ONE FLOW-TRANSFORMER FOR IMAGINATION AND CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion and flow models are effective world models for visual reinforcement learning, letting agents learn behaviors in imagination. Yet existing diffusion-based online agents use them as black-box simulators: a separate policy network learns from generated rollouts, leaving the backbone's representations unused for control. We introduce DRIFT, an efficient agent whose policy learns entirely in imagination on the same Flow Transformer backbone as its world model, trained online from scratch. DRIFT's imagination is fast: it runs in a compact continuous latent space, using $33\times$ fewer FLOPs per imagined transition than prior diffusion-based agents. On online RL benchmarks, DRIFT outperforms diffusion and flow baselines on Craftium and Crafter, and is competitive on Atari and with discrete latent-dynamics agents. Three modeling choices make this possible. First, we find that denoising alone yields a less effective control representation. We therefore train the backbone to predict the next latent from the policy's compact state, a reconstruction-free objective that improves control. Second, we observe that reward predictions are less accurate on generated latents than on real ones, which can hinder policy learning. A simple and effective mitigation is to also train the world model on its own rollouts. Third, shortcut flow matching reduces imagination to one denoising step per transition, matching multi-step sampling in rollout accuracy. These results show that one generative backbone can both imagine and control in online RL.

1 INTRODUCTION

World models let reinforcement learning (RL) agents train policies on imagined experience, with far fewer environment interactions than model-free methods (Ha & Schmidhuber, 2018; Hafner et al., 2020). The most successful online model-based agents, such as the recurrent DreamerV3 (Hafner et al., 2023) and the Transformer-based STORM (Zhang et al., 2023), share one learned representation between the world model and the policy (Burchi & Timofte, 2025). However, these agents compress observations into low-dimensional latent states or discrete tokens (Hafner et al., 2023; Micheli et al., 2023), which can discard visual details relevant for control (Alonso et al., 2024). Diffusion and flow models avoid this bottleneck by generating in continuous space, with action-conditioned rollouts faithful enough to train policies on (Cohen et al., 2026; Hafner et al., 2025). Yet existing diffusion-based online agents such as DIAMOND (Alonso et al., 2024) and Horizon Imagination (Cohen et al., 2026) use these models as black-box simulators: a separate policy network learns from their rollouts, leaving the backbone's representations unused for control. Dreamer 4 (Hafner et al., 2025) uses the features of a pretrained Diffusion Transformer (DiT) for its policy, but relies on large-scale offline video pretraining.

While sharing a denoising backbone with the policy is similar in spirit to latent world models such as DreamerV3, doing so in an online agent trained from scratch raises unique challenges for effective and efficient policy learning in imagination:

1. **Denoising–control representation gap.** To act, the policy needs a compact state, naturally read out from the Flow Transformer's denoised tokens. However, the Flow Transformer generates the next latent through attention over all tokens, so the readout plays no special role in the denoising objective. Recent work finds that strong visual reconstruction need not yield action-relevant representations (Yeom et al., 2026). For example, accurately generating a Boxing opponent's future position does not guarantee that the compact state encodes it. In contrast, Dreamer's

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.

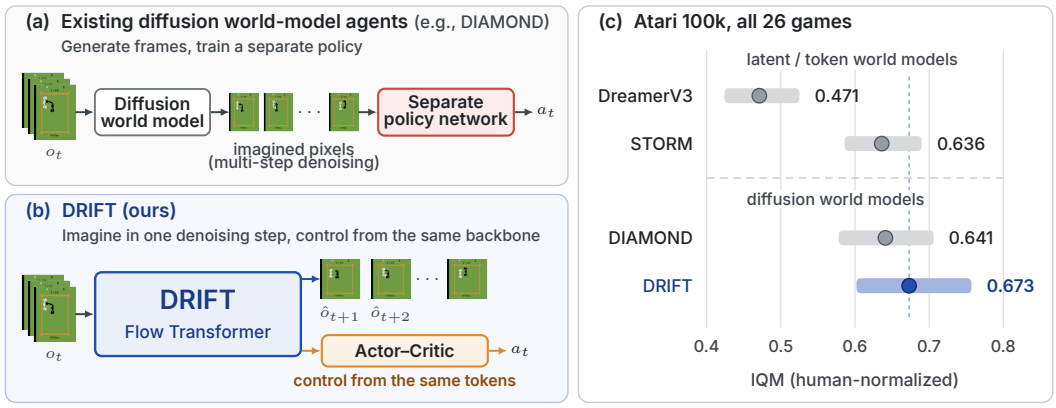

Figure 1: **One Flow Transformer for imagination and control.** (a) Diffusion world-model agents train a separate policy on generated frames. (b) DRIFT imagines with one denoising step and reads the actor–critic's compact state from the same backbone, trained with next-latent prediction and on its own rollouts. (c) Atari 100k IQM with 95% CIs.

recurrent state-space model (RSSM) (Hafner et al., 2019) predicts future states from its recurrent state and the action, encouraging predictive representations. This raises the question of whether a compact readout of denoising features retains the predictive information control needs.

2. **Train–imagination distribution gap.** In imagination, the policy learns from predicted rewards. The reward predictor is trained on real latents but scores generated ones. RSSM agents regularize predicted and encoded state distributions with a KL term (Hafner et al., 2023), while standard flow-matching objectives lack an analogous constraint. This leaves generated latents free to drift from the predictor's training distribution, potentially biasing the actor–critic's imagined returns.
3. **Imagination cost.** Actor–critic learning needs thousands of imagined transitions per epoch, and diffusion and flow models take several denoising steps per frame (Lipman et al., 2023; Alonso et al., 2024), so imagination becomes the computational bottleneck.

We introduce DRIFT, an online continuous-latent RL agent in which one Flow Transformer serves as both world model and actor–critic backbone. The actor and critic read a compact control state from it. However, we find that denoising and reward prediction alone yield a less effective control state. To tackle this denoising–control representation gap, we introduce a reconstruction-free objective that predicts the next latent from this state and the action, encouraging a predictive representation. We also find that reward predictions are less accurate on generated latents. To address this train–imagination distribution gap, we also train the world model on its own rollouts, exposing the reward predictor to generated latents. The backbone runs in a VAE latent space to reduce the imagination cost. Shortcut flow matching (Frans et al., 2025) cuts this to one denoising step per imagined transition, with past-frame attention cached for reuse.

We evaluate DRIFT at standard budgets on three benchmarks: Atari 100k (Kaiser et al., 2019) (26 games, from reactive to sparse-reward exploration), Craftium (Malagón et al., 2025) (3D voxel navigation), and Crafter (Hafner, 2022) (2D survival with sequential tool crafting). On Atari, DRIFT achieves higher median and interquartile mean (IQM) human-normalized scores than DIAMOND.

Our contributions are:

- We present DRIFT, an efficient online agent whose policy learns entirely in imagination. Its world model and policy share one Flow Transformer backbone (27.7M parameters, trained from scratch). It also uses 33× fewer FLOPs per imagined transition than DIAMOND.
- Across online RL benchmarks, DRIFT is better than the diffusion and flow baselines on Craftium and Crafter, and competitive with them on Atari and with discrete latent-dynamics agents.
- We analyze three modeling choices with ablations and probes. Next-latent prediction gives the control state a predictive signal that improves control. Reward predictions are less accurate on generated latents, and training on self-generated rollouts targets this gap and improves policy learning. Finally, one-step shortcut imagination matches multi-step sampling in rollout accuracy.

2 Related Work

World models. Latent world models learn dynamics in compact latent spaces, using recurrent (Hafner et al., 2023) or Transformer (Zhang et al., 2023; Robine et al., 2023; Micheli et al., 2023) backbones. These models represent observations as categorical latents (Hafner et al., 2023) or discrete tokens (Micheli et al., 2023; Dedieu et al., 2025), which makes rollouts fast but can discard visual details relevant for control (Alonso et al., 2024). Their representations for control can be improved by rebalancing the observation and reward losses (Ma et al., 2024a) or by adding predictive objectives (Burchi & Timofte, 2025; Maes et al., 2026). Future-latent prediction also improves model-free agents (Schwarzer et al., 2021). Like DreamerV3, DRIFT feeds the policy the world model's own state, but it keeps continuous latents instead of categorical or discrete ones.

Diffusion and flow world models. Diffusion and flow models offer richer transition modeling in continuous space. DIAMOND (Alonso et al., 2024) applies pixel-space diffusion with a U-Net backbone and trains a separate CNN–LSTM actor–critic on the generated frames. Horizon Imagination (Cohen et al., 2026) moves to latent rectified-flow dynamics with a DiT backbone but still trains a separate lightweight controller on the imagined rollouts. Dreamer 4 (Hafner et al., 2025) scales to a 2B-parameter latent DiT and shares its backbone with the policy, but relies on large-scale offline video pretraining. These agents either train a separate policy network or share the backbone only after offline pretraining. DRIFT trains a *shared* Flow Transformer backbone *online from scratch* and reads its control state from the backbone's tokens.

Efficient inference in diffusion and flow models. Standard diffusion requires many denoising steps per sample. DDIM (Song et al., 2021) and rectified flows (Lipman et al., 2023; Liu et al., 2023) reduce the number of steps with better samplers and straighter paths, and consistency models (Song et al., 2023) and distribution matching distillation (Yin et al., 2024) reach a few steps, often by distilling a pretrained teacher. A fixed teacher is poorly matched to online RL, where the world model is still being learned from a changing replay buffer. Shortcut models (Frans et al., 2025) remove the need for a pretrained teacher by conditioning the network on the integration step size, thus learning to take large steps end-to-end. DRIFT adopts this objective for online imagination.

3 Background

Online model-based reinforcement learning. In a partially observable setting (Sutton & Barto, 2018), an agent receives image observations o_t and keeps an internal state s_t that summarizes the history $(o_1, a_1, \dots, o_t)$. It selects actions $a_t \sim \pi_\phi(\cdot \mid s_t)$ and receives a scalar reward r_{t+1} and a continuation flag $\kappa_{t+1} \in \{0, 1\}$, which is 0 when the episode ends at step $t{+}1$. The objective is to maximize expected discounted return $\mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t \big(\prod_{k=1}^{t} \kappa_k\big) r_{t+1}\Big]$, where $\gamma \in [0, 1)$ is the discount factor and the product stops the sum when the episode ends. A world model approximates the environment dynamics $p_\theta(o_{t+1}, r_{t+1}, \kappa_{t+1} \mid o_{\le t}, a_{\le t})$, enabling the agent to train its policy on imagined rollouts without additional environment interaction (Ha & Schmidhuber, 2018; Hafner et al., 2021). In an online setting (Sutton, 1991; Hafner et al., 2021), the policy collects data by acting in the real environment and stores the transitions in a replay buffer. World-model and policy updates are interleaved with this collection. The policy is learned with an actor–critic framework: the actor $\pi_\phi(a_t \mid s_t)$ selects actions, and the critic $V_\psi(s_t)$ estimates the expected future return, which tells the actor whether an action did better or worse than expected (training in Sec. 4.3).

Latent diffusion and flow Transformers. Latent diffusion models (Rombach et al., 2022) separate perceptual compression from generative modeling. A variational autoencoder (VAE) with encoder Enc_φ (Kingma & Welling, 2014) and decoder Dec_φ maps an observation o_t to a compact latent $z_t = \mathrm{Enc}_\varphi(o_t)$ and reconstructs $\hat{o}_t = \mathrm{Dec}_\varphi(z_t)$. The encoder is trained with a reconstruction loss, a KL penalty, and optionally a perceptual loss (Zhang et al., 2018). The VAE is typically pretrained and kept frozen while the generative model trains on its latents (Rombach et al., 2022). The generative model is typically a Diffusion Transformer (DiT) (Peebles & Xie, 2023), which denoises these latents. SiT (Ma et al., 2024b) trains the same architecture with the flow-matching objective described next.

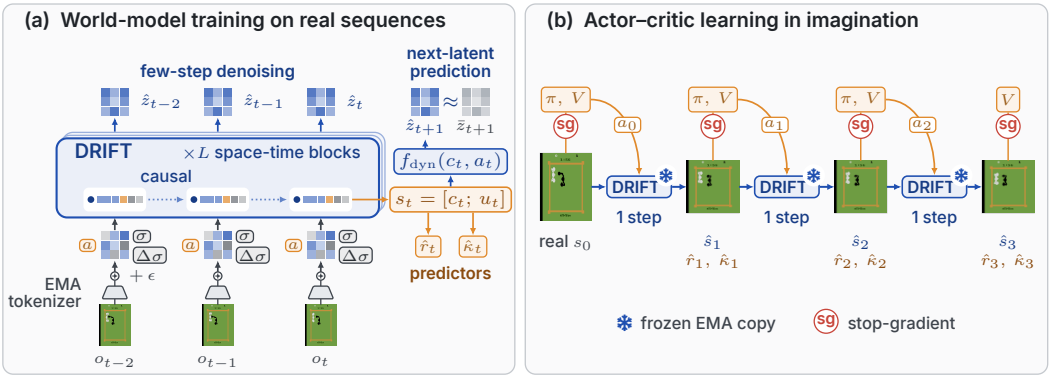

Figure 2: **Overview of DRIFT, a continuous-latent world-model agent.** (a) World-model training on real sequences. Each frame is encoded by the EMA tokenizer, noised, and passed to DRIFT as a group of tokens: CLS, latent patches, previous action, flow time, and shortcut step size, shown as the strip in each block; DRIFT attends causally over time. Three losses train the shared backbone: few-step denoising of every frame, reward and continuation prediction from the control state s_t, and next-latent prediction, which predicts the next frame's latent from the history summary and action. The reward, continuation, and denoising losses are also applied to the model's own rollouts. (b) Actor–critic learning in imagination. From a real state, the EMA world model generates each next state in one denoising step, driven by the policy's actions (frames decoded for illustration). The actor and critic read the same control state and learn from imagined rewards without updating DRIFT.

Flow matching and shortcut models. Flow matching (Lipman et al., 2023) learns a velocity field that transports noise into data along straight paths. Given a clean sample z and noise $\epsilon \sim \mathcal{N}(0, I)$, the corrupted sample $\tilde{z} = (1-\sigma)\,\epsilon + \sigma\, z$ lies on the straight path between them at flow time σ, where $\sigma{=}0$ is pure noise and $\sigma{=}1$ is clean data. The velocity along this path is $\mathrm{d}\tilde{z}/\mathrm{d}\sigma = z - \epsilon$, so a neural network v_θ is trained to predict it with $\mathcal{L}_{\mathrm{FM}} = \mathbb{E}_{\sigma,\, z,\, \epsilon}\big[\|v_\theta(\tilde{z}, \sigma) - (z - \epsilon)\|^2\big]$. Generation integrates this field from $\sigma{=}0$ to $\sigma{=}1$, but accurate integration requires many steps, making imagination rollouts expensive. Shortcut models (Frans et al., 2025) address this by also conditioning the network on the integration step size $\Delta\sigma$, giving $v_\theta(\tilde{z}, \sigma, \Delta\sigma)$. At small step sizes it learns the local velocity, as in flow matching, and at large ones an update that replaces several smaller steps. A self-consistency term trains one step of size $\Delta\sigma$ to match two consecutive steps of size $\Delta\sigma/2$, $\mathcal{L}_{\mathrm{sc}} = \mathbb{E}\big[\|v_\theta(\tilde{z}, \sigma, \Delta\sigma) - \mathrm{sg}(\bar{v})\|^2\big]$, where $\bar{v}$ is the average velocity of the two half steps taken by an EMA copy of the network and sg stops gradients. The combined shortcut flow-matching loss is $\mathcal{L}_{\mathrm{SFM}} = \mathcal{L}_{\mathrm{FM}} + \lambda_{\mathrm{sc}}\, \mathcal{L}_{\mathrm{sc}}$, where λ_{sc} weighs the self-consistency term (derivation in App. B.2).

4 DRIFT World Model Agent

We present DRIFT, an online continuous-latent model-based agent that learns to act from imagined experience (Fig. 2). The agent has two components that share a single Flow Transformer backbone: a world model and an actor–critic. The world model consists of a VAE tokenizer that compresses visual observations into continuous latents, a backbone that denoises future latents conditioned on actions, and a learned readout that extracts a compact state for predicting task outcomes such as rewards and episode continuations. The actor–critic reads the same compact state and learns from imagined rollouts. All components are trained online from scratch.

4.1 Architecture

Continuous-latent tokenizer. The VAE compresses each $64{\times}64$ observation into a continuous $16{\times}8{\times}8$ latent, rather than the discrete codes of token-based world models (Micheli et al., 2023). Latent diffusion models usually freeze a VAE pretrained on images of $256{\times}256$ pixels or more (Rombach et al., 2022), and such a VAE reconstructs $64{\times}64$ Atari frames poorly (17 dB PSNR against 56 dB for ours; Fig. 6). We therefore train the tokenizer online, which also lets the latent space adapt as the replay data changes. Online training, however, means that every encoder update moves the

backbone's targets, a known source of instability in joint encoder–generator training (Leng et al., 2025; Duggal et al., 2026). To keep the targets steady, the tokenizer maintains an exponential moving average (EMA) of its encoder, $\bar{z}_t = \mathrm{Enc}_{\bar{\varphi}}(o_t)$. The backbone reads its inputs and targets from this slowly changing copy, as with a target network (Grill et al., 2020; Schwarzer et al., 2021). World-model losses therefore do not update the tokenizer, which is trained only for reconstruction.

Flow Transformer backbone. The backbone is a causal Flow Transformer v_θ (Hafner et al., 2025) that reads the EMA latents $\bar{z}_t$, rescaled to unit variance by a batch-normalization layer. Each frame enters as a group of tokens $[\texttt{CLS}, z_t, a_{t-1}, \sigma, \Delta\sigma, \texttt{reg}]$: a CLS token that summarizes the history, the latent patches, the action that produced o_t, the flow time σ and shortcut step size $\Delta\sigma$ of the frame's latent (defined in Sec. 3), and register tokens, extra learnable tokens that give attention a place to store global information (Darcet et al., 2024). Attention is axial: layers alternate between attention within a frame and causal attention across frames, which is cheaper than attending over all tokens of all frames at once. Token packing and model sizes are given in App. B.1.

Control state and predictors. The predictors and the actor–critic read a control state $s_t = [c_t;\, u_t]$ taken directly from the backbone's tokens, without decoding back to pixels. The CLS output c_t summarizes the history up to time t (Dosovitskiy et al., 2021), but it misses spatial structure such as object positions. We add a spatial summary u_t, obtained by attention pooling g_ω with a learned query over the frame's spatial tokens. Three small MLP predictors with parameters ω read these features. A reward predictor and a continuation predictor estimate $\hat{r}_t$ and $\hat{\kappa}_t$ from s_t, and a next-latent predictor estimates the latent of the next frame from the CLS token and the current action, $\hat{z}_{t+1} = f_{\mathrm{dyn}}(c_t, a_t)$. The actor $\pi_\phi(a_t \mid s_t)$ and critic $V_\psi(s_t)$ also read s_t.

4.2 World-Model Training

Tokenizer objective. The VAE reconstructs real frames o_t as $\hat{o}_t = \mathrm{Dec}_\varphi(z_t)$ with $\mathcal{L}_{\mathrm{VAE}} = \|o_t - \hat{o}_t\|^2 + \lambda_{\mathrm{KL}}\,\mathrm{KL}\big(q_\varphi(z_t \mid o_t)\,\|\,\mathcal{N}(0, I)\big) + \lambda_{\mathrm{LPIPS}}\,\mathrm{LPIPS}(o_t, \hat{o}_t)$, where q_φ is the encoder posterior, LPIPS is a perceptual distance (Zhang et al., 2018), and the λ are weights (Tab. 6).

Shortcut denoising objective. The backbone is trained with the shortcut flow-matching loss $\mathcal{L}_{\mathrm{SFM}}$ (Sec. 3). The loss teaches it to recover the clean latent of each noised frame, and to make one large denoising step agree with two half steps so that a single step suffices at inference. It is applied to sequences of EMA latents from the replay buffer. An EMA copy of the backbone supplies the two half steps and generates all rollouts, which we call the EMA world model. Following diffusion forcing (Chen et al., 2024), each frame of a sequence gets its own noise level, so the backbone learns to generate a frame from context that is itself partly noisy, as it is during imagination. The network predicts the clean latent $\hat{z}$ rather than the velocity (Salimans & Ho, 2022; Hafner et al., 2025), and the velocity is recovered as $v = (\hat{z} - \tilde{z})/(1 - \sigma)$. The backbone also receives gradients from the reward, continuation, and next-latent objectives below, through s_t.

Reward and continuation objectives. The reward and continuation predictors are trained on the reward r_t and continuation flag κ_t stored in the replay buffer, where κ_t is 1 while the episode continues and 0 when it ends. The reward predictor classifies the sign of the reward over $\{-1, 0, +1\}$, $\mathcal{L}_r = -\log p_\omega(\mathrm{sign}(r_t) \mid s_t)$, as is standard for Atari agents (Mnih et al., 2015; Micheli et al., 2023; Alonso et al., 2024); in imagination the reward is sampled from this distribution. This bounds how much a reward error can change an imagined return. When rewards span several scales, it instead predicts $\mathrm{symlog}(r_t)$, with $\mathrm{symlog}(x) = \mathrm{sign}(x)\log(1+|x|)$, as a distribution over bins (Hafner et al., 2023). The continuation predictor uses binary cross-entropy, $\mathcal{L}_\kappa = \mathrm{BCE}(\hat{\kappa}_t, \kappa_t)$.

Next-latent objective. Denoising is conditioned on actions, but its loss acts on the latent-patch outputs, from which u_t is pooled. The history summary c_t in the control state gets this signal only indirectly, and the reward and continuation losses add just two scalars per step. We call this the *denoising–control representation gap*: DreamerV3's actor–critic reads a state trained to predict future dynamics (Hafner et al., 2023), whereas ours reads features learned mainly for denoising. To address it, the next-latent predictor estimates the EMA latent of the next real frame from c_t and a_t alone, with the reconstruction-free cosine loss $\mathcal{L}_{\mathrm{dyn}} = 1 - \cos(\hat{z}_{t+1},\, \bar{z}_{t+1})$ (Schwarzer et al., 2021). Its gradient reaches the backbone through c_t, so the history summary is directly trained to predict.

Self-generated rollout objectives. The reward and continuation predictors are trained on the latents of real frames, but during imagination they score continuous latents that the backbone generated. Discrete RSSM agents regularize their predicted and encoded states with a KL term (Hafner et al., 2021; 2023), while standard flow-matching objectives have no analogous constraint (Sec. 1), and the reward predictor is less accurate on generated latents than on real ones (Sec. 5.3). We address this *train–imagination distribution gap* with two objectives on the model's own rollouts, following self-forcing (Huang et al., 2025). From real context, the EMA world model generates a few latents under the real actions. The first objective, $\mathcal{L}_{\text{sf}}$, trains only the reward and continuation predictors on these generated latents, using the real rewards and continuations as targets. The second, $\mathcal{L}_{\text{sf-FM}}$, trains the backbone to denoise the next real latent with the generated latents as context (App. B.2).

Combined objective. The backbone (θ), readout, and predictors (ω) minimize

$$\mathcal{L}_{\text{world}} = \mathcal{L}_{\text{SFM}} + \lambda_r \mathcal{L}_r + \lambda_\kappa \mathcal{L}_\kappa + \lambda_{\text{dyn}} \mathcal{L}_{\text{dyn}} + \mathcal{L}_{\text{sf}} + \mathcal{L}_{\text{sf-FM}},$$

with loss weights λ (values in Tab. 6); the tokenizer minimizes $\mathcal{L}_{\text{VAE}}$ alongside.

4.3 Actor–Critic Learning in Imagination

The actor and critic learn only from imagined rollouts. Each rollout starts from the control state s_0 of a real replay sequence and runs for $H{=}15$ steps. At each step the actor samples an action $a_\ell \sim \pi_\phi(\cdot \mid s_\ell)$, the EMA world model generates the next latent in one shortcut step, from which the readout forms the next control state $s_{\ell+1}$ and the predictors estimate the reward $\hat{r}_{\ell+1}$ and continuation $\hat{\kappa}_{\ell+1}$. Because temporal attention is causal, the keys and values of past frames can be cached, and each imagined transition processes only the new frame. The critic learns to predict the λ-return (Sutton & Barto, 2018)

$$R^\lambda_\ell = \hat{r}_{\ell+1} + \gamma\, \hat{\kappa}_{\ell+1}\big((1-\lambda_{\text{td}})\, V_\psi(s_{\ell+1}) + \lambda_{\text{td}}\, R^\lambda_{\ell+1}\big), \qquad R^\lambda_H = V_\psi(s_H),$$

which mixes the imagined rewards with the critic's own estimates of later states, with $\lambda_{\text{td}} \in [0,1]$ setting the balance. The actor is trained with a policy gradient to make an action more likely when its return exceeds the critic's estimate, $\mathcal{L}_{\text{actor}} = -\,\text{sg}\,\big(R^\lambda_\ell - V_\psi(s_\ell)\big) \log \pi_\phi(a_\ell \mid s_\ell) - \eta\, \text{H}\big(\pi_\phi(\cdot \mid s_\ell)\big)$, where sg stops gradients and the entropy bonus, weighted by η, keeps the policy exploring. Both the actor–critic and the self-generated objectives use rollouts from the slowly changing EMA world model. The actor and critic read s_ℓ with gradients stopped, so only $\mathcal{L}_{\text{world}}$ shapes the backbone. As in DreamerV3 (Hafner et al., 2023), advantages are scaled by the range of recent returns (App. B.2).

5 Experiments

We evaluate DRIFT at the standard interaction budgets of three benchmark families: 100k environment steps for Atari and Craftium, and 1M for Crafter. Sec. 5.2 asks whether a shared backbone is competitive with agents that use discrete latents or separate policies. Sec. 5.3 examines how each modeling choice addresses its challenge from Sec. 1: the denoising–control representation gap, the train–imagination distribution gap, and the imagination cost.

5.1 Setup

Environments. (i) **Atari 100k** (Kaiser et al., 2019), all 26 games; (ii) four **Craftium** (Malagón et al., 2025) 3D voxel environments (SmallRoom, Room, Speleo, ChopTree) with procedural generation, partial observability, and richer textures; and (iii) **Crafter** (Hafner, 2022), an open-ended 2D survival game with sparse rewards and long-horizon credit assignment. All observations are $64{\times}64$ RGB, the standard resolution for world-model agents on these benchmarks.

Implementation details. DRIFT has 27.7M trainable parameters: 17.7M in the Flow Transformer (4 layers, width 512, 8 heads), 5.3M in the VAE, and 4.8M in the readout, predictors, actor, and critic. EMA copies and the frozen LPIPS network are not counted. All components are trained jointly from scratch on a single GPU. Training starts with 5k steps of random play. The tokenizer and world model then warm up in stages before online collection (schedule in App. B.2). All benchmarks share one architecture and hyperparameter set. We use sign rewards on Atari and Craftium. Crafter's rewards span two scales, and there we use full-scale rewards (DRIFT-symlog, Sec. 4.2). Both versions are reported on every benchmark. Architecture and hyperparameters are in Tabs. 5 and 6.

Table 1: **Atari 100k results on all 26 games.** Mean return per game; aggregates in HNS. Bold marks the best in each group. DreamerV3 aggregates as published; Tab. 7 recomputes them.

			Non-diffusion world models						Diffusion and flow world models		
Game *Imagines in*	**Random**	**Human**	**SimPLe** *pixels*	**TWM** *discrete*	**IRIS** *discrete*	**DreamerV3** *discrete*	**STORM** *discrete*	**TWISTER** *discrete*	**DIAMOND** *pixels*	**DRIFT-symlog** *continuous*	**DRIFT** *continuous*
Alien	227.8	7,127.7	616.9	674.6	420.0	959.0	**983.6**	970	**744.1**	703.9	594.8
Amidar	5.8	1,719.5	74.3	121.8	143.0	139.0	**204.8**	184	**225.8**	146.8	111.6
Assault	222.4	742.0	527.2	682.6	**1,524.4**	706.0	801.0	721	1,526.4	**2,013.0**	1,094.1
Asterix	210.0	8,503.3	1,128.3	1,116.6	853.6	932.0	1,028.0	**1,306**	**3,698.5**	1,484.4	1,398.1
BankHeist	14.2	753.1	34.2	466.7	53.1	649.0	641.2	**942**	19.7	790.9	**957.5**
BattleZone	2,360.0	37,187.5	4,031.2	5,068.0	13,074.0	12,250.0	**13,540.0**	9,920	4,702.0	17,482.0	**17,910.0**
Boxing	0.1	12.1	7.8	77.5	70.1	78.0	79.7	**88**	**86.9**	78.3	74.6
Breakout	1.7	30.5	16.4	20.0	**83.7**	31.0	15.9	35	132.5	**153.3**	149.1
ChopperCommand	811.0	7,387.8	979.4	1,697.4	1,565.0	420.0	**1,888.0**	910	1,369.8	**1,914.2**	1,892.2
CrazyClimber	10,780.5	35,829.4	62,583.6	71,820.4	59,324.2	**97,190.0**	66,776.0	81,880	**99,167.8**	79,998.4	80,790.0
DemonAttack	152.1	1,971.0	208.1	350.2	**2,034.4**	303.0	164.6	289	288.1	**916.3**	521.2
Freeway	0.0	29.6	16.7	24.3	31.1	0.0	**33.5**	32	**33.3**	0.0	6.4
Frostbite	65.2	4,334.7	236.9	**1,475.6**	259.1	909.0	1,316.0	305	274.1	287.6	**860.8**
Gopher	257.6	2,412.5	596.8	1,674.8	2,236.1	3,730.0	8,239.6	**22,234**	**5,897.9**	4,414.0	3,487.3
Hero	1,027.0	30,826.4	2,656.6	7,254.0	7,037.4	**11,161.0**	11,044.3	8,773	5,621.8	10,838.3	**11,719.6**
Jamesbond	29.0	302.8	100.5	362.4	462.7	445.0	509.0	**573**	427.4	**531.3**	511.0
Kangaroo	52.0	3,035.0	51.2	1,240.0	838.2	4,098.0	4,208.0	**6,016**	5,382.2	3,774.4	**6,204.0**
Krull	1,598.0	2,665.5	2,204.8	6,349.2	6,616.4	7,782.0	8,412.6	**8,839**	**8,610.1**	6,343.2	7,644.4
KungFuMaster	258.5	22,736.3	14,862.5	24,554.6	21,759.8	21,420.0	**26,182.0**	23,442	18,713.6	23,287.8	**23,607.6**
MsPacman	307.3	6,951.6	1,480.0	1,588.4	999.1	1,327.0	**2,673.5**	2,206	**1,958.2**	1,412.7	962.7
Pong	-20.7	14.6	12.8	18.8	14.6	18.0	11.3	**20**	**20.4**	18.4	18.6
PrivateEye	24.9	69,571.3	35.0	86.6	100.0	882.0	**7,781.0**	1,608	114.3	31.7	**2,426.8**
Qbert	163.9	13,455.0	1,288.8	3,330.8	745.7	3,405.0	**4,522.5**	3,197	**4,499.3**	2,945.9	2,172.0
RoadRunner	11.5	7,845.0	5,640.6	9,109.0	9,614.6	15,565.0	17,564.0	**17,832**	**20,673.2**	11,648.8	16,558.8
Seaquest	68.4	42,054.7	683.3	**774.4**	661.3	618.0	525.2	532	551.2	**645.0**	591.6
UpNDown	533.4	11,693.2	3,350.3	**15,981.7**	3,546.2	9,234.0	7,985.0	7,068	3,856.3	4,179.5	**7,152.9**
#Superhuman			1/26	8/26	10/26	9/26	10/26	**12/26**	11/26	**12/26**	**12/26**
Mean HNS			0.332	0.956	1.046	1.097	1.266	**1.620**	**1.459**	1.328	1.348
Median HNS			0.134	0.505	0.289	0.560	0.580	**0.770**	0.373	0.427	**0.520**
IQM HNS			0.130	0.459	0.501	0.497	**0.636**	–	0.641	0.620	**0.673**

Baselines and metrics. We compare against three families of model-based agents. In *latent dynamics* methods (DreamerV3 (Hafner et al., 2023), STORM (Zhang et al., 2023), TWM (Robine et al., 2023), TWISTER (Burchi & Timofte, 2025)), the policy reads the world model's latent state. *Discrete-token and pixel world models* (IRIS (Micheli et al., 2023), Δ-IRIS (Micheli et al., 2024), SimPLe (Kaiser et al., 2019)) and *diffusion/flow world models* (DIAMOND (Alonso et al., 2024), Horizon Imagination (Cohen et al., 2026)) instead train a separate policy network on the world model's rollouts. Detailed descriptions are in App. C.3. On Atari, we report the human-normalized score (HNS), on which 0 is random play and 1 is human level (Kaiser et al., 2019). Following Agarwal et al. (2021), we treat the interquartile mean (IQM) as the primary aggregate and report 95% stratified bootstrap confidence intervals (Tab. 7), since a few very high scores can dominate the mean. On Craftium we report mean return, and on Crafter the score, the geometric mean of achievement success rates (Hafner, 2022), and the return. All results use 5 seeds unless noted.

5.2 Main Results

Atari 100k. Tab. 1 lists per-game scores. Across all 26 games, DRIFT exceeds human performance on 12 games and achieves an IQM of 0.673. DRIFT has the highest score in Tab. 1 on BattleZone, BankHeist, Hero, and Kangaroo. On the first two, DIAMOND stays close to random.

Relative to DIAMOND, the closest diffusion world model, DRIFT has a higher median HNS (0.520 vs. 0.373) and IQM (0.673 vs. 0.641), while using one denoising step per imagined transition. On the four Atari games that Horizon Imagination reports, DRIFT also has higher mean and median HNS (App. D.4). The margin over DIAMOND is modest: DIAMOND retains a higher mean HNS (1.459 vs. 1.348), and DRIFT's 95% confidence intervals for median and IQM, although shifted higher, overlap with DIAMOND's (Tab. 7). Against latent and discrete world models, DRIFT remains competitive, although TWISTER achieves higher mean and median HNS (Tab. 1). TWISTER uses contrastive prediction on augmented future frames up to ten steps ahead, whereas DRIFT uses simpler one-step next-latent prediction. DRIFT trails the baselines most on Freeway, where most seeds never score, and on maze games such as MsPacman and Amidar, whose objects span only a few pixels at 64×64. Nevertheless, many learning curves still rise at 100k steps (Fig. 8). App. D.4 compares with search-based and model-free methods.

Table 2: **Craftium results.** Mean return at 100k steps.

Method	SmallRoom	Room	Speleo	ChopTree
DIAMOND	−172.6	−546.1	−2,047.0	7.3
Horizon Imag.	−40.9	−135.5	−339.6	13.8
DRIFT-symlog	**−18.6**	−649.6	669.8	**31.6**
DRIFT	−19.4	**−123.2**	**1,727.6**	15.3

Table 3: **Cost per imagined step.**

Method	Calls	GFLOPs	ms
DIAMOND	3	18.7	38.9
Horizon Imag.	0.5*	23.2	7.6*
DRIFT	2	**0.56**	11.6

A100, batch 1; *amortized.

Table 4: **Crafter results.** Mean score and return at 1M environment steps.

	Model-free		Non-diffusion				Diffusion and flow			
	PPO	Rainbow	DreamerV2	DreamerV3 M	DreamerV3 XL	Δ-IRIS	Horizon Imag.	DIAMOND	**DRIFT**	**DRIFT -symlog**
Score (%)	4.6	4.3	10.0	–	**14.5**	9.3	8.1	3.8	6.4	**10.6**
Return	4.2	6.0	9.0	6.2	**11.7**	7.7	7.3	4.3	5.6	**7.4**

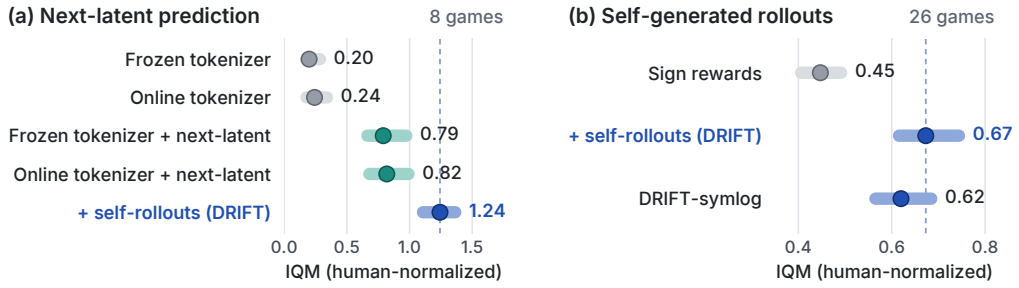

Figure 3: **Ablations.** IQM human-normalized score with 95% bootstrap CIs. (a) Online tokenizer and next-latent prediction on 8 games; rows other than DRIFT use no self-generated rollouts. (b) Self-generated rollouts and reward predictor on 26 games. Teal marks next-latent prediction and blue self-generated rollouts. Exact values are in Tab. 12.

Craftium and Crafter. On Craftium (Tab. 2), DRIFT has a higher mean return than both baselines on all four tasks, and on SmallRoom and Speleo every seed exceeds both baselines' means. In SmallRoom and Room every step is penalized, so the return measures how fast the agent reaches the target. In Speleo the reward tracks the agent's depth, and only DRIFT descends far enough for a positive return. On these three navigation tasks, DRIFT passes DIAMOND's final return within 12k steps and stays ahead (Fig. 9). On sparse-reward ChopTree, this takes about 50k steps.

On Crafter (Tab. 4), DRIFT-symlog achieves a score of 10.6%, the highest among diffusion and flow world models and second only to DreamerV3. Per-achievement success rates illustrate the effect of reward scale. With sign rewards, each ± 0.1 health change counts as much as a $+1$ achievement, so the return no longer separates progress from routine changes in health. Survival achievements such as drinking and sleeping barely change. Achievements that build on others drop: making a wood pickaxe falls from 79% to 45% and collecting stone from 52% to 15% (Tab. 9), lowering the score to 6.4%.

5.3 Analysis of Modeling Choices

Next-latent prediction improves control. To test whether next-latent prediction addresses the denoising–control representation gap, we ablate it with online tokenizer training on 8 Atari games of different types (App. C.1; Fig. 3a). With the tokenizer frozen after warmup, next-latent prediction raises IQM from 0.20 to 0.79, with non-overlapping 95% CIs. Training the tokenizer online changes it far less (0.20 to 0.24 without next-latent prediction and 0.79 to 0.82 with it), likely because Atari frames change little and the warmup tokenizer already covers most of them. This suggests that, in this setting, training the control state to predict matters more than adapting the tokenizer online. These agents use symlog rewards and no self-generated rollouts. The final DRIFT, with self-generated

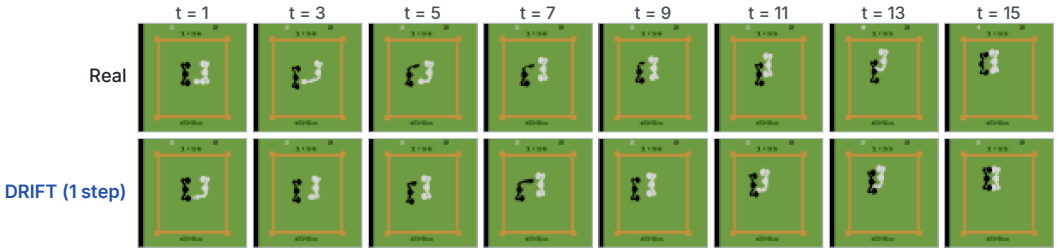

Figure 4: **One-step imagination on Boxing.** Real frames (top) and a 15-step DRIFT rollout with one denoising step per frame (bottom) under the same actions; median of 10 rollouts by pixel error.

rollouts and sign rewards, reaches 1.24. A simple extension to four-step prediction does not help; longer horizons may need a more careful design (App. E.2.2).

The control state becomes more predictive. We next examine how next-latent prediction changes what the control state s_t encodes, using two world models trained on the same data with and without this objective (App. E.2.1). With one seed per game, linear probes on held-out episodes of Boxing, Pong, Breakout and Hero show a higher explained variance (R^2) for AtariARI object positions (Anand et al., 2019) at the current step and four steps ahead (Tab. 13), from 0.64 to 0.74 four steps ahead on Boxing. Action-conditioned probes (Fig. 5) test whether s_t explains the controlled object's motion beyond the actions. Over three seeds, next-latent prediction raises this explained variance in all 12 game–seed pairs at both horizons (paired sign test, $p < 0.001$). Probes following the protocol of Maes et al. (2026) show the same trend. The gaps are smaller with nonlinear probes, suggesting that the objective makes this information more linearly accessible.

Self-generated rollouts improve policy learning. Figure 3b evaluates training on self-generated rollouts on all 26 Atari games. We first measure the train–imagination gap in these agents: under the same logged actions and reward targets, reward prediction loss is higher on generated than on real latents for 90% of game–seed pairs (median ratio 1.2; App. E.3). With sign rewards (Sec. 4.2), training on self-generated rollouts raises IQM from 0.45 to 0.67, with non-overlapping 95% confidence intervals. With symlog rewards, the same agent reaches 0.62.

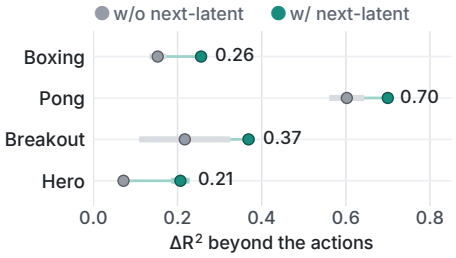

Figure 5: **Action-conditioned probes**, one step ahead (mean ± std, 3 seeds).

One denoising step suffices. DRIFT generates in a compact $16\times8\times8$ latent space with one shortcut step per imagined transition, using $33\times$ fewer FLOPs and achieving $3.4\times$ lower batch-one latency than DIAMOND (Tab. 3). Over a 15-step Boxing rollout, the generated frames track both boxers' positions, although brief punches are sometimes missed (Fig. 4). Increasing the number of denoising steps to two or four does not improve rollout accuracy on Boxing or Crafter (App. E.4.2), supporting one-step sampling for these environments. Horizon Imagination achieves lower amortized latency by denoising 32 future frames in parallel, but uses about $41\times$ more FLOPs per imagined frame (App. D.4).

6 Conclusion

We presented DRIFT, an online agent trained from scratch whose policy learns entirely in imagination on the same Flow Transformer backbone as its world model. Our experiments indicate that a generative world model's backbone can also provide the policy's representation when it is trained for how the policy uses it: next-latent prediction encourages the policy's state to be predictive, and self-generated rollouts expose the reward predictor to the latents it encounters in imagination. One-step shortcut imagination makes this sharing efficient enough for online RL. DRIFT requires no offline pretraining, and we hope the approach extends to world models pretrained on large-scale video, so that one backbone serves both imagination and control at scale. App. A discusses limitations.

AI USE STATEMENT

We used a large language model (LLM) assistant in this work. The authors implemented the method, and the assistant helped with parts of the code. It also wrote scripts for analysis and figure plotting from data provided by the authors, including the diagnostic probe analyses, confidence intervals and statistical tests. For writing, it was used primarily to polish the text and check grammar, including parts of the appendix draft. The authors manually reviewed and verified all AI-assisted code, analyses, figures and text, and take full responsibility for the content of this paper.

ETHICS STATEMENT

This work trains reinforcement learning agents in simulated video-game and voxel-world environments (Atari, Craftium and Crafter). It involves no human subjects, no personal data and no new dataset release. We see no direct pathway from these results to harmful application beyond the general dual-use considerations that apply to reinforcement learning research.

REPRODUCIBILITY STATEMENT

The paper and its appendix specify every component needed to reproduce our results. App. B gives the architecture, objectives and hyperparameters, Tabs. 5 and 6 list every setting, and App. B.3 gives the training and imagination procedures in full. App. C documents the environment configurations, the evaluation protocol and the compute used for each run. Main results are means over 5 seeds per task and the probes use 3 seeds. Aggregate statistics are computed from human-normalized scores as defined in App. C.2. We will release the code, trained models and evaluation scripts upon acceptance.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, 2021.

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

Ankesh Anand, Evan Racah, Sherjil Ozair, Yoshua Bengio, Marc-Alexandre Côté, and R Devon Hjelm. Unsupervised state representation learning in atari. In *Advances in Neural Information Processing Systems*, 2019.

Maxime Burchi and Radu Timofte. Learning transformer-based world models with contrastive predictive coding. In *International Conference on Learning Representations*, 2025.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024.

Lior Cohen, Ofir Nabati, Kaixin Wang, Navdeep Kumar, and Shie Mannor. Horizon imagination: Efficient on-policy rollout in diffusion world models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Obefq4k8iG.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *International Conference on Learning Representations*, 2024.

Antoine Dedieu, Joseph Ortiz, Xinghua Lou, Carter Wendelken, Wolfgang Lehrach, J. Swaroop Guntupati, Miguel Lazaro-Gredilla, and Kevin Patrick Murphy. Improving transformer world models for data-efficient RL. In *International Conference on Machine Learning (ICML)*, 2025.

Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. In *International Conference on Machine Learning (ICML)*, 2023.

Pierluca D'Oro, Max Schwarzer, Evgenii Nikishin, Pierre-Luc Bacon, Marc G Bellemare, and Aaron Courville. Sample-efficient reinforcement learning by breaking the replay ratio barrier. In *International Conference on Learning Representations*, 2023.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

Shivam Duggal, Xingjian Bai, Zongze Wu, Richard Zhang, Eli Shechtman, Antonio Torralba, Phillip Isola, and William T. Freeman. End-to-end training for unified tokenization and latent denoising. *arXiv preprint arXiv:2603.22283*, 2026.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *International Conference on Learning Representations*, 2025.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, 2020.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2(3), 2018.

Danijar Hafner. Benchmarking the spectrum of agent capabilities. In *International Conference on Learning Representations (ICLR)*, 2022.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. *International conference on machine learning*, pp. 2555–2565, 2019.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *International Conference on Learning Representations*, 2021.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Danijar Hafner, Wilson Yan, and Timothy Lillicrap. Training agents inside of scalable world models. *arXiv preprint arXiv:2509.24527*, 2025.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *Advances in Neural Information Processing Systems*, 2025.

Łukasz Kaiser, Mohammad Babaeizadeh, Piotr Miłos, Błażej Osiński, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, et al. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*, 2019.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations (ICLR)*, 2014.

Michael Laskin, Aravind Srinivas, and Pieter Abbeel. CURL: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, 2020.

Xingjian Leng, Jaskirat Singh, Yunzhong Hou, Zhenchang Xing, Saining Xie, and Liang Zheng. REPA-E: Unlocking VAE for end-to-end tuning with latent diffusion transformers. In *International Conference on Computer Vision*, 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations (ICLR)*, 2023.

Haoyu Ma, Jialong Wu, Ningya Feng, Chenjun Xiao, Dong Li, Jianye Hao, Jianmin Wang, and Mingsheng Long. HarmonyDream: Task harmonization inside world models. In *International Conference on Machine Learning*, 2024a.

Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, 2024b.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. LeWorldModel: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv preprint arXiv:2603.19312*, 2026.

Mikel Malagón, Josu Ceberio, and Jose A. Lozano. Craftium: Bridging flexibility and efficiency for rich 3d single- and multi-agent environments. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025.

Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models. In *International Conference on Learning Representations*, 2023.

Vincent Micheli, Eloi Alonso, and François Fleuret. Efficient world models with context-aware tokenization. In *International Conference on Machine Learning*, 2024.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

Jan Robine, Marc Höftmann, Tobias Uelwer, and Stefan Harmeling. Transformer-based world models are happy with 100k interactions. In *International Conference on Learning Representations*, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588 (7839):604–609, 2020.

Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. In *International Conference on Learning Representations*, 2021.

Max Schwarzer, Johan Samir Obando-Ceron, Aaron Courville, Marc G Bellemare, Rishabh Agarwal, and Pablo Samuel Castro. Bigger, better, faster: Human-level Atari with human-level efficiency. In *International Conference on Machine Learning*, 2023.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2021.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, 2023.

Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin*, 2(4):160–163, 1991.

Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2018.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Weirui Ye, Shaohuai Liu, Thanard Kurutach, Pieter Abbeel, and Yang Gao. Mastering Atari games with limited data. In *Advances in Neural Information Processing Systems*, 2021.

Jewon Yeom, Hanseul Kim, Jeongjae Park, Sungmok Jung, Jaejin Lee, and Taesup Kim. What makes video world model latents action-relevant: Prediction over reconstruction. *arXiv preprint arXiv:2606.07687*, 2026.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6613–6623, 2024.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Weipu Zhang, Gang Wang, Jian Sun, Yetian Yuan, and Gao Huang. Storm: Efficient stochastic transformer based world models for reinforcement learning. *Advances in Neural Information Processing Systems*, 36:27147–27166, 2023.

App. A discusses limitations and future work. The other sections follow the structure of the main text. App. B details the implementation and App. C the experimental setup and baselines. App. D extends the results of Sec. 5.2, and App. E gives the evidence behind each modeling choice of Sec. 5.3.

A Limitations and Future Work

Given our compute budget (App. C.4), we focus on low-resolution visual control with discrete actions (Atari, Craftium, Crafter). Higher-resolution observations and continuous control remain to be tested. The next-latent objective looks one step ahead at a VAE latent trained for reconstruction, which may miss longer-horizon or task-relevant structure. Training on self-generated rollouts is a simple mitigation of the train–imagination distribution gap. Sign rewards bound the effect of reward errors but discard the reward magnitude that Crafter needs, so neither reward predictor is best on every benchmark.

The most promising direction is to start from pretrained video world models, whose backbones could serve as shared control representations at scale. The next-latent target could also become native to the model by predicting the backbone's own representation of future frames, as in self-predictive and joint-embedding learning (Grill et al., 2020; Schwarzer et al., 2021; Maes et al., 2026). An action-conditioned contrastive objective over longer horizons, as in TWISTER (Burchi & Timofte, 2025), would extend this prediction further. For the train–imagination gap, more advanced approaches, such as objectives that keep generated latents close to encoded ones, could make imagined returns more reliable. Joint multi-frame generation could also reduce compounding rollout error.

Table 5: **Model configuration.** Architecture of the VAE tokenizer, the Flow Transformer backbone, and the MLPs of the readout, predictors, actor, and critic, shared across all experiments.

Component	Setting
VAE encoder and decoder (5.3M trainable)	
Latent shape (C×H×W)	$16 \times 8 \times 8$
Base channels	32
Channel multipliers	[1, 2, 4, 4]
Residual blocks / stage	2
Flow Transformer backbone	
Parameters	17.7M
Hidden dimension	512
Layers	4
Attention heads	8
Head dimension	64
Packing factor	16
Register tokens	2
Temporal attention	every 2 layers
Readout, predictors, actor, critic (4.8M total)	
Hidden dimension	512
MLP layers	2
Total trainable parameters: 27.7M	

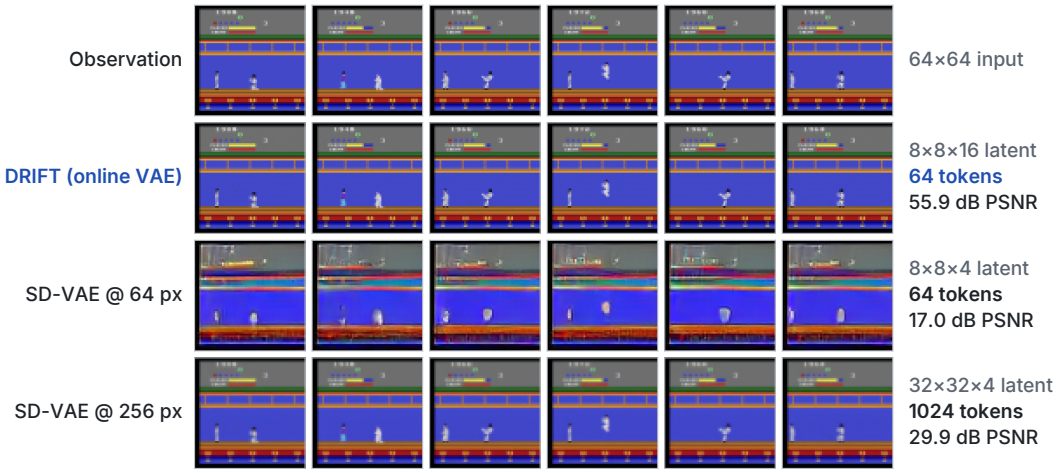

Figure 6: **Frozen versus online tokenizer.** A frozen Stable Diffusion VAE reconstructs 64×64 Atari frames poorly (row 3, 17 dB PSNR); upscaling to 256×256 recovers quality (row 4, 30 dB) but enlarges the latent grid from 8×8 to 32×32, 16 times as many tokens. The online VAE (row 2) reconstructs almost losslessly (56 dB) with 64 tokens per frame.

B Implementation Details

B.1 Architecture

VAE tokenizer. The encoder is a convolutional network with residual blocks and no self-attention, compressing 64×64 RGB observations with $8\times$ downsampling. The decoder mirrors the encoder. Tab. 5 gives the sizes of all components.

Flow Transformer backbone. The backbone follows the axial attention design of Hafner et al. (2025). A learned linear projection packs adjacent latent tokens. QK-RMSNorm (Dehghani et al., 2023) is applied to queries and keys. A learned linear layer projects one-hot-encoded discrete actions to the model dimension.

Control-state readout. The spatial summary u_t of the control state comes from single-query attention pooling: a learnable query of dimension d attends to projected spatial keys and values, followed by RMSNorm.

Next-latent predictor. The next-latent predictor f_{dyn} is a 3-layer MLP (hidden dimension 512, LayerNorm + SiLU activations) that maps $[c_t;\, a_t]$ to $\hat{z}_{t+1}$, a vector of the flattened latent size.

Predictors, actor, and critic. The reward and continuation predictors, the actor, and the critic are MLPs with ReLU activations. The critic's output depends on the reward predictor (App. B.2).

B.2 Training

Tab. 6 lists all hyperparameters.

Shortcut objective. The Flow Transformer backbone is trained as a shortcut model (Frans et al., 2025) extended with action conditioning, per-frame independent noise levels (diffusion forcing), and x-prediction. Let $z_t = \mathrm{Enc}_{\bar{\varphi}}(o_t)$ denote the EMA-tokenized latent target and $\epsilon_t \sim \mathcal{N}(0, I)$. For each future frame, we sample an independent flow time σ_t and form the interpolation

$$\tilde{z}_{t,\sigma} = (1 - \sigma_t)\epsilon_t + \sigma_t z_t.$$

Given the noisy latent sequence, actions, flow times σ_t, and step sizes $\Delta\sigma_t$, the transformer predicts clean latents $\hat{z}_{t,\theta}$. This x-prediction (Sec. 4.2) reparameterizes the velocity-based $\mathcal{L}_{\mathrm{FM}}$ from Sec. 3. Since $\hat{v} - v = (\hat{z} - z)/(1-\sigma)$, the x-space loss equals the velocity loss weighted by $(1-\sigma)^2$; we add the ramp weighting of Hafner et al. (2025). The empirical loss uses the finest step size, $d_{\min} = 1/64$:

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}_{t,\sigma,\epsilon}\left[w(\sigma_t)\,\|\hat{z}_{t,\theta} - z_t\|_2^2\right], \qquad w(\sigma) = 0.9\sigma + 0.1.$$

The ramp upweights cleaner noise levels, where perceptual detail matters most.

For self-consistency examples ($\Delta\sigma > d_{\min}$), we convert x-predictions to velocities,

$$v_\theta(\tilde{z}_\sigma, \sigma, \Delta\sigma) = \frac{\hat{z}_\theta(\tilde{z}_\sigma, \sigma, \Delta\sigma) - \tilde{z}_\sigma}{1 - \sigma},$$

and train one large step to match two EMA half-steps. With $\bar{\theta}$ the EMA dynamics model,

$$z_{\mathrm{mid}} = \tilde{z}_\sigma + \frac{\Delta\sigma}{2}\, v_{\bar{\theta}}\left(\tilde{z}_\sigma, \sigma, \frac{\Delta\sigma}{2}\right),$$

$$z_{\mathrm{two}} = z_{\mathrm{mid}} + \frac{\Delta\sigma}{2}\, v_{\bar{\theta}}\left(z_{\mathrm{mid}}, \sigma + \frac{\Delta\sigma}{2}, \frac{\Delta\sigma}{2}\right), \qquad \bar{v} = \frac{z_{\mathrm{two}} - \tilde{z}_\sigma}{\Delta\sigma}.$$

The self-consistency loss compares velocities, and its factor $(1-\sigma_t)^2$ puts it on the same x-space scale as the empirical loss, as in Hafner et al. (2025):

$$\mathcal{L}_{\mathrm{sc}} = \mathbb{E}_{t,\sigma,\epsilon,\Delta\sigma}\left[w(\sigma_t)\,(1-\sigma_t)^2\,\|v_\theta(\tilde{z}_{t,\sigma}, \sigma_t, \Delta\sigma_t) - \mathrm{sg}(\bar{v}_t)\|_2^2\right],$$

where sg stops gradients through the EMA target. We set $\lambda_{\mathrm{sc}} = 1$, and a quarter of each batch carries self-consistency targets (Tab. 6).

Noise-level sampling. Noise levels come from a discrete set rather than continuous $\mathcal{U}(0,1)$. A level index j is drawn uniformly from $\{0, \ldots, \log_2 64\}$, setting a stride $S = 2^j$; a position within that stride is then drawn uniformly, yielding $\sigma_t \in \{0, 1/64, 2/64, \ldots, 63/64\}$. Each step size $\Delta\sigma = 1/S$ used in shortcut training is therefore sampled equally often.

Self-generated rollouts. Each world-model batch draws up to 4 anchor frames τ per sequence (stride 8), each with 8 real context frames. From each anchor, the EMA world model generates $\hat{z}_{\tau+1}, \ldots, \hat{z}_{\tau+k}$ (k=4) with one shortcut step per latent, under the actions recorded in the replay buffer. With $\hat{s}_{\tau+d}$ the control state of the d-th generated latent,

$$\mathcal{L}_{\mathrm{sf}} = \frac{1}{\sum_d \rho^{d-1}} \sum_{d=1}^{k} \rho^{d-1}\big[\lambda_r\, \ell_r(\hat{s}_{\tau+d}, r_{\tau+d}) + \lambda_\kappa\, \ell_\kappa(\hat{s}_{\tau+d}, \kappa_{\tau+d})\big],$$

Table 6: **Training hyperparameters.** Shared across all experiments. The optimizer is AdamW with $(\beta_1, \beta_2) = (0.9, 0.999)$.

Hyperparameter	Value
Optimization	
Learning rate (tokenizer, world model)	10^{-4}
Learning rate (actor, critic)	3×10^{-5}
Batch size	24
Sequence length	64
Mixed precision	bf16
Gradient clipping	100.0
World model	
Flow-matching noise levels	64
Denoising steps per imagined frame (K)	1
Fraction of batch with self-consistency targets	0.25
Imagination context frames	8
Imagination horizon H	15
Ramp loss weight $w(\sigma)$	$0.9\sigma + 0.1$
Self-generated rollout depth k / decay ρ	4 / 0.7
Self-generated rollout anchors per sequence	4 (stride 8)
Loss weights	
λ_r (reward)	1.0
λ_κ (continuation)	1.0
λ_{dyn} (next-latent prediction)	0.3
λ_{KL} (VAE)	10^{-6}
λ_{LPIPS} (perceptual)	0.25
RL	
Discount γ	0.997
TD(λ)	0.95
Entropy coefficient	3×10^{-4}
Return normalization	5th–95th percentile
Value bins (symlog)	255
EMA	
Tokenizer EMA decay	0.999
World model EMA decay	0.999
Data collection	
Prefill steps	5,000
Steps collected per epoch	25
Gradient steps per epoch (tokenizer / world model / actor–critic)	50 / 50 / 25
Warmup epochs (tokenizer / world model / actor–critic)	30 / 65 / 5

where ℓ_r and ℓ_κ are the reward and continuation losses of Sec. 4.2 and ρ=0.7 down-weights later frames, which drift further from the data. The rollout runs without gradients, so $\mathcal{L}_{\text{sf}}$ updates only the reward and continuation predictors. $\mathcal{L}_{\text{sf-FM}}$ is the denoising loss on the last position of the spliced sequence $[z_{\le\tau};\, \hat{z}_{\tau+1}, \ldots, \hat{z}_{\tau+k-1};\, z_{\tau+k}]$ and updates the dynamics model. Targets that cross an episode boundary are masked.

Actor–critic objectives. Algorithm 2 generates the imagined rollouts with the EMA world model $\bar{\theta}$, without gradients through denoising. The anchor state s_0 is the control-state readout of the last real context frame. The actor–critic follows DreamerV3 (Hafner et al., 2023), as do its Transformer variants (Zhang et al., 2023; Burchi & Timofte, 2025). It uses TD(λ) returns (Sutton & Barto, 2018), REINFORCE with an entropy bonus (Williams, 1992), and percentile-scaled advantages. With v_ℓ the critic's mean value, the returns are

$$R^\lambda_\ell = \hat{r}_{\ell+1} + \gamma\,\hat{\kappa}_{\ell+1}\Big((1-\lambda_{\text{td}})\,v_{\ell+1} + \lambda_{\text{td}}\,R^\lambda_{\ell+1}\Big), \qquad R^\lambda_H = v_H.$$

With symlog rewards, the critic is DreamerV3's categorical critic over symlog-spaced bins. It is trained with a symlog cross-entropy toward the returns and toward its own EMA ψ':

$$\mathcal{L}_{\text{critic}}(\psi) = \frac{1}{BH}\sum_{b,\ell} w_\ell\Big[\,\text{SymlogCE}\big(V_\psi(s_\ell), R^\lambda_\ell\big) + \text{SymlogCE}\big(V_\psi(s_\ell), V_{\psi'}(s_\ell)\big)\Big],$$

where $w_\ell = \hat{\kappa}_0 \prod_{i=1}^{\ell} \gamma\,\hat{\kappa}_i$ discounts steps after a predicted episode end, and the EMA follows $\psi' \leftarrow 0.98\,\psi' + 0.02\,\psi$. With sign rewards, returns are bounded. The critic then outputs a scalar regressed onto R^λ_ℓ with a squared error and no EMA regularizer, as in DIAMOND (Alonso et al., 2024). The advantages are $A^\lambda_\ell = (R^\lambda_\ell - v_\ell)/\max(1, S)$, where $S = \text{EMA}\,\big(\text{Per}(R^\lambda, 95) - \text{Per}(R^\lambda, 5),\, 0.99\big)$ tracks the spread of returns. The actor minimizes

$$\mathcal{L}_{\text{actor}}(\phi) = \frac{1}{BH}\sum_{b,\ell} w_\ell\Big[-\,\text{sg}\big(A^\lambda_\ell\big)\,\log \pi_\phi\big(a_\ell \mid s_\ell\big) - \eta\,\text{H}\big(\pi_\phi(\cdot \mid s_\ell)\big)\Big].$$

Replay buffer. The replay buffer stores complete trajectories (observations, actions, rewards, continuation flags) in a FIFO queue with 100k-step capacity and samples them uniformly at random.

Warmup schedule. Warmup training on the random-policy prefill buffer is staged over 30 tokenizer, 65 world-model, and 5 actor–critic epochs (Tab. 6). The staging lets the tokenizer converge before the world model trains on its latents, and the world model stabilize before the policy shapes the data distribution.

Training dynamics. The flow-matching loss rises after warmup, when online data makes the replay buffer more diverse than the random prefill, and then decreases, while the reconstruction loss decreases throughout. On the smoothed curves of Fig. 7, the reconstruction loss ends below its initial value for 129 of the 130 game–seed pairs, and the flow-matching loss ends more than 20% below its peak for 106 of them.

B.3 Pseudocode

Algorithms 1 and 2 give the full training and imagination procedures. All quantitative experiments use one shortcut step per imagined transition (K=1).

C Experimental Details

C.1 Environments

Atari 100k. We follow the standard Atari 100k protocol (Kaiser et al., 2019): 100k environment steps (400k frames with frame skip 4), 64×64 RGB observations, up to 30 no-op actions at episode start, and fire-on-reset for games that require it. We do not terminate on life loss. Each game uses its minimal action set from the Arcade Learning Environment. The 8 ablation games are Alien and Gopher (maze and defense), Boxing and KungFuMaster (combat), ChopperCommand (shooting), CrazyClimber (climbing), Hero (platform), and RoadRunner (chase).

Algorithm 1 DRIFT training

Parameters: φ (tokenizer), θ (dynamics backbone), ω (readout, reward, continuation and next-latent predictors), ϕ (actor), ψ (critic)

procedure TRAINING
  Collect 5k steps with random policy into $\mathcal{D}$
  for warmup epoch **do** ▷ staged: tokenizer, then world model (Tab. 6)
    UPDATETOKENIZER
    UPDATEWORLDMODEL
  end for
  for epoch $= 1, 2, \ldots$ **do**
    Collect 25 steps with π_ϕ, store in $\mathcal{D}$
    $50 \times$ UPDATETOKENIZER
    $50 \times$ UPDATEWORLDMODEL
    $25 \times$ UPDATEACTORCRITIC
  end for
end procedure

procedure UPDATETOKENIZER
  Sample $\{o_t\} \sim \mathcal{D}$
  $z_t = \mathrm{Enc}_\varphi(o_t), \quad \hat{o}_t = \mathrm{Dec}_\varphi(z_t)$
  Update φ with $\mathcal{L}_{\mathrm{recon}} + \lambda_{\mathrm{KL}}\mathcal{L}_{\mathrm{KL}} + \lambda_{\mathrm{LPIPS}}\mathcal{L}_{\mathrm{LPIPS}}$
  $\bar{\varphi} \leftarrow \tau\bar{\varphi} + (1-\tau)\varphi$ ▷ EMA tokenizer, after every update
end procedure

procedure UPDATEWORLDMODEL
  Sample $\{(o_t, a_t, r_t, \kappa_t)\}_{t=1}^{L} \sim \mathcal{D}$
  $\bar{z}_t = \mathrm{Enc}_{\bar{\varphi}}(o_t)$ ▷ EMA encoder, no gradient
  Sample σ_t and $\Delta\sigma_t$ per frame from the discrete grid (App. B.2), $\epsilon \sim \mathcal{N}(0, I)$
  $\tilde{z}_t = (1-\sigma_t)\epsilon + \sigma_t \bar{z}_t$ ▷ noisy latent
  $\hat{z}_t, s_t = f_\theta(\tilde{z}_t, \sigma_t, \Delta\sigma_t, a_{t-1})$ ▷ clean latent and control state
  Predict $\hat{r}_t, \hat{\kappa}_t$ from s_t with the reward and continuation predictors, $\hat{z}_{t+1}^{\mathrm{dyn}} = f_{\mathrm{dyn}}(c_t, a_t)$
  $\mathcal{L}_{\mathrm{SFM}}$ from $\mathcal{L}_{\mathrm{FM}}$ and $\mathcal{L}_{\mathrm{sc}}$ ▷ Sec. 3
  Roll out k frames with $f_{\bar{\theta}}$ under the replayed actions (no gradient); compute $\mathcal{L}_{\mathrm{sf}}$, $\mathcal{L}_{\text{sf-FM}}$
  Update θ, ω with $\mathcal{L}_{\mathrm{world}}$ ▷ Sec. 4.2
  $\bar{\theta} \leftarrow \tau\bar{\theta} + (1-\tau)\theta$ ▷ EMA world model, after every update
end procedure

procedure UPDATEACTORCRITIC
  Sample anchor states from $\mathcal{D}$
  $\{s_\ell, a_\ell, \hat{r}_\ell, \hat{\kappa}_\ell\}_{\ell=0}^{H} \leftarrow$ IMAGINE(anchor states) ▷ Alg. 2
  Compute λ-returns R_ℓ^λ with TD(λ) ▷ App. B.2
  Update ϕ with $\mathcal{L}_{\mathrm{actor}}$ ▷ App. B.2
  Update ψ with $\mathcal{L}_{\mathrm{critic}}$, and ψ' by EMA (symlog rewards only) ▷ App. B.2
end procedure

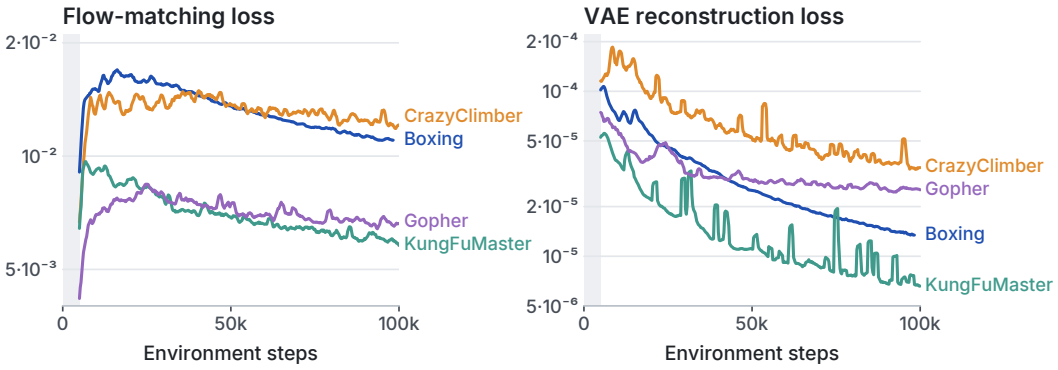

Figure 7: **Training dynamics.** Flow-matching loss (left) and VAE reconstruction loss (right) during online training on four Atari games, averaged over 5 seeds and smoothed with a 51-epoch rolling mean. Shaded regions mark the random prefill and warmup.

Algorithm 2 Imagination rollout

Require: Real context latents, EMA world model $f_{\bar{\theta}}$, readout g_ω, policy π_ϕ, horizon H, denoising steps K

Ensure: Trajectory $\{(s_\ell, a_\ell, \hat{r}_\ell, \hat{\kappa}_\ell)\}_{\ell=0}^{H}$

procedure IMAGINE(context)
  Encode the context frames, interpolated 10% toward noise, into the KV cache; read s_0 at the last frame of a clean pass over the context
  for $\ell = 0, \ldots, H-1$ **do**
    $a_\ell \sim \pi_\phi(\cdot \mid s_\ell)$
    $z^{(0)} \sim \mathcal{N}(0, I)$ ▷ start from noise (σ=0)
    for $j = 0, \ldots, K-1$ **do** ▷ shortcut steps (K=1 by default)
      $\sigma = j/K, \quad \Delta\sigma = 1/K$
      $\hat{z} = f_{\bar{\theta}}(z^{(j)}, \sigma, \Delta\sigma, a_\ell)$ ▷ x-prediction
      $z^{(j+1)} = z^{(j)} + \Delta\sigma \cdot \frac{\hat{z} - z^{(j)}}{1-\sigma}$ ▷ velocity step
    end for
    Re-encode $z^{(K)}$ at the clean noise level; append its keys and values to the KV cache
    $s_{\ell+1} = g_\omega(\text{tokens}_{\ell+1})$ ▷ readout from the re-encoded frame
    Predict $\hat{r}_{\ell+1}, \hat{\kappa}_{\ell+1}$ from $s_{\ell+1}$
  end for
end procedure

Craftium. Craftium (Malagón et al., 2025) is a suite of 3D voxel environments rendered at 64×64. We evaluate four tasks of increasing complexity: SmallRoom and Room (4 actions: noop, forward, turn left/right; dense -1 per-step penalty, goal-reaching), Speleo (7 actions adding jump and vertical look; reward proportional to descent depth), and ChopTree (8 actions adding dig; sparse $+1$ per tree chopped). Frame skip is 4 and episodes last up to 2000 steps.

Crafter. Crafter (Hafner, 2022) is an open-ended 2D survival game with procedurally generated terrain, 17 discrete actions (movement, crafting, combat, and interaction), and 22 achievement types. Rewards are sparse: $+1$ for each new achievement unlocked during an episode, plus a small health-change signal (± 0.1 per health point). Observations are 64×64 RGB with no frame skip. Episodes last up to 10000 steps. We train for the standard 1M environment steps (Hafner, 2022), the budget at which the baselines in Tab. 4 are reported.

C.2 Evaluation Protocol

We evaluate every 200 training epochs on Atari and every 100 epochs on Crafter. Actions are sampled from the policy distribution. Final results use 100 evaluation episodes per seed on Atari and Crafter and 20 on Craftium. Intermediate evaluations use 10 episodes. The human-normalized score (HNS) is

$$\text{HNS} = \frac{\text{agent score} - \text{random score}}{\text{human score} - \text{random score}}$$

Human and random reference scores follow the Atari 100k protocol of Kaiser et al. (2019). For Craftium and Crafter, we report raw episode returns since no standardized human scores exist.

C.3 Baselines

Latent dynamics models predict the next latent state in a single forward pass, which makes rollouts fast.

DreamerV3 (Hafner et al., 2023) learns a recurrent state-space model (RSSM), and its policy and value function read this latent state. DRIFT adopts its percentile return normalization, and DRIFT-symlog also its symlog predictions. On Atari 100k we report the S model of DreamerV3 (18M parameters), and on Crafter its M (37M) and XL (200M) models.

STORM (Zhang et al., 2023) replaces the RSSM recurrence with a stochastic Transformer, and its policy reads the latent state.

TWM (Robine et al., 2023) models categorical latents, actions, and rewards as separate tokens of a Transformer-XL, and its policy reads the latent state.

TWISTER (Burchi & Timofte, 2025) is a Transformer world model with categorical latents that adds action-conditioned contrastive predictive coding over ten future steps to improve its representation.

We also include discrete-token methods for reference. **SimPLe** (Kaiser et al., 2019) uses a stochastic video prediction model with PPO. **IRIS** (Micheli et al., 2023) tokenizes observations with a VQ-VAE and predicts tokens with a GPT-style transformer. **Δ-IRIS** (Micheli et al., 2024) encodes each frame as a few tokens describing its change from the previous frames.

Diffusion and flow world models generate next frames through iterative denoising, which preserves visual detail but typically takes multiple forward passes per transition.

DIAMOND (Alonso et al., 2024) is a pixel-space EDM diffusion world model (Karras et al., 2022) whose U-Net denoiser predicts the next frame from a four-frame context. A separate CNN–LSTM policy acts on the generated images.

Horizon Imagination (Cohen et al., 2026) uses a pretrained video tokenizer and a rectified-flow transformer world model. After a short context, it denoises the remaining horizon jointly with a horizon-aware sampling schedule. A separate lightweight controller trains on these imagined rollouts.

On Atari 100k, we take the scores and aggregates of SimPLe, TWM, IRIS, DreamerV3, STORM, and DIAMOND from Alonso et al. (2024) and those of TWISTER from Burchi & Timofte (2025). Medians that these sources do not report are computed from the per-game scores, and for DIAMOND

from its released per-seed results. On Craftium and Crafter, we train and evaluate DIAMOND and Horizon Imagination ourselves with the authors' code. The other Crafter baselines are taken from Hafner (2022); Hafner et al. (2023), and those for Δ-IRIS and DreamerV3-M from Micheli et al. (2024), which reports only the return for DreamerV3-M. The original DIAMOND evaluation did not record Crafter achievements, so we evaluate its five final trained models for 100 episodes each. The resulting return, 4.45 ± 0.98, matches the reported 4.34 ± 0.99.

C.4 Compute

Training one agent on one Atari 100k game takes about 4 days on a single NVIDIA A100 40GB GPU.

D Extended Results

D.1 Atari 100k Confidence Intervals

The 95% stratified bootstrap confidence intervals (Agarwal et al., 2021) of DRIFT and DIAMOND overlap on every aggregate of Tab. 1 (Tab. 7). Against DreamerV3, whose intervals we compute from its released per-seed scores, the IQM intervals do not overlap (0.673 [0.602, 0.757] against 0.471 [0.424, 0.525]). The STORM interval in Fig. 1 (0.636 [0.586, 0.690]) is read from the aggregate plot of Alonso et al. (2024), whose DIAMOND interval matches ours to within 0.002.

Table 7: **Atari 100k confidence intervals.** Aggregate human-normalized scores with 95% stratified bootstrap confidence intervals over 26 games and 5 seeds. Only methods with published per-seed scores are included; DreamerV3† is computed from its released per-seed learning curves, scoring each seed by its mean evaluation return over the last 10% of training.

Method	Mean HNS	Median HNS	IQM HNS
DreamerV3†	1.183 [1.079, 1.321]	0.437 [0.383, 0.503]	0.471 [0.424, 0.525]
DIAMOND	1.459 [1.358, 1.566]	0.373 [0.309, 0.489]	0.641 [0.578, 0.706]
DRIFT-symlog	1.328 [1.201, 1.461]	0.427 [0.379, 0.488]	0.620 [0.553, 0.699]
DRIFT	1.348 [1.252, 1.448]	0.520 [0.395, 0.730]	0.673 [0.602, 0.757]

D.2 Training Curves

On Atari (Fig. 8), we call a game still improving when its seed-averaged return over the final 20% of training exceeds that of the preceding 20% by more than 5% of the curve range. By this criterion, 21 of 26 games are still improving for DRIFT and 19 of 26 for DRIFT-symlog. DRIFT and DRIFT-symlog also reach their best return in the final quarter of training on 23 and 21 games. The main exceptions are PrivateEye for DRIFT and Freeway for DRIFT-symlog. On Freeway, the large variance comes from exploration failure.

On Craftium (Fig. 9), both DRIFT and DRIFT-symlog reach the level of Horizon Imagination on SmallRoom by about 50k steps and stay well above DIAMOND. On Room and Speleo, where reward accumulates over long episodes, the spread stays wide throughout training. DRIFT stays above DIAMOND on Room for most of training and above both baselines on Speleo from about 40k steps. ChopTree is the one task where DRIFT-symlog leads throughout training.

D.3 Seed Variability and Crafter Achievements

Much of the Craftium variance in Tab. 8 comes from the reward structure. In SmallRoom and Room, the step penalty separates episodes that reach the target from timeouts. In Speleo, the depth reward amplifies early route choices. Because all five DRIFT seeds on SmallRoom and Speleo score above the mean of both baselines (Sec. 5.2), these gains do not depend on a few seeds. On Room and ChopTree, the seeds of DRIFT overlap with those of Horizon Imagination. On Crafter, every DRIFT-symlog seed scores above every DIAMOND seed (6.6% or more against at most 4.8%). Tab. 9 gives the per-achievement success rates behind the Crafter analysis of Sec. 5.2.

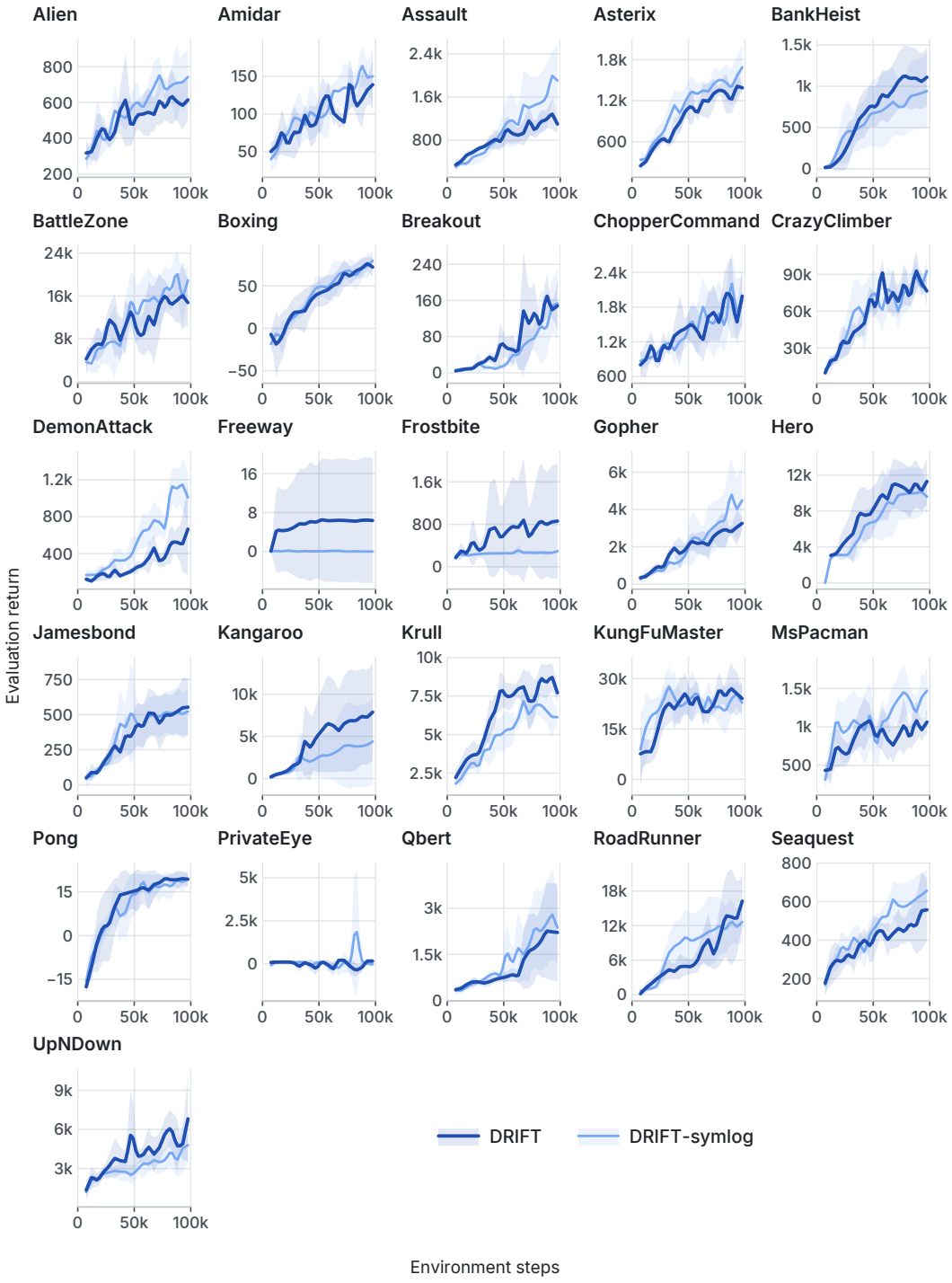

Figure 8: **Atari 100k learning curves on all 26 games.** Mean evaluation return over 5 seeds; shading shows ± 1 standard deviation.

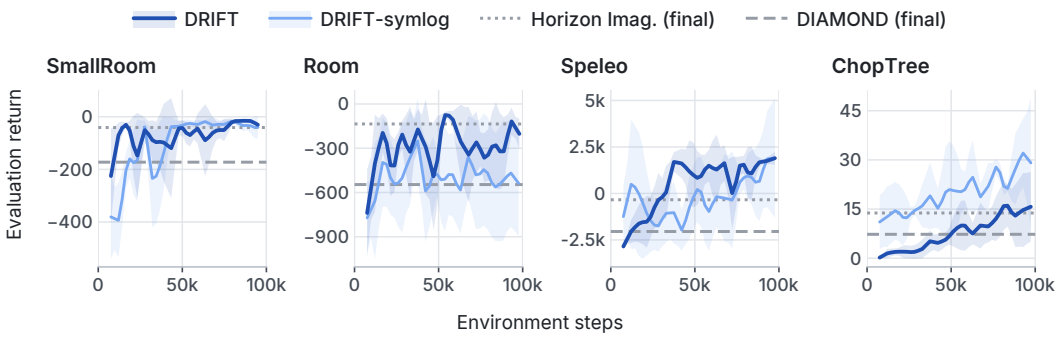

Figure 9: **Craftium learning curves.** Mean evaluation return over 5 seeds; shading shows ± 1 standard deviation, and gray lines mark the final returns of the baselines.

Table 8: **Craftium and Crafter results with standard deviations.** Mean $\pm$ standard deviation across seeds, at 100k environment steps for Craftium and 1M for Crafter; DRIFT uses 5 seeds. DreamerV3 values are from Hafner et al. (2023).

Craftium	SmallRoom	Room	Speleo	ChopTree
DIAMOND	-172.6 ± 283.6	-546.1 ± 459.9	$-2{,}047.0 \pm 903.0$	7.3 ± 2.5
Horizon Imag.	-40.9 ± 35.6	-135.5 ± 154.5	$-339.6 \pm 2{,}354.7$	13.8 ± 16.3
DRIFT-symlog	-18.6 ± 8.4	-649.6 ± 422.4	$669.8 \pm 2{,}316.1$	31.6 ± 14.3
DRIFT	-19.4 ± 4.5	-123.2 ± 46.3	$1{,}727.6 \pm 295.8$	15.3 ± 11.3

Crafter	Score (%)	Return
DreamerV3-XL	14.5 ± 1.6	11.7 ± 1.9
Horizon Imag.	8.1 ± 3.2	7.3 ± 1.9
DIAMOND	3.8 ± 0.7	4.3 ± 1.0
DRIFT-symlog	10.6 ± 2.5	7.4 ± 0.9
DRIFT	6.4 ± 2.3	5.6 ± 0.8

D.4 Comparison with Further Methods

Following Alonso et al. (2024), Tab. 10 compares DRIFT with the search-based methods MuZero (Schrittwieser et al., 2020) and EfficientZero (Ye et al., 2021) and with the model-free methods CURL (Laskin et al., 2020), SPR (Schwarzer et al., 2021), SR-SPR (D'Oro et al., 2023), and BBF (Schwarzer et al., 2023). EfficientZero plans with Monte-Carlo tree search at every step. BBF trains a larger value network at a high replay ratio with periodic network resets. Both score higher than DRIFT overall. We attribute these gains to search and to how the value network is trained, both independent of how the world model is learned. We leave combining these techniques with DRIFT to future work. DRIFT is superhuman on as many games as BBF (12) and scores higher than BBF on 7 of the 26 games, including BankHeist, Hero, and PrivateEye. In aggregate, DRIFT is close to SR-SPR (IQM 0.673 vs. 0.700; mean 1.348 vs. 1.271).

On the four Atari games that Horizon Imagination reports (Tab. 11), DRIFT scores higher on Boxing and CrazyClimber and slightly lower on Gopher and KungFuMaster. Horizon Imagination has lower amortized latency than DRIFT (App. E.4.1), but it uses about $41\times$ more FLOPs per imagined frame.

E Analysis of Modeling Choices

E.1 Ablation Results

With the tokenizer frozen, next-latent prediction raises the mean HNS from 0.533 to 1.004 (Tab. 12a), in line with its IQM gain in Sec. 5.3. Online tokenizer training helps only when combined with next-latent prediction. On its own, it improves three of the eight games and moves the mean only from 0.533 to 0.551. With next-latent prediction, it improves six of the eight and raises the mean to

Table 9: **Crafter achievement success rates.** Percentage of the 100 evaluation episodes per seed in which each achievement is unlocked at 1M environment steps, averaged over 5 seeds.

Achievement	DRIFT-symlog	DRIFT	Achievement	DRIFT-symlog	DRIFT
Collect coal	18	4	Make iron pickaxe	0	0
Collect diamond	0	0	Make iron sword	0	0
Collect drink	76	76	Make stone pickaxe	0	0
Collect iron	0	0	Make stone sword	0	0
Collect sapling	75	89	Make wood pickaxe	79	45
Collect stone	52	15	Make wood sword	75	55
Collect wood	96	93	Place furnace	0	1
Defeat skeleton	4	1	Place plant	72	70
Defeat zombie	30	2	Place stone	48	11
Eat cow	29	25	Place table	90	80
Eat plant	0	0	Wake up	91	84

Table 10: **Comparison with search-based and model-free methods on Atari 100k.** Mean return per game and human-normalized aggregates. Baseline scores, means, and IQMs are from Alonso et al. (2024); medians are computed from the per-game scores. Bold marks the best result in each row.

		Search-based		Model-free				
Game	**Human**	**MuZero**	**EfficientZero**	**CURL**	**SPR**	**SR-SPR**	**BBF**	**DRIFT**
Alien	7,127.7	530.0	808.5	711.0	841.9	1,107.8	**1,173.2**	594.8
Amidar	1,719.5	38.8	148.6	113.7	179.7	203.4	**244.6**	111.6
Assault	742.0	500.1	1,263.1	500.9	565.6	1,088.9	**2,098.5**	1,094.1
Asterix	8,503.3	1,734.0	**25,557.8**	567.2	962.5	903.1	3,946.1	1,398.1
BankHeist	753.1	192.5	351.0	65.3	345.4	531.7	732.9	**957.5**
BattleZone	37,187.5	7,687.5	13,871.2	8,997.8	14,834.1	17,671.0	**24,459.8**	17,910.0
Boxing	12.1	15.1	52.7	0.9	35.7	45.8	**85.8**	74.6
Breakout	30.5	48.0	**414.1**	2.6	19.6	25.5	370.6	149.1
ChopperCommand	7,387.8	1,350.0	1,117.3	783.5	946.3	2,362.1	**7,549.3**	1,892.2
CrazyClimber	35,829.4	56,937.0	**83,940.2**	9,154.4	36,700.5	45,544.1	58,431.8	80,790.0
DemonAttack	1,971.0	3,527.0	13,003.9	646.5	517.6	2,814.4	**13,341.4**	521.2
Freeway	29.6	21.8	21.8	**28.3**	19.3	25.4	25.5	6.4
Frostbite	4,334.7	255.0	296.3	1,226.5	1,170.7	**2,584.8**	2,384.8	860.8
Gopher	2,412.5	1,256.0	3,260.3	400.9	660.6	712.4	1,331.2	**3,487.3**
Hero	30,826.4	3,095.0	9,315.9	4,987.7	5,858.6	8,524.0	7,818.6	**11,719.6**
Jamesbond	302.8	87.5	517.0	331.0	366.5	389.1	**1,129.6**	511.0
Kangaroo	3,035.0	62.5	724.1	740.2	3,617.4	3,631.7	**6,614.7**	6,204.0
Krull	2,665.5	4,890.8	5,663.3	3,049.2	3,681.6	5,911.8	**8,223.4**	7,644.4
KungFuMaster	22,736.3	18,813.0	**30,944.8**	8,155.6	14,783.2	18,649.4	18,991.7	23,607.6
MsPacman	6,951.6	1,265.6	1,281.2	1,064.0	1,318.4	1,574.1	**2,008.3**	962.7
Pong	14.6	-6.7	**20.1**	-18.5	-5.4	2.9	16.7	18.6
PrivateEye	69,571.3	56.3	96.7	81.9	86.0	97.9	40.5	**2,426.8**
Qbert	13,455.0	3,952.0	**13,781.9**	727.0	866.3	4,044.1	4,447.1	2,172.0
RoadRunner	7,845.0	2,500.0	17,751.3	5,006.1	12,213.1	13,463.4	**33,426.8**	16,558.8
Seaquest	42,054.7	208.0	1,100.2	315.2	558.1	819.0	**1,232.5**	591.6
UpNDown	11,693.2	2,896.9	17,264.2	2,646.4	10,859.2	**112,450.3**	12,101.7	7,152.9
#Superhuman		5/26	**14/26**	2/26	6/26	9/26	12/26	12/26
Mean HNS		0.562	1.943	0.261	0.616	1.271	**2.247**	1.348
Median HNS		0.227	**1.090**	0.092	0.396	0.684	0.917	0.520
IQM HNS		0.288	1.047	0.113	0.337	0.700	**1.139**	0.673

1.482, with the largest gain on Gopher. We hypothesize that next-latent prediction trains the control state to anticipate the tokenizer's latents, so an adapted tokenizer gives it better targets. In Tab. 12b, training on self-generated rollouts raises the number of superhuman games from 8 to 12 and the median from 0.279 to 0.520. With symlog rewards, the median is 0.427.

Table 11: **Comparison with Horizon Imagination on Atari 100k.** Mean return on the four games that Cohen et al. (2026) report, with aggregates over these games. Bold marks the higher score.

Game	Horizon Imag.	DRIFT
Boxing	52.0	**74.6**
CrazyClimber	61,100.0	**80,790.0**
Gopher	**3,812.0**	3,487.3
KungFuMaster	**24,600.0**	23,607.6
Mean HNS	2.267	**2.885**
Median HNS	1.829	**2.147**

Table 12: **Ablation results.** Human-normalized scores for Fig. 3. (a) Online tokenizer training and next-latent prediction on 8 games; the four ablation rows are trained without self-generated rollouts, and the DRIFT row is the full method from Tab. 1. (b) Self-generated rollouts and reward predictor on 26 games.

(a) Next-latent prediction (8 games).

Online VAE	Next-lat. pred.	Mean	Median	IQM	Super-human
×	×	0.533	0.363	0.199	2/8
✓	×	0.551	0.202	0.242	2/8
×	✓	1.004	**1.272**	0.790	4/8
✓	✓	1.482	0.881	0.818	3/8
DRIFT		**1.778**	1.269	**1.243**	**5/8**

(b) Self-generated rollouts (26 games).

Reward predictor	Self-gen. rollouts	Mean	Median	IQM	Super-human
sign	×	1.151	0.279	0.447	8/26
sign	✓	**1.348**	**0.520**	**0.673**	**12/26**
symlog	✓	1.328	0.427	0.620	**12/26**

E.2 Denoising–Control Representation Gap

E.2.1 Linear Probes of the Control State

Setup. The four games have AtariARI object annotations and cover two-player combat (Boxing), paddle-and-ball control (Pong, Breakout), and platform navigation (Hero). For each game, the trained DRIFT policy collects 40k training frames and 10k held-out frames, taking a random action 25% of the time for coverage. Two world models are trained from scratch on the same training frames, with the same seed, batches, and number of updates (2k tokenizer and 8k world-model updates). They differ only in the next-latent loss weight (0.3 or 0). A third, randomly initialized model provides a lower reference. The ridge probes of Tab. 13 use one seed. For the action-conditioned and standard probes of Tab. 14, we retrain the three world models with three seeds on the same frames. Each comparison then pairs the two trained world models on the same seed and frames.

Ridge probes. We read the control state s_t on the held-out episodes and fit ridge regressions from it, choosing the regularization on a validation split. The three targets are the object positions annotated by AtariARI (Anand et al., 2019) at step t and at step $t{+}4$, and the summed reward over steps $t{+}1$ to $t{+}4$.

The gains in R^2 for current and future positions are largest on Boxing and Hero. The effect on the reward target is small. The probes support the interpretation in Sec. 5.3 that next-latent prediction makes the current and near-future game state more linearly accessible in s_t.

Action-conditioned probes. To test whether s_t also encodes how actions change the game state, we probe the displacement of the agent-controlled object (the player position annotated by AtariARI) from step t to $t{+}k$, for $k \in \{1, 4\}$. We fit ridge regressions to this displacement from the one-hot actions taken between t and $t{+}k$, and from these actions together with s_t. The gain ΔR^2 from adding s_t measures how much of the action-dependent motion the state explains beyond the actions alone (Tab. 14).

Table 13: **Linear probes of the control state.** Test R^2 of ridge probes on held-out episodes, one seed per game; higher is better.

Game	World model	Positions (t)	Positions (t+4)	Reward (t+1 to t+4)
Boxing	Random init.	0.78	0.57	0.10
	w/o next-latent	0.85	0.64	0.15
	w/ next-latent	0.95	0.74	0.17
Pong	Random init.	0.60	0.46	0.08
	w/o next-latent	0.86	0.74	0.44
	w/ next-latent	0.88	0.77	0.51
Breakout	Random init.	0.20	0.13	-0.01
	w/o next-latent	0.53	0.34	0.05
	w/ next-latent	0.58	0.35	0.03
Hero	Random init.	0.75	0.65	0.38
	w/o next-latent	0.79	0.67	0.49
	w/ next-latent	0.85	0.72	0.51

Table 14: **Probes over three seeds.** Mean $\pm$ standard deviation over three seeds, each retraining the three world models on the same frames. ΔR^2 is the gain in test R^2 from adding s_t to the actions when predicting the controlled object's displacement over k steps. F1 is the AtariARI weighted F1 of 256-way linear classifiers, and r is the Pearson correlation of LeWM-style linear and MLP regressors.

		Action-conditioned ΔR^2		AtariARI	LeWM r	
Game	World model	k=1	k=4	F1	Linear	MLP
Boxing	Random init.	0.01 ± 0.01	0.03 ± 0.01	0.21 ± 0.01	0.87 ± 0.01	0.92 ± 0.01
	w/o next-latent	0.15 ± 0.02	0.15 ± 0.01	0.19 ± 0.01	0.89 ± 0.02	0.97 ± 0.00
	w/ next-latent	0.26 ± 0.01	0.19 ± 0.01	0.24 ± 0.01	0.98 ± 0.01	0.99 ± 0.00
Pong	Random init.	0.17 ± 0.02	0.19 ± 0.01	0.44 ± 0.02	0.75 ± 0.02	0.85 ± 0.04
	w/o next-latent	0.60 ± 0.04	0.25 ± 0.02	0.38 ± 0.02	0.90 ± 0.01	0.96 ± 0.02
	w/ next-latent	0.70 ± 0.01	0.30 ± 0.01	0.39 ± 0.01	0.93 ± 0.01	0.95 ± 0.01
Breakout	Random init.	-0.05 ± 0.06	-0.15 ± 0.10	0.01 ± 0.00	0.38 ± 0.05	0.43 ± 0.03
	w/o next-latent	0.22 ± 0.11	0.05 ± 0.02	0.18 ± 0.13	0.62 ± 0.12	0.63 ± 0.13
	w/ next-latent	0.37 ± 0.01	0.07 ± 0.00	0.26 ± 0.00	0.69 ± 0.01	0.72 ± 0.02
Hero	Random init.	0.05 ± 0.01	0.16 ± 0.08	0.45 ± 0.01	0.82 ± 0.02	0.90 ± 0.01
	w/o next-latent	0.07 ± 0.01	0.22 ± 0.05	0.35 ± 0.01	0.78 ± 0.04	0.85 ± 0.02
	w/ next-latent	0.21 ± 0.02	0.29 ± 0.03	0.39 ± 0.02	0.85 ± 0.03	0.89 ± 0.04

At each horizon, next-latent prediction raises ΔR^2 in all 12 game–seed pairs (paired sign test, two-sided $p < 0.001$). The largest gains are one step ahead, on Boxing (0.15 to 0.26) and Hero (0.07 to 0.21).

Standard probes. We also apply the standard LeWM and AtariARI protocols to the same features. Following LeWM (Maes et al., 2026), we fit linear and MLP regressors per variable. With linear probes, next-latent prediction raises r in 11 of 12 game–seed pairs (paired sign test, two-sided $p = 0.006$), and the remaining pair differs by less than 0.01. With MLP probes, the gap between the two trained models narrows on Boxing, Pong, and Hero, which suggests that next-latent prediction mainly makes the state more linearly accessible. The AtariARI F1 (Anand et al., 2019), from one linear classifier per annotated variable (dropping variables with entropy below 0.6 and test frames that duplicate a training frame), is less informative here because the randomly initialized model scores highest on Pong and Hero.

Table 15: **Multi-step latent prediction.** Mean score over 5 seeds on the 8 ablation games (App. C.1); Δ is the change in human-normalized score. M is the number of future latents predicted, and $M{=}1$, the setting used in all other experiments, gives the scores of DRIFT in Tab. 1. Aggregates are over per-game means, except IQM, which is over all 40 game–seed scores.

Game	$M{=}1$ **(ours)**	$M{=}4$	Δ **HNS**
Alien	594.8	778.5	$+0.027$
Boxing	74.6	67.3	-0.602
ChopperCommand	1,892.2	1,495.8	-0.060
CrazyClimber	80,790.0	83,947.2	$+0.126$
Gopher	3,487.3	3,345.2	-0.066
Hero	11,719.6	13,373.8	$+0.056$
KungFuMaster	23,607.6	19,894.2	-0.165
RoadRunner	16,558.8	14,409.0	-0.274
#Superhuman	**5/8**	4/8	
Mean HNS	**1.778**	1.658	-0.120
Median HNS	**1.269**	1.153	-0.116
IQM HNS	**1.243**	1.119	-0.124

E.2.2 Multi-Step Latent Prediction

We extend next-latent prediction to the next M latents with one MLP per horizon. The predictor for $\tilde{z}_{t+m}$ receives c_t and the actions $a_t, \ldots, a_{t+m-1}$. The losses are combined as

$$\mathcal{L} = \frac{\sum_{m=1}^{M} \beta^{m-1} \mathcal{L}_m}{\sum_{m=1}^{M} \beta^{m-1}}, \qquad \beta = 0.75,$$

with the total auxiliary-loss weight fixed at 0.3.

One-step prediction performs better on all aggregate metrics (Tab. 15), and the apparent gains from $M{=}4$ on some games come largely from individual seeds. On Hero, the next-latent loss of the $M{=}4$ model grows from 0.024 one step ahead to 0.033 four steps ahead, and even its one-step loss is above the 0.021 of $M{=}1$. We hypothesize that the longer-horizon targets compete with the one-step target. Objectives designed for longer horizons, such as the contrastive prediction over ten steps in TWISTER (Burchi & Timofte, 2025), may avoid this competition.

E.3 Train–Imagination Distribution Gap

To quantify the train–imagination distribution gap, we measure the reward loss of agents trained with self-generated rollouts separately on real and on generated latents. Averaged over the second half of training, the loss on generated latents is higher than on real latents in 23 of the 26 games and for 90% of the 130 game–seed pairs. The median ratio is 1.2. Because these agents also train the reward predictor on generated latents, the gap persists even under this objective.

E.4 Efficient One-Step Imagination

E.4.1 Inference Cost

For the per-frame generation cost of each method (Tab. 3), we count FLOPs exactly and average wall-clock latency over 200 iterations after warmup. For Horizon Imagination, the reported numbers include only the DiT and therefore underestimate its total inference cost. Each imagined frame of DRIFT takes two backbone passes against the cached context: one shortcut denoising step, and one pass over the clean latent that appends its keys and values to the cache and gives the readout its tokens (Algorithm 2). The 0.56 GFLOPs include both passes and are measured with an 8-frame context. The readout and predictors add 0.006 GFLOPs. At batch size 1, kernel launches dominate latency, so the latency gap between methods is smaller than their FLOP gap. With the batch used in training (360 imagined frames per step), DRIFT generates a frame in 0.07 ms on average (about 14k frames per second) on the same GPU.

E.4.2 Qualitative Analysis

Sampling budget and horizon. Figures 10 and 11 compare rollouts with 1, 2, and 4 shortcut steps per frame, and Fig. 12 reports their pixel error over the horizon. On Boxing the error is nearly the same for all three step counts and grows over the first ten steps as the boxers move. On Crafter, one step gives the lowest error at every horizon. All three settings follow the change of scene at t=15.

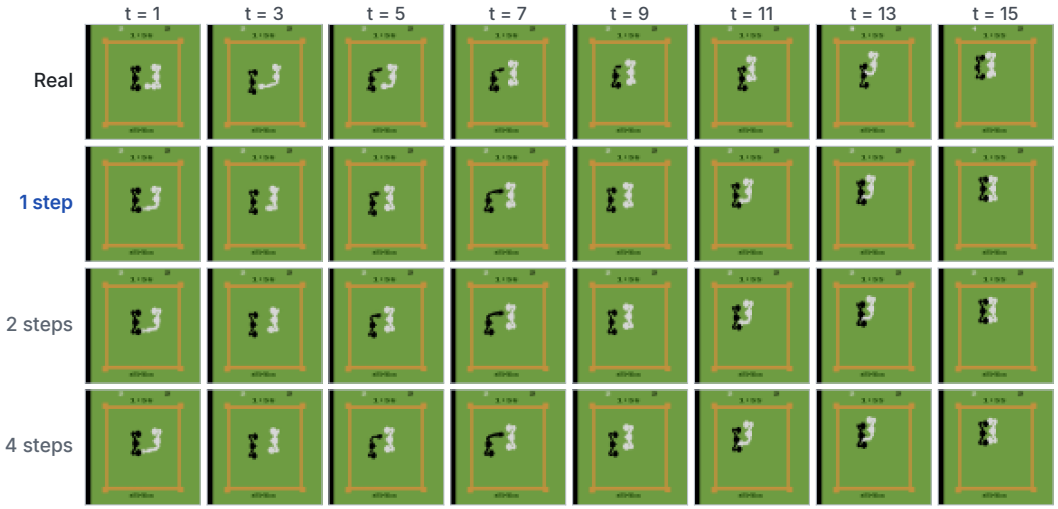

Figure 10: **Shortcut steps on Boxing.** Real frames and 15-step rollouts with 1, 2, and 4 shortcut steps per frame under the same policy actions. Each row shows the rollout with the median one-step pixel error among 10. Policy learning uses one step.

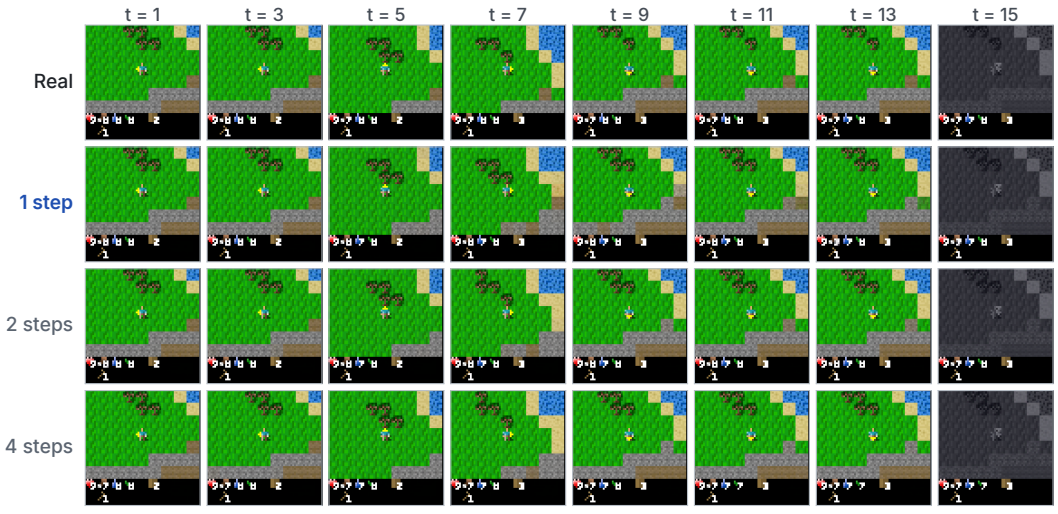

Figure 11: **Shortcut steps on Crafter.** Same setup as Fig. 10, with the median rollout among 9.

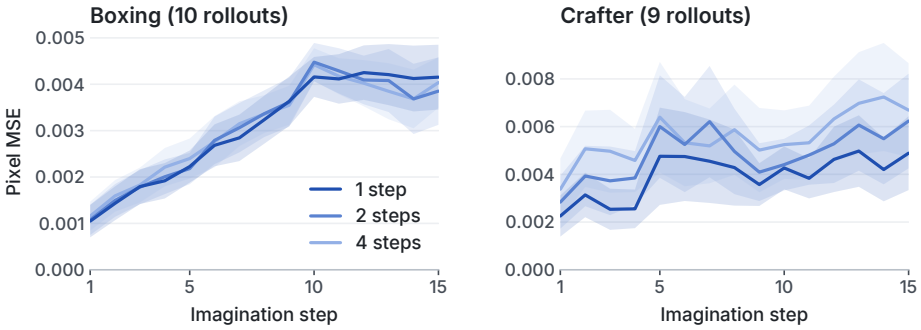

Figure 12: **Rollout error over the horizon.** Per-frame pixel MSE for 1, 2, and 4 shortcut steps per frame; mean and standard error over rollouts.