SCoT: Unifying Consistency Models and Rectified Flows via Straight-Consistent Trajectories

Zhangkai Wu
School of Computing
Macquarie University
zhangkai.wu@mq.edu.au

Xuhui Fan*
School of Computing
Macquarie University
xuhui.fan@mq.edu.au

Hongyu Wu
School of Computing
Macquarie University
hongyu.wu@students.mq.edu.au

Longbing Cao
School of Computing
Macquarie University
longbing.cao@mq.edu.au

Abstract

Pre-trained diffusion models are commonly used to generate clean data (e.g., images) from random noises, effectively forming pairs of noises and corresponding clean images. Distillation on these pre-trained models can be viewed as the process of constructing advanced trajectories within the pair to accelerate sampling. For instance, consistency model distillation develops consistent projection functions to regulate trajectories, although sampling efficiency remains a concern. Rectified flow method enforces straight trajectories to enable faster sampling, yet relies on numerical ODE solvers, which may introduce approximation errors. In this work, we bridge the gap between the consistency model and the rectified flow method by proposing a Straight-Consistent Trajectories (SCoT) model. SCoT enjoys the benefits of both approaches for fast sampling, producing trajectories with consistent and straight properties simultaneously. These dual properties are strategically balanced by targeting two critical objectives: (1) regulating the gradient of SCoT's mapping function to a constant and (2) ensuring trajectory consistency. Extensive experimental results demonstrate the effectiveness and efficiency of SCoT.

1 Introduction

Pre-trained diffusion models Ho et al. (2020); Song et al. (2021b); Rombach et al. (2022); Poole et al. (2023); Esser et al. (2024) have demonstrated impressive performance in real-world tasks such as high-quality image synthesis and image editing. However, such practical models usually require extensive computational resources to train, as well as a large number of model evaluations to generate high-quality samples (e.g. images). Using these pre-trained teacher models to generate pairs of random noises and their corresponding clean images, a popular choice for low-cost training and fast sampling is to train a student distillation model with advanced "trajectories" within the pair Salimans and Ho (2022); Wang et al. (2023, 2024); Yin et al. (2024b); Luo et al. (2024); Xie et al. (2024); Nguyen and Tran (2024); Yin et al. (2024a); Sauer et al. (2024); Xu et al. (2024); Zhu et al. (2025); Frans et al. (2025a); Sauer et al. (2025).

We can group these trajectory-based distillation methods into two categories: *consistency model distillation* Song et al. (2023); Kim et al. (2024); Lu and Song (2025) and *rectified flow distillation* Liu

*Corresponding author: xuhui.fan@mq.edu.au

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

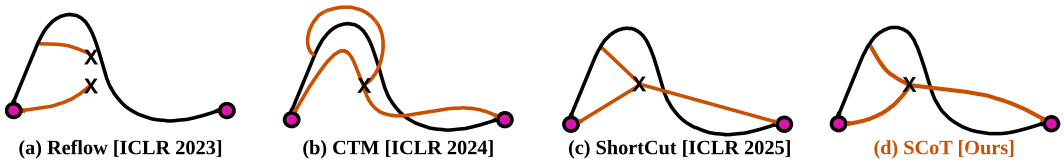

Figure 1: Comparison of trajectory distillation methods. The black line in each panel denotes the teacher trajectory of pre-trained diffusion models, which are connected within the pair of a random noise (left dot) and a clean image (right dot). The red solid line is the student trajectory of the distillation model. Panel (a) Reflow Liu et al. (2023b) straightens its student trajectory by enforcing its velocity be close to a constant. However, its trajectory maps different points to different values due to the lack of consistency. Panel (b) CTM Kim et al. (2024) places consistency requirement for the student trajectory. However, it might be difficult to track the student trajectory when it is of high curvatures. In panel (c), the Shortcut model Frans et al. (2025b) focuses on velocity estimation, while uses straight lines to approximate the trajectory and ensure the consistency. Our proposed SCoT model in panel (d) enforces straightness for consistent student trajectory. By avoiding the approximating errors of solving ODEs and by straightening the student trajectory, SCoT successfully bridges the gap between rectified flows and CTM distillations and enjoys the benefits of both approaches.

et al. (2023c); Zhu et al. (2025). Consistency model distillation focuses on the trajectory itself, requiring a "valid" consistent trajectory. That is, the student trajectory should map to the same point regardless of different initial points. Rectified flow distillation seeks a straight student trajectory by enforcing its velocities be close to the magnitude of point changes. While consistency distillation directly models pointwise changes along the trajectory, the straight trajectory of rectified flow distillation may reduce the number of steps required to generate high-quality images.

However, both categories have their own limitations. Consistency model distillation typically requires multiple steps to generate high-quality samples Song et al. (2023); Kim et al. (2024). This is likely due to the challenge of tracking the mapping function for trajectories with high curvatures. In the case of rectified flow distillation, numerical ODE solvers are required for a learned velocities, which introduce approximation errors that degrade the final output quality.

In order to address these issues and enjoy the benefits of both approaches, we bridge the gap between consistency models and rectified flows by proposing the **S**traight-**Co**nsistent **T**rajectories (SCoT) model in this paper. SCoT aims to produce trajectories that are consistent, which is a valid condition for trajectories, and are straight, which simplifies the point changes in the trajectory. In detail, SCoT regulates the trajectory through two aspects: (1) *trajectory straightness* by optimizing velocities towards the amount of point changes in the pre-trained model, (2) *trajectory consistency* by ensuring points from different time steps are projected to converge at the same value at future time steps.

In summary, the proposed SCoT is the first to produce consistent and straight trajectories between random noise and clean images, which unifies the consistency models and rectified flows. Consistency guarantees valid trajectories, while straightness facilitates the approximation of the trajectory projection function and faster sample generation. With this design, SCoT achieves state-of-the-art results, enabling compact models to generate high-quality data in N-steps, or even a single step. Figure 1 highlights the key differences among consistency distillation, rectified flow distillation, ShortCut model Frans et al. (2025b), and the proposed SCoT.

2 A trajectory perspective on Pre-trained diffusion model distillations

Notation We consider the trajectory to be working within the time interval $[0, 1]$, with the time steps specified as $1 = t_N > t_{N-1} > \ldots > t_1 = 0$. Since image synthesis is a common task for diffusion models, we denote $\mathbf{x}_0$ as the clean image, $\mathbf{x}_t$ as the noisy image at the time step t, and $\mathbf{x}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ as the random noise sampled from a standard Gaussian distribution. Furthermore, we use $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ to represent the parameters of the pre-trained teacher model and the student distillation model, respectively. Detailed descriptions of the notation are provided in **??** and Table 7.

Diffusion models Diffusion models (DMs) Ho et al. (2020); Song et al. (2021a); Dhariwal and Nichol (2021) generate data by learning the score function of noisy images at multiple noise scales. At each

time step t, a *clean* image $\mathbf{x}_0$ is first diffused into a noisy image $\mathbf{x}_t$ through a forward process $\mathbf{x}_t := \alpha_t \mathbf{x}_0 + \sigma_t \mathbf{x}_1, \mathbf{x}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, where α_t and σ_t^2 are the diffusion coefficient and variance, respectively. Using $p_t(\mathbf{x}_t)$ to denote the probability of diffusion in $\mathbf{x}_t$, DMs learn a neural network $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ that matches the score of the corrupted image $\mathbf{s}_{\text{real}}(\mathbf{x}_t) := \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) = -\sigma_t^{-1}(\mathbf{x}_t - \alpha_t \mathbf{x}_0)$ by minimizing the loss $\mathbb{E}_{t,\mathbf{x}_t}\left[\omega(t)\|\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \mathbf{s}_{\text{real}}(\mathbf{x}_t)\|^2\right]$, where $\omega(t)$ is a weighting function.

Rectified flows Lipman et al. (2023); Liu et al. (2023b); Albergo et al. (2023) can be regarded as an extension of diffusion models by defining a linear interpolation between random noise $\mathbf{x}_1$ and a clean image $\mathbf{x}_0$ as $\mathbf{x}_t = t\mathbf{x}_1 + (1-t)\mathbf{x}_0, 0 \le t \le 1$. In details, they use the ODE $\mathrm{d}\mathbf{x}_t/\mathrm{d}t = \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ to transport between the noise distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$ and data distribution $\boldsymbol{\pi}_0(\mathbf{x}_0)$, whereas the velocity $\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ is learned by minimizing the loss $\mathbb{E}_t\left[\|(\mathbf{x}_1 - \mathbf{x}_0) - \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\|^2\right]$.

Given a random noise $\mathbf{x}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, a new clean image $\widehat{\mathbf{x}}_0$ can be generated either through an iterative denoising process with a trained $\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ (in DMs), or through an ODE solver as $\widehat{\mathbf{x}}_0 = \mathbf{x}_1 + \int_1^0 \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\mathrm{d}t$ (in rectified flows). In either case, a pair of random noise and its corresponding clean image is formed as $(\mathbf{x}_1, \widehat{\mathbf{x}}_0)$. Trajectory distillation focuses on the construction of advanced trajectories to accelerate sampling without compromising image qualities.

Reflow distillation The teacher trajectory produced by the pre-trained velocity $\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ may not be straight, since their training set $\mathbf{x}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $\mathbf{x}_0 \sim \boldsymbol{\pi}_0(\mathbf{x}_0)$ are not paired. To address this issue, reflow distillation Liu et al. (2023a,c); Zhu et al. (2025); Li et al. (2025) works on the pair $(\mathbf{x}_1, \widehat{\mathbf{x}}_0)$ by training a new velocity $\mathbf{v}_{\boldsymbol{\phi}}(\mathbf{x}_t, t)$ that approximates the direction $\widehat{\mathbf{x}}_0 - \mathbf{x}_1$ within the pair. With the trained $\mathbf{v}_{\boldsymbol{\phi}}(\mathbf{x}_t, t)$ expected to approximate the straight direction from $\mathbf{x}_1$ to $\widehat{\mathbf{x}}_0$, reflow distillations may use fewer steps to generate high-quality images. Other approaches uses one neural network to approximate the magnitude of changes over the whole time period Liu et al. (2023c).

Consistency distillation Consistency model (CM) Song et al. (2023) can be regarded as one trajectory distillation method. CM studies a consistent projection function $f_{\boldsymbol{\phi}}(\mathbf{x}_t, t)$ that maps any noisy image $\mathbf{x}_t$ to the clean image $\mathbf{x}_0$: $f_{\boldsymbol{\phi}}(\mathbf{x}_t, t) = \mathbf{x}_0, \forall t \in [0, 1]$. Its distillation objective function can be written as a weighted distance metric $\mathbf{D}(\cdot, \cdot)$ between the mapped clean images from two adjacent points $\mathcal{L}_{\text{CM}} = \mathbb{E}_t\left[\omega(t)\mathbf{D}\left(\mathbf{f}_{\boldsymbol{\phi}}(\mathbf{x}_{t+\Delta t}, t+\Delta t), \mathbf{f}_{\boldsymbol{\phi}^-}(\widehat{\mathbf{x}}_t^{\boldsymbol{\theta}}, t)\right)\right]$, where $\boldsymbol{\phi}^-$ is the exponential moving average of the past values $\boldsymbol{\phi}$, and $\widehat{\mathbf{x}}_t^{\boldsymbol{\theta}}$ is obtained from the pre-trained model as $\widehat{\mathbf{x}}_t^{\boldsymbol{\theta}} = \mathbf{x}_{t+\Delta t} - t\Delta t \nabla_{\mathbf{x}_{t+\Delta t}} \log p_{t+\Delta t}(\mathbf{x}_{t+\Delta t})$. CM obtains a "valid" trajectory by mapping different points to the same clean image.

Building on CM, Consistency Trajectory Model (CTM) Kim et al. (2024) introduces a multi-step consistent mapping function defined as:

$$G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s) = (s/t)\mathbf{x}_1 + (1 - s/t) g_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s), \tag{1}$$

in which $g_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)$ is left unconstrained, and is parameterized by a neural network $g_{\boldsymbol{\phi}}(\cdot)$. Equation (1) ensures $G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)$ satisfies the boundary condition as $G_{\boldsymbol{\phi}}(\mathbf{x}_1, 1, 1) = \mathbf{x}_1$.

Concurrent works MeanFlow Geng et al. (2025) is a concurrent work to SCoT that also unifies consistency models and rectified flows via velocity integration. Its key distinction lies in introducing a correction term derived from the gradient with respect to the starting step t, unlike SCoT's use of the gradient at the terminating step s. Although rectified flow is regarded as a special case of MeanFlow under an infinitesimal integration range, it is not clear if MeanFlow itself can learn straight trajectories in practice. Another concurrent work FlowMap Sabour et al. (2025); Boffi et al. (2025) also shares the same target. By using the CTM as the backbone, it is unclear if such a setting can achieve both straight and consistent trajectories in practice.

3 Straight-Consistent Trajectories (SCoT) model

Given a pre-trained noise-image pair $(\mathbf{x}_1, \widehat{\mathbf{x}}_0)$, the proposed Straight-Consistent Trajectories (SCoT) model aims to learn a trajectory-based projection function $G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)$ that produces straight and consistent trajectories within the pair. Similar to the CTM, the projection function $G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)$ takes the current time step t, the values $\mathbf{x}_t$ at step t, as well as the future time step s as inputs, and outputs the values at time step s as $\widehat{\mathbf{x}}_s := G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)$.

3.1 Main Components of SCoT

We introduce two regulators that refine the SCoT trajectory regarding its velocity and consistency.

Constant-valued velocity In order to ensure that SCoT produces a straight trajectory, we regularize the gradient of its projection function $G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)$. In detail, we encourage the partial derivative of $G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)$ with respect to s to approximate the magnitude of the change observed within the pre-trained pair $(\mathbf{x}_1, \widehat{\mathbf{x}}_0)$. In this way, the velocity loss function can be defined as:

$$\mathcal{L}_{\text{velocity}} = \mathbb{E}_{\mathbf{x}_1,t,s}\left[\left\|\partial G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)/\partial s - (\widehat{\mathbf{x}}_0 - \mathbf{x}_1)\right\|^2\right], \tag{2}$$

where t, s are sampled based on the step schedule.

Let $\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}_s, s)$ denote the velocity of the student trajectory at time step s and let $\mathbf{x}_t$ denote the value at initial step t, $G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)$ is equivalent to the solution of the ODE $\mathrm{d}\widehat{\mathbf{x}}_s/\mathrm{d}s = \boldsymbol{\mu}_{\boldsymbol{\phi}}(\widehat{\mathbf{x}}_s, s), \forall s \in [t, 1]$. That is:

$$\boxed{\widehat{\mathbf{x}}_s = G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s) = \mathbf{x}_t + \int_t^s \boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}_r, r)\mathrm{d}r \Rightarrow \frac{\partial G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)}{\partial s} = \boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}_s, s)} \tag{3}$$

Equation (2) can be alternatively understood as enforcing the student trajectory's velocity $\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}_s, s)$ close to the magnitude of changes in the pre-trained teacher model.

In fact, this alternative formulation of the velocity loss $\left\|\boldsymbol{\mu}_{\boldsymbol{\phi}}(\mathbf{x}_s, s) - (\widehat{\mathbf{x}}_0 - \mathbf{x}_1)\right\|^2$ is equivalent to that of the reflow distillation. By minimizing $\mathcal{L}_{\text{velocity}}$, SCoT achieves the same straightening effect on the student trajectory as reflow distillation. For reflow distillation, it learns the velocity of trajectory and thus needs to numerically solve ODEs to obtain the trajectory. SCoT directly learns the student trajectory through $G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)$ and avoids such ODE approximation errors. More importantly, SCoT can be regularized to satisfy the consistency requirement.

We compute the partial derivative $\partial G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)/\partial s$ using PyTorch's `torch.autograd.grad` function, which enables automatic differentiation of $G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)$ with respect to s.

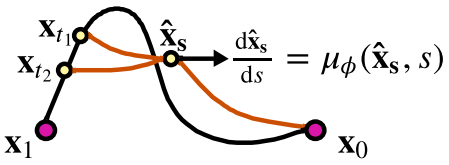

Figure 2: From two different points $\mathbf{x}_{t_1}, \mathbf{x}_{t_2}$, SCoT maps to the *same* point $\widehat{\mathbf{x}}_s$. The velocity $\boldsymbol{\mu}_{\boldsymbol{\phi}}(\widehat{\mathbf{x}}_s, s)$ at time step s is independent from previous time steps t_1, t_2.

Trajectory consistency SCoT also requires the trajectory to be consistent Song et al. (2023); Kim et al. (2024); Frans et al. (2025a). That is, the projection function $G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)$ maps the same future value to all points along the trajectory, regardless of their current time step as $G_{\boldsymbol{\phi}}(\mathbf{x}_{t_1}, t_1, s) = G_{\boldsymbol{\phi}}(\mathbf{x}_{t_2}, t_2, s)$, where $s < t_1 < t_2$. $\mathbf{x}_{t_1}$ is obtained through teacher model's ODE solver as $\mathbf{x}_{t_1} = \texttt{Solver}(\mathbf{x}_{t_2}, t_2, t_1; \boldsymbol{\theta}) = \mathbf{x}_{t_2} + \int_{t_2}^{t_1} \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}_r, r)\mathrm{d}r$, such that $\mathbf{x}_{t_1}$ and $\mathbf{x}_{t_2}$ are located on the same teacher trajectory.

We adopt the *soft-consistency loss* Kim et al. (2024) which is defined by comparing the mapped outputs $G_{\boldsymbol{\phi}}(\mathbf{x}_{t_2}, t_2, s) \approx G_{\text{sg}(\boldsymbol{\phi})}(\texttt{Solver}(\mathbf{x}_{t_2}, t_2, t_1; \boldsymbol{\theta}), t_1, s)$, where $\text{sg}(\cdot)$ is the exponential moving average stop-gradient operator. Following CTM Kim et al. (2024), these two mapping values are further projected to the clean image space to construct the consistency loss as:

$$\mathcal{L}_{\text{consistency}} = \mathbb{E}_{t_2\in[0,1],t_1\in[t_2,1],s\in[t_1,1],\mathbf{x}_{t_2}}\left[D(\mathbf{x}_{\text{target}}(\mathbf{x}_{t_2}, t_2, t_1, s), \mathbf{x}_{\text{est}}(\mathbf{x}_{t_2}, t_2, s))\right], \tag{4}$$

where $D(\cdot,\cdot)$ is the distance metric, $\mathbf{x}_{\text{est}}(\mathbf{x}_{t_2}, t_2, s) = G_{\text{sg}(\boldsymbol{\phi})}\left(G_{\boldsymbol{\phi}}(\mathbf{x}_{t_2}, t_2, s), s, 0\right)$ represents the estimated clean image obtained by projecting the point $\mathbf{x}_{t_2}$ forward to s using the student model $G_{\boldsymbol{\phi}}$, followed by decoding to time step 0 via the EMA model $G_{\text{sg}(\boldsymbol{\phi})}$, and $\mathbf{x}_{\text{target}}(\mathbf{x}_{t_2}, t_2, t_1, s) = G_{\text{sg}(\boldsymbol{\phi})}\left(G_{\text{sg}(\boldsymbol{\phi})}(\texttt{Solver}(\mathbf{x}_{t_2}, t_2, t_1; \boldsymbol{\theta}), t_1, s), s, 0\right)$ serves as the target output, computed by first integrating $\mathbf{x}_{t_2}$ to $\mathbf{x}_{t_1}$ along the teacher trajectory via the ODE solver, and then applying the same projection-decoding pipeline. This formulation ensures that both paths—although starting from different intermediate states on the same trajectory—should yield consistent outputs at the final clean image space.

This consistency ensures the "validity" of our projection function in the trajectory. Compared to the consistency models and their variants, it is expected that this trajectory straightness makes it easier to approximate the projection function than those with high curvatures.

Objective function Summarizing the above targets, SCoT's objective function may be written as:

$$\mathcal{L}_{\text{SCoT}} = \lambda_{\text{vel}}\mathcal{L}_{\text{velocity}} + \lambda_{\text{con}}\mathcal{L}_{\text{consistency}}, \tag{5}$$

where $\lambda_{\text{vel}}, \lambda_{\text{con}}$ denote the weighting factors for $\mathcal{L}_{\text{velocity}}, \mathcal{L}_{\text{consistency}}$. Each individual component addresses the straightness and consistency of trajectories, respectively. In this way, the proposed SCoT successfully unifies the consistency models and rectified flows.

The two loss terms, $\mathcal{L}_{\text{velocity}}$ and $\mathcal{L}_{\text{consistency}}$, have different dependencies on the pre-trained model. Specifically, $\mathcal{L}_{\text{velocity}}$ operates on a data-noise pair $(\widehat{\mathbf{x}}_0, \mathbf{x}_1)$, while $\mathcal{L}_{\text{consistency}}$ requires the model's parameters. To cope with their requirements, we propose generating the data point $\widehat{\mathbf{x}}_0$ dynamically for each training step using the pre-trained model $\widehat{\mathbf{x}}_0 = \mathbf{x}_1 + \int_1^0 \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)dt$. This avoids the need for pre-computed pairs. An additional potential improvement involves using a shared random noise vector $\mathbf{x}_1$ for both losses, which may enhance training consistency when sampling intermediate points. We leave a detailed study of this method to future work.

3.2 Training from scratch and Sampling

It is noted that SCoT may also be trained from scratch, by modifying consistency loss and randomly choosing noise $\mathbf{x}_1$ and data $\mathbf{x}_0$. Such exploration was not conducted due to resource constraints. Regarding sampling, Algorithm 1 outlines the SCoT sample generation process, which may be implemented as a multi-step or single step procedure. Table 2 and Table 3 demonstrate the corresponding experimental validation. Following the same setting as in Table 2 and Table 3, our model is trained with the DSM loss adopted from CTM to enhance sample quality and training stability.

3.3 Connections to Other Distillation Methods

Methods	**Reflow** Liu et al. (2023b)	**InstaFlow** Liu et al. (2023c)
Objective function	$\mathbb{E}_{\mathbf{x}_1,t}\left[\left\|\mathbf{v}_{\boldsymbol{\phi}}(\mathbf{x}_t,t) - (\widehat{\mathbf{x}}_0 - \mathbf{x}_1))\right\|^2\right]$	$\mathbb{E}_{\mathbf{x}_1}\left[\left\|\mathbf{v}_{\boldsymbol{\phi}}(\mathbf{x}_1,1) - (\widehat{\mathbf{x}}_0 - \mathbf{x}_1)\right\|^2\right]$
Methods	**FlowDreamer** Li et al. (2025)	$\mathcal{L}_{\text{velocity}}$ in **SCoT (Ours)**
Objective function	$\mathbb{E}_{\mathbf{x}_t}\left[\left\|\mathbf{v}_{\boldsymbol{\phi}}(\mathbf{x}_t,t) - \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}_t,t)\right\|^2\right]$	$\mathbb{E}_{\mathbf{x}_1,t,s}\left[\left\|\partial G_{\boldsymbol{\phi}}(\mathbf{x}_t,t,s)/\partial s - (\widehat{\mathbf{x}}_0 - \mathbf{x}_1)\right\|^2\right]$

Table 1: Different objective functions in enforcing straight student trajectory. In addition to Reflow and SCoT, InstaFlow enables one step sampling by learning the magnitude of changes based on random noise $\mathbf{x}_1$, whereas FlowDreamer aims to approximate teacher model's velocity $\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$.

The proposed SCoT bridges the gap between the consistency model and rectified flow distillations. On one hand, SCoT adopts the format of CTM Kim et al. (2024)'s project function $G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)$. SCoT's consistency loss and output reconstruction loss can be combined into CTM's integrated *soft-consistency loss*. While there is no restriction on velocity, approximating $G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)$ can be challenging when the student trajectory exhibits high curvature. As a result, multiple steps are usually required to generate high-quality images in CTM Kim et al. (2024).

Algorithm 1 Sampling Procedure of SCoT

Input: Trained SCoT projection $G_{\boldsymbol{\phi}}(\mathbf{x}_t, t, s)$; steps $t_N = 1 > \cdots > t_1 = 0$
Output: Generated image $\widehat{\mathbf{x}}_0$
1: Sample initial: $\widehat{\mathbf{x}}_{t_N} \sim \pi_1$
2: **for** $n = N, N-1, \ldots, 1$ **do**
3: $\quad \widehat{\mathbf{x}}_{t_{n-1}} = G_{\boldsymbol{\phi}}(\widehat{\mathbf{x}}_{t_n}, t_n, t_{n-1})$
4: **end for**
5: **return** $\widehat{\mathbf{x}}_0$

On the other hand, the velocity loss in SCoT shares the same target as Reflow Liu et al. (2023b). By enforcing constant velocity in Equation (2), SCoT induces a straightening effect on the student trajectory. Table 1 summarizes the different objective functions used to encourage straight student trajectories.

The Shortcut model Frans et al. (2025a) shares the same targets of trajectory consistency and straightness. While emphasizing learning trajectory velocities instead of the full trajectory, the Shortcut model still relies on an ODE solver for image sampling. Also, its usage of a one-step Euler solver to enforce trajectory consistency might not be optimal. Other concurrent works such

as MeanFlow Geng et al. (2025) and FlowMap Boffi et al. (2025); Sabour et al. (2025) have been discussed in Section 2.

Boot Gu et al. (2023) also considers the velocity alignment with the pre-trained model, by comparing the gradient of function. InstaFlow Liu et al. (2023c) and SlimFlow Zhu et al. (2025) focus on output reconstruction. As velocity and consistency are not regularized, the trajectory is not well defined.

4 Experiments

4.1 Experimental Setup

Teacher-Student Distillation Setup. To simulate the generation trajectory from $t = 0$ to $t = 1$, the pre-trained models are used to generate paired data samples $\langle \mathbf{x}_0, \mathbf{x}_1 \rangle$. Furthermore, to provide the intermediate states required for computing the consistency term in Equation (4), we distill knowledge from a teacher model $\boldsymbol{\theta}$ into a student model $\boldsymbol{\phi}$. Following established practices in CM Song et al. (2023), SlimFlow Zhu et al. (2025), and CTM Kim et al. (2024), we evaluate SCoT on both CIFAR-10 and ImageNet, using pre-trained diffusion checkpoints from EDM[2] (for CIFAR-10) and CM[3] (for ImageNet) as teacher models. For the student model, we adopt EDM's DDPM++ implementation on CIFAR-10, and the Ablated Diffusion Model (`ADM`) architecture from Dhariwal and Nichol (2021) for ImageNet. To support the additional time conditioning s in Equation (5), we incorporate several architectural modifications inspired by CTM. Specifically, we extend the temporal embedding by adding auxiliary s-embedding information to the original t-embedding. We also adopt other training heuristics from CTM (see Table 8), including: (1) using a larger value of μ in the stop-gradient EMA to slow the teacher update rate and improve training stability; (2) setting the student EMA rate to 0.999 to smooth parameter updates over time and reduce training noise; and (3) reusing skip connections from the pre-trained diffusion model to facilitate gradient flow and preserve hierarchical features.

Model Architectures. To support the additional time conditioning variable s in our generator $g_{\boldsymbol{\phi}}(\mathbf{x}, t, s)$, we incorporate a conditional embedding module into the baseline model architectures, following prior design choices in CM Song et al. (2023), SlimFlow Zhu et al. (2025), and CTM Kim et al. (2024). For CIFAR-10, we adopt the DDPM++ implementation from EDM Karras et al. (2022), extending its time embedding module to additionally encode s using a structure symmetric to the original t embedding, and applying identical normalization strategies. For ImageNet, we use the Ablated Diffusion Model (`ADM`) Dhariwal and Nichol (2021), where s is embedded jointly with the class conditioning variable c, requiring no architectural change. A detailed comparison of DDPM++ and ADM in terms of their ResNet backbones, attention configurations, and noise conditioning strategies is presented in Table 9. To improve the attention module, we address compatibility issues in `QKVFlashAttention` by modifying the dimension operations in `QKVAttentionLegacy` to match expected checkpoint formats. Additionally, we integrate `xformers`' `ScaledDotProduct` attention as an alternative backend. These improvements ensure correct weight loading while enhancing efficiency and flexibility of the attention mechanism.

Training and Sampling Hyperparameters. To ensure stable convergence and fair comparison during the training of SCoT, we adopt a higher learning rate for smaller datasets (e.g., CIFAR-10) and scale the batch size appropriately for larger datasets (e.g., ImageNet). We use the Adam optimizer for both settings, and enable mixed-precision training (FP16) to improve memory efficiency and computational speed. For CIFAR-10, we train the model for 130k iterations with a batch size of 512, while for ImageNet we use a larger batch size of 2048 and train for 40k iterations to accommodate the increased data complexity and training cost. We adopt an exponential moving average (EMA) of the student model with a decay rate of 0.9999 and employ a stop-gradient variant with EMA coefficient $\mu = 0.9999$ to stabilize the training objective. A complete list of hyperparameter settings is provided in Table 8.

[2] `https://github.com/NVlabs/edm`

[3] `https://github.com/openai/consistency_models`

NFE↓	Methods	Model Size			Generation Quality
		FLOPs (G)	MACs (G)	Params (MB)	FID↓
50	DDIM Song et al. (2021a)	12.2	6.1	35.7	4.67
20	DDIM Song et al. (2021a)	12.2	6.1	35.7	6.84
10	DDIM Song et al. (2021a)	12.2	6.1	35.7	8.23
10	DPM-solver-2 Lu et al. (2022)	12.2	6.1	35.7	5.94
10	DPM-solver-fast Lu et al. (2022)	12.2	6.1	35.7	4.70
10	3-DEIS Zhang and Chen (2022)	20.6	10.3	61.8	4.17
2	PD Salimans and Ho (2022)	41.2	20.6	55.7	5.58
2	CD Song et al. (2023)	41.2	20.6	55.7	2.93
2	CT Song et al. (2023)	41.2	20.6	55.7	5.83
2	CTM* Kim et al. (2024)	41.2	20.6	55.7	1.87
2	**SCoT**	41.2	20.6	55.7	2.30
1	1-Rectified Flow (+Distill) Liu et al. (2023a)	20.6	10.3	61.8	378
1	2-Rectified Flow (+Distill) Liu et al. (2023a)	20.6	10.3	61.8	12.21
1	3-Rectified Flow (+Distill) Liu et al. (2023a)	20.6	10.3	61.8	8.15
1	CTM* Kim et al. (2024)	41.2	20.6	55.7	1.90
1	**SCoT**	41.2	20.6	55.7	2.40

Table 2: Comparison of N-step (NFE) generation performance across diffusion models on CIFAR-10 at comparable model scales. We report sample quality metrics—FID (↓)-for $N \in \{1, 2, 10, 20, 50\}$. Entries highlighted in **bold** denote our proposed method. CTM* indicates the inclusion of GAN loss. Baseline results are sourced from Song et al. (2023); Zhu et al. (2025); Frans et al. (2025b). "–" indicates that the result is not available.

Datasets. For evaluation, we adopt two large-scale real-world datasets with different image resolutions: CIFAR-10 (32×32) and ImageNet (64×64), following standard protocol. Dataset statistics, including resolution, total size, and sample count, are summarized in Table 10.

Evaluation Metrics. We assess SCoT on unconditioned image generation using standard evaluation metrics, including Fréchet Inception Distance (FID) Heusel et al. (2017), Inception Score (IS) Salimans et al. (2016), and Recall (Rec.) Sajjadi et al. (2018). To evaluate model efficiency, we also report parameter count, floating-point operations (FLOPs), and multiply–accumulate operations (MACs). See Section C.2 for further details.

Time Efficiency. We measure the training throughput of the trajectory generator $g_{\phi}(\cdot)$ under different combinations of loss functions to evaluate time efficiency. Throughput results, reported in images per second per GPU, are summarized in Table 11.

4.2 Distillation Results

CIFAR-10. On CIFAR-10, SCoT achieves an FID of 2.30 with only 2 NFEs, outperforming baselines such as CD (FID 2.93) and CT (FID 5.83) while requiring fewer function evaluations. CTM attains a lower FID of 1.87 with just 2 NFE, but its results incorporate a GAN loss, which significantly improves sample fidelity by providing a strong adversarial signal during training. In contrast, we do not adopt the GAN loss due to its instability under our training configuration and resource constraints. Despite this, SCoT delivers competitive performance without adversarial training, demonstrating a favorable balance between generation quality and computational efficiency.

ImageNet. On ImageNet, SCoT achieves an FID of 2.60 with just 2 NFEs, outperforming baselines such as CD (FID 6.20) and PD (FID 15.39), and closely approaching the performance of EDM (FID 2.44). While CTM achieves the best FID of 1.92, this result benefits from the integration of a GAN loss, which enhances visual fidelity by introducing adversarial supervision during training. In contrast, our method does not incorporate GAN loss due to its instability under our training regime, yet still attains competitive sample quality. SCoT also demonstrates strong diversity, with a recall of 0.61 and an Inception Score of 68.20, highlighting its effectiveness in balancing quality, diversity, and computational cost.

NFE↓	Methods	Model Size			Generation Quality		
		FLOPs	MACs	Params	FID↓	Rec.↑	IS↑
250	ADM Karras et al. (2022)	—	—	—	2.07	0.63	—
79	EDM Dhariwal and Nichol (2021)	219.4	103.4	295.9	2.44	0.67	48.88
2	PD Salimans and Ho (2022)	219.4	103.4	295.9	15.39	0.62	—
2	CD Song et al. (2023)	219.4	103.4	295.9	6.20	0.63	40.08
2	CTM* Kim et al. (2024)	219.4	103.4	295.9	1.70	0.57	64.29
2	**SCoT**	219.4	103.4	295.9	2.60	0.61	68.20
1	CD Song et al. (2023)	219.4	103.4	295.9	6.20	0.63	–
1	CT Song et al. (2023)	219.4	103.4	295.9	13.00	0.47	–
1	SlimFlow Zhu et al. (2025)	67.8	31.0	80.7	12.34	—	—
1	Shortcut(unconditional) Frans et al. (2025b)	219.4	103.4	295.9	20.50	—	—
1	Shortcut(conditional) Frans et al. (2025b)	219.4	103.4	295.9	40.30	—	—
1	CTM* Kim et al. (2024)	219.4	103.4	295.9	1.92	0.57	64.29
1	**SCoT**	219.4	103.4	295.9	4.80	0.57	67.60

Table 3: Comparison of N-step generation performance by different DMs on ImageNet across corresponding model scales. Bold red numbers indicate the number of parameters for each distilled model. We report sample quality metrics—FID (↓), Rec. (↑), and IS (↑)—for $N \in \{1, 2, 10, 79, 250\}$. Entries highlighted in **bold** denote our proposed method. CTM* indicates the use of an additional GAN loss. Baseline results are taken from Song et al. (2023); Zhu et al. (2025); Frans et al. (2025b). "–" indicates that the result is not available.

4.3 Trajectory Analysis

We use CTM's soft consistency which proves to outperform local consistency and perform comparable to global consistency. Specifically, local consistency distills only 1-step teacher, so the teacher of time interval $[0, T - \Delta t]$ is not used to train the neural jump starting from $\mathbf{x}_T$. Rather, teacher on $[t - \Delta t, t]$ with $t \in [0, T - \Delta t]$ is distilled to student from neural jump starting from $\mathbf{x}_t$, not $\mathbf{x}_T$. The student, thus, has to extrapolate the learnt but scattered teacher across time intervals to estimate the jump from $\mathbf{x}_T$, which could potentially lead to imprecise estimation. In contrast, the amount of teacher to be distilled in soft consistency is determined by a random u, where μ = 0 represents distilling teacher on the entire interval $[0, T]$. Hence, soft matching serves as a computationally efficient and high-performing loss.

4.4 Consistency Guarantee

The results in Table 4 highlight the benefits of consistency. By adding this loss, SCoT is encouraged to focus on improving output reconstruction and consistency, which is reflected in the consistent reduction of FID from 15.7 to 5.6 in NFE = 1 and from 16.4 to 3.9 in NFE = 2. This optimization drives the model to generate samples closer to the target distribution. As a result, the IS increases from 30.4 to 63.1 in NFE = 1 and from 29.8 to 61.4 in NFE = 2, indicating notable improvements in both fidelity and diversity. Additionally, Precision and Recall show substantial gains rising from 0.49 to 0.72 and 0.45 to 0.57 in NFE = 1, respectively, demonstrating that the model captures more accurate and diverse samples.

4.5 Velocity Guarantee

The incorporation of the velocity guarantee loss, as shown in Table 5, further improves the generative trajectory by encouraging temporal smoothness. This constraint ensures that the trajectories are straightened, making the sampling process more stable and efficient. The benefits are evident in the substantial reduction of FID, dropping from 14.7 to 4.8 in NFE = 1 and from 15.2 to 3.6 in NFE = 2. The IS improves accordingly, reaching 67.6 and 68.2 in NFE = 1 and NFE = 2, respectively. The model also achieves stronger coverage and accuracy, with Precision improving from 0.56 to 0.71 and Recall from 0.54 to 0.58 in NFE = 1. Similar trends are observed in NFE = 2. These confirm that velocity regularization helps refine the generative path, enabling the model to produce high-quality samples with better alignment to the underlying data manifold, especially under reduced NFE.

Metric	NFE = 1				NFE = 2			
	5k	10k	15k	20k	5k	10k	15k	20k
FID (↓)	15.7	15.2	11.4	**5.6**	16.4	14.7	10.2	**3.9**
sFID (↓)	33.8	32.8	31.7	**16.2**	36.5	34.9	29.7	**13.6**
IS (↑)	30.4	34.2	43.6	**63.1**	29.8	33.7	42.2	**61.4**
Precision (↑)	0.49	0.55	0.64	**0.72**	0.47	0.53	0.65	**0.73**
Recall (↑)	0.45	0.49	**0.57**	0.54	0.44	0.47	**0.58**	0.56

Table 4: Comparison of training results for NFE = 1 and NFE = 2 for loss from Equation (4). All models are trained on ImageNet and tested on 6k samples. Bold values indicate the best performance per metric.

Metric	NFE = 1				NFE = 2			
	5k	10k	15k	20k	5k	10k	15k	20k
FID (↓)	14.7	14.1	10.4	**4.8**	15.2	13.8	9.1	**2.6**
sFID (↓)	32.0	31.0	29.9	**14.6**	35.4	33.7	28.3	**12.4**
IS (↑)	33.4	37.8	48.6	**67.6**	34.1	38.5	50.0	**68.2**
Precision (↑)	0.56	0.61	0.67	**0.71**	0.55	0.60	0.68	**0.73**
Recall (↑)	0.54	0.56	**0.58**	0.57	0.55	0.57	**0.59**	0.58

Table 5: Training Equation (5) Results for NFE = 1 and NFE = 2. For metrics with ↓, lower values are better; for metrics with ↑, higher values are better. Bold values represent the best performance in each metric.

4.6 Loss Weighting

In this section, we explore the impact of loss weighting parameters in our training objective defined by Equation (5), focusing specifically on λ_{con} for the consistency loss and λ_{vel} for the velocity loss.

Weighting on λ_{con} in Equation (5) We implemented an adaptive weighting strategy for λ_{con} to effectively balance the contributions between consistency and denoising losses during training. The primary motivation behind this adaptive mechanism is to dynamically adjust the weighting based on the relative importance and magnitude of each loss component. Specifically, this adjustment is made through gradient magnitude comparisons, ensuring that no single loss dominates excessively, thereby stabilizing training and improving convergence.

Weighting on λ_{vel} in Equation (5) To address the instability associated with the velocity loss defined in Equation (2), we explored several loss weighting strategies. The primary source of instability stems from the computation of second-order derivatives using automatic differentiation tools such as PyTorch's `torch.autograd.grad()`. These second-order computations require repeated differentiation, which can amplify small numerical inaccuracies through successive applications of the chain rule. As a result, accumulated numerical errors may lead to unstable gradient magnitudes, manifesting as gradient explosion or vanishing gradients, and ultimately impairing training stability. Motivated by these challenges, we investigated alternative weighting approaches to mitigate such instabilities.

The first strategy, **Adaptive Weighting**, dynamically adjusts the weight of the velocity loss based on the scale of individual loss elements. As shown in Table 6, this method performs relatively well in the early stages of training, but FID increases notably in later iterations. This indicates that while adaptive scaling can help stabilize early training, it may overemphasize certain loss components later on, leading to degraded generative quality.

The second strategy, **Without Scaling**, removes all adaptive scaling factors. This approach is motivated by the observation that unstable second-order gradients may produce extreme weight values, which can disrupt training. As evidenced in Table 6, this method leads to slightly worse performance at early iterations but achieves improved stability and better FID scores in later stages.

	5k	10k	15k	20k
Adaptive Weighting	15.3	14.5	11.2	6.2
Without Scaling	16.1	15.4	12.5	7.0
Normalization	15.2	13.8	9.1	3.6

Table 6: FID (↓) comparison across different velocity loss weighting strategies at various training iterations (5k, 10k, 15k, 20k), using a global batch size of 2048. **Adaptive Weighting** dynamically adjusts weights based on the relative scale of loss components; **Without Scaling** removes adaptive weighting entirely; and **Normalization** stabilizes training by bounding the magnitude of the velocity loss.

The final strategy, **Normalized Velocity Loss**, clips the adaptive weight within a predefined range $[0.01, 10]$ to prevent extreme values from dominating the loss. This normalization consistently achieves the best FID scores across all iterations, suggesting it provides a stable and balanced contribution of the velocity loss during optimization. Notably, after 20k iterations, this strategy yields the lowest FID, demonstrating its effectiveness in maintaining training stability and improving overall model performance.

5 Conclusion

In this paper, we propose the SCoT model that successfully bridges the gap between consistency model and rectified flows distillation. SCoT performs pre-trained diffusion model distillations, and produces straight and consistent trajectories for fast sampling. By aligning the output, velocity, and consistency with the teacher model, SCoT achieves high-quality generation with fewer sampling steps. Experimental results on CIFAR-10 and ImageNet show that SCoT outperforms existing distillation methods in both efficiency and generation quality. Ablation studies confirm the effectiveness of our design, highlighting the importance of trajectory consistency and velocity approximation. In the future, we aim to extend SCoT to high-resolution image synthesis and explore its integration with conditional generation tasks. We believe SCoT provides a strong foundation for efficient generative modeling and paves the way for real-time high-fidelity image synthesis.

6 Acknowledgments and Disclosure of Funding

This work is sponsored by AMD's High Performance Compute Fund project. This work is also partially sponsored by the Australian Research Council Discovery grant DP240102050, ARC LIEF grant LE240100131, and ARC Linkage grant LP230201022.

References

Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic Interpolants: A Unifying Framework for Flows and Diffusions. *Journal of Machine Learning Research*, 2023.

Nicholas M. Boffi, Michael S. Albergo, and Eric Vanden-Eijnden. Flow map matching with stochastic interpolants: A mathematical framework for consistency models, 2025.

Prafulla Dhariwal and Alexander Nichol. Diffusion Models Beat GANs on Image Synthesis. *Advances in Neural Information Processing Systems*, 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 Rlow Transformers for High-Resolution Image Synthesis. *International Conference on Machine Learning*, 2024.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One Step Diffusion via Shortcut Models. *International Conference on Learning Representations*, 2025a.

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One Step Diffusion via Shortcut Models. *International Conference on Learning Representations*, 2025b.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling, 2025.

Jiatao Gu, Shuangfei Zhai, Yizhe Zhang, Lingjie Liu, and Joshua M Susskind. Boot: Data-free Distillation of Denoising Diffusion Models with Bootstrapping. *International Conference on Machine Learning*, 2023.

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*, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. *Advances in Neural Information Processing Systems*, 2020.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the Design Space of Diffusion-based Generative Models. *Advances in Neural Information Processing Systems*, 2022.

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. *International Conference on Learning Representations*, 2024.

Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved Precision and Recall Metric for Assessing Generative Models. *Advances in Neural Information Processing Systems*, 2019.

Hangyu Li, Xiangxiang Chu, Dingyuan Shi, and Wang Lin. Flowdreamer: Exploring High Fidelity Text-to-3D Deneration via Rectified Flow. *International Conference on Learning Representations*, 2025.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. *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. *International Conference on Learning Representations*, 2023a.

Xingchao Liu, Chengyue Gong, and qiang liu. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. *International Conference on Learning Representations*, 2023b.

Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, et al. Instaflow: One Step is enough for High-quality Diffusion-based Text-to-Image Generation. *International Conference on Learning Representations*, 2023c.

Cheng Lu and Yang Song. Simplifying, Stabilizing and Scaling Continuous-time Consistency Models. *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*, 2022.

Artem Lukoianov, Haitz Sáez de Ocáriz Borde, Kristjan Greenewald, Vitor Campagnolo Guizilini, Timur Bagautdinov, Vincent Sitzmann, and Justin Solomon. Score Distillation via Reparametrized DDIM. *Advances in Neural Information Processing Systems*, 2024.

Weijian Luo, Tianyang Hu, Shifeng Zhang, Jiacheng Sun, Zhenguo Li, and Zhihua Zhang. Diff-Instruct: A Universal Approach for Transferring Knowledge From Pre-trained Diffusion Models. *Advances in Neural Information Processing Systems*, 2024.

Thuan Hoang Nguyen and Anh Tran. Swiftbrush: One-Step Text-to-Image Diffusion Model with Variational Score Distillation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3D using 2D Diffusion. *International Conference on Learning Representations*, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution Image Synthesis with Latent Diffusion Models. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet Large Scale Visual Recognition Challenge. *IJCV*, 2015.

Amirmojtaba Sabour, Sanja Fidler, and Karsten Kreis. Align your flow: Scaling continuous-time flow map distillation, 2025.

Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing Generative Models via Precision and Recall. *Advances in Neural Information Processing Systems*, 2018.

Tim Salimans and Jonathan Ho. Progressive Distillation for Fast Sampling of Diffusion Models. *International Conference on Learning Representations*, 2022.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved Techniques for Training GANs. *Advances in Neural Information Processing Systems*, 2016.

Axel Sauer, Frederic Boesel, Tim Dockhorn, Andreas Blattmann, Patrick Esser, and Robin Rombach. Fast High-Resolution Image Synthesis with Latent Adversarial Diffusion Distillation. *SIGGRAPH*, 2024.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial Diffusion Distillation. *European Conference on Computer Vision*, 2025.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising Diffusion Implicit Models. *International Conference on Learning Representations*, 2021a.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based Generative Modeling through Stochastic Differential Equations. *International Conference on Learning Representations*, 2021b.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency Models. *International Conference on Machine Learning*, 2023.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016.

Zhendong Wang, Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou. Diffusion-GAN: Training GANs with Diffusion. *International Conference on Learning Representations*, 2023.

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*, 2024.

Sirui Xie, Zhisheng Xiao, Diederik P Kingma, Tingbo Hou, Ying Nian Wu, Kevin Patrick Murphy, Tim Salimans, Ben Poole, and Ruiqi Gao. Em Distillation for One-Step Diffusion Models. *Advances in Neural Information Processing Systems*, 2024.

Yanwu Xu, Yang Zhao, Zhisheng Xiao, and Tingbo Hou. UFOGen: You Forward Once Large Scale Text-to-Image Generation via Diffusion GANs. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 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*, 2024a.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-Step Diffusion with Distribution Matching Distillation. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024b.

Qinsheng Zhang and Yongxin Chen. Fast Sampling of Diffusion Models with Exponential Integrator. *International Conference on Learning Representations*, 2022.

Mingyuan Zhou, Huangjie Zheng, Zhendong Wang, Mingzhang Yin, and Hai Huang. Score Identity Distillation: Exponentially Fast Distillation of Pretrained Diffusion Models for One-Step Generation. *International Conference on Machine Learning*, 2024.

Mingyuan Zhou, Huangjie Zheng, Yi Gu, Zhendong Wang, and Hai Huang. Adversarial Score Identity Distillation: Rapidly Surpassing the Teacher in One Step. *International Conference on Learning Representations*, 2025.

Yuanzhi Zhu, Xingchao Liu, and Qiang Liu. Slimflow: Training Smaller One-Step Diffusion Models with Rectified Flow. *European Conference on Computer Vision*, 2025.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

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

   Answer: [Yes]

   Justification: Abstract and introduction accurately describe SCoT's dual objectives (consistency and straightness) and summarize experimental results on CIFAR-10 and ImageNet (Abstract; Sec. 1).

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

Justification: Conclude in Conclusion.

Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

   Justification: We contribute to the theory in distillation.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: All hyperparameters, model architectures, data splits, and training details are fully specified in Sec. 5 and Appendix A.2–A.4.

   Guidelines:

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

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

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

   Answer: [Yes]

   Justification: Public data and code is in anonymous repo.

   Guidelines:

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

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Data splits, optimizer settings, learning rates, batch sizes, and evaluation metrics are detailed in Sec. 5.1 and App. A.2–A.4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Each group of experiments was repeated multiple times to eliminate the randomness.

   Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: GPU type, throughput measurements, and total training iterations are specified in Sec. 5.1 and App. A.5.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: Research conforms to NeurIPS Ethics Guidelines; no human or sensitive data used.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [No]

    Justification: No discussion of potential societal impacts included.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [NA]

Justification: No high-risk models or data releases involved.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

Answer: [NA]

Justification: Third-party model checkpoints and datasets are cited but licenses not explicitly stated.

Guidelines:

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

13. **New assets**

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

Answer: [NA]

Justification: No new datasets or code assets are released.

Guidelines:

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

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

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

Answer: [NA]

Justification: No human subjects research conducted.

Guidelines:

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

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

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

Answer: [No]

Justification: LLMs not used in core methodology.

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [No]

Justification: LLMs not used in core methodology.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Contents

A Theoretical Motivation for Loss Design

To better understand the design of our loss function, we provide a brief explanation of why combining velocity and consistency objectives helps the model perform better.

The trajectory projection function is defined as

$$\hat{\mathbf{x}}_s = G_\phi(\mathbf{x}_t, t, s) = \frac{s}{t}\mathbf{x}_t + \left(1 - \frac{s}{t}\right) g_\phi(\mathbf{x}_t, t, s), \tag{6}$$

where g_ϕ is a neural network that learns a residual correction. The first term encourages a straight path from $\mathbf{x}_t$ to the target, while the second term provides flexibility.

The velocity loss encourages the trajectory to follow a constant direction:

$$\mathcal{L}_{\text{velocity}} = \mathbb{E}_{x_1,t,s}\left[\left\|\frac{\partial G_\phi}{\partial s} - (\hat{\mathbf{x}}_0 - \mathbf{x}_1)\right\|^2\right]. \tag{7}$$

The consistency loss encourages points from different timesteps to align at the same target point:

$$\mathcal{L}_{\text{consistency}} = \mathbb{E}_{t_1,t_2,s}\left[\|G_\phi(\mathbf{x}_{t_1}, t_1, s) - G_\phi(\mathbf{x}_{t_2}, t_2, s)\|^2\right]. \tag{8}$$

These two losses serve different purposes. The velocity loss reduces trajectory curvature, making it easier to approximate with fewer steps. The consistency loss ensures that the mapping is valid across time. Together, they help improve both the quality and efficiency of generation, especially in low-step settings like 1-step or 2-step sampling.

B Related Work

Pre-trained diffusion distillation methods have emerged as a powerful strategy to alleviate the significant computational burden inherent in diffusion models, which traditionally require a large number of function evaluations to produce high-quality samples. By compressing the iterative denoising process into far fewer steps, these techniques not only expedite sample generation but also make it feasible to deploy such models in low-resource environments. In this work, we systematically categorize these distillation approaches into three major groups based on their underlying objectives and operational paradigms.

Output reconstruction based methods: These aim to minimize the discrepancy between the outputs of the teacher (i.e., the pre-trained diffusion model) and the student model by directly reconstructing image outputs. Some approaches, such as Progressive Distillation Salimans and Ho (2022), focus on aligning output values by enforcing a close correspondence between the denoising steps of the teacher and the student. Other methods, for instance SDS and its variants Poole et al. (2023); Wang et al. (2024); Yin et al. (2024b,a), concentrate on matching output distributions to preserve the statistical characteristics of generated images. Additionally, certain studies operate in a one-step denoising image space Lukoianov et al. (2024); Karras et al. (2022), allowing for the direct generation of high-quality images from a single function evaluation, while others employ Fisher divergence objectives Zhou et al. (2024, 2025) to more rigorously align the gradients of score functions. Together, these techniques effectively reduce the number of sampling steps required while maintaining the fidelity of generated outputs.

Trajectory distillation based methods: Instead of concentrating solely on the final output, trajectory-based methods focus on the entire denoising path—from the initial random noise to the eventual clean image. By distilling the full trajectory, these approaches ensure that the student model replicates not only the final result but also the dynamic behavior of the teacher model throughout the diffusion process. Consistency distillation techniques Song et al. (2023); Kim et al. (2024); Lu and Song (2025) emphasize the self-consistency of the denoising trajectory, ensuring stable and accurate progression across different time steps. In contrast, rectified flow distillation methods such as InstaFlow and SlimFlow Liu et al. (2023c); Zhu et al. (2025) focus on producing a straighter, more direct trajectory, thereby mitigating the accumulation of approximation errors that typically arise from curved paths. Moreover, recent studies have demonstrated that integrating consistency modeling directly into rectified flows Frans et al. (2025a) can further enhance the fidelity of generated trajectories, effectively combining the strengths of both approaches.

Adversarial distillation based methods: This category of distillation methods leverages adversarial learning to refine the student model's output distribution. By incorporating an adversarial loss—often implemented via a pre-trained classifier or discriminator, these methods drive the student model to more closely approximate the target distribution provided by the teacher. Notably, studies such as Sauer et al. (2024, 2025) have successfully employed this strategy to achieve competitive performance with significantly fewer sampling steps. The adversarial framework not only enhances the perceptual quality of generated images but also provides a flexible plug-and-play mechanism that can complement other distillation strategies.

Notation	Description
ϕ	Parameters of the generator/student model
θ	Parameters of the velocity/teacher model
t	Current time step ($t \in [0, 1]$)
s	Target time step for state projection
N	Number of discretized steps/evaluations
π_0	Data distribution at initial state ($\mathbf{t} = 0$)
π_1	Noise distribution at terminal state ($\mathbf{t} = 1$)
$g(\cdot)$	Integration approximation function
$\mathbf{v}(\cdot)$	Velocity prediction function
$G(\cdot)$	Distilled generator function

Table 7: Notations and corresponding descriptions used in training SCoT.

	Hyperparameters	CIFAR-10	ImageNet
Optimization	Learning rate	0.0004	0.000008
	Batch	512	2048
	Student's stop-grad EMA parameter μ	0.9999	0.9999
	Optimizer	Adam	Adam
	N	18	40
	Training iterations	130K	40K
	Mixed-Precision (FP16)	Enabled	Enabled
Score	ODE Solver	Heun	Heun
	EMA decay rate	0.999	0.999

Table 8: Training configuration of SCoT in different model sizes on CIFAR-10 and ImageNet.

C Technical Details

C.1 Datasets

CIFAR-10 [4]: This dataset contains 60,000 32×32 color images evenly distributed over 10 distinct classes (6,000 images per class). It is split into a training set of 50,000 images and a test set of 10,000 images, making it a standard benchmark in machine learning and computer vision.

ImageNet [5]: ImageNet is one of the most influential benchmarks in computer vision. It comprises over 1.2 million training images and around 50,000 validation images, categorized into 1,000 diverse classes. The dataset's vast scale and rich annotations have made it an essential resource for developing and evaluating deep learning models.

C.2 Metrics

Fréchet Inception Distance (FID). FID evaluates the similarity between real and generated samples by modeling their feature distributions as multivariate Gaussians in a pre-trained InceptionV3 feature

[4] `https://www.cs.toronto.edu/~kriz/cifar.html`

[5] `http://www.image-net.org/`

Component	Functionality	DDPM++ (CIFAR-10)	ADM (ImageNet)
ResNet Block	Downsampling method	Resize + Conv	Strided Conv
	Upsampling method	Resize + Conv	Transposed Conv
	Time embedding type	Fourier	Positional
	Normalization type	GroupNorm	LayerNorm
	Residual blocks per resolution	2	3
Attention Module	Applied resolutions	16	32, 16, 8
	Attention heads	1	1–8–12
	Attention blocks (Down)	2	9
	Attention blocks (Up)	1	13
	Attention implementation	Vanilla	Multi-head self-attention
Conditioning	Label embedding	None	Learned class embedding
	Extra temporal input (s)	Additive embedding	Additive embedding
	Positional encoding added	No	Yes
	Skip connections	Present	Present
Output Head	Output scaling	Yes	Yes
	Activation	Identity	Identity

Table 9: Comparison of U-Net architectural components used in DDPM++ (CIFAR-10) and ADM (ImageNet) within our distillation framework. Differences include normalization strategies, attention configuration, conditioning, and up/downsampling methods.

Data Shape	Dataset	Samples	Data Size
$(3 \times 32 \times 32)$	CIFAR-10	60K	160M
$(3 \times 64 \times 64)$	FFHQ-64	70K	5 GB
	ImageNet	1.2M	100 GB

Table 10: Experimental details of datasets.

space and computing the Fréchet distance. This metric jointly captures fidelity and diversity. We adopt `clean-fid` [6] for FID computation, which standardizes image preprocessing and Inception activations, thereby improving reproducibility across experiments.

Sliced Fréchet Inception Distance (sFID). sFID is a computationally efficient variant of FID that approximates the Fréchet distance using one-dimensional projections of feature embeddings. Instead of computing the full covariance matrix, sFID calculates the Wasserstein distance between sliced marginal distributions in the InceptionV3 feature space, offering a lightweight and scalable measure of generation quality. We follow the same evaluation protocol as `clean-fid` [7], ensuring consistency in feature extraction and preprocessing.

Precision and Recall[8]. We adopt the manifold-based metrics introduced in Kynkäänniemi et al. (2019) to assess sample fidelity and diversity. Precision quantifies the fraction of generated samples that fall within the manifold of real data, while Recall measures how much of the real data distribution is covered by generated samples. The manifolds are approximated via k-nearest neighbors in the InceptionV3 feature space. Our implementation follows the version in the ADM repository [9].

[6] `https://github.com/GaParmar/clean-fid`
[7] `https://github.com/GaParmar/clean-fid`
[8] `https://github.com/kynkaat/improved-precision-and-recall-metric`
[9] `https://github.com/openai/guided-diffusion`

Inception Score (IS)[10]**.** IS reflects both sample quality and class diversity by computing the KL divergence between the conditional class distribution and the marginal class distribution over all samples. It is calculated using logits from an InceptionV3 model Szegedy et al. (2016) trained on ImageNet Russakovsky et al. (2015). Higher scores suggest high-confidence predictions and class diversity. However, when evaluating datasets with limited categorical variation (e.g., CelebA or FFHQ), IS primarily reflects fidelity. We follow the ADM implementation [3] and evaluate IS using 10k generated images.

Parameter Counts (`MB`). We report the total number of learnable parameters in the model, measured in megabytes. This includes weights and biases from all trainable layers, such as convolutional filters and dense layer matrices. Higher parameter counts may indicate stronger representational capacity, while smaller models are better suited for deployment under resource constraints. For transformer-based architectures, we include attention and feed-forward components.

Floating Point Operations (FLOPs). FLOPs measure the total number of floating-point operations (additions and multiplications) required during a forward pass, serving as a proxy for computational cost. For a convolutional layer of kernel size $k \times k$ over a feature map of spatial size $H \times W$, the FLOPs are computed as:

$$\text{FLOPs} = 2 \times H \times W \times k^2 \times C_{\text{in}} \times C_{\text{out}}.$$

We compute FLOPs using the `calflops` utility [11], providing an estimate of overall model complexity.

Multiply–Accumulate Operations (MACs). MACs count the number of fused multiply-add computations, which are often optimized as single hardware instructions on modern accelerators. For the same convolutional layer, MACs are given by:

$$\text{MACs} = H \times W \times k^2 \times C_{\text{in}} \times C_{\text{out}}.$$

We report MACs using the same tool [2], as they offer a hardware-aware indicator of inference cost, especially relevant for edge deployment.

D Additional Experiments

D.1 Time Efficiency

Table 11 concludes the throughput in training different loss function of our SCoT.

Dataset	Equation (2)	Equation (4)
CIFAR-10	282.3	71.5
ImageNet	131.7	31.7

Table 11: Time efficiency comparison (imgs/sec. on H100 GPU) of SCoT under different loss functions across datasets.

D.2 Sample Comparisons

Figure 3, Figure 4, and Figure 5 present the qualitative results of 1-step image generation on ImageNet using the SCoT sampler described in Algorithm 1. These samples are taken at different stages of training to illustrate the progressive refinement of the model. Figure 3 shows samples generated at the initial stage of training, where the model produces blurry images with limited semantic structure. As training progresses to 10k steps (Figure 4), the generated images become more coherent, with clearer object boundaries and texture. At 30k steps (Figure 5), the model generates high-quality, semantically consistent images, demonstrating the effectiveness of the training process. To ensure efficient evaluation and maintain consistent comparison across stages, we generate 6,000 samples at each step. Following the same setting as in Table 2 and Table 3, our model is trained with the DSM loss adopted from CTM Kim et al. (2024) to enhance sample quality and training stability.

[10] `https://github.com/openai/improved-gan`

[11] `https://github.com/MrYxJ/calculate-flops.pytorch`

Figure 3: 1-Step generation on the initial training stage for ImageNet by SCoT Algorithm 1 sampler.

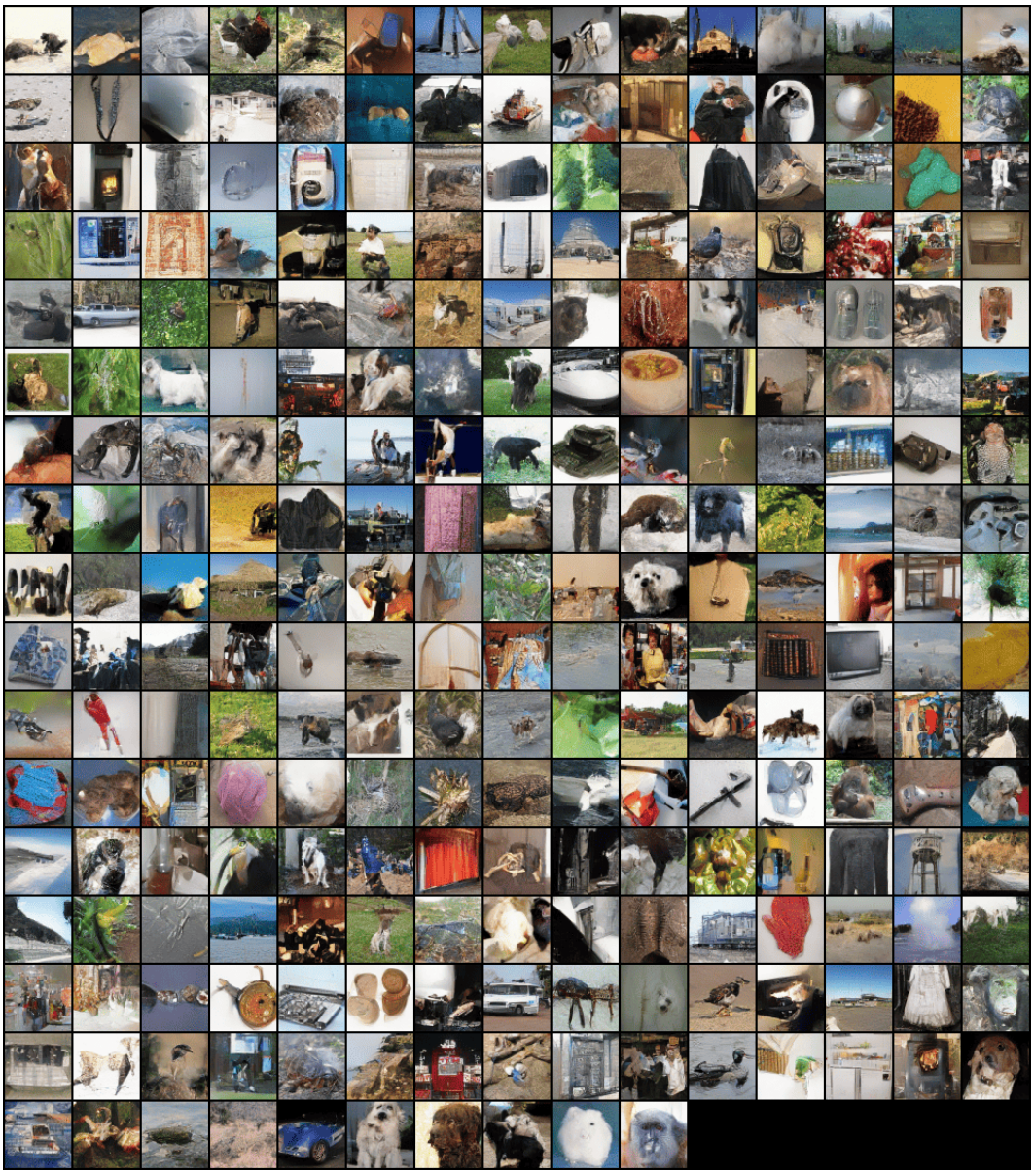

Figure 4: 1-Step generation on the 10k training stage for ImageNet by SCoT Algorithm 1 sampler.

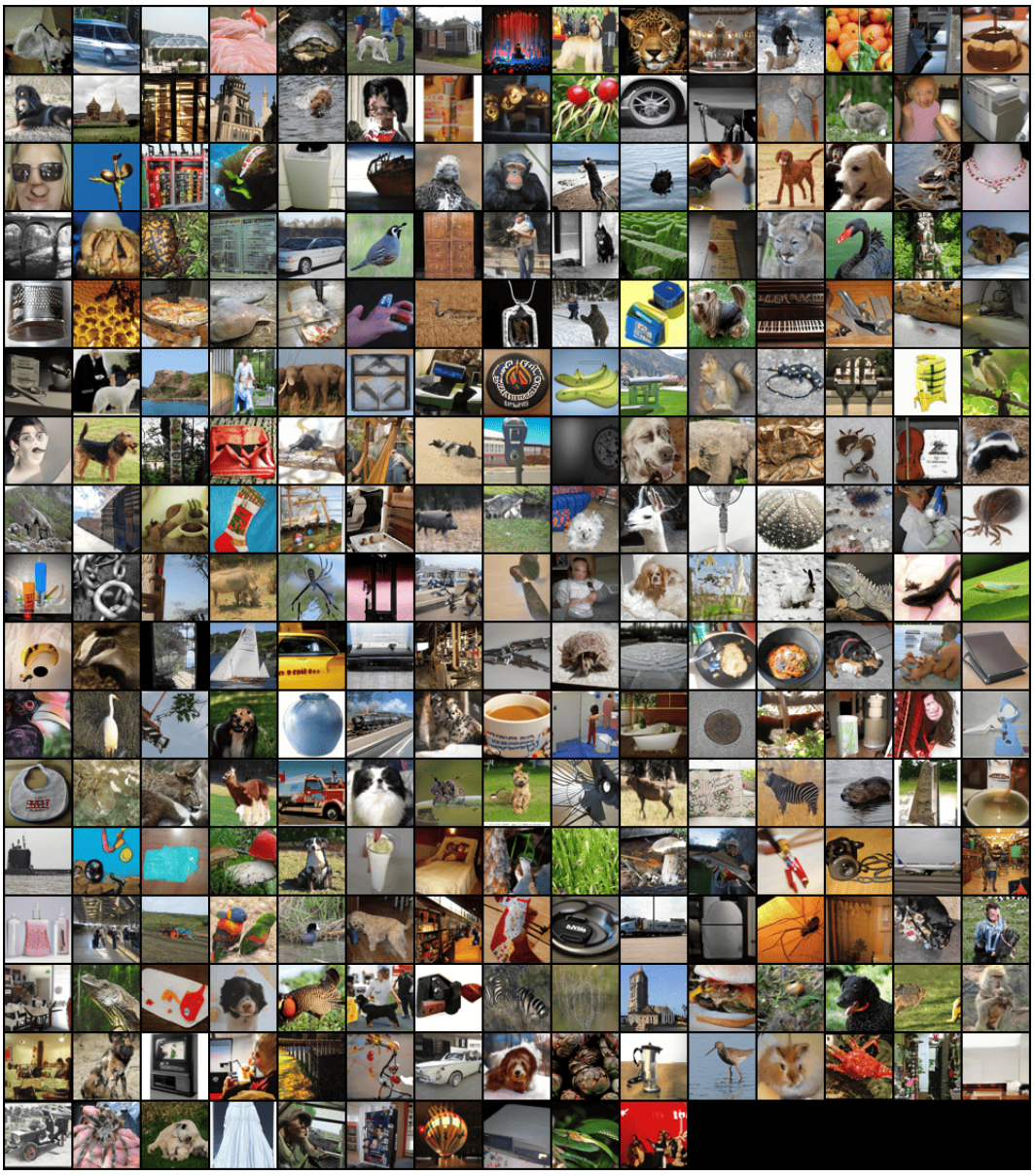

Figure 5: 1-Step generation on the 30k training stage for ImageNet by SCoT Algorithm 1 sampler.