Extended Abstract

Motivation Model-based reinforcement learning (MBRL) has made remarkable strides with Dreamer V3 (Hafner et al., 2024). Despite its breadth of success, Dreamer's vanilla vision stack remains a standard monotonic convolutional neural network (CNN). Meanwhile, the computer vision community has made rapid progress in richer representation learning—ranging from Variational Autoencoders (VAEs) with principled information bottlenecks to self–supervised Transformers such as DINOv2 (Pinheiro Cinelli et al., 2021; Oquab et al., 2023). Our motivation is to investigate whether these modern, pretrained encoders can provide more meaningful visual latents and thereby accelerate or stabilize world–model learning and downstream policy performance.

Method We extend the Dreamer V3 framework by replacing its default convolutional encoder with a range of alternative vision modules, each producing a per-frame latent vector of configurable dimension. Concretely, we implement and compare:

- A vanilla **Conv-Encoder** baseline (with optional linear projection).
- A continuous **VAE-Encoder** trained with a β–KL regularizer.
- A discrete **VQ-VAE-Encoder** using a learnable codebook.
- A **SD-VAE-Encoder** based on Stable-Diffusion's pretrained VAE.
- A self-supervised **DINOv2-Encoder** based on DINOv2's patch features.
- A minimal **Dummy-Encoder** returning a global learnable vector.
- A **Zero-Encoder** that always outputs the zero latent.

Implementation We built upon the open-source `dreamer-v3-torch` codebase by extending the unified `Multi-Encoder` class that dispatches to one of seven vision backbones—Conv, VAE, VQ-VAE, SD-VAE, DINO-V2, Dummy and Zero—based on a new `-mode` CLI flag. Each encoder defines an `outdim` and returns an `(B,T,outdim)` tensor, which is then fed into the RSSM unchanged. We augmented `Multi-Encoder` class to support `-latent_dim`, `-disable_decoder`, and encoder-specific flags (e.g. `-sd_trainable`, `-dino_trainable`), and we modified the training loop to compute additional VAE and VQ losses or skip the pixel decoder when required.

Results Our experiments reveal three key findings. First, latent capacity beyond a modest threshold (128 dimensions) yields diminishing returns: both Conv and VAE encoders recover full performance by 128 D, with less than 3% variation up to 4096 D (Sec. 5.1–5.3). Second, pretrained vision backbones (DINOv2, SD-VAE) offer no convergence speed-up or final return improvements over scratch Conv/VAE models, even when fine-tuned or frozen (Sec. 5.4). Finally, pixel-level reconstruction remains critical: decoder ablations on frozen pretrained features fail catastrophically without the reconstruction loss, whereas including the decoder restores strong performance.

Discussion These results challenge the prevailing emphasis on ever larger latent or pretrained visual representations in model-based RL. Instead, they highlight that (1) Dreamer's RSSM readily adapts to extract control-relevant information from even low-dimensional latents or frozen features, and (2) dense, per-frame supervision via the decoder is indispensable for grounding imagined trajectories. Together, these insights suggest that the principal bottleneck lies not in visual capacity, but in the expressivity of the dynamics model itself. While our experiments provide clear evidence for the limited value of scaling visual encoders or latent size, they are restricted to three benchmark domains (Walker–Walk, Walker–Run, and Crafter), and results may differ on tasks with distinct observation or reward structures. Moreover, we only consider pixel-level reconstruction as the decoder objective; alternative auxiliary losses, such as contrastive or bisimulation metrics, could alter the decoder's impact.

Conclusion We conclude that scaling up the visual encoder or latent dimension beyond the minimal effective capacity is unnecessary for Dreamer-style agents and may squander resources. Future work should pivot toward enhancing the RSSM's temporal modeling—e.g., via transformer-based architectures—to better leverage visual inputs for long-horizon planning and robust control.

Does Visual Latent Quality Improve Dreamer-Style Model-Based RL?

Hazel Chen
Department of Computer Science
Stanford University
kaiweii@stanford.edu

Yixin Li
Department of Computer Science
Stanford University
yixinli@stanford.edu

James Qian
Department of Computer Science
Stanford University
yuxiqian@stanford.edu

Abstract

Model-based reinforcement learning (MBRL) methods such as DreamerV3 leverage learned world models to enable sample-efficient, long-horizon planning from pixel observations. Despite achieving state-of-the-art results across over 150 benchmarks, DreamerV3 still relies on a relatively "vanilla" convolutional encoder to process rich visual inputs—even though the vision community has developed powerful self-supervised and multimodal representation learners. In this work, we integrate regularized autoencoders (VAE, VQ-VAE) and large, off-the-shelf backbones (DINOv2, SD-VAE) into the Dreamer pipeline and conduct extensive ablations across latent dimensions (2–4096 D) and encoder variants. We show that returns saturate by 128 D, pretrained embeddings do not accelerate convergence or improve final performance, and that a pixel-level reconstruction objective is essential—even with high-capacity frozen features. These results suggest that future improvements should prioritize richer dynamics modeling (e.g. transformer-based RSSMs) over further scaling of visual encoders or latent bottlenecks.

1 Introduction

Model Predictive Control Garcia et al. (1989); Qi et al. (2025) and Model-Based Reinforcement Learning Ha and Schmidhuber (2018); Hansen et al. (2024) have played a pivotal role in advancing robotic control and autonomous navigation by enabling agents to predict future states based on their actions. These paradigms leverage predictive world models to facilitate long-horizon planning and robust task execution, thereby significantly enhancing sample efficiency and decision-making under complex dynamics.

Despite DreamerV3 achieving state-of-the-art performance across more than 150 diverse continuous control benchmarks (Hafner et al., 2024), its default convolutional encoder-decoder architecture may lack the capacity to capture complex visual cues and semantically meaningful latent structures required for generalizable imagined trajectories and stable actor-critic updates. To address this limitation, our project investigates the integration of latent-space regularized autoencoders, including Variational Autoencoders (VAE) Pinheiro Cinelli et al. (2021) and Vector-Quantized VAEs (VQ-VAE) van den Oord et al. (2018), as well as the use of pretrained encoder features, like SD-VAE Podell et al. (2023) or DinoV2 Oquab et al. (2023), into the Dreamer-style world model pipeline.

Furthermore, the original Dreamer literature provides limited ablation studies isolating the contributions of individual architectural components within the Recurrent State-Space Model (RSSM). To

bridge this gap, we conduct extensive ablations to examine the impact of latent dimensionality and the presence or removal of the encoder and decoder modules on overall model performance. Through this investigation, we aim to better understand the design choices that influence the learning stability and representational capacity of dreamer-like model-based reinforcement learning algorithm.

2 Related Work

Model-Based Reinforcement Learning. Model-based reinforcement learning (MBRL) centers on learning a predictive model of the environment's dynamics to facilitate policy optimization through imagined trajectories. Early work by Ha and Schmidhuber introduced mixture density RNN (MDN-RNN) to train agents entirely within their own imagined latent spaces (Ha and Schmidhuber, 2018). PlaNet (Hafner et al., 2019b) proposed a Recurrent State-Space Model (RSSM) with both stochastic and deterministic latent variables, enabling gradient-based planning directly from pixel inputs. Dreamer (Hafner et al., 2019a) improved this pipeline by integrating actor-critic learning through backpropagation within the imagined latent space.

More recently, DreamerV2 and DreamerV3 extended this framework with better scalability and task generalization (Hafner et al., 2022, 2024). DreamerV3, in particular, demonstrated strong generalization across over 150 tasks using a single RSSM configuration. However, these works rely on relatively simple convolutional encoders and decoders, which may be insufficient for capturing the full richness of high-dimensional visual observations. Recent extensions such as MuDreamer (Burchi and Timofte, 2024) replace pixel-based reconstruction with value and action prediction to improve performance in visually complex environments, while HRSSM (Sun et al., 2024) introduces hierarchical masking and bisimulation-inspired losses to improve generalization. Nevertheless, few of these studies explicitly isolate the contribution of visual representations to rollout accuracy or policy quality.

Visual Representation Learning. In parallel, the representation learning community has developed a suite of methods for learning structured and semantically rich latent spaces from high-dimensional sensory input. Variational Autoencoders (VAEs) (Pinheiro Cinelli et al., 2021) impose a probabilistic prior over the latent space to enable generative sampling and regularized inference, while Vector Quantized VAEs (VQ-VAEs) (Van Den Oord et al., 2017) discretize the latent space using a codebook, which yields sharper reconstructions and categorical latent structure, albeit at the cost of optimization challenges due to non-differentiable quantization.

More recent advances integrate large-scale pretraining and multimodal priors. SD-VAE (Podell et al., 2023) and DINOv2 (Oquab et al., 2023) provide powerful pretrained vision encoders that capture geometric and semantic consistency across varied data domains.

The SD-VAE used in Stable Diffusion is trained as a variational autoencoder on hundreds of millions of internet-scale captioned image-text pairs from the LAION dataset. It optimizes a standard ELBO objective that balances pixel-space reconstruction accuracy and KL-divergence regularization:

$$\mathcal{L}_{\text{SD-VAE}} = \mathbb{E}_{q(z|x)}[\log p(x|z)] - D_{\text{KL}}(q(z|x)\|p(z)).$$

As a result, the SD-VAE encoder captures rich semantic, geometry-aware features that are well-suited for downstream generative tasks.

DINOv2 is trained using a self-supervised teacher-student framework that leverages a multi-crop strategy and Vision Transformer backbones. It uses a cross-view alignment loss between the teacher and student embeddings of multiple global and local crops of the same image:

$$\mathcal{L}_{\text{DINO}} = -\sum_i p^{(i)}_{\text{teacher}} \log p^{(i)}_{\text{student}},$$

where $p^{(i)}$ denotes normalized patch token distributions. DINOv2 is trained on curated, diverse, high-resolution images without labels, resulting in representations that exhibit strong intra-class clustering and transfer well to novel tasks, including dense prediction and few-shot learning. Its patch-token representation supports fine-grained localization and semantic abstraction, which are critical for generalizable latent dynamics in RL.

While these pretrained and generative representations have demonstrated compelling performance in vision-language and generation tasks, their integration into model-based RL pipelines remains underexplored. In particular, it is unclear how improvements in visual latent structure influence imagination fidelity, policy stability, and downstream task success in diverse control environments.

Our work bridges these two domains by incorporating regularized and pretrained encoders into the Dreamer architecture and studying their influence through targeted ablations in world modeling and actor-critic learning.

3 Method

DreamerV3 Architecture. DreamerV3 employs a Recurrent State-Space Model (RSSM) to learn compact latent dynamics from high-dimensional pixel observations. The RSSM consists of an encoder $q_\phi(z_t|h_t, x_t)$ that encodes image x_t and recurrent hidden state h_t into a latent variable z_t, a transition model $p_\phi(z_t|h_t)$ that predicts future latent states, and a decoder $p_\phi(\hat{x}_t|h_t, z_t)$ that reconstructs pixel observations. The world model is optimized to jointly predict the image, reward r_t, and terminal signal c_t via:

$$\mathcal{L}(\phi) = \mathbb{E}_{q_\phi}\left[\sum_{t=1}^{T}\left(\beta_{\text{pred}}\mathcal{L}_{\text{pred}} + \beta_{\text{dyn}}\mathcal{L}_{\text{dyn}} + \beta_{\text{rep}}\mathcal{L}_{\text{rep}}\right)\right],$$

where:

$$\begin{aligned}
\mathcal{L}_{\text{pred}} &= -\log p_\phi(x_t|z_t, h_t) - \log p_\phi(r_t|z_t, h_t) - \log p_\phi(c_t|z_t, h_t),\\
\mathcal{L}_{\text{dyn}} &= \max\left(1, \text{KL}\left[\text{sg}(q_\phi(z_t|h_t, x_t)) \,\|\, p_\phi(z_t|h_t)\right]\right),\\
\mathcal{L}_{\text{rep}} &= \max\left(1, \text{KL}\left[q_\phi(z_t|h_t, x_t) \,\|\, \text{sg}(p_\phi(z_t|h_t))\right]\right).
\end{aligned}$$

In parallel, Dreamer uses a separate actor-critic module trained on imagined trajectories from the RSSM. The policy $\pi(a_t|z_t, h_t)$ and value function $V(z_t, h_t)$ are optimized using rollouts simulated purely in latent space.

Encoder and Decoder Architecture. The standard convolutional encoder consists of $S = \log_2(H/h_{\min})$ convolutional stages with stride 2 and kernel size 4:

$$x^{(l+1)} = \text{SiLU}(\text{Norm}(\text{Conv}_{k=4,s=2}(x^{(l)})))$$

followed by a flattening and optional linear projection:

$$z_t = \text{MLP}(\text{Flatten}(x^{(S)})).$$

The decoder applies a symmetric transposed convolutional stack that upsamples from latent vectors to full-resolution images:

$$\hat{x}_t = \text{Deconv}_S(\text{Linear}(z_t)).$$

Integrating VAE into RSSM. To encourage structured and regularized latent representations, we replace the encoder with a variational autoencoder. As before, input x_t is passed through the same convolutional backbone, and two MLP heads predict:

$$q_\phi(z_t|x_t) = \mathcal{N}(z_t; \mu_t, \sigma_t), \quad \mu_t, \log\sigma_t^2 = \text{MLP}_{\mu,\sigma}(x^{(S)}).$$

A sample is drawn via reparameterization:

$$z_t = \mu_t + \sigma_t \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I).$$

The total objective becomes:

$$\mathcal{L}_{\text{VAE}} = \mathbb{E}_{q_\phi(z_t|x_t)}[\log p_\phi(x_t|z_t)] - \beta\, D_{\text{KL}}(q_\phi(z_t|x_t) \,\|\, \mathcal{N}(0, I)).$$

Compared to the standard encoder, the VAE encourages the hidden space to align with a standard Gaussian prior, promoting stability and generalization in imagination rollouts.

Integrating VQ-VAE into RSSM. For discrete latent modeling, we use VQ-VAE. The encoder first downsamples via convolutional blocks as before, then projects to a latent feature z_e:

$$z_e = \text{MLP}(\text{Flatten}(x^{(S)})).$$

Each vector in z_e is replaced with its nearest codeword from a learned codebook $\mathcal{E} = \{e_k\}_{k=1}^K$:

$$z_q = \text{Quantize}(z_e) = \arg\min_{e_k \in \mathcal{E}} \|z_e - e_k\|_2^2.$$

The loss includes reconstruction and two stop-gradient terms:

$$\mathcal{L}_{\text{VQ-VAE}} = \|x_t - \hat{x}_t\|_2^2 + \|\text{sg}(z_e) - z_q\|_2^2 + \beta_{\text{commit}}\|z_e - \text{sg}(z_q)\|_2^2.$$

This structure discretizes the latent space, encouraging the model to learn compact and semantically meaningful latent representations.

Integrating Pretrained Autoencoder into RSSM. To exploit large-scale semantic priors, we adopt the VAE encoder and decoder from Stable Diffusion (SD-VAE), trained on diverse web-scale data. Given an image x_t, the encoder produces latent $z_t = f_{\text{SD}}(x_t)$, and reconstruction is done with the pretrained decoder.

$$\hat{x}_t = f_{\text{SD}}^{-1}(z_t), \quad z_t \sim \text{SDVAE}(x_t).$$

Integrating DINOv2 Features into RSSM. We also explore using DINOv2 (Oquab et al., 2023), a self-supervised ViT trained via teacher-student contrastive learning. The input x_t is first normalized and patchified:

$$x_t \to \text{Patchify}(x_t) \to \text{ViT}_{\text{DINO}}.$$

The ViT produces a set of patch tokens, which we flatten and project:

$$z_t = \text{Proj}(\text{Concat}(\text{DINO}(x_t))).$$

This z_t is then passed to the RSSM as the latent embedding.

4 Experimental Setup

4.1 Environments

We evaluate on three visually distinct benchmarks:

- **DM-Control Walker–Walk**: 64×64 side view RGB observations, six dimensional continuous torques, dense reward = forward velocity − control cost − posture penalty.
- **DM-Control Walker–Run**: Same observation and action spaces as Walker–Walk, but with a higher required locomotion speed and sparser reward signal emphasizing rapid forward motion.
- **Crafter**: 64×64 bird's-eye RGB observations, 18 discrete actions (movement, mining, crafting, combat), sparse achievement rewards (e.g. +20 for wooden pickaxe, +1000 for diamond) and survival incentives (health, hunger).

4.2 Encoder Variants & Latent Dimensions

For each run we replace Dreamer's default ConvEncoder with one of:

- `conv`, `vae`, `vqvae`, `sdvae`, `dinov2`, `dummy`, `zero`

We sweep the latent dimension

$$\text{latent\_dim} \in \{2, 8, 16, 32, 62, 128, 256, 512, 1024, 2048, 4096\}$$

to probe capacity effects, and for VQ-VAE we test codebook sizes $K \in \{32, 128, 512\}$ with 128-D codes.

4.3 Training Details

- **Batching:** Replay-buffer samples of length $T = 64$, batch size $B = 16$.
- **World-model:** RSSM with deter $= 512$, stoch $= 32$, discrete $= 32$ (except continuous mode for VAE and SD-VAE).
- **Loss scales:** $\beta_{\text{dyn}} = 0.5$, $\beta_{\text{rep}} = 0.1$; for VAEs $\beta_{\text{vae}} = 10^{-3}$.
- **Optimizer:** AdamW, learning rate 10^{-4} for model; 3×10^{-5} for actor/critic; $\epsilon = 10^{-8}$, grad-clip=1000.
- **Steps:**
  - If evaluation return converges before 500 K steps, terminate at 500 K.
  - Otherwise continue training up to 1M steps.

4.4 Hardware

All experiments were conducted on NVIDIA A100 GPUs, except for runs involving the DINOv2 encoder, which were executed on NVIDIA H100 GPUs.

5 Results

5.1 Impact of Latent Dimensionality

We begin our analysis by examining the effect of latent dimensionality on performance in the Walker-Walk environment. Table 1 presents the episode returns for both convolutional and VAE encoders across latent dimensions ranging from 32 to 1024.

	Walker–Walk					
Encoder	**32**	**64**	**128**	**256**	**512**	**1024**
Conv	956.9	957.8	964.8	961.3	961.7	959.4
VAE	948.9	940.1	946.3	953.6	949.8	948.2

Table 1: Walker-Walk performance across standard latent dimensions. Values represent mean episode returns.

Remarkably, both encoder architectures maintain highly stable performance across this range, with less than 3% variation between the lowest and highest performing configurations. This stability is particularly striking given that the original DreamerV3 configuration uses a latent dimension of 4096, suggesting substantial over-parameterization for this task.

To further probe the limits of this robustness, we extended our evaluation to extremely low latent dimensions. Table 2 shows that even with severe information bottlenecks, both encoders maintain surprisingly competitive performance.

Encoder	**2**	**4**	**8**	**16**
VAE	843.98	893.62	945.80	945.68
Conv	843.57	909.85	939.45	940.12

Table 2: Walker-Walk performance at extremely low latent dimensions. Values represent mean episode returns.

Even at 2D, the agent still learns nontrivial locomotion. We hypothesize that this behavior may arise because (1) Walker–Walk is an easy, fully observable task that does not demand rich visual encodings; (2) the RSSM and policy may overfit the reward–signal dynamics rather than relying on per-frame latents; or (3) the pixel decoder provides an implicit auxiliary reconstruction objective that compensates for the tiny latent. We address these hypotheses in subsequent experiments.

5.2 Ablation: Global and Zero Latents

To test whether Dreamer can "guess" observations from reward signals alone, we replace the per-frame encoder with two extreme baselines:

- **DummyEncoder**: returns a single learnable vector $\bar{z}$ for every step.
- **ZeroEncoder**: always returns the zero vector, providing no information.

Both variants disable the pixel decoder and reconstruction loss.

Encoder	**Return @500k**	**Return @1M**
Dummy (global latent)	162.02	35.4
Zero (no latent)	150.28	98.8

Table 3: Evaluation returns on Walker–Walk using Dummy vs. Zero latent encoders.

These ablations demonstrate that without per-frame latents, performance degrades sharply, invalidating the hypothesis that Dreamer simply overfits the reward and the observation is not being used by the model.

5.3 Bottleneck Effects on Harder Tasks

Next, we evaluate on the more challenging `Walker-Run` task, where forward speed is harder to maintain. Table 4 shows final returns at 500 K steps for four latent sizes:

Encoder	**2**	**8**	**32**	**128**
Conv	365.9	743.3	779.6	809.2
VAE	465.2	761.5	795.8	806.8

Table 4: Final evaluation returns on `dmc_walker_run` for varying latent dimensions.

On the harder Walker–Run task, we observe a dramatic performance drop when the latent dimension is extremely constrained: the ConvEncoder falls to 365.9 return at 2 dims and 743.3 at 8 dims, while the VAE preserves more information, scoring 465.2 and 761.5 respectively. This confirms that *insufficient latent capacity* becomes a critical bottleneck as task difficulty increases. However, once the latent size reaches 32 dimensions, both Conv and VAE encoders recover to approximately 780–795 return, and at 128 dims they match the original 4096-D Conv baseline (813.1). Thus, beyond a minimal capacity threshold, *further expansion of the latent offers diminishing returns*. Notably, the VAE's stochastic bottleneck consistently outperforms the ConvEncoder in the most compact regimes, suggesting its regularisation helps compress the most task-relevant features into very low-dimensional embeddings.

5.4 VQ-VAE Mode Collapse in RSSM Dynamics

After initially integrating a Gaussian VAE, we observed that its continuous normal prior $\mathcal{N}(0, I)$ may conflict with DreamerV3's RSSM posterior, which we parameterize as a categorical distribution over a 16×16 sufficient-statistic matrix for stability. To address this mismatch, we switched to a Vector-Quantized VAE (VQ-VAE), which eschews the Gaussian assumption and leverages a discrete codebook $\mathcal{E} = \{e_k\}_{k=1}^{K}$:

$$z_q = \arg\min_{e_k \in \mathcal{E}} \| z_e - e_k \|_2^2.$$

Despite this, the VQ-VAE induces severe *mode collapse* during early training (see 1a): almost all posterior mass concentrates on a single codebook entry. We quantify this by tracking the *unclipped* forward KL divergence used in $\mathcal{L}_{\text{dyn}}$ and the posterior entropy:

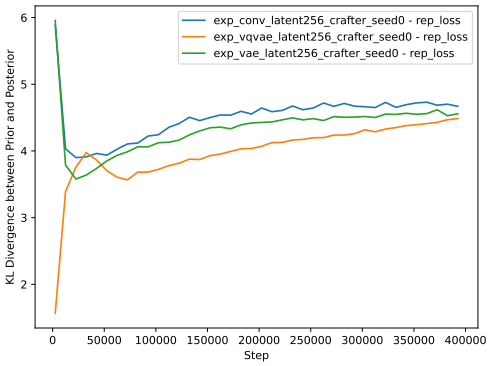
$$\mathrm{KL}_{\text{dyn}}(q_t \| p_t) = \mathbb{E}_{q_t}\big[\log q_t(z) - \log p_t(z)\big], \quad \mathcal{H}(q_t) = -\mathbb{E}_{q_t}\big[\log q_t(z)\big].$$

At initialization, VQ-VAE's $\mathrm{KL}_{\text{dyn}} \approx 1.5$ —far below the VAE and Conv baselines (≈ 6). Over ~ 600k steps, all methods converge to $\mathrm{KL}_{\text{dyn}} \approx 4.5$.

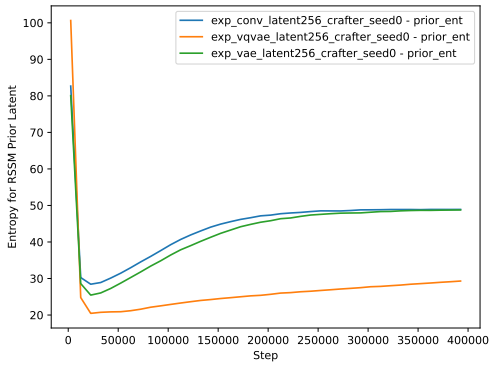
However, VQ-VAE's early "code collapse" cannot be fully reversed: its posterior entropy remains systematically lower than that of the VAE and ConvEncoder throughout training (see 1b). This persistent information bottleneck degrades the RSSM's imagined rollouts and yields consistently lower reward returns (see 5 and 2).

Encoder Type	**256**	**512**	**1024**	**2048**
Convolutional	6.09	6.64	7.79	9.21
VAE	6.30	6.42	6.31	6.77
VQ-VAE	6.36	5.35	4.84	6.77

Table 5: Final evaluation return on Crafter for various latent dimensions and encoder types.

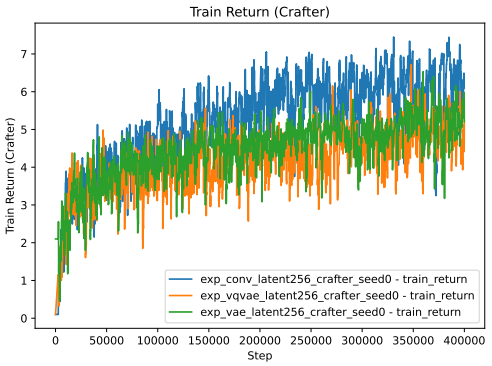


(a) Forward KL divergence used in $\mathcal{L}_{\text{dyn}}$.

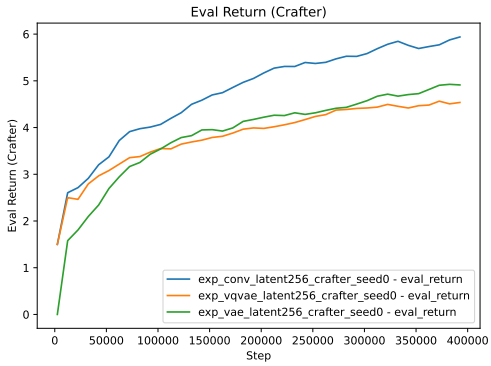


(b) Prior entropy $\mathcal{H}(q(z_t|h_t))$ of RSSM latent.

Figure 1: Latent space diagnostics for Conv, VAE, and VQ-VAE encoders in Crafter. VQ-VAE exhibits low KL and entropy early in training, indicating mode collapse.



(a) Training return on Crafter.



(b) Evaluation return on Crafter.

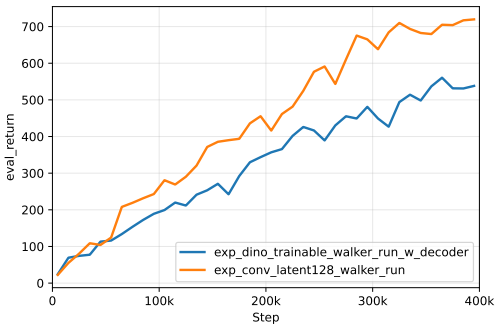
Figure 2: Training and evaluation returns on Crafter for different encoder types. VQ-VAE underperforms due to poor early latent usage.

5.5 Pretrained Features Do Not Improve Convergence

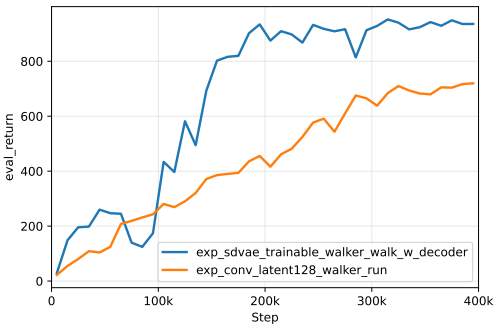
Having established the minimal latent requirements, we next evaluated whether off-the-shelf pretrained embeddings could accelerate learning. We swapped in two high-capacity backbones—a DINOv2 vision transformer and the Stable-Diffusion VAE—and trained Dreamer with these frozen or fine-tuned encoders, comparing against randomly initialized Conv and VAE baselines. Surprisingly, neither DINOv2 nor the pretrained VAE delivered any measurable speed-up. Even at very early stages, return trajectories are statistically indistinguishable. This suggests that, within the joint optimization of Dreamer's world-model and actor–critic, the RSSM quickly adapts to extract task-relevant features, and the inductive biases of large, pretrained vision models offer little extra benefit for these control tasks.

Frozen feature gap analysis. Even though pretrained features are not useful in terms of performance and covergence speed, it povides a way for us to examine the role of decoder in the model. First, we froze the pretrained encoder (either DINOv2 or SD-VAE) and trained only the RSSM transition and actor–critic modules. Despite receiving task-agnostic visual features, the model still learned to run in Walker–Run, as show in Figure 5a. This indicates that the discrepancy between pretrained and task-specific features is surprisingly small: the RSSM can leverage frozen features to extract most of the information necessary for control.

Decoder ablation with pretrained features. Next, we compared two setups using the same frozen pretrained encoder:

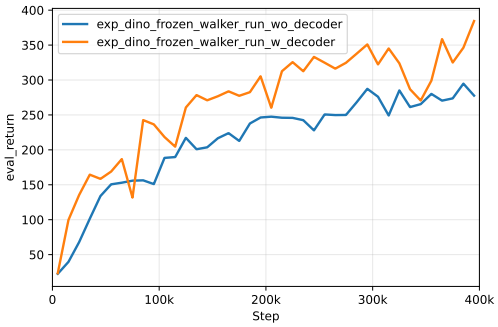


(a) Walker–Run: Conv vs. DINOv2

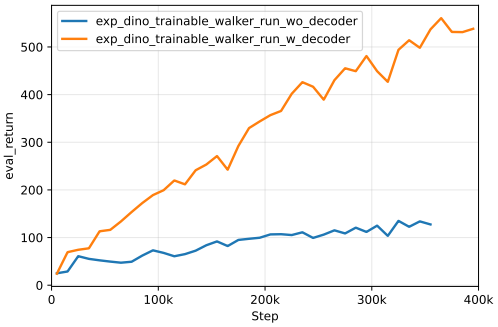


(b) Walker–Walk: Conv vs. SD-VAE

Figure 3: Evaluation Return curves comparing scratch Conv baseline against DINOv2 and SD-VAE encoders.



(a) Frozen DINOv2: w decoder vs. w/o decoder.



(b) Fine-tuned DINOv2: w decoder vs. w/o decoder.

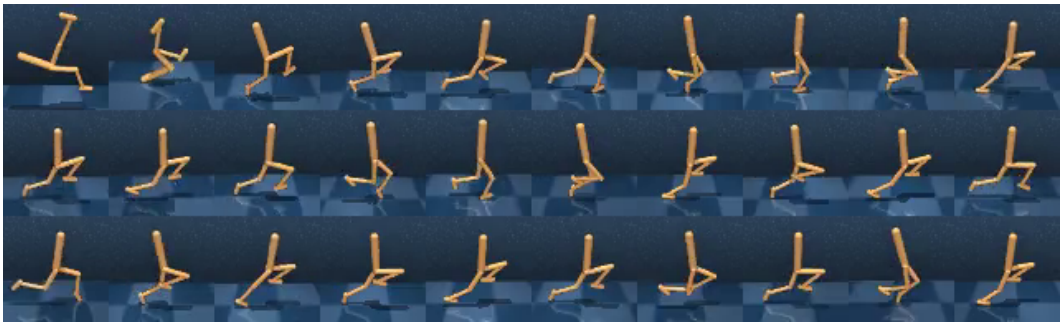
Figure 4: (a) Performance of Dreamer with frozen DINOv2 features; (b) Impact of retaining the pixel decoder when fine-tuning DINOv2. The "No-Decoder" variant fails to learn meaningful returns.

- **With-Decoder (Figure 5b)**: we retain the pixel decoder and reconstruction loss alongside dyn and rep.
- **No-Decoder (Figure 5c)**: the RSSM is trained with only the dyn and rep losses (no reconstruction), relying solely on the pretrained features for observation information.

Despite access to high-capacity frozen features, the *No-Decoder* variant failed to learn any meaningful policy (returns remained near random throughout training), showing that the reconstruction objective is still essential for grounding the RSSM in observation space. In contrast, the *With-Decoder* converged to reasonable performance.

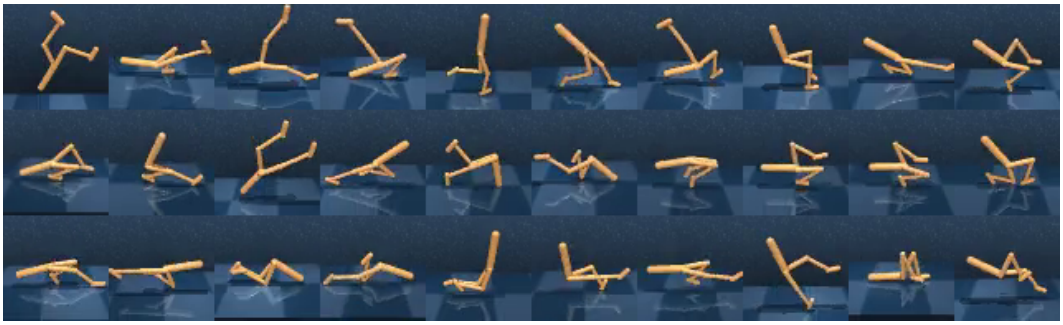
6 Limitation

While our study explores valuable insights into the roles of latent capacity, pretrained encoders, and the pixel decoder in Dreamer-style agents, it has several important limitations. First, our evaluation is confined to three benchmark domains—Walker–Walk, Walker–Run, and Crafter—which share similar observation modalities and reward structures; results may differ in environments with higher visual complexity, partial observability, or multi-agent interactions. Second, we focus exclusively on pixel-level reconstruction as the auxiliary decoder objective; alternative losses, such as contrastive representation learning or bisimulation metrics, could yield different trade-offs between grounding and abstraction. Third, our pretrained encoder experiments are limited to two models (DINOv2 and SD-VAE); other self-supervised or multimodal backbones (e.g., CLIP, masked autoencoders, or video-trained encoders) may interact differently with the RSSM.



(a) Frozen DINOv2: policy visualization

(b) Fine-tuned DINOv2 with decoder: policy visualization



(c) Fine-tuned DINOv2 without decoder: policy visualization

Figure 5: Policy visualizations for (a) frozen DINOv2 features, (b) fine-tuned DINOv2 with pixel decoder, and (c) fine-tuned DINOv2 without decoder. Each subfigure displays sampled frames arranged in a grid.

7 Conclusion

We have shown that the benefits of increasing latent dimension in Dreamer-style world models quickly saturate: beyond a modest capacity threshold (128 D), further scaling yields negligible performance gains (Sec.5.3). Similarly, swapping in large, pretrained vision backbones (DINOv2, SD-VAE) does not accelerate convergence or improve final returns, indicating that richer visual priors alone are insufficient for enhanced control performance. Through decoder ablations, we demonstrated that the pixel-level reconstruction objective remains essential—even with high-capacity frozen features—to ground the latent dynamics (Sec.5.4).

Future Work These insights suggest that, rather than further over-parameterizing the visual encoder or latent bottleneck, future efforts should focus on scaling and enriching the RSSM itself. In particular, integrating transformer-like architectures into the state-space model may enable more expressive temporal modeling and better utilization of visual features for long-horizon planning.

8 Team Contributions

- **James Qian** conceived the project idea and led the development of the overall research framework. He set up the codebase for the data pipeline, managed model training, and designed the model architecture.
- **Yixin Li** handled the execution of experiments, managed the evaluation pipeline, and played a crucial role in analyzing experimental results. He also refined and iterated on the experiments to improve the project's outcomes.
- **Hazel Chen** conducted a comprehensive literature review to support the theoretical framework of the project. She contributed to setting up the baseline models and also assisted in refining and documenting the experimental process.

We thank Professor Jitendra Malik and Professor Leonidas Guibas for providing computational resources.

References

Maxime Burchi and Radu Timofte. 2024. Mudreamer: Learning predictive world models without reconstruction. *arXiv preprint arXiv:2405.15083* (2024).

Carlos E Garcia, David M Prett, and Manfred Morari. 1989. Model predictive control: Theory and practice—A survey. *Automatica* 25, 3 (1989), 335–348.

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

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. 2019a. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603* (2019).

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

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. 2022. Mastering Atari with Discrete World Models. arXiv:2010.02193 [cs.LG] `https://arxiv.org/abs/2010.02193`

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. 2024. Mastering Diverse Domains through World Models. arXiv:2301.04104 [cs.AI] `https://arxiv.org/abs/2301.04104`

Nicklas Hansen, Hao Su, and Xiaolong Wang. 2024. TD-MPC2: Scalable, Robust World Models for Continuous Control. arXiv:2310.16828 [cs.LG] `https://arxiv.org/abs/2310.16828`

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. 2023. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193* (2023).

Lucas Pinheiro Cinelli, Matheus Araújo Marins, Eduardo Antúnio Barros da Silva, and Sérgio Lima Netto. 2021. Variational autoencoder. In *Variational methods for machine learning with applications to deep networks*. Springer, 111–149.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. 2023. SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis. arXiv:2307.01952 [cs.CV] `https://arxiv.org/abs/2307.01952`

Han Qi, Haocheng Yin, Aris Zhu, Yilun Du, and Heng Yang. 2025. Strengthening Generative Robot Policies through Predictive World Modeling. arXiv:2502.00622 [cs.RO] `https://arxiv.org/abs/2502.00622`

Ruixiang Sun, Hongyu Zang, Xin Li, and Riashat Islam. 2024. Learning latent dynamic robust representations for world models. *arXiv preprint arXiv:2405.06263* (2024).

Aaron Van Den Oord, Oriol Vinyals, et al. 2017. Neural discrete representation learning. *Advances in neural information processing systems* 30 (2017).

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2018. Neural Discrete Representation Learning. arXiv:1711.00937 [cs.LG] https://arxiv.org/abs/1711.00937