RigIt: Precise Interactive Articulation of Objects in Images

AYAN KASHYAP, International Institute of Information Technology, Hyderabad, India
AISHWARYA AGARWAL, International Institute of Information Technology, Hyderabad, India and Adobe Research, India
DEEPTI GHADIYARAM, Boston University, USA
VINEET GANDHI, International Institute of Information Technology, Hyderabad, India



Fig. 1. **Overview of articulation-based image editing with RigIt.** Given a source image and a set of user-selected parts to articulate, our method produces a photorealistic edited image with the parts in a target articulation state while preserving identity, appearance, and scene context. **Top:** The user rigs movable parts of an input photograph (a) as 3D kinematic proxies (b), articulates them to the target state (c), and our diffusion model synthesizes the resulting image (d). Here the microwave door, oven door, and drawers are opened simultaneously, with shadows, lighting, and revealed interiors handled automatically. **Bottom:** RigIt generalizes to real photographs and to object categories absent from training. Each example shows the wireframe articulation overlaid on the input (left) and the resulting output (right): a trash bin, a wooden cabinet with multi-part motion, and a car door.

Editing articulated objects within a single image requires precise kinematic control—specifying exactly what moves, along which axis, and by how much. Existing text- and drag-based image editing methods provide only weak or ambiguous control over articulated object structure, often failing to localize the correct movable part or execute precise articulation extents, particularly under occlusion. We propose RigIt, a novel framework that enables precise object articulation by directly "rigging" parts within a 2D image. Our method extracts the underlying geometry of a user-selected part to fit a 3D kinematic proxy, represented as an oriented bounding box (OBB). Users interactively manipulate this explicit 3D handle, which the system projects into a 2D wireframe conditioning signal. A fine-tuned diffusion model then processes this signal, guided by a localized articulation-aware loss, to generate photorealistic edits. Remarkably, despite being trained exclusively on a small synthetic dataset of six categories, RigIt generalizes robustly to real-world, in-the-wild images and unseen object classes. Extensive evaluations on both synthetic and real-world benchmarks demonstrate that our approach significantly outperforms state-of-the-art text and drag-based baselines, achieving over 90% user preference for preserving articulation intent and scene consistency. All code, trained models and dataset will be publicly released at https://rigit-project.github.io.

CCS Concepts: • **Computing methodologies** → **Image representations**; *Image manipulation.*

Additional Key Words and Phrases: diffusion-based image editing, interactive manipulation, articulation

ACM Reference Format:
Ayan Kashyap, Aishwarya Agarwal, Deepti Ghadiyaram, and Vineet Gandhi. 2026. RigIt: Precise Interactive Articulation of Objects in Images. In *SIGGRAPH Asia 2026 Conference Papers (SA Conference Papers '26), December 01–04, 2026, Kuala Lumpur, Malaysia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3829340.3842367

Authors' Contact Information: Ayan Kashyap, International Institute of Information Technology, Hyderabad, Hyderabad, India, ayan.k@research.iiit.ac.in; Aishwarya Agarwal, International Institute of Information Technology, Hyderabad, Hyderabad, India and Adobe Research, Hyderabad, India, agarwal.aishwarya2013@gmail.com; Deepti Ghadiyaram, Boston University, Boston, USA, dghadiya@bu.edu; Vineet Gandhi, International Institute of Information Technology, Hyderabad, Hyderabad, Telangana, India, vgandhi@iiit.ac.in.

This work is licensed under a Creative Commons Attribution 4.0 International License.
SA Conference Papers '26, Kuala Lumpur, Malaysia
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2842-6/26/12
https://doi.org/10.1145/3829340.3842367

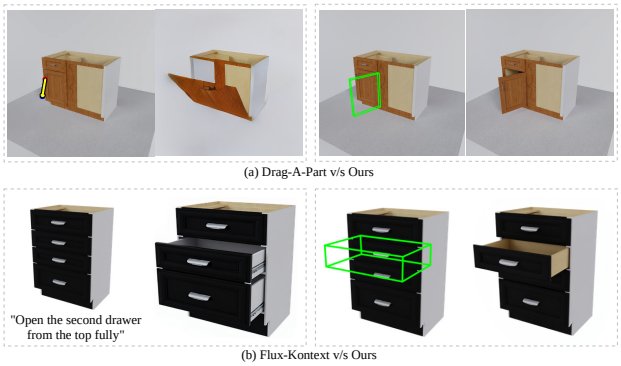


Fig. 2. Comparison of text-, drag-, and wireframe-based articulation control. Text- and drag-based methods provide limited or implicit control over articulated part identity and motion, often producing ambiguous or incorrect articulations under occlusion. In contrast, our wireframe-based interface explicitly specifies both the moving part and its articulation geometry, enabling precise, controllable, and occlusion-aware articulation.

1 Introduction

Composing images of articulated objects in specific configurations is a creative task that current tools handle poorly. Modern text prompt-based editors [Google DeepMind 2026; Labs et al. 2025; Wu et al. 2025] can handle simple cases such as "open the door," but falter where control matters: disambiguating among similar parts and specifying precise motion extent. Drag-based methods [Li et al. 2025; Pan et al. 2023] supply more spatial signal, but a sparse 2D vector still under-specifies the kinematics that govern part motion: joint type, axis, and range (fig. 2 and table 1). Object-centric editing methods [Alzayer et al. 2025; Cheng et al. 2025] provide stronger control over identity, appearance, and placement, but operate at the object level rather than exposing editable part structure. The task is also intrinsically a 3D reasoning problem: the moving part may be partially or fully occluded in the source image, and respecting joint constraints requires geometric structure that 2D pixels do not directly provide. A useful articulation interface must therefore be explicit about *what* moves, *how*, and *how far*.

We propose *RigIt*, a framework for precise object articulation in images. Our key idea is to *rig* objects directly in the image: each user-selected part is fit with a 3D *kinematic proxy* – a visually explicit handle that constrains how the part can move – much like a rig attached to an animated mesh. This rigging step cleanly separates system inference from user interaction: the system handles the hard 3D estimation, and the user manipulates the proxies as natural 3D handles, without having to specify or mentally project how their edit will look in 2D.

Users can articulate a proxy to a target state, such as opening a door at a chosen angle or sliding a drawer to a specific extent. A diffusion-based editing model then realizes the resulting appearance changes, resolving shadows, materials, and disoccluded background and interior surfaces to ensure photorealism (fig. 1).

Constructing such proxies from a single image is non-trivial: it requires recovering part geometry, joint structure, and motion constraints. We choose oriented bounding boxes (OBBs) as the proxy representation. They supply spatial extent and a coordinate frame, just enough to specify articulation for the parts that dominate everyday articulated objects. However, estimating oriented bounding boxes from a 2D image is non-trivial. To tackle this, we first segment [Carion et al. 2025] the user-selected part, lift the image to a metric point map [Wang et al. 2025], and fit a tight bounding box to the resulting 3D point set. A joint type and axis are assigned to each proxy from a small discrete label space, either through an interactive editor or by a vision-language model. Articulating the proxy and projecting its source and target configurations into the image plane produces the *wireframe condition*, a 2D control signal compatible with image-editing diffusion models.

To translate this control signal into a photorealistic edit, we fine-tune FLUX [Labs 2024], adapting it with condition-specific LoRA modules [Hu et al. 2022] on the source-image and wireframe conditions and training with an articulation-aware loss that focuses supervision on regions where parts move or background disoccludes. Training uses a synthetic dataset of six everyday articulated-object categories rendered in Blender under varied cameras and HDRI lighting. The dataset is designed to simulate exactly the kinds of edits the interface produces.

RigIt generalizes well beyond its training distribution, producing realistic and precise articulation edits on real-world scenes and unseen object categories such as trash bins, wardrobes, and car doors (fig. 1). We evaluate RigIt on three datasets: a synthetically generated benchmark, a curated real-world Common Articulated Objects (CAO) benchmark, and a collection of in-the-wild internet images. Across all studied settings, RigIt consistently outperforms state-of-the-art text- and drag-based baselines on standard reconstruction metrics and the user studies confirm that our edits are preferred in over 90% of A/B trials. Our contributions are as follows:

- RigIt, a framework that lets users articulate objects in a single image by rigging part-level 3D bounding-box proxies that are recoverable directly from the image.
- A **wireframe conditioning scheme** that turns articulated 3D proxies into a 2D control signal for diffusion-based editing, paired with an articulation-aware loss that keeps edits localized to moving and disoccluded regions.
- A study of **generalization**: training on a small synthetic dataset of six categories suffices for the model to handle real images, unseen categories, multi-part scenes, and challenging occlusion, validated by a user study and a new real-world benchmark.

2 Related Works

Controlled and reference-guided generation. A large body of work improves controllability in diffusion models by injecting spatial or appearance conditions into pretrained generative backbones. Early UNet-based methods such as ControlNet [Zhang et al. 2023], T2I-Adapter [Mou et al. 2024], IP-Adapter [Ye et al. 2023], Uni-ControlNet [Zhao et al. 2023], and OmniControlNet [Wang et al. 2024b] enable unified or multi-condition control through trainable

Table 1. **Information content of user interfaces for articulated control.** Unlike text prompt or drag-based methods, proposed wireframe explicitly encodes part identification (Part ID), articulation type (Joint), motion direction (Axis), limits (Extent), and occlusion (Occl.) to achieve high precision (Prec.). Symbols: ✓ = explicit, × = absent, ∼ = implicit or ambiguous.

Interface	Part ID	Joint	Axis	Extent	Occl.	Prec.
Text	×	×	×	×	×	Low
Drag	∼	×	×	×	×	Low
Wireframe (Ours)	✓	✓	✓	✓	✓	High

branches, lightweight adapters, or decoupled attention mechanisms. More recent DiT-based approaches, including OminiControl [Tan et al. 2025] and EasyControl [Zhang et al. 2025], emphasize parameter reuse and efficient condition injection. Orthogonal layout-grounded methods such as GLIGEN [Li et al. 2023], MultiDiffusion [Bar-Tal et al. 2023], and InstanceDiffusion [Wang et al. 2024a] provide explicit regional or instance-level control through boxes, masks, points, and scribbles. However, these methods control scene layout, object placement, or overall appearance rather than explicit articulation of parts considering joints and motion extents.

Image editing and geometric manipulation. Diffusion-based editing methods enable semantic and appearance manipulation via instructions [Brooks et al. 2023], attention control [Cao et al. 2023; Hertz et al. 2022], inversion and image-specific optimization [Kawar et al. 2023; Mokady et al. 2023], feature injection [Tumanyan et al. 2023], stochastic editing [Meng et al. 2021], masked editing [Avrahami et al. 2022], and exemplar-guided manipulation [Yang et al. 2023]. Recent work introduces stronger geometric control through lifting diffusion activations to 3D for object-centric transformations [Pandey et al. 2024], training-free object relocation [Avrahami et al. 2024], explicit object-orientation control [Parihar et al. 2025], 3D-aware layout conditioning and editing [Bhat et al. 2024; Eldesokey and Wonka 2024], occlusion-aware synthesis and rendering [Agrawal et al. 2026; Zhan and Liu 2025], and mesh-guided drag editing [Koo et al. 2025]. However, these methods primarily target whole-object transformation, layout, viewpoint, or occlusion reasoning rather than articulated editing, which requires controlled structural changes, kinematically valid part motion, preservation of unaffected regions, and coherent synthesis of disoccluded content.

Drag-Based and Articulated Editing. Interactive drag editors offer intuitive spatial control but are underspecified for articulated objects. DragDiffusion [Shi et al. 2024] optimizes intermediate latents for point-based control, while DragonDiffusion [Mou et al. 2023] preserves content via classifier-guidance gradients and cross-branch attention. FreeDrag [Ling et al. 2024] enhances reliability through adaptive feature updates, InstantDrag [Shin et al. 2024] amortizes editing with flow-conditioned models, and FlowDrag [Koo et al. 2025] leverages 3D mesh cues for geometric consistency. Despite these advances, sparse point trajectories remain an underspecified representation for articulated objects as they cannot explicitly encode part identity, joint types, or motion axes. Moreover, these methods target generic object-level manipulation, such as translation, rotation and rescaling, rather than part-level articulation, making them not directly comparable in our setting. The closest prior works, DragAPart [Li et al. 2024] and Puppet-Master [Li et al. 2025], address this by learning part-level motion priors from synthetic data for image and video generation, respectively. However, their control mechanisms remain indirect. In contrast, our approach conditions on projected 3D wireframes, providing explicit, user-editable control over part identity, motion axes, and articulation extent, while ensuring the synthesis of newly revealed regions.

3D Part Articulation. A complementary line of work models articulation directly in 3D. GAMMA [Yu et al. 2024] learns cross-category articulation and grasp affordances for robotic manipulation, while ATOP [Vora et al. 2025] predicts text-specified part motion for 3D objects. PARTICULATE [Li et al. 2026a] recovers parts, kinematic structure, and motion constraints from a 3D mesh, while its instruction-controlled extension, Instruct-PARTICULATE [Li et al. 2026b] additionally conditions on a target kinematic specification. DreamArt [Lu et al. 2025] instead generates interactable articulated 3D assets from a single image. Unlike these methods, which operate on or produce 3D assets and kinematic models, RigIt performs articulation *edits of the input image itself*, preserving appearance while synthesizing newly disoccluded content.

3 Method

Our goal is controllable image editing for articulated objects. Given a single image, users specify object parts and their desired articulation states, and the model synthesizes realistic edits while preserving object identity, appearance, and scene context. Our approach is based on explicit part-level 3D control: from a source image I_{src}, we first construct an articulated wireframe representation I_{wire} that encodes the target articulation. We then fine-tune a diffusion-based editing model conditioned on both I_{src} and I_{wire} to generate the articulated target image I_{tgt}. We describe each component of the framework in turn.

3.1 Rigging articulated objects from a single image

We now describe how we construct the wireframe condition I_{wire} from a source image I_{src} (fig. 3, left). Producing I_{wire} requires (i) identifying the parts to articulate, (ii) recovering a 3D kinematic proxy for each part, and (iii) specifying how each proxy moves. Each step can be performed interactively by a user through an editor (shown in Section B.2 of the supplementary) or executed automatically via a vision–language model. The interactive and automatic modes produce identical proxy representations, so we describe them jointly.

Step 1: Segmenting parts. The user specifies parts to articulate via text prompts (e.g., "door", "drawer") or through image point queries. Next, we use SAM3 [Carion et al. 2025] to segment the selected regions to produce a set of source masks $\{M^i_{\text{src}}\}_{i=1}^{N} = \{M_i\}$. In practice, SAM3 yields reliable masks even under moderate occlusion.

Step 2: Lifting to 3D.. We lift I_{src} to 3D using MoGE2 [Wang et al. 2025], producing a metric point map $\mathbf{P} \in \mathbb{R}^{3\times H\times W}$ together with camera intrinsics $\mathbf{K}$ that we use later to project proxies back to the image plane. For each mask M_i, the corresponding 3D points form the point set $\mathcal{X}_i = \{\mathbf{P}(u,v) \mid (u,v) \in M_i\}$.

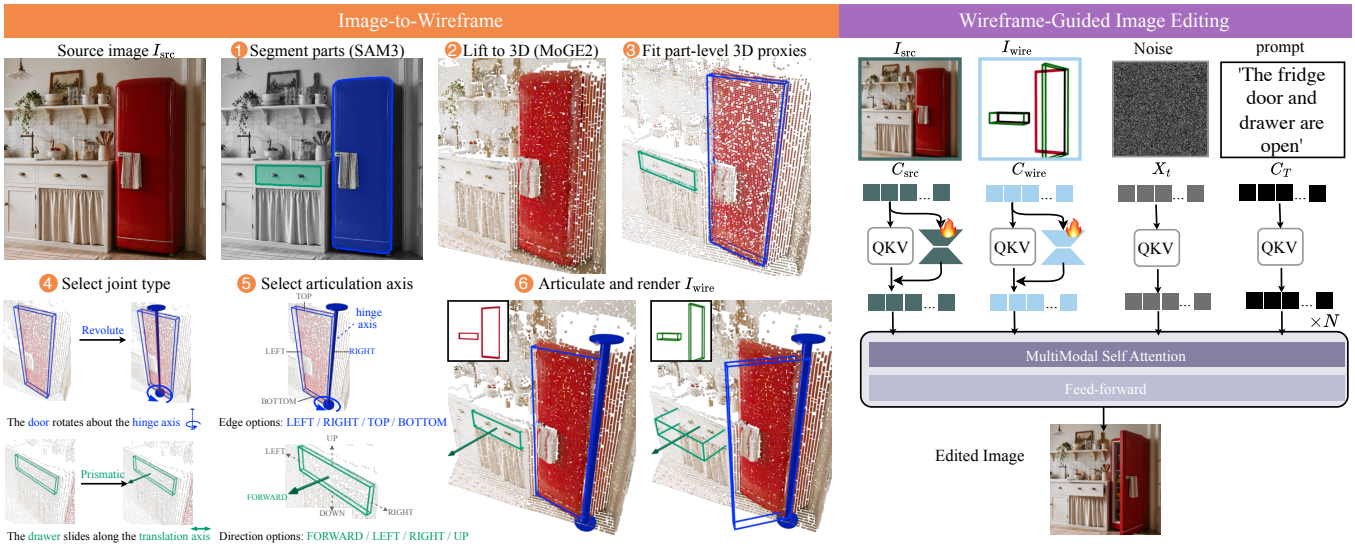


Fig. 3. **Left (Image-to-Wireframe).** Given a source image I_{src}, the user selects which parts to articulate; Step 1 uses SAM3 [Carion et al. 2025] to segment the selected regions and Step 2 uses MoGE2 [Wang et al. 2025] to lift the image to a metric point map. In Step 3, we fit an oriented 3D bounding box to each part as a kinematic proxy. A joint is attached to each proxy by selecting a type (Step 4; *revolute* or *prismatic*) and an axis (Step 5) from small discrete label spaces, either through a lightweight editor or via a vision–language model. In Step 6, the user articulates the proxies to the target state, and we project the source and target proxy edges into the image plane to render the wireframe condition I_{wire} (insets). **Right (Wireframe-Guided Image Editing).** The diffusion model receives I_{src}, I_{wire}, noise, and a text prompt as input. The source image and wireframe are encoded as condition tokens, adapted by condition-specific LoRA modules (orange flames), then concatenated with the noisy latent and text tokens and jointly processed by the FLUX mmDiT blocks to synthesize the edited image.

Step 3: Fitting part-level 3D proxies. Each $\mathcal{X}_i$ is fit with an oriented bounding box (OBB) $\mathcal{B}_i = (\mathbf{c}_i, \mathbf{R}_i, \mathbf{s}_i)$ that serves as the kinematic proxy for part i, where $\mathbf{c}_i$, $\mathbf{R}_i$, and $\mathbf{s}_i$ denote its 3D center, orientation, and side lengths, respectively. We use OBBs rather than full part meshes because (i) they are reliably recoverable from a single image, (ii) their edges project to clean wireframes that the diffusion model can interpret as spatial cues, and (iii) the parts of interest in our setting (doors, drawers, lids) are well-approximated by a planar slab. Each proxy is fit in three stages: RANSAC plane estimation on $\mathcal{X}_i$, minimum-area rectangle fitting in the recovered plane, and lifting the rectangle back to 3D as a thin slab inflated along its normal by a small factor ϵ_{wire}. Concretely, the slab thickness is $\epsilon_{\text{wire}} \cdot \max(\text{width}, \text{height})$ with $\epsilon_{\text{wire}} = 0.1$ fixed across all categories, so it scales with part size and does not encode physical depth. The inflation does not represent the part's true physical thickness; it merely ensures the otherwise-degenerate planar box renders as a visible wireframe. Full derivations are in Section A of the supplementary.

Step 4: Selecting the joint type. The fitted OBB captures spatial extent but not motion, so we attach a joint specification to each proxy. We support the two articulation types that dominate our setting: *revolute* joints, where the part rotates about a hinge, and *prismatic* joints, where the part translates along a fixed direction.

Step 5: Selecting the articulation axis. Each proxy is oriented facing the camera to establish a canonical frontal view. In this frame, the axis is parameterized over a small discrete label space to keep specification simple for both users and VLMs. For revolute joints, one of four boundary edges (`LEFT`, `RIGHT`, `TOP`, `BOTTOM`) defines the hinge axis, with the edge midpoint as the pivot. For prismatic joints, one of four translation directions (`FORWARD`, `LEFT`, `RIGHT`, `UP`) defines the slide direction. These labels cover the dominant motions in our setting; the editor additionally supports manual axis specification for uncommon cases. For automated prediction, a VLM processes I_{src} overlaid with flat-shaded part masks to sequentially predict the joint type (Step 4) and its corresponding axis (Step 5). Exact prompts are in Section B.1 of the supplementary.

Step 6: Articulating proxies and rendering I_{wire}. Once each proxy is rigged, it becomes a constrained 3D handle. The user drags a proxy to specify the target state; motion is restricted to its joint degree of freedom, yielding rotation about the selected hinge or translation along the selected direction. Because proxies are overlaid on the lifted point cloud in a shared 3D frame, the articulated motion is visualized directly against the scene. Once the target state is fixed, we project the source and target proxy edges to the image plane using $\mathbf{K}$ and pack the source wireframe into the red channel and the target wireframe into the green channel of I_{wire} image.

3.2 Articulating Images with a Diffusion Model

Given the source image I_{src} and the wireframe condition I_{wire}, we synthesize an edited image I_{tgt} matching the target articulation while preserving object identity, appearance, and surrounding scene. We build on FLUX.1-dev [Labs 2024], a rectified-flow transformer that jointly processes image and text tokens via multimodal DiT

blocks [Esser et al. 2024], and adapt it for image-conditioned articulation editing. Figure 3 (right) summarizes the architecture.

Conditioning via additional token sequences. Following recent DiT-based image-conditioning methods [Tan et al. 2025; Zhang et al. 2025], we inject spatial controls as additional token sequences rather than as cross-attention adapters or ControlNet-style branches. The source and wireframe images are encoded into condition tokens, $C_{\text{src}} = E(I_{\text{src}})$ and $C_{\text{wire}} = E(I_{\text{wire}})$, where $E(\cdot)$ denotes the FLUX VAE encoder and tokenization step. At denoising time t, these condition tokens are concatenated with the noisy latent tokens X_t and the text tokens C_T as $Z_t = [C_{\text{src}}, C_{\text{wire}}, X_t, C_T]$ and processed by the frozen FLUX mmDiT blocks. To preserve spatial alignment, C_{src} and C_{wire} share the positional encodings of the corresponding noisy latent tokens X_t.

Condition-specific LoRA adapters. The two condition streams play complementary roles: C_{src} supplies appearance and scene context, while C_{wire} encodes the spatial articulation cue. To let the model specialize for each signal without overwriting the pretrained prior, we route the two condition branches through separate LoRA-adapted projections within each mmDiT block (fig. 3, right), while the noisy latent and text tokens continue to use the frozen base weights. Only the LoRA parameters are optimized; the FLUX backbone remains frozen throughout training. This keeps the pretrained generative prior largely intact, reduces the trainable parameter count, and we find that it improves generalization to real images at inference. To ensure that each condition branch provides an independent signal, we mask the attention so that C_{src} and C_{wire} attend only to the text tokens and the noisy latent X_t and not to each other. The latent and text tokens attend to all branches.

3.3 Synthetic dataset of objects in articulated states

The model in Sec. 3.2 requires triplets $(I_{\text{src}}, I_{\text{wire}}, I_{\text{gt}})$ in which I_{src} and I_{gt} show the same object at two articulation states under identical camera, lighting, and background, and I_{wire} encodes the target state as projected bounding-box wireframes (fig. 4, top). To the best of our knowledge, no real-world dataset provides this kind of paired supervision, and capturing it would require photographing each object across many articulation extents while holding all other factors fixed. We therefore turn to synthetic supervision. Prior work [Li et al. 2024, 2025] has shown that models trained on synthetic articulated data generalize well to real objects, and because we finetune only LoRA adapters on top of a strong pretrained prior (Sec. 3.2), we expect synthetic supervision to suffice in our setting as well.

We build the dataset from PartNet-Mobility [Xiang et al. 2020] assets rendered in Blender [Blender Foundation 2024]. It spans six categories—**Microwave**, **Laptop**, **Dishwasher**, **Refrigerator**, **WashingMachine**, and **StorageFurniture** (cabinets, drawers, shelving units)—and covers both revolute joints (door and lid hinges) and prismatic joints (drawers, sliding panels).

For each object instance, we illuminate the scene using Polyhaven HDRIs for realistic lighting; position cameras at three front-facing coordinates on a spherical shell; sample articulation at five uniform intervals (0%, 25%, 50%, 75%, and 100%) of each part's motion range; and render wireframes by projecting 3D part bounding boxes onto the image plane. In total, we use **91 unique object instances** (65 train / 26 test), yielding **2,360 training and 940 test rendered images** (3,300 total). Training pairs are formed by combining images that share the same object instance and camera view but differ in articulation state, yielding **9,440 training and 3,760 test pairs**. Further details are in Section C of the supplementary.

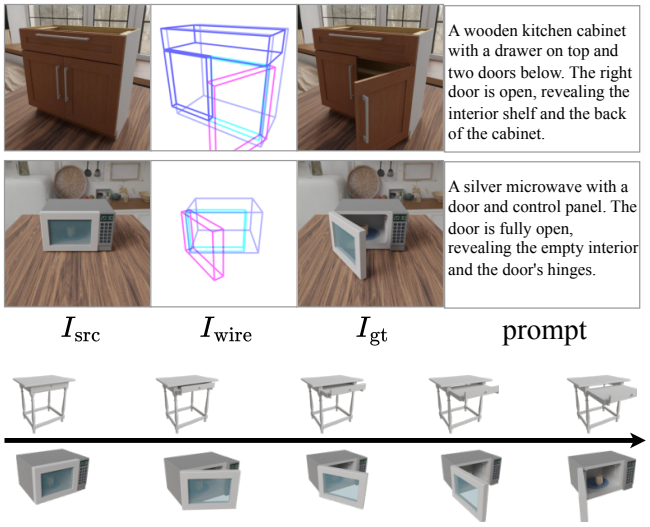


Fig. 4. Top: examples from our synthetic dataset, where each training sample consists of a source image I_{src}, articulation wireframe I_{wire}, target image I_{tgt}, and a part-aware text prompt. Bottom: uniformly sampled articulation states used to construct training pairs for both revolute and prismatic motions.

Descriptive part-aware prompting. Prompting plays a critical role in training convergence. Generic captions (e.g., "a microwave") yield slow convergence and weak condition adherence. We therefore use Qwen3-VL-8B [Bai et al. 2025] to generate detailed, appearance-grounded captions for each rendered image. For each image in an articulation sequence, the VLM produces a caption per image that explicitly localizes the moving part, describes its visual articulation degree, and details visible materials or newly exposed interiors, while strictly adhering to directly observable elements to prevent hallucination. Example captions/prompts with corresponding image triplets are shown in Figure 4.

This part-specific prompting lets the model learn the correspondence between textual descriptions, bounding-box transformations, and visual articulation changes. Such detailed prompting is needed only at training time; at inference, simple prompts such as "*the door is open*" suffice. More details are in Section C of the supplementary.

3.4 Training

With the $(I_{\text{src}}, I_{\text{wire}}, I_{\text{gt}})$ triplets and text-prompt in hand, we train the LoRA adapters with a flow-matching objective augmented by a region-weighted term emphasizing the articulated parts.

Flow-matching objective. Following standard flow matching under the velocity parameterization [Lipman et al. 2022], we sample a timestep $t \sim \text{sigmoid}(\mathcal{N}(0,1))$, form the linear interpolant $\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{x}_1$ between the clean target latent $\mathbf{x}_0$ and Gaussian noise

$\mathbf{x}_1 \sim \mathcal{N}(0, \mathbf{I})$, and supervise the model $\mathbf{v}_\theta$ against the target velocity $\mathbf{v}^* = \mathbf{x}_1 - \mathbf{x}_0$ under conditioning $C = (I_{\text{src}}, I_{\text{wire}}, \text{text})$:

$$\mathcal{L}_{\text{base}} = \left\| \mathbf{v}_\theta(\mathbf{x}_t, t, C) - \mathbf{v}^* \right\|^2. \tag{1}$$

Region-weighted mask loss. Articulated parts (e.g., a drawer handle, a cabinet hinge) typically occupy a small fraction of the image, so $\mathcal{L}_{\text{base}}$ alone is dominated by background and static-object tokens. We therefore add a region-weighted term restricted to a binary token mask $\mathbf{M}$ formed by the union of the projected source and target part bounding boxes. The union covers both the region the part vacates and the region it now occupies, focusing supervision on the pixels that change either through motion of the part itself or through disocclusion of background behind it. The masked loss is

$$\mathcal{L}_{\text{mask}} = \frac{|\mathbf{X}|}{|\mathbf{M}|} \sum_{i \in \mathbf{M}} \left\| \mathbf{v}_\theta^{(i)} - \mathbf{v}^{*(i)} \right\|^2, \tag{2}$$

where $|\mathbf{X}|$ is the total number of latent tokens and $|\mathbf{M}|$ the number of tokens inside the mask. The factor $|\mathbf{X}|/|\mathbf{M}|$ rescales the masked region so that it contributes the same total gradient magnitude as the full image would under $\mathcal{L}_{\text{base}}$, preventing small articulated parts from being drowned out by the static majority. The full objective is

$$\mathcal{L} = \mathcal{L}_{\text{base}} + \lambda \, \mathcal{L}_{\text{mask}}, \qquad \lambda = 1.0. \tag{3}$$

LoRA configuration. Each of the two condition-specific LoRA adapters has rank $r = 128$ and scaling factor $\alpha = 128$, and is applied to all linear layers within the mmDiT blocks: input embedding projections, attention $q/k/v/o$ projections, feed-forward layers, and modulation and normalization projections. Only the LoRA parameters are optimized; the FLUX backbone remains frozen throughout.

Training and inference. We optimize with Prodigy [Mishchenko and Defazio 2023] at a base learning rate of 1.0, which acts as a unit scale since Prodigy adaptively estimates the effective step size during training. We train for 100,000 iterations with batch size 8 on a single NVIDIA RTX A6000, using a masked-loss weight of $\lambda = 1.0$ throughout. During inference, we use 28 denoising steps and a classifier-free guidance scale of 5.

4 Experiments

We perform thorough quantitative and qualitative experiments. Quantitatively, we compare RigIt against state-of-the-art editing baselines on (i) a held-out test split of our synthetic dataset, generated with the same pipeline as the training set (section 3.3), and (ii) a new real-world evaluation set which we manually collect for this work (section 4.2). Qualitatively, we evaluate on articulation edits of images from Amazon Berkeley Objects [Collins et al. 2022] and CC-licensed photographs sourced from the internet. For all evaluations, wireframe conditions (I_{wire}) are generated using the proposed framework (section 3.1), a video demonstration of the interface is provided in the supplementary.

4.1 Baselines

We compare against representative methods from two families of articulation-capable editing models.

Text-based: NanoBanana 2 [Google DeepMind 2026], a state-of-the-art text-driven image editor, serves as our primary text-prompt-based baseline, together with Qwen-Image-Edit [Wu et al. 2025] and FLUX Kontext [Labs et al. 2025]. To account for the richer geometric conditioning available to RigIt, we additionally evaluate a NanoBanana2+wireframe variant that receives the same wireframe overlaid on the source image.

Drag-based: Puppet-Master [Li et al. 2025] and Drag-A-Part [Li et al. 2024] support articulated manipulation from sparse 2D drag inputs. We observe that both struggle to preserve scene content in cluttered real images, so we evaluate them only on the single-object, clean-background variant of the synthetic dataset, and exclude them from real-world evaluations.

Adapting baselines to our inputs. RigIt takes in a reference image, a wireframe image, and a short text prompt. On the other hand, the evaluated baselines expect either a descriptive caption or 2D drags. We convert each ground-truth articulation into whichever modality the baseline requires. For text-based editors, we reuse the Qwen3-VL captions generated for training; for qualitative edits, which lack a ground-truth target, we manually write the captions. For drag-based editors, we derive sparse 2D drags from the ground-truth articulation by sampling pixels in each part's source mask, applying the joint-constrained 3D motion via the recovered point map, and reprojecting to obtain target pixels.

4.2 Common Articulated Objects benchmark

Real-world evaluation is essential, as synthetic data fails to capture the appearance complexity and geometric variability encountered in practical settings, and no paired benchmark currently exists in which only the articulated state changes while all other scene attributes remain fixed. To this end, we collect the Common Articulated Objects (CAO) benchmark containing **26 object instances** across **7 object categories** (cabinet, drawer, dishwasher, fridge, laptop, oven, and table), captured over **189 images**. From these, we construct **326 directed source–target edit pairs**, each annotated with a Qwen3-VL-8B-generated caption describing the articulation change. Examples are shown in Figure 7. The data were captured in residential and laboratory environments using an iPhone 14 on a fixed tripod while objects were articulated through discrete states. We obtain wireframe conditions using the automatic VLM mode of our interface (section 3.1), manually correcting a prediction only when it is wrong.

4.3 Quantitative Evaluation

Consistent with prior works [Li et al. 2025], we evaluate methods using standard metrics PSNR, SSIM, LPIPS, and CLIP-I to measure reconstruction fidelity, perceptual quality, and semantic consistency.

Results on synthetic images. Table 2 reports results on the synthetic dataset. RigIt achieves the best performance across all four metrics by a substantial margin. Among the standard text-based baselines, NanoBanana2 performs best. NanoBanana2+wireframe improves over standard NanoBanana2, but remains below RigIt. FLUX Kontext and Qwen-Image-Edit exhibit performance comparable to NanoBanana2.

Table 2. Quantitative comparison on our synthetic test set. For each metric we report mean across all examples. Higher is better for PSNR, SSIM, and CLIP, while lower is better for LPIPS.

Method	PSNR (↑)	SSIM (full) (↑)	LPIPS (↓)	CLIP-I (↑)
PuppetMaster	21.04	0.76	0.27	0.78
DragAPart	20.52	0.73	0.24	0.86
Flux Kontext	23.54	0.89	0.13	0.94
Qwen-Image-Edit	23.86	0.87	0.16	0.94
NanoBanana2	24.12	0.88	0.15	0.95
NanoBanana2 + wireframe	25.31	0.89	0.10	0.95
RigIt	**28.47**	**0.93**	**0.08**	**0.97**

Drag-A-Part and Puppet-Master, perform substantially worse across all metrics, reflecting the difficulty of executing accurate articulated transformations from sparse 2D motion cues, together with a tendency to corrupt regions outside the moving part.

Table 3 reports results on our curated CAO benchmark, which evaluates generalization from synthetic training renders to real photographs with complex illumination, reflective materials, cluttered backgrounds, and diverse viewpoints. Despite this substantial domain shift, our method achieves the best performance across all metrics. Relative to the strongest text-based baseline, NanoBanana2, RigIt improves PSNR from 19.87 to 22.63 and reduces LPIPS from 0.25 to 0.18. The Nano Banana 2+wireframe control improves the corresponding scores to 20.15 PSNR and 0.19 LPIPS, but remains below RigIt; the distinction is more pronounced in articulation accuracy (section 4.4). Overall, RigIt's gains indicate stronger structural fidelity, perceptual quality, and semantic consistency under challenging real-world conditions. The results further demonstrate that our wireframe representation provides robust part-level articulation control and generalizes effectively beyond the synthetic training distribution.

4.4 Articulation Accuracy

Image-level metrics capture overall fidelity but not articulation accuracy. On CAO we recover the OBB from each generated image with our rigging pipeline and compare against the ground-truth target, measuring the opening-angle difference for revolute joints and the displacement difference (via MoGE2 depth) for prismatic joints. RigIt reduces the mean angular error from 23.9° to 9.8° and the mean prismatic error from 0.144 to 0.117 relative to NanoBanana2+wireframe (Table 4), demonstrating that RigIt accurately preserves both the direction and extent of the target articulation.

4.5 Qualitative Results

We present qualitative comparisons against text-prompt-based baselines on in-the-wild internet images in Figure 5 and on the CAO benchmark in Figure 7. Across all scenarios, RigIt consistently produces realistic, geometrically coherent, and accurately articulated edits. In contrast, even the state-of-the-art NanoBanana 2 fails to deliver consistent results. While it benefits from a strong generative prior that preserves scene detail and produces plausible edits at canonical articulation states (fully open or fully closed), its failures consistently arise from two key limitations: (i) incorrect localization of the articulated component when multiple movable parts

Table 3. Quantitative comparison on the CAO benchmark. Higher is better for PSNR, SSIM, and CLIP-I, while lower is better for LPIPS.

Method	PSNR (↑)	SSIM (full) (↑)	LPIPS (↓)	CLIP-I (↑)
Qwen-Image-Edit	12.61	0.36	0.54	0.81
Flux Kontext	17.62	0.71	0.21	0.93
NanoBanana2	19.87	0.72	0.25	0.93
NanoBanana2 + wireframe	20.15	0.75	0.19	0.94
RigIt	**22.63**	**0.77**	**0.18**	**0.94**

Table 4. Articulation error on the CAO benchmark. We fit an OBB to each generated image and compare it with the ground-truth target: opening-angle difference for revolute joints and absolute displacement difference for prismatic joints.

Method	Angular err. (°) (↓)	Prismatic err. (↓)
Qwen-Image-Edit	30.7	0.200
NanoBanana2	28.8	0.174
NanoBanana2 + wireframe	23.9	0.144
RigIt	**9.8**	**0.117**

are present (e.g. fig. 5, first and fourth rows), and (ii) inaccurate execution of intermediate articulation states with the correct extent (e.g. fig. 7, second and third rows). These failure modes are precisely what our wireframe conditioning is designed to address.

FLUX Kontext exhibits similar failure modes as Nano Banana 2, while further degrading scene fidelity (e.g. inaccurate door reconstruction in third row of fig. 5). We further compare against drag-based methods in Figure 6 on images from Amazon Berkeley Objects. Although drag-based approaches provide more explicit control, they remain fundamentally ambiguous in part association and articulation extent. Consequently, both Puppet-Master and Drag-A-Part exhibit poor localization and inaccurate articulations, highlighting the under-constrained nature of sparse drag supervision. Figure 8 presents additional results highlighting the versatility of RigIt for articulation editing across diverse object categories in real-world internet images.

4.6 User Study

We conducted a two-alternative forced-choice (2AFC) user study ($N = 21$) comparing our method against Nano Banana 2, FLUX Kontext, and Drag-A-Part. Given a source and an edit-intent image, participants selected the output that best satisfied the desired articulation while maintaining scene consistency. Of the 484 total pairwise comparisons, 308 were allocated to Nano Banana 2 (the most competitive baseline), and 88 each to FLUX Kontext and Drag-A-Part. As shown in fig. 9, participants preferred our method in 90.3%, 100%, and 95.5% of trials against Nano Banana 2, FLUX Kontext, and Drag-A-Part, respectively. The unanimous preference over FLUX Kontext demonstrates that general text-driven models lack the structured control required for fine-grained articulation. Similarly, the substantial margin over Drag-A-Part highlights that drag-based interfaces provide insufficient control over articulation extent and part semantics.

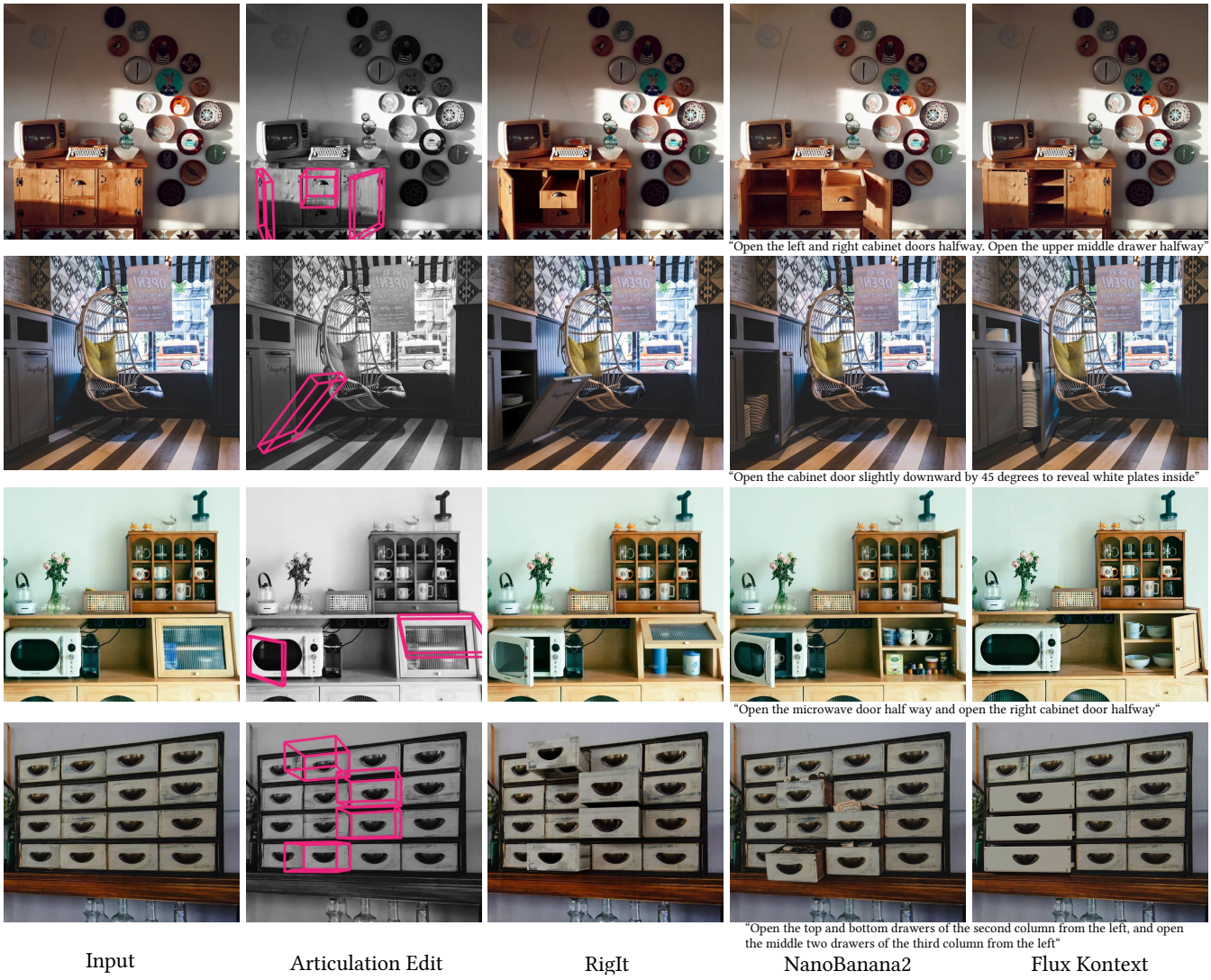


Fig. 5. Qualitative comparison against text-based baselines on in-the-wild internet images. For each row, from left to right, we show the original image, articulation wireframe, and results generated by RigIt, Nano Banana 2, and FLUX Kontext.

Table 5. Ablation study evaluating RigIt on the CAO benchmark, analyzing the impact of area-normalized masked loss weighting (λ) and LoRA adaptation capacity (r).

Variant	PSNR (↑)	SSIM (↑)	LPIPS (↓)	CLIP (↑)
RigIt ($r = 128$, $\lambda = 1$)	**22.63**	**0.77**	**0.18**	**0.94**
$r = 64$, $\lambda = 1$	18.30	0.70	0.20	0.91
$r = 64$, $\lambda = 0.5$	17.46	0.69	0.22	0.87
no mask loss ($r = 64$, $\lambda = 0$)	14.50	0.65	0.30	0.88

4.7 Ablation Studies

In Table 5 we ablate the effectiveness of the Region-weighted mask loss. Optimizing the flow-matching objective uniformly over the full image causes the loss to be dominated by the large static background, weakening supervision on the comparatively small articulated region. Consequently, removing the masked loss entirely leads to a substantial degradation in reconstruction quality (PSNR 14.50, LPIPS 0.30), with the model often ignoring the wireframe condition or articulating incorrect parts. Introducing masked weighting partially ($\lambda = 0.5$) recovers performance, while our full area-normalized formulation achieves the best results across all metrics. Notably, even with a reduced LoRA rank (r=64), the full formulation attains 18.30 PSNR and 0.20 LPIPS, substantially outperforming weaker loss variants, indicating that appropriate loss design is more critical than increased adaptation capacity for precise articulation control. Additional ablations on the text prompt, wireframe condition, and condition-specific LoRA adapters are provided in Section E of the



Fig. 6. Qualitative comparison against drag-based baselines on images from the Amazon Berkeley Objects dataset. From left to right, we show the input image, articulation wireframe and output produced by RigIt, input drag constraints, and outputs from Drag-A-Part and Puppet-Master.



Fig. 7. Qualitative comparison against Nano Banana 2 on the CAO benchmark. For each row, from left to right, we show the original image, ground-truth target, articulation wireframe, and results generated by RigIt and Nano Banana 2.

supplementary, further isolating the contribution of each conditioning component.

4.8 Robustness and Efficiency Analysis

Sensitivity to OBB perturbations. To evaluate sensitivity to inaccuracies in the fitted proxy, we perturb the 3D OBB from which the source and target wireframes are projected. This models upstream errors from depth, intrinsics, segmentation, and plane fitting, which all surface as OBB scale, position, or orientation error. On a balanced 40-pair subset of the synthetic test set (20 revolute, 20 prismatic), we independently vary scale (±5%, ±10%, ±20%), translation (±2.5%, ±5%, ±10% of the maximum OBB extent), and rotation (±3°, ±5°, ±10°), calibrated to comparable image-space error

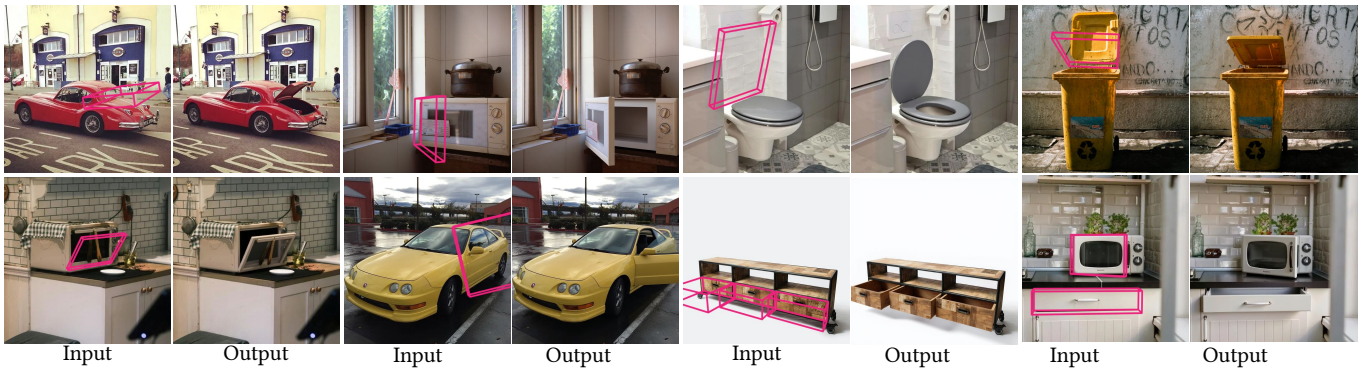


Fig. 8. Results of RigIt on diverse internet images. For each pair, the left image shows the input with the overlaid articulation wireframe, and the right image shows the edited output generated by RigIt.

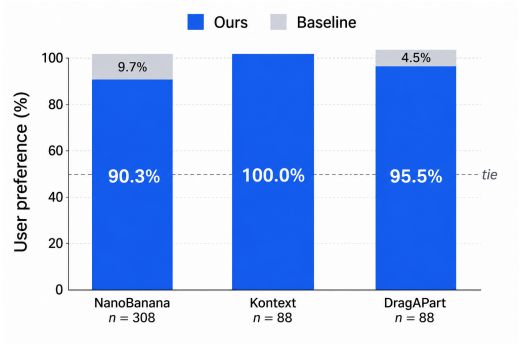


Fig. 9. Two-alternative forced-choice user-study results from 21 participants and 484 pairwise comparisons. Bars report the percentage of trials in which participants preferred RigIt over each baseline for satisfying the desired articulation while maintaining scene consistency.

(~4/8/16 px corner displacement at 512 × 512); both signs are averaged, with image, prompt, and seed fixed. Performance degrades smoothly: mild perturbations change PSNR by at most 0.08 dB and are significant in no metric, while even the strongest translation reduces PSNR by only 0.90 dB (28.12 → 27.22) and raises LPIPS from 0.079 to 0.085. CLIP-I shows no significant change anywhere. A combined moderate perturbation (±10%, ±5%, ±5°) costs only 0.56 dB. Thus RigIt tolerates mild and moderate proxy error without exact 3D reconstruction of the articulated part; approximate wireframe alignment suffices for stable editing at these magnitudes. Section F of the supplementary reports all levels with significance tests.

Runtime. On one RTX A6000, preprocessing (SAM3, MoGE2, OBB fitting) takes 1.3 s, automatic joint specification 8.5 s, and inference (28 steps) 31.1 s at 43.9 GB peak VRAM; optional interactive rigging adds ~30 s of user time. Latency is dominated by diffusion sampling, as in all diffusion-based editors.

Accuracy of the automatic mode. Finally, against manually verified annotations, the automatic mode used to rig CAO attains accuracies of 1.00 for joint type, 0.93 for hinge edge, and 1.00 for prismatic direction on CAO, and 0.94/0.82/0.99, respectively, on synthetic data. Hinge-edge errors stem from ambiguous cues such as occluded handles; any error is correctable in a few clicks (Section B.1 of the supplementary).

5 Limitations

We describe limitations of the rigging pipeline and of the generation method. RigIt requires a single dominant planar face the wireframe can approximately align to. This holds for mildly curved parts such as car trunks (fig. 8), but breaks down on strongly or doubly curved surfaces such as helmet visors and revolving doors. Spherical joints and deformable or multi-joint structures also fall outside the current scope, as such kinematic relations are not encoded in the rig. Generation quality also degrades when more than four heavily adjacent or overlapping parts are articulated simultaneously, leading to part-mixing artifacts. Qualitative examples of failure cases are presented in Section H of the supplementary.

Depth ambiguity. Projecting 3D articulations to a 2D wireframe is inherently many-to-one. Under a generic perspective view, inward and outward rotations foreshorten differently and yield distinct wireframes, and our articulation-error analysis (section 4.4) indicates that rotation direction is recovered correctly across our benchmark. Degeneracies nonetheless remain: under near-orthographic views or negligible slab thickness, motions differing in visibility and depth ordering project to nearly identical wireframes, leaving the condition underdetermined. Adding depth-ordering or visibility channels is a promising future direction.

6 Conclusions

In this work, we introduced RigIt, a framework for precise articulation editing in single images through image-space rigging. By recovering explicit 3D kinematic proxies and translating articulated states into wireframe-based diffusion conditioning, our approach enables controllable and photorealistic articulation edits, including simultaneous editing of multiple articulated parts, outperforming existing text- and drag-based methods across all evaluated settings. Remarkably, despite being trained only on synthetic single-object images from six categories, RigIt generalizes robustly to real-world scenes, unseen object categories, and challenging occlusions.

References

Vaibhav Agrawal, Rishubh Parihar, Pradhaan Bhat, Ravi Kiran Sarvadevabhatla, and R Venkatesh Babu. 2026. SeeThrough3D: Occlusion Aware 3D Control in Text-to-Image Generation. *arXiv preprint arXiv:2602.23359* (2026).

Hadi Alzayer, Zhihao Xia, Xuaner Zhang, Eli Shechtman, Jia-Bin Huang, and Michael Gharbi. 2025. Magic fixup: Streamlining photo editing by watching dynamic videos. *ACM Transactions on Graphics* 44, 5 (2025), 1–25.

Omri Avrahami, Rinon Gal, Gal Chechik, Ohad Fried, Dani Lischinski, Arash Vahdat, and Weili Nie. 2024. Diffuhaul: A training-free method for object dragging in images. In *SIGGRAPH Asia 2024 Conference Papers*. 1–12.

Omri Avrahami, Dani Lischinski, and Ohad Fried. 2022. Blended diffusion for text-driven editing of natural images. 18208–18218.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xiong-Hui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Rongyao Fang, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Li Ying Meng, Xuancheng Ren, Xin yi Ren, Sibo Song, Yu-Chen Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yihe Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Botao Zheng, Humen Zhong, Jingren Zhou, Fanxi Zhou, Jingren Zhou, Yuanzhi Zhu, and Keming Zhu. 2025. Qwen3-VL Technical Report. *ArXiv* abs/2511.21631 (2025). https://api.semanticscholar.org/CorpusID:283262018

Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. 2023. Multidiffusion: Fusing diffusion paths for controlled image generation. (2023).

Shariq Farooq Bhat, Niloy Mitra, and Peter Wonka. 2024. Loosecontrol: Lifting controlnet for generalized depth conditioning. In *ACM SIGGRAPH 2024 Conference Papers*. 1–11.

Blender Foundation. 2024. *Blender*. https://www.blender.org

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

Mingdeng Cao, Xintao Wang, Zhongang Qi, Ying Shan, Xiaohu Qie, and Yinqiang Zheng. 2023. Masactrl: Tuning-free mutual self-attention control for consistent image synthesis and editing. 22560–22570.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. 2025. Sam 3: Segment anything with concepts. *arXiv preprint arXiv:2511.16719* (2025).

Yen-Chi Cheng, Krishna Kumar Singh, Jae Shin Yoon, Alexander Schwing, Liang-Yan Gui, Matheus Gadelha, Paul Guerrero, and Nanxuan Zhao. 2025. 3d-fixup: Advancing photo editing with 3d priors. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*. 1–10.

Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F Yago Vicente, Thomas Dideriksen, Himanshu Arora, et al. 2022. Abo: Dataset and benchmarks for real-world 3d object understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 21126–21136.

Abdelrahman Eldesokey and Peter Wonka. 2024. Build-a-scene: Interactive 3d layout control for diffusion-based image generation. *arXiv preprint arXiv:2408.14819* (2024).

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. 2024. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*.

Google DeepMind. 2026. Nano Banana 2: AI Image Generator and Photo Editor. https://blog.google/innovation-and-ai/technology/ai/nano-banana-2/. Accessed: 2026-05-11.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. 2022. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626* (2022).

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. 2022. Lora: Low-rank adaptation of large language models. *Iclr* 1, 2 (2022), 3.

Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. 2023. Imagic: Text-based real image editing with diffusion models. 6007–6017.

Gwanhyeong Koo, Sunjae Yoon, Younghwan Lee, Ji Woo Hong, and Chang D Yoo. 2025. FlowDrag: 3D-aware drag-based image editing with mesh-guided deformation vector flow fields. *arXiv preprint arXiv:2507.08285* (2025).

Black Forest Labs. 2024. FLUX. https://github.com/black-forest-labs/flux.

Black Forest Labs, Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, et al. 2025. FLUX. 1 Kontext: Flow Matching for In-Context Image Generation and Editing in Latent Space. *arXiv preprint arXiv:2506.15742* (2025).

Ruining Li, Yuxin Yao, Chuanxia Zheng, Christian Rupprecht, Joan Lasenby, Shangzhe Wu, and Andrea Vedaldi. 2026a. Particulate: Feed-Forward 3D Object Articulation. In *CVPR*.

Ruining Li, Yuxin Yao, Matt Zhou, Chuanxia Zheng, Christian Rupprecht, Joan Lasenby, Shangzhe Wu, and Andrea Vedaldi. 2026b. Instruct-Particulate: Scaling Feed-Forward 3D Object Articulation with Kinematic Control. *arXiv preprint arXiv:2606.14699* (2026).

Ruining Li, Chuanxia Zheng, Christian Rupprecht, and Andrea Vedaldi. 2024. Dragapart: Learning a part-level motion prior for articulated objects. Springer, 165–183.

Ruining Li, Chuanxia Zheng, Christian Rupprecht, and Andrea Vedaldi. 2025. Puppetmaster: Scaling interactive video generation as a motion prior for part-level dynamics. 13405–13415.

Yuheng Li, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. 2023. Gligen: Open-set grounded text-to-image generation. 22511–22521.

Pengyang Ling, Lin Chen, Pan Zhang, Huaian Chen, Yi Jin, and Jinjin Zheng. 2024. Freedrag: Feature dragging for reliable point-based image editing. 6860–6870.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. 2022. Flow Matching for Generative Modeling. *ArXiv* abs/2210.02747 (2022). https://api.semanticscholar.org/CorpusID:252734897

Ruijie Lu, Yu Liu, Jiaxiang Tang, Junfeng Ni, Yuxiang Wang, Diwen Wan, Gang Zeng, Yixin Chen, and Siyuan Huang. 2025. Generating Objects with Part-Articulation from a Single Image. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*. 1–13.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. 2021. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073* (2021).

Konstantin Mishchenko and Aaron Defazio. 2023. Prodigy: An expeditiously adaptive parameter-free learner. *arXiv preprint arXiv:2306.06101* (2023).

Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. 2023. Null-text inversion for editing real images using guided diffusion models. 6038–6047.

Chong Mou, Xintao Wang, Jiechong Song, Ying Shan, and Jian Zhang. 2023. Dragondiffusion: Enabling drag-style manipulation on diffusion models. *arXiv preprint arXiv:2307.02421* (2023).

Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, and Ying Shan. 2024. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models, Vol. 38. 4296–4304.

Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimitra Meka, and Christian Theobalt. 2023. Drag your gan: Interactive point-based manipulation on the generative image manifold. In *ACM SIGGRAPH 2023 conference proceedings*. 1–11.

Karran Pandey, Paul Guerrero, Matheus Gadelha, Yannick Hold-Geoffroy, Karan Singh, and Niloy J Mitra. 2024. Diffusion handles enabling 3d edits for diffusion models by lifting activations to 3d. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7695–7704.

Rishubh Parihar, Vaibhav Agrawal, Sachidanand VS, and Venkatesh Babu Radhakrishnan. 2025. Compass control: Multi object orientation control for text-to-image generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2791–2801.

Yujun Shi, Chuhui Xue, Jun Hao Liew, Jiachun Pan, Hanshu Yan, Wenqing Zhang, Vincent YF Tan, and Song Bai. 2024. Dragdiffusion: Harnessing diffusion models for interactive point-based image editing. 8839–8849.

Joonghyuk Shin, Daehyeon Choi, and Jaesik Park. 2024. Instantdrag: Improving interactivity in drag-based image editing. In *SIGGRAPH Asia 2024 Conference Papers*. 1–10.

Zhenxiong Tan, Songhua Liu, Xingyi Yang, Qiaochu Xue, and Xinchao Wang. 2025. Ominicontrol: Minimal and universal control for diffusion transformer. 14940–14950.

Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. 2023. Plug-and-play diffusion features for text-driven image-to-image translation. 1921–1930.

Aditya Vora, Sauradip Nag, Kai Wang, and Hao Zhang. 2025. Articulate That Object Part (ATOP): 3D Part Articulation from Text and via Motion Personalization. *ACM Transactions on Graphics* (2025).

Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. 2025. Moge-2: Accurate monocular geometry with metric scale and sharp details. *arXiv preprint arXiv:2507.02546* (2025).

Xudong Wang, Trevor Darrell, Sai Saketh Rambhatla, Rohit Girdhar, and Ishan Misra. 2024a. Instancediffusion: Instance-level control for image generation. 6232–6242.

Yilin Wang, Haiyang Xu, Xiang Zhang, Zeyuan Chen, Zhizhou Sha, Zirui Wang, and Zhuowen Tu. 2024b. Omnicontrolnet: Dual-stage integration for conditional image generation. 7436–7448.

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

Fan Xiang, Yuzhe Qin, Kaichuang Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Tang, Ke Lyu, Siyuan Huang, Siyuan Jiang, et al. 2020. SAPIEN: A SimulAted

Part-Based Interactive ENvironment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 11097–11106.

Binxin Yang, Shuyang Gu, Bo Zhang, Ting Zhang, Xuejin Chen, Xiaoyan Sun, Dong Chen, and Fang Wen. 2023. Paint by example: Exemplar-based image editing with diffusion models. 18381–18391.

Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. 2023. Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arXiv:2308.06721* (2023).

Qiaojun Yu, Junbo Wang, Wenhai Liu, Ce Hao, Liu Liu, Lin Shao, Weiming Wang, and Cewu Lu. 2024. Gamma: Generalizable articulation modeling and manipulation for articulated objects. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 5419–5426.

Xiaohang Zhan and Dingming Liu. 2025. LaRender: Training-Free Occlusion Control in Image Generation via Latent Rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 19679–19688.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023. Adding conditional control to text-to-image diffusion models. 3836–3847.

Yuxuan Zhang, Yirui Yuan, Yiren Song, Haofan Wang, and Jiaming Liu. 2025. Easycontrol: Adding efficient and flexible control for diffusion transformer. 19513–19524.

Shihao Zhao, Dongdong Chen, Yen-Chun Chen, Jianmin Bao, Shaozhe Hao, Lu Yuan, and Kwan-Yee K Wong. 2023. Uni-controlnet: All-in-one control to text-to-image diffusion models. 36 (2023), 11127–11150.