COT-FM: Cluster-wise Optimal Transport Flow Matching

Chiensheng Chiang* Kuan-Hsun Tu*† Jia-Wei Liao*
Cheng-Fu Chou Tsung-Wei Ke
National Taiwan University

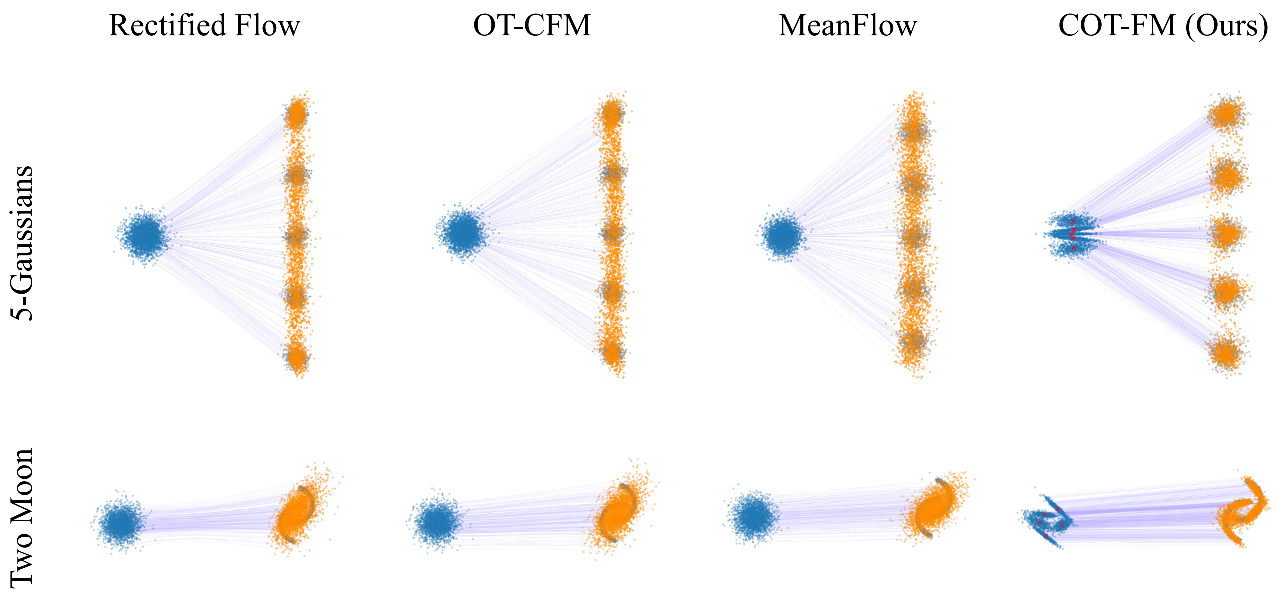

Figure 1. **COT-FM Yields Straight, Structure-Preserving Transport Flows.** Blue points show the source distribution (Gaussian), and gray points show the target distributions (a 5-component Gaussian mixture and Two Moons). Orange points are generated samples, and purple lines denote the learned transport trajectories. Red crosses mark cluster means under our Cluster-wise Optimal Transport Flow Matching (COT-FM). The method yields straight trajectories while still capturing the structure of each target distribution.

Abstract

We introduce COT-FM, a general framework that reshapes the probability path in Flow Matching (FM) to achieve faster and more reliable generation. FM models often produce curved trajectories due to random or batch-wise couplings, which increase discretization error and reduce sample quality. COT-FM fixes this by clustering target samples and assigning each cluster a dedicated source distribution obtained by reversing pretrained FM models. This divide-and-conquer strategy yields more accurate local transport and significantly straighter vector fields, all without changing the model architecture. As a plug-and-play approach, COT-FM consistently accelerates sampling and improves generation quality across 2D datasets, image generation benchmarks, and robotic manipulation tasks.

1. Introduction

Generative modeling seeks to learn a transformation that maps a simple, known source distribution to the complex, partially known data distribution [22, 25, 44, 50–52], for generating new data samples. Flow Matching (FM) [2, 30] is a specific framework: it regresses a deterministic vector field inducing the desired probability path between two distributions. During inference, source samples are transformed into data samples through integration along the learned vector field. Due to its flexibility and scalability, FM recently emerges as an effective alternative to generative models, demonstrating promising results in a wide range of tasks [4, 33, 56].

The formulation of FM is general. It encompasses dif-

* Equal contribution. † Corresponding author.
Project page: https://embodiedai-ntu.github.io/cotfm

ferent types of generative models, such as diffusion models [22, 52], if the target vector field is constructed appropriately [30]. In practice, straight vector fields are preferred over curved ones, as straighter paths incur lower time-discretization error and therefore reduce sampling steps for generation. Meanwhile, since FM models often do not learn the entire vector field during training due to computational limitations[1], time-discretization error may move samples to unseen locations, leading to distorted transport and low-quality generation. To enforce straightness, one can construct the vector field based on the optimal transport (OT) map [3]–the optimal couplings of samples between two distributions with minimal transport cost. However, the exact solution of OT map is computationally inefficient[2] to obtain for large datasets [9, 56].

To address this limitation, most FM models either adopt random, independent couplings [30] or approximate the global optimal couplings with batch optimal couplings [8, 10, 14, 20, 26, 29, 38, 41], to construct the vector field during training. While conceptually simple, the former yield frequent path crossings and the latter struggle with locality of batchwise approximations [13], both resulting in curved paths. To straighten the learned paths, k-Rectified Flow [32] proposes to iteratively optimize FM models with couplings between source and generated samples. Although this approach provably enhances straightness, it repeatedly trains FM models on self-generated samples, leading to model collapse and therefore degrading generation quality over time [63].

Another line of work bypasses the challenge of learning straight paths, attempting to distill from multi-step model [17, 45, 47, 53, 61, 62] or to learn the average vector field [5, 15, 18, 24] for accelerating generation. The latter idea is to skip sampling steps during inference with the average vector field. Despite their efficiency, these approaches do not modulate the underlying instant vector field of FM models, which remains curved due to random coupling strategy. As a result, such shortcut methods only reduce step count, but do not enhance generation quality. To illustrate these issues, we present a toy example in Fig. 1, where the data distribution comprises five modes and the source distribution is Gaussian. We have three observations: (1) training FM models with either random or batch optimal couplings like OT-CFM [57] results in curved flows, (2) shortcut approaches like MeanFlow [18] do not straighten the learned flows, and (3) curved paths lead to distorted data generation.

We introduce Cluster-wise Optimal Transport Flow Matching (COT-FM), a general and effective framework that accelerates and enhances a wide range of FM models. We observe that training data of existing generative modeling tasks naturally comprise multiple modes, while data of the same mode can be generated from similar source samples, as shown in Fig. 1. Based on these observations, our key insight is to partition data samples into clusters, each assigned its own source distribution, rather than mapping the entire data distribution from a single source. This idea brings two advantages: (1) it reduces each optimal coupling problem to a smaller cluster-level match, which reduces the number of data–source samples and makes batch optimal couplings more effective; and (2) by restricting the space of source distributions, learning the overall vector field becomes more efficient. In order to find optimal source distributions for each data cluster, our second insight is to bootstrap from pre-trained FM models. Since paths learned by these models are naturally reversible and non-intersecting, we can easily obtain source distributions for each cluster, incurring less frequent crossings, by reversing the sampling process of these trained models.

Specifically, COT-FM alternates optimization of the target vector field and the FM model. In the first stage, the FM model regresses the target vector field derived from a prior or previously estimated cluster-wise source distributions. In the second stage, we update these source distributions by reversing the learned paths for each data sample and computing their Gaussian statistics, followed by approximating OT within each cluster using batch-wise couplings. Notably, our formulation accommodates different types of clustering, including class labels or textual descriptions for conditional generation and unsupervised clustering for unconditional generation. Moreover, COT-FM only modulates the target probability path, without altering the FM architecture or input–output mechanisms, making it compatible with most existing FM models and able to improve both generation speed and quality.

COT-FM shows strong empirical gains in both one-step and few-step generation. On 2D transport benchmarks, it achieves the best results across all methods, reducing Wasserstein distance from 0.5421 to 0.1995 on Mixture of 5-Gaussians, from 0.1006 to 0.0266 on Two Moons, and from 0.3900 to 0.2550 on Checkerboard, while also attaining the lowest curvature. On CIFAR-10 [27], COT-FM improves Rectified Flow [32] from 12.6 to 8.23 FID at 10 steps and from 4.45 to 3.97 at 50 steps. At very low sampling budgets, it also reduces 1-step FID from 378.0 to 205.0 and 2-step FID from 173 to 59.1. It also enhances MeanFlow [18], lowering FID from 2.92 to 2.60 (1-step) and 2.88 to 2.53 (2-step). On LIBERO robotic manipulation [31], COT-FM reaches 96.1% (Spatial) and 94.5% (Long) with just 1 NFE, while the FLOWER policy [43] requires 4 NFEs to reach 97.1% and 93.5%. Together, these results show that clustering targets and learning cluster-wise source distributions lead to straighter transport paths and more reliable low-step

[1]For D-dimensional Gaussian distributions, $e^{\alpha D}$ samples are required to cover the entire vector field [58].

[2]The OT map requires cubic time and quadratic memory complexity in the number of samples.

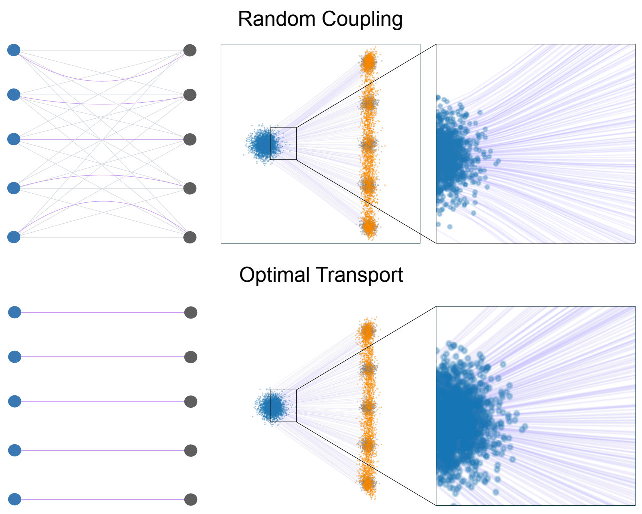

Figure 2. **Vector field results from different coupling strategies.** Random coupling forces the model to regress inconsistent (gray) velocity targets, creating ambiguous intersections and pushing the model toward an averaged (purple) direction, which produces curved velocity fields. In contrast, optimal transport provides consistent couplings with fewer intersections, enabling the model to learn much straighter velocity fields.

generation. These consistent improvements across domains confirm that reducing flow curvature is key to enabling high-quality generation under extremely low sampling budgets.

2. Preliminary of Flow Matching

The objective of flow matching (FM) models is to match a time-dependent vector field transporting samples from a source distribution p_0 to those of a target distribution p_1. This formulation is conceptually simple and computationally efficient: it avoids expensive simulation during training while enabling generation through ordinary differential equation (ODE) integration during testing.

Formally, let $\boldsymbol{v}_t : [0,1]\times\mathbb{R}^d \to \mathbb{R}^d$ denote a time-varying *vector field* that defines an ODE: $\mathrm{d}\boldsymbol{x}_t = \boldsymbol{v}_t(\boldsymbol{x}_t)\,\mathrm{d}t$. The integration map (or *flow*) of the ODE describes the transformation of sample $\boldsymbol{x}_0$ along the vector field from time 0 to t. In other words, the flow reshapes a simple source distribution p_0 to a complex target distribution p_1, where p_t denote the *probability path*–intermediate density functions transported along the vector field from time 0 to t. The FM objective regresses the vector field $\boldsymbol{v}_t$ with a neural network $\boldsymbol{v}_\theta$ parameterized by weights θ, which is formulated as:

$$\mathcal{L}_{\mathsf{FM}}(\theta) = \mathbb{E}_{t,\boldsymbol{x}_t\sim p_t}\|\boldsymbol{v}_\theta(\boldsymbol{x}_t,t) - \boldsymbol{v}_t(\boldsymbol{x}_t)\|_2^2. \tag{1}$$

However, the derivation of vector field $\boldsymbol{v}_t$ and probability path p_t is often intractable for general source and target distributions [57]. To address this limitation, [30, 57] suggest to construct the target probability path via a mixture of simpler conditional probability paths $p_t(\boldsymbol{x}_t|\boldsymbol{z})$ with variable $\boldsymbol{z}$, which may flexibly correspond to either data sample $\boldsymbol{x}_1$ or a pair of source-data sample $(\boldsymbol{x}_0,\boldsymbol{x}_1)$. Marginalizing $p_t(\boldsymbol{x}_t|\boldsymbol{z})$ over distribution $q(\boldsymbol{z})$ produces the marginal probability path:

$$p_t(\boldsymbol{x}_t) = \int p_t(\boldsymbol{x}_t|\boldsymbol{z})q(\boldsymbol{z})\,\mathrm{d}\boldsymbol{z}. \tag{2}$$

Meanwhile, we define the marginal vector field:

$$\boldsymbol{v}_t(\boldsymbol{x}_t) = \mathbb{E}_{q(\boldsymbol{z})}\left[\frac{\boldsymbol{v}_t(\boldsymbol{x}_t|\boldsymbol{z})p_t(\boldsymbol{x}_t|\boldsymbol{z})}{p_t(\boldsymbol{x}_t)}\right]. \tag{3}$$

If conditional vector field $\boldsymbol{v}_t(\boldsymbol{x}_t|\boldsymbol{z})$ generates conditional probability path $p_t(\boldsymbol{x}_t|\boldsymbol{z})$, the marginal vector field $\boldsymbol{v}_t(\boldsymbol{x}_t)$ is shown to generate the marginal probability path $p_t(\boldsymbol{x}_t)$ [57]. Lastly, we define the conditional flow matching (CFM) objective as:

$$\mathcal{L}_{\mathsf{CFM}}(\theta) = \mathbb{E}_{t,q(\boldsymbol{z}),p_t(\boldsymbol{x}_t|\boldsymbol{z})}\|\boldsymbol{v}_\theta(\boldsymbol{x}_t,t) - \boldsymbol{v}_t(\boldsymbol{x}_t|\boldsymbol{z})\|_2^2 \tag{4}$$

leading to identical gradients as FM loss:

$$\nabla_\theta\mathcal{L}_{\mathsf{FM}}(\theta) = \nabla_\theta\mathcal{L}_{\mathsf{CFM}}(\theta). \tag{5}$$

Random Coupling. Randomly pairing a source sample $\boldsymbol{x}_0 \sim p_0$ with a target sample $\boldsymbol{x}_1 \sim p_1$ is a common way for training FM models. Specifically, random coupling is formulated by setting variable $\boldsymbol{z}$ to a pair of source-target sample $(\boldsymbol{x}_0,\boldsymbol{x}_1)$, the distribution $q(\boldsymbol{z}) = p_0(\boldsymbol{x}_0)p_1(\boldsymbol{x}_1)$, and the conditional vector field $\boldsymbol{v}_t(\boldsymbol{x}_t|\boldsymbol{z}) = \boldsymbol{x}_1 - \boldsymbol{x}_0$. While each sample pair induces a straight path, the marginal vector field is curved as it aggregates paths of different sample pairs at each point $\boldsymbol{x}$ and time t. As illustrated in Fig. 2, this averaging of conflicting directions produces curved trajectories that lead to distribution misalignment.

Optimal Coupling. To enforce straightness, one can set distribution $q(z)$ to be the 2-Wasserstein OT map:

$$\pi(\boldsymbol{x}_0,\boldsymbol{x}_1) = \underset{\pi\in\Pi}{\arg\inf}\int\|\boldsymbol{x}_0 - \boldsymbol{x}_1\|_2^2\,\mathrm{d}\pi(\boldsymbol{x}_0,\boldsymbol{x}_1) \tag{6}$$

where Π denotes the set of all possible transport plans. However, calculation of OT map $\pi(\boldsymbol{x}_0,\boldsymbol{x}_1)$ requires cubic time complexity in the number of samples, while continuous source distributions like Gaussian have infinite number of samples, therefore solving global OT map for large datasets is computationally infeasible. To address this limitation, existing approaches often approximate the OT map with batch-wise OT map [8, 10, 14, 20, 26, 29, 38, 41]. However, with limited size of mini-batches, these methods struggle with locality of batchwise approximations [13], resulting in curved paths. For a more comprehensive discussion of related work, please refer to the Supplementary Material.

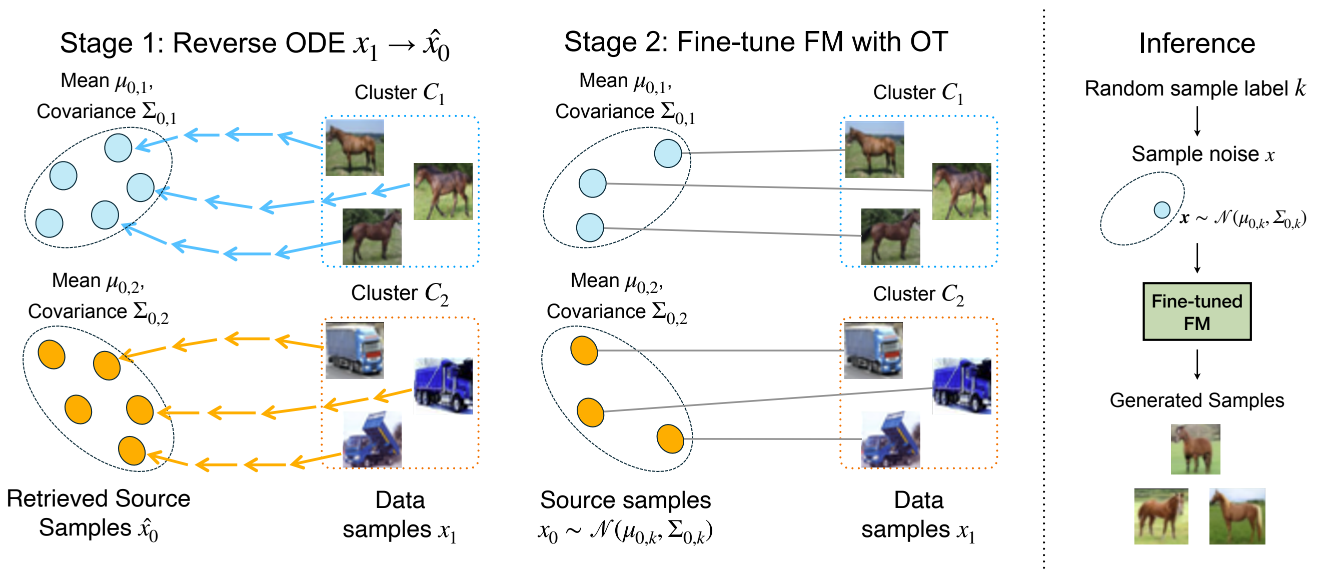
Stage 1: Reverse ODE $x_1 \to \hat{x}_0$
Mean $\mu_{0,1}$, Covariance $\Sigma_{0,1}$
Cluster C_1
Mean $\mu_{0,2}$, Covariance $\Sigma_{0,2}$
Cluster C_2
Retrieved Source Samples $\hat{x}_0$
Data samples x_1
Stage 2: Fine-tune FM with OT
Mean $\mu_{0,1}$, Covariance $\Sigma_{0,1}$
Cluster C_1
Mean $\mu_{0,2}$, Covariance $\Sigma_{0,2}$
Cluster C_2
Source samples $x_0 \sim \mathcal{N}(\mu_{0,k}, \Sigma_{0,k})$
Data samples x_1
Inference
Random sample label k
Sample noise x
$\boldsymbol{x} \sim \mathcal{N}(\mu_{0,k}, \Sigma_{0,k})$
Fine-tuned FM
Generated Samples

Figure 3. **Overview of our proposed method.** In Stage 1, we cluster the dataset, and for each image within a cluster $\mathcal{C}_k$, we reverse the ODE back to the noise space using a pretrained flow model to compute the new mean $\boldsymbol{\mu}_{0,k}$ and covariance $\boldsymbol{\Sigma}_{0,k}$ for that cluster. In Stage 2, we apply optimal transport to finetune the pretrained flow model across all clusters, encouraging it to align with the distributional structure captured in Stage 1. During inference, we first sample a cluster index k, then draw a noise $\boldsymbol{x}$ from $\mathcal{N}(\boldsymbol{\mu}_{0,k}, \boldsymbol{\Sigma}_{0,k})$, and use this noise to generate an image through the finetuned flow model.

3. Method

We propose COT-FM, a general, plug-and-play framework that seeks an optimal vector field to accelerate and enhance generation quality of FM models. The core idea is to divide-and-conquer the calculation of optimal couplings by clustering target samples, identifying corresponding source distributions for each cluster, and approximately solving optimal couplings within each cluster. The FM model is then updated by regressing this vector field with straighter flows. COT-FM alternates between optimizing the target vector field and the FM model. Notably, COT-FM only modulates the underlying target probability path of the FM model, without modifying its architecture or input-output mechanisms. This design makes COT-FM broadly applicable across diverse FM models. An overview of the proposed framework is illustrated in Fig. 3.

3.1. Constructing Cluster-wise Target Vector Field

Our COT-FM begins by partitioning target samples into clusters, then identifies source distributions and constructs a local target vector field for each cluster. This formulation is efficient and general: it reduces the number of source-target samples, making approximation of OT more effective, and it generalizes to different types of clustering, such as class labels or textual descriptions for text-conditional generation and unsupervised clustering for unconditional generation.

Identifying Cluster-wise Source Distributions. To search individual source distributions, we propose to bootstrap from pre-trained FM models. Although originally trained with random coupling, their learned flows are naturally reversible while non-intersecting. The former property allows us to estimate cluster-wise source distributions by integrating the flow backward to trace source samples that generate every data sample of a cluster; the latter property ensures paths between different clusters have few crossings. Formally, we reverse ODE integration to retrieve the source sample of data sample $\boldsymbol{x}_1$:

$$\hat{\boldsymbol{x}}_0 := \boldsymbol{x}_1 - \int_0^1 \boldsymbol{v}_\theta(\hat{\boldsymbol{x}}_t, t)\,\mathrm{d}t, \tag{7}$$

where $\boldsymbol{v}_\theta$ is the velocity field from a trained FM model and $\hat{\boldsymbol{x}}_t$ denotes reversely transported sample $\boldsymbol{x}_1$ from time 1 to t. We denote cluster $\mathcal{C}_k$ as a set of data samples whose clustering index is k. Given K clusters $\{\mathcal{C}_k\}_{k=1}^K$, we obtain source samples $\hat{X}_{0,k} = \{\hat{\boldsymbol{x}}_0 | \boldsymbol{x}_1 \in \mathcal{C}_k\}$ of every data sample within cluster $\mathcal{C}_k$. We approximate their source distribution $p_{0,k}$ as Gaussian based on estimated mean and covariance:

$$\boldsymbol{\mu}_{0,k} = \frac{1}{|\hat{X}_{0,k}|} \sum_{\hat{\boldsymbol{x}}_0 \in \hat{X}_{0,k}} \hat{\boldsymbol{x}}_0, \tag{8}$$

$$\boldsymbol{\Sigma}_{0,k} = \frac{1}{|\hat{X}_{0,k}|} \sum_{\hat{\boldsymbol{x}}_0 \in \hat{X}_{0,k}} (\hat{\boldsymbol{x}}_0 - \boldsymbol{\mu}_{0,k})(\hat{\boldsymbol{x}}_0 - \boldsymbol{\mu}_{0,k})^\top, \tag{9}$$

$$p_{0,k}(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_{0,k}, \boldsymbol{\Sigma}_{0,k}). \tag{10}$$

Algorithm 1 Constructing the cluster-wise vector field

```
1:  Input: Trained FM model $\boldsymbol{v}_\theta$ and $K$ clusters $\{\mathcal{C}_k\}_{k=1}^K$.
2:  for $k = 1$ to $K$ do
3:      Retrieve source samples with reverse ODE:
4:          $\hat{X}_{0,k} \leftarrow \{\boldsymbol{x}_1 - \int_0^1 \boldsymbol{v}_\theta(\hat{\boldsymbol{x}}_t, t)\,\mathrm{d}t | \boldsymbol{x}_1 \in \mathcal{C}_k\}$
5:      Estimate source distributions:
6:          $n_k \leftarrow |\hat{X}_{0,k}|$
7:          $\boldsymbol{\mu}_{0,k} \leftarrow \frac{1}{n_k} \sum_{\hat{\boldsymbol{x}}_0 \in \hat{X}_{0,k}} \hat{\boldsymbol{x}}_0$
8:          $\boldsymbol{\Sigma}_{0,k} \leftarrow \frac{1}{n_k} \sum_{\hat{\boldsymbol{x}}_0 \in \hat{X}_{0,k}} (\hat{\boldsymbol{x}}_0 - \boldsymbol{\mu}_{0,k})(\hat{\boldsymbol{x}}_0 - \boldsymbol{\mu}_{0,k})^\top$
9:          $p_{0,k}(\boldsymbol{x}) \leftarrow \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_{0,k}, \boldsymbol{\Sigma}_{0,k})$
10:     Draw source samples $X_{0,k} \leftarrow \{\boldsymbol{x}_0 \mid \boldsymbol{x}_0 \sim p_{0,k}\}$
11:     Cluster-wise OT map: $\pi_k \leftarrow \mathrm{OT}(X_{0,k}, \mathcal{C}_k)$
12: end for
13: Output: Cluster-wise source distributions $\{p_{0,k}\}_{k=1}^K$
    and OT maps $\{\pi_k\}_{k=1}^K$.
```

Extension to Non-fixed Clustering. So far, identifying cluster-wise source distributions assumes a fixed set of clusters, which may not hold in some applications. For example, in vision-conditioned robot policies, treating each observation as a cluster causes the number of clusters to grow during rollout. To address this, we introduce a learning-based module that predicts the source distribution for each cluster. See the Supplementary Material for additional details.

Approximating OT within Each Cluster. To enforce straight paths, COT-FM next updates the target vector field by calculating separate OT maps between each cluster of data samples and their assigned source distributions. Although exact solution of cluster-wise OT maps remains computationally intractable, our method reduces the number of source-target samples, thereby making batch-wise approximation more feasible. Specifically, for cluster $\mathcal{C}_k$, we draw a set of source samples $X_{0,k} = \{\boldsymbol{x}_0 \mid \boldsymbol{x}_0 \sim p_{0,k}\}$ from estimated source distributions $p_{0,k}$ and calculate the OT map π_k between source samples $X_{0,k}$ and data samples $\mathcal{C}_k$ based on Eqn. 6. These cluster-wise OT maps $\{\pi_k\}_{k=1}^K$ jointly, inherently composes a vector field (Eqn. 3), which the FM model is later optimized to regress. The detailed procedure of constructing cluster-wise vector field is presented in Algorithm 1.

3.2. Optimizing FM Models

The optimization stage follows the standard FM training procedure. As detailed in Algorithm 2, we compose a minibatch by randomly sampling source-target pairs based on pre-computed cluster-wise OT maps $\{\pi_k\}_{k=1}^K$. First, we randomly draw cluster index k with a probability proportional to the cluster size $\frac{|\mathcal{C}_k|}{\sum_{j=1}^K |\mathcal{C}_j|}$, followed by sampling a source-target sample pair with the corresponding OT map: $(\boldsymbol{x}_0, \boldsymbol{x}_1) \sim \pi_k$. Secondly, we sample time step t from a

Algorithm 2 Optimizing the FM model

```
1:  Input: FM model $\boldsymbol{v}_\theta$, batch size $M$, training steps $L$, $K$
    clusters $\{\mathcal{C}_k\}_{k=1}^K$, and cluster-wise OT maps $\{\pi_k\}_{k=1}^K$.
2:  for $l = 1$ to $L$ do
3:      Initialize minibatch: $B \leftarrow \{\}$
4:      for $m = 1$ to $M$ do
5:          Sample cluster $k$ with probability $\frac{|\mathcal{C}_k|}{\sum_{j=1}^K |\mathcal{C}_j|}$
6:          Sample a source-target pair: $(\boldsymbol{x}_0, \boldsymbol{x}_1) \sim \pi_k$
7:          Append to minibatch: $B \leftarrow B \cup \{(\boldsymbol{x}_0, \boldsymbol{x}_1)\}$
8:      end for
9:      $\mathcal{L}_{\mathsf{CFM}}(\theta) = \mathbb{E}_{t,(\boldsymbol{x}_0,\boldsymbol{x}_1)\sim B} \|\boldsymbol{v}_\theta(\boldsymbol{x}_t, t) - (\boldsymbol{x}_1 - \boldsymbol{x}_0)\|_2^2$
10:     $\theta \leftarrow \theta - \gamma \nabla_\theta \mathcal{L}_{\mathsf{CFM}}(\theta)$
11: end for
12: Output: Updated FM model $\boldsymbol{v}_\theta$.
```

Algorithm 3 Sampling

```
1: Input: Cluster-wise source distributions $\{p_{0,k}\}_{k=1}^K$, FM
   model $\boldsymbol{v}_\theta$, number of sampling steps $T$.
2: Sample a cluster index $k$ (as in Algorithm 2, line 5)
3: Draw source sample: $\boldsymbol{x} \sim p_{0,k}$
4: for $t = 0$ to $T - 1$ do
5:     $\boldsymbol{x} \leftarrow \boldsymbol{x} + \boldsymbol{v}_\theta(\boldsymbol{x}, \frac{t}{T}) \cdot \frac{1}{T}$
6: end for
7: Output: Generated target sample $\boldsymbol{x}$.
```

uniform distribution $\mathcal{U}(0, 1)$, and calculate the linear interpolation $\boldsymbol{x}_t = (1 - t)\boldsymbol{x}_0 + t\boldsymbol{x}_1$. Lastly, the FM model is optimized to regress the constructed vector field via the conditional flow-matching loss $\mathcal{L}_{\mathsf{CFM}}$ (Eqn. 4).

3.3. Alternating Optimization and Sampling

Starting with a pre-trained FM model, COT-FM alternates between refining cluster-wise target vector field and updating the FM model. Empirically, we find that the model performance converges within a small number of alternation rounds. Since COT-FM does not alter the FM architecture, it preserves the standard inference procedure. The only modification lies in the initialization step: whereas the original FM model samples an initial source point from a single global source distribution, COT-FM first samples a cluster index k and then draws the initial sample from the corresponding source distribution $p_{0,k}$. The cluster index k is sampled according to the probability $\frac{|\mathcal{C}_k|}{\sum_{j=1}^K |\mathcal{C}_j|}$, where $|\mathcal{C}_k|$ denotes the number of samples in cluster k, for a multinomial distribution. Other sampling methods are described in Section 4.5. The full generation process is summarized in Algorithm 3.

4. Experiments

We evaluate COT-FM under the following setups: unconditional 2D point cloud generation, unconditional and condi-

Dataset: 2D Examples	NFE	Mixture of 5-Gaussian		Two Moon		Checker Board	
		Wasserstein^2	Curvature	Wasserstein^2	Curvature	Wasserstein^2	Curvature
Rectified Flow [32]	100	0.5421	0.0316	0.1006	0.0111	0.3900	9.1946
OT-CFM [57]	100	0.6582	0.0104	0.1074	0.0020	0.3188	0.1741
MeanFlow [18]	1	0.7612	0.9170	0.1233	-	0.9170	-
COT-FM (Ours)	100	**0.1995**	**0.0084**	**0.0266**	**0.0016**	**0.2550**	**0.1505**

Table 1. **2D synthetic data distributions.** COT-FM outperforms all baselines in both Wasserstein distance and curvature.

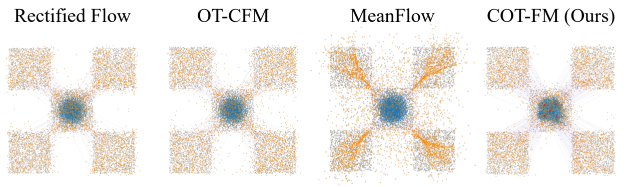

Figure 4. **Visualization of 2D checkerboard data under different methods.** Blue points denote the source distribution, gray points the target distribution, and orange points the generated samples.

Dataset: CIFAR-10	1-step	2-step	10-step	50-step
Rectified Flow [32]				
random coupling	378.0	173	12.6	4.45
+ clustering	296.0	107	10.1	4.19
OT-CFM [57]	226.0	82.2	10.6	4.78
COT-FM (Ours)	**205.0**	**59.1**	**8.23**	**3.97**
MeanFlow [18]				
random coupling	2.92	2.88	-	-
COT-FM (Ours)	**2.60**	**2.53**	-	-

Table 2. **Unconditional image generation on CIFAR-10 [27].** COT-FM improves generation quality across different flow matching baselines and sampling steps. Measured in FID (lower is better).

tional image generation on CIFAR-10 and ImageNet, and text-conditional robotic manipulation on LIBERO. In addition, we perform an extensive ablation study to assess the contribution of each model component.

4.1. Unconditional 2D Point Cloud Generation

To preliminarily validate our idea, we design a simple benchmark for unconditional 2D point cloud generation. We consider three types of data distributions: a mixture of 5 Gaussians, two moons, checkerboard.

We compare against the following baseline methods: (1) Rectified Flow [32], which uses random couplings; (2) OT-CFM [57], which uses batch-wise optimal couplings; (3) MeanFlow [18], which uses random couplings and learns an average velocity field to skip sampling. All baseline models use a standard Gaussian source distribution $p_0(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \mathbf{0}, \boldsymbol{I})$. For OT-CFM and COT-FM, we use the same backbone FM model as Rectified Flow. We apply the K-means algorithm to segment data samples into 5 clusters, and perform two alternation cycles during training.

Following its typical setup, MeanFlow generates data with 1 sampling step, while others use 100 steps. To evaluate model performance, we consider two metrics: *Wasserstein distance* and *curvature*. The former measures how close the generated distribution is to the target distribution, and the latter quantifies the straightness of the learned flows.

We present our quantitative results in Table 1, COT-FM outperforms all baselines significantly. It achieves the smallest Wasserstein distance–0.1995, 0.0266, 0.2550 compared to 0.5421, 0.1006, 0.3900, and the least curvature–0.0084, 0.0016, 0.1505 compared to 0.0104, 0.0020, 0.1741, of the second best method on the four types of data distributions. We show qualitative results in Fig. 1 and Fig. 4. Our generated 2D point cloud is much closer to the original data distributions than others. Notably, we observe that the learned flows of MeanFlow remain curved and its generated samples deviate from the data distributions. These results support our proposition that cluster-wise OT enhances generation quality of FM models, and shortcut models only accelerate but do not enhance quality of data generation. Additional details are provided in the Supplementary Material.

4.2. Unconditional Image Generation

Next, we evaluate COT-FM on the more challenging task of unconditional image generation on CIFAR-10 [27], which contains 50,000 training images of resolution 32×32. We compare COT-FM against three baseline methods: Rectified Flow, OT-CFM, and MeanFlow. Given that COT-FM is model-agnostic, we integrate it with Rectified Flow and MeanFlow, and evaluate each resulting model independently against its original counterpart.

For clustering images, we first encode an image into a latent feature vector using a self-supervised representation learning framework–DINO [6], then apply K-Means algorithm to segment all training images into 100 clusters. It has been shown that clusters derived from such self-supervised features closely correspond to image categories [39, 60]. The clustering result is shown in Fig. 7(a).

Model	Method	100	50	10	2	1
SiT-B/2	Rectified Flow [32]	5.82	5.86	8.25	119.57	264.36
	COT-FM (Ours)	**5.11**	**5.28**	**7.52**	**101.66**	**231.99**
SiT-B/4	Rectified Flow [32]	8.30	8.39	11.16	134.99	276.13
	COT-FM (Ours)	**7.65**	**7.81**	**9.87**	**114.10**	**241.18**

Table 3. **Conditional Image Generation on ImageNet 256×256.** FID (↓) at different NFE steps.

We adopt Fréchet Inception Distance (FID) [21] as the evaluation metric, which measures the similarity between generated data and real data distribution in the latent feature space [54]. Lower FID indicates better generation quality. See the Supplementary Material for architecture, training, and evaluation details.

Table 2 summarizes the results. When using the same backbone FM model as Rectified Flow, introducing cluster-wise random coupling already yields an improvement over the original Rectified Flow without clustering, achieving an absolute performance gain of 82 in FID. Building on this, computing batch-wise optimal coupling within each cluster further enhances performance, yielding an additional FID reduction of 91 with a single sampling step. Remarkably, our COT-FM consistently enhances the FM model when using the same backbone as MeanFlow. It reduces FID from 2.92 to 2.60 and 2.88 to 2.53 with one and two sampling steps. As shown in Fig. 6, our method produces noticeably clearer 1-step generations than Rectified Flow and improves the visual quality over OT-CFM on several object categories. For the 50-step generation setting, COT-FM still achieves a lower FID, reducing it from 4.45 to 3.97. These results highlight the benefit of incorporating sample clustering and optimal coupling into the training pipeline of FM models.

4.3. Conditional Image Generation

We further evaluate COT-FM on conditional image generation using ImageNet [11] at 256×256 resolution. We adopt SiT-B/2 and SiT-B/4 [35] as backbone architectures. In the conditional setting, we group training samples by class labels, treating each class as a cluster for constructing cluster-wise source distributions. For comparison, we train COT-FM by continuing from a pre-trained 1-RF model.

As shown in Table 3, COT-FM consistently improves FID across all NFE settings for both architectures. Notably, the improvement is more pronounced in the low-NFE regime, where straighter transport paths are particularly beneficial. These results demonstrate that our method scales effectively to high-resolution, class-conditional benchmarks without modification to the core algorithm.

4.4. Text-conditional Robotic Action Generation

We evaluate COT-FM on text-conditional robotic action generation using the LIBERO benchmark [31], which provides tabletop manipulation tasks paired with natural-language instructions. This simulation benchmark consists of tabletop manipulation tasks involving a Franka robot arm [16] and 3D object assets. The goal is to generate robot actions that accomplish the manipulation task conditioned on textual instructions. LIBERO comprises multiple task suites; we evaluate on LIBERO-Spatial and LIBERO-Long. The former emphasizes spatial variations, while the latter focuses on long-horizon task execution.

Dataset: LIBERO	NFE	Spatial	Long
FLOWER [43]	4	97.1%	93.5%
FLOWER [43]	1	94.2%	87.3%
2-Rectified Flow [32]	1	95.7%	91.5%
COT-FM (ours)	1	96.1%	**94.5%**

Table 4. **Robotic manipulation on LIBERO benchmarks [31].** COT-FM achieves single-step performance comparable to FLOWER's 4-step generation while outperforming other single-step baselines. Measured in success rate (higher is better).

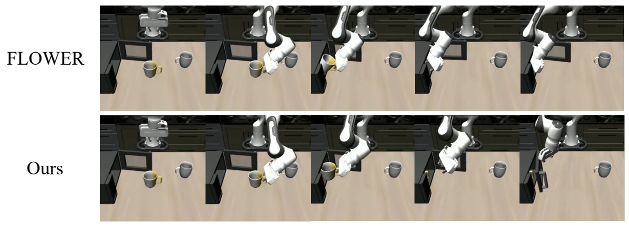

Figure 5. Comparison of two trajectories under the same initial setting in LIBERO Long. Our method successfully places the mug into the microwave and closes the door, while FLOWER fails to insert the mug smoothly and gets stuck inside the microwave.

We compare against two baselines: (1) FLOWER [43], a text-conditioned FM policy trained with random couplings, and (2) Reflow [32], which employs rectified couplings between source and generated samples. Both COT-FM and 2-Rectified Flow use the same backbone as FLOWER for fair comparison. We evaluate performance using the *success rate*, defined as the proportion of successful task completions across all evaluation trials.

Results are summarized in Table 4. Using only a single sampling step, COT-FM outperforms all baseline methods, achieving absolute gains of 0.4% on LIBERO-Spatial and 3% on LIBERO-Long over the second-best method. Remarkably, COT-FM with one sampling step surpasses FLOWER with four steps. This improvement arises because cluster-wise optimal couplings produce straighter vector fields than random couplings, enabling the FM model to learn more accurate flows. See Fig. 5 for rollout visualizations.

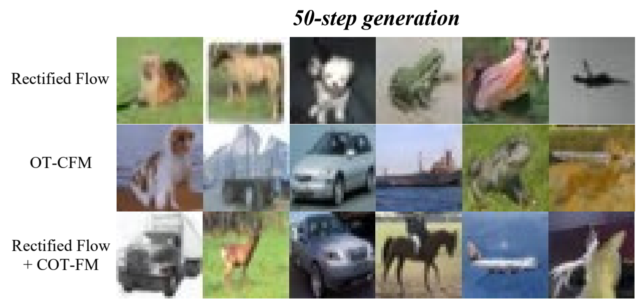

Figure 6. CIFAR-10 visualization for different flow-based methods under 50-step generation settings.

4.5. Ablation Analysis

We conduct ablation studies on CIFAR-10 to analyze design choices of COT-FM. All experiments use 1-RF with 50-step generation unless otherwise specified.

Alternating Iterations. We examine how many alternating optimization cycles are needed. Starting from a pre-trained Rectified Flow model, we alternate between training the flow matching model with cluster-wise optimal transport and refining source distributions through reverse ODE integration. Table 5(a) shows that each iteration progressively reduces FID: 4.45 for the initial model improves to 4.23, 3.97, and 4.17 after 1, 2, and 3 iterations respectively. The diminishing returns suggest that 2 iterations suffice for best performance.

Test Set Generalization. Since COT-FM modifies source distributions based on training data clusters, we verify it does not overfit. Table 5(b) compares FID on training versus held-out test images. It is expected that test dataset has higher FID than train dataset due to only 10000 images in CIFAR-10 test dataset. Both Rectified Flow baseline and COT-FM show consistent performance between train and test sets—4.45, 8.55 for Rectified Flow and 3.97, 8.19 for COT-FM—confirming that cluster-wise source distributions generalize without degrading quality on unseen data.

Different Cluster Sample Probabilities. Table 5(c) compares FID using different methods to sample cluster labels. We show that using the proportional to cluster size as a sampling probability achieves better performance than uniform sampling. We argue that this is because aligning the dataset distribution requires assigning higher sampling frequency to larger clusters, which help maintain data balance and improve FID.

5. Conclusions

We present COT-FM, a general framework that reshapes Flow Matching probability paths for faster and more reliable generation. By clustering data and assigning each cluster its own reversed source distribution, COT-FM straightens trajectories normally distorted by random or batch-wise couplings. This yields a more accurate vector field without altering model architectures or training pipelines. As a plug-and-play method, COT-FM consistently accelerates sampling while improving quality across 2D synthetic data, image generation, and robotic manipulation.

(a) Alternating iterations		**(b) Test set generalization**	
Method	FID	Method	FID
Rectified Flow (0 iter.)	4.45	Rectified Flow (train)	4.45
COT-FM (1 iter.)	4.23	Rectified Flow (test)	8.55
COT-FM (2 iter.)	**3.97**	COT-FM (train)	**3.97**
COT-FM (3 iter.)	4.17	COT-FM (test)	**8.19**

(c) Different cluster sampling probabilities

Method	FID
Uniform	4.26
Proportional to cluster size	**3.97**

Table 5. Ablation studies on CIFAR-10 at 50-step generation. (a) Additional alternating iterations progressively reduce FID. (b) COT-FM maintains consistent performance on held-out test data. (c) Size-proportional cluster sampling outperforms uniform sampling.

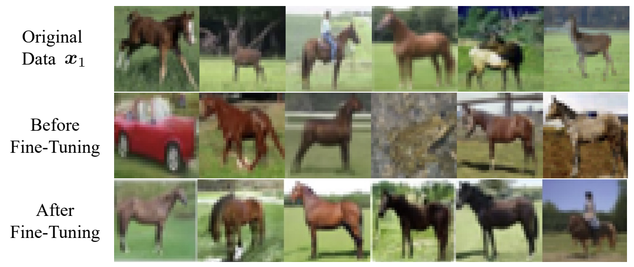

Figure 7. Visualization of a CIFAR-10 cluster. The first row shows representative images from the cluster. Using these images, we apply reverse ODE to estimate the $\mu_{0,k}$ and $\Sigma_{0,k}$ (Fig. 3, Stage 1). The second row shows images generated by sampling noise from this estimated distribution. The last row presents the results after finetuning the pretrained flow model.

Limitations and Future Work. While COT-FM improves flow straightness and sampling efficiency across diverse domains, challenges remain in scaling and generalizing the approach. The reverse ODE step can become inefficient as sample size grows, and our approach depends on clustering quality. The behavior of cluster-wise transport—particularly how local OT captures global structure and why alternating refinement converges within one or two cycles—remains not fully understood. Exploring more scalable clustering strategies and lighter source-distribution estimators offers promising directions for extending COT-FM.

Acknowledgment

This work was supported in part by the AMD–ITRI Joint Laboratory, which provided MI300X high-performance computing resources and technical support for the execution and validation of this research. This work was also supported by the AMD University Program AI & HPC Cluster. We further acknowledge Kuo-Guang Tsai for his technical support on the AMD system cluster infrastructure. This research was also supported by the National Science and Technology Council (NSTC), Taiwan, under Grants 114-2222-E-002-008, 114-2221-E-002-182-MY3, 113-2221-E-002-201, and 115-2634-F-002-001.

References

[1] Tomoharu Aizu, Takeru Oba, Yuki Kondo, and Norimichi Ukita. Robot motion planning using one-step diffusion with noise-optimized approximate motions. *arXiv preprint arXiv:2504.19652*, 2025. 15

[2] Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *International Conference on Learning Representations (ICLR)*, 2023. 1, 14

[3] Jean-David Benamou and Yann Brenier. A computational fluid mechanics solution to the monge-kantorovich mass transfer problem. *Numerische Mathematik*, 2000. 2

[4] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π0: A vision-language-action flow model for general robot control, 2024. *arXiv preprint arXiv:2410.24164*, 2024. 1

[5] 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 (TMLR)*, 2024. 2, 15

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021. 6, 14

[7] Changgu Chen, Libing Yang, Xiaoyan Yang, Lianggangxu Chen, Gaoqi He, Changbo Wang, and Yang Li. Find: Finetuning initial noise distribution with policy optimization for diffusion models. In *ACM International Conference on Multimedia (ACM MM)*, 2024. 15

[8] Ho Kei Cheng and Alexander Schwing. The curse of conditions: Analyzing and improving optimal transport for conditional flow-based generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. 2, 3

[9] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2013. 2

[10] Aram Davtyan, Leello Tadesse Dadi, Volkan Cevher, and Paolo Favaro. Faster inference of flow-based generative models via improved data-noise coupling. In *International Conference on Learning Representations (ICLR)*, 2025. 2, 3, 14

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 7, 15

[12] Luca Eyring, Shyamgopal Karthik, Karsten Roth, Alexey Dosovitskiy, and Zeynep Akata. Reno: Enhancing one-step text-to-image models through reward-based noise optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 15

[13] Kilian Fatras, Thibault Séjourné, Rémi Flamary, and Nicolas Courty. Unbalanced minibatch optimal transport; applications to domain adaptation. In *International Conference on Machine Learning (ICML)*, 2021. 2, 3

[14] Kilian Fatras, Younes Zine, Szymon Majewski, Rémi Flamary, Rémi Gribonval, and Nicolas Courty. Minibatch optimal transport distances; analysis and applications. *arXiv preprint arXiv:2101.01792*, 2021. 2, 3

[15] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. In *International Conference on Learning Representations (ICLR)*, 2025. 2, 15

[16] Claudio Gaz, Marco Cognetti, Alexander Oliva, Paolo Robuffo Giordano, and Alessandro De Luca. Dynamic identification of the franka emika panda robot with retrieval of feasible parameters using penalty-based optimization. *IEEE Robotics and Automation Letters*, 2019. 7

[17] Zhengyang Geng, Ashwini Pokle, and J Zico Kolter. One-step diffusion distillation via deep equilibrium models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 2, 14

[18] Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. 2, 6, 15

[19] Xiefan Guo, Jinlin Liu, Miaomiao Cui, Jiankai Li, Hongyu Yang, and Di Huang. Initno: Boosting text-to-image diffusion models via initial noise optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 15

[20] Etrit Haxholli, Yeti Z. Gürbüz, Oğul Can, and Eli Waxman. Minibatch optimal transport and perplexity bound estimation in discrete flow matching. *arXiv preprint arXiv:2411.00759*, 2024. 2, 3

[21] 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. In *Advances in Neural Information Processing Systems (NIPS)*, 2018. 7, 15

[22] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1, 2, 14

[23] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 14

[24] Dongjun Kim, AI Sony, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Yutong He, Yuki Mitsufuji, and

Stefano Ermon. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. In *International Conference on Learning Representations (ICLR)*, 2024. 2, 15

[25] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014. 1, 14

[26] Nikita Kornilov, Petr Mokrov, Alexander Gasnikov, and Aleksandr Korotin. Optimal flow matching: Learning straight trajectories in just one step. *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 2, 3, 14

[27] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. University of Toronto. 2, 6, 15

[28] Zeming Li, Xiangyue Liu, Xiangyu Zhang, Ping Tan, and Heung-Yeung Shum. Noisear: Autoregressing initial noise prior for diffusion models. *arXiv preprint arXiv:2506.01337*, 2025. 15

[29] Yexiong Lin, Yu Yao, and Tongliang Liu. Beyond optimal transport: Model-aligned coupling for flow matching. *arXiv preprint arXiv:2505.23346*, 2025. 2, 3

[30] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023. 1, 2, 3, 14

[31] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 2, 7, 16

[32] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations (ICLR)*, 2023. 2, 6, 7, 13, 14, 15

[33] Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, et al. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. In *International Conference on Learning Representations (ICLR)*, 2024. 1

[34] Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv preprint arXiv:2101.02388*, 2021. 14

[35] Nanye Ma, Mark Goldstein, Michael S Albergo, Nicholas M Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 7, 15

[36] Jiafeng Mao, Xueting Wang, and Kiyoharu Aizawa. Guided image synthesis via initial image editing in diffusion model. In *ACM International Conference on Multimedia (ACM MM)*, 2023. 15

[37] Jiafeng Mao, Xueting Wang, and Kiyoharu Aizawa. The lottery ticket hypothesis in denoising: Towards semantic-driven initialization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 15

[38] Khai Nguyen, Dang Nguyen, The-Anh Vu-Le, Tung Pham, and Nhat Ho. Improving mini-batch optimal transport via partial transportation. In *International Conference on Machine Learning (ICML)*, 2022. 2, 3

[39] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research (TMLR)*, 2024. 6, 14

[40] Gabriel Peyré and Marco Cuturi. Computational optimal transport. *Foundations and Trends in Machine Learning*, 2019. 12

[41] Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, and Ricky TQ Chen. Multisample flow matching: Straightening flows with minibatch couplings. In *International Conference on Machine Learning (ICML)*, 2023. 2, 3, 14

[42] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, 2024. 15

[43] Moritz Reuss, Hongyi Zhou, Marcel Rühle, Ömer Erdinç Yağmurlu, Fabian Otto, and Rudolf Lioutikov. FLOWER: Democratizing generalist robot policies with efficient vision-language-flow models. In *Conference on Robot Learning (CoRL)*, 2025. 2, 7, 16

[44] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning (ICML)*, 2015. 1, 14

[45] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations (ICLR)*, 2022. 2, 14

[46] Dvir Samuel, Rami Ben-Ari, Simon Raviv, Nir Darshan, and Gal Chechik. Generating images of rare concepts using pre-trained diffusion models. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2024. 15

[47] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 2, 14

[48] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 12, 16

[49] Andreas Sochopoulos, Nikolay Malkin, Nikolaos Tsagkas, João Moura, Michael Gienger, and Sethu Vijayakumar. Fast flow-based visuomotor policies via conditional optimal transport couplings. In *Conference on Robot Learning (CoRL)*, 2025. 14

[50] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)*, 2015. 1, 14

[51] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 15

[52] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2020. 1, 2, 14

[53] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning (ICML)*, 2023. 2, 15

[54] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 7

[55] Robert L. Thorndike. Who belongs in the family? *Psychometrika*, 1953. 14

[56] Alexander Tong, Jessie Huang, Guy Wolf, David Van Dijk, and Smita Krishnaswamy. Trajectorynet: A dynamic optimal transport network for modeling cellular dynamics. In *International Conference on Machine Learning (ICML)*, 2020. 1, 2

[57] Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research (TMLR)*, 2024. 2, 3, 6, 14, 15

[58] Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*. Cambridge university press, 2018. 2

[59] Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2025. 15

[60] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 6

[61] 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 (CVPR)*, 2024. 2, 14

[62] Mingyuan Zhou, Huangjie Zheng, Zhendong Wang, Mingzhang Yin, and Hai Huang. Score identity distillation: Exponentially fast distillation of pretrained diffusion models for one-step generation. In *International Conference on Machine Learning (ICML)*, 2024. 2, 14

[63] Huminhao Zhu, Fangyikang Wang, Tianyu Ding, Qing Qu, and Zhihui Zhu. Analyzing and mitigating model collapse in rectified flow models. *arXiv preprint arXiv:2412.08175*, 2025. 2, 14

[64] Yu Zhu. Meanflow: Pytorch implementation. `https://github.com/zhuyu-cs/MeanFlow`, 2025. PyTorch implementation of Mean Flows for One-step Generative Modeling. 15

Supplementary Material

A. Extension to Conditional Generation

The COT-FM framework naturally extends to conditional generation settings. In unconditional generation, we use unsupervised clustering to partition the target distribution. In conditional generation, the conditions themselves serve as natural cluster labels. For instance, in text-to-image generation, images corresponding to the same text prompt (e.g., "cat") naturally form a cluster. Similarly, in robot manipulation tasks, different observations correspond to different action distributions, providing an implicit cluster.

The key difference is that conditional generation may encounter unseen conditions at test time, such as input observation for robot policy. Unlike unconditional generation where we can precompute mean $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$ for each cluster k, we cannot enumerate all possible conditions in advance. To address this, we learn a conditional model ϕ that dynamically predicts the noise distribution for any given condition c_k:

$$\boldsymbol{x}_0 \sim \mathcal{N}(\boldsymbol{\mu}_\phi(c_k), \boldsymbol{\sigma}_\phi^2(c_k)I), \tag{11}$$

where $\boldsymbol{\mu}_\phi(c_k)$ and $\boldsymbol{\sigma}_\phi(c_k)$ are the predicted mean and standard deviation conditioned on c_k. Since learning a full covariance matrix in high-dimensional space is prohibitively difficult and unstable, we approximate the covariance using only the predicted standard deviation. This allows COT-FM to generate samples from appropriate cluster-wise noise distributions at inference time, maintaining the benefits of reduced trajectory curvature while generalizing to novel conditions. We formulate the problem of training a conditional model ϕ that predicts mean $\boldsymbol{\mu}_\phi(c_k)$ and standard deviation $\boldsymbol{\sigma}_\phi(c_k)$ for a given condition c_k as a reinforcement learning problem within a one-step Markov Decision Process (MDP).

The one-step MDP is defined by a tuple $(\mathcal{S}, \mathcal{A}, R)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ the action space, and R the reward function. Since our framework terminates after one step, at timestep $t = 0$, the conditional model ϕ observes a state $s_0 \in \mathcal{S}$, outputs an action $a_0 \in \mathcal{A}$ and earns a reward $R(s_0, a_0)$. The objective is to maximize the expected return over all states in $\mathcal{S}$. We adopt Proximal Policy Optimization (PPO) [48] algorithm for this reinforcement learning problem.

As shown in Algorithm 4, given an offline dataset $\mathcal{D}$ with $\{(c_k, X_{1,k})\}_{k=1}^K$, our goal is to learn a conditional distribution that best matches the distribution of $X_{1,k}$ under condition c_k. We therefore treat the condition c_k as state s_0 and specify the action as drawing an initial point $\boldsymbol{x}_0$ according to c_k

$$\boldsymbol{x}_0 \sim \mathcal{N}(\boldsymbol{\mu}_\phi(c_k), \boldsymbol{\sigma}_\phi^2(c_k)I), \tag{12}$$

as $a \in \mathcal{A}$. The sampled $\boldsymbol{x}_0$ is then propagated through the ODE to obtain a terminal point $\hat{\boldsymbol{x}}_1$. We define the reward for this transition as

$$R(s_0, a_0) = \exp(-\operatorname{MSE}(\hat{\boldsymbol{x}}_1, \boldsymbol{x}_1)). \tag{13}$$

where MSE denotes mean squared error between the generated and target terminal points. Because the MDP contains only a single transition, the value equals the reward:

$$V = \exp(-\operatorname{MSE}(\hat{\boldsymbol{x}}_1, \boldsymbol{x}_1)), \tag{14}$$

We compute the advantage as

$$\hat{A} = V - \mathbb{E}_{p_k}[V], \tag{15}$$

where p_k is the distribution from condition c_k. We then use these data to update ϕ with PPO.

During finetuning of the pretrained flow model in Algorithm 5 and during sampling in Algorithm 6, the procedure follows our original algorithm, except that the predefined $(\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k)$ are replaced with the conditional outputs $(\boldsymbol{\mu}_\phi(c_k), \boldsymbol{\sigma}_\phi(c_k))$.

Algorithm 4 Training conditional model for non-fixed condition in robot policy

```
1:  Input: Trained FM model $v_\theta$, initialized conditional model $\phi$, offline dataset $\mathcal{D}$ with $\{(c_k, X_{1,k})\}_{k=1}^K$, batch size $M$, training steps $L$.
2:  for $l = 1$ to $L$ do
3:      Initialize minibatch: $B \leftarrow \{\}$
4:      for $m = 1$ to $M$ do
5:          Sample $(c_k, x_1)$ from dataset $\mathcal{D}$
6:          $x_0 \sim \mathcal{N}(\mu_\phi(c_k), \sigma_\phi^2(c_k)I)$
7:          Compute log-likelihood $\log p_\phi(x_0 \mid c_k)$
8:          Roll out FM: $\hat{x}_1 \leftarrow \mathrm{ODESolver}(v_\theta, x_0, c_k)$
9:          $V \leftarrow \exp(-\mathrm{MSE}(\hat{x}_1, x_1))$
10:         $\hat{A} \leftarrow V - \mathbb{E}_{p_k}[V]$
11:         Append to minibatch: $B \leftarrow B \cup \{(c_k, x_0, \hat{A}, \log p_\phi(x_0 \mid c_k)\}$
12:     end for
13:     Optimize $\phi$ according to minibatch with PPO.
14: end for
15: Output: Trained conditional model $\phi$.
```

B. Computational Cost Analysis

Theoretical Analysis. Let n be the dataset size, b the batch size, and K the number of clusters. Exact OT costs $O(n^3)$, while batch OT (used by OT-CFM) costs $O(b^3 \cdot (n/b)) = O(nb^2)$ per epoch [40]. COT-FM computes exact OT within K clusters each epoch: $O((n/K)^3 \cdot K) = O(n(n/K)^2)$. Our focus is the speed–quality tradeoff: as shown in Tab. 6, COT-FM achieves better FID at matched per-epoch OT wall-clock time.

Algorithm 5 Fine-tuning FM model for non-fixed condition in robot policy

```
1:  Input: FM model $\boldsymbol{v}_\theta$, conditional model $\phi$, batch size $M$, training steps $L$, offline dataset $\mathcal{D} = \{(c_k, X_{1,k})\}_{k=1}^{K}$.
2:  for $l = 1$ to $L$ do
3:      Initialize minibatch: $B \leftarrow \{\}$
4:      for $m = 1$ to $M$ do
5:          Sample batch data $(c_k, \boldsymbol{x}_1) \sim \mathcal{D}$
6:          $\boldsymbol{x}_0 \sim \mathcal{N}(\boldsymbol{\mu}_\phi(c_k), \boldsymbol{\sigma}_\phi^2(c_k)I)$
7:          Append to minibatch: $B \leftarrow B \cup \{(\boldsymbol{x}_0, \boldsymbol{x}_1)\}$
8:      end for
9:      $\mathcal{L}_{\mathsf{CFM}}(\theta) = \mathbb{E}_{t,(\boldsymbol{x}_0,\boldsymbol{x}_1)\sim B} \|\boldsymbol{v}_\theta(\boldsymbol{x}_t, t) - (\boldsymbol{x}_1 - \boldsymbol{x}_0)\|_2^2$
10:     $\theta \leftarrow \theta - \gamma \nabla_\theta \mathcal{L}_{\mathsf{CFM}}(\theta)$
11: end for
12: Output: Updated FM model $\boldsymbol{v}_\theta$.
```

Algorithm 6 Sampling for non-fixed condition in robot policy

```
1: Input: Conditional model $\phi$, FM model $\boldsymbol{v}_\theta$, condition $c_k$, number of sampling steps $T$.
2: Draw source sample $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_\phi(c_k), \boldsymbol{\sigma}_\phi^2(c_k)I)$
3: for $t = 0$ to $T - 1$ do
4:     $\boldsymbol{x} \leftarrow \boldsymbol{x} + \boldsymbol{v}_\theta(\boldsymbol{x}, \frac{t}{T}) \cdot \frac{1}{T}$
5: end for
6: Output: Generated target sample $\boldsymbol{x}$.
```

Table 6. Speed–quality tradeoff comparison on CIFAR-10. OT-CFM with batch size $b = 512$ and COT-FM with cluster size $n/K = 500$ yield comparable per-epoch OT computation time, but COT-FM achieves significantly better FID.

OT-CFM (Batch OT)			COT-FM (Ours)		
b	Time (sec.)	FID	n/K	Time (sec.)	FID
512	6.2	4.78	500 (50000/100)	6.3	**3.97**

Empirical Time Breakdown. We measure the wall-clock time of each preprocessing stage on CIFAR-10. As shown in Tab. 7, the one-time preprocessing takes 445 seconds in total, dominated by the reverse ODE computation. The per-epoch OT computation is approximately 15 seconds.

Table 7. Wall-clock time breakdown for one-time preprocessing on CIFAR-10.

One-time Preprocessing	Time Cost (sec.)
Reverse ODE	420
DINO feature extraction	15
K-means clustering	10

Overhead of Reverse ODE vs. Training Longer. On CIFAR-10, the reverse ODE preprocessing takes 420 seconds, equivalent to approximately 5 training epochs. Training RF for these additional epochs increases FID from 4.45 to 4.49 due to overfitting, while OT-CFM shows no improvement—FID remains at 4.78. In contrast, COT-FM achieves FID **3.97**, confirming that the preprocessing overhead is well justified by the quality gains.

C. Comparison with 2-Rectified Flow

2-Rectified Flow (2-RF) [32] is a standard rectification method that also involves a two-stage pipeline: (1) train 1-RF, then (2) generate synthetic couplings via forward ODE and retrain the model. We provide a direct comparison on CIFAR-10 to demonstrate that COT-FM is complementary to 2-RF.

As shown in Tab. 8, applying COT-FM on top of 2-RF yields additional gains: 2-RF achieves 12.21 FID at 1-step generation, while 2-RF + COT-FM achieves **10.91** FID. This demonstrates that the two approaches address different aspects of the problem—2-RF straightens trajectories via retraining on synthetic couplings, while COT-FM further reduces path crossings through cluster-wise optimal transport—and can be combined for cumulative improvement.

Table 8. 1-step FID (↓) comparison on CIFAR-10 between 2-Rectified Flow and 2-Rectified Flow enhanced with COT-FM.

Method	FID (↓)
2-Rectified Flow [32]	12.21
2-Rectified Flow [32] + COT-FM (Ours)	**10.91**

D. Additional Metrics and Ablation on K

We report FID, Inception Score (IS), Precision, and Recall on CIFAR-10 with 50 NFE steps in Tab. 9. COT-FM improves FID, IS, and Recall compared to Rectified Flow while maintaining the same Precision, indicating that COT-FM achieves broader coverage of the target distribution without sacrificing sample fidelity.

Table 9. Additional metrics on CIFAR-10 (50-step) with $K = 100$.

Method	FID ↓	IS ↑	Precision ↑	Recall ↑
Rectified Flow [32]	4.45	9.11	0.78	0.65
COT-FM (Ours)	**3.97**	**9.36**	0.78	**0.67**

We also ablate the number of clusters K in Tab. 10. The results show that COT-FM is robust to the choice of K: even with as few as $K = 5$ clusters, COT-FM (FID 4.56) outperforms OT-CFM (FID 4.78). Performance peaks around

$K = 100$ and slightly degrades at $K = 120$, suggesting that overly fine-grained clustering can reduce the number of samples per cluster and diminish the effectiveness of intra-cluster OT. In practice, we recommend applying the elbow method [55] to select K.

Table 10. Ablation on number of clusters K (CIFAR-10, 50-step FID $\downarrow$).

Method	$K = 5$	$K = 50$	$K = 100$	$K = 120$
COT-FM	4.56	4.10	**3.97**	4.06

E. Discussion on Clustering Quality

A natural concern is whether COT-FM is sensitive to the quality of the clustering. We argue that the method degrades gracefully and is robust in practice for the following reasons.

First, $K = 1$ reduces to the full-dataset setting where exact OT is intractable, necessitating batch OT as in OT-CFM. Any $K > 1$ improves over this baseline because cluster-wise OT still reduces path crossings compared to batch OT, even with imperfect cluster assignments. As demonstrated in the ablation study (Sec. D), even $K = 5$ outperforms OT-CFM.

Second, for conditional generation tasks such as robot manipulation, COT-FM does not require explicit clustering. Instead, the conditional model learns source distributions directly from task embeddings, as described in Sec. A.

Third, for unconditional generation, self-supervised features from DINO [6, 39] achieve 78.3% k-NN accuracy on ImageNet, demonstrating that semantically meaningful clusters naturally emerge and are well-separated in real-world data. This is precisely the property that COT-FM relies on: as long as the clustering captures broad semantic structure, the intra-cluster OT assignment will produce straighter transport paths than global batch OT.

F. Convergence of Alternating Optimization

Table 4(a) in the main paper shows that FID improves across iterations of the alternating optimization: 4.45 (0 iterations) $\rightarrow$ 4.23 (1 iteration) $\rightarrow$ 3.97 (2 iterations), with diminishing returns at iteration 3. Fig. 8 shows that the training loss decreases monotonically across iterations.

Intuitively, the convergence behavior is stable because the reverse ODE produces non-intersecting paths by construction, and the subsequent cluster-wise OT reduces local transport cost within each cluster. Each iteration of the alternating procedure therefore starts from a better-aligned coupling than the previous one, leading to monotonic improvement. A formal convergence analysis remains an interesting direction for future work.

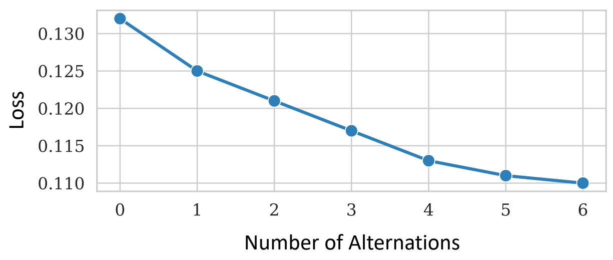

Figure 8. Training loss across alternating optimization iterations, showing monotonic decrease.

G. Related work

Diffusion Models and Flow Matching. Diffusion models [22, 25, 44, 50–52] have emerged as a powerful framework for generative modeling. These models learn a transformation that maps a prior distribution to a target distribution by progressively adding noise to the target distribution and then learning the reverse process. The reverse process is typically formulated as a stochastic differential equation (SDE) or an ordinary differential equation (ODE) [23, 52].

In contrast, Flow Matching (FM) [2, 30], learns a time-dependent velocity field that directly transports samples from the prior distribution to the target distribution. By modeling this velocity field, FM reduces the need to solve differential equations during training, providing a more direct and efficient way to learn the generative mapping.

Acceleration Strategies. Diffusion models and flow models sample slowly because of the recursive model evaluations required when solving the underlying ODE. To accelerate sampling while preserving generation quality, existing works can be broadly categorized into three directions: distilling knowledge from a pretrained model, learning a straighter velocity field, and directly learning a solution map of the ODE.

For distillation approaches [17, 34, 45, 47, 61, 62], the central idea is to distill the behavior of multi-step diffusion or score-based model into a few-step model. These methods supervise the student model using teacher trajectories, enabling efficient sampling without significantly sacrificing generation quality.

For learning straighter flows, a key challenge in training FM with a straight flow path is constructing a better coupling between the source and target distributions. The original FM formulation [30] uses random couplings for simplicity. However, random couplings induce high curvature of flow trajectories and degrade generation quality. Recent works [10, 26, 41, 49, 57] address this issue by improving the couplings using Optimal Transport (OT). Another line of work explores coupling with generated samples [32, 63].

They propose an iterative training framework that couples the source distribution with the corresponding target distribution predicted by the pre-trained model.

To directly learn the solution map of ODE [5, 15, 18, 24, 53], Consistency Model (CM) [53] learns a mapping from any point on a trajectory to its corresponding end point. Consistency Trajectory Model (CTM) [24] generalizes this idea by learning a mapping between any two points on the trajectory. Shortcut Model [15] learns discrete shortcut transitions along the trajectory, while MeanFlow [18] extends to a continuous formulation by learning the average velocity field between two points on the trajectory.

Optimized Noise Prior. Many works have adopted the strategy of optimizing the noise prior to improve the performance of diffusion or flow model. In the domain of image generation, prior works [36, 37, 46] demonstrate that a specific noise prior can induce specific pattern through diffusion model. Building on this idea, subsequent works [12, 19] enhance diffusion model performance by optimizing the initial noise using signals from reward models. Moreover, [7, 28] employ reinforcement learning (RL) and Direct Preference Optimization (DPO) [42] frameworks to modify the initial noise.

In the context of robot policy learning, several works [1, 59] show that a better prior can lead to improved performance or reduce the number of function evaluations (NFEs) required to achieve the same performance.

H. Experimental Configuration

H.1. Compute Environment

For the LIBERO robotics experiments, we use a single NVIDIA GeForce RTX 4090 GPU. For the CIFAR-10 and ImageNet experiments, we use the AMD AI & HPC Clusters Taiwan, equipped with AMD Instinct MI300X GPUs (192 GB HBM3) and AMD EPYC 9684X 96-Core Processors. GPUs within each node are interconnected via AMD Infinity Fabric (XGMI). CIFAR-10 experiments are conducted on a single MI300X GPU, while ImageNet experiments use 8 MI300X GPUs on a single node. All AMD experiments use ROCm 7.2.0.

H.2. 2D Point Cloud

We evaluate several flow-based generative models: Rectified Flow [32], OT-CFM [57], MeanFlow [18], and our COT-FM on three 2D benchmarks: Mixture of 5 Gaussians, Two Moons, and a Checkerboard Grid. The source distribution is a single Gaussian with mean $(0, 0)$ and standard deviation 0.6, while the target distribution follows the corresponding toy structure.

All methods use the same network architecture for fair comparison. We train each model using the Adam optimizer with learning rate 1e-3, batch size 512, and run 500 epochs for each task. In each iteration, source–target mini-batches are sampled, and the corresponding coupling strategy is applied. For evaluation, we measure: (1) Wasserstein distance between generated and target samples using exact 2-Wasserstein distance; (2) trajectory curvature, computed by discretizing the integrated trajectories, normalizing tangents, and measuring second-order directional change. Lower curvature corresponds to straighter probability paths.

H.3. CIFAR-10

We experiment with unconditional generation on CIFAR-10 [27], where the input to the model is $32 \times 32 \times 3$ in pixel space. We evaluate Fréchet Inception Distance (FID) [21] using 50K generated images. Optimal Transport is computed using the publicly released implementation from [57].

To ensure a fair comparison with Rectified Flow [32], our implementation follows their setting. We train the models for 600 epochs with a batch size of 128 using Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1 \times 10^{-8}$, and no weight decay. The learning rate is set to 2×10^{-4} and we apply gradient clipping with a maximum norm of 1.0. The backbone is an NCSN++-style UNet, and detailed parameters are provided in Table 11. We maintain exponential moving average (EMA) weights with decay rate of 0.999999.

For comparison with MeanFlow [18], we adopt their experimental configuration. We train the models for 240 epochs with a batch size of 512. We used Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and no weight decay. The learning rate is set to 0.0001. The UNet architecture follows [51], with model parameters summarized in Table 11. We maintain EMA weights with a decay rate of 0.99999.

H.4. ImageNet

We experiment with conditional generation on ImageNet [11] 256×256 with full dataset. The input to the model is $32 \times 32 \times 4$ in latent space obtained from the VAE. We evaluate Fréchet Inception Distance (FID) [21] using 50K generated images. Optimal Transport is computed using the publicly released implementation from [57].

Our implementation is slightly modified from the setting of a non-official PyTorch implementation [64] to 1-RF. The images are augmented through horizontal flipping. Both 1-RF and COT-FM are trained for 190 epochs in total; COT-FM is fine-tuned from a pre-trained RF model after 160 epochs, requiring only 30 additional epochs of training. We use a batch size of 256 with the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 1 \times 10^{-8}$ and no weight decay. We set the learning rate to 1×10^{-4} and apply gradient clipping with a maximum norm of 1.0. The backbone is Scalable Interpolant Transformers(SiT) [35]. We maintain EMA weights with a decay rate of 0.9999. We set ω to 2.5 and 2 for SiT-B/4 and SiT-B/2, respectively. All other hyperparameters remain identical to the original configuration.

H.5. LIBERO

The LIBERO benchmark [31] uses a Franka Emika Panda robot with end-effector control and camera control. We evaluate on Spatial for spatial relationships and Long for long horizon tasks. Each task starts with 50 different initial points and we record the average success rate for each task with 3 different seeds. The hyperparameters are the same as FLOWER [43], shown in Table 13.

We train the reinforcement learning model over 15 passes of the offline dataset. After collecting 51,200 samples in each pass, we perform Proximal Policy Optimization (PPO) [48] training for the conditional model, where condition c_k contains language instructions and the current visual observations. During PPO training, the discount factor γ is set to 0.99, the clipping parameter ϵ to 0.2, and the learning rate to 1×10^{-5}. Each PPO training session runs for 10 epochs.

For finetuning FLOWER, we follow the setting reported in FLOWER, as summarized in Table 12. To ensure training stability, we only finetune the Action Encoders, Action Heads, and Flow Transformer.

I. Additional Generated Results

We present additional generated images for CIFAR-10 in this section. We use the same noise for those sampled from $\mathcal{N}(0, I)$ and same noises for those sampled from our method for consistency.

Component	Rectified Flow	MeanFlow
Channels	128	128
Channel multiple	(1, 2, 2, 2)	(2, 2, 2)
Residual blocks per resolution	4	4
Normalization	GroupNorm	GroupNorm
Nonlinearity	Swish	Swish
Attention resolution	16	16
Dropout	0.15	0.2
Embedding type	Fourier	Positional

Table 11. UNet architecture for CIFAR-10 experiments.

Components	Number of Parameters
ViT	360M
VLM	205M
Action Encoders	3.2M
Action Heads	31.8K
Global-AdaLN	28.3M
Cond Linear Proj.	1.0M
Timestep Embedder	1.3M
Cond Norm	1.0K
FreqEmbedder	1.3M
Action Space Embedder	1.3M
Flow Transformer	339M
ϕ (ours)	17M
Total Parameters FLOWER	**964M**

Table 12. The parameters of all model components in FLOWER.

Settings	LIBERO
Action Space Encoders	2-layered MLP
Action Space Decoders	Linear Attention
Number of Flow-T Layers	18
Latent Dimension	1024
Number of Heads	16
Position Embedding	1D Rope
Sampling Distribution	Uniform
Attention Dropout	0.1
MLP Dropout	0.1
Residual Dropout	0.1
Act Seq Length	10
Denoising Steps	4
Multistep	4
Camera Views	[Primary Static, Wrist]
Use Proprio	False
Action Space	Delta EEF
Frequency	10

Table 13. LIBERO hyperparameters of FLOWER.

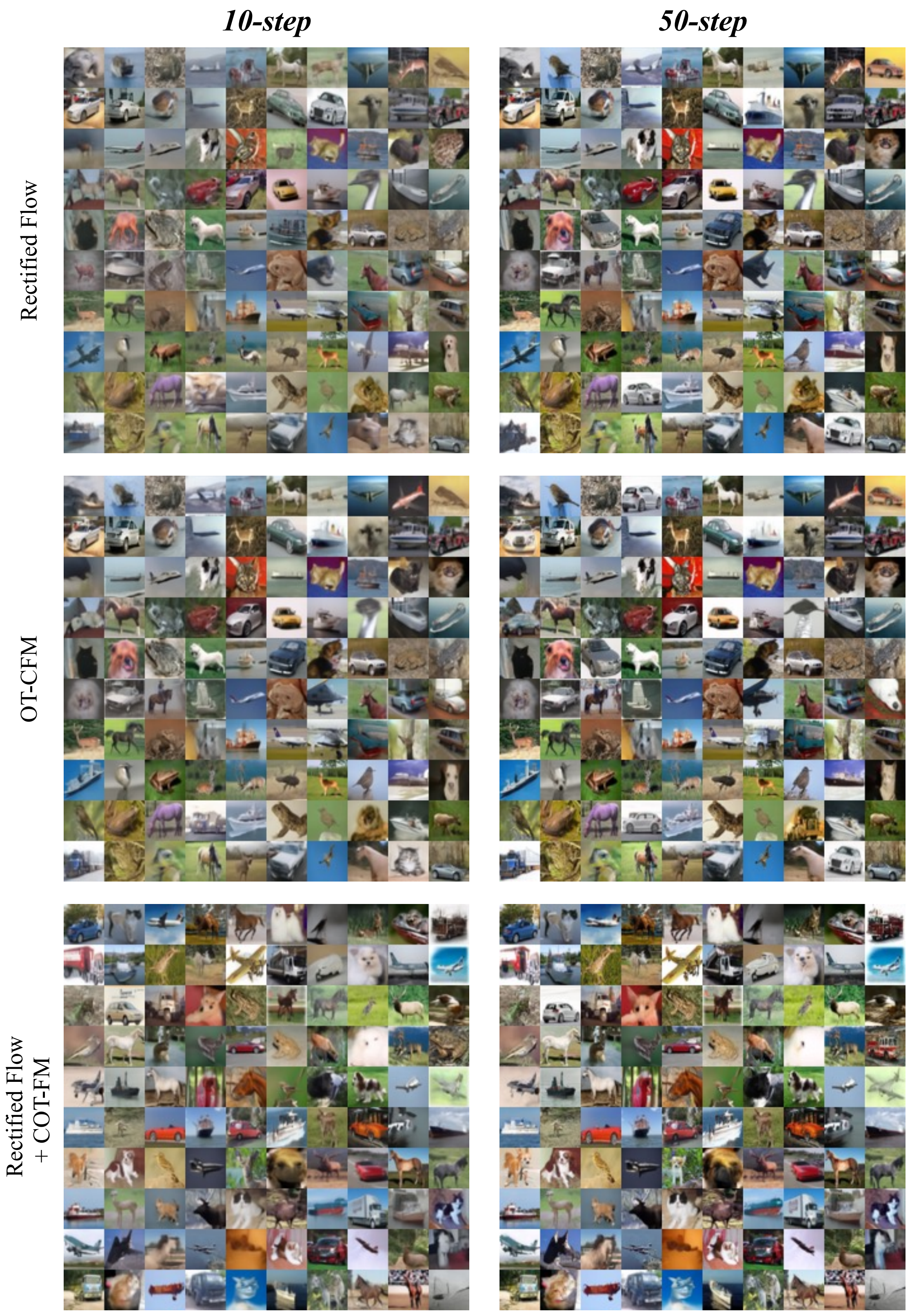

Figure 9. Visualization of CIFAR-10 for different flow-based methods under 10-step and 50-step generation settings.

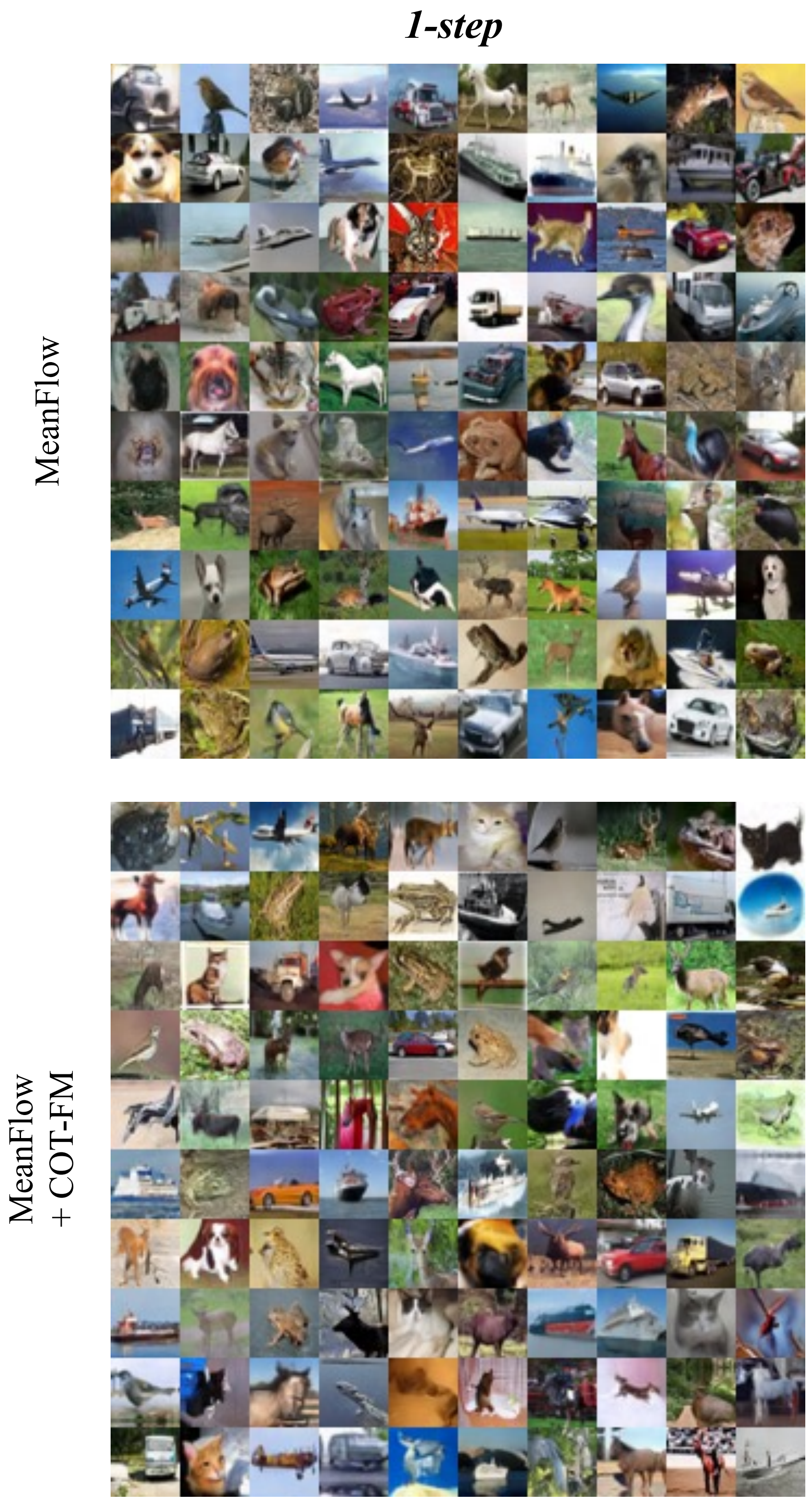

Figure 10. Visualization of CIFAR-10 for MeanFlow under 1-step generation settings.