COUPLING TRAJECTORY RECTIFICATION AND PERCEPTUAL SUPERVISION IN RECTIFIED FLOW

Anonymous authors
Paper under double-blind review

ABSTRACT

Rectified Flow straightens the trajectories of a flow model so that they can be integrated accurately with few steps, but it supervises the velocity along each path rather than the resulting image. Another family of few-step methods instead supervises that image directly against a teacher, often under a perceptual distance, and reaches high quality at small budgets, but usually trains each model for a fixed number of steps. We propose RefineFlow, which unites both forms of supervision in a single prediction. This prediction, the endpoint estimated by the model, is matched to the teacher's image under a perceptual distance and simultaneously defines the rectification target. Supervision on the output thereby shapes the velocity field along the whole trajectory and straightens it, within a reflow stage and without extra network evaluation or teacher integration. On text-to-image backbones, RefineFlow improves few-step generation quality and prompt alignment over the corresponding reflow baselines with far fewer trainable parameters, without training a separate model for each number of steps.

1 INTRODUCTION

Diffusion models and flow-based generative models (Ho et al., 2020; Song et al., 2021b; Lipman et al., 2023) produce an image by starting from noise and refining it through many sequential network evaluations, so sampling is slow and expensive. A large body of work therefore aims to reach comparable quality with only a handful of evaluations. Flow Matching (Lipman et al., 2023) casts generation as movement along a learned velocity field. The network predicts, at each point in time and space, the direction in which a sample should move, and integrating this field from noise produces an image. Sampling approximates that continuous path with a finite number of discrete steps, so a straighter path incurs less error per step and needs fewer steps overall. Rectified Flow (Liu et al., 2023) targets exactly this property by training the velocity field on straight lines drawn between a noise sample and an image.

Rectified Flow relearns the velocity field through a repeated procedure called reflow, which drives the learned paths closer to straight lines. A teacher velocity field is integrated from noise to generate paired endpoints, and a student is trained on the straight line between each pair at states sampled across the whole time interval. Repeating this yields a field that stays closer to linear along its entire path, so the same trained model can be integrated accurately with few steps or sampled with any number of steps (Lee et al., 2024). Its supervision, however, acts on the velocity at each state, while the image a trajectory arrives at is left unevaluated.

A separate line of work toward few-step generation instead trains a student to match the images a teacher generates (Song et al., 2023; Luo et al., 2023a; Yin et al., 2024b). Placing the supervision on the generated image in this way is effective at raising quality. Perceptual distances have drawn particular attention here as an alternative to matching the two images pixel by pixel. They measure agreement in visual structure and high-level content and therefore align more closely with human perceptual judgments (Zhang et al., 2018; Kang et al., 2024). Such supervision, however, serves a different purpose from straightening the trajectory. It is also often aimed at a step count fixed in advance, leaving a different model for each budget (Wang et al., 2024; Ren et al., 2024). We therefore ask whether the endpoint predicted by the current model can be supervised perceptually and at the same time serve as the destination of the straight path being learned.

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

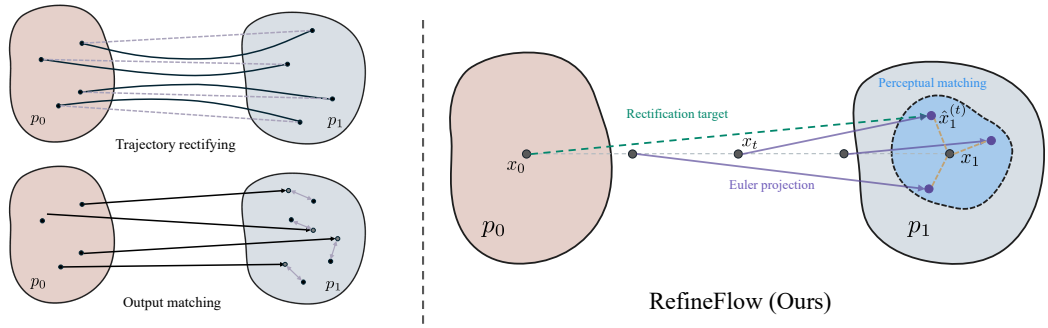


Figure 1: Connecting trajectory rectification and perceptual supervision of the output through a predicted endpoint. **Left, top:** reflow supervises the velocity along straight lines between paired endpoints. **Left, bottom:** a student is trained to match a generated image to the teacher's image. **Right:** at an intermediate state x_t, a single Euler projection of the current velocity gives an endpoint estimate, which is matched against the paired endpoint x_1 under a perceptual distance while the displacement from x_0 to that same estimate serves as the rectification target. The shaded region schematically illustrates a perceptual neighborhood of x_1.

We propose RefineFlow, a reflow method that brings supervision on the generated image into rectification. At each state along a trajectory, the endpoint predicted by the current velocity is matched to the teacher's image under a perceptual distance. The same endpoint is used as the rectification target, so the supervision placed on it propagates along the trajectory through the velocity regression. The learned field is thus encouraged to follow straight paths toward images that stay perceptually close to the teacher's image, combining the step flexibility of reflow with the output quality of methods that match a teacher.

Our contributions are as follows.

- We propose a reflow objective that learns straight paths toward perceptually aligned endpoints, using a single predicted endpoint both as the object of perceptual supervision and as the rectification target rather than combining the two as separate terms.
- The objective fits within a reflow stage. It requires no extra network evaluation or reintegration of the teacher during training, and reuses the pairs that have already been generated.
- On SD v1.5 (Rombach et al., 2022) and SDXL (Podell et al., 2023), RefineFlow improves image fidelity and prompt alignment over the corresponding reflow baselines at small step counts, and is competitive with methods that integrate a teacher during training while using far fewer trainable parameters.

2 Background

We first set up Rectified Flow, including the reflow procedure our method operates in, and then review the two lines of work on few-step generation that it draws on.

2.1 Rectified Flow

Flow Matching (Lipman et al., 2023) provides a simulation-free framework for training continuous normalizing flows (Chen et al., 2018; Papamakarios et al., 2021) by regressing velocity fields associated with prescribed probability paths. Generation proceeds by integrating a learned velocity field from a source distribution p_0 toward a target distribution p_1,

$$\frac{\mathrm{d}z_t}{\mathrm{d}t} = v_\theta(z_t, t), \qquad z_0 \sim p_0. \tag{1}$$

In text-to-image generation, the velocity field is additionally conditioned on a text prompt. We omit this conditioning from the notation for clarity.

Rectified Flow (Liu et al., 2023) learns the velocity field using linear interpolations between paired source and target samples. Given $(x_0, x_1) \sim \pi$, where π is a coupling of p_0 and p_1, the training state and its conditional velocity are

$$x_t = (1-t)x_0 + tx_1, \qquad u(x_0, x_1) = x_1 - x_0. \tag{2}$$

The resulting regression objective is

$$\mathcal{L}_{\mathrm{RF}}(\theta) = \mathbb{E}_{(x_0,x_1)\sim\pi,\ t\sim p(t)} \left[\| v_\theta(x_t, t) - (x_1 - x_0) \|_2^2 \right], \tag{3}$$

where $p(t)$ is the training distribution over $[0, 1]$. This objective is a form of conditional flow matching. Although each training pair defines a straight interpolation, the trajectories obtained by integrating the learned field need not be straight. A source sample is paired with an arbitrary target, so the regression averages over many destinations and the resulting field bends the paths it produces.

Reflow (Liu et al., 2023) removes this mismatch by constructing a new coupling from the model itself. A preceding velocity field is integrated from source samples, and each sample is paired with the endpoint it reaches, giving a coupling in which every source has the destination that model assigns to it. A subsequent velocity field is then trained on the linear interpolations of these pairs with equation 3. Repeating this procedure yields transport that is closer to straight along its entire path and better suited to coarse numerical integration. Throughout this paper x_0 denotes the source noise and x_1 the endpoint generated by the preceding model, and the pairs are produced before a training stage and held fixed within it.

2.2 Few-Step Generation

Several works apply reflow to large-scale text-to-image models (Liu et al., 2024) along different directions. Some refine supervision locally in time, at the cost of solving a teacher ODE at every training iteration (Yan et al., 2024; Ke et al., 2025). Others close the gap between the diffusion and flow matching formulations by aligning their timestep schedules and interpolants (Schusterbauer et al., 2025), or improve the reflow objective itself by changing how timesteps are sampled and how the regression error is measured (Lee et al., 2024). Across these directions, the target of the velocity regression is fixed with respect to the model being trained.

A different line trains a few-step generator by supervising its output against that of a teacher. Consistency-based methods require points along a trajectory to map to the same endpoint and bring this property to text-to-image backbones in latent and phased forms (Song et al., 2023; Luo et al., 2023a; Wang et al., 2024). Distribution matching approaches instead align the student and teacher distributions through learned score functions (Yin et al., 2024b;a), and adversarial methods add a discriminator to sharpen the generated images (Sauer et al., 2024; Lin et al., 2024; Ren et al., 2024). These methods reach strong quality at one to four steps. Each is optimized for a step count fixed during training, however, so a separate checkpoint is usually released for every budget, whereas a rectified flow can trade quality against computation by changing the number of integration steps (Lee et al., 2024).

The two lines supervise different quantities. Reflow places its signal on the velocity at intermediate states and leaves the terminal image unevaluated, while the second line evaluates the terminal image and leaves the trajectory that reaches it largely unshaped. Our method connects the two through a single predicted endpoint, described in Section 3.

3 RefineFlow: Rectification with Perceptual Endpoint Supervision

Our aim is to train the velocity field so that the straight paths it learns lead to endpoints that are perceptually aligned with the teacher's output (Figure 1). We achieve this with a single endpoint predicted by the current velocity field, which is compared with the paired output under a perceptual distance and at the same time defines the target of the velocity regression. Section 3.1 constructs this endpoint and the two objectives defined on it, and Section 3.2 describes how they are trained together.

3.1 Supervising and Rectifying Toward a Predicted Endpoint

To supervise the predicted output perceptually, we first form an endpoint estimate from the current velocity prediction. Given a fixed reflow pair (x_0, x_1) and its interpolated state x_t, we extrapolate $v_\theta(x_t, t)$ over the remaining time interval,

$$\hat{x}_1^{(t)} = x_t + \int_t^1 v_\theta(x_s, s)\,\mathrm{d}s \;\approx\; x_t + (1-t)v_\theta(x_t, t). \tag{4}$$

This single Euler projection approximates the remaining transport by holding the current velocity constant. It provides an endpoint estimate from the velocity evaluation already used for training. Comparing this estimate with the paired output lets a perceptual measure of the terminal prediction act on the velocity field at x_t.

The paired endpoint x_1, generated by the preceding model, provides a fixed teacher reference for the predicted endpoint. We measure their agreement using a perceptual distance,

$$\mathcal{L}_{\text{perc}} = \mathbb{E}_{t,(x_0,x_1)}\left[d_{\text{perc}}\left(\hat{x}_1^{(t)}, x_1\right)\right]. \tag{5}$$

Here, d_{perc} denotes a fixed perceptual distance, whose implementation is described in Section 3.3. This objective encourages the predicted endpoint to preserve the perceptual content of the paired output. Sampling t across the training interval applies this supervision to endpoint estimates formed over different remaining horizons.

The perceptual objective constrains where the predicted endpoint lies, but not the path that leads to it. We therefore use the same endpoint prediction as the rectification target, so that the straight path learned at x_t is directed toward it. Specifically, we define $\hat{u}_t = \mathrm{sg}[\hat{x}_1^{(t)}] - x_0$ as the displacement from the source to the predicted endpoint over the unit time interval, which replaces the conditional velocity $u(x_0, x_1)$ of equation 2 as the regression target at x_t. Here $\mathrm{sg}[\cdot]$ denotes the stop-gradient operation, which prevents the regression term from reducing its loss by moving the target instead of the prediction. We define the corresponding objective as

$$\mathcal{L}_{\text{flow}} = \mathbb{E}_{t,(x_0,x_1)}\left[\|v_\theta(x_t, t) - \hat{u}_t\|_2^2\right]. \tag{6}$$

Unlike the fixed displacement $x_1 - x_0$, this target moves with the model and is recomputed at every iteration.

3.2 Training with a Shared Endpoint

We train the velocity field with both objectives,

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{flow}} + \mathcal{L}_{\text{perc}}, \tag{7}$$

where the two terms carry unit weight and share the same endpoint estimate $\hat{x}_1^{(t)}$ computed from the current velocity prediction. The perceptual objective differentiates through this estimate, updating the velocity field according to its predicted endpoint's agreement with the paired output in perceptual feature space. The flow matching objective instead treats the estimate as fixed and regresses the velocity prediction onto $\hat{u}_t$.

At each iteration, we compute both terms from the current model and optimize them together, so the endpoint estimate that defines the velocity target is itself trained under the perceptual criterion. The reflow pairs are generated once before the stage and never regenerated, and each estimate is compared with the fixed paired endpoint at the same iteration, which keeps it within a perceptual neighborhood of the teacher's output.

3.3 Implementation

Training procedure. We train the model using fixed reflow pairs generated before the current training stage. For each minibatch, we sample a timestep t from the distribution used by the

Algorithm 1 RefineFlow training step

Require: Fixed reflow dataset $\mathcal{D}$, velocity model v_θ with trainable parameters θ, fixed perceptual distance d_{perc}

for each training iteration **do**
  Sample a minibatch $\mathcal{B}$ of (x_0, x_1) from $\mathcal{D}$
  Sample $t \sim p(t)$ for each example
  Construct $x_t \leftarrow (1-t)x_0 + tx_1$
  Predict $v \leftarrow v_\theta(x_t, t)$
  Estimate $\hat{x}_1^{(t)} \leftarrow x_t + (1-t)v$
  Compute $\mathcal{L}_{\text{perc}} \leftarrow \text{mean}_{\mathcal{B}}\, d_{\text{perc}}(\hat{x}_1^{(t)}, x_1)$
  Set $\hat{u}_t \leftarrow \text{sg}[\hat{x}_1^{(t)}] - x_0$
  Compute $\mathcal{L}_{\text{flow}} \leftarrow \text{mean}_{\mathcal{B}} \|v - \hat{u}_t\|_2^2$
  Update θ using $\mathcal{L}_{\text{flow}} + \mathcal{L}_{\text{perc}}$
end for

corresponding reflow baseline and construct x_t from (x_0, x_1). A single evaluation of the current velocity model yields $v_\theta(x_t, t)$, from which we compute the endpoint estimate using equation 4. We use this estimate to evaluate the perceptual objective in equation 5 and construct the velocity target for equation 6. We then update the model using equation 7. Algorithm 1 summarizes the procedure.

Perceptual distance. The text-to-image models considered in this work operate in latent space, so we instantiate d_{perc} using E-LatentLPIPS (Kang et al., 2024). This metric evaluates perceptual similarity directly between latent representations, avoiding image decoding during training. Its parameters remain fixed, while gradients propagate through the metric to the predicted endpoint. The endpoint estimate itself is an algebraic transformation of a velocity prediction already computed for training, so relative to standard reflow this term is the main source of additional cost. We examine the choice of perceptual distance and compare it with direct regression in latent space in Section 4.4.

Parameter-efficient training. In our text-to-image experiments, we train the velocity model with Low-Rank Adaptation (Hu et al., 2022), inserting trainable low-rank modules into the attention layers while keeping the backbone fixed. The reflow baseline we train ourselves on SDXL uses the same configuration, so that the number of trainable parameters matches.

4 Experiments

We evaluate whether our method improves few-step generation on two text-to-image backbones and across sampling budgets.

4.1 Experimental Setup

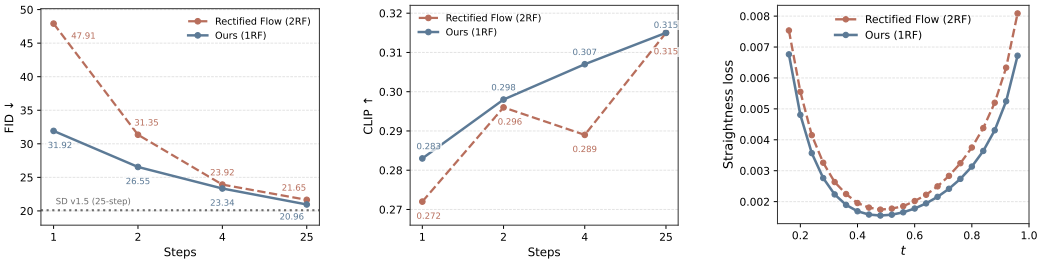
We use Stable Diffusion v1.5 (Rombach et al., 2022) at 512×512, following prior rectified flow settings (Liu et al., 2024), and SDXL (Podell et al., 2023) at 1024×1024. Training uses generated noise-image pairs constructed from the pretrained backbone with prompts sampled from LAION-Aesthetics (Schuhmann et al., 2022). We fine-tune with LoRA (Hu et al., 2022) at rank 64, starting from 1-rectified flow for SD v1.5 and from 2-rectified flow for SDXL, since a single reflow stage left the higher-resolution trajectories insufficiently straight for few-step sampling. We report FID (Heusel et al., 2017) and CLIP score (Radford et al., 2021) on the MS-COCO 2017 and COCO 2014 validation benchmarks (Lin et al., 2014), together with GenEval2 (Kamath et al., 2025), which measures how faithfully the elements described in a prompt appear in the generated image, using non-EMA weights at guidance scale 1.5.

Besides the rectified flow baseline, we compare against the reflow-based methods PeRFlow (Yan et al., 2024), Diff2Flow (Schusterbauer et al., 2025), and ProReflow (Ke et al., 2025), and against the step-distilled models LCM (Luo et al., 2023a), LCM-LoRA (Luo et al., 2023b), PCM (Wang et al., 2024), Hyper-SD (Ren et al., 2024), SDXL-Turbo (Sauer et al., 2024), and SDXL-Lightning (Lin et al., 2024), which target a budget fixed during training.

Table 1: Quantitative comparison on COCO-2017. We report FID-5k (↓) and CLIP score (↑) at different sampling steps.

Method	Params	Res.	Steps	FID ↓	CLIP ↑
SDv1.5 + DPM Solver (Lu et al., 2022)	0.9B	512	25	20.10	0.318
2-Rectified Flow (Liu et al., 2024)	0.9B	512	25	21.65	**0.315**
Diff2Flow (Schusterbauer et al., 2025)	62M	512	25	21.45	0.314
RefineFlow (ours)	62M	512	25	**20.96**	**0.315**
2-Rectified Flow (Liu et al., 2024)	0.9B	512	4	23.92	0.289
PeRFlow (Yan et al., 2024)	0.9B	512	4	23.81	0.302
ProReflow-I† (Ke et al., 2025)	0.9B	512	4	22.97	0.302
ProReflow-II† (Ke et al., 2025)	0.9B	512	4	**22.03**	0.299
Diff2Flow (Schusterbauer et al., 2025)	62M	512	4	25.29	**0.313**
RefineFlow (ours)	62M	512	4	23.34	0.307
2-Rectified Flow (Liu et al., 2024)	0.9B	512	2	31.35	0.296
ProReflow-II† (Ke et al., 2025)	0.9B	512	2	27.59	0.277
Diff2Flow (Schusterbauer et al., 2025)	62M	512	2	32.31	**0.305**
RefineFlow (ours)	62M	512	2	**26.55**	0.298
2-Rectified Flow (Liu et al., 2024)	0.9B	512	1	47.91	0.272
RefineFlow (ours)	62M	512	1	**31.92**	**0.283**

† ProReflow-I and ProReflow-II denote the first and second reflow stages, respectively.



(a) FID across sampling steps. (b) CLIP across sampling steps. (c) Straightness loss over time.

Figure 2: Comparison between 2-rectified flow (Liu et al., 2024) and RefineFlow on SD v1.5. FID and CLIP scores of 2-rectified flow are taken from the original paper. RefineFlow achieves lower FID, higher CLIP score, and lower straightness loss, with the clearest gains in the low-step regime and along the trajectory.

4.2 Quantitative Results

Table 1 reports COCO-2017 results with the SD v1.5 backbone. The gains are largest where the sampling budget is smallest. At one step RefineFlow reaches 31.92 FID against 47.91 for the 2-rectified flow baseline, and at two steps 26.55 against 31.35, also ahead of ProReflow-II, while training only 62M parameters. At four steps it improves over the baseline, PeRFlow, and Diff2Flow, and remains behind ProReflow, which trains the full 0.9B backbone with a multi-step teacher. At 25 steps the model stays close to the DPM-Solver reference, so the improvement at low budgets does not come at the cost of many-step quality. Figures 2a and 2b show the same trend across budgets, with the gap narrowing as the number of steps grows.

Table 2 evaluates on COCO-2014 under the FID-30k protocol. At two steps RefineFlow matches ProReflow-II (15.42 against 15.44) with 62M trainable parameters rather than 0.9B, and at four steps it improves over the 2-rectified flow baseline while the methods that train the full backbone with a teacher during training remain ahead.

Prompt-level alignment shows a clearer advantage. In Table 3 RefineFlow obtains the highest GenEval2 score at both two and four steps, ahead of the reflow baseline and of every step-distilled

Table 2: Quantitative comparison on COCO-2014 FID-30k (↓), all at 512×512.

Method	Params	Steps	FID ↓
2-Rectified Flow	0.9B	4	15.32
PeRFlow	0.9B	4	12.01
ProReflow-I	0.9B	4	11.16
ProReflow-II	0.9B	4	**10.70**
RefineFlow (ours)	62M	4	13.07
2-Rectified Flow	0.9B	2	20.17
ProReflow-II	0.9B	2	15.44
RefineFlow (ours)	62M	2	**15.42**

Table 3: Prompt-level alignment on SD v1.5, GenEval2 ($\times 100$, ↑).

Method	Params	Steps	GenEval2 ↑
2-Rectified Flow	0.9B	4	5.13
PeRFlow	0.9B	4	4.58
LCM-LoRA	67M	4	3.40
PCM	0.9B	4	3.71
Hyper-SD	0.9B	4	5.13
RefineFlow (ours)	62M	4	**5.91**
2-Rectified Flow	0.9B	2	4.19
LCM-LoRA	67M	2	3.15
PCM	0.9B	2	3.63
Hyper-SD	0.9B	2	4.85
RefineFlow (ours)	62M	2	**5.31**

Table 4: Quantitative comparison on SDXL at 4 sampling steps. Multi-step rollout marks methods that solve a multi-step teacher ODE at every training iteration. Params denotes trainable parameters.

Method	Params	Res.	Multi-step rollout	FID ↓	GenEval2 ↑
PeRFlow	3B	1024	✓	20.99	9.61
PCM	3B	1024	✓	21.04	9.21
ProReflow	3B	1024	✓	**19.10**	–
LCM	3B	1024	–	22.16	9.04
SDXL-Turbo	3B	512	–	23.19	8.23
SDXL-Lightning	3B	1024	–	24.56	7.99
Hyper-SD	3B	1024	–	25.57	8.22
RefineFlow (ours)	**196M**	1024	–	21.18	**10.03**

model. The score is a geometric mean over the atomic questions of a prompt, so it rewards realizing every element of a prompt in the same image rather than most of them.

Table 4 extends the evaluation to SDXL at four steps. RefineFlow adapts the backbone with 196M trainable parameters, roughly 15 times fewer than the 3B models it is compared against, and reaches the highest GenEval2 score among them. Its FID is close to that of PeRFlow and PCM, and only ProReflow is clearly ahead, with all three relying on a multi-step teacher ODE during training. Rectified flow has not been reported on SDXL, so we also train a 2-rectified flow baseline under identical settings, against which RefineFlow lowers FID from 24.30 to 21.18 (Appendix C).

4.3 Qualitative Results

Figure 3 compares the rectified flow baseline and RefineFlow at 1, 2, 4, and 25-step sampling. The gap is largest at one and two steps, where the baseline shows visible blurring and structural artifacts while RefineFlow retains coherent structure. At 4 and 25 steps the two converge to similar quality, consistent with Table 1. Additional comparisons on SD v1.5 and SDXL are provided in Appendix E.

4.4 Ablation

Table 5 applies the objective to a different pair-construction pipeline. Replacing only the training objective inside the official Diff2Flow (Schusterbauer et al., 2025) setup improves FID from 32.31 to 28.74 at two steps and from 25.29 to 24.10 at four, indicating that the gain is not tied to a single way of producing the pairs.

Table 6 examines the distance used to supervise the predicted endpoint. Compared with a direct ℓ_2 regression, the perceptual distance keeps the estimate closer to the paired endpoint on both pair-based measures, while its diversity (Friedman & Dieng, 2023) and its distance to the real distribution remain

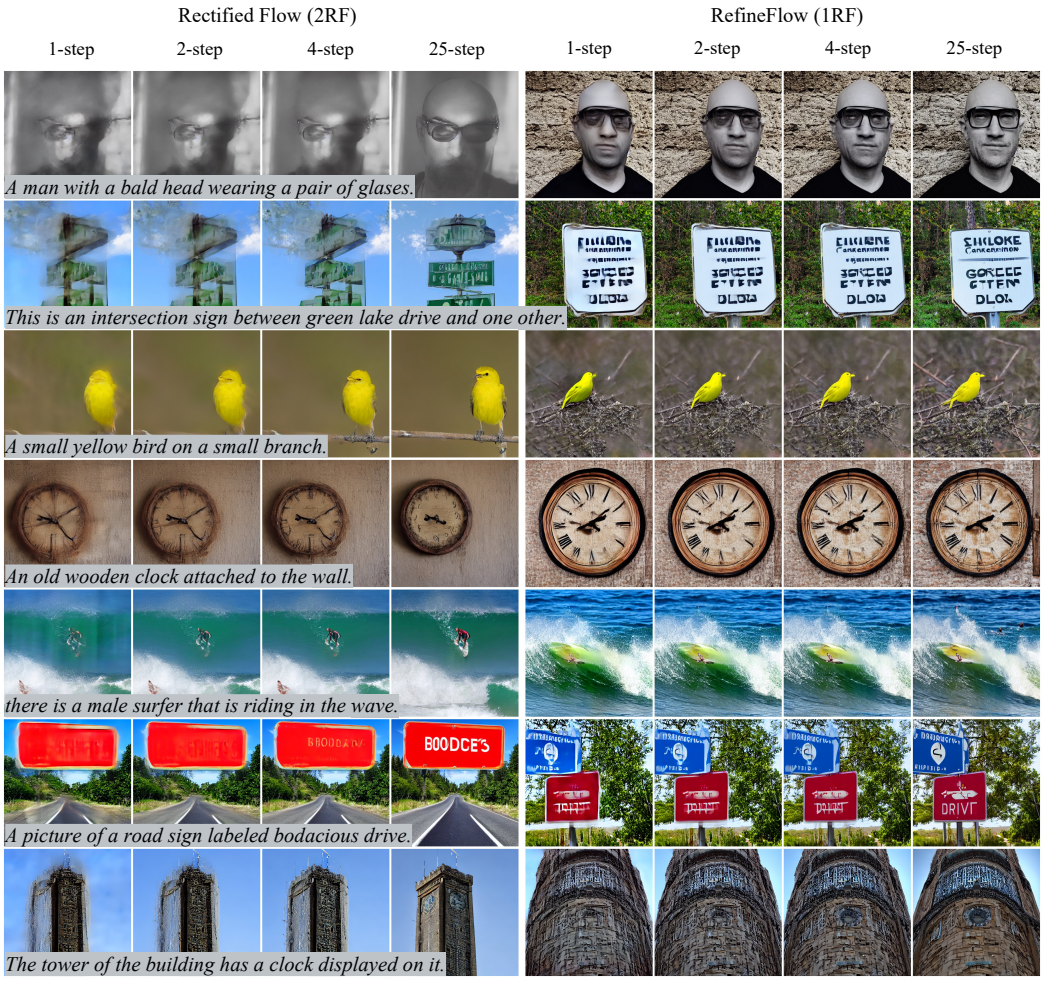


Figure 3: Qualitative comparison between the rectified flow baseline and RefineFlow on SD v1.5 at 1, 2, 4, and 25-step sampling. The gap is largest at one and two steps, where the baseline shows visible blurring and structural artifacts.

Table 5: Plug-in integration with Diff2Flow. Only the training objective is replaced, and the original pair construction is left unchanged.

Method	**1 step**	**2 steps**	**4 steps**
Diff2Flow	–	32.31	25.29
+ Ours	34.87	**28.74**	**24.10**

Table 6: Properties of the predicted endpoint under a direct and a perceptual distance, measured against the paired endpoint and, in the last column, the real distribution.

Distance	**LPIPS↓**	**FID↓**	**Vendi↑**	**FID↓ (real)**
ℓ_2	0.0311	2.42	26.13	16.75
Perceptual	**0.0264**	**1.99**	26.26	16.71

unchanged. The estimate therefore tracks the teacher's output more faithfully without losing coverage of the data distribution.

4.5 Analysis

Trajectory straightness. To check that the effect is visible in the geometry of the learned flow and not only in generation quality, we evaluate the straightness score of Liu et al. (2023),

$$S = \int_0^1 \mathbb{E}\left[\|v_\theta(x_t, t) - (x_1 - x_0)\|^2\right] \mathrm{d}t, \tag{8}$$

where a lower value means the predicted velocity is better aligned with the straight transport direction. We measure S on 1,000 MS-COCO 2017 prompts, generating noise-image pairs from each model and evaluating its own velocity along those pairs. RefineFlow reaches a mean straightness of 0.00675 against 0.00724 for the rectified flow baseline, a reduction of 6.8%. Figure 2c shows that the gap holds at every sampled timestep and is most visible toward the two ends of the interval, where both models deviate most from the straight direction.

Why the gains concentrate at small budgets. The endpoint estimate in equation 4 is the result of a single Euler step from x_t to the end of the interval, which is the operation a sampler performs when it covers the remaining interval in one step. At $t = 0$ it coincides with one-step generation from x_0, so the perceptual objective acts directly on the few-step sampler, and its influence fades as additional steps subdivide the interval. Table 1 follows this pattern. The FID gap to the 2-rectified flow baseline is 16.0 at one step and 4.8 at two, and falls below one point at four and 25 steps.

One checkpoint across budgets. The velocity field is regressed at intermediate states spread over the whole time interval, so a single trained model can be integrated with any number of steps. The step-distilled baselines instead release separate weights for different budgets. Hyper-SD (Ren et al., 2024) provides three LoRAs for one, two, and four steps, and PCM (Wang et al., 2024) several variants across budgets and guidance settings, each evaluated here with the one released for that step count. A single RefineFlow checkpoint produces every entry in Table 1, and its FID decreases steadily from one step to 25, so the same model trades computation for quality as the number of steps grows.

Pixel-space validation. The objective assumes only reflow pairs and a velocity model, and is not specific to latent or text-conditioned models. On CIFAR-10 (Krizhevsky et al., 2009), starting from the public 1-rectified flow checkpoint (Liu et al., 2023), it improves one-step FID from 12.21, the 2-rectified flow result reported from the same initialization, to 6.00. Replacing the perceptual distance with a direct regression reaches 8.10, so the ordering observed at the level of the predicted endpoint in Table 6 carries over to generation quality. Full results, together with a class-conditional ImageNet (Deng et al., 2009) setting, are reported in Appendix B.

Skipping a reflow stage. On SD v1.5, RefineFlow is trained from 1-rectified flow, yet it surpasses the reported 2-rectified flow result at every sampling budget in Tables 1 and 2. A second reflow stage requires generating a new set of pairs by integrating the first-stage model and training on them again, so reaching this level from the first stage removes that cost entirely.

5 Limitations

RefineFlow obtains its training signal from a single velocity evaluation on a fixed pair, which is weaker than the signal available to methods that integrate a multi-step teacher at every iteration. At higher resolution those methods remain ahead in FID, and reaching a comparable signal without teacher integration is left for future work. We also have not examined whether the objective brings further gains when combined with methods that rectify locally within time windows.

6 Conclusion

We presented RefineFlow, a reflow method that learns straight paths toward perceptually aligned endpoints. The endpoint predicted by the current velocity field is supervised against the teacher's image under a perceptual distance and at the same time serves as the rectification target, so supervision on the output shapes the velocity field along the whole trajectory within a reflow stage and without additional network evaluation. On SD v1.5, a model trained from the first reflow stage surpasses the reported second-stage result at every sampling budget, and on SDXL it improves over a reflow baseline trained under identical settings while attaining the highest prompt alignment among the compared few-step models.

AI USE STATEMENT

We used generative AI tools to discuss and refine the paper's conceptual framing, provide feedback on the methodology and experimental design, interpret experimental results, and draft, translate, and edit parts of the manuscript. AI tools also assisted with organizing experimental results into LaTeX tables and discussing the design of scientific figures. We did not use generative AI tools to generate synthetic datasets, implement or train the models, run experiments, or formulate or prove mathematical claims. The authors reviewed the AI-assisted text, tables, interpretations, and figure designs, and verified the reported results against the experimental records and the cited sources. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with AI assistance.

ETHICS STATEMENT

This work studies text-to-image generation using publicly available models and benchmark prompts. We do not collect data from human participants or personal data. Generated images may nevertheless reflect biases in the pretrained models and may include harmful content. Our evaluation focuses on generation quality and prompt alignment; it does not constitute a comprehensive audit of bias, privacy, or safety. The models and datasets remain subject to their respective licenses and usage policies.

REPRODUCIBILITY STATEMENT

Section 3 describes the training objective and procedure, and Algorithm 1 summarizes a training step. Section 4.1 and Appendix D report the pair generation, training configurations, sampling settings, and evaluation protocols for each backbone, and Appendix C describes the reflow baseline we train on SDXL. We provide the implementation and configuration files in the supplementary materials, and will release the trained checkpoints upon publication.

REFERENCES

James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. `https://cdn.openai.com/papers/dall-e-3.pdf`, 2023.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. *Advances in neural information processing systems*, 36:10088–10115, 2023.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

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.

Dan Friedman and Adji Bousso Dieng. The Vendi score: A diversity evaluation metric for machine learning. *Transactions on machine learning research*, 2023.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *International conference on machine learning*, pp. 2790–2799. PMLR, 2019.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Amita Kamath, Kai-Wei Chang, Ranjay Krishna, Luke Zettlemoyer, Yushi Hu, and Marjan Ghazvininejad. GenEval 2: Addressing benchmark drift in text-to-image evaluation. *arXiv preprint arXiv:2512.16853*, 2025.

Minguk Kang, Richard Zhang, Connelly Barnes, Sylvain Paris, Suha Kwak, Jaesik Park, Eli Shechtman, Jun-Yan Zhu, and Taesung Park. Distilling diffusion models into conditional GANs. In *European Conference on Computer Vision*, pp. 428–447. Springer, 2024.

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

Lei Ke, Haohang Xu, Xuefei Ning, Yu Li, Jiajun Li, Haoling Li, Yuxuan Lin, Dongsheng Jiang, Yujiu Yang, and Linfeng Zhang. ProReflow: Progressive reflow with decomposed velocity. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 28029–28038, 2025.

Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang 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 *The Twelfth International Conference on Learning Representations*, 2024.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

Sangyun Lee, Zinan Lin, and Giulia Fanti. Improving the training of rectified flows. *Advances in neural information processing systems*, 37:63082–63109, 2024.

Shanchuan Lin, Anran Wang, and Xiao Yang. SDXL-Lightning: Progressive adversarial diffusion distillation. *arXiv preprint arXiv:2402.13929*, 2024.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

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

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

Cheng Lu and Yang Song. Simplifying, stabilizing and scaling continuous-time consistency models. In *The Thirteenth International Conference on Learning Representations*, 2025.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. *Advances in neural information processing systems*, 35:5775–5787, 2022.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, 22 (4):730–751, 2025.

Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023a.

Simian Luo, Yiqin Tan, Suraj Patil, Daniel Gu, Patrick von Platen, Apolinário Passos, Longbo Huang, Jian Li, and Hang Zhao. LCM-LoRA: A universal Stable-Diffusion acceleration module. *arXiv preprint arXiv:2311.05556*, 2023b.

Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. SiT: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, pp. 23–40. Springer, 2024.

Thuan Hoang Nguyen and Anh Tran. SwiftBrush: One-step text-to-image diffusion model with variational score distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7807–7816, 2024.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pp. 8162–8171. PMLR, 2021.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, and Ricky TQ Chen. Multisample flow matching: Straightening flows with minibatch couplings. *arXiv preprint arXiv:2304.14772*, 2023.

Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. DreamFusion: Text-to-3D using 2D diffusion. In *The Eleventh International Conference on Learning Representations*, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with CLIP latents. *arXiv preprint arXiv:2204.06125*, 2022.

Yuxi Ren, Xin Xia, Yanzuo Lu, Jiacheng Zhang, Jie Wu, Pan Xie, Xing Wang, and Xuefeng Xiao. Hyper-SD: Trajectory segmented consistency model for efficient image synthesis. In *Advances in Neural Information Processing Systems*, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10674–10685. IEEE, 2022.

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. In *International Conference on Learning Representations*, 2022.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *European Conference on Computer Vision*, pp. 87–103. Springer, 2024.

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.

Johannes Schusterbauer, Ming Gui, Frank Fundel, and Bjorn Ommer. Diff2Flow: Training flow matching models via diffusion model alignment. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 28347–28357. IEEE Computer Society, 2025.

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

Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. In *The Twelfth International Conference on Learning Representations*, 2024.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

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

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 32211–32252, 2023.

Fu-Yun Wang, Zhaoyang Huang, Alexander W Bergman, Dazhong Shen, Peng Gao, Michael Lingelbach, Keqiang Sun, Weikang Bian, Guanglu Song, Yu Liu, et al. Phased consistency models. *Advances in neural information processing systems*, 37:83951–84009, 2024.

Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. ProlificDreamer: High-fidelity and diverse text-to-3D generation with variational score distillation. *Advances in neural information processing systems*, 36:8406–8441, 2023.

Hanshu Yan, Xingchao Liu, Jiachun Pan, Jun Hao Liew, Qiang Liu, and Jiashi Feng. PeRFlow: Piecewise rectified flow as universal plug-and-play accelerator. *Advances in Neural Information Processing Systems*, 37:78630–78652, 2024.

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.

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

APPENDIX

Table of Contents

Appendix A Related Work 14

Appendix B Pixel-Space Validation 15

Appendix C Reflow Baseline on SDXL 15

Appendix D Implementation and Evaluation Details 16

Appendix E Additional Qualitative Results 17

A Related Work

Diffusion and Score-based Models. Score-based generative models (Song & Ermon, 2019; Song et al., 2021b) and denoising diffusion probabilistic models (DDPMs) (Ho et al., 2020; Nichol & Dhariwal, 2021) have established the foundation for modern deep generative modeling. These models define generation as a gradual transformation from noise to data, either through reverse-time stochastic dynamics or learned denoising transitions. Subsequent work improved architecture design, noise schedules, and training stability (Dhariwal & Nichol, 2021; Karras et al., 2022; Kingma et al., 2021), enabling high-fidelity image synthesis at scale.

Fast Sampling and ODE Solvers. Reducing the number of inference steps in diffusion models has been studied extensively via improved numerical solvers (Song et al., 2021a; Lu et al., 2025) and adaptive step scheduling (Karras et al., 2022). By interpreting sampling as solving a reverse diffusion process or probability-flow ODE, these methods reduce sampling cost without changing the pretrained model. However, because they do not modify the learned generative dynamics, their performance often degrades in the extreme low-step regime.

Distillation-based Acceleration. Progressive Distillation (Salimans & Ho, 2022) and Consistency Models (Song et al., 2023; Song & Dhariwal, 2024; Kim et al., 2024; Lu & Song, 2025) compress multi-step samplers into few-step generators through teacher-student training or self-consistency objectives. These approaches directly optimize models for low-step sampling, rather than relying only on better numerical integration. Score distillation approaches (Poole et al., 2023; Wang et al., 2023; Nguyen & Tran, 2024; Yin et al., 2024a) align student distributions to teacher score functions, enabling one-step generation without paired data.

Text-to-Image Generation. Large-scale text-to-image models (Ramesh et al., 2022; Saharia et al., 2022; Betker et al., 2023; Esser et al., 2024) have demonstrated remarkable generation quality by combining diffusion objectives with powerful text encoders and large-scale image-text datasets. Latent diffusion models (Rombach et al., 2022) reduce computational cost by operating in compressed latent spaces, and have become the standard backbone for downstream acceleration research. Our experiments follow this line by evaluating on Stable Diffusion and SDXL-style latent text-to-image models under few-step sampling budgets.

Parameter-Efficient Fine-Tuning. PEFT methods (Houlsby et al., 2019; Hu et al., 2022; Dettmers et al., 2023) enable efficient adaptation of large pretrained models by updating only a small subset of parameters. In the context of diffusion model acceleration, LoRA-based fine-tuning (Hu et al., 2022) provides a practical framework for adapting large backbones without full-parameter optimization, as employed in our work and several concurrent methods (Schusterbauer et al., 2025). This is particularly useful for large text-to-image backbones, where full-model reflow or distillation can be computationally expensive.

Optimal Transport and Flow-based Models. Continuous normalizing flows (Papamakarios et al., 2021; Chen et al., 2018) and flow matching (Liu et al., 2023; Lipman et al., 2023; Geng et al., 2025; Pooladian et al., 2023) provide simulation-free alternatives to diffusion training by directly regressing vector fields. Rectified Flow (Liu et al., 2023; 2024) extends this framework with a reflow procedure that progressively straightens transport trajectories. Recent reflow-style methods further improve scalability and few-step generation through trajectory partitioning, progressive training, or diffusion-to-flow alignment. Our method builds on this family by supervising the predicted endpoint within a reflow stage, rather than changing the pair-generation pipeline or requiring additional teacher rollouts.

B Pixel-Space Validation

Although our main experiments focus on latent text-to-image models, the objective is not specific to latent diffusion or text conditioning. It requires only reflow pairs and a velocity model from which an endpoint estimate can be formed. To verify this broader applicability, we additionally evaluate RefineFlow on pixel-space flow models using CIFAR-10 (Krizhevsky et al., 2009) and ImageNet (Deng et al., 2009) 256×256. These experiments test whether the same objective improves low-step generation when the model operates directly in pixel space rather than in a VAE latent space.

We evaluate RefineFlow on CIFAR-10 starting from the official 1-Rectified Flow checkpoint (Liu et al., 2023). We generate 100K reflow pairs using 100-step sampling and train a second reflow stage on top of this checkpoint, evaluating 1-step generation quality. The baseline in Table 7 is the 2-Rectified Flow result reported in the original paper, which is trained from the same 1-Rectified Flow initialization as our results. The perceptual distance achieves the best FID and IS, consistent with our main finding that it provides a more informative supervision signal for low-step generation than direct regression.

Table 7: FID (↓) and IS (↑) on CIFAR-10 at 1-step sampling.

Method	FID ↓	IS ↑
Rectified Flow (Liu et al., 2023)	12.21†	8.08†
RefineFlow w/ ℓ_2 distance	8.10 (−4.11)	8.50 (+0.42)
RefineFlow w/ perceptual distance	6.00 (−6.21)	8.80 (+0.72)

† Results reported in the original paper.

We further evaluate RefineFlow on ImageNet 256×256 using a pretrained SiT XL/2 (Ma et al., 2024). Compared with CIFAR-10, this setting provides a higher-resolution and class-conditional pixel-space validation. We generate 1M reflow pairs using 250-step Heun sampling, with class labels sampled uniformly to avoid class imbalance in the pair set. Table 8 reports FID before and after applying RefineFlow at low-step sampling. The pretrained SiT model performs poorly when naively sampled with few steps, reflecting the large discretization error in the extreme low-step regime. After applying RefineFlow, FID improves substantially across 1, 2, and 4-step sampling, indicating that the objective also improves trajectory behavior in pixel-space class-conditional generation.

Table 8: FID on ImageNet 256×256 before and after applying RefineFlow on SiT XL/2.

Method	1-step	2-step	4-step
SiT XL/2 (Ma et al., 2024) (pretrained)	302.47	164.86	53.34
+ RefineFlow (ours)	6.56	5.96	5.66

C Reflow Baseline on SDXL

Rectified Flow has not been reported on SDXL in prior work, so we train the 2-Rectified Flow baseline ourselves to isolate the effect of the proposed objective. The baseline and RefineFlow share

the same backbone, pair generation pipeline, LoRA configuration, and training hyperparameters. The only difference is the supervision: the baseline regresses the fixed displacement $\boldsymbol{x}_1 - \boldsymbol{x}_0$, while RefineFlow regresses $\hat{\boldsymbol{x}}_1 - \boldsymbol{x}_0$ and supervises $\hat{\boldsymbol{x}}_1$ perceptually against $\boldsymbol{x}_1$.

Table 9 reports the 4-step FID on MS-COCO 2014. RefineFlow improves the baseline from 24.30 to 21.18 FID, a 12.8% relative reduction, together with a higher CLIP score. Since the pair-generation pipeline and the training setup are held fixed, the gain is attributable to the objective alone.

Table 9: Comparison against the 2-Rectified Flow baseline on SDXL at 4-step sampling on MS-COCO 2014. Both methods use the same backbone, pair generation pipeline, LoRA configuration, and training hyperparameters.

Method	FID ↓	CLIP ↑
2-Rectified Flow (trained by us)	24.30	0.318
RefineFlow (ours)	21.18 (−3.12)	0.333 (+0.015)

D Implementation and Evaluation Details

Pair generation. Training pairs $(\boldsymbol{x}_0, \boldsymbol{x}_1)$ are generated from the pretrained base model before training. For both SD v1.5 and SDXL, we sample prompts from LAION-Aesthetics and generate 1.6M noise-image pairs using classifier-free guidance with scale 5.0 and 25 DDIM steps. The source variable $\boldsymbol{x}_0$ denotes the initial noise latent, and $\boldsymbol{x}_1$ denotes the corresponding generated endpoint. These pairs are kept fixed throughout training, and our method modifies only the supervision used at each sampled intermediate state, without regenerating pairs or introducing teacher rollouts.

SD v1.5. We fine-tune at 512×512 resolution using LoRA modules inserted into the attention layers of the UNet. The LoRA rank is set to 64, while the pretrained backbone parameters are frozen. We train for 400,000 iterations with batch size 64 using the Adam optimizer with learning rate 3×10^{-5}. The perceptual distance is instantiated with E-LatentLPIPS and carries unit weight.

SDXL. We fine-tune at 1024×1024 resolution with the same LoRA configuration. Training runs for 8,000 iterations across two reflow stages with batch size 80 and learning rate 2×10^{-4}.

Evaluation. All quantitative results are obtained with the Euler ODE solver from non-EMA weights at guidance scale 1.5, evaluating SD v1.5 at 1, 2, 4, and 25 steps and SDXL at 4 steps. FID is computed with pytorch-fid using 2048 features and a fixed seed, on 5,000 COCO-2017 validation captions for the FID-5k protocol, 30,000 COCO-2014 captions for FID-30k, and the first 10,000 COCO-2014 captions for the SDXL FID-10k protocol. GenEval2 is evaluated with the official code on its 800 prompts with one image per prompt, using Qwen3-VL-8B-Instruct as the underlying VQA model, and we report Soft-TIFA GM. Baselines that release a separate checkpoint per sampling budget are evaluated with the checkpoint released for that budget.

E ADDITIONAL QUALITATIVE RESULTS



Figure 4: Comparison of qualitative results on SD v1.5.

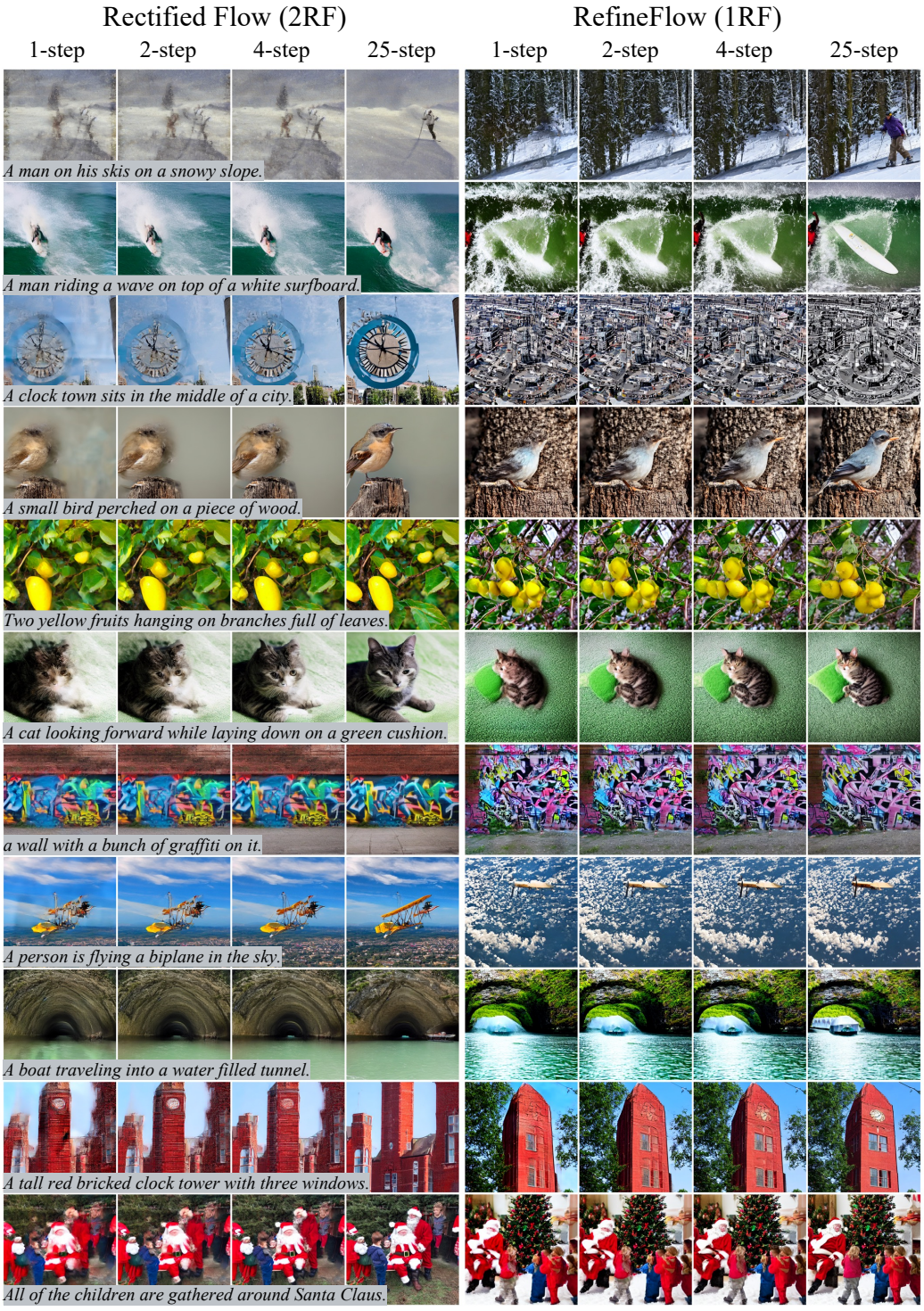
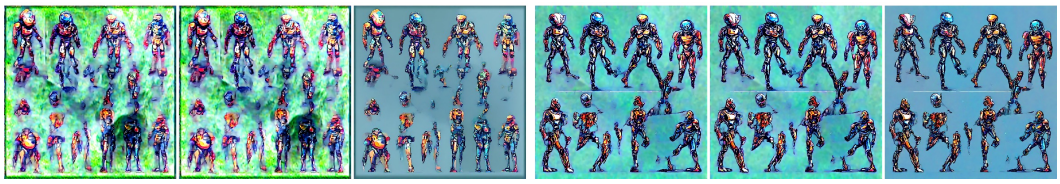


Figure 5: Comparison of qualitative results on SD v1.5.



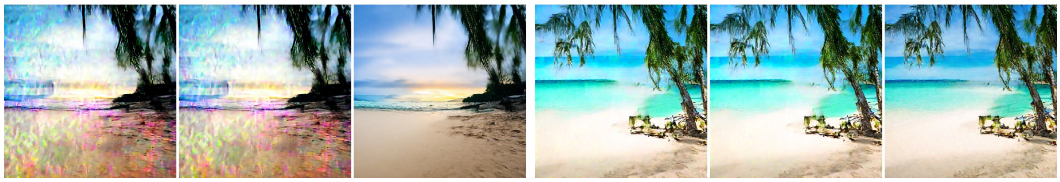
Everyone's Favourite Boogeyman John Wick Is Now Getting A TV Series.



Various color, future cyborg warriors, soldiers in futuristic armor, alien army robot Free Vector.

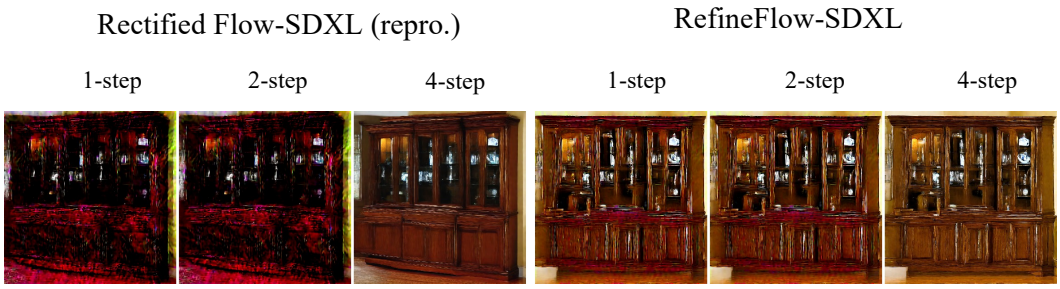


Shell Flowergirl Girl Basket.



Boracay Beach, one of the best summer destinations in Asia.

Figure 6: Comparison of qualitative results on SDXL. The baseline is the 2-rectified flow model trained under identical settings in Appendix C.



Classic Oak Burnished Large China Cabinet.



Les peintres indiens d'Amérique, American Indian painters,
Pine Tree Ceremonial Dance.



Ethopian Girl with the Pearl Earring.



Jentas Lukket Tå Leather Flower Girl Shoes med Bowknot.

Figure 7: Comparison of qualitative results on SDXL. The baseline is the 2-rectified flow model trained under identical settings in Appendix C.