ARGO: ASYMMETRIC REPLAY GROUNDING WITH OPTIMISTIC PLANNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Model-based reinforcement learning can improve sample efficiency by reusing learned dynamics, but policy optimization can exploit inaccurate predictions for actions poorly represented in real experience. We introduce ARGO, an online framework that combines replay-grounded actor learning with optimistic data collection. Its central component, support-regularized local policy improvement (SRLPI), derives an auxiliary actor target from a KL-regularized objective at replay states. A learned replay-action density anchors the target, while ensemble disagreement guides conservative action ranking and controls the strength of regularization toward that density. The actor learns from this target alongside its base objective. During collection, an actor-centered planner combines predicted return with ensemble disagreement to select among direct actor samples and local perturbations. Executed transitions enter replay and inform subsequent model and actor updates. Experiments on visual continuous-control tasks show that ARGO improves sample efficiency over strong world-model baselines, with ablations examining the roles of collection and actor learning.

1 INTRODUCTION

Model-based reinforcement learning (MBRL) improves sample efficiency by learning a world model from experience and using it to evaluate actions before additional environment interaction. Recent world-model agents use learned dynamics for latent imagination (Hafner et al., 2025), model-predictive control (Hansen et al., 2024), and tree search (Schrittwieser et al., 2020), enabling strong performance in high-dimensional control. The same mechanism, however, introduces a persistent source of bias. The world model is trained on transitions stored in replay, while the agent queries the model on actions and rollouts that appear advantageous under the model itself. When these queries fall outside the support of replay, model error is propagated into optimistic value estimates and subsequently into policy updates that reflect model exploitation rather than true environment return, leading to degraded sample efficiency.

Existing methods reduce model exploitation through uncertainty-aware model use and behavior-regularized policy learning. Uncertainty penalties and pessimistic models limit reliance on unreliable synthetic transitions (Yu et al., 2020; Kidambi et al., 2020), while behavior-regularized methods combine value-based learning with a preference for actions consistent with collected data (Wu et al., 2019; 2022). In online MBRL, COPlanner (Wang et al., 2024) combines optimistic real-environment exploration with conservative synthetic rollouts. Its planner uses an uncertainty bonus for real interaction and an uncertainty penalty when selecting actions during rollout generation.

Conservative rollout selection discourages actions with high model uncertainty, but low disagreement alone does not establish replay coverage as ensemble members can share extrapolation errors. A density learned from replay provides a complementary reference grounded in actions executed in the environment. It reflects behavior collected by earlier policies and exploratory planning, including actions that the current actor may underrepresent. Such a reference can help preserve useful discoveries while discouraging policy updates toward poorly represented actions. Replay also contains suboptimal behavior, so density alone cannot identify which actions should be reinforced. The challenge is therefore to use collected experience as a reference for policy learning while accounting for the value and reliability of model-predicted improvements.

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.

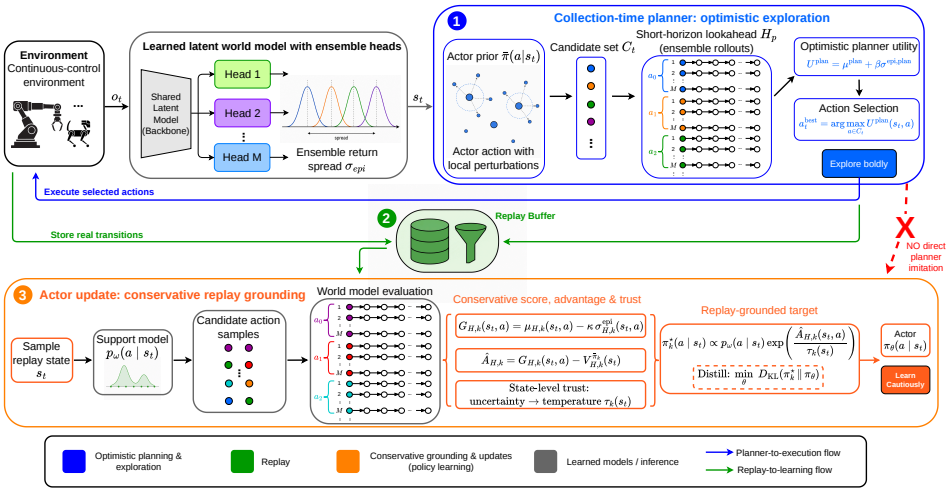


Figure 1: Overview of ARGO. The actor-centered planner ranks candidate actions by predicted return plus ensemble disagreement. Only executed transitions enter replay. At replay states, SRLPI combines learned replay-action density, conservative local returns, and an uncertainty-dependent temperature to construct an auxiliary actor target. The base actor objective is retained, and optimistic planner scores are not directly distilled into the actor.

We propose ARGO (Asymmetric Replay Grounding with Optimistic Planning), an online MBRL framework that couples replay-grounded actor learning with optimistic data collection, as illustrated in Figure 1. Its central component, support-regularized local policy improvement (SRLPI), constructs an auxiliary actor target at replay states. A learned replay-action density supplies the reference distribution, and short-horizon returns penalized by ensemble disagreement guide the reweighting of candidate actions. An uncertainty-dependent temperature controls how strongly these scores tilt the target away from the replay reference. The actor is fitted to this target alongside its base learning objective. During collection, an actor-centered planner evaluates direct actor samples and local perturbations using short-horizon predicted return plus ensemble disagreement. It executes the highest-scoring candidate, and the resulting transition enters replay. The collected experience trains the world model and replay-action density used in subsequent actor updates. The planner's optimistic scores guide interaction but are not directly used as imitation targets.

We evaluate ARGO on 13 visual DeepMind Control Suite tasks with a budget of 1M environment steps. Its aggregate mean final return is 8.2% higher than DreamerV3's (Hafner et al., 2025) and 12.5% higher than TWISTER's (Burchi & Timofte, 2025). Aggregate learning-curve analysis also shows higher normalized area under the curve than both baselines. Ablations on Quadruped Run and Cheetah Run examine the effects of collection and alternative actor-learning designs. Our contributions are threefold. First, we introduce SRLPI, a replay-grounded auxiliary actor update derived from a KL-regularized objective to mitigate model exploitation. Ensemble disagreement serves as a reliability proxy that controls both conservative action ranking and the strength of regularization toward a learned replay-action density. Second, we develop ARGO by integrating this update with actor-centered optimistic planning for data collection. Third, we demonstrate improved aggregate final return and sample efficiency on 13 visual continuous-control tasks, with ablations examining the roles of collection and actor learning.

2 Related Works

Model-based reinforcement learning uses learned dynamics for planning or policy learning, as exemplified by PETS (Chua et al., 2018), MBPO (Janner et al., 2019), and DreamerV3 (Hafner et al., 2025). Transformer-based world models include IRIS (Micheli et al., 2023), STORM (Zhang et al., 2023), and TWISTER (Burchi & Timofte, 2025), while TD-MPC and TD-MPC2 combine latent-space planning with learned value functions and policy priors (Hansen et al., 2022; 2024).

Regularized policy improvement. MPO (Abdolmaleki et al., 2018) formulates relative-entropy-constrained policy improvement, while AWR (Peng et al., 2019) and AWAC (Nair et al., 2020) use advantage-weighted regression; AWAC also studies online fine-tuning from offline data. BRAC (Wu et al., 2019) regularizes policies toward behavior, IQL (Kostrikov et al., 2022) combines in-sample value learning with weighted policy extraction, and SPOT (Wu et al., 2022) uses an explicit behavior-density model for support regularization. SRLPI builds on the established exponential-reweighting and forward-KL projection structure. It constructs an auxiliary target at replay states by combining a learned replay-action density, conservative short-horizon returns under a reference actor, and an uncertainty-dependent temperature. The density reference and local model-based score specify how this update uses the evolving online replay buffer.

Optimistic exploration. Ensemble disagreement provides intrinsic exploration signals (Pathak et al., 2019), and Plan2Explore (Sekar et al., 2020) uses latent imagination to seek informative experience. SOMBRL (Sukhija et al., 2025) optimizes a reward-plus-uncertainty objective under estimated dynamics, including in visual-control agents, and establishes regret guarantees under its modeling assumptions. ARGO uses optimism for finite-candidate real-action selection: its planner ranks actor samples and local perturbations by mean predicted return plus cross-model return disagreement. The uncertainty bonus is confined to collection-time scoring, while the base actor objective is retained and SRLPI supplies a separate conservative target.

Conservative model use. MOPO (Yu et al., 2020) penalizes uncertain model-generated rewards, MOReL (Kidambi et al., 2020) constructs a pessimistic MDP, and COMBO (Yu et al., 2021) regularizes values on model-generated data for offline learning. MOBILE (Sun et al., 2023) uses model-Bellman inconsistency to quantify uncertainty, while C-LAP (Alles et al., 2024) constrains policies through learned latent-action support. COPlanner (Wang et al., 2024) already combines optimistic real exploration with conservative synthetic rollouts, applying uncertainty-aware candidate lookahead during rollout generation. ARGO shares this asymmetric use of uncertainty but applies conservatism through auxiliary actor-target construction at replay states. Its candidate continuations follow the reference actor, and their conservative return estimates are combined with replay-action density rather than used to replan at every imagined step.

Planning and policy learning. Policy networks can guide search or learn from its results. POPLIN (Wang & Ba, 2020) combines policy-guided planning with policy distillation; MuZero and Sampled MuZero (Schrittwieser et al., 2020; Hubert et al., 2021) train on search-derived policy targets. TD-MPC and TD-MPC2 instead learn value-maximizing policy priors, and LOOP (Sikchi et al., 2021) regularizes trajectory optimization around an off-policy actor. BOOM (Zhan et al., 2025) adds a soft value-weighted alignment loss on executed planner actions stored in replay to the actor objective. TD-M(PC)^2 (Lin et al., 2025) regularizes actor samples under planner-behavior distributions stored in replay to reduce out-of-distribution value queries. These methods address planner-policy mismatch in TD-MPC-based agents that use online MPC for control. ARGO augments an imagination-trained actor with an auxiliary target constructed from learned replay-action density, conservative local returns, and an uncertainty-dependent temperature.

3 METHODOLOGY

We consider model-based actor-critic agents that learn a world model from real trajectories stored in a replay buffer. The model encodes observation and action histories into latent states and predicts subsequent states and rewards. Imagined trajectories starting from states sampled from replay are then used to train an actor to maximize predicted returns and a critic to estimate them.

Model errors can reduce sample efficiency by leading to unproductive environment interactions and reinforcing suboptimal behavior through biased actor updates. ARGO retains the base actor-critic objectives and adds two mechanisms. An actor-centered optimistic planner combines predicted returns with ensemble disagreement to select real actions for data collection. Support-regularized local policy improvement (SRLPI) adds an auxiliary actor update that combines a learned replay-action density with conservative local-return estimates. Optimism therefore guides collection, while replay support and uncertainty determine the additional actor target. The planner's selected actions

are not directly used as imitation targets. Appendix E provides pseudocode for the full ARGO training procedure and the SRLPI update.

3.1 Support-Regularized Local Policy Improvement

Let k index optimization updates. At each replay state s_t, SRLPI combines replay-action density, a conservative local score, and an uncertainty-dependent temperature in the following target:

$$\pi_k^\star(a_t \mid s_t) \propto p_\omega(a_t \mid s_t) \exp\left(\frac{\hat{A}_{H,k}(s_t, a_t)}{\tau_k(s_t)}\right), \tag{1}$$

where p_ω models replay support, $\hat{A}_{H,k}$ is a conservative local improvement signal, and $\tau_k(s_t)$ controls how aggressively the target may tilt away from replay support.

State representation and frozen reference networks. We hold the reference policy and value estimates fixed during target construction, then update the actor against the resulting target. The construction applies to any backbone that maps replay trajectories to model states s_t used for imagination and control. For example, $s_t = \{h_t, z_t\}$ combines a deterministic hidden state h_t summarizing past observations and actions with a stochastic latent representation z_t of the current observation. Let $\mathcal{D}$ denote the replay buffer of real trajectories, let $\pi_\theta(\cdot \mid s_t)$ denote the actor, and let $V_\psi(s_t)$ denote the value estimator. The actor samples actions $a_t \sim \pi_\theta(\cdot \mid s_t)$. We use the frozen reference networks

$$\bar{\pi}_k = \mathrm{sg}(\pi_{\theta_k}), \qquad \bar{V}_k = \mathrm{sg}(V_{\psi_k}), \tag{2}$$

where $\mathrm{sg}(\cdot)$ denotes stop-gradient.

Replay support model. The first ingredient in (1) is a differentiable description of which actions are actually supported by replay at a given model state. In continuous action spaces, the empirical conditional distribution in replay is atomic, so it cannot be used directly inside a KL regularizer or an importance-weighted estimator. We therefore fit a smooth conditional density model

$$p_\omega(a_t \mid s_t) \approx \hat{p}_{\mathcal{D}}(a_t \mid s_t), \qquad \mathcal{L}_{\mathrm{sup}}(\omega) = -\mathbb{E}_{(s_t,a_t)\sim\mathcal{D}}\big[\log p_\omega(a_t \mid s_t)\big]. \tag{3}$$

The role of p_ω is only to represent replay support in a differentiable form; by itself it does not decide which replayed actions are worth transferring into the actor.

Conservative local evaluation. The second component is a local model-based score. For uncertainty-aware evaluation, assume a collection of predictive models or bootstrap predictive heads $\{\hat{P}_{\phi_m}\}_{m=1}^M$ defined on the same model-state space. Under predictor m, define the H-step truncated return

$$\bar{G}^m_{H,k}(s_t, a_t) = \mathbb{E}_{\hat{P}_{\phi_m}, \bar{\pi}_k}\left[\sum_{h=0}^{H-1} \gamma^h \hat{r}^{(m)}_{t+h+1} + \gamma^H \bar{V}_k\Big(s^{(m)}_{t+H}\Big) \;\middle|\; s^{(m)}_t = s_t,\ a^{(m)}_t = a_t\right]. \tag{4}$$

When the model predicts episode continuation, future rewards and the bootstrap value in (4) are weighted by products of predicted continuation probabilities. In stochastic latent models, $\bar{G}^m_{H,k}(s_t, a_t)$ is interpreted as the model-wise expected truncated return and can be estimated by averaging a small number of rollouts within each predictor.

We aggregate model-wise predictions by their mean and epistemic disagreement,

$$\mu_{H,k}(s_t, a_t) = \frac{1}{M}\sum_{m=1}^M \bar{G}^m_{H,k}(s_t, a_t), \tag{5}$$

$$\sigma^{\mathrm{epi}}_{H,k}(s_t, a_t) = \sqrt{\frac{1}{M-1}\sum_{m=1}^M \Big(\bar{G}^m_{H,k}(s_t, a_t) - \mu_{H,k}(s_t, a_t)\Big)^2}, \tag{6}$$

and define the conservative local score

$$G_{H,k}(s_t, a_t) = \mu_{H,k}(s_t, a_t) - \kappa\, \sigma^{\mathrm{epi}}_{H,k}(s_t, a_t), \tag{7}$$

where $\kappa \geq 0$ controls the degree of conservatism. Thus, actions whose apparent gains are driven mainly by model disagreement are explicitly downweighted.

Conservative local score and advantage interpretation. The conservative score $G_{H,k}(s_t, a_t)$ estimates the value of taking root action a_t at replay state s_t, followed by the frozen actor $\bar{\pi}_k$, while penalizing ensemble disagreement. To interpret this as local policy improvement, we define the frozen-actor reference level and center the score:

$$V^{\bar{\pi}}_{H,k}(s_t) = \mathbb{E}_{b\sim\bar{\pi}_k(\cdot|s_t)}\big[G_{H,k}(s_t, b)\big], \qquad \hat{A}_{H,k}(s_t, a_t) = G_{H,k}(s_t, a_t) - V^{\bar{\pi}}_{H,k}(s_t). \tag{8}$$

By construction, $\hat{A}_{H,k}(s_t, a_t) > 0$ means that root action a_t is conservatively better than the frozen actor's average at the same replay state. Since $V^{\bar{\pi}}_{H,k}(s_t)$ is constant with respect to π, it drops from the solution $\pi^\star_k$. The advantage form is retained for its local improvement interpretation; the implementation uses $G_{H,k}$ directly, avoiding the extra actor rollouts that estimating the baseline would require.

State-dependent trust in model rankings. Equation (7) is an actionwise penalty, but a state-level correction is also needed. Even after subtracting $\kappa\sigma^{\mathrm{epi}}$, residual score gaps can be spurious when the model is broadly uncertain near s_t: a small temperature exponentiates these gaps into a concentrated target, biasing the actor toward a noise-driven candidate and shifting subsequent proposal distributions in the same direction. We therefore define

$$\Sigma_{H,k}(s_t) = \mathbb{E}_{a\sim\bar{\pi}_k(\cdot|s_t)}\big[\sigma^{\mathrm{epi}}_{H,k}(s_t, a)\big], \qquad \tau_k(s_t) = \mathrm{clip}\big(\tau_0 + \rho\,\Sigma_{H,k}(s_t),\ \tau_{\min},\ \tau_{\max}\big). \tag{9}$$

For fixed local scores, increasing $\tau_k(s_t)$ reduces score-induced deviations of the target from the replay-action density p_ω.

Statewise target and actor projection. With these ingredients in place, we define the statewise replay-regularized improvement objective that (1) previews:

$$\pi^\star_k(\cdot \mid s_t) = \arg\max_{\pi\in\Delta(\mathcal{A})} \Big\{\mathbb{E}_{a\sim\pi(\cdot|s_t)}\big[\hat{A}_{H,k}(s_t, a)\big] - \tau_k(s_t) D_{\mathrm{KL}}(\pi(\cdot \mid s_t) \,\|\, p_\omega(\cdot \mid s_t))\Big\}. \tag{10}$$

Equation (1) is the unique solution of (10); Appendix A.1 gives the short derivation. Intuitively, (10) asks for a policy that places more mass on actions with larger conservative local advantage, while paying a KL cost for drifting away from replay support. Equivalently, replacing $\hat{A}_{H,k}$ with $G_{H,k}$ in (10) yields the same optimizer, since $V^{\bar{\pi}}_{H,k}(s_t)$ is constant with respect to π. Given $\pi^\star_k$, the SRLPI auxiliary loss fits the actor to this target:

$$\mathcal{L}_{\mathrm{SRLPI}}(\theta) = \mathbb{E}_{s_t\sim\mathcal{D}}\left[D_{\mathrm{KL}}(\pi^\star_k(\cdot \mid s_t) \,\|\, \pi_\theta(\cdot \mid s_t))\right], \tag{11}$$

with no gradients propagated through $\pi^\star_k$. Using $D_{\mathrm{KL}}(p\|q) = \mathbb{E}_p[\log p - \log q]$, this is equivalently

$$\mathcal{L}_{\mathrm{SRLPI}}(\theta) = C_k - \mathbb{E}_{s_t\sim\mathcal{D}}\mathbb{E}_{a\sim\pi^\star_k(\cdot|s_t)}\big[\log \pi_\theta(a \mid s_t)\big], \tag{12}$$

where C_k is independent of θ. This form makes clear that SRLPI first constructs a replay-regularized local target and then projects that target into the actor by maximum likelihood under $\pi^\star_k$.

Continuous-action approximation. In discrete action spaces, the expectation in (12) can be evaluated by summing over actions. In continuous action spaces, direct sampling from $\pi^\star_k(\cdot \mid s_t)$ is generally intractable because the normalizing constant in (1) is unavailable in closed form. We estimate the actor-projection expectation using self-normalized importance sampling over a finite candidate set. The analytic target remains unchanged. Appendix A.2 gives the derivation.

At each replay state, we draw candidate actions from the proposal distribution ν_k, held fixed during target construction:

$$\nu_k(\cdot \mid s_t) = \alpha\, p_\omega(\cdot \mid s_t) + (1-\alpha)\,\bar{\pi}_k(\cdot \mid s_t), \tag{13}$$

with $\alpha \in (0, 1]$. The p_ω term ensures that the proposal covers replay-supported actions, while the $\bar{\pi}_k$ term improves efficiency by proposing actions near the current actor. Because these samples are drawn from ν_k rather than from the exact target $\pi^\star_k$, we reweight them to approximate expectations under (12). Given samples $a^1_t, \ldots, a^N_t \sim \nu_k(\cdot \mid s_t)$, we use the normalized weights

$$\tilde{w}_i(s_t) = \frac{p_\omega(a^i_t \mid s_t)}{\nu_k(a^i_t \mid s_t)} \exp\left(\frac{G_{H,k}(s_t, a^i_t)}{\tau_k(s_t)}\right), \qquad w_i(s_t) = \frac{\tilde{w}_i(s_t)}{\sum_{j=1}^N \tilde{w}_j(s_t)}, \tag{14}$$

which are the standard self-normalized importance weights induced by the unnormalized target density underlying (1). We then optimize the sampled projection loss

$$\widehat{\mathcal{L}}_{\mathrm{SRLPI}}(\theta) = -\mathbb{E}_{s_t\sim\mathcal{D}}\left[\sum_{i=1}^{N} w_i(s_t)\,\log \pi_\theta(a_t^i \mid s_t)\right], \tag{15}$$

with gradients stopped through $w_i(s_t)$, since $\pi_k^\star$ is treated as a fixed target during actor regression.

3.2 Planner-Augmented Replay for Targeted Support Expansion

SRLPI regularizes its auxiliary actor target toward replay-action density. We complement this update with optimistic planning to collect additional experience. The planner evaluates actions sampled from the actor and their local perturbations using short-horizon optimistic lookahead. The selected action is executed in the environment, and the resulting transition enters replay.

Structured actor-centered candidate generation. At a real interaction state mapped to model state s_t, the planner constructs a finite candidate set C_t by first including one direct actor proposal. The remaining $|C_t|-1$ candidates are drawn independently: with probability $1-\epsilon_{\mathrm{plan}}$ directly from $\bar{\pi}_k(\cdot \mid s_t)$, and with probability ϵ_{plan} from a local perturbation kernel $E_k(\cdot \mid s_t, b)$ centered at an actor anchor $b \sim \bar{\pi}_k(\cdot \mid s_t)$ (e.g., additive Gaussian perturbations in continuous control). At least one direct actor proposal is always included in C_t.

Optimistic uncertainty-aware lookahead. The planner ranks candidate root actions by predicted task return and ensemble disagreement. We evaluate each candidate using the same short-horizon local rollout template as in SRLPI. Under predictor m,

$$\bar{G}^{m,\mathrm{plan}}_{H_p,k}(s_t,a) = \mathbb{E}_{\hat{P}_{\phi^m},\bar{\pi}_k}\left[\sum_{h=0}^{H_p-1}\gamma^h \hat{r}^{(m)}_{t+h+1} + \gamma^{H_p}\bar{V}_k\left(s^{(m)}_{t+H_p}\right)\,\middle|\, s_t^{(m)} = s_t,\ a_t^{(m)} = a\right], \tag{16}$$

with the same episode-continuation weighting described for (4) when continuation is predicted. We aggregate these predictions by

$$\mu^{\mathrm{plan}}_{H_p,k}(s_t,a) = \frac{1}{M}\sum_{m=1}^{M}\bar{G}^{m,\mathrm{plan}}_{H_p,k}(s_t,a). \tag{17}$$

$$\sigma^{\mathrm{epi,plan}}_{H_p,k}(s_t,a) = \sqrt{\frac{1}{M-1}\sum_{m=1}^{M}\left(\bar{G}^{m,\mathrm{plan}}_{H_p,k}(s_t,a) - \mu^{\mathrm{plan}}_{H_p,k}(s_t,a)\right)^2}. \tag{18}$$

and define the optimistic planner utility

$$U^{\mathrm{plan}}_{H_p,k}(s_t,a) = \mu^{\mathrm{plan}}_{H_p,k}(s_t,a) + \beta\,\sigma^{\mathrm{epi,plan}}_{H_p,k}(s_t,a), \tag{19}$$

where $\beta \geq 0$ trades off predicted return and ensemble disagreement. In contrast to SRLPI, this epistemic bonus affects only which real action is tried next; it does not directly define a reusable policy target.

Execution by utility maximization over the candidate set. Given the candidate set C_t, we execute

$$a_t^{\mathrm{exec}} = \arg\max_{a\in C_t} U^{\mathrm{plan}}_{H_p,k}(s_t,a), \tag{20}$$

breaking ties in favor of direct actor samples. A perturbed candidate is selected only when its utility exceeds that of every direct actor candidate.

Replay as the filter between planning and learning. All executed transitions enter the same replay buffer and contribute to world-model learning, whether selected directly from the actor or through local perturbations. SRLPI constructs an auxiliary actor target from replay-action density and conservative model scores, while the base actor objective is retained. The actor is not trained to imitate the planner's candidate generator or optimistic utility.

Table 1: Evaluation returns on 13 visual DeepMind Control Suite tasks after 1M environment steps (mean ± sample standard deviation). ARGO uses five independent training runs per task and each baseline uses three. Bold and underline indicate the highest and second-highest mean per task. The final row averages task means. Replication details are given in Appendix F.

Task	SAC	DreamerV3	TD-MPC2	TWISTER	ARGO
Acrobot Swingup	7.6 ± 3.4	232.0 ± 59.7	$\underline{241.3 \pm 35.8}$	224.2 ± 6.0	$\mathbf{353.2 \pm 71.9}$
Cartpole Swingup Sparse	508.7 ± 441.3	746.9 ± 60.1	$\mathbf{790.0 \pm 49.3}$	738.7 ± 20.0	$\underline{753.7 \pm 3.8}$
Cheetah Run	36.6 ± 12.1	$\mathbf{834.1 \pm 81.5}$	537.3 ± 73.9	753.1 ± 16.0	$\underline{830.9 \pm 16.7}$
Finger Spin	383.3 ± 26.1	672.0 ± 95.9	$\mathbf{984.9 \pm 2.9}$	766.5 ± 158.1	$\underline{979.3 \pm 3.1}$
Finger Turn Easy	265.2 ± 112.9	$\underline{857.5 \pm 86.7}$	820.8 ± 130.1	829.2 ± 50.6	$\mathbf{924.6 \pm 36.1}$
Finger Turn Hard	165.1 ± 55.3	$\underline{807.1 \pm 114.8}$	865.6 ± 92.9	$\mathbf{962.7 \pm 4.8}$	$\underline{910.8 \pm 40.3}$
Hopper Hop	1.0 ± 1.7	231.6 ± 200.1	267.6 ± 21.7	$\underline{274.9 \pm 29.8}$	$\mathbf{325.8 \pm 16.8}$
Quadruped Run	118.1 ± 26.4	$\underline{696.8 \pm 30.9}$	283.1 ± 84.4	572.0 ± 111.8	$\mathbf{863.1 \pm 26.3}$
Quadruped Walk	99.9 ± 18.0	$\underline{886.2 \pm 55.2}$	323.5 ± 131.7	$\underline{904.8 \pm 32.4}$	$\mathbf{923.0 \pm 12.5}$
Reacher Easy	116.6 ± 5.3	957.0 ± 11.2	$\mathbf{982.2 \pm 3.6}$	$\underline{966.4 \pm 6.4}$	$\underline{976.8 \pm 5.1}$
Reacher Hard	18.8 ± 23.7	$\underline{871.6 \pm 71.4}$	$\mathbf{909.6 \pm 54.1}$	463.2 ± 191.9	$\underline{634.4 \pm 145.1}$
Walker Run	33.1 ± 3.0	$\underline{676.6 \pm 151.4}$	671.9 ± 39.2	659.6 ± 102.5	$\mathbf{759.6 \pm 4.6}$
Walker Walk	62.0 ± 13.7	$\underline{961.5 \pm 8.9}$	939.6 ± 10.1	954.0 ± 20.0	$\mathbf{969.3 \pm 5.6}$
Mean	139.7	<u>725.4</u>	662.9	697.6	**785.0**

4 Experiment

4.1 Experimental Setup

Benchmarks. We evaluate ARGO on the visual-observation DeepMind Control Suite under a fixed budget of 1M environment steps, following the evaluation protocol of recent visual world-model agents (Hafner et al., 2025; Burchi & Timofte, 2025; Hansen et al., 2024). Tasks span locomotion (*Cheetah Run*, *Hopper Hop*, *Walker Walk/Run*, *Quadruped Walk/Run*), swing-up and sparse/difficult control (*Acrobot Swingup*, *Cartpole Swingup Sparse*), reaching (*Reacher Easy/Hard*), and finger control (*Finger Spin/Turn Easy/Hard*), covering a range of exploration difficulty and control precision. ARGO uses planning for training-time data collection and the learned actor without planning at evaluation.

Baselines. We compare ARGO with **DreamerV3** (Hafner et al., 2025), **TD-MPC2** (Hansen et al., 2024), and **TWISTER** (Burchi & Timofte, 2025). We choose TWISTER as a common Transformer-based backbone for evaluating the performance and computational overhead of ARGO's collection and actor-learning mechanisms. We retain its world-model architecture and base actor-critic objectives. We also include **SAC** (Haarnoja et al., 2018) in the final-return and learning-curve comparisons. All methods use visual observations and a 1M-environment-step training budget. We use released run-level results from the official repositories for SAC, DreamerV3, and TD-MPC2. We reproduce TWISTER using its official implementation because its per-run results are not released. The comparison uses five independent training runs per task for ARGO and three per baseline.

Implementation. Appendix F describes the implementation and hyperparameters; Appendix F gives the evaluation protocol and replication details.

4.2 Experiment Results

Table 1 reports mean evaluation returns on 13 visual DeepMind Control Suite tasks after 1M environment steps. ARGO achieves a mean return of 785.0 across tasks, compared with 725.4 for DreamerV3, 697.6 for TWISTER, and 662.9 for TD-MPC2. Its mean return is 8.2% higher than DreamerV3's and 12.5% higher than TWISTER's. ARGO has the highest observed mean on 7 of 13 tasks and a higher mean than TWISTER on 12 of 13 tasks.

Figure 2 shows learning curves from the same runs. Aggregate final-return and sample-efficiency statistics, confidence intervals, and analysis details are provided in Appendix B.1.

Table 2: Mean final evaluation returns over three training runs per method on three visual DeepMind Control Suite tasks after 1M environment steps. Evaluation details are given in Appendix F.

Task	TWISTER	COPlanner	ARGO
Acrobot Swingup	224.2	236.2	347.8
Cheetah Run	753.1	772.5	831.9
Quadruped Run	572.0	612.4	869.5

Gains on exploration-sensitive and locomotion tasks. ARGO achieves the highest mean return on Acrobot Swingup, with one of its largest gains over TWISTER. On locomotion tasks, it has the highest mean on Hopper Hop, Quadruped Run, Quadruped Walk, Walker Run, and Walker Walk. DreamerV3 has a slightly higher mean on Cheetah Run (834.1 versus 830.9), while TD-MPC2 has the highest mean on Cartpole Swingup Sparse.

Performance on precision-control tasks. ARGO achieves the highest mean on Finger Turn Easy, while TWISTER leads on Finger Turn Hard (962.7 versus 910.8). On Finger Spin and Reacher Easy, ARGO's mean is within 0.6% of the highest mean, obtained by TD-MPC2. Reacher Hard remains a limitation: TD-MPC2 and DreamerV3 achieve 909.6 and 871.6, respectively, compared with 634.4 for ARGO and 463.2 for TWISTER. Learning curves for *Finger Spin* and *Reacher Easy* are provided in Appendix C.

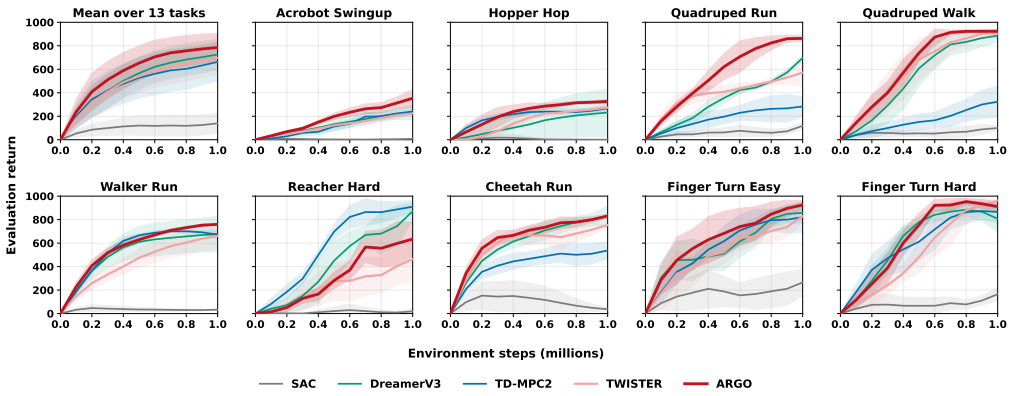


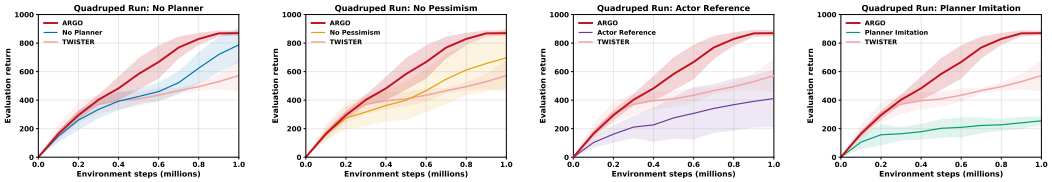
Figure 2: Learning curves on nine visual DeepMind Control Suite tasks and the mean over all 13 tasks. Task panels show mean evaluation return and one sample standard deviation across five runs for ARGO and three runs per baseline. The aggregate panel averages task means, with shading at ± 1.96 standard errors across tasks. Curves and bands are smoothed for display.

4.3 Ablations and Controlled Comparisons

Comparison with COPlanner. COPlanner (Wang et al., 2024) and ARGO both use optimistic planning for real data collection. COPlanner applies conservative planning at each step of synthetic rollout generation; SRLPI instead constructs a replay-density-regularized auxiliary actor target using conservative local-return estimates and an uncertainty-dependent temperature. SRLPI's candidate rollouts follow the frozen actor after the root action, avoiding repeated planning along imagined trajectories. For Table 2, we implement COPlanner on the TWISTER backbone and denote this adaptation as COPlanner. ARGO has a higher mean final return on each of the three tasks. Appendix F describes the adaptation and computational costs.

Component ablations. We isolate the contribution of each ARGO component on *Quadruped Run*, the task where ARGO's advantage is largest and the ablation signal is clearest.

Planner (Figure 3a). SRLPI and optimistic planning serve complementary roles: SRLPI constructs replay-regularized actor targets, while planning selects actions for data collection. The *No Planner*



(a) No Planner (b) No Pessimism (κ=0) (c) Actor Reference (d) Planner Imitation

Figure 3: Ablations on *Quadruped Run*. Each panel compares a variant with full ARGO and TWISTER. Lines show mean return and shaded regions one sample standard deviation across three runs. Curves and bands use a centered three-checkpoint moving average with endpoints preserved. Zero-step values are plotting anchors, not measured evaluations. Reported endpoint statistics use unsmoothed returns.

variant achieves higher mean returns than TWISTER, and full ARGO improves further, supporting the benefit of adding optimistic collection to the SRLPI-enabled agent. The gap between *No Planner* and full ARGO is most visible in early training.

Pessimism (Figure 3b). Setting κ=0 reduces performance compared to full ARGO, with No Pessimism ending noticeably below ARGO at 1M steps. No Pessimism still outperforms TWISTER, suggesting that replay-grounded candidate selection retains some benefit even without conservative scoring. The gap between No Pessimism and full ARGO confirms that the conservative local score in Eq. (7) is nonetheless important: without penalizing high-variance model predictions, the SRLPI targets become less reliable and actor improvement is weakened.

Replay anchor versus current-actor reference (Figure 3c). This control replaces p_ω with the frozen current actor $\bar{\pi}_k$ in the SRLPI target and uses an actor-only proposal ($\alpha = 0$), while retaining conservative scoring, the state-dependent temperature, and optimistic collection. The degraded performance may be attributable to limited candidate coverage and interference from the additional model-based actor update. Sampling exclusively from the current actor can underrepresent useful replay-supported actions that the actor assigns low probability, limiting the transfer of collected experience into the policy.

Planner imitation (Figure 3d). Replacing SRLPI with unweighted behavioral cloning of the planner's selected actions yields lower mean returns than full ARGO on Quadruped Run.

Together, these ablations separate four roles that are easy to conflate. First, removing the planner shows that replay-grounded policy improvement has limited ability to actively expand state-action coverage on its own; it can only improve within the support made available by replay. Second, removing pessimism while retaining replay support shows that support alone is not enough: replay-grounded candidates must still be ranked by a reliability-aware local evaluation, since high ensemble disagreement often co-occurs with inflated model-predicted advantages in out-of-support regions, and pessimism is what converts that signal into a penalty. Third, the actor-reference comparison tests replay anchoring against regularization toward the current policy while retaining conservative model-based evaluation. Finally, full ARGO achieves higher mean returns than the planner-imitation variant, which replaces SRLPI with unweighted behavioral cloning. The *Cheetah Run* ablations in Appendix D include the same four comparisons.

5 Conclusions and Limitations

ARGO couples optimistic collection with replay-grounded auxiliary actor learning and improves aggregate sample efficiency in visual control. Its effectiveness depends on learned latent dynamics, ensemble disagreement, and a replay-action density model. Improving the reliability of these estimates under distribution shift remains a shared challenge in model-based RL. Our experiments focus on visual continuous control, so future work could study how to apply the design on longer-horizon exploration problems, multi-task settings, and real-world robotics.

AI USE STATEMENT

We used generative AI tools to provide feedback on research methodology and experimental design, clarify mathematical arguments, assist with processing and statistical analysis of evaluation results. We take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We have taken several steps to make our results reproducible. The main paper and appendix describe the proposed method, experimental setup, datasets, evaluation procedures, and implementation details needed to understand and reproduce our experiments. We also report the key training settings and hyperparameters used in our experiments. In addition, we also provide the source code as anonymous supplementary material, including the code needed to train and evaluate our method and reproduce the main results reported in the paper.

REFERENCES

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Rémi Munos, Nicolas Heess, and Martin A. Riedmiller. Maximum a posteriori policy optimisation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=S1ANxQW0b.

Marvin Alles, Philip Becker-Ehmck, Patrick van der Smagt, and Maximilian Karl. Constrained latent action policies for model-based offline reinforcement learning. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/82389fbff376d1e8aec510916d50d054-Abstract-Conference.html.

Maxime Burchi and Radu Timofte. Learning transformer-based world models with contrastive predictive coding. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=YK9G4Htdew.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 4759–4770, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/3de568f8597b94bda53149c7d7f5958c-Abstract.html.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, Proceedings of Machine Learning Research, pp. 1856–1865. PMLR, 2018. URL http://proceedings.mlr.press/v80/haarnoja18b.html.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy P. Lillicrap. Mastering diverse control tasks through world models. *Nat.*, 640(8059):647–653, 2025. doi: 10.1038/S41586-025-08744-2. URL https://doi.org/10.1038/s41586-025-08744-2.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Temporal difference learning for model predictive control. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, Proceedings of Machine Learning Research, pp. 8387–8406. PMLR, 2022. URL https://proceedings.mlr.press/v162/hansen22a.html.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: scalable, robust world models for continuous control. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=Oxh5CstDJU.

Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Mohammadamin Barekatain, Simon Schmitt, and David Silver. Learning and planning in complex action spaces. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4476–4486. PMLR, 2021. URL http://proceedings.mlr.press/v139/hubert21a.html.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 12498–12509, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/5faf461eff3099671ad63c6f3f094f7f-Abstract.html.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/f7efa4f864ae9b88d43527f4b14f750f-Abstract.html.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=68n2s9ZJWF8.

Haotian Lin, Pengcheng Wang, Jeff Schneider, and Guanya Shi. Td-m(pc)^2: Improving temporal difference MPC through policy constraint. *CoRR*, abs/2502.03550, 2025. doi: 10.48550/ARXIV.2502.03550. URL https://doi.org/10.48550/arXiv.2502.03550.

Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=vhFu1Acb0xb.

Ashvin Nair, Murtaza Dalal, Abhishek Gupta, and Sergey Levine. Accelerating online reinforcement learning with offline datasets. *CoRR*, abs/2006.09359, 2020. URL https://arxiv.org/abs/2006.09359.

Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, Proceedings of Machine Learning Research, pp. 5062–5071. PMLR, 2019. URL http://proceedings.mlr.press/v97/pathak19a.html.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *CoRR*, abs/1910.00177, 2019. URL http://arxiv.org/abs/1910.00177.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy P. Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *Nat.*, 588(7839):604–609, 2020. doi: 10.1038/S41586-020-03051-4. URL https://doi.org/10.1038/s41586-020-03051-4.

Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, Proceedings of Machine Learning Research, pp. 8583–8592. PMLR, 2020. URL http://proceedings.mlr.press/v119/sekar20a.html.

Harshit Sikchi, Wenxuan Zhou, and David Held. Learning off-policy with online planning. In Aleksandra Faust, David Hsu, and Gerhard Neumann (eds.), *Conference on Robot Learning, 8-11 November 2021, London, UK*, volume 164 of *Proceedings of Machine Learning Research*, pp. 1622–1633. PMLR, 2021. URL https://proceedings.mlr.press/v164/sikchi22a.html.

Bhavya Sukhija, Lenart Treven, Carmelo Sferrazza, Florian Dörfler, Pieter Abbeel, and Andreas Krause. SOMBRL: scalable and optimistic model-based RL. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/d17477a5ff8b7ddb49e53f9c04305da5-Abstract-Conference.html.

Yihao Sun, Jiaji Zhang, Chengxing Jia, Haoxin Lin, Junyin Ye, and Yang Yu. Model-bellman inconsistency for model-based offline reinforcement learning. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, Proceedings of Machine Learning Research, pp. 33177–33194. PMLR, 2023. URL https://proceedings.mlr.press/v202/sun23q.html.

Tingwu Wang and Jimmy Ba. Exploring model-based planning with policy networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=H1exf64KwH.

Xiyao Wang, Ruijie Zheng, Yanchao Sun, Ruonan Jia, Wichayaporn Wongkamjan, Huazhe Xu, and Furong Huang. Coplanner: Plan to roll out conservatively but to explore optimistically for model-based RL. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=jnFcKjtUPN.

Jialong Wu, Haixu Wu, Zihan Qiu, Jianmin Wang, and Mingsheng Long. Supported policy optimization for offline reinforcement learning. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/caa934a507a952698d54efb24845fc4b-Abstract-Conference.html.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *CoRR*, abs/1911.11361, 2019. URL http://arxiv.org/abs/1911.11361.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y. Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: model-based offline policy optimization. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/a322852ce0df73e204b7e67cbbef0d0a-Abstract.html.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. COMBO: conservative offline model-based policy optimization. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp.

28954–28967, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/f29a179746902e331572c483c45e5086-Abstract.html.

Guojian Zhan, Likun Wang, Xiangteng Zhang, Jiaxin Gao, Masayoshi Tomizuka, and Shengbo Eben Li. Bootstrap off-policy with world model. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=zNqDCSokDR.

Weipu Zhang, Gang Wang, Jian Sun, Yetian Yuan, and Gao Huang. STORM: efficient stochastic transformer based world models for reinforcement learning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/5647763d4245b23e6a1cb0a8947b38c9-Abstract-Conference.html.

A Additional derivations

A.1 Derivation of the SRLPI target

We show that Eq. (1) is the unique maximizer of the replay-regularized local improvement problem in Eq. (10).

For a fixed replayed model state s_t, abbreviate

$$\hat{A}(a) := \hat{A}_{H,k}(s_t, a), \qquad \tau := \tau_k(s_t), \qquad p(a) := p_\omega(a \mid s_t).$$

Define the statewise objective

$$J(\pi) = \mathbb{E}_{a\sim\pi}[\hat{A}(a)] - \tau D_{\mathrm{KL}}(\pi\|p),$$

over probability densities π on $\mathcal{A}$ such that $\pi \ll p$. Assume $\tau > 0$ and

$$Z := \int_{\mathcal{A}} p(a) \exp\left(\frac{\hat{A}(a)}{\tau}\right) \mathrm{d}a \in (0, \infty).$$

Define

$$\pi^\star(a) := \frac{1}{Z}\, p(a) \exp\left(\frac{\hat{A}(a)}{\tau}\right).$$

Proposition 1. For any feasible π,

$$J(\pi) = \tau \log Z - \tau D_{\mathrm{KL}}(\pi\|\pi^\star).$$

Consequently, J is uniquely maximized at $\pi = \pi^\star$, and hence

$$\pi_k^\star(a_t \mid s_t) \propto p_\omega(a_t \mid s_t) \exp\left(\frac{\hat{A}_{H,k}(s_t, a_t)}{\tau_k(s_t)}\right).$$

Proof. Since

$$\log \pi^\star(a) = \log p(a) + \frac{\hat{A}(a)}{\tau} - \log Z,$$

we have

$$\begin{aligned} D_{\mathrm{KL}}(\pi\|\pi^\star) &= \mathbb{E}_{a\sim\pi}\left[\log \pi(a) - \log p(a) - \frac{\hat{A}(a)}{\tau} + \log Z\right] \\ &= D_{\mathrm{KL}}(\pi\|p) - \frac{1}{\tau}\mathbb{E}_{a\sim\pi}[\hat{A}(a)] + \log Z. \end{aligned}$$

Multiplying by τ and rearranging yields

$$\mathbb{E}_{a\sim\pi}[\hat{A}(a)] - \tau D_{\mathrm{KL}}(\pi\|p) = \tau \log Z - \tau D_{\mathrm{KL}}(\pi\|\pi^\star),$$

which is the claimed identity. Since $\tau > 0$ and $D_{\mathrm{KL}}(\pi\|\pi^\star) \geq 0$ with equality iff $\pi = \pi^\star$ almost everywhere, the maximizer is unique and equals $\pi^\star$. □

A.2 Continuous-action approximation of the projection step

We derive Eq. (14) and Eq. (15) as the self-normalized importance-sampling approximation of the exact projection objective in Eq. (12).

For a fixed replayed model state s_t, define

$$f_\theta(a, s_t) := -\log \pi_\theta(a \mid s_t).$$

Using the forward KL projection objective, the exact per-state loss can be written as

$$D_{\mathrm{KL}}(\pi_k^\star(\cdot \mid s_t)\|\pi_\theta(\cdot \mid s_t)) = C(s_t) + \mathbb{E}_{a\sim\pi_k^\star(\cdot|s_t)}[f_\theta(a, s_t)],$$

where

$$C(s_t) := \mathbb{E}_{a\sim\pi_k^\star(\cdot|s_t)}\big[\log \pi_k^\star(a \mid s_t)\big]$$

does not depend on θ. Therefore minimizing the forward KL is equivalent to minimizing

$$\ell_\theta(s_t) := \mathbb{E}_{a\sim\pi_k^\star(\cdot|s_t)}[f_\theta(a, s_t)].$$

Define the unnormalized target density

$$q_k(a \mid s_t) := p_\omega(a \mid s_t) \exp\left(\frac{\hat{A}_{H,k}(s_t, a)}{\tau_k(s_t)}\right), \qquad Z_k(s_t) := \int q_k(a \mid s_t)\, \mathrm{d}a,$$

so that

$$\pi_k^\star(a \mid s_t) = \frac{q_k(a \mid s_t)}{Z_k(s_t)}.$$

Let $\nu_k(\cdot \mid s_t)$ be any proposal distribution satisfying

$$\mathrm{supp}\big(\pi_k^\star(\cdot \mid s_t)\big) \subseteq \mathrm{supp}\big(\nu_k(\cdot \mid s_t)\big).$$

Then

$$\begin{aligned}\ell_\theta(s_t) &= \int f_\theta(a, s_t) \frac{q_k(a \mid s_t)}{Z_k(s_t)}\, \mathrm{d}a \\ &= \frac{\mathbb{E}_{a\sim\nu_k(\cdot|s_t)}\left[\frac{q_k(a|s_t)}{\nu_k(a|s_t)} f_\theta(a, s_t)\right]}{\mathbb{E}_{a\sim\nu_k(\cdot|s_t)}\left[\frac{q_k(a|s_t)}{\nu_k(a|s_t)}\right]}.\end{aligned}$$

We choose the proposal

$$\nu_k(\cdot \mid s_t) = \alpha\, p_\omega(\cdot \mid s_t) + (1 - \alpha)\, \bar{\pi}_k(\cdot \mid s_t), \qquad \alpha \in (0, 1].$$

Lemma 1. If

$$\nu_k(\cdot \mid s_t) = \alpha\, p_\omega(\cdot \mid s_t) + (1 - \alpha)\, \bar{\pi}_k(\cdot \mid s_t)$$

with $\alpha > 0$, then

$$\mathrm{supp}\big(\pi_k^\star(\cdot \mid s_t)\big) \subseteq \mathrm{supp}\big(\nu_k(\cdot \mid s_t)\big).$$

Proof. If $\pi_k^\star(a \mid s_t) > 0$, then $q_k(a \mid s_t) > 0$. Since

$$q_k(a \mid s_t) = p_\omega(a \mid s_t) \exp\left(\frac{\hat{A}_{H,k}(s_t, a)}{\tau_k(s_t)}\right),$$

and the exponential term is strictly positive, it follows that $p_\omega(a \mid s_t) > 0$. Therefore

$$\nu_k(a \mid s_t) = \alpha p_\omega(a \mid s_t) + (1 - \alpha)\bar{\pi}_k(a \mid s_t) \geq \alpha p_\omega(a \mid s_t) > 0.$$

Hence every point in the support of $\pi_k^\star(\cdot \mid s_t)$ also lies in the support of $\nu_k(\cdot \mid s_t)$. □

Given i.i.d. samples

$$a_t^1, \ldots, a_t^N \sim \nu_k(\cdot \mid s_t),$$

define the unnormalized and normalized importance weights

$$\tilde{w}_i(s_t) := \frac{q_k(a_t^i \mid s_t)}{\nu_k(a_t^i \mid s_t)} = \frac{p_\omega(a_t^i \mid s_t)}{\nu_k(a_t^i \mid s_t)} \exp\left(\frac{\hat{A}_{H,k}(s_t, a_t^i)}{\tau_k(s_t)}\right), \qquad w_i(s_t) := \frac{\tilde{w}_i(s_t)}{\sum_{j=1}^N \tilde{w}_j(s_t)}.$$

The self-normalized importance-sampling estimator of $\ell_\theta(s_t)$ is then

$$\hat{\ell}_\theta(s_t) = \sum_{i=1}^N w_i(s_t) f_\theta(a_t^i, s_t).$$

Substituting $f_\theta(a, s_t) = -\log \pi_\theta(a \mid s_t)$ yields the sampled projection loss

$$\widehat{L}_{\mathrm{SRLPI}}(\theta) = -\mathbb{E}_{s_t\sim D}\left[\sum_{i=1}^N w_i(s_t) \log \pi_\theta(a_t^i \mid s_t)\right].$$

Proposition 2. Assume

$$\mathbb{E}_{a\sim\nu_k(\cdot|s_t)}\left[\frac{q_k(a \mid s_t)}{\nu_k(a \mid s_t)}\right] < \infty, \qquad \mathbb{E}_{a\sim\nu_k(\cdot|s_t)}\left[\frac{q_k(a \mid s_t)}{\nu_k(a \mid s_t)}\,|f_\theta(a, s_t)|\right] < \infty.$$

Then

$$\sum_{i=1}^{N} w_i(s_t) f_\theta(a_t^i, s_t) \xrightarrow[N\to\infty]{\text{a.s.}} \mathbb{E}_{a\sim\pi_k^\star(\cdot|s_t)}[f_\theta(a, s_t)].$$

Proof. By the strong law of large numbers,

$$\frac{1}{N}\sum_{i=1}^{N} \tilde{w}_i(s_t) f_\theta(a_t^i, s_t) \xrightarrow[N\to\infty]{\text{a.s.}} \mathbb{E}_{a\sim\nu_k(\cdot|s_t)}\left[\frac{q_k(a \mid s_t)}{\nu_k(a \mid s_t)} f_\theta(a, s_t)\right],$$

and

$$\frac{1}{N}\sum_{i=1}^{N} \tilde{w}_i(s_t) \xrightarrow[N\to\infty]{\text{a.s.}} \mathbb{E}_{a\sim\nu_k(\cdot|s_t)}\left[\frac{q_k(a \mid s_t)}{\nu_k(a \mid s_t)}\right].$$

Taking the ratio yields the claim. The estimator is biased for finite N but consistent as $N \to \infty$. During actor regression, gradients are stopped through $w_i(s_t)$, since $\pi_k^\star$ is treated as a fixed target.

B Detailed Evaluation Results

B.1 Aggregate statistical analysis

Performance metrics. Final return measures performance at the end of training. The normalized area under the learning curve (nAUC) measures sample efficiency by averaging unsmoothed returns over 100k–1M environment steps and dividing by 1000. We use trapezoidal integration on a 100k-step grid, extending the last evaluation to 1M steps when needed. Both aggregate means weight tasks equally after averaging runs within each task.

Confidence intervals. Stratified bootstrap intervals estimate uncertainty in ARGO–baseline differences by resampling whole runs within each task and method, with task identities fixed. We use 50,000 resamples and Bonferroni adjustment across the eight mean comparisons; IQM intervals are pointwise 95% intervals. Bootstrap coverage is approximate, especially with few training runs per task.

Robust summaries. The interquartile mean (IQM) averages pooled run-level scores after trimming 25% from each tail, reducing sensitivity to extreme outcomes. The empirical probability of improvement measures how often an ARGO run outperforms a baseline run on the same task, averaging tasks equally and assigning half credit to ties.

Finite-sample check. Weighted Hoeffding intervals provide conservative bounds on mean differences under independent, bounded run outcomes, with the same eight-comparison adjustment. They exclude zero against SAC but include zero against DreamerV3, TD-MPC2, and TWISTER, so this distribution-free check is inconclusive for the world-model baselines. The approximate bootstrap intervals are positive for all eight mean comparisons.

C Additional Learning Curves

Figure 4 shows learning curves for *Cartpole Swingup Sparse*, *Finger Spin*, *Reacher Easy*, and *Walker Walk*.

D Additional Ablation Study

Figure 5 presents planner, pessimism, current-actor-reference, and planner-imitation ablations on *Cheetah Run*, complementing the Quadruped Run comparisons in Figure 3.

Table 3: Aggregate statistics for the runs in Table 1 on 13 tasks. nAUC covers 100k–1M environment steps. Differences and improvement probabilities compare ARGO with the named baseline. Mean-difference intervals are Bonferroni-adjusted across eight comparisons at nominal 95% simultaneous coverage; IQM intervals are pointwise 95% percentile-bootstrap intervals. ARGO uses five runs per task and each baseline uses three.

Aggregate performance

Method	Runs/task	Final mean	Final IQM	nAUC mean	nAUC IQM
SAC	3	139.7	82.9	0.1112	0.0592
DreamerV3	3	725.4	789.8	0.5527	0.5863
TD-MPC2	3	662.9	708.8	0.5143	0.5109
TWISTER	3	697.6	752.6	0.5289	0.5490
ARGO	5	785.0	856.0	0.6346	0.6699

Mean differences and adjusted intervals

Baseline	Metric	Δ mean	Gain (%)	Bootstrap interval
SAC	Final return	645.3	462.0	$[609.4, 694.3]$
	nAUC	0.5234	470.6	$[0.4944, 0.5617]$
DreamerV3	Final return	59.5	8.2	$[24.7, 97.0]$
	nAUC	0.0819	14.8	$[0.0530, 0.1116]$
TD-MPC2	Final return	122.1	18.4	$[94.4, 151.5]$
	nAUC	0.1204	23.4	$[0.0802, 0.1512]$
TWISTER	Final return	87.3	12.5	$[55.8, 120.2]$
	nAUC	0.1057	20.0	$[0.0791, 0.1310]$

Robust differences and empirical improvement probabilities

Baseline	Metric	Δ IQM	IQM bootstrap interval	$P(\text{ARGO} > \text{baseline})$ (%)
SAC	Final return	773.2	$[755.4, 791.7]$	97.4
	nAUC	0.6107	$[0.5908, 0.6258]$	100.0
DreamerV3	Final return	66.2	$[37.2, 96.6]$	70.3
	nAUC	0.0835	$[0.0516, 0.1153]$	77.4
TD-MPC2	Final return	147.3	$[119.5, 178.6]$	70.8
	nAUC	0.1590	$[0.1147, 0.1990]$	76.9
TWISTER	Final return	103.5	$[74.1, 133.6]$	83.3
	nAUC	0.1209	$[0.0947, 0.1455]$	95.9

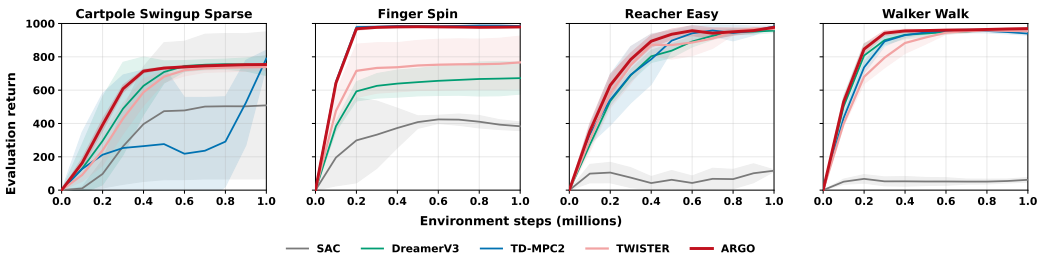
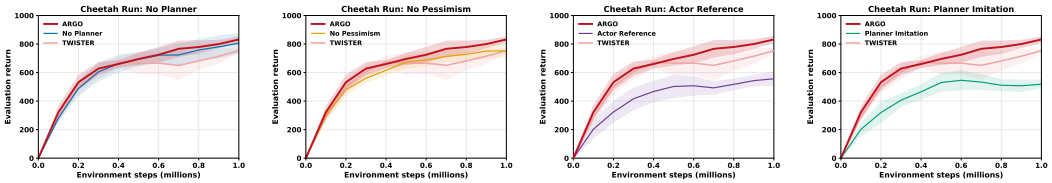


Figure 4: Learning curves on the tasks not shown in the main paper: *Cartpole Swingup Sparse*, *Finger Spin*, *Reacher Easy*, and *Walker Walk*. Lines show mean evaluation return and shaded regions one sample standard deviation across five runs for ARGO and three runs per baseline. Curves and bands are smoothed for display.



(a) No Planner (b) No Pessimism (κ=0) (c) Actor Reference (d) Planner Imitation

Figure 5: Ablations on *Cheetah Run*. Each panel compares a modified ARGO configuration (coloured line) with full ARGO (red) and TWISTER (pink). Lines show mean evaluation returns and shaded regions one sample standard deviation across three runs. Curves and bands are smoothed for display.

Planner (Figure 5a). Removing the optimistic planner reduces performance and No Planner ends marginally below full ARGO at 1M steps, with both outperforming TWISTER. The planner's contribution is smaller here than on Quadruped Run, which is consistent with Cheetah Run's smooth locomotion structure: the actor's support expands adequately on its own, leaving less opportunity for optimistic exploration to discover distinctly new replay regions. Nonetheless, the small but consistent gap confirms that the planner still brings benefit even in this setting.

Pessimism (Figure 5b). Setting κ=0 degrades performance, falling at or below the TWISTER backbone. This confirms that conservative local scoring is necessary for SRLPI targets to be reliable: without penalizing high-variance model predictions, the actor is updated toward candidates that look favorable under the model but may exploit ensemble disagreement rather than genuine environmental value.

Replay anchor versus current-actor reference (Figure 5c). The current-actor-reference variant reaches a final return of 556.2 ± 46.8, compared with 831.9 ± 22.4 for full ARGO and 753.1 ± 16.0 for TWISTER, across three runs per condition. Its lower observed mean return is consistent with the Quadruped Run comparison.

Planner imitation (Figure 5d). The planner-imitation variant, which replaces SRLPI with unweighted behavioral cloning of selected planner actions, has a lower mean return than full ARGO on Cheetah Run.

The Cheetah Run ablations examine collection-time planning and alternative actor-learning designs in a second task. The actor-reference variant has a lower observed final return than full ARGO on both Cheetah Run and Quadruped Run.

E ARGO ALGORITHM

Algorithms 1 and 2 give the full pseudocode for ARGO. Algorithm 1 describes the outer training loop, including optimistic data collection and the sequential model updates. Algorithm 2 details the SRLPI actor update, which is called as a sub-procedure in Algorithm 1.

Algorithm 1 ARGO Training

Initialize: world model $\hat{P}_\phi$ with M bootstrap heads $\{\hat{P}_{\phi_m}\}_{m=1}^{M}$ (reward and continue), actor π_θ, critic V_ψ, slow-critic EMA $\bar{V}$ (initialized to V_ψ), support model p_ω, replay buffer $\mathcal{D} \leftarrow \varnothing$
for each epoch **do**
  for each environment step **do**
    Encode observation o_t into model state s_t via the world model encoder
    Sample base action $a^\pi \sim \pi_\theta(\cdot \mid s_t)$
    // Optimistic planner (training only)
    Build candidate set $C_t = \{a^\pi\} \cup \{\tilde{a}^\ell\}_{\ell=1}^{|C_t|-1}$, where each $\tilde{a}^\ell$ is drawn independently from $\bar{\pi}_k(\cdot \mid s_t)$ (probability $1-\epsilon_{\text{plan}}$) or from a Gaussian perturbation $\mathcal{N}(b^\ell, \sigma^2_{\text{plan}} I)$ of an actor anchor $b^\ell \sim \bar{\pi}_k$ (probability ϵ_{plan})
    for each $a \in C_t$ **do**
      Compute $(\mu^{\text{plan}}, \sigma^{\text{plan}})$ via H_p-step rollout under $\{\hat{P}_{\phi_m}\}$ and $\bar{\pi}_k$, bootstrapped by $\bar{V}$ (Eq. 16–18)
      $U(a) = \mu^{\text{plan}}(s_t, a) + \beta\, \sigma^{\text{plan}}(s_t, a)$ *optimistic utility*
    end for
    $a_t^{\text{exec}} = \arg\max_{a \in C_t} U(a)$
    Execute a_t^{exec} in the environment, observe r_t, o_{t+1}; add $(o_t, a_t^{\text{exec}}, r_t, o_{t+1})$ to $\mathcal{D}$
  end for
  for each training step **do**
    Sample a batch of sequences $\{(o_{t:t+L}, a_{t:t+L}, r_{t:t+L})\}$ from $\mathcal{D}$
    Encode sequences into model states $\{s_t\}$
    World model update: optimize ϕ via ELBO (decoder, KL) and NLL for reward, continue; train bootstrap heads $\{\hat{P}_{\phi_m}\}$ with bootstrap masking (keep-prob p_b)
    Support model update: optimize $\omega \leftarrow \omega + \nabla_\omega\, \mathbb{E}_{(s_t, a_t) \sim \mathcal{D}}[\log p_\omega(a_t \mid s_t)]$
    Actor update: $\mathcal{L}^{\text{ARGO}}_{\text{actor}}(\theta) = \mathcal{L}^{\text{base}}_{\text{actor}}(\theta) + \lambda_{\text{SRLPI}}\, \widehat{\mathcal{L}}_{\text{SRLPI}}(\theta)$ ($\mathcal{L}^{\text{base}}_{\text{actor}}$: unchanged TWISTER actor objective; see Algorithm 2)
    Critic update: optimize V_ψ on TD-λ targets from H-step imagination rollouts under π_θ
    Soft-update slow critic: $\bar{V} \leftarrow (1-\eta)\, \bar{V} + \eta\, V_\psi$
  end for
end for

F Implementation Details and Hyperparameters

ARGO uses TWISTER (Burchi & Timofte, 2025) as its model-based RL backbone. All hyperparameters are shared across tasks unless otherwise noted. Table 4 lists all values used in our experiments.

Baseline results. Run-level results for SAC (Haarnoja et al., 2018) and DreamerV3 (Hafner et al., 2025) are obtained from the official DreamerV3 repository. We use the released TD-MPC2 results from its official repository, which contain three training runs per task (Hansen et al., 2024). TWISTER's per-run results are not released, so we reproduce them using its official implementation (Burchi & Timofte, 2025).

Evaluation protocol. Table 1, the learning curves, and the aggregate analysis use the same five independent training runs per task for ARGO and three training runs per task for each baseline. Table 1 reports means and sample standard deviations across training runs. Aggregate mean returns weight the 13 tasks equally. The component ablations use three runs per condition, including the full-ARGO comparator. Table 2 reports mean final returns over three training runs per task for TWISTER, the TWISTER-based COPlanner adaptation, and ARGO, without uncertainty estimates.

COPlanner–TWISTER adaptation. COPlanner–TWISTER uses the TWISTER backbone and the same bootstrap ensemble as ARGO. We use COPlanner's published visual-control planning settings (Wang et al., 2024): four action candidates, a four-step planning horizon, an optimistic coefficient of 1, and a conservative coefficient of 0.5 (2 for Quadruped Run). Ensemble disagreement

Algorithm 2 SRLPI Actor Update ($\mathcal{L}_{\mathrm{SRLPI}}$)

Require: Model states $\{s_t\}$ from current batch; frozen reference networks $\bar{\pi}_k = \mathrm{sg}(\pi_\theta)$, $\bar{V}$ (slow-critic EMA); bootstrap heads $\{\hat{P}_{\phi_m}\}_{m=1}^{M}$; support model p_ω

Select valid root states $\{s_t\} \subset \{s_t\}$ (exclude episode-initial steps)

// Conservative local evaluation

Sample N candidate actions from the mixture proposal: $a^i \sim \nu_k(\cdot \mid s_t) = \alpha\, p_\omega(\cdot \mid s_t) + (1-\alpha)\, \bar{\pi}_k(\cdot \mid s_t)$

for each candidate a^i, each head $m = 1, \ldots, M$ **do**

  $\bar{G}^m_{H,k}(s_t, a^i)$: H-step rollout under $\hat{P}_{\phi_m}$ and $\bar{\pi}_k$, bootstrapped by $\bar{V}$ at step H (Eq. 4)

end for

$\mu(s_t, a^i) = \frac{1}{M}\sum_{m=1}^{M} \bar{G}^m_{H,k}(s_t, a^i), \quad \sigma(s_t, a^i) = \mathrm{std}_m\left[\bar{G}^m_{H,k}(s_t, a^i)\right]$ (Eq. 5–6)

$G(s_t, a^i) = \mu(s_t, a^i) - \kappa\, \sigma(s_t, a^i)$ *conservative local score* (Eq. 7)

// State-adaptive temperature

Sample actor actions $\{a^j_\pi\}_{j=1}^{N} \sim \bar{\pi}_k(\cdot \mid s_t)$ and compute their epistemic uncertainties $\sigma(s_t, a^j_\pi)$

$\hat{\Sigma}(s_t) = \frac{1}{N}\sum_{j=1}^{N} \sigma(s_t, a^j_\pi)$ *Monte Carlo estimate of Eq. 9*

$\tau(s_t) = \mathrm{clip}\Big(\tau_0 + \rho\, \hat{\Sigma}(s_t),\ \tau_{\min},\ \tau_{\max}\Big)$ (Eq. 9)

// SRLPI importance weights and loss

Compute log-probabilities $\log p_\omega(a^i \mid s_t)$ and $\log \nu_k(a^i \mid s_t)$ (stop gradients)

Compute normalized weights (Eq. 14):

$$\tilde{w}_i(s_t) = \frac{p_\omega(a^i \mid s_t)}{\nu_k(a^i \mid s_t)} \exp\left(\frac{G(s_t, a^i)}{\tau(s_t)}\right), \qquad w_i(s_t) = \frac{\tilde{w}_i(s_t)}{\sum_{j=1}^{N} \tilde{w}_j(s_t)}$$

$\mathcal{L}_{\mathrm{SRLPI}}(\theta) = -\mathbb{E}_{s_t \sim \mathcal{D}}\Big[\sum_{i=1}^{N} w_i(s_t)\, \log \pi_\theta(a^i \mid s_t)\Big]$, gradients stopped through $w_i(s_t)$ (Eq. 15)

return $\mathcal{L}_{\mathrm{SRLPI}}(\theta)$

is added as an exploration bonus during real-action selection and subtracted as a penalty during imagined-action selection. The latter invokes candidate lookahead at each step of synthetic rollout generation.

Compute resources. On a single NVIDIA GeForce RTX 5090 GPU (32 GB VRAM), the measured update times are approximately 0.90 s for COPlanner–TWISTER, 0.29 s for ARGO, and 0.19 s for TWISTER.

Backbone (TWISTER). We use the small (S) model configuration throughout. The stochastic state is a 32×32 categorical representation; the deterministic recurrent state has dimension 512. The Transformer sequence model uses 4 blocks with 8 attention heads, a feedforward expansion ratio of 2, and a left-context window of 8. All MLP components (actor, critic, reward head, continue head) use 2 hidden layers of width 512.

Support model p_ω. We parameterize $p_\omega(a \mid s_t)$ as a Gaussian mixture model with $K = 5$ components whose parameters are output by a 3-layer MLP of width 512, conditioned on the model state s_t. The minimum per-component standard deviation is 0.05. The support model is trained by maximum likelihood on replay transitions, with a dedicated Adam optimizer. Log-probabilities are left unclamped (no effective clipping), so the model can assign arbitrarily low density to out-of-support actions.

Bootstrap ensemble. Epistemic uncertainty is estimated via an ensemble of $M = 3$ bootstrap heads. Each head maintains independent reward, continuation, and transition prediction networks, trained with per-step bootstrap masking (keep-probability 0.8) so that each head sees a different random subset of transitions and their predictions diverge under model uncertainty. During rollouts, each head produces 4 independent trajectory samples that are averaged to form the model-wise expected return $\bar{G}^m$ for that head. Epistemic uncertainty is then computed as the standard deviation

across the M head-wise means, as in Eq. 6. The within-head samples are used only to reduce Monte Carlo noise in estimating each $\bar{G}^m$. The value function at the rollout horizon ($\bar{V}$ in Eq. 4) is a single shared slow-critic EMA rather than a per-head estimate: this prevents value-estimation variance from compounding with reward-and-continuation variance across rollout steps, which we found important for stable temperature adaptation.

SRLPI candidate sampling. At each SRLPI update, $N = 8$ action candidates are sampled per root state from the mixture proposal $\nu_k = \alpha\, p_\omega + (1 - \alpha)\, \bar{\pi}_k$ with $\alpha = 0.2$. We also draw $N = 8$ frozen-actor samples from $\bar{\pi}_k$ to estimate the state-level uncertainty mean in Eq. 9. As noted in the main text, the implementation uses $G_{H,k}$ directly in place of $\hat{A}_{H,k}$, so the actor-value baseline in Eq. 8 is not computed. A mini-batch of 64 root states is processed per update step.

Planner candidate generation. The planner constructs $|C_t| = 32$ candidates per real step. Each candidate is independently drawn from the actor $\bar{\pi}_k$ with probability $1 - \epsilon_{\text{plan}} = 0.8$, or from a Gaussian perturbation of an actor anchor with probability $\epsilon_{\text{plan}} = 0.2$. The perturbation kernel $E_k(\cdot \mid s_t, b)$ is $\mathcal{N}(b,\, \sigma^2_{\text{plan}} I)$ clipped to the normalized action range, with $\sigma_{\text{plan}} = 0.2$. The planner is active only during training.

Table 4: ARGO hyperparameters used across all DMControl experiments.

Group	Parameter	Symbol	Value
SRLPI	Imagination horizon	H	15
	Number of action candidates	N	8
	Proposal mixture ratio (support fraction)	α	0.2
	Conservatism coefficient	κ	0.5
	Temperature baseline	τ_0	0.5
	Temperature sensitivity	ρ	1.0
	Temperature minimum	$\tau_{\min}$	0.2
	Temperature maximum	$\tau_{\max}$	2.0
	SRLPI loss weight	λ_{SRLPI}	0.01
Ensemble	Number of bootstrap heads	M	3
	Rollouts per head	—	4
	Bootstrap keep probability	—	0.8
Planner	Planning horizon	H_p	5
	Total candidates	$\lvert C_t \rvert$	32
	Perturbation fraction	ϵ_{plan}	0.2
	Perturbation std	σ_{plan}	0.2
	Optimism coefficient	β	0.2
Support model p_ω	Architecture	—	MLP, 3 layers, width 512
	Mixture components	K	5
	Minimum component std	—	0.05
	Learning rate	—	10^{-4}
Backbone (TWISTER)	Batch size	—	16
	Sequence length	L	64
	Imagination horizon (base)	—	15
	Replay buffer capacity	—	10^6
	Discount factor	γ	0.997
	TD-λ	λ	0.95
	World model learning rate	—	10^{-4}
	Actor / critic learning rate	—	3×10^{-5}
	Slow-critic EMA update rate	η	0.02
	Action repeat	—	2

F.1 Hyperparameter Selection and Sensitivity

We selected the ARGO-specific hyperparameters through a small manual search of approximately four to five configurations on Walker Run. The selected configuration was then applied unchanged

Table 5: Sensitivity to uncertainty-related hyperparameters on Quadruped Run after 1M environment steps. Bold parameter values indicate the reference configuration.

Parameter	**Value**	**Final return**
Ensemble heads M	2	672.0
	3	869.5
	5	873.0
Pessimism κ	0.0	817.0
	0.5	869.5
	2.0	775.0
Optimism β	**0.2**	869.5
	0.4	840.0

across all 13 benchmark tasks, without task-specific retuning. Walker Run is included in the reported benchmark suite.

Table 5 reports a separate sensitivity study on Quadruped Run for the ensemble size M, pessimism coefficient κ, and optimism coefficient β. All settings use a budget of 1M environment steps. The reference configuration uses $M = 3$, $\kappa = 0.5$, and $\beta = 0.2$.

The reported returns for three and five ensemble heads are similar, while two heads yield a lower return. Increasing κ from 0.5 to 2.0 or β from 0.2 to 0.4 also lowers the reported return relative to the reference configuration. All tested settings exceed the TWISTER reference mean of 572.0 ± 111.8 on this task. At $\kappa = 2.0$, late-training ensemble disagreement for the highest-weighted SRLPI candidate is 0.53, compared with approximately 0.73–0.81 at the less conservative settings. For new applications, we recommend starting from the shared configuration.