REPRESENTATION-AUGMENTED DIFFUSIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion models achieve state-of-the-art sample quality but lack an inherent mechanism to learn semantic representations from data. We propose *Representation-Augmented Diffusions* (RAD), a new class of diffusion models that natively embeds compact representations within the denoising process. Training maximizes a variational lower bound on the data likelihood, in which a recent class of models based on representation alignment emerge as special cases. Interestingly, our framework reveals a design choice overlooked by prior work: *feeding the learned representation back into the denoiser*. This simple idea leads to an order of magnitude gains in training convergence speed, data and sampling efficiency over prevalent representation alignment framework while enabling semantic discriminative representations. Furthermore, RAD extends naturally to other modalities like text-to-image synthesis, demonstrating that strong representations can be learned without compromising sample quality. Overall, RAD emerges as a strong alternative for efficiently training diffusion models.

1 INTRODUCTION

Diffusion models (Ho et al., 2020; Song et al., 2021) and related families (Lipman et al., 2023; Albergo & Vanden-Eijnden, 2023) achieve state-of-the-art sample quality by iteratively converting noise into data. Beyond sample quality, however, many downstream tasks need compact representations of the data; world models, for instance, plan and act in a learned latent state rather than in the pixel space (Hafner et al., 2023; Zhou et al., 2025; Assran et al., 2025; Maes et al., 2026). This raises a natural question: *Does denoising also yield useful representations of the data?* The answer is unclear for two reasons. First, diffusion models are trained only to denoise, and although their features carry semantic information (Xiang et al., 2023), they lag well behind those of self-supervised encoders (Yu et al., 2025). Second, unlike GANs and VAEs (Goodfellow et al., 2014; Kingma & Welling, 2014), which generate from a semantic latent space by construction, diffusion models lack such a compact latent space by construction as every intermediate latent is a noisier version of the data. Moreover, semantic features stay implicit within the denoiser.

A recent line of work based on representation alignment addresses the first problem by proposing to align the representations of an intermediate layer in the denoiser with those of a frozen pretrained encoder (Yu et al., 2025) or toward the model's own deeper features (Jiang et al., 2026). These methods show that good representations substantially accelerate the training of state-of-the-art diffusion transformers (Peebles & Xie, 2023; Ma et al., 2024). However, the representation serves only as a training-time regularizer and is discarded at generation, which leaves the second problem unresolved. Conversely, another line of work adds a latent variable that steers generation by combining diffusion models with auxiliary generative models such as a VAE (Pandey et al., 2022; Preechakul et al., 2022). These methods address the second problem but are often learned from a pixel reconstruction objective that favors low-level detail over semantics. We discuss additional related work in App. A.2.

To address both problems, we make the representation a latent variable of the diffusion model. Intuitively, at every denoising step, the model first predicts a representation from the current noisy state and then takes a denoising step conditioned on it (Fig. 2). We train the model by maximizing a variational lower bound on the data likelihood, in which the alignment loss appears as a KL regularizer rather than a heuristic add-on, and existing methods in representation alignment are recovered as a special case (Sec. 2.3 and App. A.8.1). Our framework reveals a simple design choice that representation alignment misses: *feeding the learned representation back into the denoiser* (Fig. 1). We

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.

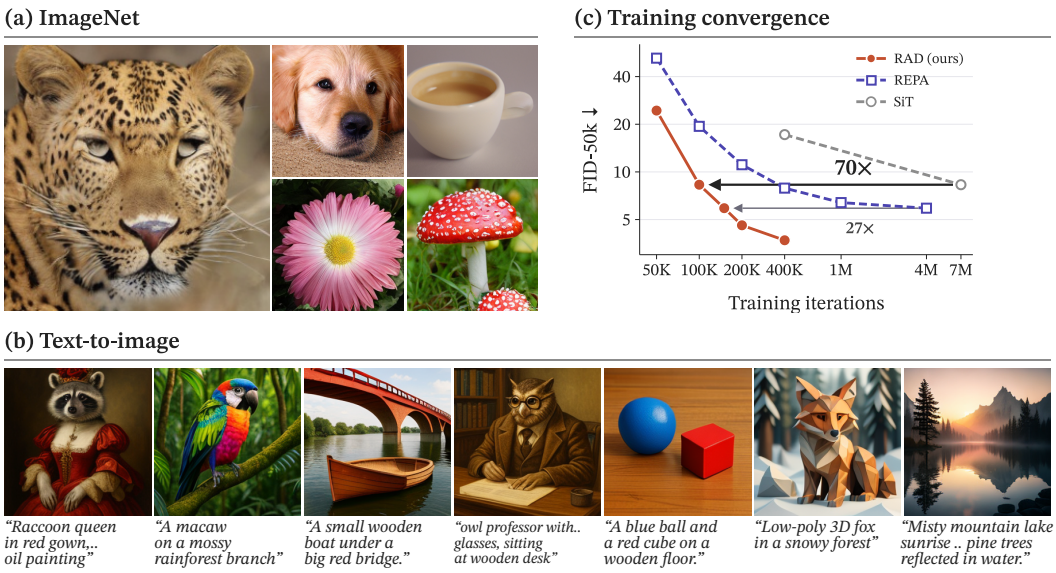


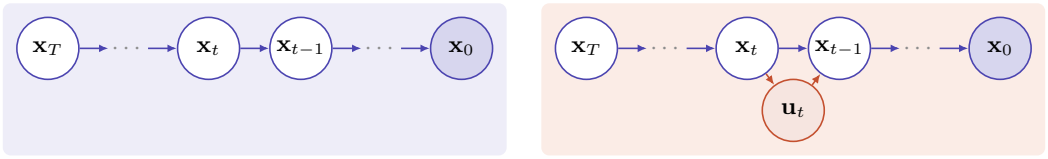
Figure 1: Representation-Augmented Diffusions (RAD). (a) ImageNet 256×256 samples. (b) Text-to-image samples. (c) Training convergence of XL/2 models on ImageNet: RAD matches the FID of SiT with $70\times$ and of REPA with $27\times$ fewer iterations.

call the resulting framework *Representation-Augmented Diffusions* (RAD). To summarize, our core contributions are:

- **A principled framework.** We formulate representation learning in diffusion models as a latent-variable model in which the denoiser conditions on a representation at every step. Our framework reveals a simple trick: feed the representation back into the denoiser, which leads to large improvements in sample quality (Sec. 3.5).
- **Faster, parameter- and data-efficient training.** On ImageNet, RAD-XL/2 surpasses REPA-XL/2 trained $27\times$ longer, while RAD-B/2 matches REPA-XL/2 with $4.8\times$ fewer parameters and outperforms REPA-B/2 trained on $10\times$ more data (Sec. 3.1).
- **Fewer sampling steps.** RAD induces more expressive denoising policies. As a result it has a better compute-vs-quality tradeoff than standard diffusion models. For instance, RAD-XL/2 needs $10\times$ fewer sampling steps than REPA-XL/2 for the same quality. (Sec. 3.2).
- **More discriminative and controllable representations.** We show that RAD learns smooth and semantic representations while preserving discriminative quality. RAD-B/2 scores 4.4 points higher than REPA under linear probing (Sec. 3.3).
- **Generalization Capabilities.** RAD can be readily applied on top of existing baselines and extends to other modalities like text-to-image synthesis (Sec. 3.4).

2 Representation-Augmented Diffusions

Intuition We introduce Representation-Augmented Diffusions (RAD), a class of diffusion models in which a semantic representation guides the generative process itself. The core idea is simple: at every denoising step, the model first predicts a representation from an intermediate noisy state, and then conditions the denoiser on it. This is in contrast to recent methods like REPA which predict representations but never condition the denoiser on them. During training, the target for learning such semantic representations can be provided by additional supervision; At inference, the model generates these representations from its own features, and no external supervision is required. Next, we discuss the core RAD modeling paradigm.



(a) Standard diffusion **(b)** RAD (ours)

Figure 2: RAD inserts a latent representation into every denoising step. Graphical models of the generative process of (a) Standard Diffusion and (b) RAD.

2.1 Denoising meets representations

Our key modeling choice is to augment the denoising process in standard diffusion models with a sequence of latent representations $\mathbf{u}_{1:T}$, one for every denoising step (see Fig. 2).

Generative process. Starting from noise $\mathbf{x}_T \sim p(\mathbf{x}_T) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, each denoising step first samples a latent representation $\mathbf{u}_t$ from a *representation policy* $p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)$. Consequently, the lesser noisy state $\mathbf{x}_{t-1}$ is then sampled from a *denoising policy* $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t)$ conditioned on the latent representation $\mathbf{u}_t$. More formally,

$$p_{\theta,\phi}(\mathbf{x}_{0:T}, \mathbf{u}_{1:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)\, p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t). \tag{1}$$

The representation $\mathbf{u}_t$ therefore acts as a hierarchical latent variable of the transition (Ranganath et al., 2016): the model first samples a semantic representation of the data and then takes a denoising step that is consistent with it. Consequently, we denote this class of diffusion models as *Representation-Augmented Diffusions* (RAD). It is worth noting that marginalizing over $\mathbf{u}_t$ yields the transition,

$$p_{\theta,\phi}(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \int p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t)\, p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)\, \mathrm{d}\mathbf{u}_t = \mathbb{E}_{\mathbf{u}_t \sim p_\phi}\left[p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t)\right], \tag{2}$$

which intuitively, is a continuous mixture that is more expressive than the Gaussian modeling choice in standard diffusion models. Since denoising is iterative and can be compute intensive, this property is specifically favorable for few-step sampling in diffusion models where the Gaussianity assumption over denoising breaks. We empirically demonstrate this property later where RAD strikes much better balance between sample quality vs inference compute (see Sec. 3.2). Moreover, when the denoising policy $p_\theta(. \mid \mathbf{x}_t, \mathbf{u}_t)$ ignores $\mathbf{u}_t$, the transition in Eq. 2 reduces to a Gaussian distribution reducing the setup to standard diffusion models.

Noising process. Next, we define the forward noising (a.k.a inference) process as,

$$q_{\theta'}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T} \mid \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1})\, q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\leq t}). \tag{3}$$

Intuitively, at each timestep t, we infer the next noisy state $\mathbf{x}_t$ from the cleaner state $\mathbf{x}_{t-1}$ via the noising kernel $q(. \mid \mathbf{x}_t)$, and then infer a representation $\mathbf{u}_t$ from the entire trajectory, $\mathbf{x}_{\leq t}$ up to time t using the *reference policy* $q_{\theta'}(. \mid \mathbf{x}_{\leq t})$. The representation $\mathbf{u}_t$ is therefore inferred from information that is unavailable to the generative model at step t, i.e. the cleaner trajectory and thus can serve as an external supervision for the representation policy $p_\phi(. \mid \mathbf{x}_t)$ (more on this in Sec. 2.3). With the modeling choices in place, we formulate a tractable variational bound for RAD next.

2.2 Tractable optimization objective

We train RAD by maximizing the variational lower bound on the log-likelihood of the data, $\log p(\mathbf{x}_0)$. With Eq. 3 as the variational distribution, we can show that (see proof in App. A.3),

$$\log p(\mathbf{x}_0) \geq \sum_{t=1}^{T} \mathbb{E}_{\mathbf{x}_{\leq t}}\Big[\underbrace{\mathbb{E}_{\mathbf{u}_t}[\log p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t)]}_{\text{Denoising}} - \underbrace{D_{\mathrm{KL}}(q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\leq t}) \| p_\phi(\mathbf{u}_t \mid \mathbf{x}_t))}_{\text{Alignment}}\Big], \tag{4}$$

where the trajectory $\mathbf{x}_{\leq t}$ is sampled from the noising process, i.e., $\mathbf{x}_t \sim q(\mathbf{x}_t \mid \mathbf{x}_0)$ and $\mathbf{x}_{t-1} \sim q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)$, and $\mathbf{u}_t$ is sampled from the reference policy $q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\leq t})$. The bound holds up to an additive constant that depends only on the noising process and not on (θ, ϕ) (see App. A.3.1). Intuitively, the objective in Eq. 4 is equivalent to learning a conditional VAE at any time t. The first term in Eq. 4 is analogous to the denoising objective of a standard diffusion model with an additional control input. During training, the denoiser learns to reconstruct $\mathbf{x}_{t-1}$ from the current noisy state $\mathbf{x}_t$ and a representation $\mathbf{u}_t$ drawn from the reference policy $q_{\theta'}(. \mid \mathbf{x}_{\leq t})$. Similarly, the second term penalizes the deviation of the learned generative policy $p_\phi(. \mid \mathbf{x}_t)$ from the reference policy $q_{\theta'}(. \mid \mathbf{x}_{\leq t})$ therefore creating an information bottleneck. Due to the nature of this objective, one would expect the latents $\mathbf{u}_t$ to capture meaningful semantic information. Indeed, we show that our model enables semantic properties like interpolation in the latent space $\mathbf{u}_t$ (Fig. 5) which is often hard to achieve with standard diffusion models. Next, we discuss parameterization of different distributions and draw connections to existing work in representation alignment which leads to a surprisingly simple recipe for training RAD.

2.3 Design Choices and Training recipe

We start by proposing different choices for the forward and the generative distributions from Sec. 2.1 and then simplify the RAD training objective bound in Eq. 4.

Noising Process. We choose the forward noising process the same as standard diffusion/stochastic interpolant formulations i.e. $q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\alpha_t \mathbf{x}_0, \sigma_t^2 \boldsymbol{I}_d)$. Following recent work on stochastic interpolants, we set $\alpha_t = 1 - t$ and $\sigma_t = t$ for $t \in [0, 1]$. Next, we parameterize the representation policy at any time t as,

$$q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\leq t}) = \mathcal{N}\left(\alpha_t f_{\text{pre}}(\mathbf{x}_0),\ \beta^2 \sigma_t^2 \boldsymbol{I}\right), \qquad \beta > 0 \tag{5}$$

Throughout this work we fix f_{pre} to a frozen pretrained DINOv2 (Oquab et al., 2024) encoder which extracts clean representations from data $\mathbf{x}_0$; in the absence of an external supervisor, alternative choices of the reference policy recover other methods (App. A.8.2). Our choice is motivated by empirical results (Table 5). Moreover, we adopt the same noising schedule for the reference policy as the forward diffusion process. Our intuition behind this choice is that during denoising at higher noise levels, only very coarse structures emerge and therefore the representations at these noise levels must encode more uncertainty. However, we emphasize that different choices of the noising schedule are possible and remains an interesting direction for further work. Moreover, since the supervision comes from $\mathbf{x}_0$ alone, the reference policy drops its dependence on the trajectory and carries no learnable parameters.

Choice of Representation Policy. We first consider a simple class of policies modeled by a unimodal Gaussian distribution,

$$p_\phi\left(\mathbf{u}_t \mid \mathbf{x}_t\right) = \mathcal{N}\left(\alpha_t \boldsymbol{h}_\phi(\mathbf{x}_t, t),\ \beta^2 \sigma_t^2 \boldsymbol{I}\right) \tag{6}$$

Similar to REPA, we model $\boldsymbol{h}_\phi(\mathbf{x}_t, t)$ as an MLP which projects the intermediate denoiser state at layer ℓ i.e $\boldsymbol{h}_t^\ell = \boldsymbol{h}_\phi(\mathbf{x}_t, t)$. The semantic representation can then be sampled via $\mathbf{u}_t = \alpha_t \boldsymbol{h}_t^\ell + \beta \sigma_t \epsilon$ where $\epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$. From a VAE perspective (Sec. 2.2), the initial ℓ blocks of the denoiser combined with the MLP head act as a noisy encoder for extracting representations from $\mathbf{x}_t$.

While the choice of unimodal Gaussian for the representation policy can induce multimodal denoising transitions (Eq. 2), we also consider conditional normalizing flow (Dinh et al., 2017) based policies with the following form (see App. A.9 for more details),

$$\mathbf{u}_t = \alpha_t \boldsymbol{h}_t^\ell + \beta \sigma_t \boldsymbol{g}_\zeta(\boldsymbol{h}_t^\ell, \epsilon), \qquad \epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d) \tag{7}$$

We find that using more expressive policies improves sample quality (Table 5).

Choice of Denoising Policy. We parameterize the denoising policy as $p_\theta\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t\right) = \mathcal{N}\left(\boldsymbol{\mu}_\theta(\mathbf{x}_t, \mathbf{u}_t, t),\ \tilde{\sigma}_{t-1|t}^2 \boldsymbol{I}\right)$ and model $\boldsymbol{\mu}_\theta$ using the SiT architecture throughout this work. The sampled representation $\mathbf{u}_t \in \mathbb{R}^{N \times D}$ enters the backbone $\boldsymbol{\mu}_\theta$ at the same layer ℓ where it is predicted, after two operations:

$$\mathbf{u}_t^n = \frac{\mathbf{u}_t - \gamma\, \bar{u}_t}{\bar{\sigma}_t + \delta} \quad \Longrightarrow \quad \mathbf{u}_t^c = \text{MLP}(\text{Scale-Shift}\left(\mathbf{u}_t^n, t\right)), \tag{8}$$

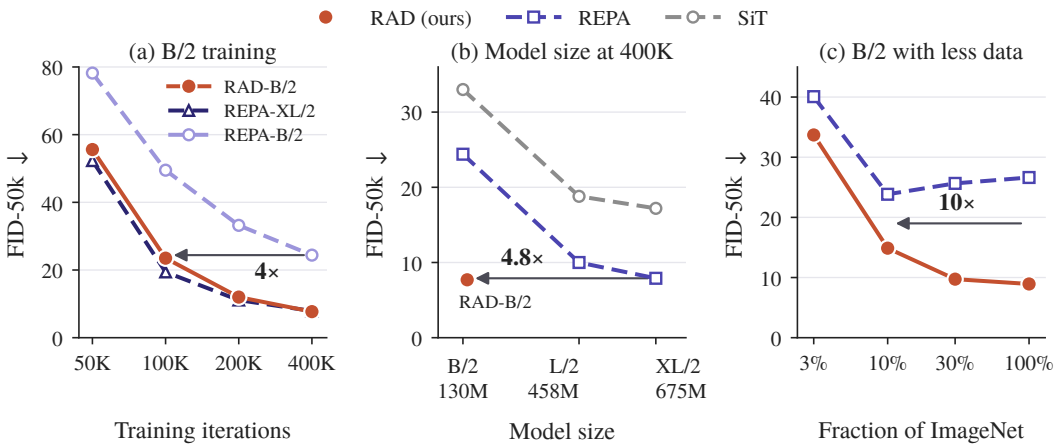


Figure 3: RAD trains faster and is more parameter- and data-efficient than REPA and SiT. (a) RAD-B/2 trains $4\times$ faster than REPA-B/2 and is comparable to REPA-XL/2. (b) RAD-B/2 is $4.8\times$ more parameter efficient than REPA. (c) RAD-B/2 is $10\times$ more data efficient than REPA-B/2.

where $\bar{u}_t, \bar{\sigma}_t \in \mathbb{R}^D$ are the per-channel mean and standard deviation of $\mathbf{u}_t$ across its N tokens, $\gamma \in [0, 1]$ controls how much of the mean is removed (we use $\gamma = 0.6$), and δ is a small constant. Intuitively, Eq. 8 first removes global information, shared across all tokens, from $\mathbf{u}_t$ (as it could be harmful for sample quality (Singh et al., 2026)) followed by a learnable AdaLN-style scale-shift transform and an MLP which compresses the information into a single token $\mathbf{u}_t^c$. We then add $\mathbf{u}_t^c$ back into the state $\mathbf{x}_t^\ell$ which is then passed into downstream SiT blocks to predict the velocity field. We present an ablation over these design choices in Sec. 3.5 and defer full details to App. A.9.

Simplified Training Objective Under these parameterizations, it can be shown that a weighted version of the variational bound in Eq. 4 reduces to the following loss (see App. A.3.2):

$$\mathcal{L}(\theta, \phi) = \mathbb{E}_{t, \mathbf{x}_0, \boldsymbol{\epsilon}, \mathbf{u}_t \sim q}\left[\left\|\boldsymbol{v}_\theta(\mathbf{x}_t, \mathbf{u}_t, t) - \boldsymbol{v}_{\text{ref}}\right\|_2^2 + \frac{\text{SNR}(t)}{2\beta^2}\left\|\mathbf{f}_{\text{pre}}(\mathbf{x}_0) - h_\phi(\mathbf{x}_t, t)\right\|_2^2\right], \tag{9}$$

where $\text{SNR}(t) = \alpha_t^2/\sigma_t^2$ and $\boldsymbol{v}_{\text{ref}} = \dot{\alpha}_t \mathbf{x}_0 + \dot{\sigma}_t \boldsymbol{\epsilon}$. The simplified objective in Eq. 9 is similar to the heuristic alignment objective proposed in REPA with the exception of the latent representations $\mathbf{u}_t$ now also part of the denoiser. Indeed, we show that existing external alignment methods like REPA are special cases of our framework (see App. A.8.1). During training, we always fix $\beta = 1.0$. However, we find that tuning β during inference improves sample quality (Sec. 3.5)

3 Experiments

We evaluate RAD primarily on class-conditional image synthesis on ImageNet 256×256. Compared to REPA, we show that RAD trains substantially faster ($27\times$), has better parameter and data efficiency, surpassing SiT-XL/2 trained $70\times$ longer (Sec. 3.1). It also exhibits a superior quality-vs-inference speed tradeoff and requires $10\times$ fewer sampling steps (Sec. 3.2). As a consequence of injecting the representations back into the denoiser, we show that RAD learns more discriminative and controllable representations (Sec. 3.3). We conclude by showing that RAD generalizes to other representation alignment baselines (Sec. 3.4) and modalities like text-to-image synthesis. We note some high-level experimental details below and defer full details to App. A.4.

Tasks and Datasets. We evaluate RAD in two settings. For **class-conditional generation**, we exactly follow the experimental setup of REPA and train on ImageNet 256×256 in the latent space of the Stable Diffusion VAE (Rombach et al., 2022). For **text-to-image synthesis**, we train on the BLIP3o pre-training captions (Chen et al., 2025) at 256×256 resolution, conditioning on caption features from a frozen Qwen3-0.6B text encoder (Qwen Team, 2025), and optionally fine-tune on the curated BLIP3o-60k instruction set.

Table 1: Comparison with existing baselines on Class-conditional ImageNet 256×256**.** FID-50k without CFG (Euler SDE, 250 steps) at 80 epochs (400K iter.); †REPA-E also fine-tunes the VAE end-to-end.

		B/2		L/2		XL/2	
Method	Epochs	#Params	FID↓	#Params	FID↓	#Params	FID↓
Diffusion transformers							
DiT [31]	80	130M	43.5	458M	23.3	675M	19.5
DiT [31]	1400	–	–	–	–	675M	9.6
SiT [26]	80	130M	33.0	458M	18.8	675M	17.2
SiT [26]	1400	–	–	–	–	675M	8.3
Representation alignment							
DiT + REPA [51]	80	–	–	458M	15.6	675M	12.3
REPA [51]	80	130M	24.4	458M	10.0	675M	7.9
iREPA [37]	80	130M	21.4	458M	9.4	675M	7.5
sREPA [48]	80	130M	20.5	458M	8.7	675M	7.2
HASTE [45]	80	130M	19.6	458M	8.9	675M	7.3
REED [42]	80	130M	19.5	458M	8.9	675M	7.3
TREAD [19]	80	130M	16.0	458M	5.9	675M	4.9
REG [46]	80	132M	15.2	460M	4.6	677M	3.4
REPA-E† [21]	80	–	–	–	–	675M	4.1
Representation augmentation							
RAD (ours)	80	142M	7.7	472M	3.9	690M	3.7

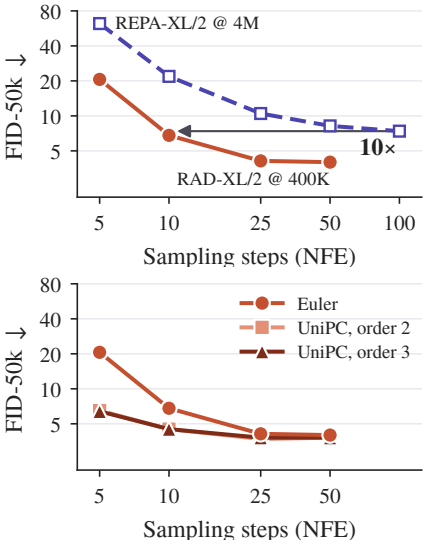


Figure 4: RAD has a better quality-vs-compute tradeoff at inference. Top: FID-50k vs. Euler ODE steps, RAD-XL/2 (400K) vs. REPA-XL/2 (4M). Bottom: Euler vs. UniPC for RAD-XL/2.

Models and Training. We use SiT-{B/2,L/2,XL/2} backbones for ImageNet and SiT-{B/2,XL/2} for text-to-image synthesis (/2 indicates 2x2 patch size). For all experiments, we use the DINOv2-B model as the default external supervisor f_{pre}. For notational convenience, we denote a SiT backbone trained with RAD by its size, e.g., RAD-XL/2. Following REPA, all models are trained with a global batch size of 256 using the AdamW optimizer (Loshchilov & Hutter, 2019) with a learning rate of 1e-4, and an Exponential Moving Average (EMA) of the weights, for up to 400K iterations (primarily due to compute limitations) unless stated otherwise.

Baselines. Our primary baseline is REPA, trained with an identical backbone, encoder, data, and optimization schedule. The only conceptual difference is whether the representation is fed back into the denoiser based on our setup in Sec. 2.3. We additionally compare against the unaligned SiT baseline and existing representation alignment methods which build directly on top of the REPA framework (Leng et al., 2025), though we hypothesize that such methods would directly benefit from our framework (see Sec. 3.4).

Evaluation Metrics. To measure sample quality, we report FID (Heusel et al., 2017), and sFID (Nash et al., 2021) on 50k samples. For class conditional ImageNet, we additionally report Inception Score (Salimans et al., 2016), and precision/recall (Kynkäänniemi et al., 2019) to measure the fidelity-diversity tradeoffs. We use the Euler–Maruyama SDE sampler with 250 steps no classifier-free guidance (Ho & Salimans, 2022) unless noted. For text-to-image synthesis, we report compositional accuracy on GenEval (Ghosh et al., 2023).

3.1 Generation Quality

Faster training convergence at every model scale. We compare the training efficiency of RAD in Fig. 3 (full results in Table 6 in App. A.5). Interestingly, after only 100K training iterations, RAD matches or surpasses REPA at 400K iterations in sample quality for all three backbones, a $4\times$ minimum reduction in training steps. Moreover, the gains grow further against long training runs (Fig. 1c): RAD-XL/2 surpasses SiT-XL/2 trained for 7M iterations after 100K iterations, and REPA-XL/2 trained for 4M iterations after 150K iterations, reducing the required training steps by $70\times$ and $27\times$, respectively. Since RAD and REPA share the backbone, encoder, and training schedule, we attribute these gains to feeding the aligned representation back into the denoiser.



Figure 5: RAD allows smooth interpolation in the representation space compared to REPA. Top: RAD-XL/2 (400K iters) preserves attributes like pose, and viewpoint consistent and allows a smooth interpolation between two images. Bottom: REPA-XL/2 (4M iters), yields undesirable artifacts often changing the semantic image content.

Better Parameter and Data efficiency. We find that RAD exhibits superior parameter efficiency compared to REPA. For instance, RAD-B/2 outperforms REPA-L/2 (FID 7.7 vs. 10.0) with $3.2\times$ fewer parameters (Fig. 3b). Similarly, RAD-B/2 outperforms SiT-XL/2 (FID: 8.3) trained for 7M iterations and also closely tracks REPA-XL/2 throughout training (Fig. 3a) and performs on par with it at 400K iterations (FID 7.7 vs. 7.9), while using $4.8\times$ fewer parameters.

RAD is also more data efficient. We train B/2 models for the same 400K iterations on subsets of ImageNet and report the best FID of each run over its checkpoints. Trained on 10% of ImageNet, RAD-B/2 outperforms REPA-B/2 (FID: 14.9 vs 26.6) trained on the full dataset i.e., RAD matches REPA with $10\times$ less data. Morever, REPA-B/2 does not improve beyond 10% of the data indicating underfitting while RAD keeps improving with more data.

Comparison with other baselines. Next, we compare RAD with additional alignment baselines (Table 1) reporting FID-50k after 80 epochs without classifier-free guidance. Across different model scales, RAD registers strong sample quality achieving best FID among competitors on SiT-B/2 and L/2 while being competitive to REG at the XL/2 scale. However, we show that our framework can be readily combined with other baselines (Sec. 3.4) and thus hypothesize that a lot of the techniques used by existing representation alignment would be complementary to RAD. We show class-conditional samples of RAD-XL/2 in App. A.10.

3.2 Sampling Efficiency

Marginalizing out the representations can lead to more expressive denoising policies than standard diffusion models (Sec. 2). This could be particularly useful in a few step sampling regime where a more expressive transition should tolerate larger steps. To test this hypothesis, we compare similar quality models: RAD-XL/2 (trained for 400k steps) with a REPA-XL/2 model trained for 4M steps in the few-step sampling regime using the Euler ODE sampler. Fig. 4 shows that RAD beats REPA in this setting at every step budget. With 10 Euler steps RAD reaches FID 6.8, below REPA's 7.4 at 100 steps, a $10\times$ reduction in sampling steps. Moreover, RAD is also compatible with other state-of-the-art samplers for diffusion models (Fig. 4, bottom). With UniPC (Zhao et al., 2023), RAD-XL/2 reaches FID 6.5 at 5 steps and 4.5 at 10 steps.

3.3 Representation Quality

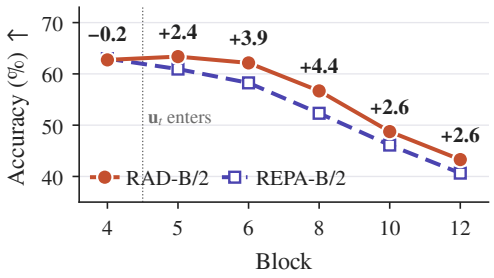
Interpolation in Representation Space As the denoiser conditions on $\mathbf{u}_t$ at every step, the learned representation can be used to manipulate the generations. We test this by interpolating between two real ImageNet images, $\mathbf{x}_0^A$ and $\mathbf{x}_0^B$. From each image we extract a representation $\bar{\mathbf{u}} = \boldsymbol{h}_\phi(\mathbf{x}_{t_c}, t_c)$ at a small noise level t_c. For a weight $m \in [0, 1]$, we linearly interpolate the representations, $\bar{\mathbf{u}}_m = m\,\bar{\mathbf{u}}^A + (1-m)\,\bar{\mathbf{u}}^B$, and likewise the class embeddings, and obtain the starting noise $\mathbf{x}_{1,m}$ by spherical linear interpolation (slerp) between the inverted noises $\mathbf{x}_1^A$ and $\mathbf{x}_1^B$. We then sample from $\mathbf{x}_{1,m}$. Fig. 5 interpolates from a baboon to a macaque with RAD-XL/2 at 400K iterations and the released REPA-XL/2 checkpoint, which shares our backbone. RAD changes the

Table 2: RAD improves REG training convergence. FID-50k, Euler SDE, w/o CFG

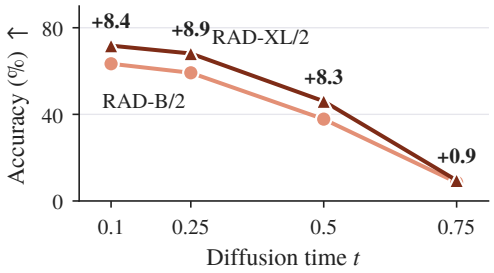
Model	#Params	Iter.	FID↓
SiT-B/2 [26]			
REG [46]	132M	100K	36.1
REG + RAD (ours)	143M	100K	**22.4**
REG [46]	132M	400K	15.2
REG + RAD (ours)	143M	400K	**8.9**
SiT-L/2 [26]			
REG [46]	460M	100K	11.4
REG + RAD (ours)	474M	100K	**7.8**
REG [46]	460M	400K	4.6
REG + RAD (ours)	474M	400K	**3.9**

Table 3: RAD improves text-to-image generation over REPA at both model scales. FID-50k and GenEval overall score on the BLIP3o pre-training captions at 256×256 with a frozen Qwen3-0.6B text encoder. Best per backbone in bold. No guidance.

	B/2		XL/2	
Method	FID↓	GenEval↑	FID↓	GenEval↑
REPA [51]	10.77	0.138	4.31	0.258
RAD (ours)	**8.42**	**0.166**	**3.94**	**0.266**



(a) Per-block accuracy for RAD-B/2 vs REPA



(b) Best block against t for RAD-{B/2, XL/2}

Figure 6: Better linear probing with RAD.

subject gradually while the pose, viewpoint, and framing stay fixed, indicating a smooth representation space. The interpolation of REPA is not consistent, with intermediate samples switching between different poses and scenes, even though it is trained for 4M iterations, i.e., $10\times$ longer. Additional samples and implementation details are given in App. A.7.

Classification Accuracy Following REPA and Xiang et al. (2023), we train a linear classifier on the hidden state at block k of a noised input $\mathbf{x}_t$ and report ImageNet top-1 accuracy for both RAD and REPA trained for 400K iterations (see App. A.4 for details). We make two key observations from Fig. 6. First, RAD learns more discriminative features than REPA (Fig. 6, top). At the aligned block, where both models regress the reference DINOv2 features, the two have similar linear probing accuracy t. For the remaining blocks, RAD is more accurate than REPA by up to 4.4 points at $t = 0.1$. More interestingly, the linear probing accuracy of RAD decreases more slowly than that of REPA. Intuitively, this is likely due to feeding the representations back into the denoiser which keeps the later blocks linearly separable. Second, the features improve with scale (Fig. 6, bottom). RAD-XL/2 improves over RAD-B/2 by 8 to 9 points consistently for the most part of the diffusion trajectory $t \leq 0.5$ (71.7% vs. 63.4% at $t = 0.1$).

3.4 Generalization Capabilities

RAD benefits REG convergence. REG denoises the class token of the encoder jointly with the image tokens. We extend REG within the RAD framework by treating the projection of the intermediate noisy patch tokens in REG as the mean of the representation policy (see App. A.4 for details). Table 2 shows that combining REG with our approach improves the training convergence of REG at both model scales. Notably, REG combined with RAD converges twice as fast: on SiT-B/2 it matches the 200K-iteration FID of REG after 100K iterations, and on SiT-L/2 it matches the 400K-iteration FID of REG after 200K iterations (Table 7).

RAD generalizes to Text-to-Image Generation Table 3 shows that RAD outperforms REPA on text-to-image synthesis in both sample quality and compositional accuracy at both scales. The gains are largest for the smaller backbone: RAD-B/2 reduces FID from 10.77 to 8.42 and improves GenEval from 0.138 to 0.166, while RAD-XL/2 improves both metrics as well (FID 3.94 vs. 4.31, GenEval 0.266 vs. 0.258).

Table 4: Elucidating the role of different design component in RAD. FLOPs is the increase in inference FLOPs over REPA (23.0 GFLOPs). Best β_{inf} per row.

Training configuration	FID↓	Δ	FLOPs
A REPA [51]	22.9	–	–
B + $\mathbf{u}_t \sim \mathcal{N}(.) \rightarrow$ denoiser	10.3	−12.5	+8.4%
C + Flow policy	8.7	−1.6	+11.8%
D + Control Norm + Mod	8.2	−0.6	+11.8%
E + Early inject + p=1.0	**7.7**	−0.5	+11.8%

Table 5: Ablation: Impact of the choice of external encoder, injection depth and alignment loss weighting. Sample quality on ImageNet 256×256 of SiT-B/2 models trained for 400K iters

Method	Encoder	Depth	p	FID↓	sFID↓	IS↑	Prec.↑	Rec.↑
SiT-B/2 [26]	–	–	–	33.0	6.5	43.7	0.53	0.63
+ REPA [51]	DINOv2-B	4	–	24.4	6.4	59.9	0.59	0.65
+ REG [46]	DINOv2-B	4	–	15.2	6.9	94.6	0.64	0.63
+ RAD (ours): encoder								
	DINOv2-B	4	0	**8.2**	6.7	129.9	0.75	0.54
	DINOv2-L	4	0	8.4	7.0	129.8	0.76	0.54
	SigLIP-L	4	0	8.9	7.1	133.1	0.77	0.51
+ RAD (ours): injection depth								
	DINOv2-B	2	0	**8.2**	7.1	141.9	0.78	0.51
	DINOv2-B	4	0	8.5	6.1	129.5	0.76	0.53
	DINOv2-B	6	0	10.5	7.4	108.8	0.71	0.55
	DINOv2-B	8	0	15.1	7.7	85.6	0.66	0.56
+ RAD (ours): alignment loss weighting								
	DINOv2-B	4	0	8.5	6.1	129.5	0.76	0.53
	DINOv2-B	4	0.5	8.7	6.5	125.7	0.75	0.53
	DINOv2-B	4	1.0	**8.3**	6.7	129.0	0.75	0.53
	DINOv2-B	4	1.5	8.6	6.5	129.6	0.75	0.52

3.5 Ablation Experiments

System Analysis. Table 4 isolates the contribution of each component of RAD towards sample quality. We start from REPA (config A) and add components sequentially, ending at the configuration used in Table 1 (config E). We emphasize that conditioning the denoiser on representations $\mathbf{u}_t$ drawn from a Gaussian policy of Eq. 6 (config B) already lowers the FID from 22.9 to 10.3. Using a more expressive representation policy, the normalizing flow of Eq. 38 (config C), lowers FID further from 10.3 to 8.7, indicating that more expressive policies can be readily adopted with RAD. Control normalization and modulation (config D) and earlier injection with the weighted alignment loss (config E) add smaller gains resulting in FID of 7.7.

Additional compute costs. Our control pathway is relatively lightweight. At B/2 model scale, the whole pathway costs 12% additional inference FLOPs over REPA, most of it from the projector h_ϕ, which REPA also trains but discards at inference. However, the control pathway has the same architecture for L/2 and XL/2: therefore the overhead falls to 3.5% at L/2 and 2.5% at XL/2.

Design Choices. Table 5 highlights the impact of the choice of external teacher, the representation injection depth and the alignment loss weighting in RAD-B/2. RAD is relatively robust to the choice of the encoder. Similar to REPA and REG, we find that DINOv2-B works the best and fix this choice for all our analysis. Moreover, we find that an injection depth of 2 and an alignment weighting coefficient of $p = 1$ works the best at RAD-B/2 scale.

4 Conclusion

We introduced Representation-Augmented Diffusions (RAD), a class of diffusion models that embed semantic representations as latent variables within the denoising process: at every step, the denoiser conditions on a representation predicted from its own noisy state. The framework recovers representation alignment methods as special cases and reveals a simple design choice they overlook, feeding the aligned representation back into the denoiser. RAD trains faster, is more data- and parameter-efficient, needs fewer sampling steps, and learns more discriminative representations.

Limitations. The gains in sample quality come at the cost of diversity. RAD attains substantially higher precision than REPA but lower recall (e.g., 0.78 vs. 0.59 precision and 0.49 vs. 0.65 recall on SiT-B/2 at 400K iterations; Table 6), and its recall stays roughly constant as sample quality improves with training and scale. Improving this fidelity–diversity tradeoff, for instance through the variance of the representation policy or the sampler, is an important direction for future work.

AI USE STATEMENT

In this work, we used generative AI tools for formulating mathematical claims, assisting in the writing of proofs design, and implementing methods. We have not used generative AI tools for helping develop theoretical models or conceptual frameworks, providing critical ingredients for proving mathematical claims, or interpreting results. The categories generating synthetic data sets, proposing or refining hypotheses, assisting with translation, cleaning and reformatting datasets, and supporting qualitative and thematic data analysis are not applicable to this work.

Additionally, we used generative AI tools for creating or modifying scientific figures or images, creating or editing software code, drafting parts of our research paper, summarizing or analyzing existing literature, sourcing/searching for information, editing our research paper to improve readability, and identifying relevant literature. We have reviewed all AI-assisted work by verifying LLM-generated code by two authors, verifying information, and by checking for potential plagiarism for research ideas through manual literature survey. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

The authors do not feel that their paper submission raises any questions regarding the ICLR Code of Ethics beyond the general risks inherent to developing and distributing AI models.

REPRODUCIBILITY STATEMENT

We have included all details necessary to replicate our experiments, and all details necessary to derive our method in the Appendix. We will publish all our code upon publication.

REFERENCES

[1] Michael S. Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *International Conference on Learning Representations*, 2023.

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

[3] Hila Chefer, Patrick Esser, Dominik Lorenz, Dustin Podell, Vikash Raja, Vinh Tong, Antonio Torralba, and Robin Rombach. Self-supervised flow matching for scalable multi-modal synthesis. In *International Conference on Machine Learning*, 2026.

[4] Jiuhai Chen, Zhiyang Xu, Xichen Pan, Yushi Hu, Can Qin, Tom Goldstein, Lifu Huang, Tianyi Zhou, Saining Xie, Silvio Savarese, Le Xue, Caiming Xiong, and Ran Xu. BLIP3-o: A family of fully open unified multimodal models – architecture, training and dataset. *arXiv preprint arXiv:2505.09568*, 2025.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[6] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, 2021.

[7] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP. In *International Conference on Learning Representations*, 2017.

[8] Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. GenEval: An object-focused framework for evaluating text-to-image alignment. In *Advances in Neural Information Processing Systems*, 2023.

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014.

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

[11] Yasaman Haghighi, Bastien van Delft, Mariam Hassan, and Alexandre Alahi. LayerSync: Self-aligning intermediate layers. In *International Conference on Learning Representations*, 2026.

[12] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, 2017.

[13] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2022.

[14] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

[15] Dengyang Jiang, Mengmeng Wang, Liuzhuozheng Li, Lei Zhang, Haoyu Wang, Wei Wei, Guang Dai, Yanning Zhang, and Jingdong Wang. No other representation component is needed: Diffusion transformers can provide representation guidance by themselves. In *International Conference on Learning Representations*, 2026.

[16] Diederik P. Kingma and Ruiqi Gao. Understanding diffusion objectives as the ELBO with simple data augmentation. In *Advances in Neural Information Processing Systems*, 2023.

[17] Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

[18] Diederik P. Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. In *Advances in Neural Information Processing Systems*, 2021.

[19] Felix Krause, Timy Phan, Ming Gui, Stefan Andreas Baumann, Vincent Tao Hu, and Björn Ommer. TREAD: Token routing for efficient architecture-agnostic diffusion training. In *IEEE/CVF International Conference on Computer Vision*, pp. 15703–15713, 2025.

[20] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. In *Advances in Neural Information Processing Systems*, 2019.

[21] 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 *IEEE/CVF International Conference on Computer Vision*, pp. 18262–18272, 2025.

[22] Tianhong Li, Dina Katabi, and Kaiming He. Return of unconditional generation: A self-supervised representation generation method. In *Advances in Neural Information Processing Systems*, 2024.

[23] 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.

[24] Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

[25] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

[26] 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*, 2024.

[27] Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworldmodel: Stable end-to-end joint-embedding predictive architecture from pixels, 2026. URL https://arxiv.org/abs/2603.19312.

[28] Charlie Nash, Jacob Menick, Sander Dieleman, and Peter W. Battaglia. Generating images with sparse representations. In *International Conference on Machine Learning*, 2021.

[29] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

[30] Kushagra Pandey, Avideep Mukherjee, Piyush Rai, and Abhishek Kumar. DiffuseVAE: Efficient, controllable and high-fidelity generation from low-dimensional latents. *Transactions on Machine Learning Research*, 2022.

[31] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision*, 2023.

[32] Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[33] Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

[34] Rajesh Ranganath, Dustin Tran, and David Blei. Hierarchical variational models. In *International conference on machine learning*, pp. 324–333. PMLR, 2016.

[35] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[36] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. In *Advances in Neural Information Processing Systems*, 2016.

[37] Jaskirat Singh, Xingjian Leng, Zongze Wu, Liang Zheng, Richard Zhang, Eli Shechtman, and Saining Xie. What matters for representation alignment: Global information or spatial structure? In *International Conference on Learning Representations*, 2026.

[38] Abhishek Sinha, Jiaming Song, Chenlin Meng, and Stefano Ermon. D2C: Diffusion-decoding models for few-shot conditional generation. In *Advances in Neural Information Processing Systems*, 2021.

[39] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, 2015.

[40] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[41] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. In *Advances in Neural Information Processing Systems*, 2021.

[42] Chenyu Wang, Cai Zhou, Sharut Gupta, Zongyu Lin, Stefanie Jegelka, Stephen Bates, and Tommi Jaakkola. Learning diffusion models with flexible representation guidance. In *Advances in Neural Information Processing Systems*, 2025a.

[43] Runqian Wang and Kaiming He. Diffuse and disperse: Image generation with representation regularization. *arXiv preprint arXiv:2506.09027*, 2025.

[44] Yingheng Wang, Yair Schiff, Aaron Gokaslan, Weishen Pan, Fei Wang, Christopher De Sa, and Volodymyr Kuleshov. InfoDiffusion: Representation learning using information maximizing diffusion models. In *International Conference on Machine Learning*, 2023.

[45] Ziqiao Wang, Wangbo Zhao, Yuhao Zhou, Zekai Li, Zhiyuan Liang, Mingjia Shi, Xuanlei Zhao, Pengfei Zhou, Kaipeng Zhang, Zhangyang Wang, Kai Wang, and Yang You. REPA works until it doesn't: Early-stopped, holistic alignment supercharges diffusion training. In *Advances in Neural Information Processing Systems*, 2025b.

[46] Ge Wu, Shen Zhang, Ruijing Shi, Shanghua Gao, Zhenyuan Chen, Lei Wang, Zhaowei Chen, Hongcheng Gao, Yao Tang, Jian Yang, Ming-Ming Cheng, and Xiang Li. Representation entanglement for generation: Training diffusion transformers is much easier than you think. In *Advances in Neural Information Processing Systems*, 2025.

[47] Weilai Xiang, Hongyu Yang, Di Huang, and Yunhong Wang. Denoising diffusion autoencoders are unified self-supervised learners. In *International Conference on Computer Vision*, 2023.

[48] Shaodong Xu, Zhendong Wang, Litong Gong, Zexian Li, Wengang Zhou, Tiezheng Ge, and Houqiang Li. Beyond point-wise matching: Structural representation alignment for accelerating diffusion transformers. *arXiv preprint arXiv:2605.16949*, 2026.

[49] Yilun Xu, Gabriele Corso, Tommi Jaakkola, Arash Vahdat, and Karsten Kreis. DisCo-Diff: Enhancing continuous diffusion models with discrete latents. In *International Conference on Machine Learning*, 2024.

[50] Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[51] Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In *International Conference on Learning Representations*, 2025.

[52] Zijian Zhang, Zhou Zhao, and Zhijie Lin. Unsupervised representation learning from pre-trained diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2022.

[53] Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. UniPC: A unified predictor-corrector framework for fast sampling of diffusion models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL `https://openreview.net/forum?id=hrkmlPhp1u`.

[54] Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. In *International Conference on Learning Representations*, 2026.

[55] Hongkai Zheng, Weili Nie, Arash Vahdat, and Anima Anandkumar. Fast training of diffusion models with masked transformers. *Transactions on Machine Learning Research*, 2024.

[56] Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. In *International Conference on Machine Learning*, 2025.

[57] Rui Zhu, Yingwei Pan, Yehao Li, Ting Yao, Zhenglong Sun, Tao Mei, and Chang Wen Chen. SD-DiT: Unleashing the power of self-supervised discrimination in diffusion transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

A Appendix

A.1 Background

A.1.1 Stochastic Interpolants

Flow matching or Stochastic Interpolants convert data to noise by defining a noising process $\mathbf{x}_t = \alpha_t \mathbf{x}_0 + \sigma_t \boldsymbol{\epsilon}$, $t \in [0, 1]$. A commonly adopted choice is the straight-line interpolant (Liu et al., 2023) where $\alpha_t = 1 - t$ and $\sigma_t = t$. The marginals of $\mathbf{x}_t$ are transported by the velocity field $\boldsymbol{v}(\mathbf{x}_t, t) = \mathbb{E}[\dot{\alpha}_t \mathbf{x}_0 + \dot{\sigma}_t \boldsymbol{\epsilon} \mid \mathbf{x}_t]$, which a network $\boldsymbol{v}_\theta$ learns by regressing onto the conditional velocity target,

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{t, \mathbf{x}_0, \boldsymbol{\epsilon}} \big[\| \boldsymbol{v}_\theta(\mathbf{x}_t, t) - (\dot{\alpha}_t \mathbf{x}_0 + \dot{\sigma}_t \boldsymbol{\epsilon}) \|_2^2 \big]. \tag{10}$$

During inference, the *probability-flow* ODE $\mathrm{d}\mathbf{x}_t = \boldsymbol{v}_\theta(\mathbf{x}_t, t)\, \mathrm{d}t$, started from $\mathbf{x}_1 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, converts noise back to data. For all our experiments in this work, we use the widely adopted Scalable Interpolant Transformer (SiT) as the denoiser backbone in combination with the straight-line interpolants.

A.1.2 Representation Alignment

Representation alignment adds an auxiliary loss to the velocity matching objective in Eq. 10 that pulls the hidden states of the velocity field toward a semantic target. Let $\mathbf{h}_t = \boldsymbol{v}_\theta^{(\ell)}(\mathbf{x}_t, t)$ denote the hidden state of the denoiser $\boldsymbol{v}_\theta$ after ℓ SiT blocks. The main idea behind representation alignment is to align the outputs $\mathbf{y} = f_{\mathrm{pre}}(\mathbf{x}_0) \in \mathbb{R}^{N \times D}$ with the intermediate layer representations $\boldsymbol{h}_t$, where f_{pre} is a frozen pretrained encoder such as DINOv2, applied to the clean image. A trainable MLP projector h_ϕ maps the hidden state to the encoder's dimension, and the two are aligned token by token with a cosine loss,

$$\mathcal{L}_{\mathrm{REPA}}(\theta, \phi) = -\mathbb{E}_{t, \mathbf{x}_0, \boldsymbol{\epsilon}} \Big[\frac{1}{N} \sum_{n=1}^{N} \cos \big(\mathbf{y}^{[n]}, h_\phi(\mathbf{h}_t)^{[n]} \big) \Big], \qquad \mathcal{L} = \mathcal{L}_{\mathrm{FM}} + \lambda \mathcal{L}_{\mathrm{REPA}}, \tag{11}$$

where $\lambda > 0$ is a scalar hyperparameter. In most representation alignment methods, the projector is discarded after training i.e. at sampling time the denoiser receives $\mathbf{x}_t$ alone. In contrast, we propose a class of diffusion models where representations are natively embedded within the denoising process and help guide generation (see Sec. 2). We show that this can largely benefit a number of aspects during training and inference of diffusion models (see Sec. yy).

A.2 Related Work

External representation alignment. A recent line of work accelerates diffusion training by borrowing the features of a frozen pretrained vision encoder. REPA pulls the hidden states of a diffusion transformer toward the DINOv2 features of the clean image and reports large gains in convergence speed and sample quality. Follow-up work varies the target and the point at which it enters the network: REG denoises the encoder's class token jointly with the image, iREPA attributes the gains to the spatial structure of the target and replaces the projector with a convolutional one, and REPA-E (Leng et al., 2025) propagates the alignment loss into the tokenizer. A related direction aligns the latent space of the tokenizer itself with foundation-model features, either by regularizing the VAE (Yao et al., 2025) or by replacing it with a frozen encoder and a trained decoder (Zheng et al., 2026). In all of these methods the representation acts as a training-time regularizer, and the sampler receives $\mathbf{x}_t$ alone. RAD keeps the same external teacher but conditions every denoising step on the representation, and the alignment loss arises as a KL term in a likelihood bound instead of as a hand-weighted addition to the denoising loss.

Internal representation alignment. A second family removes the external encoder and lets the model teach itself. Masked-token objectives (Zheng et al., 2024; Zhu et al., 2024; Krause et al., 2025) drop or route a subset of tokens and ask the network to reconstruct them while denoising. SRA and LayerSync (Haghighi et al., 2026) align the features of an earlier block at a higher noise level with those of a deeper block at a lower noise level, using an EMA copy of the network or the network itself as the teacher. Self-Flow (Chefer et al., 2026) noises a random subset of the

student's tokens and pulls its features toward an EMA teacher that sees the cleaner view, which yields a self-supervised signal that extends beyond images. Dispersive loss (Wang & He, 2025) instead spreads the hidden states apart with a contrastive-style regularizer that needs no positive pairs. These methods share the design of external alignment: an auxiliary loss on the hidden states that leaves the generative model unchanged. Our framework does not prescribe where the target comes from, but we find that an external teacher is needed to realize the gains; drawing the target from the model's own features reverts performance to that of REPA.

Combining VAEs with diffusion models. Latent-variable constructions have long been used to equip diffusion models with a compact code. Latent diffusion and LSGM (Vahdat et al., 2021) run the diffusion process in the latent space of an autoencoder, which lowers cost but treats the latent as a compressed image rather than as a semantic representation. Diffusion autoencoders (Preechakul et al., 2022), PDAE (Zhang et al., 2022), and InfoDiffusion (Wang et al., 2023) learn a global latent with an auxiliary encoder and condition the entire denoising trajectory on it, whereas DiffuseVAE (Pandey et al., 2022) and D2C (Sinha et al., 2021) pair a VAE with a diffusion decoder that refines its output. More recently, RCG (Li et al., 2024) generates a self-supervised representation with a diffusion prior and conditions image generation on it, and DisCo-Diff (Xu et al., 2024) augments a diffusion model with discrete latents inferred by an encoder. RAD shares the idea of a learned latent that steers generation but differs in two respects. First, the latent is a per-step control $\mathbf{u}_t$ with its own prior conditioned on the current state, so the model is a conditional VAE (Sohn et al., 2015) at every transition rather than a single global code. Second, the posterior over the control is anchored to a pretrained encoder, which is what connects the framework to representation alignment and lets it recover REPA as a special case.

A.3 Derivations

A.3.1 Variational Bound

We derive the variational bound in Eq. 4, using the noising process in Eq. 3 as the variational distribution. Recall the generative process,

$$p_{\theta,\phi}(\mathbf{x}_{0:T}, \mathbf{u}_{1:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)\, p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t), \tag{12}$$

and the noising process,

$$q_{\theta'}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T} \mid \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1})\, q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\le t}), \tag{13}$$

where $\mathbf{x}_{\le t} = (\mathbf{x}_0, \dots, \mathbf{x}_t)$. The reference policy may depend on any subset $\mathbf{x}_S$ of these states, $S \subseteq \{0, \dots, t\}$, i.e., $q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\le t}) = q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_S)$; for example, $S = \{0\}$ for the pretrained encoder in Sec. 2.3. Conditioning on $\mathbf{x}_{\le t}$ covers all such choices, and the derivation below holds for any of them, including a reference policy whose parameters θ' are not optimized, such as a frozen pretrained encoder. We also use that the noising process is Markov, i.e., $q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = q(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{x}_0)$.

Using the noising process as the variational distribution and applying Jensen's inequality, we obtain the evidence lower bound,

$$\begin{aligned} \log p_{\theta,\phi}(\mathbf{x}_0) &= \log \mathbb{E}_{q_{\theta'}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T} \mid \mathbf{x}_0)} \left[\frac{p_{\theta,\phi}(\mathbf{x}_{0:T}, \mathbf{u}_{1:T})}{q_{\theta'}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T} \mid \mathbf{x}_0)} \right] \\ &\ge \mathbb{E}_{q_{\theta'}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T} \mid \mathbf{x}_0)} \left[\log \frac{p_{\theta,\phi}(\mathbf{x}_{0:T}, \mathbf{u}_{1:T})}{q_{\theta'}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T} \mid \mathbf{x}_0)} \right]. \end{aligned} \tag{14}$$

Substituting the factorizations in Eqs. 12 and 13, the log-ratio in Eq. 14 becomes

$$\log \frac{p_{\theta,\phi}(\mathbf{x}_{0:T}, \mathbf{u}_{1:T})}{q_{\theta'}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T} \mid \mathbf{x}_0)} = \log \frac{p(\mathbf{x}_T) \prod_{t=1}^{T} p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)\, p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t)}{\prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1})\, q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\le t})}. \tag{15}$$

For $t \ge 2$, the Markov property and Bayes' rule give $q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)\, q(\mathbf{x}_t \mid \mathbf{x}_0) / q(\mathbf{x}_{t-1} \mid \mathbf{x}_0)$. In the product over t, the ratios of marginals cancel and, together with the factor

$q(\mathbf{x}_1 \mid \mathbf{x}_0)$ at $t = 1$, leave only $q(\mathbf{x}_T \mid \mathbf{x}_0)$:

$$\prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = q(\mathbf{x}_T \mid \mathbf{x}_0) \prod_{t=2}^{T} q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0). \tag{16}$$

Substituting Eq. 16 into Eq. 15 and grouping the factors of each step,

$$\begin{aligned} \log \frac{p_{\theta,\phi}(\mathbf{x}_{0:T}, \mathbf{u}_{1:T})}{q_{\theta'}(\mathbf{x}_{1:T}, \mathbf{u}_{1:T} \mid \mathbf{x}_0)} = \sum_{t=1}^{T} \Big[\log p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t) + \log \frac{p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)}{q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\le t})} \Big] \\ - \sum_{t=2}^{T} \log q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) + \log \frac{p(\mathbf{x}_T)}{q(\mathbf{x}_T \mid \mathbf{x}_0)}. \end{aligned} \tag{17}$$

Reducing the expectations. The summand with index t in Eq. 17 depends only on $(\mathbf{x}_{\le t}, \mathbf{u}_t)$, and no state of the noising process depends on a representation, so all other variables integrate out of the expectation in Eq. 14. We write $\mathbb{E}_{\mathbf{x}_{\le t}, \mathbf{u}_t} := \mathbb{E}_{q(\mathbf{x}_{1:t} \mid \mathbf{x}_0)} \mathbb{E}_{q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\le t})}$ for the resulting expectation and $\mathbb{E}_{\mathbf{x}_{\le t}}$ for its outer expectation.

Simplifying the terms. The first term of Eq. 17 gives the denoising term $\mathbb{E}_{\mathbf{x}_{\le t}, \mathbf{u}_t}[\log p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t)]$. For the remaining terms,

$$\mathbb{E}_{\mathbf{x}_{\le t}, \mathbf{u}_t} \left[\log \frac{p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)}{q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\le t})} \right] = -\mathbb{E}_{\mathbf{x}_{\le t}} \big[D_{\mathrm{KL}}\big(q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\le t}) \,\|\, p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)\big) \big], \tag{18}$$

$$-\mathbb{E}_{\mathbf{x}_{\le t}, \mathbf{u}_t} \big[\log q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) \big] = \mathbb{E}_{q(\mathbf{x}_t \mid \mathbf{x}_0)} \big[\mathcal{H}\big(q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)\big) \big] =: \mathcal{H}_t, \tag{19}$$

$$\mathbb{E}_{q(\mathbf{x}_T \mid \mathbf{x}_0)} \left[\log \frac{p(\mathbf{x}_T)}{q(\mathbf{x}_T \mid \mathbf{x}_0)} \right] = -D_{\mathrm{KL}}\big(q(\mathbf{x}_T \mid \mathbf{x}_0) \,\|\, p(\mathbf{x}_T)\big). \tag{20}$$

Eq. 18 is the alignment term. Eq. 19 holds because the integrand depends only on $(\mathbf{x}_{t-1}, \mathbf{x}_t)$, whose marginal is $q(\mathbf{x}_t \mid \mathbf{x}_0)\, q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)$ by the Markov property.

Combining. Substituting these terms into Eq. 14, we arrive at

$$\begin{aligned} \log p_{\theta,\phi}(\mathbf{x}_0) \ge \sum_{t=1}^{T} \Big[\underbrace{\mathbb{E}_{\mathbf{x}_{\le t}, \mathbf{u}_t} \big[\log p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t) \big]}_{\text{Denoising}} \\ \underbrace{- \mathbb{E}_{\mathbf{x}_{\le t}} \big[D_{\mathrm{KL}}\big(q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\le t}) \,\|\, p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)\big) \big]}_{\text{Alignment}} \Big] + C, \end{aligned} \tag{21}$$

where $C = \sum_{t=2}^{T} \mathcal{H}_t - D_{\mathrm{KL}}(q(\mathbf{x}_T \mid \mathbf{x}_0) \,\|\, p(\mathbf{x}_T))$ is determined by the noising process alone and does not depend on (θ, ϕ). The KL term in C vanishes when $q(\mathbf{x}_T \mid \mathbf{x}_0) = p(\mathbf{x}_T) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, as for the interpolant in Sec. 2.3. Omitting C gives Eq. 4. This concludes the proof.

Remark. In standard diffusion models, the denoising term and the entropy $\mathcal{H}_t$ combine into $-\mathbb{E}\big[D_{\mathrm{KL}}(q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) \,\|\, p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t))\big]$. In RAD, the same holds for $t \ge 2$ only if $\mathbf{u}_t$ is conditionally independent of $\mathbf{x}_{t-1}$ given $(\mathbf{x}_t, \mathbf{x}_0)$, e.g., for $S = \{0\}$. In general, $q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\le t})$ depends on $\mathbf{x}_{t-1}$, and the denoising term keeps the log-likelihood form of Eq. 4.

A.3.2 Simplified Training Objective

We derive Eq. 9 from the bound in Eq. 21 under the parameterization of Sec. 2.3. Throughout, $t \in \{1, \dots, T\}$ indexes a uniform grid on $[0, 1]$, and α_t, σ_t, $\dot{\alpha}_t$, $\dot{\sigma}_t$ denote the interpolant schedule and its time derivative at the corresponding grid time. Since in our experiments the reference policy depends on $\mathbf{x}_0$ alone, $q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\le t}) = q(\mathbf{u}_t \mid \mathbf{x}_0)$, the alignment term depends on $(\mathbf{x}_0, \mathbf{x}_t)$ only, and $\mathbf{u}_t$ is conditionally independent of $\mathbf{x}_{t-1}$ given $(\mathbf{x}_t, \mathbf{x}_0)$.

Alignment term. Both $q(\mathbf{u}_t \mid \mathbf{x}_0) = \mathcal{N}(\alpha_t f_{\text{pre}}(\mathbf{x}_0), \beta^2\sigma_t^2 \boldsymbol{I})$ and $p_\phi(\mathbf{u}_t \mid \mathbf{x}_t) = \mathcal{N}(\alpha_t \boldsymbol{h}_\phi(\mathbf{x}_t, t), \beta^2\sigma_t^2 \boldsymbol{I})$ are isotropic Gaussians with the same covariance, so their KL divergence reduces to the squared distance between the means scaled by the precision $1/(\beta^2\sigma_t^2)$,

$$D_{\text{KL}}\big(q(\mathbf{u}_t \mid \mathbf{x}_0) \,\|\, p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)\big) = \frac{\text{SNR}(t)}{2\beta^2}\big\|f_{\text{pre}}(\mathbf{x}_0) - \boldsymbol{h}_\phi(\mathbf{x}_t, t)\big\|_2^2, \qquad \text{SNR}(t) = \frac{\alpha_t^2}{\sigma_t^2}. \quad (22)$$

If the reference variance is chosen differently from $\beta^2\sigma_t^2$, the KL changes only by terms that do not depend on ϕ, since the mean term involves the variance of p_ϕ alone.

Denoising term. The forward posterior of the Gaussian noising process is (Kingma et al., 2021)

$$q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}\big(\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0),\ \tilde{\sigma}_{t-1|t}^2 \boldsymbol{I}\big), \qquad \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) = a_t \mathbf{x}_t + b_t \mathbf{x}_0, \quad (23)$$

with $a_t = \alpha_{t|t-1}\sigma_{t-1}^2/\sigma_t^2$, $b_t = \alpha_{t-1}\sigma_{t|t-1}^2/\sigma_t^2$ and $\tilde{\sigma}_{t-1|t}^2 = \sigma_{t|t-1}^2\sigma_{t-1}^2/\sigma_t^2$, where $\alpha_{t|t-1} = \alpha_t/\alpha_{t-1}$ and $\sigma_{t|t-1}^2 = \sigma_t^2 - \alpha_{t|t-1}^2\sigma_{t-1}^2$ are the coefficients of the transition $q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$. For $t = 1$, $\mathbf{x}_0$ is observed and $(a_1, b_1) = (0, 1)$; we then use a fixed decoder variance in place of $\tilde{\sigma}_{0|1}^2$. Since $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t)$ is a Gaussian with the same variance, the expectation over $\mathbf{x}_{t-1}$ in the denoising term is

$$\mathbb{E}_{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}\big[\log p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t)\big] = -\frac{1}{2\tilde{\sigma}_{t-1|t}^2}\big\|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, \mathbf{u}_t, t)\big\|_2^2 + \text{const}, \quad (24)$$

where the constant collects the entropy of the posterior and the normalizer of p_θ, neither of which depends on θ. We parameterize $\boldsymbol{\mu}_\theta$ through the velocity network by substituting the estimate of $\mathbf{x}_0$ implied by $\boldsymbol{v}_\theta$ into the posterior mean. Solving $\mathbf{x}_t = \alpha_t \mathbf{x}_0 + \sigma_t \boldsymbol{\epsilon}$ and $\boldsymbol{v}_{\text{ref}} = \dot{\alpha}_t \mathbf{x}_0 + \dot{\sigma}_t \boldsymbol{\epsilon}$ for $\mathbf{x}_0$ gives

$$\mathbf{x}_0 = \frac{\sigma_t \boldsymbol{v}_{\text{ref}} - \dot{\sigma}_t \mathbf{x}_t}{\sigma_t\dot{\alpha}_t - \dot{\sigma}_t\alpha_t}, \qquad \hat{\mathbf{x}}_\theta(\mathbf{x}_t, \mathbf{u}_t, t) := \frac{\sigma_t \boldsymbol{v}_\theta(\mathbf{x}_t, \mathbf{u}_t, t) - \dot{\sigma}_t \mathbf{x}_t}{\sigma_t\dot{\alpha}_t - \dot{\sigma}_t\alpha_t}, \quad (25)$$

and we set $\boldsymbol{\mu}_\theta(\mathbf{x}_t, \mathbf{u}_t, t) := \tilde{\boldsymbol{\mu}}_t\big(\mathbf{x}_t, \hat{\mathbf{x}}_\theta(\mathbf{x}_t, \mathbf{u}_t, t)\big)$. For the straight-line interpolant, $\sigma_t\dot{\alpha}_t - \dot{\sigma}_t\alpha_t = -1$. Since $\tilde{\boldsymbol{\mu}}_t$ is affine in its second argument, Eq. 25 yields $\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) - \boldsymbol{\mu}_\theta(\mathbf{x}_t, \mathbf{u}_t, t) = b_t(\mathbf{x}_0 - \hat{\mathbf{x}}_\theta) = b_t\sigma_t(\boldsymbol{v}_{\text{ref}} - \boldsymbol{v}_\theta)/(\sigma_t\dot{\alpha}_t - \dot{\sigma}_t\alpha_t)$, and Eq. 24 becomes a weighted velocity regression,

$$\mathbb{E}_{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}\big[\log p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t)\big] = -w_t\big\|\boldsymbol{v}_\theta(\mathbf{x}_t, \mathbf{u}_t, t) - \boldsymbol{v}_{\text{ref}}\big\|_2^2 + \text{const}, \quad (26)$$
$$w_t = \frac{b_t^2\sigma_t^2}{2\tilde{\sigma}_{t-1|t}^2(\sigma_t\dot{\alpha}_t - \dot{\sigma}_t\alpha_t)^2}.$$

Combining. Substituting Eqs. 22 and 26 into Eq. 21 and writing the noising process as $\mathbf{x}_t = \alpha_t\mathbf{x}_0 + \sigma_t\boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, the negative bound is

$$-\log p_{\theta,\phi}(\mathbf{x}_0) \le T\,\mathbb{E}_{t,\boldsymbol{\epsilon},\,\mathbf{u}_t\sim q}\Big[w_t\big\|\boldsymbol{v}_\theta(\mathbf{x}_t, \mathbf{u}_t, t) - \boldsymbol{v}_{\text{ref}}\big\|_2^2 + \tfrac{\text{SNR}(t)}{2\beta^2}\big\|f_{\text{pre}}(\mathbf{x}_0) - \boldsymbol{h}_\phi(\mathbf{x}_t, t)\big\|_2^2\Big] + \text{const}, \quad (27)$$

with t uniform on $\{1, \dots, T\}$ and $\mathbf{u}_t \sim q(\mathbf{u}_t \mid \mathbf{x}_0)$. Following common practice (Kingma & Gao, 2023), we replace the weight w_t of the denoising term by one, which changes the relative weighting of the time steps in the denoising term. Dropping the factor T and the constant, and taking the expectation over $\mathbf{x}_0$, gives Eq. 9. The coefficient of the alignment term, $\text{SNR}(t)/(2\beta^2)$, grows without bound as $t \to 0$. In practice we therefore replace it as well, by its saturated counterpart $\lambda_t = \alpha_t^2/(\alpha_t^2 + \beta^2\sigma_t^2) = r_t/(1 + r_t)$ with $r_t = \text{SNR}(t)/\beta^2$, scaled by a constant of 0.5 (App. A.4). This keeps the ordering of the time steps dictated by the bound, with the alignment term weighted most at low noise, while bounding its weight by one. We sample t uniformly on $[0, 1]$.

A.4 Experimental Details

Data and latents. For class-conditional generation we use the ImageNet training set (Deng et al., 2009) of 1.28M images at 256×256, center-cropped as in ADM (Dhariwal & Nichol, 2021), without further augmentation. Following REPA, images are encoded once with the Stable Diffusion VAE (`sd-vae-ft-mse`) (Rombach et al., 2022) into $32 \times 32 \times 4$ latents; at every training step a latent is drawn from the stored posterior and scaled by 0.18215, and samples are decoded with the `ft-ema`

decoder. The external supervisor f_{pre} is a frozen DINOv2-B/14 (Oquab et al., 2024): the 256×256 image is normalized and resized to 224×224, and the 256 patch tokens of dimension 768 (the class token is dropped) serve both as alignment targets and as the clean representation $f_{\mathrm{pre}}(\mathbf{x}_0)$ from which the training-time controls $\mathbf{u}_t$ are noised.

Backbones and injection depth. We use the unmodified SiT-B/2 (12 blocks, width 768), SiT-L/2 (24 blocks, width 1024), and SiT-XL/2 (28 blocks, width 1152) architectures with a 2×2 patch size, so that the latent is processed as 256 tokens. The hidden state after block ℓ is aligned with the encoder and, for RAD, $\mathbf{u}_t$ is injected after the same block. RAD-B/2 uses $\ell = 2$ (Table 5); RAD-L/2 and RAD-XL/2 use $\ell = 8$, the depth at which REPA aligns these backbones. Controls that live in the 768-dimensional encoder space are mapped to the backbone width by a linear layer when the two differ (L/2 and XL/2).

Optimization. All models are trained with AdamW (Loshchilov & Hutter, 2019) at a constant learning rate of 10^{-4} without warmup, $(\beta_1, \beta_2) = (0.9, 0.999)$, no weight decay, gradient clipping at norm 1, and a global batch size of 256, in fp16 mixed precision, for 400K iterations unless stated otherwise. We keep an exponential moving average of the weights with decay 0.9999, updated every step, and use the averaged weights for every evaluation. The noising process is the linear interpolant $\alpha_t = 1 - t$, $\sigma_t = t$ with $t \sim \mathcal{U}(0, 1)$, the denoiser predicts the velocity, and the class label is dropped with probability 0.1. The seed is 0 throughout.

RAD hyperparameters. The projector $\boldsymbol{h}_\phi$ is the REPA MLP (width 2048, two hidden layers, SiLU) applied to every token. Models with the flow policy are trained with Eq. 40: the alignment term is weighted by $0.5\,\lambda_t$ and the flow likelihood by $1 - \lambda_t$, with $\beta\sigma_t$ clamped below by $\sigma_{\min} = 10^{-3}$ in the whitened residual. The adaptive weighting of the alignment loss multiplies the per-sample error e by $1/(\mathrm{sg}(e) + 10^{-3})^p$, where sg stops the gradient; $p = 0$ is the unweighted loss and $p = 1$ is used for the final models. The policy scale is $\beta = 1$ during training. The control pathway of App. A.9 uses $\gamma = 0.6$ and $\delta = 10^{-6}$ for the normalization, $M = 1$ control token, and zero-initialized modulation. The flow policy has four affine coupling layers with a hidden width of 512 and a 256-dimensional context computed from $\boldsymbol{h}_\phi(\mathbf{x}_t, t)$ by one linear layer; its last layers are zero-initialized, so training starts from the Gaussian policy. At inference we draw fresh policy noise at every step, keep the controls active over the whole trajectory, and sweep $\beta_{\mathrm{inf}} \in \{0.5, 0.6, \ldots, 0.9\}$, reporting the best value per checkpoint, which is 0.7 for B/2 and 0.8 for L/2 and XL/2.

Samplers. The Euler–Maruyama SDE sampler follows REPA: 250 uniform steps from $t = 1$ to $t = 0.04$ with diffusion coefficient $w_t = 2t$, followed by one deterministic step to $t = 0$. The Euler ODE sampler uses N uniform steps from $t = 1$ to $t = 0$, so that N steps cost N network evaluations; UniPC (Zhao et al., 2023) is run with the same step budgets at orders 2 and 3 (Table 10). No classifier-free guidance is used unless stated.

Evaluation. We report FID, sFID, Inception Score, precision, and recall computed with the ADM evaluation suite (Dhariwal & Nichol, 2021) against its ImageNet 256×256 reference batch, on 50k samples with uniformly drawn class labels. For the data-efficiency experiment (Fig. 3c, Table 9), we train SiT-B/2 on random subsets of 3%, 10%, and 30% of ImageNet for the same 400K iterations at batch size 256, evaluate every checkpoint with the Euler ODE sampler with 50 steps, and report the best FID of each run.

Linear probing. Following REPA and Xiang et al. (2023), we probe the averaged weights of each 400K-iteration model with the null class label. For a diffusion time t, every image is noised once, the hidden state after block k is mean-pooled over its tokens, and a parameter-free batch normalization followed by a linear layer is trained on the ImageNet training set for 90 epochs with Adam (learning rate 10^{-3}, cosine decay, batch size 16384) and scored by top-1 accuracy on the 50k validation images; no augmentation is used. For RAD, the controls are sampled from the policy as at inference, with $\beta_{\mathrm{inf}} = 0.7$ for B/2 (Fig. 6a) and 1.0 for XL/2; Fig. 6b reports, for each t, the best block and, for B/2, the best of $\beta_{\mathrm{inf}} \in \{0.7, 1.0\}$.

REG within RAD. REG denoises the class token of the encoder jointly with the image tokens and weights its class-token loss by 0.03. We keep this recipe unchanged and add the control pathway of

RAD with the hyperparameters above, using the projection of the noisy patch tokens as the mean of the representation policy; REG's class token is not part of the control.

Text-to-image. We train on the pre-training captions of BLIP3o (Chen et al., 2025) (the JourneyDB and short-caption subsets, about 9M image–caption pairs) at 256×256 with the same VAE latents. Captions are encoded by a frozen Qwen3-0.6B (Qwen Team, 2025); its final hidden states (dimension 1024) pass through an RMSNorm and a linear layer and are prepended in context to the 256 patch tokens with a key mask, and a pooled caption embedding is added to the adaLN conditioning through a small MLP. Captions are capped at 64 tokens, which truncates 3.5% of them, and dropped with probability 0.1. The alignment loss, the flow policy, and the controls act on the patch tokens only. RAD-B/2 injects after block 2 and RAD-XL/2 after block 8 with $p = 1$; XL/2 additionally uses QK-normalization in attention, without which training diverged. Pre-training follows the class-conditional recipe (400K iterations, batch size 256). For fine-tuning on BLIP3o-60k (57,470 pairs) we start a fresh optimizer from the averaged weights and train for 11,250 steps (50 epochs) with the learning rate chosen from $\{0.5, 1, 2, 3\} \times 10^{-4}$. Samples for evaluation are drawn with the Euler ODE sampler with 50 steps; FID is computed against 50k training images, and GenEval (Ghosh et al., 2023) generates four images for each of its 553 prompts and reports the mean of its six task accuracies.

Compute. Each model is trained on a single node with 8 NVIDIA RTX PRO 6000 Blackwell GPUs. At the observed throughputs of roughly 10, 7, and 5 iterations per second, 400K iterations take about 11, 16, and 22 hours for B/2, L/2, and XL/2, respectively; the text-to-image XL/2 model takes about 37 hours, and fine-tuning on BLIP3o-60k about 25 minutes. Generating 50k samples with 250 SDE steps takes about 1.5 hours on 4 GPUs, and linear probing about one hour for B/2 and a few hours for XL/2.

A.5 Additional Generation Results

Tables 6 and 7 report the full ImageNet 256×256 results of RAD at each training budget, alongside the numbers reported in the official papers of REPA and REG at the same budgets. All results use the Euler–Maruyama SDE sampler with 250 steps and no classifier-free guidance.

Table 6: Full ImageNet 256×256 results of SiT, REPA, and RAD at every training budget (numbers behind Table 1, Fig. 1c, and Fig. 3a,b). Baseline numbers are from the respective papers. All metrics on 50k samples, best β_{inf} per row, Euler–Maruyama SDE sampler with 250 steps, no guidance.

Model	#Params	Iter.	FID↓	sFID↓	IS↑	Prec.↑	Rec.↑
SiT-B/2 [26]	130M	400K	33.0	6.46	43.7	0.53	0.63
+ REPA [51]	130M	50K	78.2	11.71	17.1	0.33	0.48
+ REPA [51]	130M	100K	49.5	7.00	27.5	0.46	0.59
+ REPA [51]	130M	200K	33.2	6.68	43.7	0.54	0.63
+ REPA [51]	130M	400K	24.4	6.40	59.9	0.59	0.65
+ RAD (ours)	142M	50K	55.6	21.94	25.4	0.44	0.46
+ RAD (ours)	142M	100K	23.5	9.05	64.8	0.64	0.51
+ RAD (ours)	142M	200K	12.0	7.49	110.5	0.74	0.50
+ RAD (ours)	142M	400K	7.7	6.72	148.0	0.78	0.49
SiT-L/2 [26]	458M	400K	18.8	5.29	72.0	0.64	0.64
+ REPA [51]	458M	50K	55.4	24.0	23.0	0.43	0.53
+ REPA [51]	458M	100K	24.1	6.25	55.7	0.62	0.60
+ REPA [51]	458M	200K	14.0	5.18	86.5	0.67	0.64
+ REPA [51]	458M	400K	10.0	5.20	109.2	0.69	0.65
+ RAD (ours)	472M	50K	28.2	16.88	48.4	0.64	0.48
+ RAD (ours)	472M	100K	9.3	8.24	120.1	0.78	0.48
+ RAD (ours)	472M	200K	5.2	7.10	169.7	0.82	0.49
+ RAD (ours)	472M	400K	3.9	5.59	203.7	0.84	0.49
SiT-XL/2 [26]	675M	7M	8.3	6.32	131.7	0.68	0.67
+ REPA [51]	675M	50K	52.3	31.24	24.3	0.45	0.53
+ REPA [51]	675M	100K	19.4	6.06	67.4	0.64	0.61
+ REPA [51]	675M	200K	11.1	5.05	100.4	0.69	0.64
+ REPA [51]	675M	400K	7.9	5.06	122.6	0.70	0.65
+ REPA [51]	675M	4M	5.9	5.73	157.8	0.70	0.69
+ RAD (ours)	690M	50K	24.4	19.65	56.7	0.65	0.45
+ RAD (ours)	690M	100K	7.8	6.36	134.3	0.82	0.47
+ RAD (ours)	690M	200K	4.6	6.38	185.3	0.84	0.49
+ RAD (ours)	690M	400K	3.7	5.66	217.8	0.85	0.50

Table 7: Full ImageNet 256×256 results of REG and REG + RAD at every training budget (numbers behind Table 2). REG numbers are from its paper. All metrics on 50k samples, best β_{inf} per row, Euler–Maruyama SDE sampler with 250 steps, no guidance.

Model	#Params	Iter.	FID↓	sFID↓	IS↑	Prec.↑	Rec.↑
SiT-B/2 [26]	130M	400K	33.0	6.46	43.7	0.53	0.63
+ REG [46]	132M	50K	64.7	9.47	23.2	0.40	0.51
+ REG [46]	132M	100K	36.1	7.74	45.5	0.53	0.61
+ REG [46]	132M	200K	22.1	7.19	72.2	0.60	0.63
+ REG [46]	132M	400K	15.2	6.69	94.6	0.64	0.63
+ REG + RAD (ours)	143M	50K	50.5	20.54	28.1	0.45	0.51
+ REG + RAD (ours)	143M	100K	22.4	7.11	67.3	0.64	0.54
+ REG + RAD (ours)	143M	200K	12.9	6.60	102.8	0.70	0.54
+ REG + RAD (ours)	143M	400K	8.9	5.76	124.6	0.74	0.53
SiT-L/2 [26]	458M	400K	18.8	5.29	72.0	0.64	0.64
+ REG [46]	460M	50K	30.1	8.92	52.6	0.58	0.57
+ REG [46]	460M	100K	11.4	5.36	108.8	0.70	0.60
+ REG [46]	460M	200K	6.6	5.16	145.4	0.73	0.63
+ REG [46]	460M	400K	4.6	5.21	167.6	0.75	0.63
+ REG + RAD (ours)	474M	50K	23.7	11.03	61.9	0.62	0.53
+ REG + RAD (ours)	474M	100K	7.8	6.13	136.1	0.77	0.52
+ REG + RAD (ours)	474M	200K	4.6	5.81	183.6	0.80	0.52
+ REG + RAD (ours)	474M	400K	3.9	4.78	211.0	0.84	0.52

A.6 Additional Quantitative Results

Tables 8–12 list the values plotted in the figures of Sec. 3 that are not already covered by Tables 1, 6, and 7.

Table 8: Training convergence (Fig. 1c and Fig. 3a). FID-50k on ImageNet 256×256 against training iterations. Baseline numbers are from the respective papers; a dash marks a checkpoint that was not evaluated. Euler–Maruyama SDE sampler with 250 steps, no guidance.

Model	50K	100K	150K	200K	400K	1M	4M	7M
SiT-XL/2 [26]	–	–	–	–	17.2	–	–	8.3
REPA-B/2 [51]	78.2	49.5	–	33.2	24.4	–	–	–
REPA-XL/2 [51]	52.3	19.4	–	11.1	7.9	6.4	5.9	–
RAD-B/2 (ours)	55.6	23.5	–	12.0	7.7	–	–	–
RAD-XL/2 (ours)	24.4	7.8	5.7	4.6	3.7	–	–	–

Table 9: Data efficiency (Fig. 3c). FID-50k of SiT-B/2 models trained for 400K iterations on a random fraction of ImageNet 256×256, best checkpoint of each run. Euler ODE sampler with 50 steps, no guidance.

Fraction of ImageNet	3%	10%	30%	100%
REPA-B/2 [51]	40.1	23.8	25.7	26.6
RAD-B/2 (ours)	**33.7**	**14.9**	**9.7**	**8.9**

Table 10: Sampling efficiency (Fig. 4). FID-50k on ImageNet 256×256 against the number of sampling steps for RAD-XL/2 trained for 400K iterations and the released REPA-XL/2 checkpoint trained for 4M iterations. No guidance. With the 250-step Euler–Maruyama SDE sampler the same models reach 3.7 and 5.9, respectively.

Model	Sampler	5	10	25	50	100
REPA-XL/2 (4M) [51]	Euler	62.3	21.9	10.5	8.2	7.4
RAD-XL/2 (400K, ours)	Euler	20.6	6.8	4.1	4.0	–
RAD-XL/2 (400K, ours)	UniPC, order 2 [53]	6.5	4.5	**3.7**	**3.8**	–
RAD-XL/2 (400K, ours)	UniPC, order 3 [53]	**6.4**	**4.5**	3.8	**3.8**	–

Table 11: Linear probing per block (Fig. 6a). ImageNet top-1 accuracy (%) of a linear classifier on the hidden state at block k of REPA-B/2 and RAD-B/2 trained for 400K iterations, for a noised input at diffusion time t. Both models are aligned at block 4, after which RAD injects $\mathbf{u}_t$ (sampled with $\beta_{\text{inf}} = 0.7$). The better value per block and t is in bold.

	$t = 0.1$		$t = 0.25$		$t = 0.5$		$t = 0.75$	
Block	REPA	RAD	REPA	RAD	REPA	RAD	REPA	RAD
4	**63.0**	62.7	**59.1**	59.0	37.2	**37.8**	8.3	**8.6**
5	60.9	**63.4**	57.4	**59.2**	36.2	**36.4**	**8.3**	6.6
6	58.3	**62.1**	54.9	**58.1**	35.4	**36.0**	**8.2**	6.8
8	52.3	**56.7**	50.2	**53.9**	33.8	**34.6**	**8.1**	6.8
10	46.1	**48.8**	44.6	**47.6**	31.7	**32.4**	**8.2**	6.9
12	40.7	**43.3**	40.2	**43.3**	29.3	**30.5**	**7.8**	6.9

Table 12: Linear probing across model scale (Fig. 6b). Best ImageNet top-1 accuracy (%) over all blocks of RAD-B/2 and RAD-XL/2 trained for 400K iterations, at each diffusion time t, with the block that attains it in parentheses. B/2 is probed with $\beta_{\text{inf}} \in \{0.7, 1.0\}$ and XL/2 with $\beta_{\text{inf}} = 1.0$.

Model	$t = 0.1$	$t = 0.25$	$t = 0.5$	$t = 0.75$
RAD-B/2	63.4 (5)	59.2 (5)	37.8 (4)	8.6 (4)
RAD-XL/2	**71.7** (9)	**68.1** (9)	**46.1** (8)	**9.5** (8)

A.7 Additional Interpolation Results

Protocol. Given two real ImageNet images $\mathbf{x}_0^A$ and $\mathbf{x}_0^B$ with labels y_A and y_B, we take the mean of the learned policy on each noised image as its representation, $\bar{\mathbf{u}} = \boldsymbol{h}_\phi(\mathbf{x}_{t_c}, t_c)$ with $t_c = 0.1$, and invert each image to noise $\mathbf{x}_1$ with the Euler ODE sampler (50 steps). For a weight $m \in [0, 1]$, we then interpolate

$$\bar{\mathbf{u}}_m = m\,\bar{\mathbf{u}}^A + (1-m)\,\bar{\mathbf{u}}^B, \qquad \mathbf{x}_{1,m} = \mathrm{slerp}(\mathbf{x}_1^A, \mathbf{x}_1^B; m), \qquad \boldsymbol{e}_m = m\,\boldsymbol{e}_{y_A} + (1-m)\,\boldsymbol{e}_{y_B}, \tag{28}$$

where $\boldsymbol{e}_y$ is the class embedding. Each sample is generated from $\mathbf{x}_{1,m}$ and $\boldsymbol{e}_m$ with the same sampler, where at every step $\mathbf{u}_t$ is sampled from the policy in Eq. 6 with $\boldsymbol{h}_\phi(\mathbf{x}_t, t)$ replaced by $\bar{\mathbf{u}}_m$. Hence, $m = 1$ and $m = 0$ reconstruct A and B, respectively. REPA does not pass the representations back into the denoiser, so its generations cannot be manipulated through them directly. Hence, we interpolate $\mathbf{x}_1$ and the class embedding in the same way.

Additional samples. We show additional interpolations between real ImageNet images across classes (Fig. 7) and within the same class (Fig. 8), using the protocol above and the checkpoints of Sec. 3.3. For same-class pairs, the class embedding is fixed, so REPA interpolates $\mathbf{x}_1$ alone. As in Fig. 5, RAD preserves the pose and framing of the endpoints while the appearance changes gradually, e.g., the iguana's pale scales darken into the textured skin of the Komodo dragon, the eye of the green chameleon grows into the turreted eye of the African chameleon, and the white-lit castle turns into the orange-lit palace while the night scene and its reflection on the water are kept. Within the same class, RAD keeps the side profile of the wood rabbit as the background changes from dry brush to green foliage. REPA again does not remain consistent: the head of the iguana dissolves into a full-body lizard among leaves, the close-up of the chameleon is replaced by full-body chameleons on branches, and the wood rabbit is replaced by a rounder, crouched rabbit facing the opposite direction for $0.35 \le m \le 0.5$.

A.8 Connections to Prior Work

Next, we discuss our choices for the parameterization of the distributions in Eq. 4. We begin with the two distributions of the alignment term, the inference distribution $q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\le t})$ and the generative distribution $p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)$. Prior work on representation alignment can be broadly divided by the source of the representation: an external pretrained model (external alignment, App. A.8.1) or the model itself (internal alignment, App. A.8.2). Our framework accommodates both, and the two settings differ only in the inference distribution. We then describe the representation-augmented denoiser $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t)$ and the forward process, which are shared by the two settings (App. A.8.3), and close with the resulting objective.

A.8.1 External Alignment

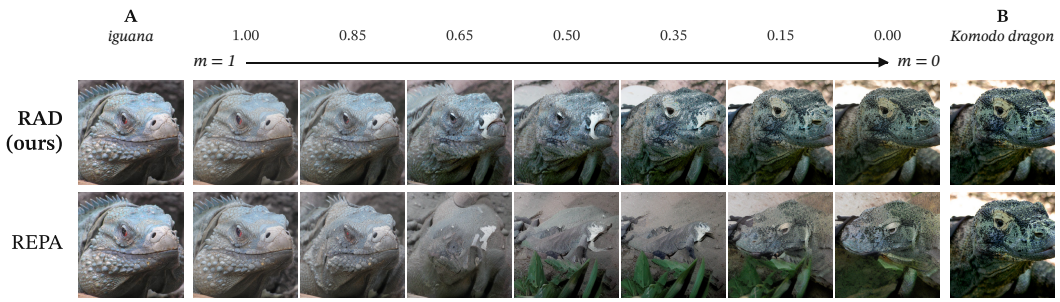
In this setting, a frozen pretrained encoder f_{pre} extracts the intermediate representations $\mathbf{u}_t$ from the clean image $\mathbf{x}_0$. Following REPA, f_{pre} can be DINOv2 or any other pretrained model that produces semantically meaningful features. Since the supervision comes from $\mathbf{x}_0$ alone, the inference distribution drops its dependence on the trajectory and carries no learnable parameters, $q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\le t}) = q(\mathbf{u}_t \mid \mathbf{x}_0)$. We parameterize it, together with the generative distribution of $\mathbf{u}_t$, as Gaussians:

$$q(\mathbf{u}_t \mid \mathbf{x}_0) = \mathcal{N}\left(\alpha_t f_{\mathrm{pre}}(\mathbf{x}_0),\ \beta^2\sigma_t^2 \boldsymbol{I}\right), \tag{29}$$

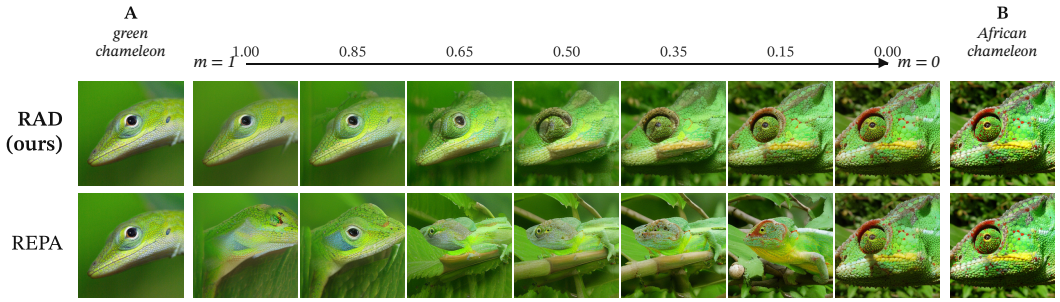
$$p_\phi(\mathbf{u}_t \mid \mathbf{x}_t) = \mathcal{N}\left(\alpha_t h_\phi(\mathbf{h}_t),\ \beta^2\sigma_t^2 \boldsymbol{I}\right), \qquad \mathbf{h}_t = f_\theta(\mathbf{x}_t). \tag{30}$$

Here (α_t, σ_t) is the interpolant schedule of App. A.1.1, shared between $\mathbf{x}_t$ and $\mathbf{u}_t$ so that the representation sits at the same signal-to-noise ratio as the state: $\mathbf{u}_t$ is pure noise at the terminal time and approaches the clean features $f_{\mathrm{pre}}(\mathbf{x}_0)$ as $t \to 0$. On the generative side, $\mathbf{h}_t$ is the hidden state of the denoiser f_θ after block ℓ, h_ϕ is an MLP projector that maps it to the feature dimension of the target, and $\beta > 0$ scales the standard deviation of both distributions, so that the two share their covariance.

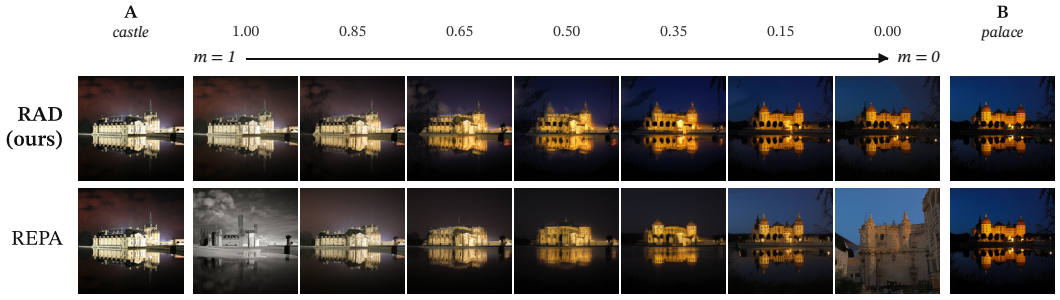
REPA as a special case. We parameterize the denoiser such to ignore $\mathbf{u}_t$, $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t) = p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t)$, so that the denoising term of Eq. 4 reduces to the velocity-matching objective of



(a) Iguana → Komodo dragon



(b) Green chameleon → African chameleon



(c) Castle → palace

Figure 7: Additional interpolations across classes. Samples interpolated from image A ($m = 1$) to image B ($m = 0$), with the real images shown in the outer columns. Top rows: RAD-XL/2 trained for 400K iterations, interpolating the representation, $\mathbf{x}_1$, and the class embedding (Eq. 28). Bottom rows: REPA-XL/2 trained for 4M iterations, i.e., $10\times$ longer than RAD, interpolating $\mathbf{x}_1$ and the class embedding. No guidance.

Eq. 10. Second, parameterize both distributions of the alignment term as von Mises–Fisher (vMF) distributions on the unit sphere, token by token, with a shared concentration $\kappa > 0$,

$$q\left(\mathbf{u}_t^{[n]} \mid \mathbf{x}_0\right) = \mathrm{vMF}\left(\bar{\mathbf{y}}^{[n]}, \kappa\right), \qquad p_\phi\left(\mathbf{u}_t^{[n]} \mid \mathbf{x}_t\right) = \mathrm{vMF}\left(\bar{h}_\phi(\mathbf{h}_t)^{[n]}, \kappa\right), \tag{31}$$

where $\mathbf{y} = f_{\mathrm{pre}}(\mathbf{x}_0)$ and the bar denotes ℓ_2 normalization. Since $\mathrm{vMF}(\boldsymbol{\mu}, \kappa) \propto \exp(\kappa\, \boldsymbol{\mu}^\top \mathbf{u})$, the KL divergence between two vMF distributions with the same concentration is

$$D_{\mathrm{KL}}\left(\mathrm{vMF}(\boldsymbol{\mu}_1, \kappa) \,\|\, \mathrm{vMF}(\boldsymbol{\mu}_2, \kappa)\right) = \kappa\left(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\right)^\top \mathbb{E}_{\mathrm{vMF}(\boldsymbol{\mu}_1, \kappa)}[\mathbf{u}] = \kappa A_D(\kappa)\left(1 - \boldsymbol{\mu}_1^\top \boldsymbol{\mu}_2\right), \tag{32}$$

where $A_D(\kappa) = I_{D/2}(\kappa)/I_{D/2-1}(\kappa)$ is the mean resultant length and I_ν the modified Bessel function of the first kind. Summing over tokens, the alignment term of Eq. 4 becomes $\kappa A_D(\kappa) \sum_n \left(1 - \cos(\mathbf{y}^{[n]}, h_\phi(\mathbf{h}_t)^{[n]})\right)$, which is REPA's cosine alignment loss of Eq. 11 up to an additive constant, with $\lambda = N\kappa A_D(\kappa)$. The bound therefore reduces to the REPA objective and REPA becomes a special case in our framework.



Figure 8: Additional interpolation within the same class. Samples interpolated between two wood rabbits, from image A ($m = 1$) to image B ($m = 0$), with the real images shown in the outer columns. Top: RAD-XL/2 trained for 400K iterations, interpolating the representation and $\mathbf{x}_1$. Bottom: REPA-XL/2 trained for 4M iterations, i.e., $10\times$ longer than RAD, interpolating $\mathbf{x}_1$ only. The class embedding is fixed in both. No guidance.

A.8.2 Internal Alignment

In this setting, the representations are produced by the model itself, as in SRA. The target is the hidden state of an exponential moving average (EMA) copy $f_{\theta'}$ of the denoiser, evaluated at a less noisy state $\mathbf{x}_{t-k}$, $k \in \{1, \dots, t\}$, of the same trajectory. Since the target depends on $\mathbf{x}_{t-k}$ alone, the inference distribution reduces to $q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{\leq t}) = q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{t-k})$, which we parameterize as

$$q_{\theta'}(\mathbf{u}_t \mid \mathbf{x}_{t-k}) = \mathcal{N}\left(\alpha_t f_{\theta'}(\mathbf{x}_{t-k}),\ \beta^2 \sigma_t^2 \boldsymbol{I}\right). \tag{33}$$

The parameters θ' track θ through the EMA update and receive no gradient. The generative distribution $p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)$ is unchanged from Eq. 30: the model at the noisier state $\mathbf{x}_t$ predicts the features that its EMA copy extracts from the cleaner state $\mathbf{x}_{t-k}$. For $k = t$ the target is computed from the clean image, as in the external setting, with $f_{\theta'}$ in place of f_{pre}.

A.8.3 Generative Transition and Forward Process

The remaining distributions are common to both settings. The generative transition is a Gaussian with fixed variance,

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{u}_t) = \mathcal{N}\left(\boldsymbol{\mu}_\theta(\mathbf{x}_t, \mathbf{u}_t, t),\ \tilde{\sigma}^2_{t-1|t} \boldsymbol{I}\right), \tag{34}$$

where $\tilde{\sigma}^2_{t-1|t}$ is the variance of the forward posterior $q(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0)$. It differs from the standard diffusion transition only in that the mean depends on $\mathbf{u}_t$. We parameterize $\boldsymbol{\mu}_\theta$ through a velocity network $\boldsymbol{v}_\theta(\mathbf{x}_t, \mathbf{u}_t, t)$ by substituting the estimate of $\mathbf{x}_0$ implied by $\boldsymbol{v}_\theta$ into the mean of the forward posterior.

The forward process is the fixed Gaussian process of App. A.1.1, with marginals $q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}\left(\alpha_t \mathbf{x}_0, \sigma_t^2 \boldsymbol{I}\right)$. The pair $(\mathbf{x}_{t-k}, \mathbf{x}_t)$ required in the internal setting is sampled as $\mathbf{x}_{t-k} \sim q(\mathbf{x}_{t-k} \mid \mathbf{x}_0)$ followed by

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-k}) = \mathcal{N}\left(\alpha_{t|t-k}\, \mathbf{x}_{t-k},\ \sigma^2_{t|t-k} \boldsymbol{I}\right), \qquad \alpha_{t|t-k} = \frac{\alpha_t}{\alpha_{t-k}}, \quad \sigma^2_{t|t-k} = \sigma_t^2 - \alpha^2_{t|t-k}\, \sigma^2_{t-k}, \tag{35}$$

which recovers the transition $q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$ of Eq. 3 for $k = 1$ and is consistent with the marginals above.

Training objective. Both settings share the inference distribution $q(\mathbf{u}_t \mid \cdot) = \mathcal{N}\left(\alpha_t \mathbf{r}_t, \beta^2 \sigma_t^2 \boldsymbol{I}\right)$ and differ only in the target representation,

$$\mathbf{r}_t = \begin{cases} f_{\text{pre}}(\mathbf{x}_0) & \text{external alignment,} \\ f_{\theta'}(\mathbf{x}_{t-k}) & \text{internal alignment.} \end{cases} \tag{36}$$

Since the inference and generative distributions of $\mathbf{u}_t$ are isotropic Gaussians with the same covariance, the KL term of Eq. 4 is available in closed form,

$$D_{\text{KL}}\left(q(\mathbf{u}_t \mid \cdot) \,\|\, p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)\right) = \frac{\text{SNR}(t)}{2\beta^2} \left\|\mathbf{r}_t - h_\phi(\mathbf{h}_t)\right\|_2^2, \qquad \text{SNR}(t) = \frac{\alpha_t^2}{\sigma_t^2}, \tag{37}$$

where $h_\phi(\mathbf{h}_t) = \boldsymbol{h}_\phi(\mathbf{x}_t, t)$ is the projection of the hidden state. The training objective is therefore Eq. 9 with $\mathbf{r}_t$ in place of $f_{\text{pre}}(\mathbf{x}_0)$ and, in the internal setting, an additional expectation over k (App. A.3.2 gives the derivation). No gradient is propagated through $\mathbf{r}_t$, and neither f_{pre} nor $f_{\theta'}$ is required at sampling time: each step draws $\mathbf{u}_t$ from the model's own policy $p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)$.

A.9 Practical Design Choices

Choice of policy. We refer to the generative distribution $p_\phi(\mathbf{u}_t \mid \mathbf{x}_t)$ as the policy. Under the Gaussian parameterization of Eq. 6, a sample takes the form $\mathbf{u}_t = \alpha_t h_\phi(\mathbf{h}_t) + \beta\sigma_t \boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, which is unimodal with a single scale per time step. To obtain a more expressive policy, we replace the noise with the output of a conditional normalizing flow (Dinh et al., 2017),

$$\mathbf{u}_t = \alpha_t h_\phi(\mathbf{h}_t) + \beta\sigma_t\, \boldsymbol{g}_\zeta(\mathbf{h}_t, \boldsymbol{\epsilon}), \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}), \tag{38}$$

where $\boldsymbol{g}_\zeta$ is a stack of affine coupling layers whose conditioning is a linear map of $h_\phi(\mathbf{h}_t)$, and ζ is trained jointly with ϕ. The flow policy changes only the KL term of Eq. 4: the closed form of Eq. 37 no longer applies, but since the density of $\mathbf{u}_t$ is available through the change of variables, the KL reduces, up to the entropy of the inference distribution (a constant), to the negative log-likelihood of the whitened residual under the flow,

$$\ell_{\text{flow}}(\phi, \zeta) = \mathbb{E}_{\mathbf{u}_t \sim q}\Big[\tfrac{1}{2}\big\|\boldsymbol{g}_\zeta^{-1}(\mathbf{h}_t, \mathbf{z}_t)\big\|_2^2 - \log\Big|\det \tfrac{\partial \boldsymbol{g}_\zeta^{-1}}{\partial \mathbf{z}_t}\Big|\Big], \qquad \mathbf{z}_t = \frac{\mathbf{u}_t - \alpha_t h_\phi(\mathbf{h}_t)}{\beta\sigma_t}, \tag{39}$$

where we have dropped terms independent of (ϕ, ζ). For the identity flow the log-determinant vanishes and the first term equals the alignment term of Eq. 37 in expectation over $\mathbf{u}_t$, so the Gaussian objective of Eq. 9 is the special case $\boldsymbol{g}_\zeta = \mathrm{id}$; this is also the initialization, since the last layer of every coupling block is zero-initialized. In practice, we keep the explicit alignment term and combine the two with complementary time weights,

$$\mathcal{L}(\theta, \phi, \zeta) = \mathbb{E}\Big[\big\|\boldsymbol{v}_\theta(\mathbf{x}_t, \mathbf{u}_t, t) - (\dot\alpha_t \mathbf{x}_0 + \dot\sigma_t \boldsymbol{\epsilon})\big\|_2^2 + \lambda_t \big\|\mathbf{r}_t - h_\phi(\mathbf{h}_t)\big\|_2^2 + (1 - \lambda_t)\, \ell_{\text{flow}}(\phi, \zeta)\Big],$$
$$\lambda_t = \frac{\alpha_t^2}{\alpha_t^2 + \beta^2\sigma_t^2}, \tag{40}$$

where $\mathbf{u}_t \sim q(\mathbf{u}_t \mid \cdot)$ is drawn from the inference distribution as before, the outer expectation is over t, $\mathbf{x}_0$ and the noise, and $h_\phi(\mathbf{h}_t)$ is not detached in ℓ_{flow}, so both terms train the projector. Here λ_t is the saturated alignment weight of App. A.3.2 (scaled by 0.5 in practice), and $1 - \lambda_t$ is the policy's share of the variance of $\mathbf{u}_t$: it vanishes at $t = 0$, where the policy is deterministic and the likelihood degenerates, and approaches one as $t \to 1$. Empirically, the flow policy consistently improves sample quality over the Gaussian policy, and retaining the explicit alignment term is necessary for the projector to train (Sec. 3). For both policies, the scale β need not coincide between training and sampling: we train with $\beta = 1$ and sample with $\beta_{\text{inf}} \le 1$, which concentrates the policy around its mean and trades diversity for fidelity without retraining.

Architecture. We build on SiT. Blocks $1, \dots, \ell$ compute the hidden state $\mathbf{h}_t$, from which the policy draws $\mathbf{u}_t \in \mathbb{R}^{N \times D}$; blocks $\ell + 1, \dots, L$ then output $\boldsymbol{v}_\theta(\mathbf{x}_t, \mathbf{u}_t, t)$. Before it re-enters the backbone, $\mathbf{u}_t$ passes through three steps. (i) *Normalization*: $\mathbf{u}_t^n = (\mathbf{u}_t - \gamma\bar u_t)/(\bar\sigma_t + \delta)$, where $\bar u_t$ and $\bar\sigma_t$ are the mean and standard deviation of $\mathbf{u}_t$ across its N tokens, $\gamma \in [0, 1]$ controls how much of the mean is removed, and δ is a small constant. This keeps the injected signal at unit scale at every noise level and makes the model robust to the choice of β_{inf}. (ii) *Compression*: an MLP along the token axis maps the N tokens of $\mathbf{u}_t^n$ to M tokens $\mathbf{u}_t^c$; we use $M = 1$. (iii) *Modulation*: $\mathbf{u}_t^f = (1 + \boldsymbol{a}(\boldsymbol{c})) \odot \mathbf{u}_t^c + \boldsymbol{b}(\boldsymbol{c})$, where $\boldsymbol{a}$ and $\boldsymbol{b}$ are linear functions of the conditioning embedding $\boldsymbol{c}$, as in adaLN, and are zero-initialized so that training starts from the unmodulated representation. The result is added to every token of the hidden state, $\bar{\mathbf{h}}_t^{[n]} = \mathbf{h}_t^{[n]} + \mathbf{u}_t^f$. We inject early, with ℓ within the first third of the backbone, which we find to be the most influential architectural choice (Sec. 3).

A.10 Additional Samples (Class Conditional ImageNet)



Figure 9: Class-conditional samples from ImageNet (Macaw).



Figure 10: Class-conditional samples from ImageNet (Lorikeet).



Figure 11: Class-conditional samples from ImageNet (Red panda).



Figure 12: Class-conditional samples from ImageNet (Giant panda).



Figure 13: Class-conditional samples from ImageNet (Tiger).



Figure 14: Class-conditional samples from ImageNet (Anemone fish).



Figure 15: Class-conditional samples from ImageNet (Agaric).



Figure 16: Class-conditional samples from ImageNet (Coral fungus).



Figure 17: Class-conditional samples from ImageNet (Strawberry).



Figure 18: Class-conditional samples from ImageNet (Teapot).



Figure 19: Class-conditional samples from ImageNet (Schooner).



Figure 20: Class-conditional samples from ImageNet (Church).

A.11 Additional Text-to-Image Samples



Figure 21: "A fox wearing a knitted scarf, reading a book by a cozy fireplace, warm storybook illustration."



Figure 22: "A cat astronaut floating inside a spaceship, looking at the Earth through a round window."



Figure 23: "A tiny island with a lighthouse inside a teacup, ocean waves swirling, whimsical 3D render."



Figure 24: “A golden retriever wearing a yellow raincoat, splashing in puddles on a rainy street.”



Figure 25: “A miniature village inside a glass terrarium with moss, tiny houses and a winding path, macro photo.”



Figure 26: “A stained glass window depicting a phoenix rising from flames, vivid red and gold.”



Figure 27: "A floating castle above the clouds at sunset, waterfalls pouring off its edges, fantasy art."



Figure 28: "An isometric 3D render of a cozy bedroom with a bed, plants and fairy lights, pastel colors."



Figure 29: "A small cactus wearing a straw hat and neon sunglasses in the Sahara desert."