FlowDC: Flow-Based Decoupling-Decay for Complex Image Editing

Supplementary Material

This supplementary document is organized as follows:

- In Sec. A, we show more details about the construction and containment of Complex-PIE-Bench
- In Sec. B.1, we describe the implementation details for both competing methods and our method, outlining how the quantitative and qualitative results were obtained.
- In Sec. B.2, we present additional qualitative results.
- In Sec. B.3, we provide additional quantitative comparisons with ParallelEdit and analyze the performance trade-offs of our method and baselines under different configurations.
- In Sec. B.4, we provide a user study to compare our method against other baselines.
- In Sec. B.5, we analyze the computation cost of our method and other baselines.
- In Sec. C, we analyze the editing trajectory, which demonstrates how VOD maintains source consistency.
- In Sec. D, we provide more ablation results and explain the selection of hyperparameters for our method
- In Sec. E, we discuss our work's limitations.
- In Sec. F, we provide the full algorithm of our method.
- In Sec. G, we provide the mathematical justification of PSO design.
- In Sec. H, we provide orthogonality analysis of editing targets.

A. Datasets

Our work builds upon the original PIE-Bench, which provides samples with a single target prompt across 10 editing categories, outlined as follows: 0) Random; 1) Change object; 2) Add object; 3) Delete object; 4) Change attribute content; 5) Change attribute pose; 6) Change attribute color; 7) Change attribute material; 8) Change background; 9) Change style.

To construct our benchmark, Complex-PIE-Bench, we extend each initial sample from a single prompt to a set of four editing targets and prompts. We employed the pre-trained multi-modal model (doubao-seed-1.6) for this expansion. The model generates new pairs conditioned on the initial pair and the source image.

The generation process adheres to two main principles: 1) Editing targets must not be conflicting (e.g., first transforming an object's color and then deleting that same object). 2) Global editing categories (i.e., background modification, style transfer) and the object deletion category are not repeated within the same sample. The resulting editing category distribution for Complex-PIE-Bench is shown in Fig. 8.

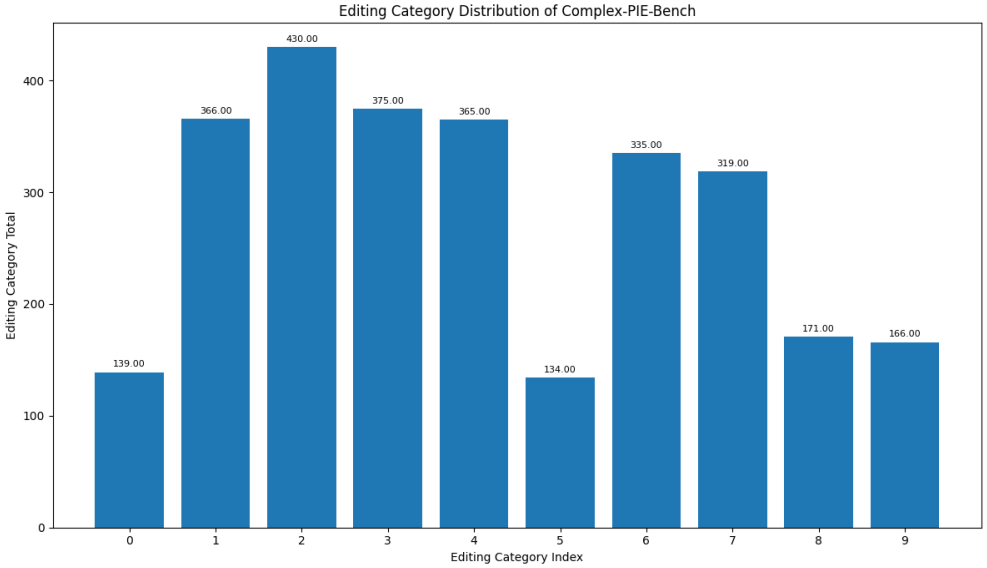

Figure 8. Editing Category Distribution of Complex-PIE-Bench

B. Additional Experiments

B.1. Additional details on the experiment settings

In Sec. 4, we compare our method against seven competing approaches: 1) ODE Inversion, 2) SDEdit [26], 3) RF Inversion [30], 4) RF Edit [37], 5) Multi Turn [46], 6) FlowEdit [20], and 7) ParallelEdit [14]. For each baseline (with the exception of ParallelEdit), we explored multiple sets of hyperparameter configurations on Complex-PIE-Bench. The hyperparameters adopted for the main experiments in Sec. 4 are highlighted in **bold**.

For ODE-Inv, SDEdit, and FlowEdit, following the setting in FlowEdit [20], we set total steps to 28. As detailed in Table 3, the CFG scales for the source and target were set to 1.5 and 5.5, respectively, while the start timestep t_1 was varied.

Table 3. Hyperparameters for ODE-Inv, SDEdit, and FlowEdit.

Method	steps T	source CFG	target CFG	start step t_1
ODE-Inv	28	1.5	5.5	20/28, **24/28**
SDEdit	28	-	5.5	**21/28**, 24/28
FlowEdit	28	1.5	5.5	23/28, **24/28**, 25/28

Regarding RF Inversion, we adhered to the official implementation settings: the control strength η was fixed at 0.9 and the starting timestep s at 0, with the stopping timestep τ being the variable parameter (see Table 4).

Table 4. Hyperparameters for RF Inversion.

Method	strength η	stopping step τ
RF Inversion	0.9	5/28, **7/28**

For RF Edit, we adopted the standard guidance scale of 2 and 30 total steps, varying injection steps as shown in Table 5.

Table 5. Hyperparameters for RF Edit.

Method	guidance	injection steps
RF Edit	2	**2**, 4

For Multi Turn, consistent with the official codebase, we set the total time steps T to 15 and experimented with different guidance scales, as detailed in Table 6.

Table 6. Hyperparameters for Multi Turn.

Method	steps T	guidance
Multi Turn	15	2.5, **3.5**

For ParallelEdit, we employed the official implementation built upon a diffusion model (LCMDreamshaperv7).

Finally, for our proposed method, we aligned our settings with FlowEdit [20] by applying source and target guidance scales of 1.5 and 5.5, respectively. We set the total time steps to $T = 28$, with key time steps configured as $t_1 = 27/28$, $t_g = 22/28$, and $t_o = 27/28$. The decay factor $\lambda(t)$ (Eq. 13) was varied as presented in Table 7.

Table 7. Hyperparameters for Ours.

Method	Decay Strength $(\lambda_1, \lambda_d, t_d)$
Ours	(0.1, 1.0, 11/28), **(0.1, 0.64, 20/28)**, (0.3, 1.0, 20/28)

B.2. Additional Qualitative Results

Fig. 9 shows additional qualitative comparisons between our method and other baselines.

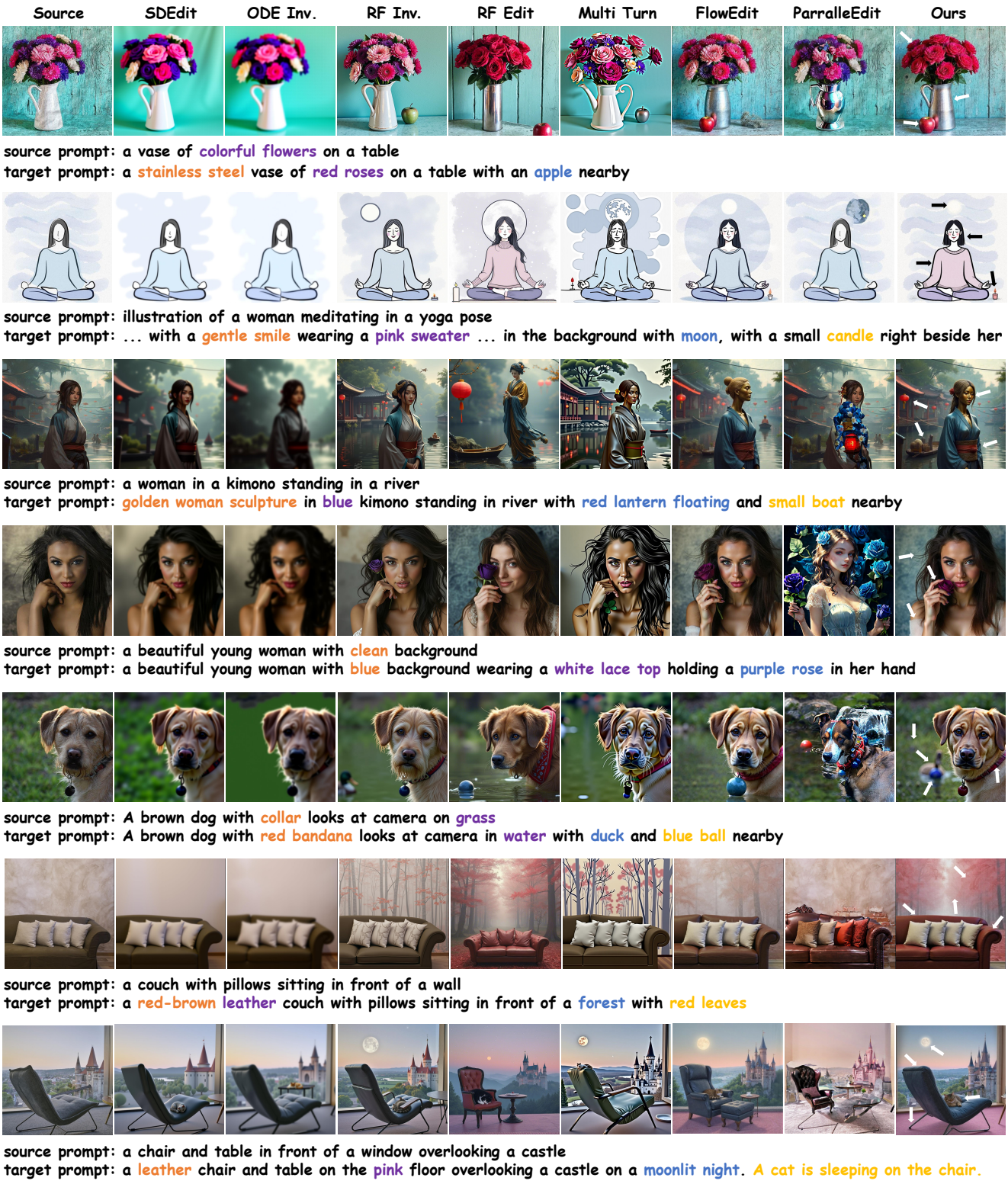

Figure 9. **Additional qualitative Comparison of complex editing.** Editing targets are indicated by differently colored text and arrows.

B.3. Additional Quantitative Results

Comparison with ParallelEdit. As the official implementation of ParallelEdit is compatible with only a subset of PIE-Bench++, we limit our comparison to this specific subset, which contains 508 samples of total 700 samples. The decay strength parameters of our method $(\lambda_1, \lambda_d, t_d)$ are set to $(0.2, 1.0, 22/28)$. As shown in Table 8, our method achieves slightly superior semantic alignment and comparable source consistency to ParallelEdit. Notably, while ParallelEdit relies on complex prompt engineering to obtain these results, our approach is more flexible and user-friendly, requiring only simple inputs.

Table 8. Additional quantitative comparison with ParallelEdit.

Method	CLIP-T (%) ↑	CLIP-I (%) ↑	DINO (%) ↑	LPIPS (%) ↓
ParallelEdit [14]	26.47	84.22	**59.88**	32.42
Ours	**26.51**	**84.47**	58.84	**31.46**

Additional Results. While Table 1 presents our quantitative comparison using fixed hyperparameters, we conduct an additional quantitative comparison on Complex-PIE-Bench across a range of hyperparameter settings (detailed in Sec. B.1), which allows us to intuitively map the performance trade-off for each method. We employ LPIPS to measure source consistency (lower is better) and CLIP-T to assess semantic alignment (higher is better). The Fig. 10 clearly demonstrate that FlowDC achieves a superior and more consistent balance across the performance envelope. In contrast, competing methods typically reveal a distinct compromise: they either rigidly preserve the original image structure, resulting in insufficient editing intensity, or they heavily modify the image, significantly compromising its source consistency.

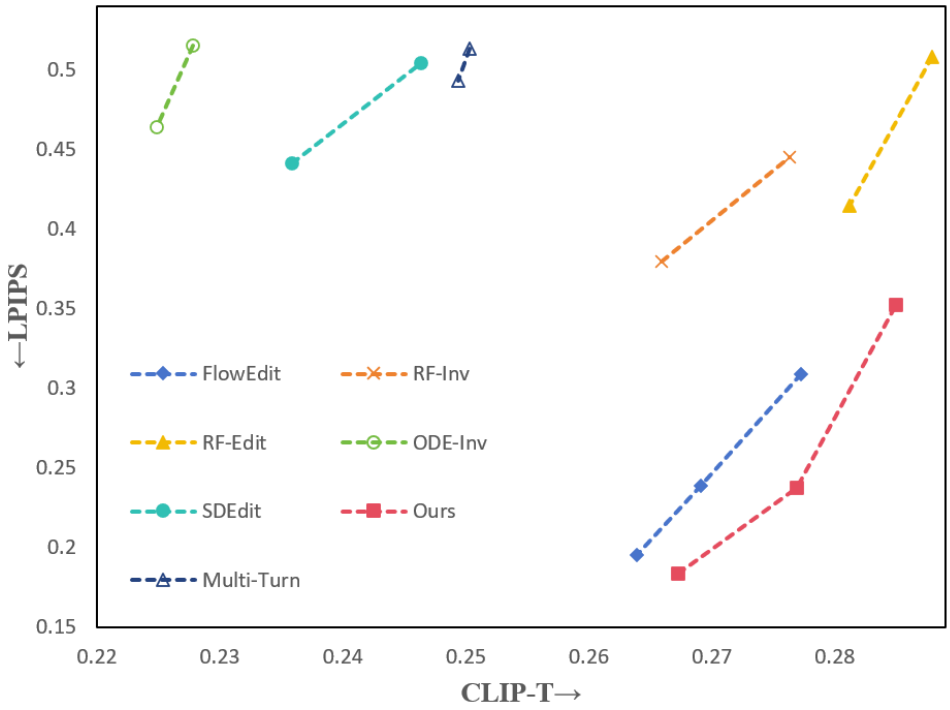

Figure 10. Additional quantitative comparisons.

B.4. User Study

Setting. We conducted a user study to compare our method against five baselines: FlowEdit [20], RF-Inversion [30], RF-Edit [37], Multi-Turn [46], and ParallelEdit [14]. The study comprised 15 trials, where each trial presented participants with a reference image, a source prompt, a target prompt and six target images. A total of 16 participants were invited to evaluate the results. In every trial, participants were asked to select one to three images based on three criteria: semantic alignment,

source consistency, and total editing quality. For each method, the preference rate reflects the selection frequency aggregated across all trials and participants. The evaluation interface is illustrated in Fig. 11.

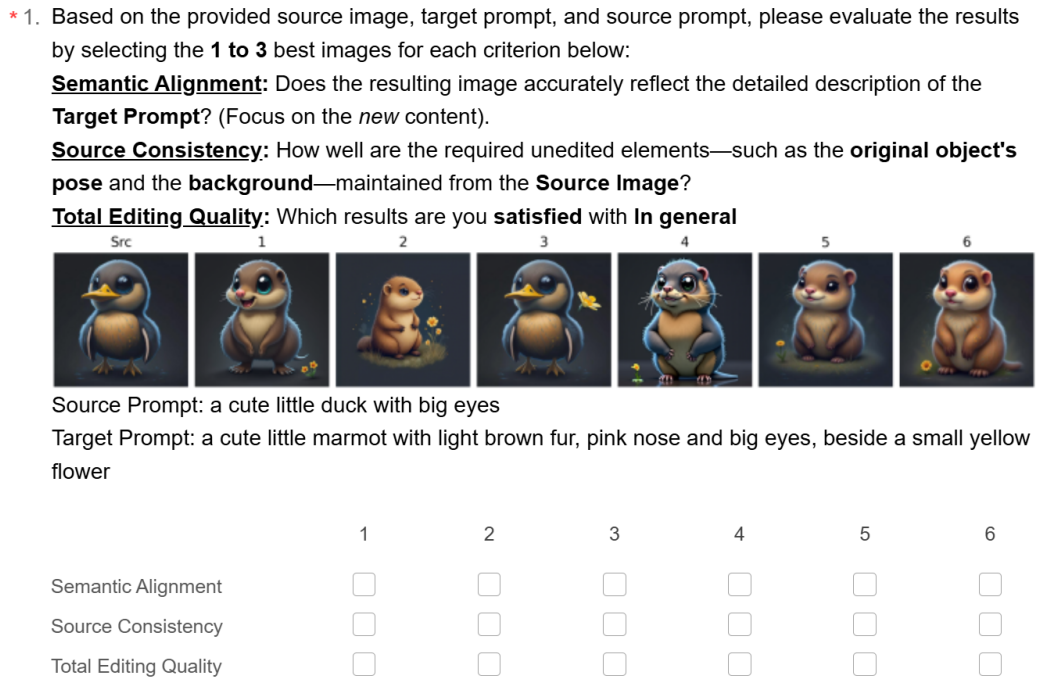

Figure 11. Example in user-study.

Results. As reported in Table. 9, our method outperforms competing approaches across all metrics, achieving the highest scores in Semantic Alignment (SA), Source Consistency (SC), and Total Editing Quality (EQ).

Table 9. User study results of preference rates (%) for Semantic Alignment (SA), Source Consistency (SC) and Total Editing Quality (EQ).

Method	Ours	FlowEdit	RF Edit	RF Inv	ParallelEdit	Multi Turn
SA	**78.67**	34.67	58.22	14.22	12.89	3.11
SC	**77.78**	30.22	12.44	44.44	21.33	16.44
EQ	**76.44**	19.11	20.44	18.22	7.56	3.11

B.5. Computation Cost

For n targets, the NFE increases by $2s(n-1)$ over s steps of parallel velocity generation. Considering efficiency and quality, we set $s = 2$ in practice. A detailed NFE comparison is provided in Tab. 10.

Table 10. NFE comparison (editing targets number $n = 4$).

Method	FlowDC	FlowEdit	MultiTurn	RF-Edit	RF-Inv.
NFE	68	48	120	60	56

C. Editing Trajectory Analysis

As illustrated in Fig. 5 and Fig. 7, the orthogonal component typically corresponds to unstable structural changes that are irrelevant to the editing objectives. By employing Velocity Orthogonal Decay (VOD), we effectively suppress this orthogonal

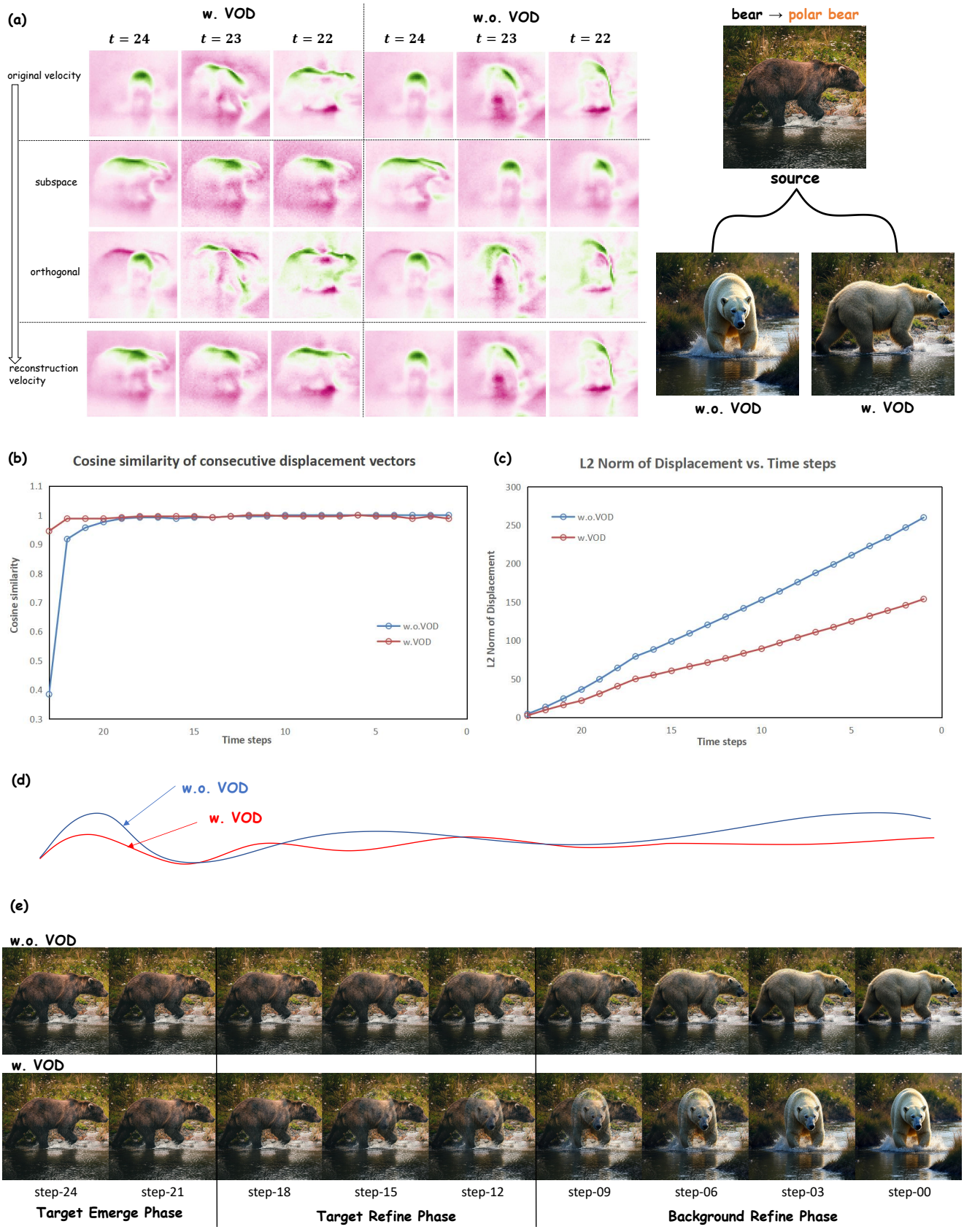

Figure 12. Visualization of the editing trajectory.

component, thereby preserving source consistency. We further evaluate the application of VOD to single-target editing with the start time step set to $t_1 = 24/28$. As shown in Fig. 12 (a), VOD successfully maintains source consistency in this setting as well.

In Fig. 12 (b), we visualize the cosine similarity of consecutive displacement vectors. The similarity is notably low during the early stages without VOD, indicating an unstable editing trajectory. In contrast, with the application of VOD, the cosine similarity approaches 1.0, resulting in a significantly more stable trajectory. As demonstrated in Fig. 12 (c), this stabilized trajectory incurs a lower transportation cost, which ultimately leads to superior source consistency. We provide a schematic abstraction of these two trajectories in Fig. 12 (d).

Furthermore, by decoding the latent at each time step, as shown in Fig. 12 (e), we can decompose the editing process into three distinct phases: *target emergence*, *target refinement*, and *background refinement*. Initially, the high-level concept of the target (e.g., 'polar bear') manifests during the *target emergence* phase. This is followed by the refinement of the subject. Finally, the model refines the overall image, including the background surrounding the target. Crucially, we observe that disruptions to source consistency (e.g., alterations in the bear's pose) originate during the target emergence phase and are subsequently refined into the final output.

D. Hyperparameter Exploration for FlowDC

Hyperparameters for VOD. Given that significant structural deviations primarily arise during the first two phases, we concentrate the application of VOD on these stages. Moreover, recognizing that earlier time steps are more prone to compromising source consistency, we employ a decaying strategy where the decay strength starts at a high value and gradually diminishes over time. We further analyze the impact of different VOD decay hyperparameters in Fig. 13. As observed in Fig. 13 (b), when the orthogonal component is completely suppressed (decayed to 0) only in the early phase, the model fails to effectively refine the 'polar bear' target. Conversely, extending this complete suppression throughout the entire process, as shown in Fig. 13 (c), introduces visible grid artifacts in the final image. These empirical observations motivate the specific parameter configurations detailed in Table 7.

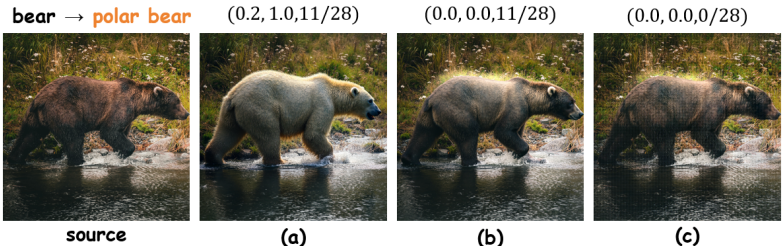

Figure 13. Ablation study on VOD decay hyperparameters. The tuple above each image denotes $(\lambda_1, \lambda_d, t_d)$.

Hyperparameters for PSO. The parameter t_o controls the operational steps of PSO. In practice, we configure PSO to run for a single step by setting $t_o = 27/28$, which is identical to t_1. Table 2 demonstrates that this single-step application is sufficient to improve semantic alignment. We further explore the impact of various PSO hyperparameters using the first 300 samples of Complex-PIE-Bench, as detailed in Table 11. We observe that increasing the number of PSO steps from 1 to 7 leads to a deterioration in both semantic alignment and source consistency. This is because the initial PSO step is derived from velocities at the designated guidance time (t_g), which ensures high precision due to the low interpolation weight of Gaussian noise. In contrast, subsequent PSO steps rely on displacements at the current time step. These displacements contain minor biases compared to the precise editing direction, causing errors of orthogonal basis to accumulate over multiple orthogonal iterations of PSO. To ensure precision, we compute the guidance velocities by averaging the results of three calculations at the designated guidance time.

Table 11. Ablation study on PSO steps.

PSO steps	CLIP-T (%) ↑	CLIP-I (%) ↑	DINO (%) ↑	LPIPS (%) ↓
0	27.32	**87.11**	**68.44**	<u>24.62</u>
1	**27.84**	<u>86.76</u>	<u>68.17</u>	**24.46**
7	<u>27.58</u>	86.64	67.63	25.21

E. Limitation

The main limitation of FlowDC lies in the underutilization of the PSO mechanism. Currently, PSO is restricted to a single step, as extending it to multiple steps introduces accumulated errors from biased displacement estimates. This constraint limits the method's potential in handling extremely complex editing scenarios that require sustained semantic guidance. Future work will focus on mitigating these displacement errors to enable stable multi-step PSO, thereby enhancing the controllability for intricate editing tasks.

F. Full Algorithm

Algorithm 3 outlines the complete pipeline of FlowDC.

Algorithm 3 Overall Pipeline of FlowDC

Input: source image X^{src}, source prompt P^{src}, complex target prompt P^{tar}, time steps t_1, t_g, t_o
Output: edited image Z_0
$\{P^{tar_i}\}_{i=1}^n \leftarrow LLM(P^{src}, P^{tar})$
$\{Z_{t_1}^i\}_{i=1}^n \leftarrow \{X^{src}\}_{i=1}^n$
$\{v_g^i\}_{i=1}^n \leftarrow \frac{1}{3}\sum_{k=1}^{3} \mathrm{PVG}(\{P^{tar_i}\}_{i=1}^n, t_g)$
for $t : t_1 \rightarrow t_o$ **do**
  for $i : 1 \rightarrow n$ **do**
    $v^i(t) \leftarrow PVG(P^{tar_i}, t)$ # original
    $d^i \leftarrow Z_t^i - X^{src}$
    if $t = t_1$ **then**
      $\mathbf{U}_i(t) \leftarrow PVO(\{v_g^i\}_{j=1}^i)$
    else
      $\mathbf{U}_i(t) \leftarrow PVO(\{d^j\}_{j=1}^i)$
    end if
    $v_{sub}^i(t) \leftarrow Proj(v^i(t), \mathbf{U}_i(t))$
    $v_{orth}^i(t) \leftarrow v^i(t) - v_{sub}^i(t)$
    $v^{i\prime}(t) \leftarrow v_{sub}^i(t) + \lambda_{orth}(t) v_{orth}^i(t)$ # precise
    $Z_t^i \leftarrow Z_t^i - v^{i\prime}(t)\Delta t$
  end for
end for
$Z_t \leftarrow Z_t^n$
for $t : t_1 \rightarrow t_o$ **do**
  $v(t) \leftarrow PVG(P^{tar}, t)$ # original
  $d \leftarrow Z_t - X^{src}$
  $\mathbf{U}(t) \leftarrow d$
  $v_{sub}(t) \leftarrow Proj(v(t), \mathbf{U}(t))$
  $v_{orth}(t) \leftarrow v(t) - v_{sub}(t)$
  $v'(t) \leftarrow v_{sub}(t) + \lambda_{orth}(t) v_{orth}(t)$ # precise
  $Z_t \leftarrow Z_t - v'(t)\Delta t$
end for

G. Mathematical Justification of PSO

In this section, we mathematically verify why PSO enhances editability in FlowDC. Let $\mathcal{U}_{\text{PSO}} = \{u_i\}_{i=1}^n$ denote the subspace spanned by the orthogonal basis derived from the cumulative displacements $\{d_i\}_{i=1}^n$, and let $\mathcal{U}_n = \{d_n\}$ be the subspace spanned solely by the final displacement d_n. Defining $\mathcal{P}_{\text{PSO}}(v)$ and $\mathcal{P}_n(v)$ as the orthogonal projections of a velocity field v onto $\mathcal{U}_{\text{PSO}}$ and $\mathcal{U}_n$ respectively, we assert:

$$\|\mathcal{P}_{\text{PSO}}(v)\|^2 \geq \|\mathcal{P}_n(v)\|^2 \tag{15}$$

The equality holds if and only if all intermediate editing directions are collinear with the final direction d_n.

Hilbert Projection Theorem. Let H be a **Hilbert space** and let M be a closed **subspace** of H. For every vector $x \in H$, there exists a unique element $y \in M$ such that:

$$\|x - y\| \leq \|x - z\|, \quad \langle x - y, z \rangle = 0, \quad \forall z \in M \tag{16}$$

Proof of Eq. 15. First, let $\mathcal{U}_{\text{latent}}$ denote the entire latent feature space. The subspaces $\mathcal{U}_n$ and $\mathcal{U}_{\text{PSO}}$ are defined as described previously. Since the final displacement d_n is a component of the generating set $\{d_1, \ldots, d_n\}$ of $\mathcal{U}_{\text{PSO}}$, it follows that $d_n \in \mathcal{U}_{\text{PSO}}$. Consequently, we establish the following nested subspace relationship:

$$\mathcal{U}_n \subseteq \mathcal{U}_{\text{PSO}} \subseteq \mathcal{U}_{\text{latent}} \tag{17}$$

Given a vector $v \in \mathcal{U}_{\text{latent}}$, according to the Hilbert Projection Theorem, v can be decomposed into its projection onto $\mathcal{U}_{\text{PSO}}$ and an orthogonal residual w_1:

$$\begin{aligned} &v = \mathcal{P}_{\text{PSO}}(v) + w_1, \\ &\langle w_1, z \rangle = 0, \quad \forall z \in \mathcal{U}_{\text{PSO}} \end{aligned} \tag{18}$$

Since $\mathcal{U}_n \subseteq \mathcal{U}_{\text{PSO}}$, w_1 is also orthogonal to $\mathcal{U}_n$:

$$\langle w_1, z \rangle = 0, \quad \forall z \in \mathcal{U}_n \tag{19}$$

Next, we expand the projection of v onto the smaller subspace $\mathcal{U}_n$. Using the linearity of the projection operator and the result from Eq. 19:

$$\begin{aligned} \mathcal{P}_n(v) &= \mathcal{P}_n(\mathcal{P}_{\text{PSO}}(v) + w_1) && \text{(by Eq. 18)} \\ &= \mathcal{P}_n(\mathcal{P}_{\text{PSO}}(v)) + \mathcal{P}_n(w_1) && \text{(by Linearity)} \\ &= \mathcal{P}_n(\mathcal{P}_{\text{PSO}}(v)) && \text{(since } w_1 \perp \mathcal{U}_n\text{)} \end{aligned} \tag{20}$$

Now, we consider the projection of the vector $\mathcal{P}_{\text{PSO}}(v)$ onto $\mathcal{U}_n$. We can decompose $\mathcal{P}_{\text{PSO}}(v)$ into the component in $\mathcal{U}_n$ and a residual w_2 orthogonal to $\mathcal{U}_n$:

$$\begin{aligned} \mathcal{P}_{\text{PSO}}(v) &= \mathcal{P}_n(\mathcal{P}_{\text{PSO}}(v)) + w_2 && \text{(by Eq. 16)} \\ &= \mathcal{P}_n(v) + w_2 && \text{(by Eq. 20)} \end{aligned} \tag{21}$$

where $\langle \mathcal{P}_n(v), w_2 \rangle = 0$ due to orthogonality. Finally, we obtain the norm relationship:

$$\|\mathcal{P}_{\text{PSO}}(v)\|^2 = \|\mathcal{P}_n(v)\|^2 + \|w_2\|^2 \tag{22}$$

Since the squared norm $\|w_2\|^2 \geq 0$, we conclude that $\|\mathcal{P}_{\text{PSO}}(v)\|^2 \geq \|\mathcal{P}_n(v)\|^2$.

H. Orthogonality Analysis of Editing Targets

Editing targets are represented as vectors in the high-dimensional VAE latent space. Spatially, non-overlapping editing targets are naturally orthogonal. For edits within the same region (*e.g.*, color vs. material), while they may not be isolated to specific single channels, they induce distinct activation patterns across the feature dimensions. Empirically, Fig. 3 (heatmaps) and Fig. 14 corroborate this assumption.

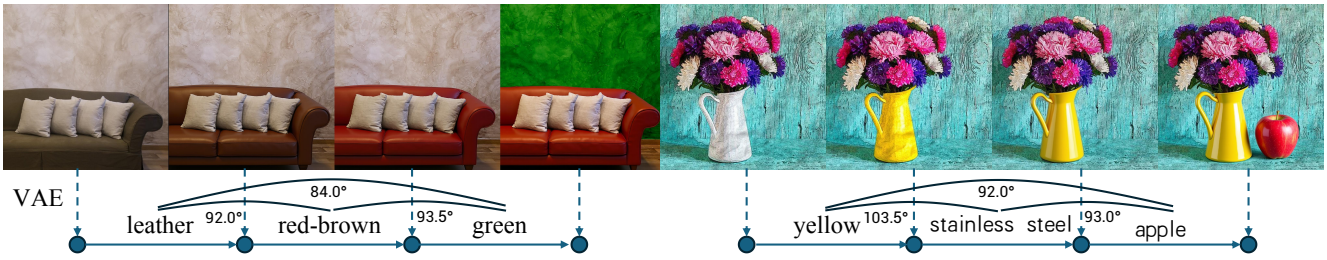

Figure 14. Editing vectors connect the VAE latent codes of different images. Angles between these vectors (derived from cosine similarity) are approximately $90°$, illustrating orthogonality.