TWINEDIT: ONE-STEP IMAGE EDITING AS SINGLE-STATE LATENT TRANSPORT

Anonymous authors
Paper under double-blind review

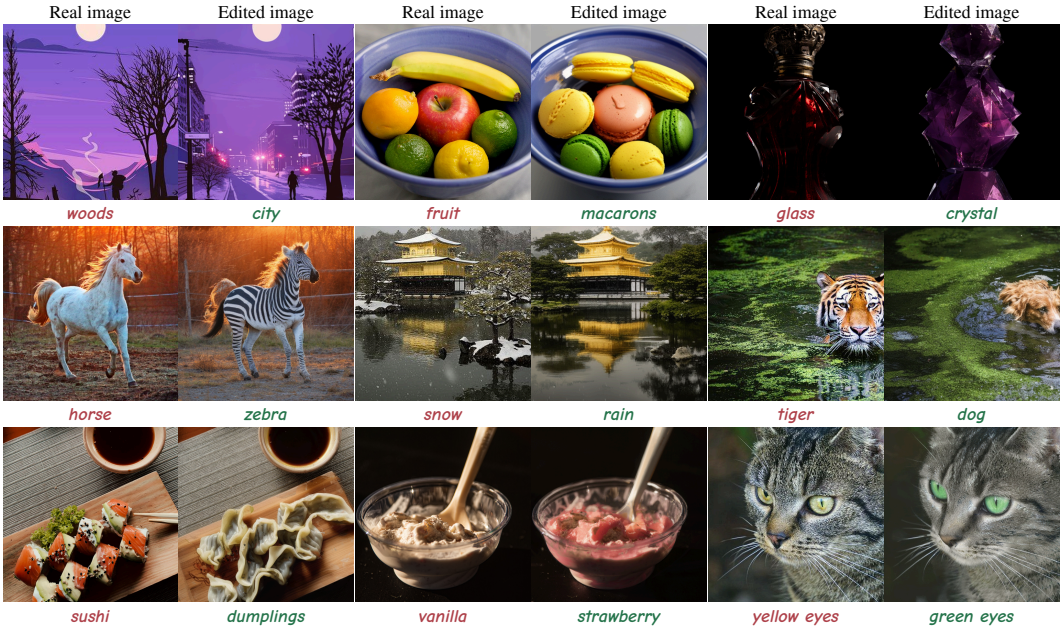

Figure 1: TwinEdit. Our training-free and inversion-free method operates on fast one-step generative models. TwinEdit separates path estimation from state transport. Paired source- and target-conditioned responses estimate the editing path, while the encoded source remains the single transported state through target-conditioned denoising. This source-anchored transport enables precise edits while preserving unrelated image content. Results use SD-Turbo in the top two rows and SwiftBrush-v2 in the bottom row. Labels indicate the desired semantic change.

ABSTRACT

One-step text-to-image models have brought image synthesis to real-time speeds, creating a natural foundation for equally fast image editing. Turning this speed into reliable editing remains challenging because a single update must introduce substantial target changes while retaining the structure and content of the source image. Recent one-step methods address this challenge through low-energy transport, post-hoc candidate selection, or localized adaptive editing. However, steering the edit and representing the edit are different problems. To address this gap, we propose *TwinEdit*, which formulates one-step editing as the transport of a single source-anchored latent state. Source- and target-conditioned latent responses estimate a target-directed path, while only the encoded source is transported along it. The transported state is refined by one target-conditioned denoising step and decoded into a target image. The final edit is then formed along the displacement from the source to this decoded target. Experiments on PIE-Bench show that TwinEdit produces clean, source-faithful edits with strong target alignment while retaining one-step inference efficiency and the lowest measured GPU-memory use in our comparison.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

Text-guided image editing aims to modify a source image according to a target description while preserving content unrelated to the requested change. Existing image editing methods use stochastic perturbation, inversion, attention control, feature reuse, or instruction-conditioned editing (Meng et al., 2022; Mokady et al., 2023; Huberman-Spiegelglas et al., 2024; Wallace et al., 2023; Hertz et al., 2023; Tumanyan et al., 2023; Cao et al., 2023; Brooks et al., 2023). Recent advances in text-to-image generation have substantially reduced generation latency while maintaining high image quality (Sauer et al., 2024; Yin et al., 2024; Dao et al., 2024; Liu et al., 2024; Luo et al., 2023; Lin et al., 2024).

Extending this efficiency to image editing remains challenging because a single update must introduce the target change while preserving the content and structure of the source image. Multi-step diffusion methods can gradually introduce and refine an edit over a denoising trajectory (Ho et al., 2020; Song et al., 2021). A one-step editor must resolve these requirements in a single update. How that update is represented therefore matters alongside its direction and magnitude. In particular, estimating a useful source-to-target displacement does not determine the latent state from which that displacement should be applied. With no long trajectory to correct an unfavorable intermediate state, this choice can directly affect the visible edit.

Recent work has explored few-step, one-step, inversion-free, and transport-based approaches to fast image editing (Deutch et al., 2024; Gong et al., 2025; Nguyen et al., 2025). FlowEdit (Kulikov et al., 2025) constructs a direct ODE between source and target distributions without explicit inversion, while FlowAlign (Kim et al., 2026) regularizes this trajectory to improve source consistency. ChordEdit (Lu et al., 2026) stabilizes one-step transport with a low-energy control field. RRLS (Yi et al., 2026) searches along a residual path to adapt editing strength, and WhereEdit (Hu et al., 2026) introduces spatial control for localized edits. These methods improve the trajectory, magnitude, or spatial support of the target-directed update.

These approaches mainly refine how the target-directed update is constructed or controlled. A different question is which latent state carries that update. Source- and target-conditioned latent responses can provide a useful estimate of the desired change. However, the same latent displacement need not produce the same visible change when applied from different latent states, because the decoder is nonlinear. Thus, a response can be informative about how the edit should move without necessarily being the state from which that edit should be realized. In a one-step editor, this representation choice cannot be corrected over a long denoising trajectory.

This distinction separates path estimation from state transport. Multiple latent responses are used to estimate the target-directed path, while the encoded source remains the single state transported along it. The transported state is then refined toward the target before decoding, and its decoded change provides the basis for final realization. We therefore formulate one-step image editing as the transport of a single latent state. TwinEdit uses paired source- and target-conditioned responses under opposite noise samples to estimate a target-directed update, applies that update to the encoded source, and refines the resulting state with one target-conditioned denoising step. The decoded target defines the appearance of the intended change, while an endpoint-derived spatial reference controls where and how strongly that change is applied relative to the source. We evaluate this separation directly. When the path construction and downstream stages are held fixed, the encoded-source carrier improves preservation before image-space realization, and this advantage remains at closely matched CLIP alignment. The same parameter setting also transfers to SwiftBrush-v2 without backbone-specific retuning.

2 BACKGROUND

2.1 ONE-STEP IMAGE EDITING

Suppose we are given a source image I_s together with source and target text conditions c_s and c_t. Let E and D denote the image encoder and decoder, respectively, and write $x = E(I_s)$ for the encoded source. At model time τ, let $x_\tau(\epsilon) = \alpha_\tau x + \sigma_\tau \epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$. With backbone noise

prediction $\hat{\epsilon}_\theta$, we define

$$F_\tau(x, c; \epsilon) = \Pi_\tau\left(\frac{x_\tau(\epsilon) - \sigma_\tau \hat{\epsilon}_\theta(x_\tau(\epsilon), c, \tau)}{\alpha_\tau}\right), \tag{1}$$

where Π_τ is scheduler endpoint postprocessing when enabled (Rombach et al., 2022; Ho et al., 2020; Song et al., 2021). We set $s_\epsilon = F_\tau(x, c_s; \epsilon)$, $t_\epsilon = F_\tau(x, c_t; \epsilon)$, and $d_\epsilon = t_\epsilon - s_\epsilon$.

In the one-step setting, these responses define the latent states from which the edit is constructed. Their geometry therefore affects both the target displacement and how source information is retained before decoding.

2.2 Problem Formulation

Consider source-conditioned and target-conditioned latent states s and t, and let $d = t - s$ denote their latent displacement. The visible effect of d can depend on the latent state at which it is applied. Between s and t, the corresponding image-space displacement is $D(t) - D(s)$, whereas applying the same displacement at the source anchor x gives $D(x + d) - D(x)$. Their discrepancy is:

$$\Gamma = \big[D(t) - D(s)\big] - \big[D(x + d) - D(x)\big]. \tag{2}$$

For an affine decoder, $\Gamma = 0$, but for a nonlinear decoder it need not be zero. The same latent displacement can therefore induce different visible changes when applied at different latent states. The response-derived displacement can guide how the edit should move, but it does not determine the latent state from which that change should be realized. This motivates applying the target-directed transport to the encoded source in latent space before decoding. Appendix B gives the corresponding integral identity and Jacobian-variation bound.

Let $\mathcal{Z}$ denote the latent space of the one-step model. A source-anchored transport is represented by a path $\gamma : [0, 1] \to \mathcal{Z}$ with $\gamma(0) = x$. The path carries the encoded source toward a target-conditioned latent state before decoding. This formulation separates path estimation from state transport. Multiple latent responses may be used to estimate a target-directed path, while a single source-anchored latent state is transported along it before decoding.

3 TwinEdit

TwinEdit represents the edit by transporting the encoded source into target-conditioned denoising, while the endpoint responses estimate the path and construct the realization reference. Figure 2 compares TwinEdit with one-step simple drift and temporal field estimation. One target-conditioned denoising step then refines the transported state into a denoised target latent. After decoding, the final stage controls how much of the resulting source-to-target change appears in the output.

3.1 Source-Anchored Transport

The target-directed path and the state that carries the edit have different roles. Source- and target-conditioned responses provide information about how the edit should move, but they do not by themselves determine which latent state should carry that change. We therefore keep the encoded source as the transported state and use the responses only to estimate its path. Because each response also depends on the sampled noise, we evaluate the same source-to-target transition with opposite noise samples $(\nu, -\nu)$ (Hammersley & Morton, 1956; Jia et al., 2026), where $\nu \sim \mathcal{N}(0, I)$. At a fixed model time τ_e, the four responses are evaluated in one batch:

$$s_+ = F_{\tau_e}(x, c_s; \nu), \quad t_+ = F_{\tau_e}(x, c_t; \nu), \qquad s_- = F_{\tau_e}(x, c_s; -\nu), \quad t_- = F_{\tau_e}(x, c_t; -\nu). \tag{3}$$

Their source-to-target displacements are $d_+ = t_+ - s_+$ and $d_- = t_- - s_-$. Both describe the same transition under opposite noise samples, so we average them as $\bar{d} = (d_+ + d_-)/2$.

Their mean is the branch-symmetric component of the two responses, but partial cancellation can shrink the transition magnitude. When their inner product is positive, we rescale $\bar{d}$ so that its squared norm matches their mean squared norm. If the inner product is negative, we set it to zero so that

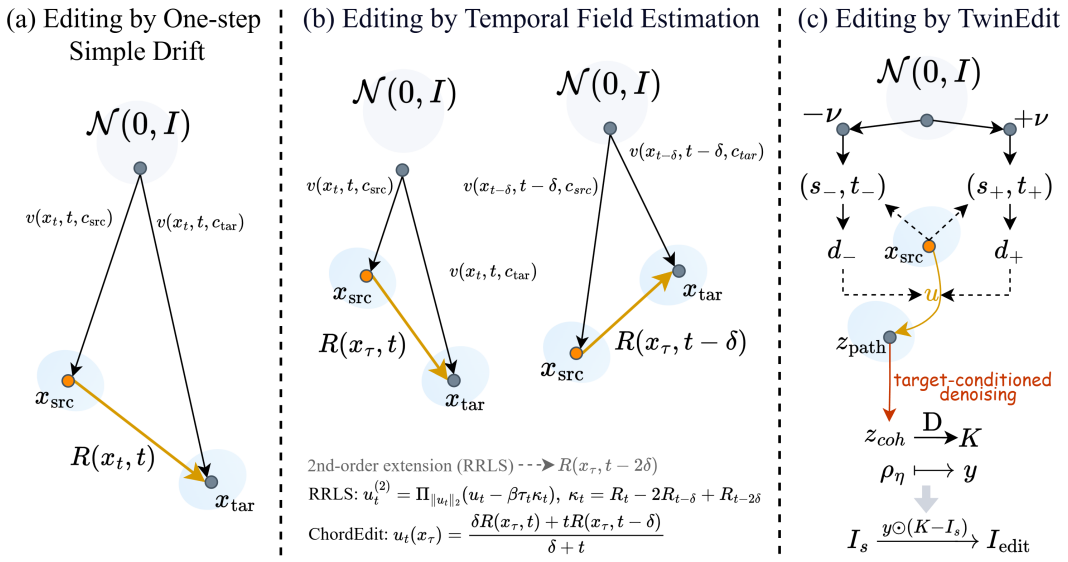

Figure 2: Comparison of one-step editing constructions. (a) simple drift uses source- and target-conditioned responses at a single noisy state to form a direct update. (b) ChordEdit (Lu et al., 2026) estimates an editing field from observable responses at neighboring times, while RRLS (Yi et al., 2026) further corrects this estimate with a second-order temporal term. (c) TwinEdit estimates a source-anchored transport from source/target responses under opposite noise samples, producing the transported latent z_{path}. Target-conditioned denoising yields the denoised target latent z_{coh}, decoded as target image K; y sets the local fraction of the source-to-K displacement.

disagreement cannot cause a larger rescaling. For nonzero responses, this gives:

$$\kappa = \sqrt{\frac{2\left(\|d_+\|_2^2 + \|d_-\|_2^2\right)}{\|d_+\|_2^2 + \|d_-\|_2^2 + 2\max(\langle d_+, d_-\rangle, 0)}}. \tag{4}$$

To retain part of the damping introduced by symmetric averaging, we apply only a fixed fraction of this correction using $s_\lambda = 1 + \lambda(\kappa - 1)$, with $\lambda \in [0, 1]$. The resulting transport state is:

$$u = s_\lambda \bar{d}, \qquad z_{\text{path}} = x + u. \tag{5}$$

Swapping the two perturbation branches leaves u unchanged, so the transport does not depend on the assignment of the two noise signs. Appendix C derives this invariance together with the magnitude-completion bounds. The transport is applied directly to the encoded source, and the edit therefore enters the next stage as the single latent state z_{path}. The complementary responses serve only to estimate its transport path.

3.2 Target-Conditioned Denoising

The transport state z_{path} carries the estimated target-directed change, but it is not yet the denoised target latent that will be decoded. We therefore refine it with one target-conditioned denoising step. For the auxiliary TwinEdit-P variant, we remove from the Gaussian noise its component parallel to u; TwinEdit-E leaves the noise unchanged.

Since u varies spatially, the projection in TwinEdit-P is applied independently at each latent token. For a token ℓ with nonzero transport direction $u_\ell \in \mathbb{R}^C$ and noise $\epsilon_\ell \sim \mathcal{N}(0, I)$, we decompose:

$$\epsilon_\ell^{\parallel} = u_\ell \frac{\langle \epsilon_\ell, u_\ell \rangle}{\|u_\ell\|_2^2}, \qquad \epsilon_\ell^{\perp} = \epsilon_\ell - \epsilon_\ell^{\parallel}. \tag{6}$$

Projecting out one direction reduces the expected noise energy. To preserve the noise scale expected by the pretrained model, we use:

$$\tilde{\epsilon}_\ell = \sqrt{\frac{C}{C-1}}\,\epsilon_\ell^{\perp}, \tag{7}$$

Algorithm 1 TwinEdit-E/P inference

Input: source image I_s, prompts (c_s, c_t), fixed one-step model, variant $o \in \{E, P\}$, $\lambda, \tau_e, \tau_c, \eta$

$x \leftarrow E(I_s)$; sample $\nu, \epsilon \sim \mathcal{N}(0, I)$

$s_\pm \leftarrow F_{\tau_e}(x, c_s; \pm\nu), \quad t_\pm \leftarrow F_{\tau_e}(x, c_t; \pm\nu), \quad d_\pm \leftarrow t_\pm - s_\pm$

$\bar{d} \leftarrow (d_+ + d_-)/2, \quad \kappa \leftarrow$ Eq. (4), $\quad u \leftarrow [1 + \lambda(\kappa - 1)]\bar{d}, \quad z_{\text{path}} \leftarrow x + u$

$(J, r_0) \leftarrow \textsc{EndpointReference}(I_s, x, s_\pm, t_\pm, u)$

choose $\tilde{\epsilon} \leftarrow \epsilon$ for E and Eq. (7) for P

$z_{\tau_c} \leftarrow \alpha_{\tau_c} z_{\text{path}} + \sigma_{\tau_c} \tilde{\epsilon}, \quad \hat{\epsilon}_t \leftarrow \epsilon_\theta(z_{\tau_c}, c_t, \tau_c)$

$z_{\text{coh}} \leftarrow (z_{\tau_c} - \sigma_{\tau_c} \hat{\epsilon}_t)/\alpha_{\tau_c}, \quad K \leftarrow D(z_{\text{coh}}), \quad \Delta \leftarrow K - I_s$

$a \leftarrow \textsc{TransferReference}(I_s, J, r_0, K, \eta), \quad \mathcal{U} \leftarrow \{i : a_i > 0\}$

solve $\{y_i\}_{i \in \mathcal{C}}$ by Eq. (12) for each connected component $\mathcal{C} \subseteq \mathcal{U}$

Output: $I_{\text{out}}(i) \leftarrow I_s(i) + y_i \Delta(i)$

where C is the latent channel dimension. The resulting noise satisfies $\langle \tilde{\epsilon}_\ell, u_\ell \rangle = 0$ and preserves the expected energy, $\mathbb{E}\|\tilde{\epsilon}_\ell\|_2^2 = \mathbb{E}\|\epsilon_\ell\|_2^2$. Appendix D gives the corresponding projector and energy derivation.

At denoising time τ_c, we form $z_{\tau_c} = \alpha_{\tau_c} z_{\text{path}} + \sigma_{\tau_c} \tilde{\epsilon}$, where α_{τ_c} and σ_{τ_c} are the signal and noise coefficients of the pretrained schedule. The target-conditioned prediction is $\hat{\epsilon}_t = \epsilon_\theta(z_{\tau_c}, c_t, \tau_c)$. We recover the denoised target latent z_{coh} and decode it to obtain K:

$$z_{\text{coh}} = \frac{z_{\tau_c} - \sigma_{\tau_c} \hat{\epsilon}_t}{\alpha_{\tau_c}}, \qquad K = D(z_{\text{coh}}). \tag{8}$$

This gives two versions of TwinEdit that differ only in the denoising noise. **TwinEdit-E** keeps the original Gaussian noise unchanged, $\tilde{\epsilon} = \epsilon$, while **TwinEdit-P** uses the projected noise in Eq. (7). Both versions share the same transport path, denoising time, realization rule, model parameters, and number of model calls.

3.3 Spatially Coordinated Realization

Forming a clean target and producing the final edit have different roles. The decoded target image K defines a single source-to-target change, while the final stage controls how much of that change is applied at each location. We write $\Delta(i) = K(i) - I_s(i)$ for this displacement. The endpoint reference sets the local amount of this change. Let J denote the fixed endpoint-derived reference proposal retained before target-conditioned denoising, and define $\Delta_{\text{ref}}(i) = J(i) - I_s(i)$. Let $r_0(i)$ denote the final endpoint radius. For a nonzero reference displacement, we convert this radius into a fraction of $\Delta_{\text{ref}}(i)$:

$$\rho_0(i) = \min\left(1, \frac{r_0(i)}{\|\Delta_{\text{ref}}(i)\|_2}\right). \tag{9}$$

Locations with vanishing reference displacement use $\rho_0(i) = 0$. A fixed global factor $\eta > 0$ controls overall edit strength while preserving the spatial pattern of the reference fraction. We use $\rho_\eta(i) = \min(1, \eta\rho_0(i))$ as the initial realization field. The field ρ_η gives the local fraction of the source-to-target displacement to be applied. Let $a_i = \rho_\eta(i)$ and $\mathcal{U} = \{i : a_i > 0\}$. Locations with $a_i = 0$ remain fixed at the source, while values with $a_i = 1$ may decrease during the following adjustment rather than being fixed at the target. We therefore adjust the coefficients jointly within each four-connected component of $\mathcal{U}$, subject to $0 \le y_i \le 1$.

The initial field may vary abruptly between neighboring locations, especially where the denoised displacement is large. We therefore regularize the realization field with the displacement-weighted energy:

$$\mathcal{E}(y) = \frac{1}{2} \sum_{(i,j) \in \mathcal{N}} w_{ij}(y_i - y_j)^2, \tag{10}$$

where $\mathcal{N}$ denotes neighboring pixel pairs and $w_{ij} = \|(\Delta_i + \Delta_j)/2\|_2^2$. This weighting penalizes coefficient discontinuities more strongly where they would induce larger changes in the decoded image. This objective is a weighted graph-Laplacian quadratic energy, and the update below uses a separable majorizing surrogate with standard constrained-optimality conditions (Chung, 1997;

Lange et al., 2000; Boyd & Vandenberghe, 2004). Let $e_i = \|\Delta_i\|_2^2$. For a component $\mathcal{C}$, define $T_{\mathcal{C}} = \sum_{i\in\mathcal{C}} e_i a_i$, the displacement-weighted amount of edit assigned to that component. We keep $T_{\mathcal{C}}$ fixed while allowing its spatial distribution to change. Let $g = \nabla\mathcal{E}(a)$ and $b_i = 2\sum_{j:(i,j)\in\mathcal{N}} w_{ij}$. We solve directly for the realized coefficients of each component:

$$\begin{aligned} \min_{\{y_i\}_{i\in\mathcal{C}}} \quad & \sum_{i\in\mathcal{C}} g_i(y_i - a_i) + \frac{1}{2}\sum_{i\in\mathcal{C}} b_i(y_i - a_i)^2 \\ \text{s.t.} \quad & \sum_{i\in\mathcal{C}} e_i y_i = T_{\mathcal{C}}, \qquad 0 \le y_i \le 1. \end{aligned} \tag{11}$$

The diagonal curvature b_i provides a local quadratic upper model of the weighted field energy, and the solution can be expressed using one scalar multiplier for each component:

$$y_i(\mu_{\mathcal{C}}) = \mathrm{clip}\left(a_i - \frac{g_i + \mu_{\mathcal{C}} e_i}{b_i}, 0, 1\right), \tag{12}$$

where $\mu_{\mathcal{C}}$ is chosen so that $\sum_{i\in\mathcal{C}} e_i y_i = T_{\mathcal{C}}$. The final edited image is:

$$I_{\mathrm{out}}(i) = I_s(i) + y_i \Delta(i). \tag{13}$$

The resulting field preserves the displacement-weighted edit amount of each connected region while remaining on the single displacement from I_s to K. Its coefficients stay in $[0, 1]$, and the weighted realization energy does not increase. Appendix F derives the majorized coordinate update and proves this energy-nonincreasing property.

Table 1: Quantitative comparison on PIE-Bench (Ju et al., 2024). Structure, MSE, and LPIPS are scaled by 10^3, and SSIM by 10^2. Superscripts T and I denote training-free and inversion-free methods, respectively. First-, second-, and third-ranked values are marked with yellow, pink, and blue backgrounds. † follows the published 100-NFE convention for the 50-step inversion/editing baselines.

Method	Structure↓	PSNR↑	MSE↓	SSIM↑	LPIPS↓	Whole↑	Edited↑	NFE↓	Runtime↓	VRAM↓
Multi-step ($\geq$ 30 steps)										
DDIM + MasaCtrl^{T}	28.33	22.17	8.71	79.77	101.93	24.81	21.38	100†	34.947	11682.95
DirectInv + MasaCtrl^{T}	24.66	22.65	8.12	81.47	81.95	25.28	21.62	100†	46.788	11689.40
DDIM + PnP^{T}	28.03	22.30	8.35	79.22	110.78	25.47	22.77	100†	21.815	7422.03
DirectInv + PnP^{T}	24.11	22.44	8.05	79.88	104.41	25.50	22.88	100†	21.874	7423.78
FlowEdit$^{\mathrm{T,I}}$ (SD3)	24.42	23.86	8.14	84.21	84.21	26.47	22.95	33	7.624	17392.63
Few-step (4 steps)										
TurboEdit$^{\mathrm{T,I}}$ (SDXL-Turbo)	29.55	22.50	9.32	79.77	102.53	25.56	23.08	4	3.712	19584.84
InstantEdit^{T} (PeRFlow-SD1.5)	11.71	24.11	2.50	85.34	60.87	25.55	22.12	8	1.349	8173.05
One-step										
SwiftEdit (SwiftBrush-v2)	20.19	23.57	6.26	81.05	81.92	25.40	22.37	2	1.079	14431.83
ChordEdit$^{\mathrm{T,I}}$ (SD-Turbo)	29.55	22.63	7.99	76.19	122.64	24.82	22.13	2	0.524	5978.08
RRLS$^{\mathrm{T,I}}$ (SD-Turbo)	26.25	23.91	6.25	81.28	112.58	25.38	22.13	4	1.262	7610.61
WhereEdit$^{\mathrm{T,I}}$ (SD-Turbo)	37.32	23.06	9.65	78.28	125.63	25.22	22.65	2	0.648	5978.31
TwinEdit-E$^{\mathrm{T,I}}$ (SD-Turbo)	29.05	22.81	7.91	78.81	118.09	25.77	22.49	2	0.991	5497.97
TwinEdit-P$^{\mathrm{T,I}}$ (SD-Turbo)	28.44	22.84	7.80	78.97	117.27	25.62	22.34	2	0.986	5497.97
TwinEdit-E$^{\mathrm{T,I}}$ (SwiftBrush-v2)	22.48	24.72	5.37	84.24	93.43	24.50	21.17	2	0.973	5505.61
TwinEdit-P$^{\mathrm{T,I}}$ (SwiftBrush-v2)	22.96	24.72	5.35	84.38	93.79	24.33	20.91	2	0.971	5505.61

4 Experiments

4.1 Experimental Setup

Dataset and evaluation metrics. We evaluate on PIE-Bench, which contains 700 image edits (Ju et al., 2024). Following the PIE-Bench protocol, we report Structure Distance based on DINO self-similarity (Caron et al., 2021); PSNR, MSE, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) on the unedited region; and CLIP similarity (Radford et al., 2021) on the whole image and edited region.

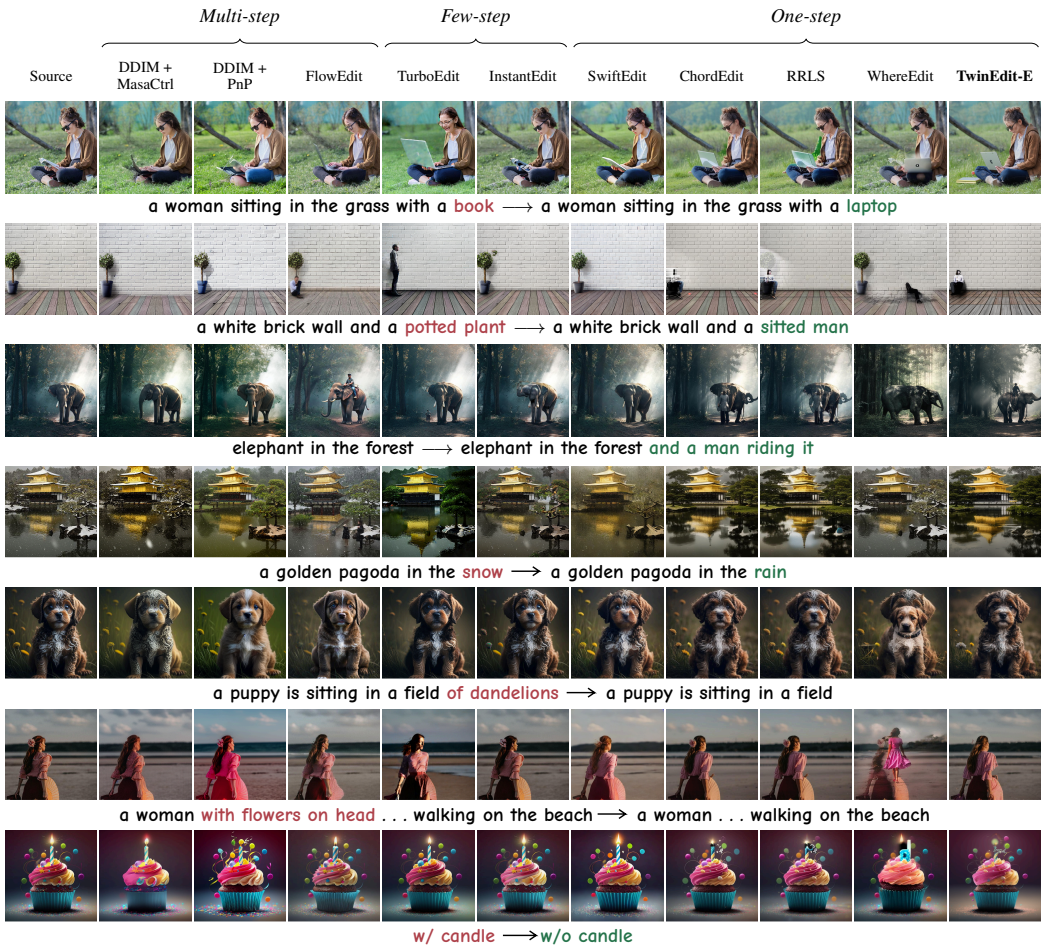

Figure 3: Comparison of edited results on PIE-Bench. Source images appear in the first column. Source and target changes are marked below each row in red and green, respectively. The final column shows TwinEdit-E. The first six rows use SD-Turbo and the last row uses SwiftBrush-v2.

Implementation details. All experiments are conducted on a single NVIDIA GeForce RTX 3090 24GB GPU. We evaluate TwinEdit with SD-Turbo (Sauer et al., 2024) and SwiftBrush-v2 (Dao et al., 2024) using $\lambda = 0.75$, endpoint time $\tau_e = 0.75$, denoising time $\tau_c = 0.45$, and realization scale $\eta = 2.2$. All main TwinEdit results use seed 42, as do the baselines evaluated using their released implementations, except FlowEdit, which follows its released seed 123. The same TwinEdit parameter setting is used for every image and both backbones. TwinEdit-E is the default Gaussian-noise configuration; TwinEdit-P is a fixed projected-noise variant reported separately. The four endpoint responses are evaluated in parallel in one model call, followed by one target-conditioned denoising call. The measured runtime protocol is reported in Appendix K.

4.2 Comparison with Prior Methods

Quantitative comparison. Table 1 compares TwinEdit with multi-step, few-step, and one-step editors on PIE-Bench. Both TwinEdit variants have better mean values than ChordEdit on all seven quality metrics. Compared with WhereEdit, TwinEdit has lower Structure Distance, MSE, and LPIPS and higher SSIM and Whole CLIP, while WhereEdit has higher PSNR and Edited CLIP. RRLS favors source preservation, whereas TwinEdit has higher Whole and Edited CLIP and runs faster (0.99 vs. 1.26 s). Across the five evaluated RRLS settings, the nearest measured TwinEdit points in Edited CLIP improve Structure, PSNR, SSIM, and LPIPS in all five comparisons and MSE in four. At the nearest measured LPIPS points, TwinEdit improves both Whole and Edited CLIP for all five settings (Appendix H.2). TwinEdit also has the lowest VRAM usage in Table 1, at about 5,500

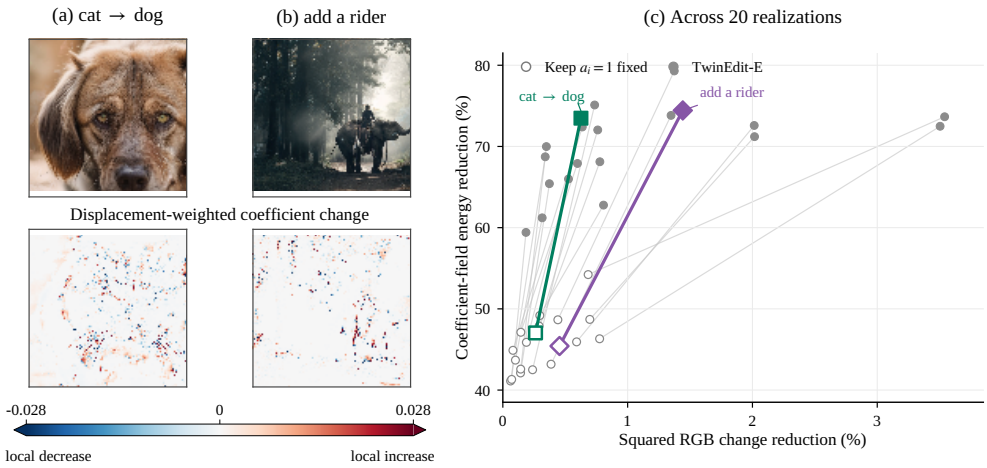

Figure 4: Fixed-K coefficient redistribution. Panels (a) and (b) show TwinEdit-E outputs and displacement-weighted coefficient changes $e_i(y_i - a_{\text{lock},i})$ relative to the control that keeps coefficients with $a_i = 1$ fixed. Blue and red indicate local decreases and increases, and both maps use the same symmetric color scale. Panel (c) connects the control and TwinEdit-E for each of 20 realizations. The coordinated solution reduces the weighted coefficient-field energy while leaving the pre-quantization squared RGB change nearly unchanged.

MiB. With SwiftBrush-v2, TwinEdit improves PSNR and MSE over SwiftEdit, while SwiftEdit has lower LPIPS and higher Whole and Edited CLIP. TwinEdit with SwiftBrush-v2 has lower Structure Distance and MSE and higher PSNR and SSIM than every multi-step baseline. It improves all five source-preservation metrics over TurboEdit and reaches the highest PSNR in Table 1, above InstantEdit. TwinEdit-E with SD-Turbo has the second-highest Whole CLIP.

Qualitative comparison. Figure 3 presents representative object replacement, object addition, weather-change, removal, and attribute-editing examples. The first six rows use SD-Turbo, the final candle row uses SwiftBrush-v2, and TwinEdit-E is shown throughout. In a blind pairwise study, TwinEdit-E receives the largest preference share in all five comparisons for both semantic alignment and preservation quality (Appendix M).

Table 2: Fixed-K realization with TwinEdit-E on PIE-Bench. The target K already includes source-anchored transport and target-conditioned denoising; only its image-space realization changes.

Realization	Structure↓	PSNR↑	MSE↓	SSIM↑	LPIPS↓	Whole↑	Edited↑
Direct target K_E	0.03666	21.2902	0.010465	0.73381	0.14588	**25.9761**	**22.7191**
Initial field	**0.02887**	22.7359	0.008019	0.78363	0.12166	25.7778	22.4854
Keep $a_i = 1$ fixed	0.02893	22.7645	0.007983	0.78645	0.11871	25.7646	22.4904
TwinEdit-E	0.02905	**22.8057**	**0.007908**	**0.78808**	**0.11809**	25.7662	22.4897

Table 3: Transported-state comparison at default $\eta = 2.2$. Entries are encoded source minus mean response; Appendix H.1 gives nearest measured comparisons across realization scales.

Output stage	Δ Structure↓	Δ PSNR↑	Δ LPIPS↓	Δ Whole↑	Δ Edited↑
Direct K	-0.04818	+3.261	-0.08384	-0.662	-0.455
Initial field	-0.02468	+2.814	-0.05909	-0.143	-0.211
Final output	-0.02556	+2.810	-0.05522	-0.158	-0.237

4.3 Fixed-K Realization Analysis

Table 2 fixes the source image, endpoint reference, and decoded target K, which already includes source-anchored transport and target-conditioned denoising. Directly decoding K favors target alignment, while the initial realization field applies the same target more conservatively and recovers source preservation. Final coordination yields small additional improvements in PSNR, MSE, SSIM,

and LPIPS, alongside a slight increase in Structure Distance and a slight decrease in Whole CLIP; Edited CLIP remains unchanged within uncertainty (Appendix H.1).

Figure 4 visualizes the final coordination step. Across 20 realizations, the weighted coefficient-field energy decreases by 59.4–79.3%, while the pre-quantization squared RGB change decreases by only 0.19–3.54%. The optimization therefore redistributes the realization coefficients rather than reducing the edit uniformly. Appendix F gives the corresponding derivation.

4.4 Ablation Study

We examine the transported state, path construction, denoising noise, and the four hyperparameters η, λ, τ_c, and τ_e.

Transported state. Table 3 compares the encoded source x with the mean source response $\bar{s} = (s_+ + s_-)/2$ at the default $\eta = 2.2$. At the directly decoded target K, x improves all five preservation metrics (Structure -0.0482, PSNR $+3.26$ dB, and LPIPS -0.0838); these gaps exceed the gains from the initial realization field and persist at the final output. Varying η gives two close Edited-CLIP pairs (gaps 0.0053 and 0.0082), where x again improves all five preservation metrics. The closest Whole-CLIP pairs agree (Appendix H.1). Thus source anchoring yields a more favorable trade-off within the overlapping measured alignment range.

Path construction. We separate the magnitude correction in Eqs. (4)–(5) from the target-conditioned denoising step in Section 3.2. Denoising raises Edited CLIP by 1.57–1.63, while magnitude correction adds 0.20–0.26; both trade source preservation for target alignment. Three independent two-sample estimates raise CLIP further but weaken all five preservation metrics, so the antithetic pair yields a more source-preserving trade-off.

Denoising noise. E and P use the same transport path and differ only in denoising noise. P removes the component parallel to the transport direction. It improves all five preservation metrics on SD-Turbo, whereas E has higher Whole and Edited CLIP; transfer to SwiftBrush-v2 is mixed.

Hyperparameter sweep. Figure 5 varies η, λ, τ_c, and τ_e one at a time, with the remaining parameters fixed at their default values. For reference, we include the five evaluated RRLS λ_{src} settings; the filled square marks its reported default, while ChordEdit and WhereEdit are shown at their reported settings. Connecting lines are visual guides. Increasing η, λ, or τ_c trades preservation for alignment. The same trend holds for τ_e through 0.85, before Whole CLIP decreases at 0.95. At $\eta = 1.4$, TwinEdit has higher Whole CLIP and lower LPIPS than all five RRLS settings.

Endpoint reference. Using only the source-matched reference anchor weakens preservation, while a uniform response radius weakens preservation and alignment. We therefore retain the complete endpoint reference in all main experiments.

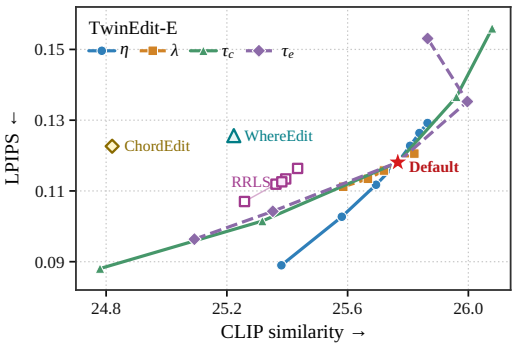

Figure 5: Hyperparameter sweeps for TwinEdit-E on PIE-Bench.

5 Conclusion

TwinEdit separates the path that guides an edit from the latent state that carries it. It keeps the encoded source as the transported state and uses source- and target-conditioned responses only to estimate its path. The transported state is refined once towards the target, and the decoded source-to-target change is then applied with spatially varying strength. Controlled transported-state comparisons show that source anchoring improves preservation before image-space realization and retains this advantage at closely matched CLIP alignment. The same construction and parameter setting transfer to SwiftBrush-v2.

Limitations. TwinEdit inherits limitations of its pretrained generator. Pixelwise source-to-target realization may produce ghosting or incomplete insertions under large geometric displacement or correspondence error (Appendix Figure 8). Two batched denoiser calls add latency.

AI USE STATEMENT

Generative AI tools were used for language polishing and critical feedback on the manuscript, experimental design, and interpretation of results.

REFERENCES

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023.

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 *ICCV*, 2023.

Mathilde Caron, Hugo Touvron, Ishan Misra, Herve Jegou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

Fan R. K. Chung. *Spectral Graph Theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, 1997.

Trung Dao, Thuan Hoang Nguyen, Thanh Le, Duc Vu, Khoi Nguyen, Cuong Pham, and Anh Tran. Swiftbrush v2: Make your one-step diffusion model better than its teacher. In *ECCV*, 2024.

Gilad Deutch, Rinon Gal, Daniel Garibi, Or Patashnik, and Daniel Cohen-Or. Turboedit: Text-based image editing using few-step diffusion models. In *SIGGRAPH Asia Conference Papers*, pp. 41:1–41:12, 2024. doi: 10.1145/3680528.3687612.

Yiming Gong, Zhen Zhu, and Minjia Zhang. Instantedit: Text-guided few-step image editing with piecewise rectified flow. In *ICCV*, pp. 16808–16817, 2025.

J. M. Hammersley and K. W. Morton. A new monte carlo technique: Antithetic variates. *Mathematical Proceedings of the Cambridge Philosophical Society*, 52(3):449–475, 1956. doi: 10.1017/S0305004100031455.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross-attention control. In *ICLR*, 2023.

Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms*. Society for Industrial and Applied Mathematics, 2nd edition, 2002. doi: 10.1137/1.9780898718027.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

Ming Hu, Mingyu Dou, Jianfu Yin, Miaomiao Zhang, Cong Hu, Yao Wang, Bingliang Hu, and Quan Wang. Whereedit: Mask-aware local latent editing for one-step image editing. *arXiv preprint arXiv:2607.20883*, 2026.

Inbar Huberman-Spiegelglas, Vladimir Kulikov, and Tomer Michaeli. An edit friendly ddpm noise space: Inversion and manipulations. In *CVPR*, 2024.

Jing Jia, Sifan Liu, Bowen Song, Wei Yuan, Liyue Shen, and Guanyang Wang. Antithetic noise in diffusion models. In *ICLR*, 2026.

Xuan Ju, Ailing Zeng, Yuxuan Bian, Shaoteng Liu, and Qiang Xu. PnP inversion: Boosting diffusion-based editing with 3 lines of code. In *ICLR*, 2024.

Jeongsol Kim, Yeobin Hong, Jonghyun Park, and Jong Chul Ye. Flowalign: Trajectory-regularized, inversion-free flow-based image editing. In *ICLR*, 2026.

Vladimir Kulikov, Matan Kleiner, Inbar Huberman-Spiegelglas, and Tomer Michaeli. Flowedit: Inversion-free text-based editing using pre-trained flow models. In *ICCV*, 2025.

Kenneth Lange, David R. Hunter, and Ilsoon Yang. Optimization transfer using surrogate objective functions. *Journal of Computational and Graphical Statistics*, 9(1):1–20, 2000. doi: 10.1080/10618600.2000.10474858.

Shanchuan Lin, Anran Wang, and Xiao Yang. SDXL-Lightning: Progressive adversarial diffusion distillation. *arXiv preprint arXiv:2402.13929*, 2024. doi: 10.48550/arXiv.2402.13929.

Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, and Qiang Liu. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. In *ICLR*, 2024.

Liangsi Lu, Xuhang Chen, Minzhe Guo, Shichu Li, Jingchao Wang, and Yang Shi. Chordedit: One-step low-energy transport for image editing. In *CVPR*, pp. 14398–14407, 2026.

Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023.

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 *ICLR*, 2022.

Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *CVPR*, 2023.

Trong-Tung Nguyen, Quang Nguyen, Khoi Nguyen, Anh Tran, and Cuong Pham. Swiftedit: Lightning fast text-guided image editing via one-step diffusion. In *CVPR*, pp. 21492–21501, 2025.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 2nd edition, 2006. doi: 10.1007/978-0-387-40065-5.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bjorn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *ECCV*, pp. 87–103, 2024. doi: 10.1007/978-3-031-73016-0_6.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.

Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *CVPR*, 2023.

Bram Wallace, Akash Gokul, and Nikhil Naik. EDICT: Exact diffusion inversion via coupled transformations. In *CVPR*, pp. 22532–22541, 2023. doi: 10.1109/CVPR52729.2023.02158.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004.

Hongzhu Yi, Zhongtian Luo, Tong Li, Yiyan Fan, and Jungang Xu. Bridging the manifold gap: Riemannian residual line search for one-step image editing. *arXiv preprint arXiv:2606.24844*, 2026.

Tianwei Yin, Michael Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T. Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *CVPR*, 2024.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

A Notation and Complete Construction

This appendix gives the complete computation used by TwinEdit and fixes the distinction between latent-space states, decoded RGB images, and the finite realization variables. Latent tensors are denoted by lower-case letters and belong to $\mathbb{R}^{C\times h\times w}$. RGB images are denoted by upper-case letters and belong to $[0,1]^{H\times W\times 3}$. The encoder and decoder are E and D. A latent token is indexed by ℓ, an RGB pixel by i, and a four-connected RGB edge by $(i,j)\in\mathcal{N}$.

The encoded source latent is:

$$x = E(I_s). \tag{A.1}$$

At endpoint time τ_e, the source and target conditions are queried under antithetic noise:

$$s_+ = F_{\tau_e}(x, c_s; \nu), \qquad t_+ = F_{\tau_e}(x, c_t; \nu), \tag{A.2}$$

$$s_- = F_{\tau_e}(x, c_s; -\nu), \qquad t_- = F_{\tau_e}(x, c_t; -\nu). \tag{A.3}$$

The four responses are evaluated in one batch. The branch displacements are:

$$d_+ = t_+ - s_+, \qquad d_- = t_- - s_-. \tag{A.4}$$

The transport update u is derived in Appendix C; TwinEdit then transports the encoded source:

$$z_{\mathrm{path}} = x + u. \tag{A.5}$$

At denoising time τ_c, one target-conditioned denoising invocation produces the denoised target latent z_{coh}, which is decoded as:

$$K = D(z_{\mathrm{coh}}). \tag{A.6}$$

The deterministic endpoint reference (J, r_0) is derived from the same endpoint responses by the construction in Appendix E. The reference is independent of K. It supplies an initial spatial fraction on the final source-to-K displacement. The constrained realization in Appendix F returns coefficients $y_i \in [0,1]$ and the final image:

$$I_{\mathrm{out}}(i) = I_s(i) + y_i\big(K(i) - I_s(i)\big). \tag{A.7}$$

Inference calls. The endpoint responses are evaluated together in one model call, followed by one target-conditioned denoising call. The normalized endpoint and denoising times correspond to scheduler indices 749 and 450, respectively.

Backbone response mapping. The endpoint query constructs $x_\tau(\epsilon) = \alpha_\tau x + \sigma_\tau \epsilon$ and evaluates the source and target conditions in one batch. If the U-Net returns $2C$ channels, only the first C channels are used for response reconstruction. For SD-Turbo, Π_τ is the identity. SwiftBrush-v2 applies the clipping or thresholding specified by its scheduler after reconstructing the endpoint latent. Both mappings return the responses $s_\pm, t_\pm$ used for path estimation, before target-conditioned denoising and image-space realization.

B Transported State and Decoder Anchor Dependence

The same latent displacement need not produce the same decoded change when applied from different latent states. For an anchor a and displacement u, define the decoded displacement operator:

$$\Phi_a(u) := D(a+u) - D(a). \tag{B.1}$$

For two anchors a and b, their anchor commutator is:

$$\Gamma_{a,b}(u) := \Phi_a(u) - \Phi_b(u). \tag{B.2}$$

Proposition B.1 (Integral form of anchor dependence)**.** *If D is continuously differentiable on the line segments $\{a+tu : t\in[0,1]\}$ and $\{b+tu : t\in[0,1]\}$, then:*

$$\Gamma_{a,b}(u) = \int_0^1 \left[J_D(a+tu) - J_D(b+tu)\right] u\, dt, \tag{B.3}$$

where J_D denotes the decoder Jacobian.

Proof. For any anchor a, the vector-valued fundamental theorem of calculus gives:

$$D(a+u) - D(a) = \int_0^1 \frac{d}{dt} D(a+tu)\, dt \tag{B.4}$$

$$= \int_0^1 J_D(a+tu)u\, dt. \tag{B.5}$$

Applying the same identity at b and subtracting the two expressions yields Eq. (B.3). □

Corollary B.2 (Affine-decoder special case)**.** *If $D(z) = Az + c$ is affine, then:*

$$\Gamma_{a,b}(u) = 0, \tag{B.6}$$

for every a, b, u.

Proof. An affine decoder has the constant Jacobian $J_D(z) = A$. The integrand in Eq. (B.3) is therefore zero. □

Proposition B.3 (Jacobian-variation bound)**.** *Assume the decoder Jacobian is L_D-Lipschitz in operator norm:*

$$\|J_D(z_1) - J_D(z_2)\|_{\mathrm{op}} \le L_D \|z_1 - z_2\|_2. \tag{B.7}$$

Then:

$$\|\Gamma_{a,b}(u)\|_2 \le L_D \|a-b\|_2 \|u\|_2. \tag{B.8}$$

Proof. Because $(a+tu) - (b+tu) = a - b$ for every t, Eq. (B.7) gives:

$$\|J_D(a+tu) - J_D(b+tu)\|_{\mathrm{op}} \le L_D \|a-b\|_2. \tag{B.9}$$

Using Eq. (B.3), submultiplicativity, and integration over an interval of unit length gives Eq. (B.8). □

A complementary first-order expansion makes the same dependence explicit. If J_D is L_D-Lipschitz, then:

$$\Phi_a(u) = J_D(a)u + R_a(u), \qquad \|R_a(u)\|_2 \le \frac{L_D}{2}\|u\|_2^2. \tag{B.10}$$

Subtracting the expansions at two anchors gives:

$$\Gamma_{a,b}(u) = [J_D(a) - J_D(b)]\, u + R_a(u) - R_b(u). \tag{B.11}$$

These identities establish anchor dependence of the decoded effect. Carrier quality is evaluated empirically in Section 4.4, where the encoded-source carrier yields a more favorable preservation–alignment trade-off than the evaluated response-derived carriers within their overlapping measured alignment range.

C Antithetic Path Construction

Define the even and odd components of the two branch displacements:

$$d_{\mathrm{e}} = \frac{d_+ + d_-}{2}, \qquad d_{\mathrm{o}} = \frac{d_+ - d_-}{2}. \tag{C.1}$$

The original branches can be recovered as:

$$d_+ = d_{\mathrm{e}} + d_{\mathrm{o}}, \qquad d_- = d_{\mathrm{e}} - d_{\mathrm{o}}. \tag{C.2}$$

Consequently:

$$\|d_+\|_2^2 = \|d_{\mathrm{e}}\|_2^2 + \|d_{\mathrm{o}}\|_2^2 + 2\langle d_{\mathrm{e}}, d_{\mathrm{o}}\rangle, \tag{C.3}$$

$$\|d_-\|_2^2 = \|d_{\mathrm{e}}\|_2^2 + \|d_{\mathrm{o}}\|_2^2 - 2\langle d_{\mathrm{e}}, d_{\mathrm{o}}\rangle. \tag{C.4}$$

Adding and subtracting these equations gives:

$$\frac{1}{2}\left(\|d_+\|_2^2 + \|d_-\|_2^2\right) = \|d_{\mathrm{e}}\|_2^2 + \|d_{\mathrm{o}}\|_2^2, \tag{C.5}$$

$$\langle d_+, d_-\rangle = \|d_{\mathrm{e}}\|_2^2 - \|d_{\mathrm{o}}\|_2^2. \tag{C.6}$$

TwinEdit uses $\bar{d} = d_{\mathrm{e}}$ as its unscaled path estimate.

Define the even- and odd-component energies as:

$$E_d = \|d_{\mathrm{e}}\|_2^2, \qquad O_d = \|d_{\mathrm{o}}\|_2^2. \tag{C.7}$$

Substituting Eqs. (C.5)–(C.6) into Eq. (4) yields:

$$\kappa^2 = \frac{2(E_d + O_d)}{E_d + O_d + \max(E_d - O_d, 0)}. \tag{C.8}$$

Proposition C.1 (Capped energy completion)**.** *For nonzero branch responses, the corrected even component satisfies:*

$$\|\kappa d_{\mathrm{e}}\|_2^2 = E_d + \min(E_d, O_d), \tag{C.9}$$

Moreover:

$$1 \le \kappa \le \sqrt{2}. \tag{C.10}$$

Proof. If $E_d \ge O_d$, then $\max(E_d - O_d, 0) = E_d - O_d$ and:

$$\kappa^2 = \frac{2(E_d + O_d)}{2E_d} = 1 + \frac{O_d}{E_d}. \tag{C.11}$$

Multiplying by E_d gives $\|\kappa d_{\mathrm{e}}\|_2^2 = E_d + O_d$. If $E_d < O_d$, the maximum term vanishes and $\kappa^2 = 2$, so $\|\kappa d_{\mathrm{e}}\|_2^2 = 2E_d$. The two cases combine as Eq. (C.9). In the first case, $O_d/E_d \in [0, 1]$; in the second, $\kappa^2 = 2$. Hence Eq. (C.10) holds. □

When $E_d \ge O_d$, Eq. (C.9) equals the mean branch energy in Eq. (C.5). When odd branch disagreement is larger, the recovered energy is capped at $2E_d$ rather than amplifying the disagreement-dominated component.

The partial correction used in the method is:

$$s_\lambda = 1 + \lambda(\kappa - 1) = (1 - \lambda) + \lambda\kappa, \qquad \lambda \in [0, 1], \tag{C.12}$$

The final transport update is:

$$u = s_\lambda d_{\mathrm{e}}. \tag{C.13}$$

Because $1 \le \kappa \le \sqrt{2}$:

$$1 \le s_\lambda \le \kappa \le \sqrt{2}, \tag{C.14}$$

which implies:

$$\|d_{\mathrm{e}}\|_2 \le \|u\|_2 \le \|\kappa d_{\mathrm{e}}\|_2. \tag{C.15}$$

For fixed responses, $\|u\|_2$ is nondecreasing in λ:

$$\frac{\partial \|u\|_2}{\partial \lambda} = (\kappa - 1)\|d_{\mathrm{e}}\|_2 \ge 0. \tag{C.16}$$

Proposition C.2 (Branch-swap invariance)**.** *Exchanging the two perturbation branches leaves the transport update unchanged.*

Proof. Under $(d_+, d_-) \mapsto (d_-, d_+)$, the even component d_{e}, the sum of squared norms, and the inner product $\langle d_+, d_-\rangle$ remain unchanged. Therefore κ, s_λ, and u remain unchanged. □

The cancellation of odd-order noise dependence can also be seen locally. If the conditional displacement $d(\nu)$ is sufficiently smooth near $\nu = 0$, then:

$$d(\nu) = d(0) + Dd(0)[\nu] + \frac{1}{2}D^2 d(0)[\nu, \nu] + \frac{1}{3!}D^3 d(0)[\nu, \nu, \nu] + \cdots, \tag{C.17}$$

$$d(-\nu) = d(0) - Dd(0)[\nu] + \frac{1}{2}D^2 d(0)[\nu, \nu] - \frac{1}{3!}D^3 d(0)[\nu, \nu, \nu] + \cdots. \tag{C.18}$$

Their mean is:

$$d_{\mathrm{e}} = d(0) + \frac{1}{2}D^2 d(0)[\nu, \nu] + \frac{1}{4!}D^4 d(0)[\nu, \nu, \nu, \nu] + \cdots, \tag{C.19}$$

so odd-order terms cancel. This is a local symmetry identity, not a claim of unconditional variance reduction.

Degenerate cases. If $d_+ = d_- = 0$, the implementation sets $u = 0$. If $d_+ = d_-$, then $d_o = 0$, $\kappa = 1$, and $u = d_+$. If $d_+ = -d_-$, then $d_e = 0$ and $u = 0$ even though $\kappa = \sqrt{2}$; the magnitude correction never invents a direction that is absent from the branch mean.

D TARGET-CONDITIONED DENOISING

The default TwinEdit-E variant uses an unchanged Gaussian noise tensor ϵ. The auxiliary TwinEdit-P variant modifies only the noise geometry. For either variant, let $\tilde{\epsilon}$ denote the selected noise. At denoising time τ_c, the noised latent is:

$$z_{\tau_c} = \alpha_{\tau_c} z_{\text{path}} + \sigma_{\tau_c} \tilde{\epsilon}. \tag{D.1}$$

The target-conditioned prediction is:

$$\hat{\epsilon}_t = \epsilon_\theta(z_{\tau_c}, c_t, \tau_c), \tag{D.2}$$

the recovered denoised latent is:

$$z_{\text{coh}} = \frac{z_{\tau_c} - \sigma_{\tau_c} \hat{\epsilon}_t}{\alpha_{\tau_c}}. \tag{D.3}$$

Substituting Eq. (D.1) into Eq. (D.3) gives the residual form:

$$z_{\text{coh}} = z_{\text{path}} + \frac{\sigma_{\tau_c}}{\alpha_{\tau_c}} \left(\tilde{\epsilon} - \hat{\epsilon}_t\right). \tag{D.4}$$

Thus the denoising step refines the already transported state; it does not replace its carrier.

D.1 AUXILIARY PROJECTED-NOISE VARIANT

For a nonzero token direction $u_\ell \in \mathbb{R}^C$, define the parallel and orthogonal projectors:

$$P_\ell^{\parallel} = \frac{u_\ell u_\ell^\top}{\|u_\ell\|_2^2}, \qquad P_\ell^{\perp} = I - P_\ell^{\parallel}. \tag{D.5}$$

Direct multiplication gives:

$$(P_\ell^{\perp})^\top = P_\ell^{\perp}, \tag{D.6}$$

$$(P_\ell^{\perp})^2 = P_\ell^{\perp}, \tag{D.7}$$

$$P_\ell^{\perp} u_\ell = 0, \tag{D.8}$$

$$\text{rank}(P_\ell^{\perp}) = C - 1. \tag{D.9}$$

For $\epsilon_\ell \sim \mathcal{N}(0, I_C)$:

$$P_\ell^{\perp} \epsilon_\ell \sim \mathcal{N}(0, P_\ell^{\perp}), \tag{D.10}$$

Therefore:

$$\mathbb{E}\|P_\ell^{\perp} \epsilon_\ell\|_2^2 = \text{tr}(P_\ell^{\perp}) = C - 1. \tag{D.11}$$

TwinEdit-P therefore uses:

$$\tilde{\epsilon}_\ell = \sqrt{\frac{C}{C-1}}\, P_\ell^{\perp} \epsilon_\ell. \tag{D.12}$$

This satisfies:

$$\langle \tilde{\epsilon}_\ell, u_\ell \rangle = 0, \tag{D.13}$$

$$\mathbb{E}\|\tilde{\epsilon}_\ell\|_2^2 = C = \mathbb{E}\|\epsilon_\ell\|_2^2. \tag{D.14}$$

Its covariance is:

$$\text{Cov}(\tilde{\epsilon}_\ell) = \frac{C}{C-1} P_\ell^{\perp}, \tag{D.15}$$

so the rescaling preserves expected total energy but not isotropic covariance. When $u_\ell = 0$, the implementation leaves the Gaussian noise unchanged.

E ENDPOINT REFERENCE AND INITIAL REALIZATION FIELD

The endpoint reference is constructed from the same source latent and four endpoint latents as the transport path. It introduces no additional denoiser invocation. The two procedures named in Algorithm 1 correspond to the construction of (J, r_0) through Eq. (E.23) and its transfer to the initial coefficient field through Eqs. (E.25)–(E.26).

E.1 SOURCE-MATCHED REFERENCE LATENTS

Define the source-response line direction as:

$$\delta_s = s_+ - s_-. \tag{E.1}$$

The source-line matching objective is:

$$f(\beta) = \|x - (s_- + \beta\delta_s)\|_2^2, \tag{E.2}$$

where $\beta \in \mathbb{R}$ denotes the scalar coordinate along the source-response line. Its first and second derivatives are:

$$\frac{\partial f}{\partial \beta} = -2\langle x - s_- - \beta\delta_s, \delta_s\rangle, \tag{E.3}$$

$$\frac{\partial^2 f}{\partial \beta^2} = 2\|\delta_s\|_2^2. \tag{E.4}$$

When $\|\delta_s\|_2^2 > 0$, the exact minimizer is:

$$\beta_{\mathrm{sm}}^\star = \frac{\langle x - s_-, \delta_s\rangle}{\|\delta_s\|_2^2}. \tag{E.5}$$

The finite implementation uses the stabilized coefficient:

$$\beta_{\mathrm{sm}} = \frac{\langle x - s_-, \delta_s\rangle}{\|\delta_s\|_2^2 + 10^{-10}}, \tag{E.6}$$

and sets $\beta_{\mathrm{sm}} = 1/2$ only when $\|\delta_s\|_2^2 = 0$. The coefficient is not clipped to $[0, 1]$; away from the numerical floor, Eq. (E.6) approaches the orthogonal projection coefficient in Eq. (E.5). Using the same coefficient on the target responses gives:

$$\bar{s} = (1 - \beta_{\mathrm{sm}})s_- + \beta_{\mathrm{sm}}s_+, \qquad \bar{t} = (1 - \beta_{\mathrm{sm}})t_- + \beta_{\mathrm{sm}}t_+. \tag{E.7}$$

The first reference latent is:

$$p = x + \bar{t} - \bar{s}. \tag{E.8}$$

Define the branch-disagreement direction as:

$$v = (t_+ - s_+) - (t_- - s_-) = d_+ - d_-. \tag{E.9}$$

The drift-corrected reference latent is:

$$q = \begin{cases} p, & \|v\|_2^2 \le 10^{-10}, \\ p - \dfrac{\langle x - \bar{s}, v\rangle}{\|v\|_2^2} v, & \text{otherwise.} \end{cases} \tag{E.10}$$

The correction in the second branch is a projection along the branch-disagreement direction; when this direction is numerically degenerate, no correction is applied.

The decoded reference anchors are mapped to the fixed 8-bit RGB representation used by the reference implementation before their source-relative radii are measured:

$$A_1 = \mathcal{Q}_{\mathrm{rgb}}(D(p)), \qquad A_2 = \mathcal{Q}_{\mathrm{rgb}}(D(q)). \tag{E.11}$$

Here $\mathcal{Q}_{\mathrm{rgb}}$ denotes the fixed RGB serialization used by the implementation. It maps both reference anchors to the uint8 RGB grid and back to float32 values in $[0, 1]$. The source image $S = I_s$ is represented in RGB and scaled by $1/255$.

E.2 RESPONSE-SHAPED RADIUS

The reference-anchor radius is the smaller of the two source-relative decoded displacements:

$$r_c(i) = \min\left\{\|A_1(i) - S(i)\|_2, \|A_2(i) - S(i)\|_2\right\}. \tag{E.12}$$

The four endpoint latents are decoded directly:

$$S_\pm = D(s_\pm), \qquad T_\pm = D(t_\pm). \tag{E.13}$$

Their mean response magnitude is:

$$m(i) = \frac{1}{2}\left(\|T_+(i) - S_+(i)\|_2 + \|T_-(i) - S_-(i)\|_2\right). \tag{E.14}$$

Define the spatial means over all $N = HW$ pixels as:

$$\bar{r}_c = \frac{1}{N}\sum_i r_c(i), \qquad \bar{m} = \frac{1}{N}\sum_i m(i). \tag{E.15}$$

For $\bar{m} > 10^{-10}$, the normalized response-shaped radius is:

$$r_{\text{shape}}(i) = m(i)\frac{\bar{r}_c}{\max(\bar{m}, 10^{-10})}. \tag{E.16}$$

Its spatial mean equals the reference-anchor-radius mean:

$$\frac{1}{N}\sum_i r_{\text{shape}}(i) = \frac{1}{N}\sum_i m(i)\frac{\bar{r}_c}{\bar{m}} \tag{E.17}$$

$$= \frac{\bar{r}_c}{\bar{m}}\left(\frac{1}{N}\sum_i m(i)\right) \tag{E.18}$$

$$= \bar{r}_c. \tag{E.19}$$

If both $\bar{m}$ and $\bar{r}_c$ are zero, the radius is zero. A zero response mean paired with a nonzero reference-anchor-radius mean is treated as an invalid construction rather than silently divided by the numerical constant.

E.3 ENDPOINT PROPOSAL AND REFERENCE SCALING

The branch-selection scores are:

$$\gamma_\pm = \frac{\|t_\pm - s_\pm\|_2}{\sqrt{\|t_\pm - s_\pm\|_2^2 + \|s_\pm - x\|_2^2 + 10^{-10}}}. \tag{E.20}$$

The selected target latent is:

$$t^\star = \begin{cases} t_+, & \gamma_+ - \gamma_- > 10^{-12}, \\ t_-, & \gamma_- - \gamma_+ > 10^{-12}, \\ \frac{1}{2}(t_+ + t_-), & \text{otherwise.} \end{cases} \tag{E.21}$$

Define the source and selected-target decodes as:

$$X = D(x), \qquad T = D(t^\star). \tag{E.22}$$

The endpoint proposal and fixed reference radius are:

$$J = \text{clip}_{[0,1]}(S + T - X), \qquad r_0 = 1.75\, r_{\text{shape}}. \tag{E.23}$$

The reference (J, r_0) is completely determined before target-conditioned denoising and does not depend on the decoded target K.

E.4 TRANSFER TO THE DECODED TARGET DISPLACEMENT

Define the reference and target displacements as:

$$\Delta_{\mathrm{ref}}(i) = J(i) - S(i), \qquad \Delta(i) = K(i) - S(i). \tag{E.24}$$

The reference-path fraction is:

$$\rho_0(i) = \begin{cases} 0, & \|\Delta_{\mathrm{ref}}(i)\|_2 \le 10^{-10}, \\ \min\left\{1, \dfrac{r_0(i)}{\|\Delta_{\mathrm{ref}}(i)\|_2}\right\}, & \text{otherwise.} \end{cases} . \tag{E.25}$$

The fraction is transferred to the clean source-to-K path using the realization scale η:

$$a_i = \begin{cases} 0, & \|\Delta(i)\|_2 \le 10^{-10}, \\ \min\{1, \eta\rho_0(i)\}, & \text{otherwise.} \end{cases} . \tag{E.26}$$

The corresponding transferred RGB radius is:

$$r(i) = a_i \|\Delta(i)\|_2. \tag{E.27}$$

Equations (E.25)–(E.27) give:

$$0 \le \rho_0(i) \le 1, \qquad 0 \le a_i \le 1, \qquad 0 \le r(i) \le \|\Delta(i)\|_2. \tag{E.28}$$

For fixed reference and K, a_i is nondecreasing in η until it saturates at one. This monotone family is the basis of the fixed η sweep in Figure 5.

Decoder workload. No extra denoiser invocation is required by this reference. Its explicit construction decodes $[s_+, t_+, s_-, t_-]$ in one batch and $[x, p, q]$ in a second batch; a branch tie requires one additional target decode.

F CONSTRAINED SPATIALLY COORDINATED REALIZATION

Define the final RGB displacement as:

$$\Delta_i = K(i) - S(i). \tag{F.1}$$

For any coefficient $y_i \in [0, 1]$:

$$S(i) + y_i\Delta_i = (1 - y_i)S(i) + y_iK(i), \tag{F.2}$$

so every realized pixel lies on the line segment between the source and the decoded target.

F.1 EDGEWISE DECOMPOSITION AND GRAPH ENERGY

For an edge (i, j), define:

$$\bar{\Delta}_{ij} = \frac{\Delta_i + \Delta_j}{2}, \qquad \bar{y}_{ij} = \frac{y_i + y_j}{2}. \tag{F.3}$$

The realized displacement difference has the exact decomposition:

$$y_i\Delta_i - y_j\Delta_j = \bar{\Delta}_{ij}(y_i - y_j) + \bar{y}_{ij}(\Delta_i - \Delta_j). \tag{F.4}$$

The first term isolates variation caused by the coefficient field. TwinEdit therefore uses:

$$w_{ij} = \|\bar{\Delta}_{ij}\|_2^2 = \left\| \frac{\Delta_i + \Delta_j}{2} \right\|_2^2, \tag{F.5}$$

The resulting weighted coefficient-field energy is:

$$\mathcal{E}(y) = \frac{1}{2} \sum_{(i,j)\in\mathcal{N}} w_{ij}(y_i - y_j)^2. \tag{F.6}$$

Let B_G be an oriented incidence matrix of the full four-connected image graph and $W = \mathrm{diag}(w_e)$. Coefficients outside the active support are held fixed at their initial values. Then:

$$\mathcal{E}(y) = \frac{1}{2}\|W^{1/2}B_G y\|_2^2 = \frac{1}{2}y^\top Ly, \qquad L = B_G^\top W B_G. \tag{F.7}$$

Hence:

$$\nabla\mathcal{E}(y) = Ly, \qquad \nabla^2\mathcal{E}(y) = L. \tag{F.8}$$

At the initial field a, define:

$$g = La. \tag{F.9}$$

F.2 DIAGONAL MAJORIZATION

Define the weighted degree and diagonal majorizer as:

$$d_i^w = \sum_{j:(i,j)\in\mathcal{N}} w_{ij}, \qquad D_w = \mathrm{diag}(d_i^w), \qquad B = 2D_w. \tag{F.10}$$

For any vector v:

$$v^\top (B - L)v = \sum_{(i,j)\in\mathcal{N}} w_{ij}(v_i + v_j)^2 \tag{F.11}$$

$$\geq 0. \tag{F.12}$$

Therefore:

$$L \preceq B. \tag{F.13}$$

Writing $\delta = y - a$, the quadratic energy has the exact expansion:

$$\mathcal{E}(a + \delta) = \mathcal{E}(a) + g^\top \delta + \frac{1}{2}\delta^\top L\delta. \tag{F.14}$$

Using Eq. (F.13) gives the upper model:

$$\mathcal{E}(y) \leq \mathcal{E}(a) + g^\top (y - a) + \frac{1}{2}(y - a)^\top B(y - a). \tag{F.15}$$

The diagonal entries of B are:

$$b_i = 2 \sum_{j:(i,j)\in\mathcal{N}} w_{ij}, \tag{F.16}$$

which yields the separable quadratic surrogate used in the method.

F.3 COMPONENT BUDGET AND FEASIBILITY

Define the squared displacement magnitude as:

$$e_i = \|\Delta_i\|_2^2. \tag{F.17}$$

For each four-connected component $\mathcal{C}$ of the support $\{i : a_i > 0\}$, define:

$$T_\mathcal{C} = \sum_{i\in\mathcal{C}} e_i a_i. \tag{F.18}$$

The finite update solves:

$$\begin{aligned} \min_{y_\mathcal{C}} \quad & q_\mathcal{C}(y; a) := \sum_{i\in\mathcal{C}} g_i(y_i - a_i) + \frac{1}{2}\sum_{i\in\mathcal{C}} b_i(y_i - a_i)^2 \\ \text{s.t.} \quad & \sum_{i\in\mathcal{C}} e_i y_i = T_\mathcal{C}, \qquad 0 \leq y_i \leq 1. \end{aligned} \tag{F.19}$$

The initial field $y = a$ is feasible because $0 \leq a_i \leq 1$ and its weighted sum equals $T_\mathcal{C}$. Therefore every component problem has a nonempty feasible set.

F.4 KARUSH–KUHN–TUCKER (KKT) SYSTEM AND CLIPPED COORDINATE SOLUTION

Introduce a multiplier μ for the equality constraint, lower-bound multipliers $\xi_i \geq 0$, and upper-bound multipliers $\zeta_i \geq 0$. The Lagrangian is:

$$\mathcal{L}(y, \mu, \xi, \zeta) = \sum_i g_i(y_i - a_i) + \frac{1}{2}\sum_i b_i(y_i - a_i)^2 \tag{F.20}$$

$$+ \mu\left(\sum_i e_i y_i - T_\mathcal{C}\right) - \sum_i \xi_i y_i + \sum_i \zeta_i(y_i - 1). \tag{F.21}$$

Stationarity requires:

$$g_i + b_i(y_i - a_i) + \mu e_i - \xi_i + \zeta_i = 0. \tag{F.22}$$

Primal and dual feasibility require:

$$\sum_i e_i y_i = T_{\mathcal{C}}, \qquad 0 \le y_i \le 1, \tag{F.23}$$

$$\xi_i \ge 0, \qquad \zeta_i \ge 0. \tag{F.24}$$

Complementary slackness is:

$$\xi_i y_i = 0, \qquad \zeta_i(y_i - 1) = 0. \tag{F.25}$$

For an interior coordinate $0 < y_i < 1$, both bound multipliers vanish, and Eq. (F.22) gives:

$$y_i = a_i - \frac{g_i + \mu e_i}{b_i}. \tag{F.26}$$

For every free coordinate with $b_i > 0$, applying the box constraints yields:

$$y_i(\mu) = \mathrm{clip}\left(a_i - \frac{g_i + \mu e_i}{b_i}, 0, 1\right). \tag{F.27}$$

Coordinates with $b_i = 0$ are never divided by b_i; the finite implementation holds them at a_i, as specified in Appendix G.

Define the equality residual as:

$$\phi_{\mathcal{C}}(\mu) = \sum_{i \in \mathcal{C}} e_i y_i(\mu) - T_{\mathcal{C}}. \tag{F.28}$$

On any interval where the unclipped active set $\mathcal{A}(\mu) = \{i : 0 < y_i(\mu) < 1\}$ is fixed:

$$\phi'_{\mathcal{C}}(\mu) = -\sum_{i \in \mathcal{A}(\mu)} \frac{e_i^2}{b_i} \le 0. \tag{F.29}$$

Thus $\phi_{\mathcal{C}}$ is continuous, piecewise linear, and nonincreasing. For $e_i > 0$, the breakpoints at which coordinate i reaches one or zero are:

$$\mu_i^{(1)} = \frac{b_i(a_i - 1) - g_i}{e_i}, \tag{F.30}$$

$$\mu_i^{(0)} = \frac{b_i a_i - g_i}{e_i}. \tag{F.31}$$

A valid root bracket is therefore:

$$\mu_{\mathrm{lo}} = \min_i \mu_i^{(1)}, \qquad \mu_{\mathrm{hi}} = \max_i \mu_i^{(0)}, \tag{F.32}$$

for which:

$$\phi_{\mathcal{C}}(\mu_{\mathrm{lo}}) \ge 0, \qquad \phi_{\mathcal{C}}(\mu_{\mathrm{hi}}) \le 0. \tag{F.33}$$

The scalar root can therefore be obtained without a multidimensional constrained optimizer.

Proposition F.1 (Weighted-energy nonincrease)**.** *Let $y^\star$ minimize Eq.* (F.19)*. Then:*

$$\mathcal{E}(y^\star) \le \mathcal{E}(a). \tag{F.34}$$

Proof. For every component, the feasible point $y_{\mathcal{C}} = a_{\mathcal{C}}$ has surrogate value $q_{\mathcal{C}}(a; a) = 0$. Optimality therefore gives:

$$q_{\mathcal{C}}(y^\star_{\mathcal{C}}; a) \le 0. \tag{F.35}$$

Define the global surrogate change as:

$$q(y; a) := \sum_{\mathcal{C}} q_{\mathcal{C}}(y_{\mathcal{C}}; a). \tag{F.36}$$

Summing the component inequalities gives $q(y^\star; a) \le 0$. Applying the majorization in Eq. (F.15) componentwise and summing gives:

$$\mathcal{E}(y^\star) \le \mathcal{E}(a) + q(y^\star; a) \le \mathcal{E}(a). \tag{F.37}$$

□

The finite solver in Appendix G restricts each component to a numerically free coordinate set and holds the remaining coordinates at a_i. Because a remains feasible for every such restricted problem, the same surrogate-nonpositivity argument applies to the evaluated finite update.

F.5 PRESERVED BUDGET VERSUS SQUARED RGB CHANGE

The equality constraint preserves the linear displacement-weighted amount:

$$M_{\mathcal{C}}(y) = \sum_{i \in \mathcal{C}} e_i y_i = T_{\mathcal{C}}. \tag{F.38}$$

The source-relative pre-quantization squared RGB change is instead:

$$Q_{\mathcal{C}}(y) = \sum_{i \in \mathcal{C}} \|y_i \Delta_i\|_2^2 = \sum_{i \in \mathcal{C}} e_i y_i^2. \tag{F.39}$$

Because $0 \le y_i \le 1$:

$$Q_{\mathcal{C}}(y) \le M_{\mathcal{C}}(y) = T_{\mathcal{C}}. \tag{F.40}$$

Cauchy–Schwarz gives:

$$\left(\sum_i e_i y_i \right)^2 \le \left(\sum_i e_i \right) \left(\sum_i e_i y_i^2 \right), \tag{F.41}$$

Therefore:

$$\frac{T_{\mathcal{C}}^2}{\sum_{i \in \mathcal{C}} e_i} \le Q_{\mathcal{C}}(y) \le T_{\mathcal{C}}. \tag{F.42}$$

The solver therefore preserves $M_{\mathcal{C}}$ exactly, not $Q_{\mathcal{C}}$. The near-constant Q values shown in Figure 4 are an empirical observation for the displayed realizations rather than an imposed equality.

Proposition F.2 (Global displacement-scale invariance of the analytical realization problem)**.** *Fix the initial coefficient field a and the set of free coordinates. If every final displacement is multiplied by a common nonzero scalar c, the analytical realization problem has the same minimizing coefficient field y.*

Proof. Under $\Delta_i' = c\Delta_i$:

$$e_i' = c^2 e_i, \qquad w_{ij}' = c^2 w_{ij}. \tag{F.43}$$

Consequently:

$$g' = c^2 g, \qquad b_i' = c^2 b_i, \qquad T_{\mathcal{C}}' = c^2 T_{\mathcal{C}}. \tag{F.44}$$

Both the objective and equality constraint in Eq. (F.19) are multiplied by c^2, while the box constraints and the fixed field a are unchanged. The minimizing coefficient field is therefore identical on the fixed free set. Appendix G separately specifies the machine-scale active-coordinate threshold used by the finite solver. □

G NUMERICAL SOLVER, EDGE CASES, AND COMPLEXITY

The optimization objective is Eq. (F.19); this section records the finite implementation used to solve it, using standard safeguarded numerical optimization and finite-precision practices (Nocedal & Wright, 2006; Higham, 2002). Source image, decoded target, initial field, displacement energies, gradient, and diagonal curvature are converted to float64 before the component solve.

For a component $\mathcal{C}$, define:

$$e_{\max} = \max_{i \in \mathcal{C}} e_i. \tag{G.1}$$

A coordinate enters the scalar root solve only if:

$$e_i > 64 \epsilon_{64} \max(e_{\max}, 1) \qquad \text{and} \qquad b_i > 0, \tag{G.2}$$

where ϵ_{64} denotes machine epsilon for float64. Coordinates that fail either condition remain fixed at a_i. Let $\mathcal{A}_{\mathcal{C}}$ denote the coordinates that pass Eq. (G.2), and let $\mathcal{F}_{\mathcal{C}} = \mathcal{C} \setminus \mathcal{A}_{\mathcal{C}}$ denote the coordinates held fixed. Since $y_i = a_i$ on $\mathcal{F}_{\mathcal{C}}$, the remaining active budget is:

$$T_{\mathcal{A}} = T_{\mathcal{C}} - \sum_{i \in \mathcal{F}_{\mathcal{C}}} e_i a_i = \sum_{i \in \mathcal{A}_{\mathcal{C}}} e_i a_i. \tag{G.3}$$

The scalar solver uses the largest active displacement energy:

$$e_{\max,\mathcal{A}} = \max_{i \in \mathcal{A}_{\mathcal{C}}} e_i. \tag{G.4}$$

It normalizes the active energies as:

$$\hat{e}_i = \frac{e_i}{e_{\max,\mathcal{A}}}, \qquad i \in \mathcal{A}_\mathcal{C}. \tag{G.5}$$

The corresponding normalized target is:

$$\hat{T}_\mathcal{A} = \frac{T_\mathcal{A}}{e_{\max,\mathcal{A}}} = \sum_{i\in\mathcal{A}_\mathcal{C}} \hat{e}_i a_i. \tag{G.6}$$

Using the normalized multiplier $\hat{\mu} = e_{\max,\mathcal{A}}\mu$, each active coordinate is evaluated as:

$$y_i(\hat{\mu}) = \operatorname{clip}\left(a_i - \frac{g_i + \hat{\mu}\hat{e}_i}{b_i}, 0, 1\right), \qquad i \in \mathcal{A}_\mathcal{C}. \tag{G.7}$$

The normalized equality residual is:

$$\hat{\phi}_\mathcal{C}(\hat{\mu}) = \sum_{i\in\mathcal{A}_\mathcal{C}} \hat{e}_i\big(y_i(\hat{\mu}) - a_i\big). \tag{G.8}$$

Thus $\hat{\phi}_\mathcal{C}(\hat{\mu}) = 0$ is equivalent to:

$$\sum_{i\in\mathcal{A}_\mathcal{C}} e_i\big(y_i - a_i\big) = 0, \tag{G.9}$$

$$\sum_{i\in\mathcal{C}} e_i y_i = T_\mathcal{C}, \tag{G.10}$$

because every coordinate in $\mathcal{F}_\mathcal{C}$ remains equal to a_i. The normalization therefore changes only the numerical scale of the scalar multiplier and residual, not the feasible coefficient field.

The numerical bracket is:

$$\hat{\mu}_{\mathrm{lo}} = \min_{i\in\mathcal{A}_\mathcal{C}} \frac{b_i(a_i - 1) - g_i}{\hat{e}_i}, \tag{G.11}$$

$$\hat{\mu}_{\mathrm{hi}} = \max_{i\in\mathcal{A}_\mathcal{C}} \frac{b_i a_i - g_i}{\hat{e}_i}. \tag{G.12}$$

Ignoring clipping, the normalized equality-constrained stationary multiplier is:

$$\hat{\mu}_{\mathrm{free}} = -\frac{\sum_{i\in\mathcal{A}_\mathcal{C}} \hat{e}_i g_i/b_i}{\sum_{i\in\mathcal{A}_\mathcal{C}} \hat{e}_i^2/b_i}. \tag{G.13}$$

The solver starts from:

$$\hat{\mu}_0 = \operatorname{clip}\left(\hat{\mu}_{\mathrm{free}}, \hat{\mu}_{\mathrm{lo}}, \hat{\mu}_{\mathrm{hi}}\right). \tag{G.14}$$

At each iteration, it evaluates the clipped coordinates, updates the bracket from the sign of $\hat{\phi}_\mathcal{C}$, and uses an in-bracket Newton step whenever the positive slope magnitude is:

$$\hat{s}(\hat{\mu}) = \sum_{i\in\mathcal{A}(\hat{\mu})} \frac{\hat{e}_i^2}{b_i} > 0, \tag{G.15}$$

where $\mathcal{A}(\hat{\mu})$ contains active coordinates whose unclipped value lies strictly between zero and one. Because $\hat{\phi}'_\mathcal{C}(\hat{\mu}) = -\hat{s}(\hat{\mu})$, the Newton candidate is:

$$\hat{\mu}_{\mathrm{new}} = \hat{\mu} + \frac{\hat{\phi}_\mathcal{C}(\hat{\mu})}{\hat{s}(\hat{\mu})}. \tag{G.16}$$

If the candidate leaves the bracket, is non-finite, or the slope magnitude is zero, the next iterate is the bracket midpoint. The solve uses at most 128 iterations and then performs at most two finite residual corrections on coordinates strictly inside the box.

For one scalar-root iteration, define the implementation round-off scale as:

$$R_\mathcal{C}(\hat{\mu}) = \sum_{i\in\mathcal{A}(\hat{\mu})} \hat{e}_i \left(\left|a_i - \frac{g_i}{b_i}\right| + \left|\frac{\hat{\mu}\hat{e}_i}{b_i}\right|\right), \tag{G.17}$$

which matches the accumulated magnitude used by the finite solver before clipping. The residual tolerance is:

$$\tau_{\mathcal{C}} = \max\left\{16\epsilon_{64}\max\left(\sum_{i\in\mathcal{A}_{\mathcal{C}}}\hat{e}_i, 1\right), 16\epsilon_{64}\max(R_{\mathcal{C}}(\hat{\mu}), 1)\right\}. \tag{G.18}$$

The convergence test applies this tolerance to $|\hat{\phi}_{\mathcal{C}}|$. A nontrivial component that fails to converge raises an error; no hidden fallback optimizer is used.

Direct exits and invariant faces. If the support is empty, the output remains $y = a$. All-zero, all-one, and single-coordinate groups have direct exits. Pixels with $a_i = 0$ are outside the support and cannot be activated by the solver. Pixels with $a_i = 1$ may leave the upper face when they pass the active-coordinate test; this is the distinction between TwinEdit and the support-locked control. If $\|K(i) - S(i)\|_2 \leq 10^{-10}$, the implementation sets both the transferred radius and the initial coefficient to zero.

Verification gates. The implementation checks finite values, $0 \leq y_i \leq 1$, per-component mass residual at most 10^{-10}, invariance of inactive and zero-displacement pixels, weighted-energy nonincrease, and normalized KKT residual at most 10^{-9}.

Final quantization. The scientific output is the float32 RGB image from Eq. (A.7). The stored PNG is produced only at the final boundary:

$$I_{\mathrm{u8}} = \mathrm{uint8}\left(\mathrm{rint}\left(255\ \mathrm{clip}_{[0,1]}(I_{\mathrm{out}})\right)\right). \tag{G.19}$$

Complexity. For a component with $n_{\mathcal{C}}$ pixels and $m_{\mathcal{C}}$ graph edges, constructing e, w, g, and b costs:

$$O(n_{\mathcal{C}} + m_{\mathcal{C}}). \tag{G.20}$$

Each scalar residual evaluation costs $O(n_{\mathcal{C}})$. With at most R root iterations, the component solve costs:

$$O(m_{\mathcal{C}} + Rn_{\mathcal{C}}). \tag{G.21}$$

Summed over the image grid, final realization is linear in the number of pixels for fixed R and introduces no neural-network evaluation.

H Additional Quantitative Results

Table 4 reports all seven metrics in their original units for both TwinEdit variants and both backbones.

Table 4: Detailed TwinEdit results on PIE-Bench.

Backbone	Variant	Structure↓	PSNR↑	MSE↓	SSIM↑	LPIPS↓	Whole↑	Edited↑
SD-Turbo	E	0.029053	22.8057	0.007908	0.788084	0.118095	**25.7662**	**22.4897**
SD-Turbo	P	**0.028440**	**22.8376**	**0.007803**	**0.789653**	**0.117274**	25.6244	22.3384
SwiftBrush-v2	E	**0.022479**	24.7209	0.005368	0.842354	**0.093431**	**24.5011**	**21.1702**
SwiftBrush-v2	P	0.022955	**24.7210**	**0.005354**	**0.843791**	0.093790	24.3256	20.9086

H.1 Carrier Effects Across Output Stages and Realization Scales

Tables 5 and 6 report the carrier-stage results at the default $\eta = 2.2$ and their paired confidence intervals. We additionally vary η for the encoded-source and mean-response carriers while holding the endpoint responses, transport update, target-conditioned denoising, endpoint reference, and final coordination fixed. Figure 6 shows the measured operating points, and Table 7 reports the closest Edited-CLIP and Whole-CLIP pairs.

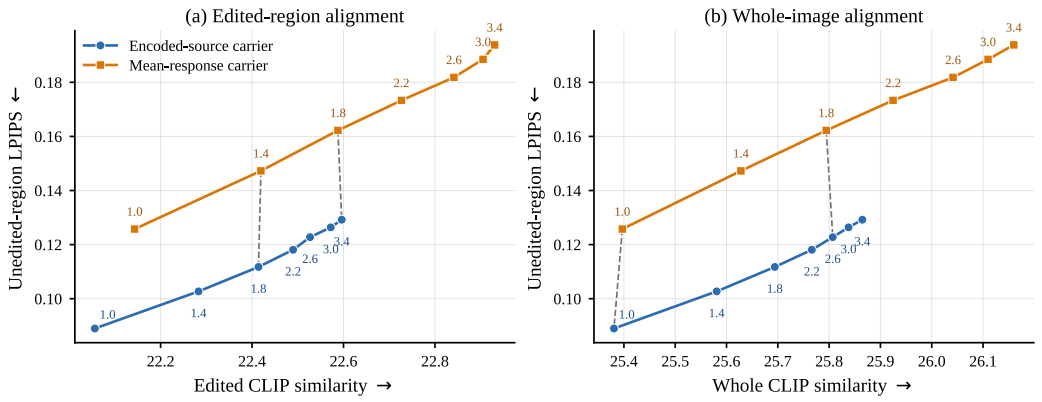

Figure 6: Measured carrier operating points on PIE-Bench. Both curves vary only the realization scale η; the endpoint responses, transport construction, target-conditioned denoising, endpoint reference, and final coordination remain fixed. Markers denote the seven measured settings, and connecting lines are visual guides. The curves overlap over part of the measured alignment range, enabling nearest-point comparisons without interpolation.

Table 5: Carrier and output-stage comparison on PIE-Bench. The endpoint responses, transport update, target-conditioned denoising step, and endpoint reference are shared within each stage comparison.

Carrier	Output stage	Structure↓	PSNR↑	MSE↓	SSIM↑	LPIPS↓	Whole↑	Edited↑
Encoded source x	Direct K	0.036659	21.2902	0.010465	0.73381	0.14588	25.9761	22.7191
	Initial field	0.028872	22.7359	0.008019	0.78363	0.12166	25.7778	22.4854
	Final output	0.029053	22.8057	0.007908	0.78808	0.11809	25.7663	22.4897
Mean response $\bar{s}$	Direct K	0.084835	18.0288	0.020461	0.67570	0.22972	26.6382	23.1740
	Initial field	0.053553	19.9217	0.013766	0.73422	0.18075	25.9206	22.6964
	Final output	0.054615	19.9957	0.013551	0.74249	0.17332	25.9246	22.7267
Source response s_+	Final output	0.059140	18.1892	0.020828	0.68798	0.19612	25.9382	22.9376
Source response s_-	Final output	0.080160	20.1770	0.012737	0.75876	0.17515	24.7049	21.0769

Table 6: Paired carrier differences at each output stage. Differences are encoded source minus mean response. Intervals use 5,000 paired image-level bootstrap resamples.

Metric	Direct K	Initial field	Final output
Structure	-0.04818 $[-0.05070, -0.04573]$	-0.02468 $[-0.02611, -0.02327]$	-0.02556 $[-0.02703, -0.02420]$
PSNR	$+3.261$ $[+3.110, +3.413]$	$+2.814$ $[+2.662, +2.971]$	$+2.810$ $[+2.659, +2.968]$
MSE	-0.009996 $[-0.010600, -0.009410]$	-0.005747 $[-0.006123, -0.005384]$	-0.005642 $[-0.006019, -0.005290]$
SSIM	$+0.05811$ $[+0.05446, +0.06194]$	$+0.04941$ $[+0.04648, +0.05232]$	$+0.04559$ $[+0.04287, +0.04838]$
LPIPS	-0.08384 $[-0.08796, -0.07982]$	-0.05909 $[-0.06189, -0.05641]$	-0.05522 $[-0.05794, -0.05263]$
Whole CLIP	-0.662 $[-0.829, -0.488]$	-0.143 $[-0.294, +0.020]$	-0.158 $[-0.312, -0.003]$
Edited CLIP	-0.455 $[-0.653, -0.252]$	-0.211 $[-0.397, -0.030]$	-0.237 $[-0.412, -0.053]$

For the encoded-source carrier, Table 8 gives paired intervals for the final coordination step relative to the initial field.

Table 8: Effect of coefficient coordination relative to the initial field. Differences are final output minus initial field.

Metric	Paired n	Mean difference	95% interval
Structure	700	+0.000180	[+0.000143,+0.000232]
PSNR	556	+0.06974	[+0.06511,+0.07461]
MSE	556	-0.000111	[-0.000120,-0.000102]
SSIM	556	+0.004455	[+0.004160,+0.004756]
LPIPS	556	-0.003565	[-0.003967,-0.003169]
Whole CLIP	700	-0.01156	[-0.02202,-0.00115]
Edited CLIP	700	+0.00430	[-0.00721,+0.01559]

Table 7: Nearest measured carrier operating points. Differences are encoded-source minus mean-response. The CLIP gap is the absolute difference between the matched operating points. All five preservation-metric confidence intervals exclude zero for the reported pairs.

(a) Edited-CLIP pairs

$\eta_{\bar{s}}$	η_x	Abs. CLIP gap	Δ Structure ↓	Δ PSNR ↑	Δ MSE ↓	Δ SSIM ↑	Δ LPIPS ↓
1.4	1.8	0.0053	-0.01558	+2.003	-0.00325	+0.01984	-0.03552
1.8	3.4	0.0082	-0.01755	+1.705	-0.00339	+0.00659	-0.03301

(b) Whole-CLIP pairs

$\eta_{\bar{s}}$	η_x	Abs. CLIP gap	Δ Structure ↓	Δ PSNR ↑	Δ MSE ↓	Δ SSIM ↑	Δ LPIPS ↓
1.0	1.0	0.0163	-0.01283	+2.431	-0.00295	+0.03347	-0.03677
1.8	2.6	0.0127	-0.01922	+2.060	-0.00396	+0.01972	-0.03945

H.2 Hyperparameter Sensitivity and RRLS Operating Points

Table 9 reports the four one-at-a-time TwinEdit-E sweeps. Tables 10 and 11 report the five RRLS selector settings and the nearest measured Edited-CLIP and LPIPS comparisons.

Table 9: One-at-a-time sensitivity of TwinEdit-E. Each block varies one parameter while holding the others at their default values.

η	LPIPS↓	Edited↑	λ	LPIPS↓	Edited↑
1.0	0.0890	22.056	0	0.1112	22.286
1.4	0.1027	22.283	0.25	0.1134	22.381
1.8	0.1117	22.414	0.50	0.1158	22.469
2.2	**0.1181**	**22.490**	**0.75**	**0.1181**	**22.490**
2.6	0.1228	22.527	1.00	0.1206	22.543
3.0	0.1264	22.572			
3.4	0.1292	22.596			
τ_c	LPIPS↓	Edited↑	τ_e	LPIPS↓	Edited↑
0.25	0.0881	21.469	0.55	0.0964	21.876
0.35	0.1015	21.990	0.65	0.1042	22.177
0.45	**0.1181**	**22.490**	**0.75**	**0.1181**	**22.490**
0.55	0.1366	22.829	0.85	0.1353	22.695
0.65	0.1560	22.992	0.95	0.1531	22.643

Table 10: RRLS operating points on PIE-Bench. All settings use the same SD-Turbo backbone and evaluator.

λ_{src}	Structure↓	LPIPS↓	Whole↑	Edited↑
0	0.02796	0.11636	25.4343	22.1855
30	0.02659	0.11339	25.3948	22.1451
40	**0.02625**	**0.11258**	**25.3819**	**22.1330**
50	0.02579	0.11191	25.3626	22.1183
100	0.02386	0.10701	25.2583	22.0328

Table 11: Nearest measured TwinEdit–RRLS operating points. Differences are TwinEdit minus RRLS; † marks a paired 95% interval excluding zero.

(a) Nearest measured Edited-CLIP comparisons

RRLS λ_{src}	TwinEdit η	Δ Edited	Δ Structure	Δ PSNR	Δ MSE	Δ SSIM	Δ LPIPS
0	1.4	+0.0974	-0.0030†	+0.199†	-0.000211	+0.0163†	-0.0137†
30	1.0	-0.0889	-0.0054†	+0.961†	-0.001013†	+0.0373†	-0.0244†
40	1.0	-0.0768	-0.0051†	+0.892†	-0.000919†	+0.0357†	-0.0236†
50	1.0	-0.0621	-0.0046†	+0.839†	-0.000842†	+0.0343†	-0.0229†
100	1.0	+0.0234	-0.0027†	+0.435†	-0.000263†	+0.0230†	-0.0180†

(b) Nearest measured LPIPS comparisons

RRLS λ_{src}	TwinEdit η	Δ LPIPS	Δ Whole	Δ Edited
0	2.2	+0.0017	+0.332†	+0.304†
30	1.8	-0.0016	+0.299†	+0.269†
40	1.8	-0.0008	+0.312†	+0.281†
50	1.8	-0.0002	+0.332†	+0.296†
100	1.4	-0.0043†	+0.322†	+0.250†

H.3 Category-Wise TwinEdit-E Results

Table 12 reports the primary TwinEdit-E configuration across the ten official PIE-Bench edit categories.

Table 12: Category-wise TwinEdit-E results on PIE-Bench. Structure and both CLIP metrics use all edits in each official category. LPIPS uses the valid unedited-region domain, whose size is shown in the final column. The style LPIPS value is based on only two valid images and is therefore descriptive.

Category	Structure↓	LPIPS↓	Whole↑	Edited↑	n_{unedit}
General / random	0.0289	0.1108	25.34	22.88	121
Object replacement	0.0356	0.1417	25.76	20.69	80
Object addition	0.0230	0.0990	25.78	23.74	53
Object deletion	0.0266	0.1530	24.59	17.88	80
Content attribute	0.0259	0.1224	25.38	21.91	36
Pose	0.0216	0.1082	26.39	22.32	35
Color	0.0273	0.1268	26.00	20.59	39
Material	0.0279	0.1031	25.88	23.55	40
Background	0.0338	0.0844	26.00	23.21	70
Style	0.0333	0.1289	27.16	27.05	2

I Additional Ablation Results

I.1 Path Construction

Table 13 separates magnitude correction from target-conditioned denoising and compares antithetic sampling with independent two-sample path estimates. The carrier, endpoint reference, and final realization are shared across the comparisons.

Table 13: Path-construction controls on PIE-Bench. Panel (a) separates magnitude correction from the target-conditioned denoising step. Panel (b) compares the antithetic pair with three independent two-sample path estimates. Each estimate uses two source–target response pairs. All rows use the encoded-source carrier, the same endpoint reference, and the same final realization.

(a) Magnitude correction and target-conditioned denoising

Path estimate	Denoising	Structure↓	PSNR↑	MSE↓	SSIM↑	LPIPS↓	Whole↑	Edited↑
Two-branch mean	No	0.013130	26.0442	0.004154	0.853512	0.060703	23.9283	20.6538
Magnitude-corrected mean	No	0.020631	24.3484	0.006102	0.830608	0.081848	24.2599	20.9161
Two-branch mean	Yes	0.026296	23.4177	0.006849	0.795318	0.111178	25.5854	22.2855
Magnitude-corrected mean	Yes	0.029053	22.8057	0.007908	0.788084	0.118095	25.7663	22.4897

(b) Antithetic and independent response pairs

Path samples	Structure↓	PSNR↑	MSE↓	SSIM↑	LPIPS↓	Whole↑	Edited↑
Antithetic pair	0.029053	22.8057	0.007908	0.788084	0.118095	25.7663	22.4897
Independent pair 1	0.030068	22.7054	0.008087	0.786672	0.120065	25.9650	22.7421
Independent pair 2	0.030218	22.7214	0.008034	0.786637	0.119889	25.9569	22.7053
Independent pair 3	0.030123	22.7547	0.008047	0.786748	0.119671	25.9788	22.7856

Magnitude correction increases Edited CLIP by 0.262 without target-conditioned denoising and 0.204 with it. The denoising step increases Edited CLIP by 1.632 without magnitude correction and 1.574 with it. Independent sampling moves the result further toward alignment, whereas the antithetic pair retains more source content.

I.2 Endpoint-Reference Simplifications

Table 14 compares the TwinEdit endpoint reference with two simpler alternatives while keeping the decoded target and final realization fixed.

Table 14: Endpoint-reference simplifications on PIE-Bench. The decoded target and final realization are fixed across all rows.

Reference	Structure↓	PSNR↑	MSE↓	SSIM↑	LPIPS↓	Whole↑	Edited↑
TwinEdit reference	0.029052	22.8057	0.007908	0.788085	0.118095	25.7664	22.4898
Source-matched reference p only	0.029438	22.7143	0.008043	0.784884	0.119569	25.7758	22.5011
Uniform response radius	0.031087	22.7877	0.007996	0.767011	0.130283	25.6519	22.3420

Using only the source-matched reference anchor weakens preservation. Removing the spatial response pattern degrades both preservation and alignment.

J Sensitivity to Random Seeds

We evaluate the default TwinEdit-E configuration on all 700 PIE-Bench edits over 20 seeds. The full-benchmark PSNR and Edited-CLIP aggregates have relative standard deviations of 0.62% and 0.25%, respectively.

Table 15: Full-benchmark seed robustness of TwinEdit-E over 700 PIE-Bench edits and 20 seeds.

Metric	Between-seed std.	Relative std.
PSNR↑	0.1393	0.62%
Edited CLIP↑	0.0552	0.25%

K Runtime Details

Runtime is measured on one RTX 3090 using 50 fixed images, five warm-up runs, and seven timed runs per image. Table 16 reports the complete end-to-end timing distribution. GPU activity profiling was unavailable for this run, so we do not report a stage-level breakdown.

Table 16: End-to-end runtime on one RTX 3090. Each variant uses 50 fixed images, five warm-up runs, and seven timed runs per image.

Backbone	Variant	Mean	Median	P90	P95	Alloc. MiB
SD-Turbo	E	0.9882	0.9910	1.0030	1.0083	5497.97
SD-Turbo	P	0.9737	0.9856	1.0042	1.0106	5497.97
SwiftBrush-v2	E	0.9755	0.9734	0.9955	1.0040	5505.61
SwiftBrush-v2	P	0.9740	0.9710	0.9966	1.0012	5505.61

L Additional Qualitative Comparisons

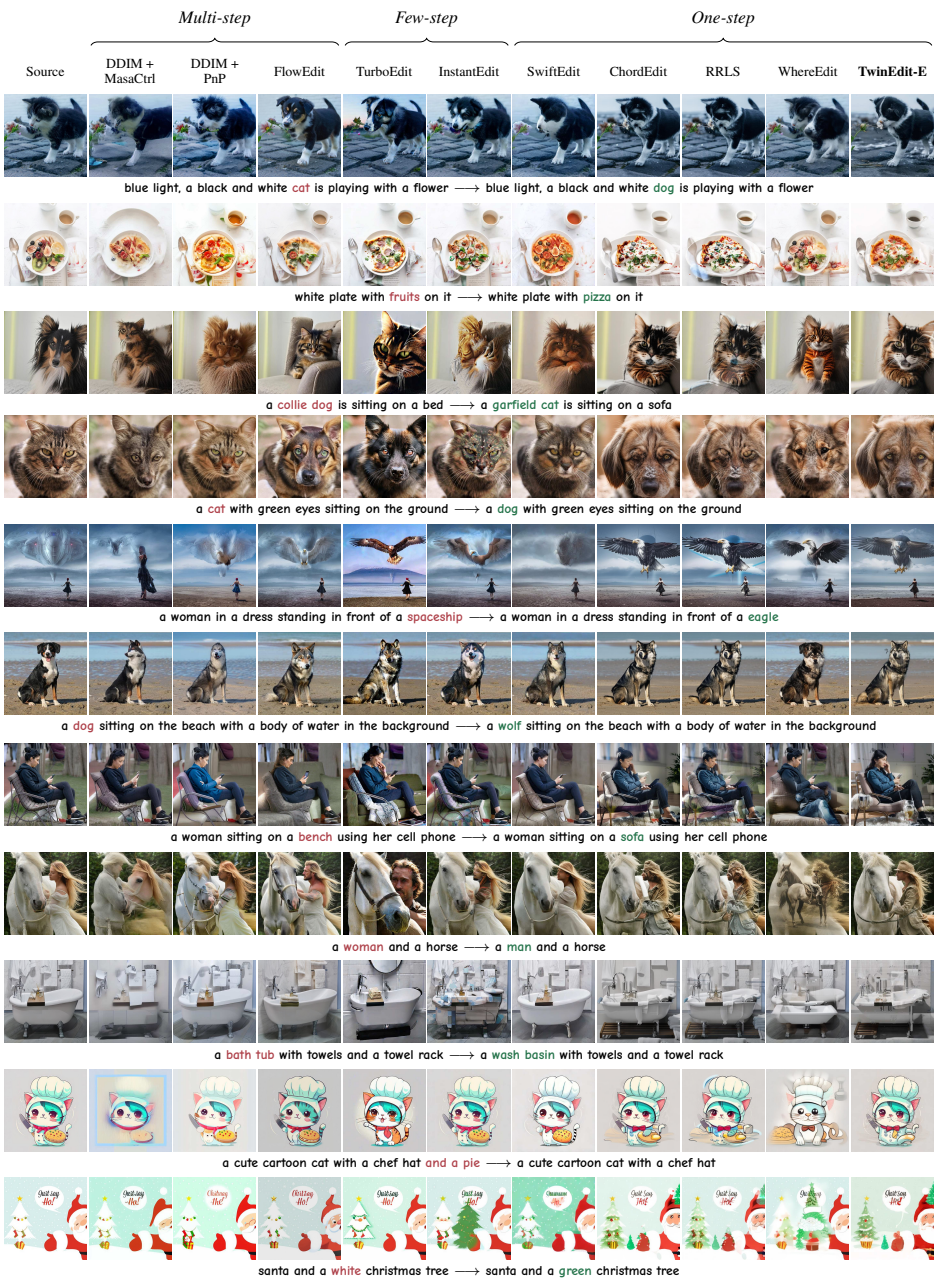

Figure 7: Additional qualitative comparisons on PIE-Bench. Source images appear in the first column. Source and target changes are marked below each row in red and green, respectively. The final column shows TwinEdit-E with SD-Turbo.

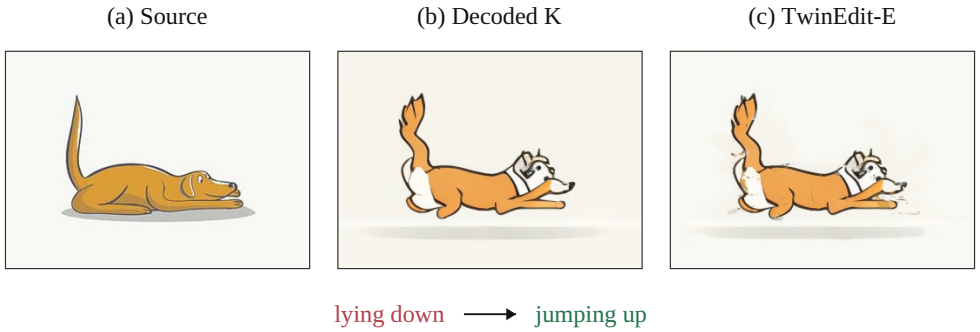

Figure 8: Challenging pose edit. The decoded target K changes the pose but is not fully aligned with the source. Under this spatial mismatch, pointwise source-to-K realization may leave faint source contours, as seen around the hind legs and tail.

M Human Preference Study

We conduct a blind pairwise human preference study on a shared pool of 80 edits sampled uniformly from PIE-Bench. Twenty participants completed 20 initial trials each, with four comparisons against each of ChordEdit, WhereEdit, SwiftEdit, TurboEdit, and RRLS, representing training-free one-step, learned one-step, and few-step editors from Table 1. Each trial showed the editing instruction, source image, and two anonymized results; trial order and left–right placement were randomized. Participants separately judged semantic alignment and preservation quality, with a no-preference option. The former assessed instruction following, while the latter assessed naturalness and preservation of unrelated source content. Participants selecting no preference in more than half of the initial trials for either criterion completed five additional trials, one per baseline. One participant met this rule, yielding 405 pairwise trials and 81 judgments per baseline for each criterion. Reported percentages pool trial-level judgments by baseline and criterion.

TwinEdit-E receives the largest preference share against every comparison method under both criteria. Its preference rate ranges from 49% to 57% for semantic alignment and from 52% to 67% for preservation quality, while no-preference responses account for 10%–11%.

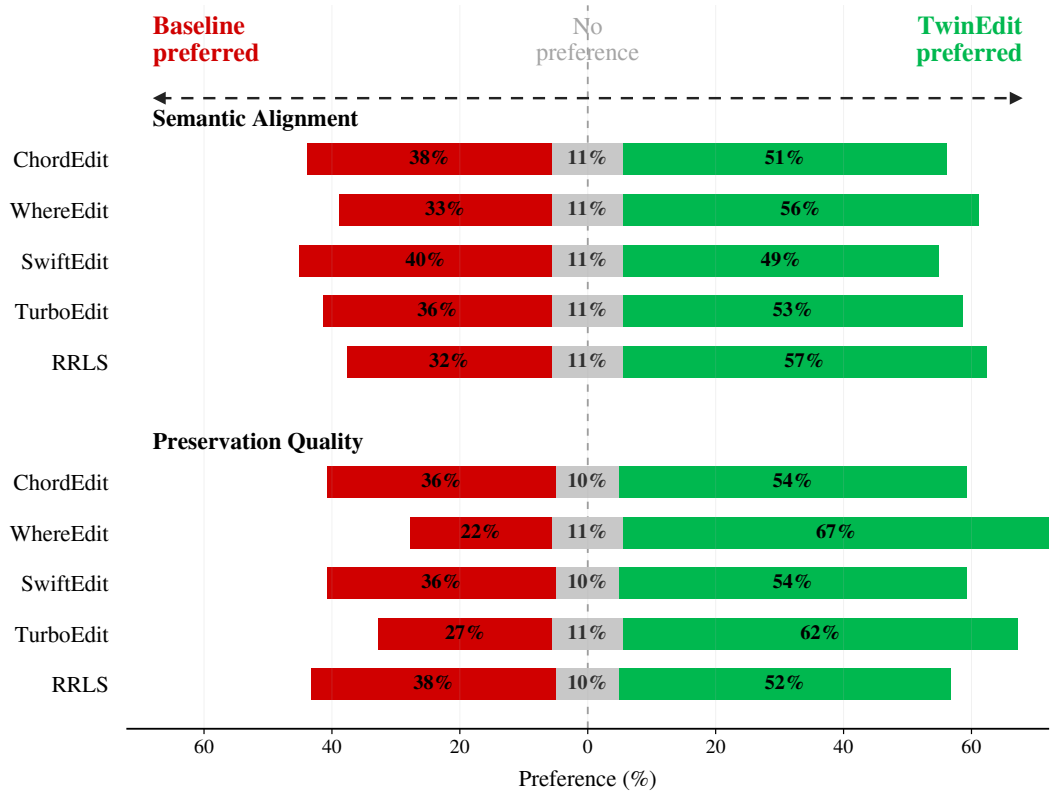

Figure 9: Blind pairwise human preferences on PIE-Bench. Bars show the percentages of judgments preferring the comparison method, expressing no preference, or preferring TwinEdit-E, separately for semantic alignment and preservation quality.

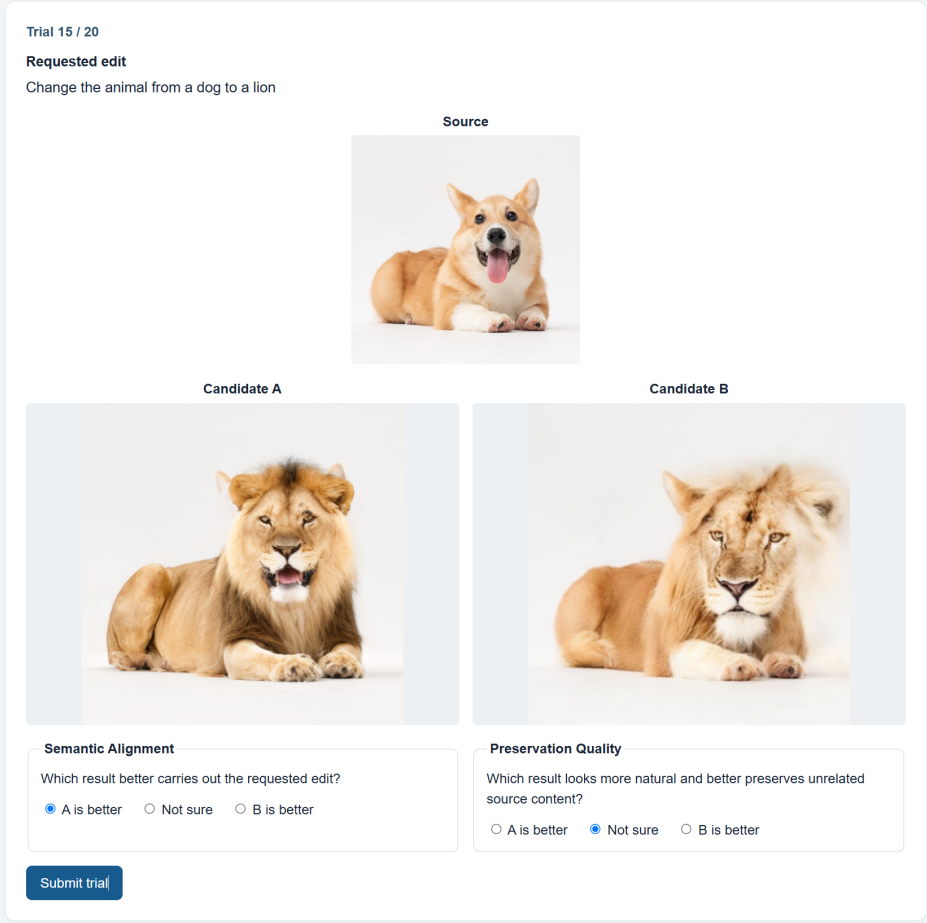

Figure 10: Human preference study interface. Each trial presents the editing instruction, source image, and anonymized candidates in randomized left–right order. Participants answer separate questions for semantic alignment and preservation quality.

N SYMBOLS TABLE

Symbol	Meaning	Symbol	Meaning
I_s, S	Source RGB image in $[0,1]^{H\times W\times 3}$.	m	Mean endpoint-response magnitude field.
x	Encoded source latent $E(I_s)$.	r_{shape}	Mean-normalized response-shaped radius.
c_s, c_t	Source and target text conditions.	J	Fixed endpoint proposal $\mathrm{clip}(S+T-X)$.
F_{τ_e}	Reconstructed endpoint latent response at time τ_e.	$\hat{\epsilon}_\theta$	Backbone noise prediction.
Π_τ	Scheduler endpoint postprocessing.	r_0	Fixed endpoint radius $1.75r_{\mathrm{shape}}$.
$s_\pm, t_\pm$	Source- and target-conditioned endpoint latents under $\pm\nu$.	ρ_0	Fraction of the endpoint proposal displacement allowed by r_0.
$d_\pm$	Branch displacements $t_\pm - s_\pm$.	η	Global realization scale.
$d_{\mathrm{e}}, d_{\mathrm{o}}$	Even and odd branch components.	a	Initial realization coefficient field after transfer to $K-S$.
κ	Capped magnitude-completion factor.	Δ_i	Clean RGB displacement $K(i)-S(i)$.
λ	Fraction of the magnitude completion applied to the branch mean.	w_{ij}	Displacement-weighted graph edge weight.
u	Final latent transport update.	$\mathcal{E}(y)$	Weighted coefficient-field energy.
z_{path}	Source-anchored transported latent $x+u$.	e_i	Squared RGB displacement $\|\Delta_i\|_2^2$.
τ_c	Target-conditioned denoising time.	$T_{\mathcal{C}}$	Preserved linear displacement-weighted budget of component $\mathcal{C}$.
$\epsilon, \tilde{\epsilon}$	Gaussian denoising noise and the selected E/P noise tensor.	g, b	Energy gradient at a and diagonal majorizer curvature.
z_{coh}	Denoised target latent recovered after target-conditioned denoising.	y	Final realized coefficient field.
K	Decoded denoised target image.	$\mu_{\mathcal{C}}$	Scalar equality multiplier for component $\mathcal{C}$.
ℓ	Latent-token index.	$\mathcal{A}_{\mathcal{C}}, \mathcal{F}_{\mathcal{C}}$	Numerically free and fixed coordinates within component $\mathcal{C}$.
β_{sm}	Stabilized source-line matching coefficient used by the endpoint reference.	$\hat{e}_i$	Active displacement energy normalized by the component's largest active e_i.
p, q	Source-matched and drift-corrected carrier latents.	$\hat{\mu}$	Equality multiplier in the normalized scalar solve.
A_1, A_2	RGB-bound decoded carrier anchors.	ϵ_{64}	Float64 machine epsilon used by the active-coordinate and residual tolerances.
r_c	Pixelwise carrier radius.	$M_{\mathcal{C}}, Q_{\mathcal{C}}$	Preserved linear budget and squared RGB change.