Noise Consistency Training: A Native Approach for One-Step Generator in Learning Additional Controls

Yihong Luo[1,*] **Shuchen Xue**[2,*] **Tianyang Hu**[3,†] **Jing Tang**[4,1,†]
[1]HKUST [2]UCAS [3]CUHK(SZ) [4]HKUST(GZ)

Abstract

The pursuit of efficient and controllable high-quality content generation remains a central challenge in artificial intelligence-generated content (AIGC). While one-step generators, enabled by diffusion distillation techniques, offer excellent generation quality and computational efficiency, adapting them to new control conditions—such as structural constraints, semantic guidelines, or external inputs—poses a significant challenge. Conventional approaches often necessitate computationally expensive modifications to the base model and subsequent diffusion distillation. This paper introduces Noise Consistency Training (NCT), a novel and lightweight approach to directly integrate new control signals into pre-trained one-step generators without requiring access to original training images or retraining the base diffusion model. NCT operates by introducing an adapter module and employs a *noise consistency loss* in the noise space of the generator. This loss aligns the adapted model's generation behavior across noises that are conditionally dependent to varying degrees, implicitly guiding it to adhere to the new control. Theoretically, this training objective can be understood as minimizing the distributional distance between the adapted generator and the conditional distribution induced by the new conditions. NCT is modular, data-efficient, and easily deployable, relying only on the pre-trained one-step generator and a control signal model. Extensive experiments demonstrate that NCT achieves state-of-the-art controllable generation in a single forward pass, surpassing existing multi-step and distillation-based methods in both generation quality and computational efficiency.

1 Introduction

The pursuit of high-quality, efficient, and controllable generation has become a central theme in the advancement of artificial intelligence-generated content (AIGC). The ability to create diverse and realistic content is crucial for a wide range of applications, from art and entertainment to scientific visualization and data augmentation. Recent breakthroughs in diffusion models and their distillation techniques have led to the development of highly capable one-step generators [1, 2, 3, 4, 5]. These models offer a compelling combination of generation quality and computational efficiency, significantly reducing the cost of content creation. Methods such as Consistency Training [6] and Inductive Moment Matching [7] have further expanded the landscape of native few-step or even one-step generative models, providing new tools and perspectives for efficient generation.

However, as AIGC applications continue to evolve, new scenarios are constantly emerging that demand models to adapt to novel conditions and controls. These conditions can take many forms, encompassing structural constraints (e.g., generating an image with specific edge arrangements), semantic guidelines (e.g., creating an image that adheres to a particular artistic style), and external

[*]Core contribution.
[†]Corresponding authors: Tianyang Hu and Jing Tang.

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

factors such as user preferences or additional sensory inputs (e.g., generating an image based on a depth map). Integrating such controls effectively and efficiently is a critical challenge.

The conventional approach to incorporating controls into diffusion models often involves modifying the base model architecture and subsequently performing diffusion distillation to obtain a one-step student model [8]. This process, while effective, can be computationally expensive and time-intensive, requiring significant resources and development time. A more efficient alternative would be to extend the distillation pipeline to accommodate new controls directly, potentially bypassing the need for extensive retraining of the base diffusion model [9]. However, even extending the distillation pipeline can still be a heavy undertaking, adding complexity and computational overhead. Therefore, the question of how to directly endow one-step generators with new controls in a lightweight and efficient manner remains a significant challenge.

In this paper, we answer this question by proposing Noise Consistency Training (NCT) — a simple yet powerful approach that enables a pre-trained one-step generator to incorporate new conditioning signals without requiring access to training images or retraining the base model. NCT achieves this by introducing an adapter module that operates in the noise space of the pre-trained generator. Specifically, we define a noise-space consistency loss that aligns the generation behavior of the adapted model across different noise levels, implicitly guiding it to satisfy the new control signal. Besides, we employ a boundary loss ensuring that when given a condition already associated with input noise, the generation should remain the same as one-step uncontrollable generation. This can ensure the distribution of the adapter generator remains in the image domain rather than collapsing. Theoretically, we demonstrate in Section 3.2 that this training objective can be understood as matching the adapted generator to the intractable conditional induced by a discriminative control model when the boundary loss is satisfied, effectively injecting the desired conditioning behavior.

Our method is highly modular, data-efficient, and easy to deploy, requiring only the pre-trained one-step generator and a control signal model, without the need for full-scale diffusion retraining or access to the original training data. Extensive experiments across various control scenarios demonstrate that NCT achieves state-of-the-art controllable generation in a single forward pass, outperforming existing multi-step and distillation-based methods in both quality and computational efficiency.

2 Preliminary

Diffusion Models (DMs). DMs [2, 1] operate via a forward diffusion process that incrementally adds Gaussian noise to data $\mathbf{x}$ over T timesteps. This process is defined as $q(\mathbf{x}_t|\mathbf{x}) \triangleq \mathcal{N}(\mathbf{x}_t; \alpha_t\mathbf{x}, \sigma_t^2\mathbf{I})$, where α_t and σ_t are hyperparameters dictating the noise schedule. The diffused samples are obtained via $\mathbf{x}_t = \alpha_t\mathbf{x} + \sigma_t\epsilon$, with $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The diffusion network, ϵ_θ is trained by denoising: $\mathbb{E}_{\mathbf{x},\epsilon,t}||\epsilon_\theta(\mathbf{x}_t, t) - \epsilon||_2^2$. Once trained, generating samples from DMs typically involves iteratively solving the corresponding diffusion stochastic differential equations (SDEs) or probability flow ordinary differential equations (PF-ODEs), a process that requires multiple evaluation steps.

ControlNet. Among other approaches for injecting conditions [10, 11, 12, 13, 14], ControlNet [15] has emerged as a prominent and effective technique for augmenting pre-trained DMs with additional conditional controls. Given a pre-trained diffusion model ϵ_θ, ControlNet introduces an auxiliary network, parameterized by ϕ. This network is trained by minimizing a conditional denoising loss $L(\phi)$ to inject the desired controls:

$$L(\phi) = \mathbb{E}_{\mathbf{x},\epsilon,t}||\epsilon - \epsilon_{\theta,\phi}(\mathbf{x}_t, \mathbf{c})||_2^2. \tag{1}$$

After training, ControlNet enables the integration of new controls into the pre-trained diffusion models.

Maximum Mean Discrepancy. Maximum Mean Discrepancy (MMD [16]) between distribution $p(\mathbf{x}), q(\mathbf{y})$ is an integral probability metric [17]:

$$\mathrm{MMD}^2(p, q) = ||\mathbb{E}_{\mathbf{x}}[\psi(\mathbf{x})] - \mathbb{E}_{\mathbf{y}}[\psi(\mathbf{y})]||^2, \tag{2}$$

where $\psi(\cdot)$ is a kernel function.

Diffusion Distillation. While significant advancements have been made in training-free acceleration methods for DMs [18, 19, 20, 21, 22], diffusion distillation remains a key strategy for achieving high-quality generation in very few steps. Broadly, these distillation methods follow two primary paradigms: 1) Trajectory distillation [23, 24, 25, 26, 27, 28], which seeks to replicate the teacher

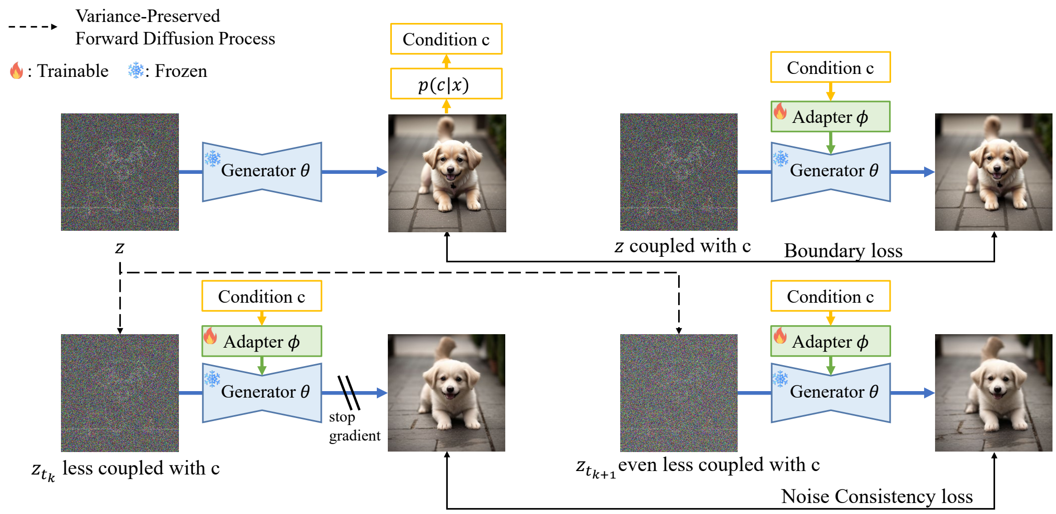

Figure 1: Framework description of our proposed **NCT**. We note that *we deliberately added some structural features to the noise to enhance readability*, rather than faithfully rendering Gaussian noise.

model's ODE trajectories on an instance-by-instance basis. These methods can encounter difficulties with precise instance-level matching. 2) Distribution matching, often realized via score distillation [5, 3, 29, 4], which aims to align the output distributions of the student and teacher models using divergence metrics. Our work utilizes a pre-trained one-step generator, which itself is a product of diffusion distillation; however, the training of our proposed NCT method does not inherently require diffusion distillation.

Additional Controls for One-step Diffusion. The distillation of multi-step DMs into one-step generators, particularly through score distillation, is an established research avenue [3, 5, 29]. However, the challenge of efficiently incorporating new controls into these pre-trained one-step generators is less explored. CCM [30], for example, integrates consistency training with ControlNet, demonstrating reasonable performance with four generation steps. In contrast, our work aims to surpass standard ControlNet performance in most cases using merely a single step. Many successful score distillation techniques [3, 5, 31, 32] rely on initializing the one-step student model with the weights of the teacher model. SDXS [8] explored learning controlled one-step generators via score distillation, but their framework requires both the teacher model and the generated "fake" scores to possess a ControlNet compatible with the specific condition being injected. JDM [9] minimizes a tractable upper bound of the joint KL divergence, which can teach a controllable student with an uncontrollable teacher. Generally, prior works are built on specific distillation techniques for adapting controls to one-step models. We argue that given an already proficient pre-trained one-step generator, performing an additional distillation for adding new controls is computationally expensive and unnecessary. However, *how to develop a **native** technique for one-step generators remains unexplored.* Our work takes the first step in designing a native approach for one-step generators to add new controls to one-step generators without requiring any diffusion distillation.

3 Method

Problem Setup. Let $\mathbf{z} \in \mathbb{R}^m$ be a latent variable following a standard Gaussian density $p(\mathbf{z})$. We have a pre-trained generator $f_\theta : \mathbb{R}^m \to \mathbb{R}^n$ that maps $\mathbf{z}$ to a data sample $\mathbf{x} = f_\theta(\mathbf{z})$. The distribution of these generated samples has a density $p_\theta(\mathbf{x})$, providing a high-quality approximation of the data distribution, such that $p_\theta(\mathbf{x}) \approx p_d(\mathbf{x})$. For any $\mathbf{x}$, there is a conditional probability density $p(\mathrm{c}|\mathbf{x})$ specifying the likelihood of condition c given $\mathbf{x}$. Our goal is to directly incorporate additional control c for a pre-trained one-step generator with additional trainable parameters ϕ (e.g. a ControlNet). More specifically, we aim to train a conditional generator $f_{\theta,\phi}(\mathbf{z}, \mathrm{c})$ that, when given a latent code $\mathbf{z}$ sampled from a standard Gaussian distribution and an independently sampled condition c, produces a sample $\mathbf{x}$ such that the joint distribution of $(\mathbf{x}, \mathrm{c})$ matches $p(\mathbf{x}, \mathrm{c}) = p_\theta(\mathbf{x})p(\mathrm{c}|\mathbf{x})$.

3.1 Failure modes of Naive Approaches for Adding Controls

Given a pre-trained diffusion model $\epsilon_\theta(\mathbf{x}_t, t)$, the adapters for injecting new conditions can be trained by minimizing a denoising loss [15, 10]. Hence, a natural idea for injecting new conditions into the pre-trained one-step generator is also adapting the denoising loss for training as follows:

$$\min_\phi d(f_{\theta,\phi}(\mathbf{z}, \mathrm{c}), \mathbf{x}),\ \mathbf{z} = \alpha_T \mathbf{x} + \sigma_T \epsilon,\ \mathrm{c} \sim p(\mathrm{c}|\mathbf{x}), \tag{3}$$

where $d(\cdot,\cdot)$ is a distance metric and T denotes the terminal timestep. This approach can potentially inject new conditions into the one-step generator f_θ, similar to existing adapter approaches for DMs. However, it fails to generate high-quality images — the resulting images are blurry, which is due to the *high variance of the optimized objective*. Specifically, its optimal solution is achieved at $f_{\theta,\phi}(\mathbf{z}, \mathrm{c}) = E[\mathbf{x}|\mathbf{z}, \mathrm{c}]$, which is an average of every potential image.

To reduce the variance, one may consider performing denoising loss over *coupled pairs* $(\mathbf{z}, \mathbf{x}, \mathrm{c})$, where $\mathbf{z} \sim \mathcal{N}(0, I)$, c is the condition corresponding to the generated samples $\mathbf{x} = f_\theta(\mathbf{z})$. However, such an approach is unable to perform conditional generation given random $\mathbf{z}$. This is because the model is only exposed to instances of $\mathbf{z}$ strongly associated with c (i.e., $\mathrm{c} \sim p(c|f_\theta(\mathbf{z}))$) during its training, and never encountered random pairings of c and $\mathbf{z}$.

High variance in denoising loss is also a key factor hindering fast sampling in diffusion models. Several methods have been proposed to accelerate the sampling of diffusion models, with optimization objectives typically characterized by low variance properties [26, 24, 33]. Among these, consistency models [26, 27] stand out as a promising approach — instead of optimizing direct denoising loss, they optimize the distance between denoising results of highly-noisy samples and lowly-noisy samples:

$$\min_\alpha L(\alpha) = d(g_\alpha(\mathbf{x}_{t_{n+1}}), \mathrm{sg}(g_\alpha(\mathbf{x}_{t_n}))), \tag{4}$$

where $\mathrm{sg}(\cdot)$ denotes the stop-gradient operator and g_α denotes the desired consistency models. Similar to denoising loss, consistency loss can also force networks to use conditions; thus, it can be used to train adapters to inject new conditions [30]. However, the consistency approach cannot be adapted to the one-step generator since it requires defining the loss over multiple noisy-level images, while the one-step generator only takes random noise as input.

3.2 Our Approach: Noise Consistency Training

To directly inject condition to one-step generator, we propose **Noise Consistency Training**, which diffuses noise to decouple it from the condition and operates the consistency training in **noise space**. Specifically, we diffuse an initial noise $\mathbf{z} \sim \mathcal{N}(0, I)$ to multiple levels $\mathbf{z}_t$ via variance-preservation diffusion as follows:

$$\mathbf{z}_t = \sqrt{1 - \sigma_t}\mathbf{z} + \sigma_t \epsilon, \tag{5}$$

where $\epsilon \sim \mathcal{N}(0, I)$. This ensures that $\mathbf{z}_t$ also follows the standard Gaussian distribution, thus it can be transformed to the high-quality image by the pre-trained one-step generator f_θ.

To inject new conditions to f_θ, we apply an adapter with parameter ϕ, which transforms $f_\theta(\cdot)$ that only takes random noise as input to $f_{\theta,\phi}(\cdot,\cdot)$ that can take an additional condition c as input. We sample coupled pairs $(\mathbf{z}, \mathrm{c})$ from $p_\theta(\mathbf{z}, \mathrm{c})$, where $p(\mathbf{z}, \mathrm{c}) = p(\mathbf{z})p_\theta(\mathrm{c}|\mathbf{z})$, and $p_\theta(\mathrm{c}|\mathbf{z}) \triangleq p_\theta(\mathrm{c}|f_\theta(\mathbf{z}))$. By the $(\mathbf{z}, \mathrm{c})$ pairs, we can perform **Noise Consistency Loss** as follows:

$$\begin{aligned} &\min_\phi \mathbb{E}_{p(\mathbf{z})p(\mathrm{c}|f_\theta(\mathbf{z}))}\mathbb{E}_{q(\mathbf{z}_{t_n}|\mathbf{z}), q(\mathbf{z}_{t_{n-1}}|\mathbf{z})}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)} d(f_{\theta,\phi}(\mathbf{z}_{t_n}, \mathrm{c}), \mathrm{sg}(f_{\theta,\phi}(\mathbf{z}_{t_{n-1}}, \mathrm{c}))) \\ &= \mathbb{E}_{\mathbf{z},\mathrm{c}|\mathbf{z},\epsilon} d(f_{\theta,\phi}(\mathbf{z}_{t_n}, \mathrm{c}), \mathrm{sg}(f_{\theta,\phi}(\mathbf{z}_{t_{n-1}}, \mathrm{c}))), \quad \#\text{Simplified Notation} \end{aligned} \tag{6}$$

where $\mathbf{z}_{t_n} = \sqrt{1 - \sigma_{t_n}^2}\mathbf{z} + \sigma_{t_n}\epsilon$ and $\mathbf{z}_{t_{n-1}} = \sqrt{1 - \sigma_{t_{n-1}}^2}\mathbf{z} + \sigma_{t_{n-1}}\epsilon$. The defined diffusion process can gradually diffuse the coupled pairs $(\mathbf{z}, \mathrm{c})$ to independent uncoupled pairs $(\mathbf{z}_T, \mathrm{c})$. By minimizing the distance between predictions given "less-coupled" pairs and "more-coupled" pairs, we can force the network to utilize the condition. Once trained, the consistency is ensured in the **noise space**. It is expected that the adapter ϕ can be trained for injecting new conditions c, while keeping the high-quality generation capability in one-step. Since the optimized objective has low variance and the generator f_θ can produce high-quality images, the adapter just need to learn how to adapt to the conditions c.

Lemma 1. *Define* $p(\mathbf{z}_0|\mathrm{c}) \triangleq \frac{p(\mathbf{z}_0)p(\mathrm{c}|f_\theta(\mathbf{z}))}{p(\mathrm{c})}$ *and* $p(\mathbf{z}_t|\mathrm{c}) \triangleq \int q(z_t|\mathbf{z}_0)p(\mathbf{z}_0|\mathrm{c})dz_0$. *The forward diffusion process defines an interpolation for the joint distribution* $p(\mathbf{z}_t,\mathrm{c}) \triangleq p(\mathbf{z}_t|\mathrm{c})p(\mathrm{c})$ *between* $p(\mathbf{z}_0,\mathrm{c}) = p(\mathrm{c}|f_\theta(\mathbf{z}_0))p(\mathbf{z}_0)$ *and* $p(\mathbf{z}_T,\mathrm{c}) = p(\mathbf{z}_T)p(\mathrm{c})$.

The above Lemma 1 provides a formal justification to our noise diffusion process as interpolation between the coupled pairs $(\mathbf{z},\mathrm{c})$ to independent pairs $(\mathbf{z}_T,\mathrm{c})$. The proof can be found in Section A.

Lemma 2. *We define the* $f_{\theta,\phi}(\sqrt{1-\sigma_{t_{k+1}}^2}\mathbf{z} + \sigma_{t_{k+1}}\epsilon,\mathrm{c})$ *induced distribution to be* $p_{\theta,\phi,t_{k+1}}$. *The proposed noise consistency loss is a practical estimation of the following loss:*

$$L(\phi) = \sum_{k=0}^{N-1} \mathrm{MMD}^2(p_{\theta,\phi,t_{k+1}}, p_{\theta,\phi,t_k}), \tag{7}$$

under specific hyper-parameter choices (e.g., set particle samples to 1).

See proof in the Section A. The above Lemma 2 builds the connection between our noise consistency training and conditional distribution matching. Technically speaking, using larger particle numbers can further reduce training variance. However, in practice, we found that directly using a single particle achieves similar performance and is more computationally feasible. More investigations on the effect of particle numbers can be found in Section B. This work serves as proof of concept that we can design an approach native to one-step generator in learning new controls, we leave other exploration for further reducing variance in future work.

Boundary Loss A core difference between NCT and CM lies in the model's behavior when reaching boundaries. Specifically, for CM, when the input reaches the boundary $\mathbf{x}_0$, the model only needs to degenerate into an identity mapping outputting $\mathbf{x}_0$, which can be easily satisfied through reparameterization g to stabilize the training. NCT, however, is fundamentally different — when the input reaches the boundary $\mathbf{z}$, the model cannot simply degenerate into an identity mapping, but needs to map $\mathbf{z}$ to high-quality clean images. This means this boundary is non-trivial — the network needs to learn to map $\mathbf{z}$ to corresponding images. Simply reparameterizing $f_{\theta,\phi}$ cannot fully stabilize the training. To satisfy this boundary condition and stabilize the training, we propose setting the clean image corresponding to $\mathbf{z}$ as $f_\theta(\mathbf{z})$ and implementing the following *boundary loss*:

$$\min_\phi \mathbb{E}_{\mathbf{z},\mathrm{c}|\mathbf{z},\epsilon} d(f_{\theta,\phi}(\mathbf{z},\mathrm{c}), f_\theta(\mathbf{z})),\ \mathbf{z} \sim \mathcal{N}(0,I),\ \mathrm{c} \sim p(\mathrm{c}|f_\theta(\mathbf{z})). \tag{8}$$

By minimizing this loss, we can ensure the boundary conditions hold and constrain the generator's output to be close to the data distribution. Intuitively, this loss is easy to understand: when the generator receives the same noise $\mathbf{z}$ and conditions corresponding to f_θ, its generation should be invariant. With the help of this loss, we can constrain the generator's output to stay near the data distribution — otherwise, if we only minimize the noise consistency loss, the model might find unwanted shortcut solutions.

Theorem 1. *Consider a parameter set* ϕ *that satisfies the following two conditions:*

1. Boundary Condition*: The parameters* ϕ *ensure the boundary loss is zero:*

$$\mathbb{E}_{p(\mathbf{z})p(\mathrm{c}|f_\theta(\mathbf{z}))}[d(f_{\theta,\phi}(\mathbf{z},\mathrm{c}), f_\theta(\mathbf{z}))] = 0,$$

2. Consistency Condition*: The parameters* ϕ *also satisfy:*

$$L(\phi) = \sum_{k=0}^{N-1} \mathrm{MMD}^2(p_{\theta,\phi,t_{k+1}}, p_{\theta,\phi,t_k}) = 0$$

Then $f_{\theta,\phi}$ *maps independent* $p(\mathbf{z})p(\mathrm{c})$ *to the target joint distribution* $p_\theta(\mathbf{x})p(\mathrm{c}|\mathbf{x})$.

See proof in the Appendix. Theorem 1 provides theoretical insight for our optimization objective, which is an empirical version for practice.

Overall Optimization We observed that the noise consistency loss is only meaningful when boundary conditions are satisfied or nearly satisfied; otherwise, the generator $f_{\theta,\phi}$ can easily find undesirable shortcut solutions, thus we suggest using a constrained optimization form as follows:

Algorithm 1 Noise Consistency Training

Require: Pre-trained One-Step Generator f_θ, Adapter ϕ, total iterations N
Ensure: Optimized adapter ϕ for injecting new condition.

for $i \leftarrow 1$ **to** N **do**
  Sample noise $\mathbf{z}$ from standard Gaussian distribution;
  Sample noise ϵ from standard Gaussian distribution;
  Sample $\mathbf{x}$ with initialized noise $\mathbf{z}$ from frozen generator f_θ, i.e., $\mathbf{x} = f_\theta(\mathbf{z})$.
  Sample condition c corresponding to $\mathbf{x}$ by $p(\mathrm{c}|\mathbf{x})$.
  `# Primal Step:`
  `## Diffuse Noise via Variance-Preserved Diffusion`
  $z_{t_{k+1}} \leftarrow \alpha_{t_{k+1}}\mathbf{z} + \sigma_{t_{k+1}}\epsilon$ and $z_{t_k} \leftarrow \alpha_{t_k}\mathbf{z} + \sigma_{t_k}\epsilon$.
  `## Compute Noise Consistency Loss`
  $\mathcal{L}_{con} \leftarrow d(f_{\theta,\phi}(\mathbf{z}_{t_{k+1}}, \mathrm{c}), \mathrm{sg}(f_{\theta,\phi}(\mathbf{z}_{t_k}, \mathrm{c})))$
  `## Compute Boundary loss`
  $\mathcal{L}_{bound} \leftarrow d(f_{\theta,\phi}(\mathbf{z}, \mathrm{c}), \mathbf{x})$
  `## Compute Total Loss and Update`
  $\mathcal{L}_{total} \leftarrow \mathcal{L}_{con} + \lambda\mathcal{L}_{bound}$
  Update ϕ using $\nabla_\phi \mathcal{L}_{total}$
  `# Dual Step:`
  Update λ according to Eq. (12).
end for

Definition 1 (Noise Consistency Training)**.** *Given a fixed margin ξ, the general optimization can be transformed into the following:*

$$\begin{aligned} \min_{\phi} \quad & \mathbb{E}_{\mathbf{z},\mathrm{c}|\mathbf{z},\epsilon} L_{\mathrm{con}}(\phi) = d(f_{\theta,\phi}(\mathbf{z}_{t_n}, \mathrm{c}), \mathrm{sg}(f_{\theta,\phi}(\mathbf{z}_{t_{n-1}}, \mathrm{c}))) \\ \text{s.t.} \quad & L_{\mathrm{bound}}(\phi) = \mathbb{E}_{\mathbf{z},\mathrm{c}|\mathbf{z},\epsilon} d(f_{\theta,\phi}(\mathbf{z}, \mathrm{c}), f_\theta(\mathbf{z})) < \xi, \end{aligned} \tag{9}$$

where $\mathbf{z}, \epsilon \sim \mathcal{N}(0, I)$, $\mathrm{c} \sim p(\mathrm{c}|f_\theta(\mathbf{z}))$, $\mathbf{z}_{t_n} = \sqrt{1-\sigma_{t_n}^2}\mathbf{z} + \sigma_{t_n}\epsilon$ and $\mathbf{z}_{t_{n+1}} = \sqrt{1-\sigma_{t_{n+1}}^2}\mathbf{z} + \sigma_{t_{n+1}}\epsilon$.

The constrained optimization problem presented in Definition 1 is hard to optimize directly. We therefore reformulate it as a corresponding saddle-point problem:

$$\max_{\lambda} \min_{\phi} \left\{ L_{\mathrm{con}}(\phi) + \lambda L_{\mathrm{bound}}(\phi) \right\}, \quad \lambda \geq 0. \tag{10}$$

Concrete Algorithm To efficiently optimize this saddle-point problem, we employ the primal-dual algorithm tailored for the saddle-point problem, which alternates between updating the primal variables ϕ and the dual variable λ. Specifically, in the *primal* step, for a given dual variable λ, the algorithm minimizes the corresponding empirical Lagrangian with respect to ϕ under a given dual variable λ, i.e.,

$$\phi_{k+1} := \arg\min_{\phi} \left\{ L_{\mathrm{con}}(\phi_k) + \lambda L_{\mathrm{bound}}(\phi_k) \right\} \tag{11}$$

In practice, this update for ϕ is performed using stochastic gradient descent. Subsequently, in the *dual* step, we update the dual variable λ as follows:

$$\lambda_{t+1} := \max \left\{ \lambda_t + \eta \cdot \left(L_{\mathrm{con}} - \xi \right), 0 \right\}, \tag{12}$$

where η is the learning rate for the dual update.

Algorithm 1 provides the pseudo-code for our primal-dual optimization of the adapter parameters ϕ. In contrast to the direct application of stochastic gradient descent in Eq. (10), the primal-dual algorithm dynamically adjusts λ. This avoids an extra hyper-parameter tuning and can provide an early-stopping condition (e.g., $\lambda = 0$). Additionally, convergence is guaranteed under sufficiently long training and an adequately small step size [34].

Table 1: Comparison of machine metrics of different methods for Canny, HED, Depth and 8× Super Resolution tasks. The mark † denotes our reimplementation with the same one-step generator as used in NCT.

Method	NFE↓	Canny		HED		Depth		8× Super Resolution		Avg	
		FID↓	Consistency↓	FID↓	Consistency↓	FID↓	Consistency↓	FID↓	Consistency↓	FID↓	Consistency↓
ControlNet	50	14.48	0.113	19.21	0.101	**15.25**	0.093	**11.93**	0.065	15.22	0.093
DI + ControlNet	1	22.74	0.141	28.04	0.113	22.49	0.097	15.57	0.126	22.21	0.119
JDM†	1	14.35	0.122	16.75	**0.055**	16.71	0.093	13.23	0.068	15.26	0.085
NCT (Ours)	1	**13.67**	**0.110**	**14.96**	0.060	16.45	**0.088**	12.17	**0.053**	**14.31**	**0.078**

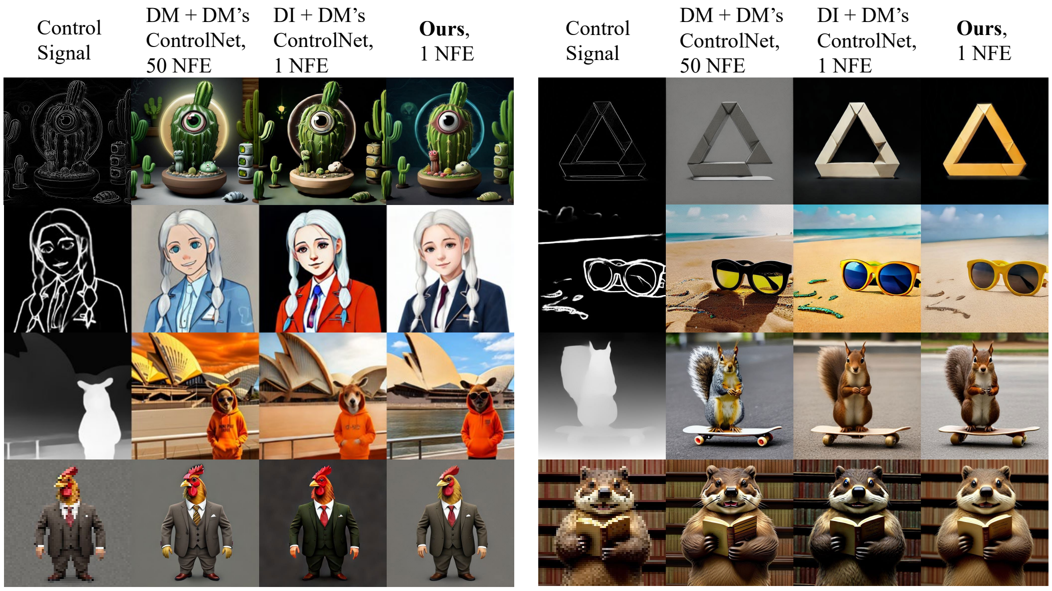

Figure 2: Qualitative comparisons on controllable generation across different control signals against competing methods.

4 Experiments

4.1 Controllable Generation

Experimental Setup. All models are trained on an internally collected dataset. The one-step generator was initialized using weights from Stable Diffusion 1.5 [35]. Subsequently, the one-step generator was pre-trained using the Diff-Instruct [3]. The ControlNet was initialized following the procedure outlined in its original publication [15]. An Exponential Moving Average (EMA) with a decay rate of 0.9999 was applied to the ControlNet parameters, denoted as ϕ.

To evaluate the performance of our proposed method in one-step controllable generation, we employed four distinct conditioning signals: Canny edges [36], HED (Holistically-Nested Edge Detection) boundaries [37], depth maps, and lower-resolution images.

Evaluation Metric. Image quality was assessed using the Fréchet Inception Distance (FID) [38]. Specifically, the FID score was computed by comparing images generated by the base diffusion model without controls against images generated with the incorporation of the aforementioned conditional inputs. The consistency metric for measuring controllability is quantified between the conditioning input c and the condition extracted from the generated image $h(\mathbf{x})$, as formulated below:

$$\text{Consistency} = ||h(\mathbf{x}) - \mathrm{c}||_1, \tag{13}$$

where $h(\cdot)$ represents the function used to extract the conditioning information (e.g., Canny edge detector, depth estimator) from a generated image $\mathbf{x}$, and c is the target conditional input. Furthermore, to assess computational efficiency, we report the NFE required to generate a single image.

Quantitative Results. We conduct comprehensive evaluations, benchmarking our proposed approach against three established baseline methods: (1) the standard diffusion model with ControlNet; (2) a

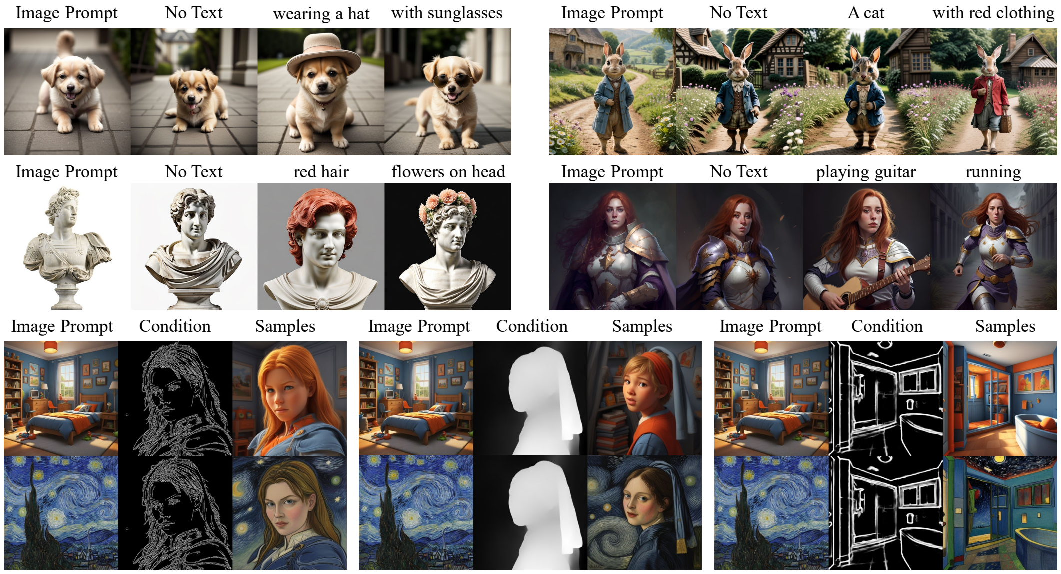

Figure 3: Visual samples of image-reference generations. The samples are generated by our NCT with 1NFE.

pre-trained one-step generator integrated with the DM's ControlNet; and (3) a crafted ControlNet specifically trained for a one-step generator trained via JDM distillation [9]. *Notably, the JDM approach necessitates an additional, computationally intensive distillation phase to incorporate control mechanisms. This step is redundant given that the one-step generator has already undergone a distillation process. In contrast, our method is tailored for one-step generators, obviating the need for further distillation and thereby enhancing computational efficiency.* The quantitative results, presented in Table 1, assess both image fidelity (FID) and adherence to conditional inputs across diverse control tasks. Our proposed method achieves a remarkable reduction in the number of function evaluations (NFEs) from 50 to 1, while concurrently maintaining or surpassing the performance metrics of the baselines. Specifically, our approach demonstrates superior FID scores and stronger consistency measures across various conditioning tasks, signifying enhanced image quality and more precise alignment with control conditions. These findings collectively establish that our method achieves a superior trade-off between computational efficiency and sample quality in controlled image generation. It delivers state-of-the-art performance with substantially reduced computational overhead and a more streamlined training pipeline.

Qualitative Comparison. A qualitative comparison of our method against baselines is presented in Fig. 2, comparing standard ControlNet and DI+ControlNet which does not require additional distillation. Visual results reveal that while the standard DM's ControlNet can impart high-level control to one-step generators, this integration frequently results in a discernible degradation of image quality. In stark contrast, our approach, which involves customized training for adding new controls to one-step generator, consistently produces images of significantly higher fidelity. These visual results substantiate the efficacy of our proposed methodology, suggesting its capability to implicitly learn the conditional distribution $p(\mathbf{x}|\mathrm{c})$ through our novel noise consistency training.

4.2 Image Prompted Generation

Experiment Setting. The pre-trained one-step generator remains consistent with that employed in the prior experiments. We employ the IP-Adapter [39] architecture to serve as the adapter for injecting image prompts. Following IP-Adapter, we use OpenCLIP ViT-H/14 as the image encoder..

Quantitative Comparison. Our method is quantitatively benchmarked against the original IP-Adapter. Following IP-Adapter [39], we generate four images conditioned on each image prompt, for every sample in the COCO-2017-5k dataset [40]. Alignment with the image condition is assessed using two established metrics: 1) CLIP-I: The cosine similarity between the CLIP image embeddings of the generated images and the respective image prompt; 2) CLIP-T: The CLIP Score measuring

Method	NFE↓	Clip-T↑	Clip-I↑
IP-Adapter†	100	0.588	0.828
JDM	1	0.585	0.826
Ours	1	0.593	0.821

Table 2: Comparison of machine metrics of different methods regarding image-prompted generation. The mark † denotes that the result is taken from the official report.

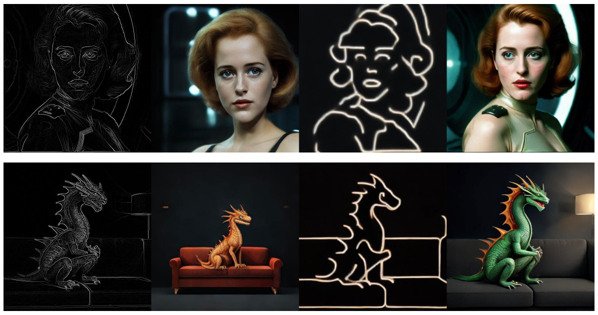

Figure 4: Both boundary loss and noise consistency loss are crucial to our NCT. Without Boundary loss, the model's distribution collapses. Without noise consistency loss, the model ignores the injected condition.

Method	FID↓	Con.↓
Ours	**13.67**	**0.110**
w/o noise consistency loss	20.56	0.165
w/o boundary loss	216.93	0.113
w/o primal-dual	14.13	0.117

Table 3: Ablation study on proposed components in our NCT.

the similarity between the generated images and the captions corresponding to the image prompts. The quantitative results, summarized in Table 2, reveal that our Noise Consistency Training (NCT) method achieves performance comparable to the original IP-Adapter (which necessitates 100 NFEs) on both CLIP-I and CLIP-T metrics. Crucially, NCT attains this level of performance with only a single NFE, signifying an approximate 100-fold improvement in computational efficiency.

Multi-modal Prompts. Our investigations indicate that NCT can concurrently process both image and textual prompts. Fig. 3 illustrates generation outcomes achieved through the use of such multimodal inputs. As demonstrated, the integration of supplementary text prompts facilitates the generation of more diverse visual outputs. This allows for capabilities such as attribute modification and scene alteration based on textual descriptions, relative to the content of the primary image prompt.

Structure Control. We observe that NCT permits the test-time compatibility of adapters designed for image prompting with those designed for controllable generation. This enables the generation of images based on image prompts while jointly incorporating additional structural or conditional controls, as shown in Fig. 3. Such test-time compatibility underscores the inherent flexibility and potential of NCT for training distinct adapters for a one-step generator, which can subsequently be combined effectively during the inference stage.

4.3 Ablation Study

The Effect of Noise Consistency Loss. The noise consistency loss is crucial to force adapter ϕ to learn condition c. Without the loss, it can be seen that the consistency metric degrades severely, and the generated samples do not follow the condition at all. This is because the adapter ϕ is trained on fully-coupled $(\mathbf{z}, \mathrm{c})$ pairs, allowing it find find a shortcut solution that directly ignores the learnable parameters to satisfy the boundary loss.

The Effect of Boundary Loss. The boundary loss can constrain the output of the generator $f_{\theta,\phi}$ in the image domain. Without the loss, although the generator can still learns some conditions, its generated samples entirely collapse as indicated by the FID and visual samples.

The Effect of Primal-Dual. We use primal-dual since it is crafted for solving the constrained problem, while it owns theoretical guarantees and dynamically balances the noise consistency loss and boundary loss. We empirically validate its effectiveness, it can be seen that without primal-dual, the performance slightly degrades regarding both fidelity and condition alignment.

5 Conclusion

This paper addressed the critical challenge of efficiently incorporating new controls into pre-trained one-step generative models, a key bottleneck in the rapidly evolving field of AIGC. We introduced Noise Consistency Training (NCT), a novel and lightweight approach that empowers existing one-step generators with new conditioning capabilities without the need for retraining the base diffusion model or additional diffusion distillation or accessing the original training dataset. By operating in the noise space and leveraging a carefully formulated noise-space consistency loss, NCT effectively aligns the adapted generator with the desired control signals. Our proposed NCT framework offers significant advantages in terms of modularity, data efficiency, and ease of deployment. The experimental results across diverse control scenarios robustly demonstrate that NCT achieves state-of-the-art performance in controllable, single-step generation. It surpasses existing multi-step and distillation-based methods in both the quality of the generated content and computational efficiency.

Acknowledgments

Jing Tang's work is partially supported by National Key R&D Program of China under Grant No. 2024YFA1012700 and No. 2023YFF0725100, by the National Natural Science Foundation of China (NSFC) under Grant No. 62402410 and No. U22B2060, by Guangdong Provincial Project (No. 2023QN10X025), by Guangdong Basic and Applied Basic Research Foundation under Grant No. 2023A1515110131, by Guangzhou Municipal Science and Technology Bureau under Grant No. 2024A04J4454, by Guangzhou Municipal Education Bureau (No. 2024312263), and by Guangzhou Industrial Information and Intelligent Key Laboratory Project (No. 2024A03J0628) and Guangzhou Municipal Key Laboratory of Financial Technology Cutting-Edge Research (No. 2024A03J0630).

References

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

[2] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[3] 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*, 36, 2023.

[4] Yihong Luo, Xiaolong Chen, Xinghua Qu, Tianyang Hu, and Jing Tang. You only sample once: Taming one-step text-to-image synthesis by self-cooperative diffusion gans, 2024.

[5] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. *arXiv preprint arXiv:2311.18828*, 2023.

[6] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models, 2023.

[7] Linqi Zhou, Stefano Ermon, and Jiaming Song. Inductive moment matching. *arXiv preprint arXiv:2503.07565*, 2025.

[8] Xuanwu Yin Yuda Song, Zehao Sun. Sdxs: Real-time one-step latent diffusion models with image conditions. *arxiv*, 2024.

[9] Yihong Luo, Tianyang Hu, Yifan Song, Jiacheng Sun, Zhenguo Li, and Jing Tang. Adding additional control to one-step diffusion with joint distribution matching. *arXiv preprint arXiv:2503.06652*, 2025.

[10] Chong Mou, Xintao Wang, Liangbin Xie, Jian Zhang, Zhongang Qi, Ying Shan, and Xiaohu Qie. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. *arXiv preprint arXiv:2302.08453*, 2023.

[11] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[12] Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Roni Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *The Twelfth International Conference on Learning Representations*, 2024.

[13] Jiajun Ma, Tianyang Hu, Wenjia Wang, and Jiacheng Sun. Elucidating the design space of classifier-guided diffusion generation. *arXiv preprint arXiv:2310.11311*, 2023.

[14] Yihong Luo, Tianyang Hu, Weijian Luo, Kenji Kawaguchi, and Jing Tang. Reward-instruct: A reward-centric approach to fast photo-realistic image generation. *arXiv preprint arXiv:2503.13070*, 2025.

[15] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.

[16] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

[17] Alfred Müller. Integral probability metrics and their generating classes of functions. *Advances in applied probability*, 29(2):429–443, 1997.

[18] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022.

[19] Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. *Advances in Neural Information Processing Systems*, 36:49842–49869, 2023.

[20] Shuchen Xue, Zhaoqiang Liu, Fei Chen, Shifeng Zhang, Tianyang Hu, Enze Xie, and Zhenguo Li. Accelerating diffusion sampling with optimized time steps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8292–8301, 2024.

[21] Chenyang Si, Ziqi Huang, Yuming Jiang, and Ziwei Liu. Freeu: Free lunch in diffusion u-net. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4733–4743, 2024.

[22] Jiajun Ma, Shuchen Xue, Tianyang Hu, Wenjia Wang, Zhaoqiang Liu, Zhenguo Li, Zhi-Ming Ma, and Kenji Kawaguchi. The surprising effectiveness of skip-tuning in diffusion sampling. *arXiv preprint arXiv:2402.15170*, 2024.

[23] Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021.

[24] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14297–14306, June 2023.

[25] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

[26] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. 2023.

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

[28] Hanshu Yan, Xingchao Liu, Jiachun Pan, Jun Hao Liew, Qiang Liu, and Jiashi Feng. Perflow: Piecewise rectified flow as universal plug-and-play accelerator. *arXiv preprint arXiv:2405.07510*, 2024.

[29] 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. In *International Conference on Machine Learning*, 2024.

[30] Jie Xiao, Kai Zhu, Han Zhang, Zhiheng Liu, Yujun Shen, Zhantao Yang, Ruili Feng, Yu Liu, Xueyang Fu, and Zheng-Jun Zha. CCM: Real-time controllable visual content creation using text-to-image consistency models. In *Forty-first International Conference on Machine Learning*, 2024.

[31] Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis, 2024.

[32] Yihong Luo, Tianyang Hu, Jiacheng Sun, Yujun Cai, and Jing Tang. Learning few-step diffusion models by trajectory distribution matching, 2025.

[33] 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. *arXiv preprint arXiv:2309.06380*, 2023.

[34] Luiz FO Chamon, Santiago Paternain, Miguel Calvo-Fullana, and Alejandro Ribeiro. Constrained learning with non-convex losses. *arXiv:2103.05134*, 2021.

[35] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[36] John Canny. A computational approach to edge detection. *PAMI*, 1986.

[37] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *ICCV*, 2015.

[38] 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.

[39] Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models. 2023.

[40] 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 *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

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: Our work proposes NCT, which can directly add new controls to one-step generators without additional diffusion distillation.

   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: We discuss the limitations in the Appendix C.

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: Provided in the appendix.

   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: Necessary information is provided in main paper and appendix.

   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: [No]

   Justification: The experiments are conducted in internally collected datasets. We have described how to form the training data and provided pseudo-code of our method.

   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: We have specified them in the appendix.

   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: [No]

   Justification: error bar is not reported, since it is too computationally expensive.

   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: We reported them in the appendix.

   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: We follow the Code of Ethics from all the perspectives stated.

   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: [Yes]

    Justification: We discussed this in Appendix D.

    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: [Yes]

    Justification: We discussed it in the Appendix E.

    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: [Yes]

    Justification: All existing assets used in this paper have been properly credited.

    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: We do not release new assets in the submission phase.

    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: This paper does not involve crowdsourcing nor research with human subjects.

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: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    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: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    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.

A Theoretical Foundation of Noise Consistency Training

This section establishes the theoretical foundation: it begins with definitions and the mathematical setup (A.1), then introduces key lemmas (A.2) that collectively build the necessary mathematical foundation—by defining critical distribution relationships and an input interpolation path—for formulating the conditions and proving the main theorem (A.3). Specifically, Lemma 1, Lemma 2, and Theorem 1 presented in the main paper are proved in Lemma A.3, Remark 1, and Theorem A.1 respectively in this section.

A.1 Definition and Setup

- **Latent Distribution:** The latent distribution Γ is the standard Gaussian measure on $(\mathbb{R}^m, \mathcal{B}(\mathbb{R}^m))$. The measure Γ has density $\gamma(\mathbf{z})$ w.r.t. $\mathrm{d}\mathbf{z}$, so $\mathrm{d}\Gamma(\mathbf{z}) = \gamma(\mathbf{z})\mathrm{d}\mathbf{z}$. Γ is a probability measure: $\int_{\mathbb{R}^m} \gamma(\mathbf{z})\mathrm{d}\mathbf{z} = 1$.
- **Implicit Generator:** $f_\theta : \mathbb{R}^m \to \mathbb{R}^n$ is a measurable function.
- **Data Distribution:** P_θ on $(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n))$ is the push-forward $P_\theta = f_{\theta\#}\Gamma$. It has density $p(\mathbf{x})$ w.r.t. $\mathrm{d}\mathbf{x}$, so $\mathrm{d}P_\theta(\mathbf{x}) = p_\theta(\mathbf{x})\mathrm{d}\mathbf{x}$. Since Γ is a probability measure, P_θ is also a probability measure: $\int_{\mathbb{R}^n} p_\theta(\mathbf{x})\mathrm{d}\mathbf{x} = 1$.
- **Condition:**. Let $(\mathcal{C}, \mathcal{B}_\mathcal{C}, \mu_\mathcal{C})$ be a measure space for the conditions. $\mathcal{B}_\mathcal{C}$ is a σ-algebra on $\mathcal{C}$, and $\mu_\mathcal{C}$ is a reference measure (e.g., Lebesgue measure if $\mathcal{C} = \mathbb{R}^k$ (such as Canny Edge), or counting measure if $\mathcal{C}$ is discrete) (such as class labels). For each $\mathbf{x} \in \mathbb{R}^n$, $p(\cdot|\mathbf{x})$ is a probability measure on $(\mathcal{C}, \mathcal{B}_\mathcal{C})$. We assume it has a density $p(\mathrm{c}|\mathbf{x})$ with respect to $\mu_\mathcal{C}$. Thus, for any $\mathbf{x} \in \mathbb{R}^n$: $\int_\mathcal{C} p(\mathrm{c}|\mathbf{x})\mathrm{d}\mu_\mathcal{C}(\mathrm{c}) = 1$.
- **Combined Map:** $T = f_\theta \times \mathrm{id} : \mathbb{R}^m \times \mathcal{C} \to \mathbb{R}^n \times \mathcal{C}$, $T(\mathbf{z}, \mathrm{c}) = (f_\theta(\mathbf{z}), \mathrm{c})$. Since f_θ and id are measurable, T is measurable with respect to the product σ-algebras $\mathcal{B}(\mathbb{R}^m) \otimes \mathcal{B}_\mathcal{C}$ and $\mathcal{B}(\mathbb{R}^n) \otimes \mathcal{B}_\mathcal{C}$.
- **Implicit Generator with Condition:** $f_{\theta,\phi} : \mathbb{R}^m \times \mathcal{C} \to \mathbb{R}^n$ be a measurable function.
- **Combined Map with Condition:** We define a new map $T_\phi : \mathbb{R}^m \times \mathcal{C} \to \mathbb{R}^n \times \mathcal{C}$ as $T_\phi(\mathbf{z}, \mathrm{c}) = (f_{\theta,\phi}(\mathbf{z}, \mathrm{c}), \mathrm{c})$.
- **Marginal Condition Density:** We define the marginal probability density $p(\mathrm{c})$ of the condition c as:

  $$p(\mathrm{c}) = \int_{\mathbb{R}^n} p_\theta(\mathbf{x}) p(\mathrm{c}|\mathbf{x})\mathrm{d}\mathbf{x} = \int_{\mathbb{R}^m} \gamma(\mathbf{z}) p(\mathrm{c}|f_\theta(\mathbf{z}))\mathrm{d}\mathbf{z}$$

  This is a probability density with respect to $\mu_\mathcal{C}$, i.e., $\int_\mathcal{C} p(\mathrm{c})\mathrm{d}\mu_\mathcal{C}(\mathrm{c}) = 1$.
- **Initial Coupled Latent-Condition Distribution**: Density $\nu(\mathbf{z}, \mathrm{c}) = \gamma(\mathbf{z}) p(\mathrm{c}|f_\theta(\mathbf{z}))$ w.r.t. $\mathrm{d}\mathbf{z}\mathrm{d}\mu_\mathcal{C}(\mathrm{c})$.
- **Independent Latent-Condition Coupling:** We define the probability measure ρ on the input space $(\mathbb{R}^m \times \mathcal{C}, \mathcal{B}(\mathbb{R}^m) \otimes \mathcal{B}_\mathcal{C})$ by its density with respect to the reference measure $\mathrm{d}\mathbf{z}\,\mathrm{d}\mu_\mathcal{C}(\mathrm{c})$:

  $$\mathrm{d}\rho(\mathbf{z}, \mathrm{c}) = \gamma(\mathbf{z}) p(\mathrm{c})\mathrm{d}\mathbf{z}\,\mathrm{d}\mu_\mathcal{C}(\mathrm{c})$$

  Here, $\gamma(\mathbf{z})$ is the density of the standard Gaussian measure Γ on $\mathbb{R}^m$. The measure ρ corresponds to sampling $\mathbf{z} \sim \Gamma$ independently from sampling $\mathrm{c} \sim p(\mathrm{c})$.
- **Target Data-Condition Distribution** η: Density $p_\eta(\mathbf{x}, \mathrm{c}) = p_\theta(\mathbf{x}) p(\mathrm{c}|\mathbf{x})$ w.r.t. $\mathrm{d}\mathbf{x}\mathrm{d}\mu_\mathcal{C}(\mathrm{c})$.
- **MMD (Maximum Mean Discrepancy)**: $\mathrm{MMD}^2(P, Q)$ is a metric between probability distributions P and Q. For a characteristic kernel, $\mathrm{MMD}^2(P, Q) = 0 \iff P = Q$.

A.2 Lemmas

Lemma A.1. *Let Γ be the standard Gaussian measure on $\mathbb{R}^m$ with density $\gamma(\mathbf{z})$ with respect to the Lebesgue measure $\mathrm{d}\mathbf{z}$. Let $f_\theta : \mathbb{R}^m \to \mathbb{R}^n$ be a measurable function, and let $P_\theta = f_{\theta\#}\Gamma$ be the push-forward measure on $\mathbb{R}^n$, assumed to have a density $p_\theta(\mathbf{x})$ with respect to the Lebesgue measure $\mathrm{d}\mathbf{x}$. Let $(\mathcal{C}, \mathcal{B}_\mathcal{C}, \mu_\mathcal{C})$ be a measure space for conditions, and let $p(\mathrm{c}|\mathbf{x})$ be a conditional probability density on $\mathcal{C}$ with respect to $\mu_\mathcal{C}$ for each $\mathbf{x} \in \mathbb{R}^n$, such that $\int_\mathcal{C} p(\mathrm{c}|\mathbf{x})\mathrm{d}\mu_\mathcal{C}(\mathrm{c}) = 1$.*

Define the measure ν on $\mathbb{R}^m \times \mathcal{C}$ by its density with respect to $\mathrm{d}\mathbf{z}\mathrm{d}\mu_{\mathcal{C}}(\mathrm{c})$:

$$\mathrm{d}\nu(\mathbf{z}, \mathrm{c}) = \gamma(\mathbf{z})p(\mathrm{c}|f_\theta(\mathbf{z}))\mathrm{d}\mathbf{z}\mathrm{d}\mu_{\mathcal{C}}(\mathrm{c})$$

Define the map $T = f_\theta \times id : \mathbb{R}^m \times \mathcal{C} \to \mathbb{R}^n \times \mathcal{C}$ by $T(\mathbf{z}, \mathrm{c}) = (f_\theta(\mathbf{z}), \mathrm{c})$.

Then the push-forward measure $T_\# \nu$ on $\mathbb{R}^n \times \mathcal{C}$ has the density $p_\theta(\mathbf{x})p(\mathrm{c}|\mathbf{x})$ with respect to $\mathrm{d}\mathbf{x}\mathrm{d}\mu_{\mathcal{C}}(\mathrm{c})$. That is,

$$(f_\theta \times id)_\#(\gamma(\mathbf{z})p(\mathrm{c}|f_\theta(\mathbf{z}))\mathrm{d}\mathbf{z}\mathrm{d}\mu_{\mathcal{C}}(\mathrm{c})) = p_\theta(\mathbf{x})p(\mathrm{c}|\mathbf{x})\mathrm{d}\mathbf{x}\mathrm{d}\mu_{\mathcal{C}}(\mathrm{c})$$

Proof. We want to show $T_\# \nu = \eta$. By definition of equality of measures, it suffices to show that for any bounded, measurable test function $\Phi : \mathbb{R}^n \times \mathcal{C} \to \mathbb{R}$:

$$\int_{\mathbb{R}^n \times \mathcal{C}} \Phi(\mathbf{x}, \mathrm{c})\mathrm{d}(T_\# \nu)(\mathbf{x}, \mathrm{c}) = \int_{\mathbb{R}^n \times \mathcal{C}} \Phi(\mathbf{x}, \mathrm{c})\mathrm{d}\eta(\mathbf{x}, \mathrm{c})$$

We start with the left-hand side (LHS). Using the change of variables formula for push-forward measures:

$$\begin{aligned}
LHS &= \int_{\mathbb{R}^n \times \mathcal{C}} \Phi(\mathbf{x}, \mathrm{c})\mathrm{d}(T_\# \nu)(\mathbf{x}, \mathrm{c}) \\
&= \int_{\mathbb{R}^m \times \mathcal{C}} \Phi(T(\mathbf{z}, \mathrm{c}))\mathrm{d}\nu(\mathbf{z}, \mathrm{c}) \quad \text{(Change of Variables)} \\
&= \int_{\mathbb{R}^m \times \mathcal{C}} \Phi(f_\theta(\mathbf{z}), \mathrm{c})\gamma(\mathbf{z})p(\mathrm{c}|f_\theta(\mathbf{z}))\mathrm{d}\mathbf{z}\mathrm{d}\mu_{\mathcal{C}}(\mathrm{c}) \quad \text{(Substitute } T \text{ and density of } \nu) \\
&= \int_{\mathbb{R}^m} \gamma(\mathbf{z}) \left[\int_{\mathcal{C}} \Phi(f_\theta(\mathbf{z}), \mathrm{c})p(\mathrm{c}|f_\theta(\mathbf{z}))\mathrm{d}\mu_{\mathcal{C}}(\mathrm{c}) \right] \mathrm{d}\mathbf{z} \quad \text{(Fubini's Theorem)}
\end{aligned}$$

The application of Fubini's theorem is justified because Φ is bounded, $\gamma(\mathbf{z}) \geq 0$, $p(\mathrm{c}|f_\theta(\mathbf{z})) \geq 0$, and ν is a finite (probability) measure.

Let's define an auxiliary function $g : \mathbb{R}^n \to \mathbb{R}$ as:

$$g(\mathbf{y}) = \int_{\mathcal{C}} \Phi(\mathbf{y}, \mathrm{c})p(\mathrm{c}|\mathbf{y})\mathrm{d}\mu_{\mathcal{C}}(\mathrm{c})$$

Since Φ is bounded (say $|\Phi| \leq M$) and $\int_{\mathcal{C}} p(\mathrm{c}|\mathbf{y})\mathrm{d}\mu_{\mathcal{C}}(\mathrm{c}) = 1$, $g(\mathbf{y})$ is also bounded ($|g(\mathbf{y})| \leq M$). If Φ is $\mathcal{B}(\mathbb{R}^n) \otimes \mathcal{B}_{\mathcal{C}}$-measurable and $p(\mathrm{c}|\mathbf{y})$ defines a measurable transition kernel, then g is $\mathcal{B}(\mathbb{R}^n)$-measurable.

Substituting g into our integral expression:

$$LHS = \int_{\mathbb{R}^m} g(f_\theta(\mathbf{z}))\gamma(\mathbf{z})\mathrm{d}\mathbf{z}$$

Now, recall the definition of the push-forward measure $P_\theta = f_\# \Gamma$. For any bounded, measurable function $h : \mathbb{R}^n \to \mathbb{R}$:

$$\int_{\mathbb{R}^n} h(\mathbf{x})\mathrm{d}P_\theta(\mathbf{x}) = \int_{\mathbb{R}^m} h(f_\theta(\mathbf{z}))\mathrm{d}\Gamma(\mathbf{z})$$

In terms of densities:

$$\int_{\mathbb{R}^n} h(\mathbf{x})p_\theta(\mathbf{x})\mathrm{d}\mathbf{x} = \int_{\mathbb{R}^m} h(f_\theta(\mathbf{z}))\gamma(\mathbf{z})\mathrm{d}\mathbf{z}$$

Applying this identity with $h = g$:

$$\int_{\mathbb{R}^m} g(f_\theta(\mathbf{z}))\gamma(\mathbf{z})\mathrm{d}\mathbf{z} = \int_{\mathbb{R}^n} g(\mathbf{x})p_\theta(\mathbf{x})\mathrm{d}\mathbf{x}$$

So,

$$LHS = \int_{\mathbb{R}^n} g(\mathbf{x})p_\theta(\mathbf{x})\mathrm{d}\mathbf{x}$$

Now, substitute back the definition of $g(\mathbf{x})$:

$$LHS = \int_{\mathbb{R}^n} \left[\int_{\mathcal{C}} \Phi(\mathbf{x}, \mathrm{c})p(\mathrm{c}|\mathbf{x})\mathrm{d}\mu_{\mathcal{C}}(\mathrm{c}) \right] p_\theta(\mathbf{x})\mathrm{d}\mathbf{x}$$

Applying Fubini's Theorem again (justified as before):

$$LHS = \int_{\mathbb{R}^n \times \mathcal{C}} \Phi(\mathbf{x}, \mathrm{c}) p_\theta(\mathbf{x}) p(\mathrm{c}|\mathbf{x}) \mathrm{d}\mathbf{x} \mathrm{d}\mu_{\mathcal{C}}(\mathrm{c})$$

This is precisely the integral with respect to the target measure η:

$$LHS = \int_{\mathbb{R}^n \times \mathcal{C}} \Phi(\mathbf{x}, \mathrm{c}) \mathrm{d}\eta(\mathbf{x}, \mathrm{c})$$

Since we have shown that $\int \Phi \mathrm{d}(T_\# \nu) = \int \Phi \mathrm{d}\eta$ for all bounded, measurable test functions Φ, the measures must be equal:

$$T_\# \nu = \eta$$

□

Lemma A.2 (Boundary Loss)**.** *Let $f_{\theta,\phi} : \mathbb{R}^m \times \mathcal{C} \to \mathbb{R}^n$ be a measurable function. Let ν be the measure on $\mathbb{R}^m \times \mathcal{C}$ with density $\gamma(\mathbf{z})p(\mathrm{c}|f_\theta(\mathbf{z}))$ w.r.t. $\mathrm{d}\mathbf{z}\mathrm{d}\mu_{\mathcal{C}}(\mathrm{c})$. Let d be a distance metric on $\mathbb{R}^n$. If the boundary loss*

$$\mathbb{E}_{(\mathbf{z},\mathrm{c})\sim\nu}[d(f_{\theta,\phi}(\mathbf{z}, \mathrm{c}), f_\theta(\mathbf{z}))] = 0$$

then:

The push-forward measure $\eta_\phi = (T_\phi)_\# \nu$ is equal to the target measure $\eta = T_\# \nu$. This means the joint distribution $p_{\theta,\phi}(\mathbf{x}, \mathrm{c})$ induced by $f_{\theta,\phi}$ is $p_\theta(\mathbf{x})p(\mathrm{c}|\mathbf{x})$, i.e., $\eta_\phi = \eta$.

Proof. The condition is $\mathbb{E}_{(\mathbf{z},\mathrm{c})\sim\nu}[d(f_{\theta,\phi}(\mathbf{z}, \mathrm{c}), f_\theta(\mathbf{z}))] = 0$. Since $d(a, b) \geq 0$ for any $a, b \in \mathbb{R}^n$, and $d(a, b) = 0$ if and only if $a = b$, the expectation of this non-negative quantity being zero implies that the integrand must be zero ν-almost everywhere. That is,

$$d(f_{\theta,\phi}(\mathbf{z}, \mathrm{c}), f_\theta(\mathbf{z})) = 0 \quad \text{for } \nu\text{-a.e. } (\mathbf{z}, \mathrm{c})$$

This implies

$$f_{\theta,\phi}(\mathbf{z}, \mathrm{c}) = f_\theta(\mathbf{z}) \quad \text{for } \nu\text{-a.e. } (\mathbf{z}, \mathrm{c})$$

We want to show that $\eta_\phi = (T_\phi)_\# \nu$ is equal to $\eta = T_\# \nu$. Recall the definitions of the maps:

$$T(\mathbf{z}, \mathrm{c}) = (f_\theta(\mathbf{z}), \mathrm{c})$$

$$T_\phi(\mathbf{z}, \mathrm{c}) = (f_{\theta,\phi}(\mathbf{z}, \mathrm{c}), \mathrm{c})$$

Since $f_{\theta,\phi}(\mathbf{z}, \mathrm{c}) = f_\theta(\mathbf{z})$ for ν-a.e. $(\mathbf{z}, \mathrm{c})$, it follows directly that the maps T_ϕ and T are equal ν-almost everywhere:

$$T_\phi(\mathbf{z}, \mathrm{c}) = (f_{\theta,\phi}(\mathbf{z}, \mathrm{c}), \mathrm{c}) = (f_\theta(\mathbf{z}), \mathrm{c}) = T(\mathbf{z}, \mathrm{c}) \quad \text{for } \nu\text{-a.e. } (\mathbf{z}, \mathrm{c})$$

If two measurable maps T and T_ϕ are equal ν-a.e., their push-forward measures $T_\# \nu$ and $(T_\phi)_\# \nu$ are identical. Let $\Psi : \mathbb{R}^n \times \mathcal{C} \to \mathbb{R}$ be any bounded, measurable test function.

$$\begin{aligned}
\int_{\mathbb{R}^n \times \mathcal{C}} \Psi(\mathbf{x}, \mathrm{c}) \mathrm{d}((T_\phi)_\# \nu)(\mathbf{x}, \mathrm{c}) &= \int_{\mathbb{R}^m \times \mathcal{C}} \Psi(T_\phi(\mathbf{z}, \mathrm{c})) \mathrm{d}\nu(\mathbf{z}, \mathrm{c}) \\
&= \int_{\mathbb{R}^m \times \mathcal{C}} \Psi(T(\mathbf{z}, \mathrm{c})) \mathrm{d}\nu(\mathbf{z}, \mathrm{c}) \quad (\text{since } T_\phi = T\ \nu\text{-a.e. and } \Psi \text{ is bounded}) \\
&= \int_{\mathbb{R}^n \times \mathcal{C}} \Psi(\mathbf{x}, \mathrm{c}) \mathrm{d}(T_\# \nu)(\mathbf{x}, \mathrm{c})
\end{aligned}$$

Since this holds for all bounded measurable Ψ, we have $(T_\phi)_\# \nu = T_\# \nu$. From Lemma 1, we know $T_\# \nu = \eta$, where η has density $p_\theta(\mathbf{x})p(\mathrm{c}|\mathbf{x})$ with respect to $\mathrm{d}\mathbf{x}\mathrm{d}\mu_{\mathcal{C}}(\mathrm{c})$. Therefore, $\eta_\phi = (T_\phi)_\# \nu = \eta$. □

Lemma A.3 (Interpolation of Joint Latent-Condition Distributions (Lemma 1 in main paper))**.** *Let $\gamma(\mathbf{z})$ be the density of the standard Gaussian measure on $\mathbb{R}^m$. Let $f_\theta : \mathbb{R}^m \to \mathbb{R}^n$ be a measurable function. Let $p(\mathrm{c}|\mathbf{x})$ be a conditional probability density on $\mathcal{C}$ (with respect to a reference measure $\mu_{\mathcal{C}}$) for each $\mathbf{x} \in \mathbb{R}^n$. Define the marginal condition density $p(\mathrm{c})$ as:*

$$p(\mathrm{c}) = \int_{\mathbb{R}^m} \gamma(\mathbf{z}')p(\mathrm{c}|f_\theta(\mathbf{z}'))\mathrm{d}\mathbf{z}'$$

Assume $p(\mathrm{c}) > 0$ for $\mu_{\mathcal{C}}$-almost every c in the support of interest. Define the conditional latent density $p_{data}(\mathbf{z}_0|\mathrm{c})$ as:

$$p_{data}(\mathbf{z}_0|\mathrm{c}) = \frac{\gamma(\mathbf{z}_0)p(\mathrm{c}|f_\theta(\mathbf{z}_0))}{p(\mathrm{c})}$$

Consider a time-dependent process for $t \in [0, 1]$ where $\mathbf{z}_t$ is generated from $\mathbf{z}_0 \sim p_{data}(\cdot|\mathrm{c})$ by:

$$\mathbf{z}_t = \alpha_t \mathbf{z}_0 + \sigma_t \boldsymbol{\epsilon}, \quad \textit{where } \boldsymbol{\epsilon} \sim \mathcal{N}(0, I_m) \textit{ independent of } \mathbf{z}_0 \textit{ and } \mathrm{c}.$$

The coefficients $\alpha_t, \sigma_t \in \mathbb{R}$ satisfy:

- $\alpha_0 = 1, \sigma_0 = 0$
- $\alpha_1 = 0, \sigma_1 = 1$
- *α_t is monotonically decreasing, σ_t is monotonically increasing.*

Let $q_t(\mathbf{z}_t|\mathbf{z}_0) = \mathcal{N}(\mathbf{z}_t; \alpha_t \mathbf{z}_0, \sigma_t^2 I_m)$ be the density of $\mathbf{z}_t$ given $\mathbf{z}_0$. Define the conditional density $p_t(\mathbf{z}|\mathrm{c})$ as:

$$p_t(\mathbf{z}|\mathrm{c}) = \int_{\mathbb{R}^m} q_t(\mathbf{z}|\mathbf{z}_0) p_{data}(\mathbf{z}_0|\mathrm{c}) \mathrm{d}\mathbf{z}_0$$

And the joint density $p_t(\mathbf{z}, \mathrm{c})$ on $\mathbb{R}^m \times \mathcal{C}$ (with respect to $\mathrm{d}\mathbf{z}\mathrm{d}\mu_{\mathcal{C}}(\mathrm{c})$) as:

$$p_t(\mathbf{z}, \mathrm{c}) = p_t(\mathbf{z}|\mathrm{c}) p(\mathrm{c})$$

Then,

1. *At $t = 0$, the joint density is $p_0(\mathbf{z}, \mathrm{c}) = \gamma(\mathbf{z}) p(\mathrm{c}|f_\theta(\mathbf{z}))$.*
2. *At $t = 1$, the joint density is $p_1(\mathbf{z}, \mathrm{c}) = \gamma(\mathbf{z}) p(\mathrm{c})$.*

Proof. The joint density at time t is given by $p_t(\mathbf{z}, \mathrm{c}) = p_t(\mathbf{z}|\mathrm{c}) p(\mathrm{c})$. Substituting the definition of $p_t(\mathbf{z}|\mathrm{c})$:

$$p_t(\mathbf{z}, \mathrm{c}) = p(\mathrm{c}) \int_{\mathbb{R}^m} \mathcal{N}(\mathbf{z}; \alpha_t \mathbf{z}_0, \sigma_t^2 I_m) p_{\text{data}}(\mathbf{z}_0|\mathrm{c}) \mathrm{d}\mathbf{z}_0$$

Now, substitute the definition of $p_{\text{data}}(\mathbf{z}_0|\mathrm{c}) = \frac{\gamma(\mathbf{z}_0)p(\mathrm{c}|f_\theta(\mathbf{z}_0))}{p(\mathrm{c})}$:

$$p_t(\mathbf{z}, \mathrm{c}) = p(\mathrm{c}) \int_{\mathbb{R}^m} \mathcal{N}(\mathbf{z}; \alpha_t \mathbf{z}_0, \sigma_t^2 I_m) \frac{\gamma(\mathbf{z}_0)p(\mathrm{c}|f_\theta(\mathbf{z}_0))}{p(\mathrm{c})} \mathrm{d}\mathbf{z}_0$$

Assuming $p(\mathrm{c}) \neq 0$ (for $\mu_{\mathcal{C}}$-a.e. c), we can cancel $p(\mathrm{c})$:

$$p_t(\mathbf{z}, \mathrm{c}) = \int_{\mathbb{R}^m} \mathcal{N}(\mathbf{z}; \alpha_t \mathbf{z}_0, \sigma_t^2 I_m) \gamma(\mathbf{z}_0) p(\mathrm{c}|f_\theta(\mathbf{z}_0)) \mathrm{d}\mathbf{z}_0$$

At $t = 0$, we have $\alpha_0 = 1$ and $\sigma_0 = 0$. The Gaussian density $\mathcal{N}(\mathbf{z}; \alpha_0 \mathbf{z}_0, \sigma_0^2 I_m)$ becomes $\mathcal{N}(\mathbf{z}; \mathbf{z}_0, 0 \cdot I_m)$. This is interpreted as the Dirac delta function $\delta(\mathbf{z} - \mathbf{z}_0)$. So,

$$\begin{aligned} p_0(\mathbf{z}, \mathrm{c}) &= \int_{\mathbb{R}^m} \delta(\mathbf{z} - \mathbf{z}_0) \gamma(\mathbf{z}_0) p(\mathrm{c}|f_\theta(\mathbf{z}_0)) \mathrm{d}\mathbf{z}_0 \\ &= \gamma(\mathbf{z}) p(\mathrm{c}|f_\theta(\mathbf{z})) \quad \text{(by the sifting property of the Dirac delta)} \end{aligned}$$

This matches the first target distribution.

At $t = 1$, we have $\alpha_1 = 0$ and $\sigma_1 = 1$. The Gaussian density $\mathcal{N}(\mathbf{z}; \alpha_1 \mathbf{z}_0, \sigma_1^2 I_m)$ becomes $\mathcal{N}(\mathbf{z}; 0 \cdot \mathbf{z}_0, 1^2 I_m) = \mathcal{N}(\mathbf{z}; 0, I_m)$. By definition, $\mathcal{N}(\mathbf{z}; 0, I_m) = \gamma(\mathbf{z})$. So,

$$\begin{aligned} p_1(\mathbf{z}, \mathrm{c}) &= \int_{\mathbb{R}^m} \gamma(\mathbf{z}) \gamma(\mathbf{z}_0) p(\mathrm{c}|f_\theta(\mathbf{z}_0)) \mathrm{d}\mathbf{z}_0 \\ &= \gamma(\mathbf{z}) \int_{\mathbb{R}^m} \gamma(\mathbf{z}_0) p(\mathrm{c}|f_\theta(\mathbf{z}_0)) \mathrm{d}\mathbf{z}_0 \end{aligned}$$

The integral $\int_{\mathbb{R}^m} \gamma(\mathbf{z}_0) p(\mathrm{c}|f_\theta(\mathbf{z}_0)) \mathrm{d}\mathbf{z}_0$ is, by definition, $p(\mathrm{c})$. Therefore,

$$p_1(\mathbf{z}, \mathrm{c}) = \gamma(\mathbf{z}) p(\mathrm{c})$$

This matches the second target distribution.

Thus, the process defines an interpolation for the joint density $p_t(\mathbf{z}, \mathrm{c})$ between $p_0(\mathbf{z}, \mathrm{c}) = \gamma(\mathbf{z}) p(\mathrm{c}|f_\theta(\mathbf{z}))$ and $p_1(\mathbf{z}, \mathrm{c}) = \gamma(\mathbf{z}) p(\mathrm{c})$. $\square$

A.3 Main Theorem and Proof

Definition 1 (Interpolation Distribution Sequence (from Lemma 3))**.** *A sequence of time points $0 = t_0 < t_1 < \cdots < t_N = 1$. For each t_k, we have a latent-condition distribution ν_{t_k} (density $p_{t_k}(\mathbf{z}, \mathrm{c})$) such that $\nu_{t_0} = \nu_0$ and $\nu_{t_N} = \rho$.*

Theorem A.1. *Assume the distributions η, ν_0, ρ and the interpolation sequence $\{\nu_{t_k}\}_{k=0}^{N}$ as defined above. Let $f_\theta : \mathbb{R}^m \to \mathbb{R}^n$ be a pre-trained generator, and $f_{\theta,\phi} : \mathbb{R}^m \times \mathcal{C} \to \mathbb{R}^n$ be a conditional generator with a single set of trainable parameters ϕ. The map T_ϕ is defined as $T_\phi(\mathbf{z}, \mathrm{c}) = (f_{\theta,\phi}(\mathbf{z}, \mathrm{c}), \mathrm{c})$.*

Consider the following two conditions:

1. ***Boundary Condition**: The parameters ϕ ensure the boundary loss is zero:*

$$\mathbb{E}_{(\mathbf{z},\mathrm{c})\sim\nu_{t_0}}[d(f_{\theta,\phi}(\mathbf{z}, \mathrm{c}), f_\theta(\mathbf{z}))] = 0$$

   where $d(\cdot, \cdot)$ is a distance metric on $\mathbb{R}^n$. By the Boundary Loss Lemma, this implies $(T_\phi)_\# \nu_{t_0} = \eta$.

2. ***Consistency Condition**: The parameters ϕ also satisfy:*

$$L_{total}(\phi) = \sum_{k=0}^{N-1} \mathrm{MMD}^2((T_\phi)_\# \nu_{t_{k+1}}, (T_\phi)_\# \nu_{t_k}) = 0$$

If such a parameter set ϕ exists and satisfies both conditions above, then $f_{\theta,\phi}$ (when its input is distributed according to ρ) generates the target data-condition distribution η:

$$(T_\phi)_\# \rho = \eta$$

That is, if $(\mathbf{z}, \mathrm{c}) \sim \rho$ (i.e., $\mathbf{z} \sim \gamma(\cdot)$ and independently $\mathrm{c} \sim p(\cdot)$), then $(f_{\theta,\phi}(\mathbf{z}, \mathrm{c}), \mathrm{c}) \sim \eta$ (i.e., its density is $p_\theta(\mathbf{x})p(\mathrm{c}|\mathbf{x})$).

Proof. Let ϕ be a parameter set that satisfies the two conditions stated in the theorem.

The first condition is $\mathbb{E}_{(\mathbf{z},\mathrm{c})\sim\nu_{t_0}}[d(f_{\theta,\phi}(\mathbf{z}, \mathrm{c}), f_\theta(\mathbf{z}))] = 0$. Recall that ν_{t_0} is the distribution with density $\gamma(\mathbf{z})p(\mathrm{c}|f_\theta(\mathbf{z}))$. According to the Boundary Loss Lemma (Lemma 2), this zero loss implies that the push-forward measure $(T_\phi)_\# \nu_{t_0}$ is equal to the target distribution η. So, $(T_\phi)_\# \nu_{t_0} = \eta$.

The second condition is $\sum_{k=0}^{N-1} \mathrm{MMD}^2((T_\phi)_\# \nu_{t_{k+1}}, (T_\phi)_\# \nu_{t_k}) = 0$. Since $\mathrm{MMD}^2(P, Q) \geq 0$ for any probability distributions P, Q, for the sum of non-negative terms to be zero, each individual term in the sum must be zero. Therefore, for each $k \in \{0, 1, \ldots, N-1\}$:

$$\mathrm{MMD}^2((T_\phi)_\# \nu_{t_{k+1}}, (T_\phi)_\# \nu_{t_k}) = 0$$

Assuming MMD is based on a characteristic kernel, $\mathrm{MMD}^2(P, Q) = 0$ if and only if $P = Q$. Thus, for each $k \in \{0, 1, \ldots, N-1\}$:

$$(T_\phi)_\# \nu_{t_{k+1}} = (T_\phi)_\# \nu_{t_k}$$

The result from step 2 implies a chain of equalities for the push-forward measures generated by T_ϕ from the sequence of input distributions ν_{t_k}:

$$\begin{aligned}
(T_\phi)_\# \nu_{t_N} &= (T_\phi)_\# \nu_{t_{N-1}} \\
&= (T_\phi)_\# \nu_{t_{N-2}} \\
&\vdots \\
&= (T_\phi)_\# \nu_{t_1} \\
&= (T_\phi)_\# \nu_{t_0}
\end{aligned}$$

So, we have $(T_\phi)_\# \nu_{t_N} = (T_\phi)_\# \nu_{t_0}$.

From the first condition, we established that $(T_\phi)_\# \nu_{t_0} = \eta$. Substituting this into the equality chain:

$$(T_\phi)_\# \nu_{t_N} = \eta$$

From Lemma A.3, we know that ν_{t_N} (which corresponds to $p_t(\mathbf{z}, \mathrm{c})$ at $t = t_N = 1$) is the independent latent-condition distribution ρ. The density of ρ is $p_\rho(\mathbf{z}, \mathrm{c}) = \gamma(\mathbf{z})p(\mathrm{c})$. Substituting $\nu_{t_N} = \rho$:

$$(T_\phi)_\# \rho = \eta$$

This is the desired conclusion. If $(\mathbf{z}, \mathrm{c})$ is sampled from ρ (meaning $\mathbf{z} \sim \gamma(\cdot)$ independently of $\mathrm{c} \sim p(\cdot)$) and then transformed by T_ϕ (i.e., forming $(f_{\theta,\phi}(\mathbf{z}, \mathrm{c}), \mathrm{c})$), the resulting distribution is the target data-condition distribution η (which has density $p_\theta(\mathbf{x})p(\mathrm{c}|\mathbf{x})$).

□

Remark 1 (Lemma 2 in main paper)**.** *Specifically, when we take $N = 1$ (particle number) in MMD loss, and take we have some specific kernel choice:*

- $k(x, y) = -\|x - y\|^2$*, although it is not a proper positive definite kernel required by MMD, we find it works well in practice*
- $k(x, y) = \mathrm{c} - \sqrt{\|x - y\|^2 + \mathrm{c}^2}$ *is a conditionally positive definite kernel.*

Then the summed MMD Loss

$$L_{total}(\phi) = \sum_{k=0}^{N-1} \mathrm{MMD}^2((T_\phi)_\# \nu_{t_{k+1}}, (T_\phi)_\# \nu_{t_k}) = 0,$$

can be implemented in a practical way:

$$L_{total}(\phi) = \sum_{k=0}^{N-1} \mathbb{E}_{\gamma(\mathbf{z})p(\mathrm{c}|f_\theta(\mathbf{z}))} \mathbb{E}_{\gamma(\mathbf{w})} d(f_{\theta,\phi}(\alpha_{t_{k+1}}\mathbf{z} + \sigma_{t_{k+1}} w, \mathrm{c}), f_{\theta,\phi}(\alpha_{t_k}\mathbf{z} + \sigma_{t_k} w, \mathrm{c})),$$

where d is l_2 loss or pseudo-huber loss, other kernel-induced losses also work.

B Experiment Details

One-step generator We adopt Diff-Instruct [3] for pre-training the one-step generator. We adopt the AdamW optimizer. The β_1 is set to be 0, and the β_2 is set to be 0.95. The learning rate for the generator is $2e - 6$, the learning rate for fake score is $1e - 5$. We apply gradient norm clipping with a value of 1.0 for both the generator and fake score. We use batch size of 256.

Controllable Generation We use Contorlnet's architecture [15] for training. We adopt the AdamW optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.95$, and the learning rate of $1e - 5$. We use batch size of 128.

Image-prompted Geneartion We use IP-adapter's architecture [39] for training. We adopt the AdamW optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.95$, and the learning rate of $1e - 4$. We use batch size of 128. We use a probability of 0.05 to drop text during training.

C Limitations

Our model shares common challenges with other controllable text-to-image diffusion models, particularly regarding fairness considerations and precise detail handling. We plan to explore these ongoing challenges in the generation domain in our future works, to improve the model's performance in text synthesis, fairness, and fine-grained control.

D Broader Impacts

This work presents NCT, a method that can inject new controls into pre-trained one-step generators. From a positive perspective, this academic contribution has potential for widespread industrial adoption, where its computational efficiency could reduce energy consumption and provide environmental advantages. Conversely, malicious use of such rapid generation technologies could facilitate the faster production of harmful content. While our focus remains on scientific advancement, we are committed to mitigating risks through measures such as removing harmful material from training datasets.

E Safeguards

The NCT is trained on an internally curated dataset that has undergone rigorous human and machine-based filtering to exclude harmful or violent content.