Neural Reflectance Surfaces (NeRS)

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recent history has seen a tremendous growth of work exploring implicit representations of geometry and radiance, popularized through Neural Radiance Fields (NeRFs). Such works are fundamentally based on a (implicit) *volumetric* representation of occupancy, allowing them to model diverse scene structure including translucent objects and atmospheric obscurants. But because the vast majority of real-world scenes are composed of well-defined surfaces, we introduce a *surface* analog of such implicit models called Neural Reflectance Surfaces (NeRS). NeRS learns a neural shape representation of a closed surface that is diffeomorphic to a sphere, guaranteeing water-tight reconstructions. Even more importantly, surface parameterizations allow NeRS to learn (neural) bidirectional surface reflectance functions (BRDFs) that factorize view-dependent appearance into environmental illumination, diffuse color (albedo), and specular "shininess." Finally, rather than illustrating our results on synthetic scenes or controlled in-the-lab capture, we assemble a novel dataset of multiview images from online marketplaces for selling goods, such as Craigslist advertisements of cars, bicycles, and other items. Such "in-the-wild" multiview image sets pose a number of challenges, including a small number of views with unknown/rough camera estimates. We demonstrate that surface-based neural reconstructions enable learning from such data, outperforming volumetric neural rendering-based reconstructions. We hope that NeRS serve as a first step toward building scalable, high-quality libraries of real-world shape, materials, and illumination.

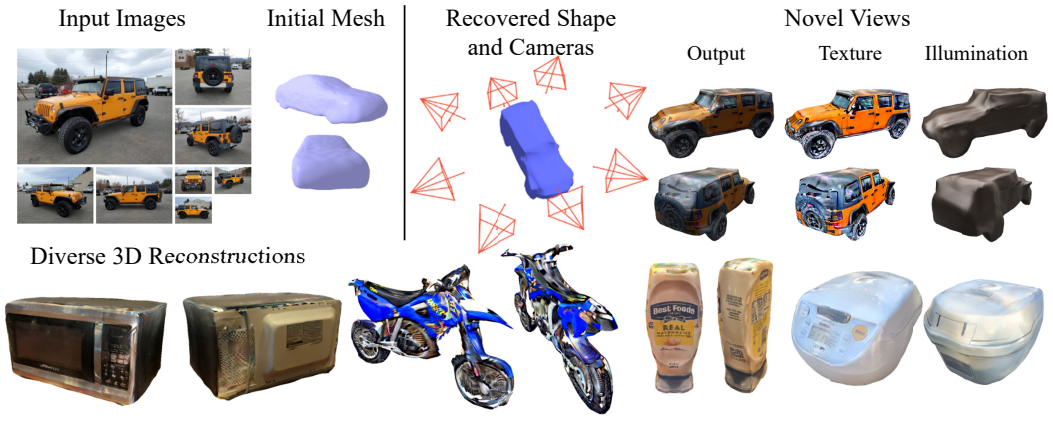

Figure 1: **3D view synthesis in the wild.** From several multiview internet images of a truck and a coarse initial mesh (top left), we recover the camera poses, 3D shape, texture, and illumination (top right). We demonstrate the scalability of our approach on a wide variety of indoor and outdoor object categories (second row). Please see the supplementary video for 360 degree visualizations.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

1 Introduction

Although we observe the surrounding world only via 2D percepts, it is undeniably 3D. The goal of recovering this underlying 3D from 2D observations has been a longstanding one in the vision community, and any computational approach aimed at this task must answer a central question about representation—how should we model the geometry and appearance of the underlying 3D structure?

An increasingly popular answer to this question is to leverage neural *volumetric* representations of density and radiance fields (Mildenhall et al., 2020). This allows modeling structures from rigid objects to translucent fluids, while further enabling arbitrary view-dependent lighting effects. However, it is precisely this unconstrained expressivity that makes it less robust and unsuitable for modeling 3D objects from sparse views in the wild. While these neural volumetric representations have been incredibly successful, they require hundreds of images, typically with precise camera poses, to model the full 3D structure and appearance of real-world objects. In contrast, when applied to 'in-the-wild' settings *e.g.* a sparse set of images with imprecise camera estimates from off-the-shelf systems (see Fig. 1), they are unable to infer a coherent 3D representation. We argue this is because these neural volumetric representations, by allowing arbitrary densities and lighting, are *too* flexible.

Is there a robust alternative that captures real-world 3D structure? The vast majority of real-world objects and scenes comprise of well-defined *surfaces*. This implies that the geometry, rather than being an unconstrained volumetric function, can be modeled as a 2D manifold embedded in euclidean 3D space—and thus encoded via a (neural) mapping from a 2D manifold to 3D. Indeed, such meshed surface manifolds form the heart of virtually all rendering engines (Foley et al., 1996). Moreover, instead of allowing arbitrary view-dependent radiance, the appearance of such surfaces can be described using (neural) bidirectional surface reflection functions (BRDFs), themselves developed by the computer graphics community over decades. We operationalize these insights into *Neural Reflectance Surfaces* (NeRS), a surface-based neural representation for geometry and appearance.

NeRS represents shape using a neural displacement field over a canonical sphere, thus constraining the geometry to be a watertight surface. This representation crucially associates a surface normal to each point, which enables modeling view-dependent lighting effects in a physically grounded manner. Unlike volumetric representations which allow unconstrained radiance, NeRS factorizes surface appearance using a combination of diffuse color (albedo) and specularity. It does so by learning neural texture fields over the sphere to capture the albedo at each surface point, while additionally inferring an environment map and surface material properties. This combination of a surface constraint and a factored appearance allows NeRS to learn efficiently and robustly from a sparse set of images in-the-wild, while being able to capture varying geometry and complex view-dependent appearance.

Using only a coarse category-level template and approximate camera poses, NeRS can reconstruct instances from a diverse set of classes. Instead of evaluating in a synthetic setup, we introduce a dataset sourced from marketplace settings where multiple images of a varied set of real-world objects under challenging illumination are easily available. We show NeRS significantly outperforms neural volumetric or classic mesh-based approaches in this challenging setup, and as illustrated in Fig. 1, is able to accurately model the view-dependent appearance via its disentangled representation. We hope that our approach and results highlight the several advantages that neural surface representations offer, and that our work serves as a stepping stone for future investigations.

2 Related Work

Surface-based 3D Representations. As they enable efficient representation and rendering, polygonal meshes are widely used in vision and graphics. In particular, morphable models (Blanz and Vetter, 1999) allow parametrizing shapes as deformations of a canonical template and can even be learned from category-level image collections (Cashman and Fitzgibbon, 2012; Kar et al., 2015). With the advances in differentiable rendering (Kato et al., 2018; Laine et al., 2020; Ravi et al., 2020), these have also been leveraged in learning based frameworks for shape prediction (Kanazawa et al., 2018; Gkioxari et al., 2019; Goel et al., 2020) and view synthesis (Riegler and Koltun, 2020). Whereas these approaches use an explicit discrete mesh, some recent methods have proposed using continuous neural surface parametrization like ours to represent shape (Groueix et al., 2018) and texture (Tulsiani et al., 2020).

However, all of these works leverage such surface representations for (coarse) single-view 3D prediction given a category-level training dataset. In contrast, our aim is to infer such a representation given multiple images of a single instance, and without prior training. Closer to this goal of representing a single instance in detail, contemporary approaches have shown the benefits of using videos (Yang et al., 2021; Li et al., 2020) to recover detailed shapes, but our work tackles a more challenging setup where correspondence/flow across images is not easily available. In addition, while these prior approaches infer the surface texture, they do not enable the view-dependent appearance effects that our representation can model.

Volumetric 3D and Radiance Fields. Volumetric representations for 3D serve as a common, and arguably more flexible alternative to surface based representations, and have been very popular for classical multi-view reconstruction approaches (Furukawa and Hernández, 2015). These have since been incorporated in deep-learning frameworks for shape prediction (Girdhar et al., 2016; Choy et al., 2016) and differentiable rendering (Yan et al., 2016; Tulsiani et al., 2017). Although these initial approaches used discrete volumetric grids, their continuous neural function analogues have since been proposed to allow finer shape (Mescheder et al., 2019; Park et al., 2019) and texture modeling (Oechsle et al., 2019).

Whereas the above methods typically aimed for category-level shape representation, subsequent approaches have shown particularly impressive results when using these representations to model a single instance from images (Sitzmann et al., 2019a,b) – which is the goal of our work. More recently, by leveraging an implicit representation in the form of a Neural Radiance Field, Mildenhall et al. (2020) showed the ability to model complex geometries and illumination from images. There has since been a flurry of impressive work to further push the boundaries of these representations and allow modeling deformation (Park et al., 2020; Pumarola et al., 2020), lighting variation (Martin-Brualla et al., 2020), and similar to ours, leveraging insights from surface rendering to model radiance (Yariv et al., 2020; Oechsle et al., 2021). However, unlike our approach which can efficiently learn from a sparse set of images with coarse cameras, these approaches rely on a dense set of multi-view images with precise camera localization to recover a coherent 3D structure of the scene. While recent concurrent work (Lin et al., 2021) does relax the constraint of precise cameras, it does so by foregoing view-dependent appearance, while still requiring a dense set of images.

Multiview Datasets. Many datasets study the longstanding problem of multiview reconstruction and view synthesis. However, they are often captured in controlled setups, small in scale, and not diverse enough to capture the span of real world objects. Middlebury (Seitz et al., 2006) benchmarks multi-view reconstruction, containing two objects with nearly Lambertian surfaces. DTU (Aanæs et al., 2016) contains eighty objects with various materials, but still captured in a lab with controlled lightnings. Freiburg cars (Sedaghat and Brox, 2015) captures 360 degree videos of fifty-two outdoor cars for multi-view reconstruction. ETH3D (Schöps et al., 2019) and Tanks and Temples (Knapitsch et al., 2017) contain both indoor and outdoor scenes but are still small in scale. Perhaps most relevant are large-scale datasets of real-world objects such as Redwood (Choi et al., 2016) and Stanford Products (Oh Song et al., 2016), but the data tends to be dominated by single-views or small baseline videos. In constrast, our Craiglist Multiview (CMV) dataset contains millions of uncurated multi-view captures of in-the-wild objects under various illumination conditions, making it suitable for studying and benchmarking algorithms for multiview reconstruction, view synthesis, and inverse rendering.

3 Method

Given a sparse set of input images of an object under natural lighting conditions, our goal is to model its shape and appearance. While recent neural volumetric approaches share a similar goal, they require a dense set of views with precise camera information to achieve it. Instead, our approach relies on only approximate camera pose estimates and a coarse category-level shape template. Our key insight is that instead of allowing unconstrained densities popularly used for volumetric representations, we can enforce a *surface* based 3D representation. Importantly, this allows view-dependent appearance variation by leveraging constrained reflection models, by decomposing appearance into diffuse and specular components. In this section, we first introduce our (neural) surface representation that captures the object's shape and texture, and then explain how illumination and specular effects can be modeled for rendering. We finally describe how our approach can learn in the challenging setting of in-the-wild images.

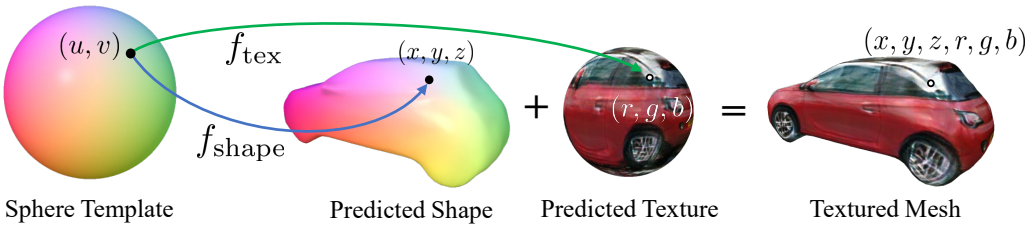

Figure 2: **Neural Surface Representation.** We propose an implicit, continuous representation of shape and texture. We model shape as a deformation of a unit sphere via a neural network f_{shape}, and texture as a learned per-uv color value via a neural network f_{tex}. We can discretize f_{shape} and f_{tex} to produce a textured mesh, as shown above.

3.1 Neural Surface Representation

We represent object shape via a deformation of a unit sphere. Previous works (Kanazawa et al., 2018; Goel et al., 2020) have generally modeled such deformations *explicitly*: the unit sphere is discretized at some resolution as a 3D mesh with V vertices. Predicting the shape deformation thus amounts to predicting vertex offsets $\delta \in \mathbb{R}^{V \times 3}$. Such *explicit discrete* representations have several drawbacks. First, they can be computationally expensive for dense meshes with fine details. Second, they lack useful spatial inductive biases as the vertex locations are predicted independently. Finally, the learned deformation model is fixed to a specific level of discretization, making it non-trivial, for instance, to allow for more resolution as needed in regions with richer detail. These limitations also extend to texture parametrization commonly used for such discrete mesh representations—using either per-vertex or per-face texture samples (Kato et al., 2018), or fixed resolution texture map, limits the ability to capture finer details.

Inspired by Groueix et al. (2018); Tulsiani et al. (2020), we address these challenges by adopting a continuous surface representation via a neural network. We illustrate this representation in Fig. 2. For any point u on the surface of a unit sphere $\mathbb{S}^2$, we represent its 3D deformation $x \in \mathbb{R}^3$ using the mapping $f_{\text{shape}}(u) = x$ where f_{shape} is parameterized as a multi-layer perceptron. This network therefore induces a deformation field over the surface of the unit sphere, and this deformed surface serves as our shape representation. We represent the surface texture in a similar manner – as a neural vector field over the surface of the sphere: $f_{\text{tex}}(u) = t \in \mathbb{R}^3$. This surface texture can be interpreted as an implicit UV texture map.

3.2 Modeling Illumination and Specular Rendering

Surface Rendering. The surface geometry and texture are not sufficient to infer appearance of the object *e.g.* a uniformly red car may appear darker on one side, and lighter on the other depending on the direction of incident light. In addition, depending on viewing direction and material properties, one may observe different appearance for the same 3D point *e.g.* shiny highlight from certain viewpoints. More formally, assuming that a surface does not emit light, the outgoing radiance L_o in direction v from a surface point x can be described by the rendering equation (Kajiya, 1986; Immel et al., 1986):

$$L_o(x, v) = \int_{\Omega} f_r(x, v, \omega) L_i(x, \omega)(\omega \cdot n) d\omega \qquad (1)$$

where Ω is the unit hemisphere centered at surface normal n, and ω denotes the negative direction of incoming light. $f_r(x, v, \omega)$ is the bidirectional reflectance function (BRDF) which captures material properties (e.g. color and shininess) of surface S at x, and $L_i(x, \omega)$ is the radiance coming toward x from ω (Refer to Fig. 3). Intuitively, this integral computes the total effect of the reflection of every possible light ray ω hitting x bouncing in the direction v.

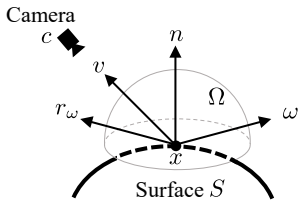

Figure 3: **Notation and convention for viewpoint and illumination parameterization.** The camera at c is looking at point x on the surface S. v denotes the direction of the camera w.r.t x, and n is the normal of S at x. Ω denotes the unit hemisphere centered about n. We compute the light arriving in the direction of every $\omega \in \Omega$, and r is the reflection of w about n.

We thus need to infer the environment lighting and surface material properties to allow realistic renderings. However, learning arbitrary lighting L_i or reflection models f_r is infeasible given sparse

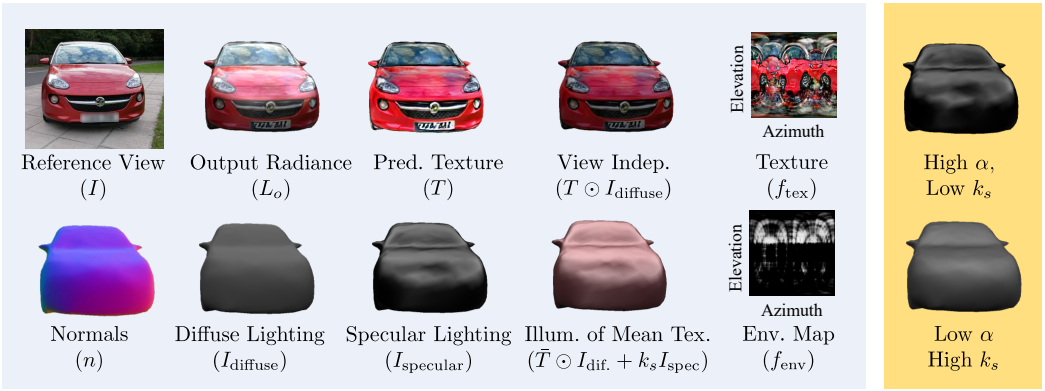

Figure 4: **Components of learned illumination model.** Given a query camera viewpoint (illustrated via the reference image I), we recover the radiance output L_o, computed using Phong shading (Phong, 1975). Here, we show the full decomposition of learned components. From the environment map f_{env} and normals n, we compute diffuse (I_{diffuse}) and specular lighting I_{specular}. The texture and diffuse lighting form the view-independent component ("View Indep.") and the specular lighting (weighted by the specular coefficient k_s) forms the view-dependent component of the radiance. Altogether, the output radiance $L_o = T \odot I_{\text{diffuse}} + k_s I_{\text{specularity}}$ (5). We also visualize the radiance using the mean texture, which is used to help learn plausible illumination. In the yellow box, we visualize the effects of the two specularity parameters. The shininess α controls the mirror-ness/roughness of the surface. The specular coefficient k_s controls the intensity of the specular highlights

views, and we need to further constrain these to allow learning. Inspired by concurrent work (Wu et al., 2021) that demonstrated its efficacy when rendering rotationally symmetric objects, we leverage the Phong reflection model (Phong, 1975) with the lighting represented as a neural environment map.

Neural Environment Map. Using environment maps intuitively corresponds to the assumption that all the light sources are infinitely far away. This allows a simplified model of illumination, where the incoming radiance only depends on the direction ω and is independent of the position x *i.e.* $L_i(x, \omega) \equiv I_\omega$. We implement this as a neural spherical environment map f_{env} which learns to predict the incoming radiance for any query direction:

$$L_i(x, \omega) \equiv I_\omega = f_{\text{env}}(\omega) \tag{2}$$

We note that there is a fundamental ambiguity between material properties and illumination, e.g. a car that appears red could be a white car under red illumination, or a red car under white illumination. To avoid this, we follow Wu et al. (2021), and further constrain the environment illumination to be grayscale, *i.e.* $f_{\text{env}}(\omega) \in \mathbb{R}$.

Appearance under Phong Reflection. Instead of allowing an arbitrary BRDF f_r, the Phong reflection model decomposes the outgoing radiance from point x in direction v into the diffuse and specular components. The *view-independent* portion of the illumination is modeled by the diffuse component:

$$I_{\text{diffuse}}(x) = \sum_{\omega \in \Omega} (\omega \cdot n) I_\omega, \tag{3}$$

while the *view-dependent* portion of the illumination is modeled by the specular component

$$I_{\text{specular}}(x, v) = \sum_{\omega \in \Omega} (r_{\omega,n} \cdot v)^\alpha I_\omega, \tag{4}$$

where $r_{\omega,n} = 2(\omega \cdot n)n - \omega$ is the reflection of ω about the normal n. The shininess coefficient $\alpha \in (0, \infty)$ is a property of the surface material and controls the "mirror-ness" of the surface. If α is high, the specular highlight will only be visible if v aligns closely with r_ω. Altogether, we compute the radiance of x in direction v as:

$$L_o(x, v) = T(x) \cdot I_{\text{diffuse}}(x) + k_s \cdot I_{\text{specular}}(x, v) \tag{5}$$

where the specularity coefficient k_s is another surface material property that controls the intensity of the specular highlight. $T(x)$ is the texture value at x computed by f_{tex}. Please see Fig. 4 for a full decomposition of these components.

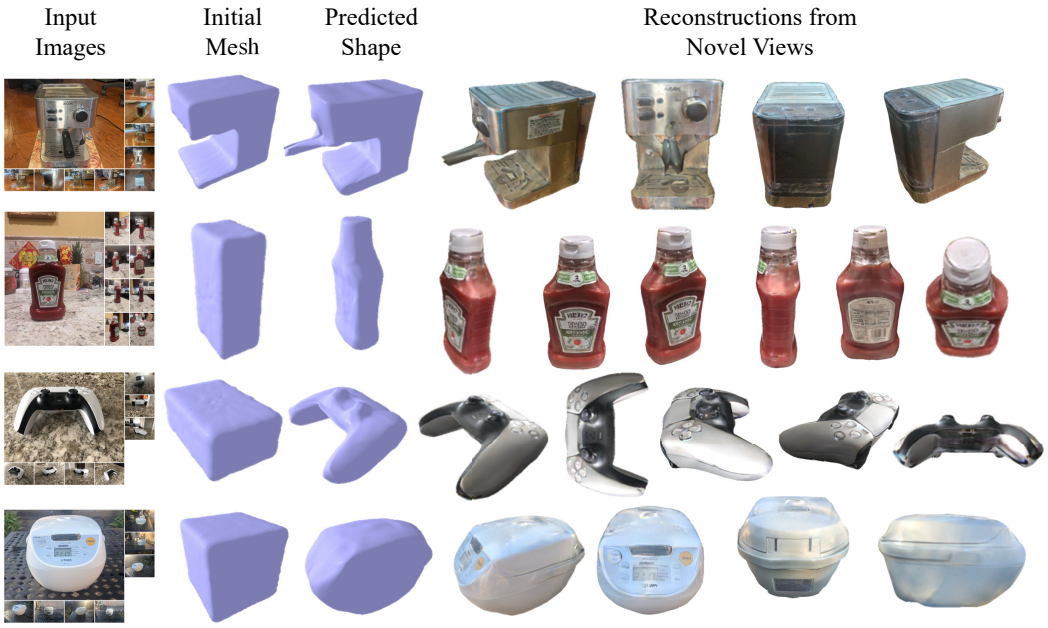

Figure 5: **Qualitative results on various household objects.** We demonstrate the versatility of our approach on an espresso machine, a bottle of ketchup, a game controller, and a rice cooker. Each instance has 8-10 input views. We find that a coarse, cuboid mesh is sufficient as an initialization to learn detailed shape and texture. We initialize the camera poses by hand, roughly binning in increments of 45 degrees azimuth.

3.3 Learning NeRS in the Wild

Given a sparse set of images in-the-wild, our approach aims to infer a NeRS representation, which when rendered, matches the available input. Concretely, our method takes as input N (typically 8) images of the same instance $\{I_i\}_{i=1}^N$, noisy camera rotations $\{R_i\}_{i=1}^N$, and a category-specific mesh initialization $\mathcal{M}$. Using these, we aim to optimize full perspective cameras $\{\Pi\}_{i=1}^N$ as well as the neural surface shape f_{shape}, surface texture f_{text}, and environment map f_{env}. In addition, we also recover the material properties of the object, parametrized by a specularity coefficient k_s and shininess coefficient α.

Initialization. Note that both the camera poses and mesh initialization are only required to be coarsely accurate. We use an off-the-shelf approach (Xiao et al., 2019) to predict camera rotations, and we find that a cuboid is sufficient as an initialization for several instances (See Fig. 5). We also use off-the-shelf approaches (Kirillov et al., 2020; Rother et al., 2004) to compute masks $\{M_i\}_{i=1}^N$. We assume that all images were taken with the same camera intrinsics. We initialize the shared global focal length f to correspond to a field of view of 60 degrees, and set the principal point at center of each image. We initialize the camera pose with the noisy initial rotations R_i and a translation t_i such that the object is fully in view. We pre-train f_{shape} to output the template mesh $\mathcal{M}$.

Rendering. To render an image, NeRS first discretizes the neural shape model $f_{\text{shape}}(u)$ over spherical coordinates u to construct an explicit triangulated surface mesh. This triangulated mesh and camera Π_i are fed into PyTorch3D's differentiable renderer (Ravi et al., 2020) to obtain per-pixel (continuous) spherical coordinates and associated surface properties:

$$[UV, N, \hat{M}_i] = \text{Rasterize}(\pi_i, f_{\text{shape}}) \tag{6}$$

where $UV[p]$, $N[p]$, and $M[p]$ are (spherical) uv-coordinates, normals, and binary foreground-background labels corresponding to each image pixel p. Together with the environment map f_{env} and specular material parameters (α, k_s), these quantities are sufficient to compute the outgoing radiance at each pixel p under camera viewpoint Π_i using (5). In particular, denoting by $v(\Pi, p)$ the viewing direction for pixel p under camera Π, and using $u \equiv UV[p], n \equiv N[p]$ for notational brevity, the intensity at pixel p can be computed as:

$$\hat{I}[p] = f_{\text{tex}}(u) \cdot \Big(\sum_{\omega \in \Omega} (\omega \cdot n)\ f_{\text{env}}(\omega) \Big) + k_s \Big(\sum_{\omega \in \Omega} (r_{\omega,n} \cdot v(\Pi, p))^a\ f_{\text{env}}(\omega) \Big) \tag{7}$$

Image loss. We compute a perceptual loss (Zhang et al., 2018) $L_{\text{perceptual}}(I_i, \hat{I}_i)$ that compares the distance between the rendered and true image using off-the-shelf VGG deep features. Note that being able to compute a perceptual loss is a significant benefit of surface-based representations over volumetric approaches such as NeRF (Mildenhall et al., 2020), which operate on batches of rays rather than images, due to the computational cost of volumetric rendering. Similar to Wu et al. (2021), we find that an additional rendering loss using the mean texture (see Fig. 4 and Fig. 7 for examples with details in the supplement) helps learn visually plausible lighting.

Mask Loss. To measure disagreement between the rendered and measured silhouettes, we compute a mask loss $L_{\text{mask}} = \frac{1}{N}\sum_{i=1}^{N}\|M_i - \hat{M}_i\|_2^2$, distance transform loss $L_{\text{dt}} = \frac{1}{N}\sum_{i=1}^{N} D_i \odot \hat{M}_i$, and 2D chamfer loss $L_{\text{chamfer}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{p\in E(M_i)} \min_{\hat{p}\in\hat{M}_i}\|p-\hat{p}\|_2^2$. D_i refers to the Euclidean distance transform of mask M_i, $E(\cdot)$ computes the 2D pixel coordinates of the edge of a mask, and $\hat{p}$ is every pixel coordinate in the predicted silhouette.

Regularization. Finally, to encourage smoother shape whenever possible, we incorporate a mesh regularization loss $L_{\text{regularize}} = L_{\text{normals}} + L_{\text{laplacian}}$ consisting of a normals consistency loss and Laplacian smoothing loss (Nealen et al., 2006; Desbrun et al., 1999). Note that such geometry regularization is another benefit of surface representations over volumetric ones. Altogether, we minimize:

$$L = \lambda_1 L_{\text{mask}} + \lambda_2 L_{\text{dt}} + \lambda_3 L_{\text{chamfer}} + \lambda_4 L_{\text{perceptual}} + \lambda_5 L_{\text{regularize}} \tag{8}$$

w.r.t $\Pi_i = [R_i, t_i, f]$, α, k_s, and the weights of f_{shape}, f_{text}, and $f_{\text{env\_map}}$. Please refer to the supplemental materials for the details of all hyperparameters.

Optimization. We find it helpful to optimize in a coarse-to-fine fashion, starting with just a few parameters and slowly increasing the number of free parameters. We initially optimize (8), w.r.t only the camera parameters Π_i. After convergence, we sequentially optimize f_{shape}, f_{tex}, and $f_{\text{env}}/\alpha/k_s$. We implement NeRS in PyTorch, using Pytorch3D (Ravi et al., 2020) as our differentiable renderer. We optimize (8) using the Adam optimizer (Kingma and Ba, 2014). We find it helpful to sample a new set of spherical coordinates u each iteration when rasterizing. This helps propagate gradients over a larger surface and prevent aliasing. With 4 Nvidia 1080TI GPUs, training NeRS requires about 45 minutes. Please see the supplement for hyperparameters and additional details.

4 Evaluation

In this section, we demonstrate the versatility of Neural Reflectance Surfaces to recover meaningful shape, texture, and illumination from in-the-wild indoor and outdoor images.

Multiview Marketplace Dataset. To address the shortage of in-the-wild multiview datasets, we introduce a new dataset, Craigslist Multiview (CMV), collected from the online marketplace Craigslist. We collected hundreds of thousands of car listings, consisting of over 5 million images. Each user-submitted listing contains seller images of the car being sold. We train a classifier to filter out images of the interior of the car. We curate a subset of size 600 with at least 8 exterior views, averaging 10 exterior images per listing. We use Xiao et al. (2019) to compute rough camera poses. CMV contains a large variety of cars under various illumination conditions (e.g. indoors, overcast, sunny, snowy, etc). The dataset contains personally identifiable information in the form of license plates and contact information. We intend to anonymize all personally identifiable information and then release filtered images, masks, initial camera poses, and optimized NeRS cameras.

Novel View Synthesis. Since NeRS can be trained with noisy camera inputs, we recover a NeRS for every instance in our dataset. We hand selected 40 instances for which the optimized cameras at the end of optimization appeared most accurate (note that these camera parameters are provided to all baselines). From these 40 instances, we randomly selected 20 as our evaluation dataset. We evaluate all approaches on novel view synthesis. Each instance in the evaluation set has N images and N optimized cameras. We treat one of the images as the target, and train each method using the remaining $N-1$ images and cameras. Finally, we compute view synthesis quality using the held out image and camera. We follow this process for every image for every instance. In total, we evaluate on 192 view synthesis tasks (20 instances, with an average of 9.6 images per instance).

Baselines. We evaluate our approach against Neural Radiance Fields (NeRF) (Mildenhall et al., 2020), the predominant approach in neural view synthesis. NeRF is normally trained with many views

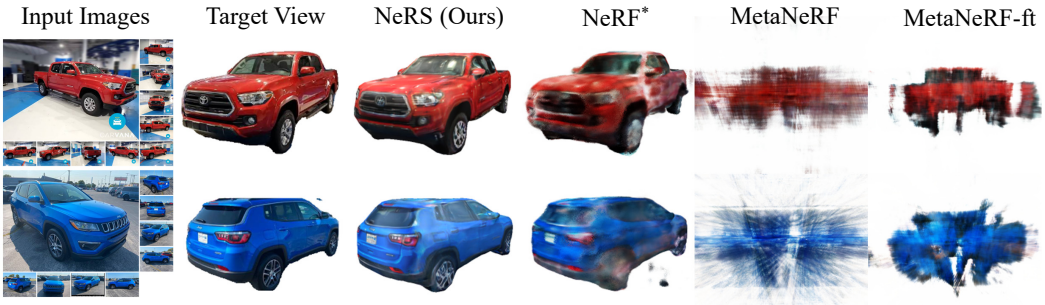

Figure 6: **Qualitative comparison of NeRS with our baselines.** We evaluate all baselines on the task of novel view synthesis on Craigslist Multiview. Since we do not have ground truth cameras, we treat the optimized cameras from our method as the camera input for the NeRF baselines. We train a modified version (See Sec. 4 for details) of NeRF (Mildenhall et al., 2020) that is more competitive in the few-shot domain (NeRF*). We also evaluate against a meta-learned initialization of NeRF with and without finetuning until convergence (Tancik et al., 2021), but found poor results perhaps due to the domain shift from Shapenet cars. We find that NeRS synthesizes novel views that are qualitatively closer to the target. Note that for both images, our method achieves the best perceptual similarity score, while NeRF* suprisingly achieves the best PSNR, MSE, and SSIM, indicating the tendency of such metrics to favor blurry results in the presence of minor camera noise. As corroborated by past work (Yan et al., 2016), we find perceptual error to far more consistent with qualitative results. The red truck has 16 total views while the blue SUV has 8 total views.

(50-100+) and ground truth cameras, neither of which we have in our in-the-wild low-data regime. In the absence of ground truth cameras, NeRF uses cameras computed from COLMAP (Schonberger and Frahm, 2016). However, we find that COLMAP consistently fails to produce reasonable camera poses due to specularities and the limited number of views. As a result, we treat the optimized cameras from our approach as the input cameras to NeRF. We find that a vanilla NeRF struggles with such few views. As such, we make the following changes to make the NeRF baseline as competitive as possible: 1. we add a mask loss that forces rays to either pass through or be absorbed entirely by the neural volume, analogous to space carving (Kutulakos and Seitz, 2000); 2. a canonical volume that zeros out the density outside of a tight box where the car is likely to be. This helps avoid spurious "cloudy" artifacts from novel views; 3. a smaller architecture to reduce overfitting; and 4. a single rendering pass rather than dual coarse/fine rendering passes. We denote this modified NeRF as NeRF*. We also evaluate against a simplified NeRF with a meta-learned initialization for cars from multiview images (Tancik et al., 2021), which we denote as MetaNeRF. MetaNeRF meta-learns an initialization such that with just a few gradient steps, it can learn a NeRF model. This setup allows the model to learn a data-driven prior about the shape of cars. Note that MetaNeRF is trained on ShapeNet (Chang et al., 2015) and thus has seen more data. We find that with the default number of gradient steps, MetaNeRF does not converge on images from our CMV dataset, and cannot recreate the training views. Thus, we also evaluate MetaNeRF-ft, which is finetuned until convergence. Finally, we reiterate that all the baselines have the additional benefit of using our final optimized cameras and do not deal with the noisier initializations (unlike our approach).

Metrics. We evaluate all approaches using the traditional image similarity metrics Mean-Square Error (MSE), Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity Index Measure (SSIM). We note that these traditional metrics correlate poorly with human perceptual distance (Zhang et al., 2018). In particular, these metrics tend to favor blurrier results (See Fig. 6). For a better metric that more accurately matches visual perceptual error, we compute the Learned Perceptual Image Patch Similarity (LPIPS) (Zhang et al., 2018). Finally, we compute Fréchet Inception Distance (Heusel et al., 2017) between the novel view renderings and original images as a measure of visual realism. In Tab. 1, we find that NeRS significant outperforms the baselines in perceptual similarity (LPIPS) and in realism (FID) while remaining competitive on the traditional metrics. NeRS consistently produces outputs with higher frequencies that match the level of detail of the real images, possibly at the cost of small pixel shifts. As a result, the blurrier results of NeRF perform better on the traditional image metrics. See Fig. 6 for a visual comparison of the methods on novel view synthesis.

Qualitative Results. In Fig. 7, we show qualitative results on our Craigslist Multiview Dataset. Each car instance has between 8 and 16 views. We visualize the outputs of our reconstruction from 3 novel views. We show the rendering for both the full radiance model and the mean texture. Both of

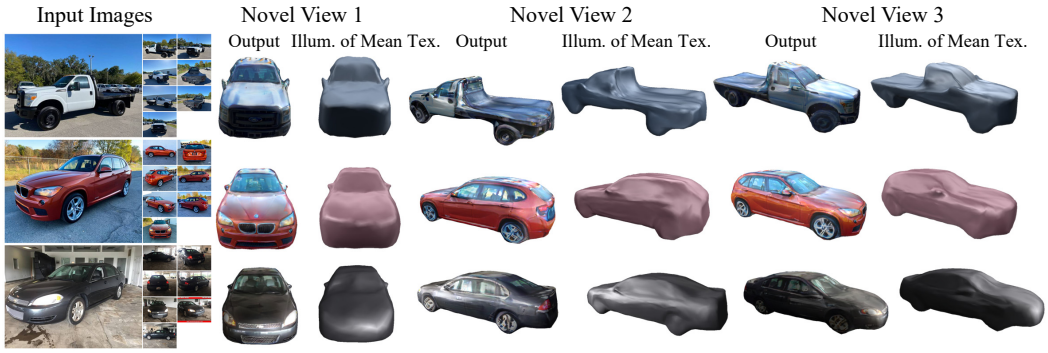

Figure 7: **Qualitative results on our in-the-wild Craigslist Multiview Dataset.** Here we visualize the NeRS outputs as well as the illumination of the mean texture on 3 of listings from the CMV dataset. We find that NeRS recovers detailed textures and plausible illumination conditions. Each instance has 8 input views.

Method	MSE ↓	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓
MetaNeRF (Tancik et al., 2021)	0.0983	10.5	0.554	0.639	390.2
MetaNeRF-ft (Tancik et al., 2021)	0.0942	10.7	0.634	0.524	329.6
NeRF* (Mildenhall et al., 2020)	0.0394	**15.8**	**0.689**	0.299	236.3
NeRS (Ours)	**0.0351**	15.3	0.675	**0.212**	**81.9**

Table 1: **Quantitative evaluation of novel-view synthesis on our in-the-wild multiview cars dataset.** We evaluate against NeRF*, a modified variant of NeRF, and a meta-learned initialization to NeRF. NeRS significant outperforms the baselines on the perceptual similarity metric (LPIPS) and realism evalauted via Fréchet Inception Distance (FID). NeRS performs competitively on the traditional image similarity metrics PSNR and SSIM. We note that the traditional similarity metrics do not correlate well with human perceptual similarity (See Fig. 6).

these renderings are used to compute the perceptual loss (See Sec. 3.2). We find that NeRS recovers detailed texture information and plausible illumination parameters. To demonstrate the scalability of our approach, we also evaluate on various household objects in Fig. 5. We find that a coarse, cuboid mesh is sufficient as an initialization to recover detailed shape, texture, and lighting conditions. Please refer to the **supplementary video for 360 degree visualizations**.

5 Discussion

We present NeRS, an approach for learning neural surface models that capture geometry and surface reflectance. In contrast to volumetric neural rendering, NeRS enforces reconstructions to be water-tight and closed manifolds. This allows NeRS to model surface-based appearance affects, including view-dependant specularities and normal-dependant diffuse appearance. We demonstrate that such regularized reconstructions allow for learning from sparse in-the-wild multiview data, enabling reconstruction of objects with diverse material properties across a variety of indoor/outdoor illumination conditions. We hope NeRS will enable construction of high-quality libraries of real-world geometry, materials, and environments through better neural approximations of shape, reflectance, and illuminants.

Limitations. Though NeRS makes use of factorized models of illumination and material reflectance, there exists some fundamental ambiguities that are difficult from which to recover. For example, it difficult to distinguish between an image of a gray car under bright illumination and an image of a white car under dark illumination. We visualize such limitations in the supplement.

Broader Impacts. NeRS reconstructions could be used to reveal identifiable or proprietary information (e.g., license plates).

References

Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, pages 1–16, 2016.

Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 187–194, 1999.

Thomas J Cashman and Andrew W Fitzgibbon. What shape are dolphins? building 3d morphable models from 2d images. *IEEE transactions on pattern analysis and machine intelligence*, 35(1): 232–244, 2012.

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

Sungjoon Choi, Qian-Yi Zhou, Stephen Miller, and Vladlen Koltun. A large dataset of object scans. *arXiv:1602.02481*, 2016.

Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016.

Mathieu Desbrun, Mark Meyer, Peter Schröder, and Alan H Barr. Implicit fairing of irregular meshes using diffusion and curvature flow. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 317–324, 1999.

James D Foley, Foley Dan Van, Andries Van Dam, Steven K Feiner, John F Hughes, Edward Angel, and J Hughes. *Computer graphics: principles and practice*, volume 12110. Addison-Wesley Professional, 1996.

Yasutaka Furukawa and Carlos Hernández. Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 9(1-2):1–148, 2015.

Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *European Conference on Computer Vision*, pages 484–499. Springer, 2016.

Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh r-cnn. In *ICCV*, 2019.

Shubham Goel, Angjoo Kanazawa, , and Jitendra Malik. Shape and viewpoints without keypoints. In *ECCV*, 2020.

Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *CVPR*, pages 216–224, 2018.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *arXiv preprint arXiv:1706.08500*, 2017.

David S Immel, Michael F Cohen, and Donald P Greenberg. A radiosity method for non-diffuse environments. *Acm Siggraph Computer Graphics*, 20(4):133–142, 1986.

James T Kajiya. The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 143–150, 1986.

Angjoo Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018.

Abhishek Kar, Shubham Tulsiani, Joao Carreira, and Jitendra Malik. Category-specific object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1966–1974, 2015.

Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3907–3916, 2018.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9799–9808, 2020.

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017.

Kiriakos N Kutulakos and Steven M Seitz. A theory of shape by space carving. *International journal of computer vision*, 38(3):199–218, 2000.

Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics*, 39(6), 2020.

Xueting Li, Sifei Liu, Shalini De Mello, Kihwan Kim, Xiaolong Wang, Ming-Hsuan Yang, and Jan Kautz. Online adaptation for consistent mesh reconstruction in the wild. In *NeurIPS*, 2020.

Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. *arXiv preprint arXiv:2104.06405*, 2021.

Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. *arXiv preprint arXiv:2008.02268*, 2020.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, pages 4460–4470, 2019.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, pages 405–421. Springer, 2020.

Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. Laplacian mesh optimization. In *Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia*, pages 381–389, 2006.

Michael Oechsle, Lars Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger. Texture fields: Learning texture representations in function space. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4531–4540, 2019.

Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. *arXiv preprint arXiv:2104.10078*, 2021.

Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4004–4012, 2016.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, pages 165–174, 2019.

Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo-Martin Brualla. Deformable neural radiance fields. *arXiv preprint arXiv:2011.12948*, 2020.

Bui Tuong Phong. Illumination for computer generated pictures. *Communications of the ACM*, 18(6): 311–317, 1975.

Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020.

Gernot Riegler and Vladlen Koltun. Free view synthesis. In *European Conference on Computer Vision*, pages 623–640. Springer, 2020.

Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. " grabcut" interactive foreground extraction using iterated graph cuts. *ACM transactions on graphics (TOG)*, 23(3):309–314, 2004.

Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.

Thomas Schöps, Torsten Sattler, and Marc Pollefeys. BAD SLAM: Bundle adjusted direct RGB-D SLAM. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

N. Sedaghat and T. Brox. Unsupervised generation of a viewpoint annotated car dataset from videos. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. URL http://lmb.informatik.uni-freiburg.de//Publications/2015/SB15.

Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, volume 1, pages 519–528. IEEE, 2006.

Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2437–2446, 2019a.

Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems*, 2019b.

Matthew Tancik, Ben Mildenhall, Terrance Wang, Divi Schmidt, Pratul P. Srinivasan, Jonathan T. Barron, and Ren Ng. Learned initializations for optimizing coordinate-based neural representations. In *CVPR*, 2021.

Shubham Tulsiani, Tinghui Zhou, Alexei A Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2626–2634, 2017.

Shubham Tulsiani, Nilesh Kulkarni, and Abhinav Gupta. Implicit mesh reconstruction from unannotated image collections. *arXiv preprint arXiv:2007.08504*, 2020.

Shangzhe Wu, Ameesh Makadia, Jiajun Wu, Noah Snavely, Richard Tucker, and Angjoo Kanazawa. De-rendering the world's revolutionary artefacts. In *CVPR*, 2021.

Yang Xiao, Xuchong Qiu, Pierre-Alain Langlois, Mathieu Aubry, and Renaud Marlet. Pose from shape: Deep pose estimation for arbitrary 3D objects. In *British Machine Vision Conference (BMVC)*, 2019.

Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. *arXiv preprint arXiv:1612.00814*, 2016.

Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Huiwen Chang, Deva Ramanan, William T Freeman, and Ce Liu. Lasr: Learning articulated shape reconstruction from a monocular video. In *CVPR*, 2021.

Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems*, 33, 2020.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] On acceptance, we will release all code and data before the conference.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] We did not observe significant variance in our results, so we did not run experiments multiple times. The computational cost would also have been prohibitive.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]