Image Sculpting: Precise Object Editing with 3D Geometry Control

Jiraphon Yenphraphai[1] Xichen Pan[1] Sainan Liu[2] Daniele Panozzo[1] Saining Xie[1]

[1]New York University [2]Intel Labs

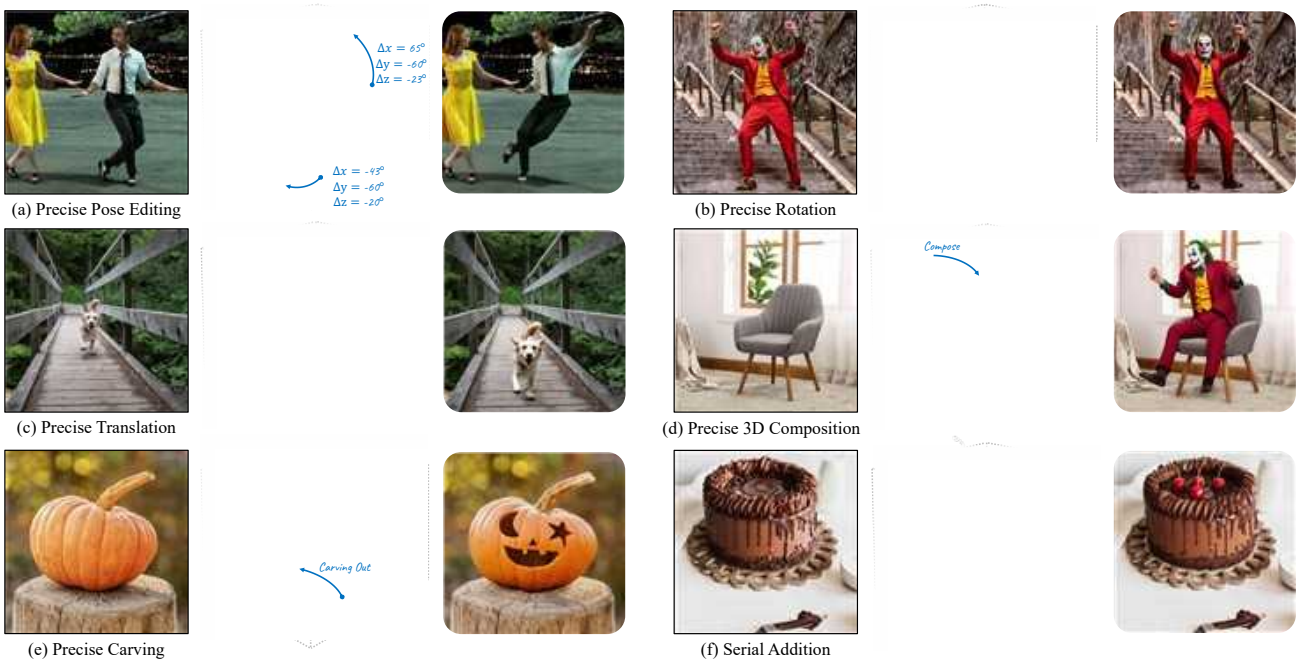

Figure 1. Achieving *precise* control in image editing tasks can be challenging with standard 2D generative pipelines. Our *Image Sculpting* framework offers the ability to interact with 3D geometry starting with a single image. This enables users to perform detailed, quantifiable, and physically-plausible edits, including *precise* pose editing, rotation, translation, 3D composition, carving, and serial addition.

Abstract

We present Image Sculpting*, a new framework for editing 2D images by incorporating tools from 3D geometry and graphics. This approach differs markedly from existing methods, which are confined to 2D spaces and typically rely on textual instructions, leading to ambiguity and limited control. Image Sculpting converts 2D objects into 3D, enabling direct interaction with their 3D geometry. Post-editing, these objects are re-rendered into 2D, merging into the original image to produce high-fidelity results through a coarse-to-fine enhancement process. The framework supports precise, quantifiable, and physically-plausible editing options such as pose editing, rotation, translation, 3D composition, carving, and serial addition. It marks an initial step towards combining the creative freedom of generative models with the precision of graphics pipelines.*

Code and project page available here.

1. Introduction

Recent developments in the field of image generative modeling [60, 63, 65, 86] have unlocked new potentials in creative content creation, offering unprecedented opportunities for the generation of diverse visual content by materializing ideas and concepts articulated through language prompts. However, the integration of these models into real-world content creation workflows still poses significant challenges. Among the most critical is the need for users to have detailed control over various aspects of generated objects, including their pose, shape, location, layout, and spatial compositions. The precision extends to quantifiable manipulations, such as rotating an object by a specific angle or making physically-realistic modifications, such as positioning a character in a way that conforms to basic anatomical and physical principles. Interestingly, such a quest for precision and controllability aligns closely with the core principles of computer graphics, which strive to generate pho-

torealistic images with artistic control.

In virtual effects (VFX) and rendering pipelines, experts meticulously craft and edit every detail within a fully controllable *3D environment*, striving for utmost realism. For decades, methods for accurately manipulating and rendering objects have been explored, leading to the development of numerous advanced techniques in 3D model acquisition, rigging, posing, lighting, texturing, and scene rendering. These methods form the bedrock of the modern computer graphics pipeline. However, it often requires custom hardware and software for (1) acquiring production-quality 3D models or designing them from scratch, (2) making these models possible to animate (rigging), (3) creating visually plausible animations (animation), (4) rendering back in the 2D world after applying material and setting up the lighting, and (5) compositing the resulting image with a background or other objects. This process often employs teams of artists and engineers for each one of these steps, as it requires substantial manual input using specialized tools (*e.g.* After Effects [2], Substance [4], and 3ds Max [27]).

In contrast, AI-based image generation avoids all this manual work, requiring only a text prompt. Leveraging the power of human language and large datasets of curated content, transforming a text description into a visually striking image is more accessible than ever. Yet, when it comes to precise object manipulation, the current 2D-based generative approach faces inherent limitations due to the lack of a third dimension, leading to incomplete information, limited user interaction on a flat plane, and possible ambiguities. The gap in controllability with respect to image generation using computer graphics techniques is striking, and closing it is a major goal of our work.

Most interfaces for image editing frameworks rely on text-based instructions. For example, techniques such as Prompt-to-Prompt [24], Plug-and-Play [76], InstructPix2Pix [10], Imagic [34] and Object 3DIT [47] offer adaptable language control. However, achieving precise manipulation through these models remains a challenge. Straightforward manipulations such as *"changing a style to mimic Van Gogh"* are manageable. However, more specific instructions such as *"lift the object by 5 cm and rotate it by 42 degrees."* are less likely to be successful, as current generative models cannot fulfill such detailed requests through textual prompts alone. 2D-based interactive methods such as DragGAN [54], FreeDrag [39], and DragDiffusion [70] demonstrate the ability to alter part of an object through transitions in the latent space. Despite this, they have their limitations: 1) they can accomplish basic deformations, but the outcomes are not entirely predictable, often leading to results that do not align with the user's intentions; 2) these latent transformations operate within the 2D feature space, which inherently limits their ability to represent 3D transformations and handle occlusions accurately; 3) they lack physics-awareness, which complicates incorporating external constraints, such as skeletal structures.

Our work draws inspiration from the computer graphics pipeline and ventures into a novel approach for 2D image-based object manipulation tasks. Our proposed *Image Sculpting* framework, which metaphorically suggests the flexible and precise sculpting of a 2D image in a 3D space, integrates three key components: (1) single-view 3D reconstruction, (2) manipulation of objects in 3D, and (3) a coarse-to-fine generative enhancement process. More specifically, 2D objects are converted into 3D models, granting users the ability to interact with and manipulate the 3D geometry directly, which allows for precision in editing. The manipulated objects are then seamlessly reincorporated into their original 2D contexts, maintaining visual coherence and fidelity. A critical hurdle in this process is the single-view 3D reconstruction method, a task that, despite rapid progress [25, 38, 40–43, 59, 68], often results in relatively low-fidelity, coarse geometric and texture representations. Unlike manually crafted 3D assets used for graphics, their rendered version is far from photo-realistic. Nonetheless, the extracted geometries are sufficient for interactive and precise control. To achieve high-quality final images, a separate enhancement procedure is necessary. In summary, our Image Sculpting pipeline has three key phases:

Phase 1. For the 3D reconstruction phase, we employ a zero-shot single image reconstruction model (Zero-1-to-3 [41]), which has been trained on extensive datasets [15] of 3D objects.

Phase 2. The deformation process utilizes established geometric processing tools, such as As-Rigid-As-Possible (ARAP) [75] and linear-based skinning [45], enabling interactive and precise manipulation of the 3D models.

Phase 3. For the generative enhancement process, we develop a coarse-to-fine enhancement approach, using an improved feature injection technique [76]. Our method strikes a balance between maintaining the original texture of the object and the modified geometry, utilizing a pre-trained text-to-image diffusion model with additional controls.

Our Image Sculpting framework showcases an array of precise and quantifiable image editing capabilities. These include precise pose editing, rotation, translation, multi-object 3D composition, carving, and serial addition. This suite of functionalities demonstrates the versatility of our approach and its superiority in precision and control compared to existing image editing methods. Our method also outperforms various baselines in image quality, as confirmed by both qualitative and quantitative evaluations on the new *SculptingBench* benchmark. We believe that our method can foster new opportunities in merging the flexibility of generative models with the precise controllability inherent in traditional graphics pipelines.

(a) Space Deformations (b) Shape-Aware Deformation (c) Linear Blend Skinning

Figure 2. Illustration of three mesh deformation methods applied to a 3D model. In cage-based space deformation (a), the model is placed in a cage and deformed when the user moves the cage vertices [31]. As-Rigid-As-Possible (ARAP) [75] deformation (b) deforms the model when user-selected blue handle points are moved towards designated red target points. Linear blend skinning (c) maps the deformation of a skeleton to the model [30]. Following deformation, a diffusion rendering process can be added for controllable generation. Each mesh deformation technique offers a different balance of control, speed, and precision. Our framework can use any of these techniques.

2. Related Work

Generative Image Editing In computer graphics, extensive research on interactive raster image editing exists, and we defer its detailed review to the next section. In computer vision, the advent of image generative models such as GANs [20, 32, 33] has expanded the scope of image editing to include style transfer [19], image-to-image translation [28, 88], latent manipulation [67, 81], and text-based manipulation [1, 55, 82]. Recently, capabilities in image editing have advanced significantly with the rise of diffusion models [16, 56, 63]. The leading systems [48, 53, 60, 65] allow users to generate image variations or use inpainting masks [51] to generate specific parts of scenes based on a text prompt. Other work explores enhancing pre-trained diffusion models with text-guided editing capabilities [10, 24, 49, 76]. Yet, text-based editing has limitations in precisely controlling object shapes and positions. ControlNet [87] incorporates additional conditional inputs such as depth [61], poses [11], and edges [83] for controllable generation. For more intuitive interactions, DragGAN [54] enables users to drag control points on objects with GANs, and similar techniques have been adapted for diffusion models [39, 70]. However, these methods are mostly confined to 2D and face challenges in tasks requiring more complex, out-of-plane transformations. 3D-aware generative models such as EG3D [12] and StyleNeRF [21] have explored this direction. OBJect-3DIT [47], a baseline in our paper, studied 3D-aware editing using language instructions. However, its effectiveness is somewhat constrained due to its training on a synthetic dataset.

Single-View Reconstruction Single-view 3D reconstruction is a long-standing problem in computer vision [23]. While algorithmic advancements are important, the significance of training data has been increasingly recognized. Earlier efforts were geared towards training models [52, 73, 80, 85] using smaller, simplistic 3D datasets [13, 62]. Recent approaches [58, 77] have started to utilize density distillation from pre-trained 2D diffusion models trained on large-scale text-image datasets, lessening the reliance on 3D data. Nonetheless, for improved view-consistency, the demand for high-quality 3D data is indispensable. The emergence of large-scale 3D datasets, such as Objaverse [14, 15], has spurred methods such as Zero-1-to-3 [41] to combine 2D score distillation with 3D data training. This has led to a surge in new models in this domain, noticeably enhancing reconstruction quality [40, 59, 69, 79]. Current 3D reconstruction models, while not perfect, have attained a level of maturity that makes them suitable for shape editing.

3. Overview of 3D Shape Deformation

The deformation of 3D shapes has been extensively studied in the last four decades, with both traditional and data-driven methods being proposed and successfully used in robotics, graphics, and engineering. We review the main approaches and their usability within our framework.

Space Deformations The older and still widely used approach is applying a volumetric warp function $f : R^3 \rightarrow R^3$ to all points of a 3D domain [66]. This approach can be applied to explicit (triangular or polygonal meshes) or implicit representations. The map can be parametrized using splines on lattices [66], vertices on a cage [31], or neural fields [17]. A limitation of these approaches is that they are unaware of the object shape, making them more challenging to use on complex articulated objects [9].

Shape-Aware Deformation Shape-aware deformations provide a set of controls linked to the objects' surface. In Computer-Aided-Design (CAD), a small set of control points define a smooth surface using spline patches [18]. Despite its flexibility and quality, extracting spline patches from 3D models or NeRFs is a challenging and open problem [7]. Partial differential equation (PDE)-based methods simulate the deformation of an object, representing it as a volumetric deformable solid [72] or as a thin rubber shell [75]. The forces guiding the deformation are applied by moving handles selected on a surface [8], making them intuitive to use and requiring minimal user interaction.

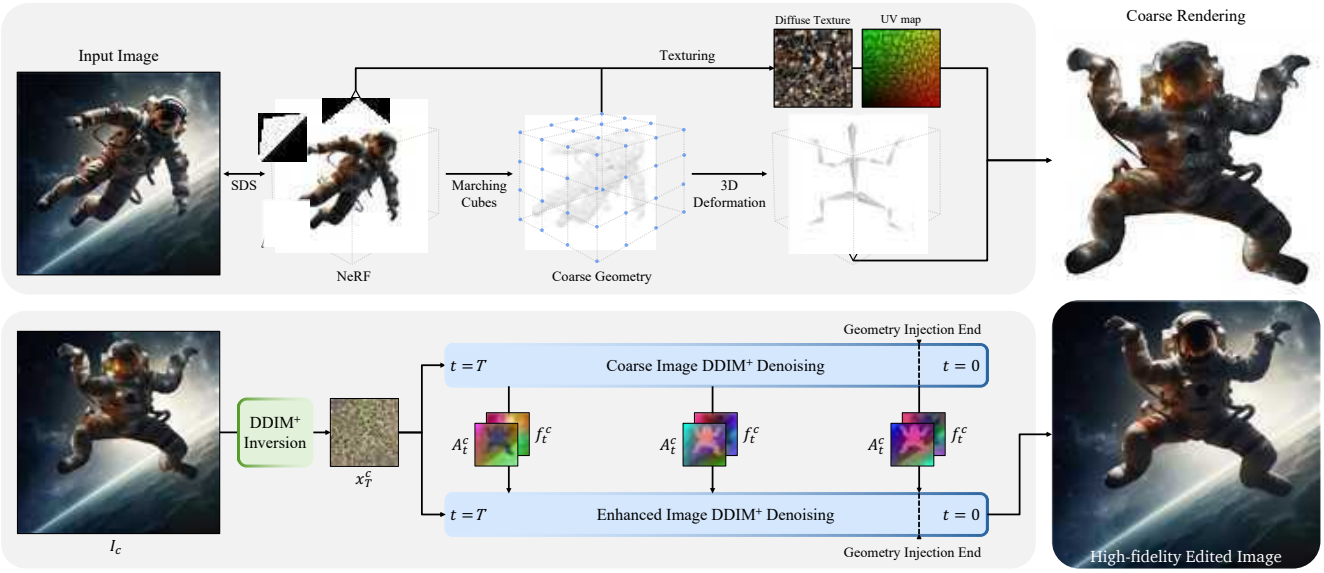

Figure 3. Overview of our *Image Sculpting* pipeline, DDIM+ represents DDIM with the DreamBooth fine-tuned and depth controlled model. The process begins by converting the input image into a textured 3D model through a de-rendering process. This model is then prepared for interactive deformation by creating a skeleton and calculating skinning weights. The user can modify the skeleton to deform the model, resulting in an initial coarse image. To refine this edited image, we invert the coarse rendering I_c into the noise $\boldsymbol{x}_T^c$. We then inject self-attention maps $\boldsymbol{A}_t^c$ and feature maps $\boldsymbol{f}_t^c$ from the initial image's denoising process into the enhanced image denoising steps. This technique helps in preserving the geometry of the modified object while restoring the visual quality of the edited image.

Linear Blend Skinning The most popular deformation approach is linear blend skinning [30], which defines a space deformation function as a blended average of a set of affine transformations weighted by shape-aware scalar functions, often computed with methods based on solutions of PDEs on surfaces [29] or manually edited. This approach offers complete control and flexibility, as the affine transformation can be attached to points, vertexes of a cage, or segments in a skeleton [6].

Our approach We can use any of these algorithms to precisely control the shape deformation and, thus, the rendered image. We show an example of one representative method for each class in Fig 2, and we leave as future work additional automation of this step.

4. Methods

Given a single 2D image, our objective is to enable precise manipulation of the objects and their orientations in 3D space, before converting this back into a high-quality edited 2D image. To achieve this, we have developed a novel editing pipeline tailored for image sculpting (see Fig 3) composed of three steps: (1) We initially convert the input image into a 3D model, (2) the 3D model is edited by deforming it in 3D space, and (3) we use a coarse-to-fine generative enhancement pipeline to turn the coarse rendering of the 3D model into a high-fidelity image.

4.1. De-Rendering and Deformation

Given an image of an object, our goal is to perform 3D reconstruction to obtain its 3D model.

Image to NeRF With advancements in text-to-image foundation models [63] and the viewpoint-conditioned image translation model [41], our initial step involves segmenting the selected object from the input image using SAM [35]. Building upon this, we then train a NeRF using Score Distillation Sampling (SDS) [58].

NeRF to 3D Model We use the implementation in threestudio [22] to convert a NeRF volume into a mesh. This algorithm transforms the volume density into a signed distance function, extracts an isosurface [44], and parameterizes it [84] for texture mapping [71]. The texture is extracted by differentiable rendering [36].

3D Model Deformation After obtaining the 3D model, a user can manually construct a skeleton and interactively manipulate it by rotating the bones to achieve the target pose. The mesh deformation affects the vertex positions of the object but not the UV coordinates used for texture mapping; this procedure thus deforms the texture mapped on the object following its deformation.

However, the resulting image quality depends on the 3D reconstruction's accuracy, which, in our case, is coarse and insufficient for the intended visual outcome (Fig 3). Therefore, we rely on an image enhancement pipeline to convert the coarse rendering into a high-quality output.

Figure 4. Comparison of our final method with various baseline methods and ablations. Our approach effectively maintains the geometric information while ensuring the texture quality. In contrast, other methods typically preserve either the texture or the geometry, but not both.

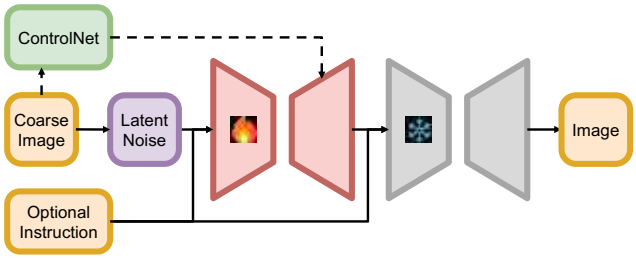

Figure 5. Overview of the coarse-to-fine generative enhancement model architecture. The red module denotes the one-shot DreamBooth [64], which requires tuning; the grey module is the SDXL Refiner [5], which is frozen in our experiments.

4.2. Coarse-to-Fine Generative Enhancement

This section focuses on blending a coarsely rendered image back to its original background. The aim is to restore textural details while keeping the edited geometry intact. Image restoration and enhancement are commonly approached as image-to-image translation tasks [78], leveraging the strong correlation between the source and target images. Our challenge, however, presents a unique scenario: despite overall similarities in appearance and texture between the input and desired output, the input object's geometry changes, sometimes significantly, after user editing.

In exploring possible solutions, one approach is to use subject-driven personalization techniques like DreamBooth [64]. They aim to preserve key details from the input, but might compromise the edited geometry. Alternatively, image-to-image translation methods like SDEdit [46] can be used to preserve the edited geometry, but this might disturb the textural consistency with the original image. This dichotomy was clear in our preliminary study, as shown in Fig 4. SDEdit can maintain the geometry, but it was unable to accurately replicate the textures. On the other hand, DreamBooth produced high-fidelity outputs, but struggled to preserve both the texture and geometry effectively.

To address the balance between preserving texture and geometry, our approach begins by "personalizing" a pre-trained text-to-image diffusion model. To capture the object's key features, we fine-tune the diffusion model with DreamBooth on *one* input reference image. To maintain the geometry, we adapt a feature and attention injection technique [76], originally designed for semantic layout control. Furthermore, we incorporate depth data from the 3D model through ControlNet [87]. We find this integration crucial in minimizing uncertainties during the enhancement process.

One-shot Dreambooth DreamBooth [64] fine-tunes a pre-trained diffusion model with a few images for subject-driven generation. The original DreamBooth paper [64] has shown its ability to leverage the semantic class priors to generate novel views of an object, given only a few frontal images of the subject. This aspect is particularly useful in our setting, since the coarse rendering we work with lacks explicit viewpoint information. In our application, we train DreamBooth using just a single example, which is the input image. Notably, this one-shot approach with DreamBooth also effectively captures the detailed texture, thereby filling in the textural gaps present in the coarse rendering.

Depth Control We use depth ControlNet [87] to preserve the geometric information of user editing. The depth map is rendered directly from the deformed 3D model, bypassing the need for any monocular depth estimation. For the background region, we don't use the depth map. This depth map serves as a spatial control signal, guiding the geometry generation in the final edited images. However, relying solely on depth control is not sufficient – although it can preserve the geometry to some extent, it still struggles in local, more nuanced editing, such as capturing the specific shapes of a pumpkin's eyes or the bent legs of a chair (Fig 4).

Feature Injection To better preserve the geometry, we use feature injection. As demonstrated in Fig 3, this step begins with DDIM inversion [74] (with the DreamBooth fine-tuned, depth controlled diffusion model) of the coarse rendering image to obtain the inverted latents. At each denoising step, we denoise the inverted latent of the coarse ren-

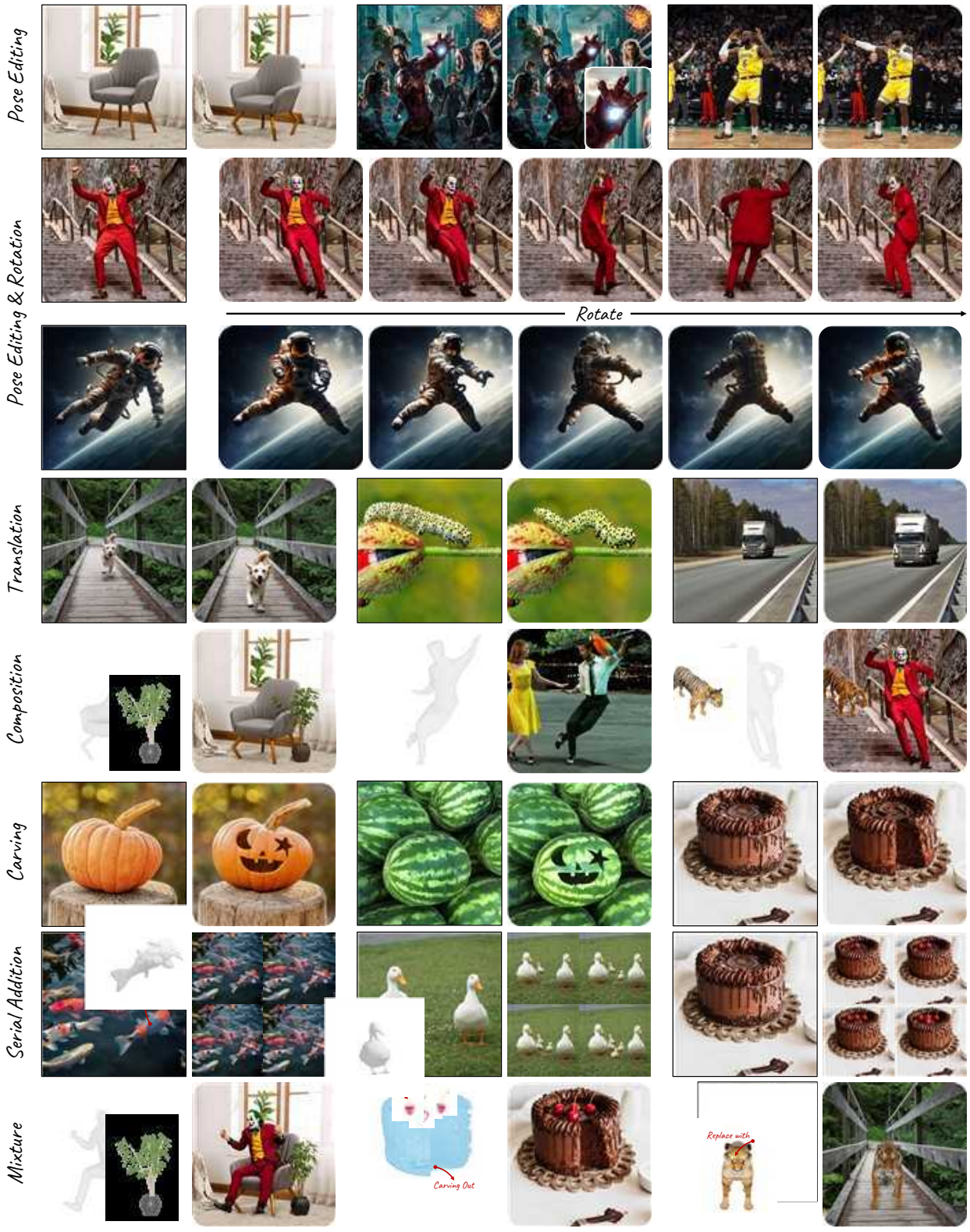

Figure 6. A compilation of qualitative results from six image editing tasks. Additionally, we include additional examples (termed as '*Mixture*' in the final row) to illustrate the versatile combination of these capabilities.

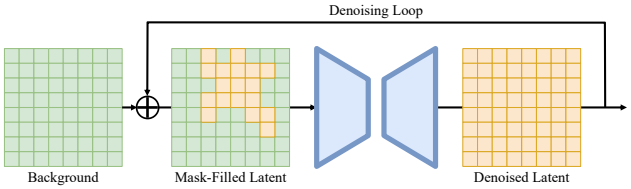

Figure 7. Our blend-in process. At every denoising step, we mask the background areas and blend them with the unmasked regions from the denoised latent. This process helps maintain visual coherence and preserve the background.

Figure 8. Our blend-in process yields visually harmonious results. *Top:* Results from direct copy-pasting. *Bottom:* Our results.

dering along with the latent of the refined image, extracting their respective feature maps (from the residual blocks) and self-attention maps (from the transformer blocks). It has been shown in [76] that the feature maps carry semantic information, while the self-attention maps contain the geometry and layout of the generated images. By overriding the feature and self-attention maps during the enhanced image denoising steps with those from the coarser version, we ensure the geometry of the enhanced image can reflect those of the coarse rendering. The pseudo code for our generative enhancement is detailed in Appendix A. Note that our method differs from the original Plug-and-Play use cases: we use feature injection to preserve the geometry during the coarse-to-fine process rather than translating the image according to a new text prompt. We present the injection layer selection and the replacement schedule in Section 5.

Background Blend-In To maintain the consistency of the background between the input and edited images, we first inpaint the area initially occupied by the object in the input image, thus obtaining an unobstructed background. However, another challenge arises in merging the edited object into this background smoothly. Merely copy-pasting it onto the background leads to an unrealistic visual effect, such as the improper water reflections over the fish and the absence of shadow casting from the truck (Fig 8). To overcome this, as demonstrated in Fig 7, our approach involves masking the background areas during the denoising steps

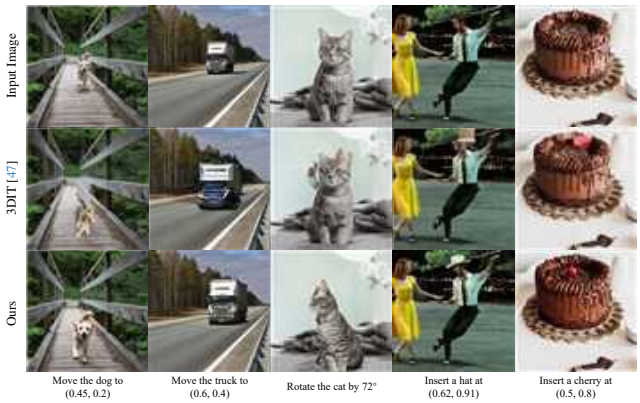

Figure 9. Comparisons with OBJect-3DIT [47] on object translation, rotation, and composition tasks.

Figure 10. Comparisons with DragDiffusion [70] and ControlNet [87] on pose editing. These techniques face difficulties in handling complex pose modifications.

to preserve their original background. This means we retain the unedited background by blending the unmasked (edited) regions from the denoising step with the masked (original) background. We use SDXL [57] as our pre-trained text-to-image model, which includes a refiner module by default. We keep this module in our pipeline, as empirically it slightly enhances the results by reducing artifacts.

5. Experiments

Experimental Setup we follow [22] to obtain the initial NeRF representation and to extract the textured 3D model. We use Instant-NGP [50] and a grid size of 256 for the 3D model extraction from NeRF. During the coarse-to-fine generative enhancement process, for one-shot DreamBooth, we fine-tune the SDXL-1.0 [57] model using LoRA [26] for 800 steps with a learning rate of 1e-5. For feature injection stage, we utilize all the self-attention layers of the SDXL decoder and the first block of the SDXL's upsampling decoder. We set $\tau_A = 0.5$ and $\tau_f = 0.2$. The SDXL refiner is applied after $t = 0.1T$. For background inpainting, we use Adobe generative fill [3].

Figure 11. Comparisons with InstructPix2Pix [10] and DALL·E 3 [53] on serial addition. These text-based editing methods fail to follow precise and quantifiable instructions.

Methods	DINO↑	CLIP-I↑	D-RMSE↓
Original Coarse Rendering	0.758	0.902	0.00
SDEdit [46] ($t_0 = 0.4$)	0.788	0.919	1.71
SDEdit [46] ($t_0 = 0.6$)	0.800	0.920	2.12
Ours w/o Feature Injection	0.848	0.925	2.33
Ours w/o Depth Control	0.851	0.921	2.15
Ours	0.853	0.921	1.99

Table 1. Ablation studies of the enhancement methods on *SculptingBench*. DINO score and CLIP-I score measure the textural details, and D-RMSE measures the geometric fidelity. We observe that depth control and feature injection can significantly enhance texture quality while maintaining geometric consistency.

Qualitative Results We showcase qualitative results in Fig 6, covering six precise image editing tasks. Detailed descriptions of these tasks are presented in Appendix B. Qualitatively, our method combines the creative freedom of generative models with the precision of graphics pipelines to achieve precise, quantifiable, and physically plausible outcomes for object editing across a variety of scenarios.

Our approach introduces new editing features through precise 3D geometry control, a capability not present in existing methods. We compare our method with the state-of-the-art object editing techniques for a comprehensive analysis. In Fig 9, we show that 3DIT [47], designed for 3D-aware editing via language instructions, faces limitations when applied to real, complex images, largely because its training is based on a synthetic dataset. In Fig 10, we compare the pose editing ability with DragDiffusion [70] and ControlNet [87]. This comparison reveals that these methods encounter difficulties with complex pose manipulations because they are constrained to the 2D domain. Furthermore, in Fig 11, we show how text-based editing methods like InstructPix2Pix [10] and DALL·E 3 [53] struggle with precise and quantifiable instructions.

Ablation Studies We create a new dataset *SculptingBench* to evaluate our new image editing capabilities. This dataset contains 28 images covering six categories: pose editing, rotation, translation, composition, carving, and serial addition (see Appendix C). We perform quantitative studies using different coarse-to-fine enhancement methods. To measure the visual similarity between the edited images and the original ones, particularly in terms of maintaining textural details through the editing process, we employ DINO and CLIP-I scores [64] as our metrics. To evaluate the geometric fidelity of user edits after enhancement, we introduce a novel metric, *D-RMSE*. This metric is specifically created to evaluate how well geometric information is retained after the enhancement procedure. *D-RMSE* measures the discrepancies between the depth maps of the coarse renderings and their enhanced counterparts:

$$\text{D-RMSE} = \sqrt{\mathbb{E}\left[(\text{depth}_{\text{coarse}} - \text{depth}_{\text{enhanced}})^2\right]}$$

where $\text{depth}_{\text{coarse}}$, $\text{depth}_{\text{enhanced}}$ denote the depth maps MiDaS [61] estimates, for the coarse rendering and the enhanced output image, respectively. In Table 1, we show that without any enhancement, the textural quality metrics (DINO and CLIP-I scores) are quite low. SDEdit effectively preserves the edited geometry with a low D-RMSE, yet the visual quality significantly deteriorates compared to the original image (see Fig. 4). Our method offers a more advantageous balance, significantly enhancing texture quality as demonstrated by higher DINO and CLIP-I scores, while preserving geometric consistency, evidenced by a low D-RMSE score. We observe that both feature injection and depth control contribute to enhanced geometric consistency and can lead to further improvement when used together. Additionally, we conduct an empirical study to explore the ideal number of self-attention layers for injection. Fig 12 shows that more layers improve alignment with user edits. In our work, we use all layers for injection.

6. Limitation

Our method is an initial step towards integrating traditional geometric processing with advanced diffusion-based generative models for precise object editing. Yet, it has limitations. A significant challenge is the dependency on the quality of single-view 3D reconstruction, which is anticipated to improve over time. Additionally, mesh deformation often requires some manual efforts for model rigging. Future research might explore data-driven techniques [37] to automate this process. The output resolution of our pipeline also falls short of industrial rendering systems, and incorporating super-resolution methods could be a solution for future improvements. Another issue is the lack of background lighting adjustment, which undermines the realism of the scene; future work could benefit from integrating dynamic (re-)lighting techniques. We present some instances of failure our our system in Appendix D.

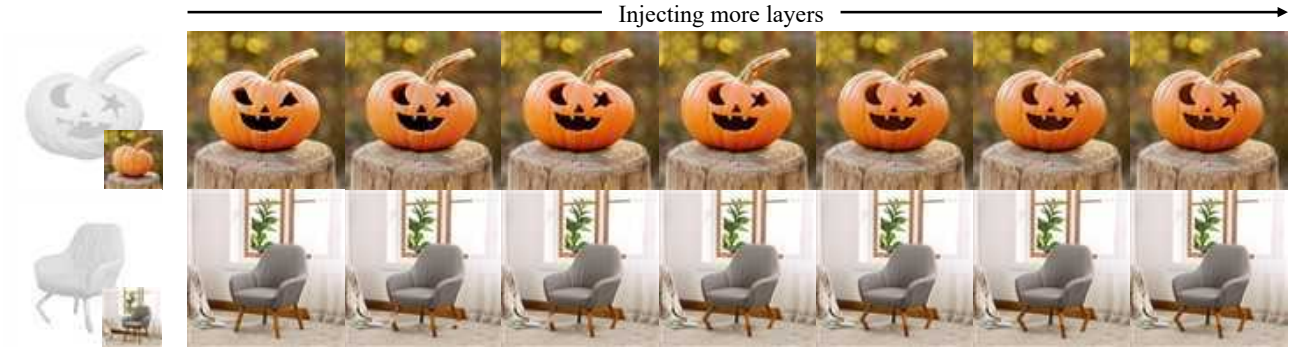

Figure 12. Ablation studies on feature injection layers. From left to right, progressively injecting more self-attention layers can result in increasingly improved alignment with user edits.

Acknowledgements. We thank Ellis Brown, Fred Lu, Sanghyun Woo, Adithya Iyer and Oscar Michel for helpful discussions. The research is partly supported by Intel, Cirrascale and the Google TRC program.

References

[1] Rameen Abdal, Peihao Zhu, John Femiani, Niloy Mitra, and Peter Wonka. CLIP2StyleGAN: Unsupervised extraction of StyleGAN edit directions. In *SIGGRAPH*, 2022. 3

[2] Adobe. Adobe After Effects. https://www.adobe.com/products/aftereffects.html, 2023. 2

[3] Adobe. Adobe Firefly. https://www.adobe.com/sensei/generative-ai/firefly.html, 2023. 7

[4] Adobe. Adobe Substance 3D. https://www.adobe.com/creativecloud/3d-ar.html, 2023. 2

[5] Stability AI. Stable Diffusion XL Refiner 1.0. https://huggingface.co/stabilityai/stable-diffusion-xl-refiner-1.0, 2023. 5

[6] Ilya Baran and Jovan Popović. Automatic rigging and animation of 3D characters. *TOG*, 2007. 4, 12

[7] D. Bommes, B. Lévy, N. Pietroni, E. Puppo, C. Silva, M. Tarini, and D. Zorin. State of the art in quad meshing. In *STARs*, 2012. 3

[8] Mario Botsch and Olga Sorkine. On linear variational surface deformation methods. *TVCG*, 2008. 3

[9] Mario Botsch, Leif Kobbelt, Mark Pauly, Pierre Alliez, and Bruno Lévy. *Polygon Mesh Processing*. AK Peters, 2010. 3

[10] Tim Brooks, Aleksander Holynski, and Alexei A Efros. InstructPix2Pix: Learning to follow image editing instructions. In *CVPR*, 2023. 2, 3, 8

[11] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. 2019. 3

[12] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *CVPR*, 2022. 3

[13] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. ShapeNet: An information-rich 3D model repository. *arXiv:1512.03012*, 2015. 3

[14] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-XL: A universe of 10M+ 3D objects. In *NeurIPS*, 2023. 3

[15] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3D objects. In *CVPR*, 2023. 2, 3

[16] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, 2021. 3

[17] Ana Dodik, Oded Stein, Vincent Sitzmann, and Justin Solomon. Variational barycentric coordinates. *TOG*, 2023. 3

[18] Gerald Farin. *Curves and Surfaces for CAGD: A Practical Guide*. Morgan Kaufmann Publishers Inc., 5th edition, 2001. 3

[19] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015. 3

[20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. In *NeurIPS*, 2014. 3

[21] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. StyleNeRF: A style-based 3D-aware generator for high-resolution image synthesis. In *ICLR*, 2022. 3

[22] Yuan-Chen Guo, Ying-Tian Liu, Ruizhi Shao, Christian Laforte, Vikram Voleti, Guan Luo, Chia-Hao Chen, Zi-Xin Zou, Chen Wang, Yan-Pei Cao, and Song-Hai Zhang. threestudio: A unified framework for 3D content generation. https://github.com/threestudio-project/threestudio, 2023. 4, 7

[23] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 3

[24] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. In *ICLR*, 2023. 2, 3

[25] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao

Tan. LRM: Large reconstruction model for single image to 3D. *arXiv:2311.04400*, 2023. 2

[26] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2021. 7

[27] Autodesk Inc. AutoDesk 3ds Max 2023. https://www.autodesk.com/products/3ds-max/overview, 2023. 2

[28] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 3

[29] Alec Jacobson, Ilya Baran, Jovan Popović, and Olga Sorkine. Bounded biharmonic weights for real-time deformation. *TOG*, 2011. 4

[30] Alec Jacobson, Zhigang Deng, Ladislav Kavan, and JP Lewis. Skinning: Real-time shape deformation. *TOG*, 2014. 3, 4

[31] Tao Ju, Scott Schaefer, and Joe Warren. Mean value coordinates for closed triangular meshes. *TOG*, 2005. 3

[32] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 3

[33] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020. 3

[34] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. In *CVPR*, 2023. 2

[35] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. *arXiv:2304.02643*, 2023. 4

[36] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *TOG*, 2020. 4

[37] Peizhuo Li, Kfir Aberman, Rana Hanocka, Libin Liu, Olga Sorkine-Hornung, and Baoquan Chen. Learning skeletal articulations with neural blend shapes. *TOG*, 2021. 8

[38] Yukang Lin, Haonan Han, Chaoqun Gong, Zunnan Xu, Yachao Zhang, and Xiu Li. Consistent123: One image to highly consistent 3D asset using case-aware diffusion priors. *arXiv:2309.17261*, 2023. 2

[39] Pengyang Ling, Lin Chen, Pan Zhang, Huaian Chen, and Yi Jin. FreeDrag: Point tracking is not you need for interactive point-based image editing. *arXiv:2307.04684*, 2023. 2, 3

[40] Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Zexiang Xu, Hao Su, et al. One-2-3-45: Any single image to 3D mesh in 45 seconds without per-shape optimization. *arXiv:2306.16928*, 2023. 2, 3

[41] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3D object. In *ICCV*, 2023. 2, 3, 4

[42] Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. SyncDreamer: Generating multiview-consistent images from a single-view image. *arXiv:2309.03453*, 2023.

[43] Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, et al. Wonder3D: Single image to 3D using cross-domain diffusion. *arXiv:2310.15008*, 2023. 2

[44] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *Seminal graphics: pioneering efforts that shaped the field*. ACM, 1998. 4

[45] Nadia. Magnenat-Thalmann, Richard Laperrière, and Daniel Thalmann. Joint-dependent local deformations for hand animation and object grasping. In *GI*. CIPS, 1989. 2

[46] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *ICLR*, 2022. 5, 8

[47] Oscar Michel, Anand Bhattad, Eli VanderBilt, Ranjay Krishna, Aniruddha Kembhavi, and Tanmay Gupta. OBJECT 3DIT: Language-guided 3D-aware image editing. *arXiv:2307.11073*, 2023. 2, 3, 7, 8

[48] MidJourney. MidJourney. www.midjourney.com. 3

[49] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *CVPR*, 2023. 3

[50] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *TOG*, 2022. 7

[51] Alexander Quinn Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob Mcgrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. 2022. 3

[52] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *CVPR*, 2020. 3

[53] OpenAI. DALL·E 3 System Card. https://openai.com/research/dall-e-3-system-card, 2023. 3, 8

[54] Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimitra Meka, and Christian Theobalt. Drag Your GAN: Interactive point-based manipulation on the generative image manifold. In *TOG*, 2023. 2, 3

[55] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery. In *ICCV*, 2021. 3

[56] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *CVPR*, 2023. 3

[57] 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:2307.01952*, 2023. 7, 12

[58] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. DreamFusion: Text-to-3D using 2D diffusion. *ICLR*, 2023. 3, 4

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

[60] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv:2204.06125*, 2022. 1, 3

[61] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *TPAMI*, 2020. 3, 8

[62] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3D: Large-scale learning and evaluation of real-life 3D category reconstruction. In *ICCV*, 2021. 3

[63] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 1, 3, 4

[64] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. DreamBooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *CVPR*, 2023. 5, 8, 12

[65] 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. In *NeurIPS*, 2022. 1, 3

[66] Thomas W. Sederberg and Scott R. Parry. Free-form deformation of solid geometric models. In *Computer Graphics*. ACM, 1986. 3

[67] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *CVPR*, 2020. 3

[68] Ruoxi Shi, Hansheng Chen, Zhuoyang Zhang, Minghua Liu, Chao Xu, Xinyue Wei, Linghao Chen, Chong Zeng, and Hao Su. Zero123++: a single image to consistent multi-view diffusion base model. *arXiv:2310.15110*, 2023. 2

[69] Yichun Shi, Peng Wang, Jianglong Ye, Long Mai, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation. *arXiv:2308.16512*, 2023. 3

[70] Yujun Shi, Chuhui Xue, Jiachun Pan, Wenqing Zhang, Vincent YF Tan, and Song Bai. DragDiffusion: Harnessing diffusion models for interactive point-based image editing. *arXiv:2306.14435*, 2023. 2, 3, 7, 8

[71] Peter Shirley and Steve Marschner. *Fundamentals of Computer Graphics*. AK Peters, 2009. 4

[72] Eftychios Sifakis and Jernej Barbic. Fem simulation of 3d deformable solids: A practitioner's guide to theory, discretization and model reduction. *TOG*, 2012. 3

[73] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3D-structure-aware neural scene representations. In *NeurIPS*, 2019. 3

[74] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021. 5, 12

[75] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *SGP*, 2007. 2, 3

[76] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *CVPR*, 2023. 2, 3, 5, 7, 12

[77] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2D diffusion models for 3D generation. In *CVPR*, 2023. 3

[78] Zhihao Wang, Jian Chen, and Steven CH Hoi. Deep learning for image super-resolution: A survey. *TPAMI*, 43(10):3365–3387, 2020. 5

[79] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. In *NeurIPS*, 2023. 3

[80] Chao-Yuan Wu, Justin Johnson, Jitendra Malik, Christoph Feichtenhofer, and Georgia Gkioxari. Multiview compressive coding for 3d reconstruction. In *CVPR*, 2023. 3

[81] Zongze Wu, Dani Lischinski, and Eli Shechtman. Stylespace analysis: Disentangled controls for stylegan image generation. In *CVPR*, 2021. 3

[82] Weihao Xia, Yujiu Yang, Jing-Hao Xue, and Baoyuan Wu. Tedigan: Text-guided diverse face image generation and manipulation. In *CVPR*, 2021. 3

[83] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *ICCV*, 2015. 3

[84] Jonathan Young. xatlas. https://github.com/jpcy/xatlas, 2023. 4

[85] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. PixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021. 3

[86] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. Scaling autoregressive models for content-rich text-to-image generation. *TMLR*, 2023. 1

[87] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, 2023. 3, 5, 7, 8

[88] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 3

A. Generative Enhancement Details

Algorithm 1: Generative Enhancement

Define: Pre-trained 2D text-to-image diffusion model M, input image I, coarse image I_c, enhanced image I_f, inversion prompt y_{inv}, prompt y, depth map D

$\hat{M} \leftarrow$ FINE-TUNE DREAMBOOTH(I)
$\boldsymbol{x}_0^c, \dots \boldsymbol{x}_T^c \leftarrow$ DDIM-INVERSION(I_c, y_{inv}, D; $\hat{M}$);
$\boldsymbol{x}_T^f \leftarrow \boldsymbol{x}_T^c$
for $t \leftarrow T$ **to** 0 **do**
  $\boldsymbol{f}_t^c, \boldsymbol{A}_t^c \leftarrow \epsilon_\theta(\boldsymbol{x}_t^c, y_{inv}, D, t; \hat{M})$
  $\boldsymbol{f}_t^f, \boldsymbol{A}_t^f \leftarrow \epsilon_\theta(\boldsymbol{x}_t^f, y, D, t; \hat{M})$
  if $t > \tau_f$ **then** $\boldsymbol{f}_t^f \leftarrow \boldsymbol{f}_t^c$
  if $t > \tau_A$ **then** $\boldsymbol{A}_t^f \leftarrow \boldsymbol{A}_t^c$
  $\boldsymbol{\epsilon}_{t-1}^f \leftarrow \epsilon_\theta(\boldsymbol{x}_t^f, y, D, t; \boldsymbol{f}_t^f, \boldsymbol{A}_t^f; \hat{M})$
  $\boldsymbol{x}_{t-1}^f \leftarrow$ DDIM-DENOISING($\boldsymbol{x}_t^f, \boldsymbol{\epsilon}_{t-1}^f$; $\hat{M}$)
end
Result: $I_f \leftarrow$ DECODER($\boldsymbol{x}_0^f$)

The generative enhancement pipeline starts with fine-tuning DreamBooth [64] with the input image. Subsequently, we apply depth control DDIM inversion [74] to the coarse rendering image. The prompt y_{inv}, which describes the coarse rendering, is used to obtain the inverted latent for each time step. During each denoising step, we denoise the inverted latent of the coarse rendering and the latent of the refined image, extracting their respective feature maps, $\boldsymbol{f}_t^c$ and $\boldsymbol{f}_t^f$, as well as their self-attention maps $\boldsymbol{A}_t^c$ and $\boldsymbol{A}_t^f$. This step is formulated as:

$$(\boldsymbol{f}_t^c, \boldsymbol{A}_t^c) \leftarrow \epsilon_\theta(\boldsymbol{x}_t^c, y_{inv}, D, t)$$
$$(\boldsymbol{f}_t^f, \boldsymbol{A}_t^f) \leftarrow \epsilon_\theta(\boldsymbol{x}_t^f, y, D, t)$$

Here, $\epsilon_\theta(\cdot)$ is the text-to-image diffusion model, specifically in our context, the Stable Diffusion XL [57] model. For the coarse rendering, the latent is denoted by $\boldsymbol{x}_t^c$, the inversion prompt by y_{inv}, and the depth map by D. For the refined image, the latent is represented by $\boldsymbol{x}_t^f$, and the prompt by y. Following Plug-and-Play [76], we replace the feature and self-attention maps of the enhanced image with those from the coarse input:

$$\boldsymbol{\epsilon}_{t-1}^f = \epsilon_\theta(\boldsymbol{x}_t^f, y, D, t; \boldsymbol{f}_t^f, \boldsymbol{A}_t^f)$$

Here $\epsilon_\theta(\cdot; \boldsymbol{f}_t^f, \boldsymbol{A}_t^f)$ represents the model with replaced feature and self-attention maps, and $\boldsymbol{\epsilon}_{t-1}^f$ is the prediction for the refined image. Replacement stops once the current time step is below the thresholds τ_f and τ_A. The threshold is important because the feature/self-attention maps may contain undesired artifacts from coarse 3D reconstruction and mesh deformation.

B. Implementation Details

In this section, we provide implementation details of all our 6 tasks.

Pose Editing Pose editing is carried out by manually creating a skeleton for each 3D model and computing its skinning weights [6]. The object's pose is edited by adjusting the skeleton's bones. A text prompt is not required to describe the pose.

Rotation Rotation is achieved by spinning the 3D model around its centroid. This allows us to rotate the model at any angle and then render it back into a 2D image. However, it becomes challenging to discern the viewpoint (*e.g.*, front, back, or side), given only the coarse rendering image. Optional text prompts are helpful in guiding the denoising step and preventing the Janus Problem. If the rotation angle ranges from $[-45°, 45°]$, we add "*front view*" to the text prompt. For angle between $[135°, 225°]$, we append "*back view*" to the text prompt. For all other angles, we use "*side view*".

Translation Translation can be done by moving the 3D model within the 3D space in any direction and over any specific distance. As the translated model is rendered, the camera perspective adjusts accordingly. As illustrated in Fig 6, moving the dog or the truck closer to the camera results in an enlarged image of the object, consistent with the camera's perspective.

Composition Our method allows for the addition of artist-created 3D objects to the scene. In Fig 6, we insert various models into the scene. Despite the 3D models not being of high quality, our coarse-to-fine strategy significantly enhances their detail, as evident in the tiger example where the texture displays hair details and a realistic face in the final output, blending well with the environment. Note that these models are not used for fine-tuning during DreamBooth training. In certain cases, text prompts prove helpful in guiding the denoising step and supplementing our geometric guidance.

Carving Beyond mesh deformation, our method enables cutting and removing parts of the mesh through the use of molds. In Fig 6, a moon-and-star-shaped mold is positioned against a pumpkin's surface. By calculating and excising the overlapping areas, the resulting mesh resembles a finely carved pumpkin in specific shapes.

Serial Addition Similar to composing elements, we can take meshes reconstructed from images and integrate them into the scene one by one. In Fig 6, we adjust each fish and duck's size, pose and their orientation before adding it to the scene. Our approach realistically merges the coarse 3D fish models into the scene, maintaining a realistic appearance even with reflections on the water's surface.

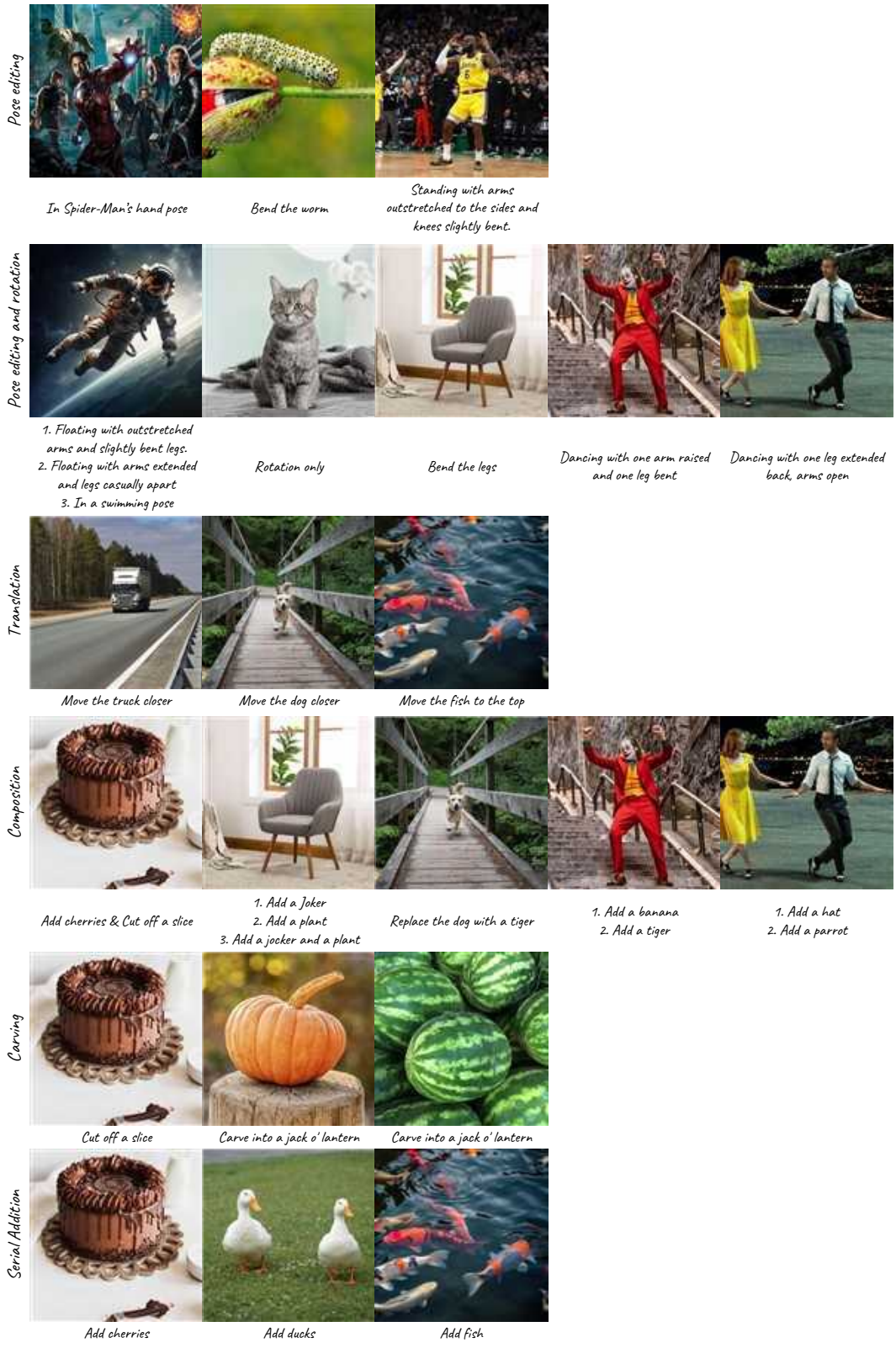

Figure 13. All 28 edits and 15 input images of our *SculptingBench*. We provide textual descriptions of the edits here. However, in practice we aim to make precise, quantifiable edits directly to 3D models, without relying on text prompts.

C. *SculptingBench*

Our *SculptingBench* dataset comprises 28 edits applied to 15 images, encompassing each of the 6 editing tasks we have developed. The full dataset is illustrated in Fig. 13. These instances present significant challenges to current object editing techniques, thereby serving as an ideal platform for testing and developing precise object editing methods.

D. Failure Cases

One key limitation lies in its dependence on the quality and reliability of single-view reconstruction techniques, particularly when dealing with unseen perspectives. Any errors in this process can result in editing failures.

As demonstrated in Fig 14, the reconstruction occasionally fails to produce detailed textures, leading to a blurred face in the top row example. Challenges also arise in mesh reconstruction and extraction. The middle row displays artifacts beneath the man's armpit, stemming from imprecise reconstruction in that region. In the bottom row example, wrong color reconstruction resulted in an less realistic final color in the output.

Figure 14. Failure cases due to inaccurate reconstruction of texture, geometry, and color.