

# Transition Models: Rethinking the Generative Learning Objective

Zidong Wang<sup>1,2,\*</sup>, Yiyuan Zhang<sup>1,2,\*†</sup>, Xiaoyu Yue<sup>2,3</sup>, Xiangyu Yue<sup>1</sup>,

Yangguang Li<sup>1,†</sup>, Wanli Ouyang<sup>1,2</sup>, Lei Bai<sup>2,†</sup>

<sup>1</sup>MMLab, CUHK

<sup>2</sup>Shanghai AI Lab

<sup>3</sup>USYD

{wangzdz2022, yiyuanzhang.ai}@gmail.com, {xyyue, wlouyang}@ie.cuhk.edu.hk

Code: <https://github.com/WZDTHU/TiM>

## Abstract

A fundamental dilemma in generative modeling persists: iterative diffusion models achieve outstanding fidelity, but at a significant computational cost, while efficient few-step alternatives are constrained by a hard quality ceiling. This conflict between generation steps and output quality arises from restrictive training objectives that focus exclusively on either infinitesimal dynamics (PF-ODEs) or direct endpoint prediction. We address this challenge by introducing an exact, continuous-time dynamics equation that analytically defines state transitions across any finite time interval  $\Delta t$ . This leads to a novel generative paradigm, Transition Models (TiM), which adapt to arbitrary-step transitions, seamlessly traversing the generative trajectory from single leaps to fine-grained refinement with more steps. Despite having only 865M parameters, TiM achieves state-of-the-art performance, surpassing leading models such as SD3.5 (8B parameters) and FLUX.1 (12B parameters) across all evaluated step counts. Importantly, unlike previous few-step generators, TiM demonstrates monotonic quality improvement as the sampling budget increases. Additionally, when employing our native-resolution strategy, TiM delivers exceptional fidelity at resolutions up to  $4096 \times 4096$ .

## 1. Introduction

Diffusion models have emerged as the dominant paradigm in visual content generation, producing state-of-the-art results across various domains [9, 20, 35, 52, 54, 83]. They generate samples from noise via iterative denoising, a process that can be formulated as numerical integration of either the reverse-time Stochastic Differential Equation (SDE) or the corresponding Probability-Flow Ordinary Differential Equation (PF-ODE), with related discrete-time solvers also widely used [46, 66, 69]. Despite its effectiveness, iterative denoising often entails a large Number of

\*: Equal contribution. †: Project lead. ‡: Corresponding authors: bailei@pjlab.org.cn, liyangguang256@gmail.com.

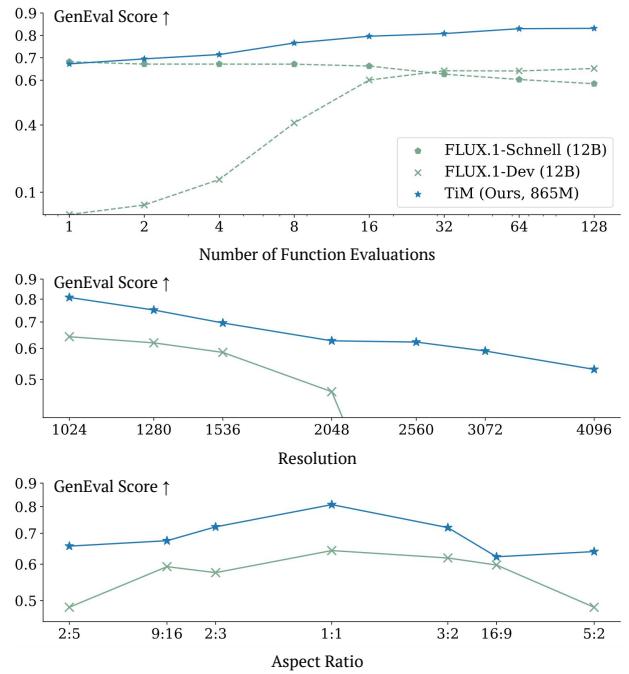


Figure 1. TiM’s superior performance across different NFEs, resolutions, and aspect ratios. On the GenEval [27] benchmark, TiM outperforms Flux.1 models [5, 6] at different NFEs (top,  $1024 \times 1024$ ), at higher resolutions (middle,  $1024 \times 1024$  to  $4096 \times 4096$ ), and diverse aspect ratios (bottom,  $2 : 5$  to  $5 : 2$ ).

Function Evaluations (NFEs)—approximately proportional to the number of integration steps—leading to increased inference latency and compute cost.

In contrast, recent approaches reduce step counts by avoiding explicit multi-step integration. Consistency models [45, 67, 70] impose PF-ODE self-consistency across different noise levels, while distribution-distillation methods [42, 61, 62, 84, 93] train students to approximate teacher distributions with fewer denoising steps. Shortcut [21], FlowMap [7, 57], and MeanFlow [26, 53] learn the average (shortcut) velocity along the flow-matching trajectory via a self-consistency objective. The principle is that a single large step should approximate the integral of

all smaller, instantaneous steps. However, by averaging the entire trajectory, *they irrevocably discard the fine-grained local dynamics necessary for high-fidelity refinement*. This leads to performance saturation—while effective for few-step generation, it offers no gains from additional sampling budget. Moreover, despite these methods deliver strong few-step results; their compute-quality scaling is typically weaker than that of high-NFE diffusion models: *quality gains plateau after only a few steps, and asymptotic performance remains below traditional multi-step diffusion.*

Thus, the entire field converges on a fundamental, yet flawed, compromise [26, 31, 44, 70, 84]: models either achieve high fidelity at the cost of computational efficiency (e.g., diffusion models), or they gain efficiency by sacrificing the very dynamics needed for high-fidelity refinement (e.g., few-step models). The root of this dilemma is not architectural, but a learning objective. It stems from a foundational choice in how these models are taught to generate, not the specific components they are built from. This trade-off is a direct and inevitable consequence of the chosen *granularity of supervision*. On one hand, local training methods that model instantaneous dynamics (such as those consistent with PF-ODEs/SDEs [31, 44, 69]) achieve high accuracy with small step sizes ( $\Delta t$ ) and scale well to many-step generation. However, their performance degrades significantly in few-step regimes. On the other hand, finite-horizon training, which learns a direct mapping over a fixed interval (flow-map and consistency models [7, 26, 70]), excels at few-step generation. Yet, these models see diminishing returns from additional intermediate steps unless specifically trained with complex, multi-interval objectives. This reveals a persistent dilemma: *objectives that model instantaneous dynamics and those that learn finite-interval mappings each entail inherent limitations*. This motivates the question: **What is an appropriate learning objective for generative models?**

We attempt to answer this question from the following perspectives:

1) Diffusion training [46, 47, 91] learns a local PF-ODE field whose numerical integration is accurate only in the small-step limit  $\Delta t \rightarrow 0$ . With large steps, the discretization error dominates; therefore, *the objective should be flexible in terms of step sizes*.

2) Few-step objectives supervise an endpoint map but do not learn a compositional flow: without an approximate semigroup over time, extra steps change the path rather than refine it, causing schedule sensitivity and early saturation. Therefore, *the objective requires a consistency along the trajectory, where intermediate steps act as refinements along a single trajectory, rather than deviations onto new ones*, which makes the sampler insensitive to step schedules and enables steady quality improvements with more steps.

Consequently, we argue that a generative model should

learn a *versatile denoising operator*, parameterized by the desired interval  $\Delta t$ . By learning the transitions between any state  $\mathbf{x}_t$  to a previous state  $\mathbf{x}_{t-\Delta t}$  for an *arbitrary*  $\Delta t$ , the generative model is no longer approximating a differential equation or a statistic map. *Instead, it is learning the solution manifold<sup>1</sup> of the generative process itself*. This formulation inherently unifies the local and finite-horizon perspectives, yielding a sampler that is both a powerful few-step generator and a precise, refinable integrator. Since the training objective is to learn the transitions between any state to a previous state, thus, it is named as Transition Models (TiM), which parameterize state-to-state transitions along the PF-ODE trajectory for arbitrary time intervals.

We validate TiM’s effectiveness through extensive experiments on text-to-image and class-guided image generation. As shown in Figure 1, TiM shows superior performance across different NFEs, resolutions, and aspect ratios. On the GenEval [27] benchmark, our compact 865M parameter model, TiM, establishes a new state-of-the-art. It achieves a score of 0.67 with a single function evaluation (1-NFE) and scales to 0.83 at 128-NFE, outperforming billion-scale industrial models including SD3.5-Large [20] (8B) and FLUX.1-Dev [5] (12B).

## 2. Related Work

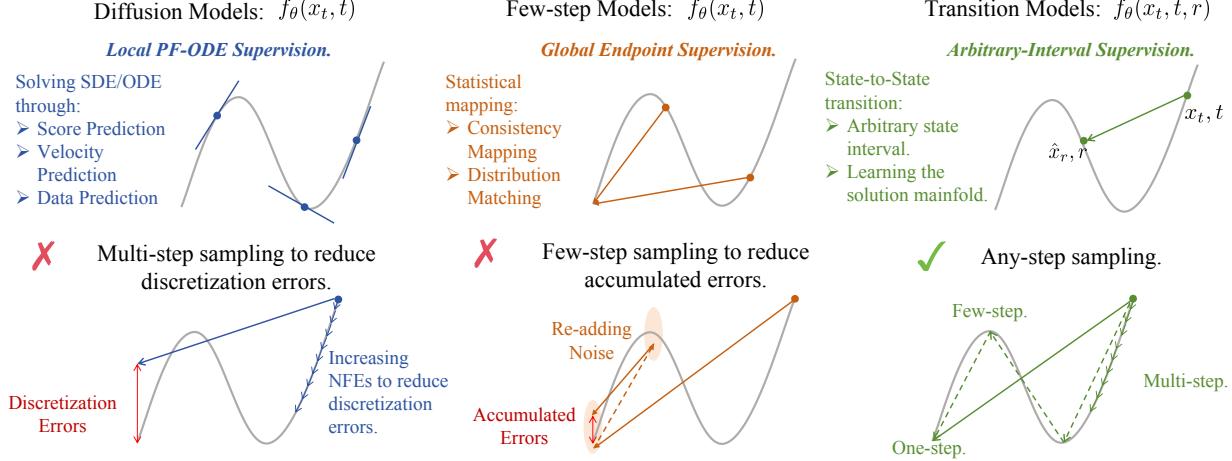
**Diffusion and Consistency Models.** Generative modeling has seen two dominant paradigms. Diffusion models [31, 35, 44] iteratively solve a PF-ODE/SDE, achieving high quality but requiring many function evaluations (NFEs). In contrast, Consistency Models [70] learn a direct mapping for few-step generation but suffer from performance saturation and complex training requirements (e.g., pre-training and stabilization [45, 67]). While recent methods like FlowMap [7, 57] and MeanFlow [26, 53] enable training CM-like models from scratch, they inherit the same limitation of stagnating quality with more steps.

To break this impasse, we propose a new learning principle: mastering state transitions over arbitrary time intervals. This transforms the model from a brittle integrator or fixed-endpoint mapper into a robust navigator on the data manifold, preserving few-step efficiency while supporting monotonic refinement by using more steps.

**Text-to-Image Generation with Few-steps.** Efficient text-to-image (T2I) sampling is currently dominated by distillation. These methods fall into two main camps: distribution distillation (e.g., SD-Turbo [61, 62], DMD [84, 85]), which matches the teacher’s output distribution, and trajectory distillation (e.g., LCM [48], PCM [77]), which mimics its generation path. Hybrid methods [55] combine both.

---

<sup>1</sup>solution manifold of a PF-ODE is the high-dimensional geometric surface formed by the collection of all possible generative trajectories that lead from noise to data.



**Figure 2. Illustration of Different Generative Paradigms.** While conventional diffusion models learn the local vector field and few-step models learn a fixed endpoint map (a single large step), our Transition Models (TiM) are trained to master arbitrary state-to-state transitions. This approach allows TiM to learn the entire solution manifold of the generative process, unifying the few-step and many-step regimes within a single, powerful model.

However, all these approaches are fundamentally limited: **1)** they require a large, pre-trained teacher, leading to complex and costly pipelines, and **2)** they produce brittle, few-step-only models whose quality stagnates or degrades with more steps. We bypass these limitations entirely by introducing TiM, the first T2I generator trained from scratch that masters arbitrary-step sampling, delivering strong few-step results that monotonically improve with more compute.

### 3. Transition Models

In this section, we first analyze the limitations of PF-ODE supervision in diffusion models, which constrain the state transition to a local, infinitesimal interval. To address the limitations, we generalize diffusion’s local state transition to an arbitrary-interval state transition, as illustrated in Fig. 2, from which we derive a novel mathematical identity that links the state  $\mathbf{x}_t$ , the interval  $\Delta t$ , and the network  $\mathbf{f}_\theta$ . From this identity, we formulate a training objective governing state evolution over any interval  $\Delta t$ , and further propose two theoretically motivated improvements for scalable and stable training. Finally, we present the architecture improvements for effective transition modeling.

#### 3.1. Limitation of PF-ODE Supervision

Given the noise distribution  $\varepsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  and the data distribution  $\mathbf{x} \sim p_{\text{data}}(\mathbf{x})$ , diffusion models learn to map the noise distribution to the data distribution. Given time range  $t \in [0, T]$ , the forward process utilizes coefficients  $\alpha_t$  and  $\sigma_t$ , such that  $\mathbf{x}_t = \alpha_t \mathbf{x} + \sigma_t \varepsilon$ , which can be described by an SDE [69]:

$$d\mathbf{x}_t = \mathbf{f}(\mathbf{x}_t, t)dt + g(t)d\mathbf{w}, \quad (1)$$

where  $\mathbf{w}$  is the standard Wiener process,  $\mathbf{f}(\mathbf{x}_t, t) = \frac{\dot{\alpha}_t}{\alpha_t} \mathbf{x}_t$  is the drift coefficient and  $g(t) = 2\sigma_t \dot{\alpha}_t - 2\frac{\dot{\alpha}_t}{\alpha_t} \sigma_t^2$  is the diffusion coefficient [35, 45, 69, 72]. Anderson [3] and Song et al. [69] have shown that the forward process can be reversed by solving the reverse-time SDE from or equivalently the probability flow ODE (PF-ODE)<sup>2</sup>:

$$\begin{aligned} \frac{d\mathbf{x}_t}{dt} &= \mathbf{f}(\mathbf{x}_t, t) - \frac{1}{2}g(t)^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) \\ &= \frac{d\alpha_t}{dt} \mathbf{x} + \frac{d\sigma_t}{dt} \varepsilon, \end{aligned} \quad (2)$$

where  $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) = -\frac{\varepsilon}{\sigma_t}$  denotes the score function.

Thus, a diffusion model can be parameterized as  $\mathbf{f}_\theta(\mathbf{x}_t, t) = \mathbf{F}_\theta(\mathbf{x}_t, c_{\text{noise}}(t))$ , where  $\theta$  denotes the parameters of the neural network and  $c_{\text{noise}}(t)$  is the time scaling function. The training objective can be given by:

$$\mathbb{E}_{\mathbf{x}, \varepsilon, t}[w(t)d(\mathbf{f}_\theta(\mathbf{x}_t, t) - (\hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \varepsilon))], \quad (3)$$

where  $\hat{\alpha}_t$  and  $\hat{\sigma}_t$  are the coefficients of diffusion target,  $w(t)$  is a weighting function,  $d(\cdot, \cdot)$  is a metric function such as the L2 loss  $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2^2$ .

Despite different transports<sup>3</sup> have instantiates coefficients  $\alpha_t$  and  $\sigma_t$ , the training objectives are equivalent to supervising the PF-ODE field<sup>4</sup>. During sampling, diffusion models

<sup>2</sup>Song et al. [69] have shown that the PF-ODE trajectory has the same marginal probability as the reverse-time SDE:  $d\mathbf{x}_t = [\mathbf{f}(\mathbf{x}_t, t) - \frac{1}{2}g(t)^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)]dt + g(t)d\mathbf{w}$ .

<sup>3</sup>For convenience, we elaborate the coefficients  $\alpha_t$ ,  $\sigma_t$ ,  $\hat{\alpha}_t$ , and  $\hat{\sigma}_t$  of different diffusion transports in Tab. 7

<sup>4</sup>For example, in VE-SDE [69], with coefficients  $\alpha_t = 1$ ,  $\sigma_t = t$ , the PF-ODE is:  $\frac{d\mathbf{x}_t}{dt} = \varepsilon$ , and the training objective is  $-\varepsilon$ . In OT-FM [43], with  $\alpha_t = 1 - t$ ,  $\sigma_t = t$ , the PF-ODE is:  $\frac{d\mathbf{x}_t}{dt} = \varepsilon - \mathbf{x}$ , which directly matches the training objective.

solve this PF-ODE, integrated from  $t = T$  to  $t = 0$  using numerical solvers. To reduce discretization error and preserve the learned continuous-time dynamics, practical solvers [47, 66, 91] typically require a small step size (i.e.,  $\Delta t \rightarrow 0$ ) or many sub-steps per interval (i.e., high-order solvers), thus inducing huge NFEs.

### 3.2. State Transition

The derivation begins with the general mathematical form for a state transition between points  $(\mathbf{x}_t, \mathbf{x}_r)$  on a PF-ODE trajectory, as given in Eq. (6). *The central principle is to treat this form not as a numerical approximation, but as an exact identity that must hold for any interval  $\Delta t = t - r$ .* It allows us to formulate a *general state transition dynamic* (Eq. (8)) that is valid across any interval. Consequently, the model’s training objective is no longer constrained to approximating a local solution of the PF-ODE. Instead, it is trained to learn the *entire solution manifold of the generative process*. By internalizing this global structure, the model inherently acquires the ability to perform inference over arbitrary step sizes, from large, single leaps to fine-grained, iterative refinement. We illustrate our derivation process step-by-step as follows:

**State Transition.** Given noisy state  $\mathbf{x}_t = \alpha_t \mathbf{x} + \sigma_t \boldsymbol{\varepsilon}$ , a diffusion model  $\mathbf{f}_\theta(\mathbf{x}_t, t)$  is optimized towards the target  $\hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \boldsymbol{\varepsilon}$ , leading to the  $x$ -prediction and  $\boldsymbol{\varepsilon}$ -prediction:

$$\hat{x} = \frac{\hat{\sigma}_t \mathbf{x}_t - \sigma_t \mathbf{f}_\theta(\mathbf{x}_t, t)}{\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t}, \quad \hat{\boldsymbol{\varepsilon}} = \frac{\alpha_t \mathbf{f}_\theta(\mathbf{x}_t, t) - \hat{\alpha}_t \mathbf{x}_t}{\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t}. \quad (4)$$

Using the prediction  $\hat{x}$  and  $\hat{\boldsymbol{\varepsilon}}$ , arbitrary previous state  $\mathbf{x}_r$  ( $r < t$ ) can be represented as:

$$\begin{aligned} \mathbf{x}_r &= \alpha_r \hat{\mathbf{x}} + \sigma_r \hat{\boldsymbol{\varepsilon}} \\ &= \frac{(\alpha_r \hat{\sigma}_t - \sigma_r \hat{\alpha}_t) \mathbf{x}_t + (\sigma_r \alpha_t - \alpha_r \sigma_t) \mathbf{f}_\theta(\mathbf{x}_t, t)}{\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t}. \end{aligned} \quad (5)$$

This represents the general form of a first-order state transition on the PF-ODE Trajectory.

**State Transition Identity.** Different from the diffusion model, our transition model learns the state transition  $\mathbf{f}_\theta(\mathbf{x}_t, t, r) = \mathbf{F}_\theta(\mathbf{x}_t, c_{\text{noise}}(t), c_{\text{noise}}(r))$  between state  $\mathbf{x}_t$  and  $\mathbf{x}_r$ . By introducing  $\mathbf{f}_\theta(\mathbf{x}_t, t, r)$  to Eq. (5), we obtain:

$$\mathbf{x}_r = \frac{(\alpha_r \hat{\sigma}_t - \sigma_r \hat{\alpha}_t) \mathbf{x}_t + (\sigma_r \alpha_t - \alpha_r \sigma_t) \mathbf{f}_\theta(\mathbf{x}_t, t, r)}{\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t}. \quad (6)$$

Here, we define  $A_{t,r} := \frac{\alpha_r \hat{\sigma}_t - \sigma_r \hat{\alpha}_t}{\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t}$ ,  $B_{t,r} := \frac{\sigma_r \alpha_t - \alpha_r \sigma_t}{\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t}$ , and  $\mathbf{f}_{\theta,t,r} := \mathbf{f}_\theta(\mathbf{x}_t, t, r)$  for simplicity, then we differentiate both sides with respect to  $t$  and rearranging yields:

$$\begin{aligned} \frac{d\mathbf{x}_r}{dt} &= \frac{d}{dt}(A_{t,r} \mathbf{x}_t + B_{t,r} \mathbf{f}_{\theta,t,r}) \Rightarrow \\ \mathbf{x}_t \frac{dA_{t,r}}{dt} + A_{t,r} \frac{d\mathbf{x}_t}{dt} &= -\mathbf{f}_{\theta,t,r} \frac{dB_{t,r}}{dt} - B_{t,r} \frac{d\mathbf{f}_{\theta,t,r}}{dt}, \end{aligned} \quad (7)$$

which can be further simplified as follows (detailed in Appx. A.1):

$$\begin{aligned} \frac{d(B_{t,r} \cdot (\hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \boldsymbol{\varepsilon} - \mathbf{f}_{\theta,t,r}))}{dt} &= 0 \Rightarrow \\ \underbrace{(\hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \boldsymbol{\varepsilon} - \mathbf{f}_{\theta,t,r})}_{\text{PF-ODE supervision}} \frac{dB_{t,r}}{dt} + B_{t,r} \underbrace{\frac{d(\hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \boldsymbol{\varepsilon} - \mathbf{f}_{\theta,t,r})}{dt}}_{\text{time-slope matching}} &= 0. \end{aligned} \quad (8)$$

We denote Equation (8) as the State Transition Identity, a product-derivative invariant. The State Transition Identity,  $\frac{d}{dt}(B_{t,r} \cdot h(t)) = 0$ , where  $h(t) = \hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \boldsymbol{\varepsilon} - \mathbf{f}_{\theta,t,r}$  is the instantaneous residual, imposes a powerful two-fold constraint on the generative model  $\mathbf{f}_\theta$ .

- **Implicit Trajectory Consistency:** The identity dictates that the weighted residual  $B_{t,r} h(t)$  must be constant for any starting time  $t$  leading to the same target  $\mathbf{x}_r$ . This directly enforces path consistency: the direct map ( $t \rightarrow r$ ) must be equivalent to any composition of intermediate steps, such as  $(t \rightarrow s) \circ (s \rightarrow r)$ . This property (Eq. (8)), absent in standard consistency models, is the core mechanism that makes TiM robust to sampling schedules and enables monotonic refinement.
- **Time-Slope Matching:** Unpacking the product rule reveals that  $(\frac{d}{dt} B_{t,r}) h(t) + B_{t,r} (\frac{d}{dt} h(t)) = 0$ . Unlike conventional diffusion training, which only minimizes the residual’s value ( $h(t) \rightarrow 0$ ), our objective forces the model to also minimize the residual’s temporal derivative ( $\frac{d}{dt} h(t) \rightarrow 0$ ). This higher-order supervision compels the model to learn a smoother solution manifold, preserving coherence during large-step sampling and ensuring stable refinement with smaller steps.

Derived from State Transition Identity (Eq. (8)), we obtain the learning target  $\hat{\mathbf{f}}$ :

$$\hat{\mathbf{f}} = \hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \boldsymbol{\varepsilon} + \frac{B_{t,r}}{\frac{dB_{t,r}}{dt}} \left( \frac{d\hat{\alpha}_t}{dt} \mathbf{x} + \frac{d\hat{\sigma}_t}{dt} \boldsymbol{\varepsilon} - \frac{d\mathbf{f}_{\theta^-, t, r}}{dt} \right), \quad (9)$$

where  $\theta^-$  indicates the fixed network parameter  $\theta$  and  $\frac{d\mathbf{f}_{\theta^-, t, r}}{dt}$  is the time derivative of the network.

### 3.3. Scalability and Stability in TiM Training

#### Remark 1: Making TiM Training Scalable.

A critical challenge in implementing our training target (Eq. (9)) is the computation of the network’s time derivative,  $\frac{d\mathbf{f}_{\theta^-, t, r}}{dt}$ . Prior work, such as MeanFlow [26, 53, 57] and SCM [45], relies on the Jacobian-Vector Product (JVP) for this task. However, JVP presents a *fundamental road-block to scalability*. It is not only compute-intensive but, more cripplingly, its reliance on backward-mode automatic differentiation is **incompatible with essential training optimizations**, including FlashAttention [17] and distributed frameworks of FSDP [89]. This incompatibility has effectively rendered JVP-based methods impractical for training billion-parameter foundation models.

Method	Operator		Training		FID		
	FLOPs (G)	Latency (ms)	Throughput (/s)	Memory (GiB)	NFE=1	NFE=8	NFE=50
JVP	48.29	213.14	1.80	14.89	<b>49.75</b>	26.22	18.11
DDE	<b>24.14</b>	<b>110.08</b>	<b>2.40</b>	15.23	49.91	<b>26.09</b>	<b>17.99</b>

Table 1. **Derivative Calculation Comparison.** We utilize a TiM-B/4 model for latency, throughput, and memory measurement, with a batch size of 256 on a NVIDIA-A100 GPU using BF16 precision.

We break this barrier with the **Differential Derivation Equation (DDE)**, a principled and highly efficient finite-difference approximation:

$$\frac{df_{\theta^-, t, r}}{dt} \approx \frac{f_{\theta^-}(\mathbf{x}_{t+\epsilon}, t + \epsilon, r) - f_{\theta^-}(\mathbf{x}_{t-\epsilon}, t - \epsilon, r)}{2\epsilon}. \quad (10)$$

As shown in Tab. 1, DDE is not only  $\sim 2\times$  faster than JVP but, crucially, its forward-pass-only structure is *natively compatible with FSDP*. This compatibility transforms a previously unscalable training process into one ready for large-scale deployment, making TiM the first model of its kind practical for from-scratch, billion-parameter pre-training.<sup>5</sup>

### Remark 2: Making TiM Training Stable.

In addition to scalability, a key challenge in training with arbitrary intervals is managing gradient variance. For example, transitions over very large intervals ( $\Delta t \rightarrow t$ ) are easier to make loss spikes. To mitigate this, we introduce a loss weighting scheme that prioritizes short-interval transitions, which are more frequent and provide a more stable learning signal.

The weighting function,  $w(t, r)$ , is a composition of a time-warping function  $\tau(\cdot)$  and a kernel function  $k(\cdot, \cdot)$ :

$$w(t, r) = k(\tau(t), \tau(r)). \quad (11)$$

Here,  $\tau(\cdot)$  is a monotonic function that re-parameterizes the time axis. For our final model, we use a tangent space transformation, which effectively stretches the time domain, yielding the specific weighting:

$$w(t, r) = (\sigma_{\text{data}} + \tan(t) - \tan(r))^{-\frac{1}{2}}, \quad (12)$$

where  $\sigma_{\text{data}}$  is the standard deviation of the clean data.<sup>6</sup>

**Learning Objective.** Our theoretical framework culminates in a scalable and stable learning objective. We train the network  $f_\theta$  to predict the dynamic target  $\hat{f}$  in Eq. (9). To manage gradient variance and ensure stable convergence, this is weighted by the interval function  $w(t, r)$  from Eq. (11). This results in the final TiM objective:

$$\mathbb{E}_{\mathbf{x}, \epsilon, t, r} \left[ w(t, r) \cdot d \left( f_\theta(\mathbf{x}_t, t, r) - \hat{f} \right) \right]. \quad (13)$$

<sup>5</sup>We provide a detailed analysis of DDE in the Appendix Tab. 10

<sup>6</sup>In the Appendix, we conduct an in-depth comparison of alternative weighting schemes is provided in Tab. 12.

This objective generalizes the standard PF-ODE supervision to arbitrary state-to-state transitions. The practical implementation, enabled by our efficient DDE calculation, is detailed in Algorithm 1. We summarize the specific parameterizations for various transport choices in Tab. 7.

### 3.4. Improved Architectures

We conduct a series of experiments to explore architectural modifications based on DiT [52] for effective state transition learning in Tab. 4. We illustrate the exact architectures in the Appendix A.

**Decoupled Time and Interval Embeddings.** To enable the model to distinguish between the absolute time  $t$  and the transition interval  $\Delta t$ , we introduce a decoupled embedding strategy. We employ two independent time encoders,  $\phi_t$  and  $\phi_{\Delta t}$ , to parameterize these two quantities. Their outputs are summed to form the final time-conditioning vector:

$$\mathbf{E}_{t, \Delta t} = \phi_t(t) + \phi_{\Delta t}(\Delta t). \quad (14)$$

This time embedding is then integrated with task-specific conditioning as follows:

- For class-guided generation, the class embedding  $\mathbf{E}_c$  is added to the time embedding, and the resulting sum,  $\mathbf{E}_{t, \Delta t} + \mathbf{E}_c$ , modulates the AdaLN layers of the model.
- For text-to-image generation, the conditioning pathways are separated. The time embedding  $\mathbf{E}_{t, \Delta t}$  solely modulates the AdaLN layers, while textual features from the prompt are injected via dedicated cross-attention mechanisms.

**Interval-Aware Attention.** We assume that the optimal way to model spatial dependencies is conditional on the transition interval  $\Delta t$ . A large step ( $\Delta t \rightarrow t$ ) may require global, coarse-grained restructuring, while a small step ( $\Delta t \rightarrow 0$ ) demands fine-grained, local refinement. Standard self-attention, which is agnostic to this context, is inappropriate for this task. We therefore introduce the Interval-Aware Attention, a mechanism that infuses the transition interval's magnitude directly into the query, key, and value computations. Specifically, we project both the spatial tokens  $\mathbf{z}$  and the interval embedding  $\mathbf{E}_{\Delta t}$  into a shared representational space before the attention calculation:

$$\begin{aligned} \mathbf{q} &= \mathbf{z}\mathbf{W}_q + \mathbf{b}_q + \mathbf{E}_{\Delta t}\mathbf{W}'_q, \\ \mathbf{k} &= \mathbf{z}\mathbf{W}_k + \mathbf{b}_k + \mathbf{E}_{\Delta t}\mathbf{W}'_k, \\ \mathbf{v} &= \mathbf{z}\mathbf{W}_v + \mathbf{b}_v + \mathbf{E}_{\Delta t}\mathbf{W}'_v. \end{aligned} \quad (15)$$

Here,  $(\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v)$  are the primary projection matrices for the spatial tokens, while  $(\mathbf{W}'_q, \mathbf{W}'_k, \mathbf{W}'_v)$  are dedicated projection matrices that modulate the attention based on the interval embedding.

Model	Param.	NFE	Overall↑	Single Obj.	Two Obj.	Counting	Colors	Position	Attr. Binding
<b>Autoregressive Models</b>									
Emu3-Gen [79]	-	-	0.54	0.98	0.71	0.34	0.81	0.17	0.21
GPT-4o [1]	-	-	0.84	0.99	0.92	0.85	0.92	0.75	0.61
<b>Multi-step Diffusion Models</b>									
SD2.1 [56]	865M	100	0.50	0.98	0.51	0.44	0.85	0.07	0.17
SD-XL [54]	2.6B	100	0.55	0.98	0.74	0.39	0.85	0.15	0.23
Seedream2.0 [28]	-	-	0.84	1.0	0.98	0.91	0.94	0.47	0.75
SD3.5-Medium [20]	2B	100	0.63	0.98	0.78	0.50	0.81	0.24	0.52
SD3.5-Large [20]	8B	128	0.69	0.99	0.89	0.67	0.81	0.24	0.56
SANA-1.5 [81]	4.6B	40	0.81	0.99	0.93	0.86	0.84	0.59	0.65
FLUX.1-Dev [5]	12B	128	0.65	0.98	0.79	0.69	0.76	0.21	0.48
<b>Few-step Distilled Diffusion Models</b>									
SDXL-LCM [48]	2.6B	8	0.40	0.97	0.50	0.12	0.67	0.09	0.07
SDXL-Turbo [62]	2.6B	8	0.50	0.99	0.75	0.07	0.89	0.11	0.20
Hyper-SDXL [55]	2.6B	8	0.46	0.92	0.58	0.26	0.78	0.11	0.15
SANA-Sprint [16]	1.6B	8	0.72	1.0	0.88	0.56	0.87	0.56	0.47
SD3.5-Turbo [61]	8B	8	0.66	0.99	0.81	0.62	0.79	0.25	0.48
FLUX.1-Schnell [5]	12B	1	0.68	0.99	0.88	0.63	0.78	0.27	0.53
<b>Transition Models</b>									
TiM	865M	1	0.67	0.98	0.75	0.52	0.80	0.54	0.44
		8	0.76	0.99	0.87	0.61	0.88	0.63	0.61
		128	0.83	1.0	0.91	0.73	0.91	0.73	0.71

Table 2. **System-level quality comparison of TiM and SOTA methods on GenEval benchmark.** In the table, 1-NFE denotes a single sampling step; 8-NFE corresponds to four sampling steps with CFG, and other multi-NFE follow the same convention. Compared with multi-step diffusion models and few-step distilled models, TiM offers any-step generation, delivering strong few-step performance and exhibiting consistent, stable improvements as NFE increases.

Model	Param.	NFE	MJHQ30K		Overall↑	DPGBench				
			FID↓	CLIP↑		Global	Entity	Attribute	Relation	Other
PixArt- $\alpha$ [12]	610M	100	6.14	27.55	71.11	74.97	79.32	78.60	82.57	76.96
PixArt- $\Sigma$ [14]	610M	100	6.15	28.26	80.54	86.89	82.89	88.94	86.59	87.68
SDXL [54]	2.6B	100	6.63	29.03	74.65	83.27	82.43	80.91	86.76	80.41
Playground v2.5 [39]	2.6B	100	6.09	29.13	75.47	83.06	82.59	81.20	84.08	83.50
Hunyuan-DiT [41]	1.5B	100	6.54	28.19	78.87	84.59	80.59	88.01	74.36	86.41
SD3.5-Medium [20]	2B	100	11.92	27.83	84.08	87.90	91.01	88.83	80.70	88.68
SD3.5-Turbo [61]	8B	8	11.97	27.35	79.03	80.12	86.13	84.73	91.86	78.29
SD3.5-Large [20]	8B	32	14.68	27.88	83.21	84.27	88.99	87.35	93.28	80.35
FLUX.1-Schnell [6]	12B	8	7.94	28.14	84.94	86.62	90.82	88.35	93.45	82.00
FLUX.1-dev [5]	12B	32	9.19	27.27	83.32	81.46	90.02	87.50	92.72	82.39
TiM	865M	1	6.68	24.80	74.93	82.98	83.64	83.54	91.99	63.20
		8	5.28	26.10	81.30	82.01	88.31	87.81	93.37	70.80
		32	5.65	26.31	82.71	82.67	89.40	88.48	93.31	79.20

Table 3. **System-level quality comparison on MJHQ30K and DPGBench benchmarks.**

## 4. Experiments

### 4.1. Setup

We use SD-VAE [56] for ImageNet-256  $\times$  256 experiments and DC-AE [13] for text-to-image (T2I) experiments. Model architecture follows DiT [52], except the modifica-

tions in Sec. 3.4. For T2I generation, we use 33M images from public datasets [2, 11, 15, 18, 33, 63–65]. We train the T2I model with 865M parameters using the native-resolution training strategies for about 30 days using 16 NVIDIA-A100 GPUs. Gemma3-1B-it [74] is utilized as a text encoder. See more details in Appx. C.2. We report the

Method	NFE=1	NFE=8	NFE=50
<i>Training Objective</i>			
(a) Baseline (SiT-B/4 [49])	309.5	77.26	20.35
(b) TiM-B/4 (w/ JVP)	<b>49.75</b>	26.22	18.11
(c) TiM-B/4 (w/ DDE)	49.91	<b>26.09</b>	<b>17.99</b>
<i>Architecture</i>			
(d) Vanilla Architecture	56.22	28.75	20.37
(e) + Decoupled Time Embedding (De-TE)	49.91	26.09	17.99
(f) + Interval-Aware Attention (IA-Attn)	48.38	26.10	17.85
(g) + De-TE + IA-Attn	<b>48.30</b>	<b>25.05</b>	<b>17.43</b>
<i>Training Strategy</i> (on top of (g))			
(h) + Time-weighting	<b>47.46</b>	<b>24.62</b>	<b>17.10</b>

Table 4. **Ablation studies of Transition Models on the standard ImageNet-256 benchmark (FID↓).** We analyze the effect of training objectives, architecture, and training strategies.

Method	NFE=1	NFE=8	NFE=32	NFE=128
SD3.5-Turbo [61]	0.50	0.66	0.70	0.70
FLUX.1-Schnell [6]	0.68	0.67	0.63	0.58
SD3.5-Large [20]	0.00	0.50	0.69	0.70
FLUX.1-Dev [5]	0.00	0.40	0.64	0.65
<b>TiM</b>	0.67	0.76	0.80	0.83

Table 5. **Benchmarking generation quality across NFEs on the GenEval benchmark (score↑).** We compare a single TiM model against diffusion models (i.e., SD3.5-Large and FLUX.1-Dev) and distilled models (i.e., SD3.5-Turbo and FLUX.1-Schnell).

Number of Function Evaluations (NFE) to quantify sampling steps. When classifier-free guidance (CFG) is used, NFE doubles, because each step requires two model evaluations: one conditioned and one unconditioned. We provide the T2I experiments and ablation experiments on ImageNet-256 × 256 below and more results on class-guided image generation in Appxs. D.1 and D.2.

**Native-Resolution Training.** Previous methods [25, 28, 80] have shown the success of native-resolution training on resolution generalization; thus, we adopt this strategy for text-to-image generation, which preserves the original image resolution and aspect ratio information to the greatest extent possible. Given the wide resolution range, we increase noise for higher-resolution images and decrease it for lower-resolution ones. Following Esser et al. [20], we therefore apply resolution-dependent timestep shifting. Please see more details in Appx. C.2.

**Sampling.** Since TiM learns the arbitrary state transition on the diffusion trajectory, it supports arbitrary-step sampling when producing images. Given a set of timesteps  $\mathcal{T} = \{t_i\}_{i=N}^0$  where  $t_N = T, t_0 = 0$ , we obtain the next state  $x_{t_{n-1}}$  given the current state  $x_{t_n}$  based on Eq. (5), as illustrated in Algorithm 2.

## 4.2. Text-to-Image Generation

TiM establishes a new state-of-the-art in performance, efficiency, and flexibility across diverse benchmarks (Tabs. 2,

3, 5 and 6). It achieves an SOTA FID of 5.25 on MJHQ-30K while resolving the core speed-quality trade-off. On GenEval, TiM’s 1-NFE performance surpasses 8-NFE distilled models (e.g., SDXL-Turbo), while its 128-NFE quality rivals closed-source models. This unique scalability starkly contrasts with competitors like SD3.5-Large, which collapse at a few steps, and FLUX.1-Schnell, which degrades at many steps. TiM alone shows monotonic quality improvement with NFE. This efficiency is further proven on DPGBench, where 8-NFE TiM outperforms 100-NFE baselines like SDXL. Finally, TiM demonstrates superior generalization across diverse resolutions and aspect ratios, validating its fundamentally more robust design.

## 4.3. Ablation Studies

We conduct a series of ablation studies to validate our design choices, building from a standard diffusion baseline (SiT-B/4 [49] here) to our final TiM configuration. We use a 131M parameter model trained on ImageNet-256 × 256 for 80 epochs and report FID at 1, 8, and 50 NFEs, corresponding to single-step, few-step, and multi-step generation<sup>7</sup>. The results are summarized in Table 4.

**Transition Objective.** As shown in Table 4 (a vs. c), switching from the standard SiT objective to our TiM objective delivers a dramatic improvement in few-step performance, *reducing the 1-NFE FID by over 6×* (309.5 → 49.91) while maintaining strong many-step quality. This confirms that learning arbitrary transitions is critical for few-step generation. Furthermore, our proposed DDE method (c) achieves this performance while being far more scalable than JVP (b), making large-scale training practical.

**Architectural Contributions.** We next analyze the impact of our architectural innovations on top of a vanilla TiM baseline (d). Both the *Decoupled Time Embedding* (e) and *Interval-Aware Attention* (f) individually provide substantial gains across all sampling steps. Crucially, combining them (g) yields the best performance, lowering the 8-NFE FID from 33.08 to 29.21. This demonstrates that enabling the model to explicitly reason about both absolute time and the transition interval is **complementary and essential** for optimal performance.

**Training Strategy.** Building on our best architecture (g), we apply our proposed *interval weighting* scheme. This final step provides a consistent boost across the board (h), further refining the model and achieving our best FID scores of 47.46 / 24.62 / 17.10.

## 5. Conclusion and Limitations

This paper introduces the Transition Models (TiM), a novel generative model that learns to navigate the entire generative trajectory with unprecedented flexibility. The success

<sup>7</sup>1-NFE: single sampling step; 8-NFE: 4 sampling steps with CFG; 50-NFE: 25 sampling steps with CFG.

Method	NFE	Aspect Ratio						Resolution					
		2 : 5	9 : 16	2 : 3	3 : 2	16 : 9	5 : 2	1280	1536	2048	2560	3072	4096
SD3.5-Turbo [61]	8	<b>X</b>	0.53	0.60	0.58	0.30	<b>X</b>	0.61	<b>X</b>	<b>X</b>	<b>X</b>	<b>X</b>	<b>X</b>
FLUX.1-Schnell [6]	8	0.57	0.61	0.63	0.62	0.59	0.57	0.64	0.58	0.46	0.14	<b>X</b>	<b>X</b>
<b>TiM</b>	8	0.55	0.58	0.63	0.64	0.58	0.56	0.70	0.61	0.49	0.48	0.45	0.39
SD3.5-Large [20]	32	0.25	0.48	0.60	0.57	0.16	<b>X</b>	0.63	<b>X</b>	<b>X</b>	<b>X</b>	<b>X</b>	<b>X</b>
FLUX.1-Dev [5]	32	0.48	0.59	0.62	0.60	0.59	0.57	0.62	0.58	0.49	0.27	<b>X</b>	<b>X</b>
<b>TiM</b>	32	0.66	0.67	0.72	0.72	0.62	0.64	0.75	0.69	0.63	0.62	0.59	0.53

Table 6. **Benchmarking resolution generation capabilities on GenEval Benchmark.** For aspect ratio generalization, the exact resolutions are:  $\{1024 \times 2560, 1024 \times 1856, 1024 \times 1536, 1536 \times 1024, 1856 \times 1024, 2560 \times 1024\}$ . **X**: when GenEval score falls below 0.10, we interpret it as evidence that the model fails to generalize to that resolution.



Figure 3. **Qualitative Analysis between TiM and existing methods under different NFEs.** TiM delivers superior fidelity and text alignment across all NFEs. In contrast, multi-step diffusion and few-step distilled models exhibit pronounced step-quality trade-offs: SDXL, SD3.5-Large, and FLUX.1-Dev fail to generate images at low NFEs, while SDXL-Turbo, SD3.5-Turbo, and FLUX.1-Schnell produce over-saturated outputs at high NFEs.

of our compact 865M model in outperforming multi-billion parameter giants is not just a new state-of-the-art; it is a testament to a more efficient and powerful paradigm. By achieving monotonic quality improvement from one step to many, and scaling to ultra-high resolutions, TiM demonstrates that a unified model is not only possible but superior. We believe this work paves the way for a new generation of foundation models that are at once efficient, scalable, and promising in their creative potential.

**Limitations.** Although TiM delivers a significant contribution to the fundamental generative models, ensuring content safety and controllability remains an open challenge, and model fidelity can degrade in scenarios requiring fine-grained detail, such as rendering text and hands. We also observe occasional artifacts at high resolutions (e.g.,  $3072 \times 4096$ ), likely attributable to biases in the underlying autoencoder.

## References

- [1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 6
- [2] adebyollin. Megalith-huggingface. <https://huggingface.co/datasets/madebyollin/megalith-10m>. 6
- [3] Brian DO Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982. 3
- [4] Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *CVPR*, 2023. 23
- [5] black-forest labs. Flux.1-dev. <https://huggingface.co/black-forest-labs/FLUX.1-dev>, . 1, 2, 6, 7, 8
- [6] black-forest labs. Flux.1-schnell. <https://huggingface.co/black-forest-labs/FLUX.1-schnell>, . 1, 6, 7, 8
- [7] Nicholas Matthew Boffi, Michael Samuel Albergo, and Eric Vanden-Eijnden. Flow map matching with stochastic interpolants: A mathematical framework for consistency models. *Transactions on Machine Learning Research*, 2025. 1, 2
- [8] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018. 22, 23
- [9] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. Accessed: 2024-5-1. 1
- [10] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 22
- [11] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *CVPR*, 2021. 6
- [12] Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Yue Wu, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, et al. Pixart-alpha: Fast training of diffusion transformer for photorealistic text-to-image synthesis. *arXiv preprint arXiv:2310.00426*, 2023. 6
- [13] Junyu Chen, Han Cai, Junsong Chen, Enze Xie, Shang Yang, Haotian Tang, Muyang Li, Yao Lu, and Song Han. Deep compression autoencoder for efficient high-resolution diffusion models. *arXiv preprint arXiv:2410.10733*, 2024. 6, 19, 21
- [14] Junsong Chen, Chongjian Ge, Enze Xie, Yue Wu, Lewei Yao, Xiaozhe Ren, Zhongdao Wang, Ping Luo, Huchuan Lu, and Zhenguo Li. Pixart-sigma: Weak-to-strong training of diffusion transformer for 4k text-to-image generation. In *European Conference on Computer Vision*, pages 74–91. Springer, 2024. 6
- [15] Jiahui Chen, Zhiyang Xu, Xichen Pan, Yushi Hu, Can Qin, Tom Goldstein, Lifu Huang, Tianyi Zhou, Saining Xie, Silvio Savarese, et al. Blip3-o: A family of fully open unified multimodal models-architecture, training and dataset. *arXiv preprint arXiv:2505.09568*, 2025. 6
- [16] Junsong Chen, Shuchen Xue, Yuyang Zhao, Jincheng Yu, Sayak Paul, Junyu Chen, Han Cai, Song Han, and Enze Xie. Sana-sprint: One-step diffusion with continuous-time consistency distillation. *arXiv preprint arXiv:2503.09641*, 2025. 6
- [17] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023. 4
- [18] Karan Desai, Gaurav Kaul, Zubin Aysola, and Justin Johnson. Redcaps: Web-curated image-text data created by the people, for the people. *arXiv preprint arXiv:2111.11431*, 2021. 6
- [19] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *NeurIPS*, 2021. 21, 23
- [20] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. 2024. 1, 2, 6, 7, 8, 20
- [21] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. *arXiv preprint arXiv:2410.12557*, 2024. 1, 17, 19, 22
- [22] Peng Gao, Le Zhuo, Ziyi Lin, Chris Liu, Junsong Chen, Ruoyi Du, Enze Xie, Xu Luo, Longtian Qiu, Yuhang Zhang, et al. Lumina-t2x: Transforming text into any modality, resolution, and duration via flow-based large diffusion transformers. *arXiv preprint arXiv:2405.05945*, 2024. 22, 23
- [23] Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan. Masked diffusion transformer is a strong image synthesizer. *arXiv preprint arXiv:2303.14389*, 2023. 22
- [24] Shanghua Gao, Pan Zhou, Ming-Ming Cheng, and Shuicheng Yan. Mdtv2: Masked diffusion transformer is a strong image synthesizer. *arXiv preprint arXiv:2303.14389*, 2023. 22
- [25] Yu Gao, Lixue Gong, Qiushan Guo, Xiaoxia Hou, Zhichao Lai, Fanshi Li, Liang Li, Xiaochen Lian, Chao Liao, Liyang Liu, et al. Seedream 3.0 technical report. *arXiv preprint arXiv:2504.11346*, 2025. 7
- [26] Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*, 2025. 1, 2, 4, 17, 19, 22
- [27] Dhruba Ghosh, Hannaneh Hajishirzi, and Ludwig Schmidt. Geneval: An object-focused framework for evaluating text-to-image alignment. *Advances in Neural Information Processing Systems*, 36:52132–52152, 2023. 1, 2
- [28] Lixue Gong, Xiaoxia Hou, Fanshi Li, Liang Li, Xiaochen Lian, Fei Liu, Liyang Liu, Wei Liu, Wei Lu, Yichun Shi, et al. Seedream 2.0: A native chinese-english bilingual image generation foundation model. *arXiv preprint arXiv:2503.07703*, 2025. 6, 7

- [29] Ali Hatamizadeh, Jiaming Song, Guilin Liu, Jan Kautz, and Arash Vahdat. Diffit: Diffusion vision transformers for image generation. *arXiv preprint arXiv:2312.02139*, 2023. 23
- [30] 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. *NeurIPS*, 2017. 21
- [31] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020. 2, 16, 20
- [32] Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. 2023. 22, 23
- [33] jackyhate. text-to-image-2m. <https://huggingface.co/datasets/jackyhat/text-to-image-2M>. 6
- [34] Minguk Kang, Jun-Yan Zhu, Richard Zhang, Jaesik Park, Eli Shechtman, Sylvain Paris, and Taesung Park. Scaling up gans for text-to-image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10124–10134, 2023. 22
- [35] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *NeurIPS*, 2022. 1, 2, 3, 15, 16, 17, 20
- [36] Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. Analyzing and improving the training dynamics of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24174–24184, 2024. 16, 17, 20, 23
- [37] Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. *arXiv preprint arXiv:2310.02279*, 2023. 17, 19
- [38] T. Kynkänniemi, T. Karras, S. Laine, and T Lehtinen, J.and Aila. Improved precision and recall metric for assessing generative models. *NeurIPS*, 2019. 21
- [39] Daiqing Li, Aleks Kamko, Ehsan Akhgari, Ali Sabet, Lin-miao Xu, and Suhail Doshi. Playground v2. 5: Three insights towards enhancing aesthetic quality in text-to-image generation. *arXiv preprint arXiv:2402.17245*, 2024. 6
- [40] Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization. *Advances in Neural Information Processing Systems*, 37:56424–56445, 2024. 22
- [41] Zhimin Li, Jianwei Zhang, Qin Lin, Jiangfeng Xiong, Yanxin Long, Xinchi Deng, Yingfang Zhang, Xingchao Liu, Minbin Huang, Zedong Xiao, et al. Hunyuan-dit: A powerful multi-resolution diffusion transformer with fine-grained chinese understanding. *arXiv preprint arXiv:2405.08748*, 2024. 6
- [42] Shanchuan Lin, Anran Wang, and Xiao Yang. Sdxl-lightning: Progressive adversarial diffusion distillation. *arXiv preprint arXiv:2402.13929*, 2024. 1
- [43] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022. 3, 13, 16, 19, 20
- [44] Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023. 2, 16, 20
- [45] Cheng Lu and Yang Song. Simplifying, stabilizing and scaling continuous-time consistency models. *arXiv preprint arXiv:2410.11081*, 2024. 1, 2, 3, 4, 13, 15, 16, 17, 20
- [46] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in neural information processing systems*, 35:5775–5787, 2022. 1, 2
- [47] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, pages 1–22, 2025. 2, 4
- [48] Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023. 2, 6
- [49] Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. *arXiv preprint arXiv:2401.08740*, 2024. 7, 22, 23
- [50] C. Nash, J. Menick, S. Dieleman, and P. W Battaglia. Generating images with sparse representations. *arXiv preprint arXiv:2103.03841*, 2021. 21
- [51] Ziqi Pang, Tianyuan Zhang, Fujun Luan, Yunze Man, Hao Tan, Kai Zhang, William T Freeman, and Yu-Xiong Wang. Randar: Decoder-only autoregressive visual generation in random orders. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 45–55, 2025. 22
- [52] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023. 1, 5, 6, 22, 23
- [53] Yansong Peng, Kai Zhu, Yu Liu, Pingyu Wu, Hebei Li, Xiaoyan Sun, and Feng Wu. Flow-anchored consistency models. *arXiv preprint arXiv:2507.03738*, 2025. 1, 2, 4, 17
- [54] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023. 1, 6
- [55] Yuxi Ren, Xin Xia, Yanzuo Lu, Jiacheng Zhang, Jie Wu, Pan Xie, Xing Wang, and Xuefeng Xiao. Hyper-sd: Trajectory segmented consistency model for efficient image synthesis. *Advances in Neural Information Processing Systems*, 37:117340–117362, 2025. 2, 6
- [56] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 6, 21, 22
- [57] Amirmojtaba Sabour, Sanja Fidler, and Karsten Kreis. Align your flow: Scaling continuous-time flow map distillation. *arXiv preprint arXiv:2506.14603*, 2025. 1, 2, 4
- [58] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. *NeurIPS*, 2016. 21

- [59] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *ACM SIGGRAPH 2022 conference proceedings*, 2022. 23
- [60] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *ACM SIGGRAPH 2022 conference proceedings*, pages 1–10, 2022. 22
- [61] Axel Sauer, Frederic Boesel, Tim Dockhorn, Andreas Blattmann, Patrick Esser, and Robin Rombach. Fast high-resolution image synthesis with latent adversarial diffusion distillation. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–11, 2024. 1, 2, 6, 7, 8
- [62] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *European Conference on Computer Vision*, pages 87–103. Springer, 2024. 1, 2, 6
- [63] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021. 6
- [64] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. 2018.
- [65] Vasu Singla, Kaiyu Yue, Sukriti Paul, Reza Shirkavand, Mayuka Jayawardhana, Alireza Ganjdanesh, Heng Huang, Abhinav Bhatele, Gowthami Somepalli, and Tom Goldstein. From pixels to prose: A large dataset of dense image captions. *arXiv preprint arXiv:2406.10328*, 2024. 6
- [66] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 1, 4
- [67] Yang Song and Prafulla Dhariwal. Improved techniques for training consistency models. In *The Twelfth International Conference on Learning Representations*. 1, 2, 17, 22
- [68] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *NeurIPS*, 2019. 16
- [69] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 1, 2, 3, 13, 14, 15, 16, 20
- [70] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, pages 32211–32252. PMLR, 2023. 1, 2, 17, 18
- [71] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *arXiv preprint arXiv:2406.06525*, 2024. 22
- [72] Peng Sun, Yi Jiang, and Tao Lin. Unified continuous generative models. *arXiv preprint arXiv:2505.07447*, 2025. 3, 13, 15
- [73] Zhicong Tang, Jianmin Bao, Dong Chen, and Baining Guo. Diffusion models without classifier-free guidance. *arXiv preprint arXiv:2502.12154*, 2025. 20
- [74] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025. 6
- [75] Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in neural information processing systems*, 37:84839–84865, 2024. 22, 23
- [76] Fu-Yun Wang, Zhengyang Geng, and Hongsheng Li. Stable consistency tuning: Understanding and improving consistency models. *arXiv preprint arXiv:2410.18958*, 2024. 23
- [77] Fu-Yun Wang, Zhaoyang Huang, Alexander Bergman, Dazhong Shen, Peng Gao, Michael Lingelbach, Keqiang Sun, Weikang Bian, Guanglu Song, Yu Liu, et al. Phased consistency models. *Advances in Neural Information Processing Systems*, 37:83951–84009, 2025. 2, 17, 19
- [78] Shuai Wang, Zexian Li, Tianhui Song, Xubin Li, Tiezheng Ge, Bo Zheng, and Limin Wang. Exploring den-like architecture for fast image generation with arbitrary resolution. *Advances in Neural Information Processing Systems*, 37:87959–87977, 2024. 22, 23
- [79] Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiyi Yu, et al. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*, 2024. 6
- [80] Zidong Wang, Lei Bai, Xiangyu Yue, Wanli Ouyang, and Yiyuan Zhang. Native-resolution image synthesis. *arXiv preprint arXiv:2506.03131*, 2025. 7
- [81] Enze Xie, Junsong Chen, Yuyang Zhao, Jincheng Yu, Ligeng Zhu, Chengyue Wu, Yujun Lin, Zhekai Zhang, Muyang Li, Junyu Chen, et al. Sana 1.5: Efficient scaling of training-time and inference-time compute in linear diffusion transformer. *arXiv preprint arXiv:2501.18427*, 2025. 6
- [82] Wanghan Xu, Xiaoyu Yue, Zidong Wang, Yao Teng, Wenlong Zhang, Xihui Liu, Luping Zhou, Wanli Ouyang, and Lei Bai. Exploring representation-aligned latent space for better generation. *arXiv preprint arXiv:2502.00359*, 2025. 22
- [83] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazhen Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024. 1
- [84] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6613–6623, 2024. 1, 2
- [85] Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and Bill Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in Neural Information Processing Systems*, 37:47455–47487, 2025. 2
- [86] Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Vervari, Kihyuk Sohn, David Minnen, Yong Cheng, Vighnesh

- Birodkar, Agrim Gupta, Xiuye Gu, et al. Language model beats diffusion–tokenizer is key to visual generation. *arXiv preprint arXiv:2310.05737*, 2023. 22, 23
- [87] Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. *arXiv preprint arXiv:2410.06940*, 2024. 22, 23
- [88] Xiaoyu Yue, Zidong Wang, Zeyu Lu, Shuyang Sun, Meng Wei, Wanli Ouyang, Lei Bai, and Luping Zhou. Diffusion models need visual priors for image generation. *arXiv preprint arXiv:2410.08531*, 2024. 22
- [89] Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023. 4, 20
- [90] Hongkai Zheng, Weili Nie, Arash Vahdat, and Anima Anandkumar. Fast training of diffusion models with masked transformers. 2023. 23
- [91] Kaiwen Zheng, Cheng Lu, Jianfei Chen, and Jun Zhu. Dpm-solver-v3: Improved diffusion ode solver with empirical model statistics. *Advances in Neural Information Processing Systems*, 36:55502–55542, 2023. 2, 4
- [92] Linqi Zhou, Stefano Ermon, and Jiaming Song. Inductive moment matching. *arXiv preprint arXiv:2503.07565*, 2025. 22
- [93] 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 *Forty-first International Conference on Machine Learning*, 2024. 1

## Appendix

We include additional derivations, experimental details, and results in the appendix.

- In Appx. A, we provide a detailed formula derivation of the State Transition Identity, training, and sampling algorithms.
- In Appx. B, we discuss TiM’s relationships with existing methods, including diffusion models, consistency models, and other training approaches.
- In Appx. C, we provide the implementation details of text-to-image generation, including native-resolution training, resolution-dependent timestep shifting, model-guidance training, and from-scratch training.
- In Appx. D, we provide additional ablation results and results on class-guided image generation.
- In Appx. E, we provide more qualitative results of TiM.

## A. Transition Model Framework

In this section, we first provide the derivation of the TiM identity equation Eq. (8). Then we provide the training and sampling algorithms. Finally, we provide a systematic analysis of the connections with other existing methods.

### A.1. TiM Identity Equation Derivation

We demonstrate the derivation from Eq. (7) to the TiM identity equation Eq. (8). We start from the detailed expansion of each term of Eq. (7). Firstly, we have:

$$\mathbf{x}_t = \alpha_t \mathbf{x} + \sigma_t \boldsymbol{\varepsilon}, \quad (16)$$

$$\frac{d\mathbf{x}_t}{dt} = \frac{d\alpha_t}{dt} \mathbf{x} + \frac{d\sigma_t}{dt} \boldsymbol{\varepsilon}, \quad (17)$$

where  $\frac{d\mathbf{x}_t}{dt}$  is the PF-ODE of diffusion and Eq. (17) has already been proved in previous works [43, 45, 69, 72]. For  $A_{t,r} = \frac{\alpha_r \hat{\sigma}_t - \sigma_r \hat{\alpha}_t}{\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t}$ ,  $B_{t,r} = \frac{\sigma_r \alpha_t - \alpha_r \sigma_t}{\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t}$ , we have:

$$\frac{dA_{t,r}}{dt} = \frac{dA_{t,r}}{d\alpha_t} \cdot \frac{d\alpha_t}{dt} + \frac{dA_{t,r}}{d\sigma_t} \cdot \frac{d\sigma_t}{dt} + \frac{dA_{t,r}}{d\hat{\alpha}_t} \cdot \frac{d\hat{\alpha}_t}{dt} + \frac{dA_{t,r}}{d\hat{\sigma}_t} \cdot \frac{d\hat{\sigma}_t}{dt} \quad (18)$$

$$\frac{dB_{t,r}}{dt} = \frac{dB_{t,r}}{d\alpha_t} \cdot \frac{d\alpha_t}{dt} + \frac{dB_{t,r}}{d\sigma_t} \cdot \frac{d\sigma_t}{dt} + \frac{dB_{t,r}}{d\hat{\alpha}_t} \cdot \frac{d\hat{\alpha}_t}{dt} + \frac{dB_{t,r}}{d\hat{\sigma}_t} \cdot \frac{d\hat{\sigma}_t}{dt} \quad (19)$$

We use  $C_{t,r} = \hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t$  for simplicity, which is the denominator of  $A_{t,r}$  and  $B_{t,r}$ . For Eq. (18) and Eq. (19), each term is calculated as:

$$\left\{ \begin{array}{l} \frac{dA_{t,r}}{d\alpha_t} = \frac{(\sigma_r \hat{\alpha}_t - \alpha_r \hat{\sigma}_t) \hat{\sigma}_t}{(\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t)^2} = -\hat{\sigma}_t \frac{A_{t,r}}{C_{t,r}} \\ \frac{dA_{t,r}}{d\sigma_t} = \frac{(\alpha_r \hat{\sigma}_t - \sigma_r \hat{\alpha}_t) \hat{\alpha}_t}{(\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t)^2} = \hat{\alpha}_t \frac{A_{t,r}}{C_{t,r}} \\ \frac{dA_{t,r}}{d\hat{\alpha}_t} = \frac{(\alpha_r \sigma_t - \sigma_r \alpha_t) \sigma_t}{(\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t)^2} = -\hat{\sigma}_t \frac{B_{t,r}}{C_{t,r}} \\ \frac{dA_{t,r}}{d\hat{\sigma}_t} = \frac{(\alpha_t \sigma_r - \alpha_r \sigma_t) \alpha_t}{(\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t)^2} = \hat{\alpha}_t \frac{B_{t,r}}{C_{t,r}} \end{array} \right. ; \quad \left\{ \begin{array}{l} \frac{dB_{t,r}}{d\alpha_t} = \frac{(\alpha_r \hat{\sigma}_t - \sigma_r \hat{\alpha}_t) \sigma_t}{(\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t)^2} = \sigma_t \frac{A_{t,r}}{C_{t,r}} \\ \frac{dB_{t,r}}{d\sigma_t} = \frac{(\sigma_r \hat{\alpha}_t - \alpha_r \hat{\sigma}_t) \alpha_t}{(\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t)^2} = -\alpha_t \frac{A_{t,r}}{C_{t,r}} \\ \frac{dB_{t,r}}{d\hat{\alpha}_t} = \frac{(\sigma_r \alpha_t - \alpha_r \sigma_t) \sigma_t}{(\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t)^2} = \sigma_t \frac{B_{t,r}}{C_{t,r}} \\ \frac{dB_{t,r}}{d\hat{\sigma}_t} = \frac{(\alpha_r \sigma_t - \sigma_r \alpha_t) \alpha_t}{(\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t)^2} = -\alpha_t \frac{B_{t,r}}{C_{t,r}} \end{array} \right. . \quad (20)$$

Substituting Eq. (20) into Eq. (18) and Eq. (19), we have:

$$\frac{dA_{t,r}}{dt} = -\hat{\sigma}_t \frac{A_{t,r}}{C_{t,r}} \cdot \frac{d\alpha_t}{dt} + \hat{\alpha}_t \frac{A_{t,r}}{C_{t,r}} \cdot \frac{d\sigma_t}{dt} - \hat{\sigma}_t \frac{B_{t,r}}{C_{t,r}} \cdot \frac{d\hat{\alpha}_t}{dt} + \hat{\alpha}_t \frac{B_{t,r}}{C_{t,r}} \cdot \frac{d\hat{\sigma}_t}{dt}, \quad (21)$$

$$\frac{dB_{t,r}}{dt} = \sigma_t \frac{A_{t,r}}{C_{t,r}} \cdot \frac{d\alpha_t}{dt} - \alpha_t \frac{A_{t,r}}{C_{t,r}} \cdot \frac{d\sigma_t}{dt} + \sigma_t \frac{B_{t,r}}{C_{t,r}} \cdot \frac{d\hat{\alpha}_t}{dt} - \alpha_t \frac{B_{t,r}}{C_{t,r}} \cdot \frac{d\hat{\sigma}_t}{dt}. \quad (22)$$

There exists some symmetry between the above two equations, which is the key to our TiM identity. Combining Eqs. (16), (17) and (21), we have:

$$\mathbf{x}_t \frac{dA_{t,r}}{dt} + A_{t,r} \frac{d\mathbf{x}_t}{dt} = (A_{t,r} \frac{d\alpha_t}{dt} + \alpha_t \frac{dA_{t,r}}{dt}) \mathbf{x} + (A_{t,r} \frac{d\sigma_t}{dt} + \sigma_t \frac{dA_{t,r}}{dt}) \boldsymbol{\varepsilon}. \quad (23)$$

The coefficient of  $x$  in the above equation can be decomposed as:

$$\begin{aligned}
& A_{t,r} \frac{d\alpha_t}{dt} + \alpha_t \frac{dA_{t,r}}{dt} \\
&= (A_{t,r} + \alpha_t \frac{dA_{t,r}}{d\alpha_t}) \frac{d\alpha_t}{dt} + \alpha_t \frac{dA_{t,r}}{d\sigma_t} \cdot \frac{d\sigma_t}{dt} + \alpha_t \frac{dA_{t,r}}{d\hat{\alpha}_t} \cdot \frac{d\hat{\alpha}_t}{dt} + \alpha_t \frac{dA_{t,r}}{d\hat{\sigma}_t} \cdot \frac{d\hat{\sigma}_t}{dt} \\
&= -\hat{\alpha}_t \sigma_t \frac{A_{t,r}}{C_{t,r}} \cdot \frac{d\alpha_t}{dt} + \hat{\alpha}_t \alpha_t \frac{A_{t,r}}{C_{t,r}} \cdot \frac{d\sigma_t}{dt} - \hat{\sigma}_t \alpha_t \frac{B_{t,r}}{C_{t,r}} \cdot \frac{d\hat{\alpha}_t}{dt} + \hat{\alpha}_t \alpha_t \frac{B_{t,r}}{C_{t,r}} \cdot \frac{d\hat{\sigma}_t}{dt} \\
&= -\hat{\alpha}_t \frac{dB_{t,r}}{dt} + (\hat{\alpha}_t \sigma_t - \hat{\sigma}_t \alpha_t) \frac{B_{t,r}}{C_{t,r}} \cdot \frac{\hat{\alpha}_t}{dt} \\
&= -\hat{\alpha}_t \frac{dB_{t,r}}{dt} - B_{t,r} \frac{d\hat{\alpha}_t}{dt}.
\end{aligned} \tag{24}$$

Similarly, the coefficient of  $\varepsilon$  in the Eq. (23) can be decomposed as:

$$\begin{aligned}
& A_{t,r} \frac{d\sigma_t}{dt} + \sigma_t \frac{dA_{t,r}}{dt} \\
&= \sigma_t \frac{dA_{t,r}}{d\alpha_t} \cdot \frac{d\alpha_t}{dt} + (A_{t,r} + \sigma_t \frac{dA_{t,r}}{d\sigma_t}) \frac{dA_{t,r}}{d\sigma_t} \cdot \frac{d\sigma_t}{dt} + \sigma_t \frac{dA_{t,r}}{d\hat{\alpha}_t} \cdot \frac{d\hat{\alpha}_t}{dt} + \sigma_t \frac{dA_{t,r}}{d\hat{\sigma}_t} \cdot \frac{d\hat{\sigma}_t}{dt} \\
&= -\hat{\sigma}_t \sigma_t \frac{A_{t,r}}{C_{t,r}} \cdot \frac{d\hat{\sigma}_t}{dt} + \hat{\sigma}_t \alpha_t \frac{A_{t,r}}{C_{t,r}} \cdot \frac{d\hat{\sigma}_t}{dt} - \hat{\sigma}_t \sigma_t \frac{B_{t,r}}{C_{t,r}} \cdot \frac{d\hat{\sigma}_t}{dt} + \hat{\alpha}_t \sigma_t \frac{B_{t,r}}{C_{t,r}} \cdot \frac{d\hat{\sigma}_t}{dt} \\
&= -\hat{\sigma}_t \frac{dB_{t,r}}{dt} + (\hat{\alpha}_t \sigma_t - \hat{\sigma}_t \alpha_t) \frac{B_{t,r}}{C_{t,r}} \cdot \frac{d\hat{\sigma}_t}{dt} \\
&= -\hat{\sigma}_t \frac{dB_{t,r}}{dt} - B_{t,r} \frac{d\hat{\sigma}_t}{dt}.
\end{aligned} \tag{25}$$

Substituting Eqs. (23) to (25) into Eq. (7), we have:

$$\begin{aligned}
& \mathbf{x}_t \frac{dA_{t,r}}{dt} + A_{t,r} \frac{d\mathbf{x}_t}{dt} + \mathbf{f}_{\theta,t,r} \frac{dB_{t,r}}{dt} + B_{t,r} \frac{d\mathbf{f}_{\theta,t,r}}{dt} = 0. \\
& \Rightarrow (-\hat{\alpha}_t \frac{dB_{t,r}}{dt} - B_{t,r} \frac{d\hat{\alpha}_t}{dt}) \mathbf{x} + (-\hat{\sigma}_t \frac{dB_{t,r}}{dt} - B_{t,r} \frac{d\hat{\sigma}_t}{dt}) \varepsilon + \mathbf{f}_{\theta,t,r} \frac{dB_{t,r}}{dt} + B_{t,r} \frac{d\mathbf{f}_{\theta,t,r}}{dt} = 0. \\
& \Rightarrow (\hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \varepsilon - \mathbf{f}_{\theta,t,r}) \frac{dB_{t,r}}{dt} + B_{t,r} \left( \frac{d\hat{\alpha}_t}{dt} + \frac{d\hat{\sigma}_t}{dt} - \frac{d\mathbf{f}_{\theta,t,r}}{dt} \right) = 0 \\
& \Rightarrow (\hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \varepsilon - \mathbf{f}_{\theta,t,r}) \frac{dB_{t,r}}{dt} + B_{t,r} \frac{d(\hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \varepsilon - \mathbf{f}_{\theta,t,r})}{dt} = 0 \\
& \Rightarrow \frac{d(B_{t,r} \cdot (\hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \varepsilon - \mathbf{f}_{\theta,t,r}))}{dt} = 0.
\end{aligned} \tag{26}$$

This is the TiM identity equation in Eq. (8), the proof is completed.

## A.2. TiM Training Algorithm

We provide the detailed training algorithm of TiM in Algorithm 1. It is noteworthy that the TiM models are entirely trained from scratch.

## A.3. TiM Sampling Algorithms

We provide the TiM sampling algorithm in Algorithm 2. For multi-step sampling, we can further incorporate stochasticity into the sampling process for improved diversity. In multi-step scenarios, the TiM sampling is similar to the diffusion sampling process, but with a new condition for the next step. Therefore, we can construct a stochastic sampling from the SDE (stochastic differential equation) diffusion process. Given  $\mathbf{x}_t = \alpha_t + \sigma_t \varepsilon$ , Song et al. [69] has shown that the SDE forward and reverse are:

$$\begin{aligned}
& \text{forward : } d\mathbf{x}_t = \mathbf{f}(\mathbf{x}_t, t) + \mathbf{g}(t) d\mathbf{w}, \\
& \text{reverse : } d\mathbf{x}_t = [\mathbf{f}(\mathbf{x}_t, t) - \frac{1}{2} \mathbf{g}(t)^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)] dt + \mathbf{g}(t) d\mathbf{w}.
\end{aligned} \tag{27}$$

---

**Algorithm 1** Training Algorithm of Diffusion Transition Models (TiM).

---

**Input:** dataset  $\mathcal{D}$  with standard deviation  $\sigma_d$ , model  $f_\theta$ , diffusion parameterization  $\{\alpha_t, \sigma_t, \hat{\alpha}_t, \hat{\sigma}_t\}$ , weighting  $w_{\Delta t}$ , learning rate  $\eta$ , time distribution  $\mathcal{T}$ , constant  $\epsilon$ , constant  $c$ .

**Init:** Iters  $\leftarrow 0$

**repeat**

- $\mathbf{x}_d \sim \mathcal{D}, \mathbf{x} = c_{\text{data}}(\mathbf{x}_d), \boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), r < t \sim \mathcal{T}, \mathbf{x}_t \leftarrow \alpha_t \mathbf{x} + \sigma_t \boldsymbol{\varepsilon}$
- $B_{t,r} \leftarrow (\sigma_r \alpha_t - \alpha_r \sigma_t) / (\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t)$
- $\frac{df_{\theta^-}}{dt} = \frac{1}{2\epsilon} (f_{\theta^-}(\mathbf{x}_{t+\epsilon}, t + \epsilon, r) - f_{\theta^-}(\mathbf{x}_{t-\epsilon}, t - \epsilon, r))$  ▷ DDE Calculation
- $\hat{f} \leftarrow \hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \boldsymbol{\varepsilon} + (\frac{d\hat{\alpha}_t}{dt} \cdot \mathbf{x} + \frac{d\hat{\sigma}_t}{dt} \cdot \boldsymbol{\varepsilon} - \frac{df_{\theta^-}}{dt}) \cdot B_{t,r} / \frac{dB_{t,r}}{dt}$  ▷ TiM Target
- $\mathcal{L}(\theta) \leftarrow \|f_\theta - \hat{f}\|_2^2 + L_{\cos}(f_\theta, \hat{f})$
- $\mathcal{L}(\theta) \leftarrow w_{\Delta t} \cdot \mathcal{L}(\theta) / (\|\mathcal{L}(\theta)\| + c)$
- $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\theta)$
- Iter  $\leftarrow$  Iters + 1

**until** convergence

---

**Algorithm 2** Piecewise Sampling Algorithm of Diffusion Transition Models (TiM).

---

**Input:** sampling step  $N$ , maximum timestep  $T_{\max}$ , model  $f_\theta$ , diffusion parameterization  $\{\alpha_t, \sigma_t, \hat{\alpha}_t, \hat{\sigma}_t\}$ , stochasticity ratio  $\rho$ .

**Init:** data  $\mathbf{x}_N \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ , timesteps  $\mathcal{T} = \{t_i\}_{i=N}^0$  where  $t_N = T_{\max}, t_0 = 0$

**for**  $i = N$  to 1 **do**

- $\mathbf{x}_{t_{i+1}} = \frac{\alpha_{t_{i+1}} \hat{\sigma}_{t_i} - \sigma_{t_{i+1}} \hat{\alpha}_{t_i}}{\hat{\sigma}_{t_i} \alpha_{t_i} - \hat{\alpha}_{t_i} \sigma_{t_i}} \mathbf{x}_{t_i} + \frac{\sigma_{t_{i+1}} \alpha_{t_i} - \alpha_{t_{i+1}} \sigma_{t_i}}{\hat{\sigma}_{t_i} \alpha_{t_i} - \hat{\alpha}_{t_i} \sigma_{t_i}} f(\mathbf{x}_{t_i}, t_i, t_{i+1})$
- if**  $\rho > 0$  **then**:

  - $\hat{\boldsymbol{\varepsilon}} \leftarrow \frac{\alpha_{t_i}}{\hat{\sigma}_{t_i} \alpha_{t_i} \hat{\alpha}_{t_i} \sigma_{t_i}} f_\theta(\mathbf{x}_{t_i}, t_i, t_0) - \frac{\hat{\alpha}_{t_i}}{\hat{\sigma}_{t_i} \alpha_{t_i} - \hat{\alpha}_{t_i} \sigma_{t_i}} \mathbf{x}_{t_i}$
  - $\boldsymbol{\varepsilon}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  - $dt = t_i - t_{i+1}$
  - $\mathbf{x}_{t_{i+1}} \leftarrow \mathbf{x}_{t_{i+1}} - \rho(\alpha_{t_i} \sigma'_{t_i} - \alpha'_{t_i} \sigma_{t_i}) \hat{\boldsymbol{\varepsilon}} dt - \sqrt{2\rho(\alpha_{t_i} \sigma'_{t_i} - \alpha'_{t_i} \sigma_{t_i})} \boldsymbol{\varepsilon}_i \sqrt{dt}$

- end if**
- $\mathbf{x}_i = \mathbf{x}_{t_{i+1}}$

**end for**

---

Previous works[35, 45, 69, 72] has provided the explicit form of  $f(\mathbf{x}_t, t)$ ,  $g(t)$  and  $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$ :

$$f(\mathbf{x}_t, t) = \frac{\dot{\alpha}_t}{\alpha_t} \mathbf{x}_t, \quad g(t) = 2\sigma_t \dot{\alpha}_t - 2\frac{\dot{\alpha}_t}{\alpha_t} \sigma_t^2, \quad \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) = -\frac{\boldsymbol{\varepsilon}}{\sigma_t}, \quad (28)$$

where  $\dot{\alpha}_t$  and  $\dot{\sigma}_t$  represent the derivation of  $\alpha_t$  and  $\sigma_t$  respectively. For PF-ODE, it is defined as:

$$\mathbf{v}_t = \frac{d\mathbf{x}_t}{dt} = f(\mathbf{x}_t, t) - \frac{1}{2} g(t)^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) = \dot{\alpha}_t \mathbf{x} + \dot{\sigma}_t \boldsymbol{\varepsilon}. \quad (29)$$

For reverse-SDE, it is defined as:

$$\begin{aligned} d\mathbf{x}_t &= [f(\mathbf{x}_t, t) - g(t)^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)] dt + g(t) d\mathbf{w} \\ &= \underbrace{[f(\mathbf{x}_t, t) - \frac{1}{2} g(t)^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)] dt}_{\text{PF-ODE Term}} - \underbrace{\frac{1}{2} g(t)^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) dt + g(t) d\mathbf{w}}_{\text{Stochastic Term}} \\ &= \mathbf{v}_t dt + [\dot{\alpha}_t - \frac{\dot{\alpha}_t}{\alpha_t} \sigma_t] \boldsymbol{\varepsilon} dt + \sqrt{2\sigma_t \dot{\sigma}_t - 2\frac{\dot{\alpha}_t}{\alpha_t} \sigma_t^2} d\mathbf{w}. \end{aligned} \quad (30)$$

In the TiM sampling, we can take the stochastic term in the above equation to enhance diversity. To balance the stochasticity and stability, we incorporate a scaling factor  $s(t) = \rho \alpha_t$ , leading to a scaled  $\tilde{g} = \rho \alpha_t g(t) = 2\rho(\alpha_t \sigma_t \dot{\sigma}_t - 2\dot{\alpha}_t \sigma_t^2)$ . Therefore, the stochastic term is:  $\rho[\alpha_t \dot{\sigma}_t - \dot{\alpha}_t \sigma_t] \boldsymbol{\varepsilon} dt + \sqrt{2\rho(\alpha_t \sigma_t \dot{\sigma}_t - 2\dot{\alpha}_t \sigma_t^2)} d\mathbf{w}$ .

Transport	Diffusion Parameterization					Transition Parameterization			
	$c_{\text{noise}}(t) =$	$\alpha_t =$	$\sigma_t =$	$\hat{\alpha}_t =$	$\hat{\sigma}_t =$	$\frac{d\hat{\alpha}_t}{dt} =$	$\frac{d\hat{\sigma}_t}{dt} =$	$B_{t,r} =$	$\frac{dB_{t,r}}{dt} =$
<b>OT-FM [43, 44]</b>	$t$	$1-t$	$t$	$-1$	$1$	$0$	$0$	$r-t$	$-1$
<b>TrigFlow [45]</b>	$t$	$\cos(t)$	$\sin(t)$	$-\sin(t)$	$\cos(t)$	$-\cos(t)$	$-\sin(t)$	$\sin(r-t)$	$-\cos(r-t)$
<b>EDM [35, 36]</b>	$\frac{1}{4} \ln(t)$	$\frac{1}{t^2 + \sigma_d^2}$	$\frac{t}{\sqrt{t^2 + \sigma_d^2}}$	$\frac{t}{\sigma_d \sqrt{t^2 + \sigma_d^2}}$	$-\frac{\sigma_d}{\sqrt{t^2 + \sigma_d^2}}$	Eq. (35)	Eq. (36)	Eq. (37)	Eq. (38)
<b>VP-SDE [31, 69]</b>	$(T-1)t$	$\frac{1}{\beta_t^2 + 1}$	$\frac{\beta_t}{\sqrt{\beta_t^2 + 1}}$	$0$	$1$	$0$	$0$	$\frac{\beta_r - \beta_t}{\sqrt{\beta_r^2 + 1}}$	$\frac{-1}{\sqrt{\beta_r^2 + 1}} \cdot \frac{d\beta_t}{dt}$
<b>VE-SDE [68, 69]</b>	$\ln(\frac{1}{2}t)$	$1$	$t$	$0$	$-1$	$0$	$0$	$t-r$	$1$

Table 7. **Transition parameterization for different diffusion transports.** For VP-SDE,  $T$  is set to 1000, and  $\beta_t = \sqrt{e^{\frac{1}{2}\beta_d t^2 + \beta_{\min} t} - 1}$ , where  $\beta_d = 19.9$  and  $\beta_{\min} = 0.1$  by default. We provide the details of timestep sampling in Tab. 8. Song et al. [69] has shown that VP-SDE is equivalent to DDPM [31] while VE-SDE is equivalent to score matching [68], so we adopt their notations for uniformity. For EDM, its TiM parameterization is too complex; we provide them in Eqs. (35) to (38).

## B. Connections with Existing Methods

In this section, we highlight the connection between TiM and other existing methods. We first demonstrate the properties of TiM compared with diffusion models. Then we demonstrate the connections of TiM with other training strategies.

Transport	Noise Level	Timestep	Time Range	Time Scaling
<b>OT-FM</b>	$\ln(\sigma) \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$	$t = \frac{\sigma}{1+\sigma}$	$t \in [0, 1]$	$c_{\text{noise}}(t) = t$
<b>Trigflow</b>	$\ln(\sigma) \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$	$t = \arctan(\frac{\sigma}{\sigma_d})$	$t \in [0, \frac{\pi}{2}]$	$c_{\text{noise}}(t) = t$
<b>EDM</b>	$\ln(\sigma) \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$	$t = \sigma$	$t \in [0, +\infty)$	$c_{\text{noise}}(t) = \frac{1}{4} \ln(t)$
<b>VP</b>	$\sigma \sim \mathcal{U}(\epsilon_t, 1)$	$t = \sigma$	$t \in [\epsilon_t, 1]$	$c_{\text{noise}}(t) = (T-1)t$
<b>VE</b>	$\sigma \sim \mathcal{U}(\epsilon_t, 1)$	$t = \sigma_{\max} \left( \frac{\sigma_{\min}^2}{\sigma_{\max}^2} \right)^{\sigma}$	$t \in [\sigma_{\min}, \sigma_{\max}]$	$c_{\text{noise}}(t) = \ln(\frac{1}{2}t)$

Table 8. Time distribution of diffusion diffusion transports.

### B.1. Connections with Diffusion Models

**TiM generalizes the standard diffusion models.** As a complement to Tab. 7, we elucidate the time distribution of different diffusion transports in Tab. 8. Our TiMs share these parameters with diffusion models, but learn a different objective. We show that the TiM training objective Eq. (13) generalizes the standard diffusion objective Eq. (3). Specifically, the TiM identity equation reduces to the diffusion identity equation in the limit as  $t \rightarrow r$ . Recall that  $B_{t,r} = \frac{\sigma_r \alpha_t - \alpha_r \sigma_t}{\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t}$ , the training target of TiM when  $t \rightarrow r$  becomes:

$$\begin{aligned}
\lim_{t \rightarrow r} \hat{f} &= \lim_{t \rightarrow r} \left( \hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \boldsymbol{\varepsilon} + \frac{B_{t,r}}{\frac{dB_{t,r}}{dt}} \left( \frac{d\hat{\alpha}_t}{dt} \cdot \mathbf{x} + \frac{d\hat{\sigma}_t}{dt} \cdot \boldsymbol{\varepsilon} - \frac{df_{\theta^{-},t,r}}{dt} \right) \right) \\
&= \hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \boldsymbol{\varepsilon} + \lim_{t \rightarrow r} \left( \frac{B_{t,r}}{\frac{dB_{t,r}}{dt}} \left( \frac{d\hat{\alpha}_t}{dt} \cdot \mathbf{x} + \frac{d\hat{\sigma}_t}{dt} \cdot \boldsymbol{\varepsilon} - \frac{df_{\theta^{-},t,r}}{dt} \right) \right) \\
&= \hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \boldsymbol{\varepsilon} + \lim_{t \rightarrow r} \left( \frac{\frac{\sigma_r \alpha_t - \alpha_r \sigma_t}{\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t}}{\frac{dB_{t,r}}{dt}} \left( \frac{d\hat{\alpha}_t}{dt} \cdot \mathbf{x} + \frac{d\hat{\sigma}_t}{dt} \cdot \boldsymbol{\varepsilon} - \frac{df_{\theta^{-},t,r}}{dt} \right) \right) \\
&= \hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \boldsymbol{\varepsilon} + \lim_{t \rightarrow r} \left( \frac{0}{\frac{dB_{t,r}}{dt}} \left( \frac{d\hat{\alpha}_t}{dt} \cdot \mathbf{x} + \frac{d\hat{\sigma}_t}{dt} \cdot \boldsymbol{\varepsilon} - \frac{df_{\theta^{-},t,r}}{dt} \right) \right) \\
&= \hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \boldsymbol{\varepsilon}.
\end{aligned} \tag{31}$$

The above target is the diffusion target. This target lacks the modeling of state transitions from state to state, thus limiting the arbitrary-step generation capabilities of diffusion models.

**EDM parametrization.** EDM [35, 36] parameterizes the diffusion model as:

$$\mathbf{D}_\theta(\mathbf{x} + t\boldsymbol{\varepsilon}, t) = \frac{\sigma_d^2}{t^2 + \sigma_d^2}(\mathbf{x} + t\boldsymbol{\varepsilon}) + \frac{t \cdot \sigma_d}{\sqrt{t^2 + \sigma_d^2}} \mathbf{F}_\theta \left( \frac{\mathbf{x} + t\boldsymbol{\varepsilon}}{\sqrt{t^2 + \sigma_d^2}}, \frac{1}{4} \ln(t) \right). \quad (32)$$

It adopts the  $\mathbf{x}$ -prediction in its training and use time weighting  $w(t) = \frac{t^2 + \sigma_d^2}{t^2 \sigma_d^2}$ , leading to training objective as:

$$\begin{aligned} \mathcal{L}(\theta) &= \frac{t^2 + \sigma_d^2}{t^2 \sigma_d^2} \|\mathbf{D}_\theta(\mathbf{x} + t\boldsymbol{\varepsilon}, t) - \mathbf{x}\|_2^2 \\ &= \frac{t^2 + \sigma_d^2}{t^2 \sigma_d^2} \left\| \frac{\sigma_d^2}{t^2 + \sigma_d^2}(\mathbf{x} + t\boldsymbol{\varepsilon}) + \frac{t \cdot \sigma_d}{\sqrt{t^2 + \sigma_d^2}} \mathbf{F}_\theta \left( \frac{\mathbf{x} + t\boldsymbol{\varepsilon}}{\sqrt{t^2 + \sigma_d^2}}, \frac{1}{4} \ln(t) \right) - \mathbf{x} \right\|_2^2 \\ &= \left\| \mathbf{F}_\theta \left( \frac{\mathbf{x} + t\boldsymbol{\varepsilon}}{\sqrt{t^2 + \sigma_d^2}}, \frac{1}{4} \ln(t) \right) - \left( \frac{t}{\sigma_d \sqrt{t^2 + \sigma_d^2}} \mathbf{x} - \frac{\sigma_d}{\sqrt{t^2 + \sigma_d^2}} \boldsymbol{\varepsilon} \right) \right\|_2^2 \end{aligned} \quad (33)$$

Therefore, let  $c_{\text{noise}}(t) = \frac{1}{4} \ln(t)$ , the original EDM parameterization can be unified into our parameterization with the following coefficients:

$$\alpha_t = \frac{1}{\sqrt{t^2 + \sigma_d^2}}, \quad \sigma_t = \frac{t}{\sqrt{t^2 + \sigma_d^2}}, \quad \hat{\alpha}_t = \frac{t}{\sigma_d \sqrt{t^2 + \sigma_d^2}}, \quad \hat{\sigma}_t = -\frac{\sigma_d}{\sqrt{t^2 + \sigma_d^2}} \quad (34)$$

Therefore, the TiM parameterization is defined as:

$$\frac{d\hat{\alpha}_t}{dt} = -\frac{t^2}{\sigma_d(t^2 + \sigma_d^2)^{\frac{3}{2}}} + \frac{1}{\sigma_d \sqrt{t^2 + \sigma_d^2}}, \quad (35)$$

$$\frac{d\hat{\sigma}_t}{dt} = \frac{t\sigma_d}{(t^2 + \sigma_d^2)^{\frac{3}{2}}}, \quad (36)$$

$$B_{t,r} = \frac{(t-r)\sigma_d^2 \sqrt{t^2 + \sigma_d^2}}{(t^2 + \sigma_d^3) \sqrt{r^2 + \sigma_d^2}}, \quad (37)$$

$$\frac{dB_{t,r}}{dt} = \sigma_d^2 \frac{t(t-r)(t^2 + \sigma_d^3) - 2t(t-r)(t^2 + \sigma_d^2) + (t^2 + \sigma_d^2)(t^2 + \sigma_d^3)}{(t^2 + \sigma_d^3)^2 \sqrt{t^2 + \sigma_d^2} \sqrt{r^2 + \sigma_d^2}}. \quad (38)$$

## B.2. Connections to Other training Methods

In this section, we discuss the connections of TiM with other training strategies, including continuous-time consistency models [45, 70], consistency trajectory models [37], phased consistency models [77], Shortcut models [21], and MeanFlow models [26, 53].

**Continuous-time consistency models.** The TiM objective Eq. (13) generalizes the continuous-time consistency models. Specifically, the CTM objective reduces to the continuous-time CM objective when  $r = 0$ . For TiM, let  $r = 0$  and  $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2^2$ , the training objective becomes:

$$\begin{aligned} &\nabla_\theta \mathbb{E}_{\mathbf{x}, \boldsymbol{\varepsilon}, t} \left[ \left\| \mathbf{f}_\theta(\mathbf{x}_t, t, 0) - (\hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \boldsymbol{\varepsilon} + \frac{B_{t,0}}{\frac{dB_{t,0}}{dt}} \left( \frac{d\hat{\alpha}_t}{dt} \cdot \mathbf{x} + \frac{d\hat{\sigma}_t}{dt} \cdot \boldsymbol{\varepsilon} - \frac{d\mathbf{f}_{\theta^-, t, 0}}{dt} \right)) \right\|_2^2 \right] \\ &= \mathbb{E}_{\mathbf{x}, \boldsymbol{\varepsilon}, t} \left[ [\nabla_\theta \mathbf{f}_{\theta, t, 0}]^T \left( \mathbf{f}_{\theta^-, t, 0} - \hat{\alpha}_t \mathbf{x} - \hat{\sigma}_t \boldsymbol{\varepsilon} + \frac{B_{t,0}}{\frac{dB_{t,0}}{dt}} \left( \frac{d\mathbf{f}_{\theta^-, t, 0}}{dt} - \frac{d\hat{\alpha}_t}{dt} \cdot \mathbf{x} - \frac{d\hat{\sigma}_t}{dt} \cdot \boldsymbol{\varepsilon} \right) \right) \right]. \end{aligned} \quad (39)$$

Continuous-time consistency models [45, 67, 70] are trained to map the noisy input  $\mathbf{x}_t$  directly to the clean data  $\mathbf{x}$  in one or a few steps. Given model  $F_\theta$ , the consistency models are formulated as:

$$\mathbf{D}_\theta(\mathbf{x}_t, t) = c_{\text{skip}}(t) \mathbf{x}_t + c_{\text{out}}(t) \mathbf{F}_\theta(c_{\text{in}}(t) \mathbf{x}_t, c_{\text{noise}}(t)). \quad (40)$$

Using the parameters  $\alpha_t$ ,  $\sigma_t$ ,  $\hat{\alpha}_t$ , and  $\hat{\sigma}_t$ , consistency parameterization corresponds to the transition from  $\mathbf{x}_t$  to  $\mathbf{x}_0$ :

$$\mathbf{D}_\theta(\mathbf{x}_t, t) = \frac{\hat{\sigma}_t \mathbf{x}_t - \sigma_t \mathbf{F}_\theta(c_{\text{in}}(t) \mathbf{x}_t, c_{\text{noise}}(t))}{\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t}, \quad (41)$$

where  $\frac{\hat{\sigma}_t}{\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t} = A_{t,0}$  and  $-\frac{\sigma_t}{\hat{\sigma}_t \alpha_t - \hat{\alpha}_t \sigma_t} = B_{t,0}$  correspond to TiM parameterizations.

When using loss function  $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2^2$ , [70] show that the gradient of continuous-time consistency models is:

$$\begin{aligned} & \nabla_\theta \mathbb{E}_{\mathbf{x}_t, t} \left[ \mathbf{D}_\theta^T(\mathbf{x}_t, t) \frac{d\mathbf{D}_{\theta^-}(\mathbf{x}_t, t)}{dt} \right] \\ &= \nabla_\theta \mathbb{E}_{\mathbf{x}_t, t} \left[ [B_{t,0} \mathbf{f}_\theta^{\text{cm}}(\mathbf{x}_t, t)]^T \frac{d\mathbf{D}_{\theta^-}(\mathbf{x}_t, t)}{dt} \right] \\ &= \mathbb{E}_{\mathbf{x}_t, t} \left[ B_{t,0} [\nabla_\theta \mathbf{f}_\theta^{\text{cm}}(\mathbf{x}_t, t)]^T \frac{d\mathbf{D}_{\theta^-}(\mathbf{x}_t, t)}{dt} \right] \\ &= \mathbb{E}_{\mathbf{x}_t, t} \left[ B_{t,0} \nabla_\theta [\mathbf{f}_\theta^{\text{cm}}(\mathbf{x}_t, t)]^T \left( \frac{dA_{t,0}}{dt} \mathbf{x}_t + A_{t,0} \frac{d\mathbf{x}_t}{dt} + \frac{dB_{t,0}}{dt} \mathbf{f}_{\theta^-}^{\text{cm}} + B_{t,0} \frac{d\mathbf{f}_{\theta^-}^{\text{cm}}}{dt} \right) \right], \end{aligned} \quad (42)$$

where  $\mathbf{f}_\theta^{\text{cm}}(\mathbf{x}_t, t) = \mathbf{F}_\theta(c_{\text{in}}(t) \mathbf{x}_t, c_{\text{noise}}(t))$  represents the network in consistency models. As  $\mathbf{x}_t = \alpha_t \mathbf{x} + \sigma_t \boldsymbol{\varepsilon}$ ,  $\frac{d\mathbf{x}_t}{dt} = \frac{d\alpha_t}{dt} \mathbf{x} + \frac{\sigma_t}{dt} \boldsymbol{\varepsilon}$ , we have:

$$\begin{aligned} & \frac{dA_{t,0}}{dt} \mathbf{x}_t + A_{t,0} \frac{d\mathbf{x}_t}{dt} + \frac{dB_{t,0}}{dt} \mathbf{f}_{\theta^-}^{\text{cm}} + B_{t,0} \frac{d\mathbf{f}_{\theta^-}^{\text{cm}}}{dt} \\ &= \frac{dA_{t,0}}{dt} (\alpha_t \mathbf{x} + \sigma_t \boldsymbol{\varepsilon}) + A_{t,0} \left( \frac{d\alpha_t}{dt} \mathbf{x} + \frac{d\sigma_t}{dt} \boldsymbol{\varepsilon} \right) + \frac{dB_{t,0}}{dt} \mathbf{f}_{\theta^-}^{\text{cm}} + B_{t,0} \frac{d\mathbf{f}_{\theta^-}^{\text{cm}}}{dt} \\ &= (\alpha_t \frac{dA_{t,0}}{dt} + A_{t,0} \frac{d\alpha_t}{dt}) \mathbf{x} + (\sigma_t \frac{dA_{t,0}}{dt} + A_{t,0} \frac{d\sigma_t}{dt}) \boldsymbol{\varepsilon} + \frac{dB_{t,0}}{dt} \mathbf{f}_{\theta^-}^{\text{cm}} + B_{t,0} \frac{d\mathbf{f}_{\theta^-}^{\text{cm}}}{dt}. \end{aligned} \quad (43)$$

Based on Eqs. (24) and (25), we have:

$$\alpha_t \frac{dA_{t,0}}{dt} + A_{t,0} \frac{d\alpha_t}{dt} = -\hat{\alpha}_t \frac{dB_{t,0}}{dt} - B_{t,0} \frac{d\hat{\alpha}_t}{dt}, \quad \sigma_t \frac{dA_{t,0}}{dt} + A_{t,0} \frac{d\sigma_t}{dt} = -\hat{\sigma}_t \frac{dB_{t,0}}{dt} - B_{t,0} \frac{d\hat{\sigma}_t}{dt}. \quad (44)$$

Therefore, we have:

$$\begin{aligned} & \frac{dA_{t,0}}{dt} \mathbf{x}_t + A_{t,0} \frac{d\mathbf{x}_t}{dt} + \frac{dB_{t,0}}{dt} \mathbf{f}_{\theta^-}^{\text{cm}} + B_{t,0} \frac{d\mathbf{f}_{\theta^-}^{\text{cm}}}{dt} \\ &= B_{t,0} \left( \frac{d\mathbf{f}_{\theta^-}^{\text{cm}}}{dt} - \frac{d\hat{\alpha}_t}{dt} \mathbf{x} - \frac{d\hat{\sigma}_t}{dt} \boldsymbol{\varepsilon} \right) + \frac{dB_{t,0}}{dt} (\mathbf{f}_{\theta^-}^{\text{cm}} - \hat{\alpha}_t \mathbf{x} - \hat{\sigma}_t \boldsymbol{\varepsilon}). \end{aligned} \quad (45)$$

Substituting the above equation into Eq. (42), the gradient of continuous-time consistency models is:

$$\begin{aligned} & \nabla_\theta \mathbb{E}_{\mathbf{x}_t, t} \left[ \mathbf{D}_\theta^T(\mathbf{x}_t, t) \frac{d\mathbf{D}_{\theta^-}(\mathbf{x}_t, t)}{dt} \right] \\ &= \mathbb{E}_{\mathbf{x}_t, t} \left[ B_{t,0} [\nabla_\theta \mathbf{f}_\theta]^T \left( B_{t,0} \left( \frac{d\mathbf{f}_{\theta^-}^{\text{cm}}}{dt} - \frac{d\hat{\alpha}_t}{dt} \mathbf{x} - \frac{d\hat{\sigma}_t}{dt} \boldsymbol{\varepsilon} \right) + \frac{dB_{t,0}}{dt} (\mathbf{f}_{\theta^-}^{\text{cm}} - \hat{\alpha}_t \mathbf{x} - \hat{\sigma}_t \boldsymbol{\varepsilon}) \right) \right], \\ &= \mathbb{E}_{\mathbf{x}_t, t} \left[ (B_{t,0} \frac{dB_{t,0}}{dt}) [\nabla_\theta \mathbf{f}_\theta]^T \left( \mathbf{f}_{\theta^-}^{\text{cm}} - \hat{\alpha}_t \mathbf{x} - \hat{\sigma}_t \boldsymbol{\varepsilon} + \frac{B_{t,0}}{\frac{dB_{t,0}}{dt}} \left( \frac{d\mathbf{f}_{\theta^-}^{\text{cm}}}{dt} - \frac{d\hat{\alpha}_t}{dt} \mathbf{x} - \frac{d\hat{\sigma}_t}{dt} \boldsymbol{\varepsilon} \right) \right) \right], \end{aligned} \quad (46)$$

Note that TiM network  $\mathbf{f}_\theta(\mathbf{x}_t, t, 0)$  corresponds to the network  $\mathbf{f}_\theta^{\text{cm}}(\mathbf{x}_t, t)$  in consistency models. The only difference between Eq. (46) and Eq. (39) is a term  $(B_{t,0} \frac{dB_{t,0}}{dt})$ , which can be bridged by a weighting function.

**Consistency trajectory models, phased consistency models, and shortcut models.** These models learn to transition from one state to another state in a discrete manner, while our TiM generalizes this to the continuous-time domain. The core of our method is the TiM identity equation Eq. (8), which determines the function for the transition between two arbitrary states.

For consistency trajectory models (CTM) [37] and phased consistency models (PCM) [77], they targets at intermediate state  $\mathbf{x}_r$ , where  $0 \leq r \leq t_{n-1}$ , thus leading to the identity equation:

$$\Psi(\mathbf{x}_{t_n}, \mathbf{f}_\theta(\mathbf{x}_{t_n}, t_n, r), r) = \Psi(\mathbf{x}_{t_{n-1}}, \mathbf{f}_\theta(\mathbf{x}_{t_{n-1}}, t_{n-1}, r), r), \quad (47)$$

where  $0 < t_1 < \dots < t_n < \dots < t_N = T$  represents the discrete timesteps, and  $\Psi$  is an ODE solver to obtain the state at timestep  $r$ . It is noteworthy that PCM splits the entire trajectory into several segments and learns this identity on each segment independently.

Shortcut models [21] adopts the OT-flow-matching [43] as the transport, the ODE solver is:  $\Psi(\mathbf{x}_t, \mathbf{f}_\theta(\mathbf{x}_t, t, r), r) = \mathbf{x}_t - (t - r)\mathbf{f}_\theta(\mathbf{x}_t, t, r)$ . The original identity equation is:  $(t - r)\mathbf{f}_\theta(\mathbf{x}_t, t, r) = (t - s)\mathbf{f}_\theta(\mathbf{x}_t, t, s) + (s - r)\mathbf{f}_\theta(\mathbf{x}_s, s, r)$ . This identity equation of shortcut models can be rearranged as:

$$\begin{aligned} (t - r)\mathbf{f}_\theta(\mathbf{x}_t, t, r) &= (t - s)\mathbf{f}_\theta(\mathbf{x}_t, t, s) + (s - r)\mathbf{f}_\theta(\mathbf{x}_s, s, r) \\ \implies \mathbf{x}_t - (t - r)\mathbf{f}_\theta(\mathbf{x}_t, t, r) &= \mathbf{x}_t - (t - s)\mathbf{f}_\theta(\mathbf{x}_t, t, s) - (s - r)\mathbf{f}_\theta(\mathbf{x}_s, s, r) \\ \implies \mathbf{x}_t - (t - r)\mathbf{f}_\theta(\mathbf{x}_t, t, r) &= \mathbf{x}_s - (s - r)\mathbf{f}_\theta(\mathbf{x}_s, s, r) \\ \implies \Psi(\mathbf{x}_t, \mathbf{f}_\theta(\mathbf{x}_t, t, r), r) &= \Psi(\mathbf{x}_s, \mathbf{f}_\theta(\mathbf{x}_s, s, r), r), \end{aligned} \quad (48)$$

where  $s = \frac{t+r}{2}$ . Based on Eq. (47) and Eq. (48), when  $t_{n-1} = \frac{t_n+r}{2}$ , CTMs are equivalent to shortcut models.

**MeanFlow models.** We show that the TiM Eq. (8) generalizes the MeanFlow [26]. In particular, the training objective of TiM reduces to the MeanFlow objective in the OT-FM [43] transport setting.

As in Tab. 7, OT-FM uses the parameterization  $\{\alpha_t = 1-t, \sigma_t = t, \hat{\alpha}_t = -1, \hat{\sigma}_t = 1\}$ , leading to the TiM parameterization  $\{B_{t,r} = r - t, \frac{dB_{t,r}}{dt} = -1\}$ . Therefore, the TiM training objective becomes:

$$\mathbb{E}_{\mathbf{x}, \varepsilon, t} \left[ d \left( \mathbf{f}_\theta(\mathbf{x}_t, t, r) - \left( \mathbf{z} - \mathbf{x} - (t - r) \frac{d\mathbf{f}_{\theta^{-}, t, r}}{dt} \right) \right) \right]. \quad (49)$$

This corresponds to the training objective of MeanFlow.

## C. Implementation Details

### C.1. Model Architecture

We illustrate the model architecture in Fig. 4. As we incorporate decoupled time embedding and interval-aware attention designs into DiT architecture, we use LoRA-AdaLN to avoid increasing model size. Specifically, given attention hidden size  $D$ , LoRA rank is set as  $r = \frac{1}{3}D$ , such as  $D = 1152$  and  $r = 384$  in XL-models. For text-to-image generation, we incorporate caption features through CrossAttention mechanism, as in Fig. 5.

### C.2. Text-to-Image Training Details

**Native-Resolution Training.** We adopt the VAE-specific native resolution training for text-to-image generation. As we use DC-AE [13] with 32 downsampling scale, an image with shape  $H \times W$  is resized to shape  $(32 \cdot \lfloor \frac{H}{32} \rfloor) \times (32 \cdot \lfloor \frac{W}{32} \rfloor)$ . For example, an image with shape  $1025 \times 513$  is resized to  $1024 \times 512$ , preserving resolution and aspect ratio as much as possible. Images of the same size are grouped into resolution buckets for batching.

We set the base batch size as 16 on a single GPU for  $1024 \times 1024$  resolution bucket, then for  $H \times W$  resolution bucket, the minimal batch size is  $B = \lfloor \frac{16 \times H \times W}{1024 \times 1024} \rfloor$ . For instance, the  $512 \times 512$  resolution bucket holds the minimal batch size as  $B = 64$ , while the  $2048 \times 2048$ -resolution bucket holds the minimal batch size as  $B = 4$ . The maximum resolution is  $4096 \times 4096$  with  $B = 1$ . For data parallelism, each device processes distinct buckets with their corresponding batch sizes, maintaining a similar token budget.

**Resolution-Dependent Timestep Shifting.** Sampling from a single timestep distribution is suboptimal across resolutions ranging from less than  $256 \times 256$  to  $4096 \times 4096$  pixels. Intuitively, higher-resolution images require stronger corruption (more noise) to destroy the signal, while lower-resolution images require less noise. Given an image with  $n = H_1 \times W_1$

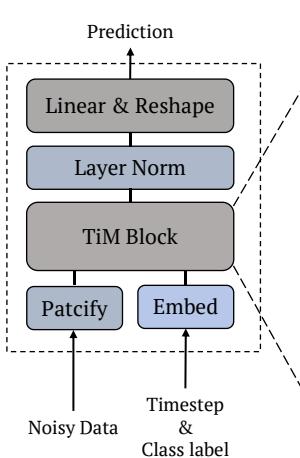


Figure 4. TiM Model Architecture.

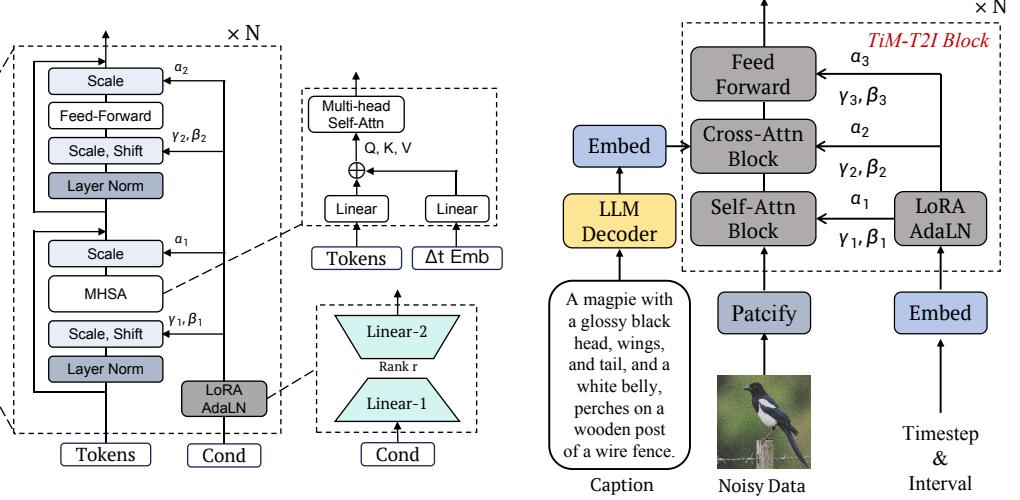


Figure 5. TiM T2I block.

pixels and its high-resolution counterpart with  $m = H_2 \times W_2$  pixels, Esser et al. [20] provides an equation to map the timestep  $t_n$  to  $t_m$ :

$$t_m = \frac{\sqrt{\frac{m}{n}} t_n}{1 + (\sqrt{\frac{m}{n}} - 1)t_n}. \quad (50)$$

In our practice, we set the base pixel number as  $n = 1024 \times 1024$ , and apply this mapping to all sampled timesteps.

**Model-Guidance Training.** Tang et al. [73] propose a model-guidance training target to improve sampling fidelity. We adopt this approach for text-to-image training. Under our formulation, the target becomes:

$$\hat{f} = \hat{\alpha}_t \mathbf{x} + \hat{\sigma}_t \boldsymbol{\varepsilon} - f_{\theta^*, t, t}^{\text{uncond}}) + \frac{B_{t,r}}{\frac{dB_{t,r}}{dt}} \left[ \frac{d\hat{\alpha}_t}{dt} \cdot \mathbf{x} + \frac{d\hat{\sigma}_t}{dt} \cdot \boldsymbol{\varepsilon} - \frac{df_{\theta^*, t, r}}{dt} \right] + (\omega - 1)(f_{\theta^*, t, t}^{\text{cond}}, \quad (51)$$

where  $\omega$  denotes the Classifier-Free Guidance (CFG) scale,  $\theta^*$  is the Exponential Moving Average (EMA) of  $\theta$ ,  $f_{\theta^*, t, t}^{\text{cond}}$  and  $f_{\theta^*, t, t}^{\text{uncond}}$  respectively represent the conditional and unconditional outputs.

**From-Scratch Training.** The TiM-T2I model contains  $865M$  parameters with the patch size of 1. We train from scratch for about 30 days across 16 NVIDIA-A100 GPUs with a constant learning rate of  $4 \times 10^{-4}$ , using PyTorch-FSDP [89] and half-precision (`torch.bfloat16`) for memory efficiency. Following Tang et al. [73], we use model-guidance target Eq. (51) with CFG scale  $\omega = 1.75$  after  $100K$  iterations.

## D. Additional Results

Transport	NFE=1	NFE=8	NFE=50
OT-FM [43, 44]	<b>49.91</b>	26.09	17.99
TrigFlow [45]	67.32	<b>25.14</b>	18.28
EDM [35, 36]	53.64	37.01	24.06
VP-SDE [31, 69]	78.98	37.44	35.72

Table 9. Ablation Studies on different transports.

Method	$\epsilon$	Speed	1-step	4-step	50-step	$t = r$	$r = 0$	1-step	4-step	50-step
JVP	n.a.	1.8 iter/s	49.75	26.22	18.11	0%	0%	52.08	31.52	24.85
DDE	0.0001	2.4 iter/s	111.25	23.34	18.38	0%	10%	53.46	32.49	25.74
DDE	0.0002	2.4 iter/s	80.14	23.83	17.58	10%	10%	51.74	29.75	22.56
DDE	0.0005	2.4 iter/s	67.09	24.33	<b>16.93</b>	20%	10%	50.98	28.41	20.74
DDE	0.001	2.4 iter/s	<b>48.83</b>	<b>24.73</b>	17.03	30%	10%	50.09	27.01	19.20
DDE	0.002	2.4 iter/s	49.07	25.54	17.59	40%	10%	49.88	26.42	18.54
DDE	0.005	2.4 iter/s	49.91	26.09	17.99	50%	10%	<b>47.46</b>	24.62	17.10
DDE	0.01	2.4 iter/s	50.05	26.53	18.33	60%	10%	48.29	24.55	16.58
DDE	0.02	2.4 iter/s	49.72	26.67	18.33	70%	10%	48.44	24.05	16.32
DDE	0.05	2.4 iter/s	49.90	27.05	18.79	80%	10%	48.26	<b>22.88</b>	<b>15.34</b>

Table 10. The impacts of DDE  $\epsilon$  on generation performance.

Table 11. Timestep sampling comparison.

Weighting	transform( $t$ ) = $t$			transform( $t$ ) = $t/(1-t)$			transform( $t$ ) = $\tan(t)$		
	NFE=1	NFE=8	NFE=50	NFE=1	NFE=8	NFE=50	NFE=1	NFE=8	NFE=50
(a) Reciprocal	48.25	25.29	17.42	56.65	<b>24.33</b>	<b>16.58</b>	49.65	25.22	17.39
(b) SMS	49.01	25.76	17.75	72.56	25.15	17.01	48.93	25.23	17.19
(c) Sqrt	<b>48.24</b>	25.82	17.87	<b>49.93</b>	24.73	16.85	<b>47.46</b>	<b>24.62</b>	<b>17.10</b>
(d) Square	48.55	<b>25.31</b>	<b>17.11</b>	70.83	25.79	17.80	48.86	24.91	17.15

Table 12. Ablation studies on different time weighting functions.

## D.1. Additional Ablations

We provide additional ablation results in this section.

**Transport Comparison.** We conduct ablation studies on different transports in Tab. 9. We find that different transports affect the convergence speed, where OT-FM and TrigFlow perform best, EDM is slightly worse, and VP-SDE performs the worst. Thus, we adopt OT-FM in all experiments.

**Differential Derivation Equation Calculation.** As we incorporate a small quantity  $\epsilon$  into Eq. (10) to calculate the time derivative of network. We systematically evaluate the impact of  $\epsilon$  on numerical accuracy in Tab. 10 and observe that  $\epsilon \in [0.001, 0.01]$  yields high precision. For training stability, we adopt  $\epsilon = 0.005$  in all experiments.

**Timestep Sampling.** Using the TiM-B/4 model, we observe improved performance when a portion of timesteps is fixed to  $t = r$ , as in Tab. 11. The best results are obtained with 50% of samples using  $t = r$  (diffusion training) and 10% using  $r = 0$  (consistency training).

**Time Weighting.** Using the TiM-B/4 model, we provide a systematic analysis of time weighting as in Eq. (11). We study three types of transformations: (1)  $\text{transform}(t) = t$ , (2)  $\text{transform}(t) = \frac{t}{1-t}$ , (3)  $\text{transform}(t) = \tan(t)$ ; and four types of weighting functions: (1) Reciprocal:  $w_{\text{fn}}(t, r) = \frac{1}{\sigma_d + t - r}$ , (2) Soft-Min-SNR (SMS):  $w_{\text{fn}}(t, r) = \frac{1}{\sigma_d^2 + (t - r)^2}$ , (3) SQRT:  $w_{\text{fn}}(t, r) = \frac{1}{\sqrt{\sigma_d + t - r}}$ , (4) Square:  $w_{\text{fn}}(t, r) = \frac{1}{(\sigma_d + t - r)^2}$ , where  $\sigma_d$  is the standard deviation of clean data  $x$  ( $\sigma_d = 1$  in our dataset). Empirically, the combination  $w(t) = w(t, r) = \frac{1}{\sqrt{\sigma_d + \tan(t) - \tan(r)}}$  achieves the best performance, slightly surpassing the best results reported in Tab. 4.

## D.2. Class-Guided Image Generation

We provide the results of class-guided image generation in this section.

**Setup.** We use SD-VAE [56] for ImageNet- $256 \times 256$  and DC-AE [13] for ImageNet- $512 \times 512$ , with patch sizes of 2 and 1, respectively. We train an XL-model with  $664M$  parameters for  $750K$  iterations with a batch size of 512 (300 epochs), using a constant learning rate of  $2 \times 10^{-4}$  and AdamW optimizer. We report FID [30], sFID [50], Inception Score (IS) [58], Precision and Recall [38] using ADM evaluation suite [19].

Method	Epochs	Params	NFE	FID↓	sFID↓	IS↑	Prec.↑	Rec.↑
<b><i>Generative Adversarial Networks</i></b>								
BigGAN [8]	-	112M	1	6.95	7.36	171.4	0.87	0.28
StyleGAN-XL [60]	-	166M	1	2.30	4.02	265.12	0.78	0.53
GigaGAN [34]	-	569M	1 × 2	3.45	-	225.52	0.84	0.61
<b><i>Masked and Autoregressive Models</i></b>								
Mask-GIT [10]	555	-	-	6.18	-	182.1	-	-
MagViT-v2 [86]	1080	307M	-	1.78	-	319.4	-	-
LlamaGen-XL [71]	300	775M	-	2.62	5.59	244.08	0.81	0.58
LlamaGen-XXL [71]	300	1.4B	-	2.34	5.97	253.90	0.81	0.59
LlamaGen-3B [71]	300	3.1B	-	2.18	5.97	263.3	0.81	0.58
VAR [75]	350	2.0B	-	1.80	-	365.4	0.83	0.57
MAR [40]	800	943M	-	1.55	-	303.7	0.81	0.62
RandAR-XL [51]	300	775M	-	2.22	-	314.21	0.80	0.60
RandAR-XXL [51]	300	1.4B	-	2.15	-	321.97	0.79	0.62
<b><i>Multi-step Diffusion Models</i></b>								
LDM-4-G [56]	170	395M	250 × 2	3.60	5.12	247.67	0.87	0.48
SimpleDiffusion [32]	800	2B	250 × 2	2.44	-	256.3	-	-
Flag-DiT-3B* [22]	200	4.23B	250 × 2	1.96	4.43	284.8	0.82	0.61
Large-DiT-3B* [22]	340	4.23B	250 × 2	2.10	4.52	304.36	0.82	0.60
MDT [23]	1300	676M	250 × 2	1.79	4.57	283.01	0.81	0.61
MDTv2 [24]	700	676M	250 × 2	1.63	4.45	311.73	0.79	0.65
DiT-XL [52]	1400	675M	250 × 2	2.27	4.60	278.24	0.83	0.57
SiT-XL [49]	1400	675M	250 × 2	2.06	4.49	277.50	0.83	0.59
FlowDCN-XL [78]	400	675M	250 × 2	2.00	4.37	263.16	0.82	0.58
SiT-REPA-XL [87]	800	675M	250 × 2	1.42	4.70	305.7	0.80	0.65
DoD-XL [88]	300	613M	250 × 2	1.73	5.14	304.31	0.79	<b>0.64</b>
SiT-RealS-XL [82]	400	675M	250 × 2	1.82	4.45	268.54	0.81	0.60
<b><i>Few-step Consistency Models</i></b>								
MeanFlow-XL† [26]	1000	675M	1	3.43	-	-	-	-
iCT-XL [67]	-	675M	1 × 2	20.30	-	-	-	-
Shortcut-XL [21]	250	675M	1 × 2	10.60	-	-	-	-
			4 × 2	7.80	-	-	-	-
IMM-XL [92]	3840	675M	1 × 2	7.77	-	-	-	-
			2 × 2	3.99	-	-	-	-
			4 × 2	2.51	-	-	-	-
<b><i>Any-step Transition Models</i></b>								
TiM-XL	300	664M	1†	3.26	4.37	210.33	0.75	0.59
			1	7.11	4.97	140.39	0.71	0.63
			1 × 2	6.14	6.21	151.79	0.74	0.59
			2 × 2	3.61	6.74	189.99	0.79	0.58
			4 × 2	2.62	5.57	203.41	0.79	0.60
			250 × 2	1.65	5.02	248.12	0.79	0.63

Table 13. **Performance comparison on ImageNet-256 × 256 class-guided generation.** \*: Flag-DiT-3B and Large-DiT-3B actually have 4.23 billion parameters, where 3B means the parameters of all transformer blocks. †: means using model-guidance in the training, therefore eliminating the usage of CFG.

**Performance Analysis.** We provide the results on ImageNet-256 × 256 and ImageNet-512 × 512 in Tabs. 13 and 14 respectively. Across both ImageNet-256 × 256 and ImageNet-512 × 512, TiM-XL demonstrates strong performance-efficiency

Method	Epochs	Params	NFE	FID↓	sFID↓	IS↑	Prec.↑	Rec.↑
<b><i>Generative Adversarial Networks</i></b>								
BigGAN [8]	-	160M	1	7.5	-	152.8	-	-
StyleGAN-XL [59]	-	168M	1 × 2	2.41	4.06	267.75	0.77	0.52
<b><i>Masked and Autoregressive Models</i></b>								
VAR [75]	1080	307M	-	2.63	-	303.2	-	-
MAGVITv2 [86]	350	2.3B	-	1.91	-	324.3	-	-
<b><i>Multi-step Diffusion Models</i></b>								
SimpleDiffusion [32]	800	2B	250 × 2	3.02	-	248.7	-	-
DiffiT [29]	-	561M	250 × 2	2.67	-	252.12	0.83	0.55
MaskDiT [90]	800	-	250 × 2	2.50	5.10	256.27	0.83	0.56
Large-DiT-3B [22]	368	4.23B	250 × 2	2.52	5.01	303.70	0.82	0.57
ADM-G,U [19]	400	774M	250 × 2	3.85	5.86	221.72	0.84	0.53
U-ViT-H/2 [4]	400	501M	250 × 2	4.05	6.44	263.79	0.84	0.48
DiT-XL [52]	600	675M	250 × 2	3.04	5.02	240.82	<b>0.84</b>	0.54
EDM2-L [36]	1468	778M	64 × 2	1.87	-	-	-	-
EDM2-XL [36]	1048	1.1B	64 × 2	1.80	-	-	-	-
EDM2-XXL [36]	734	1.5B	64 × 2	1.73	-	-	-	-
SiT-XL [49]	600	675M	250 × 2	2.62	4.18	252.21	0.84	0.57
FlowDCN-XL [78]	-	675M	250 × 2	2.44	4.53	252.8	0.84	0.54
SiT-REPA-XL [87]	200	675M	250 × 2	2.08	4.19	274.6	0.83	0.58
<b><i>Few-step Consistency Models</i></b>								
sCT-L [76]	1273	778M	1	5.15	-	-	-	-
			2	4.65	-	-	-	-
sCT-XL [76]	1117	1.1B	1	4.33	-	-	-	-
			2	3.73	-	-	-	-
sCT-XXL [76]	762	1.5B	1	4.29	-	-	-	-
			2	3.76	-	-	-	-
<b><i>Any-step Transition Models</i></b>								
TiM-XL	300	664M	1	5.07	4.29	160.06	0.79	0.59
			1 × 2	4.79	4.36	171.73	0.82	0.57
			2 × 2	4.01	4.22	171.51	0.83	0.58
			4 × 2	2.55	4.24	207.07	0.83	0.57
			250 × 2	1.69	4.66	247.52	0.82	0.62

Table 14. Performance comparison on ImageNet-512 × 512 class-guided generation.

trade-offs: at low NFE (1 to  $4 \times 2$ ), it can compete with few-step consistency models, achieving comparable FID with fewer training epochs and smaller model size. When increasing NFEs, TiM-XL matches or surpasses many multi-step diffusion models in FID, despite training for only 300 epochs. Notably, TiM demonstrates remarkable generation quality and shows stable gains as NFE increases.

## E. Qualitative Results

We provide the qualitative results in Fig. 6.



An oil-on-canvas masterpiece that captures the dynamic essence of a blue night sky, bursting with the energy of swirling azure hues and speckled with stars that seem to explode in shades of yellow. Dominating the celestial scene is a luminous, fuzzy-the canvas. Below this cosmic display, a tranquil village is depicted to the right, its quaint houses huddled in repose. On the left, a towering cypress tree stretches upwards, its branches undulating like flames, creating a stark contrast against the serene sky. In the distance, amidst the gentle roll of blue hills, the spire of a church stands tall, a silent sentinel overlooking the sleepy hamlet.



realistic, male, surfer, 80s, chill, relaxed, stoner, aviators, beach, illustration, shaggy hair, round face, round chin, brown hair, photorealistic



knight in full silver armor with a red scarf towering over the camera, sunset in the background, fantasy, mountains, clouds seeming to protude from his figure, photorealistic



portrait of wolf, lots of colour, pen and soft watercolour in style of sarah taylor art



A close-up image of an intricately designed lotus flower, which appears to be crafted entirely from crystal-clear water droplets. The flower is set against a backdrop of soft green lily pads floating on a tranquil pond. Sunlight filters through the scene, highlighting the delicate texture and the shimmering surface of the water-formed petals.



a puppy and a young cat in a cozy room, close up photorealistic image with high details. Picture shows sun flares with warm light



fantasy, a majestic sky filled with stars and galaxies, over looking a serene lake.

**Figure 6. High-resolution and multi-aspect generations from TiM (128 NFEs).** TiM attains up to  $4096 \times 4096$  resolution and reliably handles multiple aspect ratios, including  $1024 \times 4096$  and  $2560 \times 1024$ .