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

Supplementary Material

Chih-Hao Lin[1] Bohan Liu[1] Yi-Ting Chen[2] Kuan-Sheng Chen[1]
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
https://urbaninverserendering.github.io/

A. More Qualitative Results

We compare the relighting quality with FEGR [20] in Fig. 1. FEGR [20] first extracts mesh and estimates the shading from the lighting configuration, and the imperfect mesh geometry produces artifacts and loses appearance details. On the other hand, our method alleviates the original shadow and produces relighting images while preserving appearance details. We show additional night simulation results on various Kitti360 [13] sequences in Fig. 2, demonstrating the generalization capability of UrbanIR . The Instruct-Pix2Pix [3] leverages the large language model [4] and stable diffusion [17] for abundant image editing tasks. However, such a data-driven method cannot move the daylight shading and shadow in the input images. On the contrary, UrbanIR decomposes shadow-free albedo and performs physically-based rendering with new light sources (e.g., streetlights, headlights), significantly enhancing the visual quality of night simulation. The strong specular reflection is also simulated on the car region, boosting the realism of metal material. Please note that the simulation is flexible, and the user can adjust physical parameters (e.g., light color, light strength) to create various effects. Please refer to our supplementary videos to better visualize view consistency and controllable simulation.

B. Model Architecture

Instant-NGP [15] encodes the scene with a multi-scale hash table, and each entry contains learnable parameters. For point $\mathbf{x} \in \mathbb{R}^3$, the model retrieves and interpolates the parameters with hash function: $F(\mathbf{x}, \theta)$. UrbanIR adopts the hash encoding from [15] and maintain two separate hash tables for geometry and appearance, and predict the scene properties with:

$$\begin{aligned} \sigma &= F_g(\mathbf{x}, \theta_g) \\ (\mathbf{a}, \mathbf{n}, s) &= F_a(\mathbf{x}, \theta_a), \end{aligned} \tag{1}$$

where σ is density, $(\mathbf{a}, \mathbf{n}, s)$ are albedo, surface normal, and semantic. θ_g, θ_a are learnable parameters for geometry and appearance. Please note that the density field σ is not only involved in the volume rendering (Eq. **??**), but also involved in visibility estimation (Eq. **??**) and normal loss calculation. The hash encoding is implemented with tiny-cuda-nn [14]. We empirically find that maintaining separate learnable parameters for geometry and appearance leads to more stable convergence and higher rendering quality.

C. Training Details

The training procedure is illustrated in Fig. 3. We leverage pretrained networks as 2D priors during training to address the ill-posed inverse problem. Specifically, the shadow mask is estimated with MTMT [5]. Omnidata normal estimation [8] helps refine scene geometry, which is critical in the shading quality and albedo decomposition. A semantic map is provided in Kitti360 dataset [13] and can also be estimated with MMSegmentation [6] if such information is not provided. The objective function of the optimization is:

$$\min_{\theta, \mathbf{L}} \mathcal{L}_{\text{render}} + \lambda_1 \mathcal{L}_{\text{visibility}} + \lambda_2 \mathcal{L}_{\text{normal}} + \lambda_3 \mathcal{L}_{\text{semantics}} + \lambda_4 \mathcal{L}_{\text{reg}},$$

where $\lambda_1 = 0.001, \lambda_2 = 0.01, \lambda_3 = 0.04, \lambda_4 = 0.1$. We use Adam optimizer [11] with a learning rate of 0.002 for a total of 100 epochs during the optimization.

D. Application Details

We provide the implementation of relighting and object insertion as follows:
Simulating night-time proceeds by defining headlights and street lights, then illuminating with scene model considering specularity and lens flare. For sky regions $\mathbf{S}(\mathbf{r}) \in$ sky, we use $\mathbf{C}(\mathbf{r}) = \mathbf{L}_{\text{sky}}(\mathbf{r})$ and otherwise, we use

$$\mathbf{A}(\mathbf{r}) \left(\sum \mathbf{L}^{\text{i}}_{\text{dif}} \mathbf{D}_i \mathbf{V}_i + \mathbf{L}_{\text{amb}} \right) + \sum_i \mathbf{L}^i_{\text{spec}} \tag{2}$$

The spotlight we used is given by the center $\mathbf{o}^i_L \in \mathbb{R}^3$ and direction $\mathbf{d}^i_L \in \mathbb{R}^3$ of the light. This spotlight produces a

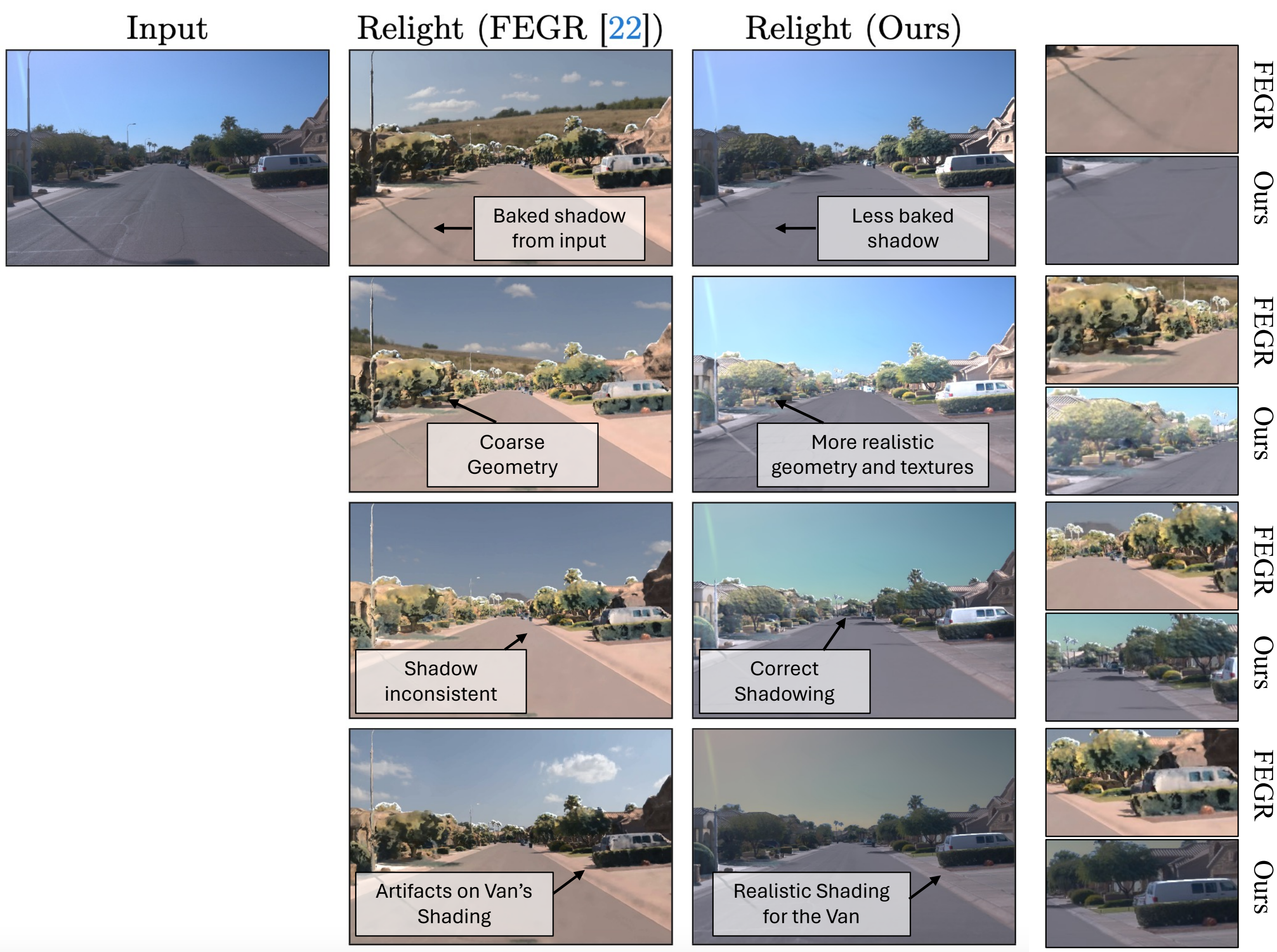

Figure 1. **Relighting Comparison on Waymo Open Dataset [19]. The second and third columns compare the relighting quality. The authors provide the FEGR results and we match the lighting condition according to the shadow direction.**

diffuse radiance at $\mathbf{r}$ given by

$$\mathbf{L}^i_{\text{dif}}(\mathbf{r}) = \frac{1}{\|\mathbf{o}^i_L - \mathbf{x}(\mathbf{r})\|^2} \left(l \cdot \mathbf{d}^i_L\right)^k, l = \frac{\mathbf{o}^i_L - \mathbf{x}(\mathbf{r})}{\|\mathbf{o}^i_L - \mathbf{x}(\mathbf{r})\|}, \tag{3}$$

Spotlight's diffuse color intensity is brightest on the central ray $\mathbf{r}(t) = \mathbf{o}_L - t\mathbf{d}_L$, decays with distance from ray $\mathbf{r}(t)$ and angle. We modulate it with constant k.

The realistic night-time simulation requires reproducing the strong specular effects on cars. We find car regions using a semantic field $\mathbf{S}$ in Eq. **??**, then simulate specular reflection with the Blinn-Phong model [2], where the γ (specular strength) parameter is inherited from the semantic field.

At night, luminaires often display lens flares. A pure simulation of lens flares is impractical, as it requires extensive ray tracing through the lens. We use the standard image-based approximation [1] to simulate such light scattering effects. For directly visible luminaires, we composite a real-world lens flare image from a similar lighting source into the image, using location and depth. As Fig. **??**, **??** in the main paper show, this simple method is effective.

Object insertion proceeds by a hybrid rendering strategy. We first cast rays from the camera and estimate ray-mesh intersections [7] for the inserted object. If the ray hits the mesh and the distance is shorter than the 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. **??**), 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 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. Similar approaches have been depicted in recent works [12, 16]. However, ours is the first to be visibility-aware, enabling us to render effects when an object enters into a shadow.

Outdoor relighting is done by simply adjusting lighting pa-

Figure 2. **Nighttime rendering.** The scene is transformed from daytime (1st row) to night-time (3rd row) by introducing new light sources: a headlight on a car and a street lamp. Top 3 and bottom 3 rows are from same driving sequence with different time stamp. Comparing with data-driven generative model and Instruct-Pix2Pix [3], the dark shadows with sharp boundaries are successfully removed with our decomposition, resulting more realistic rendering with new light sources (e.g. streetlights, headlight) during the nighttime simulation.

rameters (position or color of the sun; sky color) then re-rendering using Eq. **??** in the main paper. We also use semantics to interpret specular car surfaces and emulate their reflectance during the simulation.

E. Baseline Details

Description of the approach of baselines we compared to.

Instruct-Pix2Pix [3] edits images according to user instruction. The model leverages large language model GPT-3 [4] and Stable Diffusion [17] for generating image and instruction pairs and fine-tune diffusion model to perform editing. We use instructions "change to night", and "It's now midnight" for night image generation.

Instruct-NeRF2NeRF [10] aims to edit NeRF scenes with text instructions. It uses a generative image editing model [3] to iteratively edit input images while optimizing the underlying scene model, resulting in an optimized 3D scene that respects the instruction. We compare Instruct NeRF2NeRF in night simulation, where we provide the instruction, "*Make it look like it was taken at night*."

NeRF-OSR [18] is a recent work for outdoor scene reconstruction and relighting. We use the open-source project provided by the author to run this baseline. This method

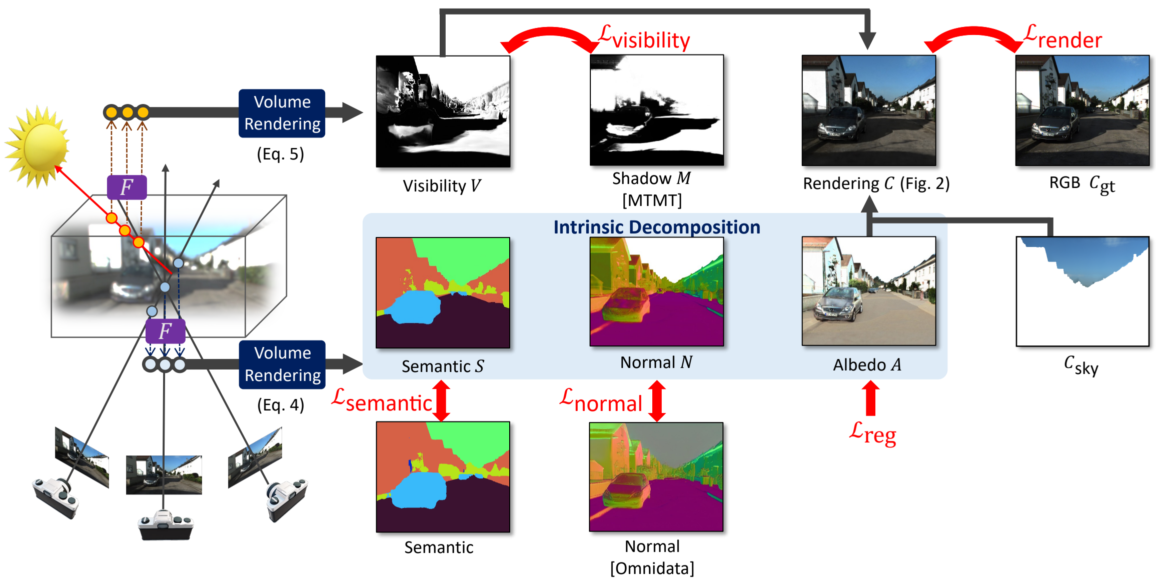

Figure 3. **Training Pipeline.** UrbanIR retrieves scene intrinsics with volume rendering from camera rays, which is guided by semantic and normal priors. Transmittance along tracing rays is supervised with shadow masks.

represents lighting as spherical harmonics parameters. It is worth noting that NeRF-OSR was designed for inverse rendering in *multi-illumination conditions*. For a fair comparison, we rotate the spherical vectors to simulate different light conditions.

RelightNet [21] is a single-image based relighting framework. We use the open-source project provided by the authors to produce intrinsic decomposition results, including shading and albedo for comparison.

ShadowFormer [9] performs single-image shadow removal task. It leverages the transformer architecture and takes the original image and shadow masks as input. In Fig. **??** in the main paper, we first estimate the shadow mask with MTMT [5], and use the open-source project and pretrained weights provided by the authors to estimate the base color of an image.

References

[1] Tomas Akenine-Moller, Eric Haines, and Naty Hoffman. *Real-time rendering*. AK Peters/crc Press, 2019. 2

[2] James F Blinn. Models of light reflection for computer synthesized pictures. In *Proceedings of the 4th annual conference on Computer graphics and interactive techniques*, pages 192–198, 1977. 2

[3] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023. 1, 3

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *NeurIPS*, 2020. 1, 3

[5] 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. 1, 4

[6] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. `https://github.com/open-mmlab/mmsegmentation`, 2020. 1

[7] Dawson-Haggerty et al. trimesh, 2019. 2

[8] 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. 1

[9] Lanqing Guo, Siyu Huang, Ding Liu, Hao Cheng, and Bihan Wen. Shadowformer: Global context helps image shadow removal. *AAAI*, 2023. 4

[10] Ayaan Haque, Matthew Tancik, Alexei Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. 3

[11] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2014. 1

[12] Yuan Li, Zhi-Hao Lin, David Forsyth, Jia-Bin Huang, and Shenlong Wang. Climatenerf: Extreme weather synthesis in neural radiance field. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3227–3238, 2023. 2

[13] 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. 1

[14] Thomas Müller. tiny-cuda-nn, 2021. 1

[15] Thomas Müller, Alex Evans, Christoph Schied, and Alexan-

der Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM TOG*, 2022. 1

[16] Yi-Ling Qiao, Alexander Gao, Yiran Xu, Yue Feng, Jia-Bin Huang, and Ming C Lin. Dynamic mesh-aware radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 385–396, 2023. 2

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

[18] 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. 3

[19] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, 2020. 2

[20] 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. 1

[21] Ye Yu, Abhimitra Meka, Mohamed Elgharib, Hans-Peter Seidel, Christian Theobalt, and William A. P. Smith. Self-supervised outdoor scene relighting. In *ECCV*, 2020. 4