Modeling Microenvironment Trajectories on Spatial Transcriptomics with NicheFlow

Kristiyan Sakalyan[*,1], **Alessandro Palma**[*,1,2,3], **Filippo Guerranti**[*,1,2,4], **Fabian J. Theis**[1,2,3,4,5], **Stephan Günnemann**[1,2,4]
[1]School of Computation, Information and Technology, Technical University of Munich
[2]Munich Centre for Machine Learning (MCML)
[3]Institute of Computational Biology, Helmholtz Munich
[4]Munich Data Science Institute (MDSI), Technical University of Munich
[5]TUM School of Life Sciences Weihenstephan, Technical University of Munich
Correspondence to: {k.sakalyan, a.palma, f.guerranti}@tum.de

Abstract

Understanding the evolution of cellular microenvironments in spatiotemporal data is essential for deciphering tissue development and disease progression. While experimental techniques like spatial transcriptomics now enable high-resolution mapping of tissue organization across space and time, current methods that model cellular evolution operate at the single-cell level, overlooking the coordinated development of cellular states in a tissue. We introduce NicheFlow, a flow-based generative model that infers the temporal trajectory of cellular microenvironments across sequential spatial slides. By representing local cell neighborhoods as point clouds, NicheFlow jointly models the evolution of cell states and spatial coordinates using optimal transport and Variational Flow Matching. Our approach successfully recovers both global spatial architecture and local microenvironment composition across diverse spatiotemporal datasets, from embryonic to brain development[1].

1 Introduction

Uncovering the principles governing tissue organization across space and time remains one of the most fundamental challenges in biology, with profound implications for evolutionary and developmental studies [1, 2]. While individual cells form the basic units of biological systems, they operate not in isolation but as integral parts of spatially organized *microenvironments*, functionally distinct neighborhoods, or *niches*, shaped by cell-to-cell interactions and extracellular components [3–5]. These spatial microenvironments influence crucial biological processes, from tumor progression to immune infiltration and tissue regeneration [6–8].

Spatial transcriptomics (ST) has transformed our ability to investigate these tissue architectures by providing single-cell resolution mapping of gene expression while preserving spatial context [9–12]. This technological breakthrough has enabled researchers to examine the molecular underpinnings of tissue organization with unprecedented detail. However, ST provides only static snapshots of inherently dynamic biological systems. *Time-resolved* spatial analysis extends ST by capturing how gene expression patterns and cellular arrangements evolve across developmental stages or experimental time [13–15]. This temporal dimension offers critical insights into the development of tissue organization in health and disease [16] that cannot be inferred from static observations alone.

[*]Equal contribution
[1]Project page: https://www.cs.cit.tum.de/daml/nicheflow

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

Despite these technological advances, modeling trajectories on time-resolved spatial slides is complicated, as no direct correspondence exists between cells across slides due to the destructive nature of acquisition practices. Moreover, current computational methods fall short in modeling the evolution of tissue organization at the level of cellular microenvironments. Most approaches infer trajectories by modeling single-cell dynamics using velocity-based models [17–19] or optimal transport between individual cells [20, 21]. While effective at capturing cell evolution, these cell-centric methods fundamentally miss the coordinated evolution of structured niches within tissues.

This limitation presents a critical research gap that we address with the following question:

How can we model the spatiotemporal evolution of cellular microenvironments preserving both local neighborhood relationships and cellular state transitions?

To address this question, we directly model the dynamics of cellular neighborhoods as cohesive units rather than focusing on isolated cell trajectories. This approach aligns naturally with tissue-scale biological processes and enables principled learning of dynamics in structured, high-dimensional, and variably sized spatial domains.

We introduce **Niche Flow** Matching (**NicheFlow**) (Fig. 1), a generative model for learning spatiotemporal dynamics of cellular niches from time-resolved spatial transcriptomics data. NicheFlow builds on recent advances in Flow Matching (FM) and Optimal Transport (OT) to operate over distributions of microenvironments, which we represent as point clouds. NicheFlow enables accurate modeling of global spatial architecture and local microenvironment composition within evolving tissues.

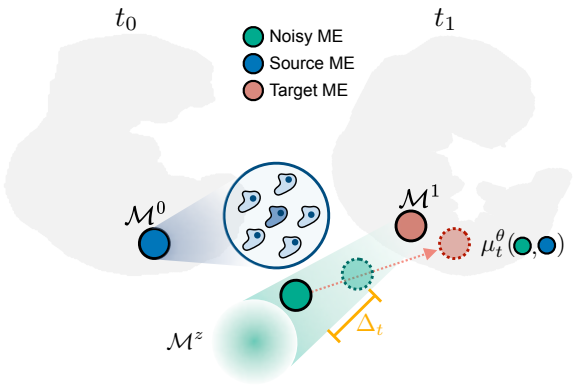


Figure 1: **Overview of NicheFlow.** At time t_1, we generate a target microenvironment $\mathcal{M}^1$ by transforming Gaussian noise $\mathcal{M}^z$ using a Variational Flow Matching model with a posterior μ_t^θ conditioned on a source microenvironment $\mathcal{M}^0$ at t_0. Source-target pairs are identified via entropic OT over pooled microenvironment coordinates and gene expression profiles.

Our contributions include:

- **A microenvironment-centered trajectory inference paradigm** that shifts from modeling individual cells in time to modeling niches as point clouds, enabling simultaneous prediction of spatial coordinates and gene expression profiles while preserving local tissue context.
- **A factorized Variational Flow Matching (VFM) approach** with distributional families (Laplace for spatial coordinates, Gaussian for gene expression) that jointly trains on spatial and cell state dynamics using a factorized loss, modeling spatial reconstruction and biological fidelity with tailored distributional assumptions.
- **A spatially-aware sampling strategy** using OT between niche representations, enabling scalable training on large tissue sections while ensuring comprehensive coverage of heterogeneous regions.

Our approach consistently outperforms baselines in recovering cell-type organization and spatial structure across embryonic, brain development, and ageing datasets. NicheFlow enables principled learning of dynamics in structured, high-dimensional, and variably sized spatial domains, a challenge with parallels in other spatiotemporal modeling domains beyond biology.

2 Related Work

NicheFlow is at the interface between generative models and spatiotemporal transcriptomic data.

FM and single-cell transcriptomics. We propose a model based on FM, a framework introduced by several seminal works [22–24]. Specifically, we adopt a variational view of the FM objective, following Eijkelboom et al. [25], but extend it to mixed-factorized distributions for point cloud generation. Our method, NicheFlow, combines FM with OT, a pairing that has proven effective in modeling cellular data [26–29]. Unlike these models, however, we focus on point clouds of

spatially-resolved transcriptomic profiles. Closest to our approach is Wasserstein FM for point cloud generation [30], applied to reconstruct cellular niches. Yet, that work does not address joint generation of spatial coordinates and cellular states, nor OT-based temporal trajectory prediction, both of which are central to our contribution.

Generative models for spatial transcriptomics. Generative models have been key to spatial tasks such as gene expression prediction from histology slides [31, 32], integration with dissociated single-cell data [32], spatial imputation [33, 34], and perturbation [35]. More recently, LUNA [36] demonstrated strong performance in predicting single-cell spatial coordinates using diffusion models [37] conditioned on transcription data. While related, our model addresses the distinct task of inferring niche trajectories, enabling the simultaneous generation of coordinates and cellular states.

Trajectory inference for spatial transcriptomics. Previous work has explored learning trajectories from spatial slides. Pham et al. [38] proposed a graph-based spatiotemporal algorithm for pseudotime inference, while others leveraged tissue-resolved transcriptomics to estimate cell velocity [17–19]. Closer to our approach, Klein et al. [20] and Bryan et al. [21] use discrete OT to link cells across time and infer the evolution of cell states from spatially-resolved gene expression. Similarly, DeST-OT [39] aligns spatial slides with semi-relaxed OT couplings, preserving transcriptomic and spatial proximity between ancestor and descendant cells, while SpaTrack [19] uses Fused Gromov-Wasserstein OT [40], balancing transcriptomic and spatial differences based on spatial autocorrelation of features. Unlike our mini-batch deep learning model, these methods do not operate on entire microenvironments and rely on exact OT at the single-cell level, resulting in limitations in scalability and generalization.

3 Background

3.1 Optimal Transport with FM

FM [23] is a generative model that transforms a source density p_0 into a target density p_1. It operates by learning a time-dependent velocity field $u_t(\boldsymbol{x})$ for $t \in [0,1]$ and $\boldsymbol{x} \in \mathbb{R}^D$, which generates a probability path $\{p_t\}_{t\in[0,1]}$. This path is constructed such that the marginals at time $t=0$ and $t=1$ match the source and target distributions, i.e., p_0 and p_1, respectively. The velocity field induces an Ordinary Differential Equation (ODE), whose solution $\phi_t(\boldsymbol{x})$ defines a *flow map* that transports samples from the source to the target distribution.

In practice, FM approximates $u_t(\boldsymbol{x})$ with a time-conditioned neural network $v_t^\theta(\boldsymbol{x})$. While the exact marginal velocity field $u_t(\boldsymbol{x})$ is intractable, it can be expressed in terms of data-conditioned velocity fields and a joint distribution $\pi(\boldsymbol{x}_0, \boldsymbol{x}_1)$ over the source and target samples $\boldsymbol{x}_0 \sim p_0$ and $\boldsymbol{x}_1 \sim p_1$:

$$u_t(\boldsymbol{x}) = \int u_t(\boldsymbol{x} \mid \boldsymbol{x}_0, \boldsymbol{x}_1) \, \frac{p_t(\boldsymbol{x} \mid \boldsymbol{x}_0, \boldsymbol{x}_1)\pi(\boldsymbol{x}_0, \boldsymbol{x}_1)}{p_t(\boldsymbol{x})} \, \mathrm{d}\boldsymbol{x}_0 \, \mathrm{d}\boldsymbol{x}_1 \,, \tag{1}$$

where $p_t(\boldsymbol{x} \mid \boldsymbol{x}_0, \boldsymbol{x}_1)$ is a pre-defined interpolating probability path. Here, we consider the tractable probability path $p_t(\boldsymbol{x} \mid \boldsymbol{x}_0, \boldsymbol{x}_1) = \delta(\boldsymbol{x} - g_t(\boldsymbol{x}_0, \boldsymbol{x}_1))$, where $g_t(\boldsymbol{x}_0, \boldsymbol{x}_1) = (1-t)\boldsymbol{x}_0 + t\boldsymbol{x}_1$ is a linear interpolation and δ denotes a Dirac delta function, representing a deterministic conditional path.

Lipman et al. [23] show that regressing the conditional field $u_t(\boldsymbol{x} \mid \boldsymbol{x}_0, \boldsymbol{x}_1)$ is equivalent to learning the marginal field $u_t(\boldsymbol{x})$ in expectation. Hence, the FM objective becomes the task of learning the velocity along the conditional probability path between any pair of source and target data points. For a linear conditional probability path, the velocity $u_t(\boldsymbol{x} \mid \boldsymbol{x}_0, \boldsymbol{x}_1)$ has a closed form, and the FM loss is:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\,(\boldsymbol{x}_0,\boldsymbol{x}_1)\sim\pi} \left[\left\| v_t^\theta\left(g_t(\boldsymbol{x}_0, \boldsymbol{x}_1)\right) - \frac{\partial}{\partial t} g_t(\boldsymbol{x}_0, \boldsymbol{x}_1) \right\|_2^2 \right]. \tag{2}$$

In practice, the coupling $\pi(\boldsymbol{x}_0, \boldsymbol{x}_1)$ is instantiated using sample pairs drawn from a mini-batch estimate. When one chooses $\pi^\star$ as the OT coupling under a squared Euclidean cost between samples from p_0 and p_1, FM approximates the dynamic OT map between source and target densities [41, 26]. Thus, the solution samples $(\boldsymbol{x}_0, \boldsymbol{x}_1) \sim \pi^\star$ from the joint distribution approximately follow:

$$\boldsymbol{x}_0 \sim p_0, \; \boldsymbol{x}_1 \sim \delta\left(\boldsymbol{x}_1 - \phi_1^\theta(\boldsymbol{x}_0)\right), \tag{3}$$

where ϕ_t^θ is the solution of the ODE with velocity field v_t^θ.

3.2 Generative OT on incomparable source and target spaces

Klein et al. [29] generalize the OT FM formulation to settings where the source and target distributions are defined on incomparable spaces and propose an approach to generative entropic OT using FM. Given a standard normal noise distribution with samples $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_D)$, the authors show that the following sampling procedure:

$$\boldsymbol{x}_0 \sim p_0,\ \boldsymbol{x}_1 \sim \delta\left(\boldsymbol{x}_1 - \phi_1^\theta(\boldsymbol{z} \mid \boldsymbol{x}_0)\right), \tag{4}$$

defines a generative model that implicitly samples from an Entropic OT (EOT) coupling, where $\phi_t^\theta(\boldsymbol{z} \mid \boldsymbol{x}_0)$ is a FM model that maps noise to target samples, conditioned on source points. To achieve this, ϕ_t^θ is trained using source-target pairs $(\boldsymbol{x}_0, \boldsymbol{x}_1)$ drawn from the EOT coupling $\pi_\epsilon^\star$, with ϵ denoting the entropic regularization parameter, which the model aims to approximate.

Crucially, this formulation enables OT between distinct source and target spaces, *as $\boldsymbol{x}_0$ does not flow directly into $\boldsymbol{x}_1$, but instead conditions the generation of target samples from noise*.

3.3 Source-conditioned VFM

Consider the source-conditioned FM formulation in Sec. 3.2. Given a conditioning source $\boldsymbol{x}_0$ and noise-based generation, the marginal field in Eq. (1) can be written as:

$$u_t(\boldsymbol{x} \mid \boldsymbol{x}_0) = \mathbb{E}_{p_t(\boldsymbol{x}_1 \mid \boldsymbol{x}, \boldsymbol{x}_0)}\left[u_t(\boldsymbol{x} \mid \boldsymbol{x}_1)\right], \tag{5}$$

where we drop the conditioning on $\boldsymbol{x}_0$ in the velocity field, as u_t is entirely determined by the target $\boldsymbol{x}_1$ when generating from noise under linear probability paths.

Since $u_t(\boldsymbol{x} \mid \boldsymbol{x}_1)$ is tractable [23], one can recast FM as a variational inference problem, following Eijkelboom et al. [25], by introducing a parameterized approximation $q_t^\theta(\boldsymbol{x}_1 \mid \boldsymbol{x}, \boldsymbol{x}_0)$ to the true posterior $p_t(\boldsymbol{x}_1 \mid \boldsymbol{x}, \boldsymbol{x}_0)$. Integrating the expected velocity in Eq. (5) over $t \in [0, 1]$ enables the generation of target points $\boldsymbol{x}_1$ from noise, conditioned on $\boldsymbol{x}_0$.

During training, the source-conditioned Variational Flow Matching (VFM) loss is:

$$\mathcal{L}_{\text{SC-VFM}}(\theta) = -\mathbb{E}_{t \sim \mathcal{U}[0,1],\, (\boldsymbol{x}_0, \boldsymbol{x}_1) \sim \pi_\epsilon^\star,\, \boldsymbol{x} \sim p_t(\boldsymbol{x} \mid \boldsymbol{x}_1)}\left[\log q_t^\theta(\boldsymbol{x}_1 \mid \boldsymbol{x}, \boldsymbol{x}_0)\right], \tag{6}$$

where $\pi_\epsilon^\star$ is an entropic OT coupling modeling the joint distribution over source and target samples, and $p_t(\boldsymbol{x} \mid \boldsymbol{x}_1)$ interpolates between target samples and noise. In the generation phase, one samples $\boldsymbol{x}_0 \sim p_0$ and noise $\boldsymbol{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_D)$, then simulates the marginal field in Eq. (5) starting from $\phi_0(\boldsymbol{x}) = \boldsymbol{z}$ to generate a target sample $\boldsymbol{x}_1$.

Crucially, under the assumption that $u_t(\boldsymbol{x} \mid \boldsymbol{x}_1)$ is linear in $\boldsymbol{x}_1$, which holds when using straight-line interpolation paths, the marginal field in Eq. (5) only depends on the posterior's first moment on $\boldsymbol{x}_1$:

$$u_t(\boldsymbol{x} \mid \boldsymbol{x}_0) = u_t\left(\boldsymbol{x} \mid \mathbb{E}_{p_t(\boldsymbol{x}_1 \mid \boldsymbol{x}, \boldsymbol{x}_0)}\left[\boldsymbol{x}_1\right]\right) \tag{7}$$

This implies that the VFM objective reduces to matching the first moment of the approximate posterior $q_t^\theta(\boldsymbol{x}_1 \mid \boldsymbol{x}, \boldsymbol{x}_0)$ to that of the true posterior $p_t(\boldsymbol{x}_1 \mid \boldsymbol{x}, \boldsymbol{x}_0)$. As a result, the approximate posterior can be chosen fully factorized under a *mean field assumption*, since each dimension can be matched independently if the mean of the true posterior is preserved; see App. C.2 for more details.

4 NicheFlow

We introduce a flow-based generative OT model to infer the temporal evolution of spatially resolved cellular microenvironments. More specifically, given a spatial microenvironment represented as a point cloud of cell states with their coordinates, NicheFlow predicts the corresponding tissue structure at a later time point. To delineate our approach, we define a list of desiderata.

Generative model on structured data. Similar to prior work [30, 36], we consider a generative model over structured point cloud data representing cellular microenvironments. This approach implicitly accounts for spatial correlations between cells, in contrast to models that study the evolution of spatial trajectories at the single-cell level [20].

Sub-regions and variable location. Crucially, for better memory efficiency, we do not consider an entire spatial slide for trajectory inference, but instead learn the dynamics of variably located

sub-regions. This design choice enables scalability and flexibility in modeling functional regions across different parts of the screened tissue.

Changes in the number of nodes. To model the temporal evolution and densification of microenvironments, we allow the source and target regions to differ in the number of nodes. We adopt a formulation similar to Sec. 3.2, implementing OT FM between non-comparable spaces.

Flexible generative models for features and coordinates. We allow flexibility in the choice of generative models for the features and coordinates. To this end, we implement the approach described in Sec. 3.3, factorizing features and coordinates into separate posteriors from different families.

4.1 Data description and problem statement

We are given a sequence of time-resolved spatial transcriptomic measurements across biological processes such as development or ageing. For simplicity, we formulate our problem in terms of two consecutive time points indexed by s, such that $s \in \{0, 1\}$, though the model can be extended to collections of more consecutive discrete temporal measurements. Each dataset at a time point is a full tissue slide that can be represented as an *attributed point cloud*:

$$\mathcal{P}_s = \{(\boldsymbol{c}_i^s, \boldsymbol{x}_i^s) \mid i = 1, \dots, N_s\}, \tag{8}$$

where $\boldsymbol{c}_i^s \in \mathbb{R}^2$ denotes the 2D spatial coordinate of cell i at time s, and $\boldsymbol{x}_i^s \in \mathbb{R}^D$ denotes its associated feature vector, typically corresponding to its gene expression profile or a low-dimensional representation thereof. Thus, each dataset is an *attributed point set* in two-dimensional space. Note that each slide can have a variable number of cells, and no direct correspondence exists between single cells across subsequent slides.

To capture spatial context beyond individual cells, we define local *microenvironments* as fixed-radius neighborhoods. Specifically, for each cell $(\boldsymbol{c}_i^s, \boldsymbol{x}_i^s)$ at time s, we construct a neighborhood $\mathcal{M}_i^s$ consisting of all neighboring cells within a spatial radius r:

$$\mathcal{M}_i^s = \left\{(\boldsymbol{c}_j^s, \boldsymbol{x}_j^s) \,\middle|\, \|\boldsymbol{c}_j^s - \boldsymbol{c}_i^s\| \leq r\right\}. \tag{9}$$

Let $\{\mathcal{M}_i^0\}_{i=1}^{N_0}$ and $\{\mathcal{M}_j^1\}_{j=1}^{N_1}$ be collections of source and target microenvironments at consecutive time points. Our goal is to train a parameterized flow model ϕ_t^θ, with $t \in [0, 1]$, that generates target microenvironments conditioned on source point clouds. Specifically, to sample a target microenvironment with k cells conditioned on a variably sized source $\mathcal{M}^0$, we define sampling as:

$$\mathcal{M}^z = \{(\boldsymbol{c}_i^z, \boldsymbol{z}_i) \mid \boldsymbol{c}_i^z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_2),\ \boldsymbol{z}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_D),\ \forall i = 1, \dots, k\}, \tag{10}$$

$$\mathcal{M}^1 = \phi_1^\theta(\mathcal{M}^z \mid \mathcal{M}^0), \tag{11}$$

where $\mathcal{M}^z$ is a point cloud composed of noisy coordinates and features, and $\mathcal{M}^1$ is a generated prediction for the evolution of $\mathcal{M}^0$ at the next time point. As explained in Sec. 3.1 and Sec. 3.2, we want our generative model to parameterize some notion of optimal entropic coupling $\pi_\epsilon^\star$ between microenvironments across slides (see Sec. 4.2).

4.2 OT formulation

To train the flow map ϕ_t^θ to perform conditional EOT, we define a cost function that induces an optimal entropic coupling $\pi_\epsilon^\star$ between source and target point clouds. This coupling is used to sample pairs of source and target microenvironment mini-batches during training. While there is no established notion of optimal cost in this setting, we propose to compute OT using source and target microenvironment representations based on the weighted average of features and coordinates.

Specifically, we compute a pooled representation for each microenvironment in the source and target slides by averaging spatial coordinates and gene expression features, weighted by a tunable hyperparameter $\lambda \in [0, 1]$ that balances spatial versus cellular state information:

$$\bar{\boldsymbol{m}}_i^s = \left[\frac{1-\lambda}{|\mathcal{M}_i^s|} \sum_{(\boldsymbol{c}_j^s, \boldsymbol{x}_j^s) \in \mathcal{M}_i^s} \boldsymbol{c}_j^s \quad \| \quad \frac{\lambda}{|\mathcal{M}_i^s|} \sum_{(\boldsymbol{c}_j^s, \boldsymbol{x}_j^s) \in \mathcal{M}_i^s} \boldsymbol{x}_j^s\right], \tag{12}$$

where $\|$ denotes concatenation, $\bar{\boldsymbol{m}}_i^s \in \mathbb{R}^{2+D}$, and $s \in \{0, 1\}$. We then apply EOT on the sets $\{\bar{\boldsymbol{m}}_i^0\}_{i=1}^{N_0}$ and $\{\bar{\boldsymbol{m}}_j^1\}_{j=1}^{N_1}$ using a squared Euclidean cost and regularization parameter ϵ, yielding the

coupling $\pi^\star_{\epsilon,\lambda}$. During training, we sample matched pairs $(\mathcal{M}^0, \mathcal{M}^1) \sim \pi^\star_{\epsilon,\lambda}$ computed over mini-batches, and use them as supervision for learning the conditional generative model. A higher value of λ prioritizes feature similarity, while lower values favor proximity in coordinates (see Fig. 12).

4.3 Mixed-factorized VFM

Once we have established a strategy for performing mini-batch OT, we proceed to describe our approach for learning the flow model ϕ^θ_t and simulating Eq. (11). To this end, we adopt a variant of VFM (Sec. 3.3), originally developed for graph generation, and adapt it to our point cloud setting.

In line with Sec. 3.2 and 3.3, we delineate an objective to train a parameterized, source-conditioned posterior over target point clouds $q^\theta_t(\mathcal{M}^1 \mid \mathcal{M}, \mathcal{M}^0)$, where $\mathcal{M}^0$ and $\mathcal{M}^1$ represent source and target niches, and $\mathcal{M}$ denotes a noisy point cloud at interpolation time t. Importantly, our posterior comes with the following characteristics: **(i)** the posterior is factorized across the single points in a point cloud; **(ii)** the posterior is factorized across cellular features and coordinate dimensions; and **(iii)** the family of posteriors can be chosen differently between cellular features and coordinates.

Following **(i)**, we model the variational distribution over $\mathcal{M}^1$ by factorizing it across individual points $(\boldsymbol{c}_1, \boldsymbol{x}_1) \in \mathcal{M}^1$. Moreover, we tackle **(ii)** and **(iii)** using the mean-field VFM assumption, modeling cellular state and positions separately (see App. C.1 and C.2 for theoretical justifications):

$$q^\theta_t(\mathcal{M}^1 \mid \mathcal{M}, \mathcal{M}^0) = \prod_{(\boldsymbol{c}_1, \boldsymbol{x}_1) \in \mathcal{M}^1} \left(\prod_{k=1}^{2} f^\theta_t(\, c^k_1 \mid \mathcal{M}, \mathcal{M}^0) \prod_{d=1}^{D} r^\theta_t(x^d_1 \mid \mathcal{M}, \mathcal{M}^0) \right). \tag{13}$$

Here, f^θ_t and r^θ_t denote distinct approximate posterior families for cellular states (x^d_1) and spatial positions (c^k_1), respectively. We use a Laplace distribution for f^θ_t due to its concentration around the mean, which supports precise modeling of coordinate features, while a Gaussian distribution is used for r^θ_t.

As explained in Sec. 3.3, if one uses FM with straight probability paths, only the first moment of q^θ_t is required to simulate the generative field in Eq. (5). Therefore, the posterior is replaced by a time-dependent predictor $(\bar{\boldsymbol{f}}^\theta_t, \bar{\boldsymbol{r}}^\theta_t) = \mu^\theta_t(\mathcal{M}, \mathcal{M}^0)$ of the mean features $\bar{\boldsymbol{r}}^\theta_t$ and coordinates $\bar{\boldsymbol{f}}^\theta_t$ of the target point clouds, learned minimizing the following loss:

$$\mathcal{L}_{\text{NicheFlow}}(\theta) = \mathop{\mathbb{E}}_{\substack{t \sim \mathcal{U}[0,1] \\ (\mathcal{M}^0, \mathcal{M}^1) \sim \pi^\star_{\epsilon,\lambda} \\ \mathcal{M} \sim p_t(\mathcal{M} \mid \mathcal{M}^1)}} \left[\sum_{(\boldsymbol{c}_1, \boldsymbol{x}_1) \in \mathcal{M}^1} \left(\|\boldsymbol{c}_1 - \bar{\boldsymbol{f}}^\theta_t\|_1 + \frac{1}{2} \|\boldsymbol{x}_1 - \bar{\boldsymbol{r}}^\theta_t\|_2^2 \right) \right]. \tag{14}$$

We derive the objective in App. C.4 and provide algorithms in App. E. Here, μ^θ_t inputs a noisy point cloud $\mathcal{M}$ and a source $\mathcal{M}^0$ and provides a mean prediction vector for coordinates and feature dimensions, respectively indicated as $\bar{\boldsymbol{f}}^\theta_t$ and $\bar{\boldsymbol{r}}^\theta_t$. We implement it as a point cloud transformer (see Sec. 4.4).

We highlight a crucial aspect about Eq. (14). While the single dimensions are fully factorized in the predictions from μ^θ_t, every feature's mean is a function of the whole noisy point cloud $\mathcal{M}$ as well as the target microenvironment $\mathcal{M}^0$. In other words, the predictions exploit structural information in the point cloud to predict the individual posterior mean of each dimension. Like most approaches, our method has modeling limitations that we outline in App. B.

4.4 Backbone architecture: Microenvironment transformer

To parameterize the conditional posterior mean μ^θ_t from Sec. 4.3, we use a *permutation-equivariant* transformer architecture designed for point clouds with variable size.

Encoder–decoder structure. The model follows an encoder-decoder layout, processing the source microenvironment $\mathcal{M}^0$ via the encoder and predicting the posterior mean from a noisy target $\mathcal{M} \sim p_t(\cdot \mid \mathcal{M}^1)$ via the decoder conditioned on the encoder's output.

Input embeddings. Each point is represented by features $\boldsymbol{x} \in \mathbb{R}^D$ and spatial coordinates $\boldsymbol{c} \in \mathbb{R}^2$, embedded separately and concatenated. Time t is encoded using sinusoidal embeddings, linearly projected and broadcast across points.

Cross-attention conditioning. The encoder processes the embedded source $\mathcal{M}^0$ microenvironment using self-attention. The decoder operates on the noisy target $\mathcal{M}$ using self-attention, followed by

Table 1: Performance comparison across three biological datasets for SPFlow, RPCFlow and NicheFlow. Models are trained using the Conditional Flow Matching (CFM), Gaussian VFM (GVFM), or Gaussian-Laplacian VFM (GLVFM) strategies. Results are reported as mean ± standard deviation over five evaluation runs on mouse embryonic development (MED), axolotl brain development (ABD), and mouse brain aging (MBA). For all experiments, we use a fixed value of $\lambda = 0.1$, enabling spatial location preservation across time.

		MED			ABD			MBA		
Model	Obj.	1NN-F1 ↑	PSD ↓ (10^2)	SPD ↓ (10^2)	1NN-F1 ↑	PSD ↓ (10^2)	SPD ↓ (10^2)	1NN-F1 ↑	PSD ↓ (10^2)	SPD ↓ (10^2)
LUNA	—	0.540 ± 0.004	—	—	0.331 ± 0.003	—	—	0.222 ± 0.000	—	—
SPFlow	CFM	0.272 ± 0.0011	1.681 ± 0.0087	0.602 ± 0.0013	0.190 ± 0.0005	2.494 ± 0.0051	1.119 ± 0.0037	0.205 ± 0.0003	1.836 ± 0.0022	0.824 ± 0.0006
SPFlow	GVFM	0.259 ± 0.0009	2.383 ± 0.0082	0.582 ± 0.0009	0.175 ± 0.0010	3.373 ± 0.0103	1.104 ± 0.0023	0.181 ± 0.0001	2.585 ± 0.0029	0.834 ± 0.0011
SPFlow	GLVFM	0.251 ± 0.0008	2.249 ± 0.0114	0.592 ± 0.0015	0.173 ± 0.0013	2.870 ± 0.0238	1.093 ± 0.0037	0.195 ± 0.0005	2.320 ± 0.0009	0.853 ± 0.0008
RPCFlow	CFM	0.546 ± 0.0012	0.981 ± 0.0024	0.564 ± 0.0015	0.524 ± 0.0020	**2.051 ± 0.0039**	1.015 ± 0.0036	0.271 ± 0.0004	**1.543 ± 0.0016**	0.810 ± 0.0010
RPCFlow	GVFM	0.503 ± 0.0013	1.155 ± 0.0044	0.578 ± 0.0007	0.477 ± 0.0008	2.260 ± 0.0077	1.036 ± 0.0031	0.249 ± 0.0003	1.753 ± 0.0020	0.784 ± 0.0010
RPCFlow	GLVFM	0.586 ± 0.0016	0.979 ± 0.0021	0.586 ± 0.0012	0.554 ± 0.0007	2.053 ± 0.0044	1.038 ± 0.0025	0.265 ± 0.0004	1.723 ± 0.0015	0.779 ± 0.0011
NicheFlow	CFM	0.609 ± 0.0030	0.979 ± 0.0228	0.402 ± 0.0036	0.604 ± 0.0018	2.086 ± 0.0058	**0.568 ± 0.0030**	0.283 ± 0.0003	1.557 ± 0.0014	0.556 ± 0.0028
NicheFlow	GVFM	0.596 ± 0.0027	0.991 ± 0.0137	0.406 ± 0.0025	0.574 ± 0.0015	2.220 ± 0.0107	0.594 ± 0.0046	0.268 ± 0.0003	1.661 ± 0.0033	**0.531 ± 0.0010**
NicheFlow	GLVFM	**0.664 ± 0.0014**	**0.883 ± 0.0094**	**0.398 ± 0.0023**	**0.628 ± 0.0013**	2.079 ± 0.0043	0.576 ± 0.0055	**0.285 ± 0.0003**	1.554 ± 0.0021	0.532 ± 0.0009

cross-attention to condition on the encoder's outputs. The cross-attention mechanism allows each target point to attend to all source points.

Output projection. The decoder outputs are linearly projected to yield posterior mean estimates $(\bar{\boldsymbol{f}}_t^\theta, \bar{\boldsymbol{r}}_t^\theta)$ for expression and coordinates.

5 Experiments

We propose quantitative and qualitative evaluations of our algorithm. Quantitatively, we test whether source-conditioned samples generated by our model preserve the biological structure and shape of future tissue states. Qualitatively, we demonstrate that our approach accurately captures compositional shifts in substructural components and developmental trajectories across time.

5.1 Quantitative evaluation

Our first research question is to assess the impact of two core modeling choices in NicheFlow: **(i)** learning trajectories over spatial microenvironments, rather than independently for each cell and **(ii)** restricting source and target point clouds to spatially co-localized neighborhoods of cells, instead of sampling them randomly across the slide.

We use NicheFlow and baseline FM approaches that do not incorporate **(i)** and **(ii)** to simulate spatial trajectories conditioned on early time point observations. Assuming that spatial arrangements and the biological composition at later stages evolve from earlier slides, the global correspondence between predicted and true slides indicates the quality of the generative trajectory.

5.1.1 Training setup

Datasets. We assess model performance across three spatiotemporal datasets: **(i)** Mouse embryogenesis [20, 8] and **(ii)** the axolotl brain development [42], two Stereo-seq datasets profiling the spatially-resolved cellular development of a mouse embryo and axolotl brain across three (E9.5, E10.5 and E11.5) and five time points, respectively. We also consider the **(iii)** mouse brain ageing dataset [43], profiled with MERFISH [44] across twenty time points (see Apps. F.1 and F.2).

Dataset construction. For each dataset and time point $s \in \mathcal{S}$, we construct a set of cellular microenvironments by applying the fixed-radius neighborhood definition introduced in Sec. 4.1. Each microenvironment $\mathcal{M}_i^s$ is centered at cell i and contains all cells within a fixed radius r. This results in:

$$\boldsymbol{\mathcal{M}}^s := \{\mathcal{M}_i^s \mid i = 1, \ldots, N_s\},$$

where N_s is the number of cells in the tissue at time s, and each $\mathcal{M}_i^s$ is an attributed point cloud encoding both spatial and gene expression information. We standardize coordinates for cross-time comparability and reduce the normalized gene expression to its top 50 Principal Components (PC).

Batching. We train NicheFlow with mini-batches of source and target cellular point clouds. To ensure spatial diversity during training, we sample individual batches uniformly from within discrete

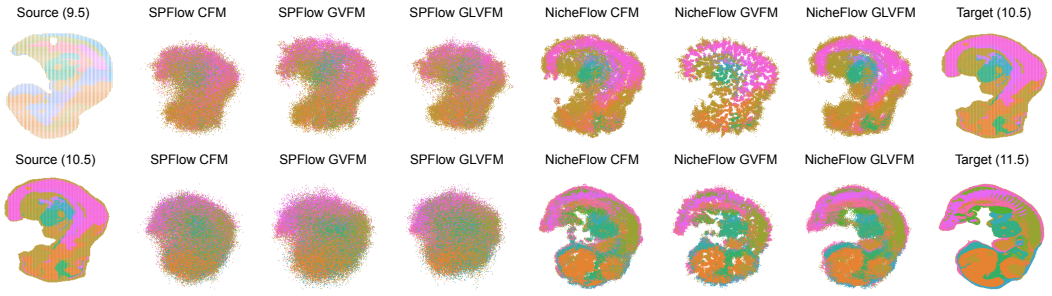


Figure 2: Qualitative comparison of generated samples on the embryonic development dataset (9.5-11.5 days). We show source and target samples alongside predictions from SPFlow and NicheFlow with different objectives.

regions of the slides computed with K-Means clustering over the 2D coordinates (see Fig. 13 for a visualization with different K values). From these regions, we collect M source and target microenvironments and resample $N \leq M$ matching pairs from the entropic OT coupling $(\boldsymbol{\mathcal{M}}^0, \boldsymbol{\mathcal{M}}^1) \sim \pi^\star_{\epsilon,\lambda}$ as described in Sec. 3.1, where $\boldsymbol{\mathcal{M}}^0$ and $\boldsymbol{\mathcal{M}}^1$ denote the sampled sets (see Sec. 4.2 for details on our OT coupling).

Evaluation data. For consistent and reproducible evaluation, we discretize each tissue into a fixed 2D grid and define evaluation microenvironments as fixed-radius neighborhoods around the nearest cells to each grid point. This guarantees full spatial coverage and ensures deterministic comparison across methods. See App. F.5 for details.

Multiple time-point. NicheFlow predicts piecewise trajectories between subsequent time points. Instead of learning one flow for each couple of subsequent slides, we train a single model with additional conditioning on source and target labels (see App. F.6).

5.1.2 Quantitative evaluation metrics

Spatial structure. We quantify coordinate generation accuracy using two asymmetric distance metrics. The *point-to-shape distance* (PSD) measures how far predicted coordinates deviate from the true structure by computing the mean squared distance from each generated point to its nearest ground truth counterpart. In contrast, the *shape-to-point distance* (SPD) evaluates how well the generated points cover the target region by averaging the squared distance from each ground truth point to its nearest generated point (see App. F.3 for a mathematical formulation of the metrics).

Cell-type organization. To assess how well the model reconstructs the spatial organization of different cell types, we use a *1-nearest-neighbor* (1NN) classification setup. Since the model generates only gene expression profiles and spatial coordinates, we assign cell type labels to generated cells using a classifier trained on ground truth gene expression data (see App. F.4). Each predicted cell is then matched to its nearest real cell, and we report the weighted F1 score (1NN-F1).

5.1.3 Models and results

Baselines. We compare against what we call *SPFlow* (Single-Point Flow), a standard FM-based model that predicts temporal trajectories across slides at a single-cell level using an MLP-based velocity field. We also consider *RPCFlow* (Random Point Cloud Flow), which has the same backbone as NicheFlow, but conditions on randomly sampled point clouds instead of radius neighborhoods. Additionally, we include *LUNA* [36], a diffusion model for spatial reconstruction from dissociated cells. Note that LUNA does not model temporal dynamics and only generates coordinates from noise with their respective biological annotations. Therefore, we use it as a reference for spatial generation accuracy via the 1NN-F1 metric rather than a proper baseline.

Ablations. We assess different training objectives by comparing standard Conditional Flow Matching (CFM) [26] with two variational formulations modeling posteriors over coordinates and features: Gaussian-only (GVFM) and Gaussian-Laplace (GLVFM). The former uses Gaussian posteriors for coordinates and features. The latter uses the factorized formulation in Sec. 4.3. For the point-cloud-based methods, we use a fixed value of $\lambda = 0.1$ and sampled batches of 64 regions chosen from

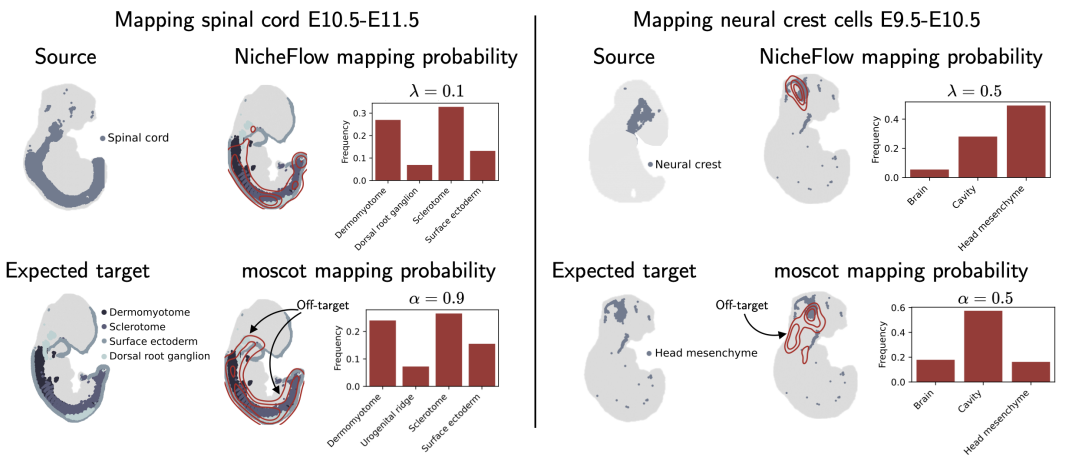


Figure 3: Left and right panels show the mapping of spinal cord (E10.5 to E11.5) and head neural crest cells (E9.5 to E10.5). In each panel, the left column shows source cells and expected targets, and the right column shows density contours of the most likely mapped regions. Bar plots display transition probabilities to the most likely descendant cell types. For NicheFlow, contours represent the proportion of samples in generated point clouds assigned to real cell coordinates across 10 samples.

the ablations in App. D.5 and D.6. We present results with a single value for λ as this experiment compares different models on reconstructing fixed tissue structures over time, for which we enforce spatial preservation (see Sec. 4.2). However, we also include model comparisons across multiple values of λ in Tab. 3 for completion.

Results. Our quantitative evaluation (Tab. 1) demonstrates that *NicheFlow* trained with the GLVFM objective consistently achieves strong performance across both spatial and semantic metrics. It outperforms all baselines in reconstructing spatial structure (PSD, SPD) and cell-type organization (1NN-F1) on developmental datasets, while remaining competitive on ageing data. These results highlight the importance of structured microenvironment modeling for capturing the spatiotemporal dynamics of complex tissues. We complement our quantitative results visually in Fig. 2 and Fig. 6 and 10 in the Appendix, where we show that SPFlow fails to capture tissue-level organization, producing blurry and spatially incoherent samples. In contrast, NicheFlow generates predictions that preserve spatial structure and cell-type organization, despite learning only from local microenvironments. In App. D.4 and App. D.7, we additionally demonstrate that NicheFlow produces conditionally consistent outputs with the source, while RPCFlow generates very diffused mappings across the slide, failing to preserve spatial consistency across time.

5.2 Qualitative evaluation and biological analysis

We explore the capabilities of NicheFlow on the spatial trajectory inference task through qualitative and biological assessments. Specifically, we focus on validating whether our model captures compositional changes within fixed spatial structures and developmental trajectories.

Experimental setup. We train the model as described in Sec. 5.1.1. Using the mouse embryonic dataset [8], we select specific microenvironments as source niches for which we want to study the trajectory over time. Specifically, we propose two scenarios for the application of NicheFlow depending on the choice of the OT parameter λ (see Sec. 4.2):

1. *Compositional changes in fixed structures across time*. We choose the evolution of the *spinal cord* of the embryo from E10.5 to E11.5 as an example and set a low value of λ to prioritize the preservation of the spatial location in the trajectory.
2. *Spatial and cellular development of immature cells*. Developing cells may displace to different areas of the embryo, requiring a higher value of λ to account for gene expression. As a case study, we inspect how neural crest cells in the head evolve into mesenchymal and cranial structures.

Baseline. We compare NicheFlow with the spatiotemporal framework in moscot [20], which models spatial trajectories at the single-cell level. In contrast to NicheFlow, moscot integrates spatial

coordinates directly into the OT formulation using a Fused Gromov-Wasserstein cost [45], where the hyperparameter α controls the trade-off between spatial and feature-based distances (see App. F.8). A high α increases the influence of spatial distances, whereas in our framework this role is played by the hyperparameter λ (see Sec. 4.2). We provide more details on the selection of the hyperparameter α for our experiments in App. D.2. Notably, moscot relies on exact OT and learns a transition matrix between source and target samples. As such, it is not a generative model over point clouds like NicheFlow. However, given the overlap in downstream tasks, we consider the comparison relevant.

Evaluation and results. For both scenarios (1) and (2), we have prior knowledge of the ground truth regions that the source microenvironments are expected to occupy at later developmental stages, as well as their corresponding biological compositions. For both moscot and NicheFlow, we assess whether the transported mass of source samples concentrates within the correct anatomical region at the target time point, and whether the predicted descendant cell types are biologically consistent (see App. F.8). Results are summarized in Fig. 3. When modeling the evolution of the spinal cord, moscot assigns considerable mass to unrelated regions such as the urogenital ridge and branchial arches, whereas NicheFlow correctly maps source niches to the maturing spinal cord. Similarly, NicheFlow captures the differentiation of neural crest cells into mesenchymal and cranial tissues within the head region, while moscot exhibits substantial off-target leakage towards lower regions.

6 Conclusions and Discussion

We introduce NicheFlow, a point-cloud-based generative model designed to capture the spatiotemporal dynamics of cellular niches in time-resolved spatial transcriptomics data. Unlike methods that model single-cell trajectories independently, NicheFlow implicitly captures spatial correlations between cells by learning trajectories on variably sized local neighborhoods. To this end, we combine OT with a new version of VFM that factorizes features and coordinates into distinct posteriors from different distribution families. We showed that NicheFlow outperforms standard FM approaches at reconstructing spatial context from previous time points and improves the mapping of biological structures in time over established exact OT approaches. With the expected increase in the volume and quality of spatial data, modeling coordinated cellular state translations with generative models is a promising avenue for generalization beyond spatiotemporal inference. We envision that models like NicheFlow will enable spatial perturbation prediction and modality translation tasks requiring principled parameterized maps beyond discrete OT to extrapolate and drive biological hypotheses.

Acknowledgments

A.P. is supported by the Helmholtz Association under the joint research school Munich School for Data Science (MUDS). A.P., F.G., S.G. and F.J.T. also acknowledge support from the German Federal Ministry of Education and Research (BMFTR) through grant numbers 031L0289A and 031L0289C. F.J.T. acknowledges support from the Helmholtz Association's Initiative and Networking Fund via the CausalCellDynamics project (grant number Interlabs-0029) and the European Union (ERC, DeepCell, grant number 101054957). The authors of this work take full responsibility for its content.

References

[1] Urs Mayr, Denise Serra, Prisca Liberali, Allon Klein, and Barbara Treutlein. Exploring single cells in space and time during tissue development, homeostasis and regeneration. *Development*, 146(12), 2019.

[2] Marietta Zinner, Ilya Lukonin, and Prisca Liberali. Design principles of tissue organisation: How single cells coordinate across scales. *Current opinion in cell biology*, 67:37–45, 2020.

[3] Yanming Ren, Zongyao Huang, Lingling Zhou, Peng Xiao, Junwei Song, Ping He, Chuanxing Xie, Ran Zhou, Menghan Li, Xiangqun Dong, Qing Mao, Chao You, Jianguo Xu, Yanhui Liu, Zhigang Lan, Tiejun Zhang, Qi Gan, Yuan Yang, Tengyun Chen, Bowen Huang, Xiang Yang, Anqi Xiao, Yun Ou, Zhengzheng Su, Lu Chen, Yan Zhang, Yan Ju, Yuekang Zhang, and Yuan Wang. Spatial transcriptomics reveals niche-specific enrichment and vulnerabilities of radial glial stem-like cells in malignant gliomas. *Nature Communications*, 14(1), February 2023.

[4] Anna Christina Schaar, Alejandro Tejada-Lapuerta, Giovanni Palla, Robert Gutgesell, Lennard Halle, Mariia Minaeva, Larsen Vornholz, Leander Dony, Francesca Drummer, Mojtaba Bahrami, and Fabian J Theis. Nicheformer: a foundation model for single-cell and spatial omics. *bioRxiv*, pages 2024–04, April 2024.

[5] Song Liu, Xiaoyan Li, Zhiyue Gu, Jiayu Wu, Shuangzheng Jia, Jinghua Shi, Yi Dai, Yushi Wu, Hailan Yan, Jing Zhang, Yan You, Xiaowei Xue, Lulu Liu, Jinghe Lang, Xiaoyue Wang, and Jinhua Leng. Single-cell and spatial transcriptomic profiling revealed niche interactions sustaining growth of endometriotic lesions. *Cell Genomics*, 5(1):100737, January 2025.

[6] Reuben Moncada, Dalia Barkley, Florian Wagner, Marta Chiodin, Joseph C. Devlin, Maayan Baron, Cristina H. Hajdu, Diane M. Simeone, and Itai Yanai. Integrating microarray-based spatial transcriptomics and single-cell RNA-seq reveals tissue architecture in pancreatic ductal adenocarcinomas. *Nature Biotechnology*, 38(3):333–342, January 2020.

[7] Ludvig Larsson, Jonas Frisén, and Joakim Lundeberg. Spatially resolved transcriptomics adds a new dimension to genomics. *Nature Methods*, 18(1):15–18, January 2021.

[8] Ao Chen, Sha Liao, Mengnan Cheng, Kailong Ma, Liang Wu, Yiwei Lai, Xiaojie Qiu, Jin Yang, Jiangshan Xu, Shijie Hao, Xin Wang, Huifang Lu, Xi Chen, Xing Liu, Xin Huang, Zhao Li, Yan Hong, Yujia Jiang, Jian Peng, Shuai Liu, Mengzhe Shen, Chuanyu Liu, Quanshui Li, Yue Yuan, Xiaoyu Wei, Huiwen Zheng, Weimin Feng, Zhifeng Wang, Yang Liu, Zhaohui Wang, Yunzhi Yang, Haitao Xiang, Lei Han, Baoming Qin, Pengcheng Guo, Guangyao Lai, Pura Muñoz-Cánoves, Patrick H. Maxwell, Jean Paul Thiery, Qing-Feng Wu, Fuxiang Zhao, Bichao Chen, Mei Li, Xi Dai, Shuai Wang, Haoyan Kuang, Junhou Hui, Liqun Wang, Ji-Feng Fei, Ou Wang, Xiaofeng Wei, Haorong Lu, Bo Wang, Shiping Liu, Ying Gu, Ming Ni, Wenwei Zhang, Feng Mu, Ye Yin, Huanming Yang, Michael Lisby, Richard J. Cornall, Jan Mulder, Mathias Uhlén, Miguel A. Esteban, Yuxiang Li, Longqi Liu, Xun Xu, and Jian Wang. Spatiotemporal transcriptomic atlas of mouse organogenesis using DNA nanoball-patterned arrays. *Cell*, 185(10):1777–1792.e21, May 2022.

[9] Patrik L. Ståhl, Fredrik Salmén, Sanja Vickovic, Anna Lundmark, José Fernández Navarro, Jens Magnusson, Stefania Giacomello, Michaela Asp, Jakub O. Westholm, Mikael Huss, Annelie Mollbrink, Sten Linnarsson, Simone Codeluppi, Åke Borg, Fredrik Pontén, Paul Igor Costea, Pelin Sahlén, Jan Mulder, Olaf Bergmann, Joakim Lundeberg, and Jonas Frisén. Visualization and analysis of gene expression in tissue sections by spatial transcriptomics. *Science*, 353(6294): 78–82, July 2016.

[10] Samuel G. Rodriques, Robert R. Stickels, Aleksandrina Goeva, Carly A. Martin, Evan Murray, Charles R. Vanderburg, Joshua Welch, Linlin M. Chen, Fei Chen, and Evan Z. Macosko. Slide-seq: A scalable technology for measuring genome-wide expression at high spatial resolution. *Science*, 363(6434):1463–1467, March 2019.

[11] Chee-Huat Linus Eng, Michael Lawson, Qian Zhu, Ruben Dries, Noushin Koulena, Yodai Takei, Jina Yun, Christopher Cronin, Christoph Karp, Guo-Cheng Yuan, and Long Cai. Transcriptome-scale super-resolved imaging in tissues by RNA seqFISH+. *Nature*, 568(7751):235–239, March 2019.

[12] Lambda Moses and Lior Pachter. Museum of spatial transcriptomics. *Nature Methods*, 19(5): 534–546, March 2022.

[13] James A. Briggs, Caleb Weinreb, Daniel E. Wagner, Sean Megason, Leonid Peshkin, Marc W. Kirschner, and Allon M. Klein. The dynamics of gene expression in vertebrate embryogenesis at single-cell resolution. *Science*, 360(6392), June 2018.

[14] Daniel E. Wagner, Caleb Weinreb, Zach M. Collins, James A. Briggs, Sean G. Megason, and Allon M. Klein. Single-cell mapping of gene expression landscapes and lineage in the zebrafish embryo. *Science*, 360(6392), June 2018.

[15] Jingyi Ren, Haowen Zhou, Hu Zeng, Connie Kangni Wang, Jiahao Huang, Xiaojie Qiu, Xin Sui, Qiang Li, Xunwei Wu, Zuwan Lin, Jennifer A. Lo, Kamal Maher, Yichun He, Xin Tang, Judson Lam, Hongyu Chen, Brian Li, David E. Fisher, Jia Liu, and Xiao Wang. Spatiotemporally

resolved transcriptomics reveals the subcellular RNA kinetic landscape. *Nature Methods*, 20(5): 695–705, April 2023.

[16] Michael Papanicolaou, Amelia L Parker, Michelle Yam, Elysse C Filipe, Sunny Z Wu, Jessica L Chitty, Kaitlin Wyllie, Emmi Tran, Ellie Mok, Audrey Nadalini, et al. Temporal profiling of the breast tumour microenvironment reveals collagen xii as a driver of metastasis. *Nature communications*, 13(1):4587, 2022.

[17] Tamim Abdelaal, Laurens M Grossouw, R Jeroen Pasterkamp, Boudewijn PF Lelieveldt, Marcel JT Reinders, and Ahmed Mahfouz. Sirv: Spatial inference of rna velocity at the single-cell resolution. *NAR genomics and bioinformatics*, 6(3):lqae100, 2024.

[18] Wenxin Long, Tianyu Liu, Lingzhou Xue, and Hongyu Zhao. spvelo: Rna velocity inference for multi-batch spatial transcriptomics data. *bioRxiv*, pages 2025–03, 2025.

[19] Xunan Shen, Lulu Zuo, Zhongfei Ye, Zhongyang Yuan, Ke Huang, Zeyu Li, Qichao Yu, Xuanxuan Zou, Xiaoyu Wei, Ping Xu, et al. Inferring cell trajectories of spatial transcriptomics via optimal transport analysis. *Cell Systems*, 16(2), 2025.

[20] Dominik Klein, Giovanni Palla, Marius Lange, Michal Klein, Zoe Piran, Manuel Gander, Laetitia Meng-Papaxanthos, Michael Sterr, Lama Saber, Changying Jing, et al. Mapping cells through time and space with moscot. *Nature*, pages 1–11, 2025.

[21] John Peterson Bryan, Samouil L Farhi, and Brian Cleary. Accurate trajectory inference in time-series spatial transcriptomics with structurally-constrained optimal transport. *bioRxiv*, pages 2025–03, 2025.

[22] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. In *The Eleventh International Conference on Learning Representations*, 2023.

[23] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

[24] Michael Samuel Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*, 2023.

[25] Floor Eijkelboom, Grigory Bartosh, Christian A. Naesseth, Max Welling, and Jan-Willem van de Meent. Variational flow matching for graph generation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[26] Alexander Tong, Kilian FATRAS, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, 2024.

[27] Alexander Y. Tong, Nikolay Malkin, Kilian Fatras, Lazar Atanackovic, Yanlei Zhang, Guillaume Huguet, Guy Wolf, and Yoshua Bengio. Simulation-free Schrödinger bridges via score and flow matching. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 1279–1287. PMLR, 02–04 May 2024.

[28] Lazar Atanackovic, Xi Zhang, Brandon Amos, Mathieu Blanchette, Leo J Lee, Yoshua Bengio, Alexander Tong, and Kirill Neklyudov. Meta flow matching: Integrating vector fields on the wasserstein manifold. In *The Thirteenth International Conference on Learning Representations*, 2025.

[29] Dominik Klein, Théo Uscidda, Fabian Theis, and Marco Cuturi. Genot: Entropic (gromov) wasserstein flow matching with applications to single-cell genomics. *Advances in Neural Information Processing Systems*, 37:103897–103944, 2024.

[30] Doron Haviv, Aram-Alexandre Pooladian, Dana Pe'er, and Brandon Amos. Wasserstein flow matching: Generative modeling over families of distributions. *arXiv preprint arXiv:2411.00698*, 2024.

[31] Sichen Zhu, Yuchen Zhu, Molei Tao, and Peng Qiu. Diffusion generative modeling for spatially resolved gene expression inference from histology images. In *The Thirteenth International Conference on Learning Representations*, 2025.

[32] Xiaomeng Wan, Jiashun Xiao, Sindy Sing Ting Tam, Mingxuan Cai, Ryohichi Sugimura, Yang Wang, Xiang Wan, Zhixiang Lin, Angela Ruohao Wu, and Can Yang. Integrating spatial and single-cell transcriptomics data using deep generative models with spatialscope. *Nature Communications*, 14(1):7848, 2023.

[33] Doron Haviv, Ján Remšík, Mohamed Gatie, Catherine Snopkowski, Meril Takizawa, Nathan Pereira, John Bashkin, Stevan Jovanovich, Tal Nawy, Ronan Chaligne, et al. The covariance environment defines cellular niches for spatial inference. *Nature Biotechnology*, 43(2):269–280, 2025.

[34] Kongming Li, Jiahao Li, Yuhao Tao, and Fei Wang. stdiff: a diffusion model for imputing spatial transcriptomics through single-cell transcriptomics. *Briefings in Bioinformatics*, 25(3): bbae171, 2024.

[35] Amir Akbarnejad, Lloyd Steele, Daniyal J Jafree, Sebastian Birk, Marta Rosa Sallese, Koen Rademaker, Adam Boxall, Benjamin Rumney, Catherine Tudor, Minal Patel, et al. Mapping and reprogramming human tissue microenvironments with mintflow. *bioRxiv*, pages 2025–06, 2025.

[36] Tingyang Yu, Chanakya Ekbote, Nikita Morozov, Jiashuo Fan, Pascal Frossard, Stéphane d'Ascoli, and Maria Brbić. Tissue reassembly with generative ai. *bioRxiv*, pages 2025–02, 2025.

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

[38] Duy Pham, Xiao Tan, Brad Balderson, Jun Xu, Laura F Grice, Sohye Yoon, Emily F Willis, Minh Tran, Pui Yeng Lam, Arti Raghubar, et al. Robust mapping of spatiotemporal trajectories and cell–cell interactions in healthy and diseased tissues. *Nature communications*, 14(1):7739, 2023.

[39] Peter Halmos, Xinhao Liu, Julian Gold, Feng Chen, Li Ding, and Benjamin J Raphael. Dest-ot: Alignment of spatiotemporal transcriptomics data. *Cell Systems*, 16(2), 2025.

[40] Titouan Vayer, Laetitia Chapel, Rémi Flamary, Romain Tavenard, and Nicolas Courty. Fused gromov-wasserstein distance for structured objects. *Algorithms*, 13(9):212, 2020.

[41] Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, and Ricky T. Q. Chen. Multisample flow matching: Straightening flows with minibatch couplings. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 28100–28127. PMLR, 23–29 Jul 2023.

[42] Xiaoyu Wei, Sulei Fu, Hanbo Li, Yang Liu, Shuai Wang, Weimin Feng, Yunzhi Yang, Xiawei Liu, Yan-Yun Zeng, Mengnan Cheng, Yiwei Lai, Xiaojie Qiu, Liang Wu, Nannan Zhang, Yujia Jiang, Jiangshan Xu, Xiaoshan Su, Cheng Peng, Lei Han, Wilson Pak-Kin Lou, Chuanyu Liu, Yue Yuan, Kailong Ma, Tao Yang, Xiangyu Pan, Shang Gao, Ao Chen, Miguel A. Esteban, Huanming Yang, Jian Wang, Guangyi Fan, Longqi Liu, Liang Chen, Xun Xu, Ji-Feng Fei, and Ying Gu. Single-cell stereo-seq reveals induced progenitor cells involved in axolotl brain regeneration. *Science*, 377(6610):eabp9444, 2022.

[43] Eric D. Sun, Olivia Y. Zhou, Max Hauptschein, Nimrod Rappoport, Lucy Xu, Paloma Navarro Negredo, Ling Liu, Thomas A. Rando, James Zou, and Anne Brunet. Spatial transcriptomic clocks reveal cell proximity effects in brain ageing. *Nature*, 638(8049):160–171, Feb 2025.

[44] Kok Hao Chen, Alistair N Boettiger, Jeffrey R Moffitt, Siyuan Wang, and Xiaowei Zhuang. Spatially resolved, highly multiplexed rna profiling in single cells. *Science*, 348(6233):aaa6090, 2015.

[45] Titouan Vayer, Nicolas Courty, Romain Tavenard, and Rémi Flamary. Optimal Transport for Structured Data with Application on Graphs. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6275–6284. PMLR, 2019.

[46] Haoqiang Fan, Hao Su, and Leonidas Guibas. A point set generation network for 3d object reconstruction from a single image. In *Computer Vision and Pattern Recognition*, 2016.

[47] Facundo Mémoli. Gromov–wasserstein distances and the metric approach to object matching. *Foundations of Computational Mathematics*, 11(4):417–487, Aug 2011.

[48] Alexander Tong, Jessie Huang, Guy Wolf, David Van Dijk, and Smita Krishnaswamy. TrajectoryNet: A dynamic optimal transport network for modeling cellular dynamics. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9526–9536. PMLR, 2020.

[49] Guillaume Huguet, Daniel Sumner Magruder, Alexander Tong, Oluwadamilola Fasina, Manik Kuchroo, Guy Wolf, and Smita Krishnaswamy. Manifold interpolating optimal-transport flows for trajectory inference. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[50] Zhenyi Zhang, Tiejun Li, and Peijie Zhou. Learning stochastic dynamics from snapshots through regularized unbalanced optimal transport. In *The Thirteenth International Conference on Learning Representations*, 2025.

A Broader impacts

This work addresses fundamental challenges in spatial transcriptomics by modeling complex spatial and compositional changes in developing tissues. We demonstrate how efficient representations of high-dimensional spatial cellular data can advance the understanding of developmental trajectories and microenvironment dynamics. We anticipate releasing NicheFlow as an open-source, user-friendly tool to enable broad application in spatial biology studies. Given its use with biological data, NicheFlow may also be applied in sensitive contexts involving clinical or patient information.

B Limitations

Our approach relies on fixed OT feature weighting by a parameter λ during training (see Sec. 4.2), limiting flexibility at inference and potentially constraining certain biological analyses. Moreover, radius-based niche definitions may also be sub-optimal for small or irregularly shaped microenvironments, where the radius captures excessive spatial context and does not allow fine-grained modeling of the functional region's evolution.

The model assumes that spatial slides can be aligned with respect to each other in time and requires normalization-based pre-processing. Future work will be directed towards rotational and translational invariant spatial constraints and the incorporation of cell-to-cell communication priors in the neighborhood definition. While the learned flow models cell population dynamics, it does not explicitly capture biological events such as division or death.

Finally, this study focuses on in-distribution testing and does not consider generalization to unseen slides or full anatomical regions excluded from the training process. To achieve prediction in unseen settings, we foresee the need for technical replicates of the same slide across time points, as the model cannot extrapolate spatial arrangements without prior exposure to the associated region. We leave these analyses to future work when spatio-temporal measurements across multiple replicates become increasingly available.

C Mixed-factorized Variational Flow Matching

C.1 Theoretical aspects of Variational Flow Matching

Variational Flow Matching (VFM) [25] relies on the observation that one can write the time-resolved marginal vector field $u_t(\boldsymbol{x})$ in FM as the expected conditional field $u_t(\boldsymbol{x} \mid \boldsymbol{x}_1)$ under the posterior $p_t(\boldsymbol{x}_1 \mid \boldsymbol{x})$ as:

$$u_t(\boldsymbol{x}) = \mathbb{E}_{p_t(\boldsymbol{x}_1 \mid \boldsymbol{x})} \left[u_t(\boldsymbol{x} \mid \boldsymbol{x}_1) \right] . \tag{15}$$

Since $u_t(\boldsymbol{x} \mid \boldsymbol{x}_1)$ has a closed form and $u_t(\boldsymbol{x})$ is all that we need to generate the probability path p_t from noise to data, this opens the door to a new interpretation of the objective as a variational inference problem, where we approximate $p_t(\boldsymbol{x}_1 \mid \boldsymbol{x})$ with a variational posterior $q_t^\theta(\boldsymbol{x}_1 \mid \boldsymbol{x})$. In other words, one can optimize the following objective:

$$\mathcal{L}_{\text{VFM}}(\theta) = -\mathbb{E}_{t \sim \mathcal{U}[0,1], \boldsymbol{x}_1 \sim p_1(\boldsymbol{x}_1), \boldsymbol{x} \sim p_t(\boldsymbol{x} \mid \boldsymbol{x}_1)} \left[\log q_t^\theta(\boldsymbol{x}_1 \mid \boldsymbol{x}) \right] ,$$

where $p_1(\boldsymbol{x})$ is the data distribution and $p_t(\boldsymbol{x} \mid \boldsymbol{x}_1)$ a straight probability path. When $u_t(\boldsymbol{x} \mid \boldsymbol{x}_1)$ is linear in $\boldsymbol{x}_1$, this model formulation acquires convenient properties listed below.

Mean parameterization. The expected conditional field under the posterior only depends on the posterior mean:

$$\mathbb{E}_{p_t(\boldsymbol{x}_1 \mid \boldsymbol{x})} \left[u_t(\boldsymbol{x} \mid \boldsymbol{x}_1) \right] = u_t(\boldsymbol{x}_1 \mid \mathbb{E}_{p_t(\boldsymbol{x}_1 \mid \boldsymbol{x})} \left[\boldsymbol{x}_1 \right]) ,$$

suggesting that it is sufficient to parameterize the posterior mean to simulate data under the marginal flow. The posterior mean can be regressed against real samples $\boldsymbol{x}_1$ during training.

Equivalence between posterior and approximate posterior formulation. From the previous point, it follows that the expectation of the conditional field is the same under the true and approximate posterior, as long as their first moments match.

Efficient simulation. Given a parameterized posterior mean μ_t^θ, simulating the generative field in Eq. (15) is efficient under the linearity condition. For example, in the standard FM setting with

straight paths [23], the marginal generative field becomes:

$$u_t(\boldsymbol{x}) = \mathbb{E}_{q_t^\theta(\boldsymbol{x}_1|\boldsymbol{x})}\left[u_t(\boldsymbol{x} \mid \boldsymbol{x}_1)\right] \tag{16}$$

$$= u_t\left(\boldsymbol{x} \mid \mathbb{E}_{q_t^\theta(\boldsymbol{x}_1|\boldsymbol{x})}\left[\boldsymbol{x}_1\right]\right) \tag{17}$$

$$= \frac{\mu_t^\theta(\boldsymbol{x}) - \boldsymbol{x}}{1-t}\,. \tag{18}$$

which can be easily simulated in the range $t \in [0, 1]$.

C.2 Factorized posterior

Similar to Eijkelboom et al. [25], in our work, we use a fully factorized posterior, where individual dimensions can follow different families of distributions with finite moments (see Sec. 4.3). Notably, a factorized approximate posterior over $\boldsymbol{x}_1$ is allowed as a choice for q_t^θ, since the only requirement to simulate $u_t(\boldsymbol{x})$ is for $q_t^\theta(\boldsymbol{x}_1 \mid \boldsymbol{x})$ to match the expectation of $p_t(\boldsymbol{x}_1 \mid \boldsymbol{x})$ over $\boldsymbol{x}_1$, irrespectively of higher moments or correlations between factors.

In this regard, it is useful to consider the following proposition.

Proposition 1. *Let $\boldsymbol{x}_1 \in \mathbb{R}^D$ be a D-dimensional target data point, $p_t(\boldsymbol{x}_1 \mid \boldsymbol{x})$ the posterior probability path conditioned on a noisy point $\boldsymbol{x} \sim p_t(\boldsymbol{x})$, and $u_t(\boldsymbol{x} \mid \boldsymbol{x}_1)$ the conditional velocity field. Assume that $u_t(\boldsymbol{x} \mid \boldsymbol{x}_1)$ is linear in $\boldsymbol{x}_1$. Then, for any dimension $d \in \{1, \dots, D\}$, the following holds:*

$$\mathbb{E}_{p_t(\boldsymbol{x}_1|\boldsymbol{x})}[x_1^d] = \mathbb{E}_{p_t(x_1^d|\boldsymbol{x})}[x_1^d] \tag{19}$$

$$u_t(x^d) = u_t\left(x^d \mid \mathbb{E}_{p_t(x_1^d|\boldsymbol{x})}[x_1^d]\right)\,, \tag{20}$$

where x^d refers to the d^{th} scalar dimension of the vector $\boldsymbol{x}$.

Proof. We begin by proving Eq. (19) using marginalization:

$$\begin{aligned}
\mathbb{E}_{p_t(\boldsymbol{x}_1|\boldsymbol{x})}[x_1^d] &= \int x_1^d\, p_t(\boldsymbol{x}_1 \mid \boldsymbol{x})\, \mathrm{d}\boldsymbol{x}_1 \\
&= \int x_1^d \left(\int p_t(\boldsymbol{x}_1 \mid \boldsymbol{x})\, \mathrm{d}\boldsymbol{x}_1^{\setminus d}\right) \mathrm{d}x_1^d \\
&= \int x_1^d\, p_t(x_1^d \mid \boldsymbol{x})\, \mathrm{d}x_1^d\,.
\end{aligned} \tag{21}$$

Next, we prove Eq. (20). Under the assumption that the conditional velocity field $u_t(\boldsymbol{x} \mid \boldsymbol{x}_1)$ is linear in $\boldsymbol{x}_1$, we have:

$$\begin{aligned}
u_t(x^d) &= \mathbb{E}_{p_t(\boldsymbol{x}_1|\boldsymbol{x})}\left[u_t(x^d \mid \boldsymbol{x}_1)\right] \\
&\overset{(1)}{=} \mathbb{E}_{p_t(\boldsymbol{x}_1|\boldsymbol{x})}\left[u_t(x^d \mid x_1^d)\right] \\
&= \mathbb{E}_{p_t(\boldsymbol{x}_1|\boldsymbol{x})}\left[\frac{x_1^d - x^d}{1-t}\right] \\
&= \frac{\mathbb{E}_{p_t(\boldsymbol{x}_1|\boldsymbol{x})}[x_1^d] - x^d}{1-t} \\
&\overset{Eq.\ (21)}{=} \frac{\mathbb{E}_{p_t(x_1^d|\boldsymbol{x})}[x_1^d] - x^d}{1-t} \\
&= u_t\left(x^d \mid \mathbb{E}_{p_t(x_1^d|\boldsymbol{x})}[x_1^d]\right)\,.
\end{aligned} \tag{22}$$

Here, step (1) follows from the linearity assumption, which ensures that the conditional velocity at x^d depends only on x_1^d.

In other words, the expected value under the posterior at an individual feature d does not depend on the other features and has an influence only on the d-th dimension of the conditional vector field. This flexibility allows each dimension's approximate posterior to be chosen from a potentially different distributional family, as long as the first moment exists and is correctly parameterized.

C.3 Marginal field derivation in source-conditioned VFM

When applying source conditioning to VFM, the marginal conditional vector field given a source $\boldsymbol{x}_0$ is:

$$u_t(\boldsymbol{x} \mid \boldsymbol{x}_0) = \int u_t(\boldsymbol{x} \mid \boldsymbol{x}_1) \, \frac{p_t(\boldsymbol{x} \mid \boldsymbol{x}_1) \, \pi(\boldsymbol{x}_1 \mid \boldsymbol{x}_0)}{p_t(\boldsymbol{x} \mid \boldsymbol{x}_0)} \, \mathrm{d}\boldsymbol{x}_1 \tag{23}$$

where the $p_t(\boldsymbol{x} \mid \boldsymbol{x}_1)$ is a probability path interpolating observations $\boldsymbol{x}_1$ with noise. Note that we omit $\boldsymbol{x}_0$ from the probability path and conditional velocity as they are fully determined by $\boldsymbol{x}_1$ under linear conditional probability paths. Furthermore, we can rewrite the marginal as an expectation:

$$\int u_t(\boldsymbol{x} \mid \boldsymbol{x}_1) \, \frac{p_t(\boldsymbol{x} \mid \boldsymbol{x}_1) \, \pi(\boldsymbol{x}_1 \mid \boldsymbol{x}_0)}{p_t(\boldsymbol{x} \mid \boldsymbol{x}_0)} \, \mathrm{d}\boldsymbol{x}_1 = \mathbb{E}_{p_t(\boldsymbol{x}_1 \mid \boldsymbol{x}, \boldsymbol{x}_0)} \left[u_t(\boldsymbol{x} \mid \boldsymbol{x}_1) \right] \, , \tag{24}$$

where we used that $p_t(\boldsymbol{x} \mid \boldsymbol{x}_1) = p_t(\boldsymbol{x} \mid \boldsymbol{x}_0, \boldsymbol{x}_1)$.

C.4 Gaussian and Laplacian Hybrid VFM

We define a hybrid Variational Flow Matching (VFM) model using a fully factorized variational distribution over individual points in the target microenvironment $\mathcal{M}^1$. Following the mean-field assumption, the variational distribution factorizes over spatial and feature dimensions:

$$q_t^\theta(\mathcal{M}^1 \mid \mathcal{M}, \mathcal{M}^0) = \prod_{(\boldsymbol{c}_1, \boldsymbol{x}_1) \in \mathcal{M}^1} q_t^\theta(\boldsymbol{c}_1, \boldsymbol{x}_1 \mid \mathcal{M}, \mathcal{M}^0) \tag{25}$$

$$= \prod_{(\boldsymbol{c}_1, \boldsymbol{x}_1) \in \mathcal{M}^1} \left(\prod_{k=1}^{2} f_t^\theta(c_1^k \mid \mathcal{M}, \mathcal{M}^0) \cdot \prod_{d=1}^{D} r_t^\theta(x_1^d \mid \mathcal{M}, \mathcal{M}^0) \right) . \tag{26}$$

In line with Eijkelboom et al. [25], using FM with straight paths enables us to efficiently simulate the marginal generating field using the *first moment* of the posterior distribution. In other words, for a fully factorized posterior, we only need to parameterize a mean predictor. In our setting, the mean prediction is a neural network μ_t^θ as a time-condition function of a noisy microenvironment $\mathcal{M}$ and a source $\mathcal{M}^0$ with outputs:

$$(\bar{\boldsymbol{f}}_t^\theta, \bar{\boldsymbol{r}}_t^\theta) = \mu_t^\theta(\mathcal{M}, \mathcal{M}^0) \, ,$$

where $\bar{f}_t^{\theta,k}$ and $\bar{r}_t^{\theta,d}$ are the expected values for the k^{th} coordinate and d^{th} cell feature. Then, we choose a parameterization for the variational factors at time $t \in [0, 1]$ as follows:

$$x_1^d \sim \mathcal{N}(\bar{r}_t^{\theta,d}, 1), \tag{27}$$

$$c_1^k \sim \text{Laplace}(\bar{f}_t^{\theta,k}, 1), \tag{28}$$

Substituting into the negative log-likelihood yields:

$$-\log(q_t^\theta(\mathcal{M}^1 \mid \mathcal{M}, \mathcal{M}^0)) \tag{29}$$

$$= -\log \left(\prod_{(\boldsymbol{c}_1, \boldsymbol{x}_1) \in \mathcal{M}^1} \left(\prod_{k=1}^{2} f_t^\theta(c_1^k \mid \mathcal{M}, \mathcal{M}^0) \cdot \prod_{d=1}^{D} r_t^\theta(x_1^d \mid \mathcal{M}, \mathcal{M}^0) \right) \right) \tag{30}$$

$$= \sum_{(\boldsymbol{c}_1, \boldsymbol{x}_1) \in \mathcal{M}^1} \left(\sum_{k=1}^{2} \left(\log 2 + |c_1^k - \bar{f}_t^{\theta,k}| \right) + \sum_{d=1}^{D} \left(\frac{1}{2} \log(2\pi) + \frac{1}{2} \left(x_1^d - \bar{r}_t^{\theta,d} \right)^2 \right) \right)$$

$$= \sum_{(\boldsymbol{c}_1, \boldsymbol{x}_1) \in \mathcal{M}^1} \left(\|\boldsymbol{c}_1 - \bar{\boldsymbol{f}}_t^\theta\|_1 + \frac{1}{2} \|\boldsymbol{x}_1 - \bar{\boldsymbol{r}}_t^\theta\|_2^2 \right) + \text{const w.r.t. } \theta \tag{31}$$

This results in a loss consisting of an ℓ_1 error on spatial coordinates and a mean squared error on gene expression features, consistent with the hybrid variational design.

D Additional results

D.1 Additional comparisons with moscot on embryonic development

We propose a similar analysis as presented in Sec. 5.2.

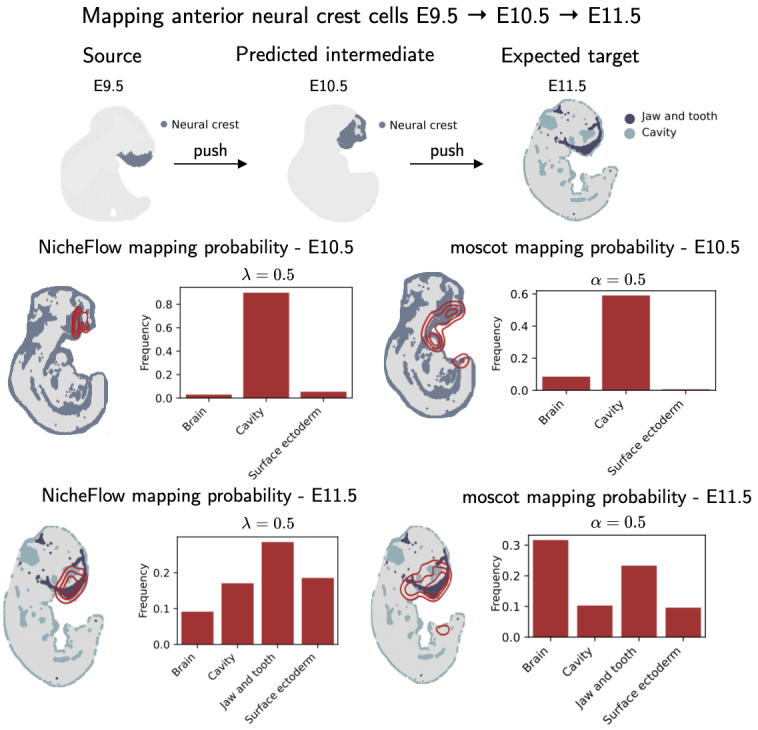


Figure 4: Comparison of NicheFlow and moscot on the prediction of the anterior neural crest cells' fate. For both models, we take source facial neural crest cells at E9.5, push them to time point E10.5, and show the compositional and density predictions in the middle panel. Then, the predictions at 10.5 are used as a source for a second trajectory prediction operation from 10.5 to 11.5, for which we inspect again the cell density over the target slide and the cell type probabilities.

In Fig. 4, we compare the ability of NicheFlow to predict an entire spatial structure trajectory by pushing an initial source cloud through all the developmental stages. To this end, the trajectory of an initial point cloud is first pushed to the next time point, and the model's prediction is used as a source for predicting the subsequent time point. An accurate niche trajectory reconstruction signifies that our model can be used to sequentially predict microenvironment evolution by treating its intermediate predictions as inputs, corroborating their accurate reflection of real point clouds.

In Fig. 4, we show that pushing anterior neural crest cells twice from E9.5 to E11.5 through the flow generates realistic target point clouds with a cell composition reflecting the expected cranial structure, mostly made of cavity cells, jaw and teeth (arising at E11.5 for the first time) and surface ectoderm. Doing the same with moscot oversamples regions outside of the cranial structure, thereby incorrectly mapping most of the neural crest density to brain cells.

In Fig. 5, we also show that NicheFlow is more accurate than moscot at transporting mass from defined organs like the liver across development. More specifically, while density leaks from the liver to the GI tract in the mapping produced by moscot, the prediction computed by NicheFlow more accurately retrieves the liver structure at the later time point. Together with previous evidence, our results underscore the importance of accounting for spatial correlations between cells during OT-based trajectory inference to buffer out the noise resulting from single-cell-based predictions.

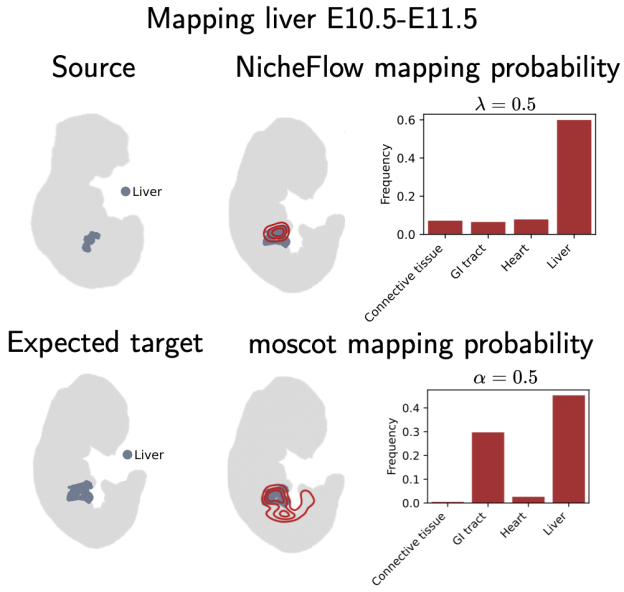


Figure 5: Comparison between moscot and NicheFlow on mapping the liver structure from E10.5 to E11.5. The liver at time E10.5 is used as a source for trajectory prediction using the different models. The left column shows the source and expected target regions highlighted on the respective E10.5 and E11.5 embryos. The middle column displays the density of the prediction obtained by transporting niches from the source to the target slide. On the right, the aggregated cell type proportions according to the density in the middle column (see App. F.8).

D.2 α parameter sweep in moscot

The comparison with moscot assesses how well each model captures compositional changes in anatomical structures or migratory patterns, depending on the use case. Qualitatively, we found that the spatial component in the OT problem considered by moscot and regulated by the hyperparameter α is comparable to our spatial term α across the 0.1–0.9 range in the Fig. 3 analysis.

In Tab. 2, we report the proportion of source density mapped to the correct cell type across α values, to support our choice for the results in Sec. 5.2. For the spinal cord (Fig. 3, left), values above 0.75 yielded better qualitative and quantitative results. For neural crest cells (Fig. 3, right), $\alpha = 0.5$ performs best.

More specifically, when mapping fixed structures over time, values below 0.75 caused excessive density dispersion outside the anatomical region. For migration, no value led to generally accurate transitions, though $\alpha = 0.5$ mapped the highest density to the expected cell type.

Table 2: Effect of the parameter α balancing spatial and cell state preservation in moscot. The results in the table indicate the percentage of source density mapped to the correct cell type from the source anatomical structure (the higher, the better).

Tissue	$\alpha = 0.1$	$\alpha = 0.25$	$\alpha = 0.5$	$\alpha = 0.75$	$\alpha = 0.9$
Spinal Cord	0.146	0.190	0.712	0.729	0.725
Neural Crests	0.159	0.160	0.162	0.155	0.112

D.3 Additional experiments on the axolotl brain development and aging datasets

We provide additional visualizations of the generated samples on the axolotl brain development dataset, presented in Fig. 6. As can be seen from the figure, NicheFlow correctly retrieves the spatial and anatomical characteristics of the brain, including hemisphere formation and cavity.

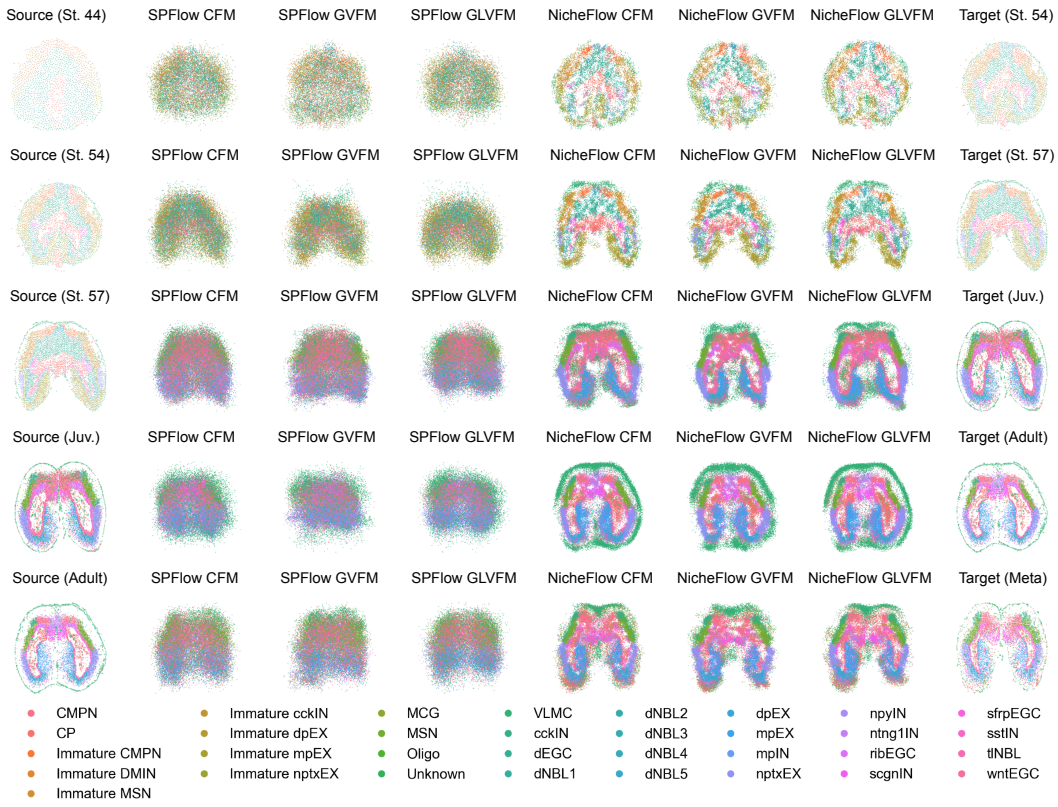


Figure 6: Qualitative comparison of generated samples on the axolotl brain development dataset (Stage 44, 54, 57, Juvenile, Adult, Meta). We show source and target samples alongside predictions from SPFlow and NicheFlow with different objectives. NicheFlow captures the spatial structure and cell-type organization more faithfully across developmental stages.

To further support our result, we propose a more in-depth qualitative analysis of the NicheFlow application on the axolotl brain development dataset. More specifically, in Fig. 7 we demonstrate that our model predicts the formation of crucial anatomical structures like the left and right lobes both spatially and compositionally. This vouches for flexibility in NichFlow's performance, which extends to non-trivial topology changes and simultaneously accounts for accurate cell state and coordinate generation in time. Similar results can be observed in Fig. 8, where we showcase the correct prediction of the formation of a left lobe cavity, predicting trajectories from an immature brain region.

Moreover, in Fig. 9 we predict the compositional and structural time evolution of the left dorsal pallium in the axolotl brain development. Following Wei et al. [42], we know that early time points populate the dorsal pallium of immature cell types like ependymoglial cells (EGC), neuroblasts (NBL), and immature neurons. In the left dorsal pallium, these disappear at the juvenile stages (the 3$^{\text{rd}}$) and lead to differentiation into mature neurons (nptxEX) and later EGC (WntEGC, sfrpEGC). This fixed structural development was accurately predicted by NicheFlow when pushing left dorsal pallium cells forward across the trajectory.

Finally, similar to Fig. 2 and Fig. 6, in Fig. 10, we qualitatively show that NicheFlow with microenvironment sampling strategy and mixed-factorized VFM is the best approach for reconstructing mouse brain trajectories in time.

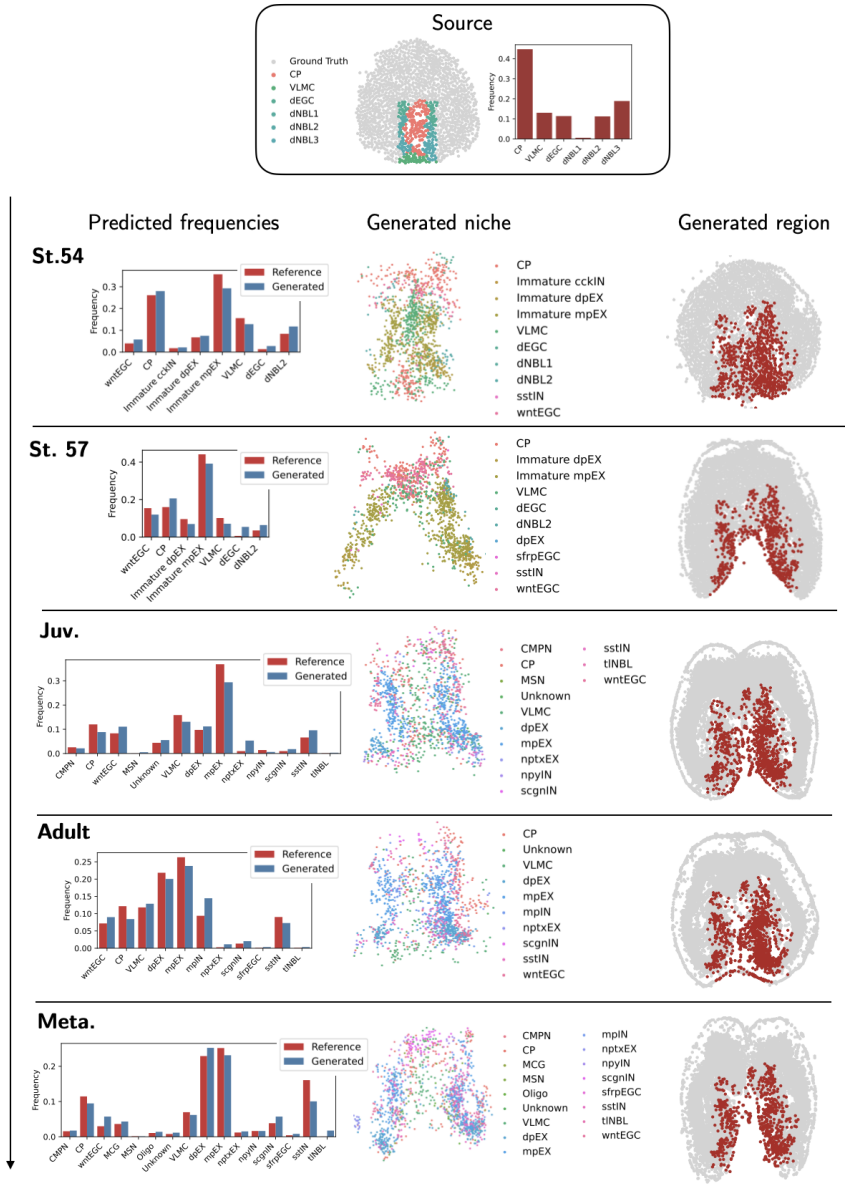


Figure 7: Prediction of hemisphere formation on the axolotl brain development. We evaluate the ability of NicheFlow to generate the anatomical splitting of central brain structure upon the formation of the right and left brain regions. The top panel shows the reference source region and cell type composition. For each stage (St.54, St.57, Juvenile, Adult, and Meta), we show: (1) predicted vs. reference cell type frequencies, (2) the generated niche visualized via 2D embedding colored by cell type, and (3) spatial projection of the generated region onto the anatomical reference (right). We set $\lambda = 0.5$.

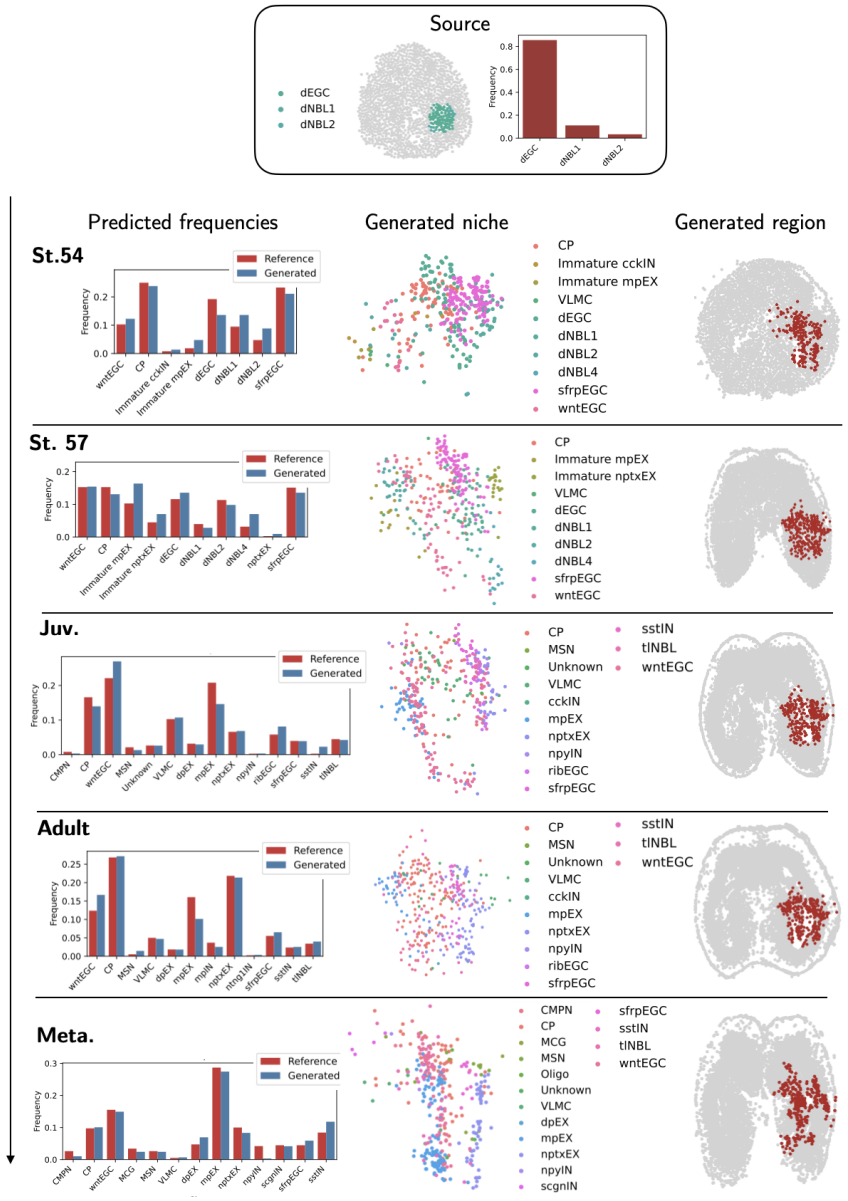


Figure 8: Prediction of cavity formation in the left brain lobe during axolotl development. We assess the ability of NicheFlow to model the emergence of a cavity structure within the left lobe of the axolotl brain. The top panel shows the reference source region and its cell type composition. For each developmental stage (St.54, St.57, Juvenile, Adult, and Meta), we display: (1) predicted versus reference cell type frequencies, (2) the generated niche visualized in 2D embedding space, colored by cell type, and (3) spatial projection of the generated region onto the anatomical brain reference (right). The progression illustrates the model's ability to recapitulate the asymmetric cavity formation localized to the left lobe. We set $\lambda = 0.5$.

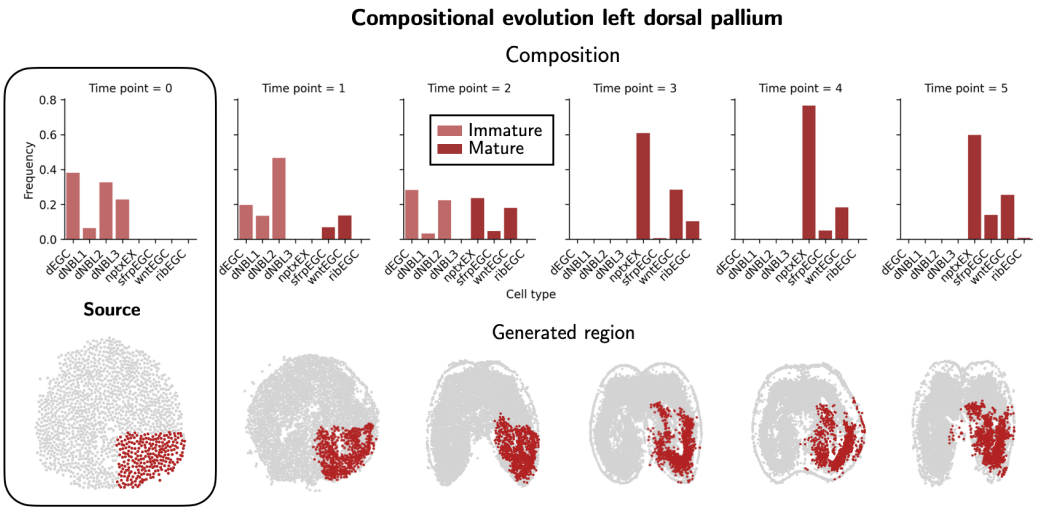


Figure 9: Structural and compositional prediction of the left dorsal pallium during the axolotl brain development. We use the left dorsal pallium region at the St.44 developmental stadium (highlighted on the left) and predict its trajectory over time. In the top row, we show the proportion of different cell types in the predicted region, colored as immature (light red) and mature (dark red). At the bottom, we show the structural prediction for the left dorsal pallium overlaid on the true slide. We set $\lambda = 0.5$.

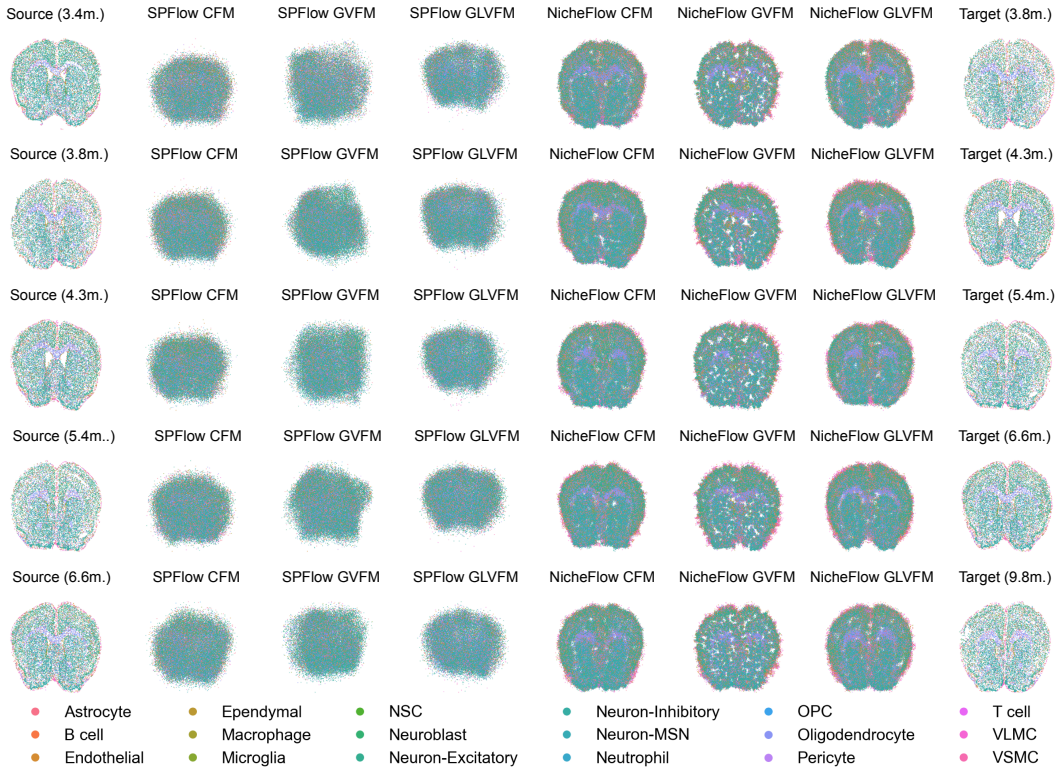


Figure 10: Qualitative comparison of generated samples on the mouse brain aging dataset (3.4, 3.8, 4.3, 5.4, 6.6, 9.8 months). We show source and target samples alongside predictions from SPFlow and NicheFlow with different objectives. NicheFlow captures the spatial structure and cell-type organization more faithfully across developmental stages.

D.4 Conditional generation

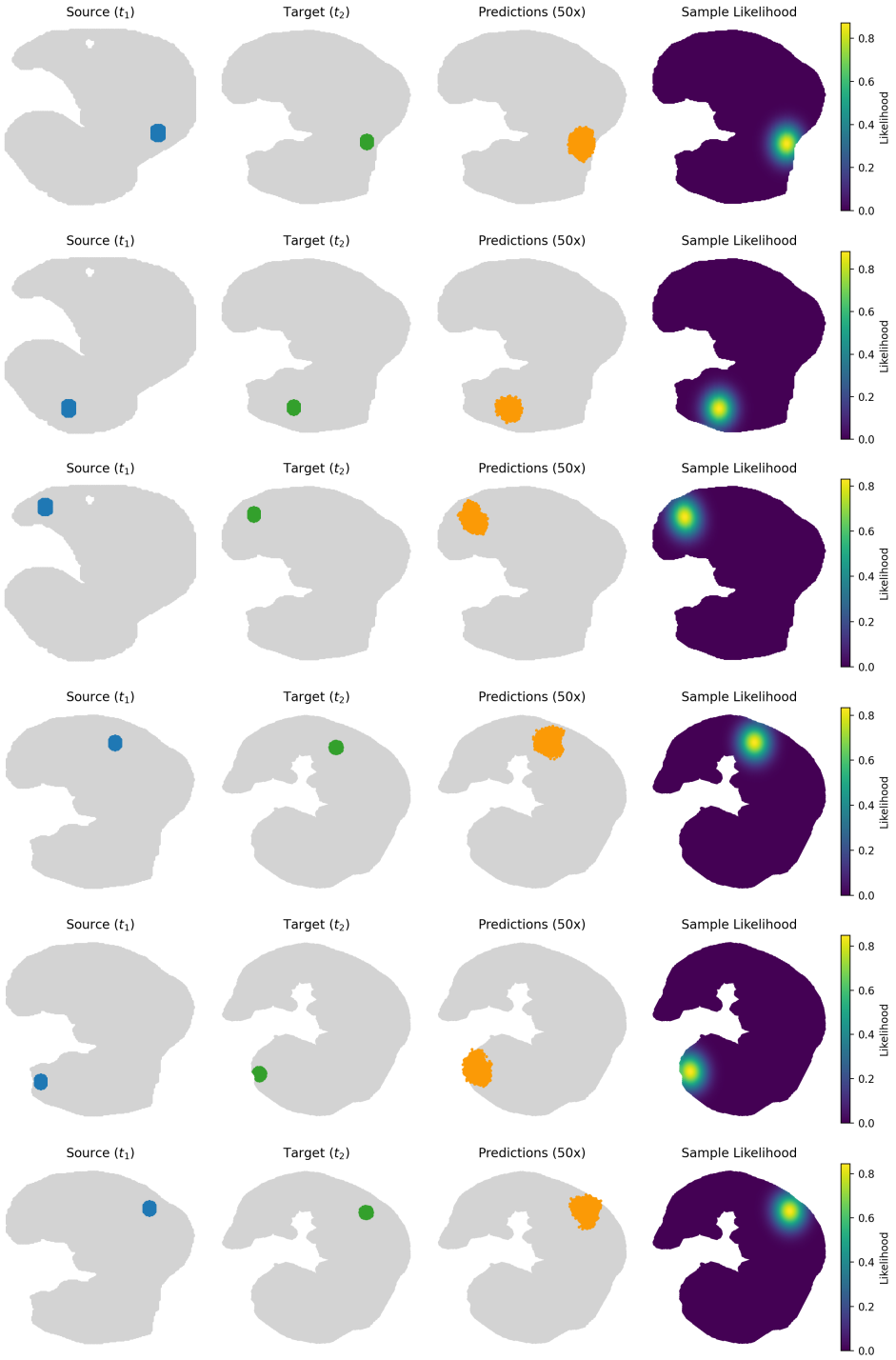


Figure 11: Qualitative evaluation of conditional generation with NicheFlow on the embryonic development dataset. Each row corresponds to a different source microenvironment at time t_1 (blue), shown alongside the ground truth target microenvironment at time t_2 (green). The third column displays 50 samples (orange) generated conditionally by the model, while the fourth column visualizes a kernel density estimate of the sample likelihood over the spatial domain.

To assess whether the model accurately conditions on input microenvironments, we visualize the spatial distribution of generated samples given a fixed source microenvironment. Figure 11 illustrates such cases on the embryonic development dataset, showing the input region at time t_1, the corresponding target at time t_2, and 50 independently generated samples from NicheFlow. Directly computing the likelihood of ground truth cell coordinates under our generative model is intractable, as it would require evaluating the density of an implicitly defined distribution over point clouds. Instead, we approximate the spatial likelihood using kernel density estimation (KDE) over Monte Carlo samples drawn from the model. Given a set of generated coordinates $\{\hat{\boldsymbol{c}}_i\}_{i=1}^N$, we estimate the likelihood at a ground truth location $\boldsymbol{c}$ as:

$$\hat{p}(\boldsymbol{c}) = \frac{1}{N}\sum_{i=1}^{N} \exp\left(-\frac{\|\boldsymbol{c}-\hat{\boldsymbol{c}}_i\|^2}{2\sigma^2}\right), \tag{32}$$

where σ is a fixed bandwidth parameter. The resulting KDE heatmap visualizes the spatial concentration of samples, allowing us to qualitatively assess whether the model produces consistent predictions conditioned on the source microenvironment.

D.5 OT Ablation Study

We begin by emphasizing that the optimal choice of the OT feature-weighting parameter λ in Eq. (12) depends on the downstream application. This parameter determines the relative importance assigned to gene expression versus spatial coordinates during OT. For developmental processes such as organogenesis or regeneration, higher values of λ prioritize transcriptional similarity, facilitating the reconstruction of continuous differentiation trajectories and capturing fate-driven transitions, which may span large spatial distances. In contrast, for applications that aim to monitor changes within a fixed spatial region, such as shifts in cell-type composition over time, a lower λ is more appropriate. In such cases, emphasizing spatial locality helps avoid spurious long-range transport assignments caused by molecular noise.

Table 3: Ablation study on the feature-coordinate trade-off parameter λ in Eq. (12) across mouse embryonic development (MED), axolotl brain development (ABD), and mouse brain aging (MBA). We report 1NN-F1, PSD, and SPD for each training objective (CFM, GVFM, GLVFM) and model variant (RPCFlow, NicheFlow). All results are averaged over five evaluation runs. Bold indicates the best value per dataset and metric.

			MED			ABD			MBA		
Model	Obj.	λ	1NN-F1 ↑	PSD ↓	SPD ↓	1NN-F1 ↑	PSD ↓	SPD ↓	1NN-F1 ↑	PSD ↓	SPD ↓
RPCFlow	CFM	0.10	0.546 ± 0.0012	0.981 ± 0.0024	0.564 ± 0.0015	0.524 ± 0.0020	2.051 ± 0.0039	1.015 ± 0.0036	0.271 ± 0.0004	1.543 ± 0.0016	0.810 ± 0.0010
RPCFlow	CFM	0.25	0.545 ± 0.0004	0.958 ± 0.0040	0.570 ± 0.0011	0.511 ± 0.0015	2.079 ± 0.0053	1.024 ± 0.0046	0.273 ± 0.0005	**1.535 ± 0.0009**	0.818 ± 0.0013
RPCFlow	CFM	0.50	0.554 ± 0.0009	0.988 ± 0.0031	0.562 ± 0.0011	0.507 ± 0.0011	2.077 ± 0.0065	1.023 ± 0.0032	0.273 ± 0.0006	1.546 ± 0.0010	0.810 ± 0.0013
RPCFlow	CFM	0.75	0.537 ± 0.0007	0.981 ± 0.0020	0.595 ± 0.0022	0.517 ± 0.0005	2.033 ± 0.0072	1.026 ± 0.0050	0.275 ± 0.0003	1.553 ± 0.0008	0.801 ± 0.0010
RPCFlow	GLVFM	0.10	0.586 ± 0.0016	0.979 ± 0.0021	0.586 ± 0.0012	0.554 ± 0.0007	2.053 ± 0.0044	1.038 ± 0.0025	0.265 ± 0.0004	1.723 ± 0.0015	0.779 ± 0.0011
RPCFlow	GLVFM	0.25	0.593 ± 0.0003	0.924 ± 0.0019	0.575 ± 0.0017	0.555 ± 0.0013	2.076 ± 0.0057	1.036 ± 0.0024	0.267 ± 0.0003	1.728 ± 0.0016	0.777 ± 0.0005
RPCFlow	GLVFM	0.50	0.586 ± 0.0013	0.934 ± 0.0019	0.569 ± 0.0014	0.551 ± 0.0008	2.038 ± 0.0045	1.032 ± 0.0034	0.269 ± 0.0002	1.715 ± 0.0014	0.783 ± 0.0006
RPCFlow	GLVFM	0.75	0.593 ± 0.0009	0.948 ± 0.0035	0.570 ± 0.0011	0.561 ± 0.0008	**2.014 ± 0.0056**	1.035 ± 0.0047	0.268 ± 0.0005	1.675 ± 0.0019	0.784 ± 0.0013
RPCFlow	GVFM	0.10	0.503 ± 0.0013	1.155 ± 0.0044	0.578 ± 0.0007	0.477 ± 0.0008	2.260 ± 0.0077	1.036 ± 0.0031	0.249 ± 0.0003	1.753 ± 0.0020	0.784 ± 0.0010
RPCFlow	GVFM	0.25	0.520 ± 0.0010	1.223 ± 0.0044	0.566 ± 0.0011	0.478 ± 0.0014	2.364 ± 0.0065	1.030 ± 0.0042	0.246 ± 0.0005	1.710 ± 0.0013	0.787 ± 0.0014
RPCFlow	GVFM	0.50	0.521 ± 0.0012	1.185 ± 0.0032	0.569 ± 0.0009	0.480 ± 0.0008	2.360 ± 0.0036	1.025 ± 0.0015	0.245 ± 0.0004	1.756 ± 0.0025	0.774 ± 0.0007
RPCFlow	GVFM	0.75	0.514 ± 0.0013	1.202 ± 0.0014	0.573 ± 0.0008	0.471 ± 0.0012	2.476 ± 0.0082	1.037 ± 0.0036	0.248 ± 0.0003	1.831 ± 0.0015	0.780 ± 0.0012
NicheFlow	CFM	0.10	0.609 ± 0.0030	0.979 ± 0.0228	0.402 ± 0.0036	0.604 ± 0.0018	2.086 ± 0.0058	0.568 ± 0.0030	0.283 ± 0.0003	1.557 ± 0.0014	0.556 ± 0.0028
NicheFlow	CFM	0.25	0.569 ± 0.0031	0.973 ± 0.0074	0.425 ± 0.0062	0.586 ± 0.0013	2.106 ± 0.0072	**0.565 ± 0.0039**	0.281 ± 0.0006	1.546 ± 0.0029	0.612 ± 0.0032
NicheFlow	CFM	0.50	0.551 ± 0.0009	1.051 ± 0.0496	0.471 ± 0.0110	0.585 ± 0.0013	2.089 ± 0.0066	0.591 ± 0.0026	0.283 ± 0.0002	1.588 ± 0.0033	0.604 ± 0.0061
NicheFlow	CFM	0.75	0.519 ± 0.0038	1.103 ± 0.0323	0.515 ± 0.0101	0.571 ± 0.0012	2.126 ± 0.0110	0.592 ± 0.0044	0.278 ± 0.0003	1.566 ± 0.0027	0.616 ± 0.0041
NicheFlow	GVFM	0.10	0.596 ± 0.0027	0.991 ± 0.0137	0.406 ± 0.0025	0.574 ± 0.0015	2.220 ± 0.0107	0.594 ± 0.0046	0.268 ± 0.0003	1.661 ± 0.0033	**0.531 ± 0.0010**
NicheFlow	GVFM	0.25	0.563 ± 0.0027	1.051 ± 0.0117	0.491 ± 0.0105	0.571 ± 0.0009	2.343 ± 0.0136	0.619 ± 0.0061	0.265 ± 0.0006	1.599 ± 0.0032	0.590 ± 0.0018
NicheFlow	GVFM	0.50	0.533 ± 0.0053	1.034 ± 0.0452	0.800 ± 0.0401	0.556 ± 0.0016	2.283 ± 0.0126	0.742 ± 0.0134	0.269 ± 0.0005	1.607 ± 0.0029	0.605 ± 0.0022
NicheFlow	GVFM	0.75	0.526 ± 0.0028	1.121 ± 0.0112	0.870 ± 0.0336	0.556 ± 0.0013	2.166 ± 0.0091	0.684 ± 0.0128	0.263 ± 0.0004	1.613 ± 0.0031	0.731 ± 0.0040
NicheFlow	GLVFM	0.10	**0.664 ± 0.0014**	0.883 ± 0.0094	0.398 ± 0.0023	**0.628 ± 0.0013**	2.079 ± 0.0043	0.576 ± 0.0055	**0.285 ± 0.0003**	1.554 ± 0.0021	0.532 ± 0.0009
NicheFlow	GLVFM	0.25	0.629 ± 0.0033	0.923 ± 0.0109	**0.394 ± 0.0035**	0.618 ± 0.0014	2.102 ± 0.0028	0.577 ± 0.0051	0.284 ± 0.0004	1.599 ± 0.0035	0.549 ± 0.0012
NicheFlow	GLVFM	0.50	0.610 ± 0.0036	0.909 ± 0.0107	0.417 ± 0.0060	0.610 ± 0.0008	2.136 ± 0.0025	0.579 ± 0.0014	0.283 ± 0.0005	1.600 ± 0.0039	0.546 ± 0.0009
NicheFlow	GLVFM	0.75	0.592 ± 0.0019	**0.879 ± 0.0054**	0.472 ± 0.0150	0.603 ± 0.0014	2.106 ± 0.0060	0.592 ± 0.0073	0.281 ± 0.0004	1.573 ± 0.0008	0.595 ± 0.0026

In Tab. 3, we sweep the value of λ across multiple models and training strategies on the same task as in Sec. 5.1, showing consistent results across settings as reported in Tab. 1.

Furthermore, Figure 12 visualizes the impact of varying λ on the OT plans described in Sec. 4.2, where intermediate values yield more coherent and biologically plausible couplings.

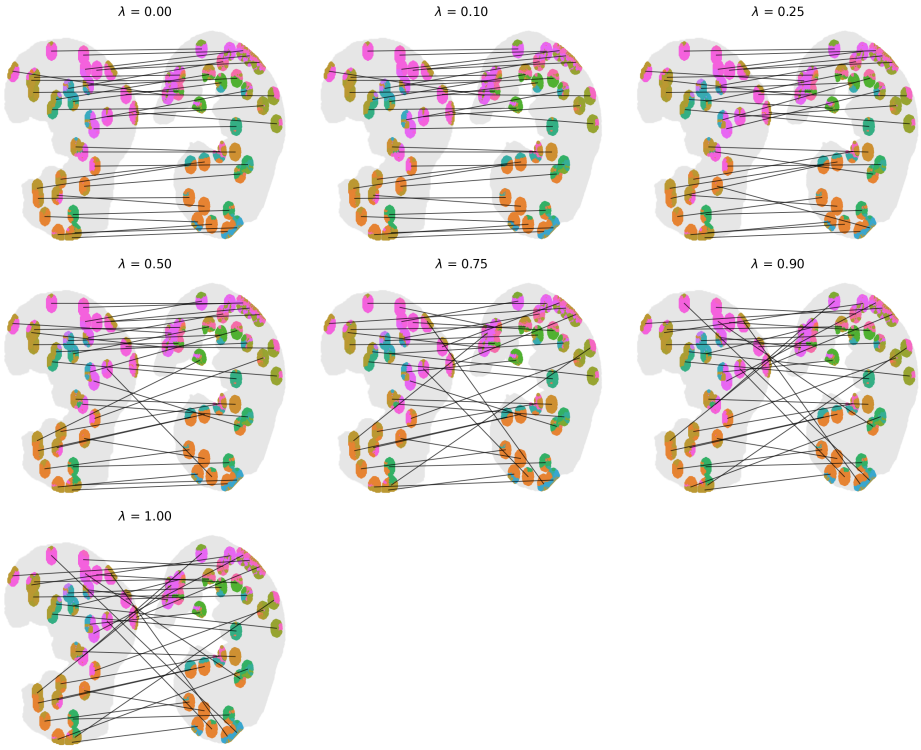


Figure 12: Visualization of OT couplings computed under varying values of the pooling parameter λ in eq. (12), which balances spatial coordinates and gene expression in microenvironment matching. Lower λ values prioritize spatial proximity, resulting in more dispersed and less structured alignments, while intermediate values yield tighter, biologically consistent mappings. Very high λ settings ignore spatial context and may lead to implausible long-range matches.

D.6 K-Means regions ablation study

To ensure diverse and spatially distributed sampling during training, we partition each tissue section into K spatial regions using K-Means clustering over the 2D cell coordinates (see Sec. 5.1.1). At each training step, microenvironments are sampled uniformly from within these regions, encouraging broad spatial coverage and preventing oversampling of densely populated areas. In this ablation study, we investigate how varying the number of spatial regions, K, affects model performance.

Table 4: Ablation study of the number of spatial regions K defined over the datasets. We evaluate NicheFlow with the GLVFM objective across three datasets: mouse embryonic development (MED), axolotl brain development (ABD), and mouse brain aging (MBA). Results are reported as mean ± standard deviation over five evaluation runs.

K	MED			ABD			MBA		
	1NN-F1 ↑	PSD ↓ (10^2)	SPD ↓ (10^2)	1NN-F1 ↑	PSD ↓ (10^2)	SPD ↓ (10^2)	1NN-F1 ↑	PSD ↓ (10^2)	SPD ↓ (10^2)
8	0.640 ± 0.0043	**0.876 ± 0.0055**	0.441 ± 0.0058	0.617 ± 0.0017	1.954 ± 0.0057	0.631 ± 0.0073	0.283 ± 0.0005	1.561 ± 0.0018	0.571 ± 0.0019
16	0.661 ± 0.0033	0.881 ± 0.0068	0.393 ± 0.0056	**0.633 ± 0.0008**	**1.936 ± 0.0027**	0.628 ± 0.0104	0.283 ± 0.0003	1.564 ± 0.0035	0.538 ± 0.0021
32	0.659 ± 0.0025	0.899 ± 0.0120	**0.391 ± 0.0029**	0.622 ± 0.0005	1.968 ± 0.0036	0.640 ± 0.0124	0.279 ± 0.0005	1.600 ± 0.0040	0.537 ± 0.0015
64	**0.664 ± 0.0014**	0.883 ± 0.0094	0.398 ± 0.0023	0.628 ± 0.0013	2.079 ± 0.0043	**0.576 ± 0.0055**	**0.285 ± 0.0003**	**1.554 ± 0.0021**	**0.532 ± 0.0009**

As shown in Figure 13, increasing K leads to increasingly fine-grained spatial partitions. While moderate values of K help improve spatial resolution, excessively high values (e.g., $K = 128$ or 256) result in overly small and fragmented regions. This can cause significant overlap between sampled microenvironments and reduce sampling diversity. Moreover, in sparsely populated tissue sections, high K values may yield regions with insufficient cells, degrading both representativeness and stability.

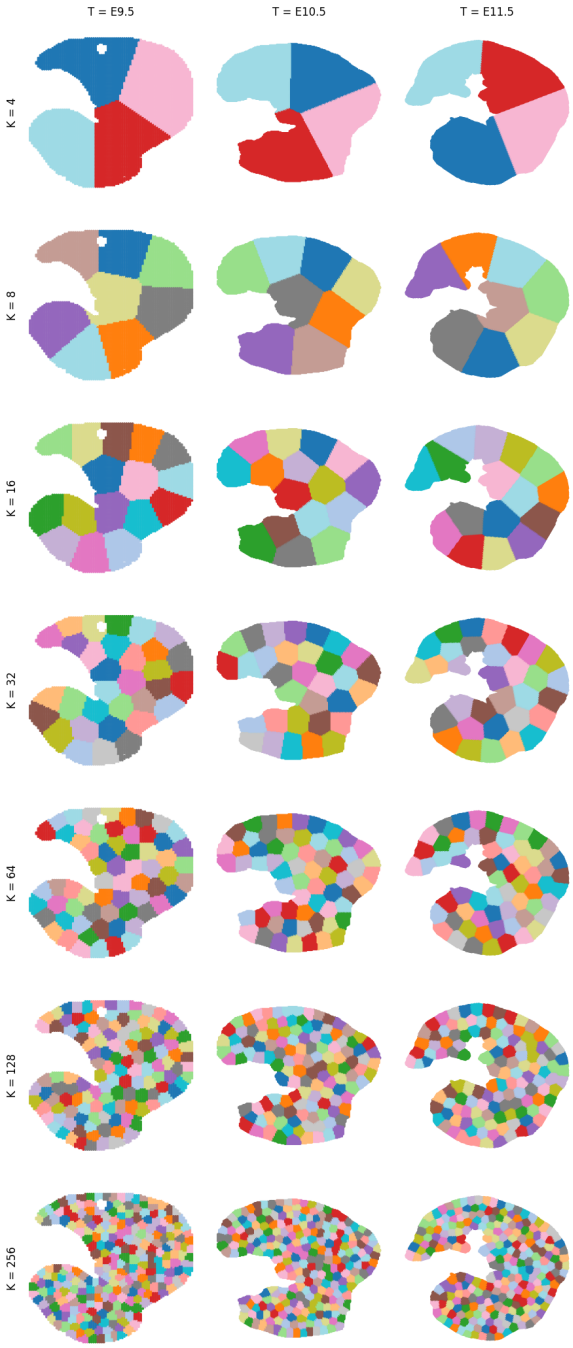


Figure 13: Visualization of spatial partitions obtained via KMeans clustering with $K = \{4, 8, 16, 32, 64, 128, 256\}$ on the embryonic development dataset. For low K, each region covers large heterogeneous areas; for high K, regions become small, dense, and highly overlapping, potentially degrading the diversity and utility of sampled microenvironments.

Conversely, using too few regions (e.g., $K = 4$) results in broad spatial partitions that may encompass multiple heterogeneous tissue compartments. This undermines the locality assumptions of our model and increases intra-region variability, which can impair the model's ability to learn.

Given the trade-offs outlined above, we focus our evaluation on $K \in \{8, 16, 32, 64\}$, which spans a range of granularities that preserve both spatial interpretability and sampling robustness. Tab. 4 sum-

marizes the quantitative results for this ablation across three datasets: mouse embryonic development (MED), axolotl brain development (ABD), and mouse brain aging (MBA). We observe that using $K = 64$ consistently yields strong performance, achieving the highest 1NN-F1 scores on both the MED and MBA datasets, while also performing competitively on ABD. These findings indicate that $K = 64$ offers an effective balance between spatial resolution and stability, and we adopt it as the default configuration in our main experiments.

D.7 Justifying the choice of NicheFlow over RPCFlow

Although RPCFlow achieves competitive performance on spatial metrics such as PSD and SPD, it lacks the essential capability of meaningful conditional generation. In RPCFlow, conditioning is performed using randomly sampled point clouds from the spatial regions without explicit microenvironment structure. As shown in Fig. 14, when conditioned on such random sources, RPCFlow tends to reconstruct the entire target tissue, rather than capturing local dynamics driven by the source input. This undermines its ability to model spatiotemporal evolution in a biologically grounded manner.

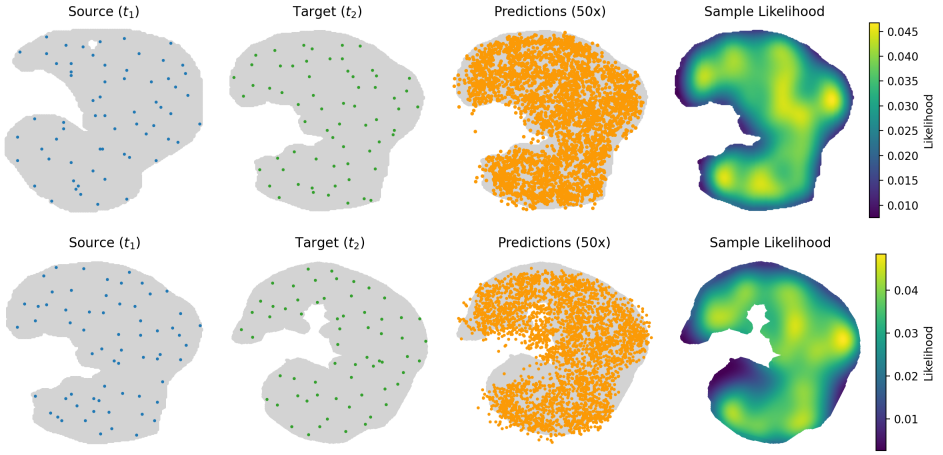


Figure 14: RPCFlow fails to perform meaningful conditional generation: the model generates a diffuse reconstruction resembling the entire target tissue rather than conditioning on the provided source points. Each row shows a different source-target pair.

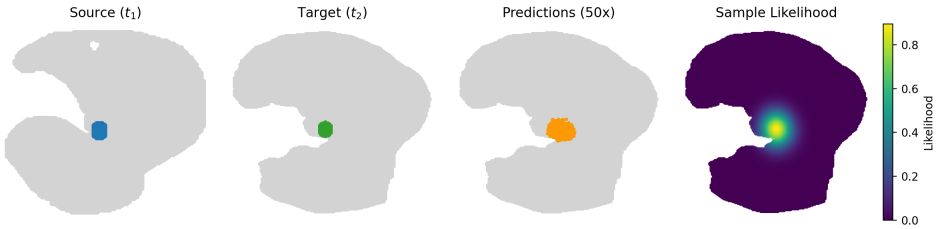
In contrast, NicheFlow is explicitly designed to push localized microenvironments through time. By conditioning on fixed-radius neighborhoods centered around specific spatial positions, the model learns how cellular contexts evolve, preserving both spatial coherence and transcriptional identity. Figure 15 visualizes this distinction: while NicheFlow consistently generates well-localized predictions aligned with the input microenvironment, RPCFlow fails to retain spatial specificity, often diffusing the prediction across broader regions.

From a biological perspective, predicting the fate of a local tissue region over time is far more relevant than mapping random point sets. Microenvironments encode structured cellular contexts, such as signaling interactions or niche-specific cell states, that are crucial for downstream analysis (e.g., lineage fate prediction, microenvironment-based intervention simulation). Because RPCFlow lacks this interpretability and fails to enable such downstream tasks, it cannot serve as a practical generative model in biological settings.

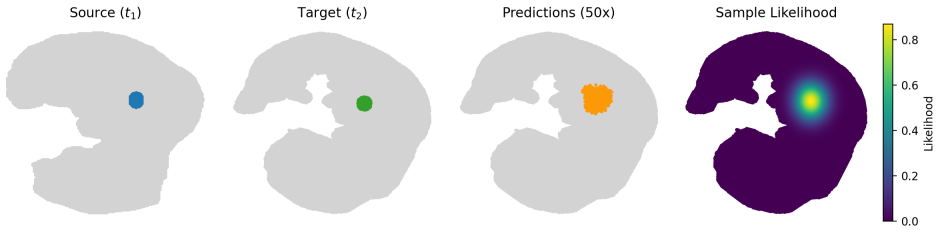
In summary, while RPCFlow may appear performant under some aggregate spatial metrics, only NicheFlow enables localized, conditionally consistent generation of evolving tissue regions. This makes it not only superior for evaluation but also for practical use in biological modeling.

D.8 Structure-aware evaluation

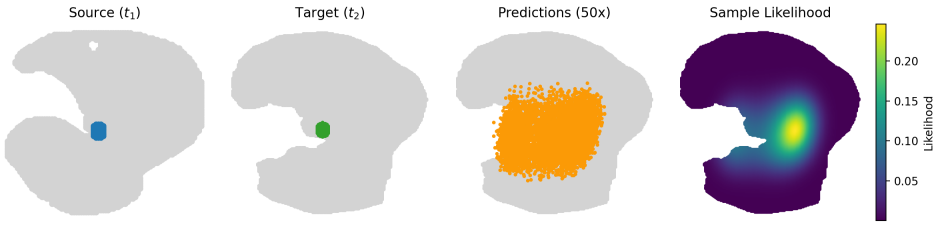
While we refer to our primary evaluation metrics as PSD and SPD, they correspond to the two asymmetric directions of the Chamfer Distance (CD) - a widely used metric in point cloud generation [46]. PSD measures how closely the predicted points adhere to the ground truth structure (fidelity), while SPD captures how well the prediction covers the full extent of the target (coverage). We



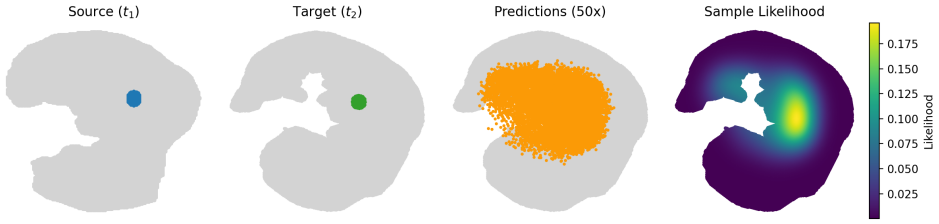
(a) NicheFlow – Sample 1



(b) NicheFlow – Sample 2



(c) RPCFlow – Sample 1



(d) RPCFlow – Sample 2

Figure 15: Comparison between NicheFlow and RPCFlow in a fixed-source microenvironment setting. Each row shows input (source), ground truth target, model prediction, and KDE-estimated likelihood. While NicheFlow (top) produces well-localized samples consistent with the input, RPCFlow (bottom) generates less structured and spatially inaccurate predictions.

compute both metrics over collections of microenvironments tiled across the tissue, providing an indirect assessment of global morphological accuracy.

To directly assess the preservation of internal structure within generated microenvironments, we additionally evaluate Gromov-Wasserstein (GW) and Fused Gromov-Wasserstein (FGW) distances [47, 40]. These metrics compare the intra-point pairwise distances in predicted and ground truth point clouds, capturing latent structural relationships. FGW further incorporates transcriptomic similarity, offering a holistic measure of structural and feature-based alignment.

Given the computational cost of GW and FGW, we restrict their evaluation to microenvironment-level generations. For each source niche, we sample predictions from the trained model and compute GW/FGW against the corresponding real microenvironment, averaging results across multiple runs.

The source–target pairs are obtained by computing OT between all microenvironments at time t and $t+1$, yielding a one-to-one coupling used for conditioning and evaluation.

As SPFlow generates individual points and RPCFlow produces randomly sampled, unstructured clouds, these models lack coherent internal structure and are not amenable to structural comparison via GW or FGW. We therefore report these metrics only for models that generate full microenvironments.

The results in Tab. 5 show that NicheFlow, particularly when trained with the GLVFM objective, consistently achieves lower GW and FGW distances compared to baseline models. These findings underscore the structural fidelity of our predictions and further validate the modeling advantages of microenvironment-aware generative training.

Table 5: Gromov–Wasserstein (GW) and Fused Gromov–Wasserstein (FGW) distances between generated and real microenvironments on mouse embryonic development (MED), axolotl brain development (ABD), and mouse brain aging (MBA). All models are trained with a fixed $\lambda = 0.1$. Results are averaged over five generation runs. Bold indicates the best (lowest) value per column.

		MED		ABD		MBA	
Model	Objective	GW (10^2)	FGW	GW (10^2)	FGW	GW (10^2)	FGW
NicheFlow	CFM	0.315 ± 0.003	3.273 ± 0.003	0.783 ± 0.002	3.543 ± 0.004	0.315 ± 0.001	3.531 ± 0.000
NicheFlow	GVFM	0.463 ± 0.004	3.167 ± 0.007	0.797 ± 0.006	**3.414 ± 0.003**	0.339 ± 0.000	3.387 ± 0.000
NicheFlow	GLVFM	**0.224 ± 0.001**	**3.147 ± 0.007**	**0.720 ± 0.004**	3.420 ± 0.003	**0.262 ± 0.000**	**3.367 ± 0.000**

D.9 Wasserstein metrics

To complement the local structural assessments with GW and FGW, we additionally include global evaluation metrics that are widely adopted in spatial transcriptomics [48–50]. Specifically, we report the 1-Wasserstein ($\mathcal{W}_1$) and 2-Wasserstein ($\mathcal{W}_2$) distances between predicted and real cell distributions, computed separately over spatial coordinates and gene expression features. These metrics are calculated at the level of individual cell types and averaged across all timepoints and generated samples. This provides a more global perspective on how well the model reconstructs tissue-wide spatial and transcriptional distributions.

Table 6: Comparison of Wasserstein distances between generated and real tissue structures on the mouse embryonic development (MED) dataset. We compute $\mathcal{W}_1$ and $\mathcal{W}_2$ distances separately for spatial coordinates (Pos.) and gene expression features (Genes), averaged across all generated cell types and timepoints. All models are trained with a fixed value of $\lambda = 0.1$, and results are averaged over five generation runs. Bold indicates the best (lowest) value per column.

Model	Obj.	$\mathcal{W}_1$ Pos.	$\mathcal{W}_1$ Genes	$\mathcal{W}_2$ Pos.	$\mathcal{W}_2$ Genes
SPFlow	CFM	0.634 ± 0.004	6.545 ± 0.004	0.773 ± 0.005	5.585 ± 0.005
SPFlow	GVFM	0.612 ± 0.005	6.517 ± 0.002	0.743 ± 0.005	5.609 ± 0.113
SPFlow	GLVFM	0.613 ± 0.001	6.457 ± 0.003	0.738 ± 0.001	5.546 ± 0.108
RPCFlow	CFM	0.290 ± 0.006	6.303 ± 0.004	0.460 ± 0.007	5.386 ± 0.004
RPCFlow	GVFM	0.258 ± 0.004	6.134 ± 0.004	0.398 ± 0.006	5.292 ± 0.114
RPCFlow	GLVFM	0.221 ± 0.003	6.087 ± 0.002	0.365 ± 0.007	5.244 ± 0.113
NicheFlow	CFM	0.253 ± 0.002	6.177 ± 0.001	0.420 ± 0.003	5.314 ± 0.111
NicheFlow	GVFM	0.222 ± 0.004	5.985 ± 0.005	0.352 ± 0.006	5.173 ± 0.122
NicheFlow	GLVFM	**0.212 ± 0.004**	**5.930 ± 0.003**	**0.342 ± 0.007**	**5.031 ± 0.003**

We evaluate all model variants and training objectives across the three datasets (MED, ABD, MBA) and report the results in Tabs. 6 to 8. NicheFlow with the GLVFM objective consistently achieves the lowest Wasserstein distances across nearly all dimensions in the embryonic development dataset (MED), outperforming both baseline models and alternative objectives.

In the ABD and MBA datasets, which exhibit more gradual spatial evolution or less distinct cell-type boundaries, we observe that RPCFlow sometimes matches or marginally outperforms NicheFlow in isolated metrics. However, as we discussed in App. D.7, RPCFlow is not a conditionally consistent model: it does not produce meaningful trajectories given a source microenvironment and lacks

Table 7: Comparison of Wasserstein distances between generated and real tissue structures on the axolotl brain development (ABD) dataset. We report $\mathcal{W}_1$ and $\mathcal{W}_2$ distances separately for spatial positions (Pos.) and gene expression features (Genes), averaged across all generated cell types and timepoints. All models use $\lambda = 0.1$, and results are averaged over five generation runs. Bold indicates the best (lowest) value per column.

Model	Obj.	$\mathcal{W}_1$ Pos.	$\mathcal{W}_1$ Genes	$\mathcal{W}_2$ Pos.	$\mathcal{W}_2$ Genes
SPFlow	CFM	0.474 ± 0.003	6.711 ± 0.002	0.573 ± 0.003	6.096 ± 0.081
SPFlow	GVFM	0.538 ± 0.002	6.681 ± 0.003	0.639 ± 0.003	6.209 ± 0.099
SPFlow	GLVFM	0.489 ± 0.002	6.533 ± 0.002	0.586 ± 0.003	6.090 ± 0.166
RPCFlow	CFM	**0.234 ± 0.003**	6.363 ± 0.004	0.374 ± 0.005	6.031 ± 0.089
RPCFlow	GVFM	0.247 ± 0.002	6.093 ± 0.006	**0.369 ± 0.005**	5.921 ± 0.051
RPCFlow	GLVFM	0.254 ± 0.002	**6.070 ± 0.004**	0.406 ± 0.003	**5.884 ± 0.042**
NicheFlow	CFM	0.267 ± 0.002	6.347 ± 0.003	0.428 ± 0.005	6.107 ± 0.003
NicheFlow	GVFM	0.284 ± 0.004	6.081 ± 0.004	0.448 ± 0.006	5.917 ± 0.036
NicheFlow	GLVFM	0.279 ± 0.006	6.085 ± 0.003	0.444 ± 0.011	5.923 ± 0.034

Table 8: Comparison of Wasserstein distances between generated and real tissue structures on the mouse brain aging (MBA) dataset. We compute $\mathcal{W}_1$ and $\mathcal{W}_2$ distances for spatial coordinates (Pos.) and gene expression features (Genes), averaging per cell type and timepoint. All models are trained with $\lambda = 0.1$, and results are averaged across five generation runs. Bold marks the lowest value in each column.

Model	Obj.	$\mathcal{W}_1$ Pos.	$\mathcal{W}_1$ Genes	$\mathcal{W}_2$ Pos.	$\mathcal{W}_2$ Genes
SPFlow	CFM	0.515 ± 0.001	7.544 ± 0.004	0.600 ± 0.002	5.237 ± 0.056
SPFlow	GVFM	0.564 ± 0.001	7.461 ± 0.004	0.646 ± 0.001	4.961 ± 0.114
SPFlow	GLVFM	0.552 ± 0.002	7.431 ± 0.003	0.632 ± 0.002	**4.117 ± 0.196**
RPCFlow	CFM	**0.385 ± 0.003**	7.318 ± 0.006	**0.480 ± 0.003**	6.659 ± 0.098
RPCFlow	GVFM	0.416 ± 0.003	7.202 ± 0.003	0.510 ± 0.003	6.346 ± 0.061
NicheFlow	CFM	0.416 ± 0.008	7.321 ± 0.012	0.510 ± 0.008	6.487 ± 0.084
NicheFlow	GVFM	0.412 ± 0.004	**7.102 ± 0.001**	0.503 ± 0.004	6.403 ± 0.087
NicheFlow	GLVFM	0.431 ± 0.008	7.121 ± 0.008	0.527 ± 0.007	6.298 ± 0.029

biological interpretability. Therefore, even when RPCFlow achieves slightly lower Wasserstein distances in aggregate, these gains are not actionable in practice, as the model cannot be used to infer biologically meaningful transitions or trace the evolution of specific spatial regions.

Taken together, Wasserstein metrics offer a complementary, global validation of the structural and semantic accuracy of NicheFlow. They reinforce the quantitative evidence already provided by Chamfer-based metrics (PSD/SPD) and Gromov-based structural evaluations (GW/FGW), confirming that our model accurately captures the tissue-scale organization of both spatial and gene expression patterns.

E Algorithms

Here, we present the algorithmic procedures underlying NicheFlow. To streamline the exposition, we first define compact notation for representing noisy microenvironments and their interpolations.

We denote a single noisy microenvironment (simplifying Eq. (11)) as:

$$\mathcal{M}^z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{D+2})^{1\times k} = \{(\boldsymbol{c}_i^z, \boldsymbol{z}_i) \mid \boldsymbol{c}_i^z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_2),\ \boldsymbol{z}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_D),\ \forall i = 1, \dots, k\}, \tag{33}$$

where k denotes the number of spatial points per microenvironment, $\boldsymbol{c}_i^z$ are 2D cell coordinates, and $\boldsymbol{z}_i$ are associated feature vectors in $\mathbb{R}^D$. For a collection of N microenvironments, we define the corresponding set of noisy microenvironments

$$\boldsymbol{\mathcal{M}}^{\boldsymbol{z}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{D+2})^{N\times k} = \left\{\mathcal{M}_i^z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{D+2})^{1\times k} \mid \forall i = 1, \dots, N\right\}. \tag{34}$$

Given a noisy sample $\mathcal{M}^z$ and a corresponding ground-truth microenvironment $\mathcal{M}^1$, we define their linear interpolation at time $t \in [0, 1]$ as: $\mathcal{M}^t = (1 - t)\mathcal{M}^z + t\mathcal{M}^1$, where the interpolation is

performed element-wise across the matrix rows. For batched data, this generalizes to:

$$\boldsymbol{\mathcal{M}}^t = (1-t)\boldsymbol{\mathcal{M}}^z + t\boldsymbol{\mathcal{M}}^1. \tag{35}$$

Algorithm 1 SAMPLEANDINTERPOLATE

Input: Number of samples N, feature dimension D, microenvironment size k, OT plan $\pi^*_{\epsilon,\lambda}$
Output: Source ($\boldsymbol{\mathcal{M}}^0$), target ($\boldsymbol{\mathcal{M}}^1$), interpolated ($\boldsymbol{\mathcal{M}}^t$), and noisy ($\boldsymbol{\mathcal{M}}^z$) microenvironments

$(\boldsymbol{\mathcal{M}}^0, \boldsymbol{\mathcal{M}}^1) \leftarrow$ Sample from K-Means regions ▷ Initial microenvironment pairs
$(\boldsymbol{\mathcal{M}}^0, \boldsymbol{\mathcal{M}}^1) \leftarrow \pi^*_{\epsilon,\lambda}(\boldsymbol{\mathcal{M}}^0, \boldsymbol{\mathcal{M}}^1)$ ▷ Resample with OT plan
$\boldsymbol{\mathcal{M}}^z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{D+2})^{N\times k}$ ▷ Noisy initial states
$t \sim \mathcal{U}(0,1)$ ▷ Random interpolation time
$\boldsymbol{\mathcal{M}}^t \leftarrow (1-t)\boldsymbol{\mathcal{M}}^z + t\boldsymbol{\mathcal{M}}^1$ ▷ Linearly interpolated states

E.1 OT Conditional Flow Matching

The OT Conditional Flow Matching (OT-CFM) algorithm consists of a training and a generation phase. During training, the model learns a time-dependent vector field u_t^θ conditioned on a source microenvironment $\mathcal{M}^0$, which maps a noisy microenvironment $\mathcal{M}^z$—sampled from Gaussian noise—to its corresponding target microenvironment $\mathcal{M}^1$. The supervision is provided via source-target pairs $(\mathcal{M}^0, \mathcal{M}^1)$ obtained through EOT over pooled microenvironment representations. At generation time, the learned vector field is integrated starting from $\mathcal{M}^z$, conditioned on a given source $\mathcal{M}^0$, to generate the predicted target microenvironment $\mathcal{M}^1$.

We optimize the following loss:

$$\mathcal{L}(\boldsymbol{\mathcal{M}}^0, \boldsymbol{\mathcal{M}}^1, \boldsymbol{\mathcal{M}}^t, \boldsymbol{\mathcal{M}}^z; \theta) = \frac{1}{2} \sum_{\substack{\mathcal{M}^0\in\boldsymbol{\mathcal{M}}^0 \\ \mathcal{M}^z\in\boldsymbol{\mathcal{M}}^z \\ \mathcal{M}^t\in\boldsymbol{\mathcal{M}}^t \\ \mathcal{M}^1\in\boldsymbol{\mathcal{M}}^1}} \left\| u_t^\theta(\mathcal{M}^t, \mathcal{M}^0) - (\mathcal{M}^1 - \mathcal{M}^z) \right\|_2^2 \tag{36}$$

The full pseudocode for both phases is provided in Algs. 2 and 3.

Algorithm 2 OT CFM — Training

Input: Number of samples N, feature dimension D, microenvironment size k, OT plan $\pi^*_{\epsilon,\lambda}$, conditional velocity field u_t^θ
Output: Trained parameters θ of u_t^θ

$(\boldsymbol{\mathcal{M}}^0, \boldsymbol{\mathcal{M}}^1, \boldsymbol{\mathcal{M}}^t, \boldsymbol{\mathcal{M}}^z) \leftarrow$ SAMPLEANDINTERPOLATE$(N, D, k, \pi^*_{\epsilon,\lambda})$ ▷ Microenvironments (Alg. 1)
$\theta \leftarrow \nabla_\theta \mathcal{L}(\boldsymbol{\mathcal{M}}^0, \boldsymbol{\mathcal{M}}^1, \boldsymbol{\mathcal{M}}^t, \boldsymbol{\mathcal{M}}^z; \theta)$ ▷ Compute loss (Eq. (36)) & update parameters θ

Algorithm 3 OT CFM — Generation

Input: Source microenvironment $\mathcal{M}^0$, learned conditional velocity field u_t^θ
Output: Generated microenvironment $\mathcal{M}^1$

$\mathcal{M}^z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{D+2})^{1\times k}$ ▷ Sample noisy sample
$\mathcal{M}^1 \leftarrow \mathcal{M}^z + \int_0^1 u_t^\theta(\mathcal{M}^t, \mathcal{M}^0)\,\mathrm{d}t$ ▷ Solve ODE

E.2 OT Gaussian Variational Flow Matching

The OT Gaussian Variational Flow Matching (OT-GVFM) algorithm adopts a variational perspective on Flow Matching. Instead of directly learning a time-dependent conditional velocity field, the model learns a factorized variational posterior $q_t^\theta(\mathcal{M}^1 \mid \mathcal{M}^t, \mathcal{M}^0)$ over target microenvironments $\mathcal{M}^1$, conditioned on an interpolated microenvironment $\mathcal{M}^t$ and a source microenvironment $\mathcal{M}^0$. The predicted velocity field is then computed as the difference between the posterior mean $\mu_t^\theta(\mathcal{M}^t, \mathcal{M}^0)$ and the current state $\mathcal{M}^1$.

The training objective minimizes the discrepancy between ground-truth targets and the predicted posterior means $(\bar{\boldsymbol{f}}_t^\theta, \bar{\boldsymbol{r}}_t^\theta)$:

$$\mathcal{L}(\boldsymbol{\mathcal{M}}^0, \boldsymbol{\mathcal{M}}^1, \boldsymbol{\mathcal{M}}^t; \mu_t^\theta) = \frac{1}{2} \sum_{\substack{\mathcal{M}^0 \in \boldsymbol{\mathcal{M}}^0 \\ \mathcal{M}^t \in \boldsymbol{\mathcal{M}}^t \\ \mathcal{M}^1 \in \boldsymbol{\mathcal{M}}^1}} \sum_{\substack{(\boldsymbol{c}_1, \boldsymbol{x}_1) \in \mathcal{M}^1 \\ (\bar{\boldsymbol{f}}_t^\theta, \bar{\boldsymbol{r}}_t^\theta) \in \mu_t^\theta(\mathcal{M}^t, \mathcal{M}^0)}} \left(\|\boldsymbol{c}_1 - \bar{\boldsymbol{f}}_t^\theta\|_2^2 + \|\boldsymbol{x}_1 - \bar{\boldsymbol{r}}_t^\theta\|_2^2 \right). \tag{37}$$

At generation time, trajectories are generated by integrating the vector field $\mu_t^\theta(\mathcal{M}^t, \mathcal{M}^0) - \mathcal{M}^t$, starting from a noise-sampled microenvironment $\mathcal{M}^z$ and conditioned on a given source $\mathcal{M}^0$. To ensure stability near $t = 1$, the vector field is scaled by a time-dependent denominator, as in prior VFM formulations.

The pseudocode for both phases is provided in Algs. 4 and 5.

Algorithm 4 OT-GVFM — Training

Input: Number of samples N, feature dimension D, microenvironment size k, OT plan $\pi^*_{\epsilon,\lambda}$, source-conditioned posterior mean predictor μ_t^θ
Output: Trained parameters θ of μ_t^θ

$(\boldsymbol{\mathcal{M}}^0, \boldsymbol{\mathcal{M}}^1, \boldsymbol{\mathcal{M}}^t, \boldsymbol{\mathcal{M}}^z) \leftarrow$ SAMPLEANDINTERPOLATE$(N, D, k, \pi^*_{\epsilon,\lambda})$ ▷ Microenvironments (Alg. 1)
$\theta \leftarrow \nabla_\theta \mathcal{L}(\boldsymbol{\mathcal{M}}^0, \boldsymbol{\mathcal{M}}^1, \boldsymbol{\mathcal{M}}^t; \mu_t^\theta)$ ▷ Compute loss (Eq. (37)) & update parameters θ

Algorithm 5 OT-GVFM — Generation

Input: Source microenvironment $\mathcal{M}^0$, learned source-conditioned posterior mean predictor μ_t^θ
Output: Generated microenvironment $\mathcal{M}^1$

$\mathcal{M}^z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{D+2})^{1\times k}$ ▷ Sample noisy sample
$\mathcal{M}^1 \leftarrow \mathcal{M}^z + \int_0^1 \frac{\mu_t^\theta(\mathcal{M}^t, \mathcal{M}^0) - \mathcal{M}^t}{1-t+\epsilon} \mathrm{d}t$ ▷ Solve ODE

E.3 NicheFlow: OT Gaussian-Laplace Variational Flow Matching

NicheFlow extends OT-GVFM by modifying the variational posterior: it assumes a Gaussian distribution for gene expression features and a Laplace distribution for spatial coordinates. This change leads to a hybrid loss that combines an L^2 loss on gene expression and an L^1 loss on spatial locations:

$$\mathcal{L}(\boldsymbol{\mathcal{M}}^0, \boldsymbol{\mathcal{M}}^1, \boldsymbol{\mathcal{M}}^t; \theta) = \sum_{\substack{\mathcal{M}^0 \in \boldsymbol{\mathcal{M}}^0 \\ \mathcal{M}^t \in \boldsymbol{\mathcal{M}}^t \\ \mathcal{M}^1 \in \boldsymbol{\mathcal{M}}^1}} \sum_{\substack{(\boldsymbol{c}_1, \boldsymbol{x}_1) \in \mathcal{M}^1 \\ (\bar{\boldsymbol{f}}_t^\theta, \bar{\boldsymbol{r}}_t^\theta) \in \mu_t^\theta(\mathcal{M}^t, \mathcal{M}^0)}} \left(\|\boldsymbol{c}_1 - \bar{\boldsymbol{f}}_t^\theta\|_1 + \frac{1}{2}\|\boldsymbol{x}_1 - \bar{\boldsymbol{r}}_t^\theta\|_2^2 \right) \tag{38}$$

At generation time, the model integrates the velocity field defined by the difference between the predicted mean $\mu_t^\theta(\mathcal{M}^t, \mathcal{M}^0)$ and the current state $\mathcal{M}^1$, starting from noise and conditioned on the source $\mathcal{M}^0$, identical to the OT-GVFM procedure.

F Experimental setup

F.1 Dataset description

We evaluate our model on three publicly available, time-resolved spatial transcriptomics datasets spanning development and aging processes. Each dataset provides single-cell resolution profiles with matched spatial coordinates and curated cell type annotations. A detailed summary of the organism, tissue, number of time points, cell types, total number of cells, and acquisition technology for each dataset is provided in Table 9.

F.2 Dataset and microenvironments preprocessing

Dataset preprocessing. All datasets used in our study underwent a preprocessing procedure appropriate for spatial transcriptomic analysis, involving total count normalization, logarithmic

Table 9: Overview of the time-resolved spatial transcriptomics datasets used in our experiments. MED: Mouse Embryonic Development, ABD: Axolotl Brain Development, MBA: Mouse Brain Aging. Each dataset varies in organism, tissue type, number of timepoints, cell types, total number of cells, and spatial transcriptomics technology.

Dataset	Organism	Tissue	Timepoints	Cell Types	Cells	Technology
MED	Mouse	Whole embryo	3	24	54k	Stereo-seq
ABD	Axolotl	Brain	6	33	36k	Stereo-seq
MBA	Mouse	Brain	20	18	1.5M	MERFISH

transformation, and principal component analysis (PCA). For the mouse embryonic development [20] and axolotl brain development [42] datasets, total count normalization, and logarithmic transformation had already been applied; we, therefore, performed PCA ourselves on the transformed data. For the mouse brain aging dataset [43], we applied all three steps: we first normalized raw count matrices so that each cell had the same total expression, followed by a natural logarithm transformation of the form $\log(X + 1)$ to stabilize variance and mitigate the influence of large values. We then computed PCA on the log-transformed data.

To reduce computational overhead due to high cell counts in the aging dataset, we subsampled the data by a factor of 0.2.

Finally, we standardized PCA components across all cells to ensure consistent scaling across time points. Spatial coordinates were standardized independently for each time point by subtracting the per-time-point mean and dividing by the standard deviation. This standardization preserves the relative spatial configuration while accounting for scale and position differences over developmental time.

Microenvironments preprocessing. To facilitate efficient training and enable consistent microenvironment construction, we precompute all fixed-radius neighborhoods for each dataset using a radius r chosen based on spatial resolution. To reduce variability in the number of neighbors and improve batching efficiency, we fix the number of nodes per microenvironment to the most frequent neighbor count observed within each slide. This standardization ensures structural comparability across microenvironments while significantly reducing computational overhead during training, as costly radius or nearest-neighbor queries are avoided at runtime.

F.3 Quantitative evaluation metrics

We assess spatial fidelity using two complementary asymmetric distance measures. The *point-to-shape distance* (PSD) captures how much predicted cell positions diverge from the actual tissue layout, computed as the average squared distance from each simulated point to its nearest neighbor in the ground truth. Conversely, the *shape-to-point distance* (SPD) quantifies how comprehensively the predicted distribution spans the target structure by averaging the squared distance from each ground truth point to its closest generated counterpart.

Let $\mathcal{G}^t$ denote the set of generated coordinates and $\mathcal{R}^t$ the set of ground truth coordinates at time t. Define $\mathrm{NN}^t_{\mathrm{ref}}(\boldsymbol{c}_i)$ as the nearest neighbor of $\boldsymbol{c}_i \in \mathcal{G}^t$ in $\mathcal{R}^t$, and $\mathrm{NN}^t_{\mathrm{gen}}(\boldsymbol{c}_i)$ as the nearest neighbor of $\boldsymbol{c}_j \in \mathcal{R}^t$ in $\mathcal{G}^t$. Then, the two metrics are given by:

$$\mathrm{PSD} = \frac{1}{|\mathcal{G}|} \sum_{\mathcal{G}^t \in \mathcal{G}} \sum_{\boldsymbol{c}_i \in \mathcal{G}^t} \|\boldsymbol{c}_i - \mathrm{NN}^t_{\mathrm{ref}}(\boldsymbol{c}_i)\|_2^2, \tag{39}$$

$$\mathrm{SPD} = \frac{1}{|\mathcal{R}|} \sum_{\mathcal{R}^t \in \mathcal{R}} \sum_{\boldsymbol{c}_j \in \mathcal{R}^t} \|\boldsymbol{c}_j - \mathrm{NN}^t_{\mathrm{gen}}(\boldsymbol{c}_j)\|_2^2. \tag{40}$$

where $\mathcal{G} := \cup_{t \in \mathcal{T}} \mathcal{G}^t$ and $\mathcal{R} := \cup_{t \in \mathcal{T}} \mathcal{R}^t$.

F.4 Cell-type classification for evaluation

To evaluate cell-type fidelity of generated microenvironments, we train a supervised classifier to assign cell type labels based on gene expression profiles. We apply the same preprocessing steps

used for training our generative models: total count normalization, log-transformation, and PCA (see App. F.2). The resulting low-dimensional embeddings are used as input features for a simple multilayer perceptron (MLP), trained to predict discrete cell type labels.

The classifier consists of a two-layer feedforward network with ReLU activations and a final linear projection to the number of cell types. It is trained using cross-entropy loss and optimized with the AdamW optimizer. We report performance using the weighted F1-score.

We obtain strong classification results across all datasets. On the mouse embryonic development dataset, the classifier achieves a weighted F1-score of 0.85; on axolotl brain development, 0.80; and on the aging dataset, 0.97. These results correlate with the number of input genes and the variance retained in the PCA-reduced space. The aging dataset contains only 300 genes, and 50 principal components explain sufficient variance to accurately distinguish most cell types. In contrast, the embryonic development dataset contains approximately 2,000 genes, and the axolotl brain development dataset includes over 12,700 genes, making classification more challenging due to higher gene expression variability.

F.5 Discretized microenvironments

To ensure consistent and reproducible evaluation across methods and datasets, we construct a fixed set of evaluation microenvironments by discretizing the spatial domain of each tissue section. For each time point, we define a regular 2D grid over the tissue and select the closest cell to each grid point as the centroid of a microenvironment. Each centroid is then used to construct a fixed-radius neighborhood, following the microenvironment definition in Section 4.1. This procedure ensures full spatial coverage by verifying that every cell belongs to at least one microenvironment.

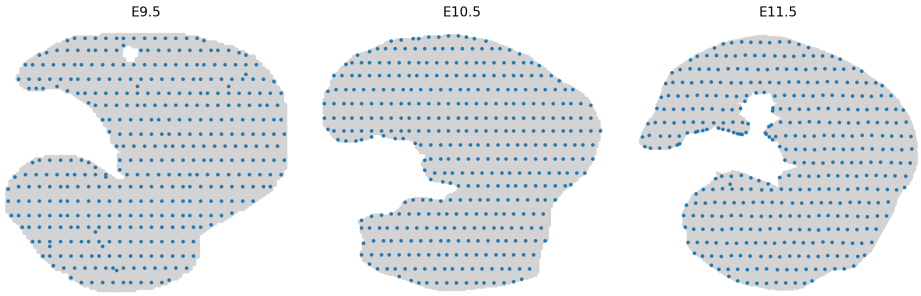


Figure 16: Discretized grid of microenvironments for the mouse embryonic development dataset. Each blue point denotes a centroid around which a microenvironment is constructed. To ensure consistent coverage across tissue sections, additional centroids are randomly sampled such that each slide contains the same number of microenvironments.

Figure 16 illustrates this discretization for the mouse embryonic development dataset. Each blue dot corresponds to a selected centroid. In cases where the number of grid-based centroids falls below a target threshold, additional centroids are randomly sampled to match a fixed total count per slide. This augmentation ensures that all slides contribute equally to the evaluation and prevents bias from sparse regions.

We apply the same discretization procedure to all three datasets used in our experiments: mouse embryogenesis, axolotl brain development, and mouse brain aging. By standardizing the evaluation regions spatially and deterministically, we eliminate the need for stochastic region sampling during evaluation, which would otherwise lead to nondeterministic and irreproducible results.

F.6 Microenvironment Transformer

To model the spatiotemporal evolution of local cellular neighborhoods, we design a permutation-equivariant transformer-based architecture tailored to structured point cloud data. Our Microenvironment Transformer processes local microenvironments—sets of cells with gene expression features and spatial coordinates—and predicts time-dependent outputs such as velocity fields or future states.

The model operates on a source $\mathcal{M}^0$ and noisy $\mathcal{M}^z$ microenvironments with per-cell features $\boldsymbol{x}_i \in \mathbb{R}^D$ and 2D coordinates $\boldsymbol{c}_i \in \mathbb{R}^2$. The architecture consists of the following components:

1. **Input Embeddings:**
   (a) **Feature Embedding:** A linear transformation is applied to the input features $\boldsymbol{x}_i$ of each cell.
   (b) **Coordinate Embedding:** Spatial coordinates $\boldsymbol{c}_i$ are linearly projected and concatenated to the feature embedding.
   (c) **Time Embedding:** For the noisy microenvironment only, time $t \in [0, 1]$ is encoded using sinusoidal functions $\cos(\omega t)$ and $\sin(\omega t)$, followed by a linear projection and concatenation with the input embedding.
2. **Transformer Encoder (Source Microenvironment):**
   (a) **Self-Attention:** A stack of transformer encoder blocks with multi-head self-attention processes the embedded source microenvironment.
   (b) **No Time Embedding:** Time information is *not* provided to the encoder, as it encodes the source $\mathcal{M}^0$.
   (c) **Residual Feedforward:** Each block contains a feedforward subnetwork with LeakyReLU activation and a residual connection.
   (d) **Layer Normalization:** Applied after both the attention and feedforward layers.
   (e) **Masking:** Binary masks are used to ignore padding in variable-length microenvironments.
3. **Transformer Decoder (Noisy Microenvironment):**
   (a) **Time Embedding:** Temporal context is injected into the decoder by embedding the time t and concatenating it to the target point embedding.
   (b) **Cross-Attention:** Decoder layers apply cross-attention between the noisy microenvironment and the encoded source microenvironment.
   (c) **Self-Attention and Feedforward:** Each decoder block includes standard self-attention and residual feedforward layers.
   (d) **Layer Normalization and Masking:** As with the encoder, normalization, and masking are applied throughout.
4. **Final Output Projection:**
   (a) **Prediction Head:** A linear layer maps the decoder outputs to the desired dimensionality.

This architecture allows for flexible and expressive modeling of temporal dynamics in cellular point clouds. By encoding only the source and decoding the temporally conditioned target, the model supports variational and flow-based training objectives with explicit temporal conditioning.

F.7 Hyperparameters and Computational Costs

Model hyperparameters. For all experiments, we use the same configuration for the Microenvironment Transformer architecture. The full set of hyperparameters is as follows:

- **Input feature dimension:** 50 PCA-based gene expression features concatenated with a one-hot encoding of the time-point, resulting in a total dimensionality of $50 + |\mathcal{T}|$, where $|\mathcal{T}|$ is the number of slides (timepoints) in the dataset.
- **Input coordinate dimension:** 2
- **Embedding dimension:** 128
- **MLP hidden dimension:** 256
- **Number of attention heads:** 4
- **Number of encoder layers:** 2
- **Number of decoder layers:** 2
- **Dropout rate:** 0.1
- **Output dimension:** 52 (gene expressions features + coordinates)

OT and mini-batching. To ensure spatial diversity and computational tractability, we uniformly sample 256 source–target microenvironment pairs from the K spatial clusters obtained via K-Means. We then compute the entropic OT plan between these sampled pairs and resample 64 source-target pairs from this plan to define a single training instance. During training, we process 16 such instances per batch.

Optimization. All models are trained using the AdamW optimizer with a learning rate of $2 \cdot 10^{-4}$ and a weight decay of $1 \cdot 10^{-5}$. We train each model until convergence.

Computational cost. All models were trained on a single NVIDIA GeForce GTX 1080 Ti GPU with 11GB of memory. Depending on the dataset and training objective (e.g., CFM or VFM), training takes approximately 12–16 hours per model.

F.8 Comparison with moscot

Here, we describe how we conduct the comparisons with moscot, as shown in Fig. 3, Fig. 4, and Fig. 5. For both NicheFlow and moscot, we select a source microenvironment that we want to track over time. In the case of NicheFlow, this corresponds to an aggregate of point clouds. For moscot, it refers to a group of single cells spatially located within the region of interest. The same set of cells is used for both methods.

NicheFlow. To generate contour plots over the spatial slide, we push forward the selected region and assign the generated points to their nearest real neighbors based on spatial coordinates. We then compute a probability value for each real position by normalizing the number of assigned generated points. In other words, the more generated points that are close to a given real point, the higher the probability assigned to that location in the contour plot. Cell type proportions are computed as the aggregated frequencies of the generated cell types across the slide. Each plot considers 10 independent generation runs from the same niche.

moscot. This baseline is not a generative model, but rather a standard discrete OT framework using a Fused Gromov-Wasserstein cost. As such, it does not generate new features or coordinates. Instead, it outputs a transition matrix that assigns matching probabilities between each source slide cell and each target slide cell. We use these transition probabilities to compute contour plots over the target slide and to aggregate cell type probabilities for the bar plots. For the latter, moscot provides a custom method called `cell_transition()`.

For the Appendix figures Fig. 4,Fig. 7, Fig. 8, and Fig. 9, we propagate the initial source region across multiple time steps. In NicheFlow, this is achieved by using the simulated point cloud from step t to predict the next state at $t + 1$, and then feeding this output as the input for the following trajectory step from $t + 1$ to $t + 2$, and so on. Ground truth points are not used as intermediate sources during this process. For moscot, pushing the source across time points can be automatically done by setting non-subsequent time points in the `cell_transition()` function.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and introduction state the core contributions. These claims are substantiated through targeted experiments and ablation studies.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss the limitations in App. B.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: We introduce the Mixed-Factorized VFM formulation alongside NicheFlow in Sec. 4, and provide complete theoretical derivations in App. C.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: The experimental setup is described in Sec. 5.1.1, with additional details provided in Apps. E and F. We outline the dataset preprocessing, evaluation discretization procedure, backbone architecture, and all hyperparameter and optimizer settings. The appendices also include training and inference algorithms for OT-CFM and OT-VFM under both GVFM and GLVFM objectives.

Guidelines:

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

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

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

   Answer: [Yes]

   Justification: A link to the code is included in our project page, which is reported in the footnotes of page 1.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We report the values used in our experiments in Sec. 5.1.1 and provide additional justification through ablations in Apps. D.5 and D.6 and App. F.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We report the mean and standard deviation across five independent evaluation runs for all experiments and evaluation metrics.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide details on the hardware that has been used in this research in App. F.7.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The research presented in this work conforms with the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: We discuss the broader impacts of our work in App. A.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

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

11. **Safeguards**

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

    Answer: [NA]

    Justification: The research discussed in this paper does not require safeguards to be put in place.

    Guidelines:

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

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Where relevant, we acknowledge and cite the original authors of prior work, including associated codebases and datasets, and specify the licenses under which these resources were released.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: We use only publicly available datasets, which are cited appropriately. The complete codebase, including documentation and data preprocessing scripts, is public.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: This work did not conduct research on human subjects or crowdsourcing experiments.

    Guidelines:

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

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

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

    Answer: [NA]

    Justification: This work did not conduct experiments where human subjects were involved and therefore does not require IRB approvals.

    Guidelines:

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

16. **Declaration of LLM usage**

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

    Answer: [NA]

    Justification: LLMs were not used for the development of our core method.

    Guidelines:

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