

TDEdit: A Unified Diffusion Framework for Text-Drag Guided Image Manipulation

Qihang Wang¹ Yaxiong Wang¹ * Lechao Cheng¹ Zhun Zhong¹
¹Hefei University of Technology

Abstract

This paper explores image editing under the joint control of text and drag interactions. While recent advances in text-driven and drag-driven editing have achieved remarkable progress, they suffer from complementary limitations: text-driven methods excel in texture manipulation but lack precise spatial control, whereas drag-driven approaches primarily modify shape and structure without fine-grained texture guidance. To address these limitations, we propose a unified diffusion-based framework for joint drag-text image editing, integrating the strengths of both paradigms. Our framework introduces two key innovations: (1) Point-Cloud Deterministic Drag, which enhances latent-space layout control through 3D feature mapping, and (2) Drag-Text Guided Denoising, dynamically balancing the influence of drag and text conditions during denoising. Notably, our model supports flexible editing modes—operating with text-only, drag-only, or combined conditions—while maintaining strong performance in each setting. Extensive quantitative and qualitative experiments demonstrate that our method not only achieves high-fidelity joint editing but also matches or surpasses the performance of specialized text-only or drag-only approaches, establishing a versatile and generalizable solution for controllable image manipulation. Code will be made publicly available to reproduce all results presented in this work.

1. Introduction

The advent of diffusion models [11, 27, 29] has marked a paradigm shift in generative image editing, offering unprecedented control over content creation and manipulation. Building upon this foundation, two distinct yet powerful editing paradigms have emerged: text-driven [1, 7, 18, 32, 33] and drag-driven [16, 24, 28, 36, 37] approaches. Text-driven methods, empowered by large-scale vision-language models like CLIP [26], enable high-level semantic edits through natural language instructions - allowing users to modify object

attributes, swap textures, or even alter scene semantics with remarkable fidelity. Concurrently, drag-driven techniques have gained traction for their intuitive spatial manipulation capabilities, where users can deform, reposition, or rotate objects through simple point-and-drag interactions. While both approaches have demonstrated impressive results in their respective domains, they represent fundamentally different axes of control: one operating in the semantic space and the other in the geometric space.

A closer examination reveals that these two editing models exhibit complementary strengths and limitations: Text-driven approaches, while excelling at texture modification like color changing, object replacement, often struggle with precise spatial control due to the inherent ambiguity of language descriptions and the limitations of cross-attention mechanisms in diffusion models. For instance, while a prompt like "make the dog fluffier" can effectively modify texture, attempts to specify exact poses or orientations through text alone frequently lead to inconsistent or unpredictable results [7, 18, 21]. Conversely, drag-driven methods provide pixel-level precision for geometric transformations but lack the semantic understanding needed for coherent texture synthesis or attribute modification [15, 24, 28]. This limitation becomes particularly apparent when users attempt to combine geometric changes with texture edits - a common requirement in practical editing scenarios. Nevertheless, current models only pay attention to either text-driven control or drag-based edit, which cannot meet the complex requirements in practical scenarios.

To address this limitation, we propose a unified editing framework capable of simultaneously processing drag-based and text-based controls while preserving the individual capabilities of each modality. Developing such a framework presents two key challenges. (1) Dynamic Guidance Balancing: The denoising process requires careful modulation between text and drag guidance to prevent one modality from dominating the other. Naively enforcing both conditions may lead to degraded control fidelity in either domain. (2) Geometric Consistency in Diffusion: Maintaining precise spatial alignment under hybrid conditions remains nontrivial, as the interplay between text and drag signals

*Corresponding author: wangyx15@stu.xjtu.edu.cn

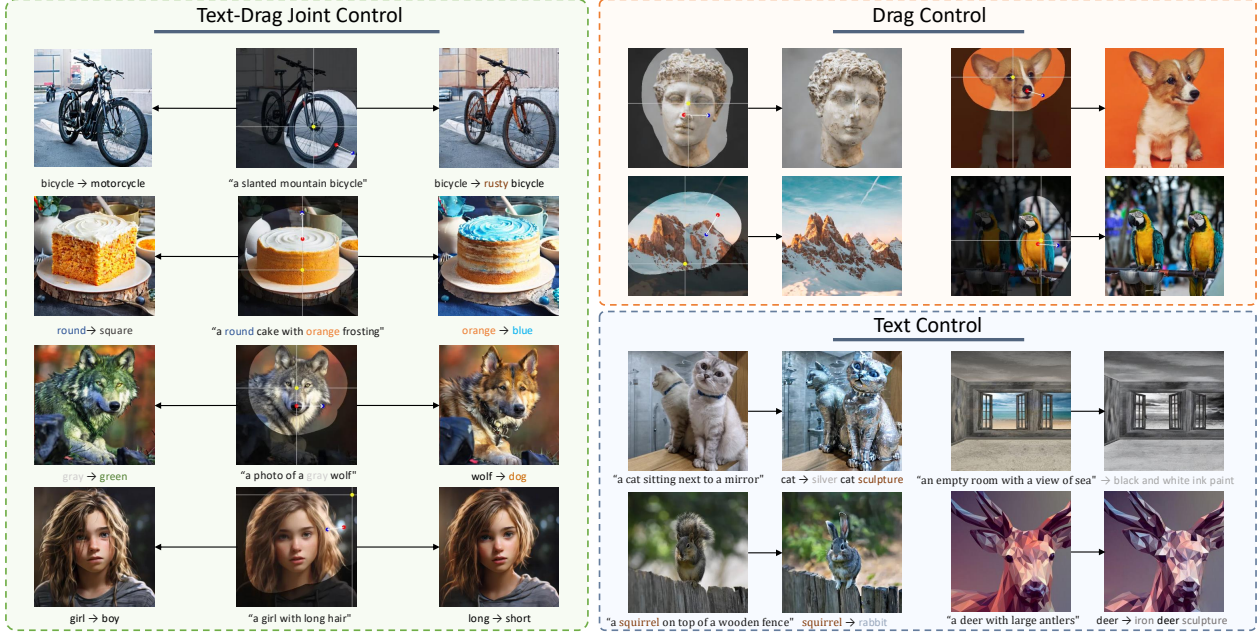


Figure 1. We present *TDEdit*, a unified diffusion-based framework for text-drag guided image editing. *TDEdit* achieves versatile manipulations through three distinct control modalities: joint text-drag guidance (left), exclusive drag-based deformation (top right), and pure textual control (bottom right).

often disrupts fine-grained layout control—a critical issue that demands resolution.

To address dynamic guidance balancing, we propose the Drag-Text Guided Denoising (DTGD) mechanism. DTGD adjusts conditional clues based on denoising steps. To balance the control signals, DTGD uses a three-branch architecture [33]: a source branch provides the original layout and details; a reference branch maintains global layout and injects target prompt semantics; and a target branch preserves the dragged region’s local layout while extracting details from the reference branch. During denoising, the target branch fuses semantic details from both branches using a dynamic factor for blended control.

To achieve precise layout control, we propose Point-Cloud Deterministic Drag (PCDD), a strategy for understanding drag intent using 3D-aware position representation. PCDD first projects 2D drag inputs into a 3D point cloud via depth-aware feature mapping. It then applies deterministic transformations to estimate post-drag pixel positions. The 2D coordinates are projected into 3D space, where rigid and non-rigid transformations are performed based on drag cues. Finally, the 3D coordinates are mapped back to 2D, adjusting latent features to align with the drag intent.

With the above designs as the main force, we finally develop our *TDEdit*: A Unified Diffusion Framework for Joint Text-Drag Guided Image Manipulation. In summary, we highlight the contributions of this paper as follows.

- We make an early exploration for a text-drag joint con-

trol framework for image editing, and present an unified framework *TDEdit* that is capable of simultaneously processing drag-based and text-based controls as well as the individual control capabilities of each modality.

- A Point-Cloud Deterministic Drag (PCDD) mechanism is proposed. PCDD distorts the latent feature via a 3D mapping to ensure the final layout of final image can precisely follow both conditions.
- To dynamically balance the control of text and drag instructions, we propose a Drag-Text Guided Denoising (DTGD) mechanism, offering a flexible trade-off between the text and drag conditions during denoising.

2. Related Work

Text-Based Image Editing Text-based image editing allows modifications via natural language prompts. Early influential works leveraged the latent space of Generative Adversarial Networks (GANs), such as StyleCLIP [25], to perform impressive edits guided by text. With the advent of diffusion models, the field has seen a surge in methods offering higher fidelity and flexibility. For instance, DiffusionCLIP [14] utilizes CLIP to fine-tune diffusion models for high-quality zero-shot edits across domains. Imagic [13] optimizes text embeddings and diffusion models for complex semantic changes, preserving image detail. Instruct-Pix2Pix [2] integrates a language model with a text-to-image model for fast instruction-based edits. Null-text Inver-

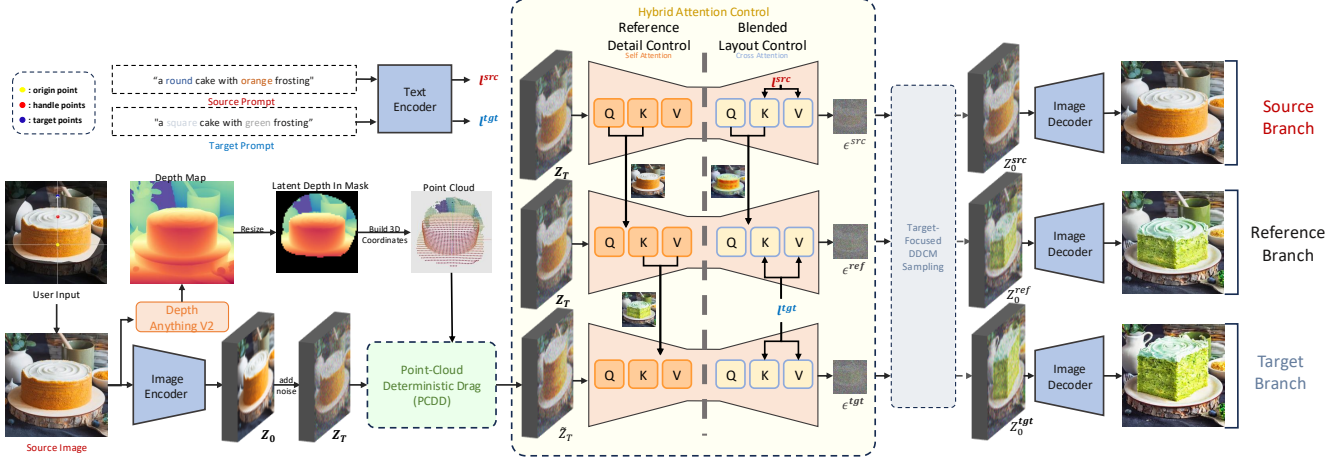


Figure 2. Illustration of our proposed TDEdit framework. The input image is first encoded via image encoder into latent feature. Next, PCDD utilizes the drag condition and estimated depth map to distort the latent features after noise addition. In the following, the distorted image latent and the original one as well as the prompts pass through DTGD with three branches to acquire the final latent feature, which is then fed into the image decoder to harvest the edited image.

sion [20] refines edits by optimizing null-text embeddings, avoiding weight adjustments. GLIDE [23] employs text-conditional diffusion for realistic editing. InfEdit [33] enhances control over intricate edits like texture using inverse techniques. Text-based methods excel in semantic flexibility but lack precise spatial control.

Drag-Based Image Editing Drag-based image editing enables precise control through user drag inputs. DragGAN [24] employs GANs with point tracking to adjust poses and shapes, though its quality trails diffusion models. DragDiffusion [28] adapts diffusion for point-based edits via latent optimization. FastDrag [37] speeds up drag edits, balancing efficiency and quality for real-time use. StableDrag [6] enhances diffusion-based motion edits for stability. DragonDiffusion [22] uses feature correspondences for object adjustments without fine-tuning. LucidDrag [5] and DragText [4] introduce text control to refine dragging, yet primarily serve drag precision rather than simultaneous semantic and pose editing. These methods achieve fine posture and layout tweaks but often sacrifice full text-driven semantic control, limiting dual editing capabilities.

3. Point-Cloud Deterministic Drag

Overview. The architecture of TDEdit is illustrated in Fig. 2. Following the paradigm of latent diffusion, we first encode the input image into a latent representation, while estimating its depth using Depth Anything v2 [34] to guide the subsequent PCDD. The PCDD module then warps the latent features using the rescaled depth and drag conditions to align with user intent. Finally, our DTGD denoiser refines the distorted latent using both original features and

source/target text prompts for high-fidelity generation.

The DTGD architecture follows InfEdit and includes three branches for precise control: (1) a source branch, which uses the source prompt and initial latent to steer the denoising process; (2) a reference branch, guided by the target prompt and initial latent, to generate semantic details while preserving the source layout; and (3) a target branch, which integrates drag information and details from the other branches using the target prompt and the warped latent from PCDD. DTGD leverages Hybrid Attention Control to manage inter-branch interactions during noise prediction, combining Blended Layout Control for global layout alignment and local drag retention, and Reference Detail Injection for maintaining semantic and detail fidelity. This process ultimately produces a denoised image that aligns with both the drag instructions and the target text description.

Before delving into the details of the denoising process, we first explain our core contribution for geometric control: the Point-Cloud Deterministic Drag (PCDD) module. Assume (x_s, y_s) is a position in the original feature $Z_T \in \mathbb{R}^{h \times w \times d}$; PCDD seeks to estimate its corresponding position after dragging, (x_t, y_t) , to warp the latent feature accordingly.

3.1. 3D Point Cloud Construction

To model complex deformations realistically, our first step is to transform the 2D problem into a 3D space.

Depth Map Normalization. For the original image I , we use the monocular depth estimation model Depth Anything V2 to obtain its corresponding depth map DP_I . As our operations occur in the latent space, this depth map is resized

to match the latent dimensions (h, w) and its values are normalized to a consistent scale, specifically to the range $[dp_{\min}, dp_{\max}]$ (default $[0, 63]$), using the following formula:

$$DP'_{Z_T} = dp_{\min} + \frac{\text{Resize}(DP_I, (h, w)) - DP_{\min}}{DP_{\max} - DP_{\min}} \cdot (dp_{\max} - dp_{\min}). \quad (1)$$

This allows us to lift every 2D point $(\hat{x}, \hat{y})$ within the user-provided *Mask* into a 3D point cloud $\hat{P}$ by assigning its depth value as the z-coordinate.

Local Coordinate System. To simplify subsequent calculations, we establish a local coordinate system centered at the object's origin $\hat{O}$. This origin is estimated from the mask's centroid and then calibrated with a slack distance d_O to better approximate the true center of rotation. All points p are converted to local coordinates via translation: $p_{\text{local}} = p_{\text{global}} - \hat{O}$.

Drag Subject Filtering. A critical step for precise editing is to isolate the target object. We achieve this by filtering the point cloud based on depth, assuming that points belonging to the same object have similar depth values. We define the set of movable points, P_{drag}^s , as those within a depth threshold d_{shield} of the primary handle point a_1 . This is formally defined as:

$$P_{\text{drag}}^s = \{p_i \in P \mid |z_{p_i} - z_{a_1}| \leq d_{\text{shield}}\}. \quad (2)$$

All other points within the mask, whose depth suggests they belong to a different object or the background, are considered static, P_{static} , and are excluded from transformations. This ensures the drag operation is precisely constrained to the intended region.

3.2. Hierarchical Drag Intention Modeling

Real-world manipulations often involve a combination of rigid body motion and local shape deformation. A key challenge when interpreting multiple drag instructions, e.g., $(a_1, b_1), (a_2, b_2), \dots$, is the inherent ambiguity in user intent. A naive approach, such as averaging the transformations implied by each pair, can lead to unpredictable or undesirable results, especially when instructions are conflicting.

To resolve this ambiguity and create an intuitive, predictable system, we propose a hierarchical interpretation of user intent. Our core design principle is to treat the first drag pair (a_1, b_1) as the representation of the user's primary, global intention, which dictates the rigid motion of the entire object. Subsequent drag pairs are then interpreted as secondary, local instructions for non-rigid deformation, refining the object's shape after its primary motion is complete. This hierarchical model offers several key advantages. It provides a predictable outcome by establishing a stable and unambiguous basis for the object's global motion. The

model also aligns with natural cognitive workflows, mirroring the coarse-to-fine process of human interaction. Finally, its design promotes algorithmic stability by elegantly decomposing the complex task into two more manageable sub-problems: a well-defined rigid body motion determined by a single vector, and a flexible non-rigid deformation field influenced by all control points. Following this principle, we implement a two-stage, coarse-to-fine hybrid drag mechanism that first establishes the object's global position based on (a_1, b_1) and then refines its local shape using all available instructions.

3.3. Hybrid-Rigid Drag

Following the hierarchical principle outlined above, our hybrid drag mechanism is implemented in two distinct stages: a coarse rigid transformation followed by a fine-grained non-rigid deformation.

Rigid Transformation. The first stage is a rigid transformation, which achieves the overall movement of the point cloud while maintaining geometric consistency. We use the primary user instruction (a_1, b_1) to define this transformation, which consists of a rotation followed by a translation. The rotation matrix R is constructed using Rodrigues' formula from the drag vector. The final position after this rigid step, P_{rigid} , serves as the basis for subsequent refinement and is obtained as:

$$P_{\text{rigid}} = R \cdot P_{\text{drag}}^s + \alpha \cdot (b_1 - R \cdot a_1). \quad (3)$$

Here, $(b_1 - R \cdot a_1)$ represents the translation vector V_{tra} , and α is a scaling factor that controls the effectiveness of the translation.

Non-Rigid Deformation. The second stage refines the object's shape based on all user instructions. We use Radial Basis Function (RBF) interpolation to model a smooth deformation field. First, we define a set of control points $C = A \cup F$, which includes both the user's handle points A (the "drivers" of the deformation) and a set of automatically determined internal fixed points F that act as structural "anchors" to preserve object integrity. The displacement vector $s(p)$ for any point p is then a weighted sum of influences from all control points, using a multiquadric kernel ϕ :

$$s(p) = \left(\sum_{c_k \in C} w_k^x \cdot \phi_{p,c_k}, \sum_{c_k \in C} w_k^y \cdot \phi_{p,c_k}, \sum_{c_k \in C} w_k^z \cdot \phi_{p,c_k} \right), \quad (4)$$

where the kernel function is defined as:

$$\phi_{p,q} = \sqrt{1 + (\mu \|p - q\|)^2}. \quad (5)$$

Here, μ is a shape parameter that controls the influence radius of the kernel. The unknown weights $w_k =$

(w_k^x, w_k^y, w_k^z) are determined by solving a system of linear equations derived from the known displacements at the control points. This step is crucial as it translates the user’s explicit instructions into a solvable mathematical system. Specifically, handle points must move towards their targets, while fixed points must remain stationary:

$$s_{c_i} = \begin{cases} b_i - a_i, & c_i \in A, \\ \mathbf{0}, & c_i \in F. \end{cases} \quad (6)$$

To ensure the deformation is localized and does not unrealistically affect distant parts of the object, we introduce a distance-based weight $\gamma(p)$. This function modulates the RBF field’s influence, making it strongest near handle points and weakest near fixed points, ensuring a natural falloff effect:

$$\gamma(p) = \frac{\min_{f_i \in F} \|p - f_i\|}{\min_{a_i \in A} \|p - a_i\| + \min_{f_i \in F} \|p - f_i\|}. \quad (7)$$

The final coordinates P_{drag}^t are obtained by additively blending the rigid motion with this localized non-rigid refinement. This combination is controlled by a non-rigid influence weight β :

$$P_{drag}^t = P_{rigid} + \beta \cdot \gamma(P_{rigid}) \cdot s(P_{rigid}), \quad (8)$$

where $V_{non-rigid} = \gamma(P_{rigid}) \cdot s(P_{rigid})$.

3.4. Feature Mapping and Interpolation

With the final 3D coordinates P_{drag}^t computed, we must apply this transformation back to the 2D latent feature map $\tilde{Z}_T$.

3D-to-2D Projection. First, the continuous 3D coordinates are discretized by rounding the x and y components to align with the latent grid:

$$\hat{P}_{drag}^t = \{(\text{round}(x_t), \text{round}(y_t), z_t) | (x_t, y_t, z_t) \in P_{drag}^t\}. \quad (9)$$

A key challenge in this 3D-to-2D projection is handling self-occlusion. To resolve this, we employ a ”z-buffering” strategy: if multiple 3D points map to the same 2D location, only the one with the maximum z-value (i.e., the one in the foreground) is retained. This process results in a sparse set of valid target points. Let $\text{Inv} : (x_t, y_t) \mapsto (x_s, y_s)$ denote the inverse mapping that retrieves the source 2D coordinates (x_s, y_s) for a given target 2D coordinate (x_t, y_t) within the dragged region. The final warped latent map $\tilde{Z}_T$ is then constructed by combining three types of features: (1) features of dragged points, which are transferred from their corresponding source locations in Z_T ; (2) features of static points within the mask, which are copied directly; and (3) features of points outside the mask, which also remain

unchanged. This comprehensive logic is formally expressed as:

$$\tilde{Z}_T(x, y) = \begin{cases} Z_T(\text{Inv}(x, y)), & \text{if } (x, y) \text{ is a dragged point,} \\ Z_T(x, y), & \text{otherwise.} \end{cases} \quad (10)$$

Interpolation of Featureless Points. The direct feature transfer and z-buffering in the previous step inevitably creates voids or ”holes” in the warped latent map. To ensure a continuous and semantically coherent latent representation for the subsequent denoising stage, we fill these featureless regions using Bidirectional Nearest Neighbor Interpolation (BNNI), as proposed by FastDrag. For any empty-feature point (x_N, y_N) , its feature is interpolated as a weighted average of its four nearest neighbors (up, down, left, right):

$$\tilde{Z}_T(x_N, y_N) = \sum_{loc=u,r,d,l} w_{loc} \cdot \tilde{Z}_T(x_{loc}, y_{loc}), \quad (11)$$

where the weights w_{loc} are inversely proportional to the distance to each neighbor, giving closer points more influence:

$$w_{loc} = \frac{1/\text{len}_{loc}}{\sum_{loc=u,r,d,l} 1/\text{len}_{loc}}. \quad (12)$$

4. Drag-Text Guided Denoising

The PCDD module injects drag guidance into the warped latent feature $\tilde{Z}_T$. Our DTGD module then integrates this distorted representation with text conditioning through attention map manipulation in the denoising UNet (as shown in Fig. 2), adaptively fusing both conditions to progressively denoise the latent space and generate the final output image.

4.1. Hybrid Attention Control

Blended Layout Control. In our practice, a key challenge in combining drag-based and text-based methods lies in layout control. We take three branches and exchange information among three branches via attention maps to solve the problem. Formally, M^{src} , M^{ref} , M^{tgt} are the attention map from the denoising UNet in source, reference and target branches, respectively, where the image always serves the key and value, and the source prompt (source branch) and target prompt (reference and target branches) act as the query.

Subsequently, we endow the reference branch with a similar layout to the source one via an attention replacement. Formally, when the j -th token in the target prompt matches the i -th token in the source prompt, we replace the j -th row of M^{ref} with the corresponding i -th row from M^{src} . The process is as:

$$\text{Replace}(M^{src}, M^{ref})_j = \begin{cases} M_i^{src} & \text{if } \mathcal{A}(j) = i, \\ M_j^{ref} & \text{if } \mathcal{A}(j) = \text{None.} \end{cases} \quad (13)$$

where $\mathcal{A}(j) = i$ means the j -th token in target prompt matches the i -th token in source prompt. To form the rough layout in early timesteps and detailed semantic layout in late timesteps, we set a timestep threshold t_c to control the degree of replacement in the cross-attention process of noise prediction:

$$M^{ref} = \begin{cases} \text{Replace}(M^{src}, M^{ref}) & \text{if } t \geq t_c, \\ M^{ref} & \text{if } t < t_c. \end{cases} \quad (14)$$

This modified attention map, M^{ref} , is then utilized within the UNet’s attention layers during the noise prediction step for the reference branch, contributing to the generation of Z_t^{ref} . Subsequently, to ensure the target branch maintains a similar layout in early timesteps without compromising the dragged region, we strategically fuse features:

$$Z_t^{tgt} = Z_t^{tgt} \odot Mask + Z_t^{ref} \odot (1 - Mask). \quad (15)$$

Reference Detail Injection. To introduce detail from the source branch, when performing cross-attention, we replace (Q^{ref}, K^{ref}) by (Q^{src}, K^{src}) between reference and source branches in the early timesteps until t_s to prevent excessive detail, and we use (K^{ref}, V^{ref}) to replace (K^{tgt}, V^{tgt}) to provide reference throughout the whole process. The detailed formulation is:

$$\begin{aligned} (Q^{ref}, K^{ref}, V^{ref}) &\leftarrow \begin{cases} (Q^{src}, K^{src}, V^{ref}) & \text{if } t \geq t_s, \\ (Q^{ref}, K^{ref}, V^{ref}) & \text{if } t < t_s, \end{cases} \\ (Q^{tgt}, K^{tgt}, V^{tgt}) &\leftarrow (Q^{tgt}, K^{ref}, V^{ref}). \end{aligned} \quad (16)$$

Subsequently, let l^{src} and l^{tgt} be the prompt feature encoded from the language encoder. We can get the noise of three branches from the noise prediction network:

$$\varepsilon_{src} = \varepsilon_\theta(Z_t^{src}, t, l^{src}), \quad (17)$$

$$\varepsilon_{ref} = \varepsilon_\theta(Z_t^{ref}, t, l^{tgt}), \quad (18)$$

$$\varepsilon_{tgt} = \varepsilon_\theta(Z_t^{tgt}, t, l^{tgt}). \quad (19)$$

In the following, we use Target-focused DDCM sampling to denoise and get the final target image.

4.2. Target-focused DDCM Sampling

The DDCM sampling formula is a modified DDIM approach for inversion-free image editing, ensuring path consistency between source (Z_t^{src}) and target (Z_t^{tgt}) branches during denoising:

$$\begin{aligned} Z_{t-1}^{tgt} &= \sqrt{\alpha_{t-1}} \left(\frac{Z_t^{tgt} - \sqrt{1 - \alpha_t} \varepsilon}{\sqrt{\alpha_t}} \right) \quad (\text{predicted } Z_0) \\ &+ \sqrt{1 - \alpha_{t-1} - \sigma_t^2} \cdot \varepsilon \quad (\text{direction to } Z_t^{tgt}) \\ &+ \sigma_t \varepsilon_t, \quad \text{where } \varepsilon_t \sim \mathcal{N}(0, \mathbf{I}) \quad (\text{random noise}), \end{aligned} \quad (20)$$

where the second term $\sqrt{1 - \alpha_{t-1} - \sigma_t^2} \cdot \varepsilon$ represents the direction toward Z_t^{tgt} . The combined noise $\varepsilon = (\varepsilon_{tgt} - \varepsilon_{src} + \varepsilon_{cons})$ reflects the source branch’s influence on the target branch. The consistency noise ε_{cons} is derived from the source branch’s latent Z_t^{src} and the initial input Z_0 , enforcing structural alignment, and is formally defined as:

$$\varepsilon_{cons} = \frac{Z_t^{src} - \sqrt{\alpha_t} Z_0}{\sqrt{1 - \alpha_t}}. \quad (21)$$

The term ε ensures the target branch adapts to editing while staying consistent with the source.

The standard DDCM eliminates the second term by setting $\sigma_t = \sqrt{1 - \alpha_{t-1}}$. However, this strategy weakens the dependence on Z_t^{tgt} , causing imprecise layout control from the drag condition. To remedy this issue, we introduce a factor η to adjust the dependency on Z_t^{tgt} , setting $\sigma_t = \eta \sqrt{1 - \alpha_{t-1}}$. Consequently, our denoising formula reads:

$$\begin{aligned} Z_{t-1}^{tgt} &= \sqrt{\alpha_{t-1}} \left(\frac{Z_t^{tgt} - \sqrt{1 - \alpha_t} \varepsilon}{\sqrt{\alpha_t}} \right) \quad (\text{predicted } Z_0) \\ &+ \sqrt{(1 - \eta^2)(1 - \alpha_{t-1})} \cdot \varepsilon \quad (\text{direction to } Z_t^{tgt}) \\ &+ \sigma_t \varepsilon_t, \quad \text{where } \varepsilon_t \sim \mathcal{N}(0, \mathbf{I}) \quad (\text{random noise}). \end{aligned} \quad (22)$$

To reserve the dragged layout, prevent it from being treated as noise and overly corrected, we set the noise control parameter $\eta < 1$ to enhance dependence on Z_t^{tgt} , and reduce random noise to further optimize the denoising effect. We implement a dynamic adjustment strategy for η .

Dynamically adjust η . We adjust η based on the progress of the current timestep t , the strategy is as:

$$\eta = \begin{cases} 0.5 & \text{if } t/T < 0.3, \\ 0.5 + 0.4 \cdot \frac{t/T - 0.3}{0.4} & \text{if } 0.3 \leq t/T \leq 0.7, \\ 0.9 & \text{if } t/T > 0.7, \end{cases} \quad (23)$$

where t/T represents the progress in the total denoising steps. This dynamic adjustment strategy is divided into three stages. In the early stage ($t/T < 0.3$), η is set to 0.5 to preserve more drag-induced details. In the middle stage ($0.3 \leq t/T \leq 0.7$), η increases linearly from 0.5 to 0.9, gradually introducing more random noise to smooth the denoising process. In the final stage ($t/T > 0.7$), η is held at 0.9 to maintain a sufficient level of randomness, which is crucial for ensuring the smoothness and detail consistency of the final generated image. These specific thresholds and values were determined empirically to achieve a robust balance between geometric fidelity and semantic quality across a wide range of editing tasks.

Both the target and reference branches are progressively denoised using this Target-focused DDCM sampling process. Ultimately, the fully denoised target latent, Z_0^{tgt} , is

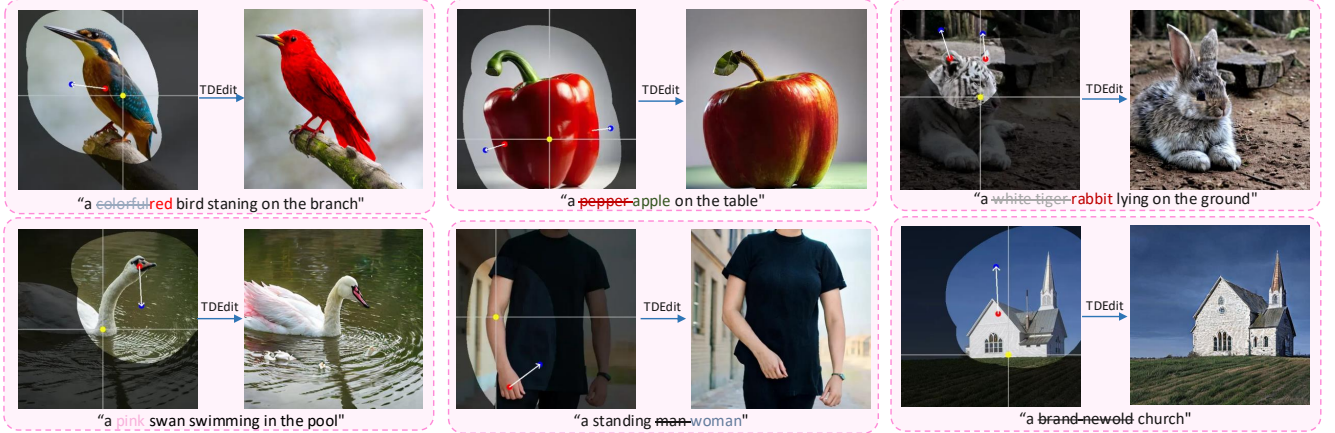


Figure 3. Results of Text-Drag joint editing with our TDEdit. The final images clearly follow both conditions and are authentic enough, revealing TDEdit well balances both control signals and synthesize plausible results.

decoded by the VAE to yield the final, high-fidelity edited image.

5. Experiments

5.1. Implementation Details

Experimental Setup. Our framework is built upon a pre-trained Latent Consistency Model (LCM), specifically the LCM Dreamshaper v7 checkpoint [17]. All experiments, including timing benchmarks, were conducted on a single NVIDIA RTX 4090 GPU. For the diffusion process, we perform editing over 10 effective denoising steps, derived from 15 total inference steps with an inversion strength of 0.7. All images are processed at a resolution of 512×512 pixels.

Denoising Process Hyperparameters. We utilize Classifier-Free Guidance (CFG) [10], setting the scale to 1.0 for the source branch and 2.0 for the target branch to balance fidelity and edit strength. The reference branch guidance is adaptively set to 1.0 or 2.0 depending on the edit complexity. For our attention control mechanisms, Blended Layout Control is active for the first 3 denoising steps, while for Reference Detail Injection, the initial 5 steps prioritize source feature integration. In our Target-focused DDCM Sampling, the noise scaling parameter η is fixed to 1.0 for the source and reference branches, while the target branch’s η is dynamically adjusted from 0.5 to 0.9 as detailed in Sec. 4.2. To protect the background, we fuse target features (inside the mask) with reference features (outside the mask) for the first 4 denoising steps.

PCDD Module Hyperparameters. The origin for rotation and deformation is set as the mask’s centroid. The default slack distance d_O used to calibrate the origin is set to 20, and the shield distance d_{shield} for isolating the drag

subject is 30. The influence weights for rigid transformation (α) and non-rigid deformation (β) are both set to 0.7.

5.2. User Study

The novel task of text-drag joint editing presents a unique evaluation challenge, as no established benchmarks currently exist to quantitatively measure the synergistic performance of both modalities. Therefore, to directly assess our framework’s effectiveness and user-perceived quality, we conducted a comprehensive user study. The study involved 23 participants recruited from university students and researchers familiar with generative AI tools. It was systematically divided into three parts to evaluate TDEdit’s performance in its core scenario of text-drag joint editing, and to benchmark its competitiveness against specialized methods in drag-only and text-only editing scenarios.

5.2.1. Text-Drag Joint Editing Evaluation

In this section, we evaluated TDEdit’s ability to handle and harmonize simultaneous text and drag instructions. Participants were shown 25 joint editing results generated by TDEdit and were asked to rate each output across four key dimensions on a 5-point Likert scale (1 = Very Poor, 5 = Very Good). The mean scores, standard deviations, and 95% confidence intervals (CI) are summarized in Table 1.

Analysis: The results in Table 1 demonstrate that TDEdit received consistently high scores across all evaluation dimensions, with all mean scores exceeding 4.3. Notably, it achieved a high score of **4.42** on ”Joint Adherence,” the most critical metric for our unified framework. The tight 95% confidence interval of [4.31, 4.53] for this metric strongly indicates that our framework reliably performs synergistic editing. Furthermore, the high ”Image Quality” score of 4.34 confirms that TDEdit produces visually plausible and

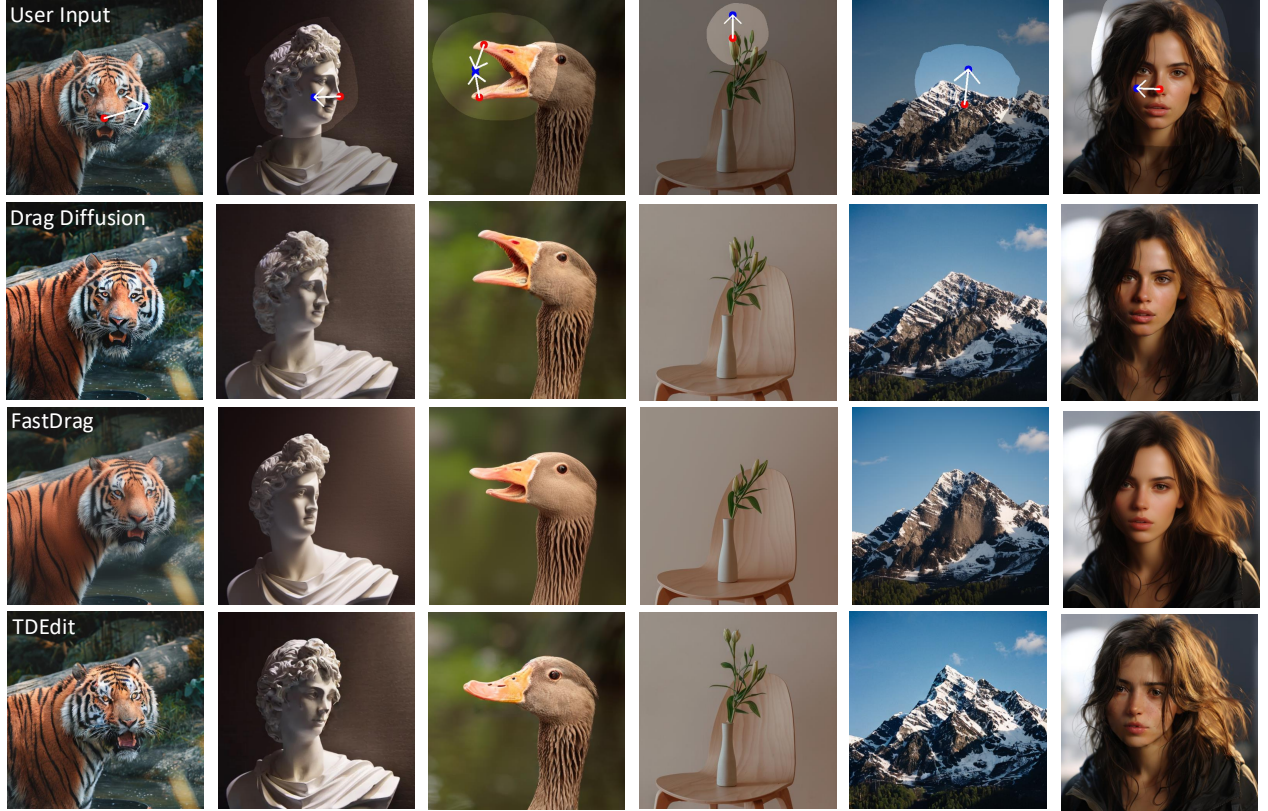


Figure 4. Qualitative comparison regarding drag control. In comparison with other models, TDEdit achieves more precise layout drag, and the results of TDEdit well follows the drag intention.

high-fidelity results.

5.2.2. Drag-Only Editing Comparison

We conducted a head-to-head comparison against two leading specialized methods: FastDrag [37] and DragDiffusion [28]. Participants were asked a forced-choice question: "Which result best follows the drag instruction while maintaining the highest image quality?". The results are aggregated in Table 2, where Preference Rate (Pref.) indicates the percentage of times a method was chosen as the best.

Analysis: As shown in Table 2, TDEdit emerged as the most preferred method (40.6%). While its lead over FastDrag (38.3%) is narrow, it significantly surpasses the classic DragDiffusion (21.1%), with their 95% CIs being clearly disjoint. This provides strong evidence that TDEdit is highly competitive in its individual modalities.

5.2.3. Text-Only Editing Comparison

Finally, to evaluate how our modifications for drag-editing impact text-only performance, we conducted a crucial comparison between TDEdit and its architectural foundation, InfEdit [33]. As InfEdit is a powerful, specialized method for text-driven editing, this head-to-head evaluation assesses

Table 1. User study scores for TDEdit on joint editing tasks. Scores are averaged over 23 users for 25 editing cases. Higher is better (out of 5).

Dimension	Score	Std. Dev.	95% CI
Text Adherence	4.51	0.81	[4.41, 4.61]
Drag Adherence	4.49	0.86	[4.38, 4.60]
Joint Adherence	4.42	0.90	[4.31, 4.53]
Image Quality	4.34	0.92	[4.23, 4.45]

whether our unified framework successfully retains (or even enhances) the strong text-editing capabilities of the base model. Participants were asked to choose which of the two results better reflected the textual edit while preserving higher overall quality. The user preference statistics are presented in Table 3.

Analysis: The data in Table 3 clearly indicates that TDEdit was preferred by a significant margin (63.0%) with non-overlapping 95% confidence intervals confirming this preference is statistically significant. This result is particularly insightful when considered alongside the quantitative met-

Table 2. User preference rates (%) in the drag-only editing comparison. Total votes were collected from 23 users over 15 cases (345 total votes).

Method	Total Votes	Pref.	95% CI
TDEdit (Ours)	140	40.6	[35.4%, 45.8%]
FastDrag	132	38.3	[33.1%, 43.5%]
DragDiffusion	73	21.1	[21.1%, 25.6%]

Table 3. User preference rates (%) in the text-only editing comparison. Total votes were collected from 23 users over 10 cases (230 total votes).

Method	Total Votes	Pref.	95% CI
TDEdit (Ours)	145	63.0	[56.7%, 69.3%]
InfEdit	85	37.0	[30.7%, 43.3%]

rics in Table 4, where InfEdit shows advantages in fidelity metrics like LPIPS and MSE. We hypothesize that this divergence between user preference and quantitative scores stems from the architectural differences between the two models. TDEdit’s dynamic multi-branch attention mechanism appears to be more responsive to Classifier-Free Guidance (CFG), effectively amplifying the strength of the textual edit to produce more visually distinct and impactful results. Conversely, InfEdit’s layout branch, which is designed to preserve source features for consistency, may inadvertently constrain the full extent of the edit, leading to results that are higher in pixel-level fidelity but sometimes less semantically aligned with the target prompt. Therefore, the strong user preference suggests that while TDEdit and InfEdit exhibit a trade-off between edit strength and detail preservation, the performance gap in text-editing is not perceived as significant by users. In fact, users tend to favor the more pronounced and semantically clear results generated by our framework. This validates that our unified model remains highly competitive in text-only tasks, offering a compelling alternative to its specialized base model.

5.3. Qualitative Comparison

Text-Drag Joint Editing. As an early exploration for text-drag joint control, no proper models can serve as the comparison methods in this setting. Fig. 3 shows the results of our TDEdit with both conditions. We can observe that the results can well follow both control signals, achieving the goals of this paper.

Text-Based Editing. In the text task, TDEdit outperforms InfEdit in certain aspects. Although TDEdit sacrifices some source branch details for dragging, the quality difference is minor in most cases. As shown in Fig. 5, InfEdit’s layout

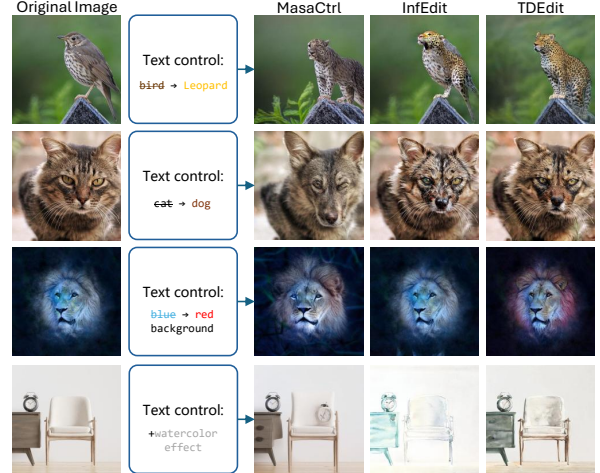


Figure 5. Qualitative comparison with InfEdit.

branch may retain excessive source features during tasks like replacement, color, and style, hindering alignment with target semantics. TDEdit achieves better generation by incorporating more target semantic information.

Drag-Based Editing. In drag tasks, TDEdit surpasses DragDiffusion and FastDrag. DragDiffusion’s n-step optimization causes uncertainty and artifacts, while FastDrag’s one-step mapping lacks realism for precise edits. TDEdit’s 3D feature mapping enables realistic transformations like rotation/stretching (Fig. 4). For details, DragDiffusion and FastDrag lose texture fidelity due to over-smoothing. TDEdit maintains details through attention control and text injection, ensuring semantic consistency.

5.4. Quantitative Comparison

Text-Based Editing. We evaluate TDEdit on PIE-Bench [12] using a range of standard metrics, including Mean Squared Error (MSE), Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM) [30], Learned Perceptual Image Patch Similarity (LPIPS) [35], a perceptual Distance [9], and CLIP scores for the whole image (CLIP-W) and the edit region (CLIP-E) [8]. The results are presented in Table 4. TDEdit shows a slightly higher Distance, LPIPS, and MSE compared to some baselines. This decline stems from our emphasis on supporting drag-based editing. Nevertheless, the impact remains minimal, suggesting editing quality is not significantly compromised.

Drag-Based Editing. We conduct quantitative experiments on DragBench [28], where MD [24] measures drag precision, and IF [13] (1-LPIPS [35]) assesses similarity. "Editing Time" is the total end-to-end duration per instruction. As presented in Table 5, our TDEdit demonstrates

Table 4. Quantitative comparison on the PIE-Bench dataset. For a fair comparison, performance of all baselines is sourced from the InfEdit paper [33]. Notably, our TDEdit, InfEdit, and MasaCtrl employ distinct editing mechanisms, while the other baselines are all based on the Prompt-to-Prompt (P2P) [7] framework for attention editing. For brevity, values for Distance, LPIPS, MSE, and SSIM are scaled by 10^3 , 10^3 , 10^4 , and 10^2 respectively.

Approach	Distance ↓	PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑	CLIP-W ↑	CLIP-E ↑
DDIM [29]	69.43	17.87	208.80	219.88	71.14	25.01	22.44
CycleD [32]	6.06	28.25	43.96	25.85	85.61	23.68	20.87
NT [20]	13.44	27.03	60.67	35.86	84.11	24.75	21.86
NP [19]	16.17	26.21	69.01	39.73	83.40	24.61	21.87
StyleD [31]	11.65	26.05	66.10	38.63	83.42	24.78	21.72
DI [12]	11.65	27.22	54.55	32.86	84.76	25.02	22.10
MasaCtrl [3]	40.45	18.82	184.90	155.20	67.29	24.00	21.49
InfEdit [33]	13.78	28.51	47.58	32.09	85.66	25.03	22.22
TDEdit (Ours)	15.09	25.66	68.77	102.91	83.58	24.81	21.90

Table 5. Quantitative comparison on DragBench.

Approach	MD ↓	IF ↑	Time(s)
DragDiffusion [28]	33.70	0.89	21.54
DragNoise [16]	33.41	0.63	20.41
FreeDrag [15]	35.00	0.70	52.63
GoodDrag [36]	22.96	0.86	45.83
DiffEditor [21]	28.46	0.89	21.68
FastDrag [37]	32.23	0.86	2.80
TDEdit (Ours)	26.91	0.92	2.52

a superior balance of precision, quality, and efficiency. It achieves a competitive Mean Distance (MD) of 26.91, significantly outperforming mainstream methods like DragDiffusion (33.70) and the efficiency-focused FastDrag (32.23). While GoodDrag obtains the lowest MD, its runtime is substantially higher. In contrast, our method not only achieves the highest Image Fidelity (IF) score of 0.92 among all baselines but also operates at the fastest speed of just 2.52 seconds. This combination of strong precision, state-of-the-art image quality, and real-time performance establishes TDEdit as a highly effective and practical solution for drag-based editing.

5.5. Ablation Study

We conduct ablation experiments on Hybrid Attention Control and η on DragBench. Table 6 shows that Blended Layout Control (BLC) has some impact, while Reference Detail Control has a greater impact on MD and IF, showing the importance of the Reference branch. Our analysis of η (Fig. 6) reveals a trade-off between image quality and dragging ease, validating our dynamic adjustment strategy.

Table 6. Ablation Study on Hybrid Attention Control.

Approach	MD ↓	IF ↑	CLIP Sim ↑
TDEdit w/o BLC & RDI	32.14	0.67	0.880
TDEdit w/o BLC	26.82	0.89	0.972
TDEdit w/o RDI	31.12	0.83	0.939
TDEdit (Full)	25.89	0.92	0.977

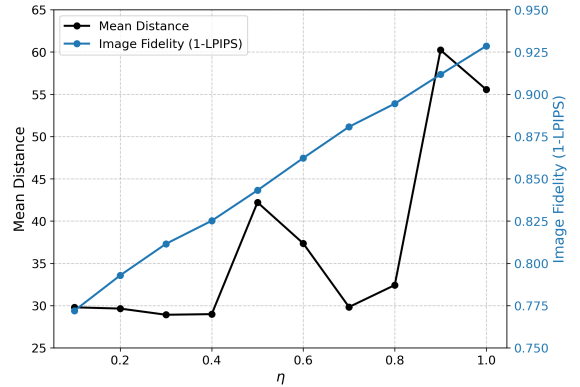


Figure 6. Illustration of Ablation Study on η .

6. Limitations

Despite the promising performance of TDEdit as a unified framework, we acknowledge several limitations that offer avenues for future research.

First, the performance of our Point-Cloud Deterministic Drag (PCDD) module can be sensitive to its hyperparameters. Unlike purely 2D drag methods that often only require handle and target points, our 3D-aware approach necessitates adjustments to parameters such as the slack distance (d_O) and shield distance (d_{shield}) to achieve optimal results for objects with varying shapes and depths. This introduces a trade-off: while the 3D representation enables more real-

istic, physically plausible deformations, it currently requires more user expertise to tune for the best effect compared to simpler 2D paradigms.

Second, artifacts can emerge in regions with extensive feature voids. These voids are an inherent consequence of the discrete latent relocation process in drag operations, particularly during large-scale displacements. While our Bidirectional Nearest Neighbor Interpolation (BNNI) strategy effectively fills small gaps by leveraging local neighborhood information, its reconstruction capabilities are limited when confronted with large, contiguous null regions. In such cases, the interpolated features may lack global semantic coherence, leading to visual artifacts or a loss of texture fidelity, as the local nature of BNNI may not fully reconstruct the complex structure of the missing area.

Finally, as our framework builds upon and modifies the architecture of InfEdit [33], it inherits some of its inherent limitations, such as occasional unintended semantic changes in non-target regions. More importantly, to seamlessly integrate drag-based control, we made a critical adaptation to the original three-branch architecture. Specifically, we repurposed InfEdit’s original layout branch, which was dedicated to fine-grained pose and layout control, into our reference branch, primarily tasked with injecting semantic details from the target text. Concurrently, the core responsibility of our target branch shifted to anchoring and preserving the spatial layout dictated by the drag operation. In InfEdit’s original design, the three branches created a delicate balance highly optimized for pure text-driven tasks, where the layout branch was crucial for precisely preserving and adjusting object poses without disturbing non-edited regions. This architectural trade-off, while fundamental to achieving our unified text-drag framework, inevitably disrupts this finely-tuned equilibrium. Specifically, the target latent Z_t^{tgt} is now primarily constrained to faithfully reflect the post-drag structure computed by our PCDD module. This strong constraint can diminish its capacity to freely blend and refine nuanced pose details from the reference latent Z_t^{ref} , particularly in non-dragged regions such as the background or secondary objects. This architectural trade-off explains the slight performance drop observed in our quantitative text-editing benchmarks (Table 4) when compared to the specialized InfEdit model. Addressing these challenges, for instance by developing more sophisticated feature fusion mechanisms or exploring more seamless architectural integrations, remains a key direction for future research.

7. Conclusion

This paper introduces a unified diffusion-based framework for image editing that combines text and drag interactions to overcome the limitations of existing methods. While text-driven editing excels in texture manipulation but lacks spatial precision, drag-based editing controls shape/structure but

misses texture guidance. The proposed method integrates both. Key innovations include: Point-Cloud Deterministic Drag: Improves spatial control via 3D feature mapping, and Drag-Text Guided Denoising: Dynamically balances text and drag conditions during denoising. The framework supports text-only, drag-only, or combined editing modes while maintaining high fidelity. Experiments show it outperforms or matches specialized single-mode methods, offering a versatile solution for controllable image manipulation.

References

- [1] Omer Bar-Tal, Dolev Ofri-Amar, Rafail Fridman, Yoni Kasten, and Tali Dekel. Text2live: Text-driven layered image and video editing. In *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XV*, pages 707–723. Springer, 2022. 1
- [2] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 18392–18402. IEEE, 2023. 2
- [3] 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 *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 22503–22513. IEEE, 2023. 10
- [4] Gayoon Choi, Taejin Jeong, Sujung Hong, Jaehoon Joo, and Seong Jae Hwang. Dragtext: Rethinking text embedding in point-based image editing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 8488–8497. IEEE, 2024. 3
- [5] Xing Cui, Peipei Li, Zekun Li, Xuannan Liu, Yueying Zou, and Zhaofeng He. Localize, understand, collaborate: Semantic-aware dragging via intention reasoner. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. 3
- [6] Yutao Cui, Xiaotong Zhao, Guozhen Zhang, Shengming Cao, Kai Ma, and Limin Wang. Stabledrag: Stable dragging for point-based image editing. In *Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part LVIII*, pages 340–356. Springer, 2024. 3
- [7] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross-attention control. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. 1, 10
- [8] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*,

- EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021, pages 7514–7528. Association for Computational Linguistics, 2021. [9](#)
- [9] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6626–6637, 2017. [9](#)
- [10] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *CoRR*, abs/2207.12598, 2022. [7](#)
- [11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. [1](#)
- [12] Xuan Ju, Ailing Zeng, Yuxuan Bian, Shaoteng Liu, and Qiang Xu. Direct inversion: Boosting diffusion-based editing with 3 lines of code. *CoRR*, abs/2310.01506, 2023. [9](#), [10](#)
- [13] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 6007–6017. IEEE, 2023. [2](#), [9](#)
- [14] Gwanghyun Kim, Taesung Kwon, and Jong Chul Ye. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 2416–2425. IEEE, 2022. [2](#)
- [15] Pengyang Ling, Lin Chen, Pan Zhang, Huaian Chen, Yi Jin, and Jinjin Zheng. Freedrag: Feature dragging for reliable point-based image editing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 6860–6870. IEEE, 2024. [1](#), [10](#)
- [16] Haofeng Liu, Chenshu Xu, Yifei Yang, Lihua Zeng, and Shengfeng He. Drag your noise: Interactive point-based editing via diffusion semantic propagation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 6743–6752. IEEE, 2024. [1](#), [10](#)
- [17] 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. [7](#)
- [18] 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 *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. [1](#)
- [19] Daiki Miyake, Akihiro Iohara, Yu Saito, and Toshiyuki Tanaka. Negative-prompt inversion: Fast image inversion for editing with text-guided diffusion models. *CoRR*, abs/2305.16807, 2023. [10](#)
- [20] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 6038–6047. IEEE, 2023. [3](#), [10](#)
- [21] Chong Mou, Xintao Wang, Jiechong Song, Ying Shan, and Jian Zhang. Diffeditor: Boosting accuracy and flexibility on diffusion-based image editing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 8488–8497. IEEE, 2024. [1](#), [10](#)
- [22] Chong Mou, Xintao Wang, Jiechong Song, Ying Shan, and Jian Zhang. Dragondiffusion: Enabling drag-style manipulation on diffusion models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. [3](#)
- [23] Alexander Quinn Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. GLIDE: towards photorealistic image generation and editing with text-guided diffusion models. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, pages 16784–16804. PMLR, 2022. [3](#)
- [24] Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimithra Meka, and Christian Theobalt. Drag your GAN: interactive point-based manipulation on the generative image manifold. In *ACM SIGGRAPH 2023 Conference Proceedings, SIGGRAPH 2023, Los Angeles, CA, USA, August 6-10, 2023*, pages 78:1–78:11. ACM, 2023. [1](#), [3](#), [9](#)
- [25] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2085–2094, 2021. [2](#)
- [26] 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 *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, pages 8748–8763. PMLR, 2021. [1](#)
- [27] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 10674–10685. IEEE, 2022. [1](#)
- [28] Yujun Shi, Chuhui Xue, Jun Hao Liew, Jiachun Pan, Han-shu Yan, Wenqing Zhang, Vincent Y. F. Tan, and Song Bai. Dragdiffusion: Harnessing diffusion models for interactive point-based image editing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 8839–8849. IEEE, 2024. [1](#), [3](#), [8](#), [9](#), [10](#)
- [29] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *9th International Confer-*

- ence on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net, 2021. [1](#), [10](#)
- [30] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4): 600–612, 2004. [9](#)
- [31] Zhizhong Wang, Lei Zhao, and Wei Xing. Stylediffusion: Controllable disentangled style transfer via diffusion models. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 7643–7655. IEEE, 2023. [10](#)
- [32] Chen Henry Wu and Fernando De la Torre. A latent space of stochastic diffusion models for zero-shot image editing and guidance. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 7344–7353. IEEE, 2023. [1](#), [10](#)
- [33] Sihan Xu, Yidong Huang, Jiayi Pan, Ziqiao Ma, and Joyce Chai. Inversion-free image editing with natural language. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, New Orleans, LA, USA, June 18-24, 2024*, pages 10674–10685. IEEE, 2024. [1](#), [2](#), [3](#), [8](#), [10](#), [11](#)
- [34] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything V2. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. [3](#)
- [35] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 586–595. Computer Vision Foundation / IEEE Computer Society, 2018. [9](#)
- [36] Zewei Zhang, Huan Liu, Jun Chen, and Xiangyu Xu. Good-drag: Towards good practices for drag editing with diffusion models. *CoRR*, abs/2404.07206, 2024. [1](#), [10](#)
- [37] Xuanjia Zhao, Jian Guan, Congyi Fan, Dongli Xu, Youtian Lin, Haiwei Pan, and Pengming Feng. Fastdrag: Manipulate anything in one step. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. [1](#), [3](#), [8](#), [10](#)