Structurally Disentangled Feature Fields Distillation for 3D Understanding and Editing

ANONYMOUS AUTHOR(S)

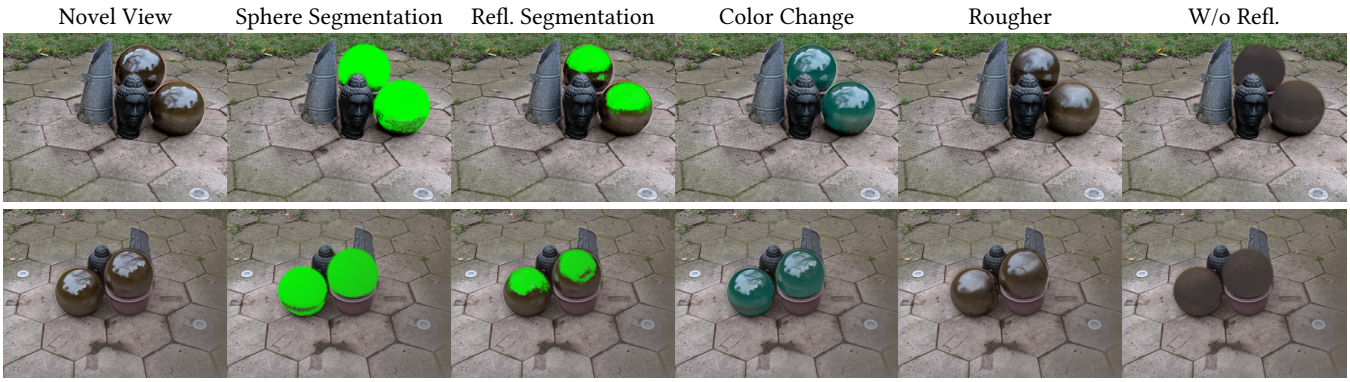

Fig. 1. An illustration of our method for the Garden-spheres scene from the real-world RefNeRF dataset [Verbin et al. 2022]. By distilling 2D DINOv2 [Oquab et al. 2023] features into our disentangled feature field representation, our method enables: (i). Generation of novel views, (ii). 3D segmentation of the spheres, (iii). 3D segmentation of the reflective region of the spheres, (iv). Color editing while adhering to reflections, (v). Changing the roughness of the spheres, (vi). Removing the reflection from the spheres (using both the reflection and camera-based components). Refl. = Reflection.

Recent work has demonstrated the ability to leverage or distill pre-trained 2D features obtained using large pre-trained 2D models into 3D features, enabling impressive 3D editing and understanding capabilities using only 2D supervision. Although impressive, the input 2D features may be 3D and physically inconsistent. Current models average such inconsistencies, resulting in inferior distilled 3D features. In this work, we propose instead to capture 3D features using multiple disentangled feature fields that capture different structural components of 3D features involving view-dependent and view-independent components. Subsequently, each element can be controlled in isolation, enabling semantic and structural understanding and editing capabilities. For instance, 3D segmentation can be achieved by considering only view-indepenent features, and discarding the view-dependent ones, resulting in significant performance compared to baselines, which average view-dependent features. Using a user click, one can segment 3D features corresponding to a given object and then segment, edit, or remove their view-dependent (reflective) properties, enabling a set of novel understanding and editing tasks.

CCS Concepts: • **Computing methodologies** → **Reflectance modeling**; **Perception**; **Image-based rendering**.

Additional Key Words and Phrases: 3D Disentanglement, Feature Distillation, Neural Radiance Fields, 3D Understanding, 3D Editing

1 INTRODUCTION

A recent paradigm in computer graphics and computer vision is 2D feature distillation, whereby 2D image features obtained using large-scale self-supervised methods are "distilled" to the parameters of an underlying 3D model. Doing so enables 3D semantic understanding and editing, given 2D RGB and feature supervision only. Several works [Kerr et al. 2023; Kobayashi et al. 2022; Ye et al. 2023] considered the setting of novel view synthesis with an underlying NeRF [Mildenhall et al. 2021] or a Gaussian Splatting [Kerbl et al. 2023] model, achieving impressive 3D understanding and editing capabilities. Each 3D point (in the former) or Gaussian (in the latter) stores a feature value, which can be rendered to different views using volumetric rendering. The rendered views correspond to 2D feature maps obtained using self-supervised 2D methods such as DINOv2 [Oquab et al. 2023]. This paradigm was later extended to text [Kerr et al. 2023; Qin et al. 2023], enabling impressive 3D text-based understanding and editing capabilities. Although impressive, current models assume that 3D features are captured using a single view-independent feature field (or a single 3D feature value per point or Gaussian). However, as demonstrated in Sec. 4.1, input 2D features from SSL models contain 3D inconsistencies and are inherently view dependent (i.e., contain significant view-dependent variations). Current models average such view-dependent variations, resulting in inferior features for downstream performance on tasks such as 3D segmentation.

In this work, we propose instead to capture 3D features using *multiple disentangled feature fields* that capture different structural components of 3D features. While these structural components could involve a variety of physical properties such as lighting and deformations, we focus on the disentanglement of view-dependent and view-independent features. Our disentangled feature fields can be learned using 2D supervision only, in an unsupervised manner, thus enabling the disentanglement of 2D (and 3D) features into components that are view-independent and view-dependent (reflections and camera-dependent features). Subsequently, each component can be controlled separately, enabling semantic and structural understanding and editing capabilities. For instance, 3D segmentation can be achieved by considering only view-indepenent features, and discarding the view-dependent ones, resulting in significant performance compared to baselines, which average view-dependent features. Using a user click, one can segment an entire object in 3D or only its reflective component, edit view-independent object

properties, such as its color, while adhering to reflections, or remove the object's reflections. An illustration is provided in Fig. 1.

To achieve our desired disentanglement, we propose computing the feature value of a 3D point along a viewing direction as the combination of the outputs of two disentangled feature fields: (i). A reflected view feature field capturing view-dependent features that arise from specular object reflections, and (ii). A view-independent feature field capturing diffuse features of the object which depend only on the location of a 3D point and not the viewing direction.

We evaluate our approach on a variety of objects from a diverse set of scenes from the Shiny Blender dataset [Verbin et al. 2022] as well as real-world scenes from the RefNeRF real-world dataset [Verbin et al. 2022] and Mip-NeRF-360 dataset [Barron et al. 2022]. In terms of 3D understanding, we consider the tasks of: (i). 3D segmentation, for which our structurally disentangled representation achieves superior results compared to a single holistic feature field [Kobayashi et al. 2022], and (ii). Segmentation of view-dependent components in a scene. For example, one can segment only the reflective component of an object selected using a user click. In terms of 3D editing, we demonstrate the applicability of our approach on (i). The ability to remove the reflective component of an object, and lastly (ii). The ability to edit individual 3D component. For example, changing the object's color while correctly adhering to reflections or manipulating its roughness.

In summary, we offer the following contributions:

(1) To our knowledge, we are the first to consider a physically-inspired decomposition of a *semantic* feature field. This is in contrast to previous works, such as RefNeRF [Verbin et al. 2022] and UniSDF [Wang et al. 2023a], which only consider appearance, and to DFF [Kobayashi et al. 2022], which uses a single view-independent feature field. Our decomposition reveals a new scientific insight: the view-dependent features of a 2D foundation model *can* be decomposed into a 3D feature field of view-independent and view-dependent components.
(2) By doing so, we can: (i) Introduce novel applications, including view-dependent and view-independent segmentation and editing capabilities. (ii) Improve 3D segmentation by only using the view-independent component and discarding the view-dependent component, demonstrating state-of-the-art performance.
(3) We improve the underlying reference features of the foundation model by *decomposing* the view-dependent component. In contrast, baseline models such as DFF *averages* all the view-dependent feature components by modeling only a single view-independent feature field.

2 RELATED WORK

2.1 Structured Novel View Synthesis

Neural Radiance Field (NeRF) [Mildenhall et al. 2021] reconstructs a 3D scene from 2D images by mapping 3D spatial locations and viewing directions to their corresponding color and density values. Different works introduced physical modeling within a NeRF-like formulation to allow for the modeling of geometry, lighting, materials, reflections, and more. For instance, to model geometry, many works integrate a signed distance function (SDF) formulation [Li et al. 2023; Long et al. 2022; Oechsle et al. 2021; Rosu and Behnke 2023; Wang et al. 2021; Yariv et al. 2021; Yu et al. 2022]. Other works extended NeRF to enable relighting by disentangling appearance into scene lighting and materials [Bi et al. 2020; Boss et al. 2021a,b; Srinivasan et al. 2021; Zhang et al. 2021a,b].

In the context of reflections modeling, approaches such as [Ramamoorthi et al. 2009] represented appearance using disentangled view-dependent specular and reflective appearance. Ref-NeRF [Verbin et al. 2022] modeled appearance through two disentangled radiance fields within an implicit radiance field formulation, one modeling view-independent diffuse properties and another modeling reflective properties. BakedSDF [Yariv et al. 2023] and ENVIDR [Liang et al. 2023] extended this representation for the reconstruction of glossy surfaces and with material decomposition. UniSDF [Wang et al. 2023a] improved upon these parameterizations by including two separate radiance fields for modeling reflections, one for the reflected view and another for camera view based radiance. Unlike the above, our focus is on structural disentanglement for feature fields, modeling 3D features together with appearance. As far as we can ascertain, our work is the first to enable such disentanglement.

2.2 2D Feature Distillation

Due to the scarcity of 3D data, recent work attempted to distill or lift 2D features trained using pre-trained self-supervised 2D models (such as DINO [Caron et al. 2021], DINOv2 [Oquab et al. 2023] or CLIP-LSeg [Li et al. 2022]). In particular, [Kobayashi et al. 2022; Tschernezki et al. 2022] extended NeRF to model not only appearance, but also semantics, through the use of an additional view-independent feature field, that maps a 3D point into a 3D feature. Other works embedded semantic information into NeRFs [Siddiqui et al. 2023; Yao et al. 2024]. Further works focused on integrating open-vocabulary text-based features, by embedding CLIP features into NeRF [Kerr et al. 2023] or Gaussian splatting [Qin et al. 2023]. Although impressive, models assume that 3D features are captured using a single view-independent feature field (or a single 3D feature per point or Gaussian) Our work instead focuses on the 3D model, capturing 3D features using multiple structurally disentangled feature fields, thus enabling multiple novel understanding and editing capabilities.

3 METHOD

3.1 Preliminaries

NeRF [Mildenhall et al. 2021] represents a 3D scene using a multilayer perceptron network (MLP) that parameterizes a continuous 5D radiance field. This field $f : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma)$, maps a 3D coordinate $\mathbf{x} \in \mathbb{R}^3$ and a viewing direction $\mathbf{d} \in \mathbb{R}^2$ to an emitted color $\mathbf{c} \in \mathbb{R}^3$ and volume density $\sigma \in \mathbb{R}$. To render 2D views, NeRF employs volumetric rendering. In particular, rays are cast through the scene, with $f(\mathbf{x}, \mathbf{d})$ queried along each ray. The volume rendering equation is then used to composite color and opacity:

$$C(r) = \int T(t)\mathbf{c}(r(t))\sigma(r(t))dt, \tag{1}$$

where $r(t)$ parameterizes the ray, $T(t)$ is the accumulated transmittance, and $\mathbf{c}(r(t))$ and $\sigma(r(t))$ are the color and density at point t. NeRF is trained by minimizing a reconstruction loss between images rendered from the field and a set of ground-truth images with

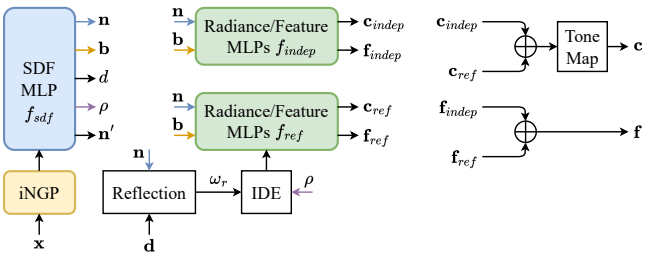

Fig. 2. **The method's pipeline.** We decompose the appearance color of the scene $\mathbf{c}$ into physical components $\mathbf{c}_{indep}$ and $\mathbf{c}_{ref}$ and sum them to compute the color of the scene at location $\mathbf{x}$ and viewing direction $\mathbf{d}$. We also learn a decomposed feature field for the scene, $\mathbf{f}_{indep}$ and $\mathbf{f}_{ref}$, which enables physically-oriented semantic understanding and editing applications. Please see section 3 for more details.

known camera poses. This differentiable process allows NeRF to implicitly learn scene geometry and appearance for photorealistic rendering from unseen viewpoints.

When considering feature distillation [Kerr et al. 2023; Kobayashi et al. 2022; Ye et al. 2023], one additionally learns an MLP that parameterizes a continuous 3D feature field f_{feat}. Unlike f, f_{feat} : $\mathbf{x} \rightarrow \mathbf{f}$ maps a 3D coordinate $\mathbf{x} \in \mathbb{R}^3$ to a feature value $\mathbf{f} \in \mathbb{R}^n$. 2D feature maps can then be rendered similarly to color values, using σ predicted by f. That is:

$$F(r) = \int T(t)\mathbf{f}(r(t))\sigma(r(t))dt, \tag{2}$$

where $r(t)$, $T(t)$ and $\sigma(r(t))$ are as above and $\mathbf{f}(r(t))$ represents the predicted feature value at point t (identical for every view direction). In addition to image-based reconstruction loss, one also minimizes a reconstruction loss between rendered features and ground-truth features, obtained by applying a pre-trained 2D feature extractor (such as DINOv2 [Oquab et al. 2023]) to ground-truth RGB views.

3.2 Structurally Disentangled Feature Fields

Fig. 2 illustrates our method, which we elaborate on below.

Signed distance representation. Similar to previous work [Wang et al. 2023a], our model's backbone is an MLP computing the signed distance function $d(\mathbf{x})$ (SDF) at each location $\mathbf{x}$. The scene's surface is defined at the zero level set of the function. $\mathbf{x}$ undergoes a contraction transformation [Barron et al. 2022; Wang et al. 2023a], limiting its values to $[0, 2)$.

The density $\sigma(\mathbf{x})$ for volume rendering is computed by $\sigma(\mathbf{x}) = \alpha\Psi_\beta(d(\mathbf{x}))$, where Ψ_β is the cumulative distribution of a Laplace distribution with zero mean and a scale parameter β that is learned during optimization. The benefit of the SDF representation is that the normal of the scene surface can be easily derived as the gradient of the distance function:

$$\mathbf{n}(\mathbf{x}) = \nabla d(\mathbf{x})/||\nabla d(\mathbf{x})||_2. \tag{3}$$

The normal is used as input to the components in our model for computing the disentangled feature representation.

To better reconstruct high-frequency details and to speed up training, we use the iNGP hash encoding [Müller et al. 2022]. Each location $\mathbf{x}$ is mapped to a high-dimensional feature vector, which is the concatenation of the iNGP's pyramid-level features. This feature vector is the input to our SDF MLP. To ease notations, we omit in this section the notation of $\mathbf{x}$ for location-dependent quantities.

Radiance and features components. We decompose the appearance of the scene into two elements: a view-independent color $\mathbf{c}_{indep}$ and a view-dependent reflected color $\mathbf{c}_{ref}$. The view-independent color is calculated as:

$$\mathbf{c}_{indep} = f_{indep}(\mathbf{n}, \mathbf{b}), \tag{4}$$

where f_{indep} is a learned MLP, $\mathbf{n}$ is the normal defined in Eq. 3, and $\mathbf{b}$ is a bottleneck vector output from the SDF MLP. This vector enables additional degrees of freedom to accommodate second-order effects, such as varying illumination over the scene [Verbin et al. 2022; Wang et al. 2023a]. $\mathbf{c}_{indep}$ represents the view-independent color and accordingly is independent of the viewing direction $\mathbf{d}$.

The other color component, $\mathbf{c}_{ref}$, is view dependent. It is responsible for reflectivity scene regions and models the reflected radiance in the scene. Following previous work [Verbin et al. 2022], we calculate the reflection for the viewing direction about the normal:

$$\omega_r = 2(\omega_o \cdot \mathbf{n})\mathbf{n} - \omega_o, \tag{5}$$

where $\omega_o = -\hat{\mathbf{d}}$, is a unit vector pointing to the camera from a point in space, and $\hat{\mathbf{d}}$ is a viewing direction. Then, we use Integrated Directional Encoding (IDE) [Verbin et al. 2022] in the computation of $\mathbf{c}_{ref}$:

$$\mathbf{c}_{ref} = f_{ref}(\mathbf{n}, \mathbf{b}, \text{IDE}(\omega_r, \kappa)), \tag{6}$$

where $\kappa = 1/\rho$, and ρ is the surface roughness predicted by the SDF MLP. Finally, the rendered color is given by:

$$\mathbf{c} = \text{tonemap}(\mathbf{c}_{indep} + \mathbf{c}_{ref}), \tag{7}$$

where tonemap$(\cdot)$ is a tone mapping function converting linear color to the sRGB format and clipping the output to the range $[0, 1]$. Similarly, we decompose the feature field of the scene $\mathbf{f}$ as follows:

$$\mathbf{f} = \mathbf{f}_{indep} + \mathbf{f}_{ref}, \tag{8}$$

where $\mathbf{f}_{indep}$ and $\mathbf{f}_{ref}$ are computed in the same manner as $\mathbf{c}_{indep}$, Eq. 4 and $\mathbf{c}_{ref}$, Eq. eqs. (5) and (6). The feature components are used as a *semantic* representation for their color counterparts. We note that the semantic feature from the independent feature field only, $\mathbf{f}_{indep}$ can be used for understanding tasks such as 3D segmentation, as 3D segmentation is inherently view-independent. This is an example where a physical disentanglement (i.e., to view-dependent and independent components) can result in "improving" or "cleaning" undesirable feature components required for 3D segmentation. By jointly modeling features and colors, one can also enable a diversity of editing applications as illustrated in Sec. 4.

3.3 Training Objective

The training of our structural decomposition of the scene is supervised by posed images and their corresponding DINOv2 [Oquab et al. 2023] feature maps. Given a set of posed images $\{C_1^{gt}, \ldots, C_m^{gt}\}$, where $C_i^{gt} \in \mathbb{R}^{H\times W\times 3}$, we obtain associated 2D feature maps $\{F_1^{gt}, \ldots, F_m^{gt}\}$ using the pretrained feature extractor DINOv2, where $F_i^{gt} \in \mathbb{R}^{H\times W\times n}$, and n is the feature dimension. The rendered color

images are compared to the given images with an l_2 loss:

$$\mathcal{L}_c = \frac{1}{m} \sum_i ||C_i - C_i^{gt}||_2^2, \tag{9}$$

where C_i is the scene appearance rendered from our model for the view direction of image C_i^{gt}. Additionally, we regularize the SDF MLP learning with the eikonal loss [Gropp et al. 2020; Wang et al. 2023a] to promote the approximation of a valid SDF:

$$\mathcal{L}_e = \frac{1}{|\mathbf{x}|} \sum_{\mathbf{x}} (||\nabla \mathbf{d}(\mathbf{x})||_2 - 1)^2. \tag{10}$$

We also encourage normal smoothness by comparing the computed normal $\mathbf{n}(\mathbf{x})$ from the SDF to the normal $\mathbf{n}'(\mathbf{x})$ predicted by the SDF MLP:

$$\mathcal{L}_p = \sum_{\mathbf{x}} w_{\mathbf{x}} \, ||\mathbf{n}(\mathbf{x}) - \mathbf{n}'(\mathbf{x})||^2. \tag{11}$$

Further, we penalize back-facing normals using the orientation loss [Verbin et al. 2022]:

$$\mathcal{L}_o = \sum_{\mathbf{x}} w_{\mathbf{x}} \max(0, \mathbf{n}(\mathbf{x}) \cdot \mathbf{d}(\mathbf{x}))^2. \tag{12}$$

For feature learning, we have found that optimizing the radiance and feature fields concurrently compromises the learning process of both. Thus, we first learn the decomposed appearance of the scene using the total loss:

$$\mathcal{L} = \mathcal{L}_c + \lambda_e \mathcal{L}_e + \lambda_p \mathcal{L}_p + \lambda_o \mathcal{L}_o. \tag{13}$$

Then, we freeze the appearance model and train MLPs for optimizing our feature field decomposition using:

$$\mathcal{L}_f = \frac{1}{m} \sum_i ||F_i - F_i^{gt}||_2^2. \tag{14}$$

3.4 Segmentation and Editing

Our structurally decomposed feature field enables physically oriented segmentation and editing of the scene. We segment the scene as follows. First, we select a rendered feature component $F_{comp}(x, y)$, where (x, y) is the pixel location, and F_{comp} can be F_{indep} or F_{ref}. Then, we correlate the selected feature with the corresponding decomposed feature field in 3D. The location of features with a correlation factor above a threshold belongs to the segmented region of interest in the scene. Once we obtain the region of interest, we can control and modify the physical properties of the scene *locally*, such as the view-independent color, roughness, level of reflection, and more, as demonstrated in Sec. 4

3.5 How is Disentanglement Achieved?

Two MLPs, $\mathbf{f}_{indep}$ and $\mathbf{f}_{dep}$, predict the view-independent and view-dependent 3D feature components (see Fig. 2). To achieve the disentanglement, we enforce three constraints: (i). $\mathbf{f}_{indep}$ cannot model the view-dependent feature component. This is achieved by design, by **not** providing $\mathbf{f}_{indep}$ the view direction as input (unlike $\mathbf{f}_{dep}$). (ii). $\mathbf{f}_{indep}$ and $\mathbf{f}_{dep}$ together model the total feature value. This is enforced by summing the outputs of $\mathbf{f}_{indep}$ and $\mathbf{f}_{dep}$ together (Eq. 8) and using volumetric rendering to ensure 2D rendered features match ground truth rendered features. (iii). $\mathbf{f}_{indep}$ models the view-independent feature component. While this is not explicitly enforced, we hypothesize that it is implicitly enforced by the model as it tries to be efficient (compressed). As $\mathbf{f}_{indep}$ does not utilize the view direction as input, for a given 3D position, it uses a single feature value for all view directions. $\mathbf{f}_{dep}$ can then model the residual view-dependent component only when this is **required**. For directions without reflections, for instance, this is not required, resulting in $\mathbf{f}_{dep}$ storing fewer feature values in total.

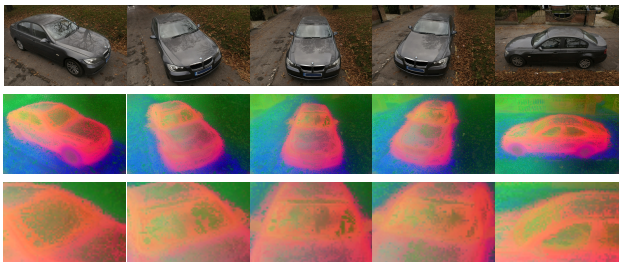

Fig. 3. PCA of DINOv2 features for ground-truth input views of the Sedan scene from real-world dataset of [Verbin et al. 2022]. We zoom in on the windshield, illustrating differences in corresponding locations between views.

4 EXPERIMENTS

We evaluate our representation on multiple vectors. First, we demonstrate that input features from DINOv2 contain both view-dependent and view-independent information. Second, we consider the ability to segment objects in 3D space. Unlike previous methods, our disentangled feature fields capture both diffuse and reflective properties and allow for a better reconstruction of the semantic components in the scene. We also enable the segmentation of the reflective component of objects in isolation from different novel views. Third, we consider the ability to remove view-dependent (reflective) components in the scene for specific semantic objects segmented using our approach. Fourth, we consider the ability to edit the scene, where one can manipulate or change only specific objects and their dependent (reflective) properties in isolation.

Datasets. We evaluate our method on synthetic scenes from the Shiny Blender [Verbin et al. 2022] dataset as well as real-world scenes from the RefNeRF real-world [Verbin et al. 2022] dataset and Mip-NeRF-360 dataset [Barron et al. 2022]. We consider a diverse set of scenes and objects from both real world and synthetic scenes, featuring multiple objects and variable lighting conditions, highlighting the generality of our approach. In particular, we consider 11 scenes and 25 objects from 3 real-world and synthetic datasets. This is comparable to recent work such as DFF or RefNeRF. We consider the standard train-view/novel-view splits provided by the respective datasets and evaluate our model on such novel views. Additionally, our method supports incorporating arbitrary 2D semantic features extracted from models like CLIP-Lseg and DINOv2. The webpage provides associated videos depicting novel views and corresponding segmentation, removal, and editing results. We also provide implementation details in the appendix.

4.1 Feature Analysis

We first consider whether the distilled DINOv2 features capture both view-dependent and view-independent components. To this

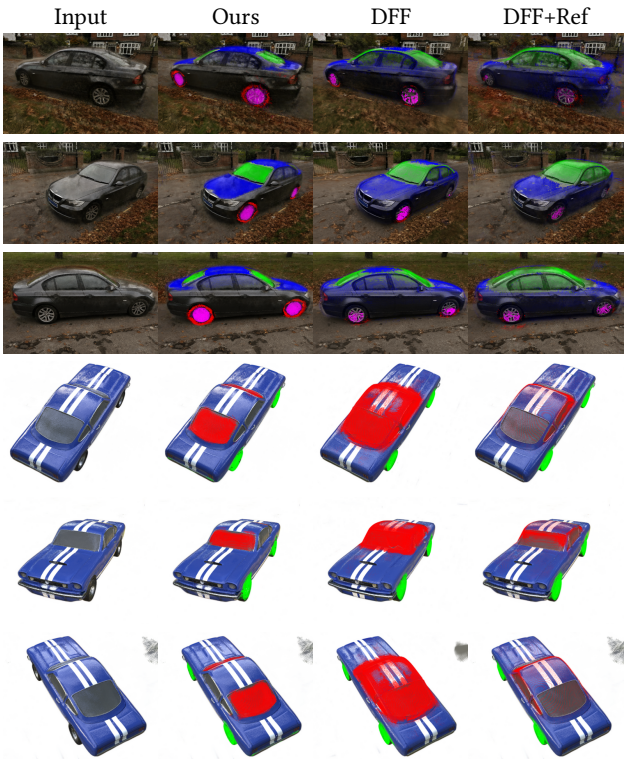

Fig. 4. 3D objects segmentation from three novel views, for the Sedan scene from real-world RefNeRF [Verbin et al. 2022] dataset for the objects of Bonet-top, Windshield, Hubcups and Wheels, and for the Car scene from synthetic Shiny Blender [Verbin et al. 2022] dataset for the objects of Windshield and Wheels. We compare our result to DFF [Kobayashi et al. 2022] and to a baseline where DFF is optimized for features while RefNeRF is optimized for appearance (see section 4.2).

Fig. 5. Segmentation of the spheres for novel views of the Garden-spheres real-world scene using either the full segmentation of the sphere (second row) or only the reflective component of the spheres (third row).

end, we visualize the PCA of the features for five ground-truth views of the Sedan scene from the real-world RefNeRF dataset. As seen in Fig. 3, while the features appear similar, there are notable differences, particularly in reflective regions such as the windshield (zoomed in). As further evidence, we note the recent work of [El Banani et al. 2024], which demonstrates that features obtained from large foundation models, DINOv2 in particular, are not 3D view consistent. As

Scene (Objects)	Ours	Ours implicit	Ours total	Ours optt	DFF	DFF+ Ref
Bicycle (Bench, Wheels)	**0.583**	0.381	0.407	0.444	0.518	0.504
Counter (Mitten, Plant, Tray)	**0.824**	0.705	0.641	0.664	0.713	0.629
Garden (Ball, Plant, Tabletop)	**0.840**	0.813	0.786	0.389	0.824	0.700
Kitchen (Lego)	**0.871**	0.761	0.631	0.778	0.856	0.850
Gardenspheres (Cone, Head, Spheres)	**0.825**	0.663	0.647	0.619	0.776	0.770
Sedan (Bonnet-top, Windshield, Hubcaps, Wheel)	**0.643**	0.337	0.377	0.485	0.625	0.512
Toycar (Body, Wheel)	**0.709**	0.499	0.176	0.283	0.689	0.654
Teapot (Cover)	**0.762**	0.125	0.654	0.768	0.529	0.399
Mean (Real World Scenes)	**0.757**	0.628	0.594	0.523	0.691	0.582
Car (Windshield, Wheels)	**0.799**	0.437	0.402	0.088	0.523	0.445
Toaster (Body, Toasts)	0.906	0.844	0.833	0.839	**0.940**	0.787
Helmet (Body, Windshield)	**0.863**	0.795	0.705	0.894	0.731	0.472
Mean (Shiny Blender)	**0.856**	0.568	0.550	0.648	0.731	0.527

Table 1. Mean IoU for segmentation of objects from the Shiny Blender synthetic dataset [Verbin et al. 2022] (bottom) and real-world scenes (top). First four scenes are taken from the real world RefNeRF [Verbin et al. 2022] dataset while the other four real world scenes are from the Mip-NeRF-360 dataset. On the RHS, we compare our approach DFF [Kobayashi et al. 2022] and a baseline where DFF is optimized for features while RefNeRF is optimized for appearance (see section 4.2). On the LHS, we also consider variants of our approach: (1). Ours-implicit: Our approach, but with implicit, instead of explicit, representation of physical properties [s.a. roughness], (2). Ours-total: Our approach, but using the total features [dependent and independent] for segmentation, (3). Our-optt: Our approach, where we optimized the color and features together. For each scene, we show, in brackets, the segmented objects.

such, applying our model has the advantage of disentangling view-dependent and view-independent components of a given feature view and can enhance downstream applications that require view-independent feature representations. This is illustrated in Sec. 4.2, where we show that our view-independent feature field yields better 3D segmentation results than using the full (both dependent and independent) feature field or baselines. Beyond 3D segmentation, one can render ground truth views only using the view-independent component, discarding their view-dependent component.

4.2 Semantic Segmentation

We consider our method's capability in segmenting both independent and view-dependent components.

4.2.1 Full Object Segmentation. In Fig.4, we visualize our segmentation of three novel views for a real-world scene and a synthetic scene. The segmentation is obtained by clicking on each object and using our disentangled view-independent feature component, described in Sec. 3.4. We compare our method to Distilled Feature Field (DFF) [Kobayashi et al. 2022], the closest method to ours, which uses a single view-independent feature field. As RefNeRF is an improved appearance model, we also consider another baseline whereby appearance is obtained as in RefNeRF, using both view-dependent and view-independent feature fields, while semantics is obtained as in DFF, using a view-independent feature field. As can be seen,

our method results in a superior segmentation and can successfully segment both reflective and non-reflective regions well, while the baseline is worse, particularly in reflective regions. Additional results are provided in the webpage.

For numerical evaluation, we compare the segmentation of objects for both synthetic scenes and real-world scenes. We manually obtain ground-truth segmentations by first applying the Segment Anything Model [Kirillov et al. 2023] and subsequently manually refining masks (see examples in the webpage). As can be seen in Tab. 1, our method results in better mean IoU scores.

4.2.2 Explicit modeling. Our approach leverages explicit components to model the view-dependent and view-independent feature fields. Specifically, we model roughness and reflections explicitly, as detailed in Eq. 5 and Eq. 6 in Sec. 3. This enables novel applications that incorporate physical properties, such as object roughness editing. However, it is unclear whether explicit molding is preferable to implicit modeling of view-dependent and view-independent feature fields when one is interested in segmentation only. To evaluate the impact of explicit modeling, we considered a baseline that implicitly models view-dependent and independent feature fields by excluding the Reflection, IDE and roughness components (Eq. 5 and Eq. 6). As seen in Tab. 1, this results in inferior performance.

4.2.3 View Dependent Segmentation. We consider the ability to segment the view-dependent reflective surfaces of given objects. To this end, we begin by segmenting semantic objects using the view-independent features according to Sec. 3.4 and subsequently select only a subset of points corresponding to features from the view-dependent (reflectance) feature field. Fig. 5 illustrates for different novel views our success in segmentation for the Garden-spheres

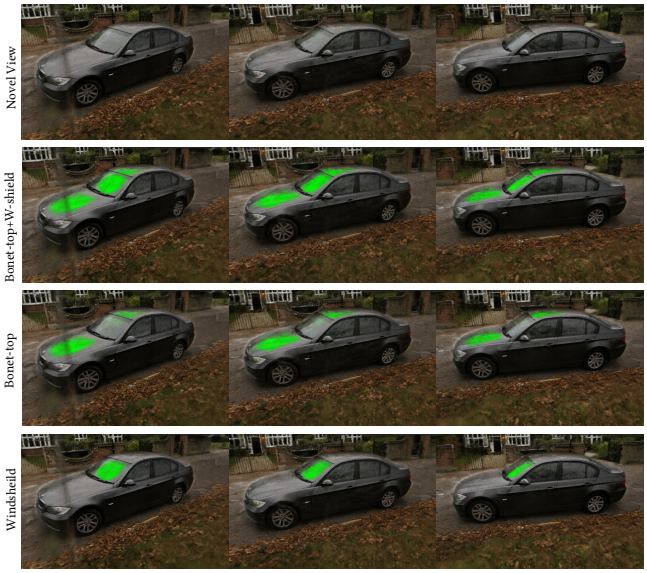

Fig. 6. Segmentation of the reflective component of different semantic components of the real-world car scene. The first row displays three novel views. We then demonstrate the segmentation of the reflective component of (1). Both the bonnet-top and the windshield (second row), (2). The bonnet-top (third row), and (3). The windshield (fourth row).

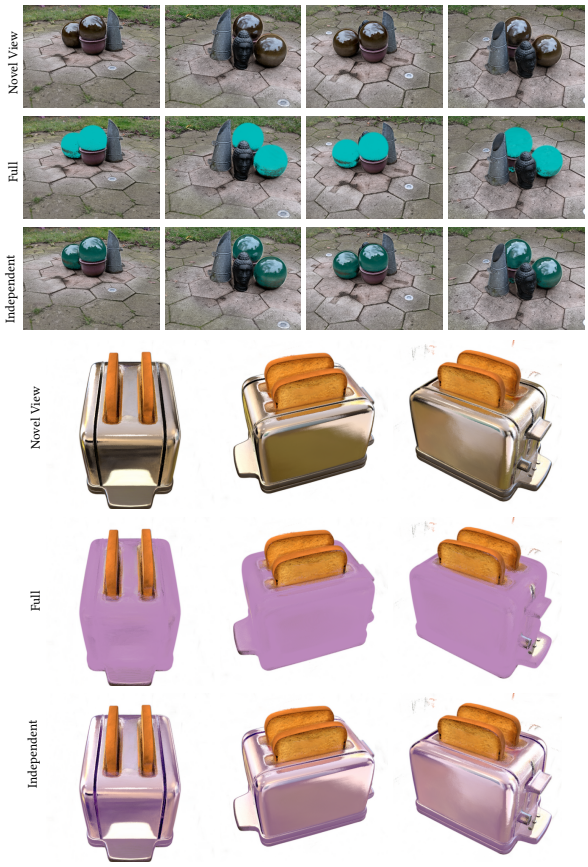

Fig. 7. Editing the color of i). The spheres in the real-world gardensphere-sand (ii). The toaster in the synthetic toaster, either (1). using all radiance fields, (*full*) or (2). using the independent component only (*indepdent*).

scene, for both the entire sphere as well as only the view-dependent reflection. Fig. 6 illustrates the ability to segment the reflective region of the (1). bonnet-top and the windshield, (2). bonnet-top, and (3). windshield. As can be seen, only the reflective region of the desired semantic entity is depicted. Additional examples are provided in the webpage.

4.3 Removal

Several works in the literature [Mirzaei et al. 2023; Wang et al. 2023b; Weder et al. 2023] consider the problem of 3D “inpainting" or 3D object “removal", where one wishes to remove a 3D foreground object, resulting in a realistic-looking background scene. Given our structurally disentangled representation, we can now explore a novel capability of “inpainting" or “removing" the reflective part of an object. This can be achieved by selecting the 3D points corresponding to a given object (using a click) and then rendering for those points only the color component from the view-independent radiance field.

Fig. 8 shows the removal of the reflective component of the bonnet-top and windshield for the car scene and the Garden-spheres scene. Our method enables the removal of the reflected radiance from the object and the retention of its diffuse color.

Fig. 8. Removing the reflective component of of the (i). Bonet-top and windshield in the sedan scene (ii). spheres in the gardenspheres scene.

4.4 Editing

4.4.1 Color Editing. We consider the ability to edit the color of an object while adhering to its reflections. In Fig. 7, we change the color of the segmented 3D points of (i). the spheres, for the real-world Garden-spheres scene, and (ii). the toaster body for the synthetic toaster scene. This color change occurs either (1). using both radiance fields (independent and reflection) or (2). using the independent component only, leaving the view-dependent reflective component unchanged. Using the latter results in a more natural manipulation that correctly adheres to reflections. Additional examples are shown in the webpage. To assess our color editing abilities numerically, we conducted a user study on colored objects. We consider 5 colors and 5 scenes (1 object each) and asked users to assess: 1. Color faithfulness ("how well does the desired color match the object?"), 2. Realism ("how realistic does object look?"), 3. Reflections match the unedited scene ("how well do the reflections match the unedited scene?"). We considered 25 users and obtained a MOS score (1-worse, 5-best) of **4.6**/**3.1**/**4.6** vs. 2.0/2.8/**4.6** in comparison to DFF for questions 1/2/3 respectively.

4.4.2 Roughness Editing. Next, we consider the ability to manipulate physical components introduced by our architectural design. In Eq. 6, we utilize the roughness parameter κ that controls the roughness. To this end, we consider the ability to manipulate the roughness of individual objects in the scene. We do so by segmenting the 3D points of an object and varying the roughness parameter κ for those points. Fig. 9 illustrates two examples: (i). the spheres in the real-world scene of the Garden-spheres scene, and (ii). the helmet case for the synthetic helmet scene. Additional examples are provided in the webpage.

4.4.3 Ablation Study. In Fig. 10, we consider an ablation in which our segmentation is performed using not only the independent feature component $\mathbf{f}_{indep}$, but also the independent and dependent components together, $\mathbf{f}_{indep} + \mathbf{f}_{ref}$. As can be seen, this results in a worse segmentation.

We also consider two additional ablations: (1) We optimized the appearance and feature model together, as opposed to first training the appearance model only and then the feature model (see Sec. 3.3). For 3D segmentation (as in Tab 1), on average, it is worse, e.g., we get mIOU 0.648 (vs our 0.856) for Shiny Blender. (2). We also conducted an ablation where we removed the tone mapping function (Eq. 7). We observe that appearance is slightly worse. Specifically, for the Gardenphere scene we obtain PSNR/LPIPS/SSIM of 28.8/0.180/0.809 vs our 28.9/0.180/0812. This results in a similar minor performance drop for 3D segmentation.

4.4.4 Limitations. While our method is designed for segmenting and editing reflective regions of semantic objects in a scene, it cannot do so at the instance-based level. For example, for the spheres in Fig. 1, selecting one of the spheres will result in capturing and editing both spheres. As in standard multiview reconstruction, errors can occur when the number of multiview ground truth features is not sufficient, or when they are noisy erroneous. We note that our task is highly unsupervised, as we aim to disentangle semantics and appearance in 3D space, given only 2D entangled appearance and semantics supervision. Improved physical modeling of, e.g., reflections and enhanced and generalizable feed-forward models may result in improved performance.

5 CONCLUSION

We have presented a novel method for 3D scene understanding and editing by distilling pretrained 2D features into a structurally disentangled feature field representation. Our approach effectively captures both view-dependent and view-independent features, enabling superior 3D segmentation compared to prior work. We have demonstrated the unique capability of our method to segment and manipulate reflective components of objects, as well as edit object colors and roughness while preserving realistic reflections. These capabilities open up new avenues for physically based understanding and editing of 3D scenes, with potential applications in augmented reality and content creation.

In future work, beyond 3D segmentation, we aim to improve the 3D consistency of underlying foundation models such as DINOv2. To do so, one can rerender GT views only using the view-independent component, discarding their view-dependent component and then fintue DINOv2 on such features. This can enable the learning of view-independent 2D features in general, which could be useful for image/semantic correspondence. Further, we aim to extend our method to incorporate additional physical properties, such as lighting, for more comprehensive 3D scene manipulation, as well as additional features, such as those derived from text.

REFERENCES

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. 2022. Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 5470–5479.

Sai Bi, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. 2020. Neural reflectance fields for appearance acquisition. *arXiv preprint arXiv:2008.03824* (2020).

Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. 2021a. Nerd: Neural reflectance decomposition from image collections. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 12684–12694.

Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan Barron, and Hendrik Lensch. 2021b. Neural-pil: Neural pre-integrated lighting for reflectance decomposition. *Advances in Neural Information Processing Systems* 34 (2021), 10691–10704.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. 2021. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*. 9650–9660.

Mohamed El Banani, Amit Raj, Kevis-Kokitsi Maninis, Abhishek Kar, Yuanzhen Li, Michael Rubinstein, Deqing Sun, Leonidas Guibas, Justin Johnson, and Varun Jampani. 2024. Probing the 3d awareness of visual foundation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 21795–21806.

Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. 2020. Implicit Geometric Regularization for Learning Shapes. In *Proceedings of Machine Learning and Systems 2020*. 3569–3579.

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

Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. 2023. LERF: Language Embedded Radiance Fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 19729–19739.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollar, and Ross Girshick. 2023. Segment Anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 4015–4026.

Sosuke Kobayashi, Eiichi Matsumoto, and Vincent Sitzmann. 2022. Decomposing NeRF for Editing via Feature Field Distillation. *Advances in Neural Information Processing Systems* 35 (2022), 23311–23330.

Boyi Li, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and Rene Ranftl. 2022. Language-driven Semantic Segmentation. In *International Conference on Learning Representations*. https://openreview.net/forum?id=RriDjddCLN

Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. 2023. Neuralangelo: High-fidelity neural surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8456–8465.

Ruofan Liang, Huiting Chen, Chunlin Li, Fan Chen, Selvakumar Panneer, and Nandita Vijaykumar. 2023. Envidr: Implicit differentiable renderer with neural environment lighting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 79–89.

Xiaoxiao Long, Cheng Lin, Peng Wang, Taku Komura, and Wenping Wang. 2022. Sparseneus: Fast generalizable neural surface reconstruction from sparse views. In *European Conference on Computer Vision*. Springer, 210–227.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. *Commun. ACM* 65, 1 (2021), 99–106.

Ashkan Mirzaei, Tristan Aumentado-Armstrong, Konstantinos G Derpanis, Jonathan Kelly, Marcus A Brubaker, Igor Gilitschenski, and Alex Levinshtein. 2023. SPIn-NeRF: Multiview segmentation and perceptual inpainting with neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 20669–20679.

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

Michael Oechsle, Songyou Peng, and Andreas Geiger. 2021. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 5589–5599.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. 2023. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193* (2023).

Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. 2023. LangSplat: 3D Language Gaussian Splatting. *arXiv preprint arXiv:2312.16084* (2023).

Ravi Ramamoorthi et al. 2009. Precomputation-based rendering. *Foundations and Trends® in Computer Graphics and Vision* 3, 4 (2009), 281–369.

Radu Alexandru Rosu and Sven Behnke. 2023. Permutosdf: Fast multi-view reconstruction with implicit surfaces using permutohedral lattices. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8466–8475.

Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulò, Norman Müller, Matthias Nießner, Angela Dai, and Peter Kontschieder. 2023. Panoptic lifting for 3d scene understanding with neural fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9043–9052.

Pratul P Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T Barron. 2021. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7495–7504.

Vadim Tschernezki, Iro Laina, Diane Larlus, and Andrea Vedaldi. 2022. Neural Feature Fusion Fields: 3D distillation of self-supervised 2D image representations. In *2022 International Conference on 3D Vision (3DV)*. IEEE, 443–453.

Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. 2022. Ref-NeRF: Structured View-Dependent Appearance for Neural Radiance Fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 5491–5500.

Dongqing Wang, Tong Zhang, Alaa Abboud, and Sabine Süsstrunk. 2023b. Inpaint-nerf360: Text-guided 3d inpainting on unbounded neural radiance fields. *arXiv preprint arXiv:2305.15094* (2023).

Fangjinhua Wang, Marie-Julie Rakotosaona, Michael Niemeyer, Richard Szeliski, Marc Pollefeys, and Federico Tombari. 2023a. UniSDF: Unifying Neural Representations for High-Fidelity 3D Reconstruction of Complex Scenes with Reflections. *arXiv preprint arXiv:2312.13285* (2023).

Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. 2021. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689* (2021).

Silvan Weder, Guillermo Garcia-Hernando, Aron Monszpart, Marc Pollefeys, Gabriel J Brostow, Michael Firman, and Sara Vicente. 2023. Removing objects from neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 16528–16538.

Jingfeng Yao, Xinggang Wang, Lang Ye, and Wenyu Liu. 2024. Matte anything: Interactive natural image matting with segment anything model. *Image and Vision Computing* (2024), 105067.

Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. 2021. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems* 34 (2021), 4805–4815.

Lior Yariv, Peter Hedman, Christian Reiser, Dor Verbin, Pratul P Srinivasan, Richard Szeliski, Jonathan T Barron, and Ben Mildenhall. 2023. Bakedsdf: Meshing neural sdfs for real-time view synthesis. In *ACM SIGGRAPH 2023 Conference Proceedings*. 1–9.

Jianglong Ye, Naiyan Wang, and Xiaolong Wang. 2023. FeatureNeRF: Learning Generalizable NeRFs by Distilling Foundation Models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 8962–8973.

Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. 2022. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. *Advances in neural information processing systems* 35 (2022), 25018–25032.

Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah Snavely. 2021a. 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.

Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. 2021b. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics (ToG)* 40, 6 (2021), 1–18.

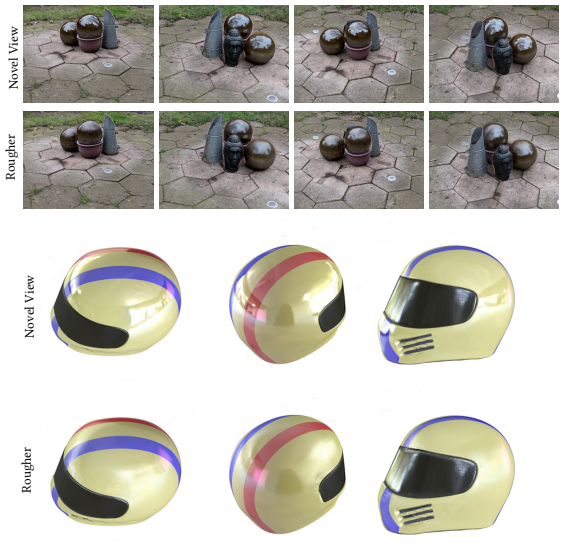

Fig. 9. Controlling the roughness of (i). The spheres in the real-world Gardenspheres, and (ii). The helmet (only the case) for the synthetic helmet scene.

A APPENDIX

A.1 Implementation Details

We implement our method using Pytorch, based on the public implementation of RefNeRF [Verbin et al. 2022]. In the iNGP hash grid hierarchy, we use 15 levels (from 32 to 4096) with 4 channels for each level, where each level has 4 channels. Two rounds of proposal sampling are used as in MipNeRF-360 [Barron et al. 2022]. We also penalize the mean of the sum of squared grid-hash values with a loss multiplier of 0.1. Our models are trained on a single A100 GPU for around 2 hours per scene. The code will also be made publically available.

A.2 Computational requirements

We require around 30 seconds to render a 500x400 resolution view at inference. This scales linearly with resolution. To perform segmentations and edits as detailed, we require about 60 seconds per frame on average. As for memory, we require an average of 22GB per scene when considering high-resolution scenes with full-resolution 2D ground views and corresponding semantics. Full per-scene memory requirements will be added in the next revision.

A.3 Per Scene IOU Results

In table 2, we provide per-scene and per-object results accompanying Tab. 1 of the main text. As can be seen, our method results in a superior performance, particularly in reflective objects such as the car windshield.

Scene - Object	Ours	DFF	DFF+ Ref
Bicycle - Bench	**0.750**	0.659	0.627
Bicycle - Wheels	**0.415**	0.378	0.380
Counter - Mittens	0.869	0.787	**0.907**
Counter - Plant	0.755	**0.791**	0.256
Counter - Tray	**0.847**	0.561	0.722
Garden - Ball	0.770	**0.838**	0.732
Garden - Plant	**0.805**	0.795	0.737
Garden - Tabletop	**0.946**	0.840	0.631
Kitchen - Lego	**0.871**	0.856	0.850
Gardenspheres - Cone	**0.845**	0.833	0.817
Gardenspheres - Head	**0.764**	0.739	0.749
Gardenspheres - Spheres	**0.866**	0.757	0.742
Sedan - Bonnet-top	0.471	0.452	**0.497**
Sedan - Windshield	**0.659**	0.569	0.342
Sedan - Hubcaps	0.746	**0.764**	0.653
Sedan - Wheel	0.697	**0.714**	0.557
Toycar - Body	0.775	0.693	**0.801**
Toycar - Wheel	0.644	**0.686**	0.507
Teapot - Cover	**0.762**	0.529	0.399
Car - Windshield	**0.838**	0.357	0.260
Car - Wheels	**0.760**	0.688	0.630
Toaster - Body	0.938	**0.967**	0.764
Toaster - Toasts	0.873	**0.913**	0.811
Helmet - Body	**0.929**	0.900	0.860
Helmet - Windshield	**0.796**	0.562	0.084

Table 2. Per Scene results accompanying Tab. 1 of the main text. We consider mean IoU for segmentation of objects from the Shiny Blender synthetic dataset [Verbin et al. 2022] (bottom) and real-world scenes (top). The first four scenes are taken from the real-world RefNeRF [Verbin et al. 2022] dataset while the other four real-world scenes are from the Mip-NeRF-360 dataset. We compare our approach to DFF [Kobayashi et al. 2022] and a baseline where DFF is optimized for features while RefNeRF is optimized for appearance. For each scene, we indicate the segmented objects.

Fig. 10. An ablation study on 3D semantic segmentation from four novel views for the real world garden scene, comparing the segmentation obtained by clicking on each object and using (i). our view-independent feature component (used as default), or (ii) using the combination of both the view-independent and view-dependent components.