UrbanIR: Large-Scale Urban Scene Inverse Rendering from a Single Video

Zhi-Hao Lin[1]   Bohan Liu[1]   Yi-Ting Chen[2]   David Forsyth[1]
Jia-Bin Huang[2]   Anand Bhattad[1]   Shenlong Wang[1]
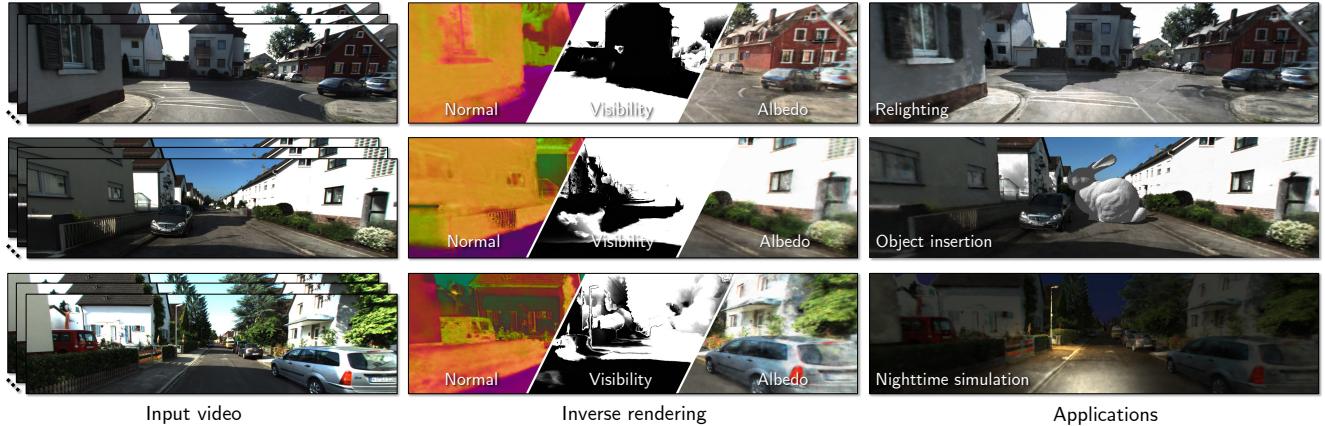[1]University of Illinois Urbana-Champaign   [2]University of Maryland, College Park

Figure 1: We present *UrbanIR* (**Urban** Scene **I**nverse **R**endering), a novel, realistic, and relightable neural scene model. UrbanIR concurrently infers shape, albedo, visibility and more from *a single video* of large-scale, unbounded scenes. The resulting representation facilitates realistic and controllable editing, delivering photorealistic free-viewpoint renderings (last column) of relit scenes (top row), inserted objects (middle row), and nighttime simulation (bottom row). Project page

Abstract

*We show how to build a model that allows realistic, free-viewpoint renderings of a scene under novel lighting conditions from video. Our method – UrbanIR (**Urban** Scene **I**nverse **R**endering) – computes an inverse graphics representation from the video. UrbanIR jointly infers shape, albedo, visibility, and sun and sky illumination from a single video of unbounded outdoor scenes with unknown lighting. UrbanIR uses videos from cameras mounted on cars (in contrast to many views of the same points in typical NeRF-style estimation). As a result, standard methods produce poor geometry estimates (for example, roofs), and there are numerous "floaters". Errors in inverse graphics inference can result in strong rendering artifacts. UrbanIR uses novel losses to control these and other sources of error. UrbanIR uses a novel loss to make very good estimates of shadow volumes in the original scene. The resulting representations facilitate controllable editing, delivering photorealistic free-viewpoint renderings of relit scenes and inserted objects. Qualitative evaluation demonstrates strong improvements over the state-of-the-art.*

1. Introduction

We show how to build a model that allows realistic, free-viewpoint renderings of a scene under novel lighting conditions from video. So, for example, a sunny afternoon video of a large urban scene can be shown at different times of day or night (as in Fig. 1), viewed from novel viewpoints, and shown with inserted objects. Our method – *UrbanIR* (**Urban** Scene **I**nverse **R**endering) – computes an inverse graphics representation from the video. UrbanIR jointly infers shape, albedo, specularity, visibility, and sun and sky illumination *from a single video of unbounded outdoor scenes* with *unknown lighting*.

Inferring these *inverse graphics* maps is challenging because doing so is ill-posed – there isn't enough of the right kind of data to recover canonical inferences, so errors are guaranteed. Errors in inverse graphics inference can result in strong rendering artifacts, including NeRF "floaters" casting shadows and shadow boundaries preserved in albedo maps. Our goal is realistic rendering (rather than canonical inference), and UrbanIR uses novel losses to control these and other sources of error. Further, UrbanIR uses a novel loss to make very good estimates of shadow volumes in the original scene. The resulting representations facilitate controllable

arXiv:2306.09349v2 [cs.CV] 16 Jun 2023

editing, delivering photorealistic free-viewpoint renderings of relit scenes and inserted objects, as demonstrated in Fig. 1.

UrbanIR combines monocular intrinsic decomposition and inverse rendering with key innovations to control error in renderings. UrbanIR uses videos from cameras mounted on cars (in contrast to many views of the same points in typical NeRF-style estimation). As a result, standard methods produce poor geometry estimates (for example, roofs) and numerous "floaters".

Our key contributions are:

- We use novel losses to control errors in geometric estimation and show significant improvements in the rendered images over alternative methods.

- We use a novel visibility rendering procedure to ensure consistency between detected shadows and scene geometry, significantly improving predicted shadows.

- We use monocular estimates of surface normal and shadows to supervise neural fields, and show that these estimates improve inverse graphics estimates.

2. Related Works

Inverse Graphics involves inferring illumination and intrinsic properties of a scene. It is difficult to achieve accurate and reliable solutions, and there is much reliance on priors [27, 20, 21, 2, 70, 1, 52, 43, 65] or on managed lighting conditions [19, 1, 15, 19, 1, 68], known geometry [51, 28, 11, 25], or material simplifications [75, 42, 70]. Recent methods use deep learning techniques to reason about material properties [37, 38, 39, 67, 73, 47]. Models trained on synthetic data [34] or pair-wise annotated data [3] have shown promising results. Some approaches focus on learning to predict monocular properties, such as albedo or shading, as demonstrated in several works [55, 14]. Others learn neural representations of materials and illumination [39, 32, 31, 29, 30]. In line with these methods, our proposed approach also leverages monocular cues, such as shadows and surface normals. In contrast, we combine learning-based monocular cues and model-based relightable NeRF optimization to infer the scene's intrinsic properties and illumination.

Relightable Neural Fields are now an active topic of research. Neural fields can capture complex and non-parametric scene structures by learning implicit representations, enabling more flexible and accurate modeling of geometry and producing visually realistic renderings. Relightable neural radiance field methods [69, 6, 72, 5, 46, 18, 61, 67] aim to factor the neural field into multiple intrinsic components and leverage neural shading equations for illumination and material modeling. These methods allow for the realistic and controllable rendering of scenes with varying lighting conditions and materials. However, most relightable NeRF methods focus on objects with surrounding views or small bounded indoor environments. There are two notable exceptions: NeRF-OSR [49], which assumes multiple lighting sources for decomposition, and Nerf meet explicit geometry [61], which either uses multiple lighting or exploits depth sensing, such as LiDAR. In contrast, our proposed approach only requires a single video captured under the same illumination, making it more applicable to a broader range of scenes.

Differentiable rendering techniques enable gradient propagation throughout the entire forward rendering process, making inverse graphics tasks more flexible and convenient. There are fast but not physically based rasterization-based methods as they assume Lambertian or simple lighting models [36, 7, 47]. Furthermore, most of these methods are based on meshes and are suitable for object-level rendering, and are difficult to apply to large urban scenes. In contrast, we leverage neural radiance fields (NeRF) [44] in conjunction with physically based differentiable rendering techniques.

Shadow modeling using images poses a significant challenge. Methods that are trained to produce cast shadows from images [60, 35, 71] work for very specific objects (pedestrians, cars, etc) and do not generalize beyond training categories. Some works have also employed deep learning techniques to detect and remove shadows from 2D images [16, 17, 59]. However, relying on 2D shadows alone does not fulfill the requirements of inverse graphics, as it requires modeling the full 3D geometry, scene properties, and ensuring temporal consistency. Model-based optimization methods have been used to infer shadows, including expensive multiple-bounce ray tracing or explicit geometry-based shadow casting, which relies on accurate scene geometry [57, 26, 63]. Other approaches utilize visibility fields to model shadows, but they often struggle to provide consistent shadows in relation to the underlying geometry [56, 64, 49, 74]. In contrast, our method combines the strengths of both learning-based monocular shadow prediction and removal and model-based inverse graphics. By blending these approaches, we achieve realistic, controllable, and consistent visibility representations that align with the underlying scene properties and the image observation. This combination allows us to overcome the limitations of relying solely on 2D shadows or visibility fields alone, offering a more comprehensive and accurate solution for modeling shadows in inverse graphics.

3. Method

UrbanIR takes as input multiple frames of video of a fixed scene under fixed illumination; the camera moves, and its motion is known. Write $\{I_i, E_i, K_i\}$, where $I_i \in \mathbb{R}^{H \times W \times 3}$ is the RGB image; $E_i \in \mathrm{SE}(3)$ is the camera pose; and K_i
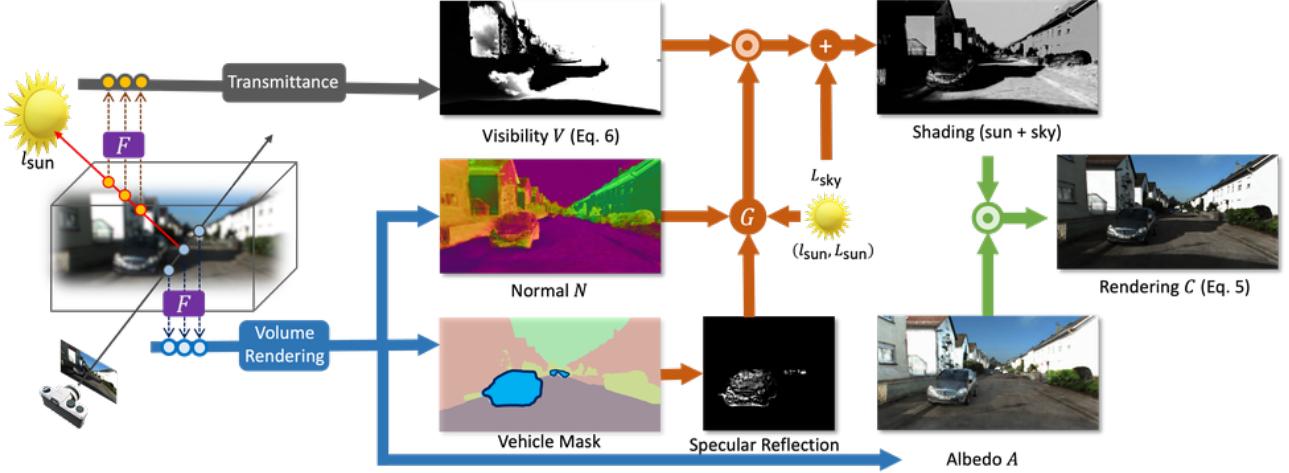
Figure 2: **Rendering Pipeline.** UrbanIR retrieves scene intrinsics (normal N, semantics S, albedo A) from camera rays, and estimate visibility V from tracing rays to the light source. Shading model computes diffuse and specular reflection with function $G(N, S, \mathbf{l}_{\text{sun}}, \mathbf{L}_{\text{sun}})$, and add sky light $\mathbf{L}_{\text{sky}}$ for final shading map. Final rendering is computed by multiplying shading and albedo A. Please refer to Eq. 5 for more details.

is camera intrinsic matrix. We produce a neural field model that can be viewed from *novel camera viewpoints* under *novel lighting conditions*. We do so by constructing a neural scene model that encodes albedo, normal, transmittance, and visibility in a unified manner (Sec. 3.1). This model is rendered from a given camera pose with given illumination using an end-to-end differentiable volume renderer (Sec. 3.2). The inference is by optimization of all properties jointly (Sec. 3.3). Applications include changing the sun angle, day-to-night transitions, and object insertion (Sec. 3.4). Fig. 3 provides an overview of our proposed inverse graphics and simulation framework.

3.1. Relightable Neural Scene Model

The scene representation is built on Instant-NGP [45, 48], a spatial hash-based voxel NeRF representation. Instant-NGP offers numerous advantages, including: low memory consumption; high efficiency in training and rendering; and compatibility with expansive outdoor scenes. Write $\mathbf{x} \in \mathbb{R}^3$ for position in 3D, $\mathbf{d}$ for query ray direction, θ for learnable scene parameters; NeRF models, including Instant-NGP, learn a radiance field $\mathbf{c}, \sigma = F(\mathbf{x}, \mathbf{d}; \theta)$, where $\mathbf{c} \in \mathbb{R}^3$ and $\sigma \in \mathbb{R}$ represent color and opacity respectively.

In contrast, UrbanIR learns a model of intrinsic scene attributes. Write albedo $\mathbf{a}$, surface normal $\mathbf{n}$, semantic vector $\mathbf{s}$, and density σ; then UrbanIR learns:

$$(\text{appearance}, \text{geometry}) = (F_a(\mathbf{x}; \theta_a); F_g(\mathbf{x}, \theta_g)) = F(\mathbf{x}; \theta) \quad (1)$$

where $\theta = \{\theta_g, \theta_a\}$ are learnable parameters. The appearance module $(\mathbf{a}, \mathbf{n}, \mathbf{s}) = F_a(\mathbf{x}, \theta_a)$ and the geometry module $\sigma = F_g(\mathbf{x}, \theta_g)$ are each linked to a separate feature hash ta-

ble, and an individual MLP header is used to encode each attribute within the semantic module. In contrast to current relightable outdoor scene models that demand coupled explicit geometry [49, 62] , our scene model is purely neural field-based, providing both compactness and consistency.

The lighting model is a parametric sun-sky model, after [22]. This encodes outdoor illumination as:

$$\mathbf{L} = \{(\mathbf{L}_{\text{sun}}, \psi_{\text{sun}}, \phi_{\text{sun}}), \mathbf{L}_{\text{sky}}\}. \quad (2)$$

The sun model is a directional 5-DoF representation, encoding sun color $\mathbf{L}_{\text{sun}}$ along with the azimuth and zenith $\psi_{\text{sun}}, \phi_{\text{sun}}$. The $\mathbf{L}_{\text{sky}}$ model is represented as a 3-DoF ambient color. Although simple and low-dimensional, this minimalist representation has proven to be highly effective in modeling various outdoor illumination effects [22].

3.2. Rendering

Rendering a query ray $\mathbf{r}$ through a pixel takes two steps: first, we compute the intrinsics that project to that pixel; and second, we shade the pixel using intrinsics and illumination model, yielding:

$$\mathbf{C}, W = \texttt{Shade}(\texttt{Intrinsics}(\mathbf{r}; \theta_a, \theta_g), \mathbf{x}, \mathbf{L}) \quad (3)$$

where $\mathbf{L}$ is the illumination model, $\mathbf{C}$ is the final RGB color and W is the alpha transparency.

Intrinsics are obtained by volume rendering. We accumulate predictions from $F(\cdot; \theta)$ along the query ray. Multiple points are sampled along the ray, and intrinsics at the query
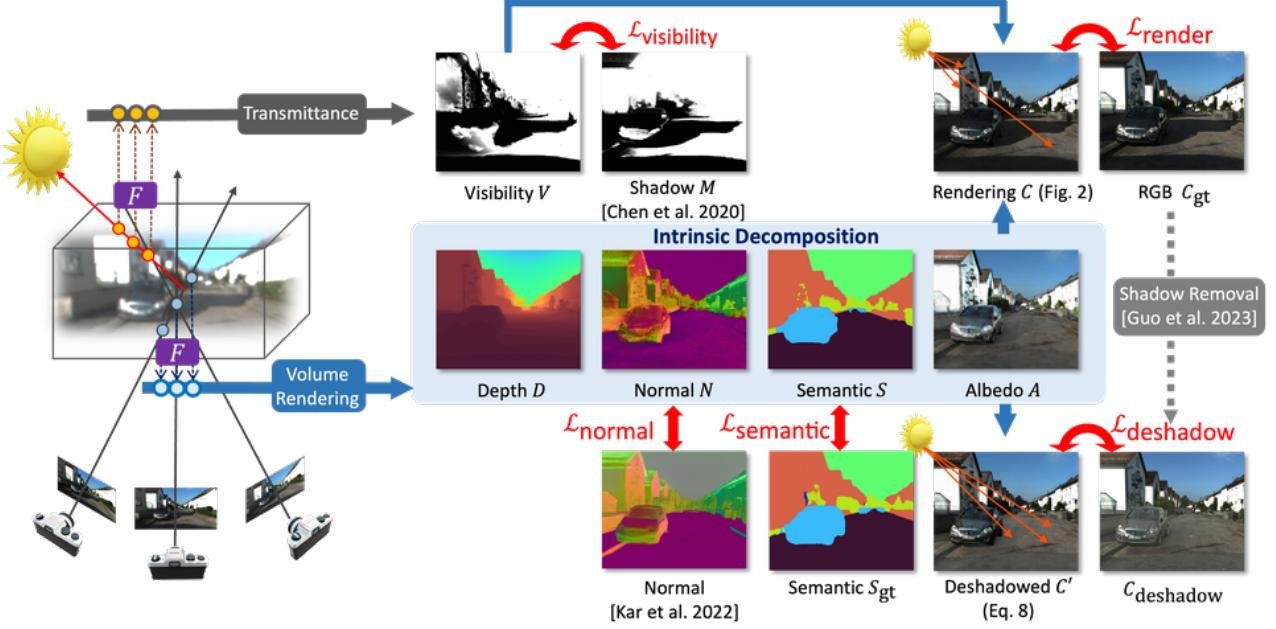
Figure 3: **Training Pipeline.** UrbanIR retrieves scene intrinsics with volume rendering from camera rays, which is guided by semantic and normal priors. Transmittance along tracing rays are supervised with shadow masks. Shading model (illustrated in Fig. 2) is performed **with** and **without** visibility term, and enforce reconstruction loss with original and deshadowed images, respectively. Please refer to Section 3.3 for more details.

pixel are obtained by volume rendering [23, 44] along the ray. In particular, albedo $\mathbf{A}$, normal $\mathbf{N}$ and semantics $\mathbf{S}$ are predicted as:

$$\mathbf{A} = \sum_{i=1}^{N} w_i \mathbf{a}_i, \quad \mathbf{N} = \sum_{i=1}^{N} w_i \mathbf{n}_i, \quad \mathbf{S} = \sum_{i=1}^{N} w_i \mathbf{s}_i, \quad (4)$$

where $w_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j)(1 - \exp(-\sigma_i \delta_i))$ is alpha-composition weight, $\delta_i = t_i - t_{i-1}$, and the intrinsic attributes follow from the neural scene model, so $(\mathbf{a}_i, \mathbf{n}_i, \mathbf{s}_i, \sigma_i) = F_\theta(\mathbf{x}_i)$.

Shading is by a local shading model (cf Blinn-Phong [4]) that incorporates sun and sky terms. We assume the sky is visible at every point, and compute

$$\begin{aligned} \mathbf{C} &= \mathbf{A} \odot (\texttt{sun} + \texttt{sky}) \\ &= \mathbf{A} \odot (\mathbf{L}_{\text{sun}} [\phi(\mathbf{N}, \mathbf{l}_{\text{sun}}) V(\mathbf{x}, \text{sun})] + \mathbf{L}_{\text{sky}}) \end{aligned} \quad (5)$$

where $\mathbf{x}$ is an estimate of the 3D position of the point being shaded (below), $\phi(\mathbf{N}, \mathbf{l}_{\text{sun}}) = \max(\mathbf{N} \cdot \mathbf{l}_{\text{sun}}, 0)$ is the cosine foreshortening at the surface, $\mathbf{l}_{\text{sun}}$ is the unit vector toward the sun (derived from $\psi_{\text{sun}}, \phi_{\text{sun}}$). The visibility $V(\mathbf{x}, \text{sun})$ is 1 if $\mathbf{x}$ can see the sun, and 0 otherwise. This shading model is capable of producing a realistic appearance with shadows in accordance with varying lighting conditions. The model can readily be extended.

Accurate visibility estimates are essential for obtaining realistic-looking shadows. Modeling the visibility of the sun with an MLP head (as in [74, 72]) is impractical because we need to change the sun's position but can learn from only one position. An alternative is to construct an explicit geometry model to cast shadows, but this model might not be consistent with the other neural fields, and imposing consistency is difficult. Instead, we first compute an estimate $\mathbf{x}$ of the 3D point being shaded, then estimate a smoothed $V(\mathbf{x}, \text{sun})$

We obtain $\mathbf{x}$ by volume rendering depth (so substitute $\hat{t} = \sum w_i t_i$ into the equation for the ray being rendered). Now to check whether $\mathbf{x}$ is visible to the light source, we evaluate the transmittance along the ray segment between $\mathbf{x}$ and the light source using volume rendering, obtaining:

$$V(\mathbf{x}, \text{sun}) = \exp\left(-\sum_i \sigma_i(\mathbf{x}_i)\delta_i\right) \quad \text{where} \quad \mathbf{x}_i = \mathbf{x} + t_i \mathbf{l}_{\text{sun}} \quad (6)$$

Lower transmittance along a ray from a surface point to a light source suggests fewer obstacles between the point and the light source. Eq. 6 establishes a strong link between transmittance, lighting, and visibility fields, which we use in training. In particular, a point in a training image that is known to be shadowed (resp. out of shadow) should have large (resp. small) accumulated transmittance. This constraint adjusts quite distant geometry during training.

3.3. Inverse graphics

We train the scene model $F(\cdot; \theta)$ (Eq. 1) and the lighting model $\mathbf{L}$ (Eq. 2) jointly using a loss:

$$\texttt{argmin}_{\theta, \mathbf{L}} \mathcal{L}_{\text{render}} + \mathcal{L}_{\text{deshadow}} + \mathcal{L}_{\text{visibility}} + \mathcal{L}_{\text{normal}} + \mathcal{L}_{\text{semantics}}, \quad (7)$$

where individual loss terms are described below.

The rendering loss measures the agreement between observed images and images rendered from the model using the training view and lighting, yielding $\mathcal{L}_{\text{render}} = \sum_{\mathbf{r}} \|\mathbf{C}_{\text{gt}}(\mathbf{r}) - \mathbf{C}(\mathbf{r})\|_2^2$, where $\mathbf{C}$ is rendered color per ray, as defined in Eq. 3, and $\mathbf{C}_{\text{gt}}$ is the observed "ground-truth" color. Minimizing the rendering loss ensures our scene model can reproduce observed images.

The deshadowed rendering loss forces shadow effects out of the estimated albedo. In particular, we compute a shadow-free version of an image using an off-the-shelf shadow detection and removal network [16, 8] to obtain $\mathbf{C}_{\text{deshadow}}$. We then render that image from the model using the training view and lighting, but assuming that every point can see the sun (equivalently $V(\mathbf{x}, \text{sun}) = 1$ for every $\mathbf{x}$). This yields $\mathbf{C}'(\theta)$. We then measure the agreement between the two to obtain $\mathcal{L}_{\text{deshadow}} = \sum_{\mathbf{r}} |\mathbf{C}_{\text{deshadow}} - \mathbf{C}'(\theta)|^2$. The combination of this loss and the original rendering loss directly gauges how the visibility map influences rendering, and helps disentangling albedo and shadows.

The visibility loss exploits shadow detection to improve geometry estimates. A pixel that is known to be in shadow must be at a point that cannot see the sun, so constraining geometry along a ray from that pixel to the sun. This loss could be computed by simply comparing visibility $V(, \text{sun}; \theta)$ with the shadow masks used for $\mathcal{L}_{\text{deshadow}}$. However, there are challenges: first, computing visibility requires another volume rendering per sample point; second, back-propagation through volume rendering, shading, and visibility computation forms a long, non-linear gradient chain, and optimization becomes difficult. Instead, we construct an intermediate "guidance" visibility estimate $V_i(\mathbf{x}; \theta_i)$ which is an MLP head trained to reproduce the shadow masks, and compute

$$\mathcal{L}_{\text{visibility}} = \sum_{\mathbf{r} \in \mathcal{R}} \text{CE}\left(M(\mathbf{r}), V_i(\mathbf{r}; \theta_i)\right) + \text{CE}\left(V(\mathbf{r}; \theta), V_i(\mathbf{r}; \theta_i)\right), \quad (8)$$

where $M(\mathbf{r})$ is the shadow mask at pixel $\mathbf{r}$, , and $\text{CE}(.,.)$ is a cross-entropy loss. Here the first term forces the (relatively easily trained) V_i to agree with the shadow masks, and the second forces V to agree with V_i.

The normal loss is computed by comparing results N_{gt} obtained from an off-the-shelf normal estimator [12, 24] to the output of the normal MLP. Recall the camera is known for training scenes and write $\mathbf{r}$ for the pixel corresponding to 3D point $\mathbf{x}(\mathbf{r})$. An alternate estimate of the normal follows from the density field: $\hat{\mathbf{n}}(\mathbf{r}) = -\frac{\nabla \sigma(\mathbf{x})}{\|\nabla \sigma(\mathbf{x})\|}$. Then the normal loss is:

$$\mathcal{L}_{\text{normal}} = \sum_{\mathbf{r} \in \mathcal{R}} \left(\|N_{\text{gt}}(\mathbf{r}) - N(\mathbf{r})\|^2 + \|\mathbf{n}(\mathbf{x}(\mathbf{r})) - \hat{\mathbf{n}}(\mathbf{x}(\mathbf{r}))\|^2 \right). \quad (9)$$

We also adopt normal regularization from Ref-NeRF [58], producing a better density field.

The semantic loss is computed by comparing predicted semantics $\mathbf{s}$ with labels in the dataset [33]. We use an additional loss to encourage high depth values in the sky region, yielding: $\mathcal{L}_{\text{semantics}} = \sum_{\mathbf{r} \in \mathcal{R}} \text{CE}\left(S_{\text{gt}}(\mathbf{r}), S(\mathbf{r})\right) - \sum_{\mathbf{r} \in \text{sky}} D(\mathbf{r})$.

3.4. Applications

Because we recover intrinsics, we can render the UrbanIR model with whatever source model appeals. Natural uses are showing scenes with different sun configurations and simulating nighttime.

Outdoor relighting proceeds by simply adjusting lighting parameters (position or color of the sun; sky color) then re-rendering the scene using Eq. 3. Additionally, we utilize semantics to interpret specular surfaces (cars) and emulate their reflectance during the simulation process.

Simulating nighttime proceeds by defining a spotlight model for headlights and street lights, then illuminating with that model. The spotlight we used is given by the center $\mathbf{o}_L \in \mathbb{R}^3$ and direction $\mathbf{d}_L \in \mathbb{R}^3$ of the light. This spotlight produces a radiance at $\mathbf{x}$ given by

$$\mathbf{L}_{\text{diffuse}}^{\text{spot}}(\mathbf{x}) = \frac{1}{\|\mathbf{o}_L - \mathbf{x}\|^2} \left(l \cdot \mathbf{d}_L\right)^k, l = \frac{\mathbf{o}_L - \mathbf{x}}{\|\mathbf{o}_L - \mathbf{x}\|}, \quad (10)$$

Intensity of spot light is brightest on the central ray $\mathbf{r}(t) = \mathbf{o}_L - t\mathbf{d}_L$, and decays with distance from the ray $\mathbf{r}(t)$ and algle, which is modulated with constant k.

Object insertion proceeds by a hybrid strategy. We first cast rays from the camera and estimate ray-mesh intersections [10]. If the ray hits the mesh and distance is shorter than volume rendering depth, the albedo $A(\mathbf{r})$, normal $N(\mathbf{r})$ and depth $D(\mathbf{r})$ are replaced with the object attributes. In the shadow pass, we calculate visibility from surface points to the light source (Eq. 6), and also estimate the ray-mesh intersection for the tracing rays. If the rays hit the mesh (meaning occlusion by the object), the visibility is also updated : $V(\mathbf{r}) = 0$. With updated $A(\mathbf{r}), N(\mathbf{r}), V(\mathbf{r})$, shading (Eq. 5) is applied to render images with virtual objects. Our method not only casts object shadows in the scene but also casts *scene shadows* on the object, enhancing realism significantly.

Figure 4: **Rendering and relighting comparison**. We show a set of two scenes comparing different methods. Each column shows a different sun position, the first column showing original images. For each set and from top to bottom, we have (a) NeRF-OSR [50] (b) COLMAP [54] + Blender [9] (c) Luma AI Unreal Engine Plugin [13] (d) Mesh-based visibility (Mesh-vis) and (e) UrbanIR (Ours). NeRF-OSR achieves good scene reconstruction but lacks the ability to perform relighting with a single lighting condition as it assumes access to images captured under multiple lighting conditions. In COLMAP, Shadows are present in the scene; however, they are "baked-in" and cannot be manipulated or relit independently. Luma has the same difficulty. For Mesh-based visibility, weak observations of the scene geometry result in poor quality visibility estimation during direct observation-based reconstruction. In contrast, for our approach, UrbanIR, our visibility optimization enables realistic and controllable relighting effects.

Figure 5: **Nighttime rendering.** The scene is transformed from daytime (top) to night-time (bottom) by introducing new light sources: a headlight on a car and a street lamp. The dark shadows with sharp boundaries that are present during the daytime (top) are successfully removed, resulting in a more evenly illuminated scene during the nighttime rendering (bottom).

4. Experiment Results

4.1. Datasets

We evaluate UrbanIR on the KITTI-360 dataset [33], which contains numerous video sequences of urban scenes. We pick 7 non-overlapping sequences for the experiments. These cover various light directions, vehicle trajectories, and layouts of buildings and vegetation. The data include RGB images from stereo cameras, semantic labels, camera poses, and RTK-GPS poses. We use Ominidata [12, 24] for monocular normal estimation from images, and use MTMT [8], ShadowFormer [16] to detect shadow masks and partially remove shadow patterns in the training images.

4.2. Baselines

We compare UrbanIR with the following methods:

NeRF-OSR [49] is recent work for outdoor scene reconstruction and relighting. We use the open-source project provided by the author for running this baseline. This method represents lighting as spherical harmonics parameters. For a fair comparison, we rotate the spherical vectors to simulate different light conditions.

Mesh + Blender [9] We compare our method with an explicit geometry-based baseline. For this, we utilize COLMAP for dense scene reconstruction [53, 54] and import the resulting scene into Blender [9] for relighting simulation.

Luma AI Unreal Engine Plugin [41, 13] Luma AI is a mobile app that employs NeRF-based reconstruction. With its Unreal Engine plugin [41, 13], models constructed by

Luma AI using NeRF can be imported into the Unreal Engine. This allows for relighting and object insertion within the engine. Luma AI performs its own custom scale-normalization, and the camera extrinsics are not readily accessible. As a result, we manually adjust its viewpoint transformation to align with other baselines. This adjustment could lead to minor misalignments in the qualitative results.

Mesh-based visibility The recent work FEGR [62] explores the relighting of outdoor scenes under singular or multiple illumination sources. However, due to the absence of open-source access to their method, we implement our own baseline model that incorporates similar visibility modeling strategies. Specifically, we employ the marching cubes technique [40] to extract a mesh from our model, excluding our proposed visibility optimization (as per Eq.8). In alignment with the shadow mapping approach adopted by FEGR[62], we cast shadows by estimating two intersections: the first between the camera rays and the mesh, and the second by tracing rays from the surface to the light source.

4.3. Relighting Quality

Relighting under various sunlight conditions are evaluated and compared in Fig. 4. NeRF-OSR [49] cannot simulate shadows under novel light conditions. While Blender [9] and Luma Field [41, 13] can change the lighting parameters explicitly, they either cast bad shadows due to incomplete geometry or do not cast new shadows at all; further, the original shadow remains unchanged in the image. Mesh-based visibility generates different shadows according to light conditions, but the mesh on the edges of and outside the training views is poor because there are few observations. This leads to noisy and incomplete shadows on the ground.
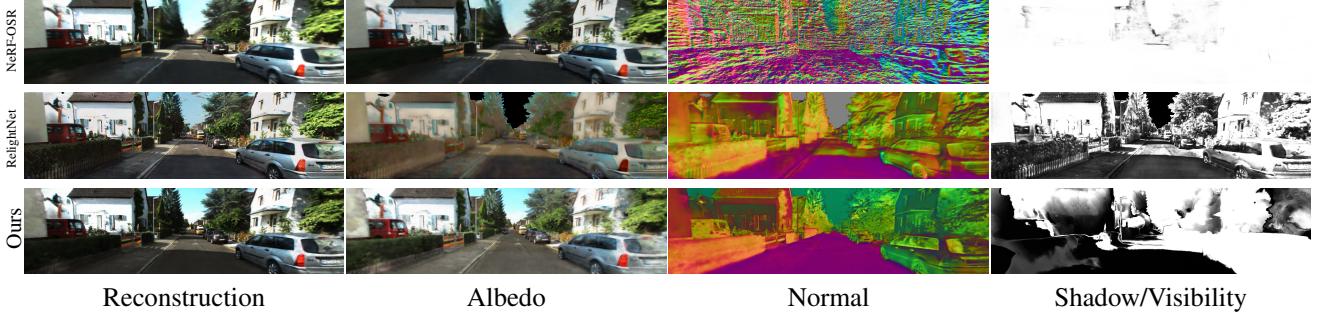
Figure 6: **Intrinsic Decomposition Quality Comparison.** (A) NeRF-OSR [50] (B) RelightNet [66] (2D-based). Note weak normals and shadow maps from NeRF-OSR; a tendency for RelightNet to make albedo *and* shadow dark in shadow regions; and dark shadows in NeRF-OSR albedo.

Figure 7: **Shadow Removal in Albedo.** Our albedo representation is guided by shadow removal model [16], which performs well on ground but cannot remove shadow on buildings and vehicles. Our method correctly recover albedo under shadow thanks to multi-view supervision and joint optimization.

UrbanIR synthesizes sharp shadows and varying surface shading following sun direction; further, the original scene shadows are largely absent. This allows synthesizing images at night (Fig. 5) by inserting car headlights and streetlights, without distracting effects from the original shadows.

4.4. Decomposition Quality

We compare with NeRF-OSR [49] and RelightNet [66] in Fig. 6. NeRF-OSR reconstructs a noisy normal map, and cannot capture the scene shadows in their shadow generation model, leaving dark shadow patterns in the albedo field. RelightNet predicts better normals but still bakes shadows into the albedo. UrbanIR generates clean and sharp albedo and normal fields and also produces a geometry-aware shadow field from the input video sequence.

In Fig. 7, we compare the learned albedo with the output of shadow removal network [16]. While ShadowFormer [16] recovers albedo well on the ground, but it cannot estimate the correct albedo for the building and vehicles. Our optimization process leverages deshadowed image as guidance ($\mathcal{L}_{\text{deshadow}}$) and further recovers a clean albedo field in most scene surfaces.

4.5. Object Insertion

The object insertion pipeline is described in Sect. 3.4. In Fig. 8, we compare with baselines by inserting a yellow cube and moving along the road.

Mesh+Blender cannot synthesize complete geometry and shadow, and LumaAI Unreal plugin cannot cast scene shadows on the object. Without visibility optimization in row (C), the scene shadow on objects is noisy. Our complete model has an accurate shadow volume so that shadows cast on the object by the environment are well represented. More complex objects appear in Fig. 9.

4.6. Ablation

Fig. 10 shows an ablation study. (A) **No deshadow loss** $\mathcal{L}_{\text{deshadow}}$: the model does not recover the albedo in the shadow region accurately, because reconstruction is ill-posed under single illumination. (B)**No intermediate visibility MLP**: If the visibility map $V(\mathbf{r})$ is supervised directly with shadow masks, the optimization is unstable and the model cannot decompose albedo successfully. (C)**No visibility loss** $\mathcal{L}_{\text{visibility}}$: the density field outside the training views is not constrained and cannot produce accurate and sharp shadows. (D)**Mesh-based visibility**: here sun visibility is calculated from a mesh, leading to sharp but inaccurate shadows.
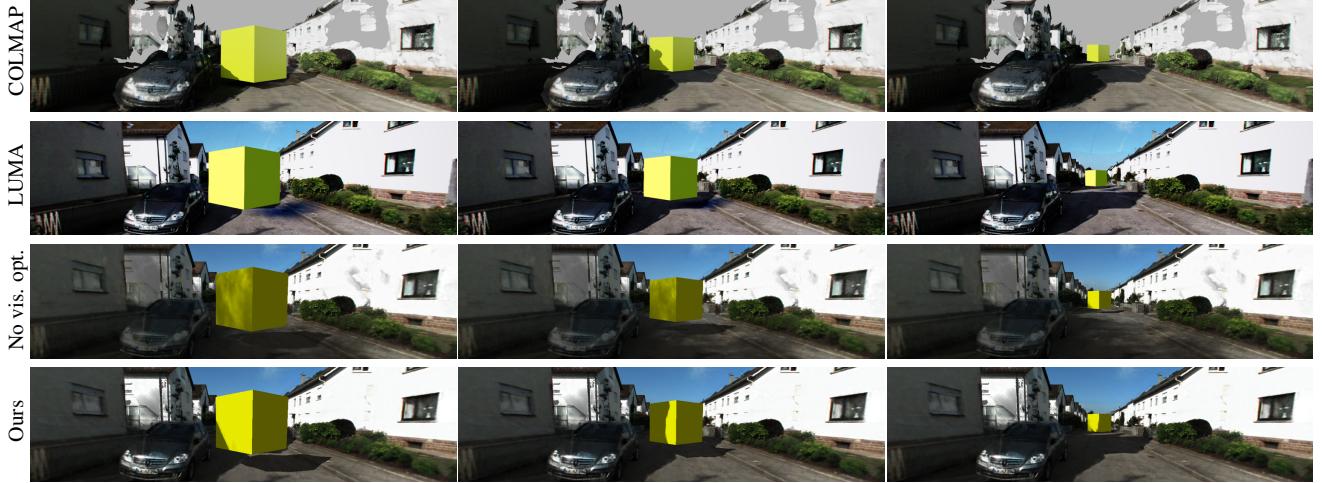
Figure 8: **Dynamic Object Insertion with Shadow Volume.** Our method produces accurate estimates of shadow volumes where others cannot. This is exposed by inserting a simple object into the scene, then looking at shadows cast onto that object. Visibility optimization makes an important contribution to accuracy. (A) COLMAP dense reconstruction [53, 54] + Blender [9] (B) Luma Unreal Engine Plugin [13]. (C) Ours without visibility optimization.
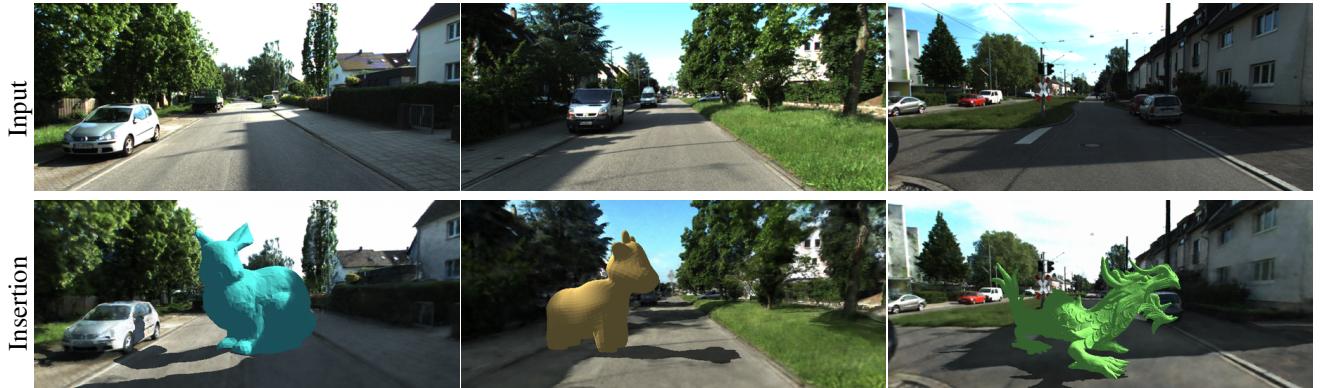

Figure 9: **Diverse Virtual Object Insertion.** Accurate shadow volume produces good looking object insertions for complex CGI objects.

4.7. Limitations

While UrbanIR provides high quality intrinsic decomposition, relighting, and insertion, it relies on multiple 2D priors during optimization. On occasion, shadow patterns cannot be removed completely in the albedo field, and appear in images. Visibility optimization refines only the geometry along the light direction in training so that large changes in sun direction can lead to poor shadows when poor geometry estimates cast shadows. Currently, we assume that light direction is known, we leave joint optimization of more complex light models and more general material parameters for future work.

5. Conclusion

We have introduced UrbanIR (Urban Scene Inverse Rendering), a novel scene model that enables realistic renderings of a scene from various viewpoints under new lighting conditions, using video as a basis. This model jointly determines shape, albedo, visibility, as well as sun and sky illumination from a single video of unbounded outdoor scenes with undefined lighting. Key innovation includes a unique visibility loss function, facilitating highly accurate shadow volume estimates within the original scene. Consequently, this allows for precise editing control, ultimately providing photorealistic renderings of relit scenes and seamlessly inserted objects from any viewpoint.
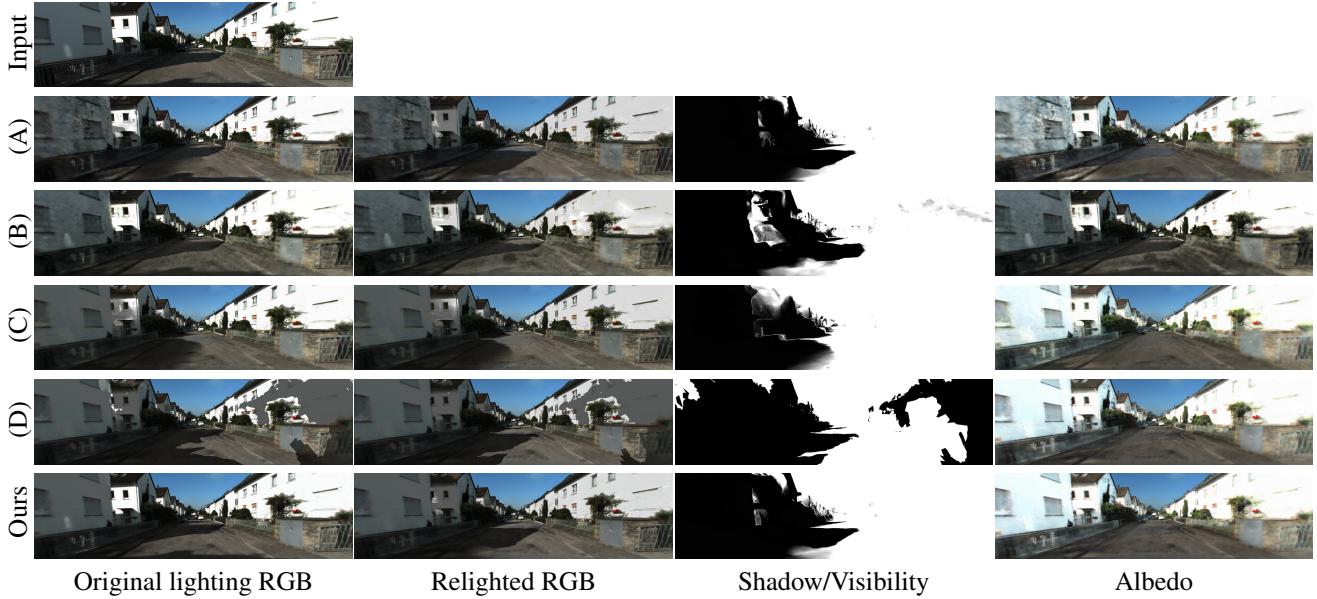
Figure 10: Ablation Study (A) **No deshadowed RGB guidance.** The albedo under shadow region cannot be reconstructed correctly. (B) **No intermediate visibility MLP.** Intrinsic decomposition cannot be conducted successfully, leaving shadow patterns in the albedo. (C) **No visibility optimization.** Geometry of unobserved region is not refined and model cannot cast sharp shadows according to novel lighting. (D) **Mesh-based visibility.** Severe artifacts are generated due to noisy mesh. Our complete model not only decompose clean albedo from input images, but also synthesize sharp shadows moving with light directions. Details in Section 4.6.

References

[1] Jonathan T Barron and Jitendra Malik. Shape, illumination, and reflectance from shading. *IEEE TPAMI*, 2014. 2

[2] H.G. Barrow and Joan M. Tenenbaum. Recovering intrinsic scene characteristics from images. *Computer Vision Systems*, 1978. 2

[3] Sean Bell, Kavita Bala, and Noah Snavely. Intrinsic images in the wild. In *ACM Transactions on Graphics (SIGGRAPH)*, 2014. 2

[4] James F Blinn. Models of light reflection for computer synthesized pictures. In *SIGGRAPH*, 1977. 4

[5] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T. Barron, Ce Liu, and Hendrik P.A. Lensch. Nerd: Neural reflectance decomposition from image collections. In *ICCV*, 2021. 2

[6] Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan T. Barron, and Hendrik P.A. Lensch. Neural-PIL: Neural Pre-Integrated Lighting for Reflectance Decomposition. 2021. 2

[7] Wenzheng Chen, Huan Ling, Jun Gao, Edward Smith, Jaakko Lehtinen, Alec Jacobson, and Sanja Fidler. Learning to predict 3d objects with an interpolation-based differentiable renderer. *NeurIPS*, 2019. 2

[8] Zhihao Chen, Lei Zhu, Liang Wan, Song Wang, Wei Feng, and Pheng-Ann Heng. A multi-task mean teacher for semi-supervised shadow detection. In *CVPR*, 2020. 5, 7

[9] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 6, 7, 9

[10] Dawson-Haggerty et al. trimesh. 5

[11] Yue Dong, Guojun Chen, Pieter Peers, Jiawan Zhang, and Xin Tong. Appearance-from-motion: Recovering spatially varying surface reflectance under unknown lighting. *TOG*, 2014. 2

[12] Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *ICCV*, 2021. 5, 7

[13] Epic Games. Unreal engine. 6, 7, 9

[14] David Forsyth and Jason J Rock. Intrinsic image decomposition using paradigms. *IEEE TPAMI*, 2021. 2

[15] Roger Grosse, Micah K Johnson, Edward H Adelson, and William T Freeman. Ground truth dataset and baseline evaluations for intrinsic image algorithms. In *ICCV*, 2009. 2

[16] Lanqing Guo, Siyu Huang, Ding Liu, Hao Cheng, and Bihan Wen. Shadowformer: Global context helps image shadow removal. *AAAI*, 2023. 2, 5, 7, 8

[17] Lanqing Guo, Chong Wang, Wenhan Yang, Siyu Huang, Yufei Wang, Hanspeter Pfister, and Bihan Wen. Shadowdiffusion: When degradation prior meets diffusion model for shadow removal. *in arXiv*, 2022. 2

[18] Jon Hasselgren, Nikolai Hofmann, and Jacob Munkberg. Shape, Light, and Material Decomposition from Images using Monte Carlo Rendering and Denoising. *in arXiv*, 2022. 2

[19] Daniel Hauagge, Scott Wehrwein, Kavita Bala, and Noah Snavely. Photometric ambient occlusion. In *CVPR*, 2013. 2

[20] Berthold KP Horn. Determining lightness from an image. *Computer graphics and image processing*, 1974. 2

[21] Berthold KP Horn. Obtaining shape from shading information. *The psychology of computer vision*, 1975. 2

[22] Xin Jin, Pengyue Deng, Xinxin Li, Kejun Zhang, Xiaodong Li, Quan Zhou, Shujiang Xie, and Xi Fang. Sun-sky model estimation from outdoor images. *Journal of Ambient Intelligence and Humanized Computing*, 2020. 3

[23] James T Kajiya and Brian P Von Herzen. Ray tracing volume densities. *ACM SIGGRAPH computer graphics*, 1984. 4

[24] Oğuzhan Fatih Kar, Teresa Yeo, Andrei Atanov, and Amir Zamir. 3d common corruptions and data augmentation. In *CVPR*, 2022. 5, 7

[25] Pierre-Yves Laffont, Adrien Bousseau, and George Drettakis. Rich intrinsic image decomposition of outdoor scenes from multiple views. *IEEE transactions on visualization and computer graphics*, 2012. 2

[26] Samuli Laine, Timo Aila, Ulf Assarsson, Jaakko Lehtinen, and Tomas Akenine-Möller. Soft shadow volumes for ray tracing. In *SIGGRAPH*. 2005. 2

[27] Edwin H Land and John J McCann. Lightness and retinex theory. *Josa*, 1971. 2

[28] Hendrik PA Lensch, Jan Kautz, Michael Goesele, Wolfgang Heidrich, and Hans-Peter Seidel. Image-based reconstruction of spatial appearance and geometric detail. *TOG*, 2003. 2

[29] Zhengqin Li, Mohammad Shafiei, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and svbrdf from a single image. In *CVPR*, 2020. 2

[30] Zhengqin Li, Jia Shi, Sai Bi, Rui Zhu, Kalyan Sunkavalli, Miloš Hašan, Zexiang Xu, Ravi Ramamoorthi, and Manmohan Chandraker. Physically-based editing of indoor scene lighting from a single image. *ECCV*, 2022. 2

[31] Zhengqin Li, Kalyan Sunkavalli, and Manmohan Chandraker. Materials for masses: SVBRDF acquisition with a single mobile phone image. In *ECCV*, 2018. 2

[32] Zhengqin Li, Zexiang Xu, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. Learning to reconstruct shape and spatially-varying reflectance from a single image. *ACM TOG*, 2018. 2

[33] Yiyi Liao, Jun Xie, and Andreas Geiger. KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *in arXiv*, 2021. 5, 7

[34] Daniel Lichy, Jiaye Wu, Soumyadip Sengupta, and David W Jacobs. Shape and material capture at home. In *CVPR*, pages 6123–6133, 2021. 2

[35] Daquan Liu, Chengjiang Long, Hongpan Zhang, Hanning Yu, Xinzhi Dong, and Chunxia Xiao. Arshadowgan: Shadow generative adversarial network for augmented reality in single light scenes. *CVPR*, 2020. 2

[36] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *ICCV*, 2019. 2

[37] Stephen Lombardi and Ko Nishino. Reflectance and illumination recovery in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2015. 2

[38] Stephen Lombardi and Ko Nishino. Radiometric scene decomposition: Scene reflectance, illumination, and geometry from rgb-d images. In *3DV*, 2016. 2

[39] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM TOG*, 2019. 2

[40] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 1987. 7

[41] Luma Labs. Luma Unreal Engine Plugin. 7

[42] Wei-Chiu Ma, Hang Chu, Bolei Zhou, Raquel Urtasun, and Antonio Torralba. Single image intrinsic decomposition without a single intrinsic image. In *ECCV*, 2018. 2

[43] Stephen Robert Marschner. *Inverse rendering for computer graphics*. Cornell University, 1998. 2

[44] 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*, 2020. 2, 4

[45] Thomas Muller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM TOG*, 2022. 3

[46] Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. Extracting Triangular 3D Models, Materials, and Lighting From Images. *arXiv:2111.12503 [cs]*, Dec. 2021. 2

[47] Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. Extracting triangular 3d models, materials, and lighting from images. In *CVPR*, 2022. 2

[48] Chen Quei-An. ngp_pl: a pytorch-lightning implementation of instant-ngp, 2022. 3

[49] Viktor Rudnev, Mohamed Elgharib, William Smith, Lingjie Liu, Vladislav Golyanik, and Christian Theobalt. Nerf for outdoor scene relighting. In *European Conference on Computer Vision (ECCV)*, 2022. 2, 3, 7, 8

[50] Viktor Rudnev, Mohamed Elgharib, William Smith, Lingjie Liu, Vladislav Golyanik, and Christian Theobalt. Neural radiance fields for outdoor scene relighting. *ECCV*, 2022. 6, 8

[51] Imari Sato, Yoichi Sato, and Katsushi Ikeuchi. Illumination from shadows. *IEEE TPAMI*, 2003. 2

[52] Yoichi Sato, Mark D Wheeler, and Katsushi Ikeuchi. Object shape and reflectance modeling from observation. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, 1997. 2

[53] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 7, 9

[54] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. 6, 7, 9

[55] Soumyadip Sengupta, Jinwei Gu, Kihwan Kim, Guilin Liu, David W Jacobs, and Jan Kautz. Neural inverse rendering of an indoor scene from a single image. In *ICCV*, 2019. 2

[56] Pratul P Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *CVPR*, 2021. 2

[57] Jon Story. Hybrid ray traced shadows. In *Game Developer Conference*, 2015. 2

[58] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. *CVPR*, 2022. 5

[59] Jin Wan, Hui Yin, Zhenyao Wu, Xinyi Wu, Yanting Liu, and Song Wang. Style-guided shadow removal. In *ECCV*. Springer, 2022. 2

[60] Yifan Wang, Andrew Liu, Richard Tucker, Jiajun Wu, Brian L Curless, Steven M Seitz, and Noah Snavely. Repopulating street scenes. In *CVPR*, 2021. 2

[61] Zian Wang, Tianchang Shen, Jun Gao, Shengyu Huang, Jacob Munkberg, Jon Hasselgren, Zan Gojcic, Wenzheng Chen, and Sanja Fidler. Neural fields meet explicit geometric representation for inverse rendering of urban scenes. *arXiv*, 2023. 2

[62] Zian Wang, Tianchang Shen, Jun Gao, Shengyu Huang, Jacob Munkberg, Jon Hasselgren, Zan Gojcic, Wenzheng Chen, and Sanja Fidler. Neural fields meet explicit geometric representations for inverse rendering of urban scenes. In *CVPR*, 2023. 3, 7

[63] Tai-Pang Wu, Chi-Keung Tang, Michael S Brown, and Heung-Yeung Shum. Natural shadow matting. *ACM TOG*, 2007. 2

[64] Wenqi Yang, Guanying Chen, Chaofeng Chen, Zhenfang Chen, and Kwan-Yee K Wong. S ^{3}-nerf: Neural reflectance field from shading and shadow under a single viewpoint. *in arXiv*, 2022. 2

[65] Yizhou Yu, Paul Debevec, Jitendra Malik, and Tim Hawkins. Inverse global illumination: Recovering reflectance models of real scenes from photographs. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, 1999. 2

[66] Ye Yu, Abhimitra Meka, Mohamed Elgharib, Hans-Peter Seidel, Christian Theobalt, and William A. P. Smith. Self-supervised outdoor scene relighting. In *ECCV*, 2020. 8

[67] Ye Yu and William AP Smith. Inverserendernet: Learning single image inverse rendering. In *CVPR*, 2019. 2

[68] Kai Zhang, Fujun Luan, Zhengqi Li, and Noah Snavely. Iron: Inverse rendering by optimizing neural sdfs and materials from photometric images. In *CVPR*, 2022. 2

[69] Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah Snavely. Physg: Inverse rendering with spherical gaussians for physics-based material editing and relighting. In *CVPR*, 2021. 2

[70] Ruo Zhang, Ping-Sing Tsai, James Edwin Cryer, and Mubarak Shah. Shape from shading: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 1999. 2

[71] Shuyang Zhang, Runze Liang, and Miao Wang. Shadowgan: Shadow synthesis for virtual objects with conditional adversarial networks. *Computational Visual Media*, 2019. 2

[72] Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM TOG*, 2021. 2, 4

[73] Yuxuan Zhang, Wenzheng Chen, Huan Ling, Jun Gao, Yinan Zhang, Antonio Torralba, and Sanja Fidler. Image gans meet differentiable rendering for inverse graphics and interpretable 3d neural rendering. *arXiv*, 2020. 2

[74] Yuanqing Zhang, Jiaming Sun, Xingyi He, Huan Fu, Rongfei Jia, and Xiaowei Zhou. Modeling indirect illumination for inverse rendering. In *CVPR*, 2022. 2, 4

[75] Tinghui Zhou, Philipp Krahenbuhl, and Alexei A Efros. Learning data-driven reflectance priors for intrinsic image decomposition. In *ICCV*, 2015. 2