LENSERASE: A FOUNDATION MODEL FOR ERASE ANYTHING

Anonymous authors
Paper under double-blind review

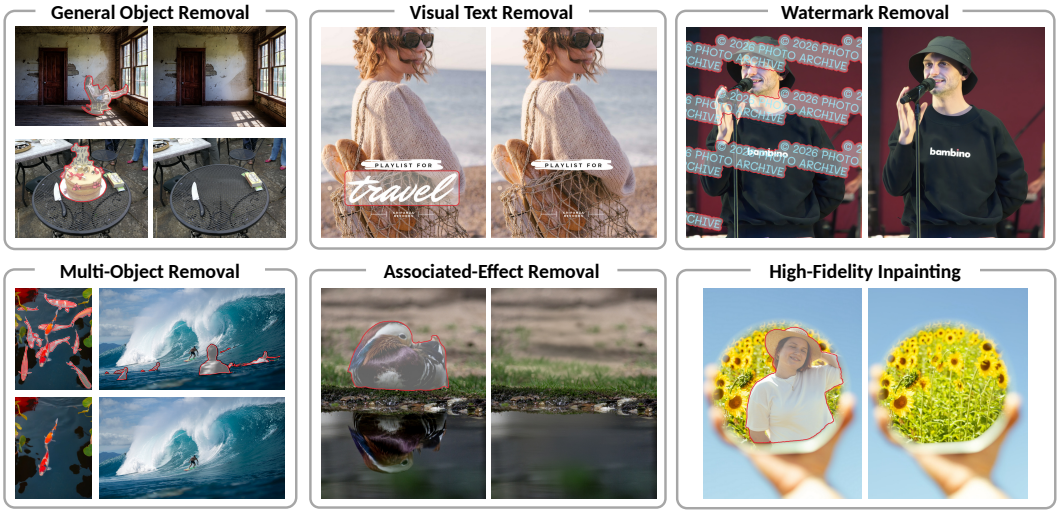

Figure 1: **LensErase across diverse removal scenarios.** Each pair shows a source image with highlighted removal targets and the corresponding LensErase result. Examples cover general objects, visual text, watermarks, multiple objects, coupled effects, and high-fidelity inpainting.

ABSTRACT

We introduce *LensErase*, a general-purpose mask-based *Erase Anything* foundation model designed to cover a broad spectrum of real-world erase scenarios. LensErase supports diverse objects, visual text, watermarks, associated visual effects, multiple-object removal, and a wide range of object scales, aspect ratios, and image resolutions. First, to enable such broad coverage, we construct *LensErase-100M*, a large-scale dataset containing 101.07 million paired source–mask–target examples spanning diverse erase scenarios. Second, we pretrain LensErase from the text-to-image model *Lens*, adapting its strong generative prior to mask-based erasing. To efficiently incorporate the source image and erase mask, we propose *Mask-Aware RoPE*, a 3D rotary positional encoding that directly injects mask information into the source-image tokens without introducing a separate mask token sequence. Finally, we refine the model through supervised fine-tuning and reinforcement learning with contract-guided, region-aware rewards, and apply *TraceFusion* at inference to preserve high-fidelity content outside the effective edit region. Across six public benchmarks and the proposed *LensErase-Bench*, LensErase achieves substantial improvements under both reference-based objective metrics and reference-free VLM evaluation. We hope LensErase can serve as a foundational model for erase-related applications and systems.

1 INTRODUCTION

Object erasing (Winter et al., 2024; Jiang et al., 2025; Zhao et al., 2026) is a fundamental image-editing capability with broad applications in photography, content creation, graphic design, and automated visual-production pipelines. Given an input image and a user-specified region, an ideal erase model should understand *what* the user intends to remove, eliminate all visual evidence as-

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.

sociated with the target, and plausibly reconstruct content that is geometrically, semantically, and visually consistent with the surrounding scene, while preserving the remainder of the image.

However, building a single erase model that works reliably across the diverse requests encountered in real-world applications remains challenging. Handling targets across a wide range of *spatial scales*, from tiny objects occupying less than 1% of an image to dominant foreground objects covering nearly half of the scene, requires both precise local editing and reconstruction of large missing regions. Erasing diverse *target types* poses distinct reconstruction challenges: text and logos require removing fine strokes without damaging nearby details, while translucent watermarks require recovering underlying content from blended pixels. *Multi-object removal* requires clearing every selected instance while maintaining scene consistency across spatially separated edits. Removing *associated visual effects*, such as reflections and shadows, requires recognizing and removing target-dependent changes that may extend beyond the selection mask (Winter et al., 2024; Zhao et al., 2026).

We introduce **LensErase**, a general-purpose *Erase Anything* model conditioned only on a source image and an erase mask. To enable broad coverage, we construct **LensErase-100M**, a large-scale dataset containing 101.07 million paired training examples, each consisting of a source image, an erase mask, and a target image. LensErase-100M is two to three orders of magnitude larger than representative object-removal datasets, which contain approximately 100K–1M training examples (Wei et al., 2025; Li et al., 2025; Jiang et al., 2025). Beyond scale, LensErase-100M is systematically constructed to cover diverse object categories, visual text, watermark types, object scales, image resolutions, and multi-object settings. We use domain-specific construction and validation pipelines to ensure reliable source–target supervision. General Data is constructed by inserting scene-compatible objects into clean real-world images and reversing the pairs, preserving the original images as removal targets. Specialized pipelines use alpha compositing for watermarks, layer rendering for posters, verified teacher edits for visual text, and foreground compositing for large objects.

LensErase builds upon a pretrained text-to-image model, *Lens* (Chen et al., 2026), leveraging its strong generative prior to reconstruct realistic and contextually coherent content in erased regions. Since both the condition and output are visual, we initialize both the *image branch* and *condition branch* from the pretrained *Lens* image branch. The condition branch receives the intact source image, allowing the model to identify the target and its associated effects, such as shadows and reflections. *Mask-Aware RoPE* injects region identity directly into the source-image tokens without introducing a separate mask token sequence. Region-weighted supervision further emphasizes the intended edit during pretraining, producing **LensErase-Base**. We then refine the model through supervised fine-tuning on curated high-quality, high-resolution data.

While large-scale supervised training equips the model with broad erase capabilities, removal quality can still vary across stochastic samples for the same input. Some outputs achieve complete and clean removal, while others may leave target fragments or residual effects, or alter neighboring content. We therefore further optimize the supervised model with reinforcement learning (RL) using DiffusionNFT (Zheng et al., 2026). Region-aware rewards encourage complete removal, plausible reconstruction, and preservation of non-target content. At inference, *TraceFusion* estimates the effective edit region from the selection mask and source–output residuals, and selectively restores the source image outside this region to improve fidelity. Together, RL and TraceFusion yield **LensErase**, combining reliable removal with high-fidelity preservation.

We also introduce **LensErase-Bench**, a benchmark of 500 removal cases spanning natural images, posters, watermarks, and visual text. Its reference-free visual evaluation assesses removal correctness, reconstruction quality, content preservation and visual cleanliness, complementing reference-based metrics when multiple plausible reconstructions exist. We evaluate LensErase against 12 existing methods on LensErase-Bench and six public benchmarks. LensErase achieves substantial gains under both reference-based objective metrics and reference-free VLM evaluation, demonstrating strong and consistent performance across diverse erase scenarios. We aim for LensErase to serve as a general-purpose erase model for diverse visual-generation and editing systems.

2 DATA

2.1 PRE-TRAINING DATA: LENSERASE-100M

We construct **LensErase-100M**, a large-scale pre-training dataset comprising 101.07M data triplets, where each triplet is represented as $(I_{\text{input}}, I_{\text{output}}, M)$, denoting the input image, erased target im-

age, and corresponding erase mask, respectively. LensErase-100M covers five major data categories: *General Data* (77.6M), *Watermark Data* (9.99M), *Large-Mask Data* (4.02M), *Visual Text Data* (1.44M), and *Poster Data* (8.02M). Figure 2 shows the distributions of mask area ratios and mask counts in LensErase-100M. Below, we briefly introduce the construction pipeline for each data category, with additional details and filtering criteria provided in Appendix B.

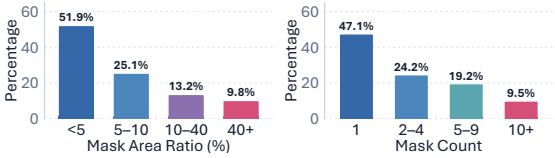

Figure 2: **Left**: Distribution of mask area ratios. **Right**: Distribution of mask counts.

General Data. We construct general object-removal pairs by adding an object to a real image and reversing the pair. We observe that general-purpose editing models perform addition more reliably than removal, which often leaves target remnants, changes unrelated backgrounds, or produces blurry fills and artifacts. Using the original real image as the clean target grounds supervision in the real-image distribution and encourages realistic outputs. The construction pipeline is as follows.

1. *Object Taxonomy.* We define a hierarchical taxonomy with 15 coarse categories (e.g., Animal, Furniture, and Infrastructure), each further divided into multiple fine-grained object classes.
2. *Instruction Generation.* We first build an image-caption pool $\mathcal{D} = (I, T)$. For each sample, we sample three distinct coarse categories. Given (I, T) and the three sampled categories, GPT-4.1 (OpenAI, 2025) is instructed to select an object that is absent from I, belongs to one of these categories, and can be naturally inserted into the scene. It outputs the corresponding coarse category, object class, and an insertion instruction P (e.g., "add a dog on the grass").
3. *Dynamic Balancing.* Sampling is dynamically adjusted according to accumulated category/object statistics. Over-represented categories or object classes are excluded from subsequent sampling to maintain data diversity and balance.
4. *Object Insertion.* Given the original image I and insertion instruction P, we use an off-the-shelf image editing model G_{add} to synthesize an image with the target object inserted. We use the inserted image as the input to the erase model, $I_{\text{input}} = G_{\text{add}}(I, P)$, and the original image as the desired erase output, $I_{\text{output}} = I$.
5. *Coarse-to-Fine Mask Generation.* We threshold low-frequency DCT and DINOv3 (Siméoni et al., 2025) feature differences between I_{input} and I_{output} to obtain complementary binary change maps. Their union defines a high-recall change region, while the interior of their overlap provides reliable positive-point candidates. Negative points are sampled outside the union. SAM2 (Ravi et al., 2024) uses these prompts to produce the final object-selection mask M.

Watermark Data. We construct watermark-removal data through the following pipeline.

1. *Watermark Collection.* We collect a diverse set of commonly seen watermark patterns in real-world images, including logo watermarks, text watermarks, repeated watermarks, diagonal cross-image watermarks, stripe-style watermarks, and small corner marks.
2. *Watermark Synthesis.* For each clean image, we randomly select a watermark asset or text. We then synthesize the watermark with randomized font, color, position, scale, rotation, opacity, stroke, and overall layout to cover diverse real-world watermark appearances.
3. *Triplet Construction.* Since the watermark is rendered programmatically, we do not estimate the mask from image differences. We treat the watermarked image as the erase-model input, the clean image as the erase target, and the rendered region as the mask, i.e., I_{input} is the synthesized watermarked image, I_{output} is the original clean image, and M is the watermark mask.

Large-Mask Data. We observe that large-mask samples are relatively under-represented in the General Data, since current general-purpose editing models struggle to insert very large objects naturally into a scene through prompting alone without occluding existing objects. We therefore construct an additional large-mask subset using a controllable copy-paste pipeline.

1. *Foreground Generation.* We define a foreground-object taxonomy containing 25 coarse categories and approximately 2.5K fine-grained classes. We randomly select a fine-grained class and generate a matching object on a white background using an image generation model. We then apply BiRefNet (Zheng et al., 2024) to extract the foreground object and its alpha mask.

2. *Copy-Paste.* We sample a clean background image I and apply YOLOE (Wang et al., 2025) to detect existing objects in the scene. The generated foreground is then scaled to occupy a relatively large image region and pasted preferentially into areas that do not overlap with detected objects.
3. *Triplet Construction.* We use the composite image after copy-paste as the erase-model input I_{input}, the original background image as the clean target I_{output}, and the transformed foreground alpha mask as the erase mask M.

Visual Text Data. We construct visual-text removal data through the following pipeline.

1. *Text Detection.* Given an image I containing visual text, we first apply PaddleOCR (Cui et al., 2025) to detect text regions and their bounding boxes.
2. *Text Editing.* We use an image editing model with a prompt that instructs it to erase text while preserving its carriers and other scene content, producing a teacher-edited image $\tilde{I}$.
3. *Candidate Removal Identification.* We run PaddleOCR again on $\tilde{I}$ and compare the OCR results before and after editing. Text regions detected in I but no longer matched in $\tilde{I}$ are treated as candidate removal regions. We further use SSIM to exclude candidates that exhibit insufficient structural change despite the OCR mismatch.
4. *Poisson Blending.* We randomly select a subset of eligible text groups and blend the corresponding regions from $\tilde{I}$ into the original image I, preserving unselected text and surrounding content. Poisson blending (Pérez et al., 2023) reduces color seams introduced by the editing model.
5. *Triplet Construction.* We use the original image with text as the erase-model input, i.e., $I_{\text{input}} = I$. The Poisson-blended image is used as the target I_{output}, and the union of the selected text regions, with a small dilation, forms the mask M. The resulting triplets are accepted as valid removal pairs after the quality filtering detailed in Appendix B.1.4.

Poster Data. We construct poster-element removal data by modeling a poster as a structured set of layers with explicit z-order.

1. *Canvas and Layout Sampling.* We first sample a canvas from either a natural image or a design background, and randomly choose a canvas aspect ratio and a layout template.
2. *Element Generation.* We populate each poster with approximately 4–9 elements. The element types include title, subtitle, paragraph, label, price, date, URL, basic shape, decorative strip, logo, QR code, and chart. For each element, we randomly sample its content and attributes, including font, language, color, opacity, and other style parameters.
3. *Structured Layer Composition.* All elements are organized as a structured layer set with explicit z-order, which determines their spatial arrangement and mutual occlusion. Under a fixed background, layout, and layer configuration, we render the complete poster.
4. *Element Deletion.* We randomly remove a subset of elements while keeping the same background, layout, and remaining layer states unchanged, and render the edited poster after deletion.
5. *Triplet Construction.* We use the complete poster as the erase-model input I_{input}, the poster after deleting the selected elements as the target image I_{output}, and the visible regions of the removed elements as the mask M. The mask is determined according to the z-order of the layered poster, so that only the actually visible pixels of the removed elements are included.

2.2 Supervised Fine-Tuning (SFT) and Reinforcement Learning (RL) Data

SFT Data. We follow the pretraining pipelines in Section 2.1, prioritizing high-resolution images. All candidate triplets undergo GPT-5.6 Sol (OpenAI, 2026a) screening for mask validity, edit correctness, non-target preservation, and reconstruction quality (Appendix B.2). We do not include Large-Mask Data in SFT, as these examples are synthetically generated and are mainly used to improve the coverage of large-mask cases during pre-training. For Poster Data, instead of using the synthetic generation pipeline adopted during pre-training, we collect high-quality poster examples designed by professional designers, providing more realistic layouts, typography, and visual compositions. The SFT set contains 140K examples: 80K General Data, 20K Watermark Data, 20K Visual Text Data, and 20K Poster Data.

RL Data. We curate 19K image–mask tasks using source images disjoint from SFT. General Data contains 8K difficulty-filtered Hard/Extreme cases and 5K targeted supplements. Manual analysis of LensErase-SFT failures guides collection of new images with similar challenges. For annotated General Data tasks, GPT-5.6 Sol (OpenAI, 2026a) localizes objects, including those with associated effects; SAM2 converts their bounding boxes into masks. GPT then assesses difficulty from the

validated source–mask pairs without observing erase-model outputs. Poster, Watermark, and Visual Text contribute 2K tasks each through high-resolution layered removal, controlled synthesis, and screened text removal with text-centered cropping, respectively. The model receives only I_{input} and M. General Data requires no paired target; Poster and Watermark use deterministic clean references, whereas Visual Text uses verified weak references for rewards. Appendix B.3 details the pipelines.

2.3 Evaluation Dataset: LensErase-Bench

LensErase-Bench. Most existing object-removal benchmarks evaluate outputs against paired reference images obtained through synthesis or real-world capture. Synthesized references may contain artifacts, while separately captured references may differ in lighting or scene details. Since object removal also admits multiple valid outcomes, agreement with a single reference may not fully reflect removal quality. To complement reference-based metrics, we introduce *LensErase-Bench*, a benchmark of 500 carefully curated removal cases evaluated without paired ground-truth target images. A VLM evaluates each output against the source image and selection mask. The Poster subset uses layered designs from Crello (Yamaguchi, 2021). We also include image–mask pairs selected from existing segmentation datasets (Qi et al., 2023; Deng et al., 2023; Xie et al., 2022). We normalize the selected images to a shorter side above 512 pixels and a longer side below 2048 pixels while preserving aspect ratio. We carefully curate the benchmark to provide balanced coverage across multiple dimensions, including application scenarios, mask area ratio, image resolution, selected-target count, and erase difficulty. These factors are jointly considered to ensure that LensErase-Bench is both diverse and representative of practical applications. Each evaluation sample consists of an input image I_{input} and an erase mask M. Appendix C details the selection constraints and distributions.

Evaluation Metrics. We use GPT-5.6 Sol (OpenAI, 2026a) as the VLM judge to evaluate each output against the input image and selection mask along four dimensions: *Removal*, the completeness of removing selected targets and their associated effects; *Inpainting*, plausible completion consistent with the surrounding context; *Preservation*, retention of non-target content; and *Cleanliness*, absence of artifacts and unrelated additions. Each dimension receives an integer rating from 0 to 4, rescaled to 0–100 for reporting. *Overall* weights these dimensions by 40%, 20%, 20%, and 20%, respectively. We additionally report the *High-Fidelity Removal Rate* (*HiFi Removal*, %), the percentage of all valid cases that achieve complete removal while meeting the quality requirements for inpainting, preservation, and visual cleanliness. Specifically, a case is successful when Removal $= 4$ and Inpainting, Preservation, and Cleanliness are all ≥ 3 on the original 0–4 scale. Mean scores are computed over all valid cases with equal weight per image. Detailed scoring rules and the evaluation protocol are provided in Appendix C.2.

3 Method

Given a source image I_{input} and a binary selection mask M, LensErase produces an output image $\hat{I}_{\text{output}}$ that removes the selected objects and their associated visual effects, while preserving the remaining scene and plausibly completing the removed regions. Figure 3 shows the architecture.

3.1 Pretraining: Adapting Generation to Erasure

Architecture and Initialization. We initialize LensErase from Lens (Chen et al., 2026), a text-to-image model whose MMDiT backbone contains a condition branch for the text modality and an image branch for the visual modality. LensErase follows the same backbone architecture, but both branches are initialized from the pretrained image branch of Lens. This is because, in the erase task, the condition input, including the mask M and source image features, is also visual rather than textual. Figure 3 illustrates the architecture of LensErase: the image branch processes the noisy target latent, while the condition branch processes the source latent. The erase mask M, with 1 denoting the masked region and 0 the unmasked region, is incorporated into the condition branch through the proposed *Mask-Aware RoPE* with negligible computational overhead.

We retain the complete source image, including the selected region. The selected object's appearance and location help identify associated shadows and reflections, while semi-transparent overlays retain useful information about the underlying background. Masking out the selected region would therefore discard information useful for both effect removal and reconstruction.

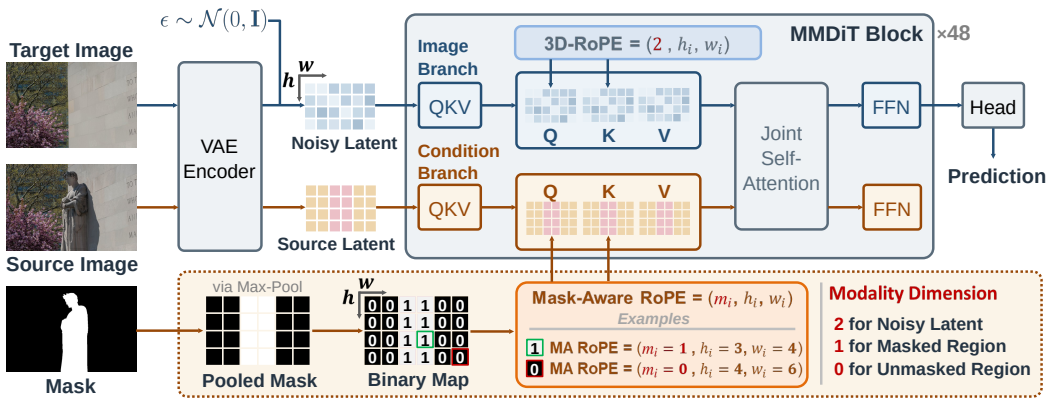

Figure 3: **Architecture of LensErase.** LensErase follows a two-branch MMDiT architecture, where the image branch processes the noisy target latent and the condition branch processes the source latent. The proposed *Mask-Aware RoPE* incorporates the pooled binary mask into the condition branch by encoding masked and unmasked source regions along an additional modality dimension.

Mask-Aware RoPE. We apply RoPE (Su et al., 2024) to both noisy target tokens and source tokens. To encode the erase mask in the source tokens, we extend conventional 2D spatial RoPE to a 3D formulation by introducing an additional modality dimension. Specifically, we first downsample the binary mask M to the source-token grid via max pooling, yielding $m_i \in \{0, 1\}$ for each source token. We then define the positional coordinate for source tokens as

$$\mathbf{p}_i^{\text{cond}} = (m_i, h_i, w_i), \tag{1}$$

where (h_i, w_i) are spatial coordinates and $m_i = 1$ ($m_i = 0$) denotes a masked (unmasked) source token. This directly encodes the mask into the positional representation without introducing additional mask tokens or embedding layers. To represent the noisy target tokens within the same 3D RoPE formulation, we assign them a proxy modality coordinate of 2:

$$\mathbf{p}_i^{\text{img}} = (2, h_i, w_i). \tag{2}$$

Region-Weighted Flow Matching. We pretrain LensErase on LensErase-100M using a region-weighted flow-matching objective. Since object removal often affects only a small image region, a uniform objective can be dominated by unchanged background. Let E denote the VAE encoder, with target latent $z^\star = E(I_{\text{output}})$ and source latent $z_{\text{src}} = E(I_{\text{input}})$. Following flow matching (Lipman et al., 2022), we construct $z_t = (1-t)z^\star + t\epsilon$ with target velocity $v^\star = \epsilon - z^\star$, where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$. Conditioned on z_{src} and the erase mask M, the model predicts the velocity $v_\theta(z_t, t, z_{\text{src}}; M)$. To emphasize the intended edit, we assign larger loss weights to tokens within the masked region:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{I_{\text{input}}, I_{\text{output}}, M, t, \epsilon}\left[\frac{\sum_{i=1}^{N} \omega_i \| v_{\theta,i}(z_t, t, z_{\text{src}}; M) - v_i^\star \|_2^2}{\sum_{i=1}^{N} \omega_i}\right]. \tag{3}$$

Here, i indexes the N target tokens and $\omega_i = 1 + (\lambda - 1)\tilde{M}_i$, where $\tilde{M}$ is the dilated mask downsampled to the token grid and $\lambda = 5$. The normalization reduces sensitivity of the overall loss scale to mask size. Further pretraining details and notation are provided in Appendices D and E.6.

Supervised Fine-Tuning (SFT). After pretraining, we perform SFT on the curated SFT set using LoRA (Hu et al., 2021), while retaining the same region-weighted flow-matching objective. Training details are provided in Appendix E.1.

3.2 RL: Contract-Guided Erasure

The SFT model already produces strong removals, but its success can vary across stochastic samples for the same image–mask input. We therefore use RL to increase the likelihood of complete removal while preserving non-target content. Figure 6 in Appendix E.2 summarizes the reward pipeline.

1. Constructing the Erase Contract. For each General Data source image I_{input} and selection mask M, we first construct a source-grounded, candidate-independent erase contract that specifies the intended edit. GPT-5.6 Sol (OpenAI, 2026a) examines the source image, mask, and mask-centered

context views to produce $\mathcal{K} = (\mathcal{T}, \mathcal{E}, \mathcal{O})$. The removal targets $\mathcal{T}$ specify the selected entities and key visual components whose presence would indicate incomplete removal. The associated effects $\mathcal{E}$ describe target-dependent effects, such as shadows or reflections, that should also disappear. The protected objects $\mathcal{O}$ identify nearby independent content that should remain unchanged. Thus, $\mathcal{T}$ and $\mathcal{E}$ specify what should disappear, while $\mathcal{O}$ specifies what should be preserved. The same contract is shared across all rollouts for a given image–mask condition, ensuring consistent reward evaluation.

2. Policy Rollouts and Reward Computation. The SFT-initialized policy generates multiple stochastic candidates per image–mask pair, which are evaluated against the shared erase contract. GPT-5.6 Sol provides four semantic scores: *Removal* assesses whether targets in $\mathcal{T}$ and effects in $\mathcal{E}$ are fully removed; *Reconstruction* evaluates the plausibility and contextual consistency of synthesized content; *Preservation* measures retention of protected objects in $\mathcal{O}$ and other non-target content; and *Cleanliness* captures residual fragments, unwanted additions, and artifacts. We additionally compute *Outside Fidelity* as source–candidate PSNR over expected unchanged regions. Reference-guided adaptations for other domains are detailed in Appendix E.2.

3. Removal-Aware Reward Gating. A weighted average can assign a high reward to an edit that performs well on reconstruction or preservation but still exhibits clear removal failures. We therefore first aggregate the five reward components, namely *Removal*, *Reconstruction*, *Preservation*, *Cleanliness*, and *Outside Fidelity*, into a base reward:

$$b = \sum_{p \in \mathcal{P}} \lambda_p P_p, \qquad \sum_{p \in \mathcal{P}} \lambda_p = 1, \tag{4}$$

where $\mathcal{P} = \{\mathrm{Removal}, \mathrm{Reconstruction}, \mathrm{Preservation}, \mathrm{Cleanliness}, \mathrm{OutsideFidelity}\}$, and $P_p \in [0, 1]$ denotes the normalized score of reward component p. We suppress the candidate index j; component and subcriterion weights are renormalized over applicable terms, as detailed in Appendix E.2. The *Removal* component is further decomposed into multiple subcriteria. Each zero-valued applicable Removal subcriterion incurs an additional penalty factor of 0.5. Specifically, let

$$n_0 = \sum_{q \in \mathcal{Q}_{\mathrm{Removal}}} \mathbb{1}[s_q = 0], \tag{5}$$

where $\mathcal{Q}_{\mathrm{Removal}}$ denotes the set of applicable Removal subcriteria and $s_q \in [0, 1]$ is the normalized score of subcriterion q. The final reward is

$$r = 0.5^{\,n_0} b. \tag{6}$$

This progressively penalizes candidates with multiple removal failures, reducing the extent to which strong performance on other criteria can compensate for them.

4. Optimization. We optimize the SFT-initialized LensErase model with DiffusionNFT (Zheng et al., 2026). Implementation details are provided in Appendix E.3.

3.3 Post-Processing with TraceFusion

Although LensErase primarily modifies the selected target and associated effects, VAE reconstruction may introduce small unintended changes in otherwise unchanged regions. TraceFusion estimates the effective edit region from source–output residuals, preserves generated content there, and restores the source image elsewhere. This improves the fidelity of unchanged content while retaining intended edits that may extend beyond the input mask, such as removed shadows or reflections.

We compute pixel-wise residuals between the generated and source images after compensating for reconstruction bias in unchanged regions. On the 0–255 scale, pixels above 4 form candidate change components, of which only those containing sufficient pixels above 10 are retained to suppress small or fragmented artifacts. This allows the estimated edit region to extend beyond the input mask to associated effects such as shadows or reflections. Detailed criteria are provided in Appendix E.5.

We then combine the detected residual support with a dilated input mask to form the primary edit support. Because changes in dark regions can produce weak residuals in the original intensity space, we additionally apply the same detector after a shared $\gamma = 0.75$ transform to both images. This gamma branch only supplements the primary support: newly detected components are retained only when they connect to the primary edit support and predominantly lie in dark regions of both images.

Table 1: **Comparison with state-of-the-art object removal methods on six public benchmarks.** *For each benchmark, we report the three baselines with the lowest mean rank across the eight metrics together with LensErase*; complete results for all 12 baselines are provided in Tables 10-15 of the Appendix. Cell backgrounds mark the **best**, second-best, and third-best scores among these four methods.

Method	PSNR ↑	FID ↓	CMMD ↓	LPIPS ↓	SSIM ↑	DISTS ↓	CLIP-I ↑	DINO ↑
OBER-Test (Zhao et al., 2026)								
ObjectClear (Zhao et al., 2026)	33.05	19.16	0.0132	0.0338	0.9353	0.0217	0.9894	**0.9839**
OSOR (Zhou et al., 2026)	30.14	27.82	0.0674	0.0524	0.8748	0.0377	0.9827	0.9693
OmniPaint (Yu et al., 2025)	28.79	23.68	0.0829	0.0549	0.8575	0.0427	0.9836	0.9803
LensErase	**34.96**	**16.97**	**0.0073**	**0.0256**	**0.9543**	**0.0176**	**0.9930**	**0.9839**
RORD-Val-343 (Sagong et al., 2022)								
ObjectClear (Zhao et al., 2026)	26.21	25.42	0.0511	0.1163	0.8577	0.0657	0.9704	0.9566
OSOR (Zhou et al., 2026)	25.67	26.33	0.0390	0.1037	0.8138	0.0542	0.9663	0.9475
OmniPaint (Yu et al., 2025)	23.64	27.45	0.0334	0.1119	0.7823	0.0623	0.9672	0.9541
LensErase	**28.23**	**20.53**	**0.0272**	**0.0601**	**0.9053**	**0.0357**	**0.9715**	**0.9568**
RemovalBench (Wei et al., 2025)								
OSOR (Zhou et al., 2026)	25.80	43.89	0.0275	0.1935	0.7656	0.0882	0.9838	0.9597
OmniPaint (Yu et al., 2025)	24.57	43.25	0.0265	0.2103	0.7535	0.0998	0.9821	0.9663
ObjectClear (Zhao et al., 2026)	25.99	46.61	0.0248	0.2853	0.7794	0.1428	0.9853	0.9718
LensErase	**28.16**	**33.67**	**0.0116**	**0.1125**	**0.8306**	**0.0586**	**0.9874**	**0.9737**
OmniPaint-Bench (captured subset) (Yu et al., 2025)								
OSOR (Zhou et al., 2026)	26.02	37.52	0.0687	0.1336	0.8421	0.0566	0.9617	0.9506
OmniPaint (Yu et al., 2025)	23.42	40.65	0.0743	0.1484	0.8193	0.0644	0.9562	0.9423
GeoRemover (Zhu et al., 2025b)	24.87	38.54	0.1376	0.1575	0.8173	0.0704	0.9524	0.9494
LensErase	**26.22**	**33.29**	**0.0245**	**0.1257**	**0.8521**	**0.0544**	**0.9660**	**0.9595**
AnimeEraseBench (Zhou et al., 2026)								
OSOR (Zhou et al., 2026)	27.96	24.45	**0.0094**	0.0967	0.8870	0.0475	0.9900	0.9751
OmniPaint (Yu et al., 2025)	25.58	27.76	0.0138	0.1195	0.8581	0.0636	0.9874	0.9764
ObjectClear (Zhao et al., 2026)	25.42	36.76	0.0121	0.1791	0.8417	0.0978	0.9914	**0.9821**
LensErase	**30.50**	**17.20**	**0.0094**	**0.0527**	**0.9398**	**0.0159**	**0.9920**	0.9798
TextEraseBench (Zhou et al., 2026)								
OSOR (Zhou et al., 2026)	32.04	18.52	0.0093	0.0728	0.9126	0.0412	0.9843	0.9771
OmniEraser (Wei et al., 2025)	30.70	21.97	0.0195	0.0763	0.9005	0.0435	0.9812	0.9758
RORem-mixed (Li et al., 2025)	29.50	24.83	0.0178	0.1034	0.8292	0.0584	0.9815	0.9744
LensErase	**36.88**	**13.51**	**0.0091**	**0.0339**	**0.9654**	**0.0135**	**0.9906**	**0.9822**

Finally, we feather the boundary of the resulting edit support to obtain a blending weight $\alpha \in [0, 1]$. The generated image is preserved inside the estimated edit region, while the source image is restored elsewhere. Blending is performed in linear RGB:

$$\hat{I}^{\mathrm{TF}}_{\mathrm{output}} = \mathcal{C}^{-1}\Big[\alpha \odot \mathcal{C}(\hat{I}_{\mathrm{output}}) + (1-\alpha) \odot \mathcal{C}(I_{\mathrm{input}})\Big], \tag{7}$$

where $\mathcal{C}$ converts sRGB values to linear RGB.

4 Experiments

Benchmarks. We evaluate on seven benchmarks, including six public benchmarks: OBER-Test (Zhao et al., 2026), RORD-Val-343 (Sagong et al., 2022), RemovalBench (Wei et al., 2025), OmniPaint-Bench (Yu et al., 2025), AnimeEraseBench, and TextEraseBench (Zhou et al., 2026), together with our proposed LensErase-Bench (Section 2.3). For OmniPaint-Bench, we use only its 300 captured cases, excluding the RORD-derived subset to avoid overlap with RORD-Val-343.

Baselines. We compare against 12 publicly released methods: OSOR (Zhou et al., 2026), ObjectClear (Zhao et al., 2026), DORS (Tang et al., 2026), PANDORA (Vo et al., 2026), GeoRemover (Zhu et al., 2025b), SmartEraser (Jiang et al., 2025), RORem-mixed (Li et al., 2025), EntityErasure (Zhu et al., 2025a), Attentive Eraser (Sun et al., 2025), DesignEdit (Jia et al., 2025), OmniEraser (Wei et al., 2025), and OmniPaint (Yu et al., 2025). Related methods are reviewed in Appendix A, and detailed baseline configurations are provided in Appendix F.2.

Implementation Details. Pretraining takes approximately two weeks on 96 NVIDIA A100 GPUs. SFT and RL are each conducted on 16 GPUs, taking approximately three and five days, respectively. Additional implementation details are provided in Appendix E.

Table 2: **Comparison on LensErase-Bench.** We report the six representative baselines together with LensErase; complete results for all 12 baselines are provided in Table 16 of the Appendix.

Method	Removal ↑	Inpainting ↑	Preservation ↑	Cleanliness ↑	Overall ↑	HiFi Removal (%) ↑
OSOR (Zhou et al., 2026)	73.85	60.45	75.55	73.55	71.45	32.60
OmniPaint (Yu et al., 2025)	69.20	66.15	80.20	83.20	73.59	42.80
ObjectClear (Zhao et al., 2026)	73.95	61.45	73.80	75.20	71.67	28.40
GeoRemover (Zhu et al., 2025b)	54.30	49.85	77.05	78.00	62.70	25.60
OmniEraser (Wei et al., 2025)	55.65	47.95	77.55	57.65	58.89	22.60
RORem-mixed (Li et al., 2025)	73.95	59.70	74.05	67.10	69.75	27.40
LensErase	**95.15**	**92.40**	**96.35**	**97.05**	**95.22**	**82.80**

Table 3: **Stage-wise ablation of LensErase.** We report performance after pre-training, SFT, RL, and TraceFusion. Results on LensErase-Bench are shown with all evaluation metrics, while results on public benchmarks report the average PSNR and LPIPS across the six public benchmarks.

Model	LensErase-Bench						Public Benchmarks	
	Removal ↑	Inpainting ↑	Preservation ↑	Cleanliness ↑	Overall ↑	HiFi Removal ↑	PSNR ↑	LPIPS ↓
Pre-Training	82.35	84.65	91.05	92.15	86.51	65.80	28.03	0.0855
+SFT	88.65	87.15	92.05	93.05	89.91	70.80	28.14	0.0826
+RL	94.95	92.65	93.35	96.15	94.41	81.60	28.70	0.0816
+TraceFusion	95.15	92.40	96.35	97.05	95.22	82.80	30.82	0.0684

Table 4: **Comparison of mask injection strategies.** Mask-Aware RoPE is compared with channel- and token-concatenation baselines, all evaluated at the pre-training stage. Both baselines obtain mask features via pixel unshuffle of 16×16 patches. *Channel Concat* concatenates source and mask features along channels and fuses them with an MLP before condition projection. *Token Concat* places source tokens alongside noisy tokens in the image branch and projects mask features to the condition branch. Both use positional indices independent of mask values.

Model	LensErase-Bench						Public Benchmarks	
	Removal ↑	Inpainting ↑	Preservation ↑	Cleanliness ↑	Overall ↑	HiFi Removal ↑	PSNR ↑	LPIPS ↓
Mask-Aware RoPE	82.35	84.65	91.05	92.15	86.51	65.80	28.03	0.0855
Channel Concat	81.50	84.55	82.90	91.15	84.32	58.80	27.84	0.0869
Token Concat	77.50	76.85	84.90	87.15	80.78	51.40	27.41	0.0965

Metrics. For the six public benchmarks with ground-truth target images, we report PSNR, SSIM, LPIPS, DISTS, CLIP-I, DINO, FID, and CMMD. For our LensErase-Bench, we adopt the evaluation metrics described in Section 2.3.

State-of-the-Art Performance Across Six Public Benchmarks. As shown in Table 1, LensErase achieves state-of-the-art performance across six public benchmarks. For clarity, we report only the three baselines with the lowest mean rank across the eight metrics per benchmark; full comparisons with all 12 baselines are provided in Tables 10–15 of the Appendix.

State-of-the-Art Performance on LensErase-Bench. We further compare LensErase with 12 publicly released baselines on the proposed LensErase-Bench, which contains more challenging and diverse removal cases. As shown in Table 2, LensErase achieves the best overall performance. For clarity, we report six representative baselines together with LensErase in the main table. Complete comparisons with all 12 baselines are provided in Table 16 of the Appendix.

Stage-wise Performance. Table 3 shows that HiFi Removal increases from 65.80% after pretraining to 70.80% after SFT and 81.60% after RL. TraceFusion further improves mean PSNR from 28.70 to 30.82 dB and LPIPS from 0.0816 to 0.0684, while reaching 82.80% HiFi Removal. Appendix G reports reward-component ablations.

Effective Design for Mask Injection. Section 3.1 introduces our Mask-Aware RoPE for injecting mask information. In Table 4, we compare it with two variants, *Channel Concat* and *Token Concat*. Appendix D provides further pretraining ablations.

Visualizations. Appendix Figures 7–17 compare LensErase with closed-source commercial models (mask-based GPT-Image 2.5-Sunburst (OpenAI, 2026b) and Flux2-Erase (Black Forest Labs, 2026)) and open-source baselines. Appendix J shows further examples using hand-drawn and SAM masks.

AI USE STATEMENT

The research ideas, methodology, experiments, and analysis were developed and conducted by the authors. During manuscript preparation, generative AI assistants were used only for language editing and polishing. Separately, generative and multimodal models were used as explicitly disclosed components of our data-construction, curation, reward-modeling, and evaluation pipelines.

ETHICS STATEMENT

LensErase is developed for research and practical image-editing applications. As with other general-purpose image-editing technologies, we encourage its responsible use in accordance with applicable data-use, copyright, and privacy requirements. All authors have read and adhere to the ICLR Code of Ethics.

REPRODUCIBILITY STATEMENT

We describe the data-construction pipelines, model architecture, training objectives, post-training procedure, inference algorithm, and evaluation setup in the main paper and Appendix. The appendices further report dataset curation criteria, model initialization and resolution buckets, key training configurations, reward construction, sampling and TraceFusion parameters, baseline configurations, and benchmark preprocessing. These details are provided to clarify the experimental protocol and facilitate reproduction of the reported results.

REFERENCES

Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18187–18197. IEEE, 2022.

Omri Avrahami, Ohad Fried, and Dani Lischinski. Blended latent diffusion. *ACM transactions on graphics (TOG)*, 42(4):1–11, 2023.

Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 28(3):24, 2009.

Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 417–424, 2000.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pp. 4965–4987, 2024.

Black Forest Labs. FLUX Erase. `https://docs.bfl.ai/flux_tools/flux_erase`, 2026. Accessed: 2026-09-21.

Chenjie Cao, Qiaole Dong, and Yanwei Fu. Zits++: Image inpainting by improving the incremental transformer on structural priors. *IEEE transactions on pattern analysis and machine intelligence*, 45(10):12667–12684, 2023.

Clément Chadebec, Onur Tasar, Sanjeev Sreetharan, and Benjamin Aubin. Lbm: Latent bridge matching for fast image-to-image translation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 29086–29098. IEEE, 2025.

Dong Chen, Fangyun Wei, Ziyu Wan, Dongdong Chen, Jiawei Zhang, Jinjing Zhao, Sirui Zhang, Yang Yue, Zhiyang Liang, Baining Guo, et al. Lens: Rethinking training efficiency for foundational text-to-image models. *arXiv preprint arXiv:2605.21573*, 2026.

Xi Chen, Lianghua Huang, Yu Liu, Yujun Shen, Deli Zhao, and Hengshuang Zhao. Anydoor: Zero-shot object-level image customization. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6593–6602. IEEE, 2024a.

Zhekai Chen, Wen Wang, Zhen Yang, Zeqing Yuan, Hao Chen, and Chunhua Shen. Freecompose: Generic zero-shot image composition with diffusion prior. In *European Conference on Computer Vision*, pp. 70–87. Springer, 2024b.

Kevin Clark, Paul Vicol, Kevin Swersky, and David Fleet. Directly fine-tuning diffusion models on differentiable rewards. In *International Conference on Learning Representations*, volume 2024, pp. 4793–4822, 2024.

Antonio Criminisi, Patrick Pérez, and Kentaro Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on image processing*, 13(9):1200–1212, 2004.

Cheng Cui, Ting Sun, Manhui Lin, Tingquan Gao, Yubo Zhang, Jiaxuan Liu, Xueqing Wang, Zelun Zhang, Changda Zhou, Hongen Liu, Yue Zhang, Wenyu Lv, Kui Huang, Yichao Zhang, Jing Zhang, Jun Zhang, Yi Liu, Dianhai Yu, and Yanjun Ma. Paddleocr 3.0 technical report, 2025. URL https://arxiv.org/abs/2507.05595.

Xinhao Deng, Pingping Zhang, Wei Liu, and Huchuan Lu. Recurrent multi-scale transformer for high-resolution salient object detection. In *Proceedings of the 31st ACM International Conference on Multimedia*, pp. 7413–7423, 2023. doi: 10.1145/3581783.3611983.

Qiaole Dong, Chenjie Cao, and Yanwei Fu. Incremental transformer structure enhanced image inpainting with masking positional encoding. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11348–11358. IEEE, 2022.

Yiğit Ekin, Ahmet B Yildirim, Erdem E Caglar, Aykut Erdem, Erkut Erdem, and Aysegul Dundar. Clipaway: Harmonizing focused embeddings for removing objects via diffusion models. *Advances in Neural Information Processing Systems*, 37:17572–17601, 2024.

Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in neural information processing systems*, 36:79858–79885, 2023.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

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. *arXiv preprint arXiv:2106.09685*, 2021.

Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ACM Transactions on Graphics (ToG)*, 36(4):1–14, 2017.

Sadeep Jayasumana, Srikumar Ramalingam, Andreas Veit, Daniel Glasner, Ayan Chakrabarti, and Sanjiv Kumar. Rethinking fid: Towards a better evaluation metric for image generation. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9307–9315. IEEE, 2024.

Yueru Jia, Aosong Cheng, Yuhui Yuan, Chuke Wang, Ji Li, Huizhu Jia, and Shanghang Zhang. Designedit: unify spatial-aware image editing via training-free inpainting with a multi-layered latent diffusion framework. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 3958–3966, 2025.

Longtao Jiang, Zhendong Wang, Jianmin Bao, Wengang Zhou, Dongdong Chen, Lei Shi, Dong Chen, and Houqiang Li. Smarteraser: Remove anything from images using masked-region guidance. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24452–24462. IEEE, 2025.

Sanghyun Jo, Donghwan Lee, Eunji Jung, Seong Je Oh, and Kyungsu Kim. Eraselora: Mllm-driven foreground exclusion and background subtype aggregation for dataset-free object removal. *arXiv preprint arXiv:2512.21545*, 2025.

Xuan Ju, Xian Liu, Xintao Wang, Yuxuan Bian, Ying Shan, and Qiang Xu. Brushnet: A plug-and-play image inpainting model with decomposed dual-branch diffusion. In *European Conference on Computer Vision*, pp. 150–168. Springer, 2024.

Maksim Kuprashevich, Grigorii Alekseenko, Irina Tolstykh, Georgii Fedorov, Bulat Suleimanov, Vladimir Dokholyan, and Aleksandr Gordeev. Nohumansrequired: Autonomous high-quality image editing triplet mining. In *2026 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 6059–6068. IEEE, 2026.

Fan Li, Zixiao Zhang, Yi Huang, Jianzhuang Liu, Renjing Pei, Bin Shao, and Songcen Xu. Magiceraser: Erasing any objects via semantics-aware control. In *European Conference on Computer Vision*, pp. 215–231. Springer, 2024.

Ruibin Li, Tao Yang, Song Guo, and Lei Zhang. Rorem: Training a robust object remover with human-in-the-loop. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14024–14035. IEEE, 2025.

Wenbo Li, Zhe Lin, Kun Zhou, Lu Qi, Yi Wang, and Jiaya Jia. Mat: Mask-aware transformer for large hole image inpainting. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10748–10758. IEEE, 2022.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Delong Liu, Haotian Hou, Zhaohui Hou, Shihao Han, Zhiyuan Huang, Mingjie Zhan, Fei Su, and Zhicheng Zhao. Raa: Achieving interactive remove/add anything via fully synthetic data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 7123–7131, 2026.

Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *European conference on computer vision*, pp. 89–105. Springer, 2018.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di ZHANG, and Wanli Ouyang. Flow-GRPO: Training flow matching models via online RL. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025a. URL https://openreview.net/forum?id=oCBKGw5HNf.

Yi Liu, Hao Zhou, Benlei Cui, Wenxiang Shang, and Ran Lin. Erase diffusion: Empowering object removal through calibrating diffusion pathways. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2418–2427. IEEE, 2025b.

Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11451–11461. IEEE, 2022.

Kamyar Nazeri, Eric Ng, Tony Joseph, Faisal Qureshi, and Mehran Ebrahimi. Edgeconnect: Structure guided image inpainting using edge prediction. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 3265–3274. IEEE, 2019.

OpenAI. Introducing GPT-4.1 in the API. https://openai.com/index/gpt-4-1/, 2025.

OpenAI. GPT-5.6 Sol. https://developers.openai.com/api/docs/models/gpt-5.6-sol, 2026a. Accessed: 2026-09-01.

OpenAI. GPT-Image-2.5 Sunburst. https://developers.openai.com/api/docs/models/gpt-image-2.5-sunburst, 2026b. Accessed: 2026-09-21.

Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2536–2544, 2016.

Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pp. 577–582. 2023.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. In *International Conference on Learning Representations*, volume 2024, pp. 1862–1874, 2024.

Mihir Prabhudesai, Anirudh Goyal, Deepak Pathak, and Katerina Fragkiadaki. Aligning text-to-image diffusion models with reward backpropagation. 2023.

Lu Qi, Jason Kuen, Tiancheng Shen, Jiuxiang Gu, Weidong Guo, Jiaya Jia, Zhe Lin, and Ming-Hsuan Yang. High-quality entity segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. URL https://arxiv.org/abs/2408.00714.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10674–10685. ieee, 2022.

Min-Cheol Sagong, Yoon-Jae Yeo, Seung-Won Jung, and Sung-Jea Ko. Rord: A real-world object removal dataset. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2022.

Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH 2022 conference proceedings*, pp. 1–10, 2022.

Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Wenhao Sun, Xue-Mei Dong, Benlei Cui, and Jingqun Tang. Attentive eraser: Unleashing diffusion model's object removal potential via self-attention redirection guidance. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 20734–20742, 2025.

Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. Resolution-robust large mask inpainting with fourier convolutions. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 3172–3182. IEEE, 2022.

Haitong Tang, Haipeng Liu, and Yang Wang. Dors: Dynamic attention routing for diffusion-based object removal in dense scenes. *arXiv preprint arXiv:2607.16656*, 2026.

Dinh-Khoi Vo, Thanh-Toan Do, Tam V Nguyen, Minh-Triet Tran, and Trung-Nghia Le. Cpam: Context-preserving adaptive manipulation for zero-shot real image editing. *arXiv preprint arXiv:2506.18438*, 2025.

Dinh-Khoi Vo, Van-Loc Nguyen, Tam V Nguyen, Minh-Triet Tran, and Trung-Nghia Le. Pandora: Pixel-wise attention dissolution and latent guidance for zero-shot object removal. *arXiv preprint arXiv:2603.27555*, 2026.

Ao Wang, Lihao Liu, Hui Chen, Zijia Lin, Jungong Han, and Guiguang Ding. Yoloe: Real-time seeing anything. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 24591–24602. IEEE, 2025.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Runpu Wei, Zijin Yin, Shuo Zhang, Lanxiang Zhou, Xueyi Wang, Chao Ban, Tianwei Cao, Hao Sun, Zhongjiang He, Kongming Liang, et al. Omnieraser: Remove objects and their effects in images with paired video-frame data. *arXiv preprint arXiv:2501.07397*, 2025.

Daniel Winter, Matan Cohen, Shlomi Fruchter, Yael Pritch, Alex Rav-Acha, and Yedid Hoshen. Objectdrop: Bootstrapping counterfactuals for photorealistic object removal and insertion. In *European Conference on Computer Vision*, pp. 112–129. Springer, 2024.

Chenxi Xie, Changqun Xia, Mingcan Ma, Zhirui Zhao, Xiaowu Chen, and Jia Li. Pyramid grafting network for one-stage high resolution saliency detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Waikit Xiu, Qiang Lu, Junbiao Chen, and Xiying Li. Prederase: Training-free object-and-effect removal with predictive latent guidance. *arXiv preprint arXiv:2609.00956*, 2026.

Kota Yamaguchi. Canvasvae: Learning to generate vector graphic documents. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5461–5469. IEEE, 2021.

Siyuan Yang, Lu Zhang, Liqian Ma, Yu Liu, JingJing Fu, and You He. Magicremover: Tuning-free text-guided image inpainting with diffusion models. *arXiv preprint arXiv:2310.02848*, 2023.

Ahmet Burak Yildirim, Vedat Baday, Erkut Erdem, Aykut Erdem, and Aysegul Dundar. Inst-inpaint: Instructing to remove objects with diffusion models. *arXiv preprint arXiv:2304.03246*, 2023.

Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 5505–5514. IEEE, 2018.

Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4471–4480, 2019.

Yongsheng Yu, Ziyun Zeng, Haitian Zheng, and Jiebo Luo. Omnipaint: Mastering object-oriented editing via disentangled insertion-removal inpainting. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 17324–17334. IEEE, 2025.

Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. Magicbrush: A manually annotated dataset for instruction-guided image editing. *Advances in neural information processing systems*, 36:31428–31449, 2023.

Ledun Zhang, Yatu Ji, Xufei Zhuang, and Xinying Yao. Clickremoval: An interactive open-source tool for object removal in diffusion models. *arXiv preprint arXiv:2605.14461*, 2026.

Jixin Zhao, Zhouxia Wang, Peiqing Yang, and Shangchen Zhou. Precise object and effect removal with adaptive target-aware attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19370–19379, 2026.

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. Diffusionnft: Online diffusion reinforcement with forward process. In *International Conference on Learning Representations*, volume 2026, pp. 134129–134150, 2026.

Peng Zheng, Dehong Gao, Deng-Ping Fan, Li Liu, Jorma Laaksonen, Wanli Ouyang, and Nicu Sebe. Bilateral reference for high-resolution dichotomous image segmentation. *CAAI Artificial Intelligence Research*, 3:9150038, 2024.

Qinming Zhou, Chenxi Sun, Deyang Kong, Junhao He, Xiangheng Tang, Peike Yu, Haotian Wu, Leilei Cao, and Linfeng Zhang. Osor: One-step diffusion inpainting for effect-aware object removal. In *European Conference on Computer Vision*, pp. 367–384. Springer, 2026.

Yixing Zhu, Qing Zhang, Yitong Wang, Yongwei Nie, and Wei-Shi Zheng. Entityerasure: Erasing entity cleanly via amodal entity segmentation and completion. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 28274–28283. IEEE, 2025a.

Zixin Zhu, Haoxiang Li, Xuelu Feng, He Wu, Chunming Qiao, and Junsong Yuan. Georemover: Removing objects and their causal visual artifacts. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=RnfyqrkOxD.

Junhao Zhuang, Yanhong Zeng, Wenran Liu, Chun Yuan, and Kai Chen. A task is worth one word: Learning with task prompts for high-quality versatile image inpainting. In *European Conference on Computer Vision*, pp. 195–211. Springer, 2024.

A Related Work

A.1 Image Inpainting and Object Removal

Classical image inpainting propagates structure from hole boundaries or retrieves compatible patches from visible regions (Bertalmio et al., 2000; Criminisi et al., 2004; Barnes et al., 2009). Learning-based methods introduce semantic completion, global–local adversarial supervision, partial or gated convolutions, contextual feature transfer, and explicit structural priors (Pathak et al., 2016; Iizuka et al., 2017; Liu et al., 2018; Yu et al., 2018; 2019; Nazeri et al., 2019). Fourier-based models and mask- or structure-aware Transformers further expand the effective context for completing large and irregular missing regions (Suvorov et al., 2022; Li et al., 2022; Dong et al., 2022; Cao et al., 2023).

Diffusion models provide strong generative priors for reconstructing missing content (Ho et al., 2020; Song et al., 2020; Rombach et al., 2022; Podell et al., 2024). Inpainting methods adapt these priors through constrained resampling, spatial blending, conditional image translation, task prompting, or dedicated conditioning branches (Lugmayr et al., 2022; Avrahami et al., 2022; 2023; Saharia et al., 2022; Zhuang et al., 2024; Ju et al., 2024). Object-removal methods further introduce language or semantic control (Yildirim et al., 2023; Ekin et al., 2024; Li et al., 2024), masked-region guidance (Jiang et al., 2025), amodal completion (Zhu et al., 2025a), and calibrated diffusion pathways (Liu et al., 2025b). A complementary line avoids task-specific training by redirecting attention or modifying latent trajectories (Yang et al., 2023; Sun et al., 2025; Vo et al., 2026; Tang et al., 2026). ClickRemoval provides interactive click-based removal through attention manipulation (Zhang et al., 2026), while EraseLoRA combines multimodal scene interpretation with test-time optimization for dataset-free removal (Jo et al., 2025).

A.2 Data and Models for Image Editing

Editing supervision ranges from manually annotated source–instruction–target triplets in MagicBrush (Zhang et al., 2023) to fully synthetic pairs in RAA (Liu et al., 2026) and automated triplet mining in NoHumansRequired (Kuprashevich et al., 2026). RORem introduces human-in-the-loop refinement to improve removal supervision (Li et al., 2025). Broader image-editing systems study image composition, object customization, and spatial control (Chen et al., 2024b;a; Jia et al., 2025). CPAM emphasizes context preservation during real-image editing (Vo et al., 2025), while LBM formulates image translation through latent bridge matching (Chadebec et al., 2025).

LensErase targets a unified source–mask interface spanning natural objects, visual text, translucent watermarks, and layered designs. Its coverage is supported by large-scale, domain-specific data construction, while visual initialization and mask-aware conditioning adapt a pretrained text-to-image prior to erasing. Supervised fine-tuning and reward-based post-training further refine removal quality and content preservation.

A.3 Object Effects and Background Preservation

Object erasing often requires removing associated effects, such as shadows and reflections, that may extend beyond the selection mask. ObjectDrop explicitly models such effects through counterfactual supervision (Winter et al., 2024). OmniEraser mines paired video frames (Wei et al., 2025), OmniPaint couples removal with insertion (Yu et al., 2025), and GeoRemover separates geometry removal from appearance rendering (Zhu et al., 2025b). ObjectClear reuses target-aware attention to localize associated effects and guide inference-time fusion (Zhao et al., 2026), whereas OSOR combines alpha-region prediction with single-step, mask-robust generation (Zhou et al., 2026). PredErase provides a training-free alternative based on predictive latent guidance (Xiu et al., 2026).

Spatial diffusion blending and gradient-domain editing provide established mechanisms for combining edited content with preserved surroundings (Avrahami et al., 2022; Pérez et al., 2023). LensErase instead retains the intact source image as a visual condition, preserving information about the selected target, its associated effects, and content visible through translucent overlays. At inference, TraceFusion estimates the effective edit support from the selection mask and source–output residuals, then combines generated content with the original image outside this support.

A.4 Reward-Based Post-Training

Reward optimization adapts generative models to criteria that paired reconstruction objectives may not fully capture. DDPO and DPOK formulate denoising as a sequential decision process and optimize terminal rewards with policy gradients, with DPOK additionally regularizing updates toward a pretrained reference (Black et al., 2024; Fan et al., 2023). For differentiable rewards, AlignProp and DRaFT backpropagate through the sampling trajectory (Prabhudesai et al., 2023; Clark et al., 2024). Flow-GRPO extends online policy optimization to flow-matching models through stochastic sampling (Liu et al., 2025a). DiffusionNFT (Zheng et al., 2026) instead learns from scored final rollouts using a forward-process flow-matching objective, with relative rewards defining positive and negative implicit objectives without backpropagating through the sampling trajectory.

LensErase adopts DiffusionNFT with removal-specific rewards that combine semantic assessment of the intended edit with deterministic fidelity measures. The reward separately evaluates removal, region reconstruction, semantic preservation, visual cleanliness, and fidelity outside the edit region, encouraging successful erasure while protecting unrelated content.

B Data Construction and Filtering Details

All supervised data pipelines produce the aligned triplet $(I_{\text{input}}, I_{\text{output}}, M)$. Language is used for data construction and screening, but is never provided as an input to the student model.

Figure 4 summarizes the composition of the 101.07M pretraining examples, 140K SFT examples, and 19K RL tasks. Pretraining and SFT use aligned input–target–mask triplets, whereas RL uses curated image–mask tasks with domain-specific references for reward computation. The following sections detail the construction and filtering procedures for pretraining, SFT, and RL data.

B.1 Pretraining Data Pipelines

B.1.1 General Data: addition-to-removal counterfactuals

Directly asking a generative teacher to remove an object makes the clean target a hallucinated inpainting. We instead begin with a real image I_{output} and ask an editing teacher G_{add} to add an object that was absent from the original scene:

$$I_{\text{input}} = G_{\text{add}}(I_{\text{output}}, P), \qquad M = \text{RecoverMask}(I_{\text{input}}, I_{\text{output}}). \tag{8}$$

The instruction P is used only by the teacher. Reversing the pair therefore provides removal supervision whose target is the untouched real image rather than a generated background.

Scene-aware proposals. Candidate additions are organized into 15 categories: human, animal, apparel and accessories, vehicles, furniture, household items, electronic devices, food and beverages, containers and tableware, sports and leisure, plants, buildings, infrastructure, tools and hardware, and natural elements. Each category contains a fine-grained object catalog. For every image, we sample three distinct coarse categories and provide them, the image, and its caption to GPT-4.1 (OpenAI, 2025). The model proposes one scene-compatible object absent from the image and belonging to one of the sampled categories, returning its coarse category, object class, and insertion instruction. Restricting the proposal to one object reduces ambiguous additions, while mask recovery can still handle occasional multi-instance outputs.

Sampling is dynamically adjusted using accumulated category and object statistics. Overrepresented categories or object classes are excluded from subsequent sampling, while proposals remain constrained by scene compatibility.

Addition teacher. We instantiate G_{add} with an off-the-shelf image editing model. It receives the real image and a short "Add …" instruction and produces only I_{input}; the original is preserved exactly as I_{output}. We retain only spatially aligned pairs. Because the teacher does not provide an accurate object mask, we recover it from the aligned before/after pair.

DCT–DINOv3–SAM2 mask recovery. A raw RGB difference is sensitive to texture noise and incidental appearance changes. We instead form two complementary maps. First, we retain only

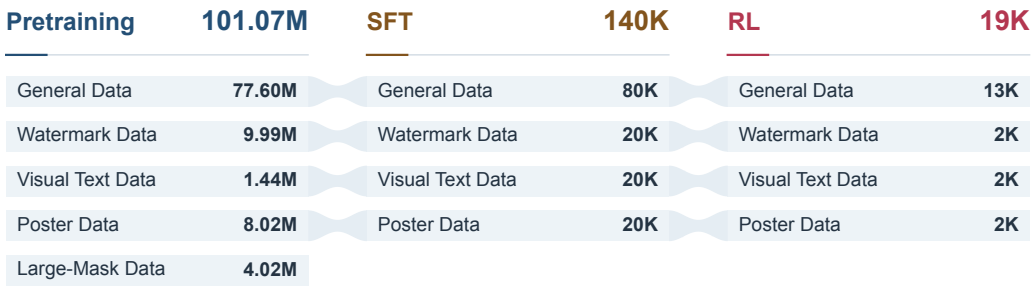

Figure 4: **Data composition across training stages.** Dataset sizes and category counts for pretraining, SFT, and RL. Bands connect shared categories across stages.

low-frequency DCT coefficients in each color channel, reconstruct both images, and compute their absolute difference D_{DCT}. This suppresses high-frequency generation noise while retaining object-scale pixel changes. Second, we compare aligned DINOv3 (Siméoni et al., 2025) dense features using cosine distance,

$$D_{\mathrm{DINO}}(i,j) = 1 - \frac{f_{\text{input}}(i,j)^\top f_{\text{output}}(i,j)}{\|f_{\text{input}}(i,j)\|_2 \|f_{\text{output}}(i,j)\|_2}. \tag{9}$$

Here f_{input} and f_{output} denote the dense features of I_{input} and I_{output}, respectively. Thresholding and morphological opening/closing give binary maps B_{DCT} and B_{DINO}. When both are nonempty, we use

$$B_{\mathrm{coarse}} = B_{\mathrm{DCT}} \cup B_{\mathrm{DINO}}, \qquad B_{\mathrm{pos}} = B_{\mathrm{DCT}} \cap B_{\mathrm{DINO}}. \tag{10}$$

The union maximizes recall for the possible edit extent, while the intersection provides conservative positive-point candidates. Cross-signal IoU is retained as a filtering indicator.

SAM2-based mask refinement. To recover object boundaries from the coarse change maps, we apply SAM2 (Ravi et al., 2024) to the object-added image I_{input} using sparse point prompts. Positive candidates are preferentially selected from the interior of B_{pos} using a distance transform, while negative candidates are sought outside B_{coarse}, preferably in a nearby exterior band. Farthest-point sampling selects up to eight positive and four negative points per image to obtain spatially distributed prompts.

The sampled positive points are then grouped by the 8-connected components of B_{pos}. Each component receiving a positive point is processed separately with its own positive prompts and the shared negative prompts. We retain the highest-scoring SAM2 prediction from each group and take their union as the final mask M. The coarse change maps guide prompt placement but are not directly merged into the final mask.

Object-focused masks and associated effects. Our mask recovery combines low-frequency DCT differences for photometric change detection, DINOv3 feature differences for semantic localization, and SAM2 for object-boundary refinement. The DCT branch can capture spatially extended appearance changes, such as cast shadows, while DINOv3 provides complementary semantic cues for localizing the added object. An associated effect may therefore enter the high-recall region B_{coarse} through the DCT branch even when the shared positive region B_{pos} remains concentrated on the object body. SAM2 can then produce an object-focused selection mask without requiring it to cover the entire detected change region.

Because I_{output} is the original image before insertion, the full input–target pair still contains the corresponding associated-effect changes, including those outside M. Such pairs provide supervision for removing the selected object together with its associated effects, without requiring every effect pixel to be explicitly included in the selection mask.

General Data pair filtering. We screen General Data for unreliable localization, ineffective additions, and unintended scene changes. The remaining pairs are assessed using the following measures:

- **Cross-signal agreement.** The intersection-over-union between B_{DCT} and B_{DINO} measures agreement between photometric and semantic change localization. Pairs flagged for low agreement are rejected to reduce supervision based on inconsistent estimates of the edited region.
- **Mask coverage.** The recorded mask-to-image area ratio identifies excessively broad selections. We reject masks with excessive coverage to limit cases in which the recovered selection extends over a large fraction of the scene.
- **Within-mask change.** We compute input–target PSNR within M and reject pairs with unusually high similarity. Such cases exhibit insufficient change in the selected region, as can occur when an insertion fails or the mask includes substantial unchanged background.
- **Outside-mask consistency.** Input–target PSNR over the complement of M measures aggregate changes outside the selection. Low values trigger rejection, targeting outpainting-induced shifts, global appearance drift, and broad background regeneration. This check bounds aggregate discrepancy rather than requiring every exterior pixel to remain identical.
- **Feature-level neighborhood change.** We compare SAM2 features in an exterior band separated from M by a guard margin to reduce contamination from receptive fields overlapping the inserted object. One minus the mean cosine similarity of the least-similar patches provides a score for suspected replacement or damage to neighboring content.
- **Pixel-level neighborhood change.** A complementary score combines the fraction of changed pixels in an adjacent exterior ring with the strength of the largest RGB differences. It measures the spatial extent and intensity of nearby appearance changes, providing pixel-level evidence for the replacement check.

The feature- and pixel-level neighborhood scores form a joint replacement criterion: this criterion triggers only when both scores indicate substantial change. The other rejection criteria are checked separately.

B.1.2 Alpha-supervised watermark synthesis

Watermark-removal pairs are generated by compositing rendered overlays onto clean images, providing known watermark support without image-difference-based mask recovery.

Assets and typography. We combine a pool of approximately 2K logo assets, in raster or SVG format, with template-based multi-lingual watermark text. The text pool includes copyright and ownership notices, preview and sample labels, usage restrictions, photographic attributions, website addresses, and identifiers. Templates are instantiated with sampled company names, domains, years, and numeric IDs. Font faces and sizes are randomized.

Layout and appearance. The renderer samples six synthesis modes: individual logos, standalone text, tiled logos or text, large diagonal text, stripe patterns with optional embedded text, and corner logos. Mode-specific randomization controls position, scale, orientation, opacity, color, and repetition spacing. Additional appearance variations include text outlines or offset shadows, slight logo blur, and translucent backdrops for corner marks. These combinations produce both localized overlays and repeated patterns extending across the image.

Triplet construction. Let (W, α) denote the rendered watermark layer in image coordinates, with $\alpha \in [0, 1]$ incorporating the sampled opacity and any rendered decorations. Alpha compositing gives

$$I_{\text{input}} = \alpha W + (1 - \alpha) I_{\text{output}}, \qquad M = \mathbf{1}[\alpha > 0]. \tag{11}$$

The clean image is retained as I_{output}, while M is derived directly from the rendered alpha support, including disconnected glyphs, repeated elements, semi-transparent marks, and any rendered outlines, shadows, or backdrops.

B.1.3 GENERATED LARGE-FOREGROUND COMPOSITING

Large removal regions are relatively under-represented in General Data. We complement this branch by generating isolated foregrounds and compositing them onto real images. Separating foreground generation from placement allows us to vary object appearance, location, and visible area while retaining the pre-compositing image as the clean removal target.

Foreground diversity. We sample from a catalog of 25 coarse categories and approximately 2.5K object descriptions, spanning people, animals, vehicles, furniture, plants, and everyday objects. Generation prompts combine the sampled object description with viewpoint or pose descriptors, including frontal, side, three-quarter, and top-down views, using varied studio-photography templates that encourage isolated objects on white backgrounds. BiRefNet (Zheng et al., 2024) predicts the foreground alpha matte. We discard degenerate extractions with negligible or nearly frame-filling foreground support, then apply random in-plane rotation and crop to the foreground bounding box. Viewpoint variation and in-plane transformations diversify both object appearance and the geometry of the resulting masks.

Background-aware placement. To vary image position while limiting interference with existing content, we apply YOLOE (Wang et al., 2025) to estimate foreground occupancy in each background image. Candidate anchors are sampled across the central image region, with preference given to locations whose neighborhoods contain less occupied area. Placement aligns the foreground's alpha-weighted centroid with the selected anchor, positioning irregular objects according to their foreground mass rather than the center of their bounding box.

Visible-area and occlusion control. We sample a desired mask-area ratio uniformly from 20%–60% and adjust foreground scale to approach this ratio. Coverage is measured from the foreground actually visible after placement and clipping to the image, rather than from its pre-placement size. We constrain occlusion at two levels: the covered fraction of each sufficiently large detected background instance, and the covered fraction of the union of all detected instances. Placements causing excessive occlusion are retried or discarded. These constraints allow partial overlap while limiting severe loss of visible context. The realized mask area is checked again after the final compositing steps, and samples outside the designated large-mask range are discarded.

Appearance adjustment and triplet construction. We partially adjust foreground color statistics toward those of the local background and feather the alpha boundary to reduce appearance and boundary discontinuities. Let F and α denote the adjusted foreground and its alpha matte in final image coordinates, with $\alpha \in [0, 1]$. We construct

$$I_{\text{input}} = \text{Composite}(I_{\text{output}}, F, \alpha), \qquad M = \mathbf{1}[\alpha > 0.5]. \tag{12}$$

The pre-compositing image remains I_{output}, so the background behind the inserted foreground is known rather than hallucinated by a removal teacher. The selection mask follows the transformed foreground silhouette rather than its rectangular extent, providing object-shaped, large-area removal supervision with a fully specified target.

B.1.4 OCR-VERIFIED PARTIAL VISUAL TEXT REMOVAL

A full-image text-removal teacher may leave residual lettering or alter unrelated scene content. Rather than adopting its output directly as the training target, we use it to propose local reconstructions and construct selection-specific removal pairs through OCR-guided region selection and local blending.

Text localization and teacher editing. Given the original text-containing image I_{input}, we apply PaddleOCR (Cui et al., 2025) with the PP-OCRv6 medium detector and recognizer to obtain text polygons. An image editing teacher receives the source image and a global instruction to erase text while preserving objects, signs, boards, and surfaces, producing a teacher-edited image $\tilde{I}$. We then run PaddleOCR on $\tilde{I}$ to obtain post-edit text detections. We use a higher OCR-confidence threshold on the source image to prioritize reliable text localization and a lower threshold after editing to increase sensitivity to weak residual text. The two OCR passes provide the source locations and residual-text evidence used by the eligibility assessment below.

Grouped selection and mask construction. We group overlapping or nearby text polygons in the source image into joint selection units. A nonempty subset of eligible groups is randomly selected for removal, with eligibility determined by the OCR-based criterion described below. Group-level selection couples the treatment of neighboring or overlapping text regions, while subset sampling varies the removal scope without requiring a separately prompted teacher edit. The union of the selected text polygons, with a small dilation, defines the binary condition mask M. This retains the selected regions' spatial geometry rather than replacing each group with its enclosing rectangle.

Region-restricted target construction. Starting from I_{input}, we blend the selected groups' teacher-edited regions back into the source through arbitrary-mask Poisson blending (Pérez et al., 2023). Blending is performed group by group using the corresponding dilated polygon masks to reduce color and exposure seams from direct patch replacement. The resulting composite is I_{output}: the source image supplies unselected text and surrounding content, while the teacher supplies candidate reconstructions for the selected removal regions. This local reuse of teacher edits constructs partial-removal targets without adopting the teacher's full-image appearance changes. Removal effectiveness and blending quality are assessed using the criteria below.

Visual Text Data pair filtering. We assess text-removal effectiveness separately from artifacts introduced by local paste-back. Failed construction records and missing, undecodable, or empty masks are excluded. Let $\tilde{I}$ denote the teacher-edited image before paste-back. The filtering criteria are:

- **OCR-based removal eligibility.** Original text detections are matched to post-edit detections by bounding-box overlap. A group is selectable only when none of its members has a post-edit match; candidates without a selectable group are discarded. This prevents regions with detected residual text from entering the partial-removal target.
- **Structural change in selected text.** We compute RGB SSIM (Wang et al., 2004) between I_{input} and $\tilde{I}$ within each selected group's dilated polygon support. High similarity indicates insufficient structural change despite the absence of an OCR match. We reject candidates with a high maximum SSIM across groups, so ineffective removal in one group cannot be averaged away by successful edits elsewhere.
- **Boundary gradient discrepancy.** From the paste-back residual $R = I_{\text{output}} - \tilde{I}$, we compute per-channel Sobel gradient magnitudes and take their maximum across RGB channels at each pixel. For each selected group, we average this magnitude in an inner boundary band. A high maximum group mean triggers rejection to screen for boundary leakage introduced by blending.
- **Regional gradient discrepancy.** Using the same residual-gradient map, we compute the 90th-percentile magnitude over each group's full selected region. A high maximum across groups triggers rejection to screen for strong internal discrepancies relative to the teacher reconstruction, complementing the boundary-focused check.

The two gradient criteria are applied independently. They measure discrepancies introduced by paste-back rather than penalizing image gradients that are also present in the teacher reconstruction.

B.1.5 LAYER-AWARE POSTER RENDERING

We construct poster-removal pairs from explicit color–alpha layers with known z-order. The same rendered design defines both the complete poster and a selected-layer removal target, allowing visual diversity without independently regenerating the two images.

Canvas and layout sampling. Background images are fitted to canvases with varied resolutions and square, portrait, landscape, and widescreen aspect ratios. Layout templates guide initial element placement through header–body–footer, centered, grid, sidebar, full-bleed, and diagonal arrangements, alongside unrestricted layouts. Sample plans balance canvas types, layout templates, element types, and element and removal counts, with an initial budget of 4–9 elements. Element sizes are also varied to encourage coverage of different removal extents.

Element and typographic diversity. Elements span text, geometric shapes, decorative graphics, synthetic logos and icons, and presentation-style components such as bullet lists, tables, and chart placeholders. Text pools distinguish titles, subtitles, paragraphs, labels, prices, dates, URLs, and mixed-script content, with multi-lingual examples. Fonts are sampled according to typographic role and, for CJK strings, character coverage; font size and line wrapping are adapted to the allocated region. We further vary color, opacity, alignment, filled versus outlined shapes, and optional drop shadows or outer glows.

Layer-consistent pair construction. Each element is rendered independently and placed in z-order. Layout constraints prevent overlap between text-element bodies, limit excessive occlusion and overlap depth, and check text contrast against the content beneath it. Retained text receives additional protection against severe occlusion, including checks at the character-region level. The planned removal set is refined by excluding candidates overlapped by higher layers designated for retention. We then reuse the identical background and rendered layers to produce I_{input} with all accepted elements and I_{output} with the selected layers omitted. The content, appearance, position, and order of the remaining layers are held fixed.

Body-focused mask construction. For mask construction, we use the element-body layers before shadow or glow is added. A front-to-back traversal assigns each pixel to the topmost body layer with nonzero alpha; pixels assigned to selected layers are unioned and slightly dilated to obtain M. This defines a body-focused selection rather than a bounding-box mask. The paired images nevertheless remove the full selected layers, including their rendered effects, so supervision for effect removal can extend beyond M without requiring an exhaustive effect mask.

B.2 SFT Data Curation and Quality Screening

We curate SFT examples from General, Watermark, Visual Text, and Poster Data, emphasizing high-resolution candidates. Construction follows the domain-specific procedures summarized in Section 2.2, including professionally designed posters rather than the procedural layouts used for pretraining. GPT-5.6 Sol (OpenAI, 2026a) examines each aligned triplet $(I_{\text{input}}, I_{\text{output}}, M)$ using the source image, a mask-overlay visualization, and the target image. When the selected region is difficult to inspect at full-image scale, aligned mask-centered crops of the source and target supplement the full-image views to reveal fine residuals and reconstruction artifacts.

- **Mask validity.** The selection must identify one or more intended targets without substantial leakage into unrelated regions. Coarse boundaries and limited inclusion of nearby background or target-associated effects are permitted when the erase request remains clear.
- **Edit correctness and preservation.** The source–target pair must realize the intended removal completely while preserving unrelated content. Target remnants, unintended deletions, and spatial misalignment lead to rejection.
- **Reconstruction quality.** The revealed region must have plausible geometry and texture without conspicuous editing artifacts. We reject seams, smearing, ghosting, abnormal blur, and implausible local reconstruction.

For General Data only, we additionally require natural photographs or credible photorealistic renderings, excluding illustrations, documents, and poster-like graphics. A triplet is accepted only when its mask is valid, all pair-quality criteria pass, and the overall judgment has medium or high confidence without contradictory failure evidence. Preservation does not require pixel-wise equality outside the intended edit. Screening focuses on correctly realizing the erase request rather than imposing additional scene-level constraints unrelated to it.

B.3 RL Data Pipeline

Task definition and composition. The dataset contains 19K tasks: 13K General Data examples and 2K examples each from Poster Data, Watermark Data, and Visual Text Data. General Data comprises 8K difficulty-filtered tasks and 5K supplementary tasks. Each task conditions LensErase only on the source image (I_{input}) and selection mask (M). General Data requires no paired target, whereas Poster Data and Watermark Data retain deterministic clean references and Visual Text Data

retains verified weak references. Available reference images and structured annotations support data curation and reward evaluation but are not additional inputs to the erase model. During RL, domains are sampled at a ratio of General:Poster:Watermark:Visual Text = 6:2:1:1, independently of their dataset cardinalities; the corresponding reward ablation protocol is reported in Section G.

General Data: failure-mode-guided collection and effect mining. The 5K supplementary tasks augment the difficulty-filtered subset through failure-mode-guided collection and targeted associated-effect mining. During extensive internal evaluation of LensErase-SFT, we manually analyze recurring failures and identify challenging scenarios. We then collect images presenting similar erasure challenges.

For associated-effect mining, GPT-5.6 Sol (OpenAI, 2026a) examines candidate images and selects those containing visibly attributable object-associated effects. These effects are already present in the images rather than synthetically added. They include cast shadows, reflections, water ripples and wakes, splashes, smoke or steam, emitted light, refraction or caustics, contact traces, and motion trails when their causal relation to a selected object is visually supported.

General Data: object localization and mask construction. For newly annotated tasks, GPT-5.6 Sol identifies the target objects and returns their physical-object bounding boxes. For effect-focused tasks, these boxes localize the objects responsible for the effects rather than the effects themselves. Source images and bounding-box overlays are used to verify object localization and, where present, the visibility and causal attribution of associated effects. We then provide the image and verified object boxes to SAM 2.1 (Ravi et al., 2024) to segment the object bodies, taking the union of masks when multiple objects are selected. Further VLM checks assess object-body completeness, instance identity, the natural-image domain, and leakage into non-target content.

The mask is an object-selection cue, not an exhaustive annotation of all pixels that should change. We do not automatically dilate or close it to absorb shadows or reflections. Accurately and compactly captured effects may be included, but do not justify covering large unrelated regions. Only after mask construction and validation do we analyze the finalized source–mask task and assess its difficulty using the procedure below. Thus, difficulty is evaluated for the actual selection, not for the source image or proposed bounding boxes alone.

General Data: quality filtering and difficulty screening. We separate source–mask validity from task difficulty. After mask construction, we exclude invalid conditions, including unidentifiable targets and degenerate masks, and suppress exact and near-duplicate source–mask pairs. The strict pool additionally excludes ambiguous or highly irregular selections, including masks with substantial overlap with independent non-target objects. Thus, strict filtering establishes usable erase conditions rather than assigning a difficulty level.

We then apply a two-stage offline analysis using GPT-5.6 Sol (OpenAI, 2026a). First, the VLM establishes a structured understanding of the selected targets, their constituent parts, associated effects and effect bounding boxes, and nearby independent content to preserve. Second, it estimates the intrinsic difficulty of the same source–mask task without observing an erase-model output. Five integer scores $s_k \in \{0, \ldots, 4\}$ assess, in order, removal scope and associated effects; reconstruction and fine-detail recovery; occlusion and object interactions; preservation and locality risk; and the risk of introducing unwanted entities. With respective weights $(w_1, \ldots, w_5) = (0.25, 0.30, 0.20, 0.20, 0.05)$, overall difficulty is

$$D = 10 \sum_{k=1}^{5} w_k \frac{s_k}{4}. \tag{13}$$

Tasks are categorized as Easy ($D \leq 2$), Moderate ($2 < D \leq 4$), Challenging ($4 < D \leq 6$), Hard ($6 < D \leq 8$), or Extreme ($8 < D \leq 10$). High ratings must reflect multiple substantive challenges rather than mask area alone. From the strict pool, we retain 6,991 Hard and 1,009 Extreme tasks, yielding the 8K difficulty-filtered subset. This score is used for task selection, not as the training reward.

Effects outside the selection remain explicitly described in the source understanding for reward evaluation. Final masks are stored as single-channel binary PNGs with values 0 and 255 at the

source-image resolution, and the associated understanding is bound to hashes of the finalized source and mask.

Poster Data: high-resolution layered removal. The RL Poster Data subset contains 2,000 unique high-quality poster examples designed by professional designers. We rank candidate designs using their native canvas area, side lengths, and layer count, and construct the subset from an ordered pool of 2,500 candidates, replacing ineligible designs until 2,000 valid tasks are obtained. The source canvases are not downscaled during pair construction. Each design contributes one task that removes a single visible non-background layer, which may contain text, graphics, icons, color blocks, or an image. The selected layer may lie behind other foreground layers. Rendering the same layer stack with and without the selected layer gives the source image and an exact removal reference (I_{ref}), following the layer-aware construction principle described in Appendix B.1.5.

We derive the mask from the final rendered RGB difference,

$$M(x) = \mathbf{1}[\exists\, c \in \{R, G, B\} : I_{\mathrm{input}}(x, c) \neq I_{\mathrm{ref}}(x, c)]\,, \tag{14}$$

rather than directly using the selected layer's alpha support. This excludes occluded or otherwise unchanged pixels from the selection. Unselected foreground content that remains visible over the selected layer's spatial extent is part of the correct reference, not evidence of incomplete removal. Candidate selection encourages diversity in removal area, element type, and layer depth, while prioritizing high resolution rather than enforcing equal counts across these attributes.

Watermark Data: controlled synthesis and mask validation. We construct 2,000 Watermark Data tasks using 2,000 unique clean background images selected from a curated collection of 25,000 high-resolution natural images. Requiring both original image dimensions to be at least 1,440 pixels leaves 24,688 eligible backgrounds. For synthesis, we preserve the aspect ratio and prepare a canvas with a maximum long side of 2,048 pixels. The upstream asset library combines logo resources with procedurally generated stamps, signatures, patterns, machine-readable codes, and multilingual text. Asset provenance describes the candidate library; the assets actually used by accepted samples are recorded in the sample manifest. Following the synthesis formulation in Appendix B.1.2, we composite sampled assets onto the clean canvas while varying layout, color, opacity, scale, and rotation, retaining the pre-compositing image as the exact clean reference.

Precise change regions are computed from the final source–reference RGB differences. Half of the tasks use precise masks, while the other half use user-style coarse local selections, such as rectangular or rounded regions, that cover all actual changes without substituting an indiscriminate whole-image box. We validate image–reference–mask dimensions, mask binarity, and change coverage. The final subset enforces marginal quotas over six dimensions: ten asset families with 200 tasks each; four layouts (corner, central, diagonal band, and tiled) with 500 each; synthesized watermark-area bins of 0.1–1%, 1–5%, 5–15%, and 15–45% with 500 each; four opacity bins with 500 each; five requested color families with 400 each; and precise versus coarse masks with 1,000 each. These are marginal quotas, not a requirement to balance every joint attribute combination.

Visual Text Data: curated weak references. The 2,000 Visual Text tasks use the OCR-verified partial-removal procedure in Appendix B.1.4. Pre- and post-edit OCR identify candidate removals, and selected teacher-edited regions are blended back into the source. RL curation further applies VLM screening and text-centered cropping. When selected text occupies only a small part of the image, we crop around the target text, applying the same crop to the source image, selection mask, and weak reference. This increases the selected text's relative extent in the resulting task while maintaining alignment. Selection also considers effective image resolution and mask coverage. The target remains a verified weak reference, since the hidden text-free appearance is not observed directly. Reference-guided rewards therefore assess removal and reconstruction without requiring every plausible completion to match the teacher reconstruction exactly, as detailed in Appendix E.2.

Interface to reward evaluation. Offline source understanding supports task curation and validation; reward evaluation follows the candidate-independent contract construction in Section 3.2 and Appendix E.2. For General Data, the erase contract and aligned detail-view geometry are defined after the training geometry transform and fixed across candidates for the same condition. Poster Data, Watermark Data, and Visual Text Data use their domain-specific references as described in that

appendix. These reward-side specifications do not change the model's source-image–mask conditioning interface. Training resolution buckets and RL optimization are described in Appendices E.1 and E.3, respectively.

C LensErase-Bench: Construction and Evaluation

LensErase-Bench contains 500 cases spanning four erase domains: 200 General cases and 100 each for Poster, Watermark, and Visual Text. The benchmark is curated to cover diverse mask scales, target counts, image resolutions, and removal difficulty. Evaluation follows the reference-free protocol in Appendix C.2.

C.1 Selection and Coverage

Mask-area coverage. We stratify each domain by mask area to ensure that the benchmark covers both highly localized edits and large-region removal. Table 5 reports the resulting domain-specific distribution. General images are evenly distributed across five mask-area ranges, while Poster, Watermark, and Visual Text use bins adapted to their characteristic edit scales.

Table 5: Domain-specific mask-area distribution in LensErase-Bench.

Domain	Mask-area bins (%)	Cases per bin
General	$< 1,\ [1,5),\ [5,15),\ [15,30),\ \geq 30$	40 / 40 / 40 / 40 / 40
Poster	$< 5,\ [5,15),\ \geq 15$	35 / 33 / 32
Watermark	$< 1,\ [1,5),\ [5,15),\ \geq 15$	25 / 25 / 25 / 25
Visual Text	$< 0.5,\ [0.5,1),\ [1,2),\ \geq 2$	25 / 25 / 25 / 25

Attribute coverage. Beyond mask area, we curate complementary attributes that characterize practical erase requests. The General subset is evenly split between cases with and without associated visual effects, and between single-target and multi-target removal. Poster cases are approximately balanced across graphic, image, and text layers, as well as back, middle, and front layer depths. Watermark cases evenly cover ten asset families, four spatial layouts, four opacity ranges, and precise versus user-style coarse masks. Visual Text includes diverse languages, layouts, and background types.

Difficulty and target count. GPT-5.6 Sol (OpenAI, 2026a) rates the filling difficulty of each case on a 0–4 scale. Table 6 reports the distribution of filling difficulty together with the number of selected semantic targets. Across the full benchmark, 59.2% of cases contain a single selected target and 40.8% contain multiple targets, with Watermark providing particularly dense multi-target cases.

Table 6: Distribution of filling difficulty and number of selected targets in LensErase-Bench.

	Filling difficulty				Selected targets		
Domain	0–1	2	3	4	1	2–3	≥ 4
General	9	50	87	54	100	87	13
Poster	34	25	25	16	93	3	4
Watermark	0	9	64	27	40	16	44
Visual Text	25	45	30	0	63	25	12

Resolution coverage. All cases preserve their original aspect ratio and are normalized without cropping or padding. The benchmark spans four long-side resolution ranges, ≤ 1024, 1025–1280, 1281–1536, and 1537–2047 pixels. Each range contains 125 cases: 50 General cases and 25 from each of Poster, Watermark, and Visual Text. This produces balanced resolution coverage across domains while retaining the original selection geometry.

C.2 Evaluation Protocol

LensErase-Bench uses the four evaluation dimensions defined in Section 2.3: Removal, Inpainting, Preservation, and Cleanliness. GPT-5.6 Sol evaluates each candidate using a shared rubric across all four domains, with domain-specific interpretation only where the semantics of the selected target differ.

Evaluation inputs. For each case, the judge receives four aligned full-image views: the source image, the source image with the selection overlaid, the candidate output, and the binary selection mask. No clean reference is provided.

Scoring rubric. Each case receives an integer score from 0 to 4 for each dimension, where 4 denotes full satisfaction of the criterion, 3 a minor defect, 2 a clear localized failure, 1 a major failure, and 0 a fundamentally unsuccessful result. Removal measures whether all selected targets and associated effects disappear; Inpainting measures the plausibility and local consistency of newly revealed content; Preservation measures fidelity of non-target content; and Cleanliness measures artifacts and unrelated additions. Multi-target cases are judged jointly, so a missed selected instance remains a Removal failure even if the remaining targets are successfully erased. To avoid double counting, a single visual defect is attributed to the most directly affected dimension.

Scores and aggregation. Let $r_i^{\mathrm{rem}}, r_i^{\mathrm{inp}}, r_i^{\mathrm{pre}}, r_i^{\mathrm{cln}} \in \{0, 1, 2, 3, 4\}$ denote the Removal, Inpainting, Preservation, and Cleanliness ratings for case i. Each dimension is mapped to 0–100 by multiplying its rating by 25. The case-level Overall Score is

$$o_i = 25 \left[0.4 r_i^{\mathrm{rem}} + 0.2 \left(r_i^{\mathrm{inp}} + r_i^{\mathrm{pre}} + r_i^{\mathrm{cln}} \right) \right]. \tag{15}$$

For each method, we report the mean of each dimension and Overall across its N valid judged cases, with equal weight assigned to every case. Here, $N = 500$ for all methods except PANDORA, for which $N = 499$ because one output was consistently rejected by the judge's safety filter; see Table 16.

HiFi Removal. We additionally report the percentage of cases that achieve complete removal while maintaining high editing quality:

$$\mathrm{HiFiRemoval} = \frac{100}{N} \sum_{i=1}^{N} \mathbb{1} \left[r_i^{\mathrm{rem}} = 4 \ \wedge \ \min\{ r_i^{\mathrm{inp}}, r_i^{\mathrm{pre}}, r_i^{\mathrm{cln}} \} \geq 3 \right], \tag{16}$$

where $\mathbb{1}$ is the indicator function. Overall summarizes average quality across the four dimensions, whereas HiFi Removal measures the fraction of cases that satisfy all removal and editing-quality requirements simultaneously.

D Pretraining Design and Ablations

This section provides additional implementation details and ablations for the initialization, mask conditioning, and two-stage adaptation strategy introduced in Section 3.1.

D.1 Visual Initialization and Mask Conditioning

Both branches of LensErase process visual inputs. We retain the pretrained Lens image branch and initialize the condition branch from the same image-branch weights, while discarding the pretrained text-specific weights. The source image remains intact, preserving visual evidence of the selected target and its associated effects. Mask-Aware RoPE injects the selection mask by assigning different modality coordinates to masked and unmasked source tokens, as defined in Equation 1.

Architecture and positional encoding. The image and condition streams operate on spatial token grids with effective stride 16 relative to the input image. The backbone contains 48 MMDiT blocks, with 24 attention heads per block and 64 channels per head. The three-dimensional RoPE allocates these channels across the modality, height, and width axes as $(8, 28, 28)$. Both streams use the

centered spatial grid of the pretrained positional encoder. The binary selection mask is adaptively max-pooled to the source-token grid and thresholded at 0.5; the resulting value selects between the two condition-stream modality indices in Equation 1. The mask is therefore encoded only through RoPE and is not embedded into the latent channels. RoPE is applied to queries and keys in every block, while values are unchanged. The condition stream is initialized from the image stream's input projection and corresponding attention, feed-forward, normalization, and modulation parameters.

Alternative mask representations. For the two feature-based alternatives in Table 4, we form mask features with $\mathrm{PixelUnshuffle}_{16}(M)$, yielding a 256-dimensional vector aligned with each source-image token. *Channel Concat* concatenates this vector with the corresponding source feature and fuses them with a two-layer MLP before the condition projection. *Token Concat* places source and noisy-image tokens together in the image branch and projects the mask features into the condition branch. Both alternatives use positional indices independent of mask values.

Text-branch initialization control. This control retains the pretrained text-Transformer weights for the condition branch while keeping the same visual inputs, branch routing, and Mask-Aware RoPE as LensErase. Its visual input projection is still initialized from the Lens image branch, and no text prompt is used. This isolates the effect of the pretrained prior used to initialize the condition Transformer.

D.2 Region-Weighted Objective Details

Training follows the flow-matching path and region-weighted objective in Section 3.1. In implementation, the per-token squared velocity error is averaged over channels; Equation 3 omits this fixed normalization factor for clarity. The conditioning mask uses the original selection, whereas the loss support uses a separate 31×31 pixel-space dilation, followed by adaptive max-pooling to the token grid and thresholding at 0.5.

D.3 Two-Stage Pretraining Schedule

To adapt the newly repurposed visual condition stream without immediately disrupting the pretrained image-generation prior, we use a two-stage schedule. Stage 1 trains the condition stream while freezing the image-specific backbone. Stage 2 then unfreezes the image backbone and jointly optimizes both streams using the same region-weighted flow-matching objective (Figure 5).

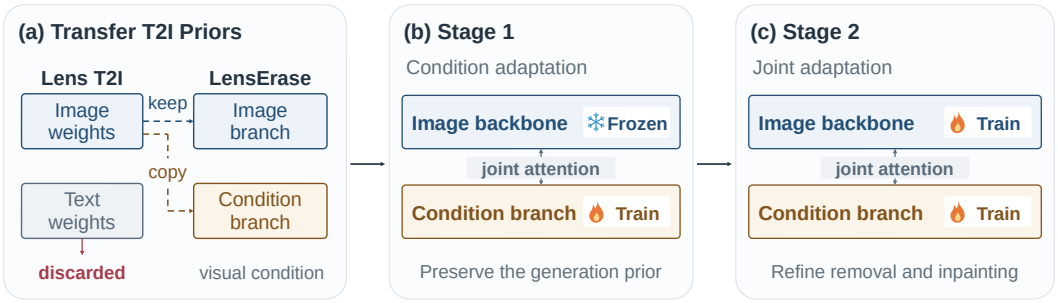

Figure 5: **Initialization and two-stage pretraining.** Both visual branches are initialized from the pretrained Lens image branch. Stage 1 adapts the condition branch while freezing the image-specific backbone. Stage 2 jointly adapts both branches for removal and reconstruction.

In Stage 1, all image-specific parameters are frozen, including the image input projection, image-side attention projections, query/key normalization, modulation, layer normalization, and feed-forward layers. Condition-side parameters remain trainable, while the shared timestep embeddings and output normalization and projection are not frozen. Stage 2 jointly optimizes both streams. Pretraining uses nominal resolution buckets of 512, 768, 1024, 1280, and 1440, each with nine aspect ratios spanning 1:2 to 2:1.

D.4 Initialization and Training Ablations

Table 7 compares alternative conditioning and initialization designs together with different pretraining schedules. The shared reference is the Stage 2 checkpoint obtained with Mask-Aware RoPE, image-branch initialization, and the two-stage schedule. *Stage 1* denotes the endpoint of the initial frozen phase. *No freeze* optimizes both streams from the outset, whereas *Always freeze* continues the Stage 1 checkpoint for the second training phase while keeping the image-specific backbone frozen.

Table 7: **Pretraining design ablations.** The shared reference uses Mask-Aware RoPE, image-branch initialization, and two-stage freeze–unfreeze training. All rows use raw outputs. LensErase-Bench metrics follow Section 2.3; PSNR and LPIPS are equal-weight means over the six public benchmarks.

Model	LensErase-Bench						Public Benchmarks	
	Removal ↑	Inpainting ↑	Preservation ↑	Cleanliness ↑	Overall ↑	HiFi Removal ↑	PSNR ↑	LPIPS ↓
Mask-Aware RoPE (Stage 2)	82.35	84.65	91.05	92.15	86.51	65.80	28.03	0.0855
Conditioning and initialization								
Channel Concat	81.50	84.55	82.90	91.15	84.32	58.80	27.84	0.0869
Token Concat	77.50	76.85	84.90	87.15	80.78	51.40	27.41	0.0965
Text initialization	89.50	1.30	0.05	1.75	36.42	0.00	10.71	0.7420
Training schedule								
Stage 1	79.10	78.95	89.40	90.05	83.32	60.40	26.74	0.1089
No freeze	75.35	76.05	89.75	89.85	81.27	55.80	26.76	0.1020
Always freeze	80.75	80.25	88.00	88.20	83.59	58.60	27.66	0.0864

Visual conditioning and initialization. Mask-Aware RoPE achieves 65.80% HiFi Removal, compared with 58.80% for Channel Concat and 51.40% for Token Concat. Text-branch initialization retains a high Removal score (89.50) but collapses Inpainting to 1.30 and HiFi Removal to 0.00%, indicating that, in this setup, visual initialization is substantially more suitable for the condition stream than text-branch initialization.

Two-stage adaptation. The freeze–unfreeze schedule achieves 65.80% HiFi Removal, compared with 55.80% when both streams are optimized from the outset and 58.60% when the image backbone remains frozen throughout. This supports first adapting the visual condition stream while preserving the pretrained image prior, followed by joint adaptation for removal and reconstruction.

E Post-Training and Inference Details

E.1 Supervised Fine-Tuning

Appendix D describes the initialization and pretraining schedule. SFT uses rank-64 LoRA with scaling parameter 64 and zero dropout, while keeping the VAE and backbone weights fixed. The adapters are applied to the joint query/key/value and attention-output projections of both streams. We retain the same image–mask conditioning interface and region-weighted flow-matching objective as in pretraining. SFT is performed for 5,000 optimization steps using a learning rate of 2×10^{-5} with a cosine schedule. SFT and RL use nominal resolution buckets of 512, 768, 1024, 1280, 1536, and 1792, with nine aspect ratios spanning 1:2 to 2:1.

E.2 Contract-Guided Reward Construction

Following Section 3.2, reward construction for General Data first establishes a candidate-independent erase contract, then evaluates each candidate against this contract and applies the removal-aware reward gate (Figure 6). GPT-5.6 Sol serves as the VLM evaluator. Reference-guided adaptations for Poster, Watermark, and Visual Text are described below.

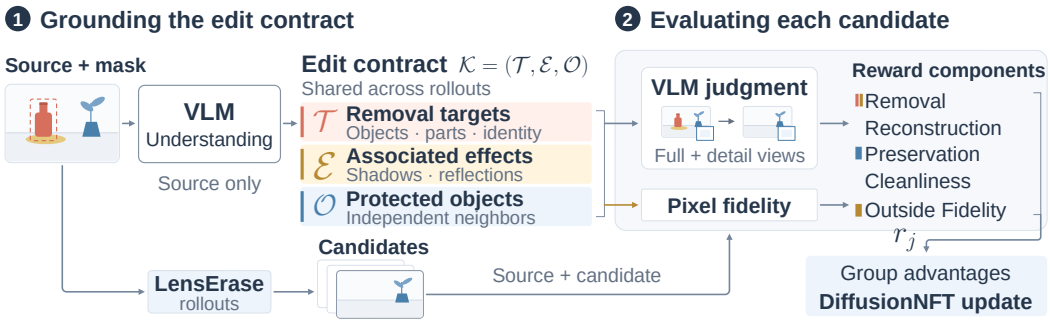

Figure 6: **Contract-guided reward construction for General Data.** Source analysis establishes an erase contract shared across rollouts. Each candidate is evaluated against this fixed contract using semantic judgments and pixel fidelity. The five reward components are combined and passed through the removal-aware gate in Equation 6; the resulting rewards define group advantages for DiffusionNFT.

Establishing the erase contract. Before observing any candidate, the VLM analyzes the source image, selection mask, and mask-centered context to construct an erase contract $\mathcal{K} = (\mathcal{T}, \mathcal{E}, \mathcal{O})$. Here, $\mathcal{T}$ denotes the removal targets, $\mathcal{E}$ their clearly attributable visual effects, and $\mathcal{O}$ nearby independent content that should be preserved. The same contract and detail-view geometry are shared by all candidates of the same image–mask condition.

Evaluating candidate edits. Each candidate is evaluated against the fixed contract using the source image, selection mask, candidate output, and aligned detail views. Removal assesses whether $\mathcal{T}$ and $\mathcal{E}$ have been fully removed; Reconstruction measures the plausibility and local consistency of newly revealed content; Preservation assesses $\mathcal{O}$ and other non-target content; and Cleanliness captures residual fragments, unwanted additions, and technical artifacts. Outside Fidelity is computed separately from source–candidate PSNR over regions expected to remain unchanged.

Reward aggregation. VLM criteria are scored on a 0–4 scale and normalized to $[0, 1]$, while computed fidelity scores are already normalized. For candidate j, let $s_{j,p,k}$ denote a normalized subscore and let $\mathcal{A}_{j,p}$ contain the enabled, applicable subcriteria for component p. We aggregate them as

$$P_{j,p} = \sum_{k \in \mathcal{A}_{j,p}} \widetilde{w}_{j,p,k} s_{j,p,k}, \tag{17}$$

where the weights are renormalized over applicable subcriteria. If an entire component is inapplicable, the remaining component weights are likewise renormalized. With the resulting weights $\widetilde{\lambda}_{j,p}$, the base reward is $b_j = \sum_p \widetilde{\lambda}_{j,p} P_{j,p}$. The final reward follows the removal-aware gate in Equation 6.

Outside Fidelity. For General Data, the valid pixel region for candidate j is

$$\Omega_j = \mathcal{V} \setminus (M_{\text{dil}} \cup B_{\mathcal{E}} \cup M_{\Delta,j}), \tag{18}$$

where $\mathcal{V}$ excludes padding, M_{dil} is a mildly dilated selection, and $B_{\mathcal{E}}$ is the union of expanded source-stage effect boxes. $M_{\Delta,j}$ further excludes size- and strength-qualified source–candidate residual components, including a gamma-0.75 pass for weak changes in dark regions. These residual components may be spatially detached from the mask, as associated reflections or shadows can extend beyond it. If this supplemental exclusion would cover more than 45% of the valid canvas, it is discarded while the selection and declared effect boxes are retained. Outside Fidelity is used only when at least 1,024 pixels and 8% of the valid canvas remain. These exclusions define the support of the pixel-fidelity metric; semantic Preservation is still evaluated over the full image.

For an applicable PSNR measurement p_j in decibels, we map the interval 20–40 dB to $[0, 1]$ with a clipped smoothstep:

$$u_j = \text{clip}\left(\frac{p_j - 20}{20}, 0, 1\right), \qquad s_j^{\text{abs}} = u_j^2(3 - 2u_j). \tag{19}$$

Table 8: Reward weights before applicability renormalization and removal-aware gating. The last column summarizes the General Data criteria. Poster, Watermark, and Visual Text share component weights, with reference-specific subcriteria described below.

Component	General Data	Poster / Watermark / Visual Text	General Data criteria
Removal	0.30	0.20	Target body absence (0.35), parts/identity absence (0.30), associated-effect removal (0.35)
Reconstruction	0.15	0.20	Semantic plausibility, structural continuity, and appearance/material consistency
Preservation	0.15	0.15	Independent-object preservation (0.60), background/scene-structure preservation (0.40)
Outside Fidelity	0.30	0.25	Outside-region PSNR score (1.00)
Cleanliness	0.10	0.20	Absence of residual fragments, unwanted additions (objects, duplicates, text, logos, or watermarks), and technical artifacts

We combine this absolute quality anchor with relative differences among valid candidates of the same condition:

$$s_j^{\mathrm{PSNR}} = \tfrac{1}{2} s_j^{\mathrm{abs}} + \tfrac{1}{2} \frac{p_j - p_{\min}}{p_{\max} - p_{\min}}. \tag{20}$$

The relative term is used only when at least two valid measurements are available and $p_{\max} - p_{\min} > 10^{-6}$ dB; otherwise the score remains s_j^{abs}.

Reference-guided domains. Poster, Watermark, and Visual Text additionally use their clean or weak references. Their component weights are 0.20 for Removal, 0.20 for Reconstruction, 0.15 for Preservation, 0.25 for Outside Fidelity, and 0.20 for Cleanliness. Removal combines a deterministic reference-based score (0.60) with the VLM target-absence score (0.40). The deterministic score measures both progress toward the reference and suppression of the original source–reference residual over changed pixels. For Watermark, a spatial low-tail term additionally checks whether repeated overlay elements remain.

For Poster and Watermark, Reconstruction combines masked reference fidelity and visual recovery quality with weights 0.70 and 0.30. For Visual Text, whose weak reference may not represent a unique valid reconstruction, the corresponding weights are 0.40 and 0.60. Visual Text Preservation additionally balances retention of unselected text and scene content with the absence of newly introduced or replacement text. Cleanliness assesses residual fragments, unwanted additions, and technical artifacts in all three reference-guided domains. The PSNR blending in Equation 20 applies to Outside Fidelity in every domain and to masked reference fidelity for Poster and Watermark; the inside text-reference score remains absolute.

Removal-aware reward gating. After component aggregation, the candidate-specific form of the removal-aware gate in Equation 6 is

$$n_{0,j} = \sum_{k \in \mathcal{A}_{j,\mathrm{Removal}}} \mathbb{1}[s_{j,\mathrm{Removal},k} = 0], \qquad r_j = 0.5^{\,n_{0,j}} b_j.$$

Each zero-valued applicable Removal subcriterion therefore contributes a factor of 0.5, so multiple removal failures compound without hard-rejecting a candidate solely because one subcriterion is zero. The same rule is applied across all data domains.

E.2.1 Reward Prompt Excerpts

The following excerpts summarize the principal semantic instructions used by the training reward evaluator. They illustrate the edit semantics and domain-specific judgments underlying Section 3.2; serialization details and output schemas are omitted. LensErase-Bench uses the separate reference-free evaluation protocol in Section 2.3.

General Data: source analysis. The mask is treated as object-selection evidence rather than an exact deletion boundary. If most of a recognizable object's visible main body is selected, the complete object—including its parts, attachments, and visible extensions—is considered a removal target. Clearly attributable effects such as cast shadows, reflections, wakes, smoke, glow, or contact traces are included in the intended removal.

General Data: candidate assessment. All selected targets are assessed jointly, and a clear failure on one target cannot be compensated by successful removal of the others. Independent non-target objects are evaluated for both presence and structural integrity, including objects not explicitly listed in the source-stage contract.

Poster. Evaluation follows the selected layer semantics and the corresponding removed reference rather than generic inpainting expectations. Foreground content may therefore remain visible inside the selected region after a back or middle layer is removed.

Watermark. All selected repeated watermark instances belong to the removal target, while text, logos, symbols, signatures, QR codes, and graphics present in the clean reference are treated as protected content.

Visual Text. Because the weak reference may not be unique, a plausible clean reconstruction is not penalized solely for differing from it. Evaluation instead focuses on residual target text, reconstruction defects, newly introduced text, and damage to content that should remain.

E.3 DIFFUSIONNFT IMPLEMENTATION

Group-relative advantages. For condition group g, let $\bar{r}_g$ and σ_g denote the mean and standard deviation of its valid candidate rewards, and let $\sigma_{\mathcal{B}}$ denote the standard deviation over all valid rewards in the current distributed collection. We compute

$$\rho_g = \min\Big(1, \frac{\sigma_g}{0.03}\Big), \qquad A_j = \mathrm{clip}\left(\rho_g \frac{r_j - \bar{r}_g}{\sigma_{\mathcal{B}} + 10^{-4}}, -5, 5\right). \tag{21}$$

A group is used for training only if at least 12 of its 16 candidates receive valid rewards and the reward spread is nonzero. Collections with $\sigma_{\mathcal{B}} < 10^{-3}$ are skipped. The factor ρ_g suppresses updates from groups with insufficient within-condition reward variation.

DiffusionNFT objective. We convert the clipped advantage into $q_j = (1 + A_j/A_{\max})/2$. Let v_θ denote the current policy, v_{old} the rollout policy, and v_{ref} the SFT reference. Following DiffusionNFT, we define

$$v_\theta^{\pm} = v_{\mathrm{old}} \pm \beta\big(v_\theta - v_{\mathrm{old}}\big). \tag{22}$$

For a generated latent $\hat{z}_j$, we sample fresh Gaussian noise at each nonterminal scheduler time and form $z_{j,t} = (1-t)\hat{z}_j + t\epsilon$, followed by $\hat{z}_{j,0}^{\pm} = z_{j,t} - t v_\theta^{\pm}$. Defining $\delta_j^{\pm} = \hat{z}_{j,0}^{\pm} - \hat{z}_j$, we use

$$d_j^{\pm} = \mathrm{sg}\big[\max\big(\mathrm{mean}\,|\delta_j^{\pm}|, 10^{-5}\big)\big], \qquad \ell_j^{\pm} = \frac{\mathrm{mean}[(\delta_j^{\pm})^2]}{d_j^{\pm}}, \tag{23}$$

where sg denotes stop-gradient. The RL objective is

$$\mathcal{L}_{\mathrm{RL}} = \mathbb{E}_{j,t,\epsilon}\left[\frac{A_{\max}}{\beta}\big(q_j \ell_j^{+} + (1-q_j)\ell_j^{-}\big) + \lambda_{\mathrm{ref}}\ \mathrm{mean}\big[(v_\theta - v_{\mathrm{ref}})^2\big]\right]. \tag{24}$$

We use $\beta = 0.5$, $A_{\max} = 5$, and $\lambda_{\mathrm{ref}} = 10^{-4}$. Gradients are propagated only through the current policy; sampled latents and the old and reference policies are treated as fixed targets. The RL reconstruction terms are averaged over the full latent and do not reuse the supervised mask weighting in Equation 3.

Policy updates. Each RL collection, corresponding to one policy update, samples conditions from a single data domain; different domains are not mixed within the same collection. We merge the SFT adapters into the backbone and use the resulting model as the fixed RL base and SFT reference. Fresh rank-64 LoRA adapters, with scaling parameter 64 and zero dropout, are attached to the same attention modules used during SFT for the current and rollout policies. We optimize the current

adapters with AdamW at a learning rate of 5×10^{-5} and clip the gradient norm to 1. We perform 100 successful policy updates in total. After the kth successful update, the rollout policy is updated as

$$\theta_{\text{old}} \leftarrow \eta_k \theta_{\text{old}} + (1 - \eta_k)\theta, \qquad \eta_k = \min(0.001k, 0.5). \tag{25}$$

The SFT reference remains fixed throughout RL training.

E.4 Inference and Sampling

All methods receive the same source images and masks and produce one output per case, with fixed per-case seeds. LensErase uses a 20-step Euler sampler throughout, pads input dimensions to multiples of 16, and crops predictions back to their original size. All outputs are restored to the source dimensions for scoring.

Sampling. At inference, we initialize $z_1 \sim \mathcal{N}(0, \mathbf{I})$ and integrate from $t = 1$ to $t = 0$ with deterministic Euler updates:

$$z_{t+\Delta t} = z_t + \Delta t \, v_\theta(z_t, t, z_{\text{src}}; M), \qquad \Delta t < 0. \tag{26}$$

Each step recomputes the velocity at the updated latent, while $z_{\text{src}} = E(I_{\text{input}})$ and M remain fixed. The Euler update has no independent learned parameters. A resolution-dependent timestep shift changes the sequence of step sizes but leaves this update rule unchanged. We use 20 Euler steps throughout. The VAE decodes only the terminal latent.

E.5 TraceFusion Parameters

Let M_{dil} denote a dilation of the input selection mask. To suppress small reconstruction biases when estimating the effective edit region, we first Gaussian-smooth the signed residual between the generated output and source image and subtract its per-channel median outside M_{dil}:

$$R = G_\sigma * (\hat{I}_{\text{output}} - I_{\text{input}}), \qquad d(x) = \left\| R(x) - \text{median}_{u \notin M_{\text{dil}}} R(u) \right\|_\infty . \tag{27}$$

Here, G_σ is a Gaussian kernel, $*$ denotes convolution, and the infinity norm takes the largest absolute response across color channels.

The detector in Section 3.3 operates on sRGB values in the 0–255 range. For an image of height H and width W, we define the resolution scale $s = \min(H, W)/512$. The Gaussian standard deviation is $\max(0.5, 5s)$ pixels with reflected boundary handling, and M_{dil} uses a dilation radius of $12s$ pixels. If M_{dil} covers the full image, the median offset in Equation 27 is set to zero.

We threshold $d(x)$ at 4 and 10 to obtain low- and high-confidence change regions. The low-threshold mask is opened with a disk of radius s, after which 8-connected components are extracted. A component is retained if it contains at least $48s^2$ pixels, including at least $8s^2$ pixels above the high threshold. Components that do not intersect M_{dil} additionally require at least 1% of their pixels to exceed the high threshold. The retained components are closed with a disk of radius $3s$ and dilated by $5s$, yielding the residual support S_{res}. The primary edit support is then

$$S_0 = M_{\text{dil}} \cup S_{\text{res}}. \tag{28}$$

To recover weak changes in dark regions, we apply the same detector after a shared gamma transform,

$$I_{\text{input},\gamma} = 255 \left(\frac{I_{\text{input}}}{255} \right)^{0.75}, \tag{29}$$

with the same transform applied to $\hat{I}_{\text{output}}$. Only changes not already covered by S_0 are considered. We extract 8-connected components from this supplemental support and retain a component only if it contains at least $8s^2$ pixels, intersects a dilation of S_0 by radius s, and lies predominantly in dark regions. Specifically, at least half of its pixels must have luma at most 96 in both the untransformed source and generated images, where luma is computed as $0.2126R + 0.7152G + 0.0722B$. Let S_γ denote the union of accepted supplemental components. The final fusion support is

$$S = S_0 \cup S_\gamma. \tag{30}$$

All integer radii and area thresholds above are rounded and clamped to at least one.

Finally, we feather the boundary of S using its normalized Euclidean distance transform. Let $d_S(x)$ be zero inside S and the distance to S outside it, with vertical and horizontal pixel spacings $1/H$ and $1/W$. The blending weight is

$$\alpha(x) = \max\left(0, 1 - \frac{d_S(x)}{0.1}\right). \tag{31}$$

If S is empty, $\alpha = 0$ everywhere. The source and generated images are converted from sRGB to linear RGB before blending and converted back to sRGB for the final output.

We use the same TraceFusion parameters for all reported experiments.

E.6 Notation Summary

Table 9 summarizes the principal notation used in Section 3 and Appendices D and E.

Table 9: Principal notation for LensErase.

Symbol	Definition
Images, masks, and positional coordinates	
I_{input}	Source image containing the selected targets.
I_{output}	Paired erase target used for supervised training.
$\hat{I}_{\text{output}}$, $\hat{I}^{\text{TF}}_{\text{output}}$	Generated image before and after TraceFusion, respectively.
M, m_i	Binary selection mask and its max-pooled value for source token i, with 1 denoting a selected region.
h_i, w_i	Height and width coordinates of token i.
$\mathbf{p}_i^{\text{cond}}$, $\mathbf{p}_i^{\text{img}}$	Source and noisy-target RoPE coordinates: (m_i, h_i, w_i) and $(2, h_i, w_i)$, respectively.
Latents and flow matching	
E	VAE encoder.
z_{src}, $z^\star$	Source latent $E(I_{\text{input}})$ and clean target latent $E(I_{\text{output}})$.
t, ϵ, $\mathbf{I}$	Noise level $t \in [0,1]$, Gaussian noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, and identity covariance.
z_t	Noisy target latent; during supervised training, $z_t = (1-t)z^\star + t\epsilon$.
v_θ, $v^\star$	Predicted velocity $v_\theta(z_t, t, z_{\text{src}}; M)$ with model parameters θ, and target velocity $v^\star = \epsilon - z^\star$.
N, i	Number of target tokens and token index, respectively.
$\tilde{M}$	Binary loss mask obtained by dilating M and downsampling to the token grid.
λ, ω_i	Masked-region loss multiplier and token weight $\omega_i = 1 + (\lambda - 1)\tilde{M}_i$.
Contract-guided rewards	
$\mathcal{K} = (\mathcal{T}, \mathcal{E}, \mathcal{O})$	Erase contract containing removal targets, associated effects, and protected objects.
$\mathcal{P}$, P_p, λ_p	Reward-component set, normalized component score, and component weight.
$\mathcal{A}_{j,p}$, $s_{j,p,k}$, $\widetilde{w}_{j,p,k}$	Applicable subcriteria, normalized subscore, and renormalized subcriterion weight for candidate j and component p.
b, n_0, r	Base reward, number of zero-valued Removal subcriteria, and gated reward $r = 0.5^{n_0} b$.
j	Candidate index, suppressed in Section 3.2 and explicit in quantities such as $P_{j,p}$, $\widetilde{\lambda}_{j,p}$, b_j, $n_{0,j}$, and r_j.
TraceFusion	
M_{dil}	Dilated selection mask used by the residual detector.
R, $d(x)$	Smoothed signed source–output residual and its median-centered magnitude at pixel x, respectively.
S_{res}	Residual support retained after component filtering and morphology.
S_0	Initial support: $M_{\text{dil}} \cup S_{\text{res}}$.
γ, S_γ	Shared power-transform exponent and the union of accepted dark-region supplemental components.
S	Final fusion support: $S_0 \cup S_\gamma$.
$d_S(x)$, $\alpha(x)$	Distance to S (zero inside), using normalized pixel spacings, and blending weight $\max(0, 1 - d_S(x)/0.1)$.
$\mathcal{C}$, $\odot$	sRGB-to-linear-RGB conversion and pixelwise multiplication.

F Complete Benchmark Results

F.1 Evaluation Protocol

All methods receive the same source images and masks and produce one output per case, with fixed per-case seeds. Clean targets are used only for scoring. Outputs are restored to the evaluation image dimensions.

CMMD (full-image resize). For CMMD (Jayasumana et al., 2024), we bicubically resize full predictions and references to 336×336 without center cropping, to retain off-center removal regions and associated effects. Within each benchmark, all methods use the same samples, preprocessing, CLIP encoder, and MMD parameters.

F.1.1 OmniPaint-Bench Mask Corrections

Among the 300 captured cases of OmniPaint-Bench (Yu et al., 2025), we correct 32 masks whose orientation is misaligned with the corresponding input image. We rotate 29 masks 90° counterclockwise and flip three horizontally; the other 268 masks are unchanged. These transformations preserve mask pixel values and foreground area without interpolation. All input images and ground-truth targets remain unchanged, and ground-truth targets are not used to generate or redraw the masks. All evaluated methods receive the same corrected masks.

F.2 Baseline Configurations

We use released checkpoints with the variants specified below. Code repositories and weights are linked for reproducibility.

ObjectClear. We use the released ObjectClear pipeline (Zhao et al., 2026)[1] with attention-guided fusion.

SmartEraser. We run the released SmartEraser controller (Jiang et al., 2025)[2].

OmniPaint. We use OmniPaint's FLUX.1-dev removal LoRA and fixed prompt embeddings (Yu et al., 2025)[3].

OmniEraser. We evaluate OmniEraser-ControlNet (Wei et al., 2025)[4] using FLUX.1-dev, the Alimama inpainting ControlNet, and the released LoRA.

OSOR. We use OSOR-FLUX-Fill Phase II (Zhou et al., 2026)[5]. Alpha-head latent blending is retained.

Attentive Eraser. We evaluate the SDXL+SIP configuration (Sun et al., 2025)[6].

PANDORA. We use the SDXL configuration of PANDORA (Vo et al., 2026)[7].

DesignEdit. We use DesignEdit's SDXL object-removal mode (Jia et al., 2025)[8] with DDIM inversion and LayerFusion.

[1] `https://github.com/zjx0101/ObjectClear`; released ObjectClear checkpoint.

[2] `https://github.com/longtaojiang/SmartEraser`; the released `smarteraser-weights.zip` archive and CLIP ViT-L/14.

[3] `https://github.com/yeates/OmniPaint`; removal weights and prompt embeddings.

[4] `https://github.com/PRIS-CV/Omnieraser`; OmniEraser-ControlNet checkpoint.

[5] `https://github.com/Zhouqm-Git/osor`; `fluxfill_phase2.pth` in the OSOR release.

[6] `https://github.com/Alibaba-VELLDEPTH/AttentiveEraser`, following the SDXL+SIP example.

[7] `https://github.com/vdkhoi20/PANDORA`, with the released SDXL configuration.

[8] `https://github.com/design-edit/DesignEdit`, using `run_remove`.

DORS. We use DORS with SDXL-Inpainting and SAM3 (Tang et al., 2026)[9].

GeoRemover. We run both GeoRemover stages (Zhu et al., 2025b)[10] on FLUX.1-Fill-dev with Video Depth Anything Large.

EntityErasure. We use EntityErasure's full amodal segmentation and completion pipeline (Zhu et al., 2025a)[11].

RORem-mixed. We use RORem's nine-channel mixed UNet (Li et al., 2025)[12] with the released fixed prompts.

F.3 Results on Public Benchmarks

Tables 10–15 expand Table 1 with all twelve baselines on each benchmark, using the same evaluation protocol and eight metrics. Red, orange, and yellow cell backgrounds mark the best, second-best, and third-best values among all thirteen methods within each benchmark. Best values are also bold; ties at the displayed precision share a color.

Table 10: Complete comparison on OBER-Test (Zhao et al., 2026).

Method	PSNR ↑	FID ↓	CMMD ↓	LPIPS ↓	SSIM ↑	DISTS ↓	CLIP-I ↑	DINO ↑
SmartEraser (Jiang et al., 2025)	26.67	94.58	0.4178	0.0762	0.9446	0.0792	0.9393	0.8423
RORem-mixed (Li et al., 2025)	25.93	61.32	0.1832	0.1122	0.8549	0.0836	0.9593	0.9186
EntityErasure (Zhu et al., 2025a)	22.75	82.02	0.2933	0.1330	0.7263	0.1037	0.9376	0.8866
Attentive Eraser (Sun et al., 2025)	26.22	59.96	0.2048	0.1123	0.8497	0.0889	0.9605	0.9228
DesignEdit (Jia et al., 2025)	26.39	71.25	0.2052	0.1302	0.8585	0.0955	0.9485	0.8997
OmniEraser (Wei et al., 2025)	25.52	85.73	0.2131	0.0996	0.9141	0.0896	0.9457	0.8698
OmniPaint (Yu et al., 2025)	28.79	23.68	0.0829	0.0549	0.8575	0.0427	0.9836	0.9803
GeoRemover (Zhu et al., 2025b)	24.66	56.24	0.4277	0.1423	0.7622	0.1128	0.9305	0.9292
PANDORA (Vo et al., 2026)	25.63	79.21	0.3854	0.1482	0.8507	0.1056	0.9397	0.8723
DORS (Tang et al., 2026)	24.13	88.09	0.4874	0.1250	0.7492	0.0966	0.9371	0.8714
ObjectClear (Zhao et al., 2026)	33.05	19.16	0.0132	0.0338	0.9353	0.0217	0.9894	**0.9839**
OSOR (Zhou et al., 2026)	30.14	27.82	0.0674	0.0524	0.8748	0.0377	0.9827	0.9693
LensErase	**34.96**	**16.97**	**0.0073**	**0.0256**	**0.9543**	**0.0176**	**0.9930**	**0.9839**

[9] https://github.com/httang1224/DORS, with the released SDXL-Inpainting and SAM3 weights.

[10] https://github.com/buxiangzhiren/GeoRemover; released stage-1 and stage-2 checkpoints.

[11] https://github.com/zyxunh/entity_erasure; released segmentation and completion checkpoints.

[12] https://github.com/leeruibin/RORem; authors' mixed-checkpoint release.

Table 11: Complete comparison on RORD-Val-343 (Sagong et al., 2022).

Method	PSNR ↑	FID ↓	CMMD ↓	LPIPS ↓	SSIM ↑	DISTS ↓	CLIP-I ↑	DINO ↑
SmartEraser (Jiang et al., 2025)	24.89	35.41	0.1104	0.1658	0.8472	0.0950	0.9644	0.9349
RORem-mixed (Li et al., 2025)	23.16	35.91	0.0983	0.1906	0.6893	0.0990	0.9482	0.9316
EntityErasure (Zhu et al., 2025a)	21.25	50.71	0.2829	0.3340	0.6142	0.1461	0.9400	0.9082
Attentive Eraser (Sun et al., 2025)	22.57	39.19	0.0948	0.1743	0.6896	0.0879	0.9501	0.9189
DesignEdit (Jia et al., 2025)	23.28	49.55	0.1603	0.1938	0.7268	0.1046	0.9304	0.8837
OmniEraser (Wei et al., 2025)	24.46	35.78	0.0426	0.1189	0.8380	0.0690	0.9580	0.9262
OmniPaint (Yu et al., 2025)	23.64	27.45	0.0334	0.1119	0.7823	0.0623	0.9672	0.9541
GeoRemover (Zhu et al., 2025b)	24.62	28.13	0.1716	0.1185	0.7659	0.0708	0.9640	0.9536
PANDORA (Vo et al., 2026)	21.61	68.85	0.5348	0.2274	0.7106	0.1211	0.8886	0.7928
DORS (Tang et al., 2026)	21.84	50.22	0.3032	0.2902	0.6253	0.1412	0.9330	0.8862
ObjectClear (Zhao et al., 2026)	26.21	25.42	0.0511	0.1163	0.8577	0.0657	0.9704	0.9566
OSOR (Zhou et al., 2026)	25.67	26.33	0.0390	0.1037	0.8138	0.0542	0.9663	0.9475
LensErase	**28.23**	**20.53**	**0.0272**	**0.0601**	**0.9053**	**0.0357**	**0.9715**	**0.9568**

Table 12: Complete comparison on RemovalBench (Wei et al., 2025).

Method	PSNR ↑	FID ↓	CMMD ↓	LPIPS ↓	SSIM ↑	DISTS ↓	CLIP-I ↑	DINO ↑
SmartEraser (Jiang et al., 2025)	23.49	151.88	0.4359	0.3308	0.7695	0.1890	0.9339	0.8239
RORem-mixed (Li et al., 2025)	23.51	90.95	0.2044	0.2262	0.7149	0.1175	0.9575	0.9098
EntityErasure (Zhu et al., 2025a)	20.36	161.80	0.3525	0.4287	0.6664	0.2288	0.9116	0.7918
Attentive Eraser (Sun et al., 2025)	23.78	80.49	0.1666	0.1941	0.7095	0.0969	0.9646	0.9263
DesignEdit (Jia et al., 2025)	23.96	108.46	0.2758	0.2343	0.7200	0.1357	0.9457	0.8676
OmniEraser (Wei et al., 2025)	22.33	147.90	0.3822	0.2150	0.7539	0.1285	0.9213	0.7990
OmniPaint (Yu et al., 2025)	24.57	43.25	0.0265	0.2103	0.7535	0.0998	0.9821	0.9663
GeoRemover (Zhu et al., 2025b)	25.35	42.05	0.2310	0.2316	0.7457	0.1181	0.9777	0.9698
PANDORA (Vo et al., 2026)	23.84	98.21	0.4140	0.2389	0.7167	0.1413	0.9359	0.8427
DORS (Tang et al., 2026)	23.07	141.61	0.4622	0.3755	0.6987	0.1935	0.9419	0.8629
ObjectClear (Zhao et al., 2026)	25.99	46.61	0.0248	0.2853	0.7794	0.1428	0.9853	0.9718
OSOR (Zhou et al., 2026)	25.80	43.89	0.0275	0.1935	0.7656	0.0882	0.9838	0.9597
LensErase	**28.16**	**33.67**	**0.0116**	**0.1125**	**0.8306**	**0.0586**	**0.9874**	**0.9737**

Table 13: Complete comparison on OmniPaint-Bench (Yu et al., 2025) (captured subset), excluding RORD-derived cases.

Method	PSNR ↑	FID ↓	CMMD ↓	LPIPS ↓	SSIM ↑	DISTS ↓	CLIP-I ↑	DINO ↑
SmartEraser (Jiang et al., 2025)	22.05	94.79	0.2854	0.2787	0.8121	0.1460	0.9189	0.8610
RORem-mixed (Li et al., 2025)	23.14	54.66	0.1398	0.1994	0.7730	0.1072	0.9433	0.9226
EntityErasure (Zhu et al., 2025a)	20.44	96.41	0.3472	0.3033	0.7262	0.1583	0.9068	0.8579
Attentive Eraser (Sun et al., 2025)	23.31	50.60	0.1324	0.1642	0.7865	0.0719	0.9452	0.9322
DesignEdit (Jia et al., 2025)	23.30	68.54	0.2245	0.1838	0.7859	0.1033	0.9261	0.8838
OmniEraser (Wei et al., 2025)	21.73	93.71	0.2645	0.1767	0.8123	0.0904	0.9059	0.8212
OmniPaint (Yu et al., 2025)	23.42	40.65	0.0743	0.1484	0.8193	0.0644	0.9562	0.9423
GeoRemover (Zhu et al., 2025b)	24.87	38.54	0.1376	0.1575	0.8173	0.0704	0.9524	0.9494
PANDORA (Vo et al., 2026)	23.26	83.41	0.3606	0.1940	0.7835	0.1122	0.9130	0.8538
DORS (Tang et al., 2026)	22.20	79.22	0.3511	0.2799	0.7479	0.1389	0.9246	0.8949
ObjectClear (Zhao et al., 2026)	24.93	48.91	0.0981	0.2233	0.8178	0.1054	0.9613	0.9549
OSOR (Zhou et al., 2026)	26.02	37.52	0.0687	0.1336	0.8421	0.0566	0.9617	0.9506
LensErase	**26.22**	**33.29**	**0.0245**	**0.1257**	**0.8521**	**0.0544**	**0.9660**	**0.9595**

Table 14: Complete comparison on AnimeEraseBench (Zhou et al., 2026).

Method	PSNR ↑	FID ↓	CMMD ↓	LPIPS ↓	SSIM ↑	DISTS ↓	CLIP-I ↑	DINO ↑
SmartEraser (Jiang et al., 2025)	23.90	55.01	0.0451	0.2116	0.8252	0.1174	0.9837	0.9577
RORem-mixed (Li et al., 2025)	24.42	33.68	0.0261	0.1305	0.7926	0.0654	0.9875	0.9756
EntityErasure (Zhu et al., 2025a)	20.26	83.39	0.0906	0.3574	0.6143	0.1776	0.9711	0.9282
Attentive Eraser (Sun et al., 2025)	22.60	37.77	0.0315	0.1329	0.7321	0.0629	0.9852	0.9699
DesignEdit (Jia et al., 2025)	24.10	49.33	0.0666	0.1582	0.7861	0.0846	0.9800	0.9519
OmniEraser (Wei et al., 2025)	24.05	49.39	0.0383	0.1155	0.8569	0.0617	0.9772	0.9273
OmniPaint (Yu et al., 2025)	25.58	27.76	0.0138	0.1195	0.8581	0.0636	0.9874	0.9764
GeoRemover (Zhu et al., 2025b)	24.80	30.72	0.0533	0.1361	0.8247	0.0734	0.9828	0.9766
PANDORA (Vo et al., 2026)	23.38	54.81	0.0968	0.1621	0.7787	0.0911	0.9716	0.9274
DORS (Tang et al., 2026)	21.59	73.28	0.0925	0.3123	0.6459	0.1548	0.9742	0.9460
ObjectClear (Zhao et al., 2026)	25.42	36.76	0.0121	0.1791	0.8417	0.0978	0.9914	**0.9821**
OSOR (Zhou et al., 2026)	27.96	24.45	**0.0094**	0.0967	0.8870	0.0475	0.9900	0.9751
LensErase	**30.50**	**17.20**	**0.0094**	**0.0527**	**0.9398**	**0.0159**	**0.9920**	0.9798

Table 15: Complete comparison on TextEraseBench (Zhou et al., 2026).

Method	PSNR ↑	FID ↓	CMMD ↓	LPIPS ↓	SSIM ↑	DISTS ↓	CLIP-I ↑	DINO ↑
SmartEraser (Jiang et al., 2025)	28.85	47.96	0.0608	0.1686	0.8885	0.1049	0.9430	0.9471
RORem-mixed (Li et al., 2025)	29.50	24.83	0.0178	0.1034	0.8292	0.0584	0.9815	0.9744
EntityErasure (Zhu et al., 2025a)	24.98	53.79	0.1175	0.2653	0.7307	0.1502	0.9427	0.9414
Attentive Eraser (Sun et al., 2025)	26.31	34.70	0.0228	0.1134	0.7956	0.0563	0.9738	0.9613
DesignEdit (Jia et al., 2025)	27.61	49.23	0.1002	0.1391	0.8289	0.0883	0.9578	0.9247
OmniEraser (Wei et al., 2025)	30.70	21.97	0.0195	0.0763	0.9005	0.0435	0.9812	0.9758
OmniPaint (Yu et al., 2025)	28.14	27.78	0.0188	0.1085	0.8798	0.0607	0.9772	0.9622
GeoRemover (Zhu et al., 2025b)	26.26	60.36	0.1997	0.1346	0.8613	0.0804	0.8907	0.9169
PANDORA (Vo et al., 2026)	26.39	53.93	0.1185	0.1412	0.8202	0.0914	0.9463	0.9106
DORS (Tang et al., 2026)	25.75	53.04	0.0676	0.2467	0.7484	0.1332	0.9612	0.9343
ObjectClear (Zhao et al., 2026)	30.37	27.81	0.0193	0.1329	0.8976	0.0822	0.9774	0.9749
OSOR (Zhou et al., 2026)	32.04	18.52	0.0093	0.0728	0.9126	0.0412	0.9843	0.9771
LensErase	**36.88**	**13.51**	**0.0091**	**0.0339**	**0.9654**	**0.0135**	**0.9906**	**0.9822**

F.4 Results on LensErase-Bench

Table 16 reports mean scores over LensErase-Bench. PANDORA uses 499 cases because one output was consistently rejected by GPT-5.6 Sol's safety filter; all other methods use 500. HiFi Removal denotes the percentage of cases achieving complete removal (Removal $= 4$) while maintaining Inpainting, Preservation, and Cleanliness scores of at least 3.

Table 16: VLM comparison on LensErase-Bench.

Method	Removal ↑	Inpainting ↑	Preservation ↑	Cleanliness ↑	Overall ↑	HiFi Removal (%) ↑
SmartEraser (Jiang et al., 2025)	23.40	24.15	66.25	57.75	38.99	3.60
RORem-mixed (Li et al., 2025)	73.95	59.70	74.05	67.10	69.75	27.40
EntityErasure (Zhu et al., 2025a)	59.65	48.50	48.45	60.20	55.29	4.80
Attentive Eraser (Sun et al., 2025)	78.40	52.25	61.50	55.05	65.12	20.60
DesignEdit (Jia et al., 2025)	76.95	60.25	63.85	62.95	68.19	25.00
OmniEraser (Wei et al., 2025)	55.65	47.95	77.55	57.65	58.89	22.60
OmniPaint (Yu et al., 2025)	69.20	66.15	80.20	83.20	73.59	42.80
GeoRemover (Zhu et al., 2025b)	54.30	49.85	77.05	78.00	62.70	25.60
PANDORA (Vo et al., 2026)	41.93	39.28	58.17	49.50	46.16	7.21
DORS (Tang et al., 2026)	55.20	27.40	39.10	32.60	41.90	0.60
ObjectClear (Zhao et al., 2026)	73.95	61.45	73.80	75.20	71.67	28.40
OSOR (Zhou et al., 2026)	73.85	60.45	75.55	73.55	71.45	32.60
LensErase	**95.15**	**92.40**	**96.35**	**97.05**	**95.22**	**82.80**

G Reward Component Ablations

We examine five variants of the reward in Section 3.2, each omitting one component: Removal, Reconstruction, Preservation, Outside Fidelity, or Cleanliness. The remaining component weights are renormalized. The variant without Removal also disables the removal-aware gate; the other four retain it. All variants start from the same SFT checkpoint, with optimization details given in Appendix E.3. Table 17 reports the VLM metrics together with PSNR and LPIPS.

Table 17: **Reward-component ablations.** PSNR and LPIPS are equal-weight means over the six public benchmarks.

Model	LensErase-Bench						Public Benchmarks	
	Removal ↑	Inpainting ↑	Preservation ↑	Cleanliness ↑	Overall ↑	HiFi Removal ↑	PSNR ↑	LPIPS ↓
RL-Raw (full reward)	94.95	92.65	93.35	96.15	94.41	81.60	28.70	0.0816
w/o Removal + gate	43.35	46.10	98.70	98.10	65.92	38.80	21.85	0.1446
w/o Reconstruction	90.85	91.40	96.10	95.70	92.98	78.00	28.32	0.0791
w/o Preservation	94.15	91.50	89.20	96.00	93.00	74.60	28.01	0.0817
w/o Outside Fidelity	92.90	91.70	94.40	96.40	93.66	78.00	26.69	0.0888
w/o Cleanliness	90.90	90.95	96.25	95.95	92.99	78.40	28.48	0.0803

Removal component and gate. Jointly removing the Removal component and the removal-aware gate causes the largest degradation, with Removal dropping from 94.95 to 43.35 and HiFi Removal from 81.60% to 38.80%.

Complementary reward components. The remaining components provide complementary benefits. Removing Preservation yields the lowest HiFi Removal among these variants (74.60%), while removing Outside Fidelity causes the largest PSNR drop, from 28.70 to 26.69 dB. Removing Reconstruction or Cleanliness also reduces HiFi Removal to 78.00% and 78.40%, respectively. Overall, the full reward achieves the best HiFi Removal.

H LIMITATIONS AND CONCLUSION

Despite its broad coverage, LensErase still has several limitations. First, scaling training to ultra-high resolutions remains challenging. Our large-scale data construction relies in part on existing general-purpose image editing models, which remain less reliable for ultra-high-resolution editing, particularly at resolutions such as 4K. This makes it difficult to construct sufficiently large and consistent 4K-scale training data. Future work could explore more reliable high-resolution data construction and more challenging erase scenarios. Overall, LensErase establishes a scalable framework for general-purpose mask-based erasing by combining large-scale data, generative-prior adaptation, reward-based post-training, and high-fidelity post-processing. We hope it provides a useful foundation for future erase-related models and visual editing systems.

I Qualitative Comparisons

I.1 Comparisons with Mask-based GPT-Image-2.5-Sunburst (OpenAI, 2026b) and Flux2-Erase (Black Forest Labs, 2026)

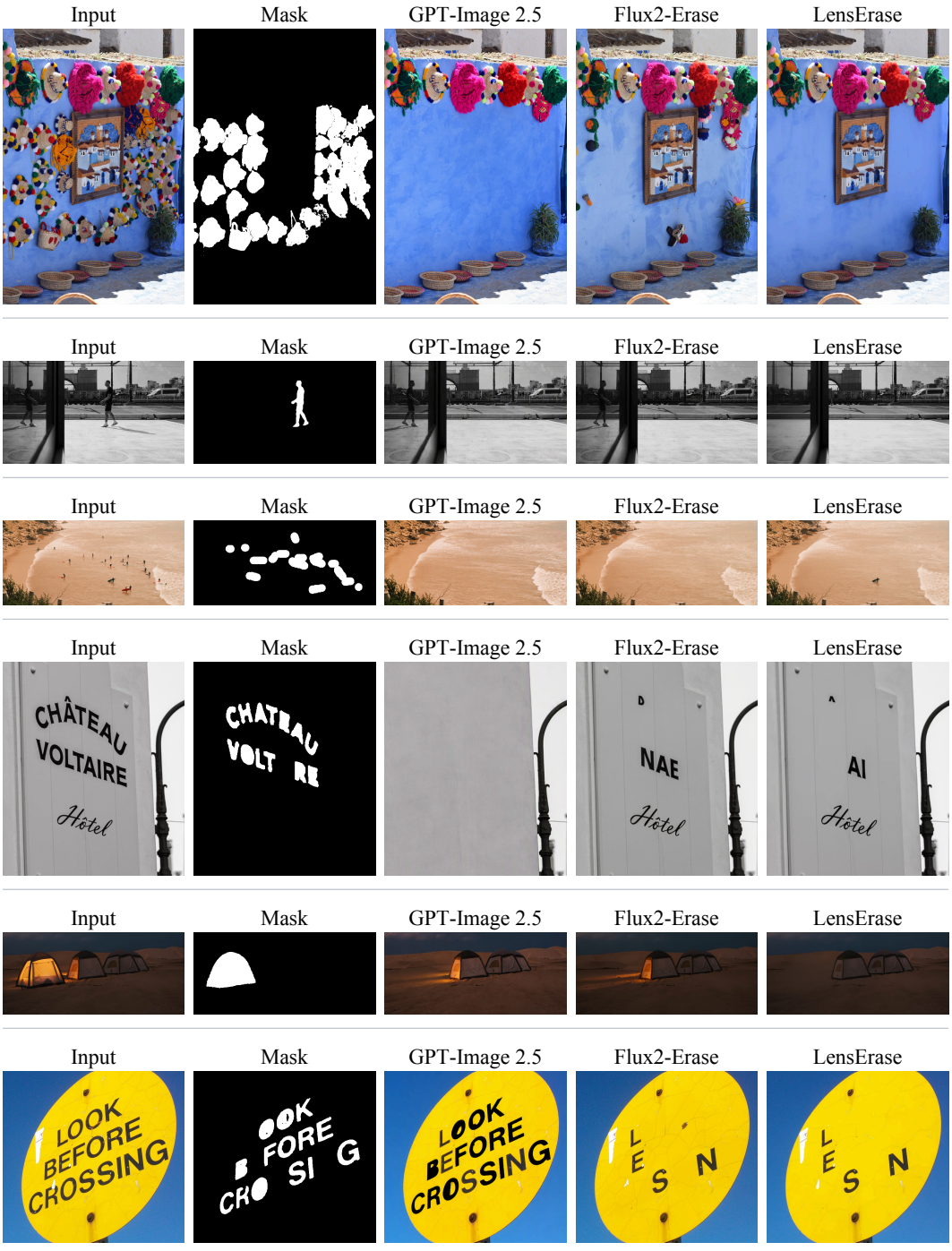

Figure 7: Qualitative comparisons with GPT-Image 2.5 and Flux2-Erase.

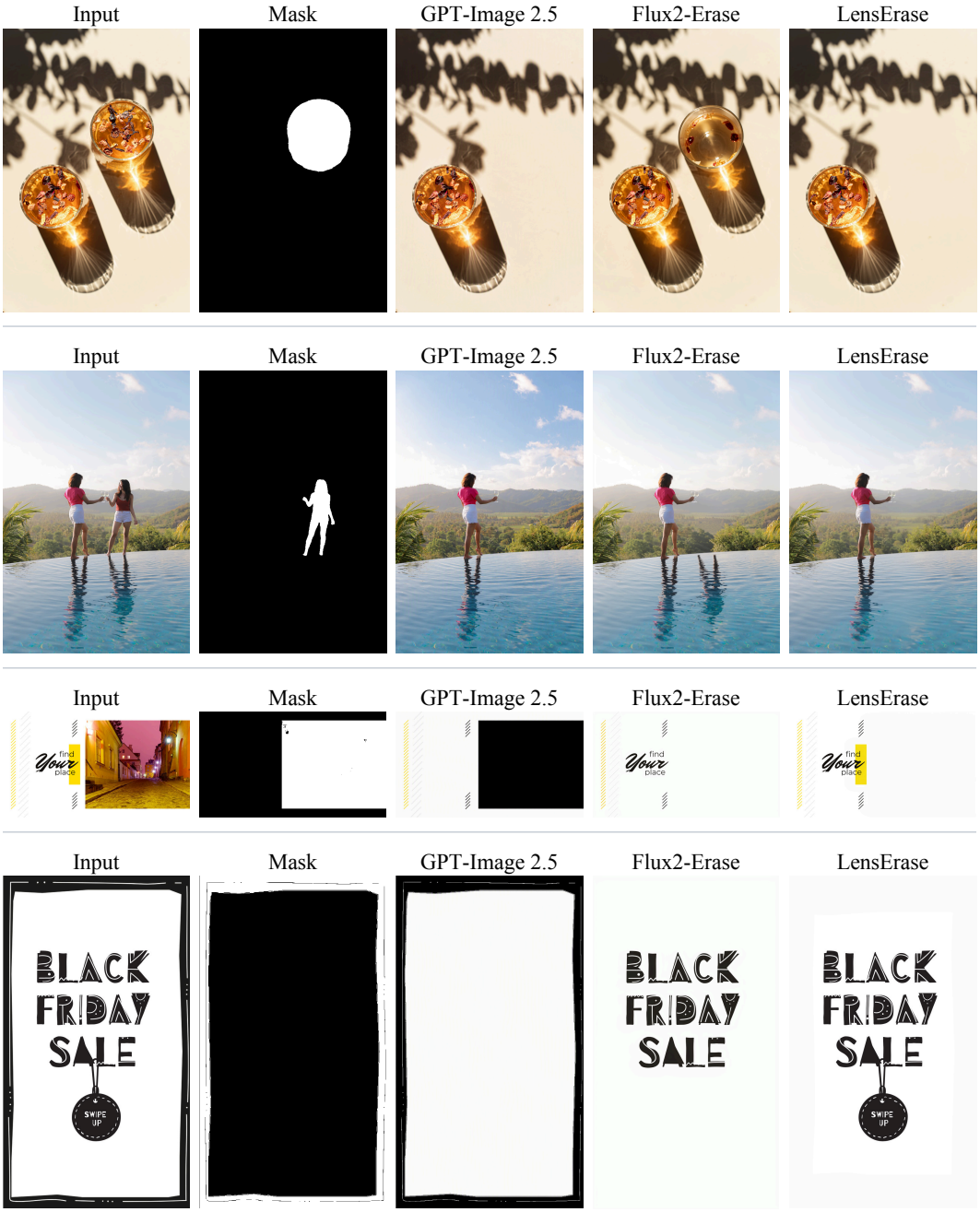

Figure 8: Qualitative comparisons with GPT-Image 2.5 and Flux2-Erase.

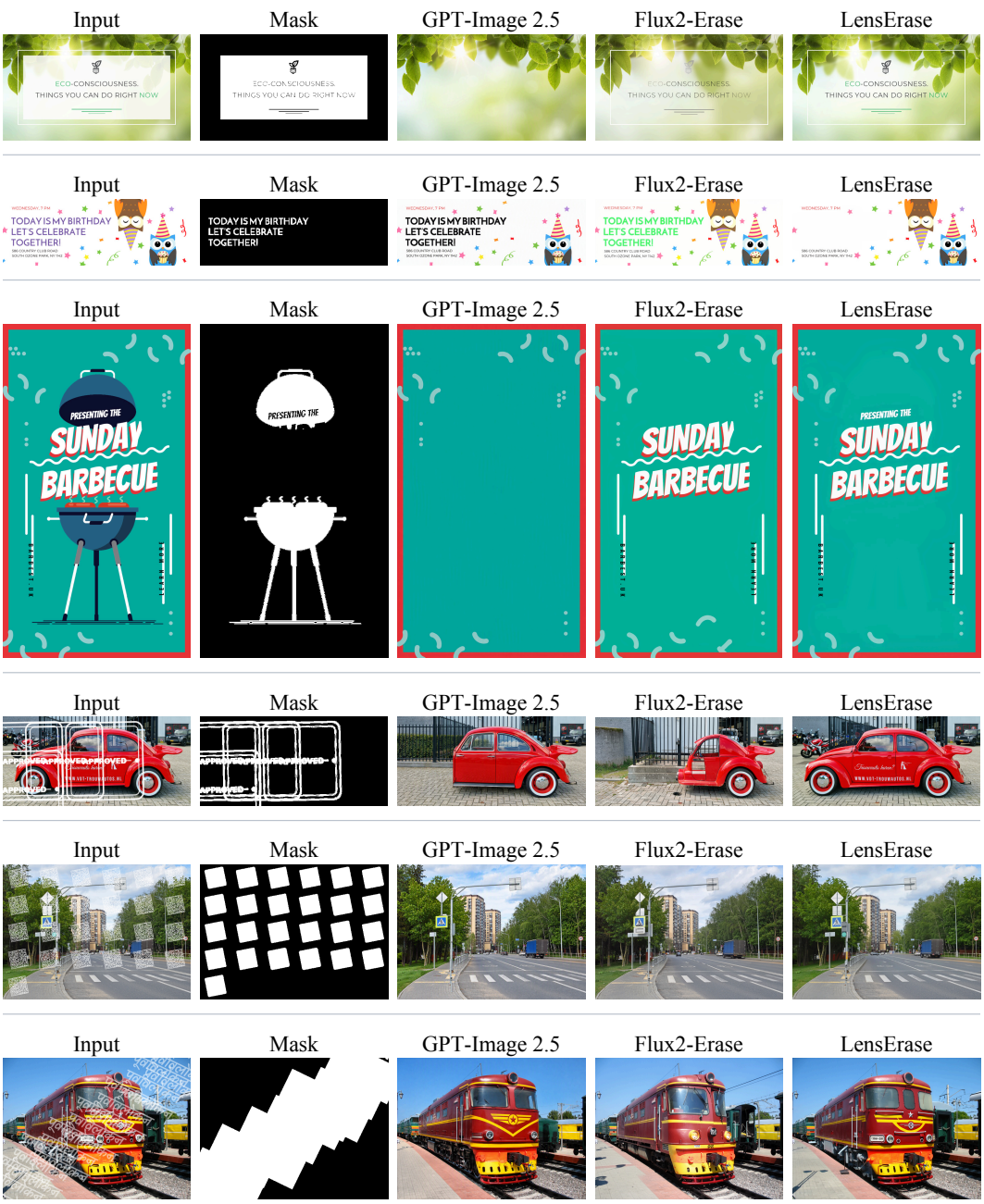

Figure 9: Qualitative comparisons with GPT-Image 2.5 and Flux2-Erase.

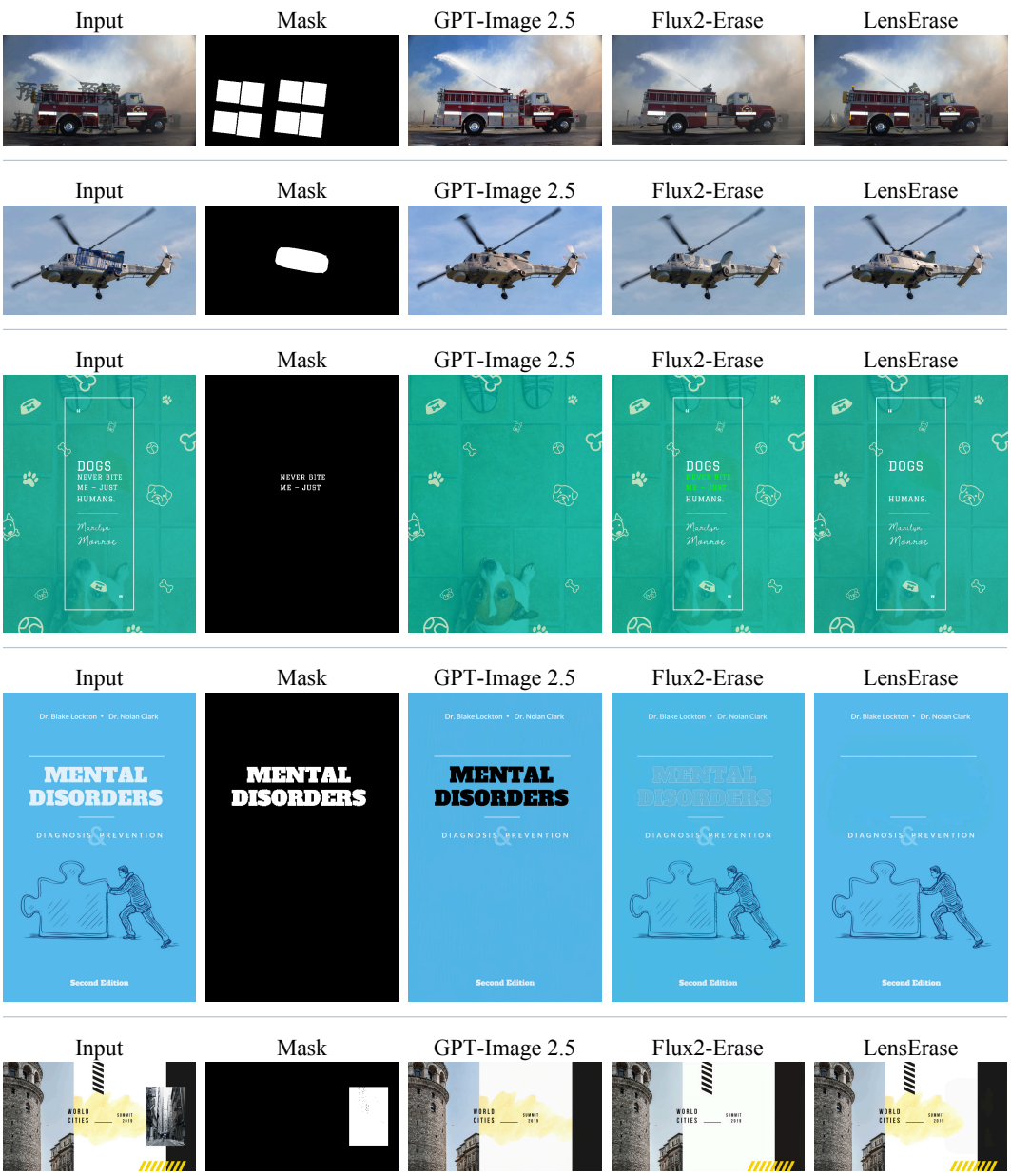

Figure 10: Qualitative comparisons with GPT-Image 2.5 and Flux2-Erase.

I.2 Comparisons with Object-Removal Baselines

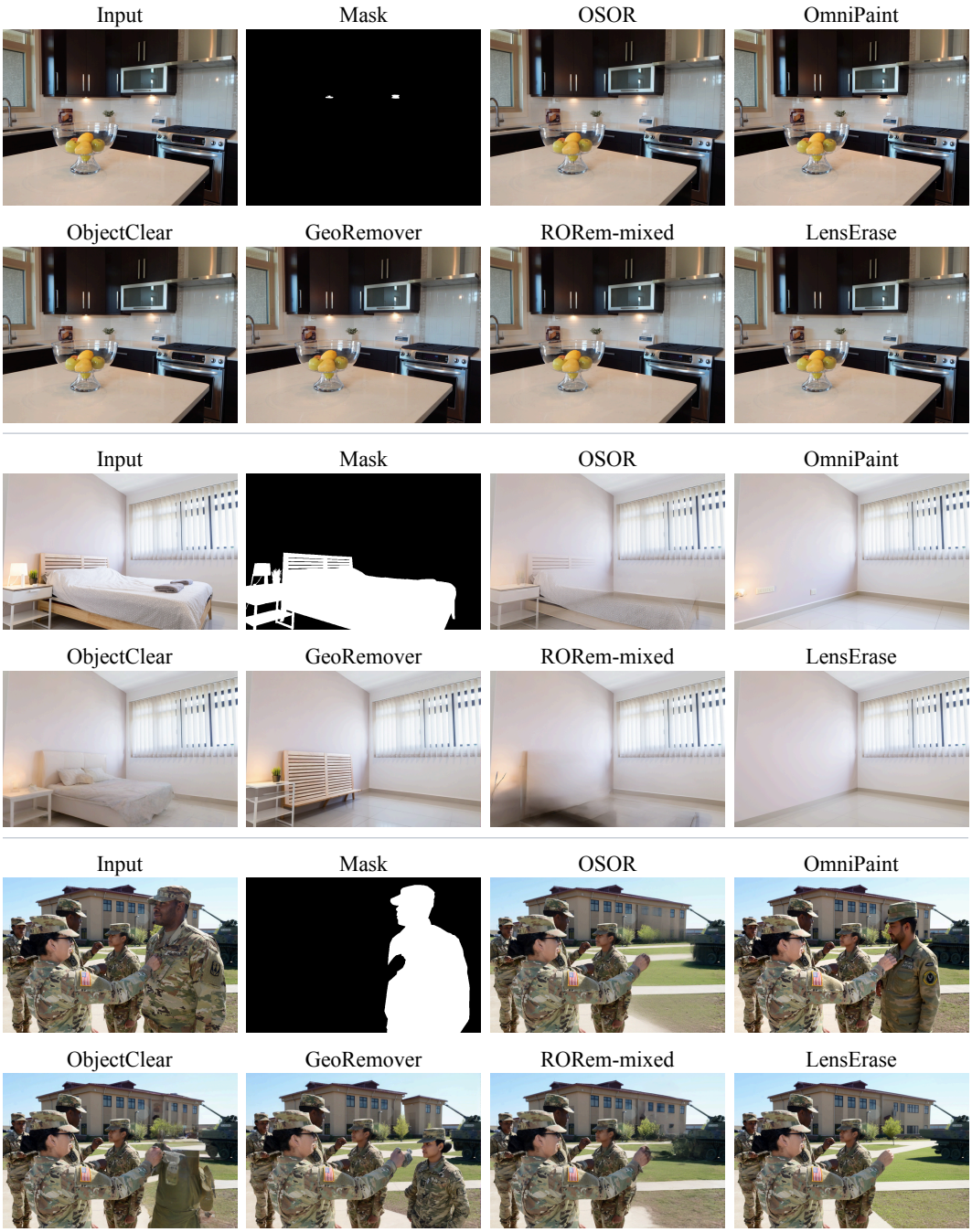

Figure 11: Qualitative comparisons on the General subset.

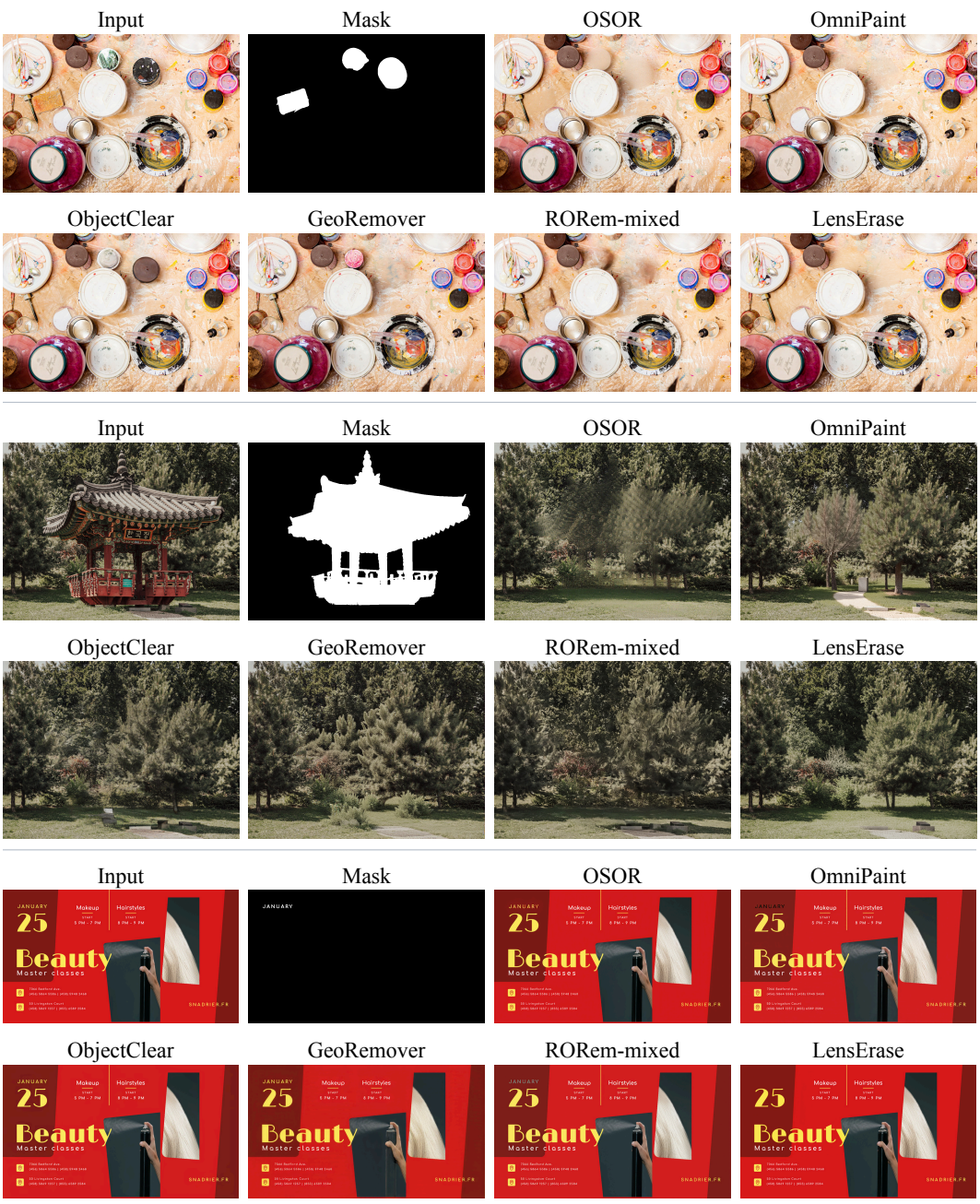

Figure 12: Qualitative comparisons on General images and posters.



Figure 13: Qualitative comparisons on posters.

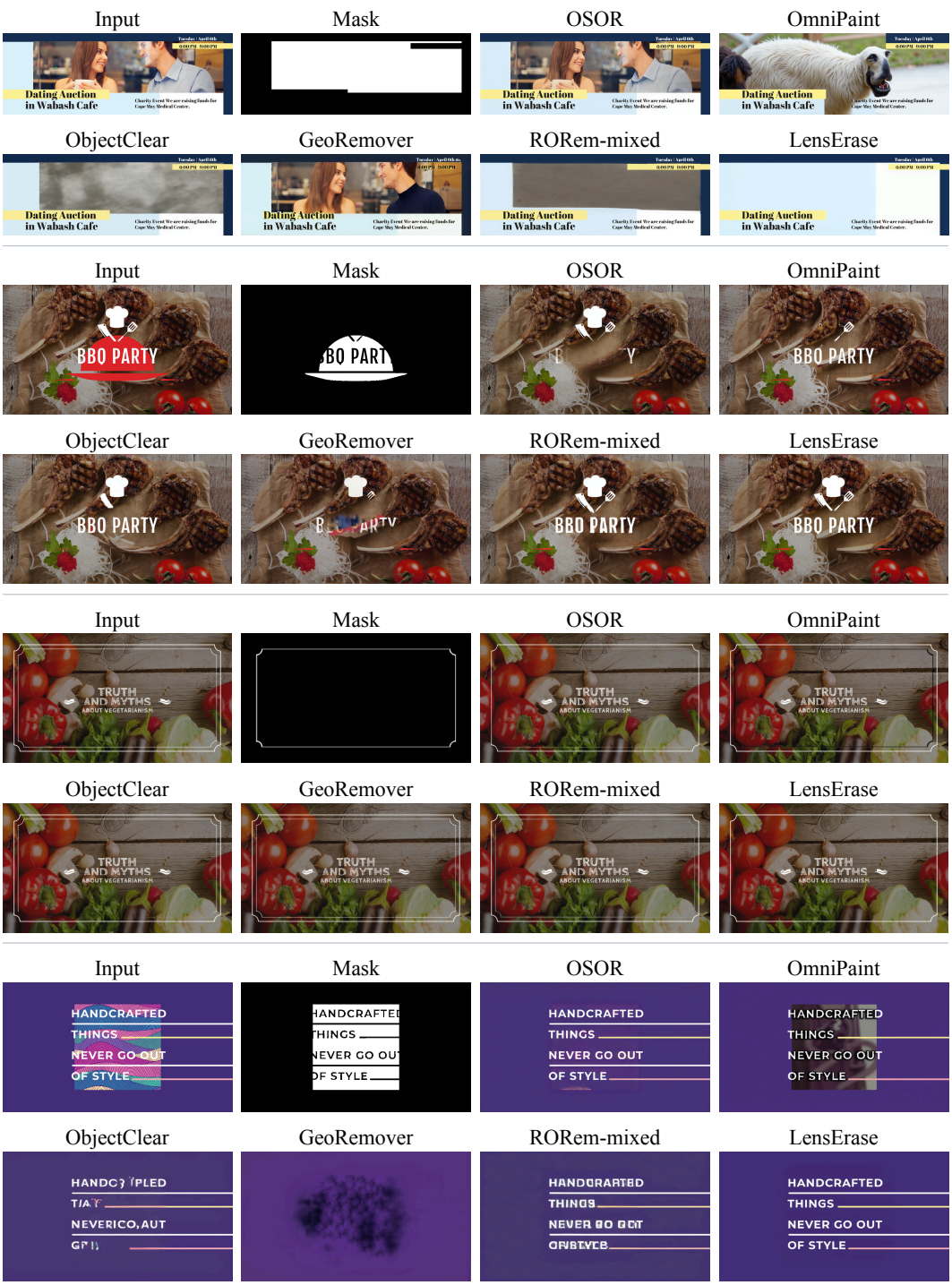

Figure 14: Qualitative comparisons on posters.

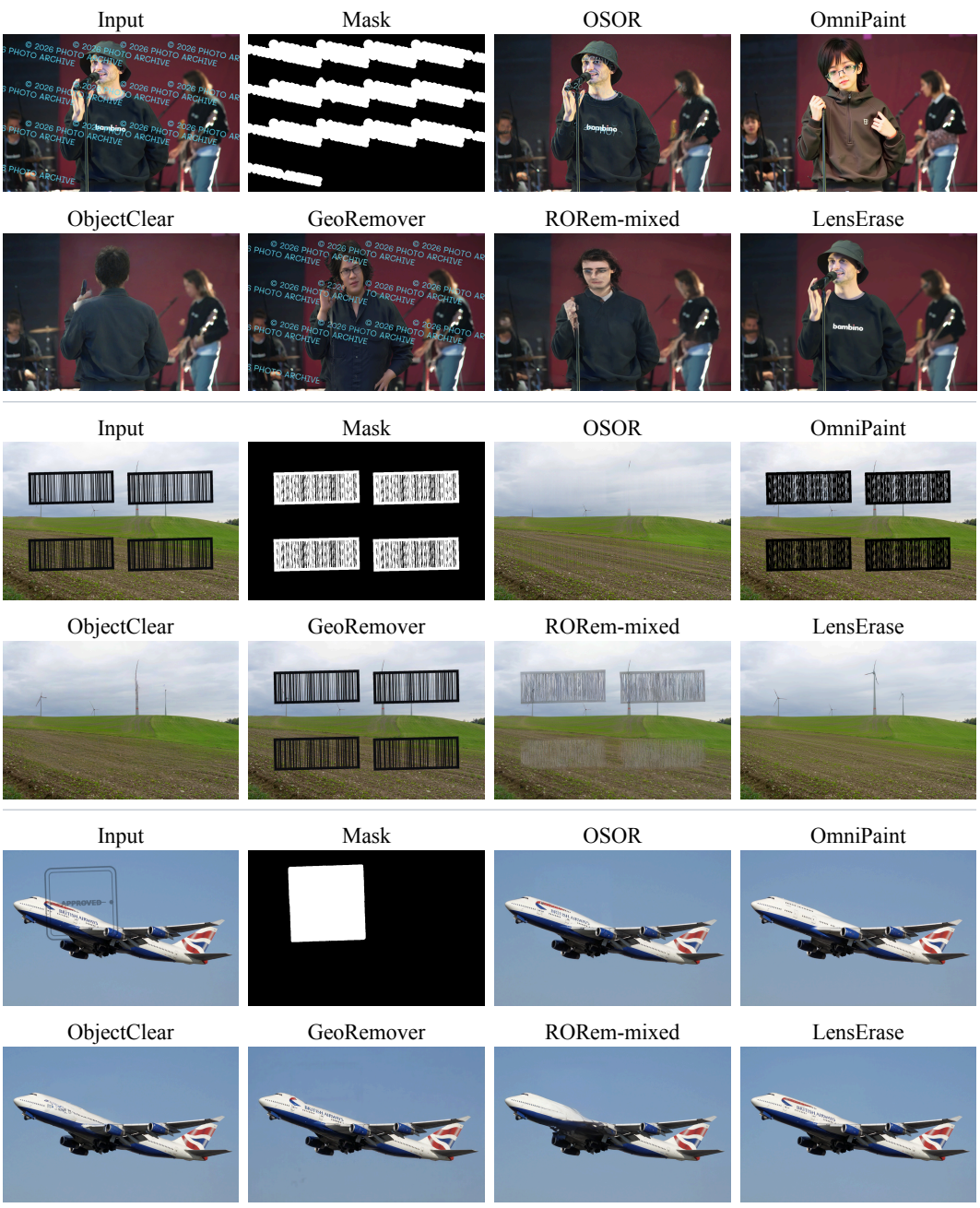

Figure 15: Qualitative comparisons on watermark removal.

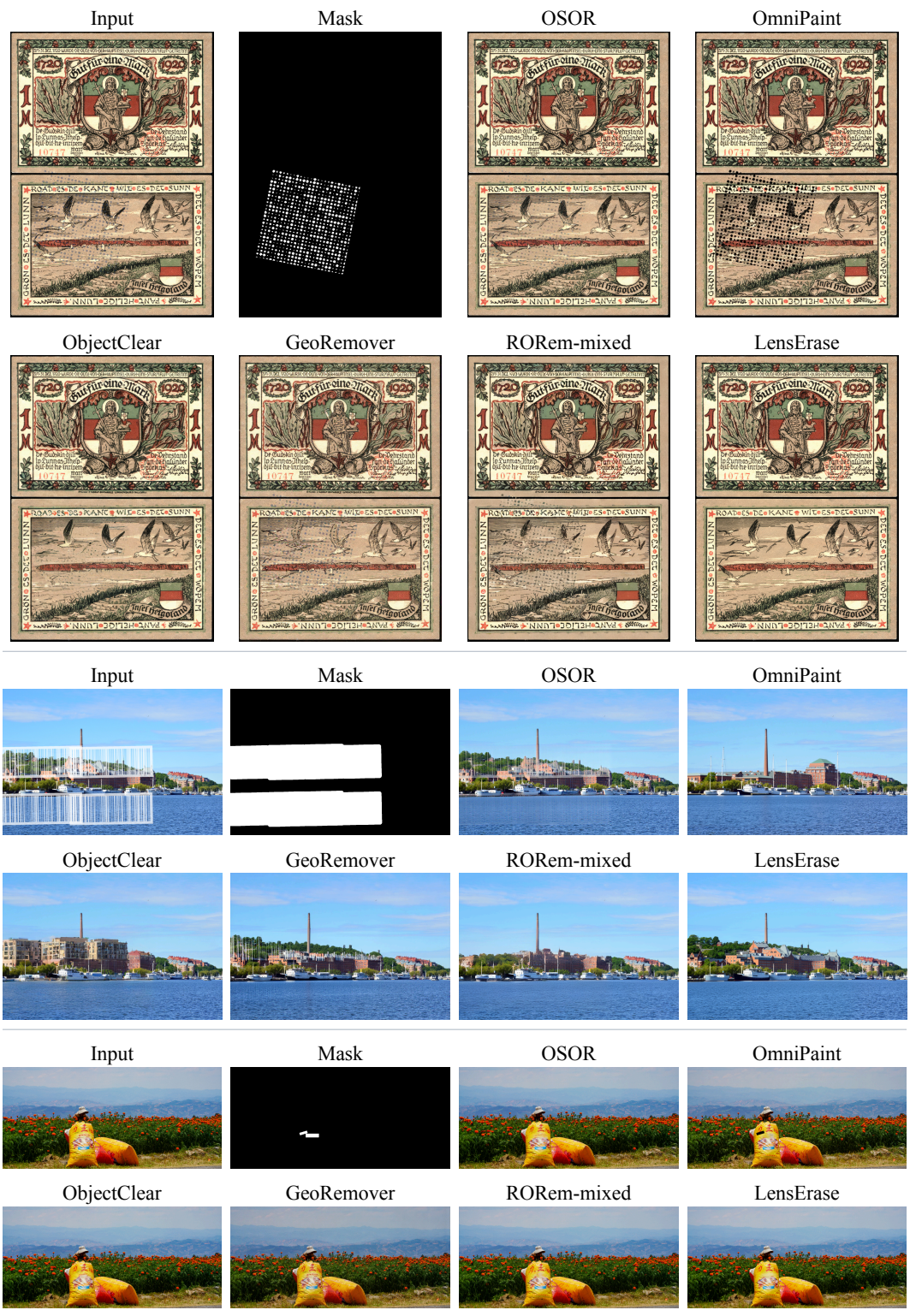

Figure 16: Qualitative comparisons on watermark and visual text removal.

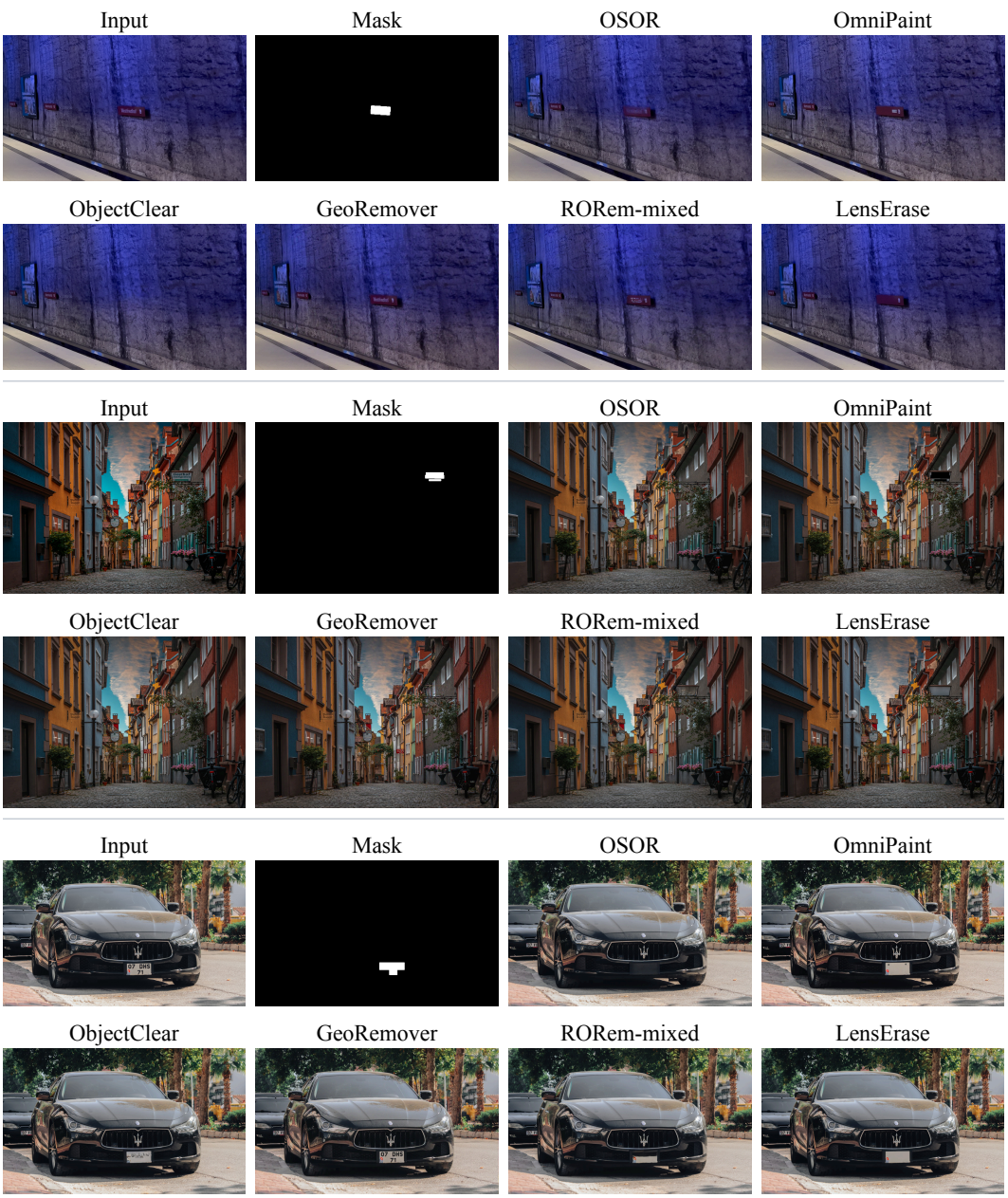

Figure 17: Qualitative comparisons on visual text removal.

J REMOVAL WITH HAND-DRAWN AND SAM MASKS

Figures 18–21 present additional LensErase results using hand-drawn masks and SAM 2 segmentations (Ravi et al., 2024). The examples include text, transparent objects, multiple targets, and associated shadows or reflections. Each row shows the input image, mask, and LensErase output.



Figure 18: LensErase results with user-provided masks.

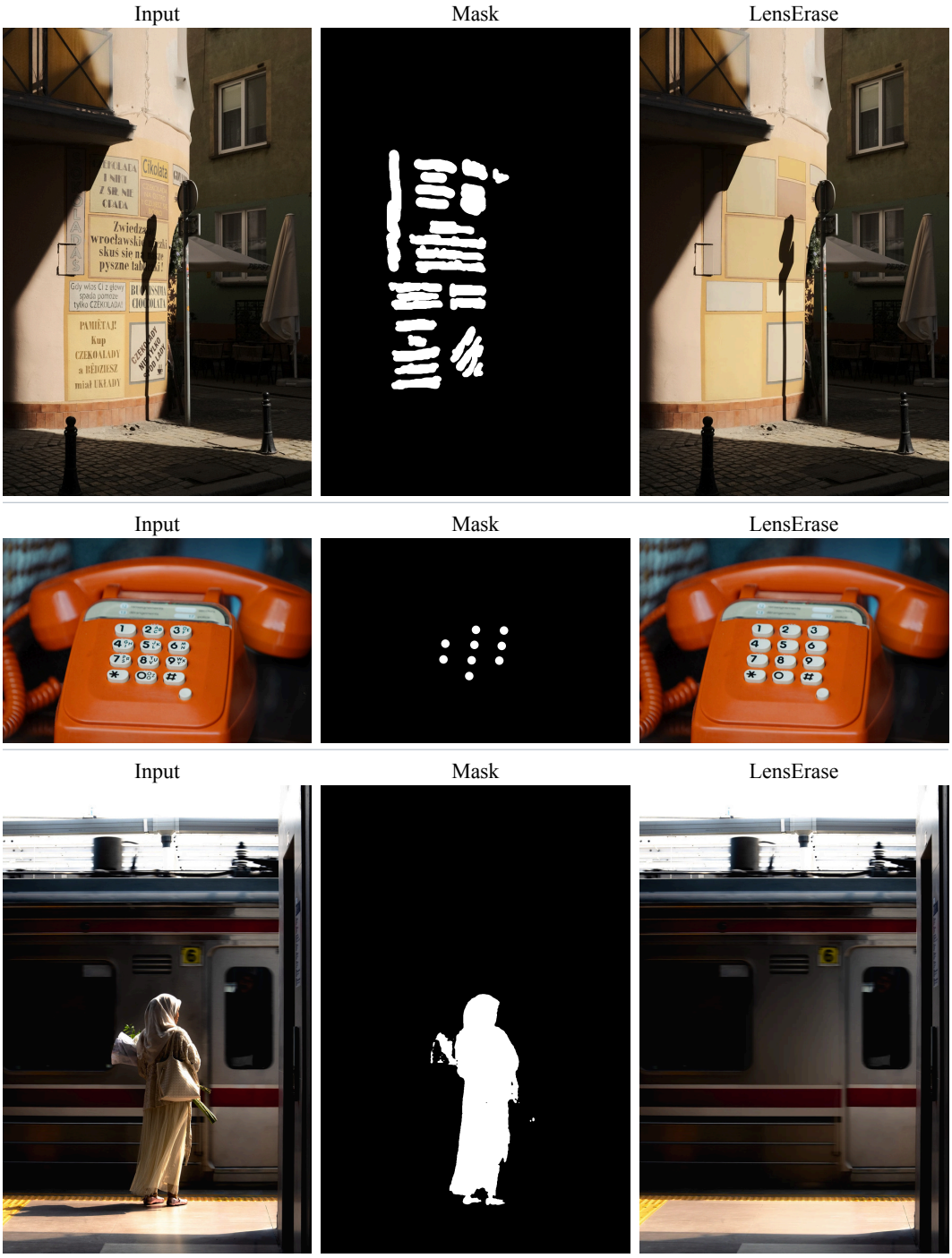

Figure 19: LensErase results with user-provided masks.

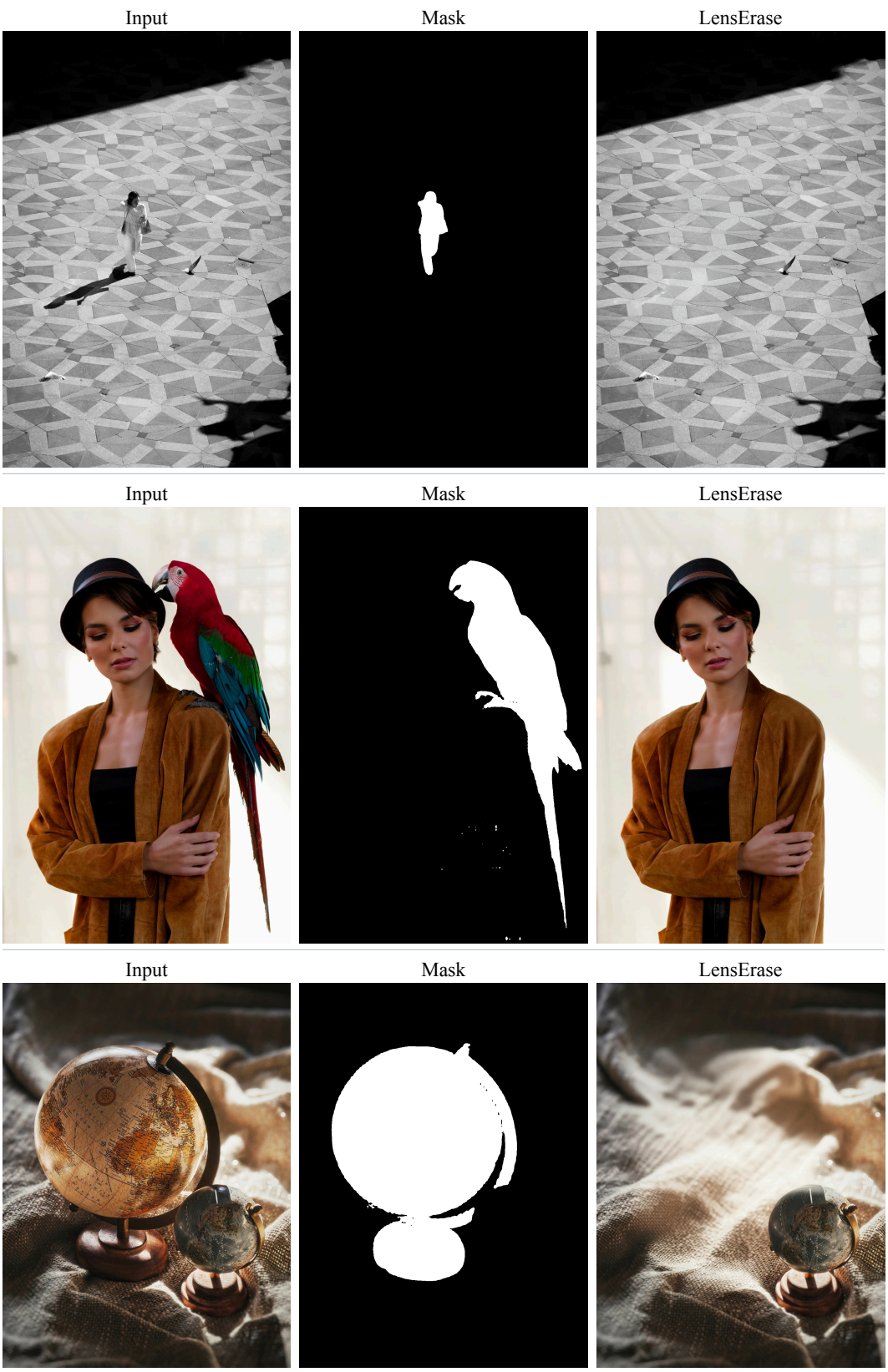

Figure 20: LensErase results with user-provided masks.

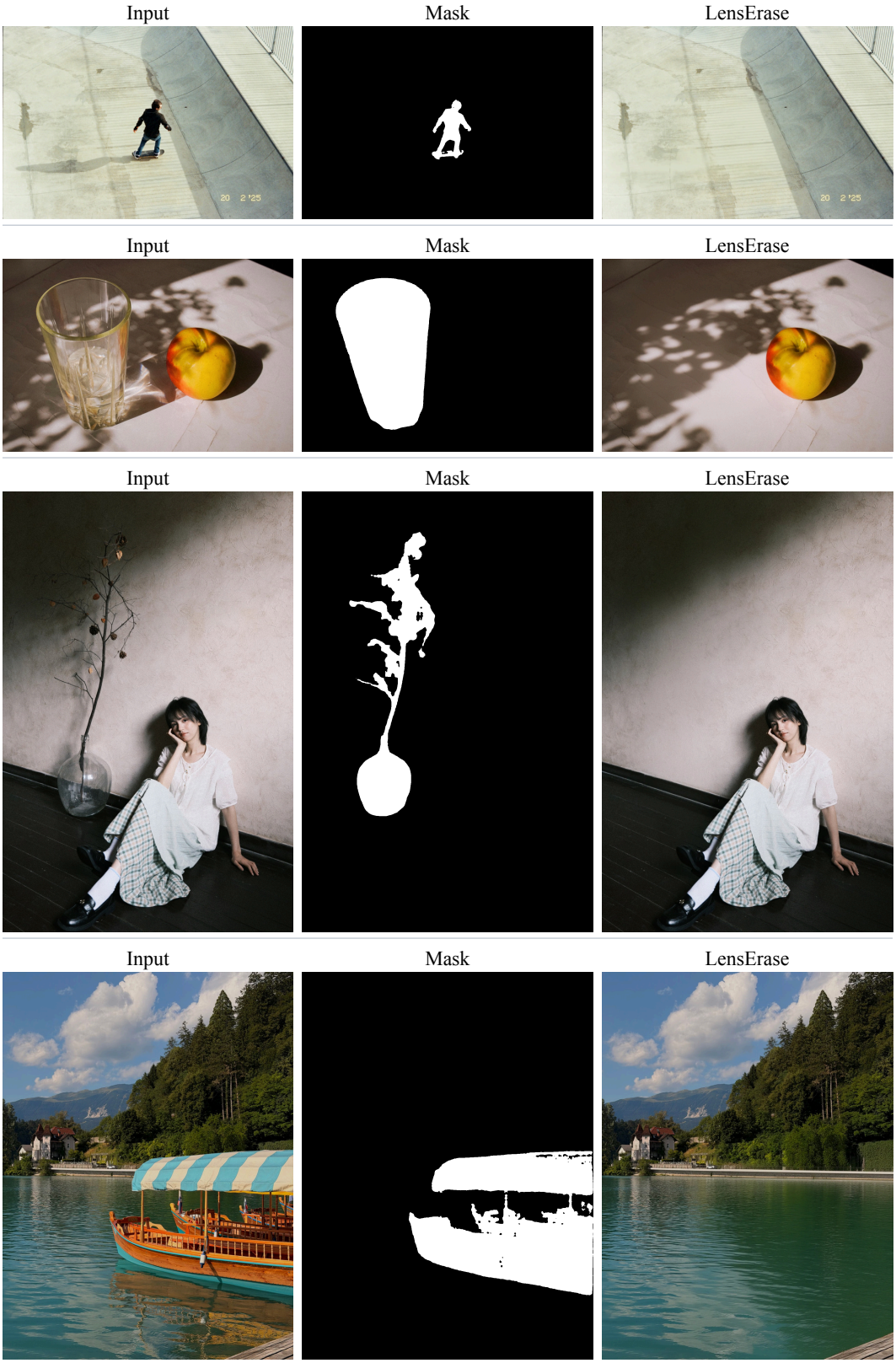

Figure 21: LensErase results with user-provided masks.