Supplementary Material
ORCa: Glossy Objects as Radiance-Field Cameras

1. Method Details

1.1. General Shape Operator

General Implicit Curvature Estimation. To approximate the virtual pixel lying on the object-cone intersection surface, we find intersection points along rays that bound the cone and approximate the surface by finding intersection points with the surface and the rays. Ideally, we would query the sdf MLP for points along the bounding rays to get the intersection points with the surface, however, due to computing requirements we approximate the surface using the second-order derivative of the local geometry. We approximate this surface in Sec. 3.2 using mean curvature sampled around a point, t_i, on ray $r_p(t)$. However, this choice was solely based on compute and efficiency constraints, and other approximations such as gaussian or principal curvature can also be used. Since our surfaces are neural implicit surfaces, we use techniques in differential geometry for neural implicit functions as proposed in [6] to estimate curvature. For a general case, we can define a shape operator, $\mathbf{d}N$, on the tangent plane at point t_i. The shape operator, $\mathbf{d}N$, can be expressed as follows, where $\mathbf{H}$ is the Hessian operator:

$$\mathbf{d}N = \left(I - \hat{\mathbf{n}}(t) \cdot \hat{\mathbf{n}}(t)^T\right) \frac{\mathbf{H}f_{\mathcal{S}}}{||\nabla f_{\mathcal{S}}||} \tag{1}$$

From the shape operator, we can find the curvature along any vector $\mathbf{v}$:

$$\kappa_{\mathbf{v}} = \left\langle -\mathbf{d}N \cdot \mathbf{v}, \mathbf{v} \right\rangle \tag{2}$$

, where $\langle ., . \rangle$ is the inner product. Using Eq. 2 we can compute principal, mean or gaussian curvatures to estimate the differential surface at t_i. By using gaussian curvature, for instance, we can approximate our surface to be locally quadric such as handling surfaces that are hyperbolic. Our ray-sphere intersection will now be able to change to ray-ellipse, ray-hyperbolic, ray-parabolic, or ray-planar intersection depending on the sign of the curvature.

Note that for concave surfaces $K_{t_i} < 0$, so $\hat{\mathbf{o}_\mathbf{S}}$ will lie outside the object and, for convex, $K_{t_i} > 0$, $\hat{\mathbf{o}_\mathbf{S}}$ will lie inside the object. This is a useful property as the virtual-cone apex changes based on the curvature and our formulation is generalizable to locally concave and convex surfaces.

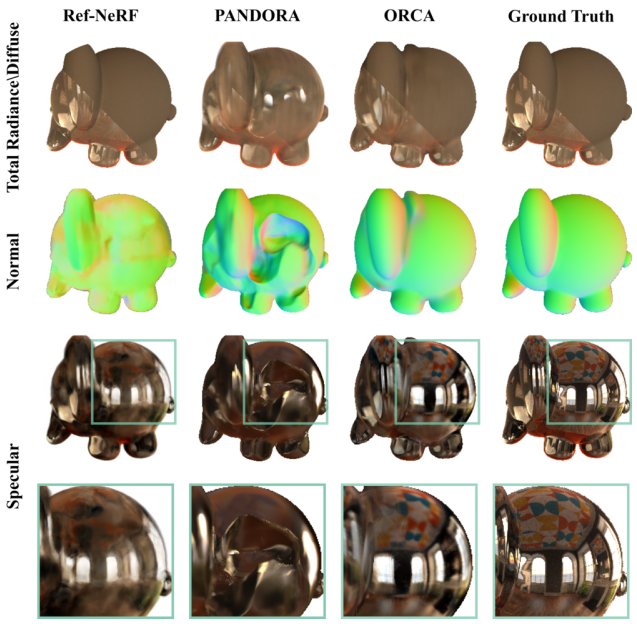

Figure 1. **Comparisons on Elephant-in-the-Room dataset**. We compare a sample test viewpoint against existing techniques that only capture an environment map. We show that our method outputs smoother surface normals, and diffuse and specular separation, in addition to the recovery of finer details such as the textured ceiling and the high-frequency illumination on the elephant through the windows.

1.2. Relation to Caustics

To convert the object into a camera, we model the object's surface as a sensor. As discussed, the center-of-projection of the object-as-camera changes wrt geometry and viewing direction, however, as shown by [4] [8], it must lie on the caustic surface of the object. One way to estimate the virtual viewpoints or the apex of the virtual cone is using the known caustic surface of the object, however, our formulation assumes unknown geometry therefore the surface is unknown. To account for this approximate the virtual viewpoint with the closest point to the reflected rays. We

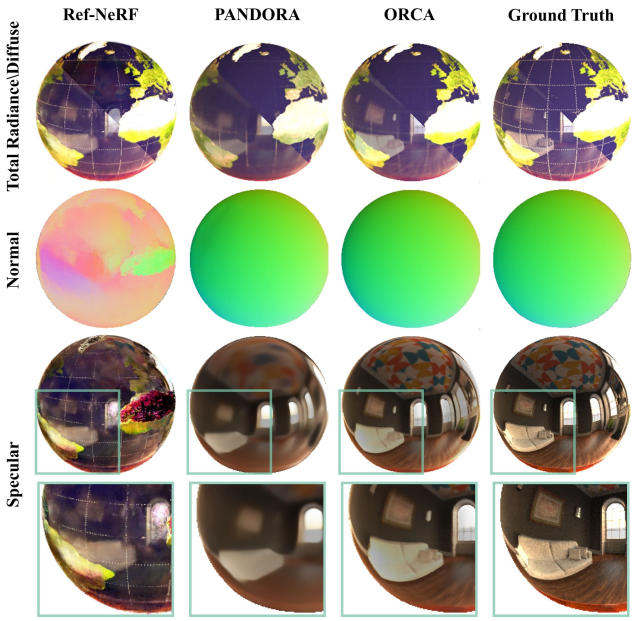

Figure 2. **Comparisons on Globe-in-the-room dataset**. We compare a sample test viewpoint against existing techniques that only capture an environment map. We get better diffuse-specular separation in addition to finer specular radiance- finer details such as the cushion are more visible.

visualize this method (Sec. 3.3) in flatland using ray-circle intersection in Figure 4. We shoot real cones from a single pixel at different angles, approximated by 2 bounding rays and 1 primary ray, and intersect the real-cone with the object. We compute surface normals (yellow) and compute the associated reflected rays (green) and the virtual viewpoint (magenta) using our closest-point to reflected-rays method in Sec. 3.3. We show that by increasing the pixel resolution, the real cone radius decreases projecting a smaller virtual-pixel surface area on the object, $\mathbf{ds}_j$. We can calculate the virtual viewpoint for this pixel and empirically show that as $\mathbf{ds}_j \rightarrow 0$, the virtual viewpoints along the surface tend to form the catacaustic of the object. We can also use our method to approximate the caustic of unknown geometry and has applications in Catadioptric Imaging Systems (CIS). Moreover, we also note that our method is limited by the resolution of the camera viewing the object- for lower resolution or objects further away, the virtual viewpoint will not be accurate.

We then calculate approximates the true caustic of the circle at higher resolutions.

2. Object-as-radiance-field Camera to Virtual Camera

We convert the glossy object into a radiance-field camera by modeling the objects' surface as the virtual sensor. The

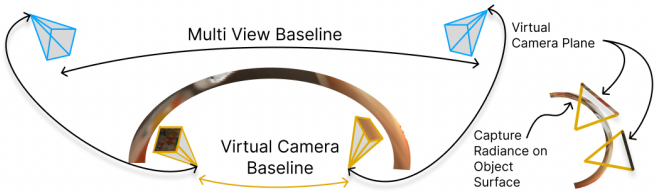

Figure 3. **Glossy object's size acts as virtual baseline** On the left, we show that the baseline for the virtual views is fundamentally limited by the object size. On the right, we show that our environment radiance field must learn to map radiance accumulated on the object-surface-as-sensor to the new virtual camera image plane with a new virtual center-of-projection to perform novel view synthesis. The distortion is high for objects with varying geometry or a low radius of curvature, but we show in our paper that our formulation of virtual cones can handle this undistortion well even for complex geometries.

virtual sensor accumulates radiance along its surface and we model the 2D radiance on this surface-as-sensor as a function of the 5D environment radiance field. Each individual virtual pixel and the associated virtual cone is defined the object surface accumulating radiance along the object surface. However, the virtual camera and the virtual view are at a different location near the object surface. As Fig. 3 shows, the virtual camera plane is different that the object surface and has a different virtual center-of-projection. Therefore Mip-NeRF must learn how to interpolate and undistort the incoming radiance from the virtual pixels. Although the difference between the virtual camera plane and the virtual pixel surface is exasperated for complex-varying geometry and low radius of curvature, we show that these networks can handle such un-distrotions well. This makes it useful for cameras that have complex or curved sensor planes.

3. Training Details

3.1. Training Procedure

We utilize a two-step training procedure for ORCa, broadly aiming to begin with rapid surface estimation from our mask network, called MaskNet, and subsequently learning diffuse and specular cues for simultaneous improved complex surface and virtual sensor estimation. We utilize 128 rays for training, 128 samples for the diffuse and VolSDF network, and 64 samples for the environment Mip-NeRF network. We begin each run with 2000 training iterations using only MaskNet loss, with the aim of estimating a convex hull of the 3D object to speed up convergence towards accurate surface estimation. Then, we utilize all three network losses, incorporating mask, diffuse, and specular network losses with the aim of both (1) recovering accurate complex surface geometry features, and (2) forming accurate virtual sensors from the object surface to estimate 5D environment radiance.

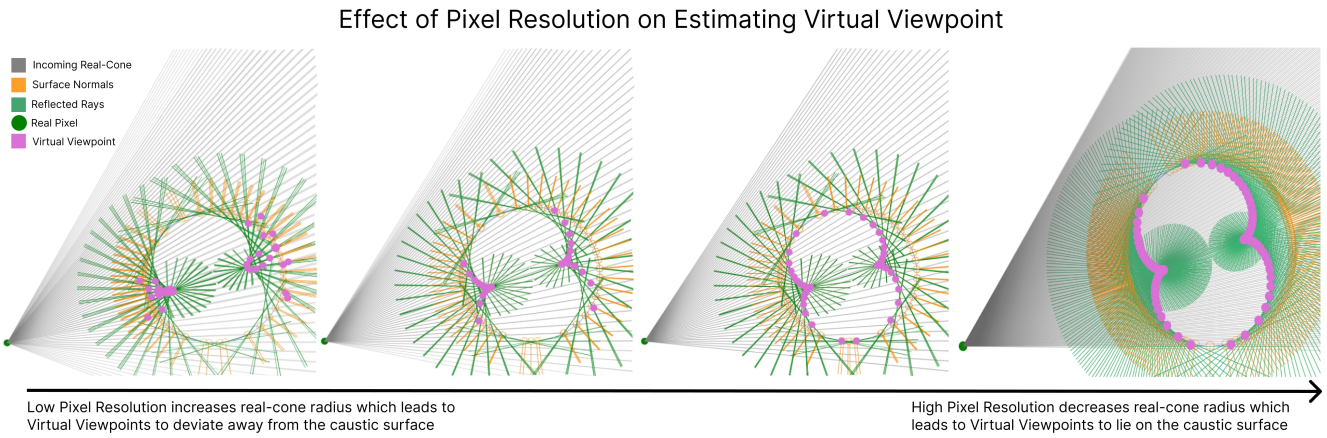

Figure 4. **Effect of Pixel Resolution On Virtual Viewpoints**. We cast a real cone (grey) from each pixel (dark green) with decreasing radii (indicating a higher resolution) in different directions. The cone, parametrized by 3 rays intersects the circle and we compute the surface normals (yellow) and reflected rays (light green). We find the closest intersection point between the reflected rays by solving least-squares and denote that as the virtual viewpoint (magenta). As we decrease the real cone radii, the virtual pixel surface area, $\mathbf{ds}_j$ also decreases and the reflected rays are closer together pointing in similar directions. As $\mathbf{ds}_j \rightarrow 0$ the virtual viewpoint starts to form a catacaustic of a circle- which denotes the true loci of virtual viewpoints of the object-as-camera.

We constrain the near and far ray sampling for both the object surface and environment radiance estimation to values specific to the given dataset for faster convergence; this, however, does not have to be known prior to training and only serves to speed up accurate estimation. We train for approximately 400k iterations, using a learning rate of 5.0e-5 and an exponential learning rate decay.

3.2. Network Details & Hyperparameters

Our model is implemented in PyTorch [7] and trained using the Adam optimizer [5]. As in PANDORA [3], we parameterize $f_{\mathcal{S}}$ with an 8-layer MLP to estimate the surface, and, as in Mip-NeRF [1], f_d with 4-layer MLP with input geometric features of size 512 from $f_{\mathcal{S}}$. We follow the sdf-to-opacity conversion and the iterative sampling of the ray, as proposed in [10]. The VolSDF network, which is used for estimating the geometry of the glossy object, is trained with an initial learning rate of 1×10^{-6}. The Mip-NeRF network, which is used to estimate the environment map surrounding the glossy object, is trained with an initial learning rate of 5×10^{-5}. Both networks are trained with exponential learning rate decay and a decay rate of 0.1. We initially train the VolSDF network for 3,000 iterations, with only the MaskNet loss being enforced for the first 1,000 iterations. After 3,000 iterations, we jointly train both networks together until they have converged on accurate object geometry and environment maps.

We set the following near/far values for each dataset. For datasets in the living room scene rendered with Mitsuba2, we use a near/far of 0.2/0.6. For datasets in the pokemon scene rendered with Mitsuba2, we use a near/far of 1.25/2.5. For real-world datasets, we use a near/far of 0/6.

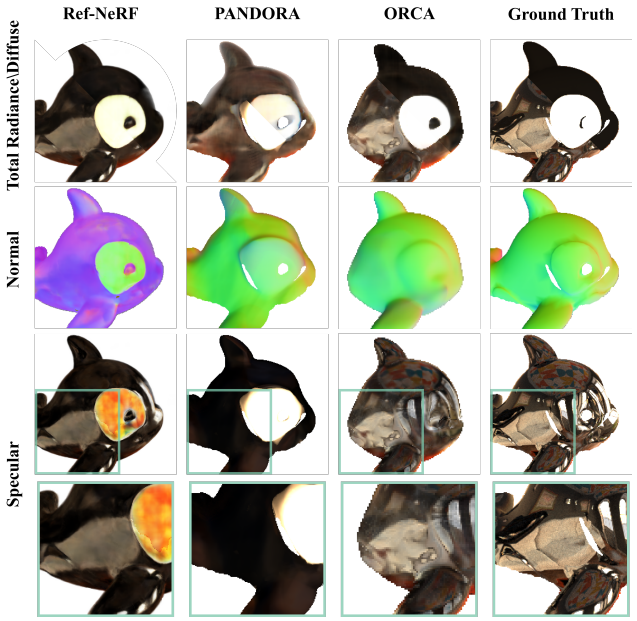

Figure 5. **Comparisons on Orca in the living room dataset**. We compare a sample test viewpoint against existing techniques that only capture an environment map.

3.3. Losses

Our method is trained end-to-end on multi-view images of the object and jointly estimates object geometry, diffuse radiance, and the 5D environment radiance field. The final loss is as follows:

$$\mathcal{L} = \lambda_{eikonal}\mathcal{L}_{eikonal} + \lambda_{mask}\mathcal{L}_{mask} + \lambda_{normal}\mathcal{L}_{normal} + \lambda_{distort}\mathcal{L}_{distort} + \left(\lambda_{coarse}\mathcal{L}_{coarse} + \lambda_{fine}\mathcal{L}_{fine}\right)$$

We use $\mathcal{L}_{eikonal}$ as proposed in [10] with a weighting of 0.1. $\mathcal{L}_{coarse}$ is the L1 loss between the measured radiance and the sum of the diffuse color and the coarse specular color predicted by the environment radiance field. Similarly, $\mathcal{L}_{fine}$ is the L1 loss between measured radiance and the sum of diffuse and fine specular radiance. We set $\lambda_{coarse} = 0.1$ and $\lambda_{spec} = 1.0$. To smooth out the normals for complex geometry, we also used the normal loss $\mathcal{L}_{normal}$ proposed in [9], $\lambda_{normal} = 1.0$. We also experimented with the distortion loss to avoid floaters in the scene similar to [2], however, we noticed little difference as a result of it.

3.4. Volumetric Masking

We utilize binary object masks to learn a 3D mask of the target object to allow for faster convergence and surface estimation. Similar to previous works [3], we estimate this 3D mask to enable a faster convergence with object geometry. We utilize a coordinate-based MLP for this 3D mask estimation, which we refer to as MaskNet, and train the network using 2D binary object masks corresponding to the multi-view image inputs. This 3D mask represents the object's 3D convex hull and thus does not represent the concave features of the target object; these concave features and other complex geometry facets are learned in subsequent training stages using diffuse and specular networks. However, we note that this 3D mask is not required for learning surface curvature and geometry and only serves to improve the speed at a coarse geometry is estimated.

4. Analysis

4.1. Roughness

We also capture the environment radiance field on a globe with high roughness and show the specular and diffuse radiance, depth from the object's surface to its surroundings in addition to a virtual view. We note that even for rougher objects our framework can recover an environment radiance field. However, the recovered radiance field is blurry due to the roughness acting like a low pass filter that removes the high-frequency components such as the cushion on the sofa or blurs the textures on the ceiling. The recovered radiance field, associated virtual views and the depth from the object surface are therefore blurrier and coarse respectively. For example, we are able to recover coarse depth in Fig. 7, however, the depth-from-object-surface is smoother at the ceiling with the globe with low roughness. In future work, we can also expand our cone formulation to include a roughness parameter, similar to RefNeRF [9], that can change the radius and apex of the virtual cone to account for rougher objects.

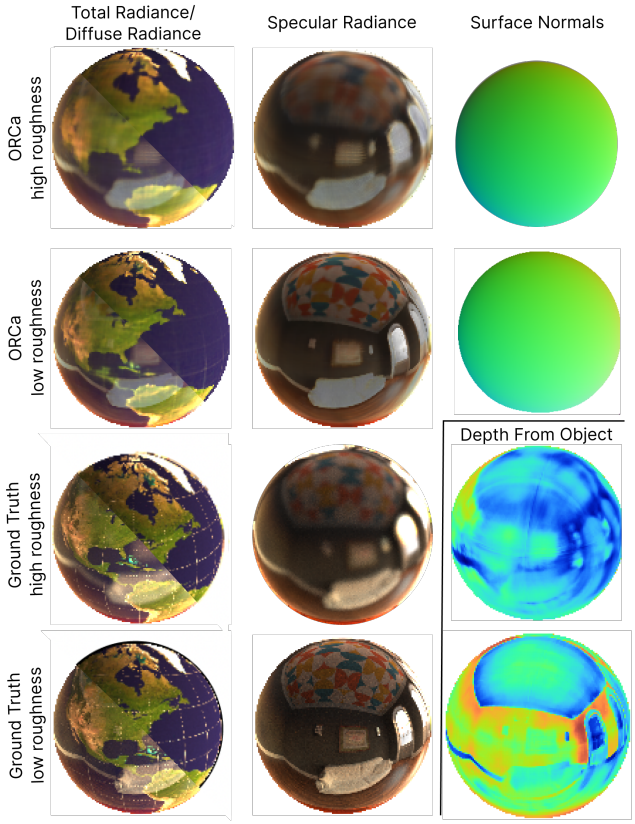

Figure 6. **Effect of adding roughness**. We show that our method is still able to recover the environment radiance field from rougher objects. Rougher objects act like a low pass filter that blurs the environment radiance visible to the real camera therefore the recoverable specular radiance and virtual views are blurry lacking the high-frequency details such as the cushion on the sofa. The recovered depth is coarse with rougher objects.

4.2. Object size as virtual baseline

As Figure 3 shows, the virtual baseline for convex objects will lie inside the object's surface or near the object's surface for concave objects. This means that the virtual baselines are much smaller and limited by object geometry- as the object size decreases, the virtual cones' apex will be close to each other, and the multi-view virtual baseline will tend to 0, effectively acting as a monocular setup. This also means that associated radiance field and depth maps will also be more coarse for small objects.

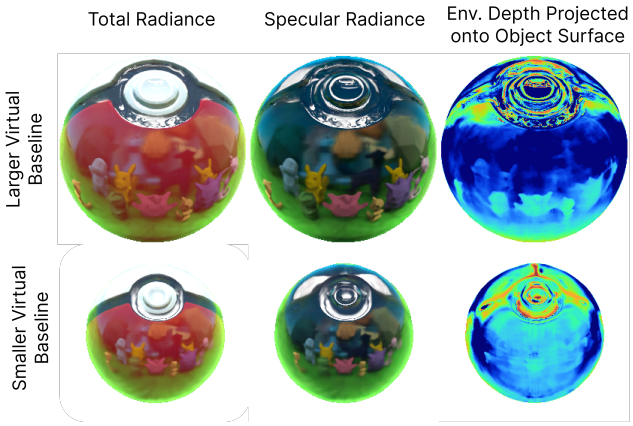

Figure 7. **Effect of Baseline on Depth**. We show that with a larger baseline, we get more accurate depth to the surroundings. This is because the virtual baseline is dependent on object geometry and for smaller objects. Similar to multi-view setups, we get less accurate depth with a smaller baseline.

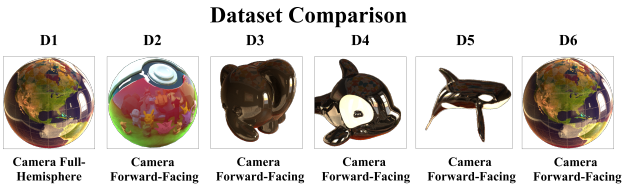

Figure 8. **Dataset Information**: We provide information about the simulated datasets used using sample images and camera distributions. We test complex geometries that have inter-reflections and test complex scenes as well in simulation.

Scene	Approach	Diffuse Radiance PSNR ↑ (dB)	Diffuse Radiance SSIM ↑	Specular Radiance PSNR ↑ (dB)	Specular Radiance SSIM ↑	Mixed Radiance PSNR ↑ (dB)	Mixed Radiance SSIM ↑	Normals MAE ↓ (°)
D1	Ref-NeRF	17.59	0.7217	14.88	0.4750	19.58	0.7956	62.45
	PANDORA	13.23	0.4759	15.12	0.5231	12.87	0.4607	2.387
	ORCA	13.29	0.4683	16.64	0.5148	18.23	0.5745	1.873
D2	Ref-NeRF	11.86	0.6090	15.28	0.7059	21.80	0.8643	33.92
	PANDORA	22.53	0.8689	17.76	0.6326	22.73	0.7787	3.693
	ORCA	23.47	0.8954	18.98	0.6954	22.31	0.8107	3.568
D3	Ref-NeRF	25.95	0.8977	21.37	0.7281	24.07	0.8502	40.369
	PANDORA	21.90	0.8536	17.87	0.6522	21.31	0.7746	11.680
	ORCA	22.98	0.9158	20.45	0.7216	22.74	0.8073	3.863
D4	Ref-NeRF	24.33	0.8454	16.34	0.6578	17.75	0.7826	52.127
	PANDORA	16.10	0.6352	13.55	0.5626	16.92	0.7393	13.757
	ORCA	17.63	0.8274	22.40	0.8153	23.00	0.8422	3.234
D5	Ref-NeRF	20.97	0.8876	19.25	0.9279	24.72	0.9694	15.203
	PANDORA	14.90	0.8452	15.22	0.8715	17.99	0.9202	5.376
	ORCA	18.66	0.8894	21.28	0.9392	21.47	0.9470	0.973
D6	Ref-NeRF	12.08	0.4211	14.80	0.4850	13.21	0.4719	58.071
	PANDORA	20.85	0.6769	17.97	0.6479	21.57	0.6970	8.747
	ORCA	23.03	0.7396	24.66	0.8346	24.25	0.7862	0.521
Average (D1 - D6)	Ref-NeRF	18.80	0.7304	16.99	0.6633	20.19	0.7890	43.690
	PANDORA	18.25	0.7260	16.25	0.6483	18.90	0.7284	7.606
	ORCA	**19.84**	**0.7893**	**20.74**	**0.7535**	**22.00**	**0.7947**	**2.339**

Table 1. **Metrics for more rendered scenes**. (D1,D2 in paper)

5. Additional Comparisons

5.1. Quantitative Simulated Comparisons

In Fig. 8, we show an example of the input RGB image for the six synthetic scenes along with the camera view distribution used. Here 'forward facing' refers to concentrating the views over one-quarter of the hemisphere. The scenes have variety in the geometry, surrounding environment and view distribution. The datasets would be released upon publication. In Table 1, we show the individual metrics obtained on each of the scenes along with the average values reported in the main text.

Figure 9. **ORCa applications**: Using the learned environment field, we can insert shiny virtual objects with realistic view and position dependent reflections (a). From the learned geometry and environment field, we can render the object under novel appearances such as metallic appearance (b). Please refer to the supplementary video for multi-view renderings.

6. Additional Applications

The estimated geometry, diffuse-specular separation, and environment radiance field can be rendered under novel configurations to enable augmented reality applications (Fig. 9). We add a synthetic shiny sphere of known geometry to the scene by querying the learned environment radiance field at the rays reflected by the sphere. Modeling the environment as 5D radiance enables realistic reflections on the sphere as it translates across the scene (Fig. 9(a)). ORCa can separate the specular radiance of the object. This specular radiance is the environment radiance field multiplied by the Fresnel reflectance of typical glossy surfaces. We can convert the object's appearance from glossy to metallic by setting the fresnel reflectance to 1 (Fig. 9(b)). ORCa can accurately model how the object's geometry distorts the near and far environment.

References

[1] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. *ICCV*, 2021. 3

[2] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 4

[3] Akshat Dave, Yongyi Zhao, and Ashok Veeraraghavan. Pandora: Polarization-aided neural decomposition of radiance. *arXiv preprint arXiv:2203.13458*, 2022. 3, 4

[4] J. Gluckman and S.K. Nayar. Planar catadioptric stereo: geometry and calibration. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, volume 1, pages 22–28 Vol. 1, 1999. 1

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[6] Tiago Novello, Guilherme Schardong, Luiz Schirmer, Vinicius da Silva, Helio Lopes, and Luiz Velho. Exploring differential geometry in neural implicits, 2022. 1

[7] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 3

[8] R. Swaminathan, M.D. Grossberg, and S.K. Nayar. Caustics of catadioptric cameras. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pages 2–9 vol.2, 2001. 1

[9] 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. 4

[10] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. 3, 4