Generative Modeling with Orbit-Space Particle Flow Matching

SINAN WANG*, Georgia Institute of Technology, USA
JINJIN HE*, Georgia Institute of Technology, USA
SHENYIFAN LU, Georgia Institute of Technology, USA
RUICHENG WANG, Georgia Institute of Technology, USA
GREG TURK, Georgia Institute of Technology, USA
BO ZHU, Georgia Institute of Technology, USA

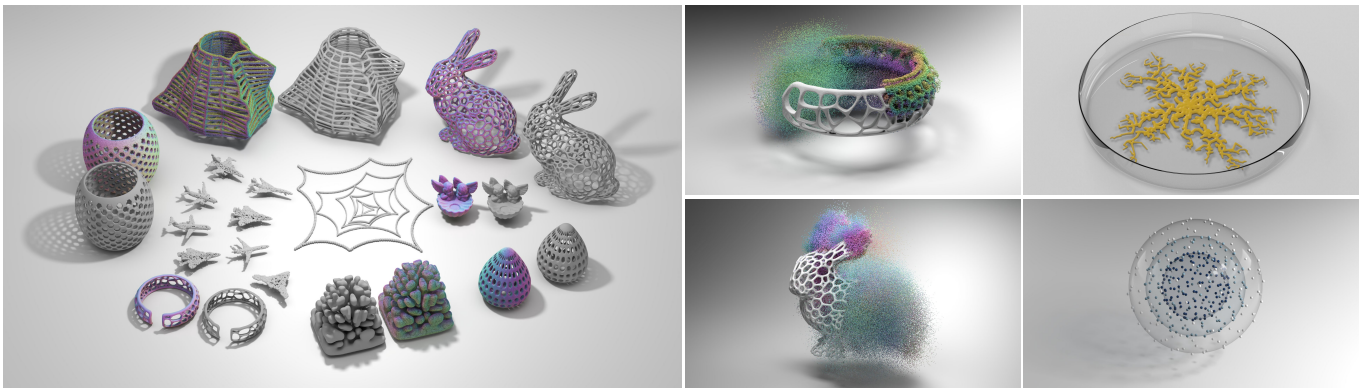
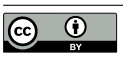


Fig. 1. *Left:* ShapeNet point cloud generation, single-shape encoding on complex Thingi10k meshes with Poisson-reconstructed surfaces, and minimal surface generation. *Middle:* Generation process visualization showing geometric probability paths transporting noise to surface points with encoded normals. *Right:* Energy-driven particle generation: diffusion-limited aggregation (top) and multilayer Thomson problem with electrons on concentric shells (bottom).

We present **Orbit-Space Geometric Probability Paths (OGPP)**, a particle-native flow-matching framework for generative modeling of particle systems. OGPP is motivated by two insights: (i) particles are defined up to *permutation symmetries*, so anonymous indexing inflates per-index target variance and yields curved, hard-to-learn flows; (ii) particles live in physical space, so the flow's *terminal velocity* has physical meaning and can encode geometric attributes (e.g., surface normals). OGPP instantiates three key components: (1) orbit-space canonicalization of the probability-path terminal endpoint, (2) particle index embeddings for role specialization, and (3) geometric probability paths with arc-length-aware terminal velocities that generate normals as a byproduct of the flow. We evaluate OGPP on minimal-surface benchmarks, where it reduces metric error by up to two orders of magnitude in a single inference step; on ShapeNet, where it matches the state-of-the-art with 5× fewer steps and reaches airplane EMD comparable to DiT-3D with 26× fewer parameters and 5× fewer steps; and on single-shape encoding, where it produces normals and reconstructions competitive with 6D generators while operating entirely in 3D.

*Both authors contributed equally to this research.

Authors' Contact Information: Sinan Wang, swang3081@gatech.edu, Georgia Institute of Technology, USA; Jinjin He, jhe433@gatech.edu, Georgia Institute of Technology, USA; Shenyifan Lu, slu361@gatech.edu, Georgia Institute of Technology, USA; Ruicheng Wang, wrc0326@outlook.com, Georgia Institute of Technology, USA; Greg Turk, turk@cc.gatech.edu, Georgia Institute of Technology, USA; Bo Zhu, bo.zhu@gatech.edu, Georgia Institute of Technology, USA.



ACM 1557-7368/2026/7-ART117
https://doi.org/10.1145/3811342

CCS Concepts: • **Computing methodologies** → **Point-based models**.

Additional Key Words and Phrases: Generative Modeling, Flow Matching, Particle Systems

ACM Reference Format:
Sinan Wang, Jinjin He, Shenyifan Lu, Ruicheng Wang, Greg Turk, and Bo Zhu. 2026. Generative Modeling with Orbit-Space Particle Flow Matching. *ACM Trans. Graph.* 45, 4, Article 117 (July 2026), 27 pages. https://doi.org/10.1145/3811342

1 Introduction

Particles constitute a central representation in computer graphics, where sampling, geometry, appearance, and physics are often modeled as structured sets of particles embedded in 2D or 3D physical space. Such particle-based representations arise across graphics pipelines for different purposes, from Poisson sampling for ray tracing [Ahmed and Wonka 2020, 2021], to point-set surfaces and clouds for geometric modeling [Alexa et al. 2001; Guennebaud and Gross 2007; Kerbl et al. 2023; Peng et al. 2021], to Lagrangian particles for solid and fluid simulations [Müller et al. 2003, 2007; Stomakhin et al. 2013; Zhou et al. 2024a], and to agent-based animation such as crowd and flock simulation [Guy et al. 2010; Narain et al. 2009; Reynolds 1987; Thalmann and Musse 2012]. Therefore, a generative model natively defined on particles and leveraging their connectivity-free structure and physical-space dynamics is well-motivated for graphics generation tasks.

However, modern generative models are built on grids rather than particles (e.g., diffusion [Blattmann et al. 2023; Ho et al. 2020;

Rombach et al. 2022; Song et al. 2020] and flow matching [Lipman et al. 2022]). In these settings, the representation lives on a fixed grid (e.g., a 2D lattice of pixels), and generation amounts to mapping noise to data distributions on the grid. Despite their successes in generating images or videos, these models do not transfer efficiently to particle generation because they ignore two fundamental differences. *First*, **particles exhibit pronounced *symmetries***: permuting particle indices leaves the underlying configuration unchanged, yet can arbitrarily alter its vectorized representation in high-dimensional space. In group-theoretic terms, the collection of all such symmetry-related configurations forms the *orbit* of a particle state. As a result, naively applying grid-based generative frameworks (e.g., flow matching [Lipman et al. 2022]) to particle data by flattening particles into long vectors leads to fundamental difficulties: For images, a pixel at a fixed coordinate exhibits consistent statistics across samples. In contrast, particle systems are defined only up to permutation symmetry: a particle at a fixed index does not correspond to any consistent spatial or statistical role across the dataset. Consequently, probability-path endpoints associated with a given index are dispersed throughout space, forcing velocity predictors to average over incompatible targets during training and yielding noisy, poorly structured per-particle flows. State-of-the-art particle generators such as Equivariant Flow Matching [Klein et al. 2023; Song et al. 2023] mitigate permutation ambiguity via optimal transport couplings. However, these methods incur high computational cost and still operate on flattened, anonymous particle representations, so individual indices must aggregate over many symmetry-induced roles, leading to increased target variance and highly curved flows. *Second*, **particles live in *physical space***. Generating a set of particles can be viewed as simulating their spatiotemporal evolution under a learned physical velocity field. This differs fundamentally from image generation, where the velocity field in flow matching merely transports pixel values and carries no intrinsic physical meaning. In particle-based settings, however, the velocity field is defined in physical space, and the terminal velocity at $t = 1$ represents a well-defined geometric quantity. For example, when particles sample a surface, this terminal velocity can encode meaningful local geometric information, such as surface normals or orientation. Standard linear paths place particles at the correct locations but do not exploit this geometric degree of freedom.

Motivated by these two insights, we propose a generative framework for particles that both respects orbit structure and exploits geometric path. Our key idea is to untangle mixed particle roles by combining orbit-space canonicalization with identity-aware particle index embeddings. On top of this, we design geometric probability paths whose terminal tangents encode per-particle normals, so a single flow jointly generates particle positions and attributes. Our framework consists of three key components (See Figure 2): (i) *orbit-space canonicalization*, (ii) *particle index embeddings*, and (iii) *geometric probability paths*. All three are expressed as choices of *conditional probability path*, which we collectively call ***Orbit-Space Geometric Probability Paths (OGPP)***.

For *orbit-space canonicalization*, we perform symmetry reduction at the terminal endpoint X_1: for each particle configuration, we sort indices according to a geometric criterion (e.g., a space-filling curve) and select a single representative from the orbit. This enforces that

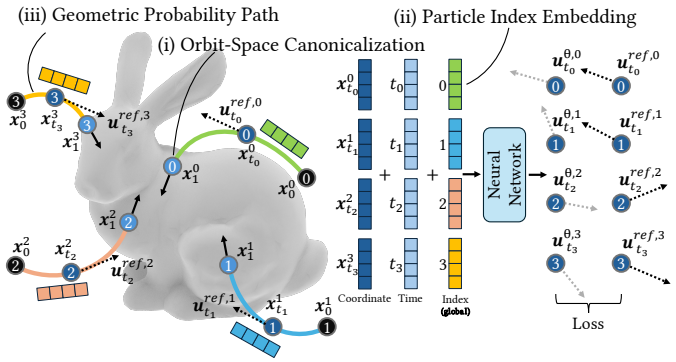


Fig. 2. **OGPP.** Our framework integrates three key components: (i) *orbit-space canonicalization* assigns canonical indices (0,1,2,3) to X_1 while keeping X_0 uncanonicalized, (ii) *particle index embeddings* (colored blocks) allow each index to specialize to its canonical role, and (iii) *geometric probability paths* encode surface normals via arc-length-aware terminal velocities. Per-particle coordinates $\boldsymbol{x}_{t_i}^i$ and learnable per-index embeddings are fed into a NN, predicting velocities $\boldsymbol{u}_{t_i}^{\theta,i}$ supervised by reference velocities $\boldsymbol{u}_{t_i}^{\text{ref},i}$.

particle index i consistently lands in a localized and stable spatial region, reducing variability in the training targets seen by each index. Next, for *particle index embeddings*, we attach a learnable identity embedding to each particle index and provide it to the velocity network. This allows the model to condition on particle identity, enabling different indices to specialize to distinct velocity-field roles, analogous to class-conditional generation. Together, canonicalization and identity embeddings convert noisy mixtures of regression targets into well-separated, easier-to-learn trajectory families, yielding straighter flows. Finally, for *geometric probability paths*, we replace linear interpolation with geometry-aware paths that exploit the structure of particle systems. Specifically, we construct Hermite-type probability paths whose terminal tangents align with per-particle normals: the endpoint specifies particle position, while the terminal velocity encodes local surface orientation. As a result, the learned flow simultaneously transports particles from noise to data and produces accurate surface normals as an intrinsic byproduct.

We evaluate **OGPP** on a range of graphics-oriented generative tasks, including geometric reconstruction, shape generation, and physics simulation. For minimal-surface generation, OGPP reduces metric error by up to two orders of magnitude in a single inference step. On ShapeNet, it improves 1-NNA, matches the particle-generator SOTA NSOT [Hui et al. 2025] with **5**× fewer inference steps, reaching airplane EMD comparable to DiT-3D [Mo et al. 2023] using **26**× fewer parameters and **5**× fewer steps. On single-shape encoding benchmarks [Zhang et al. 2025], it yields better normal estimation and reconstruction quality than generalized VP-based paths [Albergo and Vanden-Eijnden 2022; Chang et al. 2024; Ma et al. 2024], while remaining comparable to state-of-the-art 6D generators.

Contributions. Our main contributions are:

(1) **Orbit-space particle flow matching.** We introduce *particle flow matching* as a Lagrangian formulation of flow matching

for particle systems, in contrast to the Eulerian image models, combining identity embeddings on individual particles with an orbit-space canonicalization so that each particle can learn its own consistent velocity field while simplifying the learning.

(2) **Geometric probability paths.** We construct Hermite-type geometric probability paths whose terminal tangent encodes per-particle attributes such as surface normals, enabling surface normal generation as a byproduct of the learned flow.

(3) **Energy-driven evaluation for particle generators.** We propose self-referential, physics- and geometry-based metrics that directly assess the quality of generated particle sets (e.g., blue-noise spectra, fractal dimension, residual Coulomb forces, minimal-surface deviation), together with matching benchmark datasets.

Paper outline. Section 2 reviews related work, and Section 3 recalls the background on flow matching and group theories. Sections 4 and 5 present our two main theoretical contributions, orbit-space probability paths and geometric probability paths for attribute encoding, respectively, and are *essential* reading. Section 6 summarizes the overall algorithm. Section 7 reports experimental results, before Section 8 offers further discussion and Section 9 concludes.

2 Related Work

2.1 Generative Models

Continuous-Time Generative Models. From a modern continuous-time perspective [Holderrieth and Erives 2025; Lipman et al. 2024], generative models can be formulated as stochastic differential equations (SDEs), as in denoising diffusion models [Ho et al. 2020; Song et al. 2020], or ordinary differential equations (ODEs), as in flow models [Albergo et al. 2023; Chen et al. 2018; Grathwohl et al. 2018; Lipman et al. 2022; Liu et al. 2022]. This unified view has enabled broad progress across image and video synthesis [Blattmann et al. 2023; Brooks et al. 2024; Ramesh et al. 2022; Rombach et al. 2022], 3D shape generation [Hui et al. 2022; Mo et al. 2023; Zeng et al. 2022; Zhou et al. 2021], point cloud modeling [Luo and Hu 2021; Yang et al. 2019], and neural rendering [Poole et al. 2022; Wang et al. 2023]. Recently, IADB [Heitz et al. 2023] reinterprets DDIM as an ODE-based deterministic diffusion process. Within the ODE family, unlike CNFs [Chen et al. 2018; Grathwohl et al. 2018], flow matching [Lipman et al. 2022] enables simulation-free vector field learning and the use of Optimal Transport (OT) [Lipman et al. 2022] paths. Techniques like Minibatch OT [Pooladian et al. 2023; Tong et al. 2023] and Rectified Flow [Liu et al. 2022] further straighten trajectories for efficiency. However, these models are primarily tailored for objects in Euclidean space $\mathbb{R}^d$ (e.g., images), and do not naturally accommodate the unique quotient geometry of particle systems. While Riemannian Flow Matching [Chen and Lipman 2023] extends flow matching to non-Euclidean manifolds and SE(3) flow matching [Yim et al. 2023] applies it to structured proteins, both remain Eulerian and do not treat particles individually.

Alternative Probability Path Designs. Beyond linear interpolation, recent works explore alternative path designs. BNDM [Huang et al. 2024], motivated by the spectral bias of diffusion models, injects time-dependent blue noise into deterministic diffusion to modify the probability path. Generalized VP interpolants [Albergo and Vanden-Eijnden 2022; Ma et al. 2024], building on VP and VE SDEs [Song et al. 2020], enable flexible nonlinear schedules in flow matching. Recent 3D shape tokenization work [Chang et al. 2024] adopts gVP paths for latent flow matching and zero-shot normal estimation.

2.2 Point Cloud Generation

Point Cloud Generative Models. Early point cloud generation relied on GANs [Achlioptas et al. 2018; Li et al. 2021; Shu et al. 2019; Xie et al. 2021] and set-structured VAEs such as SetVAE [Kim et al. 2021], alongside CNF-based models like PointFlow [Yang et al. 2019] and SoftFlow [Kim et al. 2020], which offer exact likelihoods. To achieve scalable high-fidelity synthesis, recent works adopt a two-stage strategy: compressing high dimensional vectors into a compact latent space via VAEs [Kingma and Welling 2013] before training generative models. Early latent representations used voxel grids (e.g., ConvOccNet [Peng et al. 2020]), suffering from cubic memory costs, while later works explored more efficient structures such as irregular grids [Zhang et al. 2022], hierarchical point-based latents (LION [Zeng et al. 2022]), or latent sets without explicit spatial structure (3DShape2VecSet [Zhang et al. 2023]). Despite improved scalability, these frameworks typically rely on category-specific autoencoders. In contrast, generative models like PVD [Zhou et al. 2021], DiT-3D [Mo et al. 2023], DPM [Luo and Hu 2021] apply diffusion directly in data space. PSF [Wu et al. 2023] accelerates sampling via Reflow [Liu et al. 2022]. While effective, these works primarily advance model architectures or data representations and do not explicitly model orbit-space symmetries, retaining an Eulerian viewpoint.

Canonicalization for Permutation Handling. Permutation ambiguity in particle systems is commonly addressed via canonicalization by deterministic ordering, such as Z-order (Morton order) [Morton 1966] or Hilbert curves [Hilbert [n. d.]], which map spatial coordinates to one-dimensional sequences while preserving locality. Recent Transformer-based models adopt similar strategies to stabilize attention and improve scalability, e.g., Point Transformer v3 [Wu et al. 2024], OctFormer [Wang 2023], and FlatFormer [Liu et al. 2023]. These methods canonicalize the *Transformer input representation*, primarily to improve computational efficiency and architectural stability.

Symmetry Modeling. The recent frontier focuses on enforcing symmetries. Equivariant Flow Matching [Klein et al. 2023; Song et al. 2023] achieves this via optimal transport (OT) couplings but with a training-step complexity of $O(B^2N^3)$ [Hui et al. 2025], making it unscalable. NSOT [Hui et al. 2025] improves scalability by offline OT precomputation and hybrid coupling, and SGFM [Puny et al. 2025] extends such constraints to enforce complex space-group symmetries inherent to crystalline structures. From an architectural perspective, these permutation-equivariant models [Hui et al. 2025; Klein et al. 2023; Song et al. 2023], and more broadly, mainstream point-cloud architectures [Liu et al. 2019; Zhou et al. 2021] treat particles as anonymous coordinates, so the network is not allowed to distinguish particle indices. Diffusion Transformers such as DiT-3D [Mo et al. 2023] do employ learned positional embeddings, but

operates on voxel grids without orbit-space canonicalization. Consequently, these methods still adopt an Eulerian view, which makes the regression problem ill-conditioned and the flow highly curved.

2.3 Energy-Driven Particle Systems

Physical Particle Systems. Particle systems are a ubiquitous representation across physics, graphics, and vision, used to model phenomena ranging from N-body simulations [Barnes and Hut 1986], to molecular dynamics [Frenkel and Smit 2023], fluids via SPH [Müller et al. 2003] and vortex particles [Park and Kim 2005], and flocking or crowd behavior [Reynolds 1987; Thalmann and Musse 2012]. A fundamental subclass involves systems governed by energy functionals, where equilibrium states correspond to stationary points of pairwise or global potentials: blue-noise sampling seeks point sets with suppressed low-frequency spectra and isotropy [Cook 1986; Ulichney 1988; Yellott Jr 1983], computed via Lloyd relaxation [Lloyd 1982], capacity-constrained Voronoi tessellations [Balzer et al. 2009], optimal transport [De Goes et al. 2012; Qin et al. 2017], or kernel-based methods [Ahmed et al. 2022; Fattal 2011]; the Thomson problem [Bowick and Giomi 2009; Smale 1998; Thomson 1904] seeks minimum-energy configurations of repelling charges, tackled via basin-hopping [Wales and Doye 1997], genetic algorithms [Morris et al. 1996], or simulated annealing [Erber and Hockney 1991]; diffusion-limited aggregation [Witten Jr and Sander 1981] produces fractal clusters through Brownian-motion attachment [Halsey 2000; Meakin 1983b; s Vicsek 1992]; and minimal surfaces [Plateau 1873] minimize area under boundary constraints via variational methods [Brakke 1992; Dziuk 1990; Pinkall and Polthier 1993; Wang and Chern 2021]. Recently Geometry Distributions [Tang et al. 2025a,b; Zhang et al. 2025] represent single surfaces as infinite point distribution via diffusion models.

Physics-Aware Evaluation for Generative Models. Despite their scientific importance, these physically grounded particle systems have received limited attention from the generative modeling community, which prioritizes shape-level point cloud generation evaluated by distribution-matching metrics such as 1-NNA [Lopez-Paz and Oquab 2016]. We observe that energy-driven particle systems offer intrinsic evaluation criteria (e.g., spectral characteristics, fractal dimensions, residual forces, surface deviations) that can complement distributional metrics by more directly measuring physical fidelity. To this end, for energy-driven tasks such as blue noise, minimal surfaces, DLA, and the Thomson problem, we first use classical solvers to produce large datasets of equilibrium configurations, and then train a generative model on these datasets.

3 Background

3.1 Naming Conventions

We adopt the following conventions throughout this paper. Bold symbols (e.g., $\boldsymbol{u}$, $\boldsymbol{x}$, $\boldsymbol{n}$) denote vector fields or vectors, while regular symbols denote scalars. Capital letters (e.g., X, N) represent random variables, and lowercase bold letters (e.g., $\boldsymbol{x}$, $\boldsymbol{n}$) denote their realizations or fixed values. Specifically, X denotes position random variables, N denotes attribute random variables, and $Z = (X_1, N_1)$ denotes the joint random variable of position and attribute. Superscripts without parentheses (e.g., $\boldsymbol{x}_t^i$) denote particle indices, while superscripts in parentheses (e.g., $\boldsymbol{x}_0^{(i)}$) denote sample indices. We summarize the main symbols and notations in Table 1.

Table 1. Summary of the main symbols and notations.

Notation	Type	Definition
		General
t	scalar	time $\in [0,1]$
d, D	scalar	dimension
N	scalar	number of particles
		Flow Matching
$\boldsymbol{u}_t$	vector field	velocity field at time t
$\boldsymbol{u}_t^\theta$	vector field	neural network velocity field
$\boldsymbol{u}_t^{\text{ref}}$	vector field	reference (target) velocity field
X_0	random var.	initial point (noise)
X_1	random var.	terminal point (data)
X_t	random var.	interpolated point at time t
$\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_t$	vector	realizations of X_0, X_1, X_t
p_{init}	distribution	initial (noise) distribution
p_{data}	distribution	data distribution
p_t	distribution	marginal probability path
$p_t(\cdot\|\boldsymbol{x}_1)$	distribution	conditional probability path
Z	random var.	joint variable: (X_1, N_1)
		Canonicalization
$C(\cdot)$	map	orbit-space canonicalization map
G	group	symmetry group (e.g., permutation)
$\rho(g)$	matrix	orthogonal representation of g
$\text{Orb}(\boldsymbol{x})$	set	orbit of $\boldsymbol{x}$ under the group action
$\zeta_{\boldsymbol{x}}$	random var.	canonical representative of $\text{Orb}(X_1)$
		Geometric Path
$\boldsymbol{n}$	vector	per-particle attribute (surface normal)
$\boldsymbol{v}_1$	vector	terminal tangent velocity
$\alpha(t), \beta(t)$	scalar	Hermite basis functions
$\gamma(t)$	curve	conditional probability path curve

3.2 Flow Matching

Flow matching [Lipman et al. 2022, 2024] trains a velocity field that transports a noise distribution p_{init} to p_{data} by integrating an ODE. A flow model generates samples by solving

$$\frac{\mathrm{d}X_t}{\mathrm{d}t} = \boldsymbol{u}_t^\theta(X_t), \quad X_0 \sim p_{\text{init}}, \tag{1}$$

where $\boldsymbol{u}_t^\theta : \mathbb{R}^d \times [0,1] \to \mathbb{R}^d$ is a neural network chosen so that $X_1 \sim p_{\text{data}}$. For each data point $\boldsymbol{x}_1 \sim p_{\text{data}}$, a *conditional probability path* $p_t(\cdot \mid \boldsymbol{x}_1)$ interpolates from p_{init} at t=0 to a point mass at $\boldsymbol{x}_1$ at t=1; averaging over $\boldsymbol{x}_1$ yields the *marginal probability path* p_t. Since most of our constructions are defined at the conditional level, we refer to $p_t(\cdot \mid \boldsymbol{x}_1)$ simply as a *probability path* and reserve *marginal probability path* for p_t.

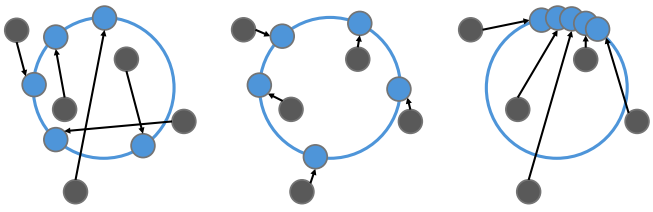


Fig. 3. Conceptual illustration of the conditional distribution of the terminal endpoint X_1^0 for a fixed particle index 0 under different coupling strategies. Each figure shows five sampled $(\boldsymbol{x}_0, \boldsymbol{x}_1)$ pairs for particles with index 0: gray points denote the noise positions $\boldsymbol{x}_0^{(i),0}$, blue points denote the corresponding targets $\boldsymbol{x}_1^{(i),0}$ on the surface, and arrows indicate their displacements. Left: independent coupling [Lipman et al. 2022]; middle: OT-based coupling [Klein et al. 2023; Song et al. 2023]; right: our orbit-space canonicalization. Independent and OT-based couplings spread the possible endpoints X_1^0 around the surface, yielding a broad conditional distribution $p(X_1^0 \mid X_t = \boldsymbol{x}, I = 0)$, whereas orbit-space canonicalization concentrates them into a smaller region, which is expected to reduce the conditional covariance and simplify the velocity regression.

Marginalization trick. The marginal velocity field can be expressed as a posterior-weighted average of conditional velocities (see Appendix A for details):

$$\boldsymbol{u}_t^{\mathrm{ref}}(\boldsymbol{x}) = \int \boldsymbol{u}_t^{\mathrm{ref}}(\boldsymbol{x} \mid \boldsymbol{x}_1)\, \frac{p_t(\boldsymbol{x} \mid \boldsymbol{x}_1)\, p_{\mathrm{data}}(\boldsymbol{x}_1)}{p_t(\boldsymbol{x})}\, \mathrm{d}\boldsymbol{x}_1. \tag{2}$$

In practice, the network is trained via the *conditional flow matching loss*:

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t,\, \boldsymbol{x}_1,\, \boldsymbol{x} \sim p_t(\cdot \mid \boldsymbol{x}_1)} \left[\left\| \boldsymbol{u}_t^{\theta}(\boldsymbol{x}) - \boldsymbol{u}_t^{\mathrm{ref}}(\boldsymbol{x} \mid \boldsymbol{x}_1) \right\|^2 \right]. \tag{3}$$

3.3 Group Theory

We briefly review concepts needed for orbit-space probability paths; extended definitions are in Appendix B. A *group* G acts on $\mathbb{R}^d$ via an orthogonal representation $\rho : G \to O(d)$, i.e., $g \cdot x = \rho(g)x$. The *orbit* of x is $\mathrm{Orb}(x) := \{\rho(g)x : g \in G\}$.

In all our experiments we normalize away global pose by recentering and PCA alignment, so the remaining symmetry is $G = S_N$ acting on $(\mathbb{R}^D)^N$ by permuting particle indices.

Canonicalization. A *canonicalization map* $C : \mathbb{R}^d \to \mathbb{R}^d$ selects a representative from each orbit in a G-invariant way, requiring: (1) $C(\rho(g)x) = C(x)$ for all $g \in G$ (G-invariance), and (2) $C(x) \in \mathrm{Orb}(x)$ (the output lies in the orbit of x). Together, these conditions imply C induces a bijection between orbits and their canonical representatives. Concretely, a G-canonicalization fixes a deterministic particle ordering (e.g., via a space-filling curve such as Morton or Hilbert).

4 Orbit-Space Probability Paths

In this section, we focus on the first two key components of OGPP, **orbit-space canonicalization** and **particle index embeddings**. These two mechanisms are designed to work in tandem: our ablation in Sec. 7.3 shows that each alone brings only limited gains, while their combination is crucial. We discuss our third key component, **geometric probability paths**, in Sec. 5.

In a traditional flow-matching framework, the neural network takes as input the particle positions $\boldsymbol{x}_t$ at time t (together with t itself) and outputs a velocity field $\boldsymbol{u}_\theta(\boldsymbol{x}_t, t)$. Both the intermediate states X_t and the reference targets Y are determined by the choice of conditional probability path $p_t(\cdot|\boldsymbol{x}_1)$. We discuss the interior geometry of $p_t(\cdot|\boldsymbol{x}_1)$ in Section 5 and focus on its endpoints in this section. As outlined in the introduction, we pursue two objectives: (i) *make the regression task easier* by reducing the conditional covariance; and (ii) *encourage straight flows* by reducing Lipschitz ratios. We begin by describing our model architecture with particle index embedding in Section 4.1. We then show in Section 4.2 that orbit-space canonicalization on X_1 reduces the conditional covariance of the regression targets. In Section 4.3 we formalize a requirement on orbit-space canonicalization maps: they must be orbit-continuous, thereby encouraging straighter flows. Finally, in Section 4.4 we study nearest-neighbor Lipschitz ratios and show that further canonicalizing the noise endpoint X_0 inflates these ratios, leading to less straight flows.

4.1 Model Architecture with Particle Index Embeddings

We instantiate the particle-indexed velocity field $\boldsymbol{u}_\theta$ with a plain Transformer encoder [Vaswani et al. 2017] that operates on sets of particles. Given an input configuration $\boldsymbol{x}_t = (\boldsymbol{x}_t^1, \ldots, \boldsymbol{x}_t^N)$, each particle is represented by a feature vector in $\mathbb{R}^{D_{\mathrm{in}}}$, consisting primarily of spatial coordinates and, in a few experiments, an additional time coordinate (see Section 7.1.3). We first apply a linear projection to an embedding dimension D_{emb}, add a *particle index embedding* $e_i \in \mathbb{R}^{D_{\mathrm{emb}}}$, and add a global time embedding $\phi_t(t) \in \mathbb{R}^{D_{\mathrm{emb}}}$: $\boldsymbol{h}_i^{(0)} = W_{\mathrm{in}}\boldsymbol{x}_t^i + e_i + \phi_t(t)$, with $i = 1, \ldots, N$. The sequence $(\boldsymbol{h}_1^{(0)}, \ldots, \boldsymbol{h}_N^{(0)})$ is then processed by a L-layer Transformer encoder with multi-head self-attention and GELU-activated MLPs, yielding representations $\boldsymbol{h}_i^{(L)}$. The final velocity prediction for particle i is obtained by a shared linear head $\boldsymbol{u}_\theta^i(X_t, t) = W_{\mathrm{out}}\boldsymbol{h}_i^{(L)}$.

Architecturally, the particle index embedding e_i is deliberately simple: it plays the same role as a standard positional embedding in Transformers, i.e., a learned token-wise bias that lets the network distinguish different positions. We refer to it as a *particle index embedding* to emphasize that the "position" here is the canonical particle index rather than a grid coordinate.

This architecture is intentionally plain; we attribute the observed improvements primarily to the probability path design and the use of identity embeddings rather than to architectural sophistication.

For conditional generation tasks such as minimal surface generation with anchor points, we extend this architecture with cross-attention layers interleaved every few self-attention blocks. Condition tokens are projected to the embedding dimension and serve as keys and values, with particle tokens as queries. To handle variable numbers of condition tokens (e.g., 3–8 anchors), we pad to a maximum count using learnable missing embeddings and apply attention masks to ignore padded positions.

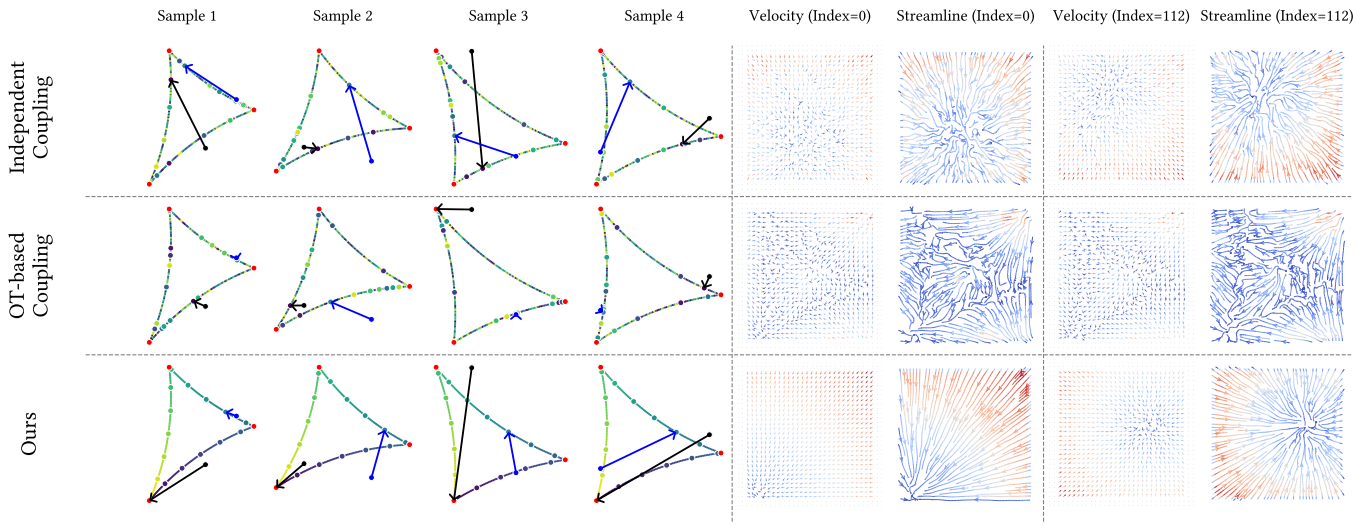


Fig. 4. Visualization of index-conditioned velocity fields in a realistic minimal-surface configuration (area-constrained). For each strategy, we sample 1000 random noise configurations $\boldsymbol{x}_0$ for each minimal-surface target $\boldsymbol{x}_1$ and construct couplings using independent coupling [Lipman et al. 2022], OT-based coupling [Klein et al. 2023; Song et al. 2023], and our orbit-space canonicalization. Left: four minimal-surface boundary point sets with particles colored by index; arrows show, for a single trial, the velocity $\boldsymbol{x}_1^i - \boldsymbol{x}_0^i$ of two highlighted particles (black: Index=0, blue: Index=112) from a shared initialization $\boldsymbol{x}_0$. Right: empirical per-particle velocity fields for the same indices, obtained by aggregating these velocities over the 1000 $(\boldsymbol{x}_0, \boldsymbol{x}_1)$ pairs and interpolating them onto a grid; streamlines visualize flow trajectories induced by these vector fields. As in the conceptual illustration Figure 3, independent and OT-based couplings spread the possible endpoints for each index around the surface, whereas our orbit-space canonicalization concentrates them into a small, stable region and yields straighter flows.

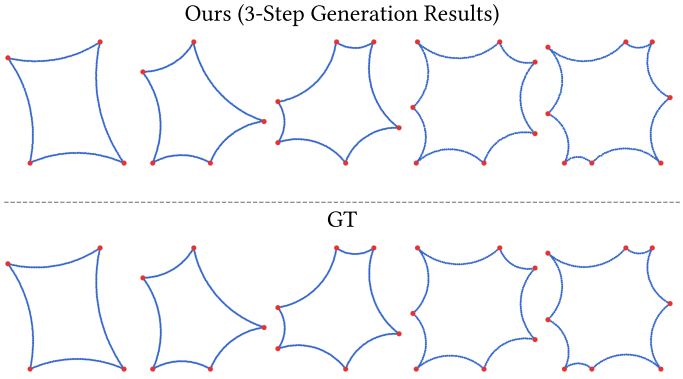


Fig. 5. **Minimal surface generation with variable anchors (3–8).** 3-step generation results from a single conditional model trained on varying anchor counts. The generated boundaries appear smooth and accurate across diverse configurations.

4.2 Orbit-Space Canonicalization on X_1

In this subsection we analyze the regression problem and show that orbit-space canonicalization of the terminal endpoint X_1 reduces the conditional covariance $\mathrm{Cov}(Y \mid X_t = \boldsymbol{x})$ of the regression target Y, which measures how noisy the regression problem is for the velocity predictor. We illustrate the conditional distribution conceptually in Figure 3 and in a real training scenario in Figure 4.

We define the regression target (for the linear path; the geometric-path extension is in Section 5.3) as

$$Y := \frac{\boldsymbol{X}_1 - \boldsymbol{X}_t}{1-t} = \boldsymbol{X}_1 - \boldsymbol{X}_0. \tag{4}$$

The Bayes-optimal velocity is

$$u^*(\boldsymbol{x}, t) = \mathbb{E}[Y \mid X_t = \boldsymbol{x}]\,. \tag{5}$$

A smaller conditional covariance $\mathrm{Cov}(Y \mid X_t = \boldsymbol{x})$ directly lowers the irreducible MSE of the Bayes-optimal predictor, making the velocity regression easier to learn. We use the trace $\mathrm{tr}\,\mathrm{Cov}(\cdot)$ as a scalar measure of this covariance.

Orbit symmetry and canonicalization. After pose normalization (Section 3.3), we model the residual permutation symmetry by assuming that, for each fixed $X_t = \boldsymbol{x}$,

$$X_1 \mid (X_t = \boldsymbol{x}) \overset{d}{=} \rho(G)\,\zeta_{\boldsymbol{x}}, \tag{6}$$

where G is a random permutation in S_N and $\zeta_{\boldsymbol{x}}$ is a canonical representative. Applying the law of total covariance to X_1 with respect to G yields the decomposition (see Appendix C for the full derivation):

$$\begin{aligned}\mathrm{Cov}(X_1 \mid X_t = \boldsymbol{x}) = &\underbrace{\mathbb{E}_G\big[\mathrm{Cov}(X_1 \mid X_t = \boldsymbol{x}, G)\big]}_{\text{intrinsic variability}} \\ &+ \underbrace{\mathrm{Cov}\big(\mathbb{E}[X_1 \mid X_t = \boldsymbol{x}, G] \;\big|\; X_t = \boldsymbol{x}\big)}_{\text{role-ambiguity term} \,\succeq\, 0}.\end{aligned} \tag{7}$$

The first term is the intrinsic variability conditioned on a fixed permutation, averaged over G; the second term captures the additional variability from random G. Exploiting the G-invariance of a canonicalization map C (Section 3.3), the second term vanishes for $\widetilde{X}_1 := C(X_1)$, giving for $\widetilde{Y} := (\widetilde{X}_1 - X_t)/(1-t)$:

$$\operatorname{tr}\operatorname{Cov}(Y \mid X_t = \boldsymbol{x}) \;\geq\; \operatorname{tr}\operatorname{Cov}\Big(\widetilde{Y} \mid X_t = \boldsymbol{x}\Big). \tag{8}$$

Therefore, without canonicalization, this covariance contains an extra component from random permutations, forcing identity embeddings to average over different roles. Orbit-space canonicalization of X_1 removes exactly this role-ambiguity term, so that each identity embedding can specialize to a well-defined canonical role and the targets become easier to learn. Our ablations in Section 7.3.2 empirically confirm that the largest gains arise when identity embeddings and one-sided canonicalization are used together.

4.3 Orbit-continuous canonicalization and straight flows

We now turn to our second objective: **encouraging straight flows**. Beyond reducing conditional variance at each fixed configuration $\boldsymbol{x}_t$, we would like the Bayes-optimal velocity field $\boldsymbol{u}^*(\boldsymbol{x}, t)$ in Eq. (5) to vary smoothly across nearby configurations. Since the full trajectory satisfies the ODE

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}_t = \boldsymbol{u}^*(\boldsymbol{x}_t, t),$$

a locally Lipschitz velocity field with a small Lipschitz constant encourages nearby trajectories to remain coherent and to change direction smoothly over time. This motivates a canonicalization map C that is well behaved on the orbit space, so that nearby orbits are mapped to nearby canonical representatives.

To make this precise, we consider the orbit space $\boldsymbol{O} = (\mathbb{R}^D)^N/G$ and equip it with a metric d_O that is invariant under the group action. We say that a canonicalization map $C : (\mathbb{R}^D)^N \to (\mathbb{R}^D)^N$ is *orbit-continuous* if it maps nearby orbits to nearby canonical representatives in $(\mathbb{R}^D)^N$ in a Lipschitz manner, i.e., if there exists a constant L_{orb} such that for all $\boldsymbol{x}, \boldsymbol{x}' \in (\mathbb{R}^D)^N$,

$$\|C(\boldsymbol{x}) - C(\boldsymbol{x}')\| \;\leq\; L_{\mathrm{orb}}\, d_O\big(\mathrm{Orb}(\boldsymbol{x}), \mathrm{Orb}(\boldsymbol{x}')\big). \tag{9}$$

The Bayes-optimal velocity can be written as

$$\boldsymbol{u}^*(\boldsymbol{x}, t) = \mathbb{E}[Y \mid X_t = \boldsymbol{x}] = \frac{1}{1-t}\,\big(\boldsymbol{m}(\boldsymbol{x}) - \boldsymbol{x}\big), \tag{10}$$

where the canonical mean is $\boldsymbol{m}(\boldsymbol{x}) := \mathbb{E}\big[\widetilde{X}_1 \mid X_t = \boldsymbol{x}\big]$.

Under natural smoothness assumptions on the endpoint distribution over the orbit space (see Appendix E for details), the canonical means $\boldsymbol{m}(\boldsymbol{x})$ inherit orbit-Lipschitz regularity from the orbit-continuity of C. Combining this with Eq. (10), we obtain a local Lipschitz bound for the Bayes-optimal velocity field:

$$\|\boldsymbol{u}^*(\boldsymbol{x}, t) - \boldsymbol{u}^*(\boldsymbol{x}', t)\| \;\leq\; L_{\mathrm{vel}}(t)\, d_O\big(\mathrm{Orb}(\boldsymbol{x}), \mathrm{Orb}(\boldsymbol{x}')\big), \tag{11}$$

where $L_{\mathrm{vel}}(t)$ is a time-dependent constant controlled by L_{orb} and the intrinsic smoothness of the canonical means. Thus, orbit-continuous canonicalization maps that align neighboring orbits with neighboring canonical representatives ensure a continuous velocity field and thereby encourage straight flows. Since each per-particle velocity is a component of the full vector field $\boldsymbol{u}^*(\boldsymbol{x}, t)$, this regularity also transfers componentwise to the individual particle trajectories.

Practical canonicalization maps. In practice, we first apply a simple pose-normalization step (recenter and align a PCA frame), which removes translations and global rotations. The main challenge lies in the remaining symmetry, the *permutation* part S_N, whose combinatorial complexity grows as $N!$. We therefore focus our design effort on a robust permutation canonicalization. We find that a Hilbert space-filling curve ordering provides a stable permutation of particle indices under small perturbations. Figure 26 compares several alternatives (e.g., Z-order, Moore curve). For joint canonicalization (See Section 5.3) we analogously apply a n-dimensional Hilbert sort [Skilling 2004]. In the minimal-surface experiments (Fig. 5 and Fig. 6) we instead use a simple rule: we pin left bottom anchor as index 0 and then enumerate boundary particles in counterclockwise order along the curve. All of these constructions are designed to satisfy the orbit-continuity intuition that neighboring orbits should induce nearby canonical representatives and avoid abrupt role flips.

4.4 Canonicalizing X_0 increases Lipschitz ratios

A natural follow-up question is whether we should also canonicalize the noise endpoint X_0. We show that further canonicalizing X_0 amplifies *directional cancellation* events and inflates the local Lipschitz ratios of the velocity field; a detailed derivation is given in Appendix D.

We measure the smoothness of the velocity field via nearest-neighbor Lipschitz ratios. For each k-NN edge (i, j) built on the interpolants $\boldsymbol{x}_t^{(i)} = (1-t)\boldsymbol{x}_0^{(i)} + t\boldsymbol{x}_1^{(i)}$, the Lipschitz ratio is

$$L_{ij}(t)^2 = \frac{\big\|\Delta_1^{(ij)} - \Delta_0^{(ij)}\big\|^2}{\big\|(1-t)\,\Delta_0^{(ij)} + t\,\Delta_1^{(ij)}\big\|^2}, \tag{12}$$

where $\Delta_\ell^{(ij)} := \boldsymbol{x}_\ell^{(i)} - \boldsymbol{x}_\ell^{(j)}$. Once X_1 is canonicalized, $\Delta_1^{(ij)}$ is typically small. If X_0 is also canonicalized, the contracted $\widetilde{\Delta}_0^{(ij)}$ reaches a comparable scale to $\Delta_1^{(ij)}$, making it much easier for the two vectors to nearly cancel in the denominator while the numerator stays large (Figure 7). By contrast, keeping X_0 uncanonicalized preserves a large spread in $\Delta_0^{(ij)}$, making such cancellation statistically unlikely.

We empirically verify in Section 7.4 (Figure 28) that our one-sided canonicalization strategy (canonicalizing X_1 only) achieves the smallest local Lipschitz ratios and the lowest prevalence of high-cancellation edges among all four canonicalization regimes (no canonicalization, X_0 only, X_1 only, and both).

5 Geometric Probability Paths for Attribute Encoding

This section focuses on the third key component of OGPP: **geometric probability paths**. The previous section focused on how to process the *endpoints* of the probability path by performing symmetry reduction on the terminal distribution. We now turn to the second design axis: the *shape* of the probability path itself.

In standard flow matching, the conditional path between noise and data is often taken to be a linear interpolation. While this choice is simple and effective for transporting particle positions (or, more generally, distributions), it leaves the terminal velocity field at $t = 1$ geometrically under-utilized: the velocity at the endpoint does not carry any intrinsic meaning. We exploit these unused degrees of freedom by constructing *geometric probability paths* whose terminal

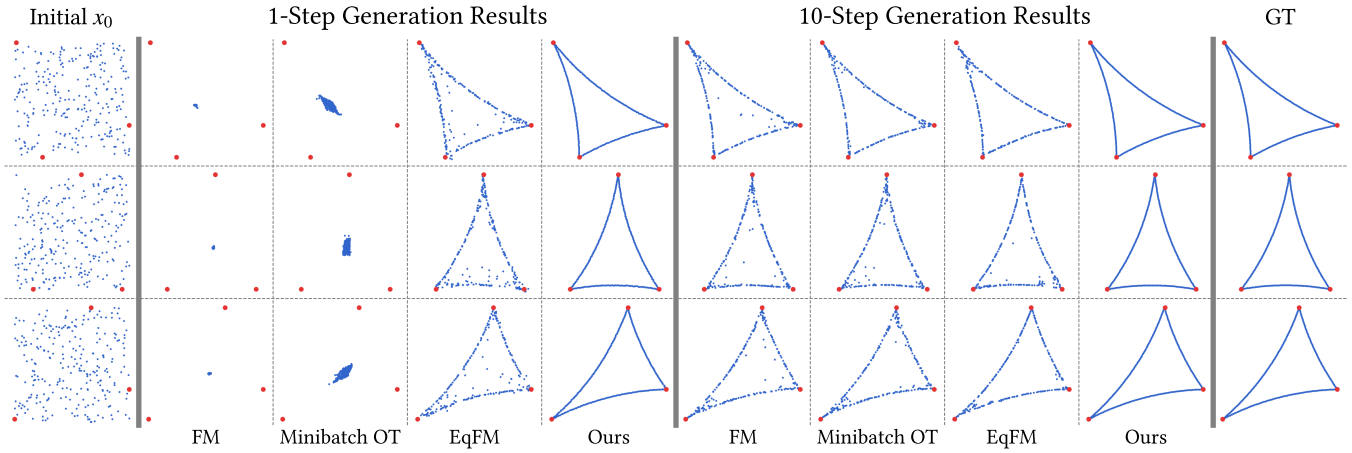


Fig. 6. **Minimal surface generation (3 anchors).** We consider the 2D analog of minimal surfaces: soap film boundaries satisfying area constraints. We compare 1-step and 10-step generation results with different methods. Red dots indicate anchor particles; blue dots show generated boundary particles. Our method produces accurate minimal surface boundaries in a single step, while baselines require multiple steps and exhibit artifacts. Ground truth (GT) shown on the right.

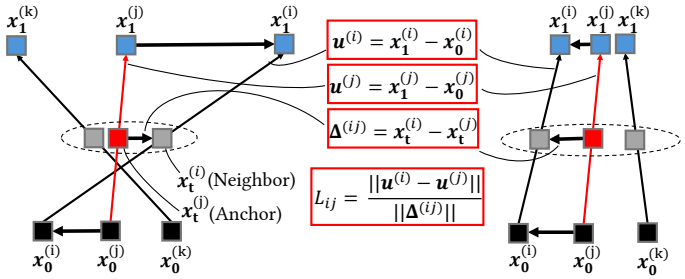


Fig. 7. Illustration of directional cancellation in the Lipschitz ratio. Squares represent stacked particle vectors in $\mathbb{R}^{DN}$. **Left:** When $\Delta_0^{(ij)}$ and $\Delta_1^{(ij)}$ point in opposite directions, the denominator nearly vanishes while the numerator remains large, yielding a large Lipschitz ratio. **Right:** Without such cancellation, the ratio stays moderate.

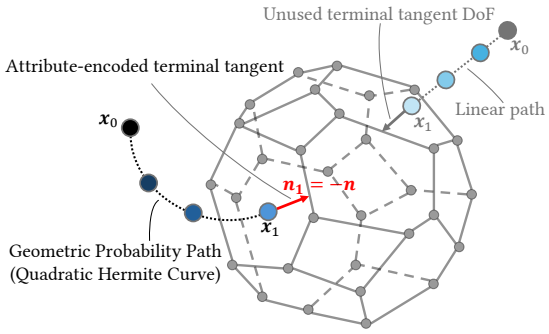


Fig. 8. **Geometric probability paths for attribute encoding.** *Left:* Our geometric probability path (quadratic Hermite curve) aligns the terminal tangent with the surface normal $\boldsymbol{n}_1$, encoding per-particle attributes into the path geometry. *Right:* Standard linear interpolation leaves the terminal velocity as an unused degree of freedom.

tangent aligns with a per-particle attribute. In this work, we instantiate this attribute as the surface normal of the shape. Formally, let the conditioning variable be $\boldsymbol{z} = (\boldsymbol{x}_1, \boldsymbol{n}_1)$, where $\boldsymbol{x}_1 \in \mathbb{R}^3$ is the target position and $\boldsymbol{n}_1 \in \mathbb{R}^3$ is the associated surface normal. We design conditional probability paths that satisfy three boundary conditions: (i) the path starts at noise, $\boldsymbol{x}(0) = \boldsymbol{x}_0 \sim p_{\text{init}}$; (ii) the path ends at the target position, $\boldsymbol{x}(1) = \boldsymbol{x}_1$; and (iii) the terminal velocity encodes the attribute, $\boldsymbol{v}_1 = \dot{\boldsymbol{x}}(1) \propto \boldsymbol{n}_1$. Conditions (i) and (ii) leave the *shape* of the path largely unconstrained; by slightly bending the path away from a straight line to enforce (iii), we turn the terminal tangent—an otherwise free degree of freedom in the linear path—into a structured carrier of geometric information (see Figure 8 and Figure 10 for an illustration).

In this section, we first introduce quadratic Hermite probability paths in Section 5.1. We then choose an arc-length-aware terminal velocity to stabilize time sampling in Section 5.2, extend canonicalization to the joint position-normal endpoints via joint canonicalization in Section 5.3, and finally characterize the marginal terminal velocity at $t = 1$ as a normal predictor together with its training and inference usage in Section 5.4 and Section 5.5.

5.1 Quadratic Hermite Curves

We construct the probability paths using a quadratic Hermite curve that satisfies the boundary conditions above. We define the curve $\gamma(t)$ as:

$$\gamma(t) = \boldsymbol{x}_0 + \alpha(t) \cdot (\boldsymbol{x}_1 - \boldsymbol{x}_0) + \beta(t) \cdot \boldsymbol{v}_1, \tag{13}$$

where the basis functions are $\alpha(t) = 2t - t^2$, $\beta(t) = t^2 - t$, and $\boldsymbol{v}_1$ denotes the terminal tangent that we assign at $t = 1$.

Conditional velocity field. Differentiating Eq. (13), we obtain the conditional velocity field along the curve:

$$\boldsymbol{u}_t^{\text{ref}}(\boldsymbol{x}_t|\boldsymbol{z}) = \frac{2}{1-t}(\boldsymbol{x}_1 - \boldsymbol{x}_t) - \boldsymbol{v}_1. \tag{14}$$

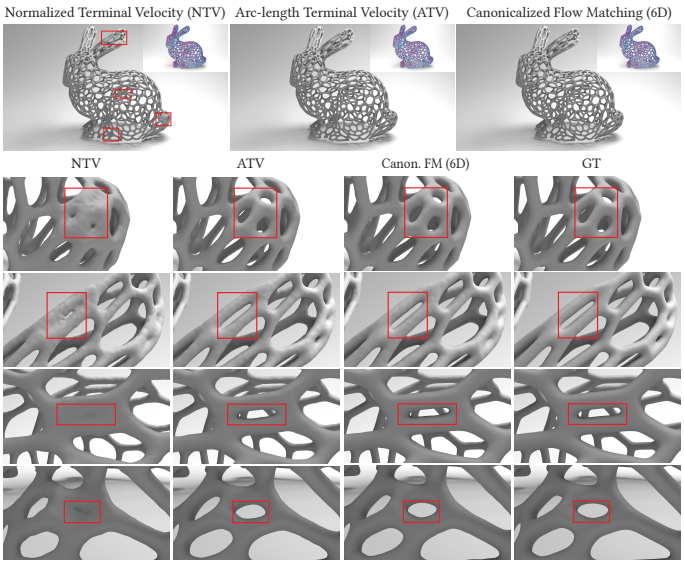


Fig. 9. **Ablation on normal encoding strategies.** Top row: Screened Poisson reconstructions from NTV, ATV, and canonicalized 6D flow matching (Canon. FM 6D), with normal-colored point clouds inset. Bottom rows: zoomed-in comparisons against the ground truth (GT). ATV and Canon. FM (6D) achieve comparable quality and accurately reconstruct small Voronoi cells and thin structures that NTV fails to capture.

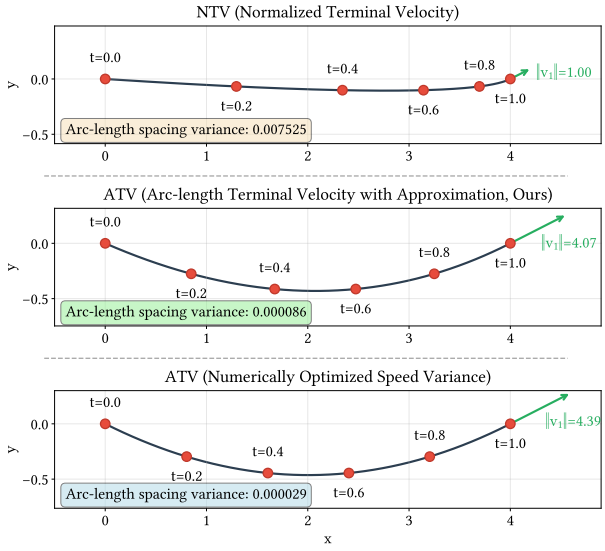


Fig. 10. Comparison of terminal velocity magnitude choices. Red dots indicate uniform time samples $t \in \{0, 0.2, 0.4, 0.6, 0.8, 1.0\}$; green arrows show $\boldsymbol{v}_1$. **Top:** NTV sets $\|\boldsymbol{v}_1\| = 1$, yielding nonuniform arc-length spacing. **Middle:** Our ATV approximation sets $\|\boldsymbol{v}_1\| = D(1 + \lambda(1 - S))$, achieving near-uniform spacing with negligible overhead. **Bottom:** Numerically optimized ATV chooses $\|\boldsymbol{v}_1\|$ to minimize speed variance along the curve, giving optimal uniformity but requiring numerical optimization.

Remark 1. *The quadratic Hermite path is the simplest polynomial path satisfying our three boundary conditions. A cubic Hermite spline would introduce an additional degree of freedom (the tangent at $t = 0$), which is unnecessary for our purpose. We verify in our ablation study (Table 9) that the quadratic path achieves the best performance.*

5.2 Arc-Length Terminal Velocity (ATV)

For surface normals, only the *direction* of $\boldsymbol{v}_1$ is constrained; its magnitude is a free parameter. We exploit this freedom to achieve approximately uniform speed profiles along each trajectory, so that uniform time sampling $t \sim \text{Uniform}(0, 1)$ correlates well with uniform sampling along the curve.

Concretely, for each particle, Algorithm 1 (lines 5–13) computes the chord length $D = \|\boldsymbol{x}_1 - \boldsymbol{x}_0\|$ and the alignment $S = \hat{\boldsymbol{d}} \cdot \hat{\boldsymbol{n}}_1$ between chord direction and normal, and sets the terminal velocity to

$$L_{\text{arc}} = D\big(1 + \lambda(1 - S)\big), \qquad \boldsymbol{v}_1 = L_{\text{arc}}\, \hat{\boldsymbol{n}}_1. \tag{15}$$

The scaling L_{arc} adapts the terminal speed to the chord length and the angle between the chord and the normal: when the normal is aligned with the chord ($S \approx 1$), the path is nearly straight and $\|\boldsymbol{v}_1\| \approx D$; when they are misaligned ($S \ll 1$), the path bends more and a larger $\|\boldsymbol{v}_1\|$ compensates to maintain uniform speed. This computation is inexpensive (only norms and dot products) and empirically produces trajectories whose speed variation over t is much smaller than the naive unit-norm baseline (normalized terminal velocity, NTV, which sets $\|\boldsymbol{v}_1\| = 1$). As shown in Figure 10, our ATV approximation closely matches a numerically optimized solution that directly minimizes speed variance. A detailed discussion of why NTV leads to nonuniform speed profiles is provided in Appendix F. We additionally compare NTV and ATV in Figure 9 (see experimental details in Section 7.3.4).

5.3 Joint Canonicalization for Attribute-Encoded Paths

The conditional-covariance analysis in Section 4.2 was derived for the linear probability path, where the regression target $Y = (X_1 - X_t)/(1 - t)$ depends only on the endpoint position X_1. Under the geometric probability path Eq. (13), differentiating yields the regression target

$$Y_t = \frac{2}{1 - t}\big(X_1 - X_t\big) - V_1, \tag{16}$$

where V_1 is the stacked terminal velocity defined by Eq. (15). For each $X_t = \boldsymbol{x}$, the randomness now comes from the joint endpoint $Z := (X_1, V_1) \in (\mathbb{R}^6)^N$, not just from X_1.

Under the same orbit-symmetry factorization as Section 4.2, extended to the joint endpoint Z, orbit-space canonicalization of Z reduces the conditional covariance $\text{Cov}(Z \mid X_t = \boldsymbol{x})$; since Y_t is an affine function of Z, this decreases the conditional covariance of Y_t as well. In particular, joint canonicalization lowers the irreducible MSE of the velocity predictor, making the geometric-path regression problem strictly easier to learn.

Implementation via 6D Hilbert curve. In practice, we concatenate the position $\boldsymbol{x}_1$ and attribute $\boldsymbol{n}_1$ for each particle into a six-dimensional vector $(\boldsymbol{x}_1, \boldsymbol{n}_1) \in \mathbb{R}^6$, and apply a N-dimensional Hilbert-curve [Skilling 2004] in this joint space. We compare 6D Hilbert ordering (position + normal) against 3D Hilbert ordering (position

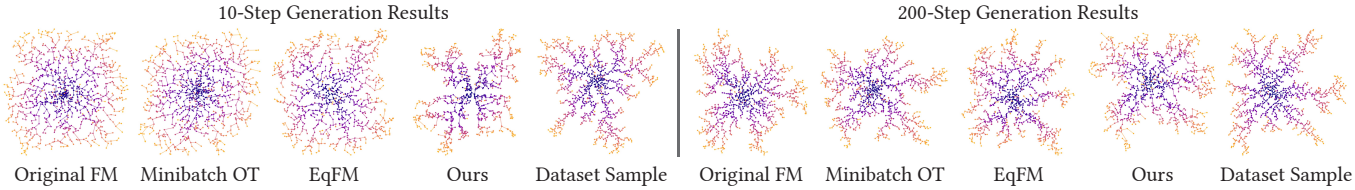


Fig. 11. **DLA generation comparison.** 10-step (left) and 200-step (right) generation results. At 10 steps, baselines produce scattered, non-fractal structures, while ours exhibits realistic dendritic branching. At 200 steps, all methods improve; ours appears closest to the ground-truth fractal morphology. Color encodes particle attachment order (early: dark, late: light).

only) in Table 9: the joint canonicalization improves normal estimation (average cosine similarity from 0.91 to 0.92, standard deviation from 0.21 to 0.19) and generation quality (1-NNA accuracy from 0.78 to 0.61).

5.4 Marginal Velocity at Terminal Time

The key theoretical question is: *what does the marginal velocity field* $\boldsymbol{u}_1^{\text{ref}}(\boldsymbol{x})$ *represent at the terminal time?* By the marginalization trick (Theorem 3.2), the trained network learns to approximate the marginal velocity, not the conditional one. We now show that at $t = 1$, the marginal velocity is precisely the expected attribute given the position.

Intuitively, at $t = 1$, the conditional path collapses to a point mass $p_1(\cdot|\boldsymbol{z}) = \delta_{\boldsymbol{x}_1}$, so only conditioning variables with $\boldsymbol{x}_1 = \boldsymbol{x}$ contribute to the marginalization integral. Since $\boldsymbol{u}_1^{\text{ref}}(\boldsymbol{x}|\boldsymbol{z}) = N_1$, the marginal velocity becomes:

$$\boldsymbol{u}_1^{\text{ref}}(\boldsymbol{x}) = \int N_1 \, p(N_1|X_1 = \boldsymbol{x}) \, \mathrm{d}N_1 = \mathbb{E}\left[N_1 \,|\, X_1 = \boldsymbol{x}\right]. \tag{17}$$

See the supplement for the full proof.

5.5 Implications for Training and Inference

The preceding analysis has direct practical implications. During training, the same network $\boldsymbol{u}_t^\theta$ learns:

- the transport velocity for $t \in [0, 1)$, and
- at $t = 1$, the conditional expectation $\mathbb{E}[N_1 \mid X_1 = \boldsymbol{x}]$.

No separate network or training procedure is needed for normal estimation. Therefore, during inference:

(1) first integrate the ODE from $t = 0$ to $t = 1$ to obtain the generated position $\boldsymbol{x}_1$;
(2) then evaluate $\boldsymbol{u}_1^\theta(\boldsymbol{x}_1)$ to get the predicted attribute N_1.

The generated point cloud comes equipped with surface normals as a *byproduct* of the flow, at no additional computational cost. Note that individual ODE trajectories are governed by the learned marginal velocity field rather than any single conditional Hermite path.

6 Algorithm Overview

We summarize our training and inference procedures in Algorithm 1 and Algorithm 2, assuming a batch size of 1. Training integrates three key components, as illustrated in Figure 2. First, *orbit-space canonicalization* (Section 4) canonicalizes only the terminal endpoint X_1 to reduce conditional covariance and straighten flows (lines 2–4). Second, *particle index embedding* (Section 4.1) allows each index to specialize to its canonical role, turning the regression problem

Algorithm 1 OGPP Training

Input: Dataset $\mathcal{D} = \{(\boldsymbol{x}_1^{(j)}, \boldsymbol{n}_1^{(j)})\}_{j=1}^M$, number of particles per sample N, canonicalization map $C(\cdot)$, hyperparameter λ
Output: Trained velocity network $\boldsymbol{u}_t^\theta$

1: **repeat**
2: Sample $\boldsymbol{x}_0^{(i)} \sim \text{Uniform}([-1, 1]^N)$ ▷ Noise
3: Sample $\boldsymbol{z}_1^{(i)} = (\boldsymbol{x}_1^{(i)}, \boldsymbol{n}_1^{(i)})$ from $\mathcal{D}$ ▷ Data with attributes
4: $(\boldsymbol{x}_1^{(i)}, \boldsymbol{n}_1^{(i)}) \leftarrow C(\boldsymbol{x}_1^{(i)}, \boldsymbol{n}_1^{(i)})$ ▷ Joint canon. (Sec. 4, Sec. 5.3)
5: **for** each particle $k = 1, \ldots, N$ **in parallel do**
6: $D^{(i),k} \leftarrow \|\boldsymbol{x}_1^{(i),k} - \boldsymbol{x}_0^{(i),k}\|$ ▷ Chord length
7: $\hat{\boldsymbol{d}}^{(i),k} \leftarrow (\boldsymbol{x}_1^{(i),k} - \boldsymbol{x}_0^{(i),k})/D^{(i),k}$ ▷ Chord direction
8: $\hat{\boldsymbol{n}}_1^{(i),k} \leftarrow \boldsymbol{n}_1^{(i),k}/\|\boldsymbol{n}_1^{(i),k}\|$ ▷ Unit normal
9: $S^{(i),k} \leftarrow \hat{\boldsymbol{d}}^{(i),k} \cdot \hat{\boldsymbol{n}}_1^{(i),k}$ ▷ Directional alignment
10: $L_{\text{arc}}^{(i),k} \leftarrow D^{(i),k} \cdot (1 + \lambda(1 - S^{(i),k}))$ ▷ Arc-length
11: $\boldsymbol{v}_1^{(i),k} \leftarrow L_{\text{arc}}^{(i),k} \cdot \hat{\boldsymbol{n}}_1^{(i),k}$ ▷ ATV (Eq. 15, Sec. 5.2)
12: Construct $\gamma^{(i),k}(t)$ from $\boldsymbol{x}_0^{(i),k}, \boldsymbol{x}_1^{(i),k}, \boldsymbol{v}_1^{(i),k}$ ▷ Sec. 5.1
13: **end for**
14: Sample $t \sim \text{Uniform}([0, 1])$
15: **for** each particle $k = 1, \ldots, N$ **in parallel do**
16: $\boldsymbol{x}_t^{(i),k} \leftarrow \gamma^{(i),k}(t)$ ▷ Interpolated position
17: $\boldsymbol{v}_t^{(i),k} \leftarrow \dot{\gamma}^{(i),k}(t)$ ▷ Reference velocity (Eq. 14)
18: **end for**
19: $\mathcal{L}^{(i)} \leftarrow \frac{1}{N} \sum_{k=1}^N \|\boldsymbol{u}_t^{\theta,k}(\boldsymbol{x}_t^{(i)}) - \boldsymbol{v}_t^{(i),k}\|^2$ ▷ MSE loss
20: Update θ via gradient descent on $\mathcal{L}^{(i)}$
21: **until** converged

from a noisy mixture into well-separated families of trajectories. Third, *geometric probability paths* (Section 5) replace linear paths with quadratic Hermite paths, encoding surface normals in the terminal tangent with arc-length-aware velocity (lines 6–12).

At inference time (Algorithm 2), we draw noise from the same prior and integrate the learned velocity field $\boldsymbol{u}_t^\theta$ forward from t=0 to t=1 using a standard ODE solver. The final positions $\boldsymbol{x}_1$ give the generated particle locations, and the terminal velocity $\boldsymbol{u}_1$ yields the surface normals after normalization.

7 Experiments

We evaluate our framework on two groups of tasks: energy-driven particle generation and 3D shape generation. By energy-driven, we refer to particle generation problems whose targets are equilibrium configurations of explicit physical or geometric energy functionals.

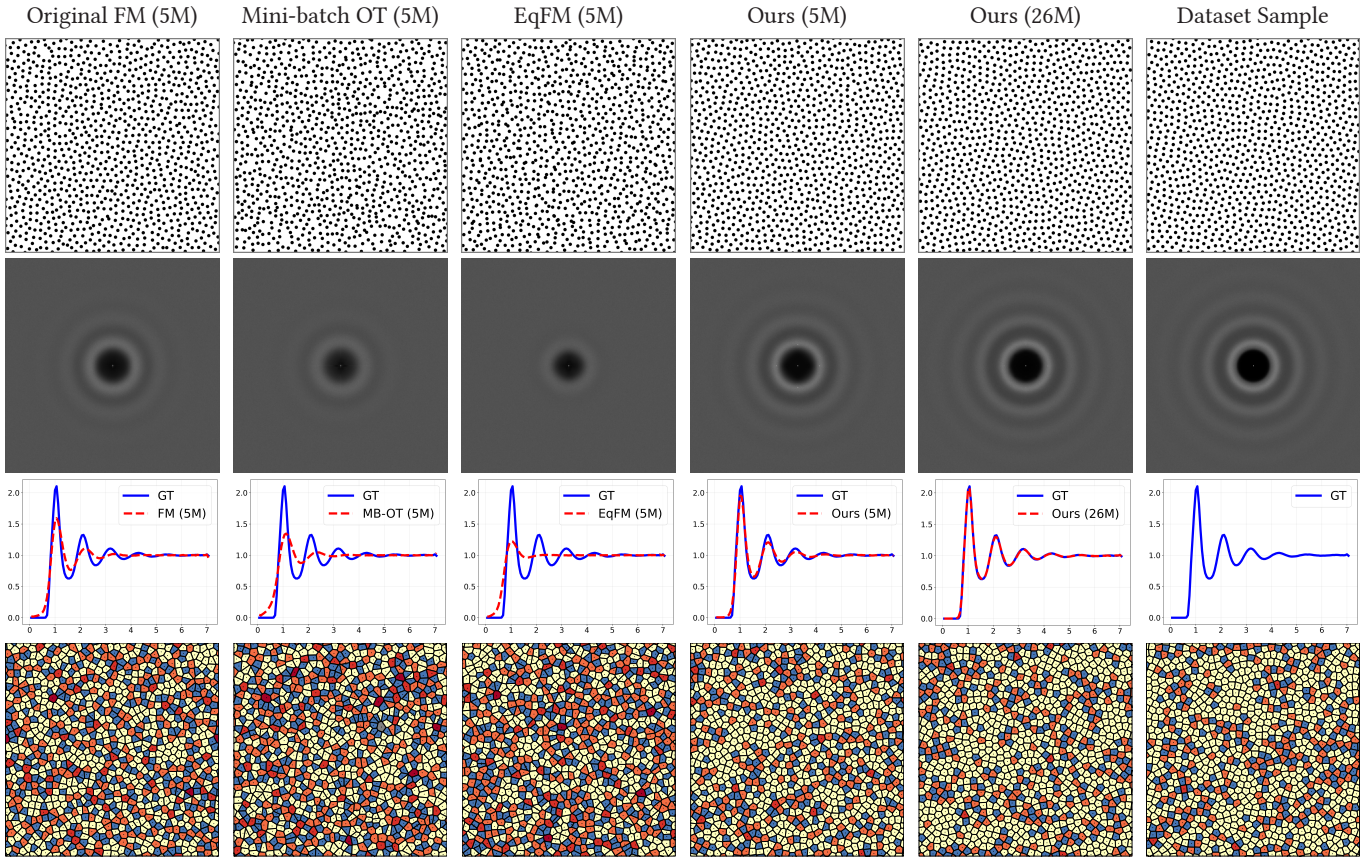


Fig. 12. **Uniform blue-noise generation.** Comparison of flow-matching variants for 1024-point uniform blue-noise generation. *Row 1:* One generated point set. *Row 2:* 2D power spectrum averaged over 1K generated samples. *Row 3:* Radial power spectrum averaged over 1K generated samples. *Row 4:* Delaunay triangulation valence (color indicates neighbor count). Our method (5M and 26M) produces the sharpest spectral ring, and the results closely match the ground truth.

Algorithm 2 OGPP Inference

Input: Trained velocity network $\boldsymbol{u}_t^\theta$, number of particles per sample N, number of steps K
Output: Generated particles $\{\boldsymbol{x}_1^i\}_{i=1}^N$ with normals $\{\hat{\boldsymbol{n}}^i\}_{i=1}^N$

Sample $\boldsymbol{x}_0 \sim \text{Uniform}([-1,1]^N)$ ▷ Initial noise
$\Delta t \leftarrow 1/K$ ▷ Step size
for $k = 0, \ldots, K-1$ **do**
  $t \leftarrow k \cdot \Delta t$
  $\boldsymbol{u}_t^\theta(\boldsymbol{x}_t) \leftarrow$ evaluate NN at $(\boldsymbol{x}_t, t)$
  for each particle $i = 1, \ldots, N$ **in parallel do**
    $\boldsymbol{x}_{t+\Delta t}^i \leftarrow \textsc{Step}(\boldsymbol{x}_t^i,\ \boldsymbol{u}_t^{\theta,i}(\boldsymbol{x}_t),\ \Delta t)$ ▷ ODE integration
  end for
end for
$\boldsymbol{v}_1^i \leftarrow \boldsymbol{u}_1^{\theta,i}(\boldsymbol{x}_1)$ for $i = 1, \ldots, N$ ▷ Terminal velocity
$\hat{\boldsymbol{n}}^i \leftarrow \boldsymbol{v}_1^i / \|\boldsymbol{v}_1^i\|$ for $i = 1, \ldots, N$ ▷ Unit normal
return $\{\boldsymbol{x}_1^i,\ \hat{\boldsymbol{n}}^i\}_{i=1}^N$

Such tasks include blue-noise sampling, minimal surfaces, diffusion-limited aggregation (DLA), and the multilayer Thomson problem. 3D shape and geometry tasks include point cloud generation on ShapeNet and single-shape encoding [Zhang et al. 2025] on complex meshes.

Model, Training, and Dataset Setup. For model architecture, we adopt a plain transformer (See Section 4.1) with 5M or 26M parameters depending on task complexity, trained on NVIDIA RTX 4090 or H200 SXM GPUs. For canonicalization strategies, we use Hilbert curve sorting as our standard strategy across all experiments, except for minimal surface generation where we use counterclockwise polygon ordering to respect boundary structure. We train for 3K epochs on energy-driven tasks (6K for 26M blue noise), 8K epochs on CelebA, and 50K epochs on ShapeNet for ours and comparison models. Our method and baselines (Original FM, Minibatch OT) use batch size 200 for most tasks, with 256 for 5M blue noise and 3360 for 26M blue noise; Similar to [Hui et al. 2025], we use batch size 8 for EqFM due to its $O(B^2N^3)$ OT coupling cost, but we train it for the same wall-clock time as the other methods to ensure a fair comparison. Full training configurations are provided in Table A-1 in the appendix. For energy-driven tasks, we generate training data using domain-specific algorithms and solvers. We use the CelebA dataset

[Liu et al. 2015] for adaptive blue noise generation, ShapeNet [Chang et al. 2015] for point cloud generation, and Thingi10k [Zhou and Jacobson 2016] for single-shape encoding. Evaluation metrics and baselines specific to each task are described in the corresponding subsections.

7.1 Energy-driven Particle Generation

We evaluate our approach on four energy-driven particle generation tasks, where equilibrium configurations arise as minimizers of physical or geometric objectives. For each task, we assess quality using intrinsic metrics that are aligned with the underlying energy functional. We first introduce the task background, then describe the dataset construction and evaluation metrics, and finally compare our method against competing baselines.

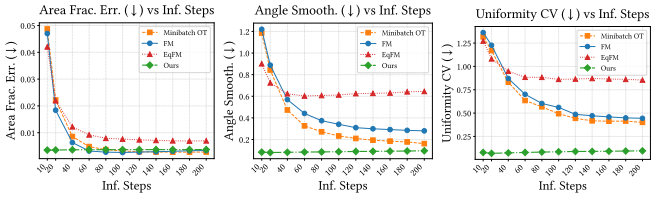
The plots that compare metrics against baselines as a function of inference steps (Figure 13) demonstrate that our method yields straighter, higher-quality flows: it achieves better metric values with fewer steps and typically converges earlier than the baselines.

7.1.1 Blue-Noise Generation. Blue-noise distributions are fundamental to rendering and scientific computing, characterized by suppressed low-frequency content and isotropy that maximize sampling efficiency while minimizing aliasing artifacts [Cook 1986; Ulichney 1988]. Yellott [Yellott Jr 1983] demonstrated that primate photoreceptor arrangements exhibit blue-noise characteristics, suggesting evolutionary optimization for visual sampling.

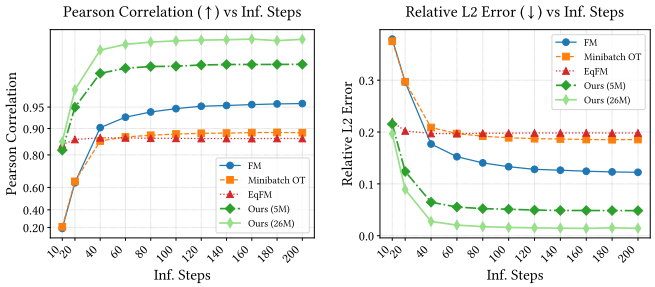
Dataset Construction and Evaluation Metrics. We use the state-of-the-art Gaussian Blue Noise (GBN) [Ahmed et al. 2022] to generate the uniform blue-noise dataset and its serial variant [Ahmed 2024] to generate the adaptive blue-noise dataset. We evaluate generation quality visually using the radial power spectrum and the valence of the Delaunay triangulation, and quantitatively using two metrics: Pearson correlation and relative L_2 error against the ground-truth spectral profile.

Uniform Blue Noise Generation Results. We generate 400K uniform blue-noise point sets of $N = 1024$ points using a constant density field as input to GBN. We train both the baseline methods and our approach on a subset of 50K point sets for unconditional generation, and additionally train a large variant of our model on the full set of 400K point sets. We compare against Original Flow Matching [Lipman et al. 2022], Minibatch OT [Pooladian et al. 2023; Tong et al. 2023], and Equivariant Flow Matching (EqFM) [Klein et al. 2023; Song et al. 2023], all trained with 5M parameters. To demonstrate scalability, we additionally train a 26M-parameter variant of our model on the full dataset.

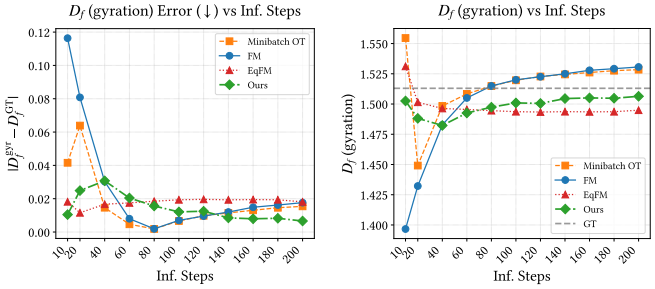
Figure 12 shows qualitative and quantitative comparisons. The first row displays generated point samples; the second row shows the 2D power spectrum averaged over 1000 generated samples; the third row plots the radial power spectrum, computed by azimuthally averaging the 2D power spectrum; and the fourth row visualizes the valence of Delaunay triangulation, where each Voronoi cell is colored by its number of neighbors, more uniform coloring indicates better spatial regularity. Among the methods compared, our method produces the closest spectral match to the ground-truth profile. Quantitative results are summarized in Table 2: our 5M model



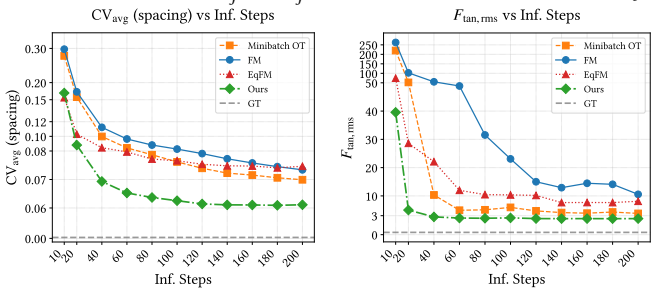
(a) Minimal surface: area fraction error, angle smoothness, and uniformity CV (all lower is better). Circled points highlight representative steps.



(b) Blue noise: Pearson correlation (higher is better) and relative L_2 error (lower is better).



(c) DLA: absolute error $|D_f^{\text{gen}} - D_f^{\text{GT}}|$ and estimated fractal dimension D_f.



(d) Multilayer Thomson: CV average (spatial uniformity) and tangential force RMS (equilibrium deviation).

Fig. 13. **Quantitative metrics vs. inference steps for all energy-driven tasks.** Our method (green) achieves low error from early steps and remains stable, while baselines converge slower and plateau at higher error levels.

outperforms the tested baselines, and our 26M model achieves Pearson correlation 0.999 and L_2 error 0.014. Figure 13b shows metric evolution across integration steps: our method reaches high Pearson correlation and low L_2 error from early steps, while the baselines require more steps to converge and plateau at higher error levels. We also conduct an ablation study on canonicalization strategies for this task, detailed in Section 7.3.1.

Table 2. Quantitative comparison on uniform blue-noise generation. Pearson correlation (higher is better) and relative L_2 error (lower is better) are computed against the ground-truth radial power spectrum over 1000 generated samples.

Method	Pearson ↑	L_2 Error ↓
Original FM (5M)	0.956	0.122
Minibatch OT (5M)	0.888	0.185
EqFM (5M)	0.867	0.198
Ours (5M)	0.994	0.049
Ours (26M)	**0.999**	**0.014**

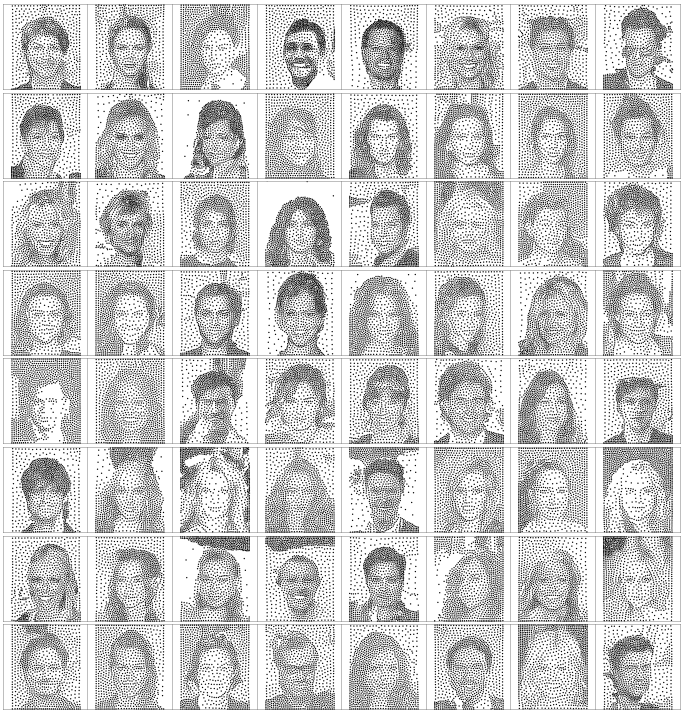


Fig. 14. **Adaptive blue-noise generation on CelebA.** Unconditionally generated face distributions using our 26M model trained on 200K adaptive blue-noise samples. Point density varies with image intensity, revealing facial features—eyes, nose, mouth, and hair contours—while maintaining local blue-noise spectral characteristics throughout.

Adaptive Blue Noise Generation Results. We extend our approach to adaptive blue-noise sampling, where point density varies spatially according to image intensity. Using the CelebA dataset [Liu et al. 2015], we generate 200K adaptive blue-noise samples by applying serial GBN to each face image, then train our 26M model for unconditional generation. As shown in Figure 14, the model successfully learns the joint distribution of facial geometries and the underlying blue-noise characteristics, generating diverse, stylized faces.

7.1.2 Minimal Surfaces (Area-Constrained). Minimal surfaces are surfaces that locally minimize area under given boundary constraints, arising naturally in soap films, biological membranes, and architectural structures [Isenberg 1992; Plateau 1873]. Given a set of anchor points defining the boundary, the minimal surface satisfies the Laplace equation with zero mean curvature. Classical computational methods solve this as a boundary-value problem through iterative optimization [Brakke 1992; Pinkall and Polthier 1993]. We reformulate this as a conditional generation task: given anchor points, directly generate boundary points that lie on the corresponding minimal surface.

Dataset Construction and Evaluation Metrics. We sample random anchor configurations on the domain boundary of a 256×256 grid and compute minimal surface boundaries using an approximate method [Israelachvili 2011] with target area fraction 0.7. Each sample consists of anchor positions as conditioning input and 256 boundary points as the target output. We conduct two experiments: (i) fixed 3-anchor configurations, and (ii) variable 3–8 anchor configurations. We evaluate generation quality using three metrics averaged over 100 samples: *area fraction error* measures deviation from the target enclosed area; *angle smoothness* quantifies boundary curve regularity via angular variation; and *uniformity CV* (coefficient of variation) assesses the evenness of point spacing along the boundary. Lower values indicate better quality for all metrics.

Fixed Anchor Count Results. Figure 6 compares 1-step and 10-step generation results for configurations with 3 anchors at random positions. Here, all methods (Original FM, Minibatch OT, EqFM, and ours) use 5M parameters. Our method produces visually accurate minimal surface boundaries in a single inference step, while the baselines tested here fail to form coherent shapes. With 10 steps, baseline methods still exhibit noticeable artifacts: scattered points, irregular spacing, and boundary distortions. Quantitative results in Table 3 confirm this observation. Specifically, with a single inference step, our method achieves area fraction error of 0.004, angle smoothness of 0.33, and uniformity CV of 0.34, while baselines show area errors exceeding 0.69 (Original FM, Minibatch OT) or poor smoothness and uniformity (EqFM). With 10 inference steps, our method further improves to area error 0.004, angle smoothness 0.08, and uniformity CV 0.08, representing an order-of-magnitude improvement over all baselines. Figure 13a shows metric evolution across inference steps: our method achieves low error from the first step and remains stable, whereas baselines improve slowly and plateau at substantially higher error levels even with 200 inference steps.

Variable Anchor Count Results. We further evaluate our method under varying anchor counts (3–8) at random boundary positions, using a conditional model architecture that generalizes across different configurations (see Section 4.1). As shown in Figure 5, our method produces smooth and accurate minimal-surface boundaries with 3 inference steps across all anchor counts and positions.

7.1.3 Diffusion-Limited Aggregation. Diffusion-limited aggregation (DLA) models fractal growth through Brownian-motion particle attachment, producing dendritic structures observed in electrodeposition, mineral formation, and biological branching [Meakin 1983b; Witten Jr and Sander 1981]. A key characteristic of DLA clusters is their fractal dimension D_f, which approaches 1.71 ± 0.01 in 2D as $N \to \infty$ [Meakin 1983a; Witten Jr and Sander 1981]. We formulate

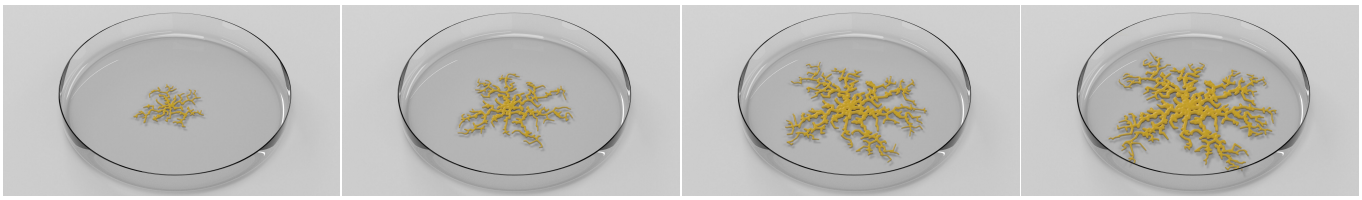


Fig. 15. Our generated DLA growth process, rendered as a growing bacterial colony in a Petri dish.

Table 3. Quantitative comparison on minimal surface generation (3 anchors, random positions). All metrics are lower-is-better, averaged over 100 samples.

Method	Area Err. ↓	Angle Smooth. ↓	Unif. CV ↓
1-step generation			
Original FM	0.700	1.974	1.123
Minibatch OT	0.689	2.011	0.906
EqFM	0.040	1.444	1.304
Ours	**0.004**	**0.330**	**0.343**
10-step generation			
Original FM	0.047	1.220	1.363
Minibatch OT	0.049	1.185	1.317
EqFM	0.042	0.901	1.272
Ours	**0.004**	**0.083**	**0.078**

DLA generation as an unconditional task where the model learns to produce realistic fractal clusters.

Dataset Construction and Evaluation Metrics. We run standard DLA simulations with $N = 1024$ particles on a 256×256 grid using a circular seed, generating 50K samples. For each sample, we record per-particle positions and attachment times as triplets (x, y, t), where t is the time step at which the particle first appears in the cluster. To visualize the DLA growth process, we first generate these (x, y, t) triplets and sort the particles by their time coordinate t. In this example, our canonicalization consists of a Hilbert sort applied to the (x, y, t) triplets. We evaluate using the fractal dimension D_f computed via the gyration method: the radius of gyration scales as $R_g(N) \sim N^{1/D_f}$, and D_f is obtained by fitting $\log R_g$ versus $\log N$. For finite $N = 1024$, the expected fractal dimension is $D_f \approx 1.58$ because of finite-size scaling effects [Meakin 1983a; Tolman and Meakin 1989], rather than the asymptotic value 1.71. Because of a slightly different dataset construction procedure and the finite-sample estimation based on 100 dataset samples, our simulated dataset has a smaller ground-truth fractal dimension of $D_f^{\mathrm{GT}} \approx 1.51$. We report the absolute error $|D_f^{\mathrm{gen}} - D_f^{\mathrm{GT}}|$ averaged over 100 generated samples.

Generation Results. Table 4 summarizes the results. Our method achieves the lowest fractal dimension error at both 10 and 200 inference steps. Figure 13c shows D_f and its error across varying inference steps: while all methods exhibit some oscillation, ours remains the most stable and converges to the lowest error. Figure 11 provides a qualitative comparison at 10 and 200 inference steps. For a fair visual comparison, we unconditionally generate 400 samples

Table 4. Fractal dimension error on DLA generation. $|D_f^{\mathrm{gen}} - D_f^{\mathrm{GT}}|$ computed via gyration method, averaged over 100 samples (lower is better).

Method	10-step ↓	200-step ↓
Original FM	0.116	0.018
Minibatch OT	0.042	0.015
EqFM	0.018	0.018
Ours	**0.011**	**0.007**

with each trained model and, for a chosen dataset example, retrieve the closest generated sample from each method using the Chamfer Distance (CD). At 10 steps, baseline methods produce scattered, non-fractal structures lacking the characteristic dendritic branching of DLA, while our method exhibits realistic fractal morphology comparable to a dataset sample. At 200 steps, all methods improve, but ours maintains the closest resemblance to the dataset sample in terms of branching density and radial structure. . Furthermore, Figure 15 visualizes the temporal evolution of our generated DLA clusters rendered as a growing bacterial colony.

7.1.4 Multilayer Thomson Problem. The Thomson problem seeks minimum-energy configurations of N electrons on a sphere under Coulomb repulsion [Smale 1998; Thomson 1904]. We extend this to a multilayer setting: particles are distributed across concentric spherical shells, interacting via pairwise Coulomb repulsion $E_{\mathrm{coul}} = \sum_{i<j} 1/|\mathbf{x}_i - \mathbf{x}_j|$, while being constrained to their respective shells by a radial potential. This models atomic shell structures and provides a challenging 3D equilibrium problem with both intra-layer and inter-layer interactions.

Dataset Construction and Evaluation Metrics. We simulate 3 concentric shells with 128 particles each (384 total), using gradient-based optimization with Coulomb forces and shell-confining springs until convergence. We generate 20K equilibrium configurations as training data. We evaluate generation quality using two metrics averaged over 100 samples: *CV average* (coefficient of variation of nearest-neighbor distances) measures spatial uniformity on each shell, and $F_{\tan}$ *RMS* (root-mean-square tangential force) quantifies deviation from force equilibrium. At a true minimum, tangential forces vanish.

Generation Results. Table 5 summarizes the results. Our method achieves the best performance on both metrics at 20 and 200 inference steps. At 20 steps, the tangential force RMS is roughly an order of magnitude lower than Original FM (4.99 vs. 102.4), suggesting

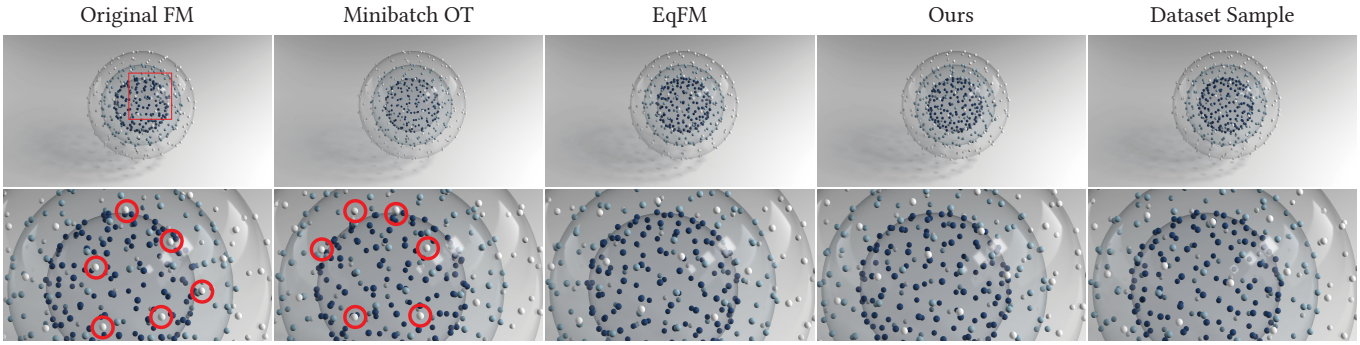


Fig. 16. **Multilayer Thomson problem generation.** Comparison of generated three-shell electron configurations (128 particles per shell). The top row shows the full configuration, and the bottom row zooms into the region indicated by the red box. Red circles mark irregular particle-spacing artifacts that remain in Original FM and Minibatch OT, while EqFM and our method produce Poisson-disk-like particle distributions on each shell that closely match the ground-truth equilibrium structure.

Table 5. Quantitative comparison on multilayer Thomson problem (3 shells × 128 particles). CV average and tangential force RMS, averaged over 100 samples (lower is better).

Method	**20-step**		**200-step**	
	CV ↓	F_{tan} ↓	CV ↓	F_{tan} ↓
Original FM	0.173	102.4	0.073	10.55
Minibatch OT	0.158	52.16	0.070	3.80
EqFM	0.103	28.61	0.075	8.11
Ours	**0.088**	**4.99**	**0.061**	**2.54**

that the generated configurations lie closer to energy minima. Figure 13d shows metric evolution across inference steps: our method converges faster and achieves lower error throughout. Figure 16 provides qualitative comparison, where our generated configurations exhibit uniform particle spacing within each shell and proper inter-shell separation, closely matching ground-truth equilibrium structures.

7.2 3D Shape Generation

We evaluate our framework on 3D shape generation tasks, testing both our canonicalization strategy for position-only generation and our geometric probability paths for joint position-normal generation. All experiments in this subsection use point clouds with $N = 2048$ points and 26M-parameter models.

7.2.1 ShapeNet Point-Cloud Generation. Following prior work, we evaluate on three ShapeNet [Chang et al. 2015] categories: airplane, chair, and car. We compare against the same baselines (Original FM, Minibatch OT, EqFM), all trained under identical settings for fair comparison. Using the evaluation protocol of Yang et al. [Yang et al. 2019], we report 1-NNA accuracy under both Chamfer Distance (CD) and Earth Mover's Distance (EMD), where values closer to 50% indicate better generation quality.

Position-Only ShapeNet Generation. Table 6 compares our method against flow-matching baselines and prior work. Results for Original FM, Minibatch OT, EqFM, and our method are from models we trained; other results are taken from respective papers with the same setting. Our method achieves the best EMD scores among flow-matching approaches across all categories (EMD is generally considered a more informative metric for global shape distribution quality than CD). Notably, we match the performance of NSOT [Hui et al. 2025] with approximately 5× fewer inference steps (200 vs. 1000), and achieve airplane EMD comparable to DiT-3D (XL) [Mo et al. 2023] using roughly 26× fewer parameters (26M vs. 675M) and 5× fewer inference steps. Figures 17, 18, and 19 show qualitative comparisons at 40 and 200 inference steps. Following Hui et al. [Hui et al. 2025], we unconditionally generate hundreds of shapes with each method and take samples generated by our model as references and, for each such sample, retrieve from every baseline the generated point cloud with the smallest Chamfer Distance (CD) to that reference, yielding shape-wise aligned comparisons. At 40 inference steps, competing baselines tend to produce less coherent shapes, whereas our model generates diverse, realistic geometries; at 200 steps, some of their samples remain less detailed and faithful than ours.

Following [Zhang et al. 2023], we further evaluate generation quality via Rendering-FID and Rendering-KID on the ShapeNet airplane category. Each generated and training shape is rendered from 10 viewpoints, and both metrics are computed between the generated and training rendering sets using Clean-FID [Parmar et al. 2022]. As shown in Table 7, our method achieves the lowest FID and KID among the flow-matching baselines.

ShapeNet Point-Cloud Generation with Encoded Normals. We further evaluate unconditional joint position-normal generation on the ShapeNet airplane category. A key advantage of our geometric probability paths is that they produce *consistently oriented* surface normals as a zero-cost byproduct of the flow, without requiring any additional network output or post-processing. For ShapeNet dataset, we first apply marching cubes to the voxelized shapes to obtain watertight meshes, so that we have consistently oriented ground-truth

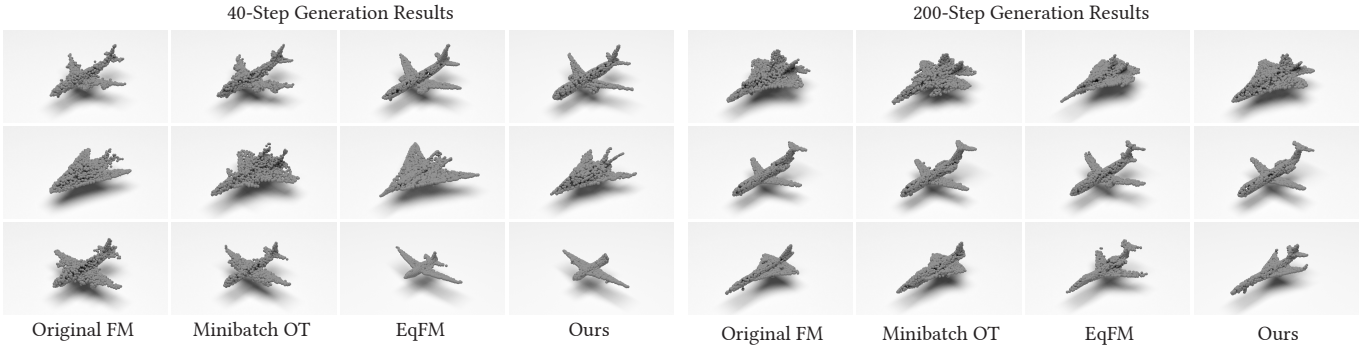


Fig. 17. **ShapeNet airplane generation.** Comparison at 40-step and 200-step inference.

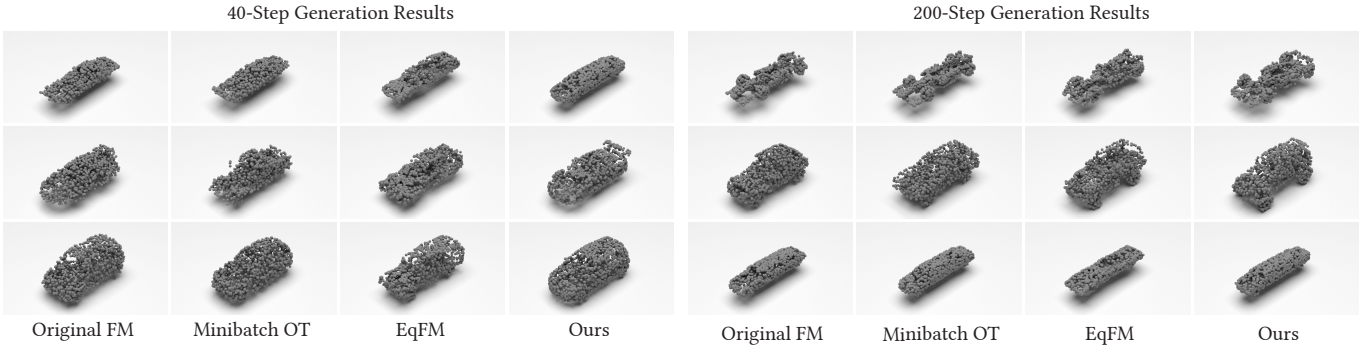


Fig. 18. **ShapeNet car generation.** Comparison at 40-step and 200-step inference.

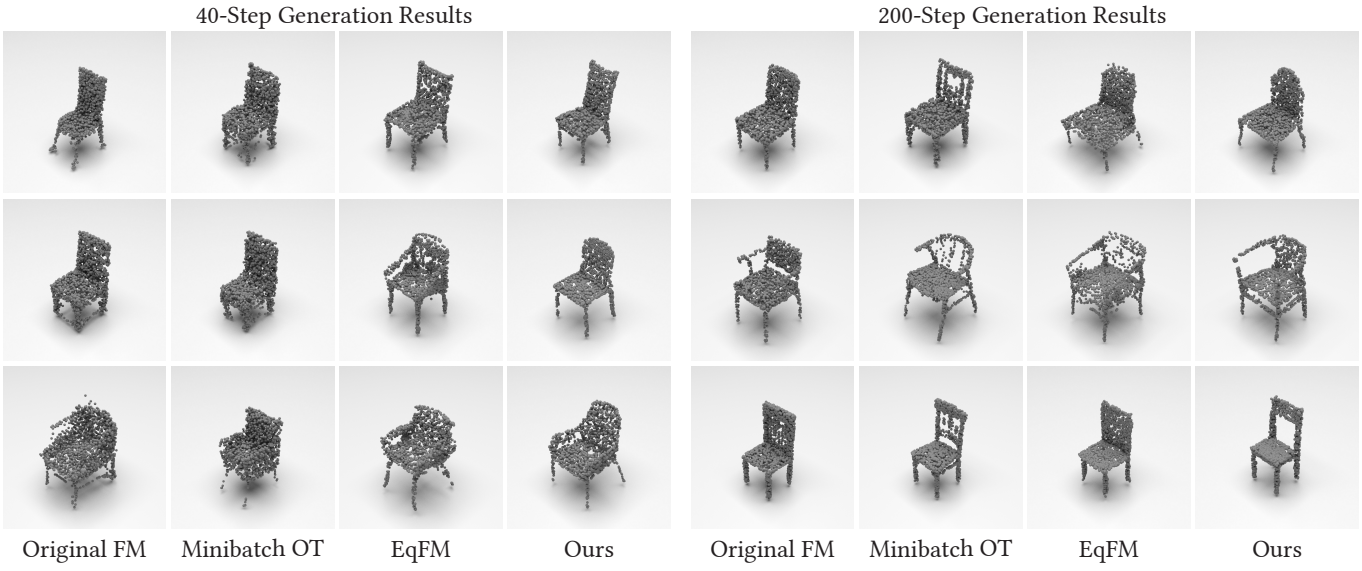


Fig. 19. **ShapeNet chair generation.** Comparison at 40-step and 200-step inference.

surface normals. Figures 20 and 21 visualize two unconditional generation processes using 200 inference steps with green line segments indicating velocity directions, which converge to surface normals at the terminal time. Figure 22 compares our generated normals against PCA-estimated normals on the same point cloud. While PCA

Table 6. Quantitative comparison on ShapeNet. 1-NNA accuracy (%) with Chamfer Distance (CD) and Earth Mover's Distance (EMD); closer to 50% is better. Two-stage latent methods first train a VAE to compress point clouds into a latent space, then train generative models in that space. †: trained by us for fair comparison; others from original papers.

Model	Method	# Params (M)	Infer Steps	Two-stage Latent	Airplane		Chair		Car	
					CD ↓	EMD ↓	CD ↓	EMD ↓	CD ↓	EMD ↓
FM	PVD-DDIM [Zhou et al. 2021]	28	100	✗	76.21	69.84	61.54	57.73	60.95	59.35
	Original FM† [Lipman et al. 2022]	25	200	✗	81.98	66.29	66.77	65.03	75.57	60.94
	Minibatch OT† [Tong et al. 2023]	25	200	✗	80.12	67.90	71.68	69.49	71.31	61.22
	Equivariant FM† [Klein et al. 2023]	25	200	✗	91.85	85.93	74.62	70.32	92.90	79.69
	NSOT [Hui et al. 2025]	-	1000	✗	**68.64**	61.85	**55.51**	57.63	59.66	53.55
	Ours	26	200	✗	69.38	**58.77**	61.93	58.38	60.65	55.39
	Ours (1000)	26	1000	✗	71.35	62.96	58.23	**55.14**	**59.54**	**53.26**
Diffusion	DPM [Luo and Hu 2021]	3.9	100	✗	76.42	86.91	60.05	74.77	68.89	79.97
	PVD [Zhou et al. 2021]	28	1000	✗	73.82	64.81	56.26	53.32	54.55	53.83
	LION [Zeng et al. 2022]	111	1000	✓	67.41	61.23	53.70	52.34	53.41	51.14
	FrePoLad [Zhou et al. 2024b]	-	1000	✓	**65.25**	62.10	52.35	53.23	51.89	50.26
	NWD [Hui et al. 2022]	31	100	✗	**59.78**	**53.84**	56.35	57.98	61.75	58.54
	3DShape2VecSet [Zhang et al. 2023]	270	18	✓	62.75	61.01	54.06	56.79	86.85	80.91
	DiT-3D (S) [Mo et al. 2023]	33	1000	✗	-	-	60.72	56.04	-	-
	DiT-3D (XL) [Mo et al. 2023]	675	1000	✗	62.35	58.67	**49.11**	**50.73**	**48.24**	**49.35**
Others	l-GAN [Achlioptas et al. 2018]	1.9	1	✓	87.30	93.95	68.58	83.84	66.49	88.78
	PointFlow [Yang et al. 2019]	1.6	var.	✓	75.68	70.74	62.84	60.57	**58.10**	56.25
	DPF-Net [Klokov et al. 2020]	3.8	var.	✓	**75.18**	**65.55**	62.00	**58.53**	62.35	**54.48**
	SoftFlow [Kim et al. 2020]	-	var.	✓	76.05	65.80	59.21	60.05	64.77	60.09
	SetVAE [Kim et al. 2021]	0.7	1	✓	75.31	77.65	**58.76**	61.48	59.66	61.48
	ShapeGF [Cai et al. 2020]	5.3	100	✓	80.00	76.17	68.96	65.48	63.20	56.53

Table 7. Rendering-FID and Rendering-KID ($\times 10^3$) on ShapeNet airplane (lower is better).

Method	FID ↓	KID ($\times 10^3$) ↓
Original FM	7.659	3.582
EqFM	11.586	6.990
Ours	**6.693**	**2.708**

can recover approximate normal directions, it cannot determine consistent orientations, leading to failures at thin structures like wings and tail fins. The top-left part additionally shows Screened Poisson reconstruction [Kazhdan et al. 2006; Kazhdan and Hoppe 2013] comparison, where PCA-based reconstruction exhibits artifacts due to these inconsistent normal orientations. We further quantify normal accuracy via unoriented angular deviation in Section 7.3.4, confirming that our method also achieves lower angular error than PCA estimation. Together, these results demonstrate that our geometric probability paths produce accurate, consistently oriented normals, enabling high-quality surface reconstruction.

7.2.2 Single-Shape Encoding. We evaluate on the single-shape encoding task proposed in Geometry Distributions [Zhang et al. 2025], where a generative model encodes a single geometry. Following this setup, we train per-shape models with our geometric probability paths on complex meshes from Thingi10k [Zhou and Jacobson 2016]. At inference time, we generate 256 batches of $N = 2048$ points, yielding about 500K points with normals using 300 inference steps, which we feed into Screened Poisson reconstruction [Kazhdan et al. 2006; Kazhdan and Hoppe 2013].

We compare against three baselines: (1) Geometry Distributions (3D) [Zhang et al. 2025], which generates positions only and estimates normals via PCA; (2) Generalized Variance Preserving (gVP) Path (3D) [Albergo and Vanden-Eijnden 2022; Chang et al. 2024; Ma et al. 2024], which, similar to our approach, interprets terminal velocities as normals but strongly relying on the assumption that the learned density collapses to a near-delta distribution around the surface; and (3) Geometry Distributions (6D), which explicitly generates 6D position-normal vectors. Figure 24 shows comparison on a challenging coral cuff mesh with thin structures. Geometry Distributions (3D) produces sparse, clustered point distributions, resulting in poor mesh quality with PCA-estimated normals. Generalized VP recovers normals from terminal velocities, but the predicted normals are noisy and often misaligned, so the reconstructed mesh exhibits pronounced artifacts. Geometry Distributions (6D) achieves results comparable to ours but requires generating 6D outputs.

Figure 23 shows additional results on diverse Thingi10k meshes, including thin structures (single tear), solid objects (dendrite, angel),

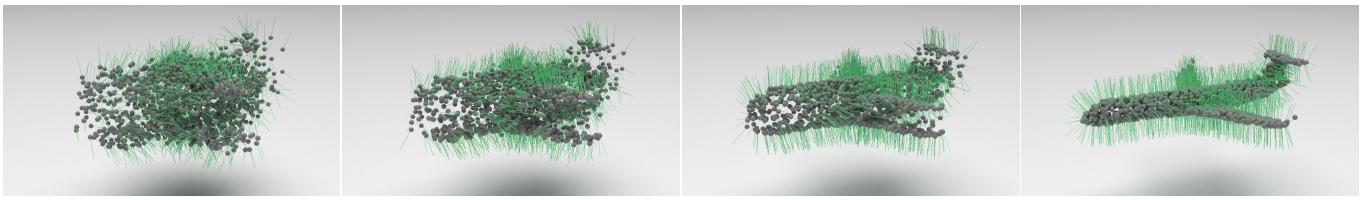


Fig. 20. **3D generation with encoded normals on ShapeNet airplane.** Green line segments show velocity directions during generation, which converge to surface normals at the terminal frame.

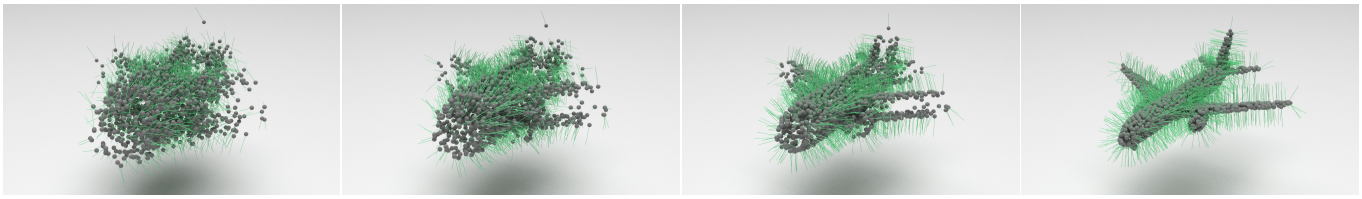


Fig. 21. **3D generation with encoded normals on ShapeNet airplane.** Additional samples demonstrating consistent normal generation across diverse airplane geometries.

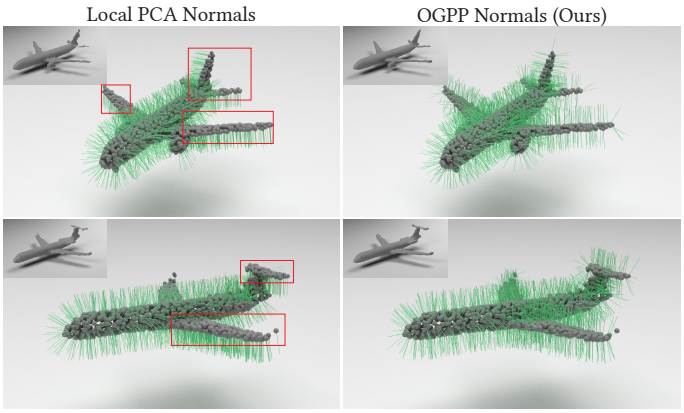


Fig. 22. **Normal comparison on ShapeNet airplane.** Top-left: Poisson reconstruction from our generated normals. Our method produces consistent, accurate normals compared to PCA-estimated normals.

and shapes with complex topology (alien egg, honeycomb jar). Left columns show generated point clouds colored by predicted normals, while right columns show meshes reconstructed with Screened Poisson. Our method consistently recovers accurate normals and supports high-fidelity reconstructions across this range of geometric complexity.

7.3 Ablation Studies

We conduct ablation studies to analyze three key design choices in our framework: (1) orbit-space canonicalization strategy and initial noise distribution, (2) particle index embeddings, and (3) geometric probability paths for normal generation.

7.3.1 Canonicalization Strategy and Initial Noise Distributions. We evaluate different canonicalization strategies and initial noise distributions on the uniform blue-noise task (Section 7.1.1) using the same 5M model and 50K training samples. Table 8 and Figure 26 summarize the results.

Table 8. Ablation study on canonicalization strategies and initial noise distributions for uniform blue-noise generation. Pearson correlation (higher is better) and L_2 error (lower is better) against ground-truth radial power spectrum.

Method	**Pearson** ↑	L_2 **Error** ↓
Canonicalized Noise	0.210	0.383
Hilbert-stratified Noise	0.896	0.184
Gaussian Noise	0.994	0.048
Scaled Gaussian Noise	0.994	0.049
Moore Curve	0.993	0.053
Z-order Curve	0.993	0.050
Hilbert Curve (Ours)	0.994	0.049
Toroidal Boundary	**0.994**	**0.046**

Several observations emerge from Table 8 and Figure 26. First, canonicalizing only the noise endpoint X_0 without sorting X_1 fails to capture blue-noise structure (Pearson 0.21), confirming our theoretical analysis that canonicalization on the X_1 side is essential. Second, Hilbert-stratified noise (second column) implements a two-sided canonicalization strategy: we sample X_0 by placing one particle at each grid-cell center, adding small jitter, and then sorting these points by their Hilbert indices, while X_1 is Hilbert-sorted in the usual way. This construction slightly improves over canonicalized noise alone (first column) but still performs markedly worse than the other configurations. Third, using Gaussian and scaled Gaussian noise (third and fourth columns) yields nearly identical power-spectrum metrics, indicating that modest changes in the noise variance have little effect on performance in this setting. Fourth, among space-filling curve orderings (Moore, Z-order, Hilbert; fifth–seventh columns), all achieve comparable performance, but we occasionally

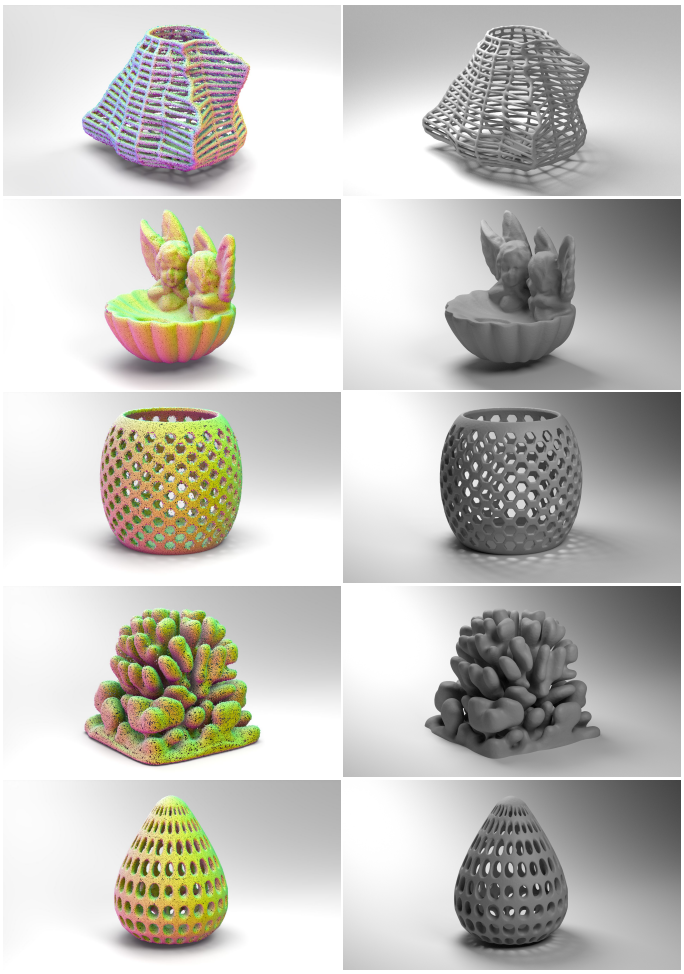


Fig. 23. **Single shape encoding.** Left: point clouds colored by predicted normals. Right: reconstructed mesh.

observe that samples generated with Z-order curves contain points that are too close to each other. We therefore adopt Hilbert ordering as our default due to its stronger locality-preserving properties. Fifth, employing a toroidal probability path (last column), which effectively imposes periodic boundary conditions so that trajectories can wrap across the domain boundary, leads to a slight additional improvement. However, this gain is marginal, we also occasionally observe point pairs that are too close, and such a path may not be equally beneficial for competing methods, so we retain the simpler linear path when comparing across baselines in Section 7.1.1.

Overall, we adopt Hilbert curve sorting with uniform noise as our standard configuration, as it offers a simple, well-standardized choice with consistently strong performance.

7.3.2 Particle Index Embedding. As shown in Figure 25, we conduct an ablation on area-constrained minimal surface generation with three anchors to disentangle the roles of orbit-space path design and per-particle identity embeddings (Figure 25). We compare four variants: vanilla flow matching in the Eulerian view, with and without index (identity) embeddings, and our OGPP framework, also with and without identity embeddings. Equipping vanilla flow matching with particle identities alone fails to recover high-quality minimal surfaces, even with more inference steps. Without per-particle identities, our method produces shapes that are only on par with vanilla flow matching. Only the full OGPP model, which combines orbit-space canonicalization with Lagrangian identity-conditioned trajectories, yields smooth, well-formed minimal surfaces in as few as one ODE step, matching the intuition from our introduction that both components are necessary to untangle mixed particle roles and straighten the learned flows.

Table 9. **Ablation study on geometric probability path design for position-normal generation**. Avg. Cos. Sim.: average cosine similarity between generated and dataset normals (higher is better). Std. Cos. Sim.: standard deviation (lower is better). 1-NNA Acc.: joint position-normal 1-NNA accuracy (closer to 50% is better).

Canon.	Hermite Degree	$\boldsymbol{n}_0$	Noise Shape	Avg. Cos. Sim. ↑	1-NNA Acc. ↓
Hilbert 6D	Cubic	$\frac{\boldsymbol{x}_1-\boldsymbol{x}_0}{\|\boldsymbol{x}_1-\boldsymbol{x}_0\|}$	Box	0.89	0.82
Hilbert 6D	Cubic	$\mathbf{0}$	Box	0.90	0.74
Hilbert 3D	Quadratic	N/A	Box	0.91	0.78
Hilbert 6D	Quadratic	N/A	Sphere	0.83	0.99
Hilbert 6D	Quadratic	N/A	Shell	0.91	0.65
Hilbert 6D	Quadratic	N/A	Box	**0.92**	**0.61**

7.3.3 Geometric Probability Path Design. We ablate geometric probability path configurations on the ShapeNet airplane category with joint position-normal generation. Our geometric probability paths involve several design choices: (1) *canonicalization dimension*, i.e., whether to sort particles using Hilbert curves in 3D position space or 6D position-normal space (joint canonicalization); (2) *Hermite degree*, where quadratic interpolation encodes only the terminal tangent (normal), while cubic interpolation additionally specifies the initial tangent $\boldsymbol{n}_0$; (3) *initial tangent* $\boldsymbol{n}_0$, which for cubic paths can be set to zero or aligned with the displacement direction $(\boldsymbol{x}_1-\boldsymbol{x}_0)/\|\boldsymbol{x}_1-\boldsymbol{x}_0\|$; and (4) *noise shape*, i.e., the distribution of initial points $\boldsymbol{x}_0$, which can be box (uniform in $[-1,1]^3$), sphere (uniform on the unit sphere), or shell (uniform in a spherical shell). Table 9 summarizes the results, evaluated by average cosine similarity between generated normals and dataset normals, its standard deviation, and joint position-normal 1-NNA accuracy.

Quadratic interpolation consistently outperforms cubic (rows 1–2 vs. 3–6), suggesting that encoding only the terminal tangent is sufficient and that additionally constraining the initial tangent over-specifies the path. Joint canonicalization, i.e., sorting in 6D position-normal space, improves over 3D sorting (row 3 vs. 6), which is consistent with our earlier analysis in Section 5.3. Noise shape has a substantial impact: sphere noise yields poor results (0.83 cosine similarity, 0.99 1-NNA), likely because particles start on a lower-dimensional manifold, whereas box noise provides full-dimensional support and achieves the best performance. For cubic paths, setting $\boldsymbol{n}_0 = \mathbf{0}$ outperforms aligning it with the displacement direction, indicating that simpler initial conditions aid optimization.

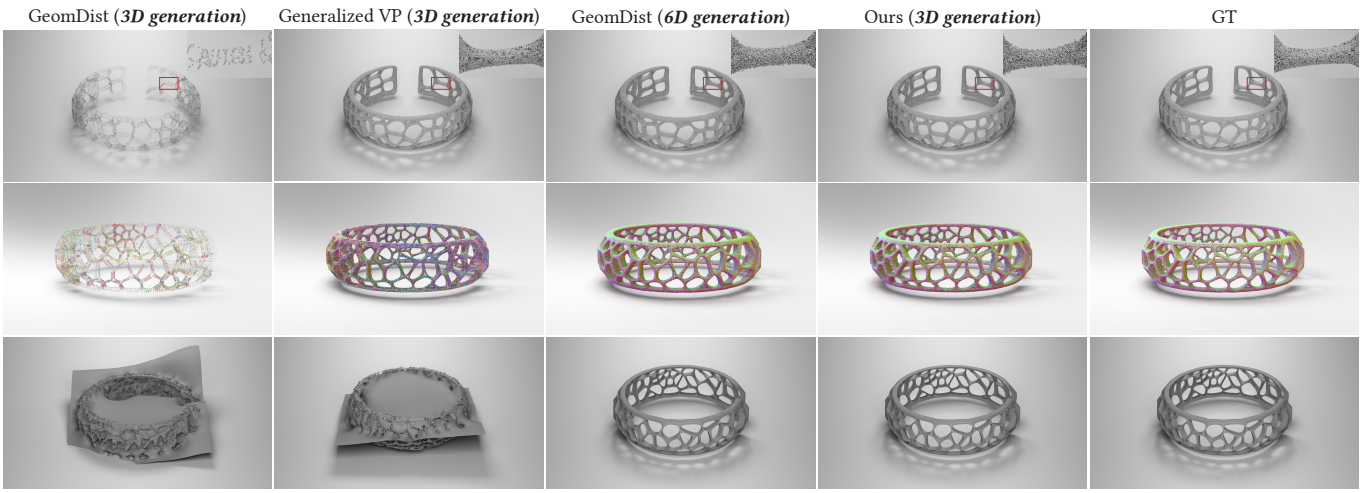


Fig. 24. **Single-shape encoding comparison on Coral Cuff.** Row 1: generated point clouds with zoomed-in details. Row 2: generated point clouds colored by normal direction. Row 3: meshes reconstructed via Screened Poisson. Geometry Distributions (3D) produces sparse, clustered points, while Generalized VP yields noisy normals on thin structures. Geometry Distributions (6D) further requires 6D outputs and higher computation. Our 3D geometric probability paths achieve quality comparable to 6D methods while maintaining the efficiency of a purely 3D generation process.

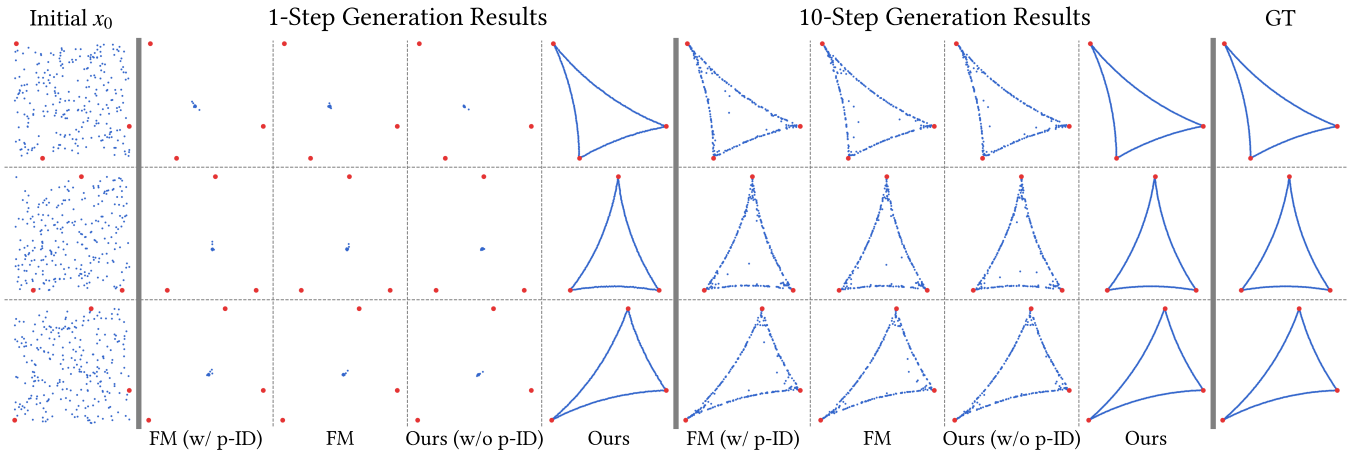


Fig. 25. **Minimal surface (area-constrained) generation ablation study (3 anchors).** Comparison of 1-step and 10-step generations; red dots indicate anchor points (conditioning locations), and ground truth (GT) is shown on the right. Without per-particle index (identity) embeddings, our method has only similar expressive power to vanilla Flow Matching (Eulerian view), while equipping vanilla Flow Matching with particle identities alone still fails to produce high-quality minimal surfaces.

Based on these results, we adopt joint canonicalization with 6D Hilbert-curve sorting, quadratic Hermite probability paths, and box noise as our default configuration.

7.3.4 Arc-length Terminal Velocity. We ablate the effect of terminal velocity magnitude on surface generation quality. As described in Section 5.2, for normal encoding, only the direction of the terminal velocity $\boldsymbol{v}_1$ is constrained, so its magnitude is a free parameter. Normalized Terminal Velocity (NTV) sets $\|\boldsymbol{v}_1\| = 1$ for all particles, while our Arc-length Terminal Velocity (ATV) scales $\|\boldsymbol{v}_1\|$ based on chord length and normal alignment (Eq. 15) to achieve more uniform speed profiles along trajectories.

Figure 9 compares NTV and ATV on the Voronoi bunny mesh. ATV reconstructs more accurate geometry, particularly visible in the zoomed-in regions (red boxes): small Voronoi cells and thin hole boundaries are clearly preserved with ATV, while NTV fails to capture these fine details. We additionally evaluate normal accuracy via the unoriented angular deviation between predicted and GT normals at the closest projected surface points on 100K generated points. The median unoriented angular error is 7.6° (ATV) vs. 12.4° (PCA-ATV), and 10.9° (NTV) vs. 17.3° (PCA-NTV). The PCA baselines differ because the two methods generate different point distributions, changing the local neighborhoods used for covariance estimation.

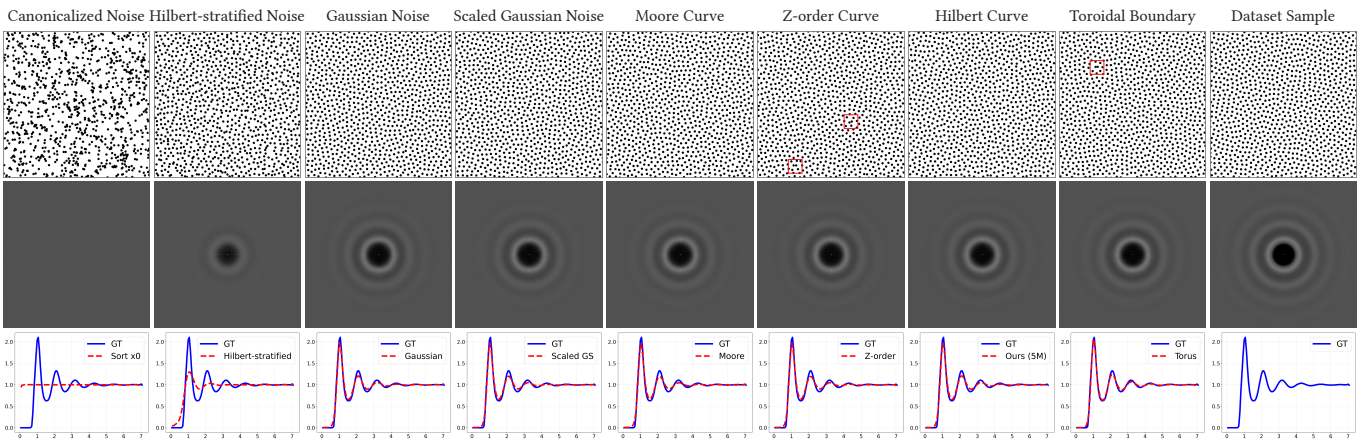


Fig. 26. **Ablation study on canonicalization strategies and initial noise distributions.** Row 1: generated point sets. Row 2: averaged 2D power spectrum. Row 3: radial power spectrum compared to ground truth. Red boxes highlight point pairs that are too close to each other.

Table 10. Inference time benchmark comparing Plain Transformer and PVCNN. Measured on a single NVIDIA H100 SXM GPU with BF16, batch size 256, averaged over 100 runs.

Model	Size	N	Dim	Params	Particle ID	Perm.-eq.	ms/samp.	samp./s
Plain Trans.	default	1024	2	5M	✓	✗	0.121	8299
	default	2048	3	5M	✓	✗	0.278	3595
	large	1024	2	26M	✓	✗	0.324	3089
	large	2048	3	26M	✓	✗	0.755	1324
	default	1024	2	5M	✗	✓	0.120	8306
	default	2048	3	5M	✗	✓	0.278	3597
	large	1024	2	25M	✗	✓	0.323	3092
	large	2048	3	25M	✗	✓	0.754	1326
PVCNN	–	1024	3	28M	✗	✓	16.952	59
	–	2048	3	28M	✗	✓	17.008	59

7.3.5 Comparison with Direct 6D Generation. We also compare our path-based normal encoding against directly generating 6D position-normal pairs via canonicalized flow matching (Canon. FM 6D). As shown in Figure 9, Canon. FM (6D) achieves comparable reconstruction quality to ATV, confirming that our geometric probability path encodes normals as effectively as explicit 6D generation. The advantage of our approach is representational economy: the flow transports only 3D positions, while normals are recovered from the terminal velocity at no extra cost.

7.3.6 Generalization: Nearest-Neighbor Analysis. To assess overfitting risks, we retrieve the nearest training neighbor under Chamfer Distance for each generated airplane. As shown in Figure 27, generated samples differ visibly from their closest matches, suggesting novel geometry synthesis rather than memorizing.

7.3.7 Permutation Equivariance and Inference Efficiency. Table 10 compares inference performance between our Plain Transformer backbone and PVCNN. Our plain transformer architecture achieves significantly higher throughput, benefiting from the efficiency of attention-based computation on modern GPUs.

Fig. 27. Generated airplanes (left of each pair) and their nearest training samples under Chamfer Distance (right of each pair). The generated shapes are visually distinct from their closest training neighbors, indicating that the model produces novel geometry rather than memorizing training data.

7.4 Mid-time analysis for Lipschitz ratio and directional cancellation

Here we empirically evaluate the Lipschitz ratio and directional cancellation introduced in Section 4.4 in a realistic training setting on our uniform blue-noise dataset.

Experimental setup. We sample $N = 500{,}000$ pairs $(\boldsymbol{x}_t^{(i)}, \boldsymbol{u}_t^{(i)})$ at the midpoint $t = 0.5$ from our uniform blue noise dataset (See Section 7.1.1), randomly select $A = 4{,}000$ anchors, and build a k-NN graph with $K = 32$ neighbors per anchor. We partition anchor–neighbor pairs into $B = 10$ equal-frequency distance bins, where bin 1 contains the closest pairs and bin 10 the most distant. Within each bin, we report summary statistics (median and 90th percentile); Details of the quantile-bin construction are given in the supplement.

At the midpoint $t = \frac{1}{2}$, the squared Lipschitz ratio is:

$$L_{ij}\left(\tfrac{1}{2}\right)^2 = 4\,\frac{\|\Delta_1^{(ij)} - \Delta_0^{(ij)}\|^2}{\|\Delta_1^{(ij)} + \Delta_0^{(ij)}\|^2}. \tag{18}$$

Large values of $L_{ij}(1/2)$ are driven by near-cancellation in the denominator, i.e., by configurations where $\Delta_0^{(ij)} \approx -\Delta_1^{(ij)}$.

Cancellation score. To quantify the degree of directional cancellation, we define the *cancellation score* for each k-NN edge (i, j):

$$s_{\text{canc}}^{(ij)} := \frac{\|(1-t)\Delta_0^{(ij)} + t\,\Delta_1^{(ij)}\|}{(1-t)\|\Delta_0^{(ij)}\| + t\|\Delta_1^{(ij)}\| + \varepsilon},$$

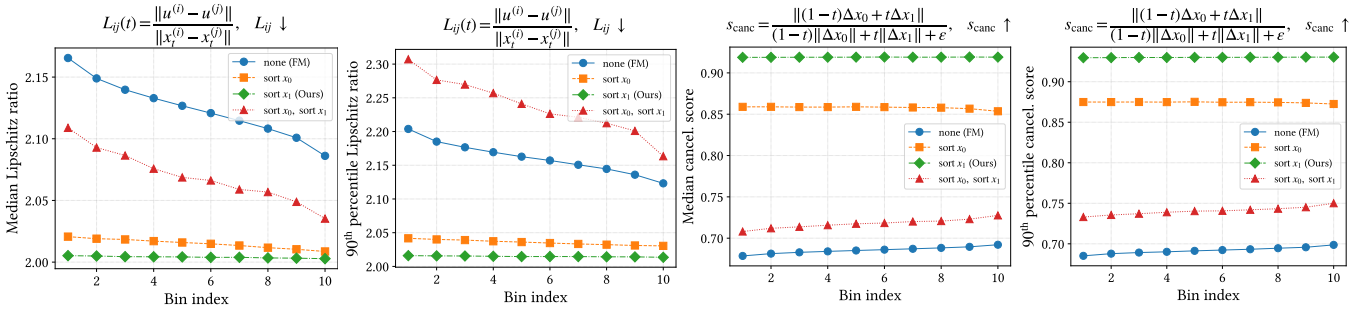


Fig. 28. **Mid-time analysis of Lipschitz ratios and directional cancellation.** From left to right: median and 90th percentile Lipschitz ratio, median and 90th percentile cancellation score at $t = 1/2$ over k-NN edges (bins ordered by distance). Lower L_{ij} and higher s_{canc} are better. Canonicalizing X_1 only ("sort x_1", Ours) yields the lowest Lipschitz ratios and highest cancellation scores, matching our directional-cancellation analysis. See more experimental details in Section 7.4.

where $\varepsilon > 0$ is a small constant for numerical stability. A score close to 1 indicates that $\Delta_0^{(ij)}$ and $\Delta_1^{(ij)}$ are roughly aligned, while a score close to 0 indicates near-perfect cancellation ($\Delta_0^{(ij)} \approx -\frac{t}{1-t}\Delta_1^{(ij)}$).

Figure 28 reports both quantities at $t = 1/2$ across four configurations (no canonicalization, canonicalize X_0 only, canonicalize X_1 only, and both). The results strongly support our theoretical analysis: one-sided canonicalization of X_1 (Ours) achieves the lowest and most stable Lipschitz ratios (median $\approx$ 2.00, P90 $\approx$ 2.02) and the highest cancellation scores ($\approx$ 0.92), indicating that most k-NN edges correspond to genuinely close pairs with minimal spurious cancellation. In contrast, two-sided canonicalization produces the highest Lipschitz ratios (P90 up to 2.30) and the lowest cancellation scores ($\approx$ 0.72), confirming that canonicalizing both endpoints contracts $\Delta_0^{(ij)}$ to the same small scale as $\Delta_1^{(ij)}$, thereby drastically increasing the frequency of directional cancellation events.

8 Discussion

Why canonicalization helps. Canonicalization fundamentally works by exploiting *structural commonalities* shared across samples. For 3D shapes, there is typically a point that is relatively "bottom-left" of the configuration; consistently assigning index 0 to that point concentrates the positional range covered by that index.

Conditions for reduced benefit. Canonicalization provides smaller gains when samples share less common structure or when the chosen ordering captures it less effectively. In the extreme case where the data has no exploitable common structure, OGPP gracefully degrades to standard flow matching. This is consistent with our experiments: on synthetic benchmarks such as minimal surfaces, where target configurations are relatively simple, canonicalization yields large improvements, while on complex real-world shapes (e.g., diverse ShapeNet categories) the gains are more moderate.

Domain-specific alternatives. When the Euclidean space-filling curve is insufficient, a more domain-appropriate canonicalization can be substituted. We already demonstrate this: for minimal surfaces (Figures 5 and 6), we use counterclockwise polygon ordering instead of Hilbert sorting. For articulated shapes, a promising direction is sorting in the spectral domain. More generally, domain knowledge about expected commonalities can be translated into a canonicalization strategy, making OGPP adaptable to diverse tasks.

9 Conclusion

In this work, we introduced *Orbit-Space Geometric Probability Paths (OGPP)*, a flow-matching framework designed for generative modeling of particle systems. While most modern generative models in graphics adopt a grid view, OGPP treats particles as persistent entities evolving through physical space, with identities, trajectories, and geometry-aware dynamics. We explored whether explicitly respecting permutation symmetry and physical semantics in the probability-path design can improve the learning problem for particle generation.

Concretely, OGPP addresses permutation symmetry through terminal canonicalization, which our analysis and experiments suggest reduces per-particle ambiguity and helps each particle assume a more consistent role. Particle index embeddings further introduce identity-aware conditioning, aiming to disentangle mixed regression targets. Our geometric probability paths show that the terminal velocity in flow matching can serve as a carrier for per-particle attributes such as surface normals. Together, these components are designed to make particle flow matching a more structured learning problem, and our experiments on the tested benchmarks indicate straighter flows and reduced inference cost. More broadly, this formulation shows that flow-based generative modeling can be designed natively for particles, and opens up new opportunities for graphics generative systems that tightly integrate sampling, geometry, and physics with particle representations. We view this work as an initial investigation into particle-centric probability-path design for flow matching, and hope it motivates further study in this direction.

Limitations. Our approach has several limitations that suggest promising directions for future research. First, our current framework operates on a fixed number of particles and relies on full attention, whose quadratic cost in particle count limits scalability to larger particle systems. Second, our geometric probability paths do not correspond to Wasserstein-2 optimal transport; lacking the geodesic property of W_2 displacement interpolation, they may induce

slightly more curved probability flows. Third, orbit-space canonicalization introduces an additional design degree of freedom and can induce useful structure in the canonical indexing (e.g., locality or ordered semantics). However, our current framework does not explicitly leverage this induced structure to encode extra information, leaving the canonicalization choice underutilized. Fourth, the benefit of canonicalization depends on how much common structure the data exhibits; on datasets with high variability, the improvements are more moderate than on structurally regular benchmarks.

Future Work. Motivated by these limitations, future work will explore sparse, hierarchical, and locality-aware architectures inspired by physical interactions, where only nearby particles exert significant influence. We also plan to extend OGPP to variable particle counts. On the probability-path side, a natural direction is to explore a richer family of geometric probability paths, e.g., higher-order or piecewise-smooth constructions that could better trade off geometric attribute encoding and transport optimality, potentially approaching W_2-consistent behavior when desired. Finally, we will investigate canonicalization as an explicit information channel by designing canonicalizers whose index order aligns with task-relevant semantics (e.g., temporal order), enabling such signals to be encoded implicitly through indices rather than introducing additional generation dimensions.

Acknowledgments

We sincerely thank the anonymous reviewers for their valuable feedback. Georgia Tech authors acknowledge NSF CAREER #2420319, IIS #2433307, OISE #2433313, IIS #2433322, ECCS #2318814 for funding support. We credit the Houdini education license for video animations.

References

Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. 2018. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*. PMLR, 40–49.

Abdalla GM Ahmed. 2024. Serial Gaussian blue noise stippling. (2024).

Abdalla GM Ahmed, Jing Ren, and Peter Wonka. 2022. Gaussian blue noise. *ACM Transactions on Graphics (TOG)* 41, 6 (2022), 1–15.

Abdalla GM Ahmed and Peter Wonka. 2020. Screen-space blue-noise diffusion of Monte Carlo sampling error via hierarchical ordering of pixels. *ACM Transactions on Graphics (TOG)* 39, 6 (2020), 1–15.

Abdalla GM Ahmed and Peter Wonka. 2021. Optimizing dyadic nets. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–17.

Michael S Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. 2023. Stochastic interpolants: A unifying framework for flows and diffusions. *arXiv preprint arXiv:2303.08797* (2023).

Michael S Albergo and Eric Vanden-Eijnden. 2022. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571* (2022).

Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T Silva. 2001. Point set surfaces. In *Proceedings Visualization, 2001. VIS'01.* IEEE, 21–29.

Michael Balzer, Thomas Schlömer, and Oliver Deussen. 2009. Capacity-constrained point distributions: A variant of Lloyd's method. *ACM Transactions on Graphics (TOG)* 28, 3 (2009), 1–8.

Josh Barnes and Piet Hut. 1986. A hierarchical O (N log N) force-calculation algorithm. *nature* 324, 6096 (1986), 446–449.

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. 2023. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127* (2023).

Mark J Bowick and Luca Giomi. 2009. Two-dimensional matter: order, curvature and defects. *Advances in Physics* 58, 5 (2009), 449–563.

Kenneth A Brakke. 1992. The surface evolver. *Experimental mathematics* 1, 2 (1992), 141–165.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Leo Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. 2024. Video generation models as world simulators. *OpenAI Blog* 1, 8 (2024), 1.

Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. 2020. Learning gradient fields for shape generation. In *European Conference on Computer Vision*. Springer, 364–381.

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. 2015. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012* (2015).

Jen-Hao Rick Chang, Yuyang Wang, Miguel Angel Bautista Martin, Jiatao Gu, Xiaoming Zhao, Josh Susskind, and Oncel Tuzel. 2024. 3D Shape Tokenization via Latent Flow Matching. *arXiv preprint arXiv:2412.15618* (2024).

Ricky TQ Chen and Yaron Lipman. 2023. Flow matching on general geometries. *arXiv preprint arXiv:2302.03660* (2023).

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

Robert L Cook. 1986. Stochastic sampling in computer graphics. *ACM Transactions on Graphics (TOG)* 5, 1 (1986), 51–72.

Fernando De Goes, Katherine Breeden, Victor Ostromoukhov, and Mathieu Desbrun. 2012. Blue noise through optimal transport. *ACM Transactions on Graphics (TOG)* 31, 6 (2012), 1–11.

Gerhard Dziuk. 1990. An algorithm for evolutionary surfaces. *Numer. Math.* 58, 1 (1990), 603–611.

T Erber and GM Hockney. 1991. Equilibrium configurations of N equal charges on a sphere. *Journal of Physics A: Mathematical and General* 24, 23 (1991), L1369–L1377.

Raanan Fattal. 2011. Blue-noise point sampling using kernel density model. *ACM Transactions on Graphics (TOG)* 30, 4 (2011), 1–12.

Daan Frenkel and Berend Smit. 2023. *Understanding molecular simulation: from algorithms to applications*. elsevier.

Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. 2018. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367* (2018).

Gaël Guennebaud and Markus Gross. 2007. Algebraic point set surfaces. In *ACM siggraph 2007 papers*. 23–es.

Stephen J Guy, Jatin Chhugani, Sean Curtis, Pradeep Dubey, Ming C Lin, and Dinesh Manocha. 2010. PLEdestrians: A Least-Effort Approach to Crowd Simulation.. In *Symposium on computer animation*. 119–128.

Thomas C Halsey. 2000. Diffusion-limited aggregation: A model for pattern formation. *Physics Today* 53, 11 (2000), 36–41.

Eric Heitz, Laurent Belcour, and Thomas Chambon. 2023. Iterative α-(de) blending: A minimalist deterministic diffusion model. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–8.

David Hilbert. [n. d.]. Über die stetige Abbildung einer Linie auf ein Flächenstück. In *Dritter Band: Analysis· Grundlagen der Mathematik· Physik Verschiedenes: Nebst Einer Lebensgeschichte*. Springer, 1–2.

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

Peter Holderrieth and Ezra Erives. 2025. An Introduction to Flow Matching and Diffusion Models. *arXiv preprint arXiv:2506.02070* (2025).

Xingchang Huang, Corentin Salaun, Cristina Vasconcelos, Christian Theobalt, Cengiz Oztireli, and Gurprit Singh. 2024. Blue noise for diffusion models. In *ACM SIGGRAPH 2024 conference papers*. 1–11.

Ka-Hei Hui, Ruihui Li, Jingyu Hu, and Chi-Wing Fu. 2022. Neural wavelet-domain diffusion for 3d shape generation. In *SIGGRAPH Asia 2022 conference papers*. 1–9.

Ka-Hei Hui, Chao Liu, Xiaohui Zeng, Chi-Wing Fu, and Arash Vahdat. 2025. Not-So-Optimal Transport Flows for 3D Point Cloud Generation. *arXiv preprint arXiv:2502.12456* (2025).

Cyril Isenberg. 1992. *The Science of Soap Films and Soap Bubbles*. Dover Publications.

Jacob N Israelachvili. 2011. *Intermolecular and surface forces*. Academic press.

Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. 2006. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, Vol. 7.

Michael Kazhdan and Hugues Hoppe. 2013. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)* 32, 3 (2013), 1–13.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–14.

Hyeongju Kim, Hyeonseung Lee, Woo Hyun Kang, Joun Yeop Lee, and Nam Soo Kim. 2020. Softflow: Probabilistic framework for normalizing flow on manifolds. *Advances in Neural Information Processing Systems* 33 (2020), 16388–16397.

Jinwoo Kim, Jaehoon Yoo, Juho Lee, and Seunghoon Hong. 2021. Setvae: Learning hierarchical composition for generative modeling of set-structured data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15059–15068.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

Leon Klein, Andreas Krämer, and Frank Noé. 2023. Equivariant flow matching. *Advances in Neural Information Processing Systems* 36 (2023), 59886–59910.

Roman Klokov, Edmond Boyer, and Jakob Verbeek. 2020. Discrete point flow networks for efficient point cloud generation. In *European Conference on Computer Vision*. Springer, 694–710.

Ruihui Li, Xianzhi Li, Ka-Hei Hui, and Chi-Wing Fu. 2021. SP-GAN: Sphere-guided 3D shape generation and manipulation. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–12.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. 2022. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747* (2022).

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. 2024. Flow matching guide and code. *arXiv preprint arXiv:2412.06264* (2024).

Xingchao Liu, Chengyue Gong, and Qiang Liu. 2022. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003* (2022).

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. 2015. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*. 3730–3738.

Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. 2019. Point-voxel cnn for efficient 3d deep learning. *Advances in neural information processing systems* 32 (2019).

Zhijian Liu, Xinyu Yang, Haotian Tang, Shang Yang, and Song Han. 2023. Flatformer: Flattened window attention for efficient point cloud transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 1200–1211.

Stuart Lloyd. 1982. Least squares quantization in PCM. *IEEE transactions on information theory* 28, 2 (1982), 129–137.

David Lopez-Paz and Maxime Oquab. 2016. Revisiting classifier two-sample tests. *arXiv preprint arXiv:1610.06545* (2016).

Shitong Luo and Wei Hu. 2021. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2837–2845.

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

Paul Meakin. 1983a. Diffusion-controlled cluster formation in 2—6-dimensional space. *Physical Review A* 27, 3 (1983), 1495.

Paul Meakin. 1983b. Formation of fractal clusters and networks by irreversible diffusion-limited aggregation. *Physical Review Letters* 51, 13 (1983), 1119.

Shentong Mo, Enze Xie, Ruihang Chu, Lanqing Hong, Matthias Niessner, and Zhenguo Li. 2023. Dit-3d: Exploring plain diffusion transformers for 3d shape generation. *Advances in neural information processing systems* 36 (2023), 67960–67971.

JR Morris, DM Deaven, and KM Ho. 1996. Genetic-algorithm energy minimization for point charges on a sphere. *Physical Review B* 53, 4 (1996), R1740.

Guy M Morton. 1966. *A computer oriented geodetic data base and a new technique in file sequencing*. International Business Machines Company.

Matthias Müller, David Charypar, and Markus Gross. 2003. Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. 154–159.

Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. 2007. Position based dynamics. *Journal of Visual Communication and Image Representation* 18, 2 (2007), 109–118.

Rahul Narain, Abhinav Golas, Sean Curtis, and Ming C Lin. 2009. Aggregate dynamics for dense crowd simulation. In *ACM SIGGRAPH Asia 2009 papers*. 1–8.

Sang Il Park and Myoung Jun Kim. 2005. Vortex fluid for gaseous phenomena. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*. 261–270.

Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. 2022. On aliased resizing and surprising subtleties in gan evaluation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11410–11420.

Songyou Peng, Chiyu Jiang, Yiyi Liao, Michael Niemeyer, Marc Pollefeys, and Andreas Geiger. 2021. Shape as points: A differentiable poisson solver. *Advances in Neural Information Processing Systems* 34 (2021), 13032–13044.

Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. 2020. Convolutional occupancy networks. In *European Conference on Computer Vision*. Springer, 523–540.

Ulrich Pinkall and Konrad Polthier. 1993. Computing discrete minimal surfaces and their conjugates. *Experimental mathematics* 2, 1 (1993), 15–36.

Joseph Plateau. 1873. *Statique expérimentale et théorique des liquides soumis aux seules forces moléculaires*. Vol. 2. Gauthier-Villars.

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

Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. 2022. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988* (2022).

Omri Puny, Yaron Lipman, and Benjamin Kurt Miller. 2025. Space group conditional flow matching. *arXiv preprint arXiv:2509.23822* (2025).

Hongxing Qin, Yi Chen, Jinlong He, and Baoquan Chen. 2017. Wasserstein blue noise sampling. *ACM Transactions on Graphics (TOG)* 36, 5 (2017), 1–13.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125* 1, 2 (2022), 3.

Craig W Reynolds. 1987. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*. 25–34.

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

Tam s Vicsek. 1992. *Fractal growth phenomena*. World scientific.

Dong Wook Shu, Sung Woo Park, and Junseok Kwon. 2019. 3d point cloud generative adversarial network based on tree structured graph convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision*. 3859–3868.

John Skilling. 2004. Programming the Hilbert curve. In *AIP Conference Proceedings*, Vol. 707. American Institute of Physics, 381–387.

Steve Smale. 1998. Mathematical problems for the next century. *The mathematical intelligencer* 20, 2 (1998), 7–15.

Yuxuan Song, Jingjing Gong, Minkai Xu, Ziyao Cao, Yanyan Lan, Stefano Ermon, Hao Zhou, and Wei-Ying Ma. 2023. Equivariant flow matching with hybrid probability transport for 3d molecule generation. *Advances in Neural Information Processing Systems* 36 (2023), 549–568.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2020. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456* (2020).

Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. 2013. A material point method for snow simulation. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 1–10.

Xiangjun Tang, Biao Zhang, and Peter Wonka. 2025a. Generative human geometry distribution. *arXiv preprint arXiv:2503.01448* (2025).

Xiangjun Tang, Biao Zhang, and Peter Wonka. 2025b. Human Geometry Distribution for 3D Animation Generation. *arXiv preprint arXiv:2512.07459* (2025).

Daniel Thalmann and Soraia Raupp Musse. 2012. *Crowd simulation*. Springer Science & Business Media.

Joseph John Thomson. 1904. XXIV. On the structure of the atom: an investigation of the stability and periods of oscillation of a number of corpuscles arranged at equal intervals around the circumference of a circle; with application of the results to the theory of atomic structure. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 7, 39 (1904), 237–265.

Susan Tolman and Paul Meakin. 1989. Off-lattice and hypercubic-lattice models for diffusion-limited aggregation in dimensionalities 2–8. *Physical Review A* 40, 1 (1989), 428.

Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. 2023. Improving and generalizing flow-based generative models with minibatch optimal transport. *arXiv preprint arXiv:2302.00482* (2023).

RA Ulichney. 1988. Dithering with blue noise. In *IEEE Proceedings*, Vol. 76. 56–79.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

David J Wales and Jonathan PK Doye. 1997. Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms. *The Journal of Physical Chemistry A* 101, 28 (1997), 5111–5116.

Peng-Shuai Wang. 2023. Octformer: Octree-based transformers for 3d point clouds. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–11.

Stephanie Wang and Albert Chern. 2021. Computing minimal surfaces with differential forms. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1–14.

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

Thomas A Witten Jr and Leonard M Sander. 1981. Diffusion-limited aggregation, a kinetic critical phenomenon. *Physical review letters* 47, 19 (1981), 1400.

Lemeng Wu, Dilin Wang, Chengyue Gong, Xingchao Liu, Yunyang Xiong, Rakesh Ranjan, Raghuraman Krishnamoorthi, Vikas Chandra, and Qiang Liu. 2023. Fast point cloud generation with straight flows. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 9445–9454.

Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. 2024. Point transformer v3: Simpler faster stronger. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 4840–4851.

Jianwen Xie, Yifei Xu, Zilong Zheng, Song-Chun Zhu, and Ying Nian Wu. 2021. Generative pointnet: Deep energy-based learning on unordered point sets for 3d generation, reconstruction and classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14976–14985.

Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. 2019. Pointflow: 3d point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE/CVF international conference on computer vision*. 4541–4550.

John I Yellott Jr. 1983. Spectral consequences of photoreceptor sampling in the rhesus retina. *Science* 221, 4608 (1983), 382–385.

Jason Yim, Andrew Campbell, Andrew YK Foong, Michael Gastegger, José Jiménez-Luna, Sarah Lewis, Victor Garcia Satorras, Bastiaan S Veeling, Regina Barzilay, Tommi Jaakkola, et al. 2023. Fast protein backbone generation with se (3) flow matching. *arXiv preprint arXiv:2310.05297* (2023).

Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, Karsten Kreis, et al. 2022. Lion: Latent point diffusion models for 3d shape generation. *Advances in Neural Information Processing Systems* 35 (2022), 10021–10039.

Biao Zhang, Matthias Nießner, and Peter Wonka. 2022. 3dilg: Irregular latent grids for 3d generative modeling. *Advances in Neural Information Processing Systems* 35 (2022), 21871–21885.

Biao Zhang, Jing Ren, and Peter Wonka. 2025. Geometry distributions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 1495–1505.

Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 2023. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Transactions On Graphics (TOG)* 42, 4 (2023), 1–16.

Chenliang Zhou, Fangcheng Zhong, Param Hanji, Zhilin Guo, Kyle Fogarty, Alejandro Sztrajman, Hongyun Gao, and Cengiz Oztireli. 2024b. Frepolad: Frequency-rectified point latent diffusion for point cloud generation. In *European Conference on Computer Vision*. Springer, 434–453.

Junwei Zhou, Duowen Chen, Molin Deng, Yitong Deng, Yuchen Sun, Sinan Wang, Shiying Xiong, and Bo Zhu. 2024a. Eulerian-lagrangian fluid simulation on particle flow maps. *ACM Transactions on Graphics (TOG)* 43, 4 (2024), 1–20.

Linqi Zhou, Yilun Du, and Jiajun Wu. 2021. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF international conference on computer vision*. 5826–5835.

Qingnan Zhou and Alec Jacobson. 2016. Thingi10K: A dataset of 10,000 3D-printing models. *arXiv preprint arXiv:1605.04797* (2016).

A Flow Matching Details

For more detailed expositions of the flow matching framework, see [Holderrieth and Erives 2025; Lipman et al. 2022, 2024].

Probability paths and velocity fields. To define suitable training targets, flow matching specifies, for each data point $\boldsymbol{x}_1 \sim p_{\text{data}}$, a *conditional probability path* $p_t(\cdot \mid \boldsymbol{x}_1)$, $t \in [0,1]$, which starts from p_{init} at $t = 0$ and collapses to a point mass at $\boldsymbol{x}_1$ at $t = 1$. Intuitively, $p_t(\cdot \mid \boldsymbol{x}_1)$ describes how noise samples are transported toward the terminal location $\boldsymbol{x}_1$. Each such path is realized by a reference *conditional velocity field* $\boldsymbol{u}_t^{\text{ref}}(\cdot \mid \boldsymbol{x}_1)$ such that the solution X_t of the induced ODE satisfies $X_t \sim p_t(\cdot \mid \boldsymbol{x}_1)$. By averaging $p_t(\cdot \mid \boldsymbol{x}_1)$ over $\boldsymbol{x}_1 \sim p_{\text{data}}$, one obtains a *marginal probability path* p_t that interpolates between p_{init} and p_{data}.

Marginalization trick. A central tool in flow matching is the *marginalization trick*, which expresses the marginal velocity field as a posterior-weighted average of conditional velocities. Let p_t be the marginal probability path induced by the conditional paths $p_t(\cdot \mid \boldsymbol{x}_1)$. Then the marginal velocity field can be written as Eq. (2), where the weighting factor is exactly the posterior of $\boldsymbol{x}_1$ given $X_t = \boldsymbol{x}$. With this choice, the ODE Eq. (1) driven by $\boldsymbol{u}_t^{\text{ref}}$ transports p_{init} along p_t and reaches p_{data} at $t = 1$.

Flow matching training. In practice, the marginal velocity field Eq. (2) is intractable to evaluate directly. Instead, flow matching trains $\boldsymbol{u}_t^\theta$ to regress onto the conditional reference velocity along the probability paths via the conditional flow matching loss Eq. (3). This objective is equivalent, up to a constant, to a marginal regression loss that matches $\boldsymbol{u}_t^\theta$ to the marginal velocity $\boldsymbol{u}_t^{\text{ref}}(\boldsymbol{x})$.

B Group Theory Details

Groups and group actions. A *group* $(G, \cdot)$ is a set G equipped with a binary operation $\cdot$ satisfying associativity, the existence of an identity element, and the existence of inverses. A group G *acts* on a set X if there is a map $G \times X \to X$, written $(g, x) \mapsto g \cdot x$, such that $e \cdot x = x$ for the identity $e \in G$ and $(g_1 \cdot g_2) \cdot x = g_1 \cdot (g_2 \cdot x)$ for all $g_1, g_2 \in G$ and $x \in X$.

Rigid motions as preprocessing. Physically, particle configurations are defined only up to global rigid motions (translations and rotations) and permutations. In all our experiments we first normalize away global pose by recentering each configuration and aligning a canonical frame (e.g., via PCA), so that the remaining symmetry is purely combinatorial: permutations of particle indices. We note that PCA-based alignment cannot resolve axis sign flips and may become ambiguous when the inertia tensor has degenerate eigenvalues (e.g., for near-isotropic shapes); we address sign ambiguity with a fixed sign convention and did not observe a noticeable impact on generation quality in our experiments.

Orthogonal representations. In our setting, we consider groups acting on Euclidean spaces via *orthogonal representations*. An orthogonal representation is a group homomorphism $\rho : G \to O(d)$, where $O(d)$ denotes the orthogonal group of $d \times d$ matrices R satisfying $R^\top R = I$. This means each group element $g \in G$ is represented by an orthogonal matrix $\rho(g)$, and the group action on $\mathbb{R}^d$ is given by $g \cdot x = \rho(g)x$.

Orbits and invariant maps. The *orbit* of a configuration $x \in \mathbb{R}^d$ under the group action is the set of all configurations reachable from x by group transformations: $\text{Orb}(x) := \{\rho(g)x : g \in G\}$. Configurations in the same orbit represent the same underlying object under symmetry transformations (here, permutations of particle indices after pose normalization). A function $f : \mathbb{R}^d \to Y$ is called *G-invariant* if $f(\rho(g)x) = f(x)$ for all $g \in G$ and $x \in \mathbb{R}^d$; that is, f is constant on each orbit.

Canonicalization (extended). A *canonicalization map* $C : \mathbb{R}^d \to \mathbb{R}^d$ selects a representative from each orbit in a G-invariant way. Formally, we require:

(1) $C(\rho(g)x) = C(x)$ for all $g \in G$ and $x \in \mathbb{R}^d$ (G-invariance);
(2) $C(x) \in \text{Orb}(x)$ for all $x \in \mathbb{R}^d$ (the output lies in the orbit of x).

The image $C(x)$ is called the *canonical representative* of x. Together, these conditions imply that $C(x_1) = C(x_2)$ if and only if x_1 and x_2 lie in the same orbit, so C induces a bijection between orbits and their canonical representatives.

C Conditional Covariance Reduction via Orbit-Space Canonicalization: Detailed Derivation

This section provides the full derivation for the conditional covariance reduction result in Section 4.2.

Setup. We consider the full configuration $\boldsymbol{X}_t = (\boldsymbol{X}_t^1, \ldots, \boldsymbol{X}_t^N) \in (\mathbb{R}^D)^N$, and a network that predicts a velocity field $\boldsymbol{u}^\theta(\boldsymbol{X}_t, t) \in (\mathbb{R}^D)^N$ for the entire configuration. For the linear path, the regression target is

$$Y := \frac{\boldsymbol{X}_1 - \boldsymbol{X}_t}{1-t} = \boldsymbol{X}_1 - \boldsymbol{X}_0. \tag{19}$$

Covariance of the regression target. Combining Eq. (19) with the Bayes-optimal velocity Eq. (5), the only source of randomness in Y given $X_t = \boldsymbol{x}$ is the endpoint X_1, and

$$\mathrm{Cov}(Y \mid X_t = \boldsymbol{x}) = \frac{1}{(1-t)^2}\,\mathrm{Cov}(X_1 \mid X_t = \boldsymbol{x})\,. \tag{20}$$

A smaller conditional covariance thus directly lowers the irreducible MSE.

Orbit-factorization details. Under the orbit-symmetry factorization Eq. (6),

$$X_1 \mid (X_t = \boldsymbol{x}) \overset{d}{=} \rho(G)\,\zeta_{\boldsymbol{x}}, \tag{21}$$

where $G \in S_N$ is a random permutation and $\rho(G)$ is its permutation representation on $(\mathbb{R}^D)^N$. Let $C : (\mathbb{R}^D)^N \to (\mathbb{R}^D)^N$ be a G-invariant canonicalization map (Section 3.3) and define $\widetilde{X}_1 := C(X_1)$. The G-invariance of C implies

$$\widetilde{X}_1 \mid (X_t = \boldsymbol{x}, G = g) \overset{d}{=} \widetilde{X}_1 \mid (X_t = \boldsymbol{x}), \tag{22}$$

so the conditional law of $\widetilde{X}_1$ no longer depends on G.

Conditional covariance decomposition. Applying the conditional law of total covariance to X_1 with respect to G gives

$$\begin{aligned}\mathrm{Cov}(X_1 \mid X_t{=}\boldsymbol{x}) &= \mathbb{E}_G\big[\mathrm{Cov}(X_1 \mid X_t{=}\boldsymbol{x}, G)\big] \\ &\quad + \mathrm{Cov}\big(\mathbb{E}[X_1 \mid X_t{=}\boldsymbol{x}, G] \,\big|\, X_t{=}\boldsymbol{x}\big).\end{aligned} \tag{23}$$

The first term is the intrinsic variability under a fixed permutation, averaged over G. The second term is a positive semidefinite covariance capturing additional variability from random G.

Applying the same decomposition to $\widetilde{X}_1$ yields

$$\begin{aligned}\mathrm{Cov}(\widetilde{X}_1 \mid X_t{=}\boldsymbol{x}) &= \mathbb{E}_G\big[\mathrm{Cov}(\widetilde{X}_1 \mid X_t{=}\boldsymbol{x}, G)\big] \\ &\quad + \underbrace{\mathrm{Cov}\big(\mathbb{E}[\widetilde{X}_1 \mid X_t{=}\boldsymbol{x}, G] \,\big|\, X_t{=}\boldsymbol{x}\big)}_{=0}.\end{aligned} \tag{24}$$

By Eq. (22), the inner conditional expectation does not depend on G, so the second term vanishes.

Trace comparison. For each fixed $G = g$, the action $\rho(g)$ is a permutation matrix on the full configuration space and is therefore orthogonal, and orthogonal transformations preserve covariance trace:

$$\mathrm{tr}\,\mathrm{Cov}(X_1 \mid X_t = \boldsymbol{x}, G = g) = \mathrm{tr}\,\mathrm{Cov}\Big(\widetilde{X}_1 \mid X_t = \boldsymbol{x}, G = g\Big). \tag{25}$$

Taking traces in Eq. (23) and Eq. (24), and using linearity of trace and expectation, we obtain

$$\begin{aligned}\mathrm{tr}\,\mathrm{Cov}(X_1 \mid X_t{=}\boldsymbol{x}) &= \mathrm{tr}\,\mathrm{Cov}(\widetilde{X}_1 \mid X_t{=}\boldsymbol{x}) \\ &\quad + \mathrm{tr}\,\mathrm{Cov}\big(\mathbb{E}[X_1 \mid X_t{=}\boldsymbol{x}, G] \,\big|\, X_t{=}\boldsymbol{x}\big) \\ &\geq \mathrm{tr}\,\mathrm{Cov}(\widetilde{X}_1 \mid X_t{=}\boldsymbol{x}).\end{aligned} \tag{26}$$

Combining Eq. (20) and Eq. (26) yields the main result Eq. (8).

D Lipschitz Ratio Analysis: Canonicalizing X_0

This section provides the detailed derivation for Section 4.4, showing why two-sided canonicalization inflates the nearest-neighbor Lipschitz ratios of the velocity field.

Setup. We view each configuration $\boldsymbol{x}_t^{(i)} = (\boldsymbol{x}_t^{(i),1}, \ldots, \boldsymbol{x}_t^{(i),N}) \in (\mathbb{R}^3)^N$ as a stacked vector in $\mathbb{R}^{3N}$.

Nearest-neighbor Lipschitz ratios. Draw i.i.d. pairs $\{(\boldsymbol{x}_0^{(i)}, \boldsymbol{x}_1^{(i)})\}_{i=1}^M$ with $\boldsymbol{x}_0^{(i)}, \boldsymbol{x}_1^{(i)} \in (\mathbb{R}^3)^N$, fix $t \in (0,1)$, and form the interpolants

$$\boldsymbol{x}_t^{(i)} = (1-t)\,\boldsymbol{x}_0^{(i)} + t\,\boldsymbol{x}_1^{(i)}, \qquad \boldsymbol{u}^{(i)} = \boldsymbol{u}\big(\boldsymbol{x}_t^{(i)}, t\big) = \frac{\boldsymbol{x}_1^{(i)} - \boldsymbol{x}_t^{(i)}}{1-t}.$$

A short calculation shows that $\boldsymbol{u}^{(i)} = \boldsymbol{x}_1^{(i)} - \boldsymbol{x}_0^{(i)}$. To quantify how $\boldsymbol{u}$ varies under small perturbations, we build a k-NN graph on $\{\boldsymbol{x}_t^{(i)}\}_{i=1}^M$, and for each edge (i, j) define the Lipschitz ratio as

$$L_{ij}(t) := \frac{\big\|\boldsymbol{u}^{(i)} - \boldsymbol{u}^{(j)}\big\|}{\big\|\boldsymbol{x}_t^{(i)} - \boldsymbol{x}_t^{(j)}\big\|},$$

where the norms are in $\mathbb{R}^{3N}$. With the per-edge differences $\Delta_0^{(ij)} := \boldsymbol{x}_0^{(i)} - \boldsymbol{x}_0^{(j)}$, $\Delta_1^{(ij)} := \boldsymbol{x}_1^{(i)} - \boldsymbol{x}_1^{(j)}$, and $\Delta_t^{(ij)} := \boldsymbol{x}_t^{(i)} - \boldsymbol{x}_t^{(j)}$, the Lipschitz ratio takes the form of Eq. (12).

One-sided vs. two-sided canonicalization. Since we focus on canonicalization at X_0, we assume X_1 has already been canonicalized so that $\Delta_1^{(ij)}$ is typically small. We compare two regimes:

- **One-sided canonicalization (ours).** Canonicalize X_1 only. Endpoint differences are $(\Delta_0^{(ij)}, \Delta_1^{(ij)})$, and $L_{ij}(t)^2$ is given by Eq. (12).
- **Two-sided canonicalization.** Also canonicalize X_0 via the same map C. Write $\widetilde{\boldsymbol{x}}_0^{(i)} := C(\boldsymbol{x}_0^{(i)})$ and $\widetilde{\Delta}_0^{(ij)} := \widetilde{\boldsymbol{x}}_0^{(i)} - \widetilde{\boldsymbol{x}}_0^{(j)}$.

By construction, canonicalization contracts the pairwise dispersion: for some $0 < \alpha_0 \leq 1$,

$$\mathbb{E}\big[\|\widetilde{\Delta}_0^{(ij)}\|^2\big] = \alpha_0^2\,\mathbb{E}\big[\|\Delta_0^{(ij)}\|^2\big],$$

with $\alpha_0 < 1$ whenever the group symmetry is non-trivial.

Directional cancellation. The key phenomenon is *directional cancellation* in the denominator of Eq. (12). When $\Delta_0^{(ij)}$ and $\Delta_1^{(ij)}$ point in approximately opposite directions and have comparable magnitudes, the denominator $(1-t)\Delta_0^{(ij)} + t\Delta_1^{(ij)}$ becomes small while the numerator $\Delta_1^{(ij)} - \Delta_0^{(ij)}$ remains large.

Since the k-NN graph is built from small values of $\|\Delta_t^{(ij)}\|$, nearest-neighbor edges are *biased* toward such cancellation events. Canonicalizing X_1 already makes $\Delta_1^{(ij)}$ small, so the denominator becomes sensitive to $\Delta_0^{(ij)}$:

- If we *keep* X_0 *uncanonicalized*, $\Delta_0^{(ij)}$ has a relatively large spread. It is statistically unlikely that $\Delta_0^{(ij)} \approx -\frac{t}{1-t}\,\Delta_1^{(ij)}$, so most nearest-neighbor edges correspond to genuinely close configurations and the denominator does not become spuriously small.
- If we also *canonicalize* X_0, $\widetilde{\Delta}_0^{(ij)}$ reaches a similar scale to $\Delta_1^{(ij)}$. It becomes much easier for the two small vectors to nearly cancel in $(1-t)\,\widetilde{\Delta}_0^{(ij)} + t\,\Delta_1^{(ij)}$, while the numerator $\Delta_1^{(ij)} - \widetilde{\Delta}_0^{(ij)}$ stays comparable. The k-NN construction then selects many edges with

tiny denominators but non-tiny numerators, yielding large $L_{ij}(t)$ and a less smooth velocity field.

E Orbit-Continuous Canonicalization and Straight Flows: Detailed Derivation

This section provides the detailed derivation for the orbit-continuous canonicalization analysis in Section 4.3.

Smoothness of endpoint distributions over the orbit space. We assume that the endpoint distribution $X_1 \mid X_t = \boldsymbol{x}$ varies smoothly over the orbit space $\boldsymbol{O}$, in the sense that nearby orbits $\mathrm{Orb}(\boldsymbol{x})$ and $\mathrm{Orb}(\boldsymbol{x}')$ induce nearby terminal endpoint distributions. Without canonicalization, this smoothness naturally lives at the level of orbits; however, a poorly behaved canonicalization map C could destroy it by introducing abrupt representative changes between nearby orbits. To avoid such pathologies, we require C to be orbit-continuous in the sense of Eq. (9). Under these conditions, the canonical means $\boldsymbol{m}(\boldsymbol{x})$ inherit orbit-Lipschitz regularity: nearby orbits induce nearby values of $\boldsymbol{m}(\boldsymbol{x})$.

Lipschitz bound derivation. Combining this orbit-Lipschitz regularity of $\boldsymbol{m}$ with Eq. (10), we obtain the local Lipschitz bound Eq. (11), where $L_{\mathrm{vel}}(t)$ is a time-dependent constant controlled by the orbit-Lipschitz constant L_{orb} (through the choice of C) and the intrinsic smoothness of the canonical means.

From orbit metric to Euclidean metric. In practice, $\boldsymbol{u}^*(\cdot, t)$ is defined on the Euclidean configuration space $(\mathbb{R}^D)^N$. When d_O is chosen as the standard orbit metric

$$d_O(\mathrm{Orb}(\boldsymbol{x}), \mathrm{Orb}(\boldsymbol{x}')) = \inf_{g \in G} \|\boldsymbol{x} - \rho(g)\boldsymbol{x}'\|,$$

we have

$$d_O(\mathrm{Orb}(\boldsymbol{x}), \mathrm{Orb}(\boldsymbol{x}')) \le \|\boldsymbol{x} - \boldsymbol{x}'\|,$$

so Eq. (11) also implies a corresponding local Lipschitz bound with respect to the Euclidean distance. Thus, the orbit-space analysis directly controls the regularity of the velocity field in the actual input space seen by the network.

F Arc-Length Terminal Velocity: Detailed Discussion

This section provides a detailed discussion of the arc-length terminal velocity (ATV) design described in Section 5.2.

Why normalized terminal velocity (NTV) is suboptimal. A naive choice is to set $\|\boldsymbol{v}_1\| = 1$ for all particles (normalized terminal velocity, NTV). However, under NTV, paths with very different chord lengths $\|\boldsymbol{x}_1 - \boldsymbol{x}_0\|$ share the same terminal speed. Distant points must either move quickly at early times and then slow down, or nearby points must start slowly and then accelerate, so different trajectories exhibit very nonuniform speed profiles in t. This makes t a poor surrogate for progress along the curve (i.e., normalized arc length), so uniform sampling in t no longer corresponds to approximately uniform sampling along the trajectory. One might try to correct this with a fixed, hand-crafted non-uniform schedule in t, but any such schedule can only compensate for a particular family of acceleration patterns (e.g., accelerate-then-decelerate trajectories) and will necessarily be suboptimal for trajectories that accelerate and decelerate in the opposite order.

Optimal speed-variance minimization. For the quadratic Hermite path Eq. (13), the speed $\|\dot\gamma(t)\|$ is the square root of a quadratic polynomial in t, and the arc length $L(\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{v}_1) = \int_0^1 \|\dot\gamma(t)\|\,\mathrm{d}t$ admits a closed-form expression in terms of $\sqrt{\cdot}$ and $\log(\cdot)$ (see the additional supplementary material for the explicit formula and derivation). Since only the *direction* of $\boldsymbol{v}_1$ is fixed by the normal, we may write $\boldsymbol{v}_1 = \alpha\hat{\boldsymbol{n}}_1$ with a scalar α, and choose α by solving a one-dimensional optimization problem that minimizes the variance of the speed profile, $\alpha^\star = \arg\min_{\alpha \in [0.5,\, 15.0]} \mathrm{Var}_{t\in[0,1]}\big(\|\dot\gamma(t;\alpha)\|\big)$, yielding, for each $(\boldsymbol{x}_0, \boldsymbol{x}_1, \hat{\boldsymbol{n}}_1)$, a Hermite trajectory whose speed over $t \in [0,1]$ is as uniform as possible.

Cheap ATV approximation. In practice, optimizing the speed variance for every particle at every training step would introduce nontrivial overhead. The ATV formula (Eq. (15)) approximates the optimal solution using only the chord length D and the chord-normal alignment S. This approximation is inexpensive (only norms and dot products), yet empirically produces trajectories with much more uniform speed profiles in t than under NTV.