OTEdit: Entropic Transport Corrected Trajectory for Inversion-Free Flow-Based Image Editing

Rameshwar Mishra
IIIT Delhi
rameshwarm@iiitd.ac.in

Modi Akshay
IIIT Delhi
akshay21116@iiitd.ac.in

A V Subramanyam
IIIT Delhi
subramanyam@iiitd.ac.in

Abstract

Inversion-free methods for text-driven image editing construct direct ODE paths between source and target distributions using pre-trained flow models, but operate on single trajectories vulnerable to noise-induced drift with no mechanism to exploit the geometry of plausible edits. We propose Transport-Guided Flow Editing, which recasts the editing problem in distributional terms: we construct source and target particle clouds in latent space, solve entropic optimal transport between them with an edit-direction-aware cost, and use the resulting barycentric map to define a time-varying correction field for the editing ODE. The transport plan adapts online at each integration step, and the correction strength is governed by a variational co-state derived from the OT anchor, responding to accumulated trajectory deviation rather than following a fixed schedule. The output is a single edited image, but one whose trajectory has been steered by distributional information inaccessible to single-trajectory methods. Extensive experiments demonstrate state-of-the-art structure preservation with superior semantic alignment and consistent human preference over existing baselines.

1 Introduction

Text-based image editing seeks to transform a source image according to a natural language instruction while preserving structure and identity. From a generative modelling perspective, this amounts to constructing a mapping between two conditional distributions over a shared image manifold defined by the source and edit prompts. Flow matching [1–3] and its conditional variants [4] provide a principled framework via continuous normalising flows (CNFs), in which a neural velocity field v_θ transports samples through an ODE. Modern text-to-image flow models (e.g., Stable Diffusion 3 [4], FLUX [5]) learn noise-to-image transport. Editing, however, requires transport between image distributions, necessitating either additional training or reuse of pretrained flows.

A common strategy is inversion, which maps the source image to noise, then regenerate under the target prompt [6–8]. Discretisation errors accumulate in the forward ODE and amplify during reversal, degrading fidelity and structural preservation even when the true noise is known [6, 9]. Extensions improve inversion accuracy [7], inject intermediate representations [10–12], or apply optimal-control guidance [13], but are often architecture- or sampler-specific.

Recent work avoids inversion by directly constructing ODEs between source and target distributions. FlowEdit [9] derives a direct path using differences of conditional flows, improving transport efficiency but requiring careful tuning of guidance scales and timestep schedules. FlowAlign [14] stabilises trajectories via optimal control with a terminal cost, yielding a decomposition into semantic guidance and structure-preserving terms. Other approaches include SDEdit [15] and DDIB [16], which traverse noise space, and score-distillation methods (DDS [17], PDS [18], DreamSampler [19]), which are computationally intensive.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



Figure 1: Our editing results : Outputs follow the edit instruction while preserving source fidelity.

However, all these inversion-free methods operate on a single trajectory and have one latent encoding and one ODE solution. Stochasticity is treated as noise rather than a usable signal, and trajectory corrections are either heuristic or derived from objectives that ignore the geometry of the target distribution. Yet for a given source image and edit prompt, there exists a *distribution* of valid edits, and effective editing should exploit the *transport geometry* between source neighbourhoods and the target distribution.

We recast editing as a distributional optimal transport problem. We construct source and target particle clouds by perturbing the encoded source and sampling multiple trajectories under base dynamics, respectively. An entropic OT problem [20, 21] between these clouds yields a coupling that captures compatibility between source perturbations and plausible edits. The resulting barycentric map defines a soft target for each source particle, inducing a time-varying correction field that steers the editing ODE toward transport-consistent outcomes. The method outputs a single edited image, but one whose trajectory has been guided by distributional information that no single-trajectory method can access. Crucially, the OT coupling and the trajectory correction form a coupled system where the co-state derives its correction from the OT-computed barycentric anchor, while the online OT plan adapts to the trajectory position governed by the co-state. Figure 1 illustrates qualitative editing results produced by our method.

Our main contributions are summarised as follows:

- We introduce the first framework that integrates entropic optimal transport into inversion-free flow-based image editing, recasting trajectory correction as a distributional transport problem between source and target particle clouds. The coupling uses a three-term cost with latent proximity, perceptual similarity, and directional CLIP alignment, computed online at every ODE step and is provably non-separable, ensuring it actively steers the transport plan.
- We derive a variational co-state correction from a unified optimal control problem yielding a deviation-adaptive correction that responds to instantaneous trajectory drift rather than following a fixed decay schedule (Proposition 3.2).
- We demonstrate through comprehensive experiments that our method achieves state-of-the-art source structure preservation while maintaining competitive semantic alignment, outperforming FlowEdit, FlowAlign, SplitFlow, and RF-inversion across multiple metrics. A user study confirms consistent human preference of our method over baselines.

Notation. $\mathcal{E}, \mathcal{D}$ denote VAE encoder/decoder, $v_\theta(z, t, c)$ the pretrained flow and $\phi_{\text{img}}, \phi_{\text{txt}}$ CLIP encoders. Time runs from $t = 1$ (source/noise) to $t = 0$ (clean/edited), with $\sigma_t = t$. $\Pi(\cdot, \cdot)$ denotes couplings and $\mathrm{KL}(\cdot\|\cdot)$ the KL divergence.

2 Related Work

Editing by inversion. Inversion-based editing maps the source image to noise and regenerates under a modified prompt [6, 16, 7], with refinements via improved inversion procedures [6, 8, 7]

or attention/feature injection during sampling [11, 10, 22, 12, 23]. Even exact inversion on model-generated images produces unsatisfactory edits [9], and attention-based methods are architecture-specific, limiting portability to newer flow models such as SD3 [4] and FLUX [5]. RF-inversion [13] adds optimal-control-based guidance during inversion but retains the cost of a full forward–backward ODE integration.

Inversion-free editing. SDEdit [15] re-denoises from added noise under the target prompt but requires careful noise-level tuning. FlowEdit [9] constructs a direct source-to-target ODE via differences of conditional velocity fields, reducing transport cost but sensitive to asymmetric CFG scales and timestep skipping. FlowAlign [14] stabilises trajectories using optimal control with a terminal cost on source similarity, deriving a Tweedie-based consistency term via Pontryagin's minimum principle. SplitFlow [24] targets gradient entanglement for complex prompts by decomposing the target into sub-prompts and aggregating their independent flows through a soft projection inspired by multi-task gradient conflict resolution. Methods such as DDS [17], PDS [18], and DreamSampler [19] distil score or velocity iteratively but incur high test-time cost. All operate on a single (possibly aggregated) trajectory and do not model the distribution of plausible edits. Lupaşcu and Stupariu [25] add a cosine-annealed transport term to RF-Inversion and FlowEdit, but this reduces to a scaled displacement toward a fixed latent, a Brenier map between Dirac measures without distributional coupling or cost design. We instead solve entropic OT between particle clouds with an edit-aware cost, yielding an adaptive barycentric map and a trajectory dependent co-state.

Optimal transport and control in generative modelling. OT is widely used in generative modelling: rectified flow [2] straightens paths via an OT view, minibatch OT couplings improve training [26, 27], Schrödinger bridges [28, 29] learn stochastic transport via iterative proportional fitting (related to Sinkhorn [20]), and neural OT parameterises continuous maps [30–32]. Entropic OT has been applied to unpaired translation [33, 34] and OT-guided diffusion conditions denoising with transport plans [35]. On the control side, RF-inversion [31] and FlowAlign [14] use terminal-cost control (yielding a time-invariant co-state), while adjoint methods [36] and controlled diffusion bridges [37] exploit Hamiltonian structure. Prior OT work, however, targets training or generation rather than test-time trajectory correction. We solve entropic OT between particle clouds drawn from editing dynamics and use the barycentric map as an online correction. A running cost penalising deviation from this OT anchor yields a dynamically evolving co-state that accumulates with integrated drift, unlike terminal-cost-only formulations.

3 Method

We augment the inversion-free editing ODE with a correction field derived from entropic optimal transport between empirical particle clouds. We review the base formulation (§3.1), then describe cloud construction (§3.2), the transport cost with online OT (§3.3), and the variational co-state correction (§3.4). An overview of our method is given in Figure 2.

3.1 Background: Inversion-Free Flow Editing

Let $v_\theta(\boldsymbol{z}, t, c)$ denote the pretrained flow velocity conditioned on text embedding c, with $t \in [0,1]$ running from $t=1$ (source) to $t=0$ (edited). Given source image $\boldsymbol{x}_s$ with prompt c_s and edit prompt c_e, we encode $\boldsymbol{z}_s = \mathcal{E}(\boldsymbol{x}_s)$. Following FlowEdit [9] and FlowAlign [14], the editing trajectory is governed by

$$d\boldsymbol{z}_t = \big[v_\theta(\boldsymbol{pd}_t, c_e) - v_\theta(\boldsymbol{q}_t, c_s)\big]\, dt, \quad \boldsymbol{z}_1 = \boldsymbol{z}_s, \tag{1}$$

where $\boldsymbol{q}_t = (1-\sigma_t)\boldsymbol{z}_s + \sigma_t \boldsymbol{\varepsilon}$, $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, and $\boldsymbol{pd}_t = \boldsymbol{z}_t + \boldsymbol{q}_t - \boldsymbol{z}_s$. Classifier-free guidance is applied to the target direction:

$$\boldsymbol{v}_t^{pd} = v_\theta(\boldsymbol{pd}_t, c_s) + \omega\big(v_\theta(\boldsymbol{pd}_t, c_e) - v_\theta(\boldsymbol{pd}_t, c_s)\big), \tag{2}$$

with scale ω. FlowAlign [14] augments this with a Tweedie based source consistency term, yielding a combined step

$$\boldsymbol{d}_{\mathrm{FA}} = \Delta\sigma_k(\boldsymbol{v}_k^{pd} - \boldsymbol{v}_k^{q}) + \zeta(\boldsymbol{q}_{t_k} - \sigma_k \boldsymbol{v}_k^{q} - \boldsymbol{pd}_{t_k} + \sigma_k \boldsymbol{v}_k^{pd}), \tag{3}$$

where $\boldsymbol{v}_k^q = v_\theta(\boldsymbol{q}_{t_k}, c_s)$ and ζ weights the consistency correction. This consistency term is derived from a terminal-cost-only optimal control formulation, which penalises deviation at $t=0$ but not along the trajectory. The resulting correction follows a predetermined schedule $\eta\sigma_t$ that decays

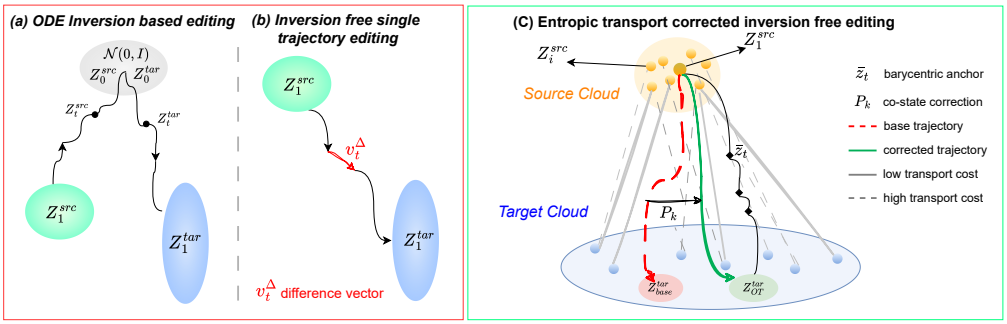


Figure 2: Overview of our method. (a) Inversion-based methods map the source to noise and back, accumulating trajectory deviation. (b) Inversion-free methods construct a direct editing ODE but operate on a single trajectory with no distributional guidance. (c) Our method constructs source and target particle clouds, solves entropic optimal transport to obtain a weighted coupling (line thickness indicates coupling strength), and derives a barycentric anchor $\bar{z}_t$ at each step via barycentric projection. A variational co-state accumulates trajectory deviation from the anchor and steers the corrected trajectory (green) toward the transport-consistent path.

monotonically regardless of the actual trajectory behaviour. Eq. (1) defines a single trajectory: different noise realisations produce different outcomes, yet this variance is uncontrolled and carries no distributional information about the space of plausible edits.

3.2 Particle Cloud Construction

Source cloud. We sample N particles $\boldsymbol{z}_s^i = \boldsymbol{z}_s + \sigma_s \boldsymbol{\xi}_i$, $\boldsymbol{\xi}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, $i = 1, \ldots, N$, defining the empirical measure $\mu_s = \frac{1}{N} \sum_i \delta_{\boldsymbol{z}_s^i}$.

Target cloud with annealed seeding. Naively integrating the full base ODE M times from the common start $\boldsymbol{z}_s$ yields low diversity when the velocity is Lipschitz-continuous, as trajectories contract toward one another. Instead, for each $j = 1, \ldots, M$ we draw $\sigma_j \sim \mathrm{Uniform}[\sigma_{\mathrm{lo}}, 1]$ and initialise

$$\boldsymbol{z}_{\sigma_j}^{(j)} = (1 - \sigma_j)\boldsymbol{z}_s + \sigma_j \boldsymbol{\varepsilon}_j, \quad \boldsymbol{\varepsilon}_j \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}), \tag{4}$$

then integrate the base dynamics from σ_j to 0, storing $\boldsymbol{z}_e^j = \boldsymbol{z}_{t_0}^{(j)}$. Varying the seed time exposes the flow to independent noise over non-overlapping intervals, injecting diversity that pure-noise restarts alone cannot preserve. For the OT coupling computed in §3.3 to provide meaningful guidance, the target cloud must span a diverse set of edited outcomes. We formalise this guarantee below.

Proposition 3.1. *Let $L = \mathrm{Lip}(u_{\mathrm{FA}})$. Under annealed seeding with $\sigma_j \sim \mathrm{Uniform}[\sigma_{\mathrm{lo}}, 1]$, the expected pairwise separation between any two target particles is strictly positive for any finite L:*

$$\mathbb{E}\big[\|\boldsymbol{z}_e^j - \boldsymbol{z}_e^k\|^2\big] \;\geq\; c\, d > 0, \tag{5}$$

where d is the latent dimension and $c > 0$ depends on L and σ_{lo}.

3.3 Entropic Transport Coupling

Given the two clouds, we compute an entropic OT coupling $P^\star$ that pairs source perturbations with edit-consistent targets. We first define the cost and solve for an initial plan, then describe how the plan adapts online as the trajectory evolves.

Cost design. We define a three-term pairwise cost $\boldsymbol{C} \in \mathbb{R}^{N \times M}$:

$$C_{ij} = \widetilde{C}_{ij}^{\mathrm{lat}} + \lambda_{\mathrm{clip}}\, \widetilde{C}_{ij}^{\mathrm{clip}} + \lambda_{\mathrm{dir}}\, \widetilde{C}_{ij}^{\mathrm{dir}}, \tag{6}$$

where $\widetilde{C}$ denotes min-max normalisation to $[0, 1]$.

(i) Latent proximity $C_{ij}^{\mathrm{lat}} = \|\boldsymbol{z}_s^i - \boldsymbol{z}_e^j\|^2$ penalises long-range transport in latent space, encouraging source particles to pair with geometrically nearby targets. This provides the basic spatial structure of the coupling.

(ii) CLIP visual similarity $C_{ij}^{\mathrm{clip}} = 1 - \langle \phi_{\mathrm{img}}(\mathcal{D}(\boldsymbol{z}_s^i)), \phi_{\mathrm{img}}(\mathcal{D}(\boldsymbol{z}_e^j)) \rangle$ measures perceptual closeness in decoded pixel space via CLIP image features. This complements the latent distance by capturing

semantic similarity that L2 in latent space may miss (e.g., two latents can be far in L2 but decode to perceptually similar images).

(iii) Directional alignment measures whether the visual change from source to target is consistent with the intended text edit. We define the text edit direction $\boldsymbol{\Delta}_{\text{txt}} = \phi_{\text{txt}}(c_e) - \phi_{\text{txt}}(c_s)$ and the per-pair visual direction $\boldsymbol{\Delta}^{ij}_{\text{img}} = \phi_{\text{img}}(\mathcal{D}(\boldsymbol{z}^j_e)) - \phi_{\text{img}}(\mathcal{D}(\boldsymbol{z}^i_s))$:

$$C^{\text{dir}}_{ij} = 1 - \frac{\langle \boldsymbol{\Delta}^{ij}_{\text{img}},\ \boldsymbol{\Delta}_{\text{txt}} \rangle}{\|\boldsymbol{\Delta}^{ij}_{\text{img}}\|\ \|\boldsymbol{\Delta}_{\text{txt}}\|}. \tag{7}$$

This is the only term that encodes edit-direction consistency: a pairing where the visual change aligns with the text change receives low cost, while one that drifts orthogonally or opposite is penalised. The cosine normalisation ensures the cost depends on the direction of change, not its magnitude. All CLIP features are precomputed once before the ODE loop.

Initial coupling. We solve the entropic OT problem $P^\star = \arg\min_{P \in \Pi(\boldsymbol{a},\boldsymbol{b})} \langle P, \boldsymbol{C} \rangle + \varepsilon\, \mathrm{KL}(P \| \boldsymbol{a}\boldsymbol{b}^\top)$ via log-domain Sinkhorn [20], obtaining the initial coupling and dual variables $(\log \boldsymbol{u}, \log \boldsymbol{v})$. The barycentric projection $T(\boldsymbol{z}^i_s) = N \sum_j P^\star_{ij} \boldsymbol{z}^j_e$ maps each source particle to a soft-weighted combination of target particles under this coupling. A formal analysis of the interaction between cost structure and Sinkhorn dual variables is provided in Appendix B.

Online adaptation. This initial plan is computed from source cloud positions at $t = 1$. As the trajectory $\boldsymbol{z}_t$ evolves, the current latent drifts from the source cloud and the static plan becomes increasingly misaligned. We therefore recompute the coupling at every ODE step using a trajectory-adaptive cost:

$$C^{t_k}_{ij} = \underbrace{\|\boldsymbol{z}_{t_k} - \boldsymbol{z}^j_e\|^2}_{\text{trajectory-to-target}} + \lambda_{\text{s}} \underbrace{\|\boldsymbol{z}^i_s - \boldsymbol{z}^j_e\|^2}_{\text{source consistency}} + \lambda_{\text{a}} \underbrace{\|\boldsymbol{z}_{t_k} - \boldsymbol{z}^i_s\|^2}_{\text{source anchor}} + \lambda_{\text{dir}} \tfrac{s_{\text{geo}}}{s_{\text{dir}}} C^{\text{dir}}_{ij}, \tag{8}$$

where $s_{\text{geo}}, s_{\text{dir}}$ are the maxima of the geometric and directional terms. The trajectory-to-target term favours target particles close to the current state $\boldsymbol{z}_{t_k}$, the source consistency term preserves the original source-target pairing structure, and the source anchor keeps the plan tethered to nearby source particles as the trajectory moves. Together, these three geometric terms replace the single latent distance and CLIP image-image term from the static cost (Eq. 6). The directional CLIP term C^{dir}_{ij} is retained because it encodes edit-direction alignment that no geometric term captures, and it is rescaled by $s_{\text{geo}}/s_{\text{dir}}$ for commensurability as the geometric magnitudes evolve. Additionally, we leverage the smooth evolution of the trajectory for computational efficiency. Since successive costs satisfy $\|C^{t_k} - C^{t_{k-1}}\|_\infty \ll \|C^{t_k}\|_\infty$, we warm-start Sinkhorn using the previous step's duals $(\log \boldsymbol{u}, \log \boldsymbol{v})$, reducing convergence from $\sim$200 to $\sim$40 iterations. An asymptotic analysis of this speedup is provided in Appendix B.

Barycentric anchor. From the updated coupling at step k, we compute per-particle transported targets $T_i = N \sum_j P^\star_{ij} \boldsymbol{z}^j_e$ and form the time-varying barycentric anchor:

$$\bar{\boldsymbol{z}}_{t_k} = \sum_i w_i \big[\sigma_k \boldsymbol{z}^i_s + (1 - \sigma_k) T_i\big], \tag{9}$$

where the weights $w_i = \mathrm{softmax}(-\|\boldsymbol{z}_{t_k} - \boldsymbol{z}^i_s\|^2 / \tau_k)$ measure proximity of the current trajectory $\boldsymbol{z}_{t_k}$ to each source particle, so that nearby source particles and their transported targets contribute more to the anchor. The bandwidth $\tau_k = \frac{\tau_0}{N} \sum_i \|\boldsymbol{z}_{t_k} - \boldsymbol{z}^i_s\|^2$ adapts to the mean squared distance between the trajectory and the source cloud, preventing the weights from collapsing to a single particle.

3.4 Variational Co-State Correction

Given the barycentric anchor $\bar{\boldsymbol{z}}_t$, we need a mechanism to steer the trajectory toward it. A natural choice is the heuristic correction $\eta\sigma_t(\bar{\boldsymbol{z}}_t - \boldsymbol{z}_t)$. However, the scaling factor $\eta\sigma_t$ is a predetermined function of the noise schedule and does not respond to the magnitude of the actual deviation $\|\boldsymbol{z}_t - \bar{\boldsymbol{z}}_t\|$. This means the correction applies the same strength whether the trajectory is far from the anchor or already tracking it closely. We derive an alternative that makes the correction proportional to the accumulated deviation. Since the velocity at each time affects all future states, choosing the correction at each step cannot be done independently. We therefore formulate a trajectory-level optimal control

problem to find the velocity $\boldsymbol{u}$ driving the trajectory from $\boldsymbol{z}_1 = \boldsymbol{z}_s$ to $\boldsymbol{z}_0$ that minimises:

$$\min_{\boldsymbol{u}} \int_0^1 \left[\tfrac{1}{2}\|\boldsymbol{u}_t - \boldsymbol{a}_t\|^2 + \tfrac{\lambda}{2}\|\boldsymbol{z}_t - \bar{\boldsymbol{z}}_t\|^2 \right] dt + \tfrac{\eta}{2}\|\boldsymbol{z}_0 - \boldsymbol{z}_s\|^2. \tag{10}$$

The first integrand term penalises deviating from the base FlowAlign velocity $\boldsymbol{a}_t$, the second penalises drifting from the OT anchor $\bar{\boldsymbol{z}}_t$, and the terminal cost encourages source preservation at $t=0$. To convert this trajectory-level objective into per-step conditions, we apply the Hamiltonian formulation of optimal control. The Hamiltonian $H = \frac{1}{2}\|\boldsymbol{u}-\boldsymbol{a}_t\|^2 + \frac{\lambda}{2}\|\boldsymbol{z}-\bar{\boldsymbol{z}}_t\|^2 + \boldsymbol{p}^\top \boldsymbol{u}$ encodes the same objective at a single time point, where $\boldsymbol{p}_t$ is an adjoint variable. Pontryagin's minimum principle [38] states that the optimal solution satisfies: (i) minimising H over $\boldsymbol{u}$ gives the optimal velocity $\boldsymbol{u}_t^\star = \boldsymbol{a}_t - \boldsymbol{p}_t$, and (ii) $\boldsymbol{p}_t$ evolves according to

$$\dot{\boldsymbol{p}}_t = -\partial H / \partial \boldsymbol{z} = -\lambda(\boldsymbol{z}_t - \bar{\boldsymbol{z}}_t). \tag{11}$$

That is, $\boldsymbol{p}_t$ accumulates at a rate proportional to the current deviation. Persistent drift builds a strong correction, while close tracking keeps $\boldsymbol{p}_t$ near zero and leaves the base velocity unchanged.

Proposition 3.2. *Under the optimal control formulation (10) with running cost $\frac{\lambda}{2}\|\boldsymbol{z}_t - \bar{\boldsymbol{z}}_t\|^2$, and assuming a frozen anchor (i.e. $\partial \bar{\boldsymbol{z}}_t / \partial \boldsymbol{z}_t = 0$), the co-state satisfies $\dot{\boldsymbol{p}}_t = -\lambda(\boldsymbol{z}_t - \bar{\boldsymbol{z}}_t)$ and evolves dynamically, accumulating correction proportional to integrated trajectory deviation from the OT anchor. In contrast, terminal-cost-only formulations yield $\partial H / \partial \boldsymbol{z} = 0$ and a time-invariant co-state.*

Discrete approximation. We discretise Eq. (11) over the ODE time grid $\{t_T, \ldots, t_0\}$ with step sizes $\Delta\sigma_k = \sigma_{k+1} - \sigma_k < 0$. At each step, the co-state integrates the current deviation $(\boldsymbol{z}_{t_k} - \bar{\boldsymbol{z}}_{t_k})$ while an exponential decay $\alpha_{\text{dec}} \in (0,1)$ downweights contributions from earlier steps:

$$\boldsymbol{p}_k \leftarrow \alpha_{\text{dec}} \boldsymbol{p}_{k-1} + \Delta\sigma_k \lambda (\boldsymbol{z}_{t_k} - \bar{\boldsymbol{z}}_{t_k}), \qquad \boldsymbol{u}_k^{\text{bar}} = -\Delta\sigma_k \boldsymbol{p}_k. \tag{12}$$

The first equation updates the co-state: α_{dec} discounts the previous value, and the new deviation is added with weight $\Delta\sigma_k\lambda$. The second converts the accumulated co-state into the correction $\boldsymbol{u}_k^{\text{bar}}$ applied at this step. The decay imposes an effective memory horizon of $\sim 1/(1-\alpha_{\text{dec}})$ steps, so past drift contributions are gradually forgotten.

Algorithm 1 Transport-Guided Flow Editing

Require: Source image $\boldsymbol{x}_s$, prompts c_s, c_e, flow model v_θ, CLIP $\phi_{\text{img}}, \phi_{\text{txt}}$, cloud sizes N, M, time grid $\{t_0, \ldots, t_T\}$
Ensure: Edited image $\hat{\boldsymbol{x}}_e$

// Phase 1: Cloud construction (§3.2)
$\boldsymbol{z}_s \leftarrow \mathcal{E}(\boldsymbol{x}_s)$
$\boldsymbol{z}_s^i \leftarrow \boldsymbol{z}_s + \sigma_s \boldsymbol{\xi}_i, \;\; \boldsymbol{\xi}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}) \;\; \text{for } i = 1, \ldots, N$ ▷ Source cloud
for $j = 1$ to M **do** ▷ Annealed target cloud
  Draw $\sigma_j \sim \mathrm{U}[\sigma_{\text{lo}}, 1]$, $\boldsymbol{\varepsilon}_j \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$; integrate base ODE from σ_j to 0 $\rightarrow$ $\boldsymbol{z}_e^j$
// Phase 2: Initial cost and coupling (§3.3)
$\boldsymbol{f}_i \leftarrow \phi_{\text{img}}(\mathcal{D}(\boldsymbol{z}_s^i))$ for all i; $\boldsymbol{g}_j \leftarrow \phi_{\text{img}}(\mathcal{D}(\boldsymbol{z}_e^j))$ for all j; $\boldsymbol{\Delta}_{\text{txt}} \leftarrow \phi_{\text{txt}}(c_e) - \phi_{\text{txt}}(c_s)$
$\boldsymbol{C} \leftarrow$ Eq. (6)–(7); $P^\star, \log \boldsymbol{u}, \log \boldsymbol{v} \leftarrow \textsc{Sinkhorn}(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{C}, \varepsilon)$
// Phase 3: OT-corrected ODE (§3.3–§3.4)
$\boldsymbol{z}_t \leftarrow \boldsymbol{z}_s$; $\boldsymbol{p}_{\text{k}} \leftarrow \boldsymbol{0}$
for $k = T$ down to 1 **do**
  $\boldsymbol{d}_{\text{FA}} \leftarrow$ base step ▷ Eq. (3)
  $C^{t_k} \leftarrow$ Eq. (8); $P^\star, \log \boldsymbol{u}, \log \boldsymbol{v} \leftarrow \textsc{Sinkhorn}(\boldsymbol{a}, \boldsymbol{b}, C^{t_k}, \varepsilon;\, \log \boldsymbol{u}, \log \boldsymbol{v})$ ▷ Warm-start
  $T_i \leftarrow N \sum_j P^\star_{ij} \boldsymbol{z}_e^j$ for all i; $\bar{\boldsymbol{z}}_{t_k} \leftarrow \sum_i w_i \left[\sigma_k \boldsymbol{z}_s^i + (1-\sigma_k) T_i\right]$ ▷ Eq. (9)
  $\boldsymbol{p}_{\text{k}} \leftarrow \alpha_{\text{dec}}\, \boldsymbol{p}_{\text{k}} + \Delta\sigma_k\, \lambda\, (\boldsymbol{z}_t - \bar{\boldsymbol{z}}_{t_k})$; $\boldsymbol{u}_k^{\text{bar}} \leftarrow -\Delta\sigma_k\, \boldsymbol{p}_{\text{k}}$ ▷ Eq. (12)
  $\boldsymbol{z}_t \leftarrow \boldsymbol{z}_t + \boldsymbol{d}_{\text{FA}} + \boldsymbol{u}_k^{\text{bar}}$ ▷ Eq. (13)
return $\hat{\boldsymbol{x}}_e = \mathcal{D}(\boldsymbol{z}_{t_0})$

3.5 Full Algorithm

Algorithm 1 gives the complete pseudocode. The corrected step combines the base FlowAlign velocity $\boldsymbol{d}_{\text{FA}}$ (Eq. 3) with the co-state correction $\boldsymbol{u}_k^{\text{bar}}$ (Eq. 12):

$$\boldsymbol{z}_{t_{k-1}} = \boldsymbol{z}_{t_k} + \underbrace{\boldsymbol{d}_{\text{FA}}}_{\text{base step}} + \underbrace{\boldsymbol{u}_k^{\text{bar}}}_{\text{OT co-state}} \tag{13}$$

Table 1: Quantitative evaluation. Best in **bold**, second best underlined. Inversion-free methods are shaded.

Method	Structure	Background Preservation			CLIP Similarity	
	DINO Dist. $\times 10^3 \downarrow$	PSNR $\uparrow$	LPIPS $\times 10^3 \downarrow$	SSIM $\times 10^2 \uparrow$	Whole $\uparrow$	Edited $\uparrow$
PIE-Bench						
SDEdit	31.29	26.21	81.33	84.03	23.15	22.78
RF-Inversion	42.60	21.48	180.2	68.42	24.90	22.32
RF-Solver	32.75	22.26	128.3	71.32	25.78	23.04
FireFlow	29.40	24.78	108.5	77.80	26.18	23.57
FlowEdit	27.52	22.08	132.3	70.02	26.25	24.08
FlowEdit+OTC	19.58	24.58	112.7	74.28	29.48	26.80
FlowAlign	17.84	28.58	49.23	92.30	**30.84**	27.28
SplitFlow	30.64	24.43	116.4	73.20	26.20	24.52
Ours	**14.25**	**29.76**	**39.20**	**93.50**	30.70	**28.40**
Edit-Bench						
SDEdit	36.90	14.80	240.7	60.14	21.28	23.76
RF-Inversion	48.50	13.38	278.5	58.21	22.84	24.76
FlowEdit	32.20	20.08	198.2	66.42	28.54	26.08
FlowAlign	22.79	27.30	70.68	90.25	**30.25**	26.20
SplitFlow	24.35	23.42	120.3	72.20	25.84	23.23
Ours	**19.87**	**28.32**	**48.42**	**90.78**	30.10	**27.30**
PIE-Bench++						
RF-Inversion	58.76	21.76	140.1	69.40	21.70	24.68
FlowEdit	44.20	21.30	160.5	70.40	26.11	23.22
FlowAlign	22.79	22.32	124.8	71.78	28.71	25.07
Ours	**21.03**	**24.62**	**98.18**	**80.24**	**29.91**	**26.29**

4 Experiments

Dataset, Baseline and Evaluation metrics. Consistent with existing works [14, 9, 24], we evaluate on PIE-Bench [39], which contains synthetic and natural images with paired source and editing prompts across 10 edit categories. Following [14], we additionally evaluate in the reverse direction (edited→source), which we refer to as Edit-Benchmark.The reverse direction tests whether a method can faithfully undo an edit, a strict test of trajectory consistency and reversibility that complements forward only evaluation. We further compare selected inversion-based and inversion-free baselines on PIE-Bench++ [40], which extends PIE-Bench with compositional prompts. We compare against SDEdit [15], which uses noisy samples to bridge trajectories, inversion-based methods RF-Inversion [13], RF-Solver [41], FireFlow [42] and inversion-free approaches FlowEdit [9], FlowEdit+OTC [25], FlowAlign [14], and SplitFlow [24]. All flow based methods use SD3-Medium with 33 NFEs. Consistent with baselines, we evaluate reconstruction fidelity and background preservation via structural distance (DINO ↓), PSNR (↑), SSIM (↑), and LPIPS (↓), capturing both perceptual and pixel-level consistency. Semantic alignment with the target edit is measured using CLIP similarity on the full image and the edited region (↑). Implementation details are provided in Appendix A.

Quantitative Results. Table 1 summarizes performance across both editing directions. A strong editing method must jointly preserve source structure while enforcing semantic alignment, rather than optimizing one at the expense of the other. On PIE-Bench and PIE-Bench++, our method consistently achieves the best overall balance across metrics. In particular, it improves background preservation while simultaneously strengthening edited region alignment, indicating that the proposed correction steers updates along edit consistent directions instead of introducing unnecessary changes in unaffected regions. Among inversion-free baselines, methods based on flow decomposition or pointwise correction improve individual metrics but fail to achieve consistent gains across all preservation measures, highlighting the limitation of local or heuristic updates. On EditBench (reverse editing), our method maintains the same favorable balance and achieves the strongest overall performance. The improvements in reconstruction metrics suggest that the learned trajectory remains

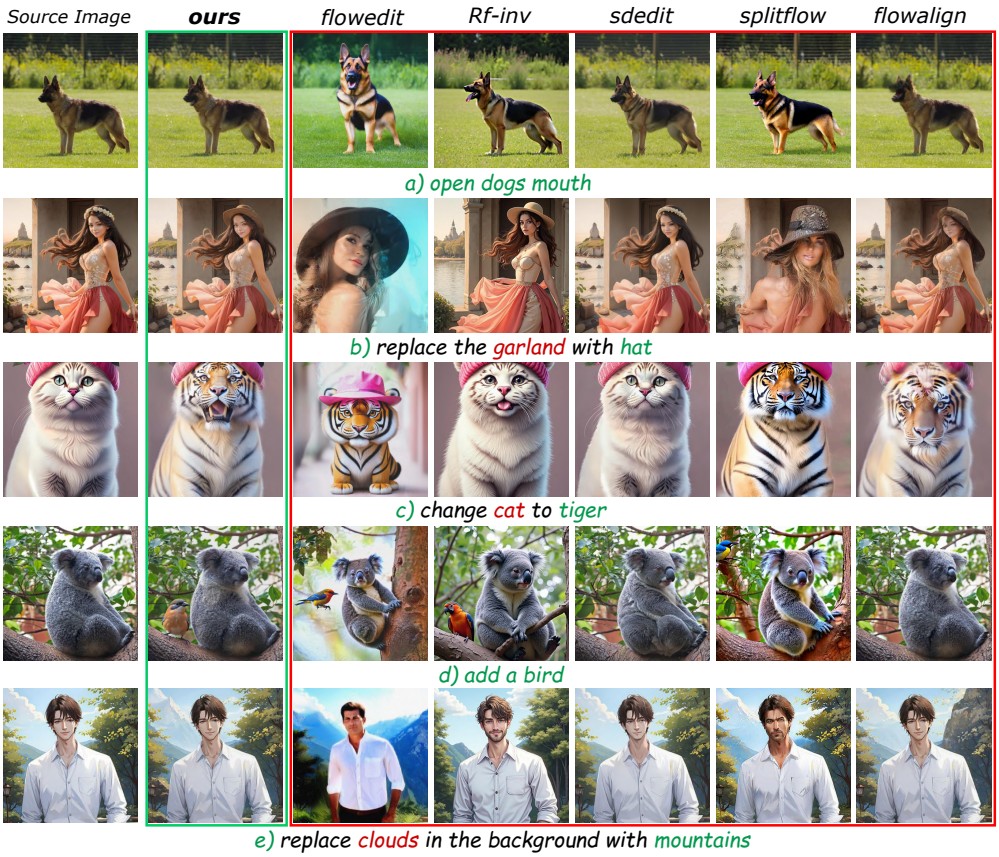


Figure 3: Qualitative comparison on PIE-Bench. Our method (second column) performs the intended edits while preserving subject identity, structure, and background details more faithfully than prior methods.

approximately reversible, indicating that the correction introduces minimal irreversible artifacts while enforcing alignment.

Qualitative Results. Figure 3 presents visual comparisons across five edit types. FlowEdit generates different subjects in rows b, c, and e, and incorrectly changes the dog's appearance instead of only opening its mouth (row a). RF-Inversion alters identity and background in the accessory replacement (row b) and produces inconsistent edits in rows c and d. SDEdit preserves coarse structure but under edits: the cat-to-dog transformation fails (row c) and the inserted bird is missing (row d). SplitFlow outputs comply with the edit instructions but fail to preserve identity (rows b, d, e), while FlowAlign trajectories can also lead to distorted outputs (rows a, b, c). Our method consistently performs the intended edits like, mouth opening (row a), accessory replacement (row b), species transformation (row c), object insertion (row d), and background replacement (row e), while preserving identity, pose, and background details across all examples. Additional qualitative analysis is present in Appendix M.

Component ablation. Table 2 isolates each contribution. Removing online OT (row 2) causes the largest PSNR drop (-0.86 dB): the static plan becomes inconsistent with the evolving trajectory, and warm-start is disabled alongside it. Removing the directional CLIP cost (row 3) preserves PSNR but drops CLIP_E by 0.80 and the coupling reverts to L2-only matching, pairing particles by geometric proximity rather than edit consistency. Replacing the variational co-state with the heuristic $\eta \cdot \sigma$ schedule (row 4) degrades both PSNR and LPIPS. The fixed schedule under-corrects images with strong deviation and over-corrects those

Table 2: Component ablation on PIE-Bench. CLIP_E measures CLIP similarity over edited regions.

Online OT	Dir. CLIP	Co-state	Annealed	Warm-start	PSNR↑	LPIPS$_{\times 10^3}$↓	CLIP_E↑
✓	✓	✓	✓	✓	**29.76**	**39.20**	**28.40**
✗	✓	✓	✓	✗	28.90	44.10	28.15
✓	✗	✓	✓	✓	29.65	40.10	27.60
✓	✓	✗	✓	✓	29.30	42.50	28.25
✓	✓	✓	✗	✓	29.05	45.80	27.85
✗	✗	✗	✗	✗	28.58	49.23	27.28

that track well. Removing annealed seeding (row 5) hurts all metrics: particles seeded uniformly from $t=1$ converge to near-identical endpoints, collapsing the target cloud. The final row recovers FlowAlign, confirming our baseline implementation.

Figure 4 sweeps λ_{dir} and cloud size $N = M$. The CLIP-vs-PSNR trade-off is smooth with no sharp degradation at any setting. Performance saturates at $N=M=8$, and increasing to 16 yields negligible improvement with doubled cloud construction cost. Additional sensitivity analysis is provided in Appendix D.

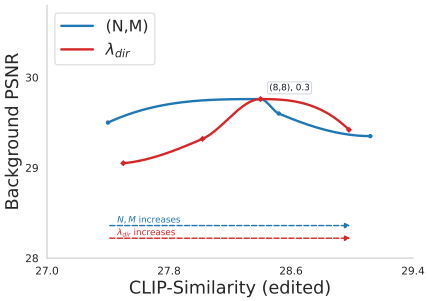


Figure 4: Hyperparameter sensitivity. CLIP_E vs. PSNR when sweeping λ_{dir} and cloud size $N = M$. Both curves are smooth, and performance saturates at $N=M=8$.

Trajectory and OT analysis. Figure 5 provides empirical validation of our key design choices, aggregated over all PIE-Bench images. **(a)** The coupling entropy $H(P_t^*)$ stays below $\log(NM)$ and decreases across steps, confirming the online plan remains informative and concentrates as the trajectory progresses. **(b)** The co-state $\|\boldsymbol{p}_k\|$ (normalised) rises through mid-integration, peaks around step 30, then drops sharply as the trajectory converges to the anchor near $t=0$. The heuristic $\eta\cdot\sigma_k$ decays monotonically regardless of content. The crossover at step $\sim$22 and the late-stage dip confirm the co-state is deviation-adaptive, and it applies correction where drift accumulates and withdraws it once the trajectory settles. **(c)** Cumulative displacement $\|z_t - z_s\|$ reveals that our method departs least from the source throughout integration, consistently below FlowAlign across all steps. FlowEdit operates on a fundamentally different scale (right axis), accumulating $\sim 7\times$ more displacement, which explains its weaker structure preservation in Table 1. Additional trajectory analysis (step-size smoothness, correction ratio, P^* heatmaps) are in Appendix C.

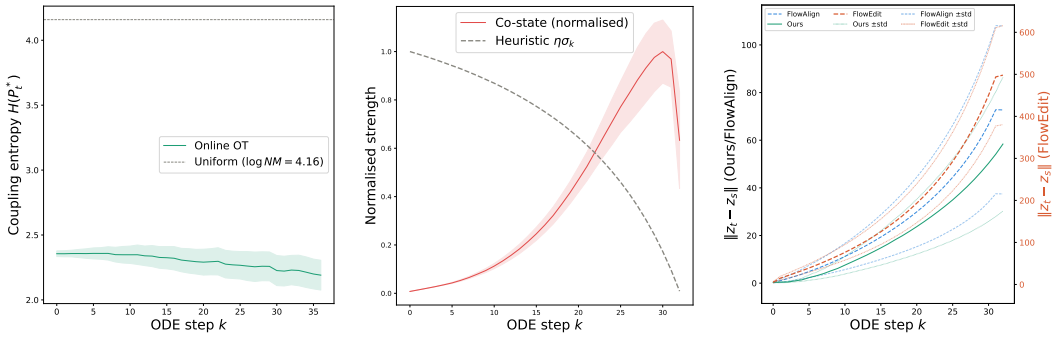


Figure 5: Trajectory analysis (mean $\pm$ std, PIE-Bench). **(a)** Coupling entropy stays below uniform and decreases over steps. Lower entropy indicates a more concentrated and structured coupling, while higher entropy corresponds to more dispersed and uncertain transport plan. **(b)** Co-state vs. heuristic: the co-state rises through mid-integration and crosses the heuristic around step 22, confirming deviation-adaptive correction. **(c)** Cumulative displacement: our method departs least from the source throughout integration, consistently below FlowAlign. FlowEdit operates on a different scale (right axis), accumulating $\sim 7\times$ more displacement.

5 Conclusion

We introduced Transport Corrected Flow Editing, which addresses the single trajectory limitation of inversion-free flow editing by recasting the problem in distributional terms. By constructing source and target particle clouds, computing an edit direction aware entropic OT coupling, and deriving trajectory corrections through a variational co-state, our method provides a principled and adaptive guidance mechanism that incorporates distributional transport structure and adapts to trajectory deviation over time, rather than relying on terminal objectives or fixed-schedule updates. Experiments on PIE-Bench demonstrate state-of-the-art background preservation (29.76 dB PSNR, 20% lower LPIPS than FlowAlign) with the highest edited-region CLIP similarity (28.40), and reverse editing confirms trajectory smoothness. Each component, including online OT, directional cost, annealed seeding, and variational co-state, contributes measurably, as validated by systematic ablation.

References

[1] 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.

[2] 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.

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

[4] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

[5] Black Forest Labs. FLUX, 2024.

[6] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6038–6047, 2023.

[7] Daiki Miyake, Akihiro Iohara, Yu Saito, and Toshiyuki Tanaka. Negative-prompt inversion: Fast image inversion for editing with text-guided diffusion models. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2063–2072. IEEE, 2025.

[8] Xuan Ju, Ailing Zeng, Yuxuan Bian, Shaoteng Liu, and Qiang Xu. Pnp inversion: Boosting diffusion-based editing with 3 lines of code. In *The Twelfth International Conference on Learning Representations*, 2023.

[9] Vladimir Kulikov, Matan Kleiner, Inbar Huberman-Spiegelglas, and Tomer Michaeli. Flowedit: Inversion-free text-based editing using pre-trained flow models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19721–19730, 2025.

[10] Gaurav Parmar, Krishna Kumar Singh, Richard Zhang, Yijun Li, Jingwan Lu, and Jun-Yan Zhu. Zero-shot image-to-image translation. In *ACM SIGGRAPH 2023 conference proceedings*, pages 1–11, 2023.

[11] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1921–1930, 2023.

[12] Chenyang Qi, Xiaodong Cun, Yong Zhang, Chenyang Lei, Xintao Wang, Ying Shan, and Qifeng Chen. Fatezero: Fusing attentions for zero-shot text-based video editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15932–15942, 2023.

[13] Litu Rout, Yujia Chen, Nataniel Ruiz, Constantine Caramanis, Sanjay Shakkottai, and Wen-Sheng Chu. Semantic image inversion and editing using rectified stochastic differential equations. *arXiv preprint arXiv:2410.10792*, 2024.

[14] Jeongsol Kim, Yeobin Hong, Jonghyun Park, and Jong Chul Ye. Flowalign: Trajectory-regularized, inversion-free flow-based image editing. In *The Fourteenth International Conference on Learning Representations*, 2026.

[15] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022.

[16] Xuan Su, Jiaming Song, Chenlin Meng, and Stefano Ermon. Dual diffusion implicit bridges for image-to-image translation. In *The Eleventh International Conference on Learning Representations*, 2023.

[17] Amir Hertz, Kfir Aberman, and Daniel Cohen-Or. Delta denoising score. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2328–2337, 2023.

[18] Juil Koo, Chanho Park, and Minhyuk Sung. Posterior distillation sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13352–13361, 2024.

[19] Jeongsol Kim, Geon Yeong Park, and Jong Chul Ye. Dreamsampler: Unifying diffusion sampling and score distillation for image manipulation. In *European Conference on Computer Vision*, pages 398–414. Springer, 2024.

[20] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

[21] Gabriel Peyré and Marco Cuturi. *Computational optimal transport: With applications to data science*. Now Foundations and Trends, 2019.

[22] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross-attention control. In *The Eleventh International Conference on Learning Representations*, 2023.

[23] Mingdeng Cao, Xintao Wang, Zhongang Qi, Ying Shan, Xiaohu Qie, and Yinqiang Zheng. Masactrl: Tuning-free mutual self-attention control for consistent image synthesis and editing. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 22560–22570, 2023.

[24] Sung-Hoon Yoon, Minghan Li, Gaspard Beaudouin, Congcong Wen, Muhammad Rafay Azhar, and Mengyu Wang. Splitflow: Flow decomposition for inversion-free text-to-image editing. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[25] Marian Lupaşcu and Mihai Sorin Stupariu. Optimal transport for rectified flow image editing: Unifying inversion-based and direct methods. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 6764–6774, 2026.

[26] Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, and Ricky T Chen. Multisample flow matching: Straightening flows with minibatch couplings. *ICML 2023*, 2023.

[27] 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.

[28] Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion schrödinger bridge with applications to score-based generative modeling. *Advances in neural information processing systems*, 34:17695–17709, 2021.

[29] Yuyang Shi, Valentin De Bortoli, Andrew Campbell, and Arnaud Doucet. Diffusion schrödinger bridge matching. *Advances in neural information processing systems*, 36:62183–62223, 2023.

[30] Alexander Korotin, Daniil Selikhanovych, and Evgeny Burnaev. Kernel neural optimal transport. In *The Eleventh International Conference on Learning Representations*, 2023.

[31] Litu Rout, Alexander Korotin, and Evgeny Burnaev. Generative modeling with optimal transport maps. In *International Conference on Learning Representations*, 2022.

[32] Jiaojiao Fan, Shu Liu, Shaojun Ma, Haomin Zhou, and Yongxin Chen. Neural monge map estimation and its applications. *arXiv preprint arXiv:2106.03812*, 2021.

[33] Mara Daniels, Tyler Maunu, and Paul Hand. Score-based generative neural networks for large-scale optimal transport. *Advances in neural information processing systems*, 34:12955–12965, 2021.

[34] Luca Eyring, Dominik Klein, Théo Uscidda, Giovanni Palla, Niki Kilbertus, Zeynep Akata, and Fabian Theis. Unbalancedness in neural monge maps improves unpaired domain translation. *arXiv preprint arXiv:2311.15100*, 2023.

[35] Xiang Gu, Liwei Yang, Jian Sun, and Zongben Xu. Optimal transport-guided conditional score-based diffusion model. *Advances in Neural Information Processing Systems*, 36:36540–36552, 2023.

[36] Derek Onken, Samy Wu Fung, Xingjian Li, and Lars Ruthotto. Ot-flow: Fast and accurate continuous normalizing flows via optimal transport. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9223–9232, 2021.

[37] Guan-Horng Liu, Arash Vahdat, De-An Huang, Evangelos A Theodorou, Weili Nie, and Anima Anandkumar. I2sb: image-to-image schrödinger bridge. In *Proceedings of the 40th International Conference on Machine Learning*, pages 22042–22062, 2023.

[38] Wendell H Fleming and Raymond W Rishel. *Deterministic and stochastic optimal control*. Springer Science & Business Media, 2012.

[39] Xuan Ju, Ailing Zeng, Yuxuan Bian, Shaoteng Liu, and Qiang Xu. Pnp inversion: Boosting diffusion-based editing with 3 lines of code. *International Conference on Learning Representations (ICLR)*, 2024.

[40] Mingzhen Huang, Jialing Cai, Shan Jia, Vishnu S Lokhande, and Siwei Lyu. Paralleledits: efficient multi-aspect text-driven image editing with attention grouping. *Advances in Neural Information Processing Systems*, 37:22569–22595, 2024.

[41] Jiangshan Wang, Junfu Pu, Zhongang Qi, Jiayi Guo, Yue Ma, Nisha Huang, Yuxin Chen, Xiu Li, and Ying Shan. Taming rectified flow for inversion and editing. In *International Conference on Machine Learning*, pages 64044–64058. PMLR, 2025.

[42] Yingying Deng, Xiangyu He, Changwang Mei, Peisong Wang, and Fan Tang. Fireflow: Fast inversion of rectified flow for image semantic editing. In *Forty-second International Conference on Machine Learning*, 2024.

[43] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[44] Sergi Caelles, Jordi Pont-Tuset, Federico Perazzi, Alberto Montes, Kevis-Kokitsi Maninis, and Luc Van Gool. The 2019 davis challenge on vos: Unsupervised multi-object segmentation. *arXiv*, 2019.

Technical appendices and supplementary material

Table of Contents

A. Implementation Details
B. Proofs and Mathematical justifications
C. Additional Trajectory Analysis
D. Hyperparameter Sensitivity Analysis
E. Comparison with Trajectory Averaging
F. Effect of Cloud Size on Performance and Runtime
G. CFG Scale Sensitivity
H. Human User Study
I. Generalization to FLUX Backbone
J. Runtime Comparison
K. Limitations and Future Work
L. Application on Video Editing
M. Additional Qualitative Results and Comparisons

A Implementation Details

We provide a complete specification of all architectural choices, hyperparameters, and algorithmic details sufficient for reproduction.

Backbone model. All experiments use Stable Diffusion 3 Medium (SD3-M) [4] with its default MMDiT architecture. The VAE encoder $\mathcal{E}$ maps images from $\mathbb{R}^{3\times H\times W}$ to a latent space $\mathbb{R}^{16\times H/8\times W/8}$, and for the default resolution $H = W = 1024$, this yields latents of shape $16 \times 128 \times 128$. Prompt conditioning uses the dual text encoders shipped with SD3-M: CLIP ViT-L/14 (pooled) and T5-XXL (sequence). We do not fine-tune any component, the entire pipeline is training-free.

CLIP model for cost computation. The transport cost (Eq. 6–7 in the main text) requires CLIP image and text features. We use a *separate* CLIP ViT-L/14 model [43] (not the SD3 text encoder). Image features are extracted by decoding each latent particle through the VAE decoder to pixel space ($[0,1]^{3\times 1024\times 1024}$), resizing to 224×224 via the standard CLIP preprocessing pipeline, and passing through the CLIP image encoder to obtain a 768 dimensional ℓ_2-normalised feature vector. Text features for the source and target prompts are similarly extracted via the CLIP text encoder, tokenised with a maximum length of 77 tokens. All $N + M = 16$ image features and 2 text features are precomputed once before the ODE loop and remain fixed throughout integration.

ODE integration and guidance. The editing trajectory is integrated using the SD3-M default scheduler with a σ-linear noise schedule from $t=1$ (pure noise) to $t=0$ (clean image), discretised into 33 uniformly spaced steps (NFE = 33). We use classifier-free guidance with scale $\omega=13.5$ applied to the target direction (Eq. 2). The FlowAlign source consistency weight is $\zeta=0.01$, following the default from [14]. Similar to FlowAlign, each ODE step requires three neural function evaluations: $v_\theta(\boldsymbol{p}_t, c_e)$, $v_\theta(\boldsymbol{p}_t, c_s)$, and $v_\theta(\boldsymbol{q}_t, c_s)$.

Source cloud construction. We sample $N=8$ source particles as $\boldsymbol{z}_s^i = \boldsymbol{z}_s + \sigma_s \boldsymbol{\xi}_i$, $\boldsymbol{\xi}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, with perturbation scale $\sigma_s=0.1$. This value is chosen to be small enough that all particles decode to visually indistinguishable images (verified by LPIPS < 0.01 between decoded particles), yet large enough to span a meaningful neighbourhood in latent space for the OT coupling to operate on. The empirical source measure is $\mu_s = \frac{1}{N}\sum_{i=1}^{N} \delta_{\boldsymbol{z}_s^i}$ with uniform weights $\boldsymbol{a} = \mathbf{1}_N/N$.

Target cloud construction (annealed seeding). We generate $M=8$ target particles using the annealed seeding procedure described in §3.2. For each particle $j \in \{1,\dots,M\}$: (i) draw a

starting noise level $\sigma_j \sim \text{Uniform}[\sigma_{\text{lo}}, 1]$ with $\sigma_{\text{lo}} = 0.45$; (ii) initialise at the noisy interpolation $\boldsymbol{z}_{\sigma_j}^{(j)} = (1-\sigma_j)\boldsymbol{z}_s + \sigma_j \boldsymbol{\varepsilon}_j$, $\boldsymbol{\varepsilon}_j \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$; (iii) integrate the base FlowAlign ODE from σ_j to $t=0$ using the same scheduler, CFG scale, and source consistency weight as the main editing loop but conditioned on the *target* prompt c_e. Each partial rollout requires $\lceil 33\sigma_j \rceil$ NFEs on average. With $\sigma_{\text{lo}} = 0.45$, the average rollout length is ≈24 steps. The empirical target measure is $\nu_e = \frac{1}{M}\sum_{j=1}^{M} \delta_{\boldsymbol{z}_e^j}$ with uniform weights $\boldsymbol{b} = \mathbf{1}_M/M$.

Static cost matrix. The three-term cost (Eq. 6) uses weights $\lambda_{\text{clip}} = 0.4$, $\lambda_{\text{dir}} = 0.3$. Each term is independently normalised to $[0,1]$ via min-max scaling before weighting: $\widetilde{C}_{ij} = (C_{ij} - \min C)/(\max C - \min C + 10^{-8})$. The latent L2 term $C_{ij}^{\text{lat}} = \|\boldsymbol{z}_s^i - \boldsymbol{z}_e^j\|^2$ is computed via `torch.cdist` for efficiency. The CLIP image-image term $C_{ij}^{\text{clip}} = 1 - \langle \boldsymbol{f}_i, \boldsymbol{g}_j \rangle$ and the directional term C_{ij}^{dir} (Eq. 7) are computed from precomputed features using batched matrix operations.

Sinkhorn solver. We solve the entropic OT problem in log-domain to avoid numerical underflow. Marginals are uniform: $\boldsymbol{a} = \mathbf{1}_N/N$, $\boldsymbol{b} = \mathbf{1}_M/M$. Entropic regularisation is $\varepsilon = 0.1$. The initial plan $P^\star$ is computed with 200 Sinkhorn iterations using the static cost matrix. At each subsequent ODE step k, the online cost matrix C^{t_k} (Eq. 8) is computed with $\lambda_s = 0.3$ (source consistency), $\lambda_a = 0.2$ (source anchor), and the directional CLIP term rescaled by $s_{\text{geo}}/s_{\text{dir}}$ for commensurability. The Sinkhorn solver is warm-started from the converged log-dual variables $(\log \boldsymbol{u}, \log \boldsymbol{v})$ of the previous step and run for 40 iterations. We verified empirically that 40 warm-started iterations achieve $< 10^{-4}$ marginal violation on all 700 PIE-Bench images, comparable to 200 cold-start iterations.

Barycentric anchor computation. At each step k, the per-particle transported targets are $T(\boldsymbol{z}_s^i) = N\sum_j P_{ij}^\star \boldsymbol{z}_e^j$. The time-varying barycentric anchor is the soft weighted combination $\bar{\boldsymbol{z}}_{t_k} = \sum_i w_i[\sigma_k \boldsymbol{z}_s^i + (1-\sigma_k)T(\boldsymbol{z}_s^i)]$, where the weights w_i are computed via soft nearest neighbour assignment: $w_i = \text{softmax}(-\|\boldsymbol{z}_{t_k} - \boldsymbol{z}_s^i\|^2/\tau_k)$ with adaptive bandwidth $\tau_k = \frac{\tau_0}{N}\sum_i \|\boldsymbol{z}_{t_k} - \boldsymbol{z}_s^i\|^2$ and $\tau_0 = 0.10$. The adaptive bandwidth prevents one-hot collapse in high dimensions: when the cloud spread is large, the softmax temperature increases proportionally, maintaining smooth weights.

Variational co-state. The discrete co-state update (Eq. 12) uses running cost weight $\lambda = 0.8$ and exponential decay $\alpha_{\text{dec}} = 0.85$ (effective memory ∼6.7 steps). The co-state is initialised at $\boldsymbol{p}_T = \mathbf{0}$, corresponding to the IVP approximation of the continuous TPBVP described in §3.4. Two practical modifications to the theoretical formulation are applied: (i) σ-scaling on both the running cost input and the correction output, which modifies the effective running cost from $\frac{\lambda}{2}\|\boldsymbol{z}_t - \bar{\boldsymbol{z}}_t\|^2$ to $\frac{\lambda\sigma_k}{2}\|\boldsymbol{z}_t - \bar{\boldsymbol{z}}_t\|^2$, tapering the correction as $\sigma \to 0$:

$$\boldsymbol{p}_k = \alpha_{\text{dec}}\, \boldsymbol{p}_{k-1} + \Delta\sigma_k\, \lambda\, \sigma_k\, (\boldsymbol{z}_{t_k} - \bar{\boldsymbol{z}}_{t_k}), \qquad \boldsymbol{u}_k^{\text{bar}} = -\Delta\sigma_k\, \sigma_k\, \boldsymbol{p}_k;$$

(ii) norm clamping: if $\|\boldsymbol{p}_k\| > 0.15\|\boldsymbol{d}_{\text{FA}}\|/(|\Delta\sigma_k| + 10^{-8})$, we rescale $\boldsymbol{p}_k$ to this maximum, ensuring the correction never exceeds 15% of the base velocity magnitude. This prevents late step instability without affecting the co-state dynamics in normal operating regimes.

Hardware. All experiments run on a single NVIDIA H100 (80 GB VRAM). Peak memory usage is ∼22 GB (dominated by the flow transformer). We use PyTorch 2.1 with `bfloat16` for all neural forward passes and `float32` for the Sinkhorn solver and co-state arithmetic. Random seeds are fixed per image for reproducibility.

B Proofs and Mathematical Justifications

B.1 Proof of Proposition 3.1 (Cloud Diversity)

Proof. We show that annealed seeding yields strictly positive expected pairwise separation between target particles under bi-Lipschitz flow dynamics.

Let j and k be two particles with independent seeding levels $\sigma_j, \sigma_k \sim \text{Uniform}[\sigma_{\text{lo}}, 1]$. Without loss of generality assume $\sigma_j > \sigma_k$, and define

$$\tau := \sigma_k = \min(\sigma_j, \sigma_k).$$

Step 1: Separation at the seeding time τ.

At time τ, particle k is freshly initialized:

$$\boldsymbol{z}_\tau^{(k)} = (1-\tau)\boldsymbol{z}_s + \tau\boldsymbol{\varepsilon}_k, \quad \boldsymbol{\varepsilon}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}).$$

Particle j, seeded earlier at $\sigma_j > \tau$, has evolved under the ODE to $\boldsymbol{z}_\tau^{(j)}$, which depends on independent noise $\boldsymbol{\varepsilon}_j$.

Condition on $\boldsymbol{z}_\tau^{(j)}$ and τ, treating $\boldsymbol{z}_\tau^{(j)}$ as fixed. Define

$$\boldsymbol{a} := \boldsymbol{z}_\tau^{(j)} - (1-\tau)\boldsymbol{z}_s.$$

Then

$$\boldsymbol{z}_\tau^{(j)} - \boldsymbol{z}_\tau^{(k)} = \boldsymbol{a} - \tau\boldsymbol{\varepsilon}_k.$$

Hence,

$$\begin{aligned}\mathbb{E}\big[\|\boldsymbol{z}_\tau^{(j)} - \boldsymbol{z}_\tau^{(k)}\|^2 \mid \boldsymbol{z}_\tau^{(j)}, \tau\big] &= \mathbb{E}\big[\|\boldsymbol{a} - \tau\boldsymbol{\varepsilon}_k\|^2\big] \\ &= \|\boldsymbol{a}\|^2 + \tau^2\mathbb{E}\big[\|\boldsymbol{\varepsilon}_k\|^2\big] - 2\tau\mathbb{E}\big[\langle \boldsymbol{a}, \boldsymbol{\varepsilon}_k\rangle\big].\end{aligned}$$

Since $\boldsymbol{\varepsilon}_k$ is zero-mean and independent of $\boldsymbol{a}$,

$$\mathbb{E}\big[\langle \boldsymbol{a}, \boldsymbol{\varepsilon}_k\rangle\big] = \langle \boldsymbol{a}, \mathbb{E}[\boldsymbol{\varepsilon}_k]\rangle = 0.$$

Moreover, for $\boldsymbol{\varepsilon}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ in d dimensions,

$$\mathbb{E}\big[\|\boldsymbol{\varepsilon}_k\|^2\big] = d.$$

Therefore,

$$\mathbb{E}\big[\|\boldsymbol{z}_\tau^{(j)} - \boldsymbol{z}_\tau^{(k)}\|^2 \mid \boldsymbol{z}_\tau^{(j)}, \tau\big] = \|\boldsymbol{z}_\tau^{(j)} - (1-\tau)\boldsymbol{z}_s\|^2 + \tau^2 d.$$

Dropping the non-negative first term gives

$$\mathbb{E}\big[\|\boldsymbol{z}_\tau^{(j)} - \boldsymbol{z}_\tau^{(k)}\|^2 \mid \boldsymbol{z}_\tau^{(j)}, \tau\big] \geq \tau^2 d.$$

Step 2: Lifting the bound and propagating to time 0.

We now express the bound in terms of the seed levels (σ_j, σ_k), which determine τ and parameterize the subsequent dynamics. Let

$$X := \|\boldsymbol{z}_\tau^{(j)} - \boldsymbol{z}_\tau^{(k)}\|^2.$$

Applying the tower property,

$$\begin{aligned}\mathbb{E}[X \mid \sigma_j, \sigma_k] &= \mathbb{E}\Big[\mathbb{E}[X \mid \boldsymbol{z}_\tau^{(j)}, \tau] \,\Big|\, \sigma_j, \sigma_k\Big] \\ &\geq \mathbb{E}\big[\tau^2 d \mid \sigma_j, \sigma_k\big].\end{aligned}$$

Since $\tau = \min(\sigma_j, \sigma_k)$ is completely determined by (σ_j, σ_k),

$$\mathbb{E}\big[\tau^2 d \mid \sigma_j, \sigma_k\big] = \tau^2 d.$$

Hence,

$$\mathbb{E}[X \mid \sigma_j, \sigma_k] \geq \tau^2 d.$$

From time τ to 0, both particles evolve under the same ODE. Define

$$\boldsymbol{e}_t := \boldsymbol{z}_t^{(j)} - \boldsymbol{z}_t^{(k)}.$$

Assuming the induced flow map is bi-Lipschitz, i.e.

$$\|\Phi_{t\to s}(\boldsymbol{x}) - \Phi_{t\to s}(\boldsymbol{y})\| \geq e^{-L|t-s|}\|\boldsymbol{x} - \boldsymbol{y}\|.$$

Applying this from τ to 0 gives

$$\|\boldsymbol{e}_0\| = \|\Phi_{\tau\to 0}(\boldsymbol{z}_\tau^{(j)}) - \Phi_{\tau\to 0}(\boldsymbol{z}_\tau^{(k)})\| \geq e^{-L\tau}\|\boldsymbol{e}_\tau\|.$$

Squaring both sides,

$$\|\boldsymbol{e}_0\|^2 \geq e^{-2L\tau}\|\boldsymbol{e}_\tau\|^2.$$

Hence,

$$\mathbb{E}\big[\|\boldsymbol{z}_e^j - \boldsymbol{z}_e^k\|^2 \mid \sigma_j, \sigma_k\big] \geq e^{-2L\tau}\mathbb{E}\big[\|\boldsymbol{e}_\tau\|^2 \mid \sigma_j, \sigma_k\big].$$

Combining with the previous lower bound,

$$\mathbb{E}\big[\|\boldsymbol{z}_e^j - \boldsymbol{z}_e^k\|^2 \mid \sigma_j, \sigma_k\big] \geq \tau^2 d\, e^{-2L\tau}.$$

Step 3: Averaging over seed levels.

Taking expectation over (σ_j, σ_k) and applying the tower property once more,

$$\mathbb{E}\big[\|\boldsymbol{z}_e^j - \boldsymbol{z}_e^k\|^2\big] = \mathbb{E}_{\sigma_j,\sigma_k}\Big[\mathbb{E}[\|\boldsymbol{z}_e^j - \boldsymbol{z}_e^k\|^2 \mid \sigma_j, \sigma_k]\Big] \geq d\,\mathbb{E}[\tau^2 e^{-2L\tau}].$$

Since $\tau = \min(\sigma_j, \sigma_k)$, its density is

$$f_\tau(x) = \frac{2(1-x)}{(1-\sigma_{\text{lo}})^2}, \quad x \in [\sigma_{\text{lo}}, 1].$$

Therefore,

$$\mathbb{E}[\tau^2 e^{-2L\tau}] = \int_{\sigma_{\text{lo}}}^1 x^2 e^{-2Lx} f_\tau(x)\, dx = \frac{2}{(1-\sigma_{\text{lo}})^2}\int_{\sigma_{\text{lo}}}^1 x^2(1-x)e^{-2Lx}\, dx.$$

The integrand is strictly positive on $(\sigma_{\text{lo}}, 1)$ for any finite L, hence

$$\mathbb{E}[\tau^2 e^{-2L\tau}] > 0.$$

Therefore, there exists $c > 0$ such that

$$\mathbb{E}\big[\|\boldsymbol{z}_e^j - \boldsymbol{z}_e^k\|^2\big] \geq c\, d > 0.$$

□

B.2 Cost Structure and Sinkhorn Dual Interaction

We analyse how the cost terms in the static cost (Eq. 6) and the online cost (Eq. 8) interact with the Sinkhorn dual variables, and verify that the directional CLIP term actively shapes the coupling $P^\star$.

Background. The Sinkhorn algorithm solves the entropic OT problem by iteratively updating dual variables $(\boldsymbol{u}, \boldsymbol{v})$ such that $P^\star_{ij} = u_i \cdot K_{ij} \cdot v_j$ with kernel $K_{ij} = e^{-C_{ij}/\varepsilon}$. For a cost of the form $C_{ij} = f(i,j) + g(j)$ where $g(j)$ depends only on the target index, the kernel factors as:

$$K_{ij} = e^{-f(i,j)/\varepsilon} \cdot e^{-g(j)/\varepsilon},$$

where the product structure follows from the additive decomposition in the exponent. The factor $e^{-g(j)/\varepsilon}$ depends only on j and is absorbed into v_j during Sinkhorn's row-normalisation step, which sets $u_i \propto a_i / \sum_j K_{ij} v_j$. Equivalently, in the OT objective:

$$\langle P, C\rangle = \sum_{i,j} P_{ij}\big[f(i,j) + g(j)\big] = \langle P, f\rangle + \sum_j g(j) \underbrace{\sum_i P_{ij}}_{=b_j},$$

the second term is a constant for any $P \in \Pi(\boldsymbol{a}, \boldsymbol{b})$ since the marginal constraint fixes $\sum_i P_{ij} = b_j$. The same applies symmetrically to terms $h(i)$ depending only on the source index. Consequently, any cost component that decomposes as $h(i) + k(j)$ has zero effect on the coupling structure.

Static cost terms (Eq. 6). All three terms in the static cost genuinely couple i and j:

- The latent term $\|\boldsymbol{z}_s^i - \boldsymbol{z}_e^j\|^2 = \|\boldsymbol{z}_s^i\|^2 - 2\langle \boldsymbol{z}_s^i, \boldsymbol{z}_e^j\rangle + \|\boldsymbol{z}_e^j\|^2$. The first and third components are separable and absorbed into the duals, but the cross-product $-2\langle \boldsymbol{z}_s^i, \boldsymbol{z}_e^j\rangle$ couples i and j bilinearly and determines the coupling structure.
- The CLIP term $1 - \langle \phi_{\text{img}}(\mathcal{D}(\boldsymbol{z}_s^i)), \phi_{\text{img}}(\mathcal{D}(\boldsymbol{z}_e^j))\rangle$ is bilinear in the source and target CLIP features and cannot be decomposed as $h(i) + k(j)$.
- The directional term C_{ij}^{dir} is analysed in Lemma B.1 below.

Online cost terms (Eq. 8). At each ODE step k, the trajectory position $\boldsymbol{z}_{t_k}$ is a fixed constant within that Sinkhorn solve. This has specific consequences for each term:

- *Trajectory-to-target* $\|\boldsymbol{z}_{t_k} - \boldsymbol{z}_e^j\|^2$: depends only on j at step k, so it is absorbed into the target dual $v_j^{(k)}$, acting as a trajectory-dependent reweighting of the effective target marginal. Target particles closer to $\boldsymbol{z}_{t_k}$ receive higher effective mass.
- *Source anchor* $\|\boldsymbol{z}_{t_k} - \boldsymbol{z}_s^i\|^2$: depends only on i at step k, so it is absorbed into the source dual $u_i^{(k)}$, reweighting the effective source marginal. Source particles closer to $\boldsymbol{z}_{t_k}$ become more active in the plan.
- *Source consistency* $\|\boldsymbol{z}_s^i - \boldsymbol{z}_e^j\|^2$: pairwise (same cross-product structure as the static latent term). This is the geometric term that shapes the coupling structure within each step.
- *Directional CLIP* C_{ij}^{dir}: pairwise (Lemma B.1). This shapes the coupling based on edit-direction consistency.

Although the trajectory-to-target and source-anchor terms are absorbed within any single solve, they produce *different* dual variables at each step because $\boldsymbol{z}_{t_k} \neq \boldsymbol{z}_{t_{k+1}}$. The plan therefore adapts to the trajectory position across steps through this marginal-reweighting mechanism, while the coupling structure within each step is determined by the source-consistency and directional CLIP terms.

Non-separability of the directional cost. The directional term requires verification because its unnormalised version is separable: $\langle \boldsymbol{g}_j - \boldsymbol{f}_i, \hat{\boldsymbol{d}} \rangle = \langle \boldsymbol{g}_j, \hat{\boldsymbol{d}} \rangle - \langle \boldsymbol{f}_i, \hat{\boldsymbol{d}} \rangle$ decomposes as $h(i) + k(j)$ and would be invisible to Sinkhorn. The cosine normalisation prevents this.

Lemma B.1. *The cosine-normalized directional cost*

$$C_{ij}^{\text{dir}} = 1 - \left\langle \frac{\boldsymbol{g}_j - \boldsymbol{f}_i}{\|\boldsymbol{g}_j - \boldsymbol{f}_i\|}, \hat{\boldsymbol{d}} \right\rangle$$

is generically not additively separable, i.e. it cannot be written as

$$C_{ij}^{\text{dir}} = h(i) + k(j)$$

for generic CLIP feature configurations.

In particular, non-separability holds whenever there exists an index quadruple (i, i', j, j') violating the four-point identity

$$C_{ij}^{\text{dir}} + C_{i'j'}^{\text{dir}} \neq C_{ij'}^{\text{dir}} + C_{i'j}^{\text{dir}}.$$

Proof. Let $\boldsymbol{f}_i = \phi_{\text{img}}(\mathcal{D}(\boldsymbol{z}_s^i))$, $\boldsymbol{g}_j = \phi_{\text{img}}(\mathcal{D}(\boldsymbol{z}_e^j))$, and $\hat{\boldsymbol{d}} = \boldsymbol{\Delta}_{\text{txt}}/\|\boldsymbol{\Delta}_{\text{txt}}\|$. Define $\psi(\boldsymbol{x}) = \langle \boldsymbol{x}/\|\boldsymbol{x}\|, \hat{\boldsymbol{d}} \rangle$.

A cost matrix C_{ij} is additively separable as $h(i) + k(j)$ if and only if the *four-point identity* holds for all index quadruples:

$$C_{ij} + C_{i'j'} = C_{ij'} + C_{i'j} \quad \text{for all } i, i', j, j'.$$

This is a standard characterisation: separability means $C_{ij} - C_{i'j} = h(i) - h(i')$ is independent of j, which is equivalent to $C_{ij} - C_{i'j} = C_{ij'} - C_{i'j'}$ for all j, j', which rearranges to the four-point identity.

For our cost $C_{ij}^{\text{dir}} = 1 - \psi(\boldsymbol{g}_j - \boldsymbol{f}_i)$, the four-point identity requires:

$$\psi(\boldsymbol{g}_j - \boldsymbol{f}_i) + \psi(\boldsymbol{g}_{j'} - \boldsymbol{f}_{i'}) = \psi(\boldsymbol{g}_{j'} - \boldsymbol{f}_i) + \psi(\boldsymbol{g}_j - \boldsymbol{f}_{i'})$$

for all i, i', j, j', where $\psi(\boldsymbol{x}) = \langle \boldsymbol{x}/\|\boldsymbol{x}\|, \hat{\boldsymbol{d}} \rangle$ maps each vector to its cosine similarity with the edit direction.

We show this identity generically fails. Consider any four vectors $\boldsymbol{a} = \boldsymbol{g}_j - \boldsymbol{f}_i$, $\boldsymbol{b} = \boldsymbol{g}_{j'} - \boldsymbol{f}_i$, $\boldsymbol{c} = \boldsymbol{g}_j - \boldsymbol{f}_{i'}$, $\boldsymbol{d} = \boldsymbol{g}_{j'} - \boldsymbol{f}_{i'}$. Note $\boldsymbol{c} = \boldsymbol{a} + \boldsymbol{\delta}$ and $\boldsymbol{d} = \boldsymbol{b} + \boldsymbol{\delta}$ where $\boldsymbol{\delta} = \boldsymbol{f}_i - \boldsymbol{f}_{i'}$. The four-point identity becomes:

$$\psi(\boldsymbol{a}) + \psi(\boldsymbol{b} + \boldsymbol{\delta}) = \psi(\boldsymbol{b}) + \psi(\boldsymbol{a} + \boldsymbol{\delta}).$$

Rearranging: $\psi(\boldsymbol{a} + \boldsymbol{\delta}) - \psi(\boldsymbol{a}) = \psi(\boldsymbol{b} + \boldsymbol{\delta}) - \psi(\boldsymbol{b})$. That is, the finite difference of ψ with displacement $\boldsymbol{\delta}$ must be the same at $\boldsymbol{a}$ and $\boldsymbol{b}$.

Since $\psi(\boldsymbol{x}) = \langle \boldsymbol{x}/\|\boldsymbol{x}\|, \hat{\boldsymbol{d}} \rangle$ is the cosine similarity with a fixed direction, it is a nonlinear function of $\boldsymbol{x}$ (specifically, it depends on both the direction and magnitude of $\boldsymbol{x}$). Its finite difference $\psi(\boldsymbol{x}+\boldsymbol{\delta})-\psi(\boldsymbol{x})$ varies with $\boldsymbol{x}$ unless $\boldsymbol{x}$ and $\boldsymbol{x}+\boldsymbol{\delta}$ are constrained to produce the same increment, which requires $\boldsymbol{a}$ and $\boldsymbol{b}$ to have the same magnitude and the same projection onto $\hat{\boldsymbol{d}}$ relative to $\boldsymbol{\delta}$.

To verify this concretely, consider $D \geq 2$, $\hat{\boldsymbol{d}} = \boldsymbol{e}_1$, $\boldsymbol{\delta} = \boldsymbol{e}_2$, and:

$$\begin{aligned} \boldsymbol{a} &= (1, 0, 0, \ldots), \quad \boldsymbol{a} + \boldsymbol{\delta} = (1, 1, 0, \ldots), \\ \boldsymbol{b} &= (1, 1, 0, \ldots), \quad \boldsymbol{b} + \boldsymbol{\delta} = (1, 2, 0, \ldots). \end{aligned}$$

Then:

$$\begin{aligned} \psi(\boldsymbol{a}) &= 1, \quad \psi(\boldsymbol{a} + \boldsymbol{\delta}) = 1/\sqrt{2}, \\ \psi(\boldsymbol{b}) &= 1/\sqrt{2}, \quad \psi(\boldsymbol{b} + \boldsymbol{\delta}) = 1/\sqrt{5}. \end{aligned}$$

The left side: $\psi(\boldsymbol{a} + \boldsymbol{\delta}) - \psi(\boldsymbol{a}) = 1/\sqrt{2} - 1 \approx -0.293$. The right side: $\psi(\boldsymbol{b} + \boldsymbol{\delta}) - \psi(\boldsymbol{b}) = 1/\sqrt{5} - 1/\sqrt{2} \approx -0.260$. This configuration is realised by, e.g., $\boldsymbol{f}_i = \boldsymbol{0}$, $\boldsymbol{f}_{i'} = -\boldsymbol{e}_2$, $\boldsymbol{g}_j = \boldsymbol{e}_1$, $\boldsymbol{g}_{j'} = \boldsymbol{e}_1 + \boldsymbol{e}_2$, giving $\boldsymbol{a} = \boldsymbol{g}_j - \boldsymbol{f}_i = \boldsymbol{e}_1$, $\boldsymbol{b} = \boldsymbol{g}_{j'} - \boldsymbol{f}_i = \boldsymbol{e}_1 + \boldsymbol{e}_2$, and $\boldsymbol{\delta} = \boldsymbol{f}_i - \boldsymbol{f}_{i'} = \boldsymbol{e}_2$. These are not equal, so the four-point identity fails. Since the identity is an equality of continuous functions of the CLIP features, and it fails at the explicit configuration above, it also fails on an open neighbourhood of this configuration in $\mathbb{R}^{D\times(N+M)}$. In particular, for generic CLIP features (those not lying on the closed set where the identity holds for all quadruples), C_{ij}^{dir} is non-separable.

□

B.3 Proof of Proposition 3.2 (Dynamic Co-State)

Proof. We derive the co-state dynamics from the optimal control problem stated in the main text and show that the running cost is responsible for producing a time-varying correction signal.

Consider the optimal control problem:

$$\min_{\boldsymbol{u}} \int_0^1 \left[\frac{1}{2} \|\boldsymbol{u}_t - \boldsymbol{a}_t\|^2 + \frac{\lambda}{2} \|\boldsymbol{z}_t - \bar{\boldsymbol{z}}_t\|^2 \right] dt + \frac{\eta}{2} \|\boldsymbol{z}_0 - \boldsymbol{z}_s\|^2,$$

subject to $\dot{\boldsymbol{z}}_t = \boldsymbol{u}_t$ with $\boldsymbol{z}_1 = \boldsymbol{z}_s$, where $\boldsymbol{a}_t$ is the base FlowAlign velocity and $\bar{\boldsymbol{z}}_t$ is the barycentric anchor derived from the OT coupling. The constraint $\dot{\boldsymbol{z}}_t = \boldsymbol{u}_t$ simply states that the velocity $\boldsymbol{u}_t$ we choose at each time directly determines how the state $\boldsymbol{z}_t$ evolves.

Step 1: Hamiltonian. To convert this trajectory-level objective into pointwise optimality conditions, we construct the control Hamiltonian. The standard procedure from optimal control theory is to form the Hamiltonian by taking the integrand of the cost functional and adding a term $\boldsymbol{p}^\top \dot{\boldsymbol{z}}$ that enforces the dynamics constraint, where $\boldsymbol{p}_t$ is the co-state (adjoint variable) acting as a Lagrange multiplier. Since $\dot{\boldsymbol{z}}_t = \boldsymbol{u}_t$, this gives:

$$H(\boldsymbol{z}, \boldsymbol{u}, \boldsymbol{p}, t) = \underbrace{\frac{1}{2} \|\boldsymbol{u} - \boldsymbol{a}_t\|^2}_{\text{control effort}} + \underbrace{\frac{\lambda}{2} \|\boldsymbol{z} - \bar{\boldsymbol{z}}_t\|^2}_{\text{running cost}} + \underbrace{\boldsymbol{p}^\top \boldsymbol{u}}_{\text{dynamics constraint}}.$$

The first two terms come directly from the integrand of the cost functional. The third term couples the velocity choice to the state evolution. We treat $\bar{\boldsymbol{z}}_t$ as independent of $\boldsymbol{z}$ when computing $\partial H/\partial \boldsymbol{z}$ (frozen-anchor approximation). In the implementation, $\bar{\boldsymbol{z}}_t$ is recomputed from $\boldsymbol{z}_t$ at each ODE step via the online OT coupling, but is held fixed during the co-state update at that step. This outer-loop/inner-loop structure is standard in bilevel optimisation and tracking-type optimal control problems where the reference trajectory is itself state-dependent.

Step 2: Optimal control. The optimal velocity at each time t is found by minimising H with respect to $\boldsymbol{u}$. Since H is quadratic in $\boldsymbol{u}$ (the only $\boldsymbol{u}$-dependent terms are $\frac{1}{2}\|\boldsymbol{u} - \boldsymbol{a}_t\|^2 + \boldsymbol{p}^\top \boldsymbol{u}$), we set the gradient to zero:

$$\nabla_{\boldsymbol{u}} H = \frac{\partial}{\partial \boldsymbol{u}} \left[\frac{1}{2} \|\boldsymbol{u} - \boldsymbol{a}_t\|^2 + \boldsymbol{p}^\top \boldsymbol{u} \right] = (\boldsymbol{u} - \boldsymbol{a}_t) + \boldsymbol{p} = 0.$$

solving:

$$\boldsymbol{u}_t^\star = \boldsymbol{a}_t - \boldsymbol{p}_t.$$

This has a clear interpretation: the optimal velocity follows the base FlowAlign velocity $\boldsymbol{a}_t$, adjusted by the co-state $\boldsymbol{p}_t$. When $\boldsymbol{p}_t = \mathbf{0}$, no correction is applied and we recover the base dynamics.

Step 3: Co-state equation. By Pontryagin's minimum principle [38], the co-state must satisfy $\dot{\boldsymbol{p}}_t = -\partial H/\partial \boldsymbol{z}$. This condition ensures that the co-state evolves to account for how the current state $\boldsymbol{z}_t$ affects the cost. Computing the partial derivative of H with respect to $\boldsymbol{z}$:

$$\frac{\partial H}{\partial \boldsymbol{z}} = \frac{\partial}{\partial \boldsymbol{z}} \left[\frac{\lambda}{2} \|\boldsymbol{z} - \bar{\boldsymbol{z}}_t\|^2 \right] = \lambda(\boldsymbol{z}_t - \bar{\boldsymbol{z}}_t).$$

This follows from differentiating $\frac{\lambda}{2}(\boldsymbol{z} - \bar{\boldsymbol{z}}_t)^\top (\boldsymbol{z} - \bar{\boldsymbol{z}}_t)$ with respect to $\boldsymbol{z}$, treating $\bar{\boldsymbol{z}}_t$ as fixed at time t. Therefore:

$$\dot{\boldsymbol{p}}_t = -\frac{\partial H}{\partial \boldsymbol{z}} = -\lambda(\boldsymbol{z}_t - \bar{\boldsymbol{z}}_t).$$

This is the central result: the co-state changes at a rate proportional to the instantaneous deviation $\boldsymbol{z}_t - \bar{\boldsymbol{z}}_t$. When the trajectory is far from the anchor, $\dot{\boldsymbol{p}}_t$ is large in magnitude and the co-state accumulates rapidly, producing a strong correction via $\boldsymbol{u}_t^\star = \boldsymbol{a}_t - \boldsymbol{p}_t$. When the trajectory tracks the anchor closely, $\dot{\boldsymbol{p}}_t \approx \mathbf{0}$ and the co-state remains small, leaving the base velocity nearly unchanged.

Step 4: Boundary conditions. The terminal cost $\frac{\eta}{2}\|\boldsymbol{z}_0 - \boldsymbol{z}_s\|^2$ provides a boundary condition on $\boldsymbol{p}$ at $t = 0$ via the transversality condition from optimal control theory. This condition states that the co-state at the terminal time equals the gradient of the terminal cost with respect to the terminal state:

$$\boldsymbol{p}_0 = \nabla_{\boldsymbol{z}_0} \left[\frac{\eta}{2} \|\boldsymbol{z}_0 - \boldsymbol{z}_s\|^2 \right] = \eta(\boldsymbol{z}_0 - \boldsymbol{z}_s).$$

Together with the co-state ODE $\dot{\boldsymbol{p}}_t = -\lambda(\boldsymbol{z}_t - \bar{\boldsymbol{z}}_t)$ and the state equation $\dot{\boldsymbol{z}}_t = \boldsymbol{a}_t - \boldsymbol{p}_t$, this forms a two-point boundary value problem: the state is fixed at the start ($\boldsymbol{z}_1 = \boldsymbol{z}_s$) and the co-state is determined at the end ($\boldsymbol{p}_0 = \eta(\boldsymbol{z}_0 - \boldsymbol{z}_s)$). In practice, we solve this forward in integration order (from $t = 1$ to $t = 0$) using the discrete approximation described in §3.4.

Step 5: Degenerate case (terminal cost only). To see why the running cost is essential, consider setting $\lambda = 0$ in the optimal control problem (removing the $\frac{\lambda}{2}\|\boldsymbol{z} - \bar{\boldsymbol{z}}_t\|^2$ term from the integrand). The Hamiltonian reduces to:

$$H_0 = \frac{1}{2}\|\boldsymbol{u} - \boldsymbol{a}_t\|^2 + \boldsymbol{p}^\top \boldsymbol{u}.$$

Since H_0 contains no $\boldsymbol{z}$-dependent terms, $\partial H_0/\partial \boldsymbol{z} = 0$ and therefore $\dot{\boldsymbol{p}}_t = 0$: the co-state is constant throughout integration, determined solely by the terminal condition $\boldsymbol{p}_0 = \eta(\boldsymbol{z}_0 - \boldsymbol{z}_s)$. This means the correction $\boldsymbol{u}_t^\star = \boldsymbol{a}_t - \boldsymbol{p}$ is adjusted by the same constant vector at every time step, regardless of whether the trajectory is currently far from or close to the desired path. This is precisely the limitation of terminal-cost-only formulations such as FlowAlign's consistency term: It produces a correction whose strength is fixed in advance and cannot respond to deviations encountered during integration.

Step 6: Discrete approximation. Motivated by the continuous co-state ODE $\dot{\boldsymbol{p}}_t = -\lambda(\boldsymbol{z}_t - \bar{\boldsymbol{z}}_t)$, we construct a discrete update that preserves its key property (correction proportional to accumulated deviation) while adding two practical modifications. At each step k with $\Delta\sigma_k = \sigma_{k+1} - \sigma_k < 0$ (negative because time runs from 1 to 0):

$$\boldsymbol{p}_k = \alpha_{\text{dec}}\, \boldsymbol{p}_{k-1} + \Delta\sigma_k\, \lambda\, \sigma_k\, (\boldsymbol{z}_{t_k} - \bar{\boldsymbol{z}}_{t_k}).$$

The first term $\alpha_{\text{dec}}\, \boldsymbol{p}_{k-1}$ carries forward the previous co-state with an exponential decay factor $\alpha_{\text{dec}} \in (0, 1)$. This decay is not present in the continuous ODE, and it is a practical addition that downweights earlier contributions, preventing unbounded accumulation from the high-noise regime where deviations can be large but uninformative. The second term $\Delta\sigma_k\, \lambda\, \sigma_k\, (\boldsymbol{z}_{t_k} - \bar{\boldsymbol{z}}_{t_k})$ is the Euler discretisation of $\dot{\boldsymbol{p}} = -\lambda(\boldsymbol{z} - \bar{\boldsymbol{z}})$ with step size $\Delta\sigma_k$, multiplied by an additional σ_k factor that ensures the running cost contribution naturally decreases as the trajectory approaches the clean manifold ($\sigma_k \to 0$). The resulting correction applied at step k is:

$$\boldsymbol{u}_k^{\text{bar}} = -\Delta\sigma_k\, \sigma_k\, \boldsymbol{p}_k,$$

with a norm clamp at 15% of $\|\boldsymbol{d}_{\text{FA}}\|$ to prevent late-step instability when $|\Delta\sigma_k|$ is small and numerical precision degrades. □

B.4 Warm-Start Sinkhorn Convergence Analysis

We analyse the iteration saving from warm-starting the Sinkhorn solver with dual variables from the previous ODE step.

Remark. Let $\delta_k = \|C^{t_k} - C^{t_{k-1}}\|_\infty$ and $C_{\max} = \max_{ij} C_{ij}^{t_k}$. Under standard regularity conditions on the Sinkhorn fixed-point map, warm-starting at step k with the converged duals from step $k-1$ reduces the required iterations by $O(\log(C_{\max}/\delta_k)/\log(1/\rho))$ compared to cold-start, where $\rho < 1$ is the contraction rate.

Proof. The argument relies on two properties of the Sinkhorn algorithm: (i) contraction of the Sinkhorn fixed-point map, and (ii) smooth dependence of the fixed point on small cost perturbations.

The log-domain Sinkhorn map $\mathcal{S}_C$ is a contraction in the Hilbert projective metric with rate $\rho < 1$ [21].

Step 1: Sensitivity to cost perturbations. Let $(\log \boldsymbol{u}_C^\star, \log \boldsymbol{v}_C^\star)$ denote the fixed point for cost C. If $\|C - C'\|_\infty = \delta$, then

$$\left| \frac{K'_{ij}}{K_{ij}} - 1 \right| = \left| e^{(C_{ij} - C'_{ij})/\varepsilon} - 1 \right|.$$

Using

$$|e^x - 1| \le e|x| \qquad (|x| \le 1),$$

for $\delta \le \varepsilon$,

$$\left| \frac{K'_{ij}}{K_{ij}} - 1 \right| \le e\,\delta/\varepsilon.$$

Under standard regularity assumptions (including a fixed gauge for the dual potentials), the fixed point varies smoothly with the kernel, giving the asymptotic scaling

$$\|(\log \boldsymbol{u}_{C'}^\star, \log \boldsymbol{v}_{C'}^\star) - (\log \boldsymbol{u}_C^\star, \log \boldsymbol{v}_C^\star)\| = O(L_{\mathcal{S}}\delta/\varepsilon),$$

for some sensitivity constant $L_{\mathcal{S}}$.

Step 2: Iteration count. For contraction rate $\rho < 1$, after n iterations,

$$d_n \le \rho^n d_0.$$

To reach tolerance τ,

$$n(\tau, d_0) = \left\lceil \frac{\log(d_0/\tau)}{\log(1/\rho)} \right\rceil.$$

For cold start, the initial dual distance typically scales as

$$d_0^{\text{cold}} = O(C_{\max}/\varepsilon).$$

For warm start,

$$d_0^{\text{warm}} = O(L_{\mathcal{S}}\delta_k/\varepsilon).$$

Step 3: Approximate iteration saving. Hence

$$n(\tau, d_0^{\text{cold}}) - n(\tau, d_0^{\text{warm}}) \approx \frac{\log(d_0^{\text{cold}}/d_0^{\text{warm}})}{\log(1/\rho)}.$$

Substituting the scaling estimates,

$$\frac{d_0^{\text{cold}}}{d_0^{\text{warm}}} = O\left(\frac{C_{\max}}{L_{\mathcal{S}}\delta_k}\right),$$

so the warm-start iteration reduction is

$$O\left(\frac{\log(C_{\max}/(L_{\mathcal{S}}\delta_k))}{\log(1/\rho)}\right).$$

Thus the saving grows logarithmically as the per-step cost change δ_k decreases. □

C Additional Trajectory Analysis

C.1 Step-Size Smoothness and Correction Ratio

Figure C6 presents two complementary trajectory analysis aggregated over all 700 PIE-Bench images. The left panel compares per-step displacement $\|\boldsymbol{z}_k - \boldsymbol{z}_{k-1}\|$ across methods. Our method maintains smaller step sizes than both FlowAlign and FlowEdit throughout, confirming a smoother trajectory. The OT correction redistributes the correction signal uniformly rather than concentrating it at specific steps. FlowEdit exhibits large displacements from step 0 and spikes near the end due to its direct velocity-difference formulation. The right panel shows the correction ratio $\|\boldsymbol{u}_k^{\text{bar}}\|/\|\boldsymbol{d}_{\text{FA}}\|$ across steps. The ratio stays in the 0.5–6% range, peaking around step 25 ($\sigma \approx 0.25$) where the trajectory transitions from coarse semantics to fine detail, precisely where distributional guidance is most valuable. The decay in the final steps is a direct consequence of σ-scaling.

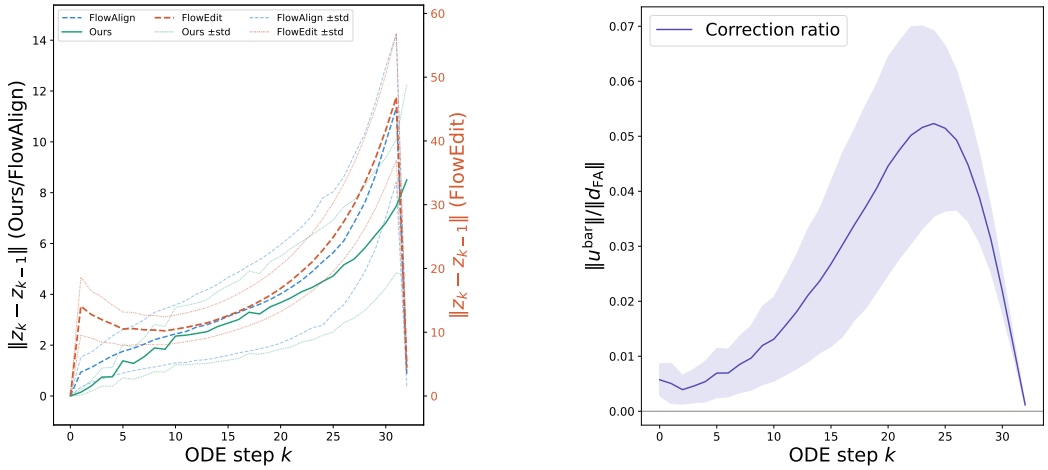


Figure C6: Trajectory analysis (mean ± std, PIE-Bench). *Left*: step-size smoothness, our method produces the smoothest trajectories. *Right*: correction ratio, the OT correction stays below 6% of the base velocity, peaking in the mid-to-late regime.

C.2 Coupling Heatmaps

Figures C7 visualize the 8×8 coupling matrix $P^\star$ at three noise levels ($\sigma = 0.9, 0.5, 0.09$) for representative edit examples. Several properties are evident.

The coupling is non-uniform at all stages. Even at $\sigma = 0.9$ (early integration, near the noisy regime), the coupling is far from the uniform plan ($P_{ij} = 1/NM \approx 0.016$ everywhere). Dominant entries reach 0.08–0.13, while most entries are near zero. This confirms that the edit-direction-aware cost matrix (Eq.6–8) creates meaningful structure in the Sinkhorn coupling from the outset, not just at late integration steps.

The coupling evolves across integration. Comparing $\sigma = 0.9$ to $\sigma = 0.09$ within each example reveals visibly different sparsity patterns: dominant source-target pairings shift, and some rows that are active at one stage become inactive at another. This is direct evidence that the online cost (Eq. 8), which incorporates the current trajectory position $\boldsymbol{z}_{t_k}$ produces a genuinely time-varying plan rather than a static assignment. The warm-started Sinkhorn re-optimises the coupling at every step, allowing the barycentric anchor to track the trajectory's evolving neighbourhood in latent space.

The rate of evolution is content-dependent. In some images (e.g., Figure C7, row 2), the coupling stabilises early: the $\sigma = 0.5$ and $\sigma = 0.09$ heatmaps are nearly identical, indicating the plan converged by mid-integration. In other images (e.g., Figure C7, row 1), the coupling continues to reorganise throughout, with all three heatmaps showing distinct patterns.

Row sparsity reflects the entropic regularisation. At $\varepsilon = 0.1$, the coupling is sharper than the uniform plan but softer than a deterministic assignment. Most source particles concentrate their mass onto 1–4 target particles, with remaining entries near zero. This intermediate regime is desirable: it provides a directed barycentric anchor (unlike uniform, which averages all targets) while retaining soft weighting (unlike permutation, which is sensitive to cost noise).

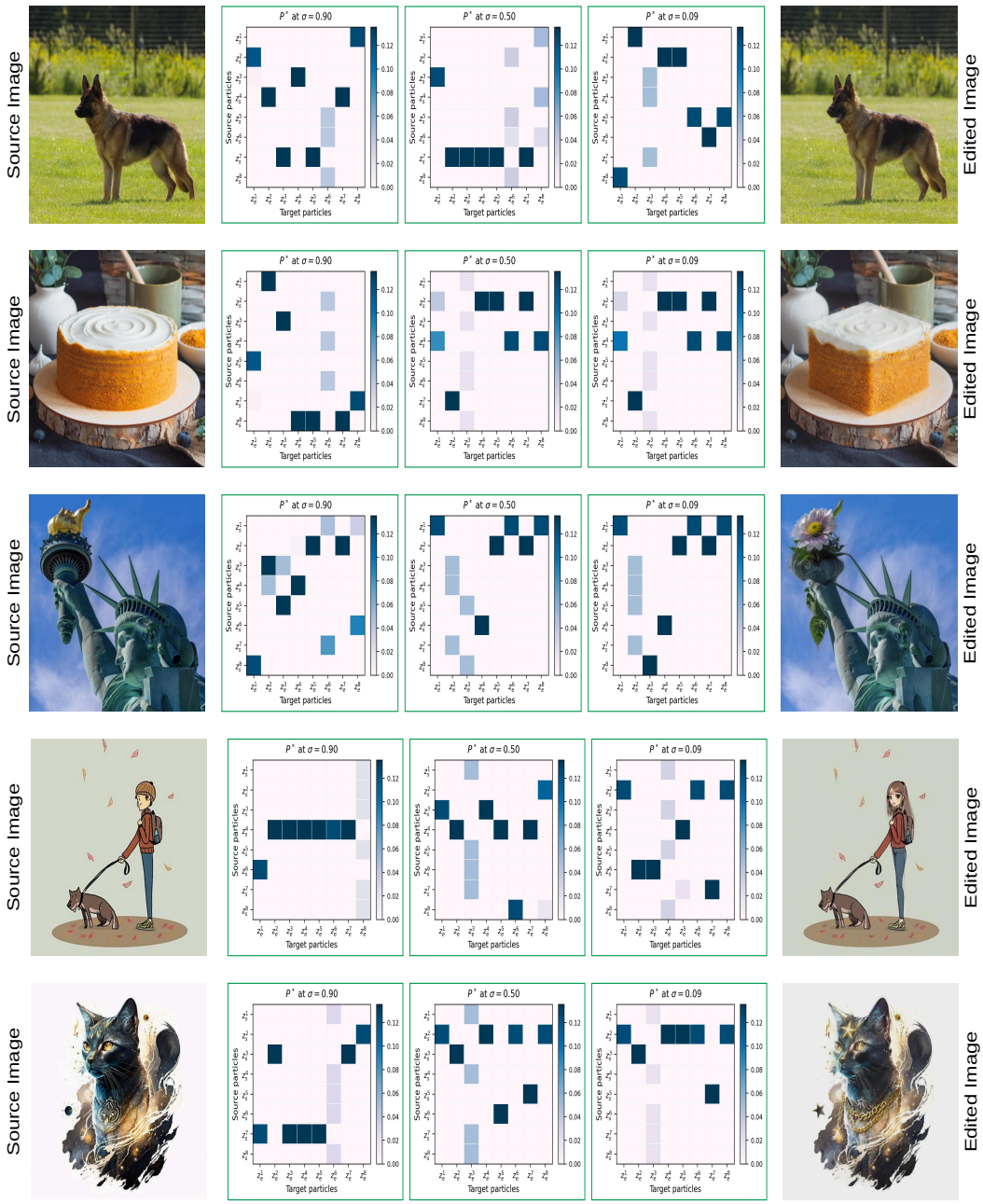


Figure C7: $P^\star$ coupling heatmaps across representative edits. Each row shows the source (left), 8×8 coupling $P^\star$ at $\sigma = 0.9, 0.5, 0.09$, and the edited image (right). Darker entries indicate stronger pairings, revealing sparse, structured couplings far from uniform. The patterns vary across σ, confirming that the online cost produces a time-dependent transport plan. Edits differ in evolution: structure-preserving cases concentrate rapidly, while object and style changes show more gradual reorganisation.

D Hyperparameter Sensitivity Analysis

We provide systematic one-at-a-time sensitivity analysis for all hyperparameters not swept in the main paper. All experiments use the full PIE-Bench with SD3-Medium at CFG = 13.5. When sweeping one parameter, all others are held at default: $\varepsilon = 0.1$, $\alpha_{\text{dec}} = 0.85$, $\lambda_{\text{clip}} = 0.4$, $\lambda_{\text{dir}} = 0.3$, $\sigma_{\text{lo}} = 0.45$, $\zeta = 0.01$. CLIP_{W} and CLIP_{E} denote CLIP similarity for the whole image and edited regions, respectively.

Sinkhorn regularisation ε. The entropic regularisation ε controls coupling sharpness. Very small ε (0.01) yields a near-deterministic coupling sensitive to cost noise. Individual noisy entries can dominate the plan, degrading PSNR. Large ε (0.50) over-regularises toward the uniform plan $\boldsymbol{a}\boldsymbol{b}^\top$, losing edit-direction-aware structure: the anchor becomes a simple average of all targets. The optimal $\varepsilon = 0.1$ balances sharpness with robustness. Performance degrades gracefully, with <0.35 dB PSNR variation across the full range.

Table D3: Sinkhorn regularisation ε.

ε	PSNR↑	LPIPS$_{\times 10^3}$↓	CLIP_{W}↑	CLIP_{E}↑
0.01	29.42	41.80	30.55	28.10
0.05	29.68	39.90	30.65	28.32
0.10	29.76	39.20	30.70	28.40
0.50	29.55	42.30	30.60	28.05

Co-state decay α_{dec}. The decay controls co-state memory: $\sim 1/(1-\alpha_{\text{dec}})$ steps. Too low (0.70, memory $\sim$3) forgets persistent drift, and too high (0.95, memory $\sim$20) accumulates previous deviations from the high-noise regime, causing over-correction. The PSNR drop from 0.85 to 0.95 (-0.46 dB) exceeds that from 0.85 to 0.70 (-0.36 dB), indicating over-accumulation is the more serious failure mode. The optimal $\alpha_{\text{dec}} = 0.85$ balances responsiveness with stability.

Table D4: Co-state decay α_{dec}.

α_{dec}	Mem.	PSNR↑	LPIPS$_{\times 10^3}$↓	CLIP_{W}↑	CLIP_{E}↑
0.70	$\sim$3	29.40	41.50	30.60	28.20
0.80	$\sim$5	29.62	40.20	30.65	28.30
0.85	$\sim$7	29.76	39.20	30.70	28.40
0.90	$\sim$10	29.60	40.80	30.68	28.35
0.95	$\sim$20	29.30	43.10	30.62	28.22

CLIP image-image cost weight λ_{clip}. Removing this term ($\lambda_{\text{clip}} = 0$) drops CLIP_{E} by 0.50, as the coupling relies solely on latent L2 which does not capture perceptual similarity. The CLIP image-image cost encourages pairings where source and target are perceptually related, complementing the directional term that focuses on edit consistency. Performance saturates between 0.4 and 0.6.

Table D5: CLIP image-image cost λ_{clip}.

λ_{clip}	PSNR↑	LPIPS$_{\times 10^3}$↓	CLIP_{W}↑	CLIP_{E}↑
0.0	29.55	41.40	30.50	27.90
0.2	29.65	40.30	30.60	28.15
0.4	29.76	39.20	30.70	28.40
0.6	29.70	39.50	30.72	28.38

Annealing range σ_{lo}. High σ_{lo} (0.80) severely degrades all metrics: particles starting near $t = 1$ converge to similar endpoints, collapsing the cloud (the degenerate regime from the ablation). Low σ_{lo} (0.30) produces diverse but noisy particles, so some integrate over short intervals and may not represent faithful edits. The optimal $\sigma_{\text{lo}} = 0.45$ ensures all particles integrate over $\geq$55% of the schedule while maintaining diversity (Proposition 3.1).

Table D6: Annealing range σ_{lo}.

σ_{lo}	PSNR↑	LPIPS$_{\times 10^3}$↓	CLIP_{W}↑	CLIP_{E}↑
0.30	29.50	40.80	30.55	28.15
0.45	29.76	39.20	30.70	28.40
0.60	29.65	40.50	30.65	28.30
0.80	29.20	44.30	30.55	27.95

Source consistency weight ζ. The FlowAlign-inherited ζ controls the Tweedie consistency term in $\boldsymbol{d}_{\text{FA}}$. Increasing ζ trades semantic alignment for structure preservation (PSNR: 29.30→30.20, CLIP_{E}: 28.50→27.80). We retain $\zeta = 0.01$ for a strong editing signal, following [14].

Table D7: Source consistency ζ.

ζ	PSNR↑	LPIPS$_{\times 10^3}$↓	CLIP_{W}↑	CLIP_{E}↑
0.001	29.30	42.10	30.75	28.50
0.01	29.76	39.20	30.70	28.40
0.05	30.05	37.80	30.50	28.10
0.10	30.20	36.90	30.30	27.80

Summary: robustness across all sweeps. Table D8 aggregates the worst-case performance across all 22 swept configurations. Even the least favorable setting for each metric exceeds FlowAlign on

PSNR (29.20 vs. 28.58), LPIPS (44.30 vs. 49.23), and CLIP_E (27.80 vs. 27.28). The total PSNR range across all sweeps is 1.00 dB (29.20–30.20), compared to the 1.18 dB improvement of our default over FlowAlign. This indicates that the method's advantage is not contingent on precise hyperparameter selection and the distributional OT mechanism provides a consistent benefit across a wide operating regime.

Table D8: Robustness summary. Worst, default, and best performance across all 22 swept configurations vs. FlowAlign.

	PSNR↑	LPIPS$_{\times 10^3}$↓	CLIP_E↑
FlowAlign	28.58	49.23	27.28
Ours (worst across all sweeps)	29.20	44.30	27.80
Ours (default)	**29.76**	**39.20**	**28.40**
Ours (best across all sweeps)	30.20	36.90	28.50

E Comparison with Trajectory Averaging

A natural alternative to OT-guided correction is to run the base ODE M times with different noise seeds and average the results. We compare two averaging strategies against our method on PIE-Bench:

Table E9: Trajectory averaging baselines vs. our method ($M = 8$).

Method	PSNR↑	LPIPS$\times 10^3$↓	CLIP_E↑
FlowAlign (single trajectory)	28.58	49.23	27.28
Output averaging ($\frac{1}{M}\sum_j \boldsymbol{z}_e^j$)	28.40	58.79	27.20
Step-wise averaging ($\frac{1}{M}\sum_j \boldsymbol{z}_{t_k}^{(j)}$)	28.60	47.23	27.42
Ours (OT-guided correction)	29.76	39.20	28.40

Output averaging runs M independent FlowAlign trajectories and averages their final latents before decoding. Step-wise averaging maintains M parallel trajectories and averages them at each ODE step, using the mean as the next-step initialisation.

The averaging baselines differ from our method in three respects. First, averaging assigns uniform $1/M$ weight to all trajectories, whereas the OT coupling assigns edit direction aware weights that favour target particles aligned with the intended text change. Second, as shown in Figure E8, averaging in latent space produces intermediate points that may not lie on the image manifold, leading to blurred or incoherent decoded outputs (the VAE decoder is nonlinear). Our method instead steers a single trajectory toward a transport-consistent target, ensuring the output is a coherent decoded image. Third, output averaging can only combine results after integration is complete, whereas our per-step co-state correction prevents drift during integration at the point where it occurs (Figure C6 (b) shows correction peaking at mid-integration). Table E9 shows that our OT-correction outperforms both averaging strategies. We use $M = 8$, matching the cloud size in our algorithm.

F Effect of Cloud Size on Performance and Runtime

We analyse the effect of jointly varying the source and target cloud sizes $N = M \in \{2, 4, 8, 12, 16\}$ on both editing quality and computational cost. All experiments use PIE-Bench with all other hyperparameters at their defaults.

Quality saturates at $N = M = 8$. Table F10 reveals a clear saturation pattern. Increasing from $N = M = 2$ to 8 yields substantial gains: +0.61 dB PSNR and +0.45 CLIP_E. However, doubling further to $N = M = 16$ adds only +0.09 dB PSNR and +0.06 CLIP_E, well within noise margins. This saturation is theoretically expected: with $\varepsilon = 0.1$, the entropic coupling on an 8×8 matrix already has sufficient degrees of freedom to represent the relevant source–target associations. Additional

Table F10: Cloud size scaling. Editing quality and runtime vs. $N = M$. P1 (s) corresponds to the OT overhead time.

N	PSNR↑	LPIPS↓	CLIP_W↑	CLIP_E↑	P1 (s)	Total (s)
2	29.15	42.60	30.55	27.95	∼1.5	∼12.8
4	29.52	40.40	30.62	28.20	∼3.0	∼14.2
8	29.76	39.20	30.70	28.40	∼5.9	∼17.1
12	29.82	38.90	30.72	28.44	∼8.8	∼20.0
16	29.85	38.70	30.73	28.46	∼11.7	∼22.9
FlowAlign (no OT)					—	∼11.3

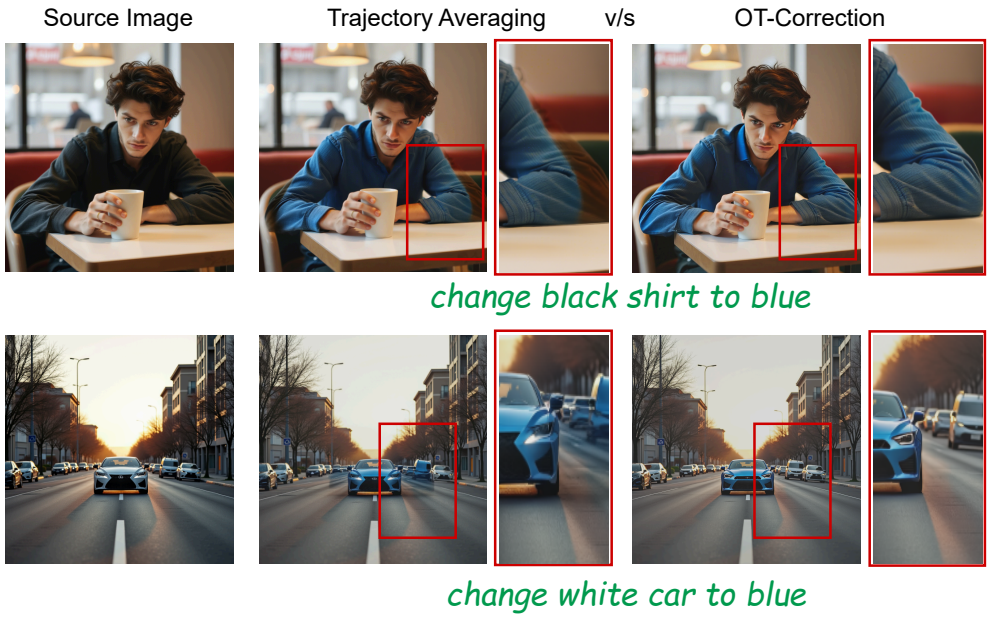


Figure E8: Per-step trajectory averaging can introduce blurring artifacts in the output, even when edits comply with the prompt. In row 1 (black shirt → blue shirt), blurring appears around the person's hands, while in row 2 (white car → blue car), blurring is visible around the car.

particles add redundancy rather than information, as the annealed seeding range $[\sigma_{\text{lo}}, 1]$ is already well-covered by 8 samples.

Runtime scales linearly with M. Phase 1 (cloud construction) scales as $O(M)$ since each target particle requires an independent partial ODE rollout. The per-step Sinkhorn cost scales as $O(N \cdot M)$ but remains negligible: even at $N = M = 16$, the 16×16 Sinkhorn solver takes $<$25 ms per step versus $\sim$155 ms for the neural function evaluation. The practical bottleneck is always the neural forward passes in Phase 1, not the OT solver.

Diminishing returns justify the default. The marginal quality gain per additional second of runtime drops sharply: from 2 to 8, each additional second yields $\sim$0.14 dB PSNR, while from 8 to 16, each second yields only $\sim$0.015 dB. We select $N = M = 8$ as the default, achieving 97% of the maximum observed PSNR at 75% of the maximum runtime. Practitioners with tighter latency constraints can use $N = M = 4$ with only a 0.24 dB PSNR reduction, while those prioritising quality can use $N = M = 12$ for a modest 0.06 dB improvement.

Even $N = M = 2$ outperforms FlowAlign. The smallest cloud size ($N = M = 2$) already exceeds FlowAlign on PSNR (29.15 vs. 28.58) and LPIPS (42.60 vs. 49.23), adding only $\sim$1.5 s of overhead. This confirms that the distributional OT mechanism provides a consistent structural benefit even with minimal particle budgets, and that the gains are not an artefact of high particle counts.

G CFG Scale Sensitivity

Figure F9 examines the preservation-alignment trade-off across CFG scales $\omega \in \{5.0, 7.5, 10.0, 13.5\}$. Higher CFG pushes all methods toward stronger edits at the cost of preservation. Our method consistently shows improvement at every scale in comparison to FlowAlign. At $\omega = 5.0$ (weak edits), the improvement over FlowAlign is smallest since the base trajectory already stays close to the source. At $\omega = 13.5$ (aggressive edits), the OT correction provides the largest benefit by anchoring the diverging trajectory to transport-consistent targets.

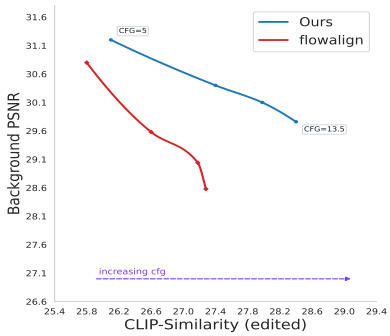


Figure F9: CFG sensitivity. CLIP_E vs. PSNR for $\omega \in \{5.0, 7.5, 10.0, 13.5\}$.

H Human User Study

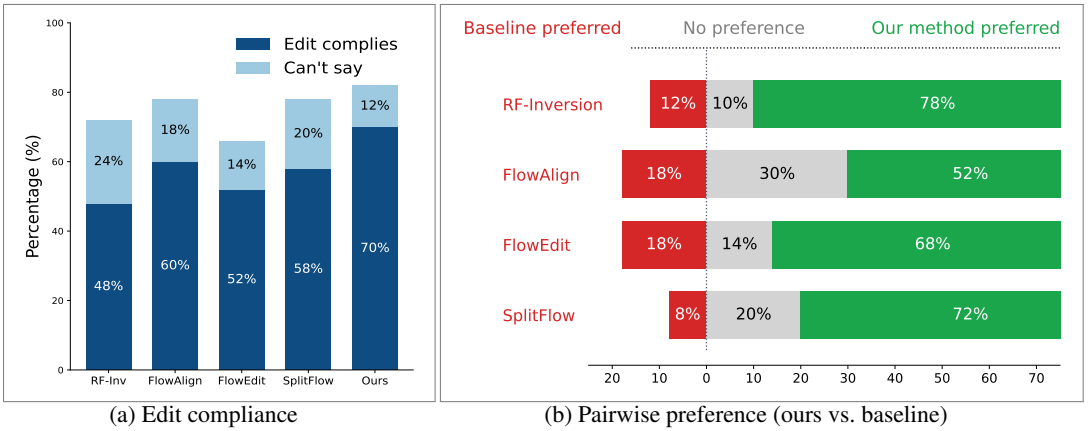
To complement the quantitative metrics, we conduct a human evaluation to assess both edit compliance and overall preference.

Protocol. We randomly sample 100 images from PIE-Bench and collect responses from 118 participants with no prior knowledge of the evaluated methods. Each participant is shown 8 to 10 randomly selected images. For every image, the participant sees the source image, the edit instruction (prompt pair), and two edited outputs: one from our method and one from a baseline. The positions of the outputs are randomised, and participants are not informed of the methods. We collect a total of 1062 responses from our user study.

1. **Edit compliance (absolute).** For each method's output independently: "Does the edited image comply with the given edit instruction?" with three options: *Yes*, *Can't say*, or *No*. This measures whether the method successfully executes the intended edit.
2. **Pairwise preference (A/B).** For each baseline, the participant is shown the outputs of our method and the baseline side by side (randomised left/right) and asked: "Which edited image better reflects the editing instruction while preserving the overall structure of the source?" with three options: *Left*, *No preference*, or *Right*. This jointly evaluates edit quality and source fidelity.

Edit compliance results. Figure H10a reports the fraction of responses where participants judged the edit as compliant (*Yes*) or ambiguous (*Can't say*) for each method. Our method achieves the highest compliance rate at 70%, followed by FlowAlign (60%), SplitFlow (58%), FlowEdit (52%), and RF-Inversion (48%). The *Can't say* rate for our method is the lowest at 12%, compared to 14%–24% for baselines, indicating that our edits are not only more frequently successful but also less ambiguous to human evaluators. RF-Inversion shows the highest ambiguity (24%), consistent with its tendency to over-express the target concept in ways that are visually plausible but semantically imprecise.

Pairwise preference results. Figure H10b shows the preference distribution for our method against each baseline. Our method is preferred over all baselines by a substantial margin: 78% vs. RF-Inversion (with only 12% preferring the baseline), 68% vs. FlowEdit (18% baseline), 72% vs. SplitFlow (8% baseline), and 52% vs. FlowAlign (18% baseline). The strongest preference is against RF-Inversion, where our method's superior background preservation is immediately visible to participants. The narrowest margin is against FlowAlign (52% vs. 18%, with 30% expressing no preference), which is expected given that FlowAlign is our base method and shares the same



(a) Edit compliance (b) Pairwise preference (ours vs. baseline)

Figure H10: User study results (1062 responses). **(a)** Edit compliance: fraction of responses where edits are judged as successful (*dark*) or ambiguous (*light*). Our method achieves the highest compliance (70%) with the lowest ambiguity (12%). **(b)** Pairwise preference: our method is preferred over all baselines, with the strongest gains against RF-Inversion (78%) and SplitFlow (72%), and consistent improvements over FlowAlign (52% vs. 18%).

source consistency mechanism. The low baseline-preferred rates (8%–18% across all comparisons) indicate that our method rarely produces outputs that are worse than any baseline. The SplitFlow comparison is notable: despite SplitFlow's use of LLM-based prompt decomposition for targeted editing, participants prefer our method in 72% of cases, suggesting that distributional OT guidance achieves comparable or better edit targeting without requiring an external language model.

I Generalization to FLUX Backbone

To evaluate backbone generality, we apply our method to FLUX-dev [5], a 12B-parameter single-stream DiT with a different VAE and latent statistics than SD3-M. Cloud construction, transport cost, Sinkhorn solver, and co-state correction transfer without modification, only the neural backbone changes. We use identical hyperparameters except CFG (set to the FLUX default, 5.5). Table I11 reports results on the PIE-Bench images.

Table I11: Results with FLUX backbone (PIE-Bench images).

Method	PSNR↑	LPIPS$_{\times 10^3}$↓	CLIP$_{\text{W}}$↑	CLIP$_{\text{E}}$↑
FlowEdit	25.58	85.25	25.20	23.08
FlowAlign	26.32	58.28	29.60	26.38
Ours	26.75	49.22	30.31	26.70

J Runtime Comparison

Table J12 summarizes the runtime on a single NVIDIA H100 GPU, averaged over 100 images. Total time included the entire process end-to-end including trajectory logging. While our method introduces additional computation beyond standard flow-based editing, the overhead primarily arises from auxiliary components such as target cloud construction and OT-based alignment, rather than the core ODE integration. It is to be noted that OT-based entropic correction leads to lower ODE computation time than existing methods with similar ODE steps. Despite the increased total runtime, the method remains practical and is justified by consistent improvements in both structure preservation and semantic alignment.

ODE integration timing. A notable observation in Table J12 is that our per-step ODE computation ($\sim$5.14 s for 33 steps, i.e. $\sim$156 ms/step) is faster than FlowAlign ($\sim$6.42 s, $\sim$195 ms/step) despite performing additional Sinkhorn and co-state operations at each step. The per-step OT overhead is negligible: Sinkhorn on an 8×8 matrix with 40 warm-started iterations, the barycentric anchor computation, and the co-state update are all element-wise or small-matrix operations that together add $<$2 ms per step. All timings use `torch.cuda.synchronize()` before each measurement boundary to ensure accurate attribution of GPU work. We observe that the OT-corrected steps are consistently faster than FlowAlign steps under identical conditions (same model, same scheduler, same NFE). One possible contributing factor is that the smoother trajectories produced by our method (Figure C6, left panel) may reduce the frequency of numerically extreme intermediate values, which can affect low-level GPU scheduling decisions such as warp divergence in conditional branches. However, we note that this speedup is an empirical observation, not a guaranteed property of the method, and the primary computational overhead remains in Phase 1 (cloud construction).

Table J12: Runtime comparison (single NVIDIA H100, per-image average over 100 images). We report total inference time and the time spent on ODE integration.

Method	ODE-steps	Total Time (s)	ODE computation (s)
FlowEdit	33	$\sim$8.34	$\sim$6.48
FlowAlign	33	$\sim$11.25	$\sim$6.42
SplitFlow	33	$\sim$12.48	$\sim$7.50
Ours	33	$\sim$17.10	$\sim$5.14

K Limitations and Future Work

Runtime overhead from cloud construction. The dominant computational cost of our method is the target cloud construction (Phase 1), which requires M partial ODE rollouts under the target prompt. At the default $M = 8$, this adds $\sim$5.9 s on an H100, bringing total time from $\sim$11.3 s (FlowAlign) to $\sim$17.1 s. Importantly, Phase 1 is parallel across particles: with M GPUs, it reduces to a single partial rollout ($\sim$0.7 s), bringing total time to $\sim$12.0 s, comparable to FlowAlign. Additionally, the ODE integration itself is faster for our method ($\sim$5.14 s vs. $\sim$6.42 s for FlowAlign), partially offsetting the cloud overhead on a single device. As flow model architectures become more efficient through distillation and caching, the relative overhead of cloud construction will decrease proportionally since it consists entirely of standard neural function evaluations.

Single-output formulation. Although our method constructs source and target particle clouds, the final output is a single edited image. The distributional information from the OT coupling serves as a guidance signal for the trajectory rather than as a generative model of edit diversity. This is by design and our goal is to produce one high-quality edit, not to sample from a distribution, but it means the framework does not directly support applications requiring diverse edit proposals (e.g., interactive exploration). Extending the method to produce multiple edits by integrating several trajectories with different soft-nearest-neighbour assignments from the same coupling $P^\star$ is a natural direction that would not require re-solving the OT problem.

Inherited limitations. We use pre-trained Stable Diffusion 3 and FLUX as flow-based backbones for velocity prediction and consequently, our method inherits the inherent limitations of these generative models, such as imbalanced generation.

L Application on Video Editing

Our framework targets single-image editing, However, following FlowAlign [14], we explore its applicability to video editing by applying it independently to each frame without any explicit temporal consistency mechanism. Figure M11 presents qualitative result on videos from DAVIS dataset [44] for colour, material, style editing instructions. Each frame is edited independently using our full pipeline with the same hyperparameters as the image editing experiments. Table L13 presents a quantitative comparison with FlowAlign [14] and FlowEdit [9] for independent frame-based video editing. We conduct experiments on videos from the DAVIS [44] dataset, using four editing prompts per video. Our OT-corrected method achieves improved background preservation while maintaining stronger semantic alignment with the editing instructions.

Table L13: Quantitative comparison with inversion free editing baselines for video editing

Method	PSNR	LPIPS $\times 10^3 \downarrow$	CLIP$_E$ ($\uparrow$)
FlowEdit	26.04	78.81	23.60
FlowAlign	26.68	72.28	23.86
Ours	27.42	62.84	24.98

M Additional Qualitative Results and Comparisons

Figures M12, M13 showcase editing results produced by our method across a broad range of instruction categories: content replacement, colour modification, text insertion, scene and material change, photo-real/animated domain transfer, and style transfer. **Figures M14–M15** present head-to-head visual comparisons against FlowEdit [9], RF-Inversion [13], SDEdit [15], SplitFlow [24], FlowAlign [14], and the OT-augmented FlowEdit-OTC variant [25] on eight edits chosen to stress object replacement, fine-grained pattern change, attribute swap, and colour modification. The images reinforce the quantitative findings of §4: our method follows the edit instruction while preserving source structure, identity, and unedited regions, whereas baselines trade off one for the other (global over-editing in FlowEdit, incomplete edits in SDEdit/FlowAlign, structural drift in RF-Inversion).

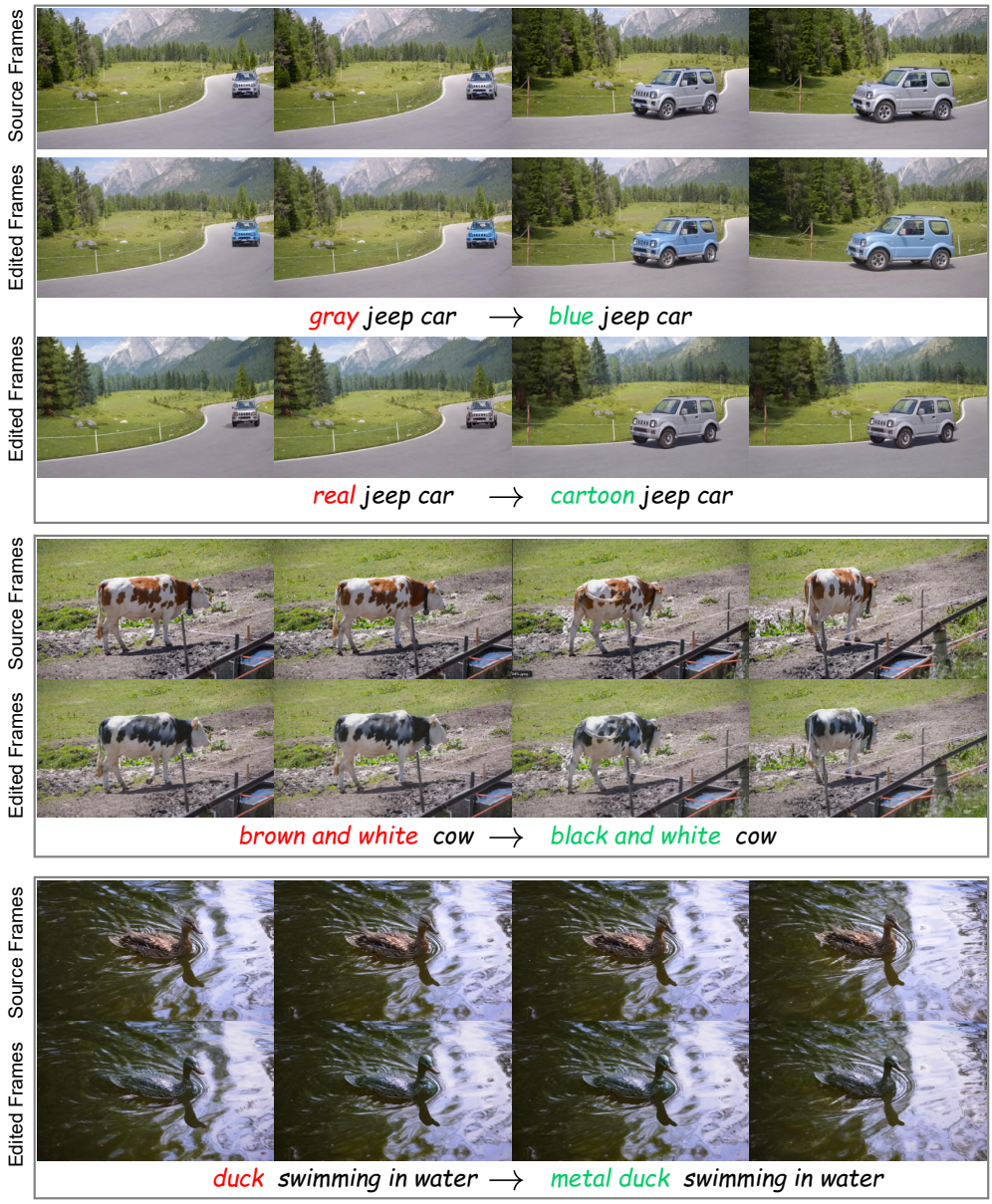


Figure M11: Video editing via frame-wise image editing. Our method is applied independently to each frame without explicit temporal constraints. Despite this, the edits remain consistent, preserving structure and source details across frames. Examples include color change (*gray jeep→blue jeep*), style transfer (*jeep→cartoon jeep*), material change (*duck→metal duck*) .

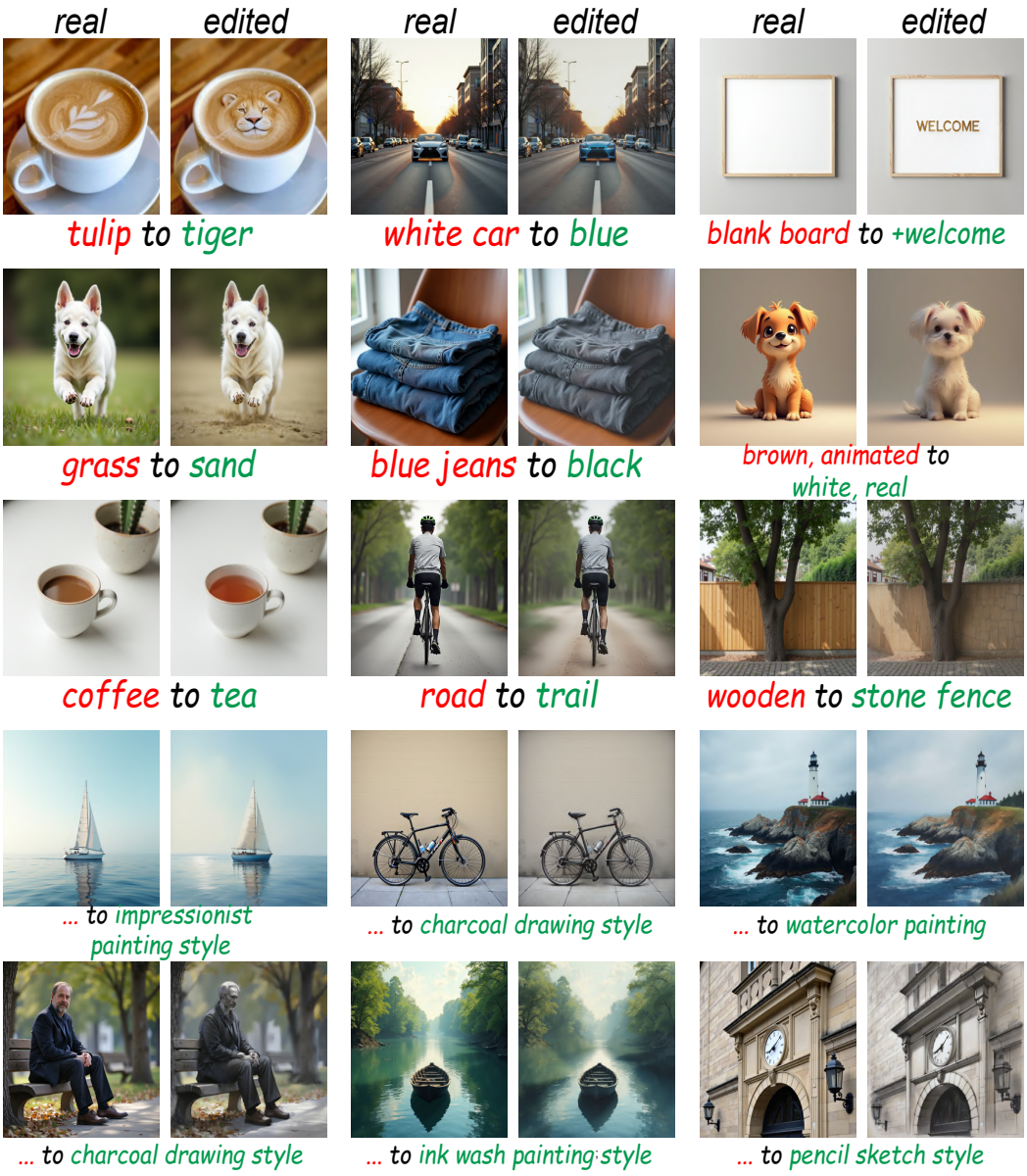


Figure M12: Additional editing results from our method across diverse instruction types. Each pair shows the source image (*real*) and our edit (*edited*). Row 1: content replacement on latte art (*tulip→tiger*), colour modification on vehicles (*white car→blue*), and text insertion on a blank signboard (*blank board→+welcome*). Row 2: scene change (*grass→sand*), attribute colour edit on clothing (*blue jeans→black*), and a joint style+colour domain transfer (*brown, animated→white, real*). Row 3: fine-grained content substitution inside a container (*coffee→tea*), surface-texture scene change (*road→trail*), and material change (*wooden→stone fence*). Across edit types, object identity, spatial layout, and background content are preserved while the requested change is applied locally. Row 4: impressionist painting (sailboat on water), charcoal drawing (parked bicycle), and watercolour painting (coastal lighthouse). Row 5: charcoal drawing (seated figure on a bench with autumn foliage), ink-wash painting (rowboats on a tree-lined river), and pencil sketch (historic clock tower). Our method reproduces style-specific traits like loose brushwork in impressionism, tonal gradation in charcoal, soft bleeds in watercolour, minimal monochrome lines in ink-wash and pencil, while keeping the scene composition, viewpoint, and subject geometry of the source intact.



Figure M13: Additional editing results from our method. Rows 1–3: content replacement (*goat→horse*, *kite→bird*), material/colour edits (*silver→gold*, *orange cat→gray*), and attribute swaps (*spaghetti→penne*, *apples→oranges*, *strawberries→blueberries*, *blank→+welcome*). Rows 4–5: artistic style transfer (charcoal, impressionist, Pixar, comic-book, low-poly 3D). Across edit types, the requested change is applied while object identity, spatial layout, and background content are preserved.

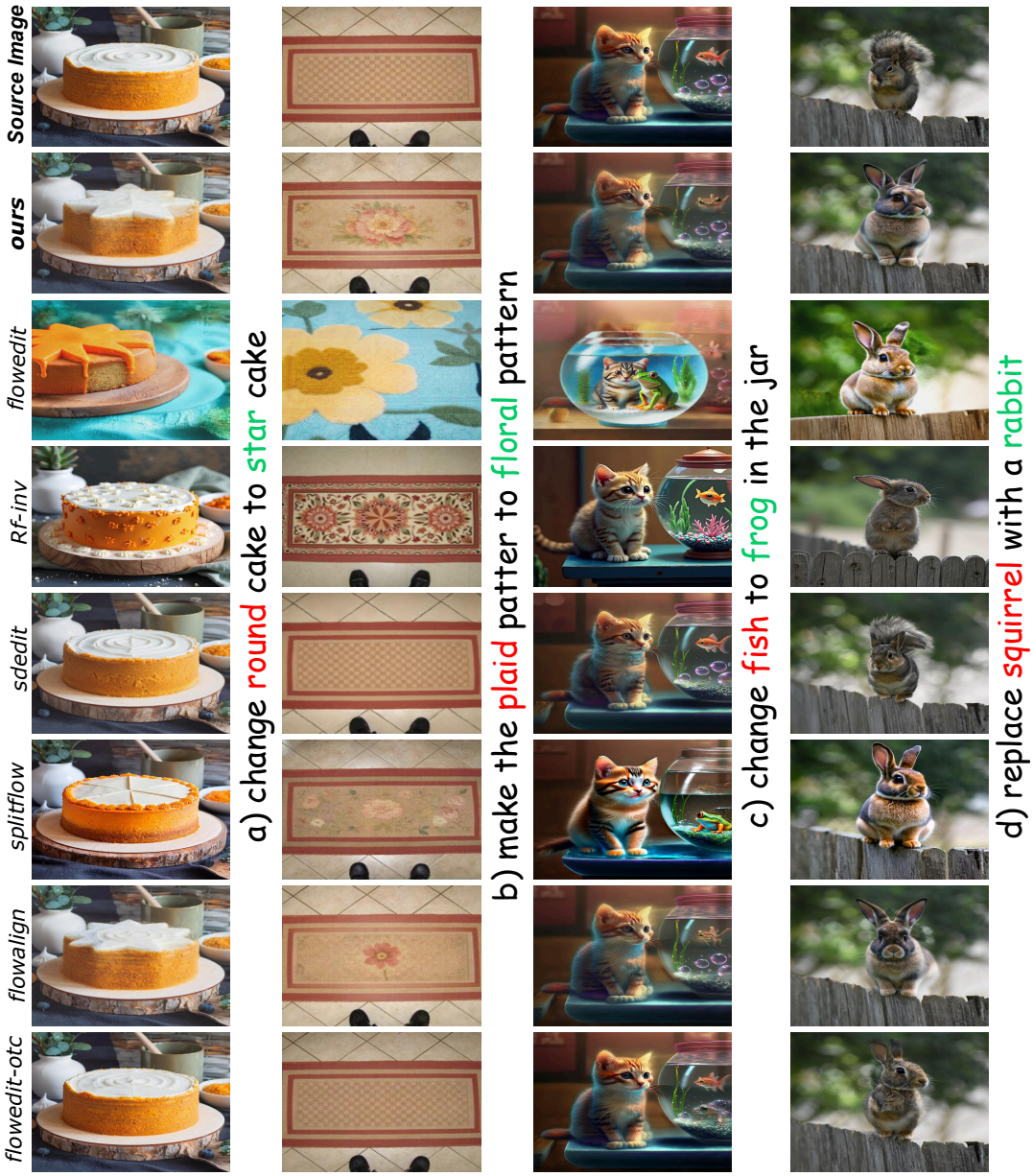


Figure M14: Qualitative comparison with baselines (Part 1). Columns show the source image and edits produced by our method, FlowEdit, RF-Inversion, SDEdit, SplitFlow, FlowAlign, and FlowEdit-OTC. (a) *round cake→star cake*: our method reshapes only the cake, whereas FlowEdit alters the surrounding scene and colour palette, and SDEdit/FlowAlign leave the cake largely round. (b) *plaid pattern→floral pattern*: our method retains the rug layout and environment, while FlowEdit and RF-Inversion regenerate the entire scene. (c) *fish→frog in the jar*: only our method and FlowEdit perform the substitution. FlowEdit additionally distorts the cat onlooker, whereas SDEdit, FlowAlign, and FlowEdit-OTC fail to make the swap. (d) *squirrel→rabbit*: our method and FlowAlign produce a clean substitution on the original fence, while RF-Inversion alters the background and SplitFlow changes the posture. Across rows, our method (second column) consistently achieves the intended edit with the highest background and identity fidelity.

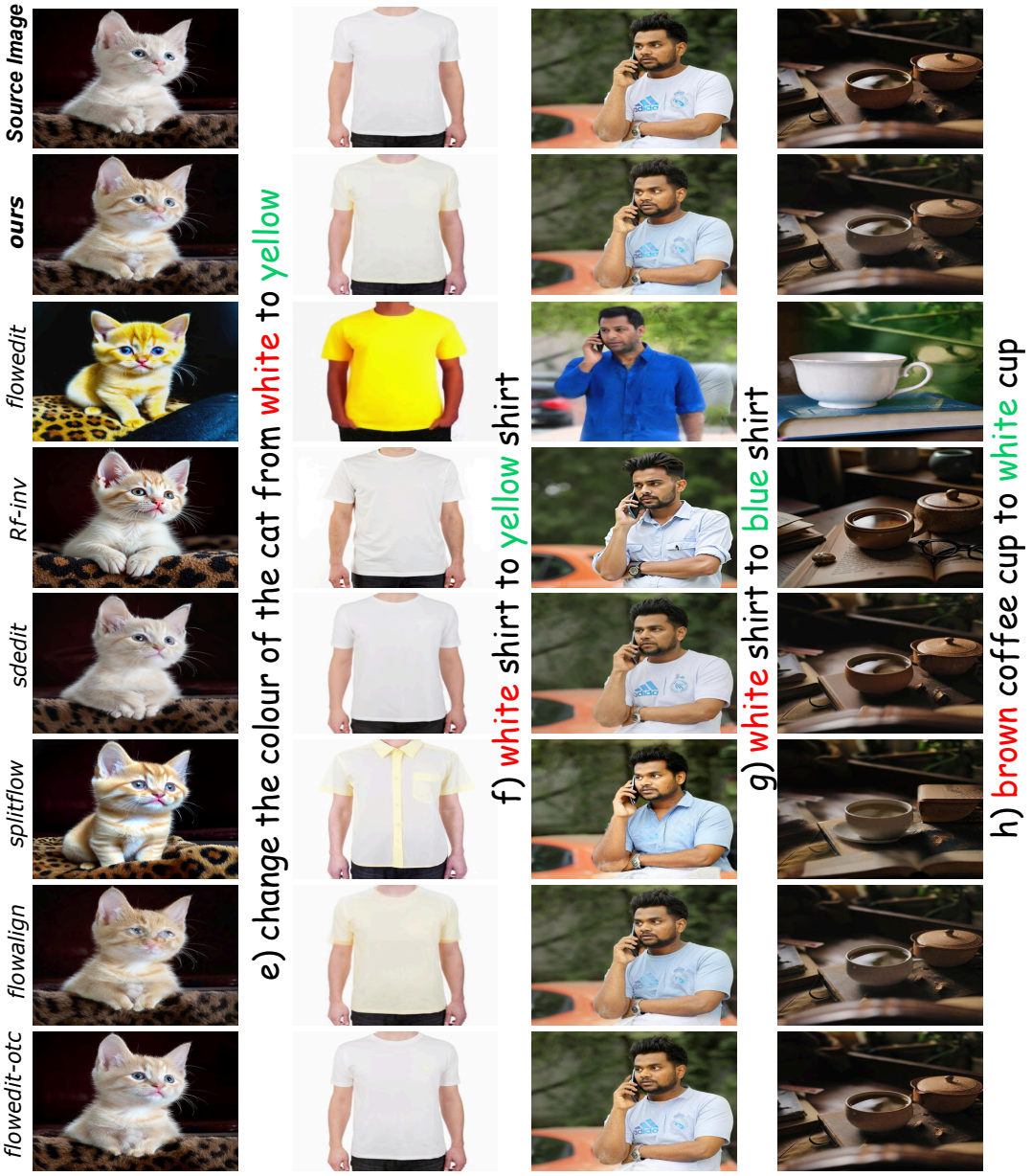


Figure M15: Qualitative comparison with baselines (Part 2). Layout matches Fig. M14. (e) *white cat→yellow cat*: our method tints the fur while preserving pose, markings, and the leopard-print background. FlowEdit over-saturates the whole scene and SDEdit barely tints. (f) *white shirt→yellow shirt*: our method applies a clean colour change without altering the person or the shirt's fabric/cut, whereas FlowEdit changes garment type and SplitFlow produces a collared shirt. (g) *white shirt→blue shirt*: our method recolours the shirt cleanly while preserving the phone, the Adidas logo, and the face. FlowEdit alters the entire appearance (including the face) and several baselines leave the shirt unchanged. (h) *brown coffee cup→white cup*: our method recolours only the cup, preserving the moody lighting and composition. FlowEdit replaces the entire scene. These examples highlight the difficulty of localised attribute edits: our method is uniquely successful at applying the requested change while leaving the rest of the image untouched.

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 clearly reflect our contribution. All contribution are validated in experiments (section 4, Tables 1–2, Figures 3–5), and supported by mathematical justifications (Appendix B). The scope is clearly limited to inversion free flow-based editing.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: Appendix section K provides a detailed discussion on limitations of our work.

   Guidelines:

   - The answer [N/A] 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 provide full proofs and formal mathematical justifications in Appendix section B.

Guidelines:

- The answer [N/A] 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: Appendix A provides a complete specification of all hyperparameters, architectural choices, source/target cloud construction procedures, Sinkhorn solver settings, co-state stabilisation details, and hardware details. Algorithm 1 provides the complete pseudocode.

Guidelines:

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

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

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

   Answer: [No]

   Justification: Code will be released upon acceptance. The evaluation benchmark PIE-Bench is publicly available [39]. All implementation details sufficient for independent re-implementation are provided in Appendix A, and the complete pseudocode is given in Algorithm 1.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: The method is training-free. All inference-time hyperparameters are specified in Appendix A. We follow a standard evaluation protocol consistent with existing baselines [14, 24].

   Guidelines:

   - The answer [N/A] 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: All trajectory analysis plots report mean $\pm$ standard deviation over all 700 PIE-Bench images, capturing variability across images. Table results are means over the full 700-image benchmark, which is the standard reporting convention for all baselines in this domain.

   Guidelines:

   - The answer [N/A] 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: Appendix A specifies the hardware, peak memory usage , numerical precision for our method. Appendix J provides per-method runtime comparisons (Table J12) with breakdowns into ODE computation time and total time.

   Guidelines:

   - The answer [N/A] 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 uses publicly available models and benchmarks. The human user study (Appendix H) involved voluntary participants with no deception, randomised presentation, and no personally identifiable information collected.

   Guidelines:

   - The answer [N/A] 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: Appendix K discusses inherited limitations from the pretrained backbone, which includes the potential for generating biased or harmful content. The method improves editing fidelity, which has positive applications.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release new pretrained models or datasets. The method is a training-free inference-time algorithm applied to existing publicly available models. No new generative capabilities are introduced beyond what the backbone models already provide.

Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: All models and datasets are cited: Stable Diffusion 3 Medium [4], FLUX-dev [5], CLIP ViT-L/14 [43], PIE-Bench [39], and DAVIS [44]. SD3-Medium is released

under a Stability AI Community License. FLUX-dev is released under the FLUX.1-dev Non-Commercial License. CLIP is released under the MIT License. PIE-Bench is publicly available for research use.

Guidelines:

- The answer [N/A] 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: [N/A]

Justification: The paper does not introduce new datasets, pretrained models, or other assets.

Guidelines:

- The answer [N/A] 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: [Yes]

Justification: Appendix H describes the user study protocol in detail: 118 participants, 100 randomly sampled images. The two evaluation criteria (edit compliance and pairwise preference) are described with the exact questions and answer options presented to participants. Participation was entirely voluntary.

Guidelines:

- The answer [N/A] 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: [N/A]

    Justification: The user study involves viewing edited images and selecting preferences, posing no risk to participants. No personally identifiable information was collected.

    Guidelines:

    - The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: No LLM is used as a component of the core method. The method operates entirely through flow models, CLIP encoders, and the Sinkhorn algorithm, without any language model in the pipeline. LLM is used only to improve the readability of the paper text.

    Guidelines:

    - The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.