D-OPSD: On-Policy Self-Distillation for Continuously Tuning Step-Distilled Diffusion Models

Dengyang Jiang[1,2] **Xin Jin**[2] **Dongyang Liu**[4,2] **Zanyi Wang**[3]
Mingzhe Zheng[1] **Ruoyi Du**[2] **Xiangpeng Yang**[2] **Qilong Wu**[2]
Zhen Li[2] **Peng Gao**[2✉] **Harry Yang**[1✉] **Steven C.H. Hoi**[2]
[1]HKUST [2]Alibaba Group [3]UCSD [4]CUHK
https://vvvvvjdy.github.io/d-opsd

Abstract

The landscape of high-performance image generation models is currently shifting from the inefficient multi-step ones to the efficient few-step counterparts (e.g, Z-Image-Turbo and FLUX.2-klein). However, these models present significant challenges for direct continuous supervised fine-tuning. For example, applying the commonly used fine-tuning technique would compromise their inherent few-step inference capability. To address this, we propose **D-OPSD**, a novel training paradigm for step-distilled diffusion models that enables on-policy learning during supervised fine-tuning. We first find that the modern diffusion models, where the LLM/VLM serves as the encoder, can inherit its encoder's in-context capabilities. This enables us to formulate the training as an on-policy self-distillation process. Specifically, during training, we make the model act as both the teacher and the student with different contexts, where the student is conditioned only on the text feature, while the teacher is conditioned on the multimodal feature of both the text prompt and the target image. Training minimizes the two predicted distributions over the student's own roll-outs. By optimizing on the model's own trajectory and under its own supervision, D-OPSD enables the model to learn new concepts, styles, etc., without sacrificing the original few-step capacity.

1 Introduction

Recent years have seen significant progress in text-to-image (T2I) generation, with models advancing from synthesizing rudimentary textures to producing images that exhibit strong adherence to semantic descriptions [87, 3, 4, 16, 95, 6, 78, 22, 60]. However, the sampling process typically requires numerous iterative denoising steps [82, 30, 45], leading to substantial latency and computational cost in practice. To address this, researchers have developed various step-distillation techniques [53, 106, 55, 105, 9] that substantially reduce the number of function evaluations (NFEs). Furthermore, recent advances in distillation methodology [46, 36, 105, 12, 56] have enabled state-of-the-art open-source few-step diffusion models to surpass their multi-step predecessors not only in sampling efficiency but also in generated image quality. As a result, such few-step models are increasingly adopted in practical production settings.

Despite these advances, how to continually finetune these step-distilled diffusion models remains unclear. A straightforward solution is to apply the standard supervised fine-tuning objective used in the multi-step counterpart [45, 49], i.e., feeding a noised target image into the model and supervising it with the corresponding flow-matching target.[2] However, this training signal is defined on externally induced states of the target image that belong to an offline data distribution, rather than

✉Corresponding authors
[2]In this paper, we mainly discuss flow-matching models, as they are currently the default choice in the field.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



Figure 1: We empirically investigate the visual appearance of generated images when conditioned on only the text prompt feature or the multimodal feature of target image and the text prompt using Z-Image Turbo [87] with 8 steps. Investigation of FLUX.2-klein [4] is provided in Appendix A.

on the states actually visited by the model's own few-step sampler. For step-distilled models, whose generation quality relies on a small number of carefully distilled denoising updates, such a mismatch can easily perturb the learned few-step dynamics and degrade inference quality. This effect is also borne out empirically: across our experiments, and echoed by community reports, standard SFT often compromises the model's original distilled few-step ability to generate high-quality images. Online Reinforcement learning (RL), in contrast, would not impair the few-step capabilities when used as the training algorithm for the model [36, 54]. This is because it optimizes the model on samples generated by the current model and derives the learning signal on the same sampled trajectory. However, it requires a well-designed reward function [99, 48, 100, 111, 97, 27, 39], which is not feasible for most secondary developers in the community, as they usually have only the image-text pairs to customize concepts, styles, etc.

Thus, we assume that a suitable continuous-tuning strategy should be a combination of the two: it should update the model on its own roll-outs, and it should incorporate supervision from paired image-text data on those same visited states. A natural candidate is on-policy self-distillation (OPSD), which has recently been studied in autoregressive large language models (LLMs) [79, 102, 110, 26]. OPSD retains the appeal of on-policy learning while avoiding explicit reward design: the model samples from its current policy as a student, while a stronger teacher distribution is obtained by conditioning the same model on richer in-context information [94, 5]. This perspective is particularly appealing in our setting because the target image in each training pair naturally provides the supervision. However, directly transferring the idea of OPSD to diffusion models is nontrivial. In text generation with LLMs, the context can simply be appended to the input sequence. In diffusion models, by contrast, directly feeding the target image into the denoising process would alter the trajectory itself, reducing the formulation back to the standard off-policy SFT regime. The key challenge is therefore: *how can target-image information be introduced as stronger context while keeping the student's few-step roll-out unchanged?*

We address this challenge by proposing **D-OPSD**. Unlike earlier T2I diffusion models that used T5 [69] or CLIP [66] as the encoder [62, 3, 16], current state-of-the-art diffusion models increasingly adopt LLM/VLM backbones [93, 101] as their encoders [4, 87, 95]. This raises a natural question: can the subsequent diffusion model inherit the encoder's in-context capability? As shown in Figure 1, we find that the answer is yes. When replacing text-only features with multimodal features extracted from both the text prompt and the target image, the diffusion model can already generate variations that preserve the target concept or style (see gen w/text+img), even without any additional training. This emergent behavior enables us to instantiate OPSD in diffusion models. Specifically, during training, we assign the same model two roles: a student conditioned only on the text feature, and a teacher conditioned on the multimodal feature of the text prompt and the target image. We then distill the teacher's predictions into the student along the student's own roll-outs, yielding a one-stage on-policy framework that injects target-image information without requiring external modules or reward design.

We evaluate D-OPSD in the settings of both LoRA training on small customized dataset and full fine-tuning on larger dataset, the results show that our method enables the model to acquire new knowledge (e.g, specific concept, style) from the target image-text pair while preserving its original few-step inference capability. Furthermore, rather than learning via overfitting to the training pair, the acquired knowledge with our method demonstrates strong generalization across unseen prompts (e.g, generating training concepts in different scenarios). These results suggest promising prospects for the continual learning of step-distilled diffusion models.

In summary, our main contributions are as follows:

- We identify an emergent property of modern text-to-image diffusion models with LLM/VLM encoders and utilize this property to the continuous tuning of the step-distilled diffusion model.
- We propose D-OPSD, a novel diffusion model on-policy self-distillation framework. By assigning the same model two roles with different contexts, D-OPSD enables supervised tuning on the student's own roll-outs without requiring any external reward function or extra modules.
- We validate D-OPSD in different settings. The results show that our method enables the model to learn new concepts, styles, and domain preferences while preserving its original few-step inference capability and previous knowledge.

2 Method

2.1 Background

In this study, our goal is to continually tune a step-distilled diffusion model on supervised image-text pairs while preserving its original few-step inference capability. As discussed in Section 1, this is difficult for conventional fine-tuning. Vanilla SFT optimizes the model on noised target images rather than on the states visited by its own sampler, and the supervision is provided by an external target velocity that is unavailable at inference time [45, 49]. Such a train test mismatch may make the model acquire new concepts or styles at the cost of distorting the previously distilled few-step generation distribution (distribution shift). Online RL-style methods are more compatible with this setting because they optimize the model on its own roll-outs and derive supervision from the same on-policy samples [48, 100, 99], but they rely on carefully designed reward functions or preference signals [97, 39], which are typically unavailable in practical customization scenarios. We address this gap by constructing an on-policy self-distillation framework for diffusion models, which uses only paired image-text data and does not require any external reward.

2.2 D-OPSD

OPSD in LLMs and our solution for implication in diffusion models. On-policy self-distillation (OPSD) is first proposed in language models with a simple idea: the same model can act as both a student and a teacher under different contexts. Given an input query q, let r denote additional in-context information, such as demonstrations, intermediate reasoning, or the ground-truth response [110, 79, 26, 74]. The student predicts under the weaker context q, while the teacher predicts under the stronger context (q, r). Let $\pi_\theta(\cdot \mid q)$ denote the student distribution and $\pi_{\bar{\theta}}(\cdot \mid q, r)$ the teacher distribution. OPSD optimizes the student on its own sampled outputs $\hat{o} \sim \pi_\theta(\cdot \mid q)$, and minimizes a divergence between the teacher and student predictions on that on-policy sample:

$$\mathcal{L}_{\text{OPSD}}^{\text{LLM}} = \mathbb{E}_{\hat{o}\sim\pi_\theta(\cdot|q)} \left[D(\pi_{\bar{\theta}}(\cdot \mid q, r),\, \pi_\theta(\cdot \mid q)) \right]. \tag{1}$$

This formulation inherits two key properties of on-policy learning: it updates the model on samples produced by the current policy, and the supervision is computed under the same sampled trajectory instead of being borrowed from an external offline distribution.

The challenge in transferring this training paradigm to diffusion models lies in how to construct a stronger context. In LLMs, the extra information r can be natively appended to the input sequence [94, 5]. In diffusion models, however, the desired supervision is an image, and one cannot simply insert the target image into the denoising trajectory in the same way without returning to the standard off-policy SFT setting (e.g, Once the noisy target image is fed directly into the model like traditional training does, the sampling trajectory is disrupted, reducing the process to a supervision paradigm analogous to teacher forcing in large language models [67, 68, 85].). This challenge suggests that the stronger context in diffusion models needs to be introduced through a representation

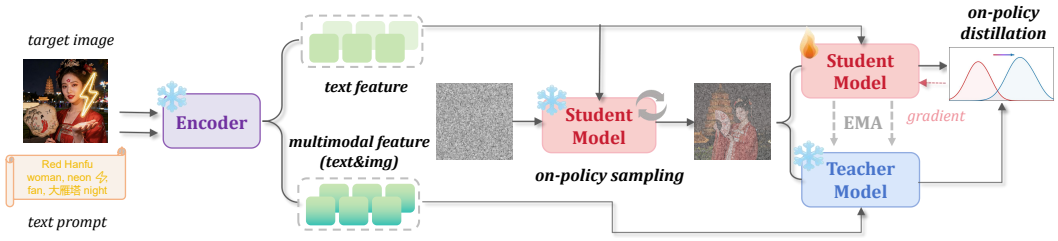


Figure 2: **Method overview.** For each training pair, we first pass the prompt alone and the prompt together with the target image through the encoder to obtain c_s and c_t, respectively. We then sample a few-step trajectory using the student branch conditioned on c_s. After that, the teacher and student predict velocities on the same trajectory states, and the student is updated by Equation 6. After training, the teacher branch is discarded, and inference uses exactly the same few-step text-to-image pipeline as the original step-distilled model.

that enriches the model's conditioning space while leaving the student's roll-outs unchanged. In other words, to make OPSD applicable to diffusion models, we need a mechanism that incorporates target-image information without replacing the student's own sampled states.

We solve this by utilizing the property of modern diffusion models. As we analyse in Section 1 and Figure 1, current SOTA few-step models often adopt LLM/VLM backbones as their encoders and we find that the subsequent diffusion model can inherit the encoder's in-context capability: when conditioned on the multimodal feature extracted from both the text prompt and the target image, the model can already produce variations that preserve the target concept or style, even without additional training. This observation allows us to instantiate OPSD in diffusion models by treating the target image as in-context supervision, rather than as a direct denoising target.

Formulating OPSD for diffusion models. The overall framework and pseudocode of our method D-OPSD can be seen in Figure 2 and Algorithm 1. Specifically, we consider the model parameterized by θ, whose velocity field is denoted by $v_\theta(x_t, t, c)$, where x_t is the latent state at time $t \in [0, 1]$ and c is the condition feature. During inference, the model defines an ODE trajectory:

$$\frac{dx_t}{dt} = v_\theta(x_t, t, c), \tag{2}$$

which is solved with a small number of time steps by the few-step sampler (e.g, 4 or 8). Let $1 = t_K > t_{K-1} > \cdots > t_0 = 0$ denote the inference schedule. Starting from Gaussian noise $x^s_{t_K} \sim \mathcal{N}(0, I)$, the student roll-outs is generated by:

$$x^s_{t_{k-1}} = \Phi\big(x^s_{t_k}, t_k, t_{k-1}, v_\theta(\cdot, \cdot, c_s)\big), \quad k = K, \ldots, 1, \tag{3}$$

where Φ denotes the same few-step solver used at test time, and c_s is the student condition.

For each training pair (x_0, y), we construct two conditions from the same encoder:

$$c_s = f_{\text{text}}(y), \qquad c_t = f_{\text{mm}}(y, x_0), \tag{4}$$

where f_{text} encodes only the text prompt and f_{mm} encodes the multimodal context consisting of the text prompt and the target image. The student is conditioned only on c_s, so its inference pathway is exactly the original text-to-image generation process. The teacher is conditioned on c_t, which provides additional information about the target concept, style, or preference to be learned.

Within f_{mm}, the ViT and Connector convert the target image into visual tokens that are processed jointly with text tokens. The resulting hidden states are passed through the diffusion backbone's original conditioning interface. This construction requires no additional encoder training; its implementation and quantitative validation are provided in Appendix D.

Given the on-policy trajectory $\{x^s_{t_k}\}_{k=1}^K$, we evaluate both branches on the same visited states:

$$u^s_k = v_\theta(x^s_{t_k}, t_k, c_s), \quad u^t_k = v_{\bar{\theta}}(x^s_{t_k}, t_k, c_t), \tag{5}$$

where $\bar{\theta}$ denotes the teacher parameters. We then train the student to match the teacher's velocity prediction by minimizing:

$$\mathcal{L}_{\text{D-OPSD}} = \mathbb{E}_{(x_0,y)}\left[\frac{1}{K}\sum_{k=1}^{K}\left\|u^s_k - \text{sg}(u^t_k)\right\|_2^2\right], \tag{6}$$

where $\mathrm{sg}(\cdot)$ denotes stop-gradient operation. In this way, the student is optimized on its own roll-outs states, while the teacher provides a stronger supervision signal through multimodal context.

Note that Equation 6 can be viewed as the diffusion counterpart of Equation 1. At a high level, the analogy is straightforward: the student's sampled response in LLMs corresponds here to the student's denoising trajectory, and the teacher's stronger prediction under a richer context is realized as a stronger conditional denoising field. The main difference lies in the output space of the model. Autoregressive LLMs produce a discrete token distribution [67, 88], so the teacher-student alignment can be written directly as a divergence between vocabulary distributions [110, 79]. Flow-matching diffusion models, by contrast, do not expose such a discrete predictive distribution at each step. Instead, they parameterize the denoising dynamics through a conditional velocity field, whose predictions determine the evolution of the sample trajectory [45, 49, 84]. For this reason, we instantiate the teacher-student alignment in Eq. 6 as a mean-squared error between velocity predictions on the same on-policy states. Although this objective is not a token-level KL divergence [41], it serves the same role in our setting: it pulls the student's conditional generation dynamics toward those of the teacher, thereby aligning the induced trajectory distribution under a stronger multimodal context. The underlying principle therefore remains unchanged: the model learns from its own trajectory under a stronger self-generated supervision signal.

Discussion on why D-OPSD preserves few-step capability. Compared with vanilla SFT, our method does not force the model to fit states induced by the target image that never appear under its own few-step sample trajectory. Instead, both the optimization states and the supervision signal are defined on the student's actual roll-outs, which substantially reduces the mismatch between training and inference. This distinction is particularly important for step-distilled diffusion models, where even small perturbations to the learned few-step dynamics can directly harm generation quality. As a result, D-OPSD provides an on-policy supervised training paradigm for step-distilled diffusion models, enabling them to learn new concepts, styles, or domain preferences from paired image-text data while preserving the original few-step sampling behavior. More discussion and comparison of different training paradigms are provided in Appendix B.

Algorithm 1: Training Procedure of D-OPSD

Require: Training pairs $\mathcal{D} = \{(x_0, y)\}$
Require: Inference schedule $\{t_0, \ldots, t_K\}$ and Solver Φ
Require: Base model v_ϕ, Student and Teacher model v_θ, $v_{\bar{\theta}}$
Initialize student weights $\theta \leftarrow \phi$
Initialize teacher weights $\bar{\theta} \leftarrow \phi$
while not converged **do**
  Sample a mini-batch $(x_0, y) \sim \mathcal{D}$
  Encode student condition $c_s \leftarrow f_{\text{text}}(y)$
  Encode teacher condition $c_t \leftarrow f_{\text{mm}}(y, x_0)$
  Initialize $x^s_{t_K} \sim \mathcal{N}(0, I)$
  Initialize $\mathcal{L}_{\text{D-OPSD}} \leftarrow 0$
  for $k = K, \ldots, 1$ **do**
    $u^s_k \leftarrow v_\theta(x^s_{t_k}, t_k, c_s)$
    $u^t_k \leftarrow v_{\bar{\theta}}(x^s_{t_k}, t_k, c_t)$
    $\mathcal{L}_{\text{D-OPSD}} \leftarrow \mathcal{L}_{\text{D-OPSD}} + \frac{1}{K}\|u^s_k - \mathrm{sg}(u^t_k)\|_2^2$
    if $k > 1$ **then**
      $x^s_{t_{k-1}} \leftarrow \mathrm{sg}(\Phi(x^s_{t_k}, t_k, t_{k-1}, u^s_k))$
    end if
  end for
  Update student model θ by minimizing $\mathcal{L}_{\text{D-OPSD}}$
  Update teacher model $\bar{\theta}$ via EMA
end while

3 Experiment

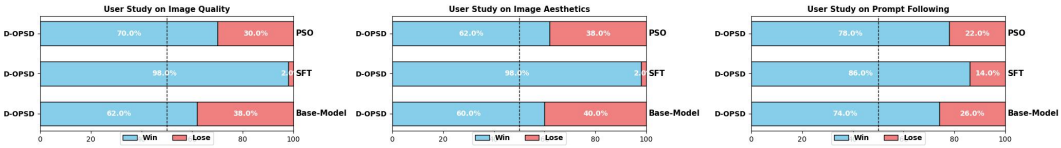
3.1 Experimental Setup

Implementation. We use Z-Image-Turbo 6B [87] and FLUX.2-klein 4B [4] as our baseline model to conduct experiment. Detailed experimental implementation, including hyperparameter settings, GPU resources, and other training configs, is provided in Appendix D.

Evaluation. We use the same inference settings as the original step-distilled model across all methods. We choose to report DINO distance (DINO-D) [7], LPIPS distance (LPIPS-D) [109], Fréchet Inception Distance (FID [28]) for testing whether the model can learn from the target images, VLM's judgment of subject or style consistency (VLM-J), CLIP Score (CLIP-S) [27] for testing whether the model can generalize with the learned new knowledge, the Quality Score (Quality-S) and Aes-

Table 1: **System-level comparison against baseline methods in LoRA training settings.** The **best** and second-best results on each metric are highlighted in bold and underlined.

Method	DINO-D↓	LPIPS-D↓	VLM-J↑	CLIP-S↑	Quality-S↑	Aesthetic-S↑
Z-Image-Turbo						
Base-Model	0.2373	0.7520	1.0000	0.3043	3.5088	2.9667
Vanilla SFT	0.2212	0.6501	1.3293	0.3095	2.4236	2.3582
SFT + LoRA on distilled	0.1588	0.7243	1.8525	0.3201	3.6081	3.0027
Dreambooth	0.0902	0.6424	3.0625	0.3328	2.5582	2.3755
PSO	**0.0570**	**0.4974**	**3.3333**	0.2893	3.3422	3.0820
D-OPSD (ours)	0.0823	0.5803	**3.3333**	**0.3664**	**3.7965**	**3.1710**
FLUX.2-klein						
Base-Model	0.2386	0.7518	1.0000	0.3020	3.5076	2.9665
Vanilla SFT	0.2212	0.6500	1.2083	0.3002	2.4119	2.3595
SFT + LoRA on distilled	0.1564	0.7288	1.0000	0.3151	3.6079	3.0021
Dreambooth	0.1053	0.6824	3.0283	0.3266	2.5492	2.3683
PSO	**0.0593**	**0.5120**	**3.3015**	0.2801	3.3258	3.0566
D-OPSD (ours)	0.0822	0.5648	3.1255	**0.3689**	**3.7957**	**3.1521**



(a) User Study on Image Quality (b) User Study on Image Aesthetics (c) User Study on Prompt Following

Figure 3: **User study comparing with baselines.** All images are generated using identical noise. Our method produces images with superior aesthetics and prompt coherence.

thetic Score (Aesthetic-S) from the reward model for testing whether the model maintain its few-step sampling capacity, as well as GenEval [20] and DPG [32] score to test whether the model retain its previous knowledge. We also provide a user study for evaluation. Detailed explanation of how the evaluation set is constructed and how each metric is obtained are in Appendix E.

Methods for comparison. We compare with several representative baseline methods: (a). directly training with vanilla flow-matching loss [45] (Vanilla SFT). (b). training on the original multi-step model then adding LoRA on the step-distilled model (SFT + LoRA on distilled). (c). Dreambooth style training [73] (Dreambooth). (d). PSO training [59] (PSO). Note that we do not show the results of two-stage training strategies, such as first performing supervised fine-tuning on a multi-step model and then applying step distillation. One practical reason is that, the exact distillation recipe, including key hyperparameter configurations, is not publicly available for the base model we use. Reproducing such pipelines would therefore require substantial implementation assumptions and could introduce unfairness into the comparison. However, we still provide a comparison with a two-stage variant that uses the open-source DMD method in the Appendix C for reference.

3.2 Main Results

D-OPSD for LoRA training on small customized dataset. We first evaluate D-OPSD in the setting of LoRA training on small customized datasets. In this setting, the goal is to learn a new concept from only a few image–text pairs (e.g., 4 examples) while still being able to generalize beyond the training set. We conduct training and evaluation on the DreamBooth dataset [73] together with a small amount of stylized data. As shown in Table 1, our method substantially outperforms both the base model and SFT style training on DINO-D, LPIPS-D, and VLM-J. Moreover, as illustrated in Figure 4, after training, our method can generalize the newly learned concept beyond the training distribution, e.g., generating the learned object in novel scene that do not appear in the training data, while preserving the original model's ability to produce high-quality images with a small number of inference steps. In contrast, other baselines such as SFT and DreamBooth training lose the ability to generate high-quality images under the few-step inference setting, as reflected by the large drops in Quality-S and Aesthetic-S in Table 1, as well as the blurry images shown in the Figure 4. PSO, on the other hand, tends to overfit the training set: although it can capture the target concept, its

Table 2: **System-level comparison against baseline methods in full-finetuning settings.** The **best** and second-best results on each metric are highlighted in bold and underlined.

Method	FID↓	DINO-D↓	LPIPS-D↓	Quality-S↑	Aesthetic-S↑	GenEval↑	DPG↑
Z-Image-Turbo							
Base-Model	48.6858	0.1274	0.7036	3.7624	3.5218	**0.7543**	**84.7645**
Vanilla SFT	82.2036	0.1896	0.6787	2.6121	2.4852	0.1588	69.9746
PSO	88.4343	0.1716	0.6530	2.8653	2.6424	0.2475	72.7363
D-OPSD (ours)	**40.4938**	**0.1088**	**0.6419**	**3.8438**	**3.6195**	0.7170	84.1116
FLUX.2-klein							
Base Model	45.2335	0.1210	0.6894	3.7626	3.4465	**0.8155**	**85.5624**
Vanilla SFT	98.7893	0.2565	0.6567	2.5038	2.4121	0.1113	56.2138
PSO	82.1438	0.2185	0.6511	2.8066	2.5952	0.2406	63.4174
D-OPSD (ours)	**38.2932**	**0.0902**	**0.6476**	**3.8328**	**3.5511**	0.7298	83.8927

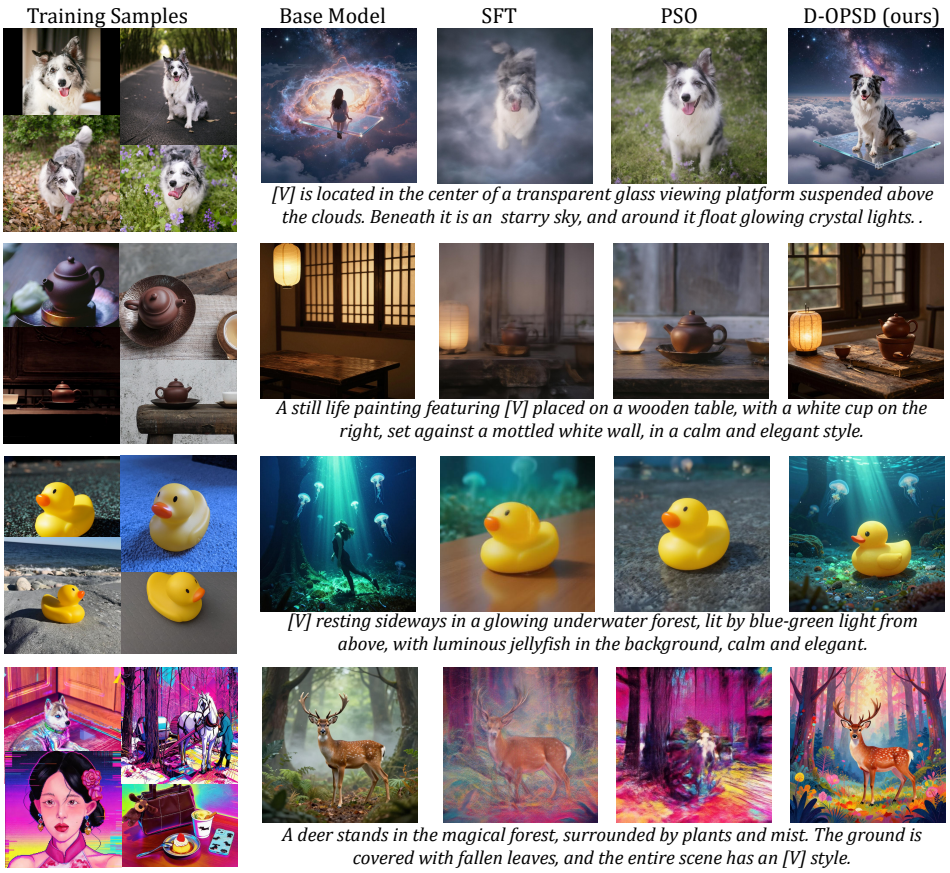


Figure 4: **Visual comparison between baseline methods and ours finetuned on Z-Image-Turbo under customized training settings.** Vanilla SFT training sacrifices the original few-step capacity, and PSO suffers from the overfitting to training set, whereas our method enables the step-distilled model to continuously learn new concepts while maintaining the few-step capacity.

ability to follow novel instructions degrades substantially, as indicated by the decline in CLIP-S and its failure to generate scenes beyond those in the training data.

D-OPSD for full finetuning on larger scale dataset. We next evaluate D-OPSD in the setting of full finetuning on larger scale dataset. In this setting, the goal is to test whether by fine-tuning, the model can be biased towards a certain preference or domain (in our experiment, it is "anime" domain), and suffer from catastrophic forgetting of previously learned knowledge. We conduct training and evaluation on the high-quality anime dataset. As shown in Table 2, our method substantially outperforms both the base model and other training methods on FID, DINO-D, and LPIPS-D, suggesting the output of the model after finetuning is more likely to be closer to the target distribution.

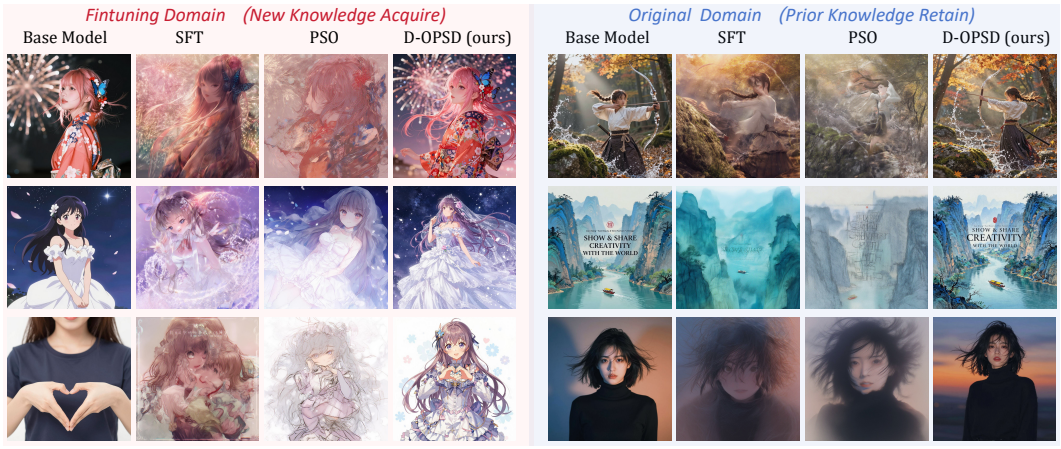


Figure 5: **Visual comparison between baseline methods and ours finetuned on Z-Image-Turbo under full-finetuning settings.** SFT and PSO training sacrifices the original few-step capacity, whereas our method enables the step-distilled model to continuously learn to bias target domain while maintaining the few-step capacity as well as the learned knowledge in the original domain.

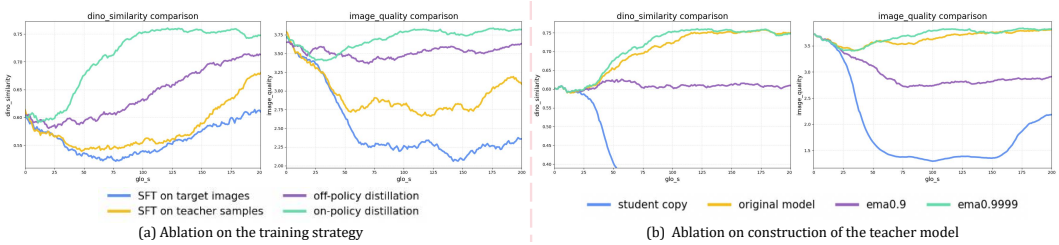
Meanwhile, our method is still able to adapt to the new distribution while retaining the model's original knowledge as well as few-step inference ability. This can be observed from the GenEval and DPG results in the Table 2 and Figure 5, where our method shows no catastrophic degradation after fine-tuning. Although there is a slight drop in benchmark score compared with the base model, we believe this reflects a trade-off introduced by adapting the model to a new distribution whose domain differs from those emphasized by the benchmarks. In contrast, both SFT and PSO fail to simultaneously adapt to the new domain and preserve the model's few-step inference capability in the full-finetuning on large-scale dataset setting. This is evident from the sharp declines across multiple metrics in Table 2, as well as the blurry generated images shown in Figure 5.

We also report user study results in Figure 3. D-OPSD is preferred over PSO, Vanilla SFT, and even the base model across all three aspects: image quality, aesthetics, and prompt following. The preference margin is particularly large against SFT, indicating that our method better preserves the original few-step generation quality during continual tuning. Notably, D-OPSD also largely outperforms the base model in prompt following, suggesting that the learned target-domain knowledge improves performance without sacrificing overall generation quality.

3.3 Ablation Study

Effect of on-policy self-distillation. Our method consists of two key components: on-policy sampling and on-policy distillation. To elucidate the role of each component, we conduct four groups of ablation studies in isolation: (1) SFT from target images, which is identical to the vanilla training setting with flow-matching loss. (2) SFT from teacher samples, where we replace the target images with samples generated by conditioning on multimodal features extracted from the target image and the text prompt, and use these generated samples as the new targets for SFT. (3) Off-policy distillation, where the student model is trained to align with the teacher's outputs on a fixed dataset. (4) On-policy distillation, which corresponds to our proposed method. As shown in the two left plots of Figure 6, vanilla SFT with the flow-matching loss gradually impairs the model's ability to generate high-quality images with few steps, as reflected by the progressive decline in the Quality Score. Self-distillation-based schemes effectively mitigate this issue. Compared with the off-policy distillation variant, our method achieves the fastest training convergence, as evidenced by the highest DINO similarity to the target images, while simultaneously maintaining the best generation quality.

Construction of the teacher model. As our method is a self-distillation framework, we study several way to build the teacher model. First, we find that using the frozen base model as the teacher yields stable training and effective results. We then study the commonly used EMA operations [34], like other self-distillation works [37, 7, 113], we also find that it requires a large momentum coefficient to stabilize training; for example, directly using the student copy leads to training collapse. In our experiment, using the EMA teacher with the momentum coefficient 0.9999 leads to the best



(a) Ablation on the training strategy (b) Ablation on construction of the teacher model

Figure 6: **Ablation on (a) the different training strategies, and (b) the different way to build teacher model**. We report the curves across training steps of DINO feature similarity between the generated images and the targets, as well as the Quality Score of the generated images. Training conducted on Z-Image-Turbo with LoRA.

results, we assume that it is because this can not only extremely smooth the high-variance alignment target to make training stable, but also tracks the student's progress for better distillation.

Teacher–student alignment. We further quantify the supervision provided by the multimodal teacher on the 40 concept/style tasks used in our LoRA evaluation with Z-Image-Turbo. We compare teacher–target alignment before training with student–target alignment after training using the DINO-D and LPIPS-D protocols in Appendix E. As shown in Table 3, the trained student approaches the teacher's target-image alignment, with gaps of 0.0016 in DINO-D and 0.0075 in LPIPS-D. The teacher uses both the prompt and the target image, whereas the student uses only the prompt at inference. These results support effective transfer of target-specific information from the stronger multimodal context to the text-conditioned student.

Table 3: **Teacher–student alignment.** Average target-image distances over 40 concept/style tasks.

Model	DINO-D↓	LPIPS-D↓
Teacher (before training)	0.0807	0.5728
Student (after training)	0.0823	0.5803

4 Discussion on Limitations and Future Works

The success of D-OPSD is contingent upon the base model's in-context learning abilities. Specifically, as shown in Figure 7, even when conditioned on the multimodal feature of the target image and the text prompt, the subsequent diffusion model still can not generate meaningful supervision signals, and the training would fail. However, we successfully introduce on-policy self-distillation into the image generation domain and demonstrate promising results. Building upon our method, several interesting directions for future work, such as how to better construct the teacher model's context (conditioning), for example, integrating image editing models [22, 40, 43] or video generation models [90, 25, 77], or how to incorporate target image information more effectively, such as combining our framework with additional constraints [64, 37], remain to be explored.

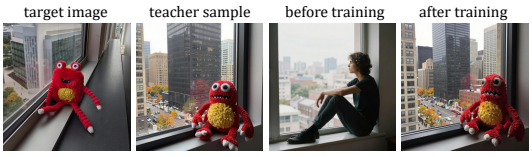


Figure 7: When the teacher model fails to generate images consistent with the concept ID under multimodal condition and therefore cannot provide an effective supervision signal, training will fail.

5 Related Work

We highlight key related studies here and defer discussion of the others to Appendix F.

Step-distilled diffusion models. To accelerate diffusion model inference for enhancing productivity, various timestep distillation methods have been proposed to compress the original model into a generator capable of few-step sampling [106, 53, 76, 55, 52, 57]. This is typically achieved either by distilling the trajectory [53, 71, 112, 9] or from the distribution [106, 105, 36, 12, 75, 76, 51]. Although a significant effort has been made to ensure the quality of the generated contents while

reducing the number of inference steps, continual fine-tuning on these distilled models still faces the challenge of how to preserve their few-step inference capability when learning new things. In this work, we solve this by utilizing the in-context capacity of the diffusion model's encoders to develop an on-policy self-distillation framework that enables the continuous learning of the model under its own supervision, without sacrificing its original few-step inference capability.

On-policy self-distillation. In the field of large language models, on-policy distillation is proposed to mitigate the train-test mismatch issues caused by off-policy SFT or knowledge distillation [1, 83, 50, 35]. However, it still requires a stronger external teacher model for guidance. Therefore, on-policy self-distillation is proposed, which enables the model itself to act as a teacher by leveraging its own in-context capabilities within the pre-existing context (e.g, demonstration, answer) [110, 104, 79, 74, 102, 26]. Unlike these works, which focused on text generation with autoregressive large language models, our approach shows how to utilize on-policy self-distillation in image generation for continuously training step-distilled diffusion models.

6 Conclusion

In this work, we present D-OPSD, an on-policy self-distillation framework for continually tuning step-distilled diffusion models. Our method builds on the observation that modern diffusion models with LLM/VLM encoders exhibit an emergent in-context capability, enabling the same model to serve as a student under text-only conditioning and as a teacher under stronger multimodal conditioning. By distilling the teacher's predictions along the student's own few-step roll-outs, D-OPSD enables supervised adaptation without external rewards or auxiliary training stages. Experiments on both LoRA adaptation and full fine-tuning show that our method learns new concepts, styles, and domain preferences while preserving few-step generation quality and prior knowledge.

Acknowledgments and Disclosure of Funding

This work was supported in part by the Research Grants Council of the Hong Kong Special Administrative Region, China, under the Early Career Scheme (Project No. 26210225). The authors declare no additional competing interests.

We also wish to note that, since the release of Z-Image-Turbo, we have been greatly inspired by the community's extensive explorations into the internal mechanisms of step-distilled models and continuous training strategies. While we cannot enumerate every contribution here, we extend our sincere gratitude to all the talented 'artist' in the community.

References

[1] Agarwal, R., Vieillard, N., Zhou, Y., Stanczyk, P., Garea, S.R., Geist, M., Bachem, O.: On-policy distillation of language models: Learning from self-generated mistakes. In: The twelfth international conference on learning representations (2024)

[2] Bai, S., Cai, Y., Chen, R., Chen, K., Chen, X., Cheng, Z., Deng, L., Ding, W., Gao, C., Ge, C., et al.: Qwen3-vl technical report. arXiv preprint arXiv:2511.21631 (2025)

[3] Black Forest Labs: FLUX. https://github.com/black-forest-labs/flux (2023)

[4] Black Forest Labs: FLUX.2: Frontier Visual Intelligence. https://bfl.ai/blog/flux-2 (2025)

[5] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. Advances in neural information processing systems **33**, 1877–1901 (2020)

[6] Cao, S., Chen, H., Chen, P., Cheng, Y., Cui, Y., Deng, X., Dong, Y., Gong, K., Gu, T., Gu, X., et al.: Hunyuanimage 3.0 technical report. arXiv preprint arXiv:2509.23951 (2025)

[7] Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 9650–9660 (2021)

[8] Chefer, H., Esser, P., Lorenz, D., Podell, D., Raja, V., Tong, V., Torralba, A., Rombach, R.: Self-supervised flow matching for scalable multi-modal synthesis. arXiv preprint arXiv:2603.06507 (2026)

[9] Chen, H., Zhang, K., Tan, H., Guibas, L., Wetzstein, G., Bi, S.: pi-flow: Policy-based few-step generation via imitation distillation. arXiv preprint arXiv:2510.14974 (2025)

[10] Chen, J., Yu, J., Ge, C., Yao, L., Xie, E., Wu, Y., Wang, Z., Kwok, J., Luo, P., Lu, H., Li, Z.: Pixart-α: Fast training of diffusion transformer for photorealistic text-to-image synthesis. In: International Conference on Learning Representations (2024)

[11] Chen, Z., Wang, W., Tian, H., Ye, S., Gao, Z., Cui, E., Tong, W., Hu, K., Luo, J., Ma, Z., et al.: How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites. Science China Information Sciences **67**(12), 220101 (2024)

[12] Cheng, Z., Sun, P., Li, J., Lin, T.: Twinflow: Realizing one-step generation on large models with self-adversarial flows. arXiv preprint arXiv:2512.05150 (2025)

[13] Comanici, G., Bieber, E., Schaekermann, M., Pasupat, I., Sachdeva, N., Dhillon, I., Blistein, M., Ram, O., Zhang, D., Rosen, E., et al.: Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. arXiv preprint arXiv:2507.06261 (2025)

[14] Daniel Verdú, J.M.: Flux.1 lite: Distilling flux1.dev for efficient text-to-image generation. https://huggingface.co/Freepik (2024)

[15] Dhariwal, P., Nichol, A.: Diffusion models beat gans on image synthesis. Advances in neural information processing systems **34**, 8780–8794 (2021)

[16] Esser, P., Kulal, S., Blattmann, A., Entezari, R., Muller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., Podell, D., Dockhorn, T., English, Z., Lacey, K., Goodwin, A., Marek, Y., Rombach, R.: Scaling rectified flow transformers for high-resolution image synthesis. In: Forty-first international conference on machine learning (2024)

[17] Fang, A., Jose, A.M., Jain, A., Schmidt, L., Toshev, A., Shankar, V.: Data filtering networks. arXiv preprint arXiv:2309.17425 (2023)

[18] Fang, G., Li, K., Ma, X., Wang, X.: Tinyfusion: Diffusion transformers learned shallow. arXiv preprint arXiv:2412.01199 (2024)

[19] Gal, R., Alaluf, Y., Atzmon, Y., Patashnik, O., Bermano, A.H., Chechik, G., Cohen-Or, D.: An image is worth one word: Personalizing text-to-image generation using textual inversion. arXiv preprint arXiv:2208.01618 (2022)

[20] Ghosh, D., Hajishirzi, H., Schmidt, L.: Geneval: An object-focused framework for evaluating text-to-image alignment. Advances in Neural Information Processing Systems **36**, 52132–52152 (2023)

[21] Google DeepMind: Gemini 3. https://deepmind.google/models/gemini/ (2025)

[22] Google DeepMind: Gemini 3 pro image model card. https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-Pro-Image-Model-Card.pdf (2025)

[23] Gou, J., Yu, B., Maybank, S.J., Tao, D.: Knowledge distillation: A survey. International Journal of Computer Vision **129**(6), 1789–1819 (2021)

[24] Goyal, S., Agrawal, S., Anil, G.G., Jain, P., Paul, S., Kusupati, A.: Elt: Elastic looped transformers for visual generation. arXiv preprint arXiv:2604.09168 (2026)

[25] HaCohen, Y., Brazowski, B., Chiprut, N., Bitterman, Y., Kvochko, A., Berkowitz, A., Shalem, D., Lifschitz, D., Moshe, D., Porat, E., et al.: Ltx-2: Efficient joint audio-visual foundation model. arXiv preprint arXiv:2601.03233 (2026)

[26] He, Y., Kaur, S., Bhaskar, A., Yang, Y., Liu, J., Ri, N., Fowl, L., Panigrahi, A., Chen, D., Arora, S.: Self-distillation zero: Self-revision turns binary rewards into dense supervision. arXiv preprint arXiv:2604.12002 (2026)

[27] Hessel, J., Holtzman, A., Forbes, M., Bras, R.L., Choi, Y.: Clipscore: A reference-free evaluation metric for image captioning. arXiv preprint arXiv:2104.08718 (2021)

[28] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017)

[29] Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)

[30] Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in neural information processing systems **33**, 6840–6851 (2020)

[31] Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al.: Lora: Low-rank adaptation of large language models. ICLR **1**(2), 3 (2022)

[32] Hu, X., Wang, R., Fang, Y., Fu, B., Cheng, P., Yu, G.: Ella: Equip diffusion models with llm for enhanced semantic alignment. arXiv preprint arXiv:2403.05135 (2024)

[33] Huang, X., Li, Z., He, G., Zhou, M., Shechtman, E.: Self forcing: Bridging the train-test gap in autoregressive video diffusion. arXiv preprint arXiv:2506.08009 (2025)

[34] Hunter, J.S.: The exponentially weighted moving average. Journal of quality technology **18**(4), 203–210 (1986)

[35] Jang, I., Yeom, J., Yeo, J., Lim, H., Kim, T.: Stable on-policy distillation through adaptive target reformulation. arXiv preprint arXiv:2601.07155 (2026)

[36] Jiang, D., Liu, D., Wang, Z., Wu, Q., Li, L., Li, H., Jin, X., Liu, D., Li, Z., Zhang, B., et al.: Distribution matching distillation meets reinforcement learning. arXiv preprint arXiv:2511.13649 (2025)

[37] Jiang, D., Wang, M., Li, L., Zhang, L., Wang, H., Wei, W., Dai, G., Zhang, Y., Wang, J.: No other representation component is needed: Diffusion transformers can provide representation guidance by themselves. arXiv preprint arXiv:2505.02831 (2025)

[38] Karras, T., Aittala, M., Lehtinen, J., Hellsten, J., Aila, T., Laine, S.: Analyzing and improving the training dynamics of diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 24174–24184 (2024)

[39] Kirstain, Y., Polyak, A., Singer, U., Matiana, S., Penna, J., Levy, O.: Pick-a-pic: An open dataset of user preferences for text-to-image generation. Advances in neural information processing systems **36**, 36652–36663 (2023)

[40] Kulikov, V., Kleiner, M., Huberman-Spiegelglas, I., Michaeli, T.: Flowedit: Inversion-free text-based editing using pre-trained flow models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 19721–19730 (2025)

[41] Kullback, S., Leibler, R.A.: On information and sufficiency. The annals of mathematical statistics **22**(1), 79–86 (1951)

[42] Kumari, N., Zhang, B., Zhang, R., Shechtman, E., Zhu, J.Y.: Multi-concept customization of text-to-image diffusion. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1931–1941 (2023)

[43] Labs, B.F., Batifol, S., Blattmann, A., Boesel, F., Consul, S., Diagne, C., Dockhorn, T., English, J., English, Z., Esser, P., et al.: Flux. 1 kontext: Flow matching for in-context image generation and editing in latent space. arXiv preprint arXiv:2506.15742 (2025)

[44] Li, L., Gong, Y., Liu, S., Cheng, B., Ma, Y., Wu, L., Jiang, D., Wang, Z., Leng, D., Yin, Y.: Refvton: person-to-person try on with additional unpaired visual reference. arXiv preprint arXiv:2511.00956 (2025)

[45] Lipman, Y., Chen, R.T., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. arXiv preprint arXiv:2210.02747 (2022)

[46] Liu, D., Gao, P., Liu, D., Du, R., Li, Z., Wu, Q., Jin, X., Cao, S., Zhang, S., Li, H., Hoi, S.: Decoupled dmd: Cfg augmentation as the spear, distribution matching as the shield. arXiv preprint arXiv:2511.22677 (2025)

[47] Liu, H., Li, C., Wu, Q., Lee, Y.J.: Visual instruction tuning. Advances in neural information processing systems **36**, 34892–34916 (2023)

[48] Liu, J., Liu, G., Liang, J., Li, Y., Liu, J., Wang, X., Wan, P., Zhang, D., Ouyang, W.: Flow-grpo: Training flow matching models via online rl. arXiv preprint arXiv:2505.05470 (2025)

[49] Liu, X., Gong, C., Liu, Q.: Flow straight and fast: Learning to generate and transfer data with rectified flow. arXiv preprint arXiv:2209.03003 (2022)

[50] Lu, K., Lab, T.M.: On-policy distillation. Thinking Machines Lab: Connectionism (2025). https://doi.org/10.64434/tml.20251026, https://thinkingmachines.ai/blog/on-policy-distillation

[51] Lu, Y., Ren, Y., Xia, X., Lin, S., Wang, X., Xiao, X., Ma, A.J., Xie, X., Lai, J.H.: Adversarial distribution matching for diffusion distillation towards efficient image and video synthesis. arXiv preprint arXiv:2507.18569 (2025)

[52] Luhman, E., Luhman, T.: Knowledge distillation in iterative generative models for improved sampling speed. arXiv preprint arXiv:2101.02388 (2021)

[53] Luo, S., Tan, Y., Huang, L., Li, J., Zhao, H.: Latent consistency models: Synthesizing high-resolution images with few-step inference. arXiv preprint arXiv:2310.04378 (2023)

[54] Luo, W.: Diff-instruct++: Training one-step text-to-image generator model to align with human preferences. arXiv preprint arXiv:2410.18881 (2024)

[55] Luo, W., Hu, T., Zhang, S., Sun, J., Li, Z., Zhang, Z.: Diff-instruct: A universal approach for transferring knowledge from pre-trained diffusion models. Advances in Neural Information Processing Systems **36**, 76525–76546 (2023)

[56] Luo, Y., Hu, T., Luo, W., Tang, J.: Tdm-r1: Reinforcing few-step diffusion models with non-differentiable reward. arXiv preprint arXiv:2603.07700 (2026)

[57] Luo, Y., Hu, T., Sun, J., Cai, Y., Tang, J.: Learning few-step diffusion models by trajectory distribution matching. arXiv preprint arXiv:2503.06674 (2025)

[58] Ma, N., Goldstein, M., Albergo, M.S., Boffi, N.M., Vanden-Eijnden, E., Xie, S.: Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In: European Conference on Computer Vision. pp. 23–40. Springer (2024)

[59] Miao, Z., Yang, Z., Lin, K., Wang, Z., Liu, Z., Wang, L., Qiu, Q.: Tuning timestep-distilled diffusion model using pairwise sample optimization. arXiv preprint arXiv:2410.03190 (2024)

[60] OpenAI: Gpt-Image-1. https://openai.com/index/introducing-4o-image-generation/ (2025)

[61] Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 4195–4205 (2023)

[62] Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., Rombach, R.: Sdxl: Improving latent diffusion models for high-resolution image synthesis. arXiv preprint arXiv:2307.01952 (2023)

[63] Qin, Q., Zhuo, L., Xin, Y., Du, R., Li, Z., Fu, B., Lu, Y., Li, X., Liu, D., Zhu, X., et al.: Lumina-image 2.0: A unified and efficient image generative framework. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 20031–20042 (2025)

[64] Qin, Y., Wang, L., Fei, H., Zimmermann, R., Bo, L., Lu, Q., Wang, C.: Soar: Self-correction for optimal alignment and refinement in diffusion models. arXiv preprint arXiv:2604.12617 (2026)

[65] Qwen Team: Qwen3.5: Towards native multimodal agents. https://qwen.ai/blog?id=qwen3.5 (2026)

[66] Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021)

[67] Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al.: Improving language understanding by generative pre-training (2018)

[68] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)

[69] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of machine learning research **21**(140), 1–67 (2020)

[70] Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with clip latents. arXiv preprint arXiv:2204.06125 **1**(2), 3 (2022)

[71] Ren, Y., Xia, X., Lu, Y., Zhang, J., Wu, J., Xie, P., Wang, X., Xiao, X.: Hyper-sd: Trajectory segmented consistency model for efficient image synthesis. arXiv preprint arXiv:2404.13686 (2024)

[72] Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695 (2022)

[73] Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., Aberman, K.: Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2023)

[74] Sang, H., Xu, Y., Zhou, Z., He, R., Wang, Z., Sun, J.: On-policy self-distillation for reasoning compression. arXiv preprint arXiv:2603.05433 (2026)

[75] Sauer, A., Boesel, F., Dockhorn, T., Blattmann, A., Esser, P., Rombach, R.: Fast high-resolution image synthesis with latent adversarial diffusion distillation. In: SIGGRAPH Asia 2024 Conference Papers. pp. 1–11 (2024)

[76] Sauer, A., Lorenz, D., Blattmann, A., Rombach, R.: Adversarial diffusion distillation. In: European Conference on Computer Vision. pp. 87–103. Springer (2024)

[77] Seedance, T., Chen, D., Chen, L., Chen, X., Chen, Y., Chen, Z., Chen, Z., Cheng, F., Cheng, T., Cheng, Y., et al.: Seedance 2.0: Advancing video generation for world complexity. arXiv preprint arXiv:2604.14148 (2026)

[78] Seedream, T., Chen, Y., Gao, Y., Gong, L., Guo, M., Guo, Q., Guo, Z., Hou, X., Huang, W., Huang, Y., et al.: Seedream 4.0: Toward next-generation multimodal image generation. arXiv preprint arXiv:2509.20427 (2025)

[79] Shenfeld, I., Damani, M., Hübotter, J., Agrawal, P.: Self-distillation enables continual learning. arXiv preprint arXiv:2601.19897 (2026)

[80] Siméoni, O., Vo, H.V., Seitzer, M., Baldassarre, F., Oquab, M., Jose, C., Khalidov, V., Szafraniec, M., Yi, S., Ramamonjisoa, M., et al.: Dinov3. arXiv preprint arXiv:2508.10104 (2025)

[81] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)

[82] Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. arXiv preprint arXiv:2010.02502 (2020)

[83] Song, M., Zheng, M.: A survey of on-policy distillation for large language models. arXiv preprint arXiv:2604.00626 (2026)

[84] Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. arXiv preprint arXiv:2011.13456 (2020)

[85] Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. Advances in neural information processing systems **27** (2014)

[86] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2818–2826 (2016)

[87] Team, Z.I., Cai, H., Cao, S., Du, R., Gao, P., Hoi, S., Hou, Z., Huang, S., Jiang, D., Jin, X., Li, L., Li, Z., Li, Z.Y., Liu, D., Liu, D., Shi, J., Wu, Q., Yu, F., Zhang, C., Zhang, S., Zhou, S.: Z-image: An efficient image generation foundation model with single-stream diffusion transformer. arXiv preprint arXiv:2511.22699 (2025)

[88] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al.: Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971 (2023)

[89] Wallace, B., Dang, M., Rafailov, R., Zhou, L., Lou, A., Purushwalkam, S., Ermon, S., Xiong, C., Joty, S., Naik, N.: Diffusion model alignment using direct preference optimization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8228–8238 (2024)

[90] Wan, T., Wang, A., Ai, B., Wen, B., Mao, C., Xie, C.W., Chen, D., Yu, F., Zhao, H., Yang, J., et al.: Wan: Open and advanced large-scale video generative models. arXiv preprint arXiv:2503.20314 (2025)

[91] Wang, L., Xing, X., Cheng, Y., Zhao, Z., Li, D., Hang, T., Tao, J., Wang, Q., Li, R., Chen, C., et al.: Promptenhancer: A simple approach to enhance text-to-image models via chain-of-thought prompt rewriting. arXiv preprint arXiv:2509.04545 (2025)

[92] Wang, M., Jiang, D., Li, L., Lin, Y., Shen, G., Kong, X., Liu, Y., Dai, G., Wang, J.: Vae-repa: Variational autoencoder representation alignment for efficient diffusion training. arXiv preprint arXiv:2601.17830 (2026)

[93] Wang, P., Bai, S., Tan, S., Wang, S., Fan, Z., Bai, J., Chen, K., Liu, X., Wang, J., Ge, W., et al.: Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. arXiv preprint arXiv:2409.12191 (2024)

[94] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q.V., Zhou, D., et al.: Chain-of-thought prompting elicits reasoning in large language models. Advances in neural information processing systems **35**, 24824–24837 (2022)

[95] Wu, C., Li, J., Zhou, J., Lin, J., Gao, K., Yan, K., Yin, S.m., Bai, S., Xu, X., Chen, Y., et al.: Qwen-image technical report. arXiv preprint arXiv:2508.02324 (2025)

[96] Wu, G., Zhang, S., Shi, R., Gao, S., Chen, Z., Wang, L., Chen, Z., Gao, H., Tang, Y., Yang, J., et al.: Representation entanglement for generation: Training diffusion transformers is much easier than you think. arXiv preprint arXiv:2507.01467 (2025)

[97] Wu, X., Hao, Y., Sun, K., Chen, Y., Zhu, F., Zhao, R., Li, H.: Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. arXiv preprint arXiv:2306.09341 (2023)

[98] Xie, E., Chen, J., Chen, J., Cai, H., Tang, H., Lin, Y., Zhang, Z., Li, M., Zhu, L., Lu, Y., et al.: Sana: Efficient high-resolution image synthesis with linear diffusion transformers. arXiv preprint arXiv:2410.10629 (2024)

[99] Xu, J., Liu, X., Wu, Y., Tong, Y., Li, Q., Ding, M., Tang, J., Dong, Y.: Imagereward: Learning and evaluating human preferences for text-to-image generation. Advances in Neural Information Processing Systems **36**, 15903–15935 (2023)

[100] Xue, Z., Wu, J., Gao, Y., Kong, F., Zhu, L., Chen, M., Liu, Z., Liu, W., Guo, Q., Huang, W., et al.: Dancegrpo: Unleashing grpo on visual generation. arXiv preprint arXiv:2505.07818 (2025)

[101] Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., et al.: Qwen3 technical report. arXiv preprint arXiv:2505.09388 (2025)

[102] Yang, C., Qin, C., Si, Q., Chen, M., Gu, N., Yao, D., Lin, Z., Wang, W., Wang, J., Duan, N.: Self-distilled rlvr. arXiv preprint arXiv:2604.03128 (2026)

[103] Ye, H., Zhang, J., Liu, S., Han, X., Yang, W.: Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models. arXiv preprint arXiv:2308.06721 (2023)

[104] Ye, T., Dong, L., Wu, X., Huang, S., Wei, F.: On-policy context distillation for language models. arXiv preprint arXiv:2602.12275 (2026)

[105] Yin, T., Gharbi, M., Park, T., Zhang, R., Shechtman, E., Durand, F., Freeman, B.: Improved distribution matching distillation for fast image synthesis. Advances in neural information processing systems **37**, 47455–47487 (2024)

[106] Yin, T., Gharbi, M., Zhang, R., Shechtman, E., Durand, F., Freeman, W.T., Park, T.: One-step diffusion with distribution matching distillation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 6613–6623 (2024)

[107] Yu, S., Kwak, S., Jang, H., Jeong, J., Huang, J., Shin, J., Xie, S.: Representation alignment for generation: Training diffusion transformers is easier than you think. In: International Conference on Learning Representations (2025)

[108] Zhang, L., Rao, A., Agrawala, M.: Adding conditional control to text-to-image diffusion models. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 3836–3847 (2023)

[109] Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 586–595 (2018)

[110] Zhao, S., Xie, Z., Liu, M., Huang, J., Pang, G., Chen, F., Grover, A.: Self-distilled reasoner: On-policy self-distillation for large language models. arXiv preprint arXiv:2601.18734 (2026)

[111] Zheng, K., Chen, H., Ye, H., Wang, H., Zhang, Q., Jiang, K., Su, H., Ermon, S., Zhu, J., Liu, M.Y.: Diffusionnft: Online diffusion reinforcement with forward process. arXiv preprint arXiv:2509.16117 (2025)

[112] Zheng, K., Wang, Y., Ma, Q., Chen, H., Zhang, J., Balaji, Y., Chen, J., Liu, M.Y., Zhu, J., Zhang, Q.: Large scale diffusion distillation via score-regularized continuous-time consistency. arXiv preprint arXiv:2510.08431 (2025)

[113] Zhou, J., Wei, C., Wang, H., Shen, W., Xie, C., Yuille, A., Kong, T.: ibot: Image bert pre-training with online tokenizer. In: International Conference on Learning Representations (2022)

[114] Zhu, R., Pan, Y., Li, Y., Yao, T., Sun, Z., Mei, T., Chen, C.W.: Sd-dit: Unleashing the power of self-supervised discrimination in diffusion transformer. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8435–8445 (2024)

Appendix

A Investigation of FLUX.2-klein

We also perform a similar analysis with FLUX.2-klein [4] like those have done in Figure 1, as shown in Figure 8, the similar behavior can be seen, suggesting that this inherited in-context capability is broadly applicable to diffusion models that employ LLM/VLMs as encoders.



Figure 8: We also empirically investigate the difference of generated images when conditioned on only the text feature or the multimodal feature of target image and the text prompt using FLUX.2-klein-4B with 4 steps. Similar to Z-Image-Turbo, using multimodal features as condition instead of text-only features allows the model to produce image variations while maintaining the target image's underlying concept or stylistic identity.

B Discussion and Comparison of Different Training Paradigms

Table 4 summarizes the main differences among representative training paradigms for continually tuning step-distilled diffusion models. We compare them along four dimensions: the form of supervision signal, whether training is on-policy, whether an additional reward model or reward function is required, and whether the training process matches the model's inference behavior.

Table 4: Comparison of different training paradigms regarding the source of the supervision signal, the learning paradigm (whether its in an on-policy way), the necessity of an auxiliary reward model, and the consistency between training-time inference-time state.

Method	Supervision Signal	On-Policy	Reward Model	Train-Inference Match
SFT	GT velocity	×	×	×
RL-offline (e.g, Diffusion-DPO [89], PSO [59])	GT velocity pairs $(+/-)$	×	×	×
RL-online (e.g, ReFL [99], flow-grpo [48]	reward function	✓	✓	✓
D-OPSD (ours)	self-distilled velocity	✓	×	✓

Vanilla SFT. Vanilla supervised fine-tuning optimizes the model using the target image as the source of supervision. In flow-matching models, this means that training is performed on noised states constructed from the ground-truth image, with the model regressed toward the corresponding ground-truth velocity. While this objective is standard for training diffusion models from scratch, it is mismatched to continual tuning of step-distilled models. The reason is that both the optimization states and the supervision signal are induced by the target image, rather than by the model's own few-step sampling trajectory. As a result, although the model can absorb new concepts or styles from the training pairs, we suppose that it may do so by altering the distilled generation dynamics that are responsible for high-quality few-step sampling.

Offline RL-style methods. A natural alternative is to replace direct regression on a single target with preference-style or pairwise supervision. Representative examples include Diffusion-DPO [89]

and PSO [59][3]. These methods can be viewed as offline RL-style objectives, in the sense that their supervision is still derived from a fixed dataset (use target image related states as inputs and rely on ground-truth velocity-based pairwise supervision). Therefore, although methods like PSO try to specially adapt the few-step models, their optimization states and supervision signal are still not fully induced by the student's own current distribution. This also explains why, in our experimental results, PSO can often learn the target appearance but tends to overfit the training set in the small dataset and fails to learn in the large-scale settings.

Online RL-style methods. Online RL methods, such as ReFL [99] and flow-GRPO [48], are conceptually more suitable for preserving the behavior of step-distilled models because they optimize the model on its own sampled trajectories. In these methods, the model first generates images on-policy, and the resulting samples are then scored by a reward function or reward model. As such, both the optimization states and the supervision signal are tied to the current sampling process, substantially reducing the mismatch between training and inference. This is also an important reason why some studies that perform RL on the step-distilled model can make the model align with human preference without compromising the original few-step ability [36, 56]. However, this advantage comes at the cost of requiring a well-designed reward function or preference model. In practical customization scenarios, especially when secondary developers only possess a small number of image-text pairs, such reward design is often the main bottleneck.

Our method. Our method occupies a different point in this design space. Like online RL, D-OPSD is on-policy: optimization is performed on the student's own few-step roll-outs, so the model is always updated on states that it actually visits at inference time. More importantly, the supervision signal is also defined on these same states. Instead of introducing the target image as an external denoising target, D-OPSD uses it only to enrich the teacher's condition through multimodal in-context encoding, and supervises the student with self-distilled velocity predictions evaluated on the student's current trajectory. At the same time, unlike RL-based approaches, D-OPSD does not require any external reward model or manually designed reward function. In this sense, D-OPSD combines the main advantage of online optimization, train-inference consistency, with the practicality of supervised learning from paired image-text data.

Why this distinction matters for step-distilled models. For multi-step diffusion models, moderate train-test mismatch can sometimes be tolerated because iterative denoising provides room for error correction [84, 30, 82, 45]. Step-distilled models are less forgiving: with only a few denoising steps, even small deviations in the learned dynamics can directly harm image quality [105, 33]. For this reason, in our setting, it is important not only whether the training states are aligned with the model's own sampling trajectory, but also whether the supervision signal is defined under that same trajectory. This is the main motivation behind the design in Table 4. Among the compared paradigms, D-OPSD is the only one that simultaneously satisfies four desirable properties for continual tuning of few-step models: it is on-policy, does not require a reward model, preserves train-inference consistency, and still incorporates target image-text pairs into training through self-distillation.

C Comparison with Two-stage Training Pipeline

In this section, we compare D-OPSD with a two-stage training pipeline that first fine-tunes a multi-step model on the target data and then distills the tuned model back into a few-step model. We conduct the comparison using open-source Z-Image as the multi-step model and Z-Image-Turbo as its few-step counterpart. For the distillation stage, we adopt the open-source DMD algorithm [106, 105]. In the LoRA customization setting, we first train a LoRA adapter on Z-Image, and then train an additional step-distillation LoRA on the tuned model (Main model with LoRA). In the full-finetuning setting, both the tuning stage and the DMD distillation stage are performed with full model updates. The Training GPU hours are tested with the H800 GPU.

The results are reported in Table 5. Directly applying SFT to Z-Image-Turbo is computationally simple, but it substantially weakens the model's few-step generation capability, especially in the

[3] PSO can be regarded as a variant of the Diffusion-DPO [89] for the step-distilled model because it only conducts training at the few-step sampling timestep, but it still uses the target image state as input and uses the ground truth velocity for supervision..

Table 5: **System-level comparison with two-stage training pipelines.** We compare one-stage D-OPSD with SFT on the few-step model, SFT on the multi-step model, and a two-stage pipeline that applies DMD distillation after multi-step SFT. Experiments are conducted with Z-Image and Z-Image-Turbo. The **best** and second-best results on each metric are highlighted.

Method	DINO-D↓	Quality-S↑	GenEval↑	Inference NFE↓	Training GPU hours↓
LoRA customization setting					
SFT on Z-Image-Turbo	0.2212	2.4236	-	**8**	**5**
SFT on Z-Image	**0.0547**	3.3488	-	100	**5**
SFT on Z-Image + DMD	0.1024	3.3372	-	**8**	5 + 882
D-OPSD (ours)	0.0823	**3.7965**	-	**8**	9
Full-finetuning setting					
SFT on Z-Image-Turbo	0.2565	2.5038	0.1588	**8**	**1066**
SFT on Z-Image	**0.0843**	3.3801	0.6578	100	**1066**
SFT on Z-Image + DMD	0.1094	3.3652	0.6084	**8**	1066 + 2054
D-OPSD (ours)	0.1088	**3.8438**	**0.7170**	**8**	1918

LoRA setting, where the Quality-S score drops from the original few-step model quality regime. This is consistent with our main observation: vanilla SFT optimizes on target-image-induced states rather than on the states visited by the model's own few-step sampler, thereby perturbing the distilled few-step dynamics. Fine-tuning the multi-step model instead gives the best DINO-D in both settings, indicating strong fitting to the target domain. However, this model requires 100 NFEs at inference time and therefore loses the main efficiency advantage of step-distilled models. Moreover, in the full-finetuning setting, it shows a clear degradation in Quality-S and GenEval, suggesting that adapting to the target domain alone does not preserve the generation behavior desired from the few-step model.

The two-stage pipeline partially addresses the inference cost by applying DMD after multi-step SFT, reducing inference to 8 NFEs. However, it introduces substantial extra training cost and still does not recover the overall performance of the tuned model. In particular, the DMD-distilled variants obtain lower Quality-S and GenEval scores than D-OPSD, indicating degradation in both few-step generation quality and retained prior knowledge. This limitation is partly practical: the strongest distillation recipes used by SOTA few-step foundation models are often not fully public, including key hyperparameters and engineering details [46, 36]. Therefore, reproducing the closed-source distillation pipeline of the original base model is not feasible for most secondary developers. With the publicly available DMD implementation, the additional distillation stage is expensive and can still introduce performance loss.

In contrast, D-OPSD performs continual tuning in a single stage. It learns the new target-domain information while keeping both optimization and supervision on the student's own roll-outs, which preserves the original few-step sampling behavior. Although D-OPSD requires on-policy student roll-outs and teacher inference during training, leading to roughly $4\times$ FLOPs and about $2\times$ wall-clock training time per iteration compared with vanilla SFT, we consider this cost acceptable for continual tuning of few-step diffusion models. The roll-out and teacher-inference computations do not require gradient computation and can be further optimized in engineering. More importantly, once the cost of re-distillation is included, D-OPSD is more resource-efficient than the two-stage pipeline while achieving better few-step quality and stronger retention of prior knowledge.

D Implementation Details.

We now provide a detailed training and implementation settings of D-OPSD as follows:

Encoder settings of student and teacher. During training, for the student model, we use the original text encoder of the diffusion model (for both Z-Image-Turbo and FLUX.2-klein, text prompts are encoded using Qwen3-4B [101]). For the teacher model, since target image information must be incorporated, a straightforward solution is to replace Qwen3-4B with the corresponding Qwen3-VL-4B [2]. However, in practice, we find that this naive substitution introduces high-frequency artifacts and excessive sharpening in the generated images (See Figure 9, the middle column). We attribute this issue to a mismatch of the feature space between inference and training distribution.[4]

[4]Since Qwen3-VL is a continually trained variant of Qwen3-LM, the model can still drive image generation, but its output distribution no longer aligns well with that of the diffusion model's original training setup.

To address this issue, we replace the weights of the LLM component in Qwen3-VL-4B with those from the more compatible Qwen3-4B, while keeping the ViT and Connector weights unchanged. In this way, we preserve multimodal in-context capability while making the output distribution as consistent as possible with that seen during diffusion model training (See Figure 9 the last column). Notably, this operation can be viewed as approximately reverting the VLM to the stage where the Connector has been trained, but the LLM parameters remain unchanged [47, 93, 11]. Although the final VLM exhibits stronger multimodal capabilities, the model at this earlier stage still retains a certain degree of multimodal understanding. This reflects a trade-off between preserving the output image quality of the final diffusion model and maintaining the in-context capability provided by the VLM. As more LLMs/VLMs evolve toward native multimodal architectures [13, 21, 65], we expect this trade-off to be naturally alleviated in future diffusion models using stronger multimodal encoders, such as Qwen3.5 [65].

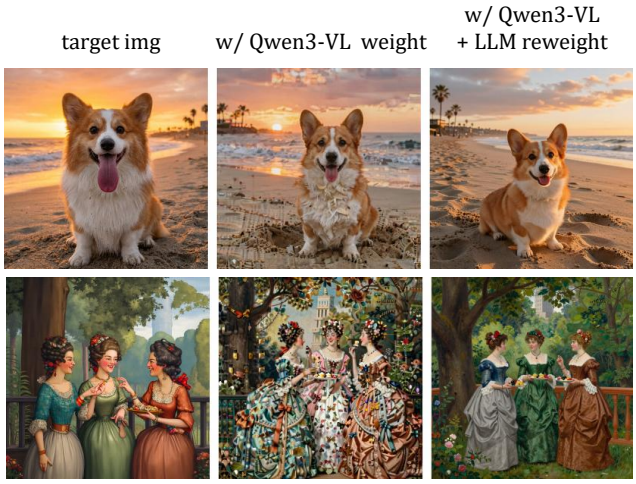


Figure 9: Comparison of generated images of Z-Image-Turbo conditioned on multimodal feature using Qwen3-VL 4B and Qwen3-VL 4B with LLM part reweighted by Qwen3-4B LM.

Compatibility and quantitative validation. The construction uses Qwen3-4B and Qwen3-VL-4B, whose language backbones belong to the same model family and are architecture-compatible. The unchanged ViT and Connector convert the target image into visual tokens, which are processed jointly with the text tokens; the resulting hidden states enter the diffusion backbone through its original conditioning interface. Swapping the pretrained language weights requires no additional training. The feature-space compatibility explanation above is an interpretation of this empirical result.

Table 6: **Encoder ablation before fine-tuning.** Results on the 40 concept/style tasks with Z-Image-Turbo.

Encoder	DINO-D↓	LPIPS-D↓	Quality-S↑
Qwen3-4B (text only)	0.2373	0.7520	3.7455
Qwen3-VL-4B (naive replacement)	0.0974	0.6046	3.0061
Weight-swapped encoder (ours)	**0.0807**	**0.5728**	**3.8242**

Table 6 compares the original text encoder, naive VLM replacement, and our weight-swapped encoder before fine-tuning. Naive replacement improves target-image alignment but substantially reduces image quality. Weight swapping further improves alignment while recovering generation quality, supporting the encoder construction illustrated in Figure 9.

LoRA training on small customized dataset:

- **Z-Image-Turbo**: We use LoRA with rank 64 and alpha 128 to finetune the model. We set the total batch size to 4 and the learning rate to 4e-5[5]. The momentum coefficient of EMA decay is set to 0.9999 by default. (Note that since the model is trained with LoRA, we only need to use one main weight and then perform the EMA operation on the LoRA weights. This can effectively save the memory). The model is trained with 1K iterations on a single H800 GPU.

[5]The learning rate for Z-Image-Turbo is consistently set higher than that for FLUX.2-klein because we find that the norm of the parameters in Z-Image-Turbo is greater than that of other models, including FLUX.2-klein. Consequently, a proportionally larger learning rate is required to maintain effective parameter updates during optimization.

- **FLUX.2-klein**: We also use LoRA with rank 64 and alpha 128 to finetune the model. We set the total batch size to 4 and the learning rate to 1e-5. The momentum coefficient of EMA decay is set to 0.9999 by default. The model is trained with 1K iterations on a single H800 GPU.

Full finetuning on larger scale dataset:

- **Z-Image-Turbo**: In this setup, we unlock all the parameters in the diffusion transformer backbone. We set the total batch size to 256 and the learning rate to 3e-5. The momentum coefficient of EMA decay is set to 0.9999 by default. The model is trained with 10k iterations on 32 H800 GPUs.
- **FLUX.2-klein**: We also unlock all the parameters in the diffusion transformer backbone. We set the total batch size to 256 and the learning rate to 8e-6. The momentum coefficient of EMA decay is set to 0.9999 by default. The model is trained with 10K iterations on 32 H800 GPUs.

Optional prior-preservation regularization. We further evaluate an optional regularizer in the Z-Image-Turbo full-finetuning setting of Table 2. On replay prompts, the regularizer matches the current model's velocity predictions to those of the frozen base model, with weight λ. Table 7 shows that increasing λ generally improves prior preservation at the cost of target-domain adaptation. The regularizer requires an additional base-model forward pass; standard D-OPSD uses $\lambda = 0$ and requires no replay data.

Table 7: **Adaptation–preservation trade-off.** Optional base-velocity regularization during full fine-tuning.

λ	DINO-D↓	LPIPS-D↓	GenEval↑	DPG↑
0	0.1088	0.6419	0.7170	84.1116
0.1	0.1106	0.6463	0.7246	84.2241
0.5	0.1168	0.6607	0.7392	84.5027
1.0	0.1431	0.6814	0.7490	84.2778

E More Details of Evaluation

We now provide the detailed evaluation protocols.

LoRA training on a small customized dataset. In this setting, we attempt to follow the community's common secondary fine-tuning setup, where a concept is learned through LoRA training on the base model using only a small number of text–image pairs (e.g, less then 10). Thus, we adopt DreamBooth-style data [73] for both fine-tuning and evaluation. To ensure stable results, the scores in Table 1 are averaged over multiple categories (30 concept class and 10 style class) rather than selected from one or two cherry-picked cases. We use the following metric to evaluate the model:

- **DINO distance (DINO-D)**: We first take the training captions of the training images and use the LLM to paraphrase the captions without changing their core semantics. These rewritten prompts are then used to generate images with the fine-tuned model. We compute the cosine distance between the DINO features of the generated images and those of the corresponding target images. For a specific model, we use DINOv3-ViT-S-plus [80] as the feature extractor.
- **LPIPS distance (LPIPS-D)**: Similar to DINO-D, we compute the LPIPS distance [109] between the generated images and the corresponding target images. For a specific model, we adopt the commonly used VGG network [81] for this metric.
- **VLM's judgment of subject or style consistency (VLM-J)**: We fix the learned training concept and ask the LLM to construct four groups of prompts that differ from the training prompts but still contain the same concept. For object concepts, the new prompts vary aspects such as scene and composition; for style concepts, the new prompts describe specific image contents different from those seen during training. We use these prompts to generate images with the fine-tuned model, and then feed the generated images together with the target image into the VLM, which is asked to assess the similarity of the concept and assign a score followed the rule of scoring: 4 points(basically the same); 3 points(relatively similar); 2 points (slightly similar); 1 point (completely different). For specific model, we use Qwen3-VL-8B-Instruct [2] for evaluation.

- **CLIP Score (CLIP-S)**: Using the same images as in VLM-J, we further evaluate whether the model still follows the non-concept parts of the prompt, such as the background. Specifically, we compute the image-text alignment between the generated image and the prompt using CLIP [66, 27]. Note that we replace the special DreamBooth class token ([V]) with the original class name when computing this metric. For a specific model, we use DFN-CLIP-H [17] for evaluation.
- **Quality Score (Quality-S) and Aesthetic Score (Aesthetic-S)**: We use our internal VLM-based reward model for scoring. Compared with traditional open-source CLIP-based reward models, such as ImageReward [99] and PickScore [39], the model does not require the text prompt used for image generation as input, and provides more reliable scores thanks to large-scale training with dedicated human preference annotations and a reasoning process before giving a final score during inference. Note that we do not explicitly optimize either of these two scores during fine-tuning. Moreover, all compared methods are evaluated under the same scoring protocol and criteria.

Full finetuning on larger scale dataset: In this setting, we fully fine-tune the model like a normal large-scale SFT. Given that we use the latest state-of-the-art open-source models [87, 4] as our baseline, most publicly available open-source datasets are unsuitable for our full finetuning experiment, as their overall quality is lower than that of the data used to train these baseline models. Therefore, we rely on an in-house dataset of 25K high-quality anime images. We use the following metric to evaluate the model:

- **Fréchet Inception Distance (FID) [28]**: We randomly sample 2K data from the training set, and use the fine-tuned model to generate the images from these prompts. the two sets of images are then fed into the Inception-v3 network [86] to extract features. Assumes both feature distributions are multivariate Gaussian and computes the Fr´echet distance between them.
- **DINO distance (DINO-D)**: The images generated for calculating FID score will also be used to calculate DINO distance, following the rules used in the above LoRA evaluation settings.
- **LPIPS distance (LPIPS-D)**: Similar to DINO-D, the images generated for calculating FID score will also be used to calculate LPIPS distance following the rules used in the above LoRA evaluation settings.
- **Quality Score (Quality-S) and Aesthetic Score (Aesthetic-S)**: Similar to both DINO-D and LPIPS-D, the images generated for calculating FID score will also be used to calculate Quality Score and Aesthetic Score from the Reward model we introduced above.
- **GenEval and DPG score**: We follow the evaluation settings in these benchmarks [20, 32] to generate images and calculate score. Note that we use prompts from the original benchmark instead of the Prompt-Enhanced (PE) [91] variants for generating images.

User study. Following prior work [105, 57, 73], we conduct a comprehensive human evaluation to assess overall generation performance. Specifically, we sample 50 prompts from the small-scale LoRA customization setting and another 50 prompts from the large-scale full-finetuning setting, yielding 100 evaluation prompts in total. For the small-scale setting, we additionally provide annotators with the corresponding reference images or style exemplars, so that prompt following can be judged with respect to both the textual instruction and the target concept/style to be learned. To ensure fair and systematic assessment, we compare paired outputs from our method and the baselines under randomized presentation, and ask annotators to evaluate them along three dimensions: image quality, aesthetic quality, and prompt following. We then aggregate the preferences from both settings into a single overall score, and report the final results in Figure 3.

F More Related Works

We now provide a detailed literature review of other related work.

Image generation diffusion models. Diffusion models have achieved remarkable success in image generation. Early works mainly study unconditional or class-conditional generation, demonstrating that progressively denoising noise can produce highly realistic images [15, 38]. This framework

was then extended to text-to-image synthesis, where text conditions are injected into the denoising network to guide image generation according to natural language descriptions [70, 72]. Later, latent-space diffusion models and stronger text-conditioning designs further improve both efficiency and generation quality, making diffusion models the mainstream solution for text-to-image generation [62, 10, 72]. Building on this, recent studies introduce more scalable backbones and objectives, such as diffusion transformers and flow/rectified-flow formulations, which continue to push the frontier of image fidelity, prompt following, and training scalability [61, 58, 10, 16, 3, 98]. Meanwhile, unlike earlier models that usually rely on CLIP [66] or T5 [69] as the condition encoder, the latest high-performance image generation models increasingly adopt large language models or vision-language models as the encoder [63, 87, 95, 4]. Our method is built upon this evolution: we show that the diffusion model can benefit from the in-context capability inherited from these modern encoders, which makes on-policy self-distillation feasible for continuously tuning step-distilled diffusion models.

Knowledge distillation for diffusion model. Besides step-distillation for faster sampling, other forms of knowledge distillation [29, 23] is also widely used in diffusion models. Commonly, a more powerful pretrained model is often used to guide the diffusion model during training [14, 107, 18, 96]. Meanwhile, self-distillation frameworks also demonstrates effectiveness even without external components (e.g, stronger model) [37, 92, 114, 8, 24]. For example, SRA [37] aligns the output latent representation of the diffusion transformer in earlier layer with higher noise to that in later layer with lower noise to progressively enhance the overall representation learning during only generative process and accelerate the training convergence of the model, which has also been demonstrated in subsequent study [8] to be applicable in multiple modalities (video, audio, etc.). Our work is also related to self-distillation, while our approach is conducted in an on-policy way for preserving few-step inference capacity during supervised fine-tuning.

Diffusion model fine-tuning. A large body of work studies how to adapt pretrained diffusion models to new concepts, styles, or downstream domains [73, 103, 19, 42, 108, 44]. Representative approaches include standard supervised fine-tuning with all model parameters, subject-driven customization methods such as DreamBooth [73], textual inversion [19], and parameter-efficient adaptation techniques such as LoRA [31]. Among them, full-parameter SFT remains a common choice when sufficient paired data are available, as it directly optimizes the full generative model on the target distribution. However, such standard fine-tuning paradigms are mainly developed for conventional multi-step diffusion models, and they do not explicitly consider whether the adapted model can preserve the few-step inference capability of a step-distilled generator. In particular, directly applying the commonly used denoising or flow-matching objective during SFT would lead to degradation of their original fast-sampling behavior. In contrast, our work focuses on continuously tuning already distilled diffusion models and specifically targets preserving their native few-step generation ability during supervised adaptation.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main contributions of D-OPSD, including on-policy self-distillation for step-distilled diffusion models and validation under LoRA and full fine-tuning settings. These claims are supported by the method in Section 2 and the experimental results in Section 3.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 4 discusses limitations and future work, including the additional computational cost of on-policy roll-outs and the dependence on the teacher/base model's in-context capability. It also describes cases where ineffective teacher supervision can cause training failure.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not have formal theoretical results requiring theorem statements or proofs. Section 2 provides the method formulation and training objective rather than a theoretical guarantee.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Sections 3 and Appendices C–D disclose the models, training hyperparameters, evaluation metrics, and protocols in detail. We will release the training code and provide access to the internal reward-model evaluation, either through APIs or model weights, so that the main results can be partially reproduced; the remaining metrics rely on standard evaluation protocols can be reproduced from the detailed information provided in the paper.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The paper provides implementation and evaluation details in the appendices, and publicly known assets such as DreamBooth-style data and benchmark protocols are referenced where applicable. For the full fine-tuning setting, it uses an in-house anime dataset because, as discussed in the appendix, publicly available datasets are not sufficiently aligned with the quality and domain requirements of the evaluated state-of-the-art base models; Note that all compared methods are trained and evaluated under the same dataset setting to ensure fair comparison, and the core contribution of this paper is the proposed training algorithm rather than the dataset itself.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 3 describes the experimental setup, baselines, datasets, and evaluation metrics, while Appendices C and D provide detailed training hyperparameters, GPU settings, and evaluation protocols. These details are sufficient to understand the reported comparisons and conclusions.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.

- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The main experiments are evaluated using standard aggregate image-generation metrics and benchmark scores, rather than hypothesis-testing-based comparisons. Therefore, statistical significance tests or error bars are not central to the evaluation protocol of this paper.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: Appendix C reports the number and type of GPUs used, such as one H800 GPU for LoRA training and 32 H800 GPUs for full fine-tuning, along with iteration counts and batch sizes.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

Justification: To the best of our knowledge, the research uses existing models, image-text datasets, and evaluation protocols in a manner consistent with responsible ML research practice. The paper does not report human-subject experiments or procedures that would require deviation from the NeurIPS Code of Ethics.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [No]

Justification: This paper proposes a new training algorithm for scientific research on continual tuning of step-distilled diffusion models, and does not release a new model, dataset, or deployed system. Therefore, direct societal impacts are not the primary focus of this work and are not discussed in detail.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper proposes a training method and does not describe releasing a new high-risk pretrained model, image generator, or scraped dataset. Therefore, release safeguards for such assets are not applicable to the current submission.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper cites existing models, datasets, benchmarks, and evaluation models, including Z-Image-Turbo, FLUX.2-klein, DreamBooth-style data, GenEval, DPG, CLIP, DINO, and Qwen models.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release a new dataset, code package, or model asset as a contribution.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [No]

Justification: The authors and other researchers evaluated image quality, aesthetics, and prompt alignment, as described in Appendix E.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [No]

Justification: The paper includes a human evaluation of generated images, as described in Appendix E. However, it does not report potential risks to participants, whether these risks were disclosed, or the status of IRB approval or equivalent institutional review.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: This paper is only corrected for grammar and words using LLM.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.