TIP-Editor: An Accurate 3D Editor Following Both Text-Prompts And Image-Prompts

JINGYU ZHUANG*, Sun Yat-sen University, China and Tencent AI Lab, China
DI KANG, Tencent AI Lab, China
YAN-PEI CAO, Tencent AI Lab, China
GUANBIN LI†, Sun Yat-sen University, China and Peng Cheng Laboratory, China
LIANG LIN, Sun Yat-sen University, China and Peng Cheng Laboratory, China
YING SHAN, Tencent AI Lab, China

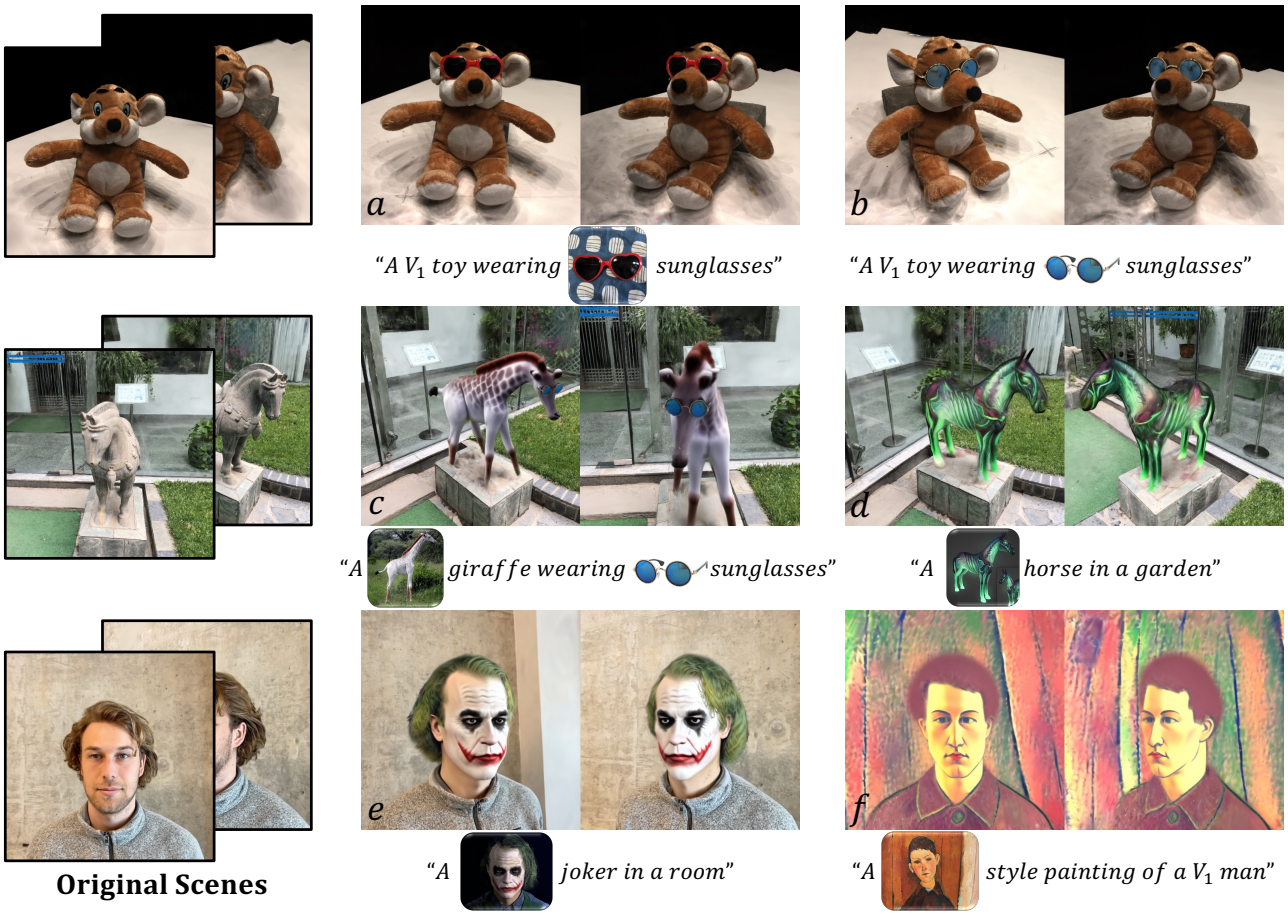

Fig. 1. TIP-Editor excels in precise and high-quality localized editing given a 3D bounding box, and allows the users to perform various types of editing on a 3D scene, such as object insertion (a, b), whole object replacement (d), part-level object editing (e), combination of these editing types (i.e. sequential editing, (c)), and stylization (f). The editing process is guided by not only the text but also one reference image, which serves as the complement of the *textual* description and results in more accurate editing control. Images in the text prompts denote their associated *rare tokens*, which are fixed without optimization.

*Work done during an internship at Tencent AI Lab.
†Corresponding author. Welcome to *Project page*

Authors' Contact Information: Jingyu Zhuang, zhuangjy6@mail2.sysu.edu.cn, Sun Yat-sen University, China and Tencent AI Lab, China; Di Kang, di.kang@outlook.com, Tencent AI Lab, China; Yan-Pei Cao, caoyanpei@gmail.com, Tencent AI Lab, China; Guanbin Li, liguanbin@mail.sysu.edu.cn, Sun Yat-sen University, China and Peng Cheng Laboratory, China; Liang Lin, linliang@ieee.org, Sun Yat-sen University, China and Peng Cheng Laboratory, China; Ying Shan, yingsshan@tencent.com, Tencent AI Lab, China.

Text-driven 3D scene editing has gained significant attention owing to its convenience and user-friendliness. However, existing methods still lack accurate control of the specified appearance and location of the editing result due to the inherent limitations of the text description. To this end, we propose a 3D scene editing framework, TIP-Editor, that accepts both **t**ext and **i**mage **p**rompts and a 3D bounding box to specify the editing region. With the image prompt, users can conveniently specify the detailed appearance/style of the target content in complement to the text description, enabling accurate control of the appearance. Specifically, TIP-Editor employs a stepwise 2D personalization strategy to better learn the representation of the existing scene and the reference image, in which a localization loss is proposed to encourage correct object placement as specified by the bounding box. Additionally, TIP-Editor utilizes explicit and flexible 3D Gaussian splatting (GS) as the 3D representation to facilitate local editing while keeping the background unchanged. Extensive experiments have demonstrated that TIP-Editor conducts accurate editing following the text and image prompts in the specified bounding box region, consistently outperforming the baselines in editing quality, and the alignment to the prompts, qualitatively and quantitatively.

CCS Concepts: • **Computing methodologies** → **Rendering**.

Additional Key Words and Phrases: 3D Gaussian splatting, Scene editing

ACM Reference Format:
Jingyu Zhuang, Di Kang, Yan-Pei Cao, Guanbin Li, Liang Lin, and Ying Shan. 2024. TIP-Editor: An Accurate 3D Editor Following Both Text-Prompts And Image-Prompts. *ACM Trans. Graph.* 43, 4, Article 121 (July 2024), 12 pages. https://doi.org/10.1145/3658205

1 INTRODUCTION

Due to the unprecedented photorealistic rendering quality, methods that use radiance field-related representations (e.g. NeRF [Mildenhall et al. 2021] and 3D Gaussian Splatting [Kerbl et al. 2023]) have been more and more popular in 3D reconstruction field [Cao 2023; Li et al. 2021; Yu et al. 2021] and various downstream 3D editing tasks, such as texture editing [Richardson et al. 2023; Xiang et al. 2021], shape deformation [Xu and Harada 2022; Yang et al. 2022], scene decomposition [Tang et al. 2022], and stylization [Wang et al. 2023b].

Generative editing, which only requires high-level instructions (e.g. text prompts), emerges as a new approach in complement to previous painting-like and sculpting-like editing approaches [Xiang et al. 2021; Yang et al. 2022] that require *extensive* user interactions. Among these methods, text-driven methods [Haque et al. 2023; Zhuang et al. 2023] have gained significant attention due to their convenience and have achieved remarkable progress due to the success of large-scale text-to-image (T2I) models.

However, methods using only text as the condition struggle to precisely generate editing results with the specified appearance at the specified location due to the inherent limitations of the text description. For example, existing text-driven methods usually produce less satisfactory results (Fig. 6) if we want to dress the toy in a special heart-shaped sunglasses or give the male the Joker makeup appeared in the movie *The Dark Knight*. Moreover, it is hard to specify the accurate editing location by text guidance (Fig. 7). These challenges primarily stem from the diverse appearances of the generated objects and the diverse spatial layout of the generated scenes.

ACM 1557-7368/2024/7-ART121
https://doi.org/10.1145/3658205

To overcome the challenges above, we present TIP-Editor, which allows the users to intuitively, conveniently, and accurately edit the exiting GS-based (3D Gaussian Splatting [Kerbl et al. 2023]) radiance fields using both **t**ext prompts and **i**mage **p**rompts. Our framework achieves such capabilities through two crucial designs. (1) The first one is a novel stepwise 2D personalization strategy that enables precise appearance control (via a reference image) and location control (via a 3D bounding box). Specifically, it contains a scene personalization step, which includes a localization loss to ensure the editing occurs inside the user-defined editing region, and a separate novel content personalization step dedicated to the reference image based on LoRA [Hu et al. 2021]. (2) The second one is adopting explicit and flexible 3D Gaussian splatting [Kerbl et al. 2023] as the 3D representation since it is efficient and, more importantly, highly suitable for local editing.

We conduct comprehensive evaluations of TIP-Editor across various real-world scenes, including objects, human faces, and outdoor scenes. Our editing results (Fig. 1 and Fig. 3) successfully capture the unique characteristics specified in the reference images. This significantly enhances the controllability of the editing process, presenting considerable practical value. In both qualitative and quantitative comparisons, TIP-Editor consistently demonstrates superior performance in editing quality, visual fidelity, and user satisfaction when compared to existing methods.

Our contributions can be summarized as follows:

- We present TIP-Editor, a versatile 3D scene editing framework that allows the users to perform various editing operations (e.g. object insertion, object replacement, re-texturing, and stylization) guided by not only the text prompt but also by a reference image.
- We present a novel stepwise 2D personalization strategy, which features a localization loss in the scene personalization step and a separate novel content personalization step dedicated to the reference image based on LoRA, to enable accurate location and appearance control.
- We adopt 3D Gaussian splatting to represent scenes due to its rendering efficiency and, more importantly, its explicit point data structure, which is very suitable for precise local editing.

2 RELATED WORKS

2.1 Text-guided image generation and editing

Text-to-image (T2I) diffusion models [Ramesh et al. 2022; Rombach et al. 2022; Saharia et al. 2022], trained on large-scale paired image-text datasets, have gained significant attention since they can generate diverse and high-quality images that match the complicated text prompt. Instead of directly generating images from scratch, another popular and closely related task is to edit the given image according to the text prompt [Avrahami et al. 2022; Brooks et al. 2022; Couairon et al. 2022; Hertz et al. 2022; Kawar et al. 2022; Meng et al. 2021].

Another popular task is object/concept personalization, which aims at generating images for a specified object/concept defined in the given image collection. Textual Inversion (TI) [Gal et al. 2022] optimizes special text token(s) in the text embedding space to represent the specified concept. DreamBooth [Ruiz et al. 2022] fine-tunes the entire diffusion model with a class-specific prior preservation

loss as regularization. In general, DreamBooth generates higher-quality images since it involves a larger amount of updated model parameters (i.e. the whole UNet model). However, all the aforementioned methods do not support generating images containing multiple personalized objects simultaneously.

Custom Diffusion [Kumari et al. 2023] extends the above task to generate multiple personalized *concepts* in one image simultaneously. Although separate special text tokens are assigned to each *concept*, the UNet is updated by all *concepts*, resulting in less satisfactory personalization results. Furthermore, it lacks a localization mechanism to specify the interaction between two *concepts* (Fig. 8). In contrast, we propose a stepwise 2D personalization strategy to learn the existing scene and the new content separately, achieving high-quality and faithful personalization results and being generalizable to sequential editing scenarios.

2.2 Radiance field-based 3D object/scene generation

The success of T2I diffusion models has largely advanced the development of 3D object/scene generation. One seminal contribution, DreamFusion [Poole et al. 2022], introduces score distillation sampling (SDS), which distills knowledge from a pre-trained 2D T2I model to *optimize* a radiance field without the reliance on any 3D data. Most of the subsequent works adopt such an optimization-based pipeline and make further progresses by introducing an extra refinement stage (e.g., Magic3D [Lin et al. 2022] and DreamBooth3D [Raj et al. 2023]), or proposing more suitable SDS variants (e.g., VSD [Wang et al. 2023c]), or using more powerful 3D representations [Chen et al. 2023a,d; Yi et al. 2023], such as GS [Kerbl et al. 2023].

Furthermore, a body of research [Deng et al. 2022; Melas-Kyriazi et al. 2023; Qian et al. 2023; Tang et al. 2023] endeavors to integrate reference images within the optimization framework. This integration is facilitated by various techniques, including the application of reconstruction loss, employment of predicted depth maps, and the execution of a fine-tuning process. Nevertheless, these methods are constrained to generate a single object from scratch and cannot edit existing 3D scenes.

2.3 Radiance field-based 3D object/scene editing

Earlier works [Wang et al. 2022, 2023b] mainly focus on global style transformation of a given 3D scene, which takes text prompts or reference images as input and usually leverage a CLIP-based similarity measure [Radford et al. 2021] during optimization. Several studies enable local editing on generic scenes by utilizing 2D image manipulation techniques (e.g. inpainting) [Bao et al. 2023; Kobayashi et al. 2022; Liu et al. 2022] to obtain new training images to update the existing radiance field. Some works adopt 3D modeling techniques (e.g. mesh deformation [Wang et al. 2023a; Xu and Harada 2022; Yang et al. 2022; Yuan et al. 2022] or point clouds deformation [Chen et al. 2023c; Wu et al. 2023]) to propagate the shape deformation to the underlying radiance field. However, these methods require extensive user interactions.

Recently, text-driven radiance field editing methods have gained more and more attention for their editing flexibility and accessibility. For example, Instruct-NeRF2NeRF [Haque et al. 2023] and GenN2N [Liu et al. 2024] employ an image-based diffusion model (InstructPix2Pix [Brooks et al. 2022]) to modify the rendered image by the users' instructions, and subsequently update the 3D radiance field with the modified image. DreamEditor [Zhuang et al. 2023], 3D Paintbrush [Decatur et al. 2023] and Vox-E [Sella et al. 2023] enable better local editing by adopting explicit 3D representations (i.e. mesh and voxel), where the editing region is automatically determined by the 2D cross-attention maps. GaussianEditor [Chen et al. 2023b; Fang et al. 2023] adopts GS as the scene representation and incorporates 3D semantic segmentation [Cen et al. 2023; Kirillov et al. 2023b] to facilitate efficient and precise scene editing. However, these text-driven approaches lack precise control over the specified appearance and position of the editing results.

A concurrent work, CustomNeRF [He et al. 2023], is most related to our task setting. But CustomNeRF only supports the object replacement task, since it requires an object that can be detected by the segmentation tool [Kirillov et al. 2023a] existing in the implicit NeRF scene, as the editing target. In contrast, we adopt explicit GS as the 3D representation which facilitates our method to perform more editing tasks (e.g., object insertion and stylization).

3 BACKGROUND

3.1 3D Gaussian Splatting.

3D Gaussian Splatting (GS) [Kerbl et al. 2023] quickly draws tremendous attention due to its high rendering quality and efficiency. GS utilizes a set of point-like anisotropic Gaussians g_i to represent the scene: $\mathcal{G} = \{g_1, g_2, ..., g_N\}$. Each g_i contains a series of optimizable attributes, including center position $\mu \in \mathbb{R}^3$, opacity $\alpha \in \mathbb{R}^1$, 3D covariance matrix Σ, and color c. The differentiable splatting rendering process is outlined as follows:

$$C = \sum_{i \in \mathcal{N}} c_i \sigma_i \prod_{i-1}^{j=1} (1 - \sigma_j), \quad \sigma_i = \alpha_i G(x) = \alpha_i e^{-\frac{1}{2}(x)^T \Sigma^{-1}(x)}, \quad (1)$$

where j indexes the Gaussians in front of g_i according to their distances to the optical center in ascending order, $\mathcal{N}$ is the number of Gaussians that have contributed to the ray, and c_i, α_i, and x_i represent the color, density, and distance to the center point of the i-th Gaussian, respectively.

3.2 Optimizing Radiance Fields with SDS Loss.

Score distillation sampling (SDS) [Poole et al. 2022] optimizes a radiance field by distilling the priors from a Text-to-Image (T2I) diffusion model for 3D generation. The pre-trained diffusion model ϕ is used to predict the added noise given a noised image $\hat{I}_t$ and its text condition y.

$$\nabla_\theta \mathcal{L}_{SDS}(\phi, \hat{I} = f(\theta)) = \mathbb{E}_{\epsilon,t}\left[w(t)(\epsilon_\phi(\hat{I}_t; y, t) - \epsilon) \frac{\partial \hat{I}}{\partial \theta} \right], \quad (2)$$

where θ denotes the parameters of the radiance field, $f(\cdot)$ is the differentiable image formation process, and $w(t)$ is a predefined weighting function derived from noise level t.

4 METHOD

Given posed images (i.e., images and their associated camera parameters estimated by COLMAP [Schönberger and Frahm 2016]) of the

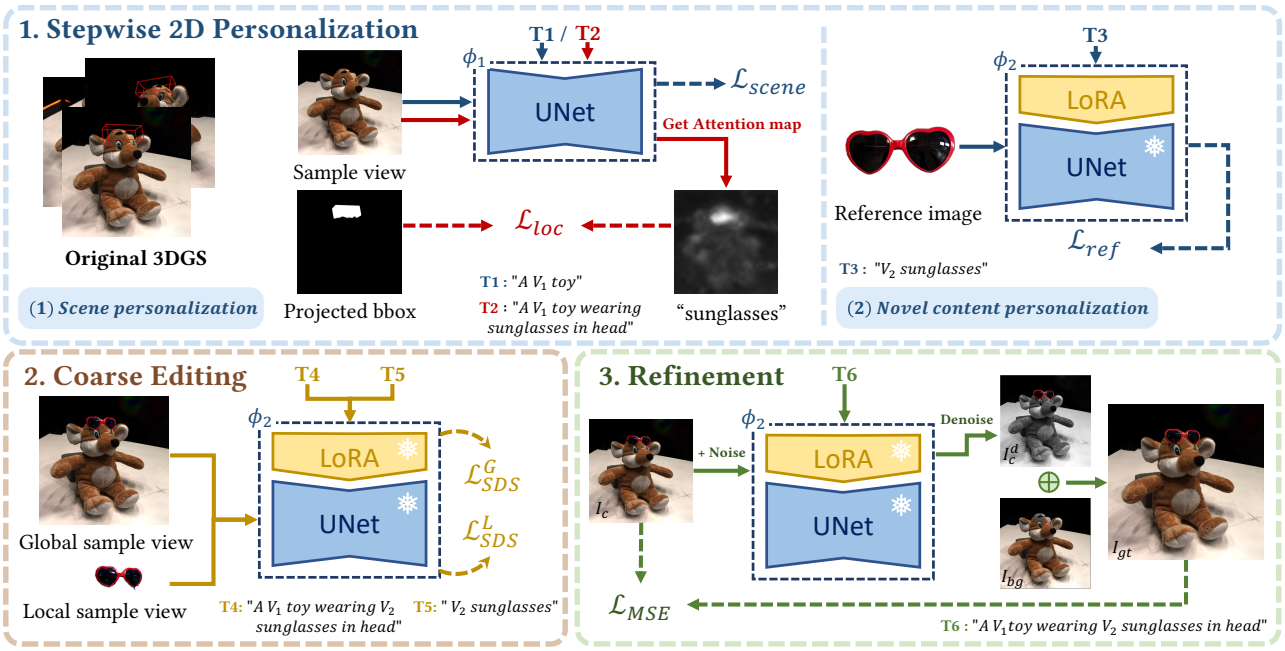

Fig. 2. **Method overview.** TIP-Editor optimizes a 3D scene that is represented as 3D Gaussian splatting (GS) to conform with a given hybrid text-image prompt. The editing process includes three stages: 1) a stepwise 2D personalization strategy, which features a localization loss in the scene personalization step and a separate novel content personalization step dedicated to the reference image based on LoRA (Sec. 4.1); 2) a coarse editing stage using SDS (Sec. 4.2); and 3) a pixel-level texture refinement stage, utilizing carefully generated pseudo-GT image from both the rendered image I_c and the denoised image I_c^d (Sec. 4.3).

target scene, our goal is to enable more accurate editing following a hybrid text-image prompt within a user-specified 3D bounding box. We choose 3D Gaussian splatting (GS) [Kerbl et al. 2023] to represent the 3D scene since GS is an explicit and highly flexible 3D representation method, which is beneficial for the following editing operations, especially local editing.

As shown in Fig. 2, TIP-Editor contains three major steps, including 1) a stepwise 2D personalization of the existing scene and the novel content (Sec. 4.1), 2) a coarse 3D editing stage using score distillation sampling (SDS) [Poole et al. 2022] (Sec. 4.2), and 3) a pixel-level refinement of the 3D scene (Sec. 4.3).

4.1 Stepwise 2D Personalization

In general, our stepwise personalization of the pre-trained T2I model (i.e., Stable Diffusion (SD) [Rombach et al. 2022]) is based on DreamBooth [Ruiz et al. 2022], but with two significant modifications. These changes are essential to personalize both the existing scene and the novel content in the reference image. First, in the 2D personalization of the existing scene, we propose an attention-based localization loss to enforce the interaction between the existing and the novel content specified by the provided 3D bounding box (e.g., sunglasses on the forehead, see Fig. 7). Note that the reference image is not involved in this step. Second, in the 2D personalization of the novel content, we introduce LoRA layers to better capture the unique characteristics of the specified item in the reference image.

4.1.1 2D personalization of the existing scene. We first personalize the SD to the given scene to facilitate various types of editing of the scene afterward. Specifically, the initial text prompt (e.g. "a toy") is obtained using an image captioning model, BLIP-2 [Li et al. 2023]. To enhance the specificity of the scene, we add a special token V_1 in front of the noun describing the scene, resulting in a scene-specific text prompt (e.g., "a V_1 toy") as in [Zhuang et al. 2023]. The UNet ϵ_ϕ of the T2I model is fine-tuned with the reconstruction loss and the prior preservation loss [Ruiz et al. 2022]. The input of the reconstruction training includes the scene-specific text and a rendered image of the 3D scene from a random view. The input of the prior preservation training includes the initial text and a random image generated by SD using the initial text as input (omitted in Fig. 2 to reduce clutter). The above losses are computed as follows:

$$\begin{aligned}\mathcal{L}_{scene} =& \mathbb{E}_{z,y,\epsilon,t}||\epsilon_{\phi_1}(z_t,t,p,y)-\epsilon||_2^2+ \\ & \mathbb{E}_{z^*,y^*,\epsilon,t^*}||\epsilon_{\phi_1}(z_t^*,t^*,p^*,y^*)-\epsilon||_2^2\end{aligned} \tag{3}$$

where y denotes the text prompt, t the timestep, z_t the noised latent code at t-th timestep extracted from the input scene image, and p the camera pose. Superscript $*$ denotes the corresponding variables used in prior preservation training. Note that we add an additional camera pose p to the condition embeddings in the network to have a better viewpoint control of the generated images from the SD, facilitating the subsequent SDS-based 3D scene optimization. Since randomly generated images for prior preservation training do not have a meaningful "scene pose", we assign a fixed camera pose $p^* = I_4$ that will never be used for rendering.

To encourage accurate localization of the target object, we introduce an attention-based localization loss (Fig. 2) during the 2D scene personalization to encourage the SD to generate images containing

the required scene-object interaction. This step is particularly important if the target object is specified at a rarely seen location (e.g., sunglasses on the forehead, see Fig. 7). The actual location of the target object generated by SD is extracted from the cross-attention map A_t of the object keyword (e.g., "sunglasses") following [Hertz et al. 2022]. The wanted location of the target object (i.e., GT editing region) is obtained by projecting the provided 3D bounding box to the image plane. The loss between the actual and the wanted location is defined as:

$$\mathcal{L}_{loc} = (1 - \underset{s \in \mathcal{S}}{max}(A_t^s)) + \lambda \sum_{s \in \bar{\mathcal{S}}} ||A_t^s||_2^2 \tag{4}$$

where, λ is a weight to balance two terms, $\mathcal{S}$ the GT editing mask region (projection of the 3D bounding box $\mathcal{B}$) and $\bar{\mathcal{S}}$ the otherwise. Intuitively, this loss encourages a high probability inside the editing area and penalizes the presence of the target object outside the editing area. As demonstrated in our ablation study (Fig. 7), this loss is crucial for ensuring precise editing within the specified region.

4.1.2 2D personalization of the novel content. We introduce a dedicated personalization step using LoRA [Hu et al. 2021] (with the UNet fixed) to better capture the unique characteristics contained in the reference image. This step is essential to reduce the negative influence (e.g. concept forgetting [Kumari et al. 2023]) when learning (personalizing) multiple concepts, resulting in a better representation of both the scene and the novel content. Specifically, we train the additional LoRA layers inserted to the previously personalized and fixed T2I model ϵ_{ϕ^*}. Similar to the last step, we obtain the initial text prompt using BLIP-2 model and insert a special token V_2 into it, yielding an object-specific text prompt y^r of the reference object (e.g. "V_2 sunglasses"). The new LoRA layers are trained with the following loss function:

$$\mathcal{L}_{ref} = \mathbb{E}_{z^r, y^r, \epsilon, t} ||\epsilon_{\phi_2}(z_t^r, t, p^*, y^r) - \epsilon||_2^2 \tag{5}$$

After training, the content of the scene and the reference image are stored in UNet and added LoRA layers, respectively, resulting in largely reduced mutual interference.

4.2 Coarse Editing via SDS Loss

We optimize the selected Gaussians $\mathcal{G}^{\mathcal{B}} \in \mathcal{B}$ (i.e., those inside the bounding box $\mathcal{B}$) with SDS loss from the personalized T2I diffusion model ϵ_{ϕ_2}. Specifically, we input randomly rendered images $\hat{I}$ using sampled camera poses p and the text prompt y^G into the T2I model ϵ_{ϕ_2}, and calculate the global scene SDS Loss as follows:

$$\nabla_{\mathcal{G}} \mathcal{L}_{SDS}^{G}(\phi_2, f(\mathcal{G})) = \mathbb{E}_{\epsilon,t}\left[w(t)(\epsilon_{\phi_2}(z_t; t, p, y^G) - \epsilon) \frac{\partial z}{\partial \hat{I}} \frac{\partial \hat{I}}{\partial \mathcal{G}} \right] \tag{6}$$

where y^G is the text prompt including special tokens V_1, V_2 and describes our wanted result, $f(\cdot)$ the GS rendering algorithm.

It is noteworthy that the selection and update criteria of the Gaussians $\mathcal{G}^{\mathcal{B}}$ to be optimized are slightly different for different types of editing tasks. For object insertion, we duplicate all the Gaussians inside the bounding box and exclusively optimize all the attributes of these new Gaussians. For object replacement and re-texturing, all the Gaussians inside the bounding box will be updated. For stylization, optimization is applied to all the Gaussians in the scene. Note that we only update the colors (i.e., the spherical harmonic coefficients) for re-texturing instead of updating all the attributes.

Since the foreground and background of a GS-based scene are readily separable given the bounding box $\mathcal{G}^{\mathcal{B}}$, we introduce another local SDS loss for object-centric editing (e.g., object insertion/replacement) to reduce artifacts as follows:

$$\nabla_{\mathcal{G}^{\mathcal{B}}} \mathcal{L}_{SDS}^{L}(\phi_2, f(\mathcal{G}^{\mathcal{B}})) = \mathbb{E}_{\epsilon,t}\left[w(t)(\epsilon_{\phi_2}(z_t; t, p, y^L) - \epsilon) \frac{\partial z}{\partial \hat{I}} \frac{\partial \hat{I}}{\partial \mathcal{G}^{\mathcal{B}}} \right] \tag{7}$$

where y^L is the text prompt including the special tokens V_2 and only describes our wanted new object, $\hat{I}$ the rendered images containing only the foreground object.

We employ $\mathcal{L}_{SDS}^{G}$ and $\mathcal{L}_{SDS}^{L}$ with weight γ to optimize $\mathcal{G}^{\mathcal{B}}$:

$$\mathcal{L}_{\mathcal{SDS}} = \gamma \mathcal{L}_{SDS}^{G} + (1 - \gamma) \mathcal{L}_{SDS}^{L} \tag{8}$$

4.3 Pixel-Level Image Refinement

In this stage, we introduce a pixel-level reconstruction loss to effectively enhance the quality of the editing results, since the 3D results directly optimized with SDS loss usually contain artifacts (e.g. green noise on the glasses' frame, needle-like noise on the hair in Fig. 11).

The core of this stage is to create a pseudo-GT image I_{gt} to supervise the rendered image I_c from the coarse GS. Firstly, we follow SDEdit [Meng et al. 2021] to add noise on the rendered image I_c to obtain the noised image I_c^d and then utilized the personalized T2I model ϵ_{ϕ_2} as a denoising network and obtain the denoised image I_c^d. The denoising process effectively reduces the artifacts in I_c (see Fig. D.1 in the supplementary), but also alters the background image. Secondly, we obtain the binary instance mask M^{inst} of the edited object/part by rendering only the editable Gaussians $\mathcal{G}^{\mathcal{B}}$ and thresholding its opacity mask. Then, we render a background image I_{bg} with only the fixed Gaussians. Finally, the pseudo-GT image I_{gt} is obtained as:

$$I_{gt} = M^{inst} \odot I_c^d + (1 - M^{inst}) \odot I_{bg} \tag{9}$$

This process ensures that the background image is clean and the same as the original scene while the foreground editable region is enhanced by the T2I model ϵ_{ϕ_2}. Using this pseudo-GT image as pixel-level supervision effectively enhances the resultant texture and reduces floaters (Fig. 11). MSE loss is applied between the rendered image I_c and the created pseudo-GT image I_{gt}. A flowchart depicting the complete preparation of I_{gt} is included in the supplementary.

To maintain better coverage, the rendering camera poses cover all elevation and azimuth angles with an interval of 30° within a predefined range. To maintain better view-consistency of the denoised images, we set a small noise level ($t_0 = 0.05$, .i.e, "intermediate time" in SDEdit). Using such a small noise level effectively enhances fine texture details, removes small artifacts, and does not introduce significant shape and appearance change, maintaining better view consistency for the target editing region.

5 EXPERIMENTS

5.1 Experimental Setup

5.1.1 Implementation Details. We use the official code to train the original scene GS, with the default hyper-parameters. In the stepwise 2D personalization stage, the scene personalization step consists of

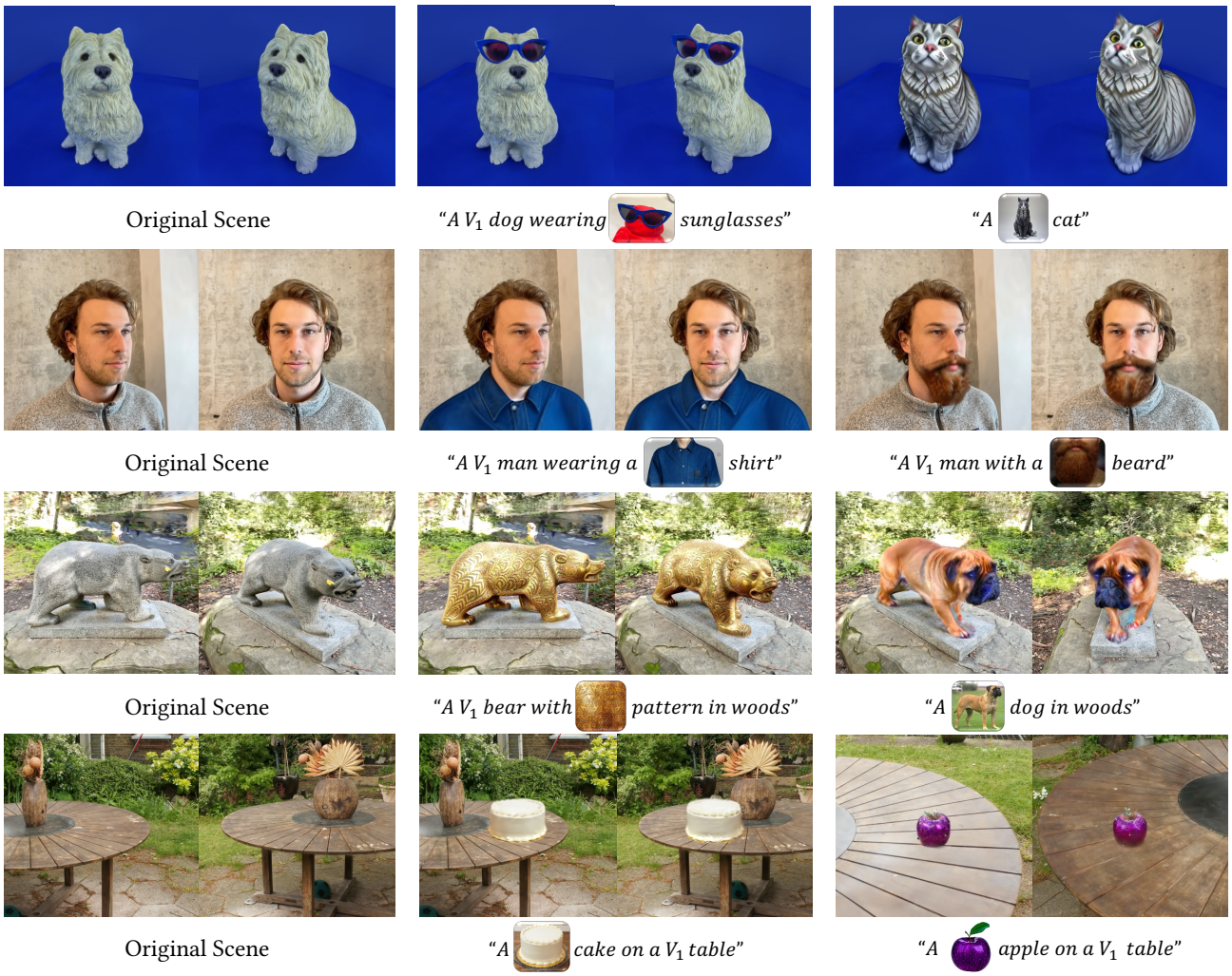

Fig. 3. Editing results of the proposed TIP-Editor. Images in the text prompts denote their associated *rare tokens*, which are fixed without optimization.

1k iterations, while the novel content personalization contains 500. We set $\lambda = 0.1$ in $\mathcal{L}_{loc}$. In the coarse editing stage, we adopt the sampling strategy of views from [Zhuang et al. 2023]. The size of the rendered images is 512×512. Owing to the different complexity of the editing task, this stage requires optimizing for 1K∼5K iterations, consuming approximately 5∼25 minutes. The refinement stage takes 3K iterations with the supervision of the generated I_{gt}, concluding in less than 3 minutes. More implementation details can be found in the supplementary.

5.1.2 Dataset. To comprehensively evaluate our method, We select six representative scenes with different levels of complexity following previous works [Chen et al. 2023b; Haque et al. 2023; Zhuang et al. 2023]. These scenes include objects in simple backgrounds, human faces, and complex outdoor scenes. For each editing, a hybrid prompt, consisting of text and a reference image obtained from the Internet, is employed to guide the editing. Additionally, we manually set a 3D bounding box to define the editing region.

5.1.3 Baselines. Due to the lack of dedicated image-based editing baselines, we compare with two state-of-the-art text-based radiance field editing methods, including Instruct-NeRF2NeRF ("I-N2N") [Haque et al. 2023] and DreamEditor [Zhuang et al. 2023]. I-N2N utilizes Instruct-pix2pix [Brooks et al. 2022] to update the rendered multi-view images according to special text instructions. DreamEditor adopts a mesh-based representation and includes an attention-based localization operation to support local editing. For a fair comparison, we replace its automatic localization with a more accurate manual selection. See our supplementary for more implementation details.

5.1.4 Evaluation Criteria. For quantitative evaluation, we adopt CLIP Text-Image directional similarity following [Haque et al. 2023; Zhuang et al. 2023] to assess the alignment of the editing outcomes with the given text prompt. To evaluate image-image alignment (between the edited scene and the reference image), we follow [He et al. 2023] to calculate the average DINO similarity [Oquab et al.

Fig. 4. Sequential editing results. We show two rendered images of the 3D scene after every editing step, indicated by the number in the top-left corner. V_*, V_{**}, and V_{***} represent the special tokens of the scene in different sequences of editing.

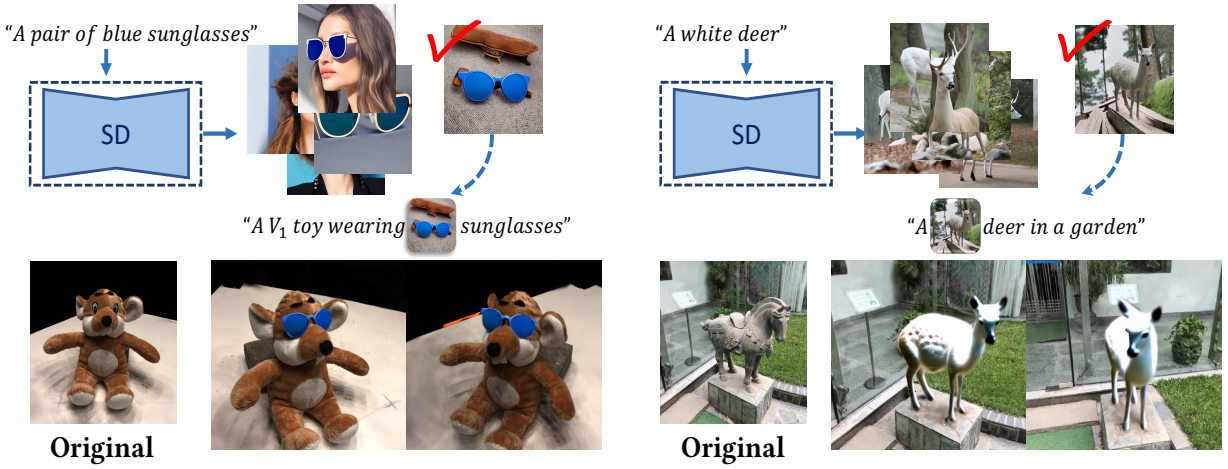

Fig. 5. Results of using a generated image as the reference. We first generate several candidate images by the diffusion model using text prompts, then we choose one as the reference image for editing.

2023] between the reference image and the rendered multi-view images of the edited 3D scene. Detailed information about these calculations is available in the supplementary.

Additionally, we conduct a user study and ask the participants (50 in total) to evaluate the results of different methods from two aspects (overall "Quality", and "Alignment" to the reference image). The user study includes 10 questions, each containing the edited results of the two baselines and ours rendered into rotating videos in random order (see our supplementary). The 10 questions have covered various scenes and editing types to better compare the methods under different scenarios.

5.2 Visual Results of TIP-Editor

In Fig.1 and Fig. 3, we present qualitative results of TIP-Editor. Video demonstrations are included in the supplementary. Experiments on diverse 3D scenes demonstrate that TIP-Editor effectively executes various editing tasks, including re-texturing, object insertion, object replacement, and stylization, achieving both high-quality results and strictly following the provided text prompt and reference image.

5.2.1 Keeping unique characteristics specified by the reference image. One of the most distinguishable differences between TIP-Editor and previous methods is that TIP-Editor also supports an image prompt, which offers more accurate control and makes it more user-friendly in real applications. Results in Fig. 1&3 demonstrate high consistency between the updated 3D scene and the reference image (e.g. the *styles* of the sunglasses; the *white* giraffe; the *virtual ghost* horse; the joker make-up appeared in movie *The Dark Knight*). Moreover, as depicted in the bottom of Fig. 1, our method can also perform global scene editing, such as transferring the entire scene in the *Modigliani* style of the reference image.

5.2.2 Sequential editing. TIP-Editor can sequentially edit the initial scene multiple times thanks to the local update of the GS and the stepwise 2D personalization strategy, which effectively reduces

Fig. 6. Visual comparisons between different methods. Our method produces obviously higher-quality results and *accurately* follows the reference image input (bottom-right corner in column 1). Instruct-N2N sometimes misunderstands (row 1) or overlooks (row 2) the keywords. DreamEditor faces difficulty in making obvious shape changes (row 2). Both of them do not support image prompts to specify detailed appearance/style, producing less controlled results.

Table 1. Quantitative comparisons. CLIP_{dir} denotes the CLIP Text-Image directional similarity. DINO_{sim} is the DINO similarity.

Method	$\text{CLIP}_{dir}\uparrow$	$\text{DINO}_{sim}\uparrow$	$\text{Vote}_{quality}$	$\text{Vote}_{alignment}$
Instruct-N2N	8.3	36.4	21.6%	8.8%
DreamEditor	11.4	36.8	7.6%	10.0%
Ours	**15.5**	**39.5**	**70.8%**	**81.2%**

the interference between the existing scene and the novel content. Results in Fig.4 demonstrate the sequential editing capability. There is no observable quality degradation after multiple times of editing and no interference between different editing operations.

5.2.3 Using generated image as the reference. In the absence of the reference image, we can generate multiple candidates from a T2I model and let the user choose a satisfactory one. This interaction offers the user more control and makes the final result more predictable. Fig. 5 shows some examples.

5.3 Comparisons with State-of-the-Art Methods

5.3.1 Qualitative comparisons. Fig.6 shows visual comparisons between our method and the baselines. Since both baselines do not support image prompts as input, they generate an uncontrolled (probably the most common) item belonging to the object category. In contrast, our results consistently maintain the unique characteristics specified in the reference images (i.e., the *heart-shaped* sunglasses; the *white* giraffe; the joker from the movie *The Dark Knight*).

Moreover, Instruct-N2N sometimes misunderstands (row 1) or overlooks (row 2) the keywords, or cannot generate a specified

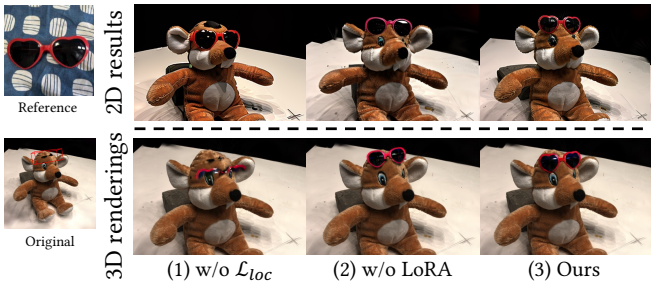

Fig. 7. Ablation study on the components proposed in stepwise 2D personalization. We compare the generated images of the personalized T2I model (top row) and the rendered images of the updated 3D scene (bottom row). Removing the localization loss $\mathcal{L}_{loc}$ fails to place the new object in the specified place. Removing the separate LoRA layers dedicated for the personalization of the reference image produces less similar results (heart-shaped vs. regular round shape).

Table 2. Quantitative evaluation of the 3D renderings in the ablation study on the components proposed in stepwise 2D personalization (Fig. 7).

	w/o $\mathcal{L}_{loc}$	w/o LoRA	Ours
$\text{CLIP}_{dir}\uparrow$	4.4	24.0	**25.4**
$\text{DINO}_{sim}\uparrow$	18.3	28.0	**28.8**

appearance in limited experiments (row 3), probably due to limited supported instructions in Instruct-Pix2Pix. DreamEditor also faces difficulty if the user wants to add a specified sunglasses item (row 1). Additionally, it is difficult for DreamEditor to make obvious shape

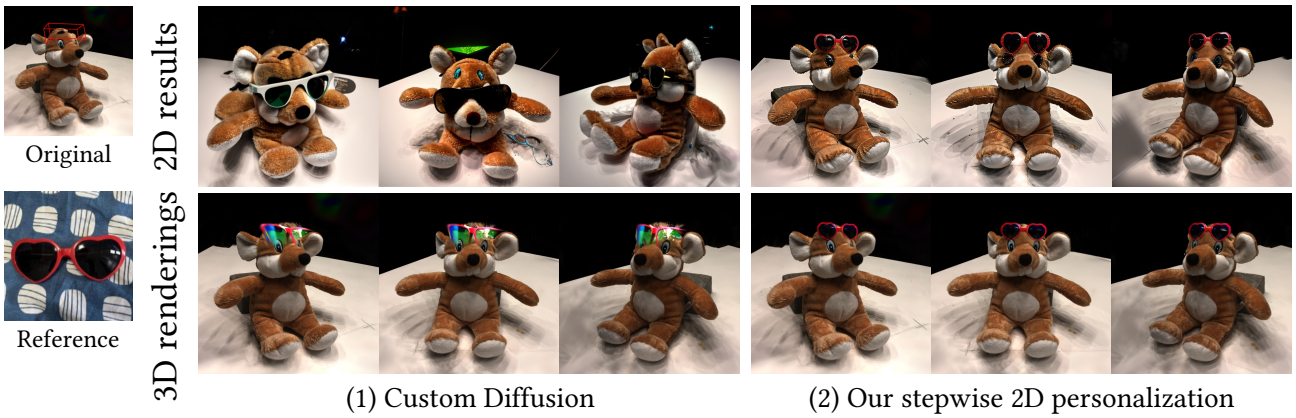

Fig. 8. Comparison of different 2D personalization methods. Generated images of the T2I models after personalization (top) and the final updated 3D scene (bottom) are presented. *Text prompt*: "A V_1 toy wearing V_2 sunglasses on the forehead"

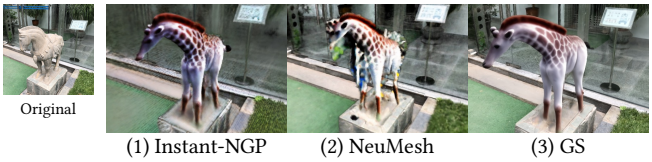

Fig. 9. Ablation study on different 3D representations to show the advantage of GS for this task. Using Instant-NGP results in a changed background while using NeuMesh cannot produce large enough shape deformation. In contrast, using *explicit* and *flexible* GS obtains the best foreground editing result while keeping the background unchanged.

Table 3. Quantitative evaluation of the 3D renderings in the ablation study on different 3D representations (Fig. 9).

	Instant-NGP	NeuMesh	Ours
CLIP$_{dir}$ ↑	17.8	18.4	**18.7**
DINO$_{sim}$ ↑	31.8	27.0	**33.5**

changes (row 2) to the existing object due to its adoption of a less flexible mesh-based representation (i.e., NeuMesh).

5.3.2 Quantitative comparisons. Tab. 1 shows the results of the CLIP Text-Image directional similarity and DINO similarity. The results clearly demonstrate the superiority of our method in both metrics, suggesting that the appearance generated by our method aligns better with both the text prompt and the image prompt. A similar conclusion has been drawn according to the user study. Our results surpass the baselines with a substantial margin on both the *quality* evaluation (70.8% votes) and the *alignment* evaluation (81.2% votes).

5.4 Ablation Study

5.4.1 Ablation studies on the stepwise 2D personalization. We conduct ablative experiments in Fig.7 and Tab. 2 to demonstrate the benefit of using $\mathcal{L}_{loc}$ and LoRA Layers in the stepwise 2d personalization. Without $\mathcal{L}_{loc}$, the fine-tuned T2I model fails to place the

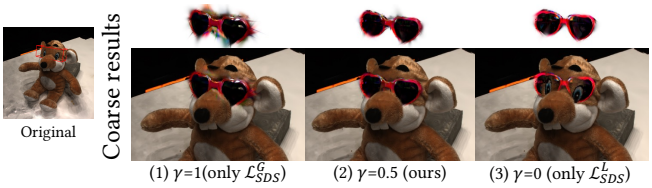

Fig. 10. Ablation study on the influence of global and local SDS (Eq. 8) in the coarse stage. The top row shows the rendering of the editable Gaussians $\mathcal{G}^{\mathcal{B}}$. Only using global SDS $\mathcal{L}^G_{SDS}$ produces low-quality foreground object/part, while only using local SDS $\mathcal{L}^L_{SDS}$ produces unnatural foreground when composited with the existing scene (e.g., color, placement).

Table 4. Quantitative evaluation of the 3D renderings in the ablation study on different γ in coarse editing (Fig. 10).

	$\gamma = 1$	$\gamma = 0.5$	$\gamma = 0$
CLIP$_{dir}$ ↑	9.0	**18.1**	12.3
DINO$_{sim}$ ↑	29.0	**29.5**	27.5

sunglasses in the specified region (i.e. on the forehead) due to the bias present in the training data of the original T2I model. Introducing dedicated LoRA layers to personalize the unique features in the reference image results in more faithful output, demonstrating the effectiveness of the proposed stepwise 2D personalization strategy in capturing details in the reference image.

We also compare our stepwise 2D personalization with another multiple object personalization method, i.e., Custom Diffusion. As shown in Fig. 8, our stepwise 2D personalization achieves high-quality and faithful personalization results and ensures precise editing within the specified region. In contrast, although Custom Diffusion assigns separate special text tokens to each *concept*, the UNet is updated by all *concepts*, resulting in less satisfactory personalization results. In addition, Custom Diffusion lacks a localization mechanism to specify the interaction between the sunglasses and the toy, failing to place the new object on the toy's forehead.

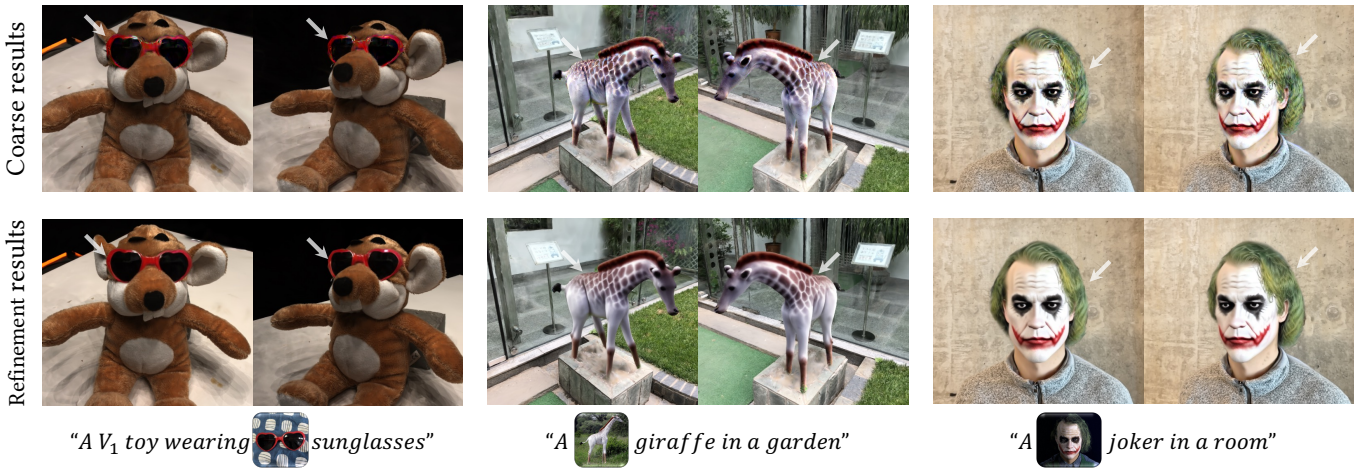

Fig. 11. Comparison of the coarse editing results and the refinement results. The region indicated by the arrow demonstrates the efficacy of the refinement step in enhancing the quality of the editing results.

5.4.2 Ablation study on different 3D representations. We test different 3D representations in Fig. 9 and Tab. 4 while keeping all the other settings the same. Using GS obtains the best editing result while keeping the background unchanged. For Instant-NGP [Müller et al. 2022], we observe undesired changes in the background since its content in different locations is not independent due to its adoption of a shared MLP and multi-resolution grid. For NeuMesh [Yang et al. 2022] (used in DreamEditor), we observe obvious artifacts if the editing requires (relatively) large shape change since it adopts mesh as its geometry proxy, which is less flexible than GS due to topology constraints.

5.4.3 Influence of different γ in coarse editing. As in Fig.10, both the global and local SDS loss are necessary and our solution achieves the best result. Specifically, only using global SDS loss $\mathcal{L}_{SDS}^{G}$ results in obvious artifacts in the editable region. Only using local SDS loss $\mathcal{L}_{SDS}^{L}$ results in inaccurate placement of the object and unnatural color discrepancy between the background and the novel content since the context information is missing during optimization.

5.4.4 Effectiveness of the pixel-level refinement step. As in Fig.11, introducing the refinement stage effectively reduces artifacts and enhances the texture, resulting in substantially improved quality.

5.4.5 Ablation study on different formulas of $\mathcal{L}_{loc}$. In Eq. 4, we only encourage the max attention value of the pixels in the region $\bar{\mathcal{S}}$ to approach 1. This is different from previous works [Avrahami et al. 2023; Xiao et al. 2023], which force all the attention values in the region $\bar{\mathcal{S}}$ to 1. We compare these two settings in Fig. 12. Using the latter one tends to create an object (even multiple objects) covering the entire bounding box regardless of whether its placement is appropriate (e.g. overfill), resulting in various artifacts.

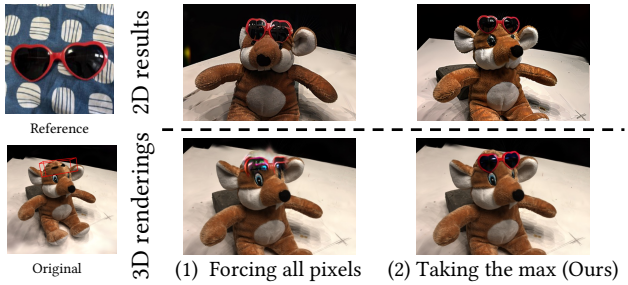

Fig. 12. Ablation study on different formulas of $\mathcal{L}_{loc}$. We compare the generated images of the personalized T2I model (top row) and the rendered images of the updated 3D scene (bottom row). Forcing all attention values in the region $\bar{\mathcal{S}}$ to 1 tends to create an object (even multiple objects) covering the entire bounding box regardless of whether its placement is appropriate (e.g. overfill), resulting in various artifacts.

6 CONCLUSION AND LIMITATIONS

In this paper, our proposed TIP-Editor equips the emerging text-driven 3D editing with an additional image prompt as a complement to the textual description and produces high-quality editing results accurately aligned with the text and image prompts while keeping the background unchanged. TIP-Editor offers significantly enhanced controllability and enables versatile applications, including object insertion, object replacement, re-texturing, and stylization.

One limitation of TIP-Editor is the coarse bounding box input. Although convenient, it struggles in complex scenes where bounding boxes may include unwanted elements. It would be very beneficial to automatically obtain 3D instance segmentation of the scene. Another limitation is related to geometry extraction since it is hard to extract a smooth and accurate mesh from GS-represented scenes.

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China (NO. 62325605, NO. 62322608), in part by the CAAI-MindSpore Open Fund, developed on OpenI Community,in part by the Open Project Program of State Key Laboratory of Virtual Reality Technology and Systems, Beihang University (No.VRLAB2023A01).

REFERENCES

Omri Avrahami, Kfir Aberman, Ohad Fried, Daniel Cohen-Or, and Dani Lischinski. 2023. Break-a-scene: Extracting multiple concepts from a single image. In *SIGGRAPH Asia 2023 Conference Papers*. 1–12.

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

Chong Bao, Yinda Zhang, Bangbang Yang, Tianxing Fan, Zesong Yang, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. 2023. SINE: Semantic-driven image-based nerf editing with prior-guided editing field. In *CVPR 2023*. 20919–20929.

Tim Brooks, Aleksander Holynski, and Alexei A Efros. 2022. InstructPix2Pix: Learning to follow image editing instructions. *arXiv preprint arXiv:2211.09800* (2022).

de Charette Raoul Cao, Anh-Quan. 2023. SceneRF: Self-supervised monocular 3D scene reconstruction with radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 9387–9398.

Jiazhong Cen, Zanwei Zhou, Jiemin Fang, Chen Yang, Wei Shen, Lingxi Xie, Xiaopeng Zhang, and Qi Tian. 2023. Segment Anything in 3D with NeRFs. In *NeurIPS*.

Jun-Kun Chen, Jipeng Lyu, and Yu-Xiong Wang. 2023c. Neuraleditor: Editing neural radiance fields via manipulating point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12439–12448.

Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. 2023a. Fantasia3D: Disentangling Geometry and Appearance for High-quality Text-to-3D Content Creation. *arXiv preprint arXiv:2303.13873* (2023).

Yiwen Chen, Zilong Chen, Chi Zhang, Feng Wang, Xiaofeng Yang, Yikai Wang, Zhongang Cai, Lei Yang, Huaping Liu, and Guosheng Lin. 2023b. GaussianEditor: Swift and Controllable 3D Editing with Gaussian Splatting. *arXiv preprint arXiv:2311.14521* (2023).

Zilong Chen, Feng Wang, and Huaping Liu. 2023d. Text-to-3d using gaussian splatting. *arXiv preprint arXiv:2309.16585* (2023).

Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. 2022. DiffEdit: Diffusion-based semantic image editing with mask guidance. *arXiv preprint arXiv:2210.11427* (2022).

Dale Decatur, Itai Lang, Kfir Aberman, and Rana Hanocka. 2023. 3D Paintbrush: Local Stylization of 3D Shapes with Cascaded Score Distillation. *arXiv preprint arXiv:2311.09571* (2023).

Congyue Deng, Chiyu Jiang, Charles R Qi, Xinchen Yan, Yin Zhou, Leonidas Guibas, Dragomir Anguelov, et al. 2022. NeRDi: Single-View NeRF Synthesis with Language-Guided Diffusion as General Image Priors. *arXiv preprint arXiv:2212.03267* (2022).

Jiemin Fang, Junjie Wang, Xiaopeng Zhang, Lingxi Xie, and Qi Tian. 2023. GaussianEditor: Editing 3D gaussians delicately with text instructions. *arXiv preprint arXiv:2311.16037* (2023).

Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. 2022. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618* (2022).

Ayaan Haque, Matthew Tancik, Alexei A Efros, Aleksander Holynski, and Angjoo Kanazawa. 2023. Instruct-NeRF2NeRF: Editing 3D Scenes with Instructions. *arXiv preprint arXiv:2303.12789* (2023).

Runze He, Shaofei Huang, Xuecheng Nie, Tianrui Hui, Luoqi Liu, Jiao Dai, Jizhong Han, Guanbin Li, and Si Liu. 2023. Customize your NeRF: Adaptive Source Driven 3D Scene Editing via Local-Global Iterative Training. *arXiv preprint arXiv:2312.01663* (2023).

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

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685* (2021).

Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. 2022. Imagic: Text-based real image editing with diffusion models. *arXiv preprint arXiv:2210.09276* (2022).

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics* 42, 4 (2023).

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023b. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 4015–4026.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. 2023a. Segment Anything. *arXiv:2304.02643* (2023).

Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. 2022. Decomposing NeRF for editing via feature field distillation. *arXiv preprint arXiv:2205.15585* (2022).

Nupur Kumari, Bingliang Zhang, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. 2023. Multi-concept customization of text-to-image diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1931–1941.

Jiaxin Li, Zijian Feng, Qi She, Henghui Ding, Changhu Wang, and Gim Hee Lee. 2021. Mine: Towards continuous depth mpi with nerf for novel view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 12578–12588.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597* (2023).

Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. 2022. Magic3D: High-Resolution Text-to-3D Content Creation. *arXiv preprint arXiv:2211.10440* (2022).

Hao-Kang Liu, I Shen, Bing-Yu Chen, et al. 2022. NeRF-In: Free-form NeRF inpainting with RGB-D priors. *arXiv preprint arXiv:2206.04901* (2022).

Xiangyue Liu, Han Xue, Kunming Luo, Ping Tan, and Li Yi. 2024. GenN2N: Generative NeRF2NeRF Translation. *arXiv preprint arXiv:2404.02788* (2024).

Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. 2023. Realfusion: 360deg reconstruction of any object from a single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 8446–8455.

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

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. NeRF: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)* 41, 4 (2022), 1–15.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. 2023. DINOv2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193* (2023).

Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. 2022. DreamFusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988* (2022).

Guocheng Qian, Jinjie Mai, Abdullah Hamdi, Jian Ren, Aliaksandr Siarohin, Bing Li, Hsin-Ying Lee, Ivan Skorokhodov, Peter Wonka, Sergey Tulyakov, et al. 2023. Magic123: One image to high-quality 3D object generation using both 2D and 3D diffusion priors. *arXiv preprint arXiv:2306.17843* (2023).

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *ICML 2021*. 8748–8763.

Amit Raj, Srinivas Kaza, Ben Poole, Michael Niemeyer, Nataniel Ruiz, Ben Mildenhall, Shiran Zada, Kfir Aberman, Michael Rubinstein, Jonathan Barron, et al. 2023. DreamBooth3D: Subject-Driven Text-to-3D Generation. *arXiv preprint arXiv:2303.13508* (2023).

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with CLIP latents. *arXiv preprint arXiv:2204.06125* (2022).

Elad Richardson, Gal Metzer, Yuval Alaluf, Raja Giryes, and Daniel Cohen-Or. 2023. Texture: Text-guided texturing of 3D shapes. *arXiv preprint arXiv:2302.01721* (2023).

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *CVPR 2022*. 10684–10695.

Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. 2022. DreamBooth: Fine tuning text-to-image diffusion models for subject-driven generation. *arXiv preprint arXiv:2208.12242* (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. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS 2022* 35 (2022), 36479–36494.

Johannes Lutz Schönberger and Jan-Michael Frahm. 2016. Structure-from-Motion Revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Etai Sella, Gal Fiebelman, Peter Hedman, and Hadar Averbuch-Elor. 2023. Vox-E: Text-guided Voxel Editing of 3D Objects. *arXiv preprint arXiv:2303.12048* (2023).

Jiaxiang Tang, Xiaokang Chen, Jingbo Wang, and Gang Zeng. 2022. Compressible-composable nerf via rank-residual decomposition. *Advances in Neural Information Processing Systems* 35 (2022), 14798–14809.

Junshu Tang, Tengfei Wang, Bo Zhang, Ting Zhang, Ran Yi, Lizhuang Ma, and Dong Chen. 2023. Make-it-3D: High-fidelity 3D creation from a single image with diffusion prior. *arXiv preprint arXiv:2303.14184* (2023).

Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. 2022. CLIP-NeRF: Text-and-image driven manipulation of neural radiance fields. In *CVPR 2022*. 3835–3844.

Can Wang, Mingming He, Menglei Chai, Dongdong Chen, and Jing Liao. 2023a. Mesh-Guided Neural Implicit Field Editing. *arXiv preprint arXiv:2312.02157* (2023).

Can Wang, Ruixiang Jiang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. 2023b. NeRF-Art: Text-driven neural radiance fields stylization. *IEEE Transactions on Visualization and Computer Graphics* (2023).

Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. 2023c. ProlificDreamer: High-Fidelity and Diverse Text-to-3D Generation with Variational Score Distillation. *arXiv preprint arXiv:2305.16213* (2023).

Tong Wu, Jia-Mu Sun, Yu-Kun Lai, and Lin Gao. 2023. De-nerf: Decoupled neural radiance fields for view-consistent appearance editing and high-frequency environmental relighting. In *ACM SIGGRAPH 2023 conference proceedings*. 1–11.

Fanbo Xiang, Zexiang Xu, Milos Hasan, Yannick Hold-Geoffroy, Kalyan Sunkavalli, and Hao Su. 2021. Neutex: Neural texture mapping for volumetric neural rendering. In *CVPR 2021*. 7119–7128.

Guangxuan Xiao, Tianwei Yin, William T Freeman, Frédo Durand, and Song Han. 2023. Fastcomposer: Tuning-free multi-subject image generation with localized attention. *arXiv preprint arXiv:2305.10431* (2023).

Tianhan Xu and Tatsuya Harada. 2022. Deforming radiance fields with cages. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIII*. Springer, 159–175.

Bangbang Yang, Chong Bao, Junyi Zeng, Hujun Bao, Yinda Zhang, Zhaopeng Cui, and Guofeng Zhang. 2022. NeuMesh: Learning disentangled neural mesh-based implicit field for geometry and texture editing. In *ECCV 2022*. Springer, 597–614.

Taoran Yi, Jiemin Fang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. 2023. Gaussiandreamer: Fast generation from text to 3d gaussian splatting with point cloud priors. *arXiv preprint arXiv:2310.08529* (2023).

Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. 2021. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4578–4587.

Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. 2022. NeRF-editing: geometry editing of neural radiance fields. In *CVPR 2022*. 18353–18364.

Jingyu Zhuang, Chen Wang, Liang Lin, Lingjie Liu, and Guanbin Li. 2023. DreamEditor: Text-driven 3d scene editing with neural fields. In *SIGGRAPH Asia 2023 Conference Papers*. 1–10.