IRON: A Neural Inverse Rendering Method

Jiaping Zhang

Robitics, Cognition and Intelligence - Technische Universität München

Abstract

"IRON: Inverse Rendering by Optimizing Neural SDFs and Materials from Photometric Images" was published in 2022 by Zhang, Kai, et al. It is a novel inverse rendering pipeline that applies hybrid optimization of neural fields to obtain meshes and material textures from a set of photometric images. The optimization stages optimize a thick surface via a volumetric radiance field and refine the surface by edge-aware physical-based surface rendering. The second step of optimization proposes a new edge sampling algorithm for SDFs according to mesh-based differentiable rendering. Compared to its baseline (DRV), this method could obtain a more precise shape and appearance.

1 Introduction

Reconstruction of real-world objects and scenes is a prevailing topic that can be applied in various scenarios, such as autonomous driving and movie scene construction. Inverse Rendering is a powerful way to obtain 3D objects from 2D images. One method for inverse rendering is fully differentiable Monte Carlo path tracing methods [2], which focus on realistic functions and equations. The challenge of this method lies in computing the edge derivatives. For mesh-based differentiable rendering, shape optimization is also difficult to process because of changing topology and avoiding self-intersection; for signed distance field representation, recent studies only consider interior derivatives and ignore edge derivatives [8]. The other inverse rendering direction concentrates on neural representations for the radiance field. The limitation of this method reflects in the difficulties of disentangling material from lightning. Furthermore, The details of the shape and appearance in NeRF could not be fully presented since volume rendering is rough based on volume density [4]. IDR utilizes surface rendering, which could obtain better performance but constraints application scenarios [8].

For this sake, IRON introduces a novel approach for inverse rendering with better performance. IRON makes use of neural representations for geometry and materials. A two-step optimization scheme, namely volumetric radiance field and edge-aware physical-based surface rendering, could recover an object with more accurate geometry and sharper texture details. The whole process starts from a set of photometric photos, transfers the data into neural representations for optimization, and finally obtains the object in the form of mesh with textures (The pipeline is shown in Figure 1). The improvement of this method is that it disentangles materials with lightning, and based on the new edge sampling method, it could recover the 3D object more precisely [9].

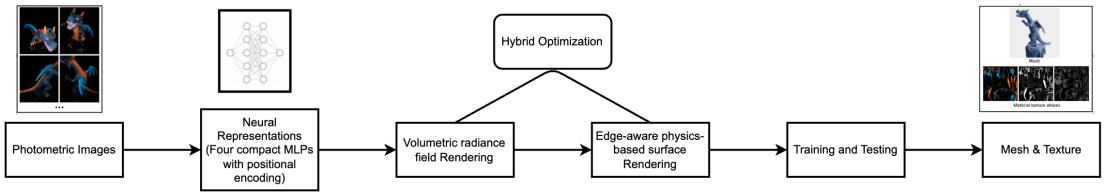

Figure 1: Pipeline of IRON

2 Method description

The proposed method works for opaque objects without ambient light and ignores shadows for efficiency. Under the conditions mentioned above, the IRON system could optimize the neural shape and material representations through two stages when provided with input photometric images. The first stage is the volumetric radiance field, which aims to recover the correct object topology and serves as an initialization for the second phase. The second stage is trying to refine geometric details and factorize materials from lighting. Four compact MLPs are Neural SDF $S_{\Theta_S} : x \rightarrow (S, f)$, Neural diffuse albedo $\beta_{\Theta_\beta} : (x, n, n, f) \rightarrow \beta$, Neural specular albedo $\kappa_{\Theta_\kappa} : (x, n, f) \rightarrow \kappa$, Neural roughness $\alpha_{\Theta_\alpha} : (x, n, f) \rightarrow \alpha$, where x, n, f are 3D location, surface normal and feature descriptor.

2.1 Volumetric radiance field rendering

The volumetric radiance field optimizes neural SDF (S_{Θ_S}) and diffuse albedo (β_{Θ_β}) to rebuild the geometry. This phase makes use of the method declared by the former research NeuS. Compared to the surface rendering method IDR and volume rendering method NeRF, NeuS is effective in complex geometries and self-occlusion. Surface rendering IDR can hardly process abrupt depth change since it considers only a single surface intersection of each ray. On the other hand, volume rendering NeRF can handle sudden depth changes, but the reconstruction results are noisy. NeuS uses a novel volume rendering to learn a neural SDF representation, which could get a more accurate and robust surface representation. The comparison of these three methods is shown in figure 2 [7].

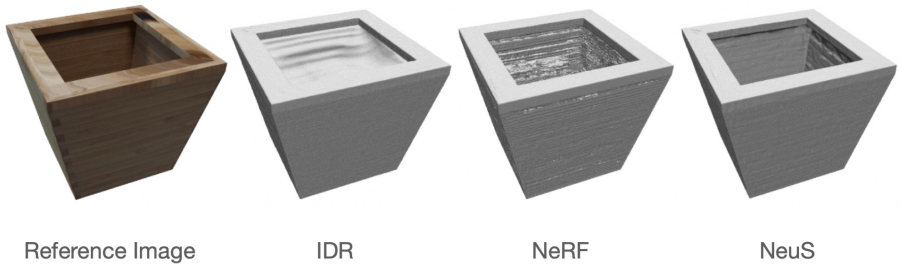

Figure 2: Reconstruction Performance Comparison of IDR, NeRF and NeuS [7]

The NeuS could train the neural representations by 2D supervision. Given a pixel, the corresponding ray emitted from the pixel is defined as $\{p(t) = o + tv | t \geq 0\}$,

and the color of the pixel is expressed as an integral:

$$C(o,v) = \int_0^{+\infty} w(t)c(p(t),v)dt \tag{1}$$

o, v, $w(t)$ are camera center, unit direction vector of the ray, and weight of point $p(t)$ in the observation direction v. And $w(t)$ should fulfill two requirements, unbiased and occlusion-aware, namely the point near the surface and the camera should have a greater weight to the final result. The weight function in standard volume rendering formulation is denoted as the equation below.

$$w(t) = T(t)\sigma(t),\ where\ T(t) = exp(-\int_0^t \sigma(u)du) \tag{2}$$

$\sigma(t)$, $T(t)$ are volume density and accumulated transmittance. The surface of the object S is represented by a zero-level set of SDF.

$$S = x \in \mathbb{R}^3 | f(x) = 0 \tag{3}$$

Moreover, NeuS introduces a probability density function $\phi_s(f(x))$ (S-density).

$$\phi_s(x) = \frac{se^{-sx}}{(1+e^{-sx})^2} \tag{4}$$

An opaque density function $\rho(t)$ is used instead of $\sigma(t)$ to obtain an unbiased and occlusion-aware weight.

$$\rho(t) = max(\frac{\frac{-d\Phi_s}{dt}(f(p(t))}{\Phi_s(f(p(t))}, 0) \tag{5}$$

The final step of the volume rendering model is to do discretization. Along the ray, The system samples n points $\{p_i = o + t_i v | i = 1, ..., n, t_i < t_{i+1}\}$, computes the approximate pixel color of the ray.

$$\hat{C} = \sum_{i=1}^{n} T_i \alpha_i c_i \tag{6}$$

T_i, α_i are discrete accumulated transmittance and discrete opacity value, which can be denoted as the following equation:

$$T_i = \prod_{j=1}^{i-1}(1-\alpha_j) \tag{7}$$

$$\alpha_i = 1 - exp(-\int_{t_i}^{t_{i+1}} \rho(t)dt) \tag{8}$$

After training, which minimizes the difference between the rendered colors and ground truth colors, parameters of neural SDF (S_{Θ_S}) and colors ($\beta_{\Theta_\beta}(x, n, -d, f)$)

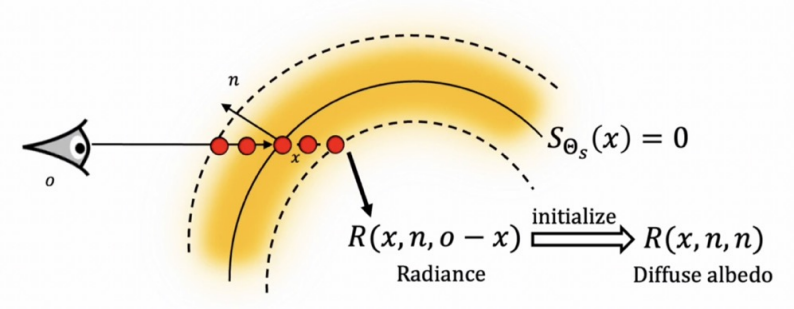

Figure 3: Volumetric rendering [9]

are obtained. To explain the process used in IRON visually, as shown in figure 3, it implements volume rendering by passing the output of the neural SDF through a flipped sigmoid light function with a notable width to obtain density value at each location. Then it samples multiple points along each ray to perform volume rendering. The color at each point is computed using a radiance MLP that predicts a view-dependent color. Once this volume rendering or position is complete, it uses these radiance MLP to initialize the diffuse albedo MLP (by recovering the second n).

This stage is also recognized as the initialization of the second optimization phase. Michael Oechsle et al. observed that if they do not implement volumetric rendering and directly optimize surface rendering, the optimization requires object segmentation masks. It will easily get stuck in local minima with incorrect topology since existing surface rendering methods can only reason about rays that intersect a surface [6].

2.2 Edge-aware physics-based surface rendering

The second step, optimization Edge-aware physics-based surface rendering, jointly optimizes neural SDF, neural materials, and light intensity. The rendering involves finding ray surface intersections and evaluating the rendering equation at each surface point(figure 4), including two critical components: differentiable physics-based shading and edge-aware surface rendering.

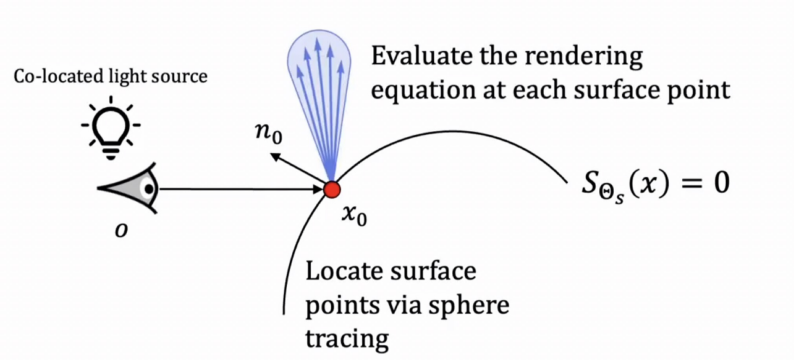

Figure 4: Surface rendering [9]

During the differentiable physics-based shading phase, we could compute the simplified rendering equation as follows because the inputs are photometric images.

$$L_o(w_o, x) \approx L_i(w_o, x) f_r(w_o, w_o, x)(w_o \cdot n), \tag{9}$$

where L_o, x, n, $_o$, L_i, f_r are observed light, surface location, surface normal, view(light) direction, incident light and BRDF. Besides, we could model the flash-light as a point light source as below.

$$L_i(w_o; x) = \frac{L}{\|x - o\|_2^2}, \tag{10}$$

where L and o are scalar light intensity and light(camera) location. Through Eqns. 9 and 10, it's obvious that the gradient of rendered image $L_o(w_o, x)$ must back-propagate to the shape and material parameters through x, n and f_r.

For previous research, IDR and DRV fail to consider the gradient of the image reconstruction loss concerning the geometrical parameters and visibility discontinuous. The differentiable rendering module only works for interior pixels, but geometric discontinuities are introduced by edge pixels, where multiple depth values exist in a single pixel [5]. IRON introduces a novel edge sampling algorithm for neural SDF to fix this problem, including three steps:

- The first step is to detect edge points in 3D by walking on the zero-level set of the neural SDF, and only consider the point where the ray-surface intersections at depth discontinuity pixels for efficiency, and then project 3D edge points into 2D for subpixel edge localization (as shown in figure 5). The walking direction is defined as $x_t \leftarrow x_t + \epsilon \cdot (n_t - \frac{o - x_t}{(o - x_t)^T n_t})$

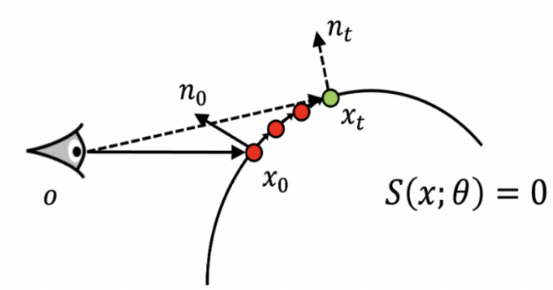

Figure 5: Edge point detection [9]

- After obtaining the edge points, the second step is to reparametrize edge points so that it is differentiable with respect to the network weights of neural SDF. For interior points, the differentiable ray-surface intersection equation is like Eqns. 11, and for edge points, the only difference is to replace fire direction $(o - x)$ by surface normal n, as shown in Eqns. 12.

$$x_{\Theta s} = x - \frac{o - x}{n^T (o - x)} S_{\Theta s}(x) \tag{11}$$

$$x_{\Theta s} = x - \frac{n}{n^T n} S_{\Theta s}(x) = x - n S_{\Theta s}(x) \tag{12}$$

- Finally, we need to compute the shading at each edge pixel. As shown in figure 6, the edge pixel needs to be rendered by blending colors on both sides of the edge. The blending weights are functions of the subpixel-accurate differentiable edge point. Furthermore, it approximates the square pixel footprint to simplify computation. The predicted color of the edge pixel should be like Eqns. 13.

$$C = w_A C_A + (1 - w_A) C_B, \tag{13}$$

  where C_A and C_B denote shaded colors (here purple and blue), and w_A represents weight of C_A, which is proportional to the segment areas separated by the edge. Therefore, edge pixel color C could back-propagate to the neural SDF and materials by C_A, C_B and w_A.

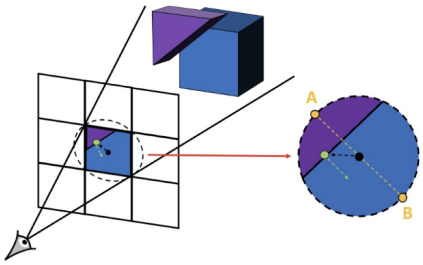

Figure 6: Edge pixel shading computing [9]

Volumetric radiance rendering and physical-based edge-aware surface rendering reconstruct the object with a more accurate and precise result. After training, it uses the marching cubes algorithm and UV unwrapping to transfer neural representations into meshes and textures.

3 Experiments and results

There are four main experiments have been implemented. Two of them show the superior performance of the IRON method compared to DRV [1] and PSDR [3]; The other two small experiments are done to verify the efficiency of the edge sampling algorithm and the loss function of the IRON method.

- For the first experiment, the goal is to address the critical effect of the proposed edge sampling algorithm. The input of this experiment is a single target image of an object with known color. As shown in figure 7, DRV failed to process this problem since it lacks edge pixel handling; PSDR could recover silhouettes with degraded quality because of fixed mesh topology; on the other hand, IRON could reconstruct the object perfectly.

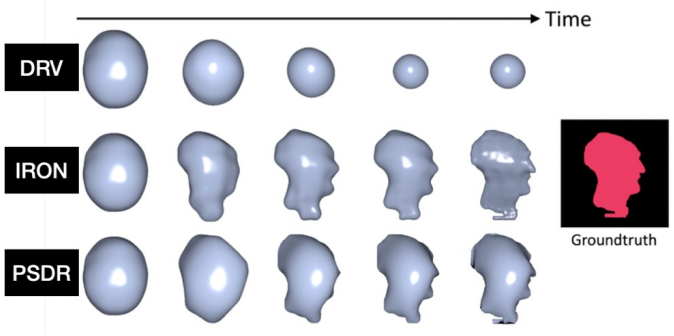

Figure 7: Optimize model to fit single image [9]

- The second experiment evaluates the performance of three methods on inverse rendering from multi-view photometric images on co-located flashlight synthetic Dataset, natural environment lightning synthetic Dataset, and real-world Dataset. The dataset and performance of each method are explained in figure 8. Moreover, part of the comparison results is shown in Figures 9 and 10.

Dataset	Synthetic Dataset		• **Training data:** Images are rendered from 200 randomly sampled viewpoints using the Mitsuba path-tracing renderer with co-locating point light source and camera without other light sources. • **Testing data:** 100 images under novel co-located flashlight illumination, and another 100 images under novel natural environmental illumination.
	Real Dataset		• **Training data:** 70 % of images, which are acquired using co-located flashlight setup in a dark environment • **Testing data:** 30% of the images, which are acquired using co-located flashlight setup in a dark environment
Performance	Volume-based DRV	Synthetic Dataset (co-located flashlight)	• Has slightly better LPIPS and PSNR scores, but it significantly blurs detail.
		Synthetic Dataset (natural environmental lighting)	• /
	Mesh-based PSDR	Synthetic Dataset (co-located flashlight)	• Produces geometry with incorrect topology sometimes • Face difficulties in maintaining mesh regularity and avoiding self-intersections during optimization • Require repeated application of error-prone remeshing and uv-mapping procedures.
		Synthetic Dataset (natural environmental lighting)	• Matches less for specular highlight regions on synthetic data
	IRON	Synthetic Dataset (co-located flashlight)	• Perform better on geometric accuracy and generalization
		Synthetic Dataset (natural environmental lighting)	• Have better synthesized specular highlights and more perceptually convincing relighting

Figure 8: Inverse rendering from photometric images [9] [1] [3]

From the view of optimization and deployment, IRON is easy for both optimization and deployment, while PSDR is only easy to deploy and DRV is only easy to optimize. However, IRON also has limitations, like no inter-reflections modeling, a more involved capture process that requires a dark room and assuming opaque objects with only diffuse and specular reflections, lacking in modeling translucent/transparent effects.

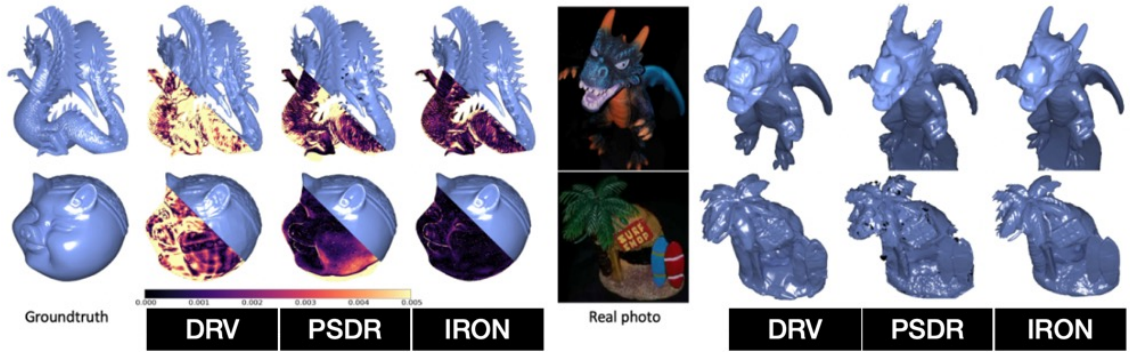

Figure 9: Comparison of recovered geometry on synthetic (left) and real data (right) with DRV and PSDR [9]

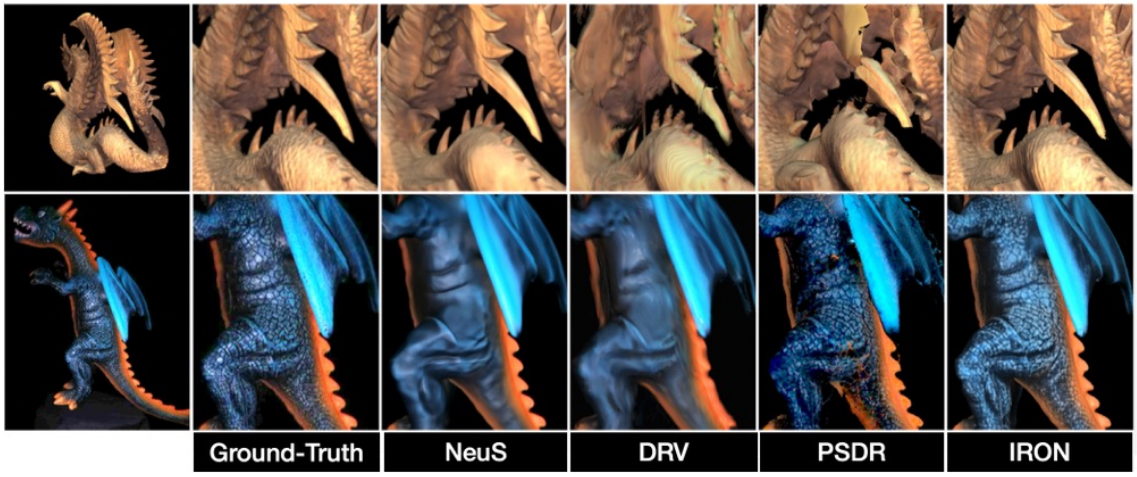

Figure 10: Qualitative comparison of generalization to novel co-located flashlight relighting using both synthetic (top row) and real (bottom row) data with NeuS, DRV and PSDR [9]

4 Discussion / Conclusion

From the descriptions and experiments, we could know that IRON can recover more accurate geometry and sharper texture details and can be easily re-lit under environmental lighting using an existing graphic renderer. Besides, it is also less prone to shape optimization artifacts. However, NeRF [4] and IDR [8] could hardly reach this performance since NeRF has insufficient surface constraints, and IDR requires foreground masks as supervision. Besides, IDR is easily trapped in local minima and struggles with the reconstruction of objects with severe self-occlusion or thin structures. Compared to NeuS [7], IRON introduces a novel edge sampling algorithm, which works better for refining the surface.

Neural fields benefit optimization, and Mesh and texture representation leads to easier deployment. Moreover, for neural fields, it is better to optimize in a hybrid way, where volumetric radiance rendering is for global optimization, and edge-aware surface rendering is for local optimization. It not only works to the advantage of better reconstruction performance but also makes full use of the features of neural

representation and is computationally effective.

It would be efficient if IRON could work not only for photometric images and also for the situation when the capture process is implemented in a natural environment because it is hard to create such an environment for every object.

References

[1] Sai Bi, Zexiang Xu, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Deep reflectance volumes: Relightable reconstructions from multi-view photometric images. In *European Conference on Computer Vision*, pages 294–311. Springer, 2020.

[2] Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. Differentiable monte carlo ray tracing through edge sampling. *ACM Transactions on Graphics (TOG)*, 37(6):1–11, 2018.

[3] Fujun Luan, Shuang Zhao, Kavita Bala, and Zhao Dong. Unified shape and svbrdf recovery using differentiable monte carlo rendering. In *Computer Graphics Forum*, volume 40, pages 101–113. Wiley Online Library, 2021.

[4] 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. *Communications of the ACM*, 65(1):99–106, 2021.

[5] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3504–3515, 2020.

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

[7] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *Advances in Neural Information Processing Systems*, 34:27171–27183, 2021.

[8] 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*, 33:2492–2502, 2020.

[9] Kai Zhang, Fujun Luan, Zhengqi Li, and Noah Snavely. Iron: Inverse rendering by optimizing neural sdfs and materials from photometric images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5565–5574, 2022.