FLOWCTS: ON-POLICY CONTINUOUS TRAJECTORY SUPERVISION OF FLOW MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

While on-policy distillation (OPD) effectively addresses sparse rewards and exposure bias in large language model post-training, its extension to flow models remains underexplored. To this end, we propose Flow Continuous Trajectory Supervision (FLOWCTS), which matches subsequent student and reference trajectories initialized from the same student-visited state. Using the integral relation between trajectories and velocity fields, we derive a temporally weighted velocity-matching upper bound and discretize it into practical objectives parameterized by the number of supervision steps. Under a multi-reference setup, FLOWCTS-OPD outperforms vanilla KL-based OPD with faster convergence. Compared with the SD3.5-M base model, it improves GenEval from 0.63 to 0.93, OCR from 0.59 to 0.92, and PickScore from 21.72 to 23.06, matching task experts and surpassing the GenEval expert under multi-step supervision, while outperforming mixed-reward RL across all metrics. Further analysis reveals a clear temporal supervision mismatch in vanilla KL-based OPD arising from its auxiliary SDE transition kernels. Beyond on-policy setting, FLOWCTS also consistently outperforms vanilla SFT especially on OCR, while increasing supervision steps exhibit a trade-off between richer trajectory information and greater optimization difficulty.

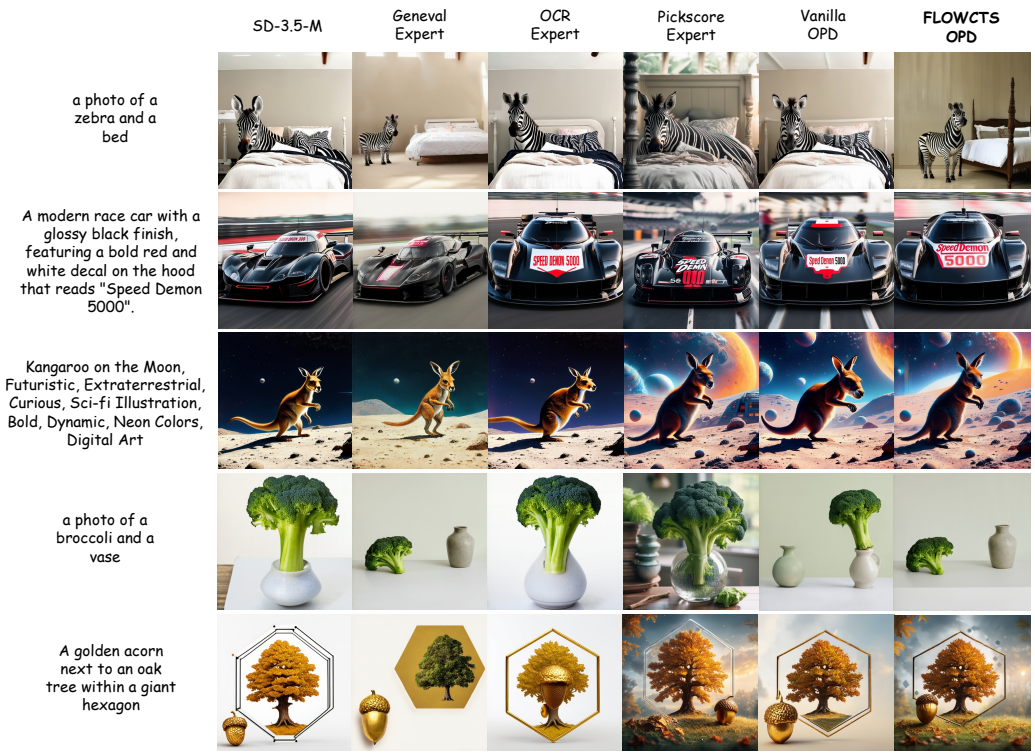


Figure 1: Qualitative comparison of generated images. Our method FLOWCTS-OPD produces images with improved alignment and visual quality compared with previous approaches.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

Qwen3, MiMo-v2, and GLM-5 have recently attributed part of their post-training gains to an emerging paradigm: on-policy distillation (OPD) (Yang et al., 2025; Xiao et al., 2026; Zeng et al., 2026). This approach, as illustrated in Figure 2 (a), combines on-policy training with dense reference supervision (Agarwal et al., 2024; Gu et al., 2024). Although its training dynamics remain incompletely understood, these successes highlight the value of OPD for future generative-model development (Li et al., 2026c; Hou et al., 2026). In imitation learning, it is well established that learned policies benefit from training under their induced state distributions, rather than under those encountered by the expert (Ross & Bagnell, 2010; Ross et al., 2011). This facilitates recovery from mistakes and helps reduce compounding errors. Moreover, OPD provides such benefits by training on the student's rollout distribution with dense token-level reference supervision, thereby reducing reliance on sparse outcome rewards (Lu & Lab, 2025; Ma et al., 2026a; Yang et al., 2026). Inspired by these, this work investigates how to effectively formulate dense OPD supervision for flow models.

To this end, we first revisit the form of supervision in token-based autoregressive language models (Gu et al., 2024; Li et al., 2026c), whose original objective minimizes the sequence-level reverse KL divergence between the student's and the reference's output sequence distributions given a prompt. The chain-rule factorization of language modeling decomposes this objective exactly into *next token supervision* over a finite vocabulary at student-generated prefixes (Agarwal et al., 2024; Wen et al., 2023). However, flow models parameterize a continuous velocity field to model the continuous state trajectory from noise to data and lack a natural next-step conditional distribution (Lipman et al., 2022; Liu et al., 2022; Tong et al., 2023; Liu et al., 2024; Kornilov et al., 2024; Frans et al., 2025; Geng et al., 2025). This motivates us to revisit the essence of *sequence supervision* in OPD, shifting the supervision target from next-step distributions to the trajectory itself.

We therefore propose Flow Continuous Trajectory Supervision (FLOWCTS), which supervises the discrepancy between the student and reference continuous trajectories that evolve from the same shared state. FLOWCTS-OPD instantiates this objective in the on-policy setting, where the shared starting state is sampled from the current student's rollout. We further derive an upper bound of the continuous trajectory supervision as a tractable optimization objective and discretize it into a finite number of optimization steps. The discretized objective is parameterized by the number of supervision steps K, covering single-state velocity matching and multi-step trajectory supervision. In the on-policy setting, single-state FLOWCTS-OPD outperforms vanilla KL-based OPD with faster convergence. Specifically, it improves GenEval from 0.63 to 0.93, OCR from 0.59 to 0.92, and PickScore from 21.72 to 23.06, while also showing improved out-of-distribution generalization. Qualitative examples are shown in Figure 1. Further analysis reveals a clear temporal supervision mismatch in vanilla KL-based OPD, which constructs transition kernels via an auxiliary SDE.

More broadly, FLOWCTS does not restrict how the shared starting state is sampled. When the starting state is constructed from an off-policy data trajectory, the same formulation extends to FLOWCTS-SFT. Experiments show that FLOWCTS-SFT consistently outperforms vanilla SFT in overall score, with the most pronounced gains on OCR. Although increasing the supervision steps does not bring consistent improvements, FLOWCTS provides a new scaling law dimension of flow models that trades richer trajectory information against greater optimization difficulty. In summary, our contributions are:

- We take an early exploration toward dense on-policy supervision for flow models. Revisiting the essence of sequence supervision in OPD for language modeling, we propose FLOWCTS from a continuous trajectory supervision perspective.
- The discretized objective of FLOWCTS covers both single-state and multi-step supervision, introducing the supervision step as a new scaling dimension for flow models.
- We extended FLOWCTS to both on-policy distillation and off-policy SFT settings, with experiments validating its effectiveness in both settings.
- We attribute the advantage of FLOWCTS to the dense supervision in early denoising stages, where the student–reference discrepancy is largest. In contrast, KL-based FlowOPD concentrates supervision in late stages and suffers from this temporal supervision mismatch.

2 PRELIMINARIES

In this section, we begin by reviewing flow matching and standard on-policy distillation, which together motivate our trajectory-based OPD formulation.

Flow Matching. Flow matching defines a transport trajectory between the data and noise distributions via an ordinary differential equation (ODE). Formally, let $\boldsymbol{x}_0 \sim p_{\text{data}}$ be a data sample and $\boldsymbol{x}_1 \sim p_{\text{noise}} = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ be a noise sample. The transport trajectory is commonly defined by linear interpolation (Liu et al., 2022):

$$\boldsymbol{x}_t = (1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1, \quad t \in [0,1]. \tag{1}$$

The trajectory satisfies the ODE $d\boldsymbol{x}_t = \boldsymbol{v}_t(\boldsymbol{x}_t, t)dt$, and under the linear path, the conditional target velocity is $\boldsymbol{u}_t(\boldsymbol{x}_t \mid \boldsymbol{x}_0, \boldsymbol{x}_1) = d\boldsymbol{x}_t/dt = \boldsymbol{x}_1 - \boldsymbol{x}_0$. Flow matching trains a neural velocity field $\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ by minimizing

$$\mathcal{L}_{\text{FM}}(\boldsymbol{\theta}) = \mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{x}_1}\left[\left\|\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - (\boldsymbol{x}_1 - \boldsymbol{x}_0)\right\|^2\right]. \tag{2}$$

On-Policy Distillation. OPD provides reference supervision on samples generated by the current student. Given a prompt $x \sim \mathcal{D}_x$, the student samples a response $\hat{y}_{1:T} \sim \pi_{\boldsymbol{\theta}}(\cdot \mid x)$. At each student-generated prefix $(x, \hat{y}_{<t})$, the student and reference define next-token distributions $p^t_{\boldsymbol{\theta}}(\cdot) = \pi_{\boldsymbol{\theta}}(\cdot \mid x, \hat{y}_{<t})$ and $p^t_{\text{T}}(\cdot) = \pi_{\text{T}}(\cdot \mid x, \hat{y}_{<t})$.

A standard OPD objective minimizes the sequence-level reverse KL between the student and reference:

$$\mathcal{L}_{\text{OPD}}(\boldsymbol{\theta}) = \mathbb{E}_{x\sim\mathcal{D}_x}\left[D_{\text{KL}}\left(\pi_{\boldsymbol{\theta}}(\cdot \mid x) \,\|\, \pi_{\text{T}}(\cdot \mid x)\right)\right]. \tag{3}$$

By autoregressive factorization, this objective can be written as the sum of token-level KL terms on student-generated prefixes.

$$\mathcal{L}_{\text{OPD}}(\boldsymbol{\theta}) = \mathbb{E}_{x\sim\mathcal{D}_x,\ \hat{y}_{1:T}\sim\pi_{\boldsymbol{\theta}}(\cdot|x)}\left[\sum_{t=1}^{T} D_{\text{KL}}\left(p^t_{\boldsymbol{\theta}} \,\|\, p^t_{\text{T}}\right)\right]. \tag{4}$$

Unlike autoregressive models, whose sequence factorization naturally yields next-token conditional distributions for local matching, flow models generate samples through a continuous state trajectory from noise to data and do not natively provide such a next-step distribution. Yet, given the same visited state, the student and reference models define distinct subsequent trajectories. This observation motivates us to formulate OPD at the trajectory level by matching their subsequent evolution over a finite time interval.

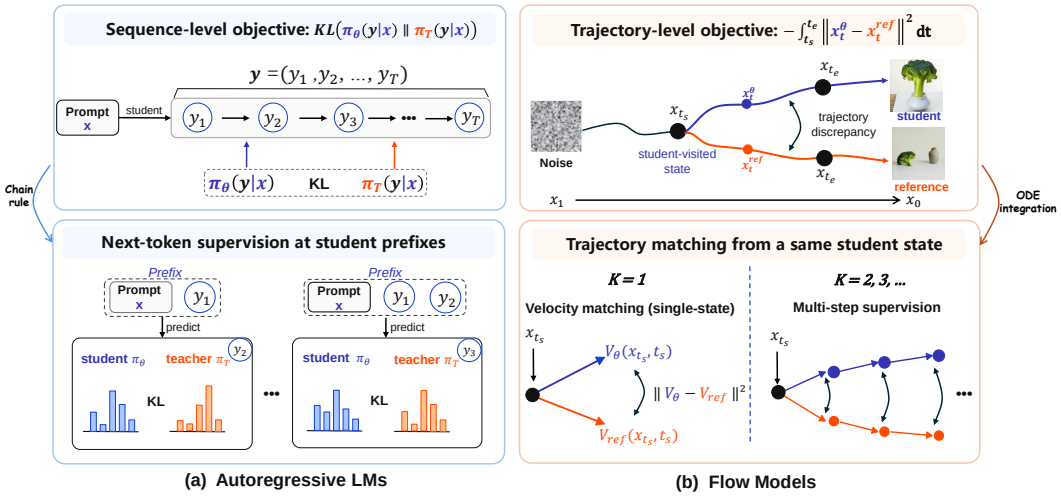
3 A CONTINUOUS-TRAJECTORY PERSPECTIVE ON OPD FOR FLOW MODELS

In this section, we study how to formulate OPD for flow matching models from a continuous-trajectory perspective, as illustrated in Figure 2. Under OPD, we first sample a student-visited state $\boldsymbol{x}_t$ along the current student's generation trajectory. From this state, the student and reference velocity fields induce two subsequent trajectories. We formulate their discrepancy, derive a tractable velocity-based upper bound, and discretize it into a family of practical training objectives. We further discuss how the same formulation extends to off-policy starting states.

3.1 A TRAJECTORY-LEVEL FORMULATION

We begin with a trajectory-level formulation over a reverse-time segment. Let $0 \le t_{\text{e}} < t_{\text{s}} \le 1$, where t_{s} and t_{e} denote the start and end times of the segment, respectively. Let $\Phi^{\boldsymbol{\theta}}_{t_{\text{s}}\to t}$ and $\Phi^{\text{ref}}_{t_{\text{s}}\to t}$ denote the flow maps induced by the student and reference velocity fields, respectively. Given a starting state $\boldsymbol{x}_{t_{\text{s}}}$ and a condition c, the corresponding trajectories are $\boldsymbol{x}^{\boldsymbol{\theta}}_t = \Phi^{\boldsymbol{\theta}}_{t_{\text{s}}\to t}(\boldsymbol{x}_{t_{\text{s}}}, c)$ and $\boldsymbol{x}^{\text{ref}}_t = \Phi^{\text{ref}}_{t_{\text{s}}\to t}(\boldsymbol{x}_{t_{\text{s}}}, c)$. Their discrepancy is defined as:

$$\mathcal{L}_{\text{CTS}}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}_{t_{\text{s}}},c}\left[\int_{t_{\text{e}}}^{t_{\text{s}}} \left\|\boldsymbol{x}^{\boldsymbol{\theta}}_t - \boldsymbol{x}^{\text{ref}}_t\right\|^2 dt\right]. \tag{5}$$



(a) Autoregressive LMs (b) Flow Models

Figure 2: **Autoregressive OPD and trajectory-level supervision for flow models.** In autoregressive models, sequence-level reverse KL decomposes by the chain rule into next-token supervision at student-generated prefixes. Since flow models do not naturally admit an analogous next-step factorization, FlowCTS instead matches student and reference trajectories initialized from the same student-visited state. Using the integral relation between trajectories and velocity fields, we derive a temporally weighted velocity-matching upper bound and discretize it into practical objectives parameterized by the supervision steps K, with K=1 recovering single-state velocity matching.

We refer to this trajectory-level formulation as Continuous Trajectory Supervision (CTS). It defines an on-policy distillation objective when the starting state $\boldsymbol{x}_{t_\mathrm{s}}$ is sampled from the current student rollout, and an off-policy supervision objective when it is constructed from an offline data trajectory.

3.2 A Velocity-Based Upper Bound

Flow trajectories are generated by integrating time-dependent velocity fields (Lipman et al., 2022; Liu et al., 2022). Therefore, the discrepancy between the student and reference trajectories is determined by the accumulated mismatch between their velocities. Based on this relation, we derive a weighted velocity-matching upper bound of the trajectory loss, yielding a tractable objective defined directly on the model outputs.

Specifically, since both trajectories start from the same state $\boldsymbol{x}_{t_\mathrm{s}}$, their states at any $t \in [t_\mathrm{e}, t_\mathrm{s}]$ are obtained by integrating the corresponding velocity fields:

$$\boldsymbol{x}_t^{\boldsymbol{\theta}} = \boldsymbol{x}_{t_\mathrm{s}} - \int_t^{t_\mathrm{s}} \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_r^{\boldsymbol{\theta}}, r, c)\, dr, \qquad \boldsymbol{x}_t^{\mathrm{ref}} = \boldsymbol{x}_{t_\mathrm{s}} - \int_t^{t_\mathrm{s}} \boldsymbol{v}_{\mathrm{ref}}(\boldsymbol{x}_r^{\mathrm{ref}}, r, c)\, dr. \tag{6}$$

Subtracting the two equations gives:

$$\boldsymbol{x}_t^{\boldsymbol{\theta}} - \boldsymbol{x}_t^{\mathrm{ref}} = -\int_t^{t_\mathrm{s}} \left[\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_r^{\boldsymbol{\theta}}, r, c) - \boldsymbol{v}_{\mathrm{ref}}(\boldsymbol{x}_r^{\mathrm{ref}}, r, c)\right] dr. \tag{7}$$

This identity shows that the trajectory discrepancy arises from the accumulated velocity mismatch along the integration path. It leads to the following weighted velocity-matching upper bound, which we term the velocity upper bound (VUB):

$$\mathcal{L}_{\mathrm{CTS}}(\boldsymbol{\theta}) \le \mathcal{L}_{\mathrm{VUB}}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}_{t_\mathrm{s}}, c}\left[\int_{t_\mathrm{e}}^{t_\mathrm{s}} w(r) \left\|\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_r^{\boldsymbol{\theta}}, r, c) - \boldsymbol{v}_{\mathrm{ref}}(\boldsymbol{x}_r^{\mathrm{ref}}, r, c)\right\|^2 dr\right], \tag{8}$$

where the trajectory-derived weight is $w(r) = \int_{t_\mathrm{e}}^{r} (t_\mathrm{s} - t)\, dt = \left[(t_\mathrm{s} - t_\mathrm{e})^2 - (t_\mathrm{s} - r)^2\right]/2$. This weight arises from the trajectory bound rather than manual design. It increases from zero at the segment end t_e to its maximum at the segment start t_s, assigning larger weights to velocity errors closer to t_s, as they affect a longer subsequent trajectory. A full derivation is provided in Appendix A.

3.3 Discrete Training Objective

We now discretize the reverse-time segment $[t_e, t_s]$ into K Euler steps. Let $h = (t_s - t_e)/K$ and define $t_i = t_s - ih$ for $i = 0, \ldots, K$. Starting from the same state $\boldsymbol{x}_0^{\boldsymbol{\theta}} = \boldsymbol{x}_0^{\text{ref}} = \boldsymbol{x}_{t_s}$, the student and reference rollouts are given by

$$\boldsymbol{x}_{i+1}^{\boldsymbol{\theta}} = \boldsymbol{x}_i^{\boldsymbol{\theta}} - h\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_i^{\boldsymbol{\theta}}, t_i, c), \qquad \boldsymbol{x}_{i+1}^{\text{ref}} = \boldsymbol{x}_i^{\text{ref}} - h\boldsymbol{v}_{\text{ref}}(\boldsymbol{x}_i^{\text{ref}}, t_i, c), \tag{9}$$

for $i = 0, \ldots, K-1$. Applying the same error-accumulation argument to the discretized trajectories yields the following practical objective:

$$\mathcal{L}_{\text{CTS}}^{(K)}(\boldsymbol{\theta}) = \mathbb{E}\left[\sum_{i=0}^{K-1} \alpha_i \left\|\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_i^{\boldsymbol{\theta}}, t_i, c) - \boldsymbol{v}_{\text{ref}}(\boldsymbol{x}_i^{\text{ref}}, t_i, c)\right\|^2\right], \tag{10}$$

where $\alpha_i = [K(K+1) - i(i+1)]/2$ for $i = 0, \ldots, K-1$ is the trajectory-derived discrete weight. It reflects how the velocity mismatch at step i accumulates over the subsequent trajectory. We present the weights in unnormalized form to make their relative weighting explicit and normalize them to sum to one in implementation. See Appendix B for the full derivation.

When the starting state is sampled from the current student's rollout, the resulting formulation defines a family of OPD objectives parameterized by the supervision step K. For $K = 1$, we have $\alpha_0 = 1$, and the objective reduces to velocity matching at a single student-visited state. For $K > 1$, supervision extends to multiple subsequent states along the student and reference trajectories. Thus, K explicitly controls how far supervision extends along the trajectory, and we study its effect empirically in Section 4.3. When the starting state is constructed from an off-policy data trajectory, the same formulation yields the corresponding off-policy supervision objective.

4 Experiments

4.1 Experimental Setup

On-policy setting. We evaluate FlowCTS in a multi-reference OPD setting based on SD3.5-Medium (Esser et al., 2024). We consider GenEval, OCR, and PickScore as target capabilities and use the corresponding officially released task-specific Flow-GRPO checkpoints as references (Liu et al., 2025), without additional models. For each mini-batch, prompts are sampled from a target-specific prompt set and routed to the corresponding reference model, while student-visited states are generated using the SDE rollout procedure adopted in Flow-GRPO. All OPD variants share the same prompt mixture, rollout procedure, reference routing, optimization hyperparameters, and inference configuration. We compare SDE and ODE rollouts in Appendix C.3

Off-policy SFT setting. We further evaluate FlowCTS in an off-policy setting. Starting states and target trajectories are constructed from 50K image–text pairs sampled from Fine-T2I (Ma et al., 2026b). We use SD3.5-Medium as the base model and train Vanilla SFT and all off-policy variants with LoRA under same optimization settings.

Baselines. For the on-policy setting, we report the pretrained SD3.5-Medium and compare FlowCTS-OPD against three groups of competitive baselines: (1) *Single-task references*, including the task-specific Flow-GRPO models specialized for GenEval, OCR, and PickScore; (2) *Multi-objective RL*, represented by Flow-GRPO-Mix, which jointly optimizes the three target capabilities; and (3) *KL-based OPD*, implemented as Vanilla OPD following the transition-kernel KL objective of Flow-OPD (Fang et al., 2026). For the off-policy setting, Vanilla SFT with single-state velocity supervision serves as the baseline.

Evaluation. Following the evaluation protocol of Flow-GRPO, we evaluate in-domain performance on GenEval, OCR, and PickScore. We further assess out-of-distribution compositional generalization on T2I-CompBench (Huang et al., 2023), and evaluate general image quality and human preference on DrawBench (Saharia et al., 2022) using ImageReward, Aesthetic Score, UnifiedReward, and HPS-v2.1. Full training and evaluation details are provided in Appendices C and D.

Table 1: Main results for multi-reference on-policy distillation and its off-policy supervision extension. Higher values are better for all metrics. Bold and underlined values indicate the best and second-best results within each supervision setting, respectively. The Overall values are computed by averaging three 0-1 normalized metrics

Method	GenEval ↑	OCR ↑	PickScore ↑	Overall ↑
I. On-policy multi-objective alignment				
Pretrained model				
SD3.5-Medium	0.63	0.59	21.72	0.685
Reinforcement-learning baselines				
FlowGRPO (GenEval)	<u>0.93</u>	0.65	21.60	0.804
FlowGRPO (OCR)	0.66	**0.92**	21.74	0.805
FlowGRPO (PickScore)	0.51	0.70	**23.25**	0.701
FlowGRPO (Mixed rewards)	0.73	0.83	21.84	0.800
KL-based on-policy distillation				
Vanilla OPD w/ KL ($K = 1$)	0.90	<u>0.90</u>	22.75	0.890
One-step velocity supervision				
FLOWCTS-OPD ($K = 1$)	<u>0.93</u>	**0.92**	<u>23.06</u>	<u>0.912</u>
Multi-step trajectory supervision				
FlowCTS-OPD ($K = 2$)	**0.94**	**0.92**	22.96	**0.914**
FlowCTS-OPD ($K = 3$)	<u>0.93</u>	**0.92**	22.60	0.906
II. Off-policy trajectory supervision				
Single-step supervised fine-tuning				
Vanilla SFT	<u>0.71</u>	0.70	<u>21.68</u>	0.747
Multi-step trajectory supervision				
FlowCTS-SFT ($K = 2$)	**0.72**	0.73	21.66	<u>0.761</u>
FlowCTS-SFT ($K = 3$)	<u>0.71</u>	**0.75**	**21.71**	**0.763**
FlowCTS-SFT ($K = 4$)	0.70	<u>0.74</u>	21.63	0.756

4.2 MAIN RESULTS

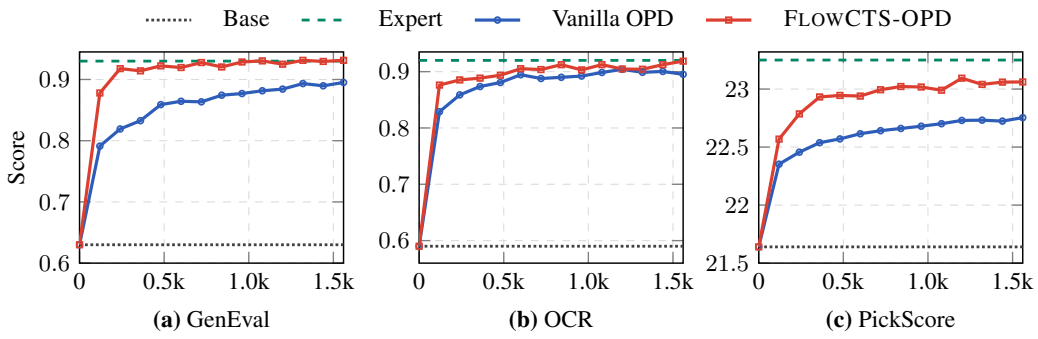
Table 1 reports the main results under the on-policy and off-policy settings, we first examine multi-objective alignment in the on-policy setting. The task-specific Flow-GRPO models perform strongly on their respective targets but exhibit substantial trade-offs on the others. Flow-GRPO-Mix improves the balance across objectives, yet fails to match the specialized capabilities of individual experts. In contrast, FLOWCTS-OPD achieves balanced performance across all three objectives, matching the GenEval and OCR experts while retaining competitive PickScore performance. These results demonstrate the effectiveness of the trajectory-derived formulation for multi-reference OPD.

To isolate the effect of the trajectory-derived objective, we compare FLOWCTS-OPD with Vanilla OPD within the same multi-reference on-policy pipeline. FLOWCTS-OPD improves GenEval from 0.90 to 0.93, OCR from 0.90 to 0.92, and PickScore from 22.75 to 23.06. Figure 3 further shows that FLOWCTS-OPD converges faster and reaches higher performance across all three metrics, approaching the corresponding GenEval and OCR expert levels substantially earlier than Vanilla OPD. Because the two methods differ only in their supervision objectives, these controlled results suggest that the trajectory-derived velocity-matching objective is more effective than Vanilla OPD.

The off-policy results provide complementary evidence under a different starting-state distribution. Compared with Vanilla SFT, the FlowCTS-SFT objectives mainly improve OCR while maintaining comparable GenEval and PickScore performance, suggesting that the same trajectory-derived formulation also provides useful supervision in the off-policy setting. The effect of the supervision step in both settings is analyzed in Section 4.3.

4.3 EFFECT OF THE SUPERVISION STEP IN OPD

We next study how far reference supervision within FLOWCTS-OPD should extend beyond a student-visited state. As shown in Table 1, we compare different supervision steps under both



(a) GenEval **(b)** OCR **(c)** PickScore

Figure 3: Training dynamics of Vanilla OPD and FlowCTS-OPD. Dotted and dashed lines indicate base-model and expert performance, respectively; blue and red curves denote Vanilla OPD and FlowCTS-OPD.

on-policy and off-policy settings. The step K ranges from velocity matching at the current state when $K = 1$ to supervision over a short student–reference trajectory segment when $K > 1$. The single-state setting provides a direct comparison with transition-kernel KL, while Table 1 and the supervision-budget control in Appendix E examine the additional effects of multi-step supervision.

Under the on-policy setting, extending the supervision step from $K = 1$ to $K = 2$ slightly improves GenEval from 0.93 to 0.94, yielding performance beyond that of the reference, while preserving the OCR score with only a minor reduction in PickScore. Extending the step further to $K = 3$ provides no additional benefit and leads to a larger drop in PickScore. These results suggest that the supervision step should be chosen carefully: short-steps objectives already capture the main training signal from the trajectory, while further extending the step leads to inconsistent improvements across metrics. In the small-scale budget control in Appendix E, we compare K = 2 trained for N steps with K = 1 trained for 2N steps, matching the number of supervised state evaluations. K = 2 achieves a higher GenEval score and the same OCR score.

The off-policy setting shows a similar pattern. Compared with Vanilla SFT, extending the supervision step to $K = 2$ and $K = 3$ improves the overall score, with the most pronounced gains observed on OCR, which increases from 0.70 to 0.73 and 0.75, respectively. However, increasing it further to $K = 4$ reduces all three metrics relative to $K = 3$, indicating that the gains from extending the supervision steps are not sustained at larger K.

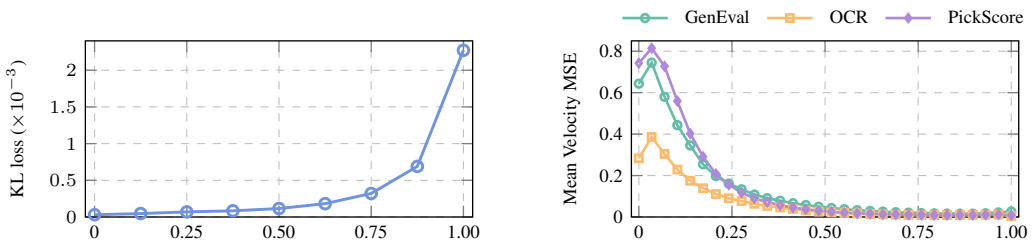
We attribute this non-monotonic behavior to a trade-off between richer trajectory supervision and increased optimization difficulty. A larger K provides reference signals at more future states, promoting structurally consistent evolution and improving compositional and spatial correctness. However, longer student–reference rollouts amplify the effect of early prediction errors, making later supervision harder to optimize. Stronger cross-step constraints may also suppress fine-grained details, weakening perceptual preference. Thus, the supervision step should be treated as an explicit design choice rather than simply maximized.

4.4 TEMPORAL MISALLOCATION OF KL SUPERVISION

We next analyze why the FLOWCTS-OPD optimizes more effectively than Vanilla OPD. In Vanilla OPD, the deterministic ODE dynamics are first reformulated as stochastic SDE transitions, and the KL divergence between the resulting transition kernels reduces to velocity regression with a time-dependent weight, where $w_{\mathrm{KL}}(t)$ is induced by the auxiliary SDE transition variance. We provide the full derivation of this KL-induced weighting in Appendix F.

$$D_{\mathrm{KL}}\left(\pi_\theta(\cdot \mid x_t) \| \pi_{\mathrm{ref}}(\cdot \mid x_t)\right) = w_{\mathrm{KL}}(t)\, \| v_\theta(x_t, t, c) - v_{\mathrm{ref}}(x_t, t, c) \|_2^2 , \tag{11}$$

Figure 4 reveals a clear mismatch in temporal signal allocation. The KL loss is strongly concentrated near the image endpoint, whereas the largest velocity-field differences between the pretrained model and the three reference experts consistently occur during early denoising. This observation is consistent with TempFlow-GRPO (He et al., 2025), which shows that perturbations during early structural stages have a substantially greater influence on the final generation outcome than those during late refinement stages. We further bias $K = 1$ supervision toward early, middle, or late



(a) Per-step KL loss

(b) Velocity MSE gap

Figure 4: Temporal signal allocation in KL-based OPD. Both horizontal axes denote normalized denoising progress from noise to image. The two quantities are computed from the same student-visited states and training observations. (a) Effective per-step KL loss: $w_{\mathrm{KL}}(t)\|v_\theta(x_t,t,c) - v_{\mathrm{ref}}(x_t,t,c)\|_2^2$. (b) Unweighted velocity gap: $\|v_\theta(x_t,t,c) - v_{\mathrm{ref}}(x_t,t,c)\|_2^2$. The KL objective concentrates its effective supervision near the image endpoint, while the underlying student–reference velocity mismatch is largest during earlier denoising stages.

Table 2: **Results on T2I-CompBench.** The best and second-best results within the OPD group are shown in bold and underlined, respectively.

Model	Color	Shape	Texture	Complex	3D-Spatial	Numeracy	Non-Spatial
Pretrained model							
SD3.5-M	0.7994	0.5669	0.7338	0.3800	0.3739	0.5927	0.3146
Reinforcement learning							
GRPO-Mix	0.7966	0.5803	0.7392	0.3677	0.3681	0.6388	0.3130
On-policy distillation							
Vanilla OPD	**0.8350**	<u>0.6211</u>	**0.7508**	<u>0.3834</u>	<u>0.4411</u>	<u>0.6840</u>	**0.3135**
FLOWCTS-OPD	<u>0.8295</u>	**0.6285**	<u>0.7414</u>	**0.3869**	**0.4473**	**0.6939**	<u>0.3096</u>

trajectory regions, where earlier supervision consistently yields better GenEval performance; see Appendix G.

Therefore, the SDE-induced KL weight emphasizes late timesteps while underweighting the stages where the reference models introduce their largest corrections. In contrast, our $K = 1$ objective directly matches velocities at student-visited states without this temporal weighting. This difference provides an explanation for its faster convergence and stronger final performance in Figure 3.

4.5 Capability and Generalization Analysis

We further examine whether the FLOWCTS-OPD objective generalizes beyond the three training metrics. As shown in Table 3, FLOWCTS-OPD consistently outperforms Vanilla OPD across ImageReward (Xu et al., 2023), Aesthetic Score (Schuhmann et al., 2022), UnifiedReward (Wang et al., 2025), and HPS-v2.1 (Wu et al., 2023). These results show that its improvements are not limited to the training targets, but extend to broader measures of visual quality and human preference.

Table 2 further evaluates out-of-distribution compositional capabilities on T2I-CompBench. Compared with Vanilla OPD, FLOWCTS-OPD performs better on shape, spatial relations, and numeracy, while the differences on color and texture remain limited. This pattern is consistent with the temporal analysis in Section 4.4: since early denoising stages play a greater role in establishing global structure, correcting the under-allocation of supervision at these stages is expected to benefit structural and relational capabilities more than visual attributes. These results suggest that the allocation of supervision across timesteps plays an important role in effective post-training.

5 Related Work

Post-training for Flow Models. Diffusion post-training includes policy-gradient methods such as DDPO (Black et al., 2024) and DPOK (Fan et al., 2023), differentiable reward optimization such as ReFL (Xu et al., 2023) and DRaFT (Clark et al., 2024), and offline preference learning such as

Table 3: Comparison on general image quality and human preference metrics. The best and second-best results within the OPD group are shown in bold and underlined, respectively. ‡ Evaluated at a resolution of 1024×1024.

Model	ImageReward	Aesthetic	UnifiedReward	HPS-v2.1
Pretrained models				
SD-XL‡	0.76	5.60	2.93	0.28
SD3.5-L‡	0.96	5.50	3.25	0.29
FLUX.1-Dev	0.96	5.71	3.27	0.27
SD3.5-M	0.82	5.37	3.00	0.28
On-policy distillation				
Vanilla OPD	<u>1.21</u>	<u>5.46</u>	<u>3.28</u>	<u>0.30</u>
FLOWCTS-OPD	**1.27**	**5.64**	**3.32**	**0.31**

Diffusion-DPO (Wallace et al., 2024). Recent methods extend online optimization to flow models: DanceGRPO unifies GRPO across diffusion and rectified-flow generators (Xue et al., 2025), Flow-GRPO uses stochastic rollouts (Liu et al., 2025), TempFlow-GRPO introduces temporal credit assignment (He et al., 2025), and DiffusionNFT incorporates reward feedback into forward-process flow matching (Zheng et al., 2025). These methods adapt generative models through reward or preference feedback. Complementary to these approaches, we study on-policy distillation from reference models, focusing on how to construct dense supervision along student-generated trajectories.

On-Policy Distillation for Flow Models. On-policy distillation was initially developed for autoregressive language models, where the student generates its own sequences and receives dense reference supervision at the visited prefixes. MiniLLM optimizes a reverse-KL objective on student rollouts, while GKD generalizes this paradigm to different divergence objectives and sampling strategies (Gu et al., 2024; Agarwal et al., 2024).Recent studies further use OPD to consolidate capabilities from multiple specialized references into a unified student (Yang et al., 2026; Ma et al., 2026a; Zeng et al., 2026; Yang et al., 2025; Xiao et al., 2026; Li et al., 2026a). Recent works extend this idea to flow models. Flow-OPD and DiffusionOPD construct per-step KL objectives on student-generated states (Fang et al., 2026; Li et al., 2026b). Concurrent to our work, DanceOPD focuses on capability composition with one low-noise velocity query per student rollout (Zhou et al., 2026). FlowCTS instead derives a family of objectives from continuous student–reference trajectory differences. It further analyzes the temporal misallocation of transition-kernel KL relative to reference-student velocity errors, studies different supervision steps, and extends the same formulation to off-policy supervision.

Trajectory and Distribution Distillation. Trajectory-based distillation primarily aims to accelerate sampling by compressing multi-step generation into few-step students, through matching intermediate transitions or enforcing cross-timestep consistency (Salimans & Ho, 2022; Liu et al., 2024; Song et al., 2023; Kim et al., 2024; Boffi et al., 2024). DMD instead aligns student and reference distributions through score-based distribution matching, without requiring trajectory-wise correspondence (Yin et al., 2024a;b). Although FLOWCTS-OPD adopts trajectory-based supervision, it shares DMD's high-level goal of reference alignment. We further discuss their connection through a velocity-residual interpretation in Appendix H.

6 CONCLUSION

We study on-policy distillation for flow matching models and derive FLOWCTS-OPD from continuous student and reference trajectories initialized at shared student-visited states. The resulting objectives cover both local velocity matching and short-steps supervision. In a controlled multi-reference setting, FLOWCTS-OPD converges faster and outperforms transition-kernel KL. Longer steps yield non-monotonic gains, reflecting a trade-off between additional trajectory information and optimization difficulty. Temporal analysis further reveals that KL supervision is concentrated at late denoising stages, despite larger reference-field corrections occurring earlier. Together, these results identify temporal weighting and supervision step as two key design choices for OPD in continuous-time generative models.

AI USE STATEMENT

We used generative AI tools to assist with language polishing and to help identify potentially relevant literature. All references suggested by AI tools were independently verified by the authors. The AI tools were not used to develop the research ideas, methodology, theoretical results, or experimental findings. All AI-assisted text was reviewed and revised by the authors, who take full responsibility for the final content of this paper.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pp. 21246–21263, 2024.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pp. 4965–4987, 2024.

Nicholas M Boffi, Michael S Albergo, and Eric Vanden-Eijnden. Flow map matching with stochastic interpolants: A mathematical framework for consistency models. *arXiv preprint arXiv:2406.07507*, 2024.

Kevin Clark, Paul Vicol, Kevin Swersky, and David Fleet. Directly fine-tuning diffusion models on differentiable rewards. In *International Conference on Learning Representations*, volume 2024, pp. 4793–4822, 2024.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in Neural Information Processing Systems*, 36:79858–79885, 2023.

Zhen Fang, Wenxuan Huang, Yu Zeng, Yiming Zhao, Shuang Chen, Kaituo Feng, Yunlong Lin, Lin Chen, Zehui Chen, Shaosheng Cao, et al. Flow-opd: On-policy distillation for flow matching models. *arXiv preprint arXiv:2605.08063*, 2026.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *International Conference on Learning Representations*, volume 2025, pp. 34668–34684, 2025.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling, 2025. URL `https://arxiv.org/abs/2505.13447`.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. In *International Conference on Learning Representations*, volume 2024, pp. 32694–32717, 2024.

Xiaoxuan He, Siming Fu, Yuke Zhao, Wanli Li, Jian Yang, Dacheng Yin, Fengyun Rao, and Bo Zhang. Tempflow-grpo: When timing matters for grpo in flow models. *arXiv preprint arXiv:2508.04324*, 2025.

Wenjin Hou, Shangpin Peng, Weinong Wang, Zheng Ruan, Yue Zhang, Zhenglin Zhou, Mingqi Gao, Yifei Chen, Kaiqi Wang, Hongming Yang, et al. Uni-opd: Unifying on-policy distillation with a dual-perspective recipe. *arXiv preprint arXiv:2605.03677*, 2026.

Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. T2i-compbench: A comprehensive benchmark for open-world compositional text-to-image generation. *Advances in Neural Information Processing Systems*, 36:78723–78747, 2023.

Dongjun Kim, Chieh-Hsin Lai, WeiHsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. In *International Conference on Learning Representations*, volume 2024, pp. 44493–44525, 2024.

Nikita Kornilov, Petr Mokrov, Alexander Gasnikov, and Alexander Korotin. Optimal flow matching: Learning straight trajectories in just one step. *arXiv preprint arXiv:2403.13117*, 2024.

Fengxiang Li, Han Zhang, Haoyang Huang, Jinghui Wang, Jinhua Hao, Kun Yuan, Mengtong Li, Minglei Zhang, Pengcheng Xu, Wenhao Zhuang, et al. Kat-coder-v2 technical report. *arXiv preprint arXiv:2603.27703*, 2026a.

Quanhao Li, Junqiu Yu, Kaixun Jiang, Yujie Wei, Zhen Xing, Pandeng Li, Ruihang Chu, Shiwei Zhang, Yu Liu, and Zuxuan Wu. Diffusionopd: A unified perspective of on-policy distillation in diffusion models. *arXiv preprint arXiv:2605.15055*, 2026b.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huanang Gao, Wenkai Yang, Zhiyuan Liu, et al. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv preprint arXiv:2604.13016*, 2026c.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online rl, 2025. URL https://arxiv.org/abs/2505.05470.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, et al. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. In *The Twelfth International Conference on Learning Representations*, 2024.

Kevin Lu and Thinking Machines Lab. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20251026. https://thinkingmachines.ai/blog/on-policy-distillation.

Wenhan Ma, Jianyu Wei, Liang Zhao, Hailin Zhang, Bangjun Xiao, Lei Li, Qibin Yang, Bofei Gao, Yudong Wang, Rang Li, et al. Mopd: Multi-teacher on-policy distillation for capability integration in llm post-training. *arXiv preprint arXiv:2606.30406*, 2026a.

Xu Ma, Yitian Zhang, Qihua Dong, and Yun Fu. Fine-t2i: An open, large-scale, and diverse dataset for high-quality t2i fine-tuning. *arXiv preprint arXiv:2602.09439*, 2026b.

Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 661–668. JMLR Workshop and Conference Proceedings, 2010.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in neural information processing systems*, 35:25278–25294, 2022.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *arXiv preprint arXiv:2302.00482*, 2023.

Bram Wallace, Meihua Dang, Rafael Rafailov, Linqi Zhou, Aaron Lou, Senthil Purushwalkam, Stefano Ermon, Caiming Xiong, Shafiq Joty, and Nikhil Naik. Diffusion model alignment using direct preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8228–8238, 2024.

Yibin Wang, Yuhang Zang, Hao Li, Cheng Jin, and Jiaqi Wang. Unified reward model for multimodal understanding and generation. *arXiv preprint arXiv:2503.05236*, 2025.

Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. F-divergence minimization for sequence-level knowledge distillation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10817–10834, 2023.

Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. *arXiv preprint arXiv:2306.09341*, 2023.

Bangjun Xiao, Bingquan Xia, Bo Yang, Bofei Gao, Bowen Shen, Chen Zhang, Chenhong He, Chiheng Lou, Fuli Luo, Gang Wang, et al. Mimo-v2-flash technical report. *arXiv preprint arXiv:2601.02780*, 2026.

Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36:15903–15935, 2023.

Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, et al. Dancegrpo: Unleashing grpo on visual generation. *arXiv preprint arXiv:2505.07818*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Wenkai Yang, Weijie Liu, Ruobing Xie, Kai Yang, Saiyong Yang, and Yankai Lin. Learning beyond teacher: Generalized on-policy distillation with reward extrapolation. *arXiv preprint arXiv:2602.12125*, 2026.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in neural information processing systems*, 37:47455–47487, 2024a.

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

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. Glm-5: from vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. Diffusionnft: Online diffusion reinforcement with forward process. *arXiv preprint arXiv:2509.16117*, 2025.

Wei Zhou, Xiongwei Zhu, Zelin Xu, Bo Dong, Lixue Gong, Yongyuan Liang, Meng Chu, Leigang Qu, Lingdong Kong, Wei Liu, et al. Danceopd: On-policy generative field distillation. *arXiv preprint arXiv:2606.27377*, 2026.

A Derivation of the Velocity Upper Bound

We provide the derivation of the velocity upper bound used in Section 3.2. Consider a reverse-time segment $[t_e, t_s]$ with $0 \le t_e < t_s \le 1$. The student and reference trajectories start from the same state $\boldsymbol{x}_{t_s}$ and are defined as

$$\boldsymbol{x}_t^{\boldsymbol{\theta}} = \boldsymbol{x}_{t_s} - \int_t^{t_s} \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_r^{\boldsymbol{\theta}}, r, c)\, dr, \qquad \boldsymbol{x}_t^{\mathrm{ref}} = \boldsymbol{x}_{t_s} - \int_t^{t_s} \boldsymbol{v}_{\mathrm{ref}}(\boldsymbol{x}_r^{\mathrm{ref}}, r, c)\, dr. \tag{12}$$

Subtracting the two trajectories gives

$$\boldsymbol{x}_t^{\boldsymbol{\theta}} - \boldsymbol{x}_t^{\mathrm{ref}} = -\int_t^{t_s} \left[\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_r^{\boldsymbol{\theta}}, r, c) - \boldsymbol{v}_{\mathrm{ref}}(\boldsymbol{x}_r^{\mathrm{ref}}, r, c)\right] dr. \tag{13}$$

For simplicity, define the velocity mismatch along the two trajectories as

$$\Delta \boldsymbol{v}_r = \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_r^{\boldsymbol{\theta}}, r, c) - \boldsymbol{v}_{\mathrm{ref}}(\boldsymbol{x}_r^{\mathrm{ref}}, r, c). \tag{14}$$

Then

$$\boldsymbol{x}_t^{\boldsymbol{\theta}} - \boldsymbol{x}_t^{\mathrm{ref}} = -\int_t^{t_s} \Delta \boldsymbol{v}_r\, dr. \tag{15}$$

The continuous trajectory objective is

$$\mathcal{L}_{\mathrm{CTS}}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}_{t_s}, c}\left[\int_{t_e}^{t_s} \left\|\boldsymbol{x}_t^{\boldsymbol{\theta}} - \boldsymbol{x}_t^{\mathrm{ref}}\right\|^2 dt\right]. \tag{16}$$

Substituting the integral form of the state discrepancy gives

$$\begin{aligned}
\mathcal{L}_{\mathrm{CTS}}(\boldsymbol{\theta}) &= \mathbb{E}_{\boldsymbol{x}_{t_s}, c}\left[\int_{t_e}^{t_s} \left\|\int_t^{t_s} \Delta \boldsymbol{v}_r\, dr\right\|^2 dt\right] \\
&\le \mathbb{E}_{\boldsymbol{x}_{t_s}, c}\left[\int_{t_e}^{t_s} (t_s - t) \int_t^{t_s} \|\Delta \boldsymbol{v}_r\|^2\, dr\, dt\right] \\
&= \mathbb{E}_{\boldsymbol{x}_{t_s}, c}\left[\int_{t_e}^{t_s} \left(\int_{t_e}^{r} (t_s - t)\, dt\right) \|\Delta \boldsymbol{v}_r\|^2\, dr\right].
\end{aligned} \tag{17}$$

The inner trajectory integral is first bounded by Cauchy–Schwarz, after which the resulting double integral is reordered. The trajectory-derived weight is

$$w(r) = \int_{t_e}^{r} (t_s - t) dt = \frac{(t_s - t_e)^2 - (t_s - r)^2}{2}. \tag{18}$$

Substituting $\Delta \boldsymbol{v}_r$ back gives the velocity upper-bound objective

$$\mathcal{L}_{\mathrm{CTS}}(\boldsymbol{\theta}) \le \mathcal{L}_{\mathrm{VUB}}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x}_{t_s}, c}\left[\int_{t_e}^{t_s} w(r) \left\|\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_r^{\boldsymbol{\theta}}, r, c) - \boldsymbol{v}_{\mathrm{ref}}(\boldsymbol{x}_r^{\mathrm{ref}}, r, c)\right\|^2 dr\right]. \tag{19}$$

B Derivation of the Discrete Training Objective

We derive the discrete form of the trajectory objective. We discretize the reverse-time segment $[t_e, t_s]$ into K uniform Euler steps. Let

$$h = \frac{t_s - t_e}{K}, \qquad t_i = t_s - ih, \qquad i = 0, \dots, K. \tag{20}$$

The student and reference rollouts start from the same state:

$$\boldsymbol{x}_0^{\boldsymbol{\theta}} = \boldsymbol{x}_0^{\mathrm{ref}} = \boldsymbol{x}_{t_s}. \tag{21}$$

Using Euler discretization, the two rollouts are

$$\boldsymbol{x}_{i+1}^{\boldsymbol{\theta}} = \boldsymbol{x}_i^{\boldsymbol{\theta}} - h\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_i^{\boldsymbol{\theta}}, t_i, c), \qquad \boldsymbol{x}_{i+1}^{\mathrm{ref}} = \boldsymbol{x}_i^{\mathrm{ref}} - h\boldsymbol{v}_{\mathrm{ref}}(\boldsymbol{x}_i^{\mathrm{ref}}, t_i, c), \tag{22}$$

for $i = 0, \ldots, K-1$.

Define the velocity mismatch at step i as

$$\delta_i = \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_i^{\boldsymbol{\theta}}, t_i, c) - \boldsymbol{v}_{\mathrm{ref}}(\boldsymbol{x}_i^{\mathrm{ref}}, t_i, c). \tag{23}$$

Then the state discrepancy follows

$$\boldsymbol{x}_{i+1}^{\boldsymbol{\theta}} - \boldsymbol{x}_{i+1}^{\mathrm{ref}} = \boldsymbol{x}_i^{\boldsymbol{\theta}} - \boldsymbol{x}_i^{\mathrm{ref}} - h\delta_i. \tag{24}$$

Since $\boldsymbol{x}_0^{\boldsymbol{\theta}} - \boldsymbol{x}_0^{\mathrm{ref}} = \mathbf{0}$, we have

$$\boldsymbol{x}_i^{\boldsymbol{\theta}} - \boldsymbol{x}_i^{\mathrm{ref}} = -h \sum_{j=0}^{i-1} \delta_j, \qquad i = 1, \ldots, K. \tag{25}$$

A discrete state-space trajectory loss can be written as

$$\mathcal{L}_{\mathrm{state,disc}} = h \sum_{i=1}^{K} \left\| \boldsymbol{x}_i^{\boldsymbol{\theta}} - \boldsymbol{x}_i^{\mathrm{ref}} \right\|^2. \tag{26}$$

Substituting the accumulated error form gives

$$\mathcal{L}_{\mathrm{state,disc}} = h^3 \sum_{i=1}^{K} \left\| \sum_{j=0}^{i-1} \delta_j \right\|^2. \tag{27}$$

By the discrete Cauchy–Schwarz inequality,

$$\left\| \sum_{j=0}^{i-1} \delta_j \right\|^2 \leq i \sum_{j=0}^{i-1} \|\delta_j\|^2. \tag{28}$$

Therefore,

$$\mathcal{L}_{\mathrm{state,disc}} \leq h^3 \sum_{i=1}^{K} i \sum_{j=0}^{i-1} \|\delta_j\|^2. \tag{29}$$

Exchanging the summation order, we obtain

$$\mathcal{L}_{\mathrm{state,disc}} \leq h^3 \sum_{j=0}^{K-1} \left(\sum_{i=j+1}^{K} i \right) \|\delta_j\|^2. \tag{30}$$

The inner sum has the closed form

$$\sum_{i=j+1}^{K} i = \frac{K(K+1) - j(j+1)}{2}. \tag{31}$$

Thus,

$$\mathcal{L}_{\mathrm{state,disc}} \leq h^3 \sum_{j=0}^{K-1} \frac{K(K+1) - j(j+1)}{2} \|\delta_j\|^2. \tag{32}$$

The discrete upper bound determines the relative contribution of the velocity mismatch at each supervised state. For a fixed supervision step K and trajectory segment, h^3 is a positive constant independent of $\boldsymbol{\theta}$. It therefore does not affect the minimizer or the relative weighting across timesteps, and can be absorbed into the overall loss coefficient. We define the unnormalized trajectory-derived weights as

$$\alpha_j = \frac{K(K+1) - j(j+1)}{2}, \qquad j = 0, \ldots, K-1. \tag{33}$$

For training, we normalize these coefficients as

$$\bar{\alpha}_j = \frac{\alpha_j}{\sum_{i=0}^{K-1} \alpha_i}, \qquad \sum_{i=0}^{K-1} \alpha_i = \frac{K(K+1)(2K+1)}{6}, \tag{34}$$

which preserves the trajectory-derived temporal profile while keeping the overall loss scale comparable across different values of K. The practical objective is therefore

$$\mathcal{L}_{\mathrm{CTS}}^{(K)}(\boldsymbol{\theta}) = \mathbb{E}\left[\sum_{j=0}^{K-1} \bar{\alpha}_j \left\| \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_j^{\boldsymbol{\theta}}, t_j, c) - \boldsymbol{v}_{\mathrm{ref}}(\boldsymbol{x}_j^{\mathrm{ref}}, t_j, c) \right\|_2^2\right]. \tag{35}$$

When $K = 1$, $\bar{\alpha}_0 = 1$, and the objective reduces to velocity matching at the segment starting state.

C Training Details

C.1 On-Policy multi-reference Distillation

Implementation details. We use Stable Diffusion 3.5 Medium as the base model and initialize the student from the averaged multi-task Flow-GRPO LoRA checkpoint (Liu et al., 2025). All OPD variants use LoRA with rank 32 and alpha 64, and are trained on four H200 GPUs.

Student rollouts are generated at a resolution of 512×512 using 10 sampling steps and a classifier-free guidance scale of 4.5. A timestep fraction of 0.99 is used, resulting in nine denoising transitions for optimization. The sampling batch size is 16 per GPU, with four sampling batches per epoch and gradient accumulation over two steps. We optimize the student with AdamW using a learning rate of 1×10^{-4}, weight decay 1×10^{-4}, and a maximum gradient norm of 1.0. We maintain an exponential moving average of the trainable parameters with decay 0.9, updated every eight optimization steps.

Prompt routing and reference models. We use the OCR, GenEval, and PickScore prompt sets released with Flow-GRPO. Each mini-batch is drawn from one prompt set and routed to its corresponding task-specific reference model: the text-rendering, GenEval, or PickScore Flow-GRPO expert. The three datasets are sampled alternately with a cycle ratio of 1:3:1 for OCR, GenEval, and PickScore, respectively. The Vanilla OPD (Fang et al., 2026) additionally employs MAR, which relies on a separately trained frozen task-agnostic auxiliary teacher. We do not use this auxiliary model. For a controlled comparison, we disable MAR and use only the three task-specific Flow-GRPO experts as reference models for both vanilla OPD and FlowCTS-OPD.

OPD objectives. Vanilla OPD and FlowCTS-OPD use the same initialization, prompt mixture, student rollouts, reference routing, batch construction, and optimization hyperparameters. Vanilla OPD uses the step-wise transition-kernel KL as the distillation objective.

For FlowCTS-OPD, we replace the transition-kernel policy objective with the trajectory-derived velocity objective in Eq. 10. We treat the reference velocity predictions as stop-gradient targets in the per-step losses. When $K = 1$, the objective reduces to mean-squared velocity matching between the student and the routed reference model at a student-visited state. For $K > 1$, the student and reference models are rolled out over the subsequent trajectory segment, and the per-step losses are weighted according to Eq. 33.

C.2 off-policy Extension

For the off-policy experiments, we construct training trajectories from 50K image–text pairs sampled from Fine-T2I (Ma et al., 2026b) and use SD3.5-Medium as the base model. As in the on-policy setting, the student and reference trajectories start from a shared state. Here, the starting state is constructed from an offline image–noise pair, and the continuation of the resulting straight-line interpolation serves as the reference trajectory. Vanilla SFT and all FlowCTS-SFT variants are trained with LoRA under the same data and optimization configuration. Vanilla SFT uses single-state velocity supervision, while FlowCTS-SFT supervises $K \in \{2, 3, 4\}$ states using the trajectory-derived weights. All off-policy experiments are conducted on four A800 GPUs.

C.3 Effect of SDE and ODE Sampling

We compare SDE and ODE sampling for constructing the student-visited states during on-policy distillation. The two variants use the same velocity objective and training configuration, differing only in the rollout process used to obtain on-policy states.

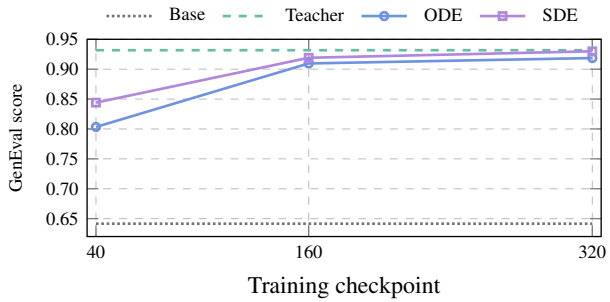


Figure 5: Comparison of SDE and ODE sampling for constructing student-visited states during on-policy distillation. Both variants use the same velocity-matching objective and training configuration, differing only in the rollout process used to generate on-policy states. SDE sampling consistently achieves higher GenEval performance across training checkpoints, with a larger advantage during early optimization. The horizontal dashed lines denote the pretrained base model and the corresponding task-specific reference.

Table 4: Training-budget control. The extended $K = 1$ run matches the number of supervised state evaluations used by $K = 2$.

Method	Training steps	GenEval ↑	OCR ↑	PickScore ↑
FLOWCTS-OPD ($K = 1$)	N	0.91	0.89	22.93
FLOWCTS-OPD ($K = 1$)	$2N$	0.92	0.91	23.01
FLOWCTS-OPD ($K = 2$)	N	0.93	0.91	22.80

As shown in Figure 5, SDE sampling consistently outperforms deterministic ODE sampling throughout training. The difference is most pronounced at the early checkpoint, where SDE improves GenEval from 0.80 to 0.84. The advantage remains at the end of training, with SDE reaching 0.93 compared with 0.92 for ODE sampling and approaching the corresponding reference performance. These results suggest that stochastic exploration improves the states encountered during on-policy distillation, particularly during early optimization.

D EVALUATION

Unless otherwise specified, we generate images at a resolution of 512×512 using 40 inference steps and a classifier-free guidance scale of 4.5. We evaluate each model using multiple random seeds and report the mean scores across seeds. All compared methods use identical prompts, generation settings, random seeds, and evaluator implementations. Models marked with ‡ in Table 3 are evaluated at 1024×1024.

Following Flow-GRPO (Liu et al., 2025), we evaluate GenEval, OCR, and PickScore on their corresponding test sets. Compositional generalization is evaluated on the validation split of T2I-CompBench (Huang et al., 2023), with ten images generated per prompt using the official evaluation implementation. General image quality and preference alignment are evaluated on DrawBench using ImageReward, Aesthetic Score, UnifiedReward, and HPS-v2.1, following the Flow-GRPO evaluation protocol. All compared methods use identical generation and evaluator settings.

E CONTROLLING FOR THE TRAINING BUDGET

Since FLOWCTS-OPD ($K = 2$) evaluates two supervised states per optimization step, we conduct a small-scale auxiliary study to efficiently control for the supervision budget. Specifically, we compare $K = 2$ trained for N steps with $K = 1$ trained for N and $2N$ steps, where the latter matches the total number of supervised state evaluations. All runs in this study use the same reduced training configuration and are intended for controlled comparison within Table 4.

As shown in Table 4, doubling the number of training steps for $K = 1$ produces modest improvements across all three metrics, but does not reproduce the behavior of $K = 2$. The latter

achieves a higher GenEval score and comparable OCR performance. This comparison holds the number of supervised state evaluations fixed while changing how they are allocated: K = 1 receives more single-state updates, whereas K = 2 supervises subsequent student–reference evolution within each update. The higher GenEval and lower PickScore of K = 2 follow the same pattern as Table 1, supporting a task-dependent effect of multi-step supervision under the matched supervision count. Meanwhile, qualitative inspection suggests that this stronger trajectory constraint may suppress some fine-grained details, resulting in slightly smoother images and a lower preference score. We therefore view the result as a trade-off introduced by the supervision span, rather than a uniform improvement obtained from a larger training budget.

F Derivation of the KL Temporal Weight

We provide the derivation of the time-dependent weight $w_{\mathrm{KL}}(t)$ used in Section 4.4. Following the auxiliary SDE formulation of Flow-GRPO, the deterministic flow dynamics are reformulated as

$$d\boldsymbol{x}_t = \left[\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c) + \frac{\sigma_t^2}{2t}\left(\boldsymbol{x}_t + (1-t)\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c)\right)\right] dt + \sigma_t d\boldsymbol{w}_t. \tag{36}$$

Applying Euler–Maruyama discretization with step size Δt, the student transition is an isotropic Gaussian,

$$\pi_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-\Delta t} \mid \boldsymbol{x}_t, c) = \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c), \sigma_t^2 \Delta t\, \boldsymbol{I}\right), \tag{37}$$

where

$$\boldsymbol{\mu}_{\boldsymbol{\theta}} = \boldsymbol{x}_t - \left[\boldsymbol{v}_{\boldsymbol{\theta}} + \frac{\sigma_t^2}{2t}\left(\boldsymbol{x}_t + (1-t)\boldsymbol{v}_{\boldsymbol{\theta}}\right)\right]\Delta t. \tag{38}$$

Similarly, the reference transition is

$$\pi_{\mathrm{ref}}(\boldsymbol{x}_{t-\Delta t} \mid \boldsymbol{x}_t, c) = \mathcal{N}\left(\boldsymbol{\mu}_{\mathrm{ref}}(\boldsymbol{x}_t, t, c), \sigma_t^2 \Delta t\, \boldsymbol{I}\right). \tag{39}$$

Since the two transition distributions share the same covariance, their KL divergence reduces to

$$D_{\mathrm{KL}}\left(\pi_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}_t, c) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid \boldsymbol{x}_t, c)\right) = \frac{\|\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_{\mathrm{ref}}\|_2^2}{2\sigma_t^2 \Delta t}. \tag{40}$$

Because both transition means are evaluated at the same state $\boldsymbol{x}_t$, their difference is

$$\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_{\mathrm{ref}} = -\Delta t\left(1 + \frac{\sigma_t^2(1-t)}{2t}\right)\left[\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c) - \boldsymbol{v}_{\mathrm{ref}}(\boldsymbol{x}_t, t, c)\right]. \tag{41}$$

Substituting this expression into Eq. (40) gives

$$\begin{aligned} &D_{\mathrm{KL}}\left(\pi_{\boldsymbol{\theta}}(\cdot \mid \boldsymbol{x}_t, c) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid \boldsymbol{x}_t, c)\right) \\ &= \frac{\Delta t}{2\sigma_t^2}\left(1 + \frac{\sigma_t^2(1-t)}{2t}\right)^2 \|\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c) - \boldsymbol{v}_{\mathrm{ref}}(\boldsymbol{x}_t, t, c)\|_2^2 \\ &= w_{\mathrm{KL}}(t)\, \|\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c) - \boldsymbol{v}_{\mathrm{ref}}(\boldsymbol{x}_t, t, c)\|_2^2, \end{aligned} \tag{42}$$

where

$$\boxed{w_{\mathrm{KL}}(t) = \frac{\Delta t}{2}\left(\frac{1}{\sigma_t} + \frac{\sigma_t(1-t)}{2t}\right)^2}. \tag{43}$$

G Effect of Supervision at Different Trajectory Regions

Section 4.4 reveals a clear temporal discrepancy between KL-based OPD and direct velocity supervision: the SDE-induced KL objective places more emphasis on later stages of generation, whereas the student–reference velocity discrepancy is more pronounced earlier. To examine whether the supervision location itself affects optimization, we conduct a controlled query-location ablation using FlowCTS-OPD with $K = 1$.

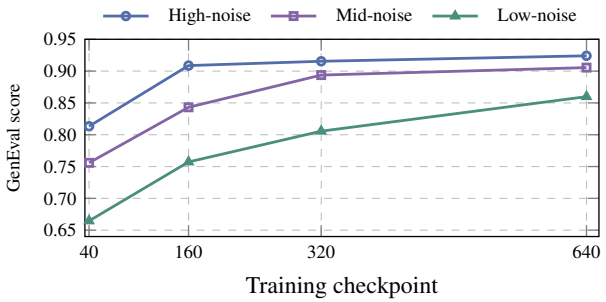


Figure 6: **Effect of supervision at different noise levels on GenEval.** High-noise-biased supervision consistently yields stronger performance throughout training, while low-noise-biased supervision converges substantially more slowly.

Setup. We keep the student initialization, reference model, optimization configuration, and single-state velocity-matching objective unchanged, and vary only the queried trajectory position. Specifically, we sample a normalized trajectory index $s \in [0, 1]$, where $s = 0$ and $s = 1$ correspond to the beginning and end of the student rollout, respectively. Since the rollout proceeds from high to low noise, earlier positions correspond to relatively higher-noise states. We use

$$s_{\text{early}} \sim \text{Beta}(2, 5), \qquad s_{\text{mid}} \sim \text{Beta}(5, 5), \qquad s_{\text{late}} \sim \text{Beta}(5, 2).$$

These distributions bias supervision toward different trajectory regions rather than defining explicit intervals in physical flow time.

Results. As shown in Fig. 6, earlier-trajectory supervision consistently achieves the strongest GenEval performance throughout training. At checkpoint 40, the early, middle, and late variants achieve 0.81, 0.76, and 0.66, respectively; at checkpoint 640, they reach 0.92, 0.91, and 0.86. The stable ordering across training indicates that moving supervision toward earlier trajectory states substantially improves both optimization speed and final performance.

Discussion. These results highlight the importance of earlier trajectory states for compositional text-to-image generation. At these relatively higher-noise stages, global structure and semantic composition are still being established, so deviations can propagate through a longer portion of the subsequent trajectory. This observation further supports the temporal analysis in Section 4.4: although the auxiliary SDE transition induces stronger KL weighting toward later stages, our controlled ablation shows that earlier supervision is more effective for improving GenEval in this setting. This preference is task dependent; tasks dominated by local appearance refinement or image editing may benefit more from later, low-noise states, whereas our experiments focus on multi-reference text-to-image generation, where global semantic and compositional decisions are central.

H Connection to Distribution Matching Distillation

Distribution Matching Distillation (DMD) (Yin et al., 2024b) and OPD share a high-level goal of aligning student-generated distributions with a reference model, but construct supervision differently. DMD minimizes an approximate reverse KL divergence and derives its gradient from the difference between the student and reference scores. In contrast, FlowCTS derives its supervision from continuous student–reference trajectory discrepancies. We discuss their connection at the level of local supervision.

From scores to velocities. For the rectified-flow interpolation $\boldsymbol{x}_t = (1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1$, the marginal score and velocity field satisfy (Liu et al., 2025)

$$s(\boldsymbol{x}_t, t) = -\frac{\boldsymbol{x}_t}{t} - \frac{1-t}{t}\boldsymbol{v}(\boldsymbol{x}_t, t). \tag{44}$$

Consequently, at the same state $\boldsymbol{x}_t$,

$$s_{\text{fake}} - s_{\text{ref}} = -\frac{1-t}{t}\left(\boldsymbol{v}_{\text{fake}} - \boldsymbol{v}_{\text{ref}}\right). \tag{45}$$

This reveals that the score discrepancy is proportional to the corresponding velocity discrepancy, with an additional noise-dependent scaling factor.

Connection to velocity regression. DMD computes its distribution-matching gradient as

$$\nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathrm{DMD}} \simeq \mathbb{E}\left[w_t \alpha_t (s_{\mathrm{fake}} - s_{\mathrm{ref}})^\top \frac{\partial G_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}} \right], \tag{46}$$

where $\boldsymbol{x} = G_{\boldsymbol{\theta}}(z)$. To normalize gradient magnitudes across noise levels, DMD introduces (Yin et al., 2024b)

$$w_t = \frac{\sigma_t^2}{\alpha_t} \frac{CS}{\|\mu_{\mathrm{ref}}(\boldsymbol{x}_t, t) - \boldsymbol{x}\|_1}, \tag{47}$$

where μ_{ref} denotes the denoised prediction of the reference model, and S is the number of spatial locations and C is the number of channel.

Substituting Eq. (45) into the DMD gradient yields

$$\boxed{\nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathrm{DMD}} \simeq -\mathbb{E}\left[w_t \alpha_t \frac{1-t}{t} (\boldsymbol{v}_{\mathrm{fake}} - \boldsymbol{v}_{\mathrm{ref}})^\top \frac{\partial G_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}} \right].} \tag{48}$$

Under the linear rectified-flow interpolation, $\alpha_t = 1 - t$ and $\sigma_t = t$. The factor σ_t^2/α_t in w_t cancels the explicit inverse-noise scaling introduced by the score parameterization, leaving an adaptively weighted velocity residual. The remaining normalization is sample-dependent and does not make all timesteps equally weighted.

In comparison, FlowCTS with $K = 1$ directly minimizes the velocity discrepancy, yielding

$$\boxed{\nabla_{\boldsymbol{\theta}} \mathcal{L}_{\mathrm{CTS}}^{(1)} = 2\mathbb{E}\left[(\boldsymbol{v}_{\boldsymbol{\theta}} - \boldsymbol{v}_{\mathrm{ref}})^\top \frac{\partial \boldsymbol{v}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}} \right].} \tag{49}$$

Both formulations involve a local velocity residual between a student-side model and the reference. In DMD, the one- or few-step generator does not generally provide the score of its own noised output distribution, so an auxiliary fake-score model is trained to estimate this quantity. FlowCTS retains the student's time-dependent velocity field and evaluates it directly at student-visited states. Under the assumption that this field approximates the marginal velocity associated with the same noising process, it can serve as a surrogate for the velocity implied by the fake-score model. The two formulations then share a common local residual structure, while their weighting and parameter gradients differ because of their distinct normalization and backpropagation paths.

FlowCTS computes this residual directly from the current student, avoiding auxiliary fake-model training and its associated tracking lag. Its trajectory-based formulation does not require the above score-consistency assumption: it derives local velocity supervision from continuous student–reference trajectory discrepancies and extends it to multi-step trajectory supervision with $K > 1$. This provides a direct supervision objective for the student velocity field without introducing the scaling associated with score conversion.