THINKDRAG: SEMANTIC DRAG-BASED IMAGE EDITING WITH VISUAL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Drag-based image editing provides an intuitive interface for spatially controlled image manipulation by allowing users to specify handle and target points. Existing methods have made substantial progress through optimization-based, one-step, and feed-forward formulations, but they often interpret drag instructions as geometric displacements. This limits their effectiveness when the desired edit depends on interpreting image content and inferring a plausible semantic transformation from sparse point constraints. We introduce THINKDRAG, a unified multimodal framework for drag-based image editing with visual reasoning. THINKDRAG is trained to associate point constraints with meaningful object- and scene-level transformations, and can optionally produce an explicit reasoning trace that interprets the intended edit before image generation. This reasoning trace improves interpretability, particularly for ambiguous drags. Drag Endpoint Markers, motivated by an analysis of cross-image attention, provide an explicit correspondence signal that grounds each drag in image space. To support this framework, we construct a supervised dataset of semantic drag transformations paired with reasoning traces and introduce DragBench++, a benchmark targeting challenging drag-based editing scenarios with reference edit solutions. Experiments show that THINKDRAG achieves state-of-the-art performance, improving generation quality and plausibility while maintaining competitive point-following precision.

1 INTRODUCTION

Image editing is a fundamental problem in generative modeling, where a model must modify visual content while keeping the rest of the scene intact (Meng et al., 2022; Hertz et al., 2023; Brooks et al., 2023). Beyond content creation, it also tests how well a model connects visual understanding with controllable synthesis (Deng et al., 2025). Text-guided editing has made image manipulation more accessible, but language alone often provides limited spatial precision. Many edits are easier to specify by directly indicating where visual content should move, such as repositioning an object part or changing a pose. Drag-based image editing (Pan et al., 2023; Shi et al., 2024) provides this interface by allowing users to specify pairs of handle and target points, after which the model edits the image by moving the content associated with each handle point toward its target.

Early methods such as DragGAN (Pan et al., 2023) demonstrated drag-based editing with GAN priors, while diffusion-based methods extended it to broader visual domains (Shi et al., 2024; Mou et al., 2024). Existing approaches typically rely on latent optimization (Zhang et al., 2025; Xia et al., 2025), one-step latent editing (Zhao et al., 2024; Pu et al., 2026), or feed-forward generation conditioned on point coordinates (Shin et al., 2024; Shi et al., 2025). However, they largely treat drag instructions as spatial displacements and rely on the base generative prior for plausibility. This is limiting because the same handle–target displacement can correspond to different valid edits depending on the object and on what the user intends. For example, moving a point on an articulated object may require rotating a part rather than translating a local patch.

Recent unified multimodal models (Zhou et al., 2025; Xie et al., 2025a; Deng et al., 2025) provide a new opportunity for drag-based image editing. Unlike conventional image editors designed primarily around synthesis (Brooks et al., 2023; Wu et al., 2025a; Black Forest Labs, 2025), these models couple visual and language understanding with image generation in a single framework. This extends the intuition behind DragGAN (Pan et al., 2023): if a generative prior enables drag editing within a limited domain, then unified multimodal models can provide a broader prior for arbitrary images by drawing on their knowledge of the visual world. Such a prior is particularly suitable for drag-based

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

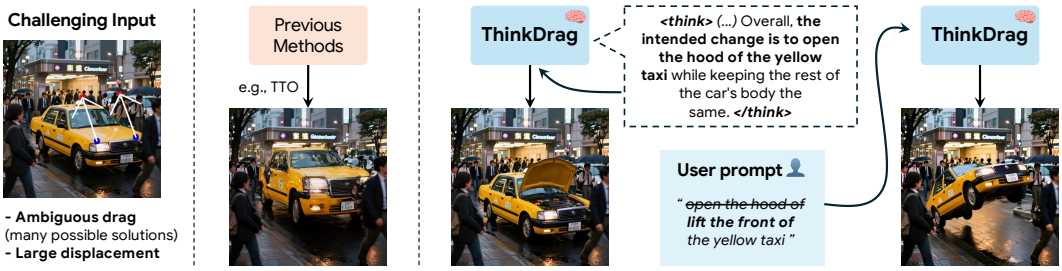

Figure 1: THINKDRAG handles challenging and ambiguous drags, optionally making its interpretation explicit as a reasoning trace that the user can refine, while the drags keep the edit spatially grounded.

editing, where a point instruction is not simply a displacement vector but an under-specified editing request. Executing it correctly requires understanding what the scene contains and how its objects can move or change. We therefore formulate drag-based editing as semantic interpretation coupled with correspondence-grounded generation.

In this work, we introduce THINKDRAG, a unified multimodal framework for drag-based image editing with visual reasoning. Given an input image and drag instructions, THINKDRAG generates an edited image that satisfies the point constraints while preserving the semantic structure of the scene. This is enabled by our new dataset of high-quality semantic drag edits, which teach the model to associate sparse point constraints with plausible object- and scene-level transformations, paired with reasoning traces that teach it to interpret the intended edit before generation. To execute drags precisely, we further introduce Drag Endpoint Markers (DEM). Motivated by the observation that each region of the edited image attends predominantly to the source region from which its content originates, DEM explicitly encodes the correspondence between each handle and its target. As illustrated in Figure 1, THINKDRAG uses visual reasoning to interpret ambiguous drag instructions and can further incorporate user text when additional semantic guidance is provided. The model can edit directly, or first reason when the edit is ambiguous or requires higher-level scene understanding.

We evaluate THINKDRAG on standard drag-editing benchmarks and introduce DragBench++, a new benchmark targeting more challenging scenarios that require higher-level scene understanding and semantic interpretation of point constraints. Extensive experiments show that THINKDRAG achieves state-of-the-art performance, improving generation quality while maintaining competitive point-following precision. The direct editing mode already performs strongly, reflecting the value of semantic drag supervision and unified multimodal modeling. The reasoning mode complements direct editing by making the model's interpretation explicit, and it is particularly useful for ambiguous multi-drag edits. In a user study, participants prefer THINKDRAG to prior methods in 80% of the overall-preference votes on DragBench++ and 61% on DragBench.

Our contributions are summarized as follows:

- We introduce THINKDRAG, a unified multimodal framework for drag-based image editing with visual reasoning, interpreting drag instructions with scene semantics and object structure.
- We construct, to our knowledge, the first supervised dataset of high-quality semantic drag transformations with reasoning traces for training drag-based image editing models.
- We propose DragBench++, a benchmark targeting more challenging and semantically ambiguous drag-based editing scenarios, with reference edit solutions.
- We achieve state-of-the-art results on drag-based editing benchmarks and support joint text-drag conditioning, where text disambiguates user intent and drags provide precise spatial control.

2 RELATED WORK

2.1 UNIFIED MULTIMODAL UNDERSTANDING AND GENERATION

Multimodal understanding and image generation have traditionally followed different modeling paradigms, with autoregressive vision-language models driving perception and reasoning (Liu et al.,

2023a; 2024b; Bai et al., 2023), and diffusion models dominating high-fidelity synthesis (Ho et al., 2020; Song et al., 2021; Dhariwal & Nichol, 2021). Recent unified multimodal models combine both capabilities in a single framework, either by tokenizing images and modeling them jointly with text through next-token prediction (Chameleon Team, 2024; Wang et al., 2024; Wu et al., 2025b; Chen et al., 2025) or by coupling autoregressive language modeling with diffusion-based visual generation (Xie et al., 2025b; Zhou et al., 2025; Deng et al., 2025; Liao et al., 2025). Multimodal chain-of-thought (Wang et al., 2025; Shao et al., 2024) and generator-coupled reasoning (Qin et al., 2026; He et al., 2026) further show a growing ability to connect visual perception, language reasoning, and image synthesis, but their use for spatially grounded image manipulation remains underexplored.

2.2 Drag-based Image Editing

Drag-based image editing provides intuitive image manipulation by dragging user-specified handle points to target positions. DragGAN (Pan et al., 2023) introduced this paradigm through latent optimization with motion supervision and point tracking, but its GAN prior limits generalization to arbitrary real images. Diffusion-based methods extend drag editing to broader domains, mostly through test-time optimization (Shi et al., 2024; Mou et al., 2024; Hou et al., 2024; Ling et al., 2024; Liu et al., 2024a; Cui et al., 2024; Zhang et al., 2025; Jiang et al., 2025; Choi et al., 2025; Xia et al., 2025; Zhou et al., 2026; Lin et al., 2025) or faster one-step and training-free alternatives (Nie et al., 2024; Zhao et al., 2024; Lu et al., 2024; Lu & Han, 2025; Yin et al., 2026; Pu et al., 2026; Jhong et al., 2026; He et al., 2025). Most related to our work are feed-forward methods that frame drag-based editing as conditional generation. InstantDrag (Shin et al., 2024) and LightningDrag (Shi et al., 2025) synthesize edited images directly from point or motion conditions derived from videos, treating drags as geometric constraints, while Cui et al. (2024) infer the intended edit with an LLM at test time. In contrast, ThinkDrag interprets drag instructions within a unified multimodal model and grounds edits through an explicit correspondence signal in image space, learned from semantic drag edits.

3 ThinkDrag

3.1 Dataset Construction

Prior work (Shin et al., 2024; Shi et al., 2025) has shown that videos provide a natural source of paired supervision for drag-based image editing. These pipelines typically sample source and target images from a temporal window and recover motion using optical flow or point tracking. However, video-derived supervision has several limitations. Extracted correspondences are primarily geometric and may not reflect semantically meaningful edit intent, optical flow can be unreliable under large displacement or occlusion, and natural videos often contain camera motion or background variation that corrupts the desired edit supervision. We therefore construct the ThinkDrag dataset, a new drag-based image editing dataset that emphasizes high-quality image pairs and semantically meaningful point instructions. Figure 2 gives an overview of the pipeline, described below.

Synthetic Image Pair Generation. Drag-based image editing is best suited to transformations that can be expressed through spatial motion of existing visual content, such as object translation, pose change, non-rigid deformation, and local geometric warping. It is less appropriate for edits involving semantic discontinuities (*e.g.*, object insertion and removal, or global style transfer). Based on this distinction, we construct synthetic image pairs through an automated MLLM-assisted pipeline. We first use an MLLM (Bai et al., 2025) to generate diverse source-image prompts from a pool of subjects and environments, and synthesize source images using state-of-the-art text-to-image models, including Qwen-Image (Wu et al., 2025a), FLUX.2 (Black Forest Labs, 2026), and Z-Image (Cai et al., 2025). Given each source image, the MLLM proposes image-specific edit prompts constrained to be compatible with drag-like transformations. The source image and edit prompt are then passed to the editing model FLUX.2 (Black Forest Labs, 2026; fal, 2025) to synthesize the target image, and an MLLM-based binary filter discards low-quality or invalid pairs.

Semantic Keypoint Matching. From the filtered pairs, we recover each edit as semantic drag instructions. We prompt Gemini 3 Pro (Google, 2025) to identify keypoints that explain the transformation between the source and target images. Each drag consists of a handle point, a target point, and a semantic description of the matched element in both images. This converts the synthetic pair into

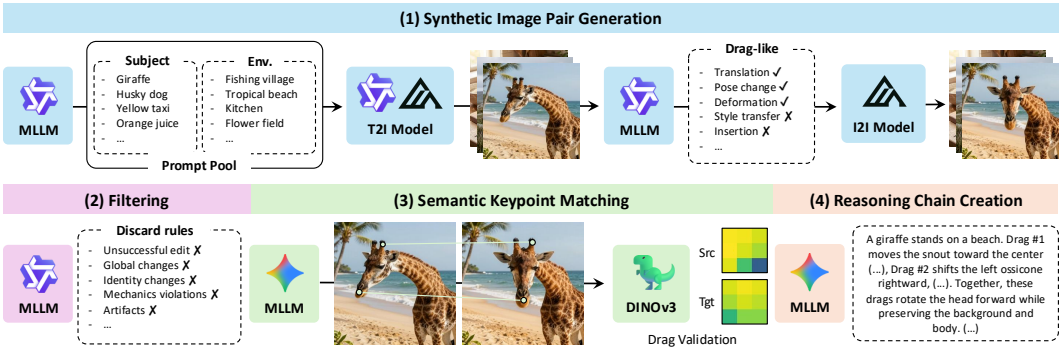

Figure 2: Dataset construction pipeline of the THINKDRAG dataset.

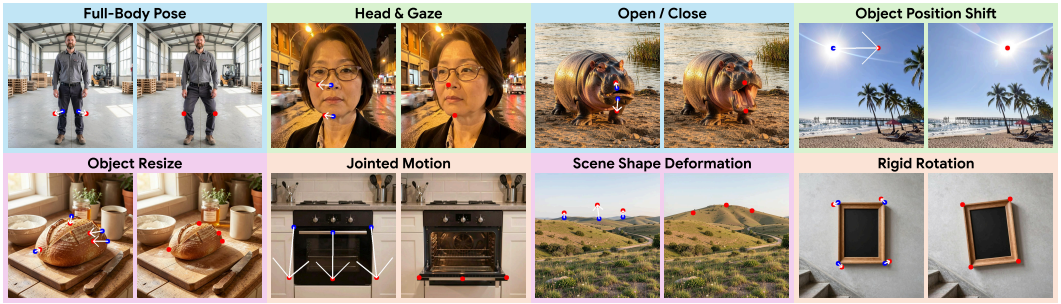

Figure 3: Diverse examples of the THINKDRAG dataset.

drag-based supervision while preserving the semantic role of each correspondence. Unlike video-derived annotations from optical flow, this procedure targets the user-facing edit intent rather than only low-level motion. Since MLLMs can hallucinate incorrect matches, we validate the predicted correspondences using DINOv3 (Siméoni et al., 2025) features and manual inspection.

Reasoning Chain Creation. Finally, we generate reasoning chains for each drag-edit example using Gemini 3 Flash (Google, 2025). The dataset construction pipeline makes the intended edit recoverable by construction: the MLLM has access to the source image, edited image, edit prompt, and semantic keypoint correspondence annotations. We use this information to prompt the MLLM to produce a reasoning chain that explains the source scene, interprets each drag in semantic terms, and summarizes the intended transformation while specifying which regions should remain unchanged. Since each source–target pair can be used bidirectionally, we generate reasoning chains for both edit directions. These traces provide supervision for reasoning-driven drag-based editing.

Dataset Statistics. Starting from approximately 193K generated pairs, MLLM filtering retains 97K, keypoint matching 50K, and verification yields the final dataset of approximately 11K source–target image pairs from 7K source images. Since each pair can be used in both directions, by swapping the source and target images and reversing the drag endpoints, this yields approximately 22K drag-based editing samples. During training, we further apply standard spatial augmentations such as horizontal flips and random crops. Figure 3 shows how the dataset spans diverse semantic drag transformations. Samples contain 2.15 drags on average (74% with multiple drags) and reasoning traces of 123 words on average. Edit category distribution and per-stage details are provided in Appendix C.

3.2 DRAG CONDITIONING IN A UNIFIED MULTIMODAL MODEL

We build on BAGEL (Deng et al., 2025), a unified multimodal model with a mixture-of-transformers architecture (Liang et al., 2025), and adapt it to drag-based editing by representing drag instructions through its existing multimodal interface. As shown in Figure 4, rather than introducing a separate point encoder or treating drags only as low-level geometric inputs, we express each drag through both structured text tokens and spatial endpoint markers. The text tokens expose the paired source and target coordinates, normalized to $[0, 1]$ with three decimals, to the model in a form compatible

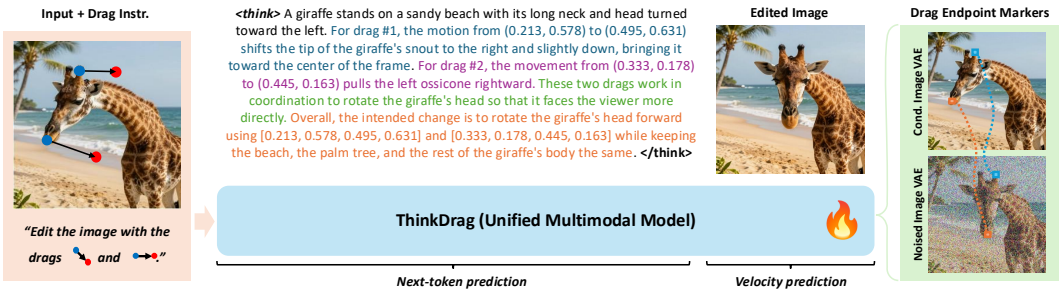

Figure 4: Overview of the THINKDRAG unified multimodal model pipeline.

with its language and visual understanding capabilities, allowing the drag instruction to be processed together with the image context and used for reasoning through the model's *understanding expert*. The endpoint markers complement this representation by injecting the same drag information at the corresponding VAE token locations, providing explicit spatial anchors for image generation through the model's *generation expert*. This design is motivated by the observation that each region of the generated image attends predominantly to the source region from which its content originates (Figure 8). This suggests that a drag is more likely to be executed correctly when the target location attends to its handle, a correspondence that textual coordinates convey only indirectly.

Drag Endpoint Markers. Let the drag instruction be $\mathcal{D} = \{(h_k, t_k)\}_{k=1}^{K}$, where each drag k consists of a handle point $h_k = (x_k^h, y_k^h)$ and a target point $t_k = (x_k^t, y_k^t)$. We associate each drag index k with a marker vector $\mathbf{m}_k = \mathbf{m}_{\text{shared}} + \mathbf{e}_k$, where $\mathbf{m}_{\text{shared}} \in \mathbb{R}^d$ is shared across all drags, initialized to zero and learned, and $\mathbf{e}_k \in \mathbb{R}^d$ is a fixed orthogonal slot embedding that preserves the identity of drag k. Let $\mathbf{z}_{\text{src}}^{(\ell)}$ and $\mathbf{z}_{\text{tgt}}^{(\ell)}$ denote the source and target VAE hidden features before layer ℓ. We inject drag markers by applying additive updates at the corresponding spatial locations:

$$\mathbf{z}_{\text{src}}^{(\ell)}(h_k) \leftarrow \mathbf{z}_{\text{src}}^{(\ell)}(h_k) + \alpha_{\text{src}}\, \mathbf{m}_k, \tag{1}$$

$$\mathbf{z}_{\text{tgt}}^{(\ell)}(t_k) \leftarrow \mathbf{z}_{\text{tgt}}^{(\ell)}(t_k) + \alpha_{\text{tgt}}\, \mathbf{m}_k, \tag{2}$$

where $\alpha_{\text{src}}, \alpha_{\text{tgt}} \in \mathbb{R}$ are learnable scalars for source and target endpoints, initialized to 1. We apply this injection before each layer, so the markers act as drag-specific encodings over the VAE tokens. Because the same marker $\mathbf{m}_k$ is added at both endpoints, the handle and target locations share a signature that biases the target location to attend to its handle (Appendix A.4). Unlike training-free methods that impose correspondence at inference (Yin et al., 2026; He et al., 2025), DEM learns it during training.

3.3 JOINT GENERATION MODELING

We represent an input image I_{src} and drag instruction $\mathcal{D}$ as an interleaved multimodal prompt P with source-image tokens, textual drag tokens, and endpoint marker conditioning. The model supports direct editing, which generates the edited image from P, and reasoning-based editing, which first generates a reasoning chain $R = \{r_i\}_{i=1}^{M}$ and then conditions image generation on (P, R).

For reasoning, we train the model with the standard autoregressive language-modeling objective

$$\mathcal{L}_{\text{think}}(\theta) = -\sum_{i=1}^{M} \log p_\theta(r_i \mid P, r_{<i}). \tag{3}$$

The reasoning chain describes the source scene, interprets the semantic role of each drag, and summarizes the intended transformation. This supervision teaches the model to ground sparse point constraints in the visual content before image generation.

For image generation, we follow the rectified-flow objective (Liu et al., 2023b) used by BAGEL (Deng et al., 2025). Given the target latent x_0 and noise $x_1 \sim \mathcal{N}(0, I)$, we sample $t \in [0, 1]$ and define $x_t = (1-t)x_0 + tx_1$ with target velocity $v = x_1 - x_0$. The image-generation loss is

$$\mathcal{L}_{\text{img}}(\theta) = \mathbb{E}_{t,x_0,x_1}\left[\|v - v_\theta(x_t, t, P, R)\|_2^2\right], \tag{4}$$

where R is used for reasoning-based editing and omitted for direct editing. The final objective is $\mathcal{L}(\theta) = \mathcal{L}_{\text{img}}(\theta) + \lambda_{\text{think}}\mathcal{L}_{\text{think}}(\theta)$, where λ_{think} controls the weight of reasoning supervision.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate on DragBench (Shi et al., 2024), the standard benchmark for drag-based image editing. We also introduce DragBench++, a new benchmark constructed using our dataset construction pipeline in Sec. 3.1 for image pair generation. DragBench++ contains 145 samples with manually annotated drag instructions and emphasizes scenarios that require higher-level scene understanding, object-level reasoning, and semantic interpretation, with more drags per sample and longer displacements than DragBench (Table 2). Unlike prior benchmarks, each input also includes a pseudo-ground-truth edited image as reference. Although drag-based editing is under-constrained and multiple outputs may satisfy the same instruction, this reference edit verifies that each example has at least one plausible solution. It also enables reference-based evaluation and a more controlled analysis of whether existing metrics reflect meaningful editing quality. Reference edits are used only for evaluation. Baselines that require an edit mask receive manually annotated masks.

Evaluation Metrics. We evaluate along two axes: editing accuracy and generation quality. For editing accuracy, we follow prior work (Shi et al., 2024) and report *Mean Distance* (MD), which measures the distance between each target point and the corresponding handle point tracked in the generated image using DIFT (Tang et al., 2023), restricted to the edit mask. On DragBench++, reference edits show that DIFT-based correspondence can be unreliable under semantic ambiguity or large structural changes. We therefore also report MD_{MLLM}, which uses the same distance definition but replaces DIFT with GPT-5.5 (OpenAI, 2026) for point tracking, computed for the strongest method of each category (Appendix B.2). Since GPT-5.5 is not used in our data pipeline, data construction and evaluation are decoupled. For generation quality, we report *Image Fidelity* (IF) and FID (Heusel et al., 2017). Following prior work, IF is computed as $1 - \text{LPIPS}$ (Zhang et al., 2018) between the input image and generated edit. Since fidelity to the input is ambiguous for editing tasks, where the output should preserve irrelevant content while changing the edited region, we also report FID on DragBench to measure overall visual quality and realism. For DragBench++, we instead report IF* against the reference edits as the primary fidelity metric.

4.2 COMPARISON WITH BASELINES

Quantitative Evaluation. We compare against state-of-the-art drag-based image editing methods with public official implementations: test-time optimization methods DragDiffusion (Shi et al., 2024), DragNoise (Liu et al., 2024a), and DragLoRA (Xia et al., 2025); one-step methods FastDrag (Zhao et al., 2024) and GeoDrag (Pu et al., 2026); and fine-tuned conditional generation methods InstantDrag (Shin et al., 2024) and LightningDrag (Shi et al., 2025). Table 1 reports quantitative results on DragBench and DragBench++, with THINKDRAG in direct mode. Across both benchmarks, THINKDRAG achieves the best generation quality, with the highest IF and lowest FID on DragBench, and the best IF and IF* on DragBench++. On DragBench, THINKDRAG remains competitive in point-following accuracy, although the strongest MD scores come from test-time optimization and one-step methods that explicitly enforce the point displacement at inference. The advantage is clearer on DragBench++, where edits require stronger semantic interpretation. THINKDRAG achieves the best MD and MD_{MLLM}. The reference row also shows that DIFT-based MD is less reliable in this setting, since even reference edits obtain a relatively high MD. MD_{MLLM} provides a more semantic estimate, under which THINKDRAG substantially outperforms the baselines. The improvement in IF* further shows that THINKDRAG does not improve point accuracy by producing implausible distortions; its outputs remain closest to the reference edits.

Qualitative Analysis. Figure 5 shows qualitative comparisons on DragBench and DragBench++. On DragBench, most methods handle relatively direct point movements, while THINKDRAG better preserves identity and local appearance during the edit. On DragBench++, the gap is more pronounced. Baselines often under-follow the drag instruction, introduce artifacts, or satisfy the point movement through implausible changes. In contrast, THINKDRAG more consistently produces edits that are

Table 1: Results on both benchmarks. TTO: test-time optimization, OS: one-step, FT: fine-tuned.

Method	Type	DragBench++ IF ↑	IF* ↑	MD ↓	MD_{MLLM} ↓	DragBench IF ↑	FID ↓	MD ↓	MD_{MLLM} ↓
Reference		–	–	43.42	27.38	–	–	–	–
DragDiffusion	TTO	0.833	0.819	71.69	–	0.882	38.90	30.88	–
DragNoise	TTO	0.867	0.829	74.44	–	0.900	44.39	32.70	–
DragLoRA	TTO	0.808	0.810	48.36	59.98	0.874	43.35	23.20	**36.54**
FastDrag	OS	0.780	0.771	79.02	–	0.859	49.34	27.45	–
GeoDrag	OS	0.767	0.759	70.55	74.78	0.842	53.96	**22.77**	39.46
InstantDrag	FT	0.821	0.793	76.45	–	0.882	45.10	30.78	–
LightningDrag	FT	0.842	0.836	57.49	57.28	0.888	40.73	26.36	40.99
ThinkDrag	FT	**0.875**	**0.905**	**47.66**	**34.24**	**0.918**	**30.05**	28.84	37.10

Table 2: Benchmark statistics.

	DragBench++ (n=145)	DragBench (n=205)
Drags/sample	2.51	1.70
Multi-drag	80.7%	47.3%
Displacement (px)	107.8	58.5
Reference edit	✓	×

Table 3: User study: share of overall-preference votes (%) on each benchmark.

Method	DragBench++	DragBench
DragLoRA	6.3	13.3
GeoDrag	6.3	8.7
LightningDrag	7.7	17.0
ThinkDrag	**79.7**	**61.0**

both spatially grounded and semantically plausible. Figure 6 compares direct generation with the reasoning mode. Direct generation already benefits from semantic drag supervision, but reasoning can provide an additional advantage when the transformation is under-specified by the points alone. In such cases, explicitly interpreting the edit before generation helps resolve ambiguity and coordinate non-local changes, such as opening a refrigerator door or making waves taller in a painting.

User Study. We conduct a user study with 30 participants, each answering 20 questions sampled evenly from DragBench and DragBench++. For each question, participants see the input image, the drag instruction, and anonymized outputs of DragLoRA, GeoDrag, LightningDrag, and ThinkDrag in random order, and select the preferred result for edit accuracy, generation quality, and overall preference. As shown in Table 3, participants clearly prefer ThinkDrag, with 80% of the overall-preference votes on DragBench++ and 61% on DragBench, an external benchmark independent of our data pipeline, and similar shares for the other criteria (Appendix B.3). On DragBench, ThinkDrag also receives 62% of the edit-accuracy votes despite a higher DIFT-based MD than all three baselines, suggesting that MD does not fully capture semantic correctness.

4.3 Joint Text-Drag Conditioning

In the default setting, ThinkDrag infers the intended edit from the input image and drag points alone, either directly or through its reasoning chain. Joint text-drag conditioning provides a complementary mode where the user specifies part of the semantic intent. This is useful when the same point motion admits multiple plausible interpretations, since drag points provide spatial anchors while text clarifies the desired transformation. Prior work has explicitly targeted this setting with specialized mechanisms (Jiang et al., 2025). In ThinkDrag, it follows naturally from the unified multimodal formulation. During training, we condition the model on both drag endpoints and the corresponding text edit description for 10% of samples, allowing it to learn text-drag interaction without making text mandatory at inference time. As shown in Figure 7, the same drag instruction can be resolved into different valid edits when paired with different text prompts, such as widening a flower versus opening its petals, or enlarging the candle body versus extending the flame.

4.4 Ablation Studies

Reasoning and Language-based Conditioning. Table 4 ablates reasoning and user-provided text conditioning. The reasoning mode improves reference-based fidelity over direct generation and

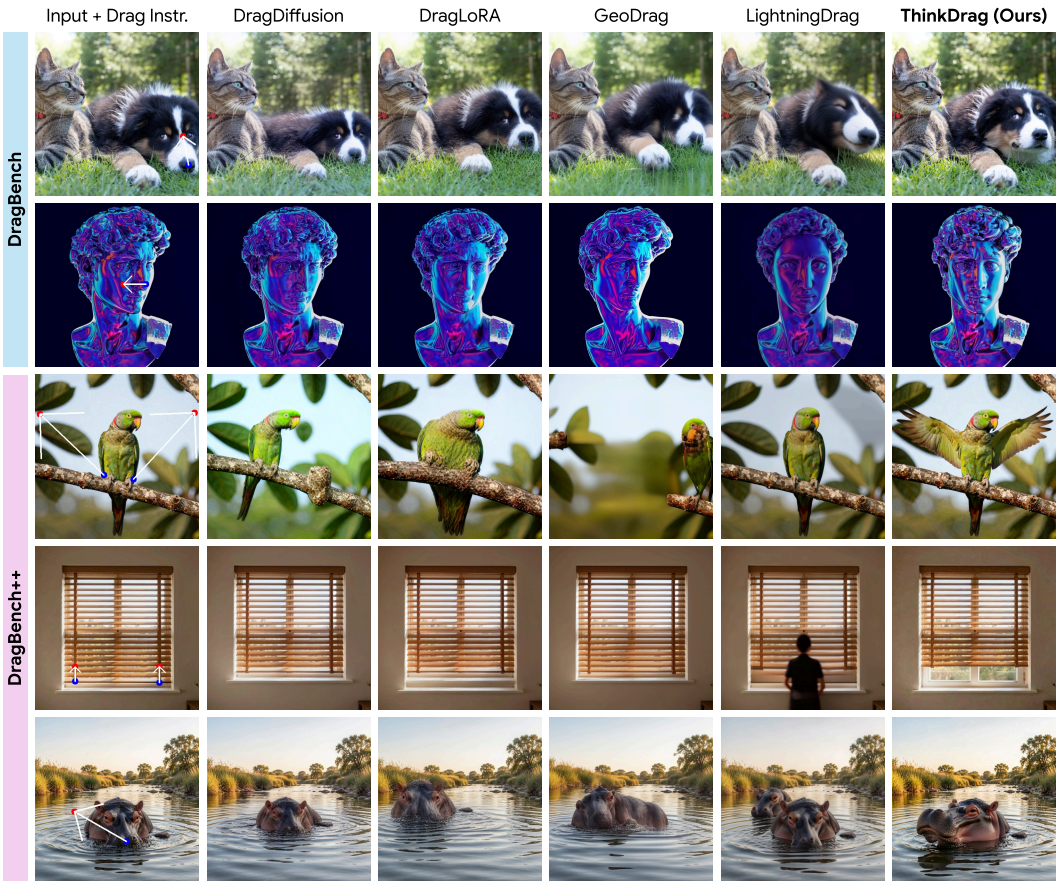

Figure 5: Qualitative comparison with other models on DragBench and DragBench++.

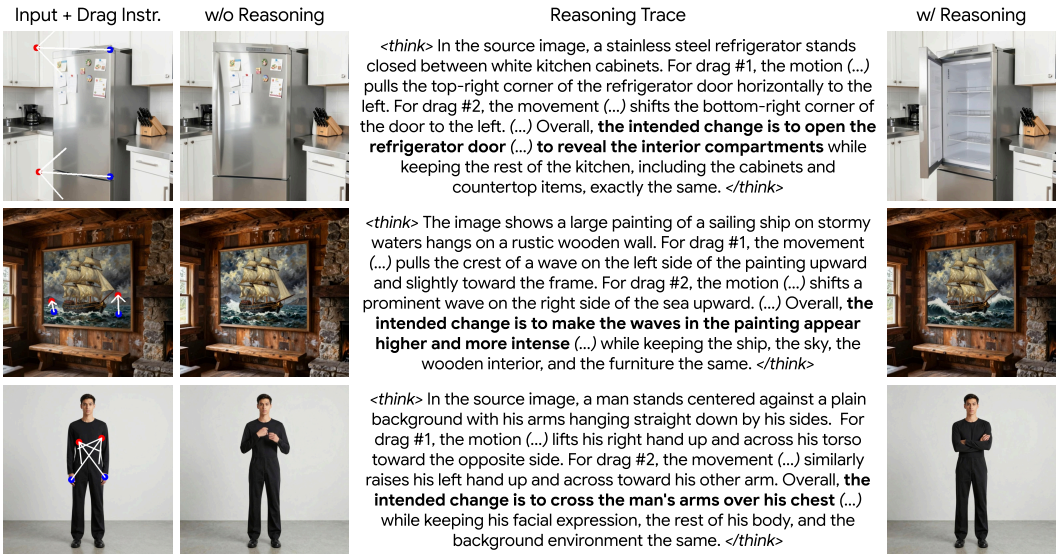

Figure 6: Qualitative comparison between direct generation and the reasoning mode on DragBench++.

slightly lowers MD_{MLLM}, approaching the setting in which the reference edit prompt removes the ambiguity of the drag instruction. The prompt itself changes MD_{MLLM} only slightly, indicating that the drags already determine the spatial outcome, while the interpretation mainly affects how faithfully the intended edit is realized. Per sample, reasoning yields the lower MD_{MLLM} more often on multi-

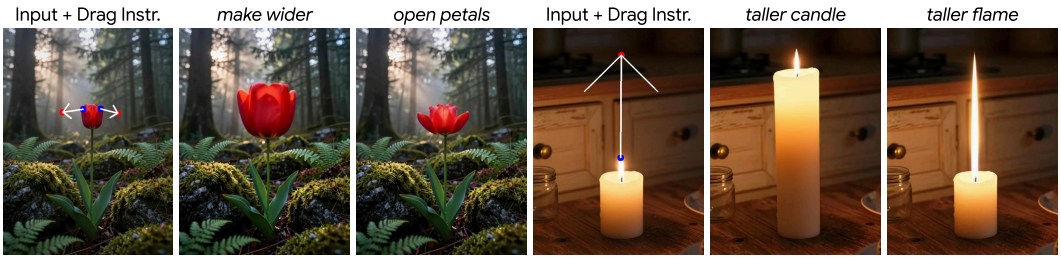

Figure 7: Joint text-drag conditioning. Given the same drag instruction, different text prompts guide the model toward different valid edits, while the drags keep each edit spatially grounded.

Table 4: Reasoning and text conditioning on DragBench++ (MD_{MLLM} with GPT-5.5).

Method	IF* ↑	MD_{MLLM} ↓
THINKDRAG	0.905	34.24
w/ Reasoning	0.911	33.92
w/ Reference Prompt	0.914	33.82

Table 5: Component analysis on DragBench, from zero-shot base models to the full model.

Method	IF ↑	FID ↓	MD ↓
BAGEL (zero-shot)	0.866	51.66	58.92
FLUX.2 (zero-shot)	**0.933**	32.30	58.56
THINKDRAG	0.918	**30.05**	**28.84**
w/o DEM	0.911	30.99	45.42

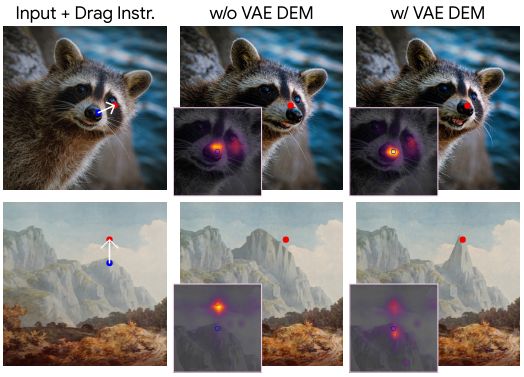

Figure 8: Qualitative ablation of DEM. Insets show the averaged attention from the target point to the source image. With DEM, it peaks closer to the handle and the edit is more precise.

drag edits (56%) than on single-drag edits (36%; Appendix A.2). Figure 6 shows examples where reasoning helps resolve edits that require a more explicit interpretation of the intended transformation.

Drag Endpoint Markers. Table 5 separates the contributions of the base model, the data, and DEM. Neither BAGEL nor the editing model FLUX.2 follows drag instructions zero-shot, and FLUX.2 attains a high IF largely because it leaves the image unchanged. Training on our dataset without DEM substantially reduces MD, and adding DEM reduces it much further with similar IF and FID. This gap indicates that the structured text representation alone is insufficient for precise spatial control at this scale, and that endpoint markers provide important grounding. Training with textual drag coordinates nevertheless yields an emergent ability to ground coordinates in image content (Appendix A.3). Figure 8 further shows that without DEM the model still captures the overall intent of the drag instruction but lacks the spatial precision needed to align the edited content with the target point. The attention maps provide insight into this. Without DEM, the attention of the target location rarely peaks at the handle, resulting in the reduced precision observed. With DEM, the attention peaks at the handle for 51% of drags on DragBench++, compared to 6% without (Appendix A.4).

5 CONCLUSION

We introduced THINKDRAG, a unified multimodal framework for drag-based image editing with visual reasoning. By training on a new dataset of high-quality semantic drag transformations and reasoning traces, and by grounding each drag with Drag Endpoint Markers, THINKDRAG learns to interpret sparse point constraints in relation to scene content and object structure, rather than treating them only as geometric displacements. We also introduced DragBench++, a benchmark of more challenging drag-based editing scenarios with reference edits. Experiments show that THINKDRAG achieves state-of-the-art generation quality while maintaining competitive point-following precision, with the reasoning mode providing additional interpretability, particularly for ambiguous multi-drag edits.

AI USE STATEMENT

Generative AI is part of this work's method and evaluation, as described in the paper. An MLLM (Qwen3-VL) writes source-image and edit prompts, and text-to-image and editing models (Qwen-Image, FLUX.2, Z-Image) synthesize the image pairs. Gemini 3 Pro estimates semantic keypoints and Gemini 3 Flash writes reasoning traces for the dataset (Sec. 3.1). For evaluation, GPT-5.5 tracks points for MD_{MLLM} (Appendix B.2), and Qwen3.6-27B judges reasoning traces and coordinate grounding (Appendix A.2 and A.3). Generated data was filtered automatically, and the final dataset was verified with feature-based checks and manual inspection. Beyond these uses, we used AI assistants to polish the writing, such as finding grammar or spelling mistakes and improving readability, and to assist with code development, such as preparing experiment environments and finding and fixing bugs. The authors reviewed and verified AI-assisted text and code. We take responsibility for the final content of this work, including the content produced with the aid of generative AI.

ETHICS STATEMENT

Our user study collected anonymous preference judgments over generated images from 30 volunteer participants. No personal or sensitive information was collected, and the study did not require IRB approval. The instructions shown to participants are provided in Appendix B.3. The THINKDRAG dataset and DragBench++ consist of synthetic images produced by generative models. As with other image editing methods, THINKDRAG could be misused to create misleading manipulated images or alter visual evidence. We evaluate it only as a research system, and responsible deployment should include provenance tracking or watermarking and disclosure to users.

REPRODUCIBILITY STATEMENT

The paper and appendix describe each part of this work in detail. The dataset construction and its per-stage statistics are described in Sec. 3.1, with the edit categories and the model and key settings of each stage in Appendix C. The model, drag conditioning, and training objectives are specified in Sec. 3, and the training and inference hyper-parameters and DEM implementation details are given in Appendix B.1. The benchmarks and metrics are defined in Sec. 4, with the MLLM tracking protocol and prompt in Appendix B.2. The attention analysis and the user study are detailed in Appendix A.4 and Appendix B.3. The code, weights, the THINKDRAG dataset, and DragBench++ will be released.

REFERENCES

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Black Forest Labs. FLUX. 1 kontext: Flow matching for in-context image generation and editing in latent space. *arXiv preprint arXiv:2506.15742*, 2025.

Black Forest Labs. FLUX.2 [klein] 9B, 2026.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. InstructPix2Pix: Learning to Follow Image Editing Instructions. In *CVPR*, 2023.

Huanqia Cai, Sihan Cao, Ruoyi Du, Peng Gao, Steven Hoi, Zhaohui Hou, Shijie Huang, Dengyang Jiang, Xin Jin, Liangchen Li, et al. Z-image: An efficient image generation foundation model with single-stream diffusion transformer. *arXiv preprint arXiv:2511.22699*, 2025.

Chameleon Team. Chameleon: Mixed-modal early-fusion foundation models. *arXiv preprint arXiv:2405.09818*, 2024.

Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-pro: Unified multimodal understanding and generation with data and model scaling. *arXiv preprint arXiv:2501.17811*, 2025.

Gayoon Choi, Taejin Jeong, Sujung Hong, and Seong Jae Hwang. Dragtext: Rethinking text embedding in point-based image editing. In *WACV*, 2025.

Xing Cui, Peipei Li, Zekun Li, Xuannan Liu, Yueying Zou, and Zhaofeng He. Localize, understand, collaborate: Semantic-aware dragging via intention reasoner. In *NeurIPS*, 2024.

Chaorui Deng, Deyao Zhu, Kunchang Li, Chenhui Gou, Feng Li, Zeyu Wang, Shu Zhong, Weihao Yu, Xiaonan Nie, Ziang Song, et al. Emerging properties in unified multimodal pretraining. *arXiv preprint arXiv:2505.14683*, 2025.

Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion Models Beat GANs on Image Synthesis. In *NeurIPS*, 2021.

fal. FLUX.2 [dev] turbo LoRA, 2025.

Google. Gemini 3 developer guide, 2025.

Huiguo He, Pengyu Yan, Ziqi Yi, Weizhi Zhong, Zheng Liu, Yejun Tang, Huan Yang, Kun Gai, Guanbin Li, and Lianwen Jin. ContextDrag: Precise drag-based image editing via context-preserving token injection and position-consistent attention. *arXiv preprint arXiv:2512.08477*, 2025.

Runze He, Yiji Cheng, Tiankai Hang, Zhimin Li, Yu Xu, Zijin Yin, Shiyi Zhang, Wenxun Dai, Penghui Du, Ao Ma, et al. Re-align: Structured reasoning-guided alignment for in-context image generation and editing. *arXiv preprint arXiv:2601.05124*, 2026.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-Prompt Image Editing with Cross-Attention Control. In *ICLR*, 2023.

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 *NeurIPS*, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *NeurIPS*, 2020.

Xingzhong Hou, Boxiao Liu, Yi Zhang, Jihao Liu, Yu Liu, and Haihang You. Easydrag: Efficient point-based manipulation on diffusion models. In *CVPR*, 2024.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *ICLR*, 2022.

Meng-Ting Jhong, Tai-Ming Huang, Shang-Fu Chen, Wen-Huang Cheng, and Kai-Lung Hua. Dragonite: Single-step drag-based image editing with geometric-semantic guidance. In *WACV*, 2026.

Ziqi Jiang, Zhen Wang, and Long Chen. CLIPDrag: Combining text-based and drag-based instructions for image editing. In *ICLR*, 2025.

Weixin Liang, LILI YU, Liang Luo, Srini Iyer, Ning Dong, Chunting Zhou, Gargi Ghosh, Mike Lewis, Wen-tau Yih, Luke Zettlemoyer, et al. Mixture-of-transformers: A sparse and scalable architecture for multi-modal foundation models. In *ICLR 2025 Workshop*, 2025.

Chao Liao, Liyang Liu, Xun Wang, Zhengxiong Luo, Xinyu Zhang, Wenliang Zhao, Jie Wu, Liang Li, Zhi Tian, and Weilin Huang. Mogao: An omni foundation model for interleaved multi-modal generation. *arXiv preprint arXiv:2505.05472*, 2025.

Xiaojian Lin, Hanhui Li, Yuhao Cheng, Yiqiang Yan, and Xiaodan Liang. GDrag: Towards general-purpose interactive editing with anti-ambiguity point diffusion. In *ICLR*, 2025.

Pengyang Ling, Lin Chen, Pan Zhang, Huaian Chen, Yi Jin, and Jinjin Zheng. Freedrag: Feature dragging for reliable point-based image editing. In *CVPR*, 2024.

Haofeng Liu, Chenshu Xu, Yifei Yang, Lihua Zeng, and Shengfeng He. Drag your noise: Interactive point-based editing via diffusion semantic propagation. In *CVPR*, 2024a.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual Instruction Tuning. In *NeurIPS*, 2023a.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *CVPR*, 2024b.

Xingchao Liu, Chengyue Gong, et al. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023b.

Jingyi Lu and Kai Han. Inpaint4Drag: Repurposing inpainting models for drag-based image editing via bidirectional warping. In *ICCV*, 2025.

Jingyi Lu, Xinghui Li, and Kai Han. Regiondrag: Fast region-based image editing with diffusion models. In *ECCV*, 2024.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *ICLR*, 2022.

Chong Mou, Xintao Wang, Jiechong Song, Ying Shan, and Jian Zhang. DragonDiffusion: Enabling drag-style manipulation on diffusion models. In *ICLR*, 2024.

Shen Nie, Hanzhong Allan Guo, Cheng Lu, Yuhao Zhou, Chenyu Zheng, and Chongxuan Li. The blessing of randomness: SDE beats ODE in general diffusion-based image editing. In *ICLR*, 2024.

OpenAI. GPT-5.5 system card. `https://openai.com/index/gpt-5-5-system-card`, April 2026.

Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimitra Meka, and Christian Theobalt. Drag your gan: Interactive point-based manipulation on the generative image manifold. In *ACM SIGGRAPH*, 2023.

Xinyu Pu, Hongsong Wang, Jie Gui, and Pan Zhou. Dragging with geometry: From pixels to geometry-guided image editing. In *ICLR*, 2026.

Luozheng Qin, GONG JIA, Yuqing Sun, Tianjiao Li, Haoyu Pan, Mengping Yang, Xiaomeng Yang, Chao Qu, Zhiyu Tan, and Hao Li. Uni-CoT: Towards unified chain-of-thought reasoning across text and vision. In *ICLR*, 2026.

Hao Shao, Shengju Qian, Han Xiao, Guanglu Song, Zhuofan Zong, Letian Wang, Yu Liu, and Hongsheng Li. Visual cot: Advancing multi-modal language models with a comprehensive dataset and benchmark for chain-of-thought reasoning. In *NeurIPS*, 2024.

Yujun Shi, Chuhui Xue, Jun Hao Liew, Jiachun Pan, Hanshu Yan, Wenqing Zhang, Vincent YF Tan, and Song Bai. Dragdiffusion: Harnessing diffusion models for interactive point-based image editing. In *CVPR*, 2024.

Yujun Shi, Jun Hao Liew, Hanshu Yan, Vincent Y. F. Tan, and Jiashi Feng. LightningDrag: Lightning fast and accurate drag-based image editing emerging from videos. In *ICML*, 2025.

Joonghyuk Shin, Daehyeon Choi, and Jaesik Park. Instantdrag: Improving interactivity in drag-based image editing. In *ACM SIGGRAPH Asia*, 2024.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising Diffusion Implicit Models. In *ICLR*, 2021.

Luming Tang, Menglin Jia, Qianqian Wang, Cheng Perng Phoo, and Bharath Hariharan. Emergent correspondence from image diffusion. In *NeurIPS*, 2023.

Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, et al. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*, 2024.

Yaoting Wang, Shengqiong Wu, Yuecheng Zhang, Shuicheng Yan, Ziwei Liu, Jiebo Luo, and Hao Fei. Multimodal chain-of-thought reasoning: A comprehensive survey. *arXiv preprint arXiv:2503.12605*, 2025.

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025a.

Chengyue Wu, Xiaokang Chen, Zhiyu Wu, Yiyang Ma, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, Chong Ruan, et al. Janus: Decoupling visual encoding for unified multimodal understanding and generation. In *CVPR*, 2025b.

Siwei Xia, Li Sun, Tiantian Sun, and Qingli Li. DragLoRA: Online optimization of LoRA adapters for drag-based image editing in diffusion model. In *ICML*, 2025.

Jinheng Xie, Weijia Mao, Zechen Bai, David Junhao Zhang, Weihao Wang, Kevin Qinghong Lin, Yuchao Gu, Zhijie Chen, Zhenheng Yang, and Mike Zheng Shou. Show-o: One single transformer to unify multimodal understanding and generation. In *ICLR*, 2025a.

Jinheng Xie, Zhenheng Yang, and Mike Zheng Shou. Show-O2: Improved native unified multimodal models. *arXiv preprint arXiv:2506.15564*, 2025b.

Zixin Yin, Xili Dai, Duomin Wang, Xianfang Zeng, Lionel Ni, Gang YU, and Heung-Yeung Shum. LazyDrag: Enabling stable drag-based editing on multi-modal diffusion transformers via explicit correspondence. In *ICLR*, 2026.

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

Zewei Zhang, Huan Liu, Jun Chen, and Xiangyu Xu. GoodDrag: Towards good practices for drag editing with diffusion models. In *ICLR*, 2025.

Xuanjia Zhao, Jian Guan, Congyi Fan, Dongli Xu, Youtian Lin, Haiwei Pan, and Pengming Feng. FastDrag: Manipulate anything in one step. In *NeurIPS*, 2024.

Chunting Zhou, LILI YU, Arun Babu, Kushal Tirumala, Michihiro Yasunaga, Leonid Shamis, Jacob Kahn, Xuezhe Ma, Luke Zettlemoyer, and Omer Levy. Transfusion: Predict the next token and diffuse images with one multi-modal model. In *ICLR*, 2025.

Yuan Zhou, Junbao Zhou, Qingshan Xu, Kesen Zhao, Yuxuan Wang, Hao Fei, Richang Hong, and Hanwang Zhang. Dragnext: Rethinking drag-based image editing. In *AAAI*, 2026.

A Additional Experiments

A.1 Additional Visual Results

Figure 9 provides additional qualitative comparisons on DragBench and DragBench++. For easier inspection of editing accuracy, we overlay the target points on the generated images. These examples further illustrate the ability of ThinkDrag to produce spatially accurate edits while preserving image identity and visual quality.

A.2 Reasoning Analysis

Table 6 reports the win rate of reasoning over direct generation on DragBench++, where a mode wins a sample if its output has the lower MD_{MLLM}. Overall, the two modes win about equally often (reasoning wins 75 of the 145 samples, 51.7%), but reasoning wins on 56% of multi-drag edits compared with 36% of single-drag edits. Table 7 shows win rates across edit types. To assess faithfulness, Qwen3.6-27B judges whether each generated trace describes the same edit as the annotated edit prompt, and as a control we construct a deliberately incorrect edit prompt for each sample. The traces are judged consistent with the annotated prompt for 78.6% of samples and with the incorrect prompt for 2.8%. Since an ambiguous drag can admit several valid interpretations, the first number is a conservative estimate of trace correctness.

Table 6: Win rate of reasoning over direct generation on DragBench++ by drag number (MD_{MLLM}).

Number of drags	1	2	3	$\geq$4	All
Samples	28	51	37	29	145
Reasoning wins	10	25	24	16	75
Win rate	35.7%	49.0%	64.9%	55.2%	51.7%

Table 7: Win rate of reasoning over direct generation on DragBench++ by edit type, determined by the first verb of the edit instruction. Verbs with fewer than five samples are grouped under Other.

Edit type	Open	Adjust	Move	Rotate	Raise	Other
Samples	20	23	20	19	19	44
Win rate	60.0%	52.2%	50.0%	47.4%	42.1%	54.5%

A.3 Coordinate Grounding

Training with textual drag coordinates gives rise to an emergent ability to ground coordinates in image content. To evaluate this ability, we manually annotate the object and object part at 250 handle points from DragBench++. Given the image and a normalized coordinate, each model is asked to identify the object and the object part at that position, and Qwen3.6-27B compares the answers with the labels. Table 8 reports the results. Figure 10 shows examples where ThinkDrag identifies the correct object and part at the queried coordinate, while BAGEL describes an unrelated object.

Table 8: Coordinate grounding on 250 annotated handle points from DragBench++.

Model	Correct	Accuracy
BAGEL	95	38.0%
ThinkDrag	182	72.8%

A.4 Attention Analysis: Target Point to Handle Correspondence

We process every DragBench and DragBench++ sample with and without DEM from the same initial noise. For each drag, we extract the attention from the generated-image token at t_k to the source VAE tokens, averaged over all heads, layers, and denoising steps. Since each region of the generated

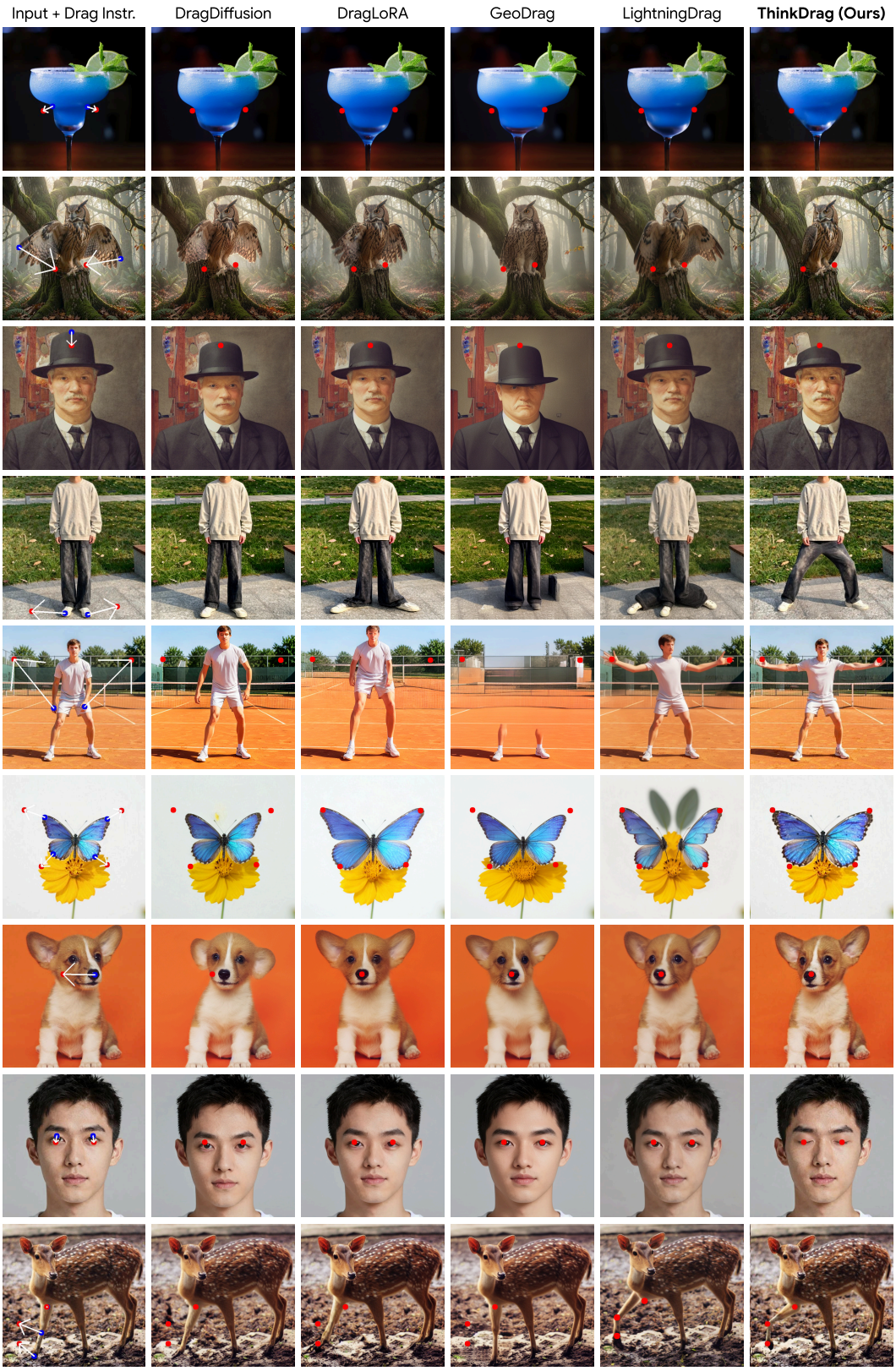

Figure 9: Additional visual comparison with other models on DragBench and DragBench++. Target points are overlaid on the generated images to facilitate analysis of editing accuracy.

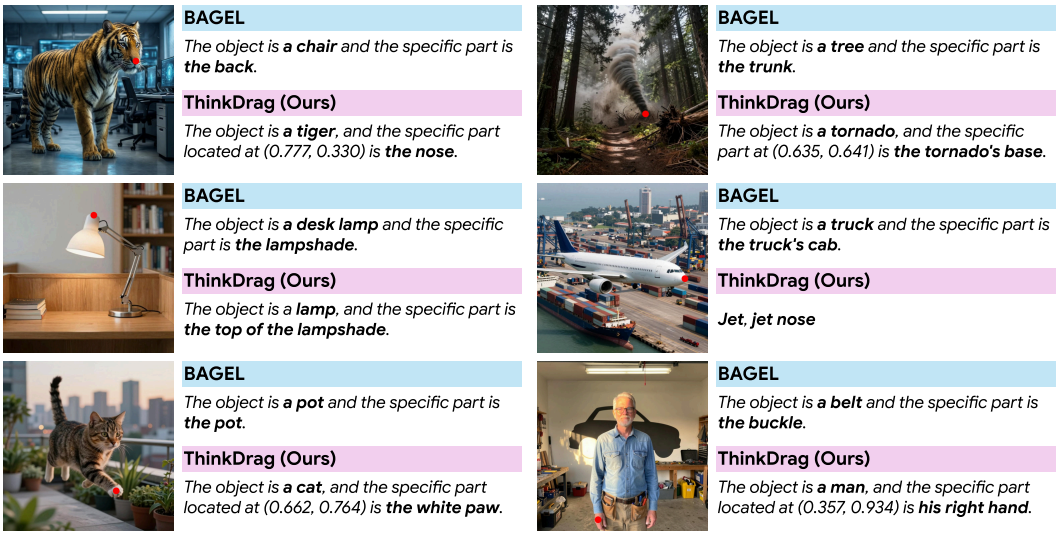

Figure 10: Grounding of textual coordinates. Each input image shows the queried normalized coordinate as a red point. Given the image and the coordinate, BAGEL often describes an unrelated object, whereas THINKDRAG identifies the correct object and part at the queried location.

image draws its content predominantly from the source region it attends to most, the location of this attention peak is a good indicator of which source content matters most for the transfer to the target point. A drag executed as intended should therefore have its attention peak at the corresponding handle. A drag counts as a hit if the attention peak lies in the latent cell of h_k or one of its eight neighbors, where one cell corresponds to 16 pixels. Drags shorter than three cells are excluded because their handle and target patches overlap. Table 9 summarizes the results. DEM substantially increases the fraction of drags whose attention peaks at the handle on both benchmarks.

Table 9: Fraction of drags whose cross-image attention from the target point peaks at the handle (within one latent cell), with and without DEM.

Benchmark	Drags	w/o DEM	w/ DEM
DRAGBENCH++	294	6.1%	50.7%
DRAGBENCH	226	13.3%	46.0%

A.5 RUNTIME AND MEMORY MEASUREMENT

We report runtime and peak GPU memory usage in Table 10, measured on a single NVIDIA H200 GPU. The reasoning mode increases runtime due to autoregressive reasoning-chain generation before image synthesis, while peak memory remains nearly unchanged.

Table 10: Runtime and peak memory usage.

Method	Runtime (s)	Peak Memory (GB)
THINKDRAG	14.3	35.5
THINKDRAG w/ Reasoning	22.9	35.6

B Additional Experimental Details

B.1 Implementation Details

We implement ThinkDrag using BAGEL (Deng et al., 2025) as the base unified multimodal model. We train LoRA adapters (Hu et al., 2022) of rank 256 on the attention and MLP projections of the base transformer for 100K steps, using AdamW with a learning rate of 2×10^{-5} and a cosine schedule. The effective batch size is approximately 16 samples, distributed across 4 NVIDIA H200 GPUs. The training procedure takes around 60 hours. We set the reasoning loss weight to $\lambda_{\text{think}} = 0.25$. We support up to 10 drag pairs per image during training; this limit is configurable. Unless otherwise specified, inference uses 50 sampling steps with classifier-free guidance scales of 3.0 for text and 1.5 for image conditioning. The learned endpoint scales remain close to their initialization ($\alpha_{\text{tgt}} \approx 1.16$, $\alpha_{\text{src}} \approx 1.12$). Images are processed at their native aspect ratio, with training images mostly square and the remainder in 4:3 and 16:9 in both orientations. The number of drags per sample (1–7) follows the annotation without random subsampling. The arrows and points drawn in our figures serve only as visualization, since the model receives drags solely as text and DEM. For the zero-shot rows of Table 5, BAGEL and FLUX.2 receive the drags as structured text instructions.

B.2 MLLM-based Evaluation

For the MD_{MLLM} metric, we use GPT-5.5 (OpenAI, 2026) to track source handle points in the generated image. The model receives the source image, the generated image, and the source handle-point coordinates. It is prompted to return the corresponding physical or semantic point locations in the generated image using normalized coordinates in $[0, 1000]$. Due to API cost, we compute MD_{MLLM} for the strongest method of each category (DragLoRA, GeoDrag, and LightningDrag), the same baselines as in the user study, and mark the remaining methods with "–" in Table 1. Figure 11 compares this protocol with DIFT-based tracking. Although MLLM-based tracking is also imperfect, it is more robust in challenging cases involving semantic ambiguity, large structural changes, or visually similar distractors, where local feature matching can select an incorrect correspondence.

On the reference edits of DragBench++, which satisfy their drags by construction, DIFT reports a mean distance of 43.42, whereas GPT-5.5 reports 27.38. Most of the DIFT distance on this benchmark therefore reflects tracking failures under large structural changes. On DragBench, where edits are more local, DIFT remains accurate, so we report both trackers and regard them as complementary.

Prompt for MLLM-based point tracking

```
You are a highly precise vision-language model. You see two images in order:
SOURCE then GENERATED.

Your task is point tracking. The user provides handle point coordinates in the
SOURCE image. For each provided source handle point, return the corresponding
same physical/semantic point location in the GENERATED image.

Critical rules:
- Track the exact same point or landmark.
- The source handle point may have moved, deformed, or changed appearance in
  the generated image.
- Use nearby object geometry, part boundaries, corners/tips/joints/hinges,
  texture, and semantic context to identify the correspondence.
- If a point is occluded or not visible in the GENERATED image, set its target
  to the nearest visible point.
- Coordinates are normalized to 0-1000: (0,0) top-left, (1000,1000)
  bottom-right of the GENERATED image.
- Return exactly one entry per input point, preserving the input point index.
- Do not include markdown or prose.

Output STRICT JSON:
{"tracked_points":[{"index":0,"target":[x,y]}]}
```

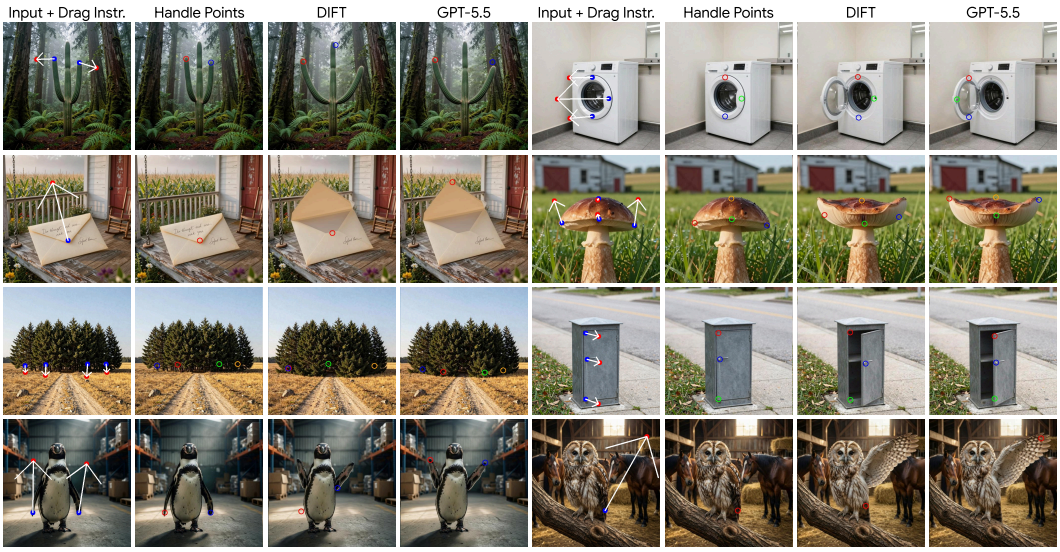

Figure 11: Comparison between DIFT-based and MLLM-based point tracking. MLLM-based tracking is more robust in challenging cases involving large structural changes or distractors.

B.3 User Study

Each of the 30 participants answered 20 questions sampled evenly from DragBench and DragBench++. For each question, edited results from different methods were anonymized and shown in random order. *Edit accuracy* measures how well the output follows the drag instruction and moves the intended content toward the target location. *Generation quality* measures whether the edited image remains faithful to the input, preserving identity, appearance, and unrelated regions while avoiding artifacts. *Overall preference* considers both criteria. We compare against DragLoRA, GeoDrag, and LightningDrag, the strongest baseline of each category. Table 11 reports all three criteria for each benchmark and for both benchmarks combined. Over both benchmarks, ThinkDrag receives about 70% of the votes under each criterion and the most votes on 18 of the 20 questions.

Table 11: User study vote share (%) by benchmark and criterion. Each benchmark contributes 10 of the 20 questions.

Benchmark	Method	Edit accuracy	Generation quality	Overall preference
DragBench++	DragLoRA	7.7	6.3	6.3
	GeoDrag	7.0	6.7	6.3
	LightningDrag	7.7	8.3	7.7
	ThinkDrag	**77.7**	**78.7**	**79.7**
DragBench	DragLoRA	13.7	15.7	13.3
	GeoDrag	7.7	8.0	8.7
	LightningDrag	16.7	15.7	17.0
	ThinkDrag	**62.0**	**60.7**	**61.0**
Both	DragLoRA	10.7	11.0	9.8
	GeoDrag	7.3	7.3	7.5
	LightningDrag	12.2	12.0	12.3
	ThinkDrag	**69.8**	**69.7**	**70.3**

Figure 12 shows an example question from the study interface. We provide the exact instructions shown to participants below.

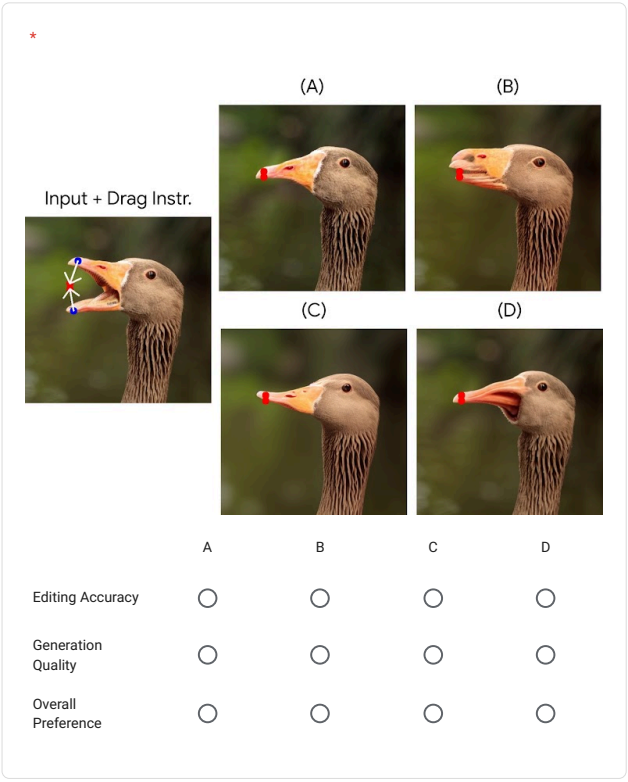

Figure 12: Example question from the user study interface.

User study instructions

```
User Study on Drag-based Image Editing

You will be shown 20 questions with an input image, a drag instruction,
and edited results.

For each example, answer the following three questions:

Editing Accuracy:
Select the result that best follows the drag instruction. Focus on whether
image content moves from the handle point to the target point and the intended
edit is achieved.

Generation Quality:
Select the result with the best visual quality for the given edit. Focus on
image quality, realism, lack of artifacts, and preservation of regions that
should not change.

Overall Preference:
Select the result you prefer overall, considering both editing accuracy and
generation quality.

The edited results are shown in random order.
```

C Dataset Details

The final dataset covers rotations (28.9%), raising and lowering (24.3%), translations (16.6%), opening and closing (10.8%), deformations (5.5%), scaling (3.8%), and other edits. DragBench++

image pairs are built with the same pipeline from source images that do not appear in the training data, and their drags are annotated manually. To check the edit masks provided to baselines, we measure LPIPS between each source image and its reference edit, which is 0.118 inside the mask and 0.018 outside it, so the masks cover the edited regions. Table 12 summarizes the model and key settings of each stage of the pipeline.

Table 12: Stages of the dataset construction pipeline.

Stage	Model	Output and key details
Source prompts	Qwen3-VL	One-sentence scene prompt of 45–110 words from a subject and a location drawn from a pool. The subject is dominant, object parts appear with their parent object, and 8–14 location details vary the scene.
Source images	Qwen-Image, FLUX.2, Z-Image	One image per prompt, square in about 70% of cases and otherwise landscape or portrait, as chosen by the prompt model.
Edit proposal	Qwen3-VL	Five edit prompts per source image, grounded in visible parts and achievable with a few drags: articulation (*e.g.*, opening a door, raising a limb), repositioning of an object, or deformation of a boundary. Insertion, removal, and style, lighting, or material changes are excluded.
Target images	FLUX.2	One edited image per source image and edit prompt.
Filtering	Qwen3-VL	Binary decision on each pair. A pair is kept only if the edit is clearly achieved in the intended direction, identity and unrelated content are preserved, the change is local, and there are no artifacts, anatomy or mechanics violations, or global color and lighting shifts.
Keypoints	Gemini 3 Pro	One to seven keypoints on moved and anchor parts, located in both images on a 0–1000 grid, preferring corners, tips, and joints. Pairs judged unsuitable receive no keypoints and are discarded, as are drags shorter than 32 grid units.
Verification	DINOv3, manual	Feature consistency in local windows around each handle and target point, followed by manual inspection of the source image, target image, and drags.
Reasoning	Gemini 3 Flash	One paragraph of 50–200 tokens per edit direction. It describes each drag and its local effect, relates drags that act together, and ends with a sentence stating what changes and what stays the same. Coordinates are written as placeholders and filled in at training time.

D Limitations and Future Work

While ThinkDrag achieves strong drag-based editing performance across both standard and challenging scenarios, several limitations remain. First, the reasoning mode introduces additional inference latency due to autoregressive reasoning generation, and its benefit is concentrated in ambiguous multi-drag edits. Although direct editing remains available and already performs strongly, future work could study distillation or adaptive reasoning strategies that invoke reasoning only when the edit is ambiguous. Second, our reasoning is expressed in language space, which provides interpretability but may not fully capture fine-grained visual transformations. A promising direction is to explore reasoning directly in latent or pixel space, where intermediate representations could encode spatial transformations more explicitly. Finally, although DragBench++ expands evaluation toward more semantic and ambiguous edits, drag-based editing remains under-constrained, and automatic metrics only partially reflect human preference. Future benchmarks could develop more robust evaluation protocols for settings where multiple valid edits can satisfy the same instruction.