ and the Adam optimizer (Kingma & Ba, 2014).

B ESNR ALGORITHM

Code 1: ESNR pseudo-code where N is number of action samples and B is number of observation samples.

```
def compute_esnr(actions, J, grad_f):
    """
    :param actions: tensor of shape (B, act_dim)
    :param J: function with signature J(actions) -> float
    :param grad_f: function with signature grad_f(J, actions) -> grads
    """
    actions = actions.detach().requires_grad_()
    grads = grad_f(J, actions)                          # (N, B, action_dim)
    grad_mean = grads.mean(dim=0)                     # (B, action_dim)
    grad_std = grads.std(dim=0)                       # (B, action_dim)
    snrs = grad_mean**2 / (grad_std**2 + 1e-8)        # (B, action_dim)
    return snrs.mean()
```

Across all experiments, we use a fixed budget of 500 environment seeds and 100 gradient samples per initial state. All remaining hyperparameters (e.g., horizon length, elite set size) are matched to each algorithm's original settings from the respective works.

Algorithm 1 ESNR (Score / log-prob form): SNR of REINFORCE gradients induced by a world model

Inputs: world model $\mathcal{M}$ (encode, reward, next), horizon H, discount γ, MC samples S
Policy (per step): $a_t \sim \pi_\theta(\cdot \mid z_t)$ with log-prob $\log \pi_\theta(a_t \mid z_t)$ (*diag-Gaussian example:* $\nabla_{a_t} \log \pi = -(a_t - \mu_t) \oslash \sigma_t^2$)
Outer expectation set: initial observations $\mathcal{O}_0$; **Output:** ESNR
for $o_0 \in \mathcal{O}_0$ **do** ▷ approx. $\mathbb{E}_{o_0}[\cdot]$
  $z_0 \leftarrow \text{encode}_{\mathcal{M}}(o_0)$
  for $s = 1 \dots S$ **do** ▷ Monte Carlo over action sequences
    $z \leftarrow z_0$, $J \leftarrow 0$, $d \leftarrow 1$
    for $t = 1 \dots H$ **do** ▷ sample, rollout, and accumulate discounted return
      sample $a_t \sim \pi_\theta(\cdot \mid z)$; $r_t \leftarrow \text{reward}_{\mathcal{M}}(z, a_t)$
      $J \leftarrow J + d \cdot r_t$; $z \leftarrow \text{next}_{\mathcal{M}}(z, a_t)$; $d \leftarrow \gamma \cdot d$
    end for
    Score gradients via autograd: for each t, compute $\mathbf{s}_t \leftarrow \nabla_{a_t} \log \pi_\theta(a_t \mid z)$
    REINFORCE vector: $\mathbf{g}^{(s)} \leftarrow \sum_{t=1}^{H} \gamma^{t-1} r_t \mathbf{s}_t \in \mathbb{R}^{H \cdot A}$
  end for
  $\boldsymbol{\mu} \leftarrow \frac{1}{S} \sum_s \mathbf{g}^{(s)}$, $\boldsymbol{\sigma}^2 \leftarrow \frac{1}{S-1} \sum_s \left(\mathbf{g}^{(s)} - \boldsymbol{\mu}\right)^2$
  $\text{SNR}(o_0) \leftarrow \frac{1}{H \cdot A} \sum_{i=1}^{H \cdot A} \frac{\mu_i^2}{\sigma_i^2 + \varepsilon}$
end for
return $\text{ESNR} = \frac{1}{|\mathcal{O}_0|} \sum_{o_0} \text{SNR}(o_0)$

Algorithm 2 ESNR for FoG

Inputs: world model $\mathcal{M}$ (encode, reward, next), horizon H, discount γ, samples S
Outer expectation set: initial observations $\mathcal{O}_0$
Output: ESNR
for $o_0 \in \mathcal{O}_0$ **do** ▷ Approximate $\mathbb{E}_{o_0}[\cdot]$ by averaging
  $z \leftarrow \text{encode}_{\mathcal{M}}(o_0)$
  for $s = 1 \dots S$ **do** ▷ Monte Carlo over action noise
    sample $a_{1:H} \sim \mathcal{N}(0, I)$ (requires_grad)
    $J \leftarrow 0$, $d \leftarrow 1$, $z_s \leftarrow z$
    for $t = 1 \dots H$ **do** ▷ Latent rollout and discounted return
      $r_t \leftarrow \text{reward}_{\mathcal{M}}(z_s, a_t)$
      $J \leftarrow J + d \cdot r_t$
      $z_s \leftarrow \text{next}_{\mathcal{M}}(z_s, a_t)$
      $d \leftarrow \gamma \cdot d$
    end for
    $\mathbf{g}^{(s)} \leftarrow \nabla_{a_{1:H}} J \in \mathbb{R}^{H \cdot A}$
  end for
  $\boldsymbol{\mu} \leftarrow \frac{1}{S} \sum_{s=1}^{S} \mathbf{g}^{(s)}$, $\boldsymbol{\sigma}^2 \leftarrow \frac{1}{S-1} \sum_{s=1}^{S} \left(\mathbf{g}^{(s)} - \boldsymbol{\mu}\right)^2$
  $\text{SNR}(o_0) \leftarrow \frac{1}{H \cdot A} \sum_{i=1}^{H \cdot A} \frac{\mu_i^2}{\sigma_i^2 + \varepsilon}$
end for
return $\text{ESNR} = \frac{1}{|\mathcal{O}_0|} \sum_{o_0} \text{SNR}(o_0)$

Algorithm 3 ESNR (CEM / TD-MPC2): SNR of CEM-weighted action "gradients" under a world model

Inputs: world model $\mathcal{M}$ (encode, reward, next), horizon H, discount γ, pop. size N, elites E, temperature τ, grad samples S
Action stats: mean $\boldsymbol{\mu} \in \mathbb{R}^{H\times A}$ (init. $= \mathbf{0}$), diag std $\boldsymbol{\sigma} \in \mathbb{R}^{H\times A}$ (fixed)
Outer expectation set: initial observations $\mathcal{O}_0$; **Output:** ESNR
for $o_0 \in \mathcal{O}_0$ **do** ▷ approx. $\mathbb{E}_{o_0}[\cdot]$
  for $s = 1..S$ **do** ▷ Monte Carlo over action populations
    sample $\varepsilon \in \mathbb{R}^{N\times H\times A}$ i.i.d. $\mathcal{N}(0,1)$; $\mathbf{A} \leftarrow \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \varepsilon$
    $J_n \leftarrow \sum_{t=1}^{H} \gamma^{t-1}\,\text{reward}_{\mathcal{M}}(z_t^{(n)}, A_t^{(n)})$ with $z_{t+1}^{(n)} = \text{next}_{\mathcal{M}}(z_t^{(n)}, A_t^{(n)})$, $z_1^{(n)} = \text{encode}_{\mathcal{M}}(o_0)$
    Elites: keep indices $\mathcal{E}$ of top-E returns; restrict $\{J_n\}, \{\mathbf{A}^{(n)}\}$ to $n \in \mathcal{E}$
    Softmax weights: $w_n \leftarrow \text{softmax}\big(\tau\,(J_n - \max_m J_m)\big)$ over $n \in \mathcal{E}$
    CEM action "gradient" (as in code): $\mathbf{g}^{(s)} \leftarrow \sum_{n\in\mathcal{E}} w_n\,\mathbf{A}^{(n)} \in \mathbb{R}^{H\times A}$ ▷ *no* σ^{-2} factor
  end for
  stack $\mathbf{g}^{(1:S)} \in \mathbb{R}^{S\times H\times A}$; reshape to $\mathbb{R}^{S\times(HA)}$
  $\boldsymbol{\mu}_g \leftarrow \frac{1}{S}\sum_s \mathbf{g}^{(s)}, \quad \boldsymbol{\sigma}_g^2 \leftarrow \frac{1}{S-1}\sum_s \big(\mathbf{g}^{(s)} - \boldsymbol{\mu}_g\big)^2$
  $\text{SNR}(o_0) \leftarrow \frac{1}{HA}\sum_{i=1}^{HA} \frac{\mu_{g,i}^2}{\sigma_{g,i}^2 + \varepsilon}$
end for
return $\text{ESNR} = \frac{1}{|\mathcal{O}_0|}\sum_{o_0} \text{SNR}(o_0)$

C Additional Experimental Results

Policy Extraction Performance

Raw Large World Model Planning Results

Table 3: Rope task: Minimum Reduced Chamfer Distance (CD) per configuration (1–10) over 10 MPC steps, shown across all epochs.

		Configs (1–10)									
Model	Epoch	1	2	3	4	5	6	7	8	9	10
dino	1	0.09459	0.04193	0.48082	0.21772	0.11608	0.53926	0.28854	1.03296	0.10842	0.22533
dino	25	0.11339	1.46259	0.09250	0.99083	0.87808	0.32004	0.23874	1.31385	0.07392	0.26680
dino	50	0.10308	0.14979	1.05701	0.19003	0.96528	0.13502	0.30932	0.32733	0.07042	0.16416
dino	75	0.09642	0.21736	0.15945	0.38226	0.77112	0.04774	0.53940	0.19794	0.06206	0.09256
dino	100	0.07289	0.25567	0.10827	0.15935	1.03337	0.28235	0.35111	0.22349	0.06045	1.13459
dino	150	0.06513	0.89921	0.65513	0.24267	0.16757	0.11382	0.21172	0.15394	0.09686	0.12657
r3m	1	2.41500	5.88838	2.31113	2.84423	2.11243	3.12261	2.27891	2.12716	1.86786	1.38681
r3m	25	0.53223	2.17162	2.56594	1.52169	0.94276	0.23914	1.06509	1.95005	0.27470	1.91671
r3m	50	1.27117	1.42981	3.24293	0.93250	2.03880	0.70580	1.93068	1.17572	1.62150	1.46967
r3m	75	0.68322	0.41783	1.20674	0.56782	0.14875	0.09701	0.42042	0.96485	0.72479	1.62964
r3m	100	1.00865	0.21431	1.89690	0.46082	1.33459	0.20927	0.58501	1.52318	0.88556	2.01774
r3m	150	0.07504	2.50585	1.12173	1.76768	0.71625	1.32995	1.75855	1.00132	0.85643	1.31559
resnet	1	0.35964	2.08357	1.98353	0.66609	1.79421	1.43631	1.53532	2.32653	0.93529	2.65346
resnet	25	1.39610	1.92685	1.53913	1.38198	2.69483	0.52702	1.49306	2.02805	1.56138	1.72422
resnet	50	0.55921	3.55125	1.48298	1.63048	1.50783	0.39569	0.66874	0.73747	0.81317	2.07038
resnet	75	0.49132	1.67828	3.18766	1.50089	1.66228	0.89216	1.65985	2.10973	0.49975	1.99412
resnet	100	1.24596	1.59694	1.69936	1.24382	1.67177	0.23903	0.48933	2.19391	2.01874	2.34193
resnet	150	1.56192	1.89500	4.26792	1.40917	1.27476	0.38079	1.36250	4.08078	1.69061	1.86746
vjepa2	1	0.09648	0.91299	1.81233	1.35777	0.23752	0.78538	1.44923	0.89511	0.43192	1.59231
vjepa2	25	0.29945	1.83534	2.20004	1.41878	0.16598	0.10504	1.66310	1.52258	0.41726	1.05544
vjepa2	50	0.34753	1.51121	1.78945	1.33096	0.05454	0.64381	0.43591	1.55015	0.34436	3.10286
vjepa2	75	0.19621	1.54930	1.76645	1.38654	0.14202	0.72667	1.08851	1.73931	0.90187	2.27225
vjepa2	100	0.82727	2.84179	2.13063	1.99864	0.11719	0.70772	1.14918	1.71183	1.93167	2.44238
vjepa2	150	0.24537	2.70079	1.88925	1.28137	0.03945	0.40495	1.13179	1.65183	0.20860	1.45995

Rope Task Details. Rope (SoftGym). A 7-DoF arm manipulates a deformable rope to a target configuration. Planning uses MPC with CEM as the inner optimizer. Unless stated otherwise: MPC

runs for 10 iterations with prediction horizon 5 and 5 action updates per iteration; CEM draws 100 samples for 10 optimization steps, keeps the top-30 elites, and uses a variance scale of 1.

USING MEAN AS A METRIC

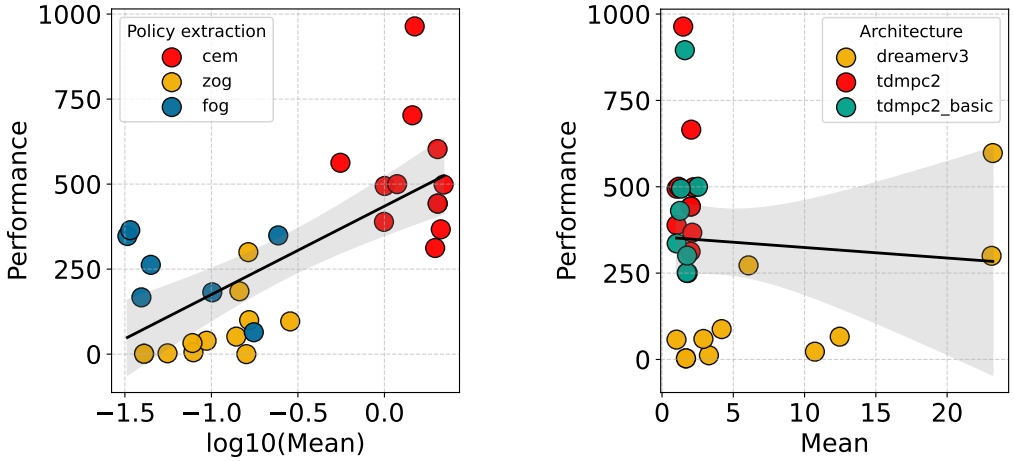


Figure 8: Mean metric vs. performance. Relationship between the mean of the policy gradient and downstream policy performance, across policy extraction method (left) and across architectures (right) across all tasks

USING VARIANCE AS A METRIC

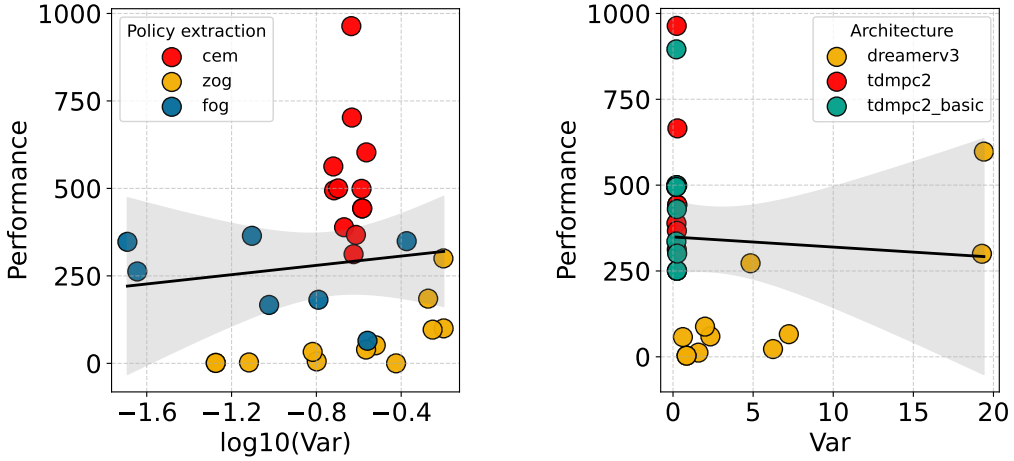


Figure 9: Variance statistic vs. performance. Relationship between the variance of the policy gradient and downstream policy performance performance, across policy extraction method (left) and across architectures (right) across all tasks

LARGE WORLD MODEL EVALUATION TASKS AND METRICS

We evaluate our models on three manipulation tasks from the ManiSkill benchmark: PickCube-v1, PushCube-v1, and PokeCube-v1. Each task requires the agent to navigate a 7-DOF end-effector (3D position, 3D orientation, and gripper state) to achieve task-specific goals. For PickCube, the agent must grasp and lift a cube; for PushCube, the agent must push a cube to a target location; and for PokeCube, the agent must poke a cube with precise contact. We measure performance

using the inverse accumulated state difference metric, computed as 1 / (accumulated L2 distance between predicted and goal end-effector states), where the accumulation occurs over all steps in an evaluation episode. This metric rewards models that rapidly converge to goal configurations, with higher values indicating better performance. The state representation consists of the end-effector's 7D pose (xyz position + quaternion orientation + gripper position), and the L2 distance is computed at each timestep and summed across the episode. All results are averaged across 4 environment seeds to ensure statistical robustness.

D When can ESNR be read during training?

Early in training, a world model can produce large but unreliable gradient estimates, and ESNR computed on such a model can be high even when the policy extracted from it is poor. We therefore restrict attention to checkpoints whose action gradients have stopped changing, which can be determined without evaluating any policy: we use the split-half cosine (SHC) between the CEM action gradients of consecutive checkpoints on the same probe states. We train • TD-MPC2 world models from state observations on five MetaWorld tasks (5 seeds, 65 checkpoints over 500K steps), evaluate every checkpoint with TD-MPC2's planner, and, for each task, keep every checkpoint from the median first crossing of SHC ≥ 0.9 onward, a threshold chosen on push (Figure 10). Within this settled region, ESNR is positively rank-correlated with return on push ($\rho = 0.48$), pick-place ($\rho = 0.65$) and sweep-into ($\rho = 0.28$); after controlling for training step, the correlations fall to 0.20, 0.51 and 0.02. On the remaining two tasks, the settled region either shows a negative relationship (assembly, $\rho = -0.32$, $n = 230$) or begins only at 450K steps, with 3 of 5 seeds settling and almost no relationship (stick-pull, $\rho = 0.07$, $n = 30$), and the gradients of • DreamerV3 never reach SHC 0.9 on any of the five tasks. Settling therefore identifies checkpoints at which ESNR is measured consistently, but it does not by itself make ESNR a reliable predictor of performance: on pick-place, early settled checkpoints reach ESNR values comparable to those of the best checkpoints while their return is still near zero.

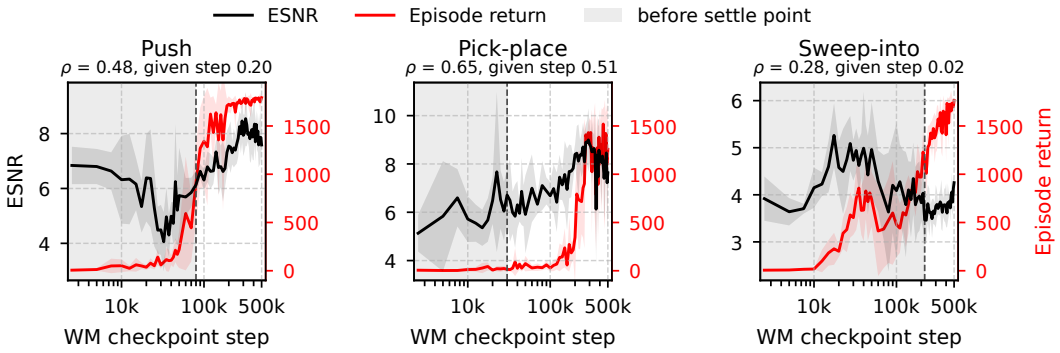


Figure 10: ESNR vs. policy performance over training (TD-MPC2). CEM ESNR (black) overlaid on the return of TD-MPC2's planner with each frozen checkpoint (red; 30 episodes), as means over five world-model seeds with ± 1 s.d. bands. Each panel shows the full training run; the shaded region precedes the settle point defined above, and ρ is computed on the checkpoints after it. ρ: Spearman correlation between ESNR and return over all settled seed–checkpoint pairs, with and without controlling for training step. State observations.

E Fast-WAM training and evaluation details

Model. We train the Joint variant of Fast-WAM (Yuan et al., 2026). Its video expert is the Wan2.2-TI2V-5B diffusion transformer (Team Wan, 2025); its action expert, an ActionDiT, is initialized by interpolating that transformer. Action tokens attend to video tokens but not vice versa, so the video branch never sees actions. The Wan2.2 VAE and the text encoder are frozen (text embeddings are precomputed), and 6.02B parameters are trained. Each action chunk holds 32 actions of 14 joint positions.

Training. All variants are fine-tuned from these pretrained weights, not trained from scratch, on 5,500 episodes (550 per task; 5,445 for training and 55 held out for validation) of ten RoboTwin 2.0 tasks: dump_bin_bigbin, hanging_mug, lift_pot, open_microwave, pick_diverse_bottles, place_can_basket,

put_object_cabinet, scan_object, stack_blocks_three and turn_switch. Training runs for 2,780 steps (two epochs) on 4 H100 GPUs with a global batch of 1,024, AdamW (betas 0.9 and 0.95, weight decay 0.01), 139 linear warm-up steps to a learning rate of 10^{-4}, cosine decay to 10^{-6}, and seed 42.

Variants. The video timestep is $t = 1000\,\varphi(u, S)$ with $\varphi(u, S) = Su/(1 + (S - 1)u)$ and $u \sim \mathcal{U}(0, 1)$, where $t = 1000$ is pure noise; $S = 1$ gives uniform timesteps, and larger S places more mass on high-noise denoising (Table 4). The per-timestep loss weight is renormalized for each S, so S also changes the loss weighting. Variant f blurs only the future-frame training targets; validation and evaluation use sharp frames. The action loss and its timestep schedule are identical across variants, as is inference: 10 denoising steps, an inference shift of 5, a fixed noise seed, and 24 executed actions per chunk before replanning.

Evaluation. We evaluate in RoboTwin 2.0 (SAPIEN simulator, dual-arm aloha-agilex robot). The domain-randomized configuration differs from the clean one only by random backgrounds, table clutter, table height and lighting. Each task and configuration has 100 fixed episode seeds shared by all variants, giving 2,000 episodes per variant. An episode succeeds if the task's own success check passes within its step budget (400–1,500 actions per task). Confidence intervals on success are 95% bootstrap intervals over paired episodes.

Metrics. Fidelity uses two windows per held-out episode, at 25% and 75% of its valid start range (110 clips); the eight future frames are generated with 20 denoising steps and scored against sharp ground truth. The high-frequency power ratio is the grayscale spectral power of the prediction above a quarter of the Nyquist frequency, divided by that of the ground truth. Validation action RMSE is the per-clip RMSE in the original action units, averaged over 64 held-out clips predicted with 10 denoising steps. The gradient SNR is probed every 250 steps on 8 fixed held-out clips, with video timesteps pinned to $\{250, 833, 950\}$ and the action timestep to 833. The two-batch estimator of McCandlish et al. (2018) ($b = 2$ clips per GPU, $B = 8$ in total) sums the gradient moments over all parameters and the three noise levels before taking the ratio. The final checkpoint uses the probe at step 2,750.

Table 4: Fast-WAM variants. All other training and inference settings are shared. Mean t: measured mean of the video training timestep under the variant's shift S.

Variant	Shift S	Future-frame targets	Mean t
a (default)	5	sharp	747.6
b	1	sharp	500.8
c	10	sharp	827.1
f	5	Gaussian blur, $\sigma = 2$ px	747.6