

BFS: BACK-TO-FRONT LAYERED IMAGE SYNTHESIS VIA KNOWLEDGE TRANSFER

Anonymous authors

Paper under double-blind review

ABSTRACT

Layered images have long served as a crucial representation for creative editing, and the advent of large-scale generative models has recently spurred interest in their automatic generation. Nevertheless, existing approaches remain limited. Decomposition-based methods often struggle to achieve clean separation of layers in complex scenes, while generation-based methods face challenges stemming from their training data construction pipeline, which lead to suboptimal visual quality and limited scene diversity. In this paper, we propose BFS, a novel generation-based framework for layered image synthesis. We approach the task through the highly practical background-to-foreground formulation. Specifically, given a background layer and user guidance, it synthesizes a foreground layer that incorporates visual effects such as shadow and reflection while harmonizing with the background to form a coherent composite image. Since constructing suitable training data is difficult, we instead leverage the comparatively easy-to-learn knowledge of composite synthesis for the foreground synthesis. To this end, we design a dual-branch framework that jointly generates a composite image and a foreground layer, enabling knowledge transfer through bidirectional information exchange between the two branches. To promote this transfer, we also propose a two-stage training scheme that does not rely on ground-truth foreground layer, main dataset bottleneck. Extensive experiments show that BFS produces high-quality layered images, consistently outperforming prior methods.

1 INTRODUCTION

Layered images, typically organized as a stack of multiple RGBA layers, have long been a standard representation in professional image creation and editing. Such a representation enables independent manipulation of visual elements without affecting the rest of the image, and this non-destructive property not only streamlines iterative creative workflows but also enables dynamic reuse of visual elements across different compositions.

Recent advances in image generative models (Ramesh et al., 2022; Saharia et al., 2022; Rombach et al., 2022; Podell et al., 2023) have opened new opportunities for automatic layered image synthesis. According to their layer construction strategy, existing approaches can be broadly categorized into two paradigms: *decomposition-* and *generation-based* methods. *Decomposition-based* methods (Kang et al., 2025; Yang et al., 2025) extend traditional image matting approaches. They break down an existing composite image into its constituent foreground and background layers, typically guided by user-provided foreground masks. However, these methods often struggle to accurately extract foreground objects together with their associated visual effects, such as shadows and reflections, when applied to complex scenes. Moreover, they are inherently limited to separating existing images and lack the ability to synthesize new layers to construct novel compositions.

In contrast, *generation-based* methods (Zhang et al., 2023b; Huang et al., 2024; Zhang & Agrawala, 2024; Dalva et al., 2024; Huang et al., 2025b; Pu et al., 2025; Huang et al., 2025a) directly synthesize layers from random noise and text descriptions. They produce sharp and accurate alpha masks and allow the straightforward creation of new layered images using text description, which has driven growing research interest. However, since learning the distribution of layered images demands a large-scale layered image dataset, which is difficult to obtain, developing a scalable training data construction pipeline has emerged as a central challenge.

For example, Text2Layer (Zhang et al., 2023b), the MuLAn dataset (Tudosiu et al., 2024), and DreamLayer (Huang et al., 2025b) propose separation-based data construction pipelines. Starting from a composite image dataset, they extract foreground layers and then inpaint the residual regions to produce background layers. Starting from a composite image dataset, they extract foreground

layers and then inpaint the residual regions to produce background layers. However, this process often leaves visual effects in the background instead of the foreground, which in turn causes the trained model to generate incorrect outputs. Although DreamLayer (Huang et al., 2025b) filters out low-quality layered images through manual review, such reliance on human labor makes the approach unsuitable for scalable dataset construction.

Other pipelines proposed by Zhang & Agrawala (2024) and Huang et al. (2025a), start from an RGBA foreground dataset. Specifically, Zhang & Agrawala (2024) first apply outpainting outside the foreground regions to obtain composite images, then remove the foregrounds and inpaint the regions to produce background layers. While this approach allows foreground layers to retain their visual effects, these effects are typically not harmonized with the background. As a result, the generated layered images, and therefore the output trained model often exhibit poor harmonization between foreground and background layers. Moreover, the initial RGBA foreground set is limited in both diversity and scale, making the resulting composite images less representative of real-world imagery. Consequently, the generated layered images are confined to a narrow range of object categories and scene variations, limiting their applicability to broader real-world scenarios. Meanwhile, PSDiffusion (Huang et al., 2025a) hires professional designers to select assets from foreground and background datasets and carefully combine them, but this labor-intensive curation is prohibitively costly. These limitations of previous works highlight that layered image synthesis remains challenging, particularly the collecting high-quality data at scale without heavy reliance on human labor.

In this paper, we propose BFS (Back-to-Front layered image Synthesis), a novel generation-based framework for high-quality layered image synthesis. Specifically, we focus on BG2FG (background-to-foreground) synthesis, i.e., synthesizing a new foreground layer conditioned on a given background image. While alternative strategies exist such as generating all layers at once or foreground-to-background synthesis, BG2FG synthesis holds a couple of distinct practical importance. Firstly, it closely aligns with the common design process where additional objects are introduced into a scene one at a time to explore diverse layer compositions. Secondly, it naturally extends to multi-layered image generation by sequentially adding new layers.

Constructing a suitable training dataset for this task, however, is challenging. Such a dataset should be sufficiently large to provide reliable supervision for the scale, position, and pose of the foreground objects relative to the background. In addition, the foreground layers should faithfully incorporate realistic visual effects consistent with the background’s contextual semantics, but collecting such foreground-background pairs remains extremely difficult.

To overcome this dataset challenge, we reduce reliance on ground-truth foreground layers and instead leverage the comparatively easier-to-learn knowledge of composite synthesis for the foreground synthesis. To this end, we first design a dual-branch generation framework that jointly synthesizes the composite image and the foreground layer, facilitating the knowledge transfer by aligning their generation pathways with bidirectional information exchange. Concretely, on a shared pretrained diffusion transformer, we attach two content LoRA modules dedicated to each modality and an information-sharing module that couples the two branches.

This dual-branch framework enables effective knowledge transfer in two respects. First, the foreground layer is generated in conjunction with the composite image rather than in isolation, allowing composition-aware synthesis that naturally incorporates realistic visual effects. Second, learning the composite branch is relatively easy, as composite generation closely aligns with the pretrained knowledge of producing high-quality images. By conditioning on the composite branch, the foreground branch can be trained stably and efficiently, even with limited data.

Furthermore, we propose a two-stage training strategy to promote knowledge transfer, without the reliance on ground-truth foreground layers. In the first stage, we learn the representations of the outputs of both branches and their compositional relationships. Specifically, we train on a synthetic dataset of well-aligned triplets of foregrounds with simulated effects, backgrounds, and composite images, using a diffusion loss and our composition loss. In the second stage, we enhance the realism of the composite branch, which naturally transfers to the foreground branch through the information-sharing learned in the previous stage. Specifically, we supervise the composite branch using a realism-rich dataset composed of backgrounds and composites, which are much easier to obtain than ground-truth foreground layers. We also introduce a regularization scheme for the foreground branch to keep its distribution during this stage.

With extensive experiments, we demonstrate that BFS achieves high-quality layered image synthesis, in which foreground and background layers are well harmonized, along with strong generalization capability compared to existing approaches. In addition, our method proves effective across diverse practical applications such as reference-based foreground generation and foreground layer extraction from composite and background images.

2 RELATED WORK: LAYERED IMAGE SYNTHESIS

Motivated by the practical value of layered image representations, interest in their automatic generation has grown rapidly. In this section, we review the architectural designs adopted by existing methods. Beyond early CLIP-based attempt of Text2Live (Bar-Tal et al., 2022), most contemporary methods leverage the generative power of pretrained diffusion models via fine-tuning. Decomposition-based methods (Kang et al., 2025; Yang et al., 2025) adapt the input/output layers of diffusion models to decompose a composite image into its constituent layers, thereby producing separate foreground and background layers.

Generative-based methods directly synthesize layers from random noise and text descriptions. Text2Layer (Zhang et al., 2023b) introduces a unified latent space in which foreground and background are jointly embedded, enabling both layers to be generated in a single pass. Subsequent methods (Huang et al., 2024; Zhang & Agrawala, 2024; Dalva et al., 2024; Huang et al., 2025b;a) have established a dominant paradigm of multi-pass joint synthesis, allocating a dedicated generative pathway to each layer. For instance, LayerDiff (Huang et al., 2024) synthesizes layers from layer-specific text prompts while promoting both intra- and inter-layer interactions via a collaborative attention block, and LayerDiffuse (Zhang & Agrawala, 2024) leverages layer-specific LoRA adapters for each layer and aggregates all attention vectors across all pathways to form a unified model. More recent works (Dalva et al., 2024; Huang et al., 2025b;a) augment the multi-pass design with an additional composite branch, leveraging cross-attention maps to inject global scene context into the individual layers. Despite these advances, they remain tied to U-Net-based diffusion backbones and depend heavily on cross-attention maps, with only limited exploration of emerging diffusion transformer (DiT) architectures. In contrast, the only DiT-based work, ART (Pu et al., 2025), adopts a fundamentally different strategy. It first generates a global layout, assigns latent tokens to spatial regions, and then synthesizes all layers simultaneously using a diffusion transformer. While this design enables efficient region-wise generation, it inherently limits the ability to capture holistic visual effects that span across multiple regions.

3 BFS (BACK-TO-FRONT LAYERED IMAGE SYNTHESIS)

As introduced in Sec. 1, we tackle the BG2FG problem in layered image synthesis by proposing BFS, which is illustrated in Fig. 1. Specifically, given a background image B , a bounding-box mask M , and a foreground text description T , it generates a foreground RGBA layer $F = (F_{rgb}, F_{\alpha})$ that contains the object specified by T within M . The generated foreground layer is expected to (1) incorporate associated visual effects such as shadow and reflections, and (2) naturally harmonize with the background B , thereby producing a natural-looking composite image C . The composition follows the standard alpha blending equation:

$$C = F_{\alpha} \cdot F_{rgb} + (1 - F_{\alpha}) \cdot B, \quad (1)$$

where F_{rgb} and F_{α} represent the RGB color components and opacity map of F , respectively, and $\cdot$ denotes element-wise multiplication.

The core contribution of BFS lies in transferring the relatively easier-to-learn knowledge of composite synthesis to guide more challenging foreground synthesis, thereby effectively bypassing the difficulty of constructing training data. To this end, we design a dual-branch framework that facilitates this transfer. Specifically, it simultaneously synthesizes both a composite image and a foreground layer through two dedicated branches. At the same time, it enables bidirectional information exchange between the two generation pathways, which ensures that the two modalities remain well aligned, effectively guiding the foreground branch to follow the composite branch. Architecturally, we take inspiration from UniCon (Li et al., 2024), a UNet-based joint generation framework, and adapt it to our diffusion transformer backbone.

BFS builds on Flux-Fill, an image inpainting model chosen for its task similarity. We attach two *content* LoRA modules (a composite LoRA and a foreground LoRA) to all the transformer blocks in

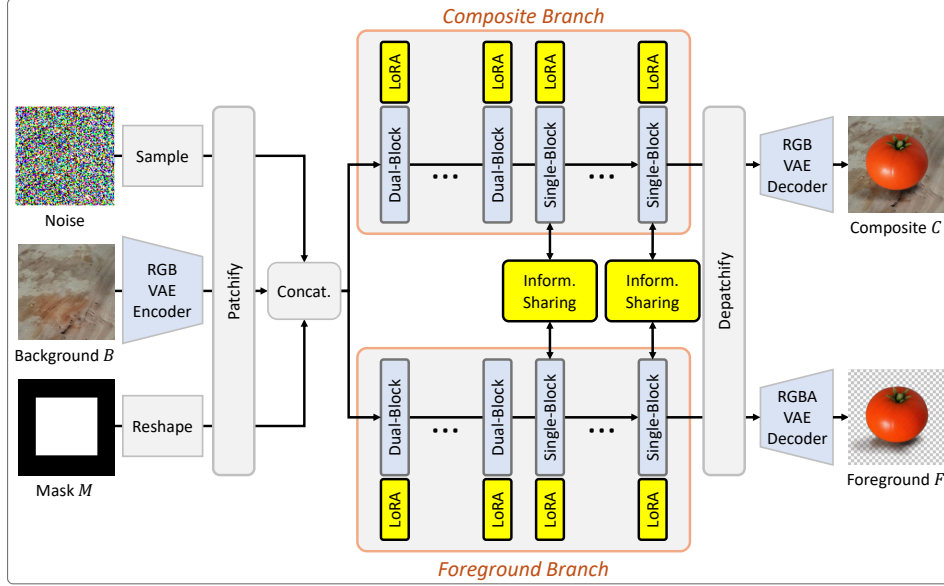


Figure 1: Overall framework of BFS. Given a background image B , a bounding-box M , and a foreground text description T , the model generates a foreground RGBA layer F . To enhance generation quality, it simultaneously synthesizes a composite image and transfers knowledge from composite generation to foreground generation. We train only the modules highlighted in yellow.

Flux-Fill. Each LoRA operates exclusively on its corresponding pathway, guiding the synthesis of each modality. To enable bidirectional information exchange between the pathways, thereby aligning both modalities, we introduce an *information-sharing* module featuring a symmetric cross-attention layer. For efficiency, this module is inserted only into the 38 single-stream blocks that follow the initial 19 dual-stream blocks.

The information-sharing module operates as follows. Let H^C and H^F be the intermediate features before the self-attention layer of the composite and foreground pathways at a single-stream block, respectively. We compute two cross-attentions by querying one pathway with keys/values from the other:

$$Z^{C \rightarrow F} = \text{softmax}\left(\frac{Q^F(K^C)^\top}{\sqrt{d}}\right) V^C, \quad Z^{F \rightarrow C} = \text{softmax}\left(\frac{Q^C(K^F)^\top}{\sqrt{d}}\right) V^F, \quad (2)$$

where Q , K , and V denote the query, key, and value embeddings of each branch, and d is the embedding dimension. The two cross-attention messages are concatenated, passed through a lightweight MLP $g(\cdot)$, and split into each branch via residual updates:

$$[\Delta H^C, \Delta H^F] = g(\text{concat}(Z^{F \rightarrow C}, Z^{C \rightarrow F})), \quad \tilde{H}^C = H^C + \Delta H^C, \quad \tilde{H}^F = H^F + \Delta H^F. \quad (3)$$

For cross-attention projection, we use shared and frozen base matrices W_Q , W_K , and W_V , and introduce two *direction-specific* LoRAs. The resulting feature embeddings are computed as:

$$\begin{aligned} Q^F &= H^F(W_Q + \Delta W_Q^{C \rightarrow F}), \quad K^C = H^C(W_K + \Delta W_K^{C \rightarrow F}), \quad V^C = H^C(W_V + \Delta W_V^{C \rightarrow F}), \\ Q^C &= H^C(W_Q + \Delta W_Q^{F \rightarrow C}), \quad K^F = H^F(W_K + \Delta W_K^{F \rightarrow C}), \quad V^F = H^F(W_V + \Delta W_V^{F \rightarrow C}). \end{aligned} \quad (4)$$

The base projections (W_Q, W_K, W_V) are initialized from the pretrained self-attention weights of each block. The final trainable components of BFS are the two *content* LoRAs, the two *direction-specific information-sharing* LoRAs $\{\Delta W_{\cdot}^{C \rightarrow F}, \Delta W_{\cdot}^{F \rightarrow C}\}$, and the lightweight MLP.

At inference time, the diffusion denoising process operates in the latent space of the pretrained VAE. Two same random noise tensors are first channel-wise concatenated with the conditioning inputs (the background latent and a bounding box mask reshaped following Flux-Fill). The concatenated tensors are then stacked along the batch dimension and passed through the diffusion transformer with text embeddings computed from foreground descriptions. The two content LoRAs are applied independently to different batch elements, while the information-sharing module attends jointly to both.

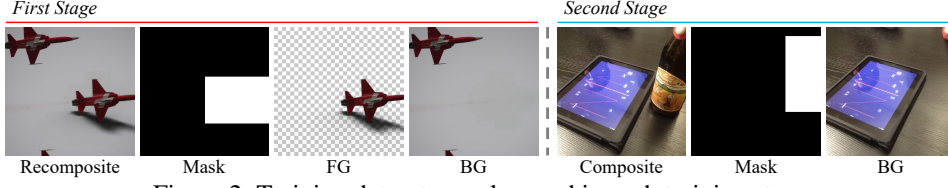


Figure 2: Training dataset samples used in each training stage.

After iterative denoising, the model produces a composite latent and a foreground latent. Finally, the only foreground latent is decoded to image domain using RGBA VAE introduced in LayerDiffuse (Zhang & Agrawala, 2024). Further implementation details are provided in the appendix.

4 TWO-STAGE TRAINING STRATEGY

To train BFS, the most straightforward dataset would comprise foreground, background, composite images, and bounding-box masks. Such a dataset must be sufficiently large to provide reliable supervision for the scale, position, and pose of the foreground object, while the foreground layers should be represented in RGBA format and faithfully capture realistic visual effects. Yet, as discussed in Sec. 1, constructing a layered image dataset in practice is highly challenging. Although background and composite image pairs can be obtained relatively easily through recent object removal techniques, acquiring corresponding foreground representations remains extremely difficult. To address this challenge, we propose a two-stage training strategy that avoids reliance on explicit ground-truth foreground layers. In brief, we first train our model on a simulation-based synthetic dataset containing all modalities, and then fine-tune it on realism-rich background and composite pairs to enhance visual fidelity and contextual harmonization of the composites, which in turn improves the quality of the foreground layers through the information-sharing module. The following paragraphs provide details of the datasets and the training losses used in each stage.

Representation Learning with Compositional Consistency BFS first learns modality-specific representations and their compositional relationships using a simulation-based synthetic dataset composed of triplets of foreground, background, composite images, along with bounding-box masks. To construct the dataset, we adopt an approach similar to LayerDecomp (Yang et al., 2025) on the RORem object removal dataset (Li et al., 2025), which provides composite images, object masks, and background images where both objects and their visual effects have been removed. For each composite image C , we first extract a foreground layer F from the object mask M using a matting technique (Yao et al., 2024), then augment it to include simulated shadows using an internal shadow generation network. The augmented foreground layer $\hat{F}$ is recomposited with the background layer B to form a new composite image $\hat{C}$. Finally, we replace the original object mask with a bounding-box mask $\hat{M}$, resulting in a training corpus of $\{\hat{F}, B, \hat{M}, \hat{C}\}$, as shown in the left side in Fig. 2. We also provide visual examples of the dataset construction pipeline in the appendix. For notational simplicity, we drop the hat in the rest of the paper.

Our data construction pipeline produces image layers with precise pixel-level alignment and embeds simulated visual effect into the foreground layer, even though these effects are less realistic and not harmonized with the background layer. In contrast to LayerDecomp, which constructs composites by randomly pairing foregrounds with backgrounds, our method derives foregrounds directly from real composite images, thereby offering more reliable supervision for object placement and scale.

Using the synthetic dataset, we employ a flow-matching loss. Let z_F, z_B, z_C denote the VAE latents of the foreground, background, and composite images, respectively. At each training iteration, we sample Gaussian noise $z_0 \sim \mathcal{N}(0, I)$ and linearly interpolate it with each data latent $z_1 \in \{z_F, z_C\}$ at a random time step $t \sim \mathcal{U}(0, 1)$: $z^{(t)} = (1 - t)z_0 + tz_1$. The flow-matching objective is then defined as:

$$\mathcal{L}_{\text{flow}} = \mathbb{E}_{t, z_0, z_1} \left[\|v_C^* - v_\theta^C(z_C^{(t)}, z_F^{(t)}, t, \kappa)\|_2^2 + \|v_F^* - v_\theta^F(z_C^{(t)}, z_F^{(t)}, t, \kappa)\|_2^2 \right], \quad (5)$$

where v_C^* and v_F^* are the ground-truth velocity fields (given by $v^* = z_1 - z_0$). v_θ^C and v_θ^F are velocity fields predicted by the composite and foreground branches, respectively. Note that v_θ^C and v_θ^F are computed from both $z_C^{(t)}$ and $z_F^{(t)}$, since the composite and foreground branches are connected through information-sharing modules. Here, κ represents the conditioning inputs, including the foreground text description T , the background latent z_B , and a down-sampled mask M .

To further enforce compositional consistency between the branches, we introduce a composition loss, which is defined as:

$$\mathcal{L}_{\text{comp}} = \|\Phi(\hat{z}_F, z_B) - \hat{z}_C\|_1, \quad (6)$$

where $\hat{z}_C$ and $\hat{z}_F$ are the model’s one-step denoised estimates at time t , (i.e., the predicted clean latents from $z_C^{(t)}$ and $z_F^{(t)}$). Φ is a function that computes a composite of $\hat{z}_F$ and z_B in the latent space. Φ is modeled as a neural network with learnable parameters and trained in advance to train our framework. Finally, the overall objective of the first stage is defined as:

$$\mathcal{L}_{\text{first}} = \mathcal{L}_{\text{flow}} + 0.1\mathcal{L}_{\text{comp}}. \quad (7)$$

Details of the auxiliary networks used during training are provided in the appendix.

Realism Enhancement with Distribution Regularization In this stage, we enhance the realism of the generated images using a realism-rich dataset that captures diverse real-world visual effects. Specifically, we utilize paired background and composite images to supervise only the composite LoRA so that the generated composite latent better matches realistic image statistics. The resulting improvements then propagate to the foreground branch through the information-sharing modules, aligning the foreground with the refined composite and calibrating its visual effects.

For the second stage, we construct a dataset based on the object-removal dataset RORem (Li et al., 2025), which provides background and composite images and bounding-box masks. The background images of RORem are generated by inpainting the masked regions in the composite images, but they often retain residual visual effects of foreground objects such as shadows and reflections. To obtain cleaner backgrounds, we instead synthesize background images using ObjectClear (Zhao et al., 2025), a more advanced object-removal method that removes both objects and their associated visual effects. This dataset construction pipeline is scalable and readily allows further expansion of the training corpus in the future.

Based on the dataset, we define a realism-boosting loss that supervises only the composite LoRA:

$$\mathcal{L}_{\text{real}} = \mathbb{E}_t \left[\|v_C^* - v_\theta^C(z_C^{(t)}, z_F^{(t)}, t, \kappa)\|_2^2 \right], \quad (8)$$

where $v^* = z_1 - z_0$. As no ground-truth foreground layer F is available, we instead use a *surrogate* input for the foreground branch, specifically $\bar{F} = C \cdot M$, obtained by masking the composite image C with the binary mask M . We empirically find that this surrogate input has negligible influence on learning dynamics.

While the composite supervision improves realism, relying on it alone may cause the foreground distribution to drift away from the distribution established in the previous stage, degrading the quality of the foreground layer. To mitigate this, we introduce additional supervision for the foreground LoRA and the information-sharing module using the synthetic dataset of the previous stage. Although the dataset is not fully realistic, it benefits from the pixel-wise alignment across $\{F, B, C\}$. Specifically, when we feed the clean composite latent (without adding noise) as the input of the composite branch with timestep $t = 0$, the branch is expected to faithfully reconstruct the clean composite latent. At the same time, the foreground branch should estimate the residual between C and B (i.e., F) to satisfy the compositional relationship. To encourage this behavior, we introduce a regularization loss:

$$\mathcal{L}_{\text{reg}} = \mathbb{E}_t \left[\|v_F^* - v_\theta^F(z_C, z_F^{(t)}, (0, t), \kappa)\|_2^2 \right] \quad (9)$$

where $(0, t)$ indicates that we use different timesteps for the composite and foreground branches, i.e., we set the timestep to 0 for the composite branch, while setting it to t for the foreground branch. During training iterations, we alternate between optimizing the realism-boosting loss and the foreground regularization loss, each combined with the composition loss. Specifically, we optimize $\mathcal{L}_{\text{real}}$ and $\mathcal{L}_{\text{reg}} + 0.1\mathcal{L}_{\text{comp}}$ in turn.

5 EXPERIMENTS

5.1 COMPARISON WITH OTHER LAYERED IMAGE SYNTHESIS METHODS

Baselines We compare the quality of the layered images generated by BFS and existing approaches on multiple datasets, including SAM-FB (He et al., 2024), RORem (Li et al., 2025), and AnyInsertion (Song et al., 2025). Each dataset provides background images and masks specifying the regions

Table 1: Quantitative evaluation of existing layered image synthesis methods on SAM-FB dataset.

Method	FG	Recomposite (or Background for LayeringDiff + ObjectClear)			
	CLIP ($\uparrow$)	MUSIQ ($\uparrow$)	MANIQA ($\uparrow$)	KID $\times$ 1000 ($\downarrow$)	DINO ($\uparrow$)
LayeringDiff + ObjectClear	0.59	70.33	0.430	0.72	1.00
LayerDiffuse	0.78	70.47	0.411	10.89	0.50
Ours	0.63	70.99	0.432	0.62	0.96

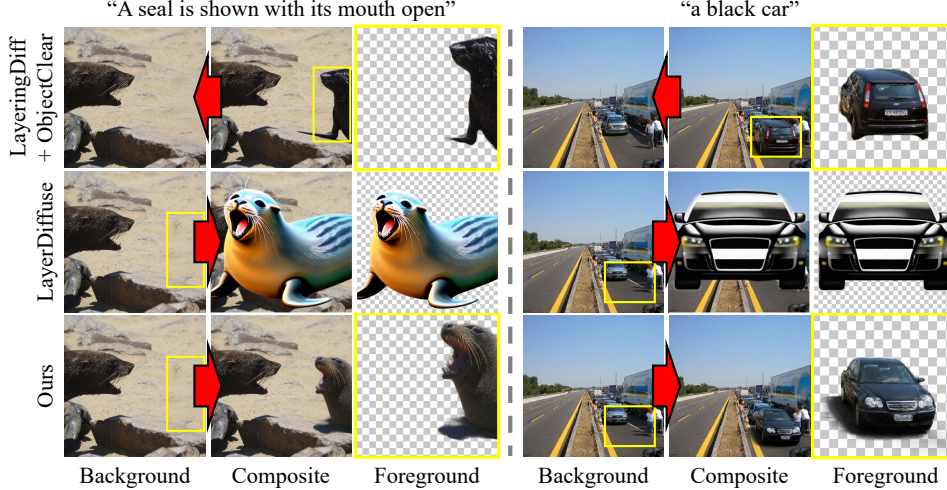


Figure 3: Qualitative comparison with existing layered image synthesis approaches. For better visual comparison, we present zoom-in views of the generated foreground layer except for LayerDiffuse, as its foreground spans the entire region.

where foreground objects should be placed. For consistency, all masks are converted into bounding-box form. All input images (backgrounds and masks) are resized such that the shorter side is 512 pixels before being fed into the models. For the target foreground text descriptions, we use the captions provided by the SAM-FB and AnyInsertion datasets, while those for the RORem dataset are generated using BLIP2 (Li et al., 2023).

As the decomposition-based methods cannot generate new foreground layers, we instead use them to decompose the ground-truth composite images into layered images and evaluate them. Specifically, we compute foreground masks using SAM (Ravi et al., 2024) based on the bounding boxes of the object masks, and employ LayeringDiff (Kang et al., 2025) to estimate the foreground layers. Background layers are generated with ObjectClear (Zhao et al., 2025), since LayeringDiff often leaves residual visual effects in the background. For LayeringDiff, we use the code provided by the authors, while for ObjectClear we use the official implementation. LayerDecomp (Yang et al., 2025) could not be evaluated because its code is not publicly available, but the quality of its generated background layers can be reasonably assumed to be comparable to that of ObjectClear, given the close similarity in their training strategies.

For generation-based methods, we compare only with LayerDiffuse (Zhang & Agrawala, 2024), as it is the only method with publicly available code. Specifically, we adopt its strongest SDXL-based two-stage variant, which first generates a composite image and then estimates the foreground layer conditioned on the background and the composite.

Qualitative Comparison Fig. 3 presents a qualitative comparison. For clearer visualization, we enlarge the foreground regions generated by our method and by LayeringDiff combined with ObjectClear. As shown in the figure, LayeringDiff fails to disentangle fine details in complex scenes, such as the whiskers of a seal or the side mirror of a car. The generated foreground objects by LayerDiffuse are frequently oversized and inconsistent with the background context, leading to poor visual harmony. This indicates that the synthetic dataset used for training does not generalize well to real-world backgrounds, leaving the model uncertain about how to position objects, and simply producing large, centrally placed ones that follow the typical distribution of RGBA images. In contrast, our method generates foreground objects that are contextually appropriate, visually consistent, and equipped with proper visual effects, even in complex scenes, yielding more coherent composites.

Quantitative Comparison Quantitative evaluation of generated layered images is inherently challenging, as ground-truth layered images are not available. In this work, we evaluate both the gener-

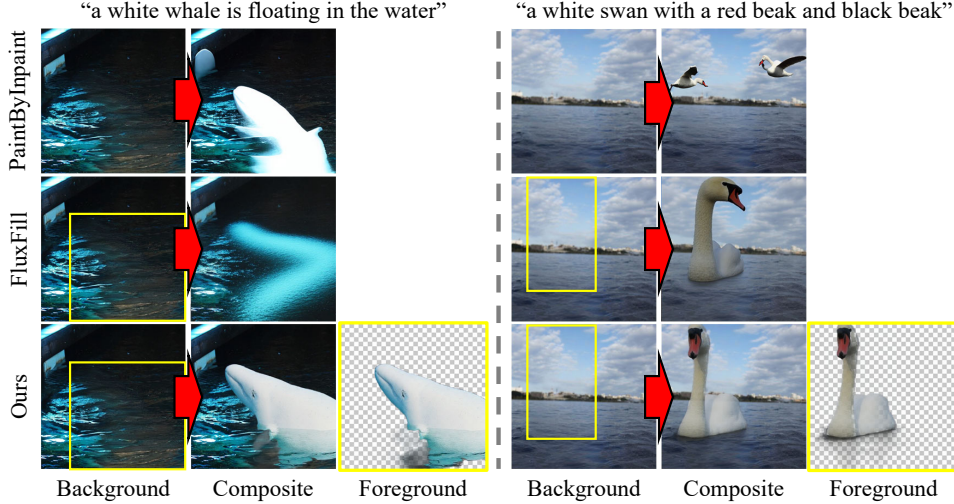


Figure 4: Qualitative comparison with an object insertion method PaintByInpaint (Wasserman et al., 2025) and a image inpainting method FluxFill (Labs, 2024)

Table 2: Quantitative evaluation with an object insertion method FluxFill and an image inpainting method PaintByInpaint on SAM-FB dataset.

Method	MUSIQ ($\uparrow$)	MANIQA ($\uparrow$)	KID $\times 1000$ ($\downarrow$)	DINO ($\uparrow$)
PaintByInpaint (Wasserman et al., 2025)	70.89	0.41	0.76	0.93
FluxFill (Labs, 2024)	70.71	0.42	0.79	0.96
Ours	70.99	0.43	0.62	0.96

ated foreground layer and the recomposite image obtained by combining the generated foreground with the input background. We report results on the SAM-FB dataset, while results on two additional datasets are provided in the appendix. For decomposition-based methods, where the input is a composite image, we instead evaluate the estimated background layers rather than the composite image. For evaluation metrics, the quality of the foreground layers is assessed using the CLIP score (Hessel et al., 2021), which measures the alignment between the generated (or decomposed) foreground and the input foreground caption. For recomposite evaluation, we employ non-reference aesthetic quality metrics, MUSIQ (Yu et al., 2025), MANIQA (Yu et al., 2025), as well as FID (Heusel et al., 2017) to measure distributional similarity against the target distribution. In addition, we report the DINO score (Oquab et al., 2023) to directly compare with ground-truth composite images. The results are summarized in Tab. 1. LayerDiffuse attains the highest CLIP scores, as it often generates large foreground objects that dominate the image and are therefore favored by the metrics. However, its performance on recomposite evaluation is notably worse than other methods, especially in KID and DINO, where the generated results deviate substantially from the ground-truth distribution. Our method achieves a higher CLIP score than LayeringDiff, while also producing foreground layers that yield recomposite images with superior aesthetic quality and closer alignment to the ground-truth.

5.2 COMPARATIVE WITH OTHER OBJECT INSERTION METHODS

BFS is conceptually related to naturally adding new objects into a given scene. To evaluate this capability, we compare it against two recent models designed for related tasks: the image inpainting model Flux-Fill (Labs, 2024) and the object insertion method PaintByInpaint (Wasserman et al., 2025). For Flux-Fill, we adopt the default configuration. Since PaintByInpaint does not take an input mask and follows an instruction-based text description, we prepend the word “add” to the foreground caption before providing it to the model.

Fig. 4 presents a qualitative comparison of the composites generated by these baselines and recomposite image by our approach. For reference, we also show the foreground layers produced by BFS. As shown in the figure, Our method synthesizes high-quality recomposite images comparable to existing approaches, while additionally generating explicit and reusable foreground layers that facilitate further editing and composition. Tab. 2 summarizes the quantitative evaluation on the SAM-FB dataset. The results show that our method achieves slightly higher scores than existing approaches, demonstrating that BFS produces highly natural recomposite images comparable to those of state-of-the-art insertion models trained solely with real-world supervision.

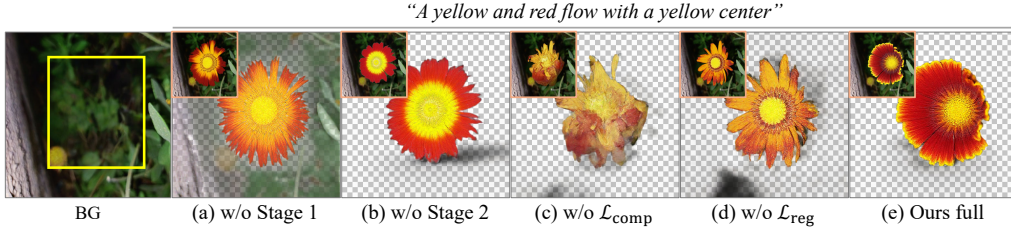


Figure 5: Ablation study. Qualitative comparison of BFS with different components removed. The inset in the top-left shows the output of the composite branch for reference.

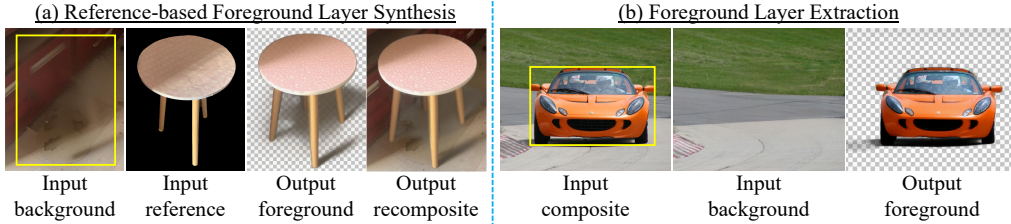


Figure 6: Practical applications of BFS. (a) Reference-based foreground layer synthesis guided by a reference image. (b) Foreground layer extraction from a composite and its background.

5.3 ABLATION STUDY

We conduct an extensive ablation study to evaluate the contribution of each component in BFS. Fig. 5 presents qualitative comparisons of the results obtained by different model variants, given a background image and a bounding-box mask (highlighted in yellow). We also include the outputs of the composite branch for each variant in the figure inset. Without Stage 1 representation learning, directly training Stage 2 completely fails to capture the foreground distribution. In contrast, omitting Stage 2 synthesizes foreground layers that lack realism showing unnatural shadow effect. Removing the composition loss in Stage 1 severely degrades both branches, as effective knowledge transfer does not occur, which diminishes the quality of the foreground layers and the low-quality information is passed to the composite branch, which consequently harms the composite branch. Excluding the regularization step in Stage 2 causes the RGBA distribution of the foreground layer to collapse, introducing irrelevant visual effects into the representation. By contrast, our full model successfully aligns the two branches and produces high-quality RGBA layers.

5.4 FURTHER APPLICATIONS

We present practical applications of BFS. It enables *reference-based foreground synthesis*: by substituting the caption embedding with that of a reference image via a Flux-Redux adapter, the synthesized foreground inherits the reference characteristics (Fig. 6 (a)). BFS also supports *foreground extraction* from existing images. Given a composite and its background, we input the composite into the composite branch (instead of random noise) with its timestep fixed to zero, yielding a residual layer corresponding to the foreground. Unlike conventional matting, this produces a layer that preserves both the object and realistic visual effects (Fig. 6 (b)).

6 CONCLUSION

We present BFS, a generation-based framework for high-quality layered image synthesis. To overcome the difficulty of constructing a suitable training dataset, we transfer the easier-to-learn knowledge of composite synthesis to foreground synthesis, by using a dual-branch framework that jointly generates composite images and foreground layers, with bidirectional information exchange aligning the two branches. To further promote effective knowledge transfer, we also propose two-stage training scheme avoids reliance on ground-truth foreground supervision, which is the main dataset bottleneck. Through extensive experiments, we demonstrated that BFS produces high-quality layered images with strong generalization ability, consistently surpassing prior methods.

BFS has certain limitations. The dual-branch design increases inference time, as generating a single output with BFS takes 31 seconds compared to 12 seconds with Flux-Fill backbone. Furthermore, the visual effects are not guaranteed to be physically correct. We will integrate rendering toolchains to capture complex, physically valid effects and use them to supervise future models.

REFERENCES

- Omer Bar-Tal, Dolev Ofri-Amar, Rafail Fridman, Yoni Kasten, and Tali Dekel. Text2live: Text-driven layered image and video editing. In *European conference on computer vision*, pp. 707–723. Springer, 2022.
- Ryan D Burgert, Brian L Price, Jason Kuen, Yijun Li, and Michael S Ryoo. Magick: A large-scale captioned dataset from matting generated images using chroma keying. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22595–22604, 2024.
- Alper Canberk, Maksym Bondarenko, Ege Ozguroglu, Ruoshi Liu, and Carl Vondrick. Erasedraw: Learning to insert objects by erasing them from images. In *European Conference on Computer Vision*, pp. 144–160. Springer, 2024.
- Gemma Canet Tarrés, Zhe Lin, Zhifei Zhang, Jianming Zhang, Yizhi Song, Dan Ruta, Andrew Gilbert, John Collomosse, and Soo Ye Kim. Thinking outside the bbox: Unconstrained generative object compositing. In *European Conference on Computer Vision*, pp. 476–495. Springer, 2024.
- Xi Chen, Lianghua Huang, Yu Liu, Yujun Shen, Deli Zhao, and Hengshuang Zhao. Anydoor: Zero-shot object-level image customization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6593–6602, 2024.
- Yusuf Dalva, Yijun Li, Qing Liu, Nanxuan Zhao, Jianming Zhang, Zhe Lin, and Pinar Yanardag. Layerfusion: Harmonized multi-layer text-to-image generation with generative priors. *arXiv preprint arXiv:2412.04460*, 2024.
- Jixuan He, Wanhua Li, Ye Liu, Junsik Kim, Donglai Wei, and Hanspeter Pfister. Affordance-aware object insertion via mask-aware dual diffusion. *arXiv preprint arXiv:2412.14462*, 2024.
- Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning. *arXiv preprint arXiv:2104.08718*, 2021.
- Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.
- Dingbang Huang, Wenbo Li, Yifei Zhao, Xinyu Pan, Yanhong Zeng, and Bo Dai. Psdiffusion: Harmonized multi-layer image generation via layout and appearance alignment. *arXiv preprint arXiv:2505.11468*, 2025a.
- Junjia Huang, Pengxiang Yan, Jinhang Cai, Jiyang Liu, Zhao Wang, Yitong Wang, Xinglong Wu, and Guanbin Li. Dreamlayer: Simultaneous multi-layer generation via diffusion mode. *arXiv preprint arXiv:2503.12838*, 2025b.
- Junjia Huang, Pengxiang Yan, Jiyang Liu, Jie Wu, Zhao Wang, Yitong Wang, Liang Lin, and Guanbin Li. Dreamfuse: Adaptive image fusion with diffusion transformer. *arXiv preprint arXiv:2504.08291*, 2025c.
- Runhui Huang, Kaixin Cai, Jianhua Han, Xiaodan Liang, Renjing Pei, Guansong Lu, Songcen Xu, Wei Zhang, and Hang Xu. Layerdiff: Exploring text-guided multi-layered composable image synthesis via layer-collaborative diffusion model. *arXiv preprint arXiv:2403.11929*, 2024.
- Kyoungkook Kang, Gyujin Sim, Geonung Kim, Donguk Kim, Seungho Nam, and Sunghyun Cho. Layeringdiff: Layered image synthesis via generation, then disassembly with generative knowledge. *arXiv preprint arXiv:2501.01197*, 2025.
- Jinwoo Kim, Sangmin Han, Jinho Jeong, Jiwoo Choi, Dongyeoung Kim, and Seon Joo Kim. Orida: Object-centric real-world image composition dataset. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 3051–3060, 2025.
- Black Forest Labs. Flux. <https://github.com/black-forest-labs/flux>, 2024.
- Jizhizi Li, Jing Zhang, Stephen J Maybank, and Dacheng Tao. Bridging composite and real: towards end-to-end deep image matting. *International Journal of Computer Vision*, 130(2):246–266, 2022.

- Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pp. 19730–19742. PMLR, 2023.
- Ruibin Li, Tao Yang, Song Guo, and Lei Zhang. Rorem: Training a robust object remover with human-in-the-loop. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 14024–14035, 2025.
- Xirui Li, Charles Herrmann, Kelvin CK Chan, Yinxiao Li, Deqing Sun, Chao Ma, and Ming-Hsuan Yang. A simple approach to unifying diffusion-based conditional generation. *arXiv preprint arXiv:2410.11439*, 2024.
- Ruofan Liang, Zan Gojcic, Merlin Nimier-David, David Acuna, Nandita Vijaykumar, Sanja Fidler, and Zian Wang. Photorealistic object insertion with diffusion-guided inverse rendering. In *Euro-pean Conference on Computer Vision*, pp. 446–465. Springer, 2024.
- Qingyang Liu, Junqi You, Jianting Wang, Xinhao Tao, Bo Zhang, and Li Niu. Shadow generation for composite image using diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8121–8130, 2024.
- Lingxiao Lu, Jiangtong Li, Bo Zhang, and Li Niu. Dreamcom: Finetuning text-guided inpainting model for image composition. *arXiv preprint arXiv:2309.15508*, 2023a.
- Shilin Lu, Yanzhu Liu, and Adams Wai-Kin Kong. Tf-icon: Diffusion-based training-free cross-domain image composition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2294–2305, 2023b.
- Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 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. *arXiv preprint arXiv:2307.01952*, 2023.
- Yifan Pu, Yiming Zhao, Zhicong Tang, Ruihong Yin, Haoxing Ye, Yuhui Yuan, Dong Chen, Jianmin Bao, Sirui Zhang, Yanbin Wang, et al. Art: Anonymous region transformer for variable multi-layer transparent image generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 7952–7962, 2025.
- Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.
- Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.
- Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.
- Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.
- Jaskirat Singh, Jianming Zhang, Qing Liu, Cameron Smith, Zhe Lin, and Liang Zheng. Smartmask: Context aware high-fidelity mask generation for fine-grained object insertion and layout control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6497–6506, 2024.
- Wensong Song, Hong Jiang, Zongxing Yang, Ruijie Quan, and Yi Yang. Insert anything: Image insertion via in-context editing in dit. *arXiv preprint arXiv:2504.15009*, 2025.

-
- Yizhi Song, Zhifei Zhang, Zhe Lin, Scott Cohen, Brian Price, Jianming Zhang, Soo Ye Kim, and Daniel Aliaga. Objectstitch: Object compositing with diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18310–18319, 2023.
- Yizhi Song, Zhifei Zhang, Zhe Lin, Scott Cohen, Brian Price, Jianming Zhang, Soo Ye Kim, He Zhang, Wei Xiong, and Daniel Aliaga. Imprint: Generative object compositing by learning identity-preserving representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8048–8058, 2024.
- Gemma Canet Tarrés, Zhe Lin, Zhifei Zhang, He Zhang, Andrew Gilbert, John Collomosse, and Soo Ye Kim. Multitwine: Multi-object compositing with text and layout control. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 8094–8104, 2025.
- Onur Tasar, Clément Chadebec, and Benjamin Aubin. Controllable shadow generation with single-step diffusion models from synthetic data. *arXiv preprint arXiv:2412.11972*, 2024.
- Yoad Tewel, Rinon Gal, Dvir Samuel, Yuval Atzmon, Lior Wolf, and Gal Chechik. Add-it: Training-free object insertion in images with pretrained diffusion models. *arXiv preprint arXiv:2411.07232*, 2024.
- Petru-Daniel Tudosiu, Yongxin Yang, Shifeng Zhang, Fei Chen, Steven McDonagh, Gerasimos Lampouras, Ignacio Iacobacci, and Sarah Parisot. Mulan: A multi layer annotated dataset for controllable text-to-image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22413–22422, 2024.
- Weicheng Wang, Guoli Jia, Zhongqi Zhang, Liang Lin, and Jufeng Yang. Ps-diffusion: Photorealistic subject-driven image editing with disentangled control and attention. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18302–18312. IEEE Computer Society, 2025.
- Navve Wasserman, Noam Rotstein, Roy Ganz, and Ron Kimmel. Paint by inpaint: Learning to add image objects by removing them first. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 18313–18324, 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, 2024a.
- Daniel Winter, Asaf Shul, Matan Cohen, Dana Berman, Yael Pritch, Alex Rav-Acha, and Yedid Hoshen. Objectmate: A recurrence prior for object insertion and subject-driven generation. *arXiv preprint arXiv:2412.08645*, 2024b.
- Binxin Yang, Shuyang Gu, Bo Zhang, Ting Zhang, Xuejin Chen, Xiaoyan Sun, Dong Chen, and Fang Wen. Paint by example: Exemplar-based image editing with diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 18381–18391, 2023.
- Jinrui Yang, Qing Liu, Yijun Li, Soo Ye Kim, Daniil Pakhomov, Mengwei Ren, Jianming Zhang, Zhe Lin, Cihang Xie, and Yuyin Zhou. Generative image layer decomposition with visual effects. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 7643–7653, 2025.
- Jingfeng Yao, Xinggang Wang, Shusheng Yang, and Baoyuan Wang. Vitmatte: Boosting image matting with pre-trained plain vision transformers. *Information Fusion*, 103:102091, 2024.
- Yongsheng Yu, Ziyun Zeng, Haitian Zheng, and Jiebo Luo. Omnipaint: Mastering object-oriented editing via disentangled insertion-removal inpainting. *arXiv preprint arXiv:2503.08677*, 2025.
- Jooyeol Yun, Davide Abati, Mohamed Omran, Jaegul Choo, Amirhossein Habibian, and Auke Wiggers. Generative location modeling for spatially aware object insertion. *arXiv preprint arXiv:2410.13564*, 2024.
- Bo Zhang, Yuxuan Duan, Jun Lan, Yan Hong, Huijia Zhu, Weiqiang Wang, and Li Niu. Controlcom: Controllable image composition using diffusion model. *arXiv preprint arXiv:2308.10040*, 2023a.

648 Lvmin Zhang and Maneesh Agrawala. Transparent image layer diffusion using latent transparency.
649 *arXiv preprint arXiv:2402.17113*, 2024.
650

651 Xinyang Zhang, Wentian Zhao, Xin Lu, and Jeff Chien. Text2layer: Layered image generation using
652 latent diffusion model. *arXiv preprint arXiv:2307.09781*, 2023b.

653 Jixin Zhao, Shangchen Zhou, Zhouxia Wang, Peiqing Yang, and Chen Change Loy. Objectclear:
654 Complete object removal via object-effect attention. *arXiv preprint arXiv:2505.22636*, 2025.
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701

A APPENDIX

In this appendix, we provide:

1. Related Work: Object Insertion	14
2. Implementation Details	14
3. Simulation-Based Synthetic Dataset Construction Pipeline	14
4. Shadow Generation Network G_{shadow}	16
5. Latent Composition Network Φ	16
6. Extended Experimental Results	17
7. LLM Usage Disclosure	18

A.1 RELATED WORK: OBJECT INSERTION

Another closely related line of research is object insertion, which aims to integrate an object into a target scene while ensuring object-level plausibility (pose, scale, and identity) and scene-level consistency (lighting, shadows, reflections, and other visual effects). Recently, large-scale image generative models have motivated *single framework solutions* that address all the subproblems simultaneously, directly producing composite images in a single synthesis. Among them, training-free methods (Lu et al., 2023b; Tewel et al., 2024) demonstrate that high-quality insertion results can be achieved by guiding the diffusion model generation process. A more mainstream direction is to fine-tune diffusion models to insert a reference image into a given background. Early studies (Yang et al., 2023; Song et al., 2023) replace text embeddings with embeddings of a given reference image to reproduce the target instance, while subsequent methods (Zhang et al., 2023a; Chen et al., 2024; Song et al., 2024) place greater emphasis on identity preservation by devising more effective identity injection strategies.

More recent researches (Huang et al., 2025c; He et al., 2024; Song et al., 2025; Canet Tarrés et al., 2024; Liang et al., 2024; Wang et al., 2025; Winter et al., 2024a; Canberk et al., 2024; Yu et al., 2025) have shifted their focus toward enhancing the realism of generated composites. To this end, some works (Huang et al., 2025c; He et al., 2024; Song et al., 2025; Canet Tarrés et al., 2024) curate dataset that explicitly capture object–scene interactions. Others, such as ObjectDrop (Winter et al., 2024a), EraseDraw (Canberk et al., 2024), ORIDa (Kim et al., 2025), and Ompaint (Yu et al., 2025), leverage object removal data to strengthen the fidelity of inserted objects. Meanwhile, Multitwine (Tarrés et al., 2025) extends the task to multi-object insertion, and ObjectMate (Winter et al., 2024b) and DreamCom (Lu et al., 2023a) leverage multiple reference images to utilize richer view-dependent information. Finally, layout estimation and then inpaint approaches, such as SmartMask (Singh et al., 2024) and Generative Location Modeling (Yun et al., 2024), first estimate insertion layouts and exploit pretrained inpainting models to generate natural composite results.

Despite these efforts, most object insertion methods concentrate on composite image generation rather than learning disentangled representations. As a result, the inserted objects cannot be easily separated, reused, or independently edited after synthesis, limiting post-hoc editing flexibility. In contrast, our approach explicitly generates a foreground RGBA layer preserving contextual realism while providing an editable and reusable representation of the inserted object.

A.2 IMPLEMENTATION DETAILS

For inference, we use the flow-matching Euler discrete sampler with 50 steps and a guidance scale of 30. We summarize the training configurations for both training stages of BFS in Tab. A1.

A.3 SIMULATION-BASED SYNTHETIC DATASET CONSTRUCTION PIPELINE

We describe in detail the pipeline for constructing our simulation-based synthetic dataset used in the training Stage 1, with visual examples in Fig. A1. We begin with the RORem (Li et al., 2025) dataset, an object removal dataset that contains a composite image C , a rough binary mask M of an object in the composite, and a background image B where the object has been removed together with its visual effects.

Table A1: Training setup for Stage 1 and Stage 2 of BFS.

Item	Stage 1	Stage 2
Datasets	Synthetic dataset from the RORem dataset	RORem dataset
Number of Samples	57,304	114,352
Image Resolution	512×512	512×512
RANK	64	64
Optimizer	AdamW 8bit ($\beta_1 = 0.9, \beta_2 = 0.999$)	AdamW 8bit ($\beta_1 = 0.9, \beta_2 = 0.999$)
Learning Rate	$8e - 6$	$8e - 6$
Weight Decay	$1e - 2$	$1e - 2$
Batch Size	32	32
Training Steps	3,000	9,000
LR Schedule	Cosine with restarts	Cosine with restarts
Gradient Clipping	1.0	1.0
Hardware	4×A100 80GB	4×A100 80GB
Precision	BF16	BF16

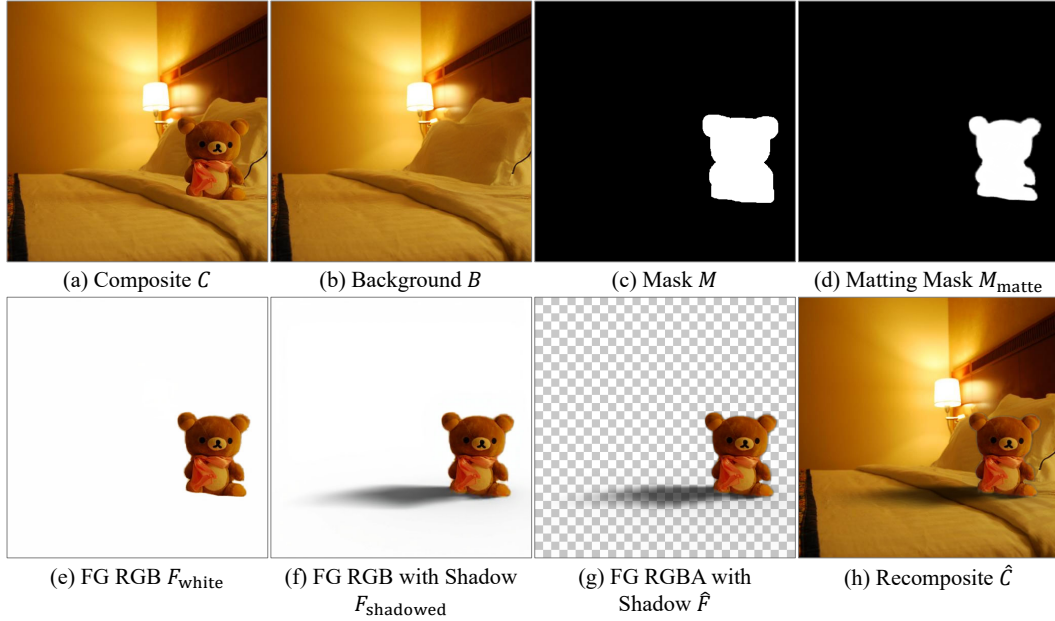


Figure A1: Visual examples from the pipeline used to construct the synthetic dataset.

Starting from the rough mask M , we first convert it into a bounding-box form M_{bbox} and apply SAM2 (Ravi et al., 2024) to detect an object mask M_{sam} based on the bounding-box mask M_{bbox} . Since SAM2 may capture regions misaligned with the original rough mask M , we filter out cases where the intersection-over-union (IoU) between the two masks (M and M_{sam}) is below 0.8, resulting in 57,304 valid samples out of the initial 114,352 images. For the retained masks, we perform an 11-pixel morphological dilation and erosion operations to construct a trimap, where pixels outside the dilated region are set to 0, those inside the eroded region to 1, and the intermediate band to 0.5. Based on the trimap, we estimate a detailed fine mask M_{matte} using a image matting method, ViTMatte (Yao et al., 2024). Based on the matting mask M_{matte} , we extract the foreground object from the composite C and place it on a white background (F_{white}). We then apply a shadow generation network G_{shadow} to synthesize shadows cast by the object, producing a shadow-augmented foreground F_{shadowed} (the details of G_{shadow} are provided in Sec. A.4).

Lastly, we compute the alpha channel to form the final RGBA representation $\hat{F}$ from the shadow-augmented foreground F_{shadowed} . Specifically, we assume that the synthesized shadow can be represented as the product of the shadow opacity α_{shadow} and a pure black image I_{black} . Accordingly,

F_{shadowed} can be expressed as

$$F_{\text{shadowed}} = \alpha_F F + \alpha_{\text{shadow}} I_{\text{black}} + (1 - \alpha_F - \alpha_{\text{shadow}}) I_{\text{white}} \quad (10)$$

where F denotes the foreground object, α_F is the object opacity, and I_{white} is the pure white background. Substituting $I_{\text{black}} = 0$ and $I_{\text{white}} = 1$ simplifies the equation to

$$F_{\text{shadowed}} = \alpha_F F + 1 - \alpha_F - \alpha_{\text{shadow}}, \quad (11)$$

which can be rearranged as

$$\alpha_F + \alpha_{\text{shadow}} = \alpha_F F + 1 - F_{\text{shadowed}}. \quad (12)$$

The left-hand side represents the combined opacity of the object and its shadow, which we denote as α . The right-hand side is fully computable, since $\alpha_F F$ corresponds to F_{black} , i.e., the object placed on a black background. Based on this, we can derive the foreground RGBA representation with the computed alpha α as

$$\hat{F} = (\hat{F}_{\text{color}}, \alpha), \quad (13)$$

where the color component is obtained by normalizing the shadowed foreground with its opacity,

$$\hat{F}_{\text{color}} = \frac{F_{\text{shadowed}} - (1 - \alpha)}{\alpha + \epsilon}, \quad (14)$$

and ϵ is a small constant to avoid division by zero. Finally, the recomposite image can be produced by alpha blending the RGBA foreground with the background B :

$$\hat{C} = \alpha \cdot \hat{F}_{\text{color}} + (1 - \alpha) \cdot B. \quad (15)$$

A.4 SHADOW GENERATION NETWORK

We introduce our shadow generation network G_{shadow} , which takes as input an object composited on a white background together with a fine-grained object mask, and outputs a shadow-casted image. Since no publicly available model and dataset align with our target scenario, we train our own network using a two-stage fine-tuning strategy.

In Stage 1, the model is trained to generate shadows that harmonize with the surrounding scene, given an input image and a fine-grained object mask. For this purpose, we leverage the large-scale DESOBAv2 dataset (Liu et al., 2024), which consists of 28,573 triplets containing (i) an object copy-pasted onto a background without shadows, (ii) the corresponding scene with naturally cast shadows, and (iii) a fine-grained object mask. We fine-tune the Flux backbone (Labs, 2024) by attaching a LoRA adapter, modifying the input layer to jointly process the image and mask, and optimizing the LoRA parameters with an ℓ_2 loss between the model outputs and the ground-truth shadowed images.

Although the Stage 1 model can produce realistic shadows, its outputs are not directly aligned with our target scenario, where objects are placed on a uniform white background. To address this mismatch, Stage 2 performs additional fine-tuning on a smaller dataset (Tasar et al., 2024) specifically tailored to white-background scenes. This dataset is constructed using Blender and provides 555 pairs of (i) an object composited on a white background with rendered shadows and (ii) a fine-grained object mask. From these pairs, we construct training triplets by extracting the object using the mask and placing it onto a white background. The network is trained with the same ℓ_2 loss as in Stage 1. This refinement enables the model to generate shadows consistent with objects placed on a white background.

Through this two-stage training, we obtain a robust shadow generation network, which serves as a crucial component in constructing the simulation-based dataset used in our synthetic dataset construction pipeline. We summarize the training configurations for both stages in Tab. A2.

A.5 LATENT COMPOSITION NETWORK

In this section, we introduce the latent composition network Φ used in the training Stage 1. Learning the compositional relationship where the output of the composite branch should equal the composition of the input background latent and the output of the foreground branch is crucial, as also demonstrated in the ablation study of the main paper.

Table A2: Training setup for Stage 1 and Stage 2 of the shadow generation network G_{shadow} .

Item	Stage 1	Stage 2
Datasets	DESOVAv2 (Liu et al., 2024)	Tasar et al. (2024)
Number of Samples	28,573	555
Image Resolution	512×512	512×512
RANK	128	128
Optimizer	AdamW 8bit ($\beta_1 = 0.9, \beta_2 = 0.999$)	AdamW 8bit ($\beta_1 = 0.9, \beta_2 = 0.999$)
Learning Rate	$3e - 5$	$3e - 5$
Weight Decay	$1e - 2$	$1e - 2$
Batch Size	32	32
Training Steps	22,000	2,000
LR Schedule	Cosine with restarts	Cosine with restarts
Gradient Clipping	1.0	1.0
Hardware	1×A100 80GB	1×A100 80GB
Precision	BF16	BF16

However, since both branches operate in the latent space of the pretrained VAE rather than directly in the image domain, the standard alpha blending equation cannot be applied. Furthermore, the encoding and decoding of the foreground latent rely on the RGBA VAE introduced in LayerDiffuse (Zhang & Agrawala, 2024), which projects transparency information into the latent representation, meaning that no explicit blending mask is available. While one could follow the approach of LayerDecomp (Yang et al., 2025) and design a composition loss in the image domain using the VAE decodings of the generated latents of both branches, this requires backpropagation through the VAE decoder. Such a strategy is computationally expensive and carries the risk of inaccurate gradient propagation due to the long backpropagation path.

To address this, we introduce the latent composition network Φ , which performs composition directly in the latent space. This network adopts a simple U-Net architecture and is trained to map the channel-wise concatenation of a foreground latent and a background latent to the corresponding composite latent. To this end, the training objective is defined as the reconstruction error between the predicted composite latent $\hat{z}_C$ and the ground-truth composite latent z_C :

$$\mathcal{L}_{\text{comp}} = \|z_C - \Phi(z_B, z_F)\|_2^2. \quad (16)$$

Below, we provide the code for constructing the network architecture using the Diffusers library, along with details of the training setup (Tab. A3).

```

from diffusers import UNet2DModel
composition_net = UNet2DModel(
    sample_size=128,          # latent resolution
    in_channels=32,           # foreground + background latents channels
    out_channels=16,          # composite latent channels
    layers_per_block=2,
    block_out_channels=[320, 640],
    down_block_types=("DownBlock2D", "DownBlock2D"),
    up_block_types=("UpBlock2D", "UpBlock2D"),
)

```

A.6 EXTENDED EXPERIMENTAL RESULTS

We present additional qualitative comparison examples with layered image synthesis methods in Fig. A2 and Fig. A3.

We present additional qualitative comparison examples with inpainting methods in Fig. A4, Fig. A5, Fig. A6, Fig. A7, and Fig. A8.

We present additional quantitative comparisons on the RORem dataset (Li et al., 2025), comparing existing layered image synthesis methods in Tab. A4, and object insertion methods in Tab. A5.

We present additional quantitative comparisons on the AnyInsertion dataset (Song et al., 2025), comparing existing layered image synthesis methods in Tab. A6, and object insertion methods in Tab. A7.

A.7 LLM USAGE DISCLOSURE

We used OpenAI’s ChatGPT as a writing assistant during the preparation of this paper. Specifically, ChatGPT was employed for language refinement, grammar correction, and improving the clarity of exposition. It did not contribute to research ideation, experimental design, data analysis, or the derivation of technical results. All scientific content, including the conceptual framework, methodology, experiments, and conclusions, was created entirely by the authors.

Table A3: Training setup for the latent composition network Φ .

Item	Setting
Datasets	Foreground from MAGICK dataset (Burgert et al., 2024) Background from BG20k dataset (Li et al., 2022)
Number of Samples	20,000
Latent Resolution	64×64
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.999$)
Learning Rate	5×10^{-6}
Weight Decay	$1e - 2$
Batch Size	32
LR Schedule	Cosine with restarts
Training Steps	10,000
Gradient Clipping	1.0
Hardware	1×A100 80GB
Precision	Float32

Table A4: Quantitative evaluation of existing layered image synthesis methods on RORem dataset.

Method	FG	Recomposite (or Background for LayeringDiff + ObjectClear)			
	CLIP ($\uparrow$)	MUSIQ ($\uparrow$)	MANIQA ($\uparrow$)	KID×1000 ($\downarrow$)	DINO ($\uparrow$)
LayeringDiff + ObjectClear	0.63	66.98	0.41	5.19	0.99
LayerDiffuse	0.77	70.05	0.43	9.04	0.50
Ours	0.70	68.34	0.43	4.73	0.94

Table A5: Quantitative evaluation with an object insertion method FluxFill and an image inpainting method PaintByInpaint on RORem dataset.

Method	MUSIQ ($\uparrow$)	MANIQA ($\uparrow$)	KID×1000 ($\downarrow$)	DINO ($\uparrow$)
PaintByInpaint (Wasserman et al., 2025)	68.08	0.38	4.60	0.93
FluxFill (Labs, 2024)	67.72	0.42	4.59	0.93
Ours	68.34	0.43	4.73	0.94

Table A6: Quantitative evaluation of existing layered image synthesis methods on AnyInsertion dataset.

Method	FG	Recomposite (or Background for LayeringDiff + ObjectClear)			
	CLIP ($\uparrow$)	MUSIQ ($\uparrow$)	MANIQA ($\uparrow$)	KID×1000 ($\downarrow$)	DINO ($\uparrow$)
LayeringDiff + ObjectClear	0.70	54.18	0.35	10.47	0.71
LayerDiffuse	0.73	65.36	0.40	8.51	0.41
Ours	0.71	55.54	0.32	3.88	0.45

Table A7: Quantitative evaluation with an object insertion method FluxFill and an image inpainting method PaintByInpaint on AnyInsertion dataset.

Method	MUSIQ ($\uparrow$)	MANIQA ($\uparrow$)	KID $\times$ 1000 ($\downarrow$)	DINO ($\uparrow$)
PaintByInpaint (Wasserman et al., 2025)	58.10	0.33	5.49	0.49
FluxFill (Labs, 2024)	55.01	0.33	3.02	0.56
Ours	55.54	0.32	3.88	0.45

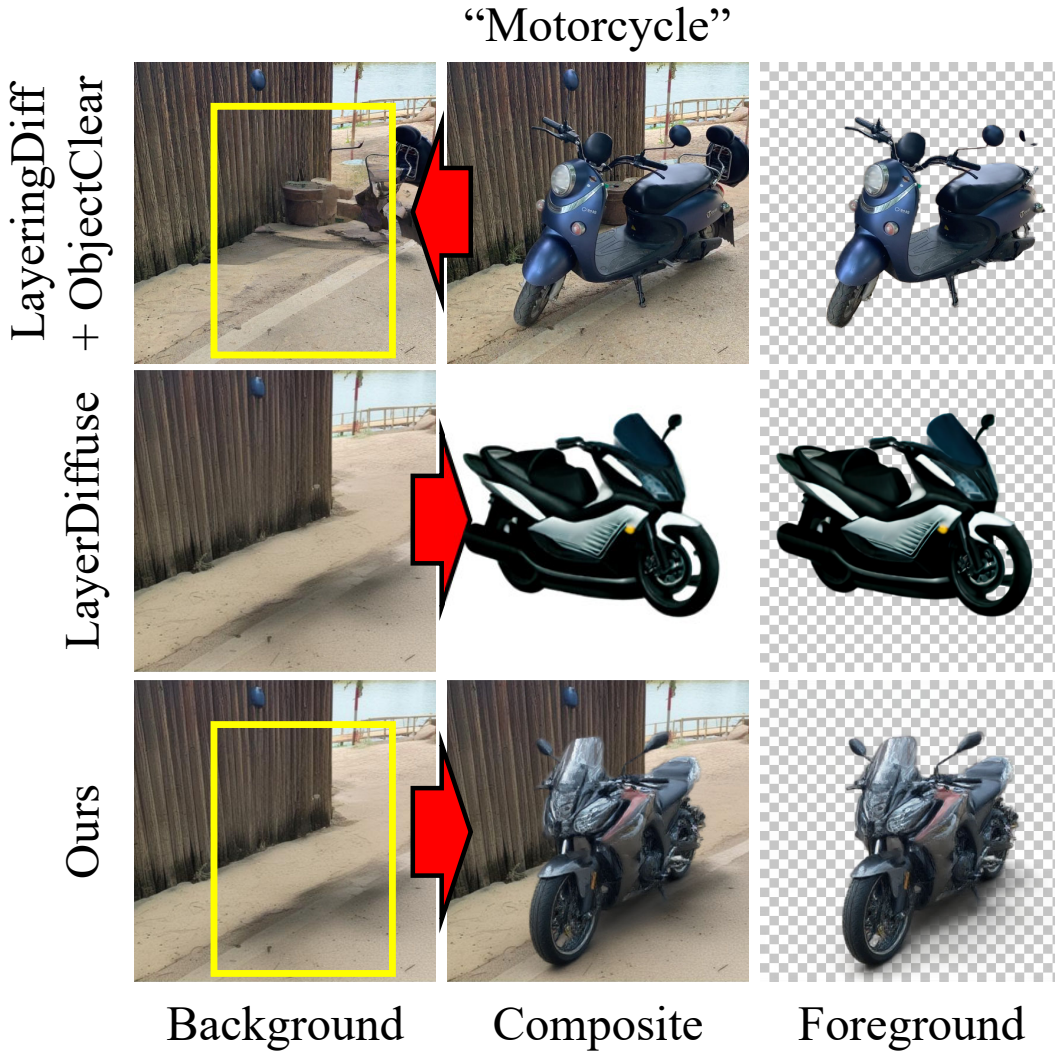


Figure A2: Qualitative comparison with existing layered image synthesis approaches.

1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079

“The new opel astra hatchback is shown in yellow”

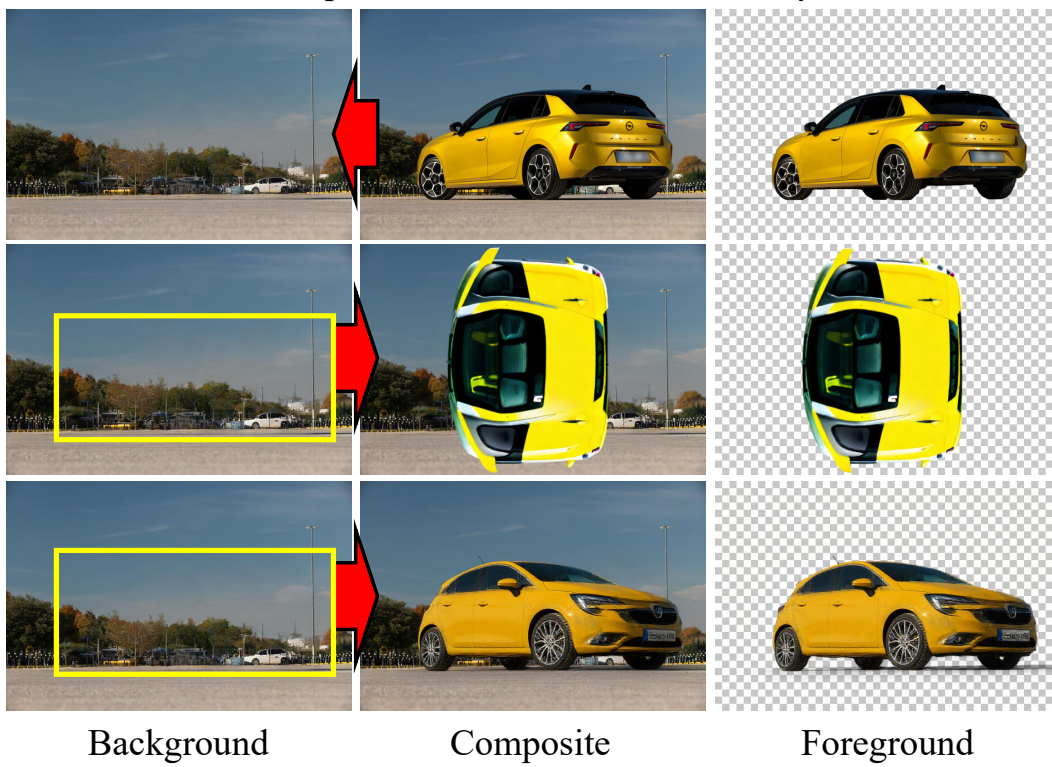


Figure A3: Qualitative comparison with existing layered image synthesis approaches.

1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133

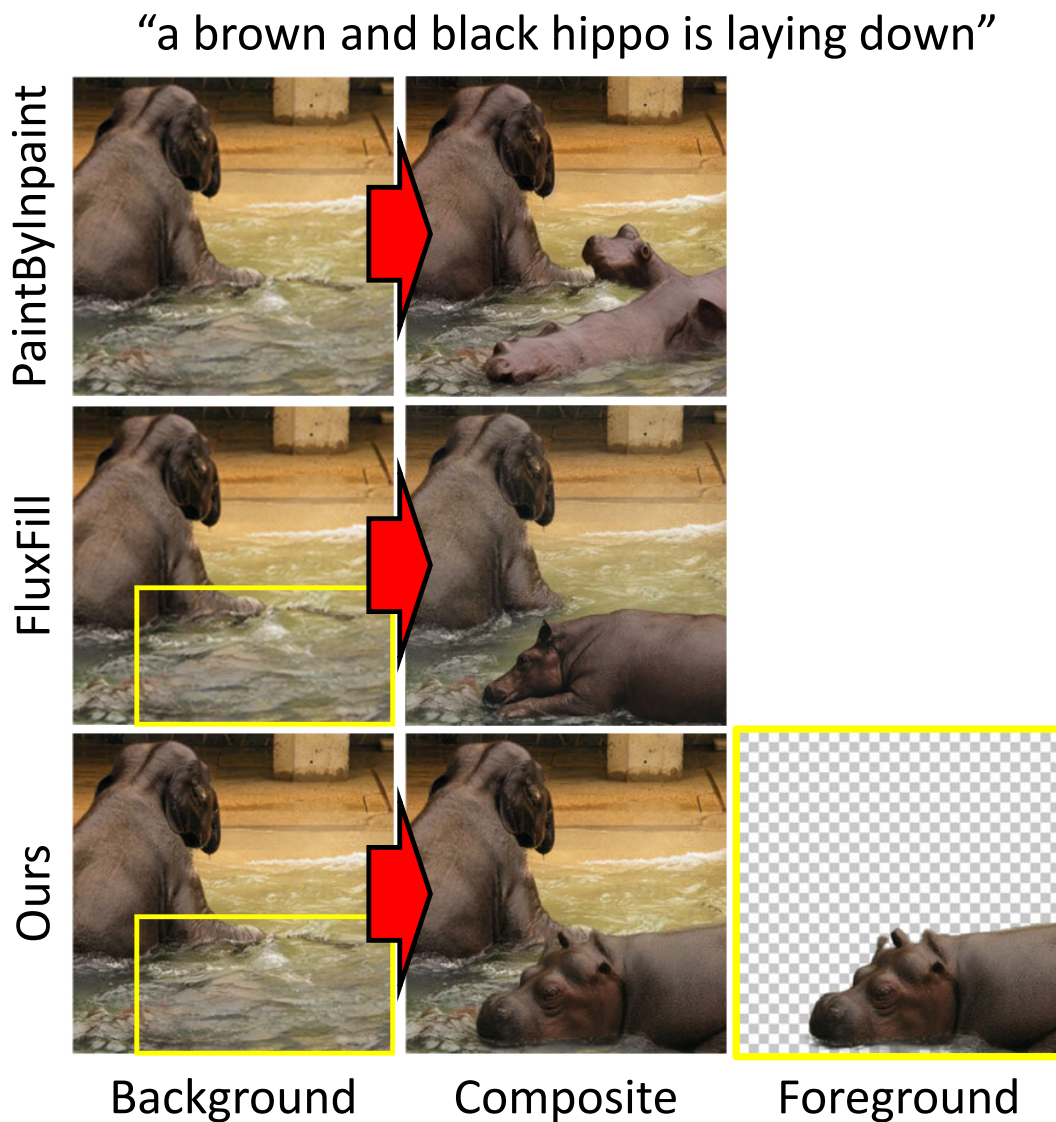


Figure A4: Qualitative comparison with an object insertion method PaintByInpaint (Wasserman et al., 2025) and a image inpainting method FluxFill (Labs, 2024)

1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187

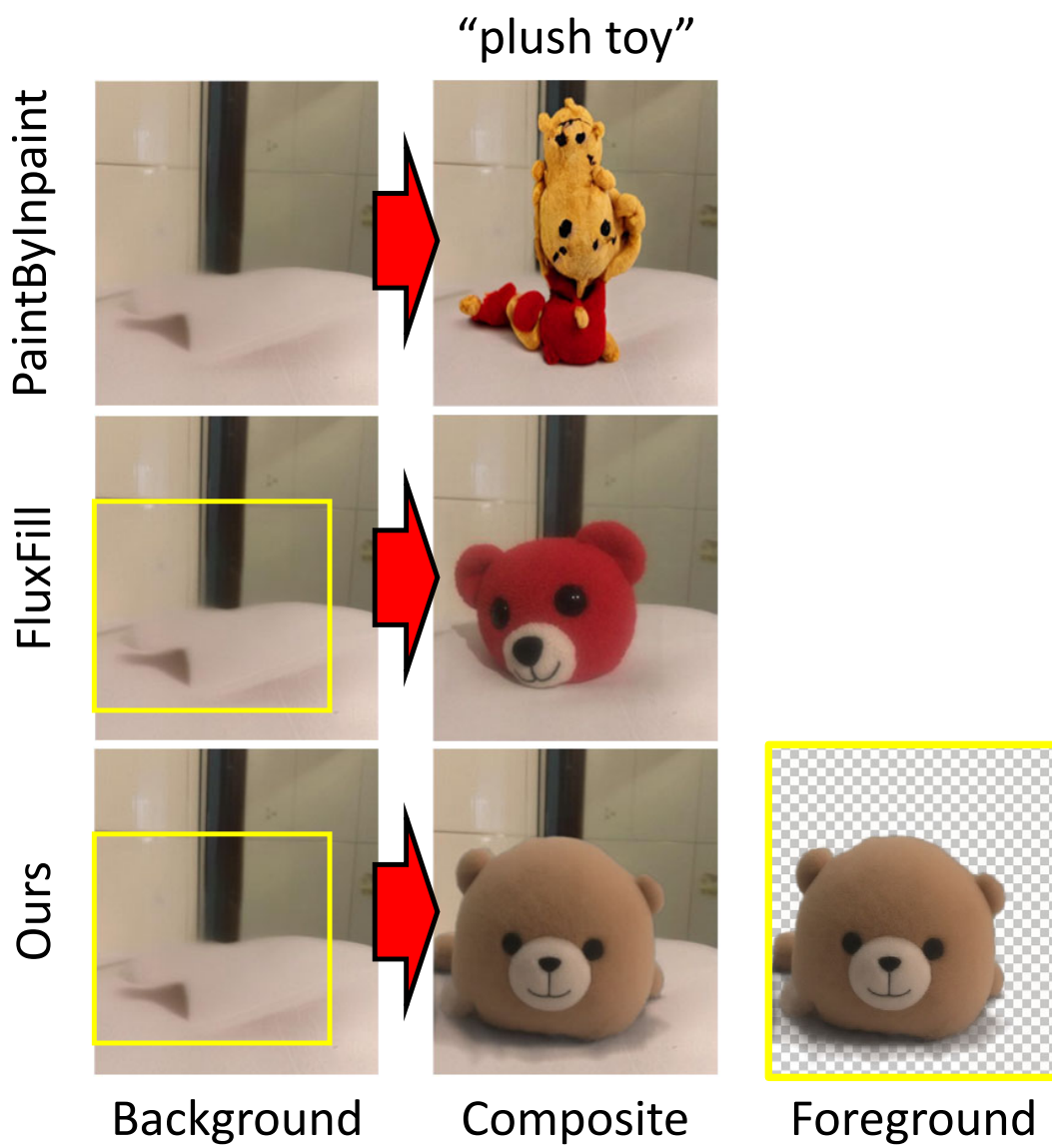


Figure A5: Qualitative comparison with an object insertion method PaintByInpaint (Wasserman et al., 2025) and a image inpainting method FluxFill (Labs, 2024)

1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241

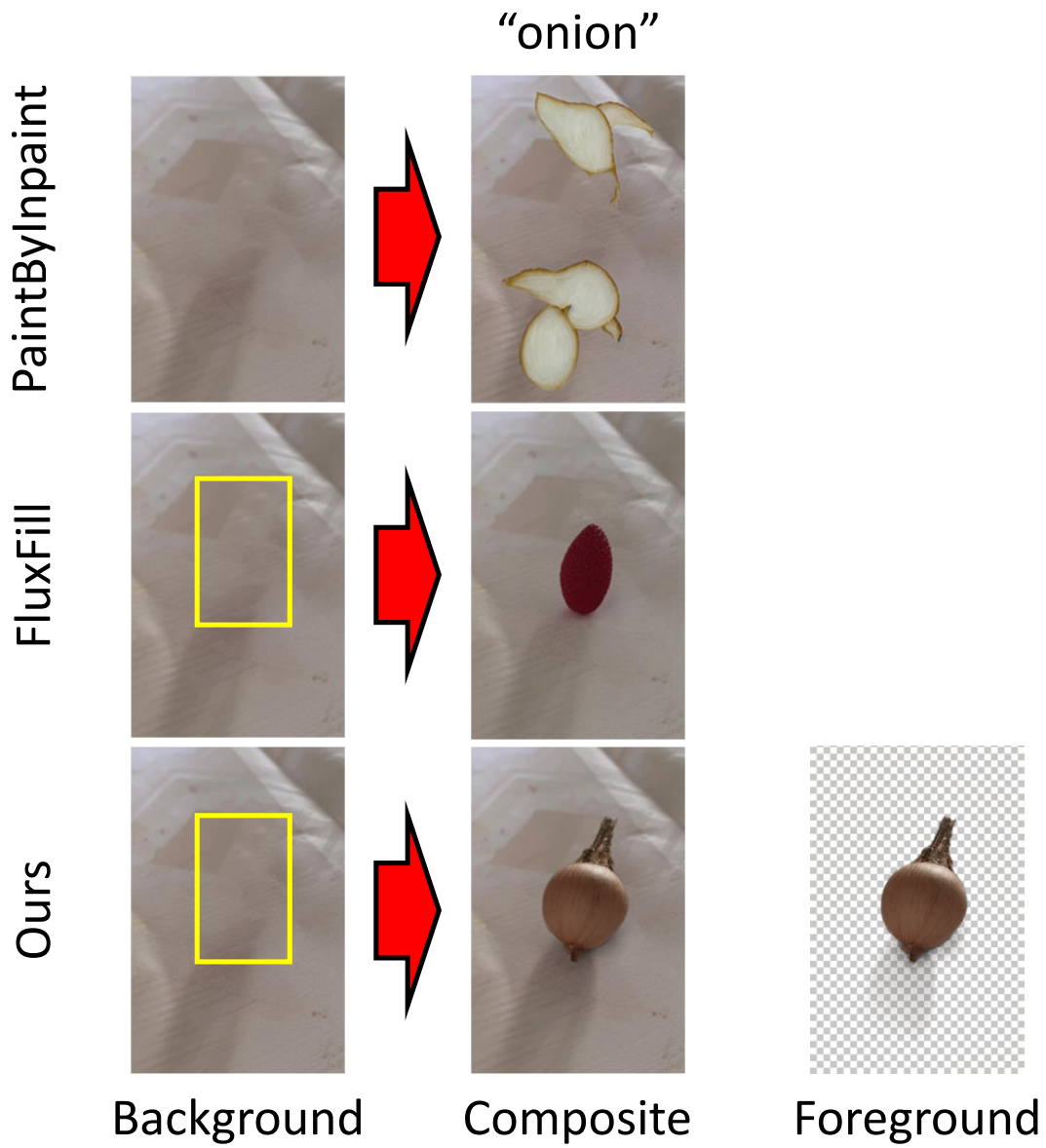


Figure A6: Qualitative comparison with an object insertion method PaintByInpaint (Wasserman et al., 2025) and a image inpainting method FluxFill (Labs, 2024)

1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295

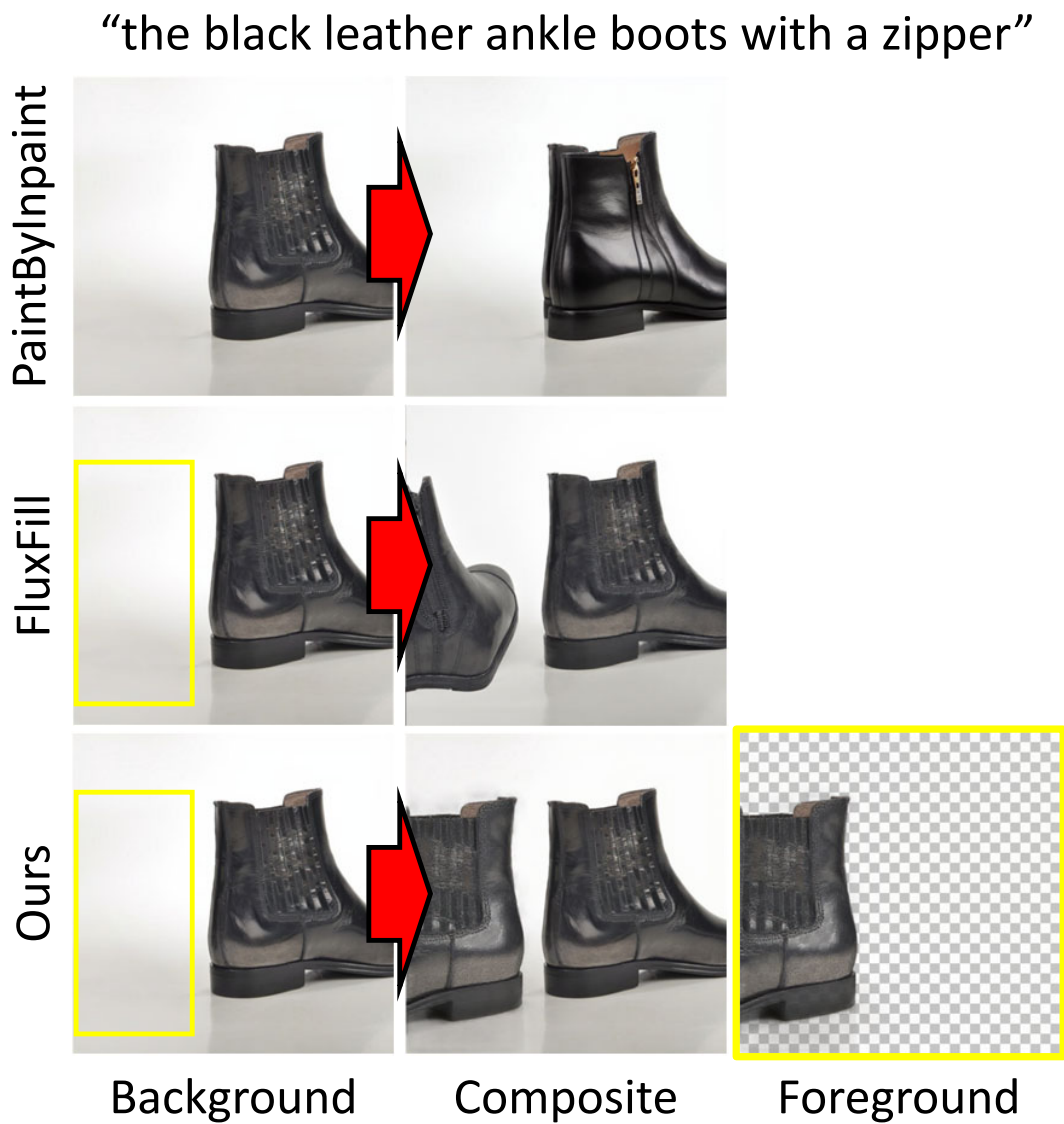


Figure A7: Qualitative comparison with an object insertion method PaintByInpaint (Wasserman et al., 2025) and a image inpainting method FluxFill (Labs, 2024)

1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349

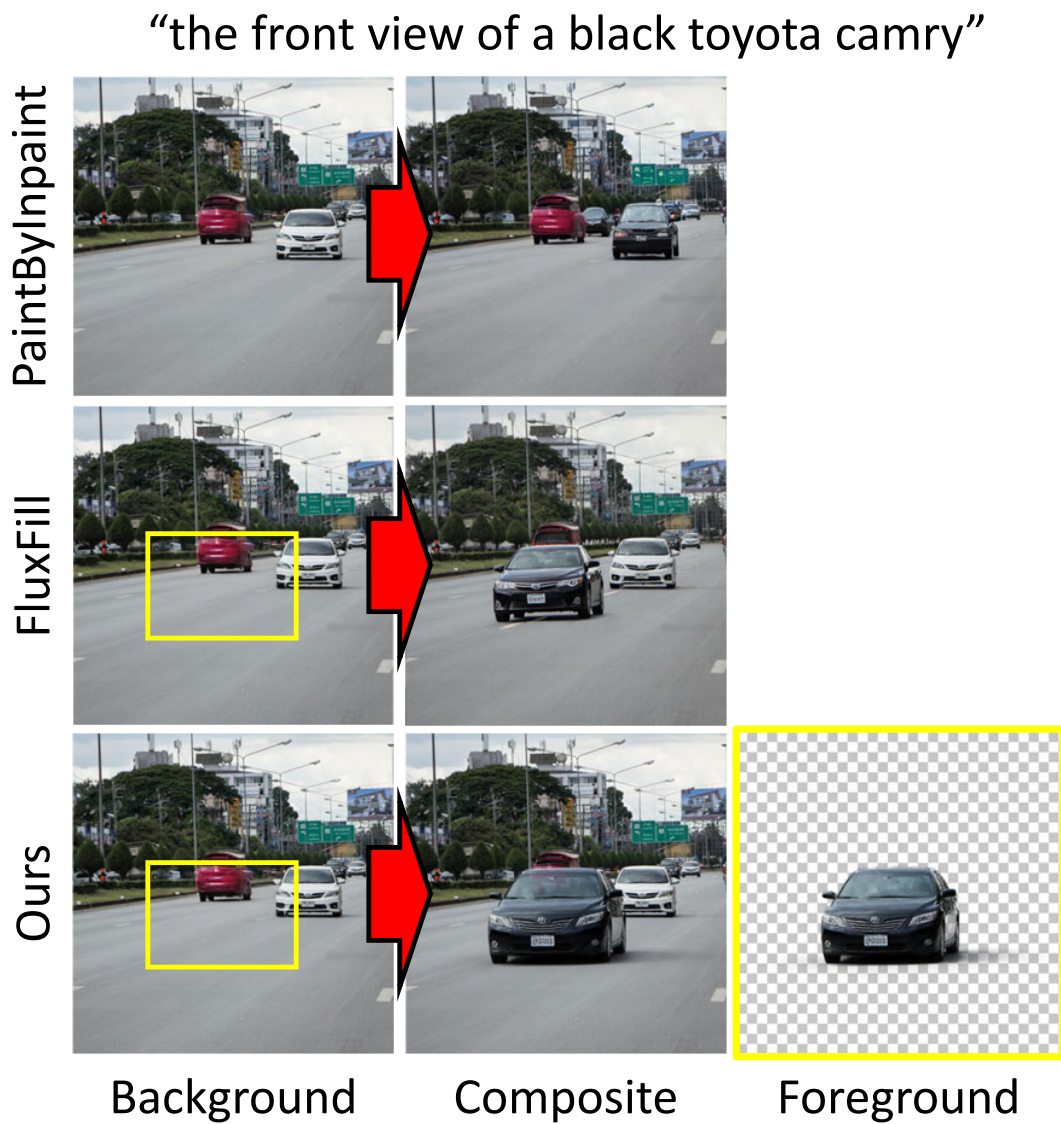


Figure A8: Qualitative comparison with an object insertion method PaintByInpaint (Wasserman et al., 2025) and a image inpainting method FluxFill (Labs, 2024)