DragNeXt: Rethinking Drag-Based Image Editing

Yuan Zhou[1], Junbao Zhou[1], Qingshan Xu[1]*, Kesen Zhao[1], Yuxuan Wang[1], Hao Fei[2], Richang Hong[3], Hanwang Zhang[1]

[1]Nanyang Technological University
[2]National University of Singapore
[3]Hefei University of Technology

Abstract

Drag-Based Image Editing (DBIE), which allows users to manipulate images by directly dragging objects within them, has recently attracted much attention from the community. However, it faces two key challenges: (*i*) point-based drag is often highly ambiguous and difficult to align with user intentions; (*ii*) current DBIE methods primarily rely on alternating between motion supervision and point tracking, which is not only cumbersome but also fails to produce high-quality results. These limitations motivate us to explore DBIE from a new perspective—unifying it as a Latent Region Optimization (LRO) problem that aims to use region-level geometric transformations to optimize latent code to realize drag manipulation. Thus, by specifying the areas and types of geometric transformations, we can effectively address the ambiguity issue. We also propose a simple yet effective editing framework, dubbed **DragNeXt**. It solves LRO through Progressive Backward Self-Intervention (PBSI), simplifying the overall procedure of the alternating workflow while further enhancing quality by fully leveraging region-level structure information and progressive guidance from intermediate drag states. We validate **DragNeXt** on our NextBench, and extensive experiments demonstrate that our proposed method can significantly outperform existing approaches.

Code — https://github.com/zhouyuan888888/DragNeXt
Extended version — https://arxiv.org/pdf/2506.07611

Introduction

Diffusion models (Rombach et al. 2022; Dhariwal and Nichol 2021) have made remarkable progress in the field of text-to-image generation, serving as foundational models for a wide range of generative tasks, such as image super-resolution (Wu et al. 2024; Sun et al. 2024), style transfer (Zhang et al. 2023; Chung, Hyun, and Heo 2024), text-based image editing (Brooks, Holynski, and Efros 2023; Hertz et al. 2022), to name but a few. Nevertheless, an inherent limitation of diffusion models lies in their poor controllability, which brings more challenges to fine-grained editing tasks, especially those that require interactive-style manipulation (Liu et al. 2024).

Recent studies (Shi et al. 2024; Cui et al. 2024) have explored the use of diffusion models for Drag-Based Image Editing (DBIE), which enables users to manipulate images by directly dragging objects via a set of user-specified handle and target points. Existing diffusion-based DBIE methods predominantly employ a point-based alternating optimization strategy (Liu et al. 2024; Cui et al. 2024; Ling et al. 2024; Hou et al. 2024; Choi et al. 2024; Shi et al. 2024), where *Step-1*: optimizing the features of handle points toward corresponding target positions by performing point motion supervision; *Step-2*: updating handle point positions iteratively via KNN-based point tracking.

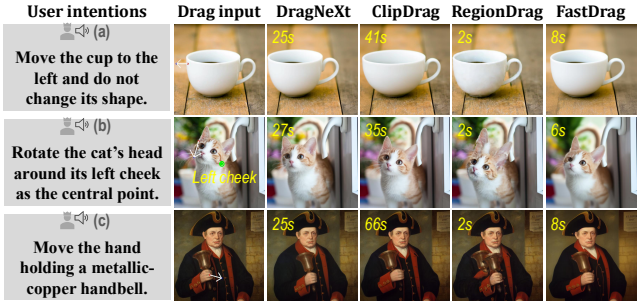

Figure 1: Examples of the key issues in current DBIE. (*i*) Text prompts used in ClipDrag (Jiang, Wang, and Chen 2025) remain insufficient for solving the ambiguity issue; (*ii*) predefined mapping functions employed by FastDrag (Zhao et al. 2024) and RegionDrag (Lu, Li, and Han 2024) boost efficiency but severely compromise editing quality. The numbers given in the upper-left corner of images indicate the latency for dragging the regions of handle points to target positions.

However, the point-based alternating workflow inevitably brings two issues to DBIE: (*i*) point-based drag suffers from high ambiguity and struggles to align with users' intentions, thereby severely compromising the precision of the editing process; (*ii*) tackling DBIE through an alternating procedure of motion supervision and point tracking is not only cumbersome but also fails to always yield high-quality results, as accurately estimating the updated positions of handle points in each drag iteration is both challenging and time-consuming (Ling et al. 2024; Cui et al. 2024). Besides, given that point-based motion supervision offers only limited structural cues about visual scenes, it cannot effectively guide DBIE.

Recently, ClipDrag (Jiang, Wang, and Chen 2025) sought

*Corresponding author.

to mitigate ambiguity by incorporating constraints from text prompts. Nonetheless, as a form of high-level descriptions, texts are often too vague to provide control signals required by fine-grained image manipulation (Shi et al. 2024; Zhang, Rao, and Agrawala 2023). For example, as shown in Figure 1, even with the guidance of the prompt "rotate the cat's head around its left cheek as the central point", ClipDrag still fails to achieve the desired outcome. To boost DBIE's efficiency, FastDrag (Zhao et al. 2024) and RegionDrag (Lu, Li, and Han 2024) proposed using predefined mapping functions, rather than the learnable alternating paradigm. However, unfortunately, the warpage function and the copy-and-paste strategy used in FastDrag and RegionDrag are not flexible enough to handle all editing tasks and are prone to yielding unrealistic or unnatural results—such as the distorted cat's head, deformed handbell, and visible artifacts in the edited areas shown in Figure 1—thus severely degrading image quality.

Point-based motion supervision and tracking are cumbersome and often difficult to align with users' intentions, whereas relying solely on the warpage function or the copy-and-paste strategy is far from delivering high-quality results. **These observations naturally lead us to ask two questions:** $\mapsto\boldsymbol{\mathcal{Q}}$**1.** ***Is there a more effective solution to solve the ambiguity issue?*** $\mapsto\boldsymbol{\mathcal{Q}}$**2.** ***How to enhance the efficiency of DBIE approaches based on alternating point motion supervision and tracking, while further improving their editing quality?***

These two questions motivate us to revisit DBIE from a new perspective—unifying it as a Latent Region Optimization (LRO) problem, which aims to leverage region-level geometric transformations to optimize latent embeddings and realize drag manipulation. Therefore, by specifying the regions and types of geometric transformations, we can effectively address $\boldsymbol{\mathcal{Q}}$**1**. Furthermore, we design a simple-yet-effective editing framework, **DragNeXt**, to tackle $\boldsymbol{\mathcal{Q}}$**2**. For efficiency, it unifies DBIE as LRO and thus eliminates the necessity of conducting handle point tracking by upgrading point motion supervision to region-level optimization of latent embeddings. For editing quality, we propose a Progressive Backward Self-Intervention (PBSI) strategy that solves LRO by fully leveraging region-level self-intervention from intermediate drag states. By bypassing point tracking and considering region-level guidance from intermediate states, it can achieve a better trade-off between efficiency and quality.

Contribution Summary: (*i*) We propose to unify DBIE as an LRO problem. Therefore, by specifying the regions and types of geometric transformations, we can effectively resolve the ambiguity issue. (*ii*) We propose a simple yet effective editing framework, **DragNeXt**, which tackles DBIE via LRO and further enhancing editing quality via performing PBSI. (*iii*) We introduce NextBench, a benchmark with explicit user-intention annotations for evaluating alignment between user expectations and edited results. (*iv*) On NextBench, the extensive experiments demonstrate that our **DragNeXt** can achieve a better trade-off between editing efficiency and quality.

Related Work

DragDiffusion (Shi et al. 2024) is the first work using diffusion models to achieve DBIE, which followed (Pan et al. 2023) and conducted motion supervision and point tracking alternately. Based on (Shi et al. 2024), GoodDrag (Zhang et al. 2024) further enhanced the fidelity of dragged areas by rearranging the drag process across multiple denoising timesteps. DragText (Choi et al. 2024) proposed refining text embeddings to avoid drag halting. DragonDiffusion (Mou et al. 2023) and DiffEitor (Mou et al. 2024) discarded the tracking phase and directly applied point motion supervision between initial handle points and target points. To estimate handle point positions more accurately, StableDrag (Cui et al. 2024) proposed a discriminative point tracking strategy, and FreeDrag (Ling et al. 2024) designed a line search backtracking mechanism. EasyDrag (Hou et al. 2024) advanced (Shi et al. 2024) via introducing a stable motion supervision, which is beneficial for improving the quality of final results. FastDrag (Zhao et al. 2024) and RegionDrag (Lu, Li, and Han 2024) improved the efficiency of DBIE by employing fixed predefined mapping functions, where (Lu, Li, and Han 2024) is based on copy-and-paste and thus requires users to specify both handle and target areas. ClipDrag (Jiang, Wang, and Chen 2025) reduced the ambiguity of DBIE via using text prompts. DragNoise (Liu et al. 2024) proposed editing on UNet's bottleneck features, which inherently contain more semantic information and can stabilize dragging.

REMARK 1. Our method differs from DragDiffusion, GoodDrag, DragText, StableDrag, FreeDrag, EasyDrag, and DragNoise fundamentally, since it does not rely on alternating between point motion supervision and tracking. Instead of only considering initial relationships between handle and target points, as in DragonDiffusion and DiffEditor, we fully leverage progressive region-level guidance from intermediate drag states. CLIPDrag and RegionDrag overlook the ambiguity issue arising from the type of geometric transformations, whereas our approach does not rely on texts to reduce ambiguity. Different from FastDrag and RegionDrag, we only use geometric mapping functions to provide interventional signal and our learnable backward self-intervention strategy can fully leverage inherent prior knowledge learned by diffusion models via back-propagated gradients, avoiding unnatural deformation led by using a fixed transformation pattern.

Methodology

Preliminaries

Diffusion Models. Diffusion models (Ho, Jain, and Abbeel 2020; Rombach et al. 2022; Dhariwal and Nichol 2021) are composed of a diffusion process and a reverse process. During the diffusion, an image $\boldsymbol{x}$ is encoded into latent space $\boldsymbol{z}_0$ and undergoes a gradual addition of Gaussian noise, $q(\boldsymbol{z}_t|\boldsymbol{z}_0) = \mathcal{N}(\sqrt{\alpha_t}\boldsymbol{z}_0, (1-\alpha_t)\boldsymbol{I})$, where α_t is a non-learnable parameter and decreases *w.r.t.* the timestep t. The reverse process is to recover $\boldsymbol{z}_0$ from $\boldsymbol{z}_T$ by training a denoiser $\boldsymbol{\varepsilon}_{\boldsymbol{\Theta}}(\cdot)$:

$$\mathcal{L}_{\boldsymbol{\Theta}} = \mathbb{E}_{t\sim\mathcal{U}(1,T),\boldsymbol{\varepsilon}_t\sim\mathcal{N}(0,I)}\left[||\boldsymbol{\varepsilon}_t - \boldsymbol{\varepsilon}_{\boldsymbol{\Theta}}(\boldsymbol{z}_t; t, \boldsymbol{c})||^2\right] \quad (1)$$

where $\boldsymbol{\varepsilon}_t$ denotes the groundtruth noise in the timestep t, and $\boldsymbol{c}$ represents an extra condition. Following the prior works (Liu et al. 2024; Shi et al. 2024; Zhang et al. 2024), we employ DDIM (Song, Meng, and Ermon 2020) in our approach.

Drag-Based Image Editing. Given n pairs of handle and target points $\boldsymbol{\mathcal{O}} = \{\boldsymbol{h}_i = (x_i^h, y_i^h), \boldsymbol{g}_i = (x_i^g, y_i^g)\}_{i=1,\ldots,n}$,

DBIE aims to edit an image $\boldsymbol{x}$ by dragging objects or regions indicated by handle points to target ones. Usually, an extra binary mask $\boldsymbol{M}$ is used to specify the uneditable region of $\boldsymbol{x}$.

Motion Supervision and Point Tracking. Current DBIE methods (Shi et al. 2024; Zhang et al. 2024; Choi et al. 2024; Cui et al. 2024; Ling et al. 2024; Hou et al. 2024; Liu et al. 2024) mainly rely on performing motion supervision and point tracking alternately, where the former aims to transfer the features of handle points to target positions while the latter updates handle points iteratively and prevents dragging halt. We use $\mathcal{F}_{\boldsymbol{h}_i/\boldsymbol{g}_i}(\boldsymbol{z}_t)$ to denote the features extracted by $\boldsymbol{\varepsilon}_{\boldsymbol{\Theta}}(\cdot)$ at the location $\boldsymbol{h}_i$ or $\boldsymbol{g}_i$. Therefore, the objective function of the motion supervision can be described by Equation (2):

$$\mathcal{L}_m(\boldsymbol{z}_t^k) = \sum_{i=1}^{n} \sum_{\boldsymbol{q}\in\boldsymbol{\pi}(\boldsymbol{h}_i^k)} ||\mathcal{F}_{\boldsymbol{q}+\boldsymbol{d}_i}(\boldsymbol{z}_t^k) - \mathcal{SG}(F_{\boldsymbol{q}}(\boldsymbol{z}_t^k))||_1 + \mathcal{R}_{\boldsymbol{M}} \tag{2}$$

where $\boldsymbol{z}_t^k$ and $\boldsymbol{h}_i^k$ denote the latent code $\boldsymbol{z}_t$ and the handle point $\boldsymbol{h}_i$ updated by k iterations, $\boldsymbol{d}_i = (\boldsymbol{g}_i - \boldsymbol{h}_i^k)/||\boldsymbol{g}_i - \boldsymbol{h}_i^k||_2$ is the normalized vector from $\boldsymbol{h}_i^k$ to $\boldsymbol{g}_i$, $\boldsymbol{\pi}(\boldsymbol{h}_i^k)$ denotes the neighborhood of $\boldsymbol{h}_i^k$, $\mathcal{SG}(\cdot)$ stops gradients from being back-propagated to variables, and $\mathcal{R}_{\boldsymbol{M}}$ is a constraint term to ensure the consistency of uneditable regions. After the motion supervision in each iteration k, point tracking is performed:

$$\boldsymbol{h}_i^{k+1} = \arg\min_{\boldsymbol{q}\in\boldsymbol{\pi}(\boldsymbol{h}_i^k)} ||\mathcal{F}_{\boldsymbol{q}}(\boldsymbol{z}_t^{k+1}) - \mathcal{F}_{\boldsymbol{h}_i}(\boldsymbol{z}_t)||_1 \tag{3}$$

where $\mathcal{F}_{\boldsymbol{h}_i}(\boldsymbol{z}_t)$ indicates the features of the initial handle point $\boldsymbol{h}_i$ in the original latent code $\boldsymbol{z}_t$.

REMARK 2. ↦**Why is point tracking critical for motion-based methods?** Point-based motion supervision is too local to provide enough guidance for the whole editing procedure. Losing the positions of handle points will severely interrupt the drag process, as no alternative guidance for editing remains, thereby significantly damaging the quality of edited images. ↦**What are limitations of methods based on motion supervision and point tracking?** Firstly, although the use of point tracking can alleviate the inherent limitation of point-based motion supervision, it is still very challenging to precisely estimate the updated positions of handle points. Inaccurate coordinate estimation can significantly mislead the dragging process, resulting in suboptimal outcomes. Secondly, the alternating execution of Equation (2) and (3) results in low efficiency of DBIE, since motion supervision is repeatedly disrupted by iterative point tracking. Thirdly, point motion supervision suffers from high ambiguity and easily leads to gaps between user expectations and actual results. Last but not least, while using dense points can reduce ambiguity in some situations, this will substantially decrease the efficiency of the alternating workflow, as shown in Figure 9.

Latent Region Optimization for Reliable Drag-Based Image Editing

We begin by outlining key factors behind the ambiguity issue of DBIE in **Proposition 1**, which can be further summarized as two key questions: *how to drag?* and *what to drag?*

Proposition 1 (**Key Factors to Ambiguity**). The ambiguity of DBIE is twofold: ↦**Factor-1.** *drag operations inherently involve multiple types—such as translation, deformation, and rotation—and treating them as type-agnostic induces ambiguity about users' intentions (**how to drag?**);* ↦**Factor-2.** *point indicators are insufficient for accurately specifying objects or regions that need to be dragged (**what to drag?**).*

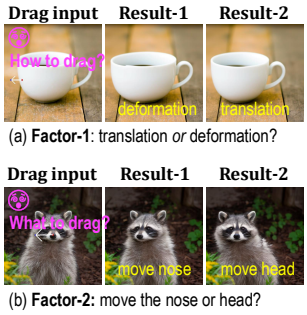

Figure 2: Factor-1 and -2.

Figure 3: Rethink DBIE.

In Figure 2, we provide an illustration for the two key factors, **Factor-1** and **Factor-2**. On one hand, the drag operation in Figure 2(a) is inherently ambiguous since it could be interpreted as either a translational movement of the cup or a deformation of its edge region. This ambiguity stems from uncertainty about the types of drag operations (*how to drag?*), which inevitably increases gaps between user expectations and model behaviors, thus damaging the precision of the editing process. On the other hand, in Figure 2 (b), the drag instruction could be either dragging the raccoon's nose, its head, or even its whole body. This type of ambiguity arises from uncertainty about which areas or objects to drag (*what to drag?*) since points are too ambiguous to clearly reflect users' intentions. How to drag and what to drag are two fundamental problems in DBIE. Although textual description appears to be a shortcut, it actually does not work well as exemplified in Figure 1. We argue there is **No Free Lunch** in resolving these ambiguity issues, which means it is necessary to enable models to perceive drag operation types and areas in a more explicit way and design a more effective approach to guide them toward producing user-intended results.

REMARK 3. ↦Some previous methods have noticed the ambiguity issue in DBIE, but few of them consider both **Factor-1** and **Factor-2**, or provide a systematic and clear analysis for this problem, which we believe is critical and valuable for inspiring the further development of DBIE.

Based on the above observations, we step-by-step introduce how to explore DBIE from a new perspective, i.e., unifying DBIE as a Latent Region Optimization (LRO) problem. We first rethink the DBIE task in **Proposition** 2.

Proposition 2 (**Rethink DBIE**). *DBIE can be regarded as performing geometric transformations on user-specified regions of images.*

For instance, Result-1 given in Figure 2 (a) can be seens as applying a deformation transformation to the white coffee cup region. Form this perspective, by specifying the regions and types of geometric transformations, we can effectively resolve the ambiguity issue caused by Factor-1 and Factor-2, because of clarifying both how and what to drag as illustrated

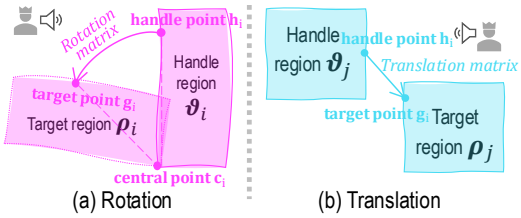

Figure 4: Examples of estimating target regions.

in Figure 3.

Based on **Proposition 2**, we further give the definition of our region-level **Reliable Drag-based Image Editing** (Reliable DBIE) in **Definition** 1, which aims to help users to yield reliable editing results and narrow gaps between their expectations and actual outcomes.

Definition 1 (**Reliable DBIE**). *Reliable DBIE is to manipulate user-specified regions* $\boldsymbol{\mathcal{E}} = \{\boldsymbol{\vartheta}_i\}_{i=1,\dots,n}$ *of an image* $\boldsymbol{x}$ *based on the corresponding geometric transformations* $\boldsymbol{\Gamma} = \{f_i\}_{i=1,\dots,n}$ *inferred from instructions given by users.*

Currently, the editing process is primarily performed on noise latent embeddings encoded by diffusion models, as they are more editable than original images (Mokady et al. 2023; Ruiz et al. 2023). Thus, we can further extend **Definition 1** to **Definition 2**, and unify DBIE as a Latent Region Optimization (LRO) problem.

Definition 2 (***Unify DBIE as LRO***). *DBIE can be unified as optimizing specific target regions* $\boldsymbol{\mathcal{P}} = \{\boldsymbol{\rho}_i\}_{i=1,\dots,n}$ *within a latent code* $\boldsymbol{z}_t$ *based on user-specified handle regions* $\boldsymbol{\mathcal{E}} = \{\boldsymbol{\vartheta}_i\}_{i=1,\dots,n}$ *and the corresponding geometric transformations* $\boldsymbol{\Gamma} = \{f_i\}_{i=1,\dots,n}$ *involved in user instructions:*

$$\boldsymbol{z}_t^* = \arg\min_{\boldsymbol{z}_t} \mathcal{L}_{LRO}\left(\boldsymbol{z}_t, \{\boldsymbol{\rho}_i\}_{i=1,\dots,n}\right) \tag{4}$$
$$s.t., \{\boldsymbol{\rho}_i\}_{i=1,\dots,n} = \delta(\boldsymbol{\mathcal{E}}, \boldsymbol{\Gamma})$$

where $\mathcal{L}_{LRO}$ *is the objective function of LRO, and* $\delta(\cdot)$ *aims to produce binary masks* $\{\boldsymbol{\rho}_i\}_{i=1,\dots,n}$ *to identify target regions required to be optimized in* $\boldsymbol{z}_t$ *according to* $\boldsymbol{\mathcal{E}}$ *and* $\boldsymbol{\Gamma}$.

REMARK 4. $\mapsto$**Why LRO?** LRO serves as a bridge between DBIE and region-level geometric transformations. Therefore, we can leverage many well-studied geometric transformation functions in computer graphics to realize more reliable DBIE via explicitly controlling the dragging process. $\mapsto$**What can LRO do?** Different from methods based on alternating point motion supervision and tracking, LRO takes into account region-level visual information, which provides more robust guidance for latent code manipulation. Under such regional supervision, it is unnecessary to excessively focus on positions of some specific points, as there exists sufficient context information to guide dragging. $\mapsto$**How to estimate target latent regions?** Target latent regions are estimated using geometric transformation functions widely adopted in computer graphics. As exemplified in Figure 4, if users intend to rotate the handle region $\boldsymbol{\vartheta}_i$, the region can be multiplied by a rotation matrix to achieve the desired geometric transformation.

We observe that rotation and translation geometric transformations can cover most DBIE scenarios. We reformulate DBIE's user input: users specify a set of handle regions $\boldsymbol{\mathcal{E}} = \{\boldsymbol{\vartheta}_i\}_{i=1,\dots,n}$ for an input image and give the corresponding drag instructions $\boldsymbol{\mathcal{C}} = \{\mathcal{T}_i, \boldsymbol{\mathcal{O}}_i\}_{i=1,\dots,n}$. If the operation type $\mathcal{T}_i$ ="rotation", $\boldsymbol{\mathcal{O}}_i = \{\boldsymbol{h}_i, \boldsymbol{g}_i, \boldsymbol{c}_i\}$ where $\boldsymbol{h}_i$ and $\boldsymbol{g}_i$ denote a pair of a handle point and a target point, and $\boldsymbol{c}_i$ represents a rotation center of $\boldsymbol{\vartheta}_i$; otherwise, $\boldsymbol{\mathcal{O}}_i = \{\boldsymbol{h}_i, \boldsymbol{g}_i\}$. Also, a binary mask $\boldsymbol{M}$ is adopted to specify the uneditable region. Based on $\{\boldsymbol{h}_i, \boldsymbol{g}_i\}$ or $\{\boldsymbol{h}_i, \boldsymbol{g}_i, \boldsymbol{c}_i\}$, the transformation function f_i can be constructed by determining the corresponding rotation and translation matrix. For details on converting input points to f_i, please refer to the supplementary material.

REMARK 5. DBIE via regional geometric transformations. Object movement can be achieved by translating an object' entire region; **deformation** can be realized by translating only its edge region; **2D rotation** can be achieved by applying a rotation transformation; and **3D rotation** can be interpreted as translating the sub-region of an object, assisted by prior knowledge inherently learned in diffusion models (as shown in Figure 7 (j), the car's 3D rotation can be realized by translating its front to the right). Explicit geometric functions and region-level guidance can help achieve better DBIE.

Progressive Backward Self-Intervention: Less Meets More!

Based on **Definition 2**, we further design **DragNeXt** to enhance both editing quality and efficiency. As mentioned before, the alternating workflow lowers the efficiency of DBIE, while inaccurate handle point tracking easily leads to dragging halt and makes results unsatisfactory. Therefore, **DragNeXt** addresses DBIE from an LRO perspective, eliminating the need for KNN-based point tracking by explicitly advancing point-based motion supervision to region-level optimization of latent embeddings. Moreover, it employs a Progressive Backward Self-Intervention (PBSI) strategy, which does not require accurately tracking point positions but still achieves superior editing results by fully leveraging progressive region-level guidance from intermediate transformation states.

Progressive Backward Self-Intervention. Figure 5 gives a brief illustration of our approach. Given an input image $\boldsymbol{x}$, we first encode it into latent space and perform DDIM inversion to produce $\boldsymbol{z}_T$. Then, PBSI is conducted from T to T' during denoising with K iterations per timestep. We take the handle region $\boldsymbol{\vartheta}_i$ at the k-th iteration of the timestep t as an example to illustrate PBSI. We first extract the features of $\boldsymbol{z}_t^k$ by concatenating outputs from the last upsample blocks of all stages of $\boldsymbol{\varepsilon}_{\boldsymbol{\Theta}}(\boldsymbol{z}_t^k)$ and upsampling them to half of the resolution of $\boldsymbol{x}$, denoted as $\mathcal{F}(\boldsymbol{z}_t^k)$. Then, we estimate the intermediate transformation state $\boldsymbol{\rho}_i^{t,k}$ for the handle region $\boldsymbol{\vartheta}_i$ within the extracted features $\mathcal{F}(\boldsymbol{z}_t^k)$ based on user-given conditions $\boldsymbol{\mathcal{C}}$, which can be described by Equation (5):

$$\boldsymbol{\rho}_i^{t,k}, \boldsymbol{\Pi}_{\boldsymbol{\vartheta}_i \to \boldsymbol{\rho}_i^{t,k}} = \delta(\boldsymbol{\vartheta}_i, f_i^{t,k}) \tag{5}$$
$$s.t., \delta(\boldsymbol{\vartheta}_i, f_i^{t,k}) = \begin{cases} \texttt{Rot}(\boldsymbol{\vartheta}_i, \boldsymbol{c}_i, \theta), & \text{if } \mathcal{T}_i = \text{"rotation"} \\ \texttt{Trans}(\boldsymbol{\vartheta}_i, \boldsymbol{\omega}), & \text{else.} \end{cases}$$

In the equation, $\texttt{Rot}(\boldsymbol{\vartheta}_i, \boldsymbol{c}_i, \theta)$ aims to rotate the handle region $\boldsymbol{\vartheta}_i$ around the center point $\boldsymbol{c}_i$ by an angle $\theta = \eta^{t,k} * \angle \boldsymbol{h}_i \boldsymbol{c}_i \boldsymbol{g}_i$, $\texttt{Trans}(\boldsymbol{\vartheta}_i, \boldsymbol{\omega})$ translates $\boldsymbol{\vartheta}_i$ according to the offset vector $\boldsymbol{\omega} = \eta^{t,k} * (\boldsymbol{g}_i - \boldsymbol{h}_i)$, and $\eta^{t,k} = \frac{K*(T-t)+k}{K*(T-T'+1)}$

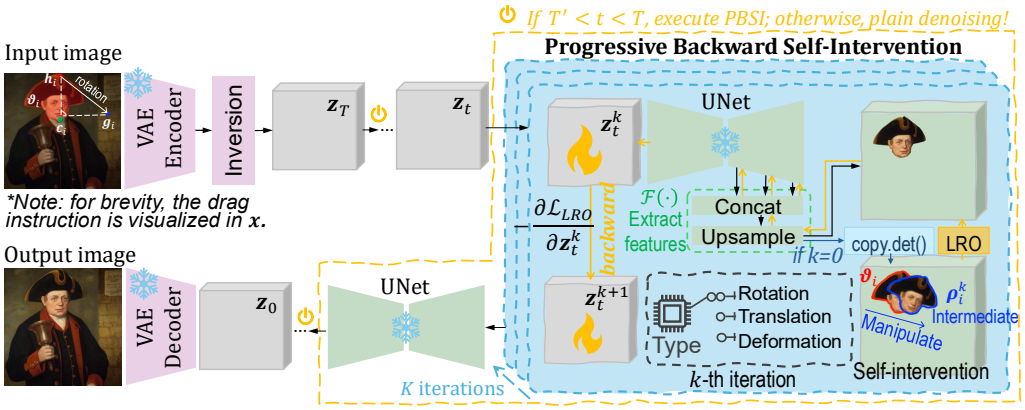

Figure 5: A brief illustration of our DragNeXt.

is a weighting factor that determines angles or offsets of intermediate states. Also, $\boldsymbol{\rho}_i^{t,k}$ is a binary mask that identifies the target intermediate region in $\mathcal{F}(\boldsymbol{z}_t^k)$, and $\boldsymbol{\Pi}_{\boldsymbol{\vartheta}_i \to \boldsymbol{\rho}_i^{t,k}}$ represents the coordinate mapping from the handle region $\boldsymbol{\vartheta}_i$ to the intermediate state $\boldsymbol{\rho}_i^{t,k}$. Finally, we copy and detach the features extracted from the original latent code, $\mathcal{F}'(\boldsymbol{z}_t) = \mathcal{F}(\boldsymbol{z}_t)\texttt{.copy.detach()}$. Moreover, we interventionally adjust the detached features according to the obtained coordinate mapping, $\mathcal{F}'(\boldsymbol{z}_t)[\boldsymbol{\Pi}_{\boldsymbol{\vartheta}_i \to \boldsymbol{\rho}_i^{t,k}}]$, thereby perturbing the original latent representations and transforming the features of the handle region $\boldsymbol{\vartheta}_i$ to the intermediate target position $\boldsymbol{\rho}_i^{t,k}$. We consider self-intervention from the perturbed features to $\mathcal{F}(\boldsymbol{z}_t^k)$ and back-propagate interventional signal to the latent code $\boldsymbol{z}_t^k$ along the denoiser to update latent features. This can be depicted by Equation (6) and (7):

$$\boldsymbol{z}_t^{k+1} \longleftarrow \boldsymbol{z}_t^k - \frac{\partial \mathcal{L}_{LRO}}{\partial \boldsymbol{z}_t^k}, \tag{6}$$

$$\mathcal{L}_{LRO} = \|\mathcal{F}(\boldsymbol{z}_t^k) * \boldsymbol{\rho}_i^{t,k} - \mathcal{F}'(\boldsymbol{z}_t)[\boldsymbol{\Pi}_{\boldsymbol{\vartheta}_i \to \boldsymbol{\rho}_i^{t,k}}] * \boldsymbol{\rho}_i^{t,k}\|_1 + \mathcal{R}_M. \tag{7}$$

Minimizing $\mathcal{L}_{LRO}$ back-propagates self-intervention gradients to the latent code, thus progressively dragging handle regions to target positions. Once PBSI is complete, we denoise $\boldsymbol{z}_{T'}$ to $\boldsymbol{z}_0$ and decode it into image space. The pseudocode of our **DragNeXt** is provided in Algorithm 1 of the appendix.

REMARK 6. $\mapsto$**Why backward self-intervention?** Our method also adopts geometric mapping functions. However, unlike RegionDrag and FastDrag, which directly use them to manipulate latent code, we instead leverage them to provide interventional signal. By optimizing latent code through back-propagated gradients from the denoiser, our approach fully exploits the prior of pretrained diffusion models, thereby mitigating unnatural results caused by fixed mapping functions. $\mapsto$**Difference between Equation (7) and (2).** $\mathcal{L}_{LRO}$ considers region-level guidance, whereas $\mathcal{L}_m$ performs point supervision and needs to iteratively track handle point positions. $\mapsto$**Discussion on Equation (5).** We unify translation, deformation, and 3D rotation into a single mapping function, $\texttt{Trans}(\cdot)$, as deformation and 3D rotation can be interpreted by translating the partial regions of an object, assisted by

Figure 6: Comparison between our NextBench and the previous benchmarks, DragBench and Drag100.

priors inherently encoded in diffusion models, *e.g.,* in Figure 7 (j), the car's 3D rotation can be achieved by translating its front to the right. We extend drag points to regional guidance to more clearly specify regions to move, deform, or rotate.

Experiments

We first introduce our NextBench and evaluation metrics, followed by the main results of our method and ablation studies. We provide implementation details in the supplementary material due to the limited space of the paper's main body.

NextBench: a Benchmark for Reliable DBIE

To better evaluate model performance on Reliable DBIE, we propose a new benchmark, **NextBench**, that comprises 234 test samples with drag operations including translation, 2D/3D rotation, and deformation. Each sample is clearly annotated with user intentions to better assess how well model outputs align with user expectations, as shown in Figure 6.

Why NextBench? Existing benchmarks, such as DragBench (Shi et al. 2024) and Drag100 (Zhang et al. 2024), still contain ambiguous drag instructions, *e.g.*, as shown in Figure 6, the raccoon's and dog's heads could either be translated or rotated, and both would be considered as satisfactory results. Our NextBench explicitly annotates each sample with a clear user expectation, handle regions, and editable areas, therefore enabling a more reliable assessment of intention–result alignment and regional consistency. NextBench also treats 3D and 2D rotations as two distinct operations, showing that current

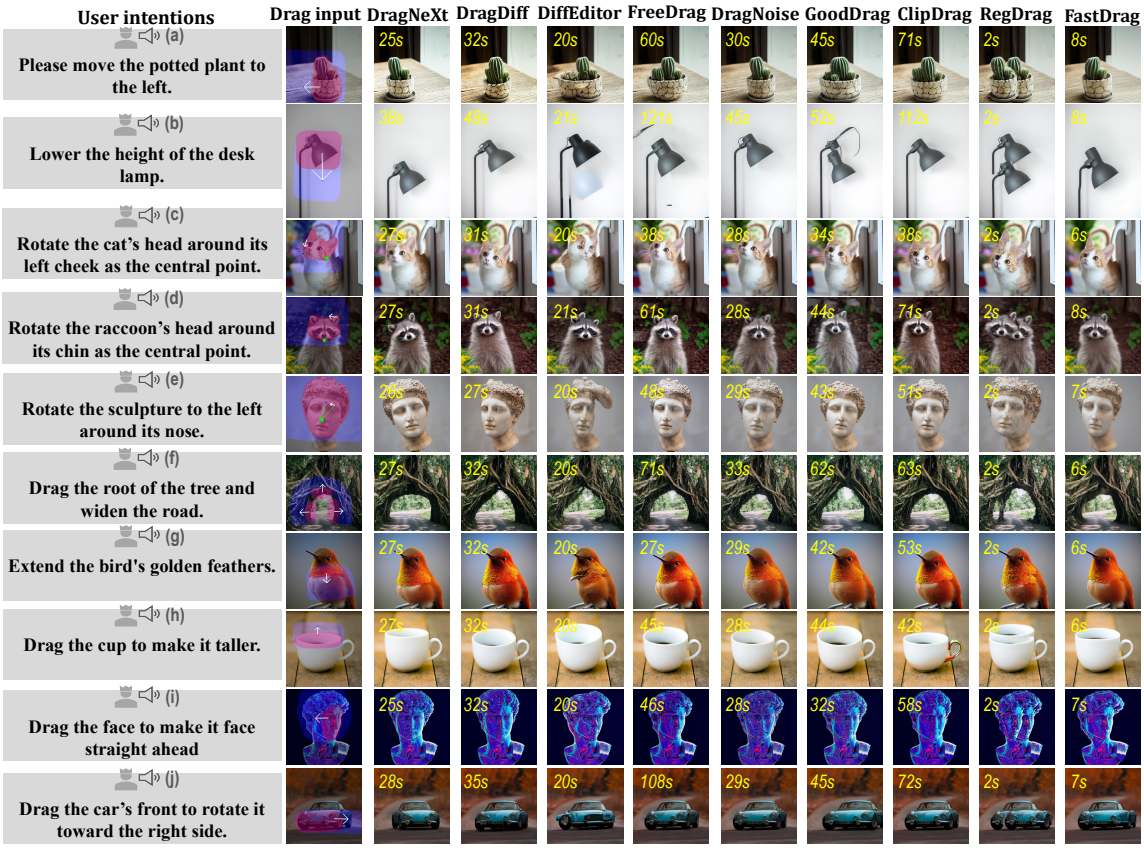

Figure 7: Qualitative results achieved by our DragNeXt.

approaches struggle with 2D rotation and excel at yielding 3D rotation based on the prior of pretrained diffusion models.

Evaluation Metrics

Following prior work (Zhang et al. 2024; Shi et al. 2024), we use LPIPS and DAI to evaluate performance on NextBench, where the radius of DAI is set as 20. To better assess region-level DBIE, LPIPS is computed between original images and edited results in three parts: (*i*) LPIPS$_{ue}$ for uneditable regions (*a lower LPIPS$_{ue}$ indicates better preservation of uneditable regions*); (*ii*) LPIPS$_{th}$ for consistency between handle and target regions (*a lower LPIPS$_{th}$ means handle regions are successfully dragged to target positions*); (*iii*) LPIPS$_{hh}$ for handle regions (*Successful dragging handle regions to target positions should result in a higher LPIPS$_{hh}$ between the handle regions of original input images and edited results, reflecting the change of visual content*).

Main results

We compare our method with eight typical open-source DBIE methods: DragDiffusion, DiffEditor, DragNoise, FreeDrag, GoodDrag, ClipDrag, RegionDrag, and FastDrag.

Qualitative Results. We present qualitative results of our method in Figure 7, from which we can make the following observations. By specifying the regions and the types of geometric transformations, **DragNeXt** achieves better alignment between user expectations and results, *e.g.*, as shown in Figure 7 (a), our models translate the potted plant leftward without damaging its shape. Based on LRO and PBSI, **DragNeXt** achieves a better trade-off between efficiency and quality, *e.g.*, it has obviously higher efficiency than methods based on alternating point motion supervision and tracking, while delivering obviously higher editing quality than those relying solely on predefined mapping functions.

Method	Lat↓	DAI↓	LPIPS$_{ue}$ ↓	LPIPS$_{th}$ ↓	LPIPS$_{hh}$ ↑
DragDiff	36s	0.10333	0.05914	0.32314	0.23663
DiffEditor	24s	0.07255	0.05960	0.30570	0.23291
DragNoise	34s	0.10001	0.05891	0.35697	0.24789
FreeDrag	69s	0.11113	0.07693	0.36907	0.26723
GoodDrag	51s	0.06878	**0.05794**	0.28667	0.25291
ClipDrag	58s	0.11909	0.06243	0.37292	0.20528
RegionDrag	**3s**	0.05968	0.05930	0.18780	0.19736
FastDrag	8s	0.08530	0.06624	0.32646	0.28411
DragNeXt	28s	**0.05775**	0.05886	**0.16223**	**0.36653**

Table 1: Quantitative results on NextBench. "Lat" indicates mean latency per image for dragging handle point regions to target positions. ↑ / ↓ denotes higher/lower values are better.

REMARK 7. Discussion on 2D/3D rotation. ↦We observe that existing DBIE methods particularly excel at yielding 3D rotation effects, *e.g.*, as shown in Figure 7 (i) and (j), dragging the face or the vehicle front enables most methods to rotate them. This capability arises from the strong prior of

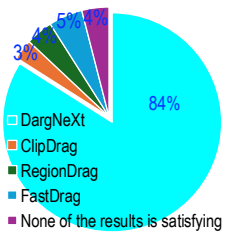

Figure 8: Voting results.

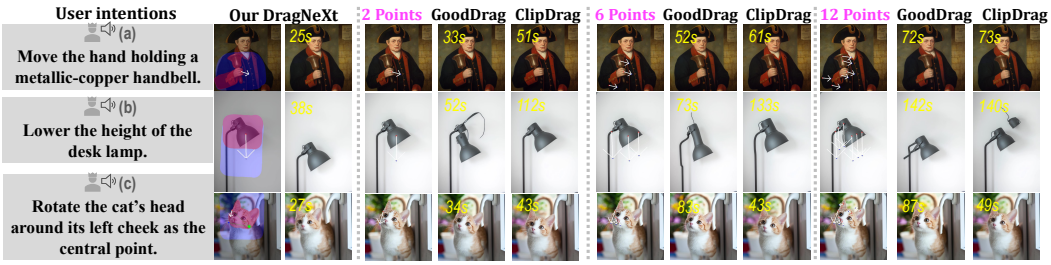

Figure 9: Efficiency and quality improvements over the point-based alternating workflow.

pretrained diffusion models, which are creative to generate rotated objects based on latent features perturbed by drag operations. *Among these methods,* **DragNeXt** *leverages region-level visual cues, which can obviously guide diffusion models to realize better 3D rotation effects.* ↦Current DBIE methods generally fail to perform 2D rotation, as these patterns are not well captured and learned by diffusion models. **DragNeXt** can mitigate this by explicitly using regional geometric transformations.

Quantitative Results. The quantitative results are summarized in Table 1. The table shows that our method achieves the lowest DAI and LPIPS$_{th}$ and the highest LPIPS$_{hh}$, demonstrating that our method can effectively drag objects from handle regions to target positions. Unsuccessfully dragging objects to target positions results in low LPIPS$_{hh}$—indicating little change in handle regions—and high LPIPS$_{th}$ due to mismatch between original handle regions and edited target regions. Also, the value of LPIPS$_{ue}$ indicates that our method can preserve the high fidelity of uneditable areas.

Anonymous User Study. We further conducted user studies to validate our method, where 26 participants are invited. Selecting the most appropriate result from too many options is time-consuming; to reduce volunteers' workload, we include the three most relevant methods: ClipDrag, RegionDrag, and FastDrag. The results consistently demonstrate the superiority of **DragNeXt**, *e.g.,* 84% of the votes favored our results, demonstrating higher quality and better alignment with user expectations (see the supplementary material for details).

Method Analysis

Efficiency and Quality Improvements. In Figure 9, we analyze the efficiency and quality improvements of our method over the point-based alternating workflow. Without losing generality, we choose two recent typical works—GoodDrag and ClipDrag—as compared baselines. The results consistently validate the effectiveness of our method. The point-based methods are often difficult to align with user intentions, e.g., as shown in Figure 9 (a), GoodDrag and ClipDrag drag only the hand and leave the handbell unmoved using 2 points. Although increasing the number of points can mitigate ambiguity, this largely slows down the alternating drag process and fail to always guide models to yield satisfactory results.

Ablation Study on PBSI. We provide ablation studies for our PBSI strategy in Figure 10. Based on the results shown in the figure, we have the following observations. Firstly, removing the guidance of intermediate states significantly degrades

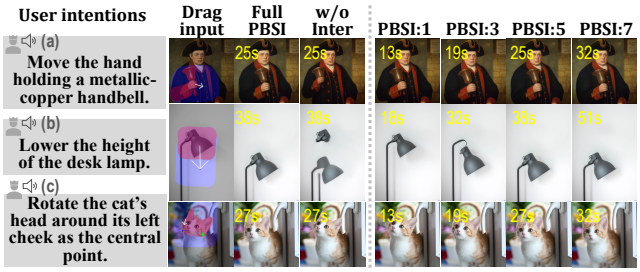

Figure 10: Ablation study on PBSI. "Full PBSI" indicates using the full PBSI strategy, "w/o inter" represents the guidance from intermediate drag states is not considered in PBSI, and "PBSI: N" indicates that PBSI is performed over N timesteps. Zoom in for a better view.

output quality, *e.g.*, the hand in Figure 10 (a) is dragged to an incorrect position, and unnatural results are yielded in Figure 10 (b), thereby demonstrating its important role in achieving high-quality results of DBIE. We also study the impact of performing PBSI over different numbers of timesteps. When PBSI is applied to only a single denoising timestep, objects cannot be successfully dragged to target positions. By contrast, increasing the timesteps of performing PBSI obviously improves the quality of edited results, saturating after 5 timesteps, which also indicate the effectiveness of our method in guiding diffusion models to achieve DBIE.

Conclusion

We propose to address Drag-Based Image Editing (DBIE) from a new perspective—unifying it as a Latent Region Optimization (LRO) problem that aims to use region-level geometric transformations to optimize latent code to realize drag-based manipulation. By specifying the areas and types of geometric transformations, we can effectively reduce gaps between users' intentions and actual model behaviors. We also design a new simple-yet-effective editing framework, dubbed **DragNeXt**. It solves LRO through a Progressive Backward Self-Intervention (PBSI), which simplifies the procedure of DBIE while further enhancing editing quality by fully leveraging region-level structure information and progressive guidance from intermediate transformation states. Physically driven DBIE remains highly challenging. Therefore, in the future, we plan to enhance our **DragNeXt** by integrating physics-based geometric transformation functions.

Acknowledgments

This research is supported by the RIE2025 Industry Alignment Fund – Industry Collaboration Projects (IAF-ICP) (Award I2301E0026), administered by A*STAR, as well as supported by Alibaba Group and NTU Singapore through Alibaba-NTU Global e-Sustainability CorpLab (ANGEL).

References

Brooks, T.; Holynski, A.; and Efros, A. A. 2023. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 18392–18402.

Choi, G.; Jeong, T.; Hong, S.; and Hwang, S. J. 2024. DragText: Rethinking Text Embedding in Point-based Image Editing. *arXiv preprint arXiv:2407.17843*.

Chung, J.; Hyun, S.; and Heo, J.-P. 2024. Style injection in diffusion: A training-free approach for adapting large-scale diffusion models for style transfer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 8795–8805.

Cui, Y.; Zhao, X.; Zhang, G.; Cao, S.; Ma, K.; and Wang, L. 2024. StableDrag: Stable dragging for point-based image editing. In *European Conference on Computer Vision*, 340–356. Springer.

Dhariwal, P.; and Nichol, A. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34: 8780–8794.

Hertz, A.; Mokady, R.; Tenenbaum, J.; Aberman, K.; Pritch, Y.; and Cohen-Or, D. 2022. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33: 6840–6851.

Hou, X.; Liu, B.; Zhang, Y.; Liu, J.; Liu, Y.; and You, H. 2024. Easydrag: Efficient point-based manipulation on diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8404–8413.

Jiang, Z.; Wang, Z.; and Chen, L. 2025. CLIPDrag: Combining Text-based and Drag-based Instructions for Image Editing. In *Proceedings of the Thirteenth International Conference on Learning Representations*.

Ling, P.; Chen, L.; Zhang, P.; Chen, H.; Jin, Y.; and Zheng, J. 2024. Freedrag: Feature dragging for reliable point-based image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6860–6870.

Liu, H.; Xu, C.; Yang, Y.; Zeng, L.; and He, S. 2024. Drag your noise: Interactive point-based editing via diffusion semantic propagation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 6743–6752.

Lu, J.; Li, X.; and Han, K. 2024. Regiondrag: Fast region-based image editing with diffusion models. In *European Conference on Computer Vision*, 231–246. Springer.

Mokady, R.; Hertz, A.; Aberman, K.; Pritch, Y.; and Cohen-Or, D. 2023. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 6038–6047.

Mou, C.; Wang, X.; Song, J.; Shan, Y.; and Zhang, J. 2023. Dragondiffusion: Enabling drag-style manipulation on diffusion models. *arXiv preprint arXiv:2307.02421*.

Mou, C.; Wang, X.; Song, J.; Shan, Y.; and Zhang, J. 2024. Diffeditor: Boosting accuracy and flexibility on diffusion-based image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8488–8497.

Pan, X.; Tewari, A.; Leimkühler, T.; Liu, L.; Meka, A.; and Theobalt, C. 2023. Drag your gan: Interactive point-based manipulation on the generative image manifold. In *ACM SIGGRAPH 2023 conference proceedings*, 1–11.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10684–10695.

Ruiz, N.; Li, Y.; Jampani, V.; Pritch, Y.; Rubinstein, M.; and Aberman, K. 2023. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 22500–22510.

Shi, Y.; Xue, C.; Liew, J. H.; Pan, J.; Yan, H.; Zhang, W.; Tan, V. Y.; and Bai, S. 2024. Dragdiffusion: Harnessing diffusion models for interactive point-based image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8839–8849.

Song, J.; Meng, C.; and Ermon, S. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.

Sun, H.; Li, W.; Liu, J.; Chen, H.; Pei, R.; Zou, X.; Yan, Y.; and Yang, Y. 2024. Coser: Bridging image and language for cognitive super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 25868–25878.

Wu, R.; Yang, T.; Sun, L.; Zhang, Z.; Li, S.; and Zhang, L. 2024. Seesr: Towards semantics-aware real-world image super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 25456–25467.

Zhang, L.; Rao, A.; and Agrawala, M. 2023. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, 3836–3847.

Zhang, Y.; Huang, N.; Tang, F.; Huang, H.; Ma, C.; Dong, W.; and Xu, C. 2023. Inversion-based style transfer with diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10146–10156.

Zhang, Z.; Liu, H.; Chen, J.; and Xu, X. 2024. GoodDrag: Towards good practices for drag editing with diffusion models. *arXiv preprint arXiv:2404.07206*.

Zhao, X.; Guan, J.; Fan, C.; Xu, D.; Lin, Y.; Pan, H.; and Feng, P. 2024. FastDrag: Manipulate anything in one step. *arXiv preprint arXiv:2405.15769*.