InstantSticker: Realistic Decal Blending via Disentangled Object Reconstruction

Yi Zhang[1], Xiaoyang Huang[1], Yishun Dou[2], Yue Shi[1], Rui Shi[1], Ye Chen[1], Bingbing Ni[1]*, Wenjun Zhang[1]

[1]Shanghai Jiao Tong University, Shanghai 200240, China
[2]Huawei
{yizhangphd, nibingbing}@sjtu.edu.cn

Abstract

We present InstantSticker, a disentangled reconstruction pipeline based on Image-Based Lighting (IBL), which focuses on highly realistic decal blending, simulates stickers attached to the reconstructed surface, and allows for instant editing and real-time rendering. To achieve stereoscopic impression of the decal, we introduce *shadow factor* into IBL, which can be adaptively optimized during training. This allows the shadow brightness of surfaces to be accurately decomposed rather than baked into the diffuse color, ensuring that the edited texture exhibits authentic shading. To address the issues of warping and blurriness in previous methods, we apply As-Rigid-As-Possible (ARAP) parameterization to pre-unfold a specified area of the mesh and use the local UV mapping combined with a neural texture map to enhance the ability to express high-frequency details in that area. For instant editing, we utilize the Disney BRDF model, explicitly defining material colors with 3-channel diffuse albedo. This enables instant replacement of albedo RGB values during the editing process, avoiding the prolonged optimization required in previous approaches. In our experiment, we introduce the Ratio Variance Warping (RVW) metric to evaluate the local geometric warping of the decal area. Extensive experimental results demonstrate that our method surpasses previous decal blending methods in terms of editing quality, editing speed and rendering speed, achieving the state-of-the-art.

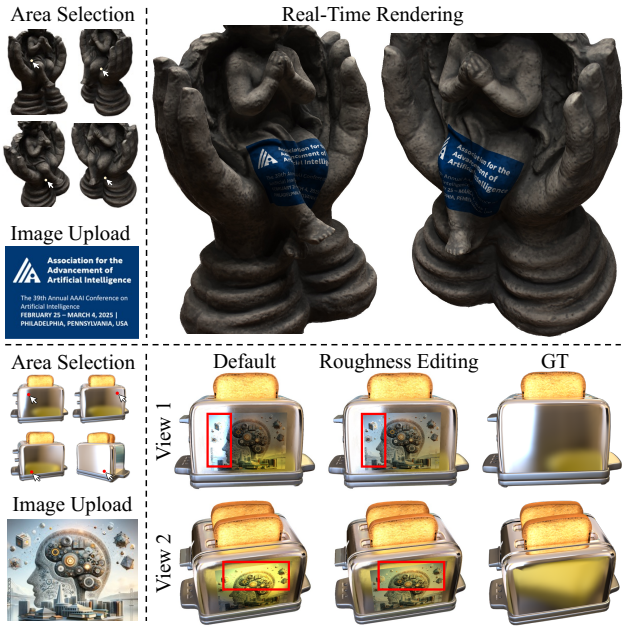

Figure 1: We select a target area in the user interface to position the decal by adjusting the viewpoint and selecting four anchors to define a quadrilateral. Then, we upload an image, which will blend realistically within this quadrilateral.

1 Introduction

Decal blending for multi-view reconstructed objects allows users to upload an image and apply it to a specified area on the 3D object. This is a flexible and controllable editing mode that creates highly realistic visual effects, making it valuable for applications in AR/VR, advertising, and digital content creation. However, highly coupled 3D representations such as NeRF and its variants (Mildenhall et al. 2020; Tang et al. 2022), and 3D Gaussian (Kerbl et al. 2023) often perform poorly in this task. To achieve realistic decal appearance, it should conform to the surface's bumps and depressions while accurately capturing environmental lighting and shadows, which necessitates disentangled reconstruction.

Since simultaneously disentangling all components, including geometry, often results in convergence to local minima, most methods focus on disentangling appearance components based on pre-reconstructed geometry. In DE-NeRF (Wu et al. 2023), a signed distance function (SDF) is first trained on multi-view images, followed by mesh reconstruction using the marching cubes algorithm (Lorensen and Cline 1987). Each vertex is assigned learnable geometry and appearance features, with lighting approximated by a learnable environment map and an implicit lighting network. During rendering, multiple points are sampled along each camera ray, with each point's feature derived from the weighted sum of its k nearest vertices' features, leading to significant computational overhead. Volume rendering is then performed along each ray, using multi-view images as supervision to optimize vertex features and lighting. During editing, a manually edited single-view rendered image is used as the optimization target. Raycasting from the camera to the

*Corresponding author: Bingbing Ni.

reconstructed mesh locates the corresponding vertices, and the editing effect is achieved by iteratively optimizing the features of these vertices. However, this process introduces significant latency in the editing workflow and causes decal warping on curved surfaces. Additionally, the decal often appears blurred due to pixel colors being influenced by multiple neighboring vertices, limiting the texture frequency.

In this paper, we develop a novel disentangled reconstruction pipeline based on IBL (Debevec 1998), enabling real-time rendering and supporting instant, highly realistic decal blending with integrated environmental lighting and shadows. Unlike DE-NeRF, we employ advanced surface reconstruction methods that provide accurate geometry, eliminating the need for calculations beyond the first-hit intersections. This enables us to adopt mesh rasterization for real-time rendering, which in turn allows us to leverage barycentric interpolation for estimating fragment properties from vertex features, further enhancing rendering efficiency. For material representation, we use the Disney BRDF model (Burley and Studios 2012), which integrates Lambertian (Pharr, Jakob, and Humphreys 2016) for diffuse reflection and microfacet BRDF (Cook and Torrance 1982) for specular reflection. The corresponding texture and material features are assigned to vertices for optimization, which is more efficient than the code-decoder framework. Additionally, the Lambertian BRDF uses a 3-channel albedo to model the diffuse component, enabling instant RGB value replacement during editing. To address warping issues in decal blending, we enable users to select an area of the mesh for ARAP parameterization (Liu et al. 2008). In this area, the albedo is represented using a UV mapping combined with a high-resolution neural texture map, rather than vertex features. This texture map is generated by a fixed UV coordinate grid combined with a multi-layer perceptron (MLP), effectively addressing the issue of insufficient texture space learning due to gaps in the training samples.

Furthermore, we tackle the limitation of IBL's inability to model shadows. Specifically, IBL only accounts for direct lighting, which leads to shadows being baked into the albedo during reconstruction, resulting in edited albedo that lacks realistic shadow effects. To address this, we introduce *shadow factor* into IBL rendering equation, which can be adaptively optimized in the pre-selected area during reconstruction. Since the diffuse albedo in the pre-selected area is generated by an MLP, which is well-known for its strong spectral bias towards producing lower frequency, smoother textures (Rahaman et al. 2019), the shadow factor is capable of capturing higher frequency shadows without global frequency limitations. Moreover, to improve the reconstruction quality of specular objects, we design a *local reflection network* to handle reflections within the object that are local and cannot be modeled by global environment maps.

In our experiment, we develop a Ratio Variance Warping (RVW) metric to evaluate decal warping by measuring the variance in scaling ratios between texture space Euclidean distances and mesh surface geodesic distances. We benchmark our method against other advanced techniques on the NeRF-Synthetic (Mildenhall et al. 2020), ShinyBlender (Verbin et al. 2022), and DTU (Jensen et al. 2014) datasets. Compared to the state-of-the-art, our method achieves comparable reconstruction quality with significantly less warping in decal blending on curved surfaces, more realistic shadow effects, and an 800× increase in rendering speed (80 FPS vs. 0.1 FPS). Additionally, we implement a simple user interface (UI). As shown in Fig. 1, the decal appears like a sticker on diffuse surfaces, while on metal surfaces, it blends with environment specular reflections by default. We can also achieve a diffuse effect on metal surfaces by manually increasing the roughness value of the blending area, making it resemble a sticker.

2 Related Work

Mesh Parameterization. Mesh parameterization is a fundamental technique in computer graphics and geometric processing, often used for texture mapping, remeshing, and other applications. LSCM (Lévy et al. 2002) approximates conformal parameterization by solving a linear system derived from conformal mapping equations. ARAP parameterization (Liu et al. 2008) minimizes local distortions by preserving local geometry as much as possible, iteratively solving linear systems to minimize an energy penalizing deviations from original edge lengths. NeuTex (Xiang et al. 2021) introduces an inverse mapping network and a cycle consistency loss to learn a 3D-to-2D texture mapping network for neural volumetric rendering. Nuvo (Srinivasan et al. 2023) presents a UV mapping method for 3D reconstructed geometry, using a neural field to create a continuous UV map optimized for visible points, resulting in editable UV mappings that are robust to ill-behaved geometry.

Material and Environment Estimation. PhySG (Zhang et al. 2021) models geometry as an SDF network and its lighting is approximated by a composition of several Spherical Gaussian (SG) functions. NvDiffRec (Munkberg et al. 2022) jointly decouples the geometry, texture, materials, and lighting from multi-view images, allowing these components to be deployed in traditional graphics engines. NeRO (Liu et al. 2023) uses accurate sampling to recover the environment lighting and the BRDF of objects while keeping the object geometry fixed. SOL-NeRF (Sun et al. 2023) uses a single SG lobe for sunlight approximation and models skylight with spherical harmonics, while shadows are calculated via raycasting and ambient occlusion. GIR (Shi et al. 2023) uses 3D Gaussians to estimate the material, illumination, and geometry of an object from multi-view images.

Decal Blending for Reconstructed Objects. NeuMesh (Yang et al. 2022) encodes disentangled geometry and texture on mesh vertices, enabling various editing functions, including decal blending. Building on NeuMesh, DE-NeRF (Wu et al. 2023) decouples appearance into texture and lighting, enabling decals to reflect environment lighting effects. Seal-3D (Wang et al. 2023) achieves shadow-like decal blending by transferring luminance offsets in HSL space from the original surface color to the target surface color. However, these methods rely on overlaying an image on single-view renderings to achieve decal blending, which often leads to warping across different views due to the optimization being limited to a single viewpoint.

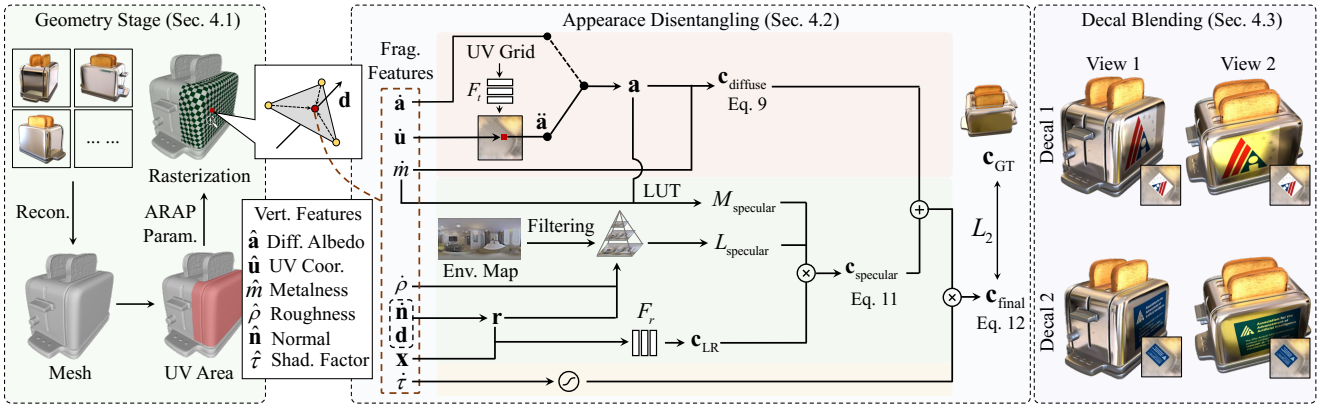

Figure 2: During the geometry stage, we reconstruct the mesh, select and parameterize a UV area, and rasterize the mesh. During appearance disentangling, we assign learnable features to each vertex and use barycentric interpolation to obtain fragment features. The environment lighting is represented as a mipmap, filtered from a learnable environment map. After reconstruction, users can upload an image and blend it into any position in UV area by overwritting its pixel colors to the albedo texture map. Please note the distinction between different forms of the same symbol. For example, the symbol $\mathbf{a}$ represents the general form used in equations, $\hat{\mathbf{a}}$ denotes the learnable vertex feature, and $\dot{\mathbf{a}}$ indicates the fragment feature interpolated from $\hat{\mathbf{a}}$.

3 Preliminary

Image-Based Lighting. IBL is a widely used real-time lighting technique in computer graphics. The core idea of IBL is to use an HDR environment map to simulate environment lighting from an infinite distance. The outgoing radiance $\mathbf{c}\left(\omega_o\right)$ in direction ω_o is calculated by

$$\mathbf{c}\left(\omega_o\right)=\int_{\Omega} L\left(\omega_i\right) f\left(\omega_i, \omega_o\right)\left(\omega_i \cdot \mathbf{n}\right) d \omega_i, \tag{1}$$

which is an integral of the product of the incident radiance $L\left(\omega_i\right)$ from direction ω_i and the BRDF $f\left(\omega_i, \omega_o\right)$. The integration domain is the hemisphere Ω around the surface normal $\mathbf{n}$ at the ray-surface intersection.

Disney BRDF. We use the Disney BRDF model (Burley and Studios 2012), which integrates Lambertian (Pharr, Jakob, and Humphreys 2016) for diffuse reflection and microfacet BRDF (Cook and Torrance 1982) for specular reflection as follows

$$f\left(\omega_i, \omega_o\right)=\underbrace{(1-m) \frac{\mathbf{a}}{\pi}}_{\text{diffuse}}+\underbrace{\frac{D F G}{4\left(\omega_i \cdot \mathbf{n}\right)\left(\omega_o \cdot \mathbf{n}\right)}}_{\text{specular}}, \tag{2}$$

where $m \in[0,1]$ and $\mathbf{a} \in[0,1]^3$ represent the metalness and diffuse albedo of the shading point, respectively. D, F and G represent the GGX normal distribution function (Walter et al. 2007), Fresnel term and geometry attenuation, respectively. They are all determined by the diffuse albedo $\mathbf{a}$, the metalness m, and the roughness $\rho \in[0,1]$, which is detailed in the appendix. Substitute Eq. 2 into Eq. 1 to get

$$\mathbf{c}\left(\boldsymbol{\omega_o}\right)=\mathbf{c}_{\text{diffuse}}+\mathbf{c}_{\text{specular}}, \tag{3}$$

in which

$$\mathbf{c}_{\text{diffuse}}=\mathbf{a}(1-m) \int_{\Omega} L\left(\omega_i\right) \frac{\omega_i \cdot \mathbf{n}}{\pi} d \omega_i, \tag{4}$$

$$\mathbf{c}_{\text{specular}}=\int_{\Omega} \frac{D F G}{4\left(\omega_i \cdot \mathbf{n}\right)\left(\omega_o \cdot \mathbf{n}\right)} L\left(\omega_i\right)\left(\omega_i \cdot \mathbf{n}\right) d \omega_i. \tag{5}$$

Split-Sum Approximation. For Eq. 5, we use the split-sum approximation (Karis and Games 2013) to simplify the integral and achieve real-time rendering via precomputation

$$\mathbf{c}_{\text{specular}} \approx \underbrace{\int_{\Omega} L\left(\omega_i\right) D\left(\omega_i, \omega_0 ; \rho\right) d \omega_i}_{L_{\text{specular}}} \cdot \underbrace{\int_{\Omega} \frac{D F G}{4\left(\omega_0 \cdot \mathbf{n}\right)} d \omega_i}_{M_{\text{specular}}}, \tag{6}$$

where L_{specular} denotes the integral of the lights on the normal distribution function $D(\omega_i, \omega_0; \rho) \in [0,1]$, which can be precomputed and stored in a mipmap. M_{specular} is the integral of the specular BRDF, which can be approximated using a precomputed look-up table (LUT) as

$$M_{\text{specular}}=(0.04 \cdot(1-m)+m \cdot \mathbf{a}) \cdot F_1+F_2, \tag{7}$$

where F_1 and F_2 are two precomputed scalars that depend on the roughness ρ and the parameters $\cos\theta = \omega_i \cdot \mathbf{n}$. The integral in Eq. 4 can likewise be precomputed, but we have another way, which will be described in Sec. 4.2.

4 Method

Our method consists of three stages, as illustrated in Fig. 2. In Sec. 4.1, we reconstruct the mesh and assign learnable appearance features to each vertex based on the material model discussed in preliminary, and explain how to calculate the appearance features and UVs of the shading point for each pixel. In Sec. 4.2, we discuss the limitations of directly applying Eq. 3 along with Eqs. 4 and 6, and introduce our improved disentangling pipeline. Finally, in Sec. 4.3, we describe the process of achieving instant decal blending.

4.1 Geometry Stage

We begin by utilizing advanced surface reconstruction techniques to generate high-quality meshes, specifically employing Voxurf (Wu et al. 2022) for diffuse objects and Ref-NeuS (Ge et al. 2023) for specular objects. After generating the mesh, we select a continuous area, referred to as

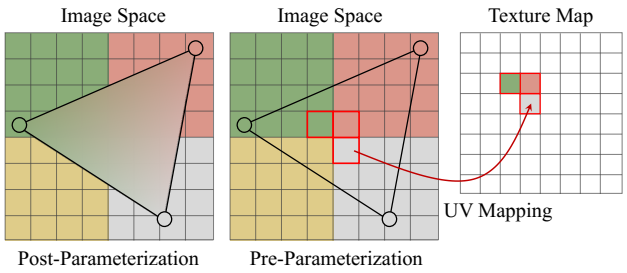

Figure 3: For post-parameterization, the rendered color within a triangle is limited to interpolating the features of its three vertices, hindering high-frequency texture representation. We address this by pre-parameterizing to obtain UVs and training a high-resolution texture map.

UV area. Each face of the mesh is assigned an indicator denoting whether it is included in UV area. We then apply ARAP parameterization (Liu et al. 2008) to UV area to obtain the UV coordinates for each vertex and allocate a sufficiently high-resolution texture map to store vertex features. For rendering, we utilize rasterization techniques, a surface-based approach commonly employed in real-time applications. The color of each pixel is determined by the first-hit fragment, which includes the z-buffer, face index buffer, and barycentric coordinates. Following (Huang et al. 2023), we use the z-buffer to estimate the intersection coordinate $\mathbf{x} = (x, y, z)$. Additionally, the ray direction $\mathbf{d}$ (which equals $-\omega_0$) is also stored in the fragments.

Following Lambertian and microfacet models, we assign each vertex the following learnable appearance features: diffuse albedo $\hat{\mathbf{a}}$, roughness $\hat{\rho}$, and metalness $\hat{m}$. Here, $\hat{\mathbf{a}}$ is a 3-dimensional vector, while both $\hat{\rho}$ and $\hat{m}$ are 1-dimensional. In addition to these learnable features, each vertex also includes two fixed attributes: normal $\hat{\mathbf{n}}$ and UV coordinate $\hat{\mathbf{u}}$ (set to (0,0) for non-UV areas). The fragment features are then calculated by performing barycentric interpolation, which yields a weighted sum of the vertex features. To distinguish these interpolated fragment features from the original vertex features, we use the dot symbol to denote them. As illustrated in Fig. 2, the final color of each pixel is computed using the interpolated fragment features: $\dot{\mathbf{a}}$, $\dot{\mathbf{u}}$, $\dot{m}$, etc.

Discussion of Rasterization. The following three reasons support our decision to use surface rasterization rather than volumetric representation employed in previous methods. Firstly, existing surface reconstruction methods (Wu et al. 2022; Ge et al. 2023; Fu et al. 2022) are capable of producing high-quality meshes, even for specular objects. This ensures that the contributions of samples beyond ray-surface intersections are negligible. Additionally, surface representations are more closely aligned with real-world objects than volumetric ones, which is crucial for achieving realistic sticker effects. Furthermore, since ray-surface intersections (or first-hit fragments) are guaranteed to lie on mesh triangles, we can utilize barycentric interpolation on vertex features, thereby avoiding the significant computational overhead of k-nearest neighbor searches.

Discussion of Pre-Parameterization. We plan to store the vertex diffuse albedo in a texture map using the UV mapping derived from ARAP, enabling surface-aligned texture editing. As noted by (Do Carmo 2016), only surfaces that are simply connected and homeomorphic to a disk can be continuously unfolded. Since most objects do not meet these criteria, they require cutting for unfolding (Srinivasan et al. 2023; Young 2016), which leads to fragmented texture spaces unsuitable for decal blending. Therefore, we focus on parameterizing only a specific part of the mesh. We refer to a naive method as **post-parameterization**, where the target area is parameterized after appearance disentangling is completed. Specifically, this method converts the vertex diffuse albedo of the target area to a texture map using UV mapping, overlays the decal image onto this texture map, and then re-applies the edited texture back to the original vertices for rendering. However, this approach may result in low-frequency details, as illustrated in Fig. 3. This occurs because a triangle can cover multiple pixels in image space, with the color determined by interpolating the three vertex colors, thereby limiting high-frequency texture detail. To address this, we propose **pre-parameterizing** the UV area before appearance disentangling. For the UV area, each fragment diffuse albedo is indexed into the texture map using its interpolated fragment UV coordinates, allowing a single triangle to correspond to multiple pixels on the high-resolution texture map. During appearance disentangling, these albedos stored in the texture map can be correctly optimized, enabling a single triangle to express multiple colors, thus significantly enhancing the frequency representation capability. Additionally, pre-parameterization enables instant editing without ARAP computation delays.

4.2 Appearance Disentangling

Neural Albedo Texture Map. For UV area, we use a learnable texture map combined with fragment UV coordinates to represent the diffuse albedo, denoted as $\ddot{\mathbf{a}}$. Regarding the learnable texture map, the naive approach is to directly optimize its pixels. However, this approach often results in incomplete learning of the texture map, as certain UVs may not be sampled during training. Therefore, we propose a neural texture map implemented as an MLP that generates color values based on the input UV coordinates.

$$\ddot{\mathbf{a}} = F_t(\gamma(\dot{\mathbf{u}})), \tag{8}$$

where F_t is a 3-layers MLP, γ is a positional encoding function. During training, at each iteration, the texture map is generated by inputting a UV grid of shape $(H, W, 2)$. This map will be precomputed for use during inference.

Diffuse Color Revising. In preliminary, we cover the precomputation of specular color. Now, we will focus on the treatment of diffuse color in Eq. 4. We point out that due to their differing physical properties, disentangling lighting and albedo from the diffuse color is significantly more challenging than disentangling specular component. The view-dependent nature of specular reflection allows for a more robust decoupling of environmental lighting. In contrast, diffuse reflection, being view-independent, arises from multiple scattering and reflection of light within the surface,

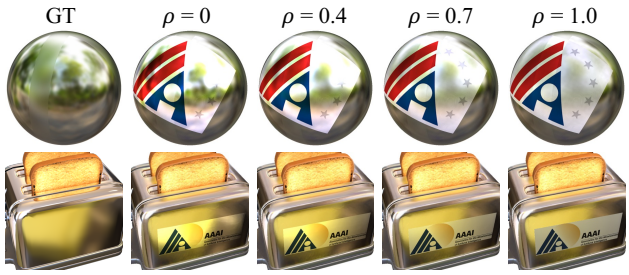

Figure 4: Manually editing the roughness of the blending area can create a diffuse effect on a specular surface.

making it more challenging to separate environment lighting from the albedo. Introducing environment lighting into the diffuse color can result in gradient descent baking the object's albedo into the lighting, leading to erroneous optimization. In many PBR techniques, the diffuse component typically excludes direct lighting and is combined with ambient lighting in a separate step, common in modern game engines and renderers (Pharr, Jakob, and Humphreys 2016). Therefore, in our approach, we simplify the diffuse term Eq. 4 by omitting the integral over the light source

$$\mathbf{c}_{\text{diffuse}} \approx \mathbf{a}(1-m). \tag{9}$$

Modeling Local Reflection. IBL employs a global environment map to simulate lighting, which generally yields favorable results. However, objects with specular surfaces may reflect parts of themselves, making it challenging for the global environment map to accurately model these local reflections. For example, the front panel of the *Toaster* in Fig. 2 reflects the pattern of its base, a phenomenon we refer to as **local reflection**. To address this, the following local reflection (LR) net is incorporated into the specular color (Eq. 6) to model the local reflection

$$\mathbf{c}_{\text{LR}} = F_r(\gamma(\mathbf{x}), \gamma(\mathbf{r})), \tag{10}$$

in which F_r is a 3-layers MLP, γ is a positional encoding function, $\mathbf{x}$ is the intersection coordinate. Inspired by Ref-NeRF (Verbin et al. 2022), we replace MLP's parameterization of view-dependent radiance with a representation of reflected radiance direction $\mathbf{r} = 2\left(\omega_o \cdot \mathbf{n}\right)\mathbf{n} - \omega_o$, instead of view direction. In this way, Eq. 6 is rewritten as

$$\mathbf{c}_{\text{specular}} = \mathbf{c}_{\text{LR}} \cdot L_{\text{specular}} \cdot M_{\text{specular}}. \tag{11}$$

Modeling Shadow. Another limitation of IBL is its inability to model shadows, as it only considers direct lighting from the environment map. This is often not an issue in reconstruction tasks, gradient optimization during reconstruction tends to bake shadows into the diffuse albedo. However, this is particularly detrimental for decal blending, as edited textures lose the 3D perception provided by shadows, significantly reducing realism. Inspired by ambient occlusion techniques (Zhukov, Iones, and Kronin 1998), we aim to enable the model in multi-view reconstruction to adaptively learn an ambient occlusion factor that controls the brightness of each shading point, thereby creating a realistic 3D effect. We refer to this factor as *shadow factor*, denoted as τ. This 1-dimensional factor directly influences the final color, indicating the intensity of light received by the shading point. We assign $\hat{\tau}$ to each vertex and use barycentric interpolation to obtain $\dot{\tau}$ for fragment. Thus, Eq. 3 is rewritten as follows

$$\mathbf{c}\left(\omega_{\boldsymbol{o}}\right) = \sigma(\tau) \cdot \left(\mathbf{c}_{\text{diffuse}} + \mathbf{c}_{\text{specular}}\right), \tag{12}$$

where σ is sigmoid function. Since the albedo in the UV area is generated by an MLP, which is well-known for its strong spectral bias towards producing low frequency textures, whereas the shadow factor is not constrained by global frequencies and can capture higher frequency shadows. In our experiments, we found that the shadow factor in the UV area can be adaptively optimized correctly without any supervision or constraints.

Pipeline. As shown in Fig. 2, we optimize the parameters using only MSE loss, denoted as L_2, between the ground truth and the final color. The parameters being optimized include the vertex features, the parameters of F_t and F_r, and an HDR environment map, represented as a float tensor. Detailed information on the filtering process used to generate mipmaps from the environment map is provided in the appendix. During inference, we precompute F_t to generate and cache the texture map for UV area, and similarly, the environment map is prefiltered and cached as a mipmap.

4.3 Decal Blending

Once the reconstruction is complete, we can upload an image and select four anchors in UV area to define a quadrilateral in the texture space. The uploaded image is then directly overlaid onto the corresponding position in the precomputed albedo texture map, instantly achieving the desired 3D editing effect. Please see the appendix for details. The decal appears like a sticker on diffuse surfaces, while on metal surfaces, it naturally blends with environment specular reflections. We can also manually adjust the roughness of the blending area to blur the environment light, creating a diffuse sticker effect, as shown in Fig. 4.

5 Experiments

5.1 Experimental Settings

Implementation Details. For convenience, we use PyTorch3D (Ravi et al. 2020) to pre-rasterize the mesh and store the fragments locally, with OpenGL (Kessenich, Sellers, and Shreiner 2016) employed for deployment. The Adam optimizer is used, with a learning rate of 5e-4 for F_t and F_r, and 5e-3 for the appearance features and environment map. Training is performed on a single NVIDIA RTX 3090 GPU and takes only 1 hour. The positional encoding dimension is set to 4, and the hidden layer dimension of MLP is 256. We utilize ARAP interface provided by Libigl library (Jacobson and Panozzo 2017).

Datasets and Baselines. We evaluate our method on NeRF-Synthetic (Mildenhall et al. 2020), Shiny Blender (Verbin et al. 2022), and DTU (Jensen et al. 2014) dataset. We mainly compare our method with the following methods capable of performing decal blending on reconstructed objects: NeuMesh, Seal-3D, and DE-NeRF.

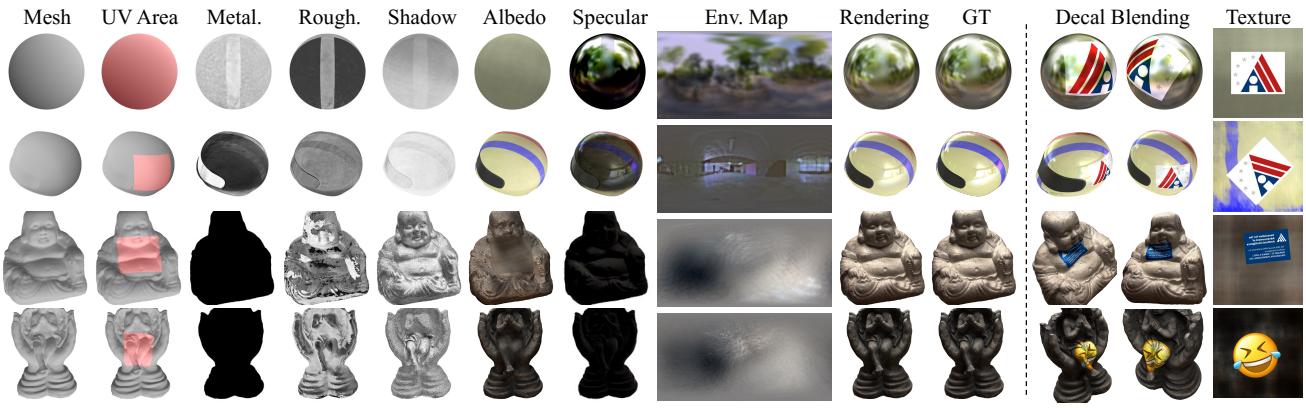

Figure 5: Disentangling and decal blending results of our method.

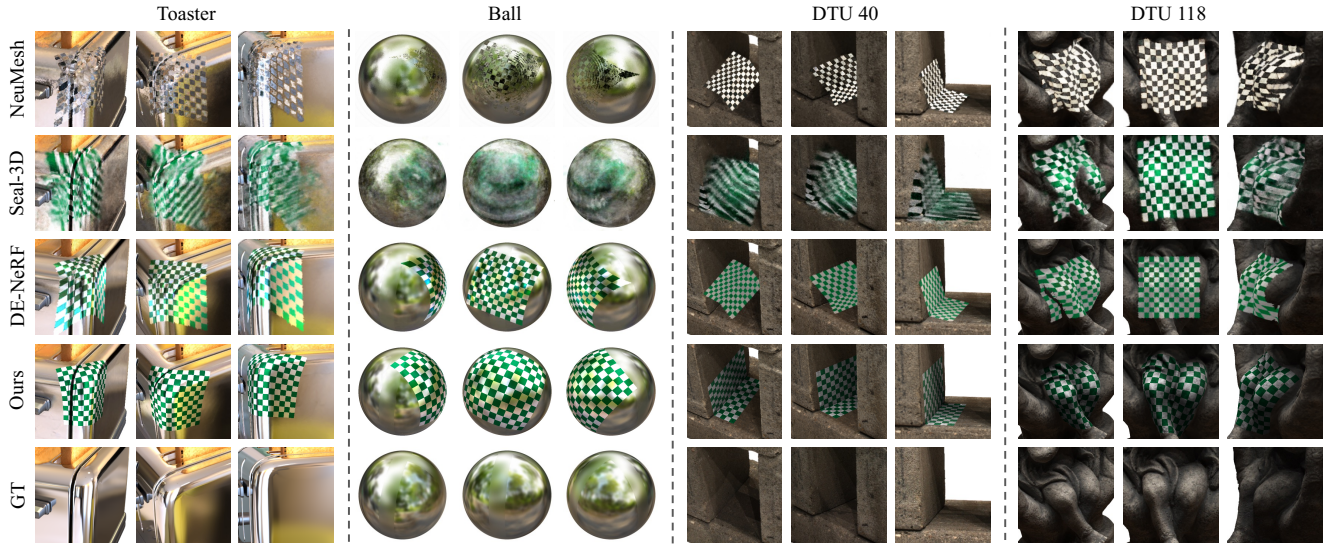

Figure 6: The results demonstrate that our method allows the decal to conform to the surface geometry with minimal warping, while the integration of lighting and shadows remains highly realistic.

5.2 Reconstruction Results

We present the components of disentangling and editing results in Fig. 5. Since disentangled reconstruction is more challenging than free-form reconstruction, we compare our approach with advanced disentangled reconstruction methods for fairness, in terms of PSNR. The results demonstrate that our reconstruction quality is comparable to that of DE-NeRF, as shown in Tab. 1. Comparison results in terms of SSIM and LPIPS can be found in the appendix.

5.3 Editing Results

Qualitative Comparison. For the qualitative evaluation of decal blending, we select a standard checkerboard pattern as the decal, apply it to two specular objects and two diffuse objects, and compare our method with three previous decal blending methods. The decal in NeuMesh appears similar in color to the object because the decoder is fixed, and only the vertex features are optimized. Seal3D, using a volumetric representation, fails to accurately locate the surface, making the decal appear embedded within the surface. Although DE-NeRF achieves decent lighting integration on specular objects, it fails to handle shadow effects. Moreover, all three methods use a manually edited single-view image as the optimization target, leading to significant warping. In contrast, our method not only incorporates environment lighting and shadows on decals but also maintains geometric alignment.

Quantitative Comparison. The Ratio Variance Warping (RVW) metric is designed to assess decal area warping. We sample N vertex pairs in the UV area, forming set V, and map them to 2D texture space as set U. Geodesic distances g_i in V and Euclidean distances d_i in U are used to compute ratios $r_i = \frac{g_i}{d_i}$. The variance $\mathrm{Var}(r_i)$ measures warping, with zero variance indicating uniform scaling. Additionally, to further evaluate the practicality of each method, we compare training time, rendering FPS, and editing time. As shown in Tab. 2, our method not only minimizes decal warping but also significantly surpasses previous methods in terms of

Method	NeRF Syn.	Shiny Blen.	DTU
PhySG	20.60	26.21	24.69
NeRFactor	27.86	27.04	25.75
NvDiffRec	29.05	28.11	25.54
DE-NeRF	**29.18**	28.79	25.93
Ours	28.96	**28.81**	**26.12**

Table 1: PSNR of novel view synthesis with disentangled reconstruction methods.

Method	Training↓	FPS↑	Editing↓	Warping↓
NeuMesh	16 hours	0.017	1 hour	0.472
Seal-3D	**0.1 hours**	28.4	1 min	0.565
DE-NeRF	18 hours	0.013	1 hour	0.489
Ours	1 hour	**81.5**	**0**	**0.001**

Table 2: We compare training time, inference frame rate (FPS), editing time, and the RVW metric for warping. While our training time is longer than Seal-3D, our method significantly outperforms other methods in the other three metrics.

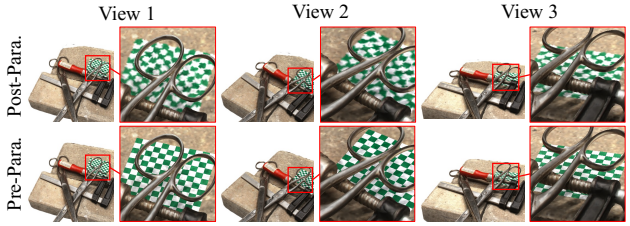

Figure 7: Our proposed pre-parameterizaton shows high-frequency details of the decal (i.e., the cell edges are clear), while the other one exhibits significant blurriness.

practicality. Furthermore, we conduct a user study with 18 participants to evaluate the integration of decal texture with lighting and shadows. We ask the participants to give a preference score (ranging from 0 to 1.0), with 1.0 indicating that the integration appears completely natural and 0 indicating no environmental lighting effect. Our score is **0.86**, while the second-place method, DE-NeRF, scores 0.55.

5.4 Ablation Study

In this section, we evaluate several technical components in our pipeline through a series of ablation studies.

Pre-Parameterization. As shown in Fig. 7, the baseline model with post-parameterization (as discussed in Sec. 4.1) exhibits significant blurriness on the decal. In contrast, our method shows high-frequency details of the decal (i.e., the cell edges are clear), as previously analyzed in Fig. 3.

Neural Albedo Texture Map. We use the optimization of the pixels' RGB of texture map as a baseline (pixel-based), comparing it to our approach, which encodes textures using an MLP. As shown in Fig. 8, the baseline approach, which lacks sufficient UV sampling during training, leads to noise artifacts in novel views. By contrast, encoding the texture

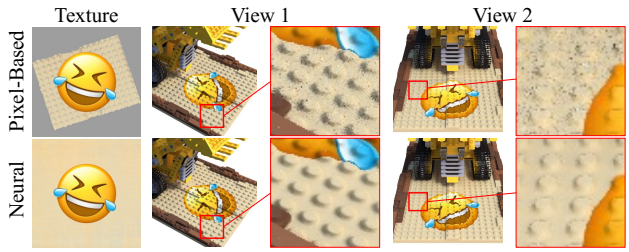

Figure 8: Pixel-based texture maps result in insufficient sampling, leading to noisy renderings in novel views.

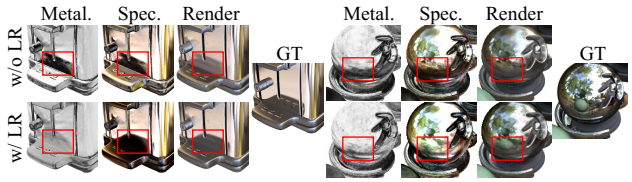

Figure 9: LR net can model local reflections within objects, improving the quality of the reconstruction.

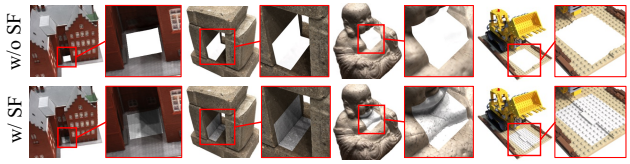

Figure 10: Our proposed shadow factor (SF) accurately captures shadow brightness, enhancing the white decal's three-dimensional appearance and sense of depth.

map using an MLP allows the model to generalize across the entire UV space from supervision on a subset of pixels.

LR Net. We remove LR net from our pipeline to create a baseline. As shown in Fig. 9, our method reconstructs the local reflections, whereas the baseline method fails.

Shadow Factor. We remove shadow factor from Eq. 12 to build a baseline and use a plain white image as the decal to highlight the shadow effect. As shown in Fig. 10, our proposed method effectively decouples shadows and projects them onto the decal, achieving stereoscopic impression.

6 Conclusion

In this paper, we present InstantSticker, a disentangled reconstruction pipeline based on IBL, which focuses on highly realistic decal blending, simulates stickers attached to the reconstructed surface, and allows for instant editing and real-time rendering. Extensive experimental results demonstrate that our method surpasses previous decal blending methods in terms of editing speed, rendering speed, and editing quality, achieving the state-of-the-art.

Limitation. Our method is dependent on high-quality mesh input, a limitation we plan to address in future work by exploring approaches that relax this requirement.

Acknowledgments

This work was supported by National Science Foundation of China (U20B2072, 61976137). This work was also partially supported by Grant YG2021ZD18 from Shanghai Jiao Tong University Medical Engineering Cross Research. This work was partially supported by STCSM 22DZ2229005.

References

Burley, B.; and Studios, W. D. A. 2012. Physically-based shading at disney. In *Acm Siggraph*, volume 2012, 1–7. vol. 2012.

Cook, R. L.; and Torrance, K. E. 1982. A reflectance model for computer graphics. *ACM Transactions on Graphics (ToG)*, 1(1): 7–24.

Debevec, P. 1998. Rendering synthetic objects into real scenes: bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, 189–198.

Do Carmo, M. P. 2016. *Differential geometry of curves and surfaces: revised and updated second edition*. Courier Dover Publications.

Fu, Q.; Xu, Q.; Ong, Y. S.; and Tao, W. 2022. Geo-neus: Geometry-consistent neural implicit surfaces learning for multi-view reconstruction. *Advances in Neural Information Processing Systems*, 35: 3403–3416.

Ge, W.; Hu, T.; Zhao, H.; Liu, S.; and Chen, Y.-C. 2023. Ref-neus: Ambiguity-reduced neural implicit surface learning for multi-view reconstruction with reflection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 4251–4260.

Huang, X.; Zhang, Y.; Ni, B.; Li, T.; Chen, K.; and Zhang, W. 2023. Boosting point clouds rendering via radiance mapping. In *Proceedings of the AAAI conference on artificial intelligence*, 953–961.

Jacobson, A.; and Panozzo, D. 2017. libigl: Prototyping Geometry Processing Research in C++. In *SIGGRAPH Asia 2017 Courses, SA 2017*. Association for Computing Machinery.

Jensen, R.; Dahl, A.; Vogiatzis, G.; Tola, E.; and Aanæs, H. 2014. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 406–413.

Karis, B.; and Games, E. 2013. Real shading in unreal engine 4. *Proc. Physically Based Shading Theory Practice*, 4(3): 1.

Kerbl, B.; Kopanas, G.; Leimkühler, T.; and Drettakis, G. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Trans. Graph.*, 42(4): 139–1.

Kessenich, J.; Sellers, G.; and Shreiner, D. 2016. *OpenGL Programming Guide: The official guide to learning OpenGL, version 4.5 with SPIR-V*. Addison-Wesley Professional.

Lévy, B.; Petitjean, S.; Ray, N.; and Maillot, J. 2002. Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics (TOG)*, 21(3): 362–371.

Liu, L.; Zhang, L.; Xu, Y.; Gotsman, C.; and Gortler, S. J. 2008. A local/global approach to mesh parameterization. In *Proceedings of the Symposium on Geometry Processing*, 1495–1504.

Liu, Y.; Wang, P.; Lin, C.; Long, X.; Wang, J.; Liu, L.; Komura, T.; and Wang, W. 2023. Nero: Neural geometry and brdf reconstruction of reflective objects from multiview images. *ACM Transactions on Graphics (TOG)*, 42(4): 1–22.

Lorensen, W. E.; and Cline, H. E. 1987. Marching cubes: A high resolution 3D surface construction algorithm. *ACM SIGGRAPH Computer Graphics*, 21(4): 163–169.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *European Conference on Computer Vision*, 405–421. Springer.

Munkberg, J.; Hasselgren, J.; Shen, T.; Gao, J.; Chen, W.; Evans, A.; Müller, T.; and Fidler, S. 2022. Extracting triangular 3d models, materials, and lighting from images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8280–8290.

Pharr, M.; Jakob, W.; and Humphreys, G. 2016. Physically Based Rendering: From Theory to Implementation.

Rahaman, N.; Baratin, A.; Arpit, D.; Draxler, F.; Lin, M.; Hamprecht, F.; Bengio, Y.; and Courville, A. 2019. On the spectral bias of neural networks. In *International conference on machine learning*, 5301–5310. PMLR.

Ravi, N.; Reizenstein, J.; Novotny, D.; Gordon, T.; Lo, W.-Y.; Johnson, J.; and Gkioxari, G. 2020. Accelerating 3d deep learning with pytorch3d. *arXiv preprint arXiv:2007.08501*.

Shi, Y.; Wu, Y.; Wu, C.; Liu, X.; Zhao, C.; Feng, H.; Liu, J.; Zhang, L.; Zhang, J.; Zhou, B.; et al. 2023. Gir: 3d gaussian inverse rendering for relightable scene factorization. *arXiv preprint arXiv:2312.05133*.

Srinivasan, P. P.; Garbin, S. J.; Verbin, D.; Barron, J. T.; and Mildenhall, B. 2023. Nuvo: Neural UV Mapping for Unruly 3D Representations. arXiv:2312.05283.

Sun, J.-M.; Wu, T.; Yang, Y.-L.; Lai, Y.-K.; and Gao, L. 2023. SOL-NeRF: Sunlight modeling for outdoor scene decomposition and relighting. In *SIGGRAPH Asia 2023 Conference Papers*, 1–11.

Tang, J.; Chen, X.; Wang, J.; and Zeng, G. 2022. Compressible-composable nerf via rank-residual decomposition. *Advances in Neural Information Processing Systems*, 35: 14798–14809.

Verbin, D.; Hedman, P.; Mildenhall, B.; Zickler, T.; Barron, J. T.; and Srinivasan, P. P. 2022. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5481–5490. IEEE.

Walter, B.; Marschner, S. R.; Li, H.; and Torrance, K. E. 2007. Microfacet Models for Refraction through Rough Surfaces. *Rendering techniques*, 2007: 18th.

Wang, X.; Zhu, J.; Ye, Q.; Huo, Y.; Ran, Y.; Zhong, Z.; and Chen, J. 2023. Seal-3d: Interactive pixel-level editing for neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 17683–17693.

Wu, T.; Sun, J.-M.; Lai, Y.-K.; and Gao, L. 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.

Wu, T.; Wang, J.; Pan, X.; Xudong, X.; Theobalt, C.; Liu, Z.; and Lin, D. 2022. Voxurf: Voxel-based Efficient and Accurate Neural Surface Reconstruction. In *The Eleventh International Conference on Learning Representations*.

Xiang, F.; Xu, Z.; Hasan, M.; Hold-Geoffroy, Y.; Sunkavalli, K.; and Su, H. 2021. Neutex: Neural texture mapping for volumetric neural rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7119–7128.

Yang, B.; Bao, C.; Zeng, J.; Bao, H.; Zhang, Y.; Cui, Z.; and Zhang, G. 2022. Neumesh: Learning disentangled neural mesh-based implicit field for geometry and texture editing. In *European Conference on Computer Vision*, 597–614. Springer.

Young, J. 2016. xatlas. https://github.com/jpcy/xatlas.

Zhang, K.; Luan, F.; Wang, Q.; Bala, K.; and Snavely, N. 2021. Physg: Inverse rendering with spherical gaussians for physics-based material editing and relighting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5453–5462.

Zhukov, S.; Iones, A.; and Kronin, G. 1998. An ambient light illumination model. In *Rendering Techniques' 98: Proceedings of the Eurographics Workshop in Vienna, Austria, June 29—July 1, 1998 9*, 45–55. Springer.