# Neural Surface Refinement for Modeling Transparent Objects

Weijian Deng[1] Dylan Campbell[1] Chunyi Sun[1]
Shubham Kanitkar[2] Matthew E. Shaffer[2] Stephen Gould[1]

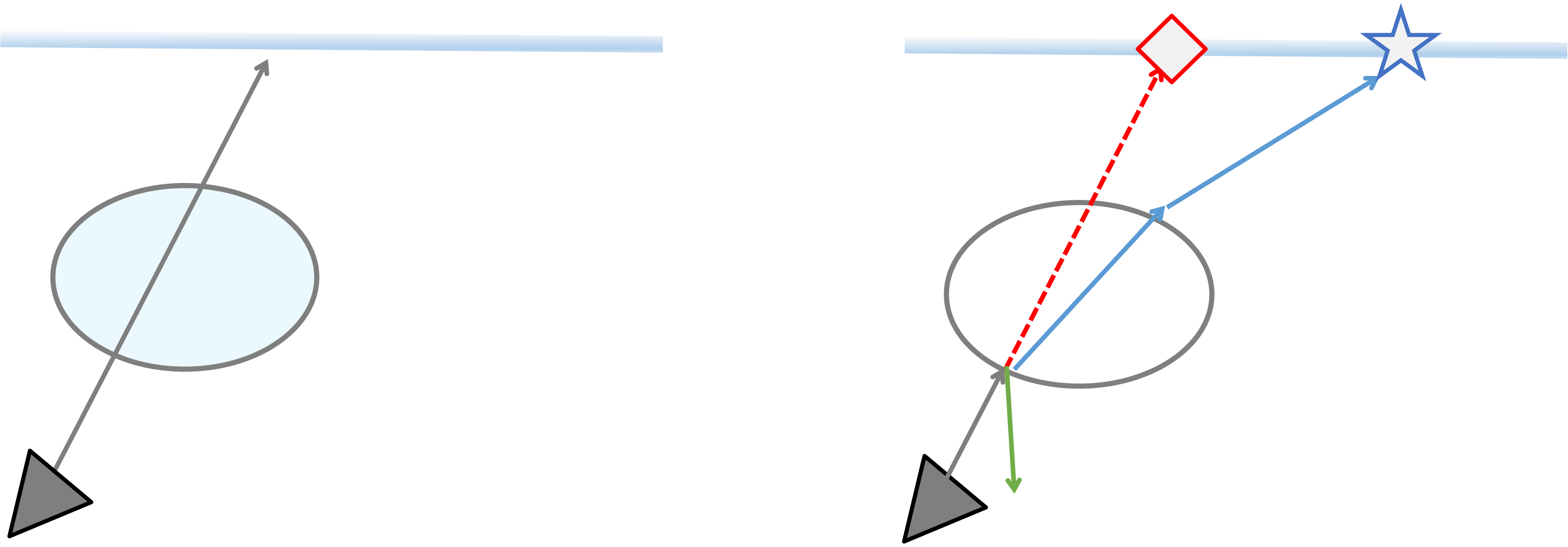
[1]Australian National University [2]RIOS Intelligent Machines

# B A C K G R O U N D

Light Behavior in Transparent Object

# B A C K G R O U N D

## Light Behavior in Transparent Object



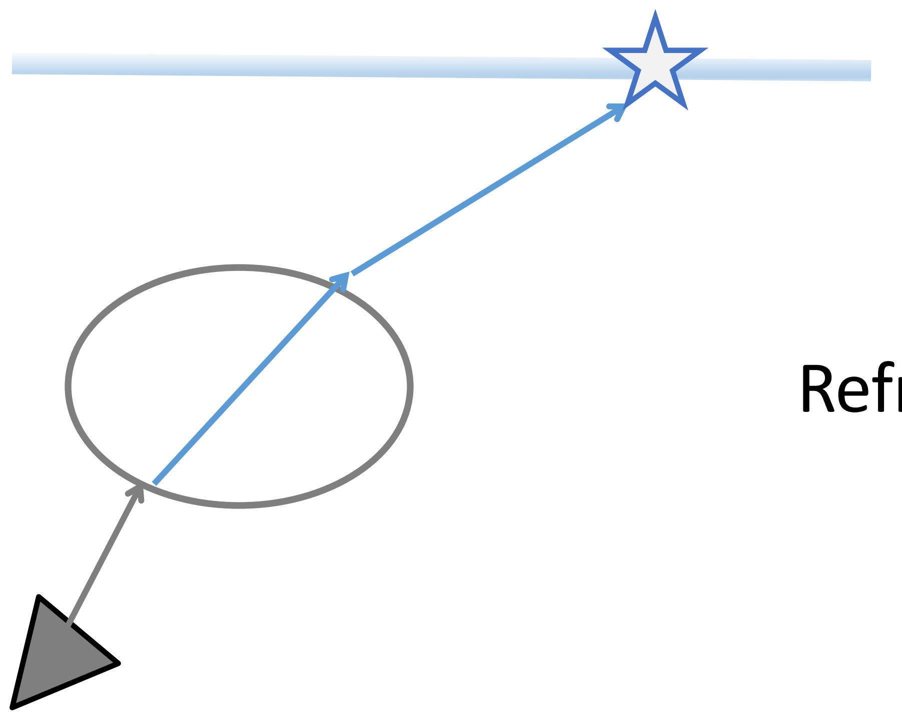
opaque object

transparent object

Straight Ray vs. Curved Ray

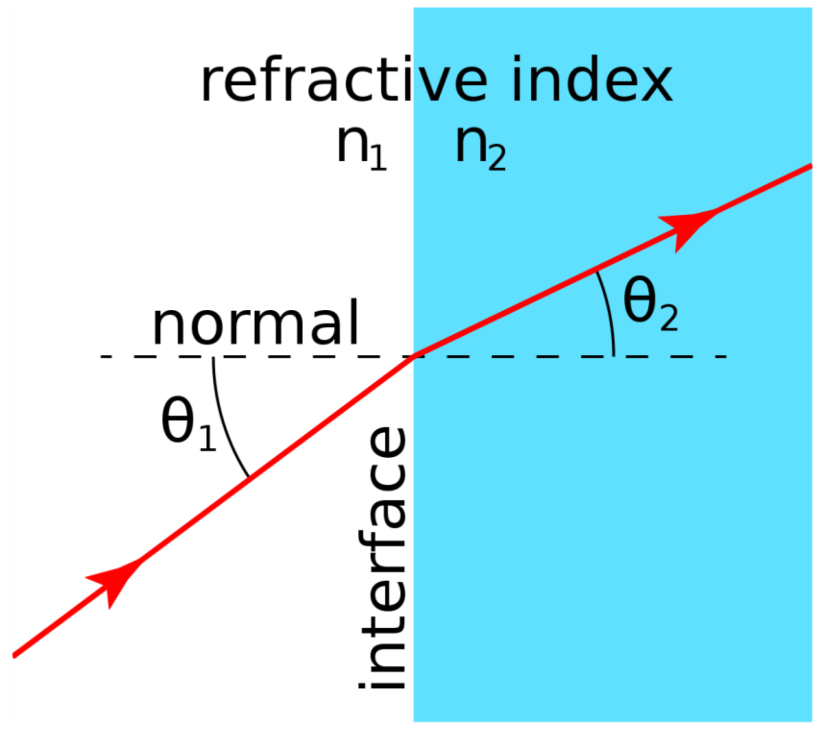
# B A C K G R O U N D

## Light Behavior in Transparent Object



transparent object

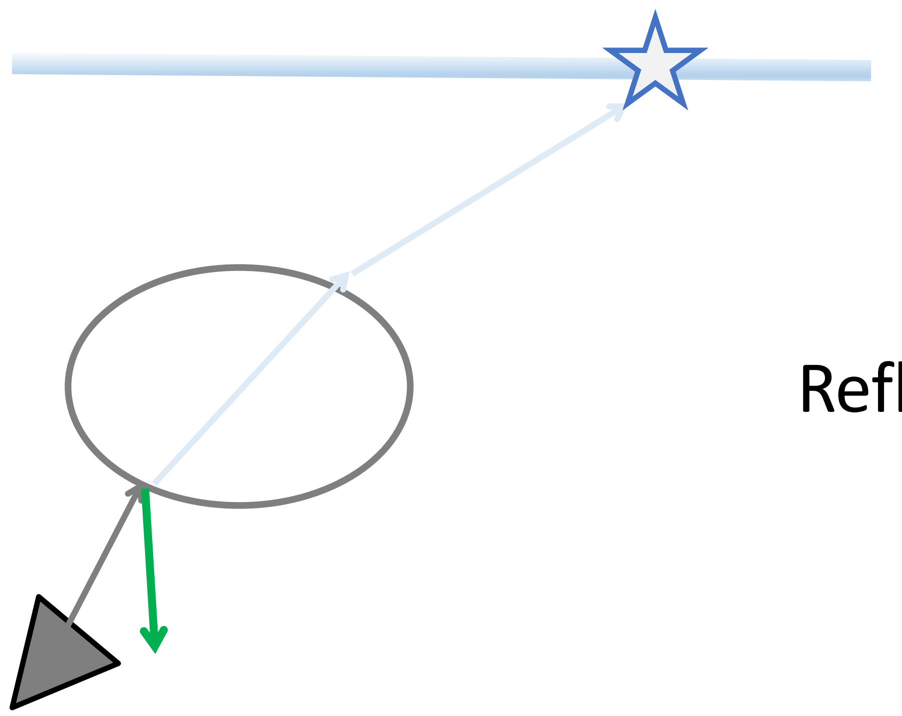
Refraction: bending light (Snell's law)




Snell's law

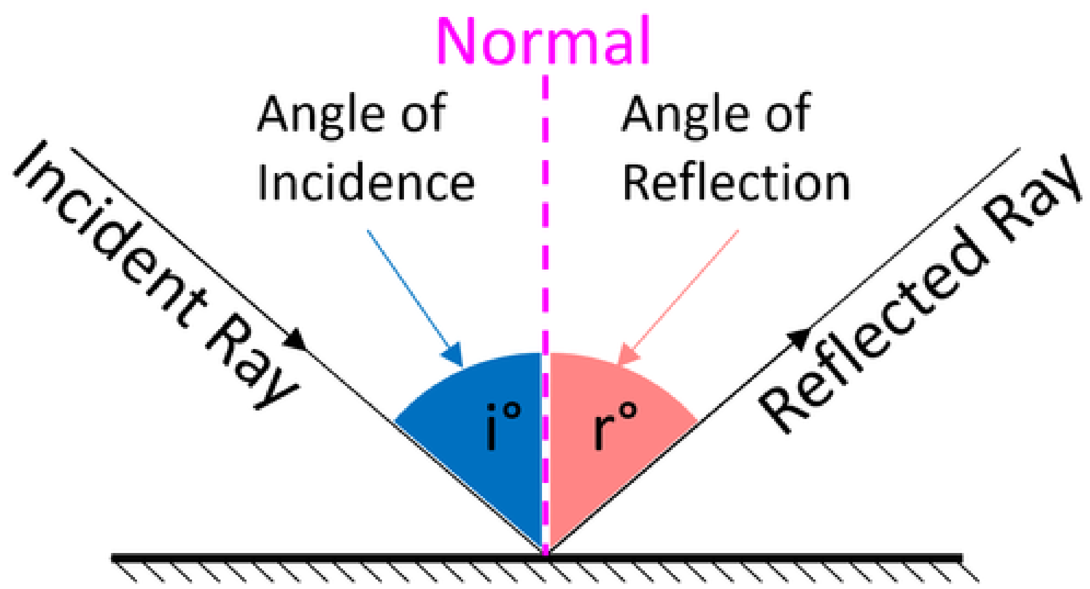
# B A C K G R O U N D

## Light Behavior in Transparent Object



transparent object

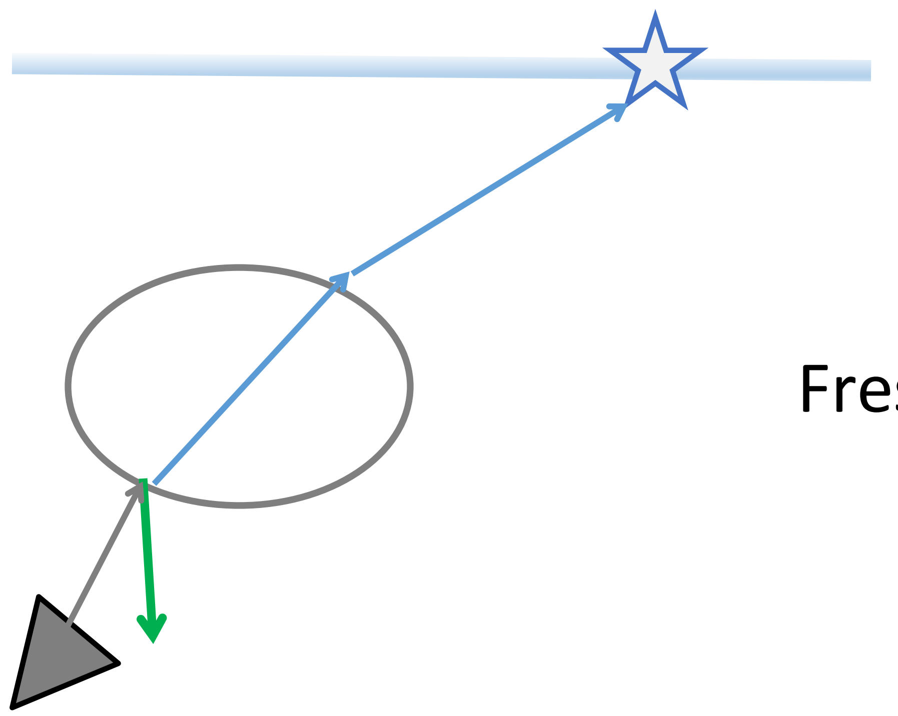
Reflection: surface reflection




surface reflection

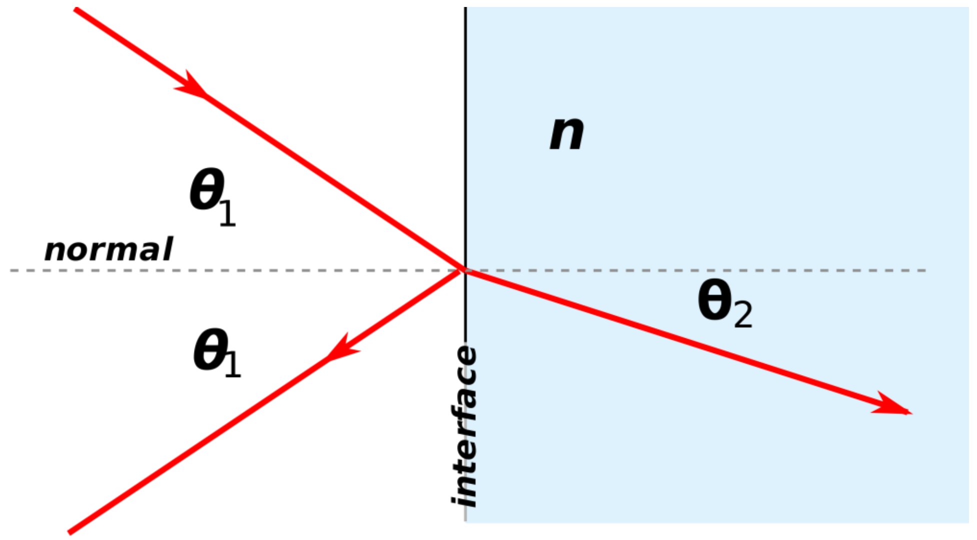
# B A C K G R O U N D

## Light Behavior in Transparent Object



transparent object

Fresnel Equation*:
merging contribution




$$F_r = \frac{1}{2}\left(\frac{\eta \mathbf{d}_0^\top \mathbf{n}_1 - \eta_t \mathbf{d}_1^\top \mathbf{n}_1}{\eta \mathbf{d}_0^\top \mathbf{n}_1 + \eta_t \mathbf{d}_1^\top \mathbf{n}_1}\right)^2 + \frac{1}{2}\left(\frac{\eta_t \mathbf{d}_0^\top \mathbf{n}_1 - \eta \mathbf{d}_1^\top \mathbf{n}_1}{\eta_t \mathbf{d}_0^\top \mathbf{n}_1 + \eta \mathbf{d}_1^\top \mathbf{n}_1}\right)^2$$

**Principles of optics: electromagnetic theory of propagation, interference and diffraction of light*

# P R O B L E M

Standard NeRFs assume the light ray transports along the straight path

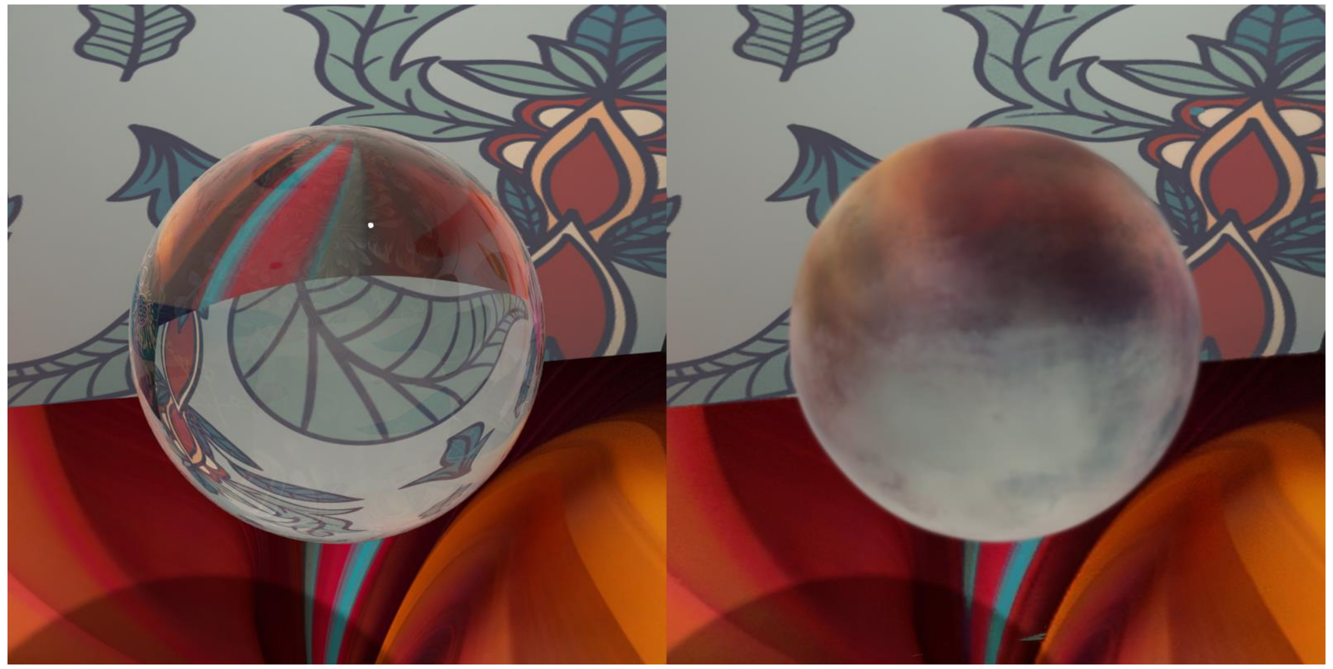
While this assumption works for opaque objects, it fells in short for transparent objects where light is bent by refraction.

# P R O B L E M

Standard NeRFs assume the light ray transports along the straight path

While this assumption works for opaque objects, it fells in short for transparent objects where light is bent by refraction.

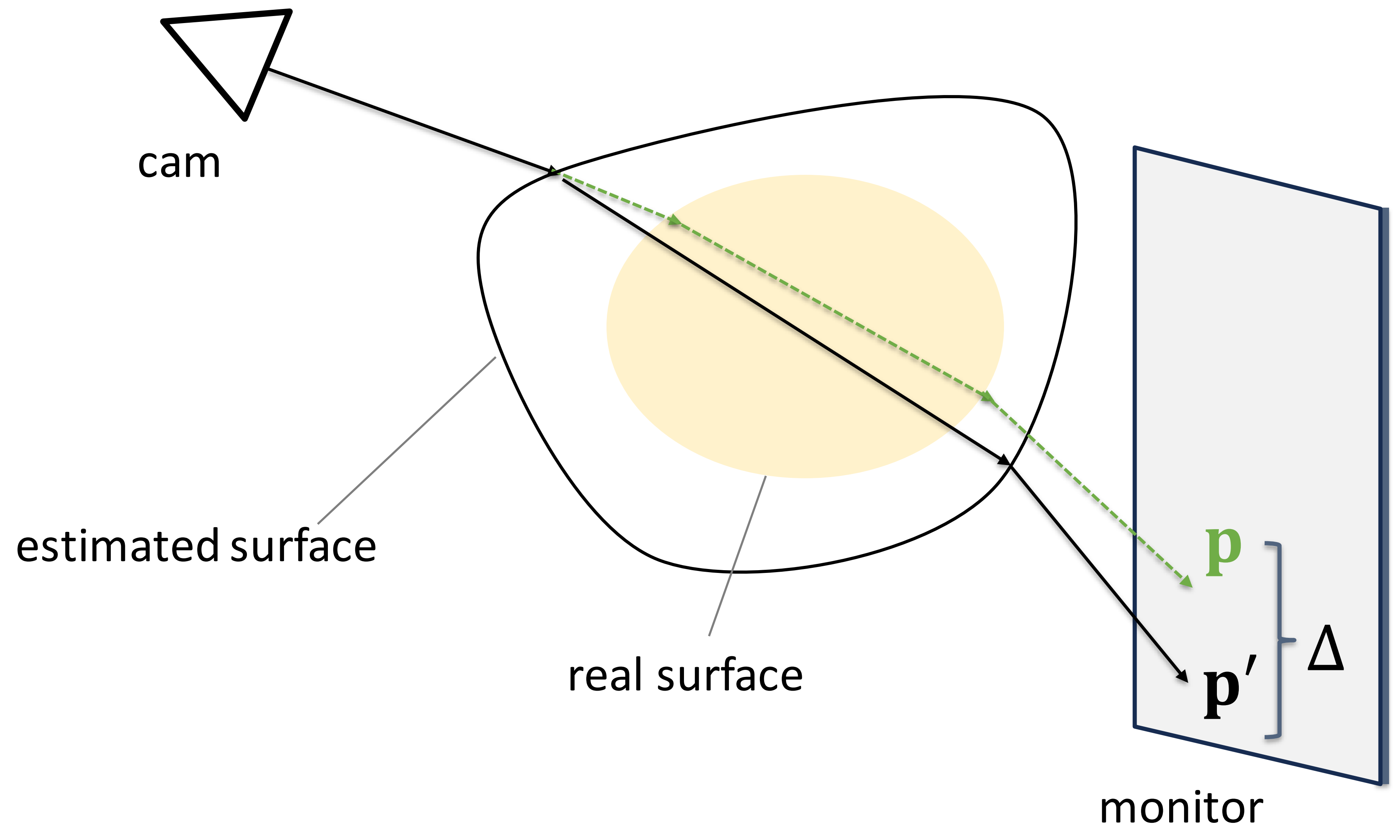
Ground Truth NeRF



Ground Truth NeRF

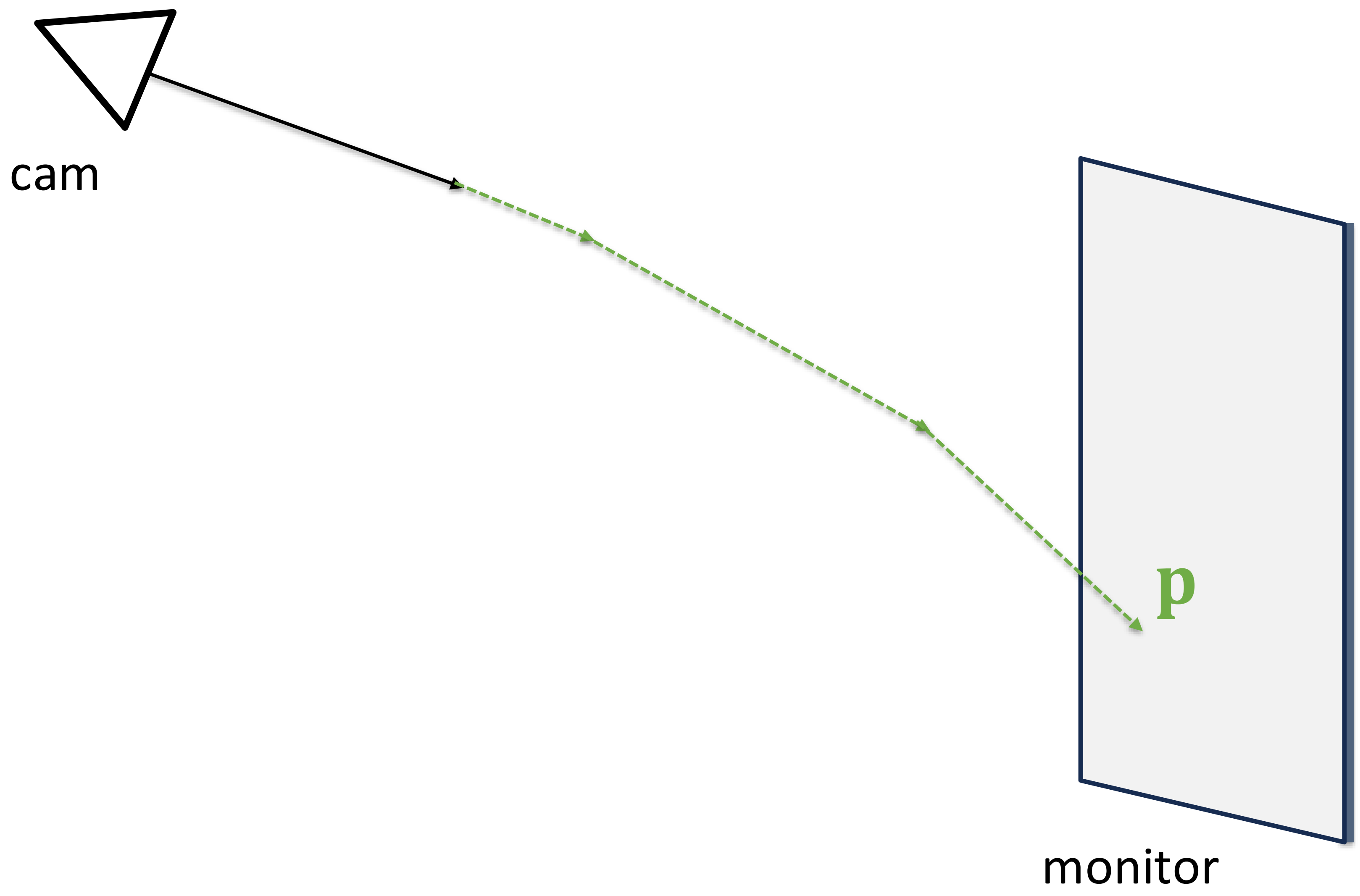
# SURFACE REFINEMENT

Ray-location correspondence[‡]




‡Lyu et al., *"Differentiable refraction-tracing for mesh reconstruction of transparent objects." In ACM Trans. Graph. 2020*

# SURFACE REFINEMENT

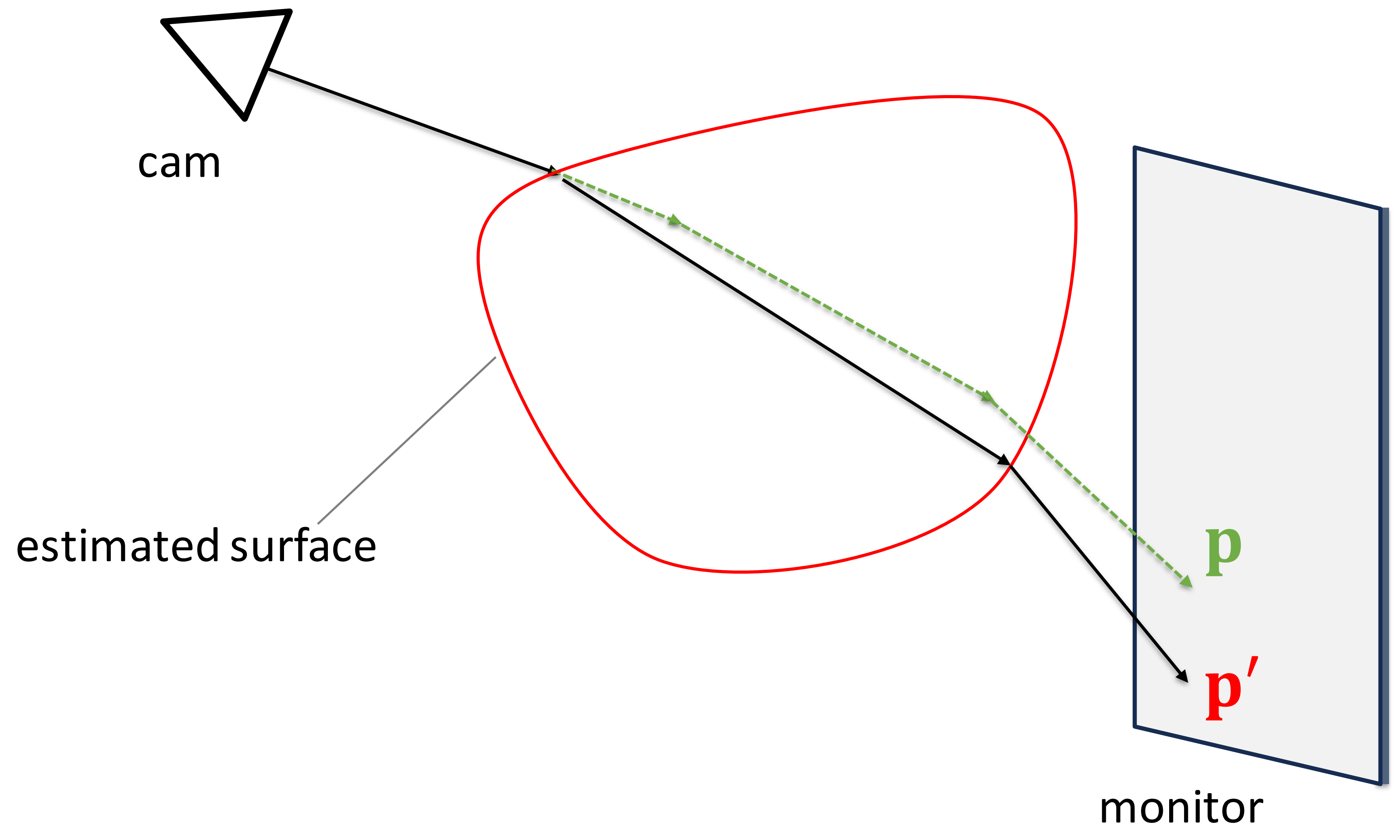



Ray-location correspondence‡

Given a well-controlled setup, the termination point of each ray is accurately obtained

‡Lyu et al., "Differentiable refraction-tracing for mesh reconstruction of transparent objects." In ACM Trans. Graph 2020

# SURFACE REFINEMENT



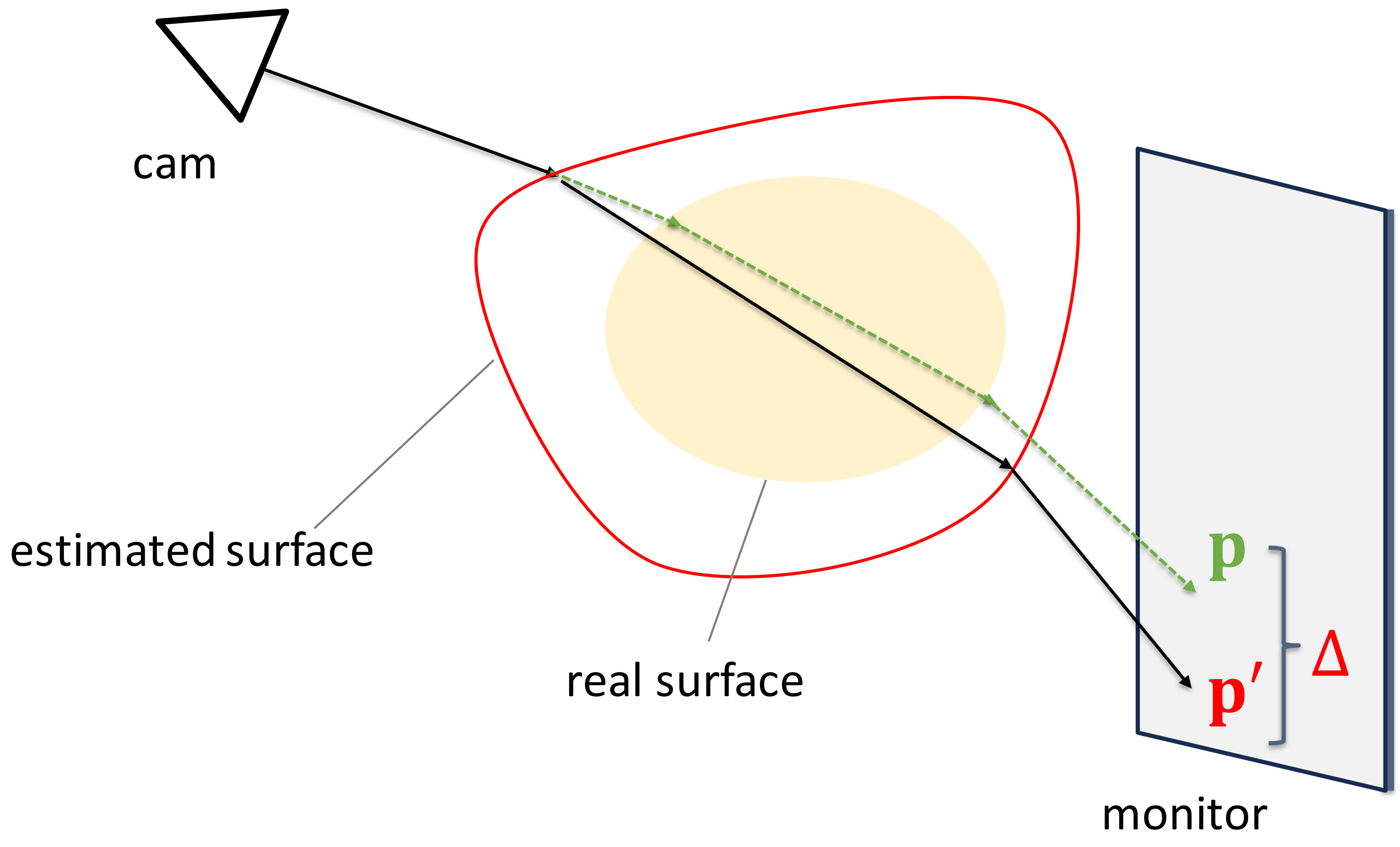

Ray-location correspondence‡

Given a well-controlled setup, the termination point of each ray is accurately obtained

Given an estimated surface, we use ray tracing to compute the end point

‡Lyu et al., "Differentiable refraction-tracing for mesh reconstruction of transparent objects." In ACM Trans. Graph. 2020

# SURFACE REFINEMENT




## Ray-location correspondence‡

Given a well-controlled setup, the termination point of each ray is accurately obtained

Given an estimated surface, we use ray tracing to compute the end point

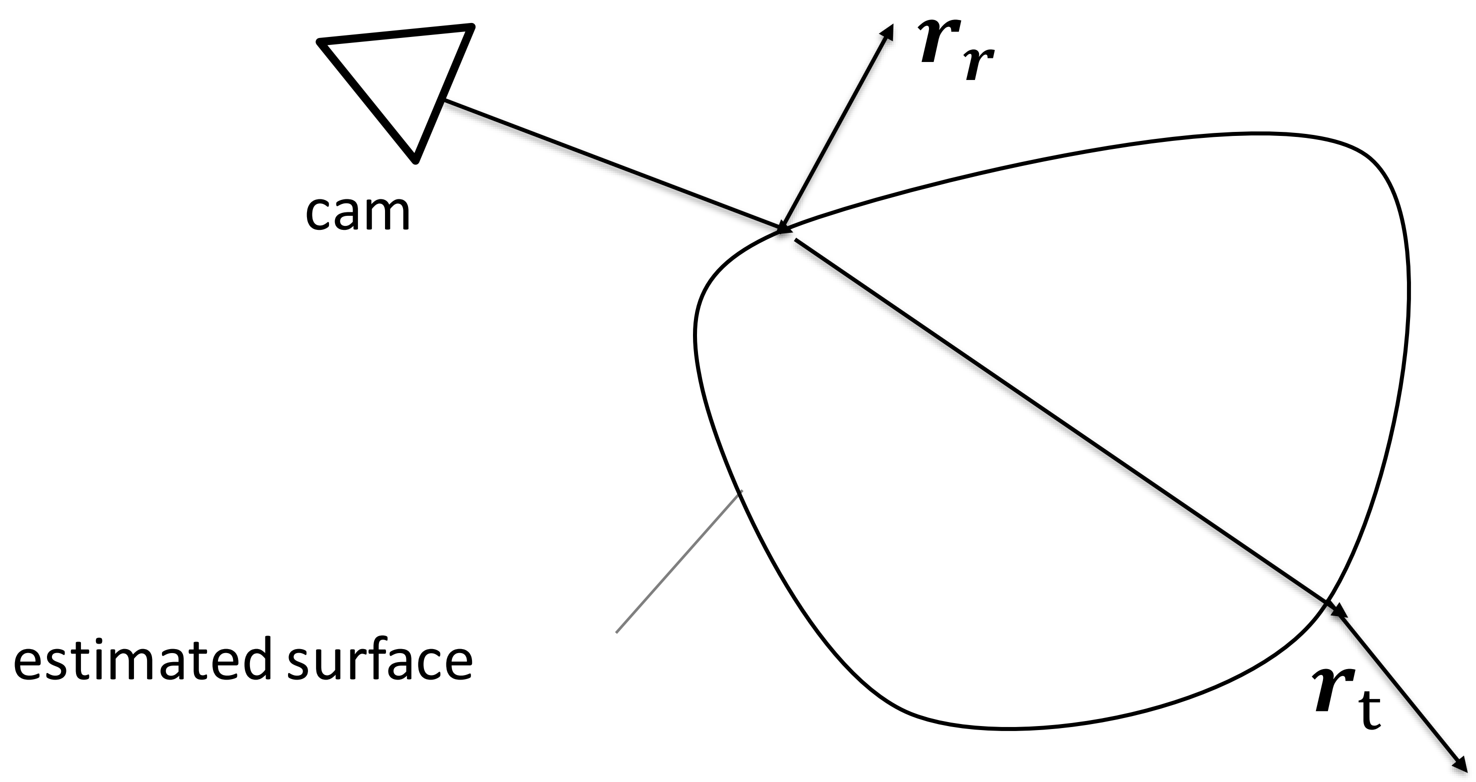
The gap between estimated and ground-truth termination points guides the surface refinement

‡Lyu et al., *"Differentiable refraction-tracing for mesh reconstruction of transparent objects." In ACM Trans. Graph. 2020*

# OUR PROPOSAL

Can we use only color information?

# OUR PROPOSAL

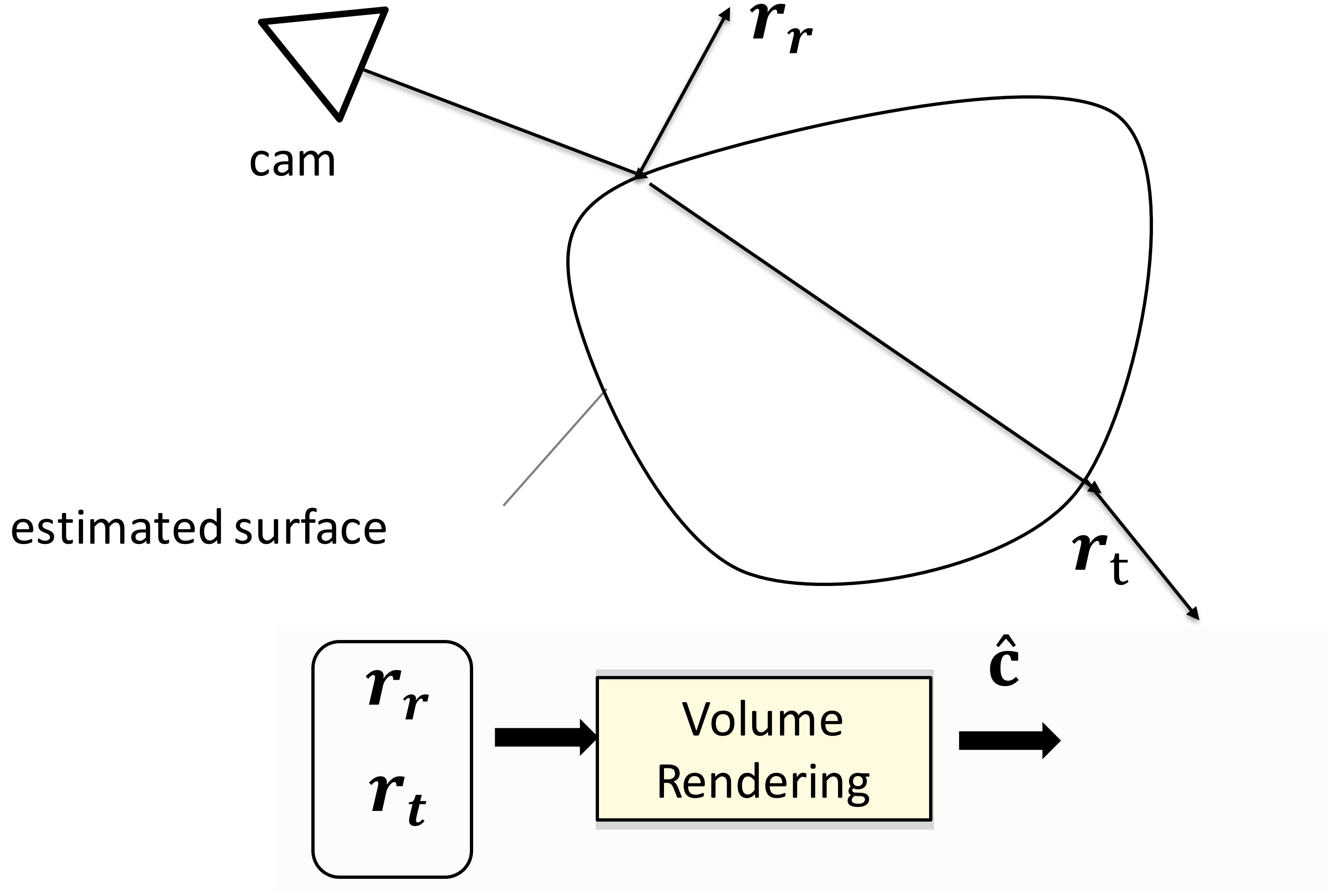



Can we use only color information?

Color Consistency

Given an estimated surface, we use ray tracing to compute the reflected and refracted rays

# OUR PROPOSAL



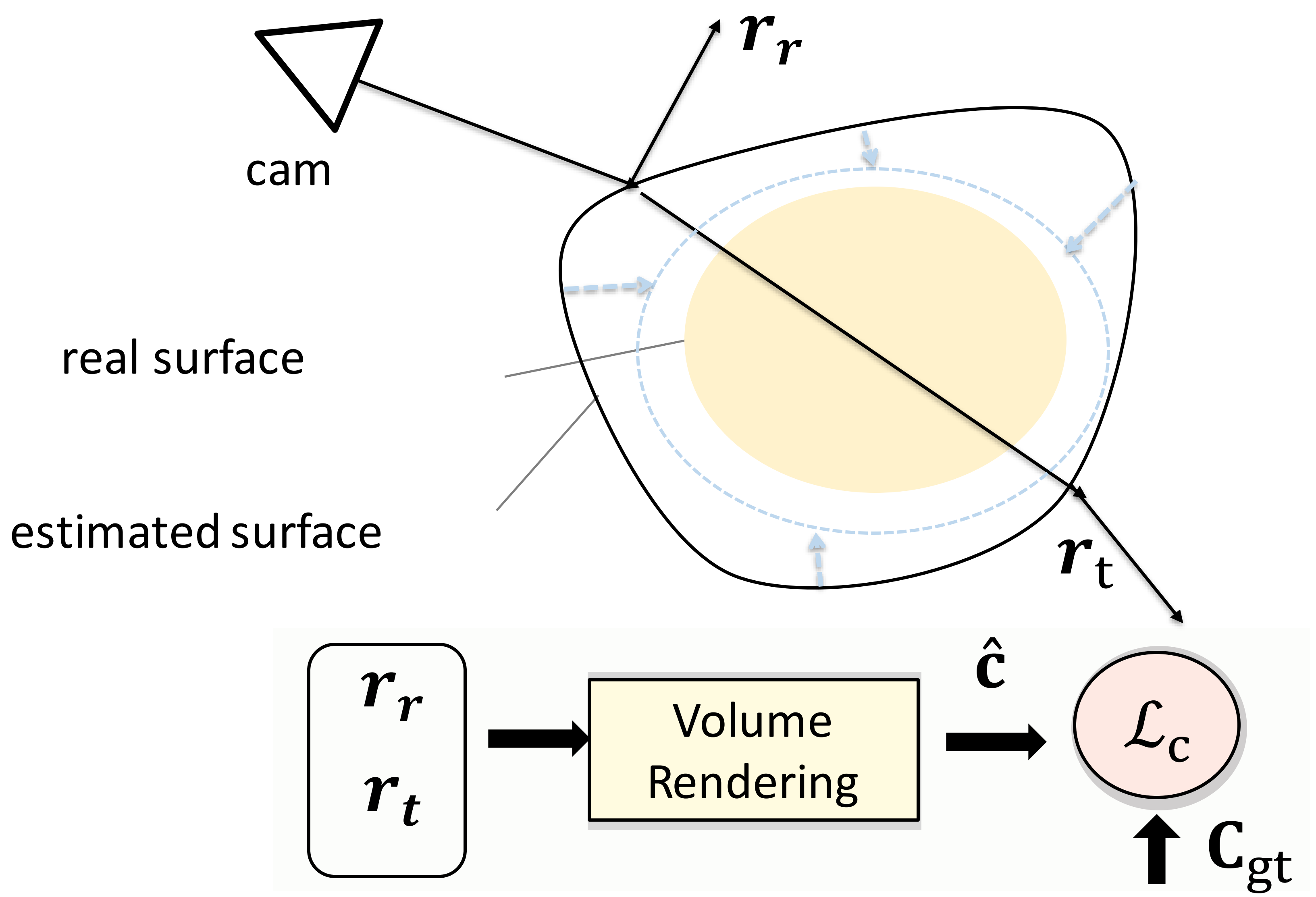

Can we use only color information?

Color Consistency

Given an estimated surface, we use ray tracing to compute the reflection and refracted rays

With volume rendering, we render the color

# OUR PROPOSAL




Can we use only color information?

Color Consistency

Given an estimated surface, we use ray tracing to compute the reflection and refracted rays

With volume rendering, we render the color

The difference between ground-truth and rendered color is the supervision for refinement

# NEURAL SURFACE REFIMENT

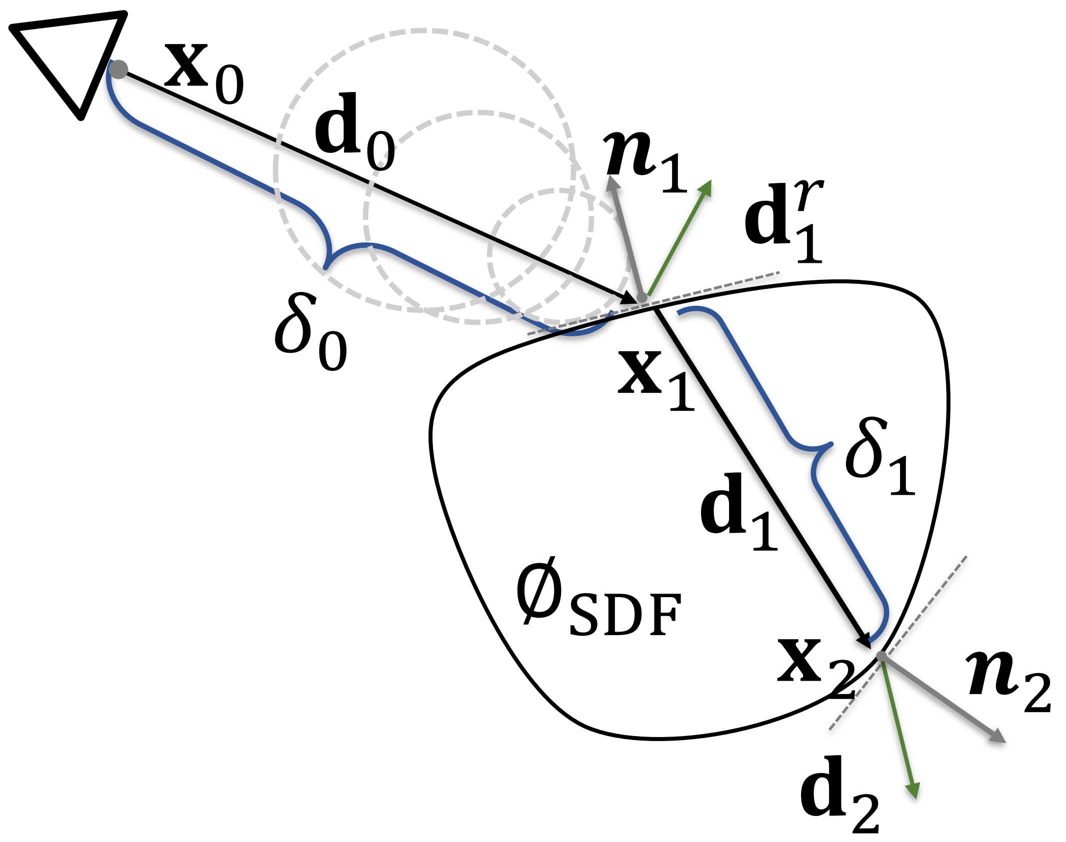
How to incorporate color consistency?

# NEURAL SURFACE REFIMENT

How to incorporate color consistency?

Sphere Tracing

Give a ray from $\mathrm{x}_0$ with direction $\mathrm{d}_0$ and $\emptyset_{\mathrm{SDF}}$, find ray-surface intersection

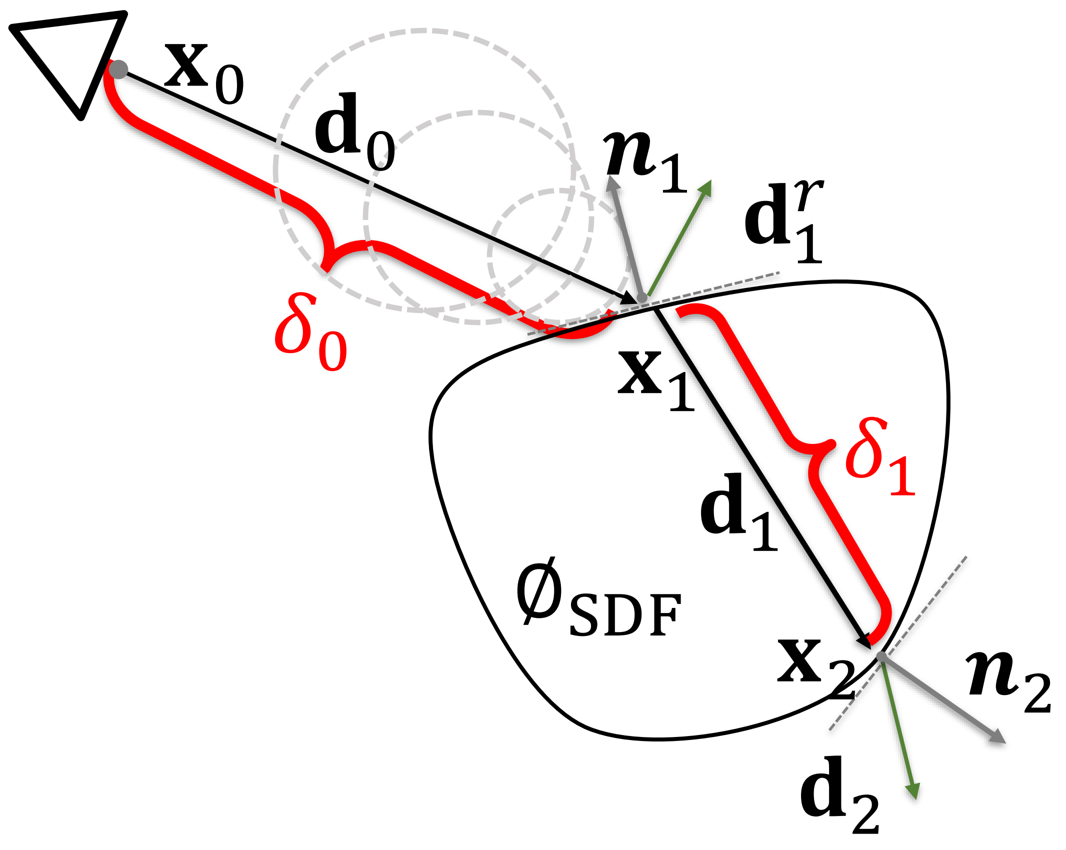


# NEURAL SURFACE REFIMENT

How to incorporate color consistency?

Ray tracing is an optimization problem

*Finding the smallest distance to the surface*



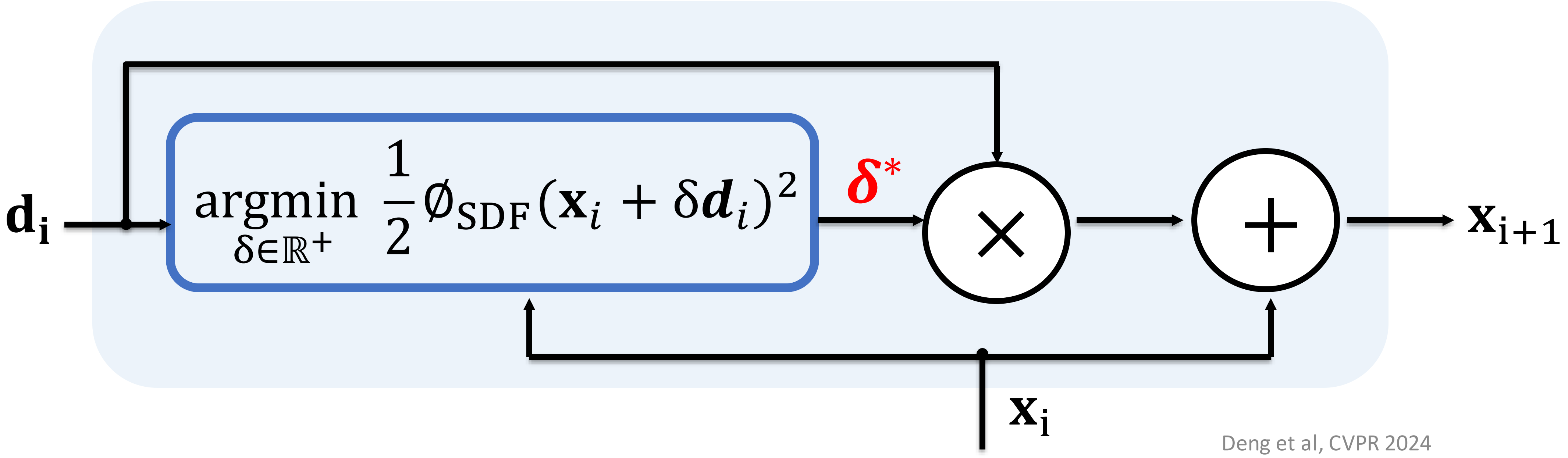
# NEURAL SURFACE REFIMENT

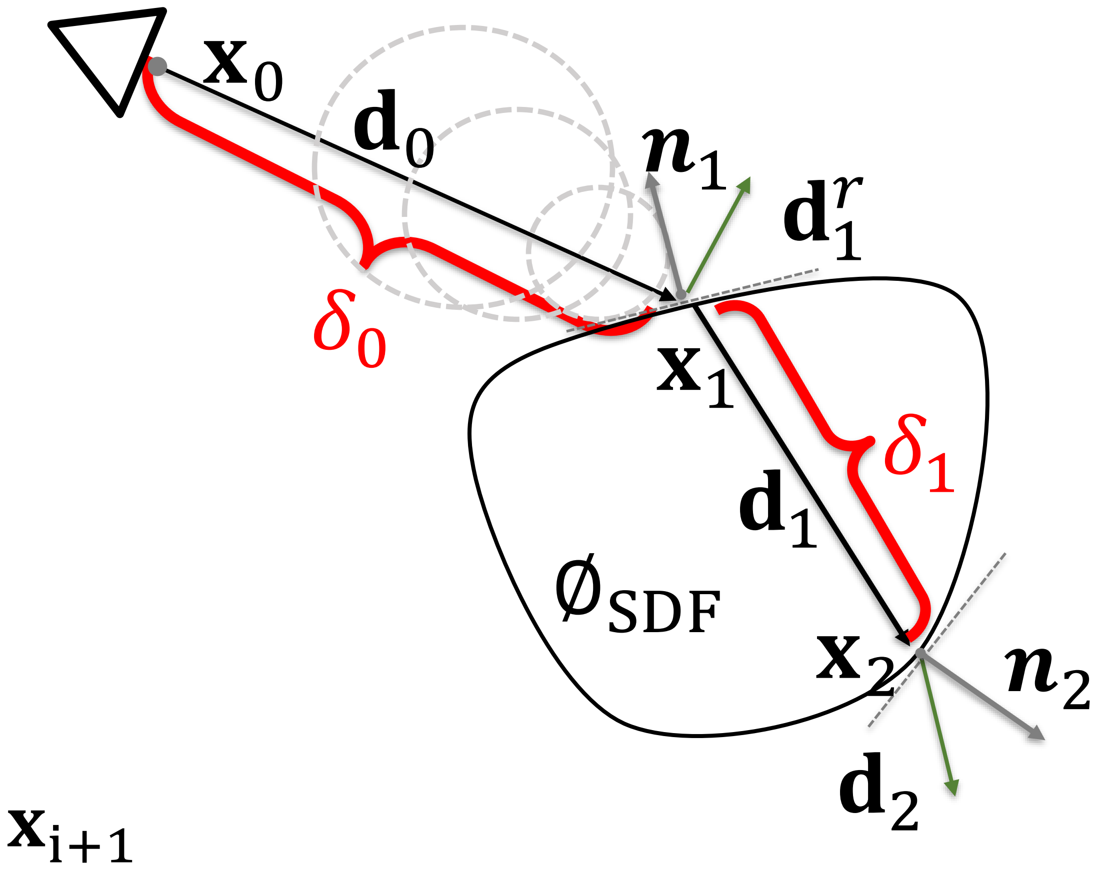
How to incorporate color consistency?

Ray tracing is an optimization problem

*Finding the smallest distance to the surface*

Intersection node

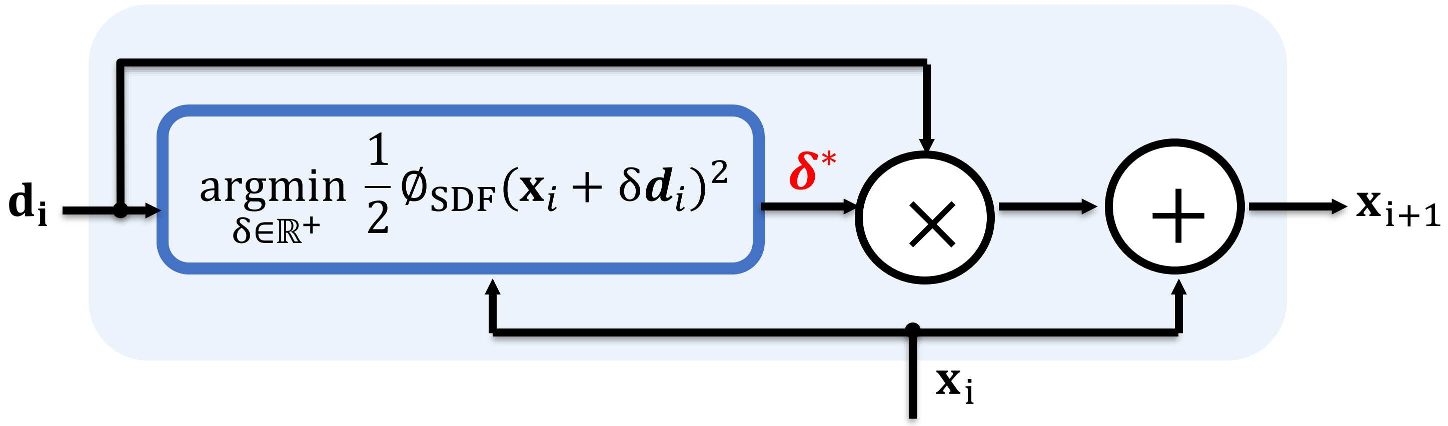


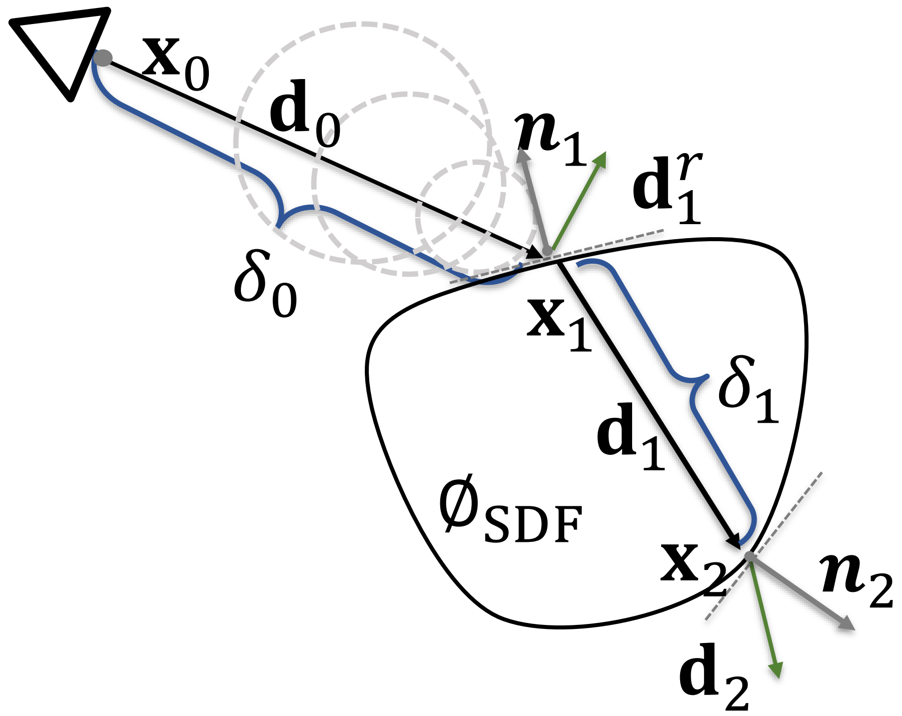



# NEURAL SURFACE REFIMENT

## Analytic Derivatives

### Intersection node



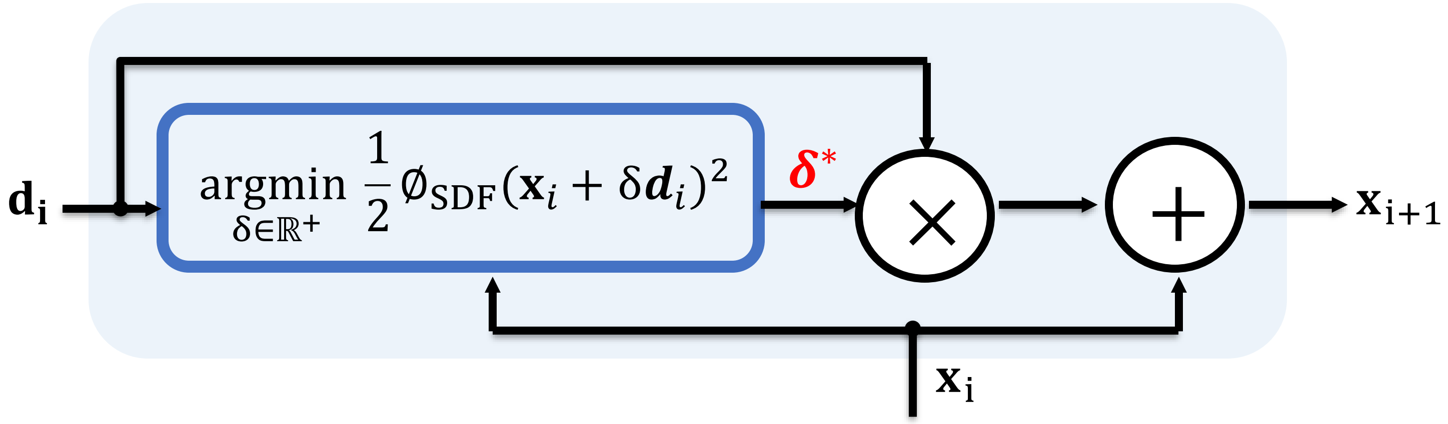

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \delta_i \mathbf{d}_i$$

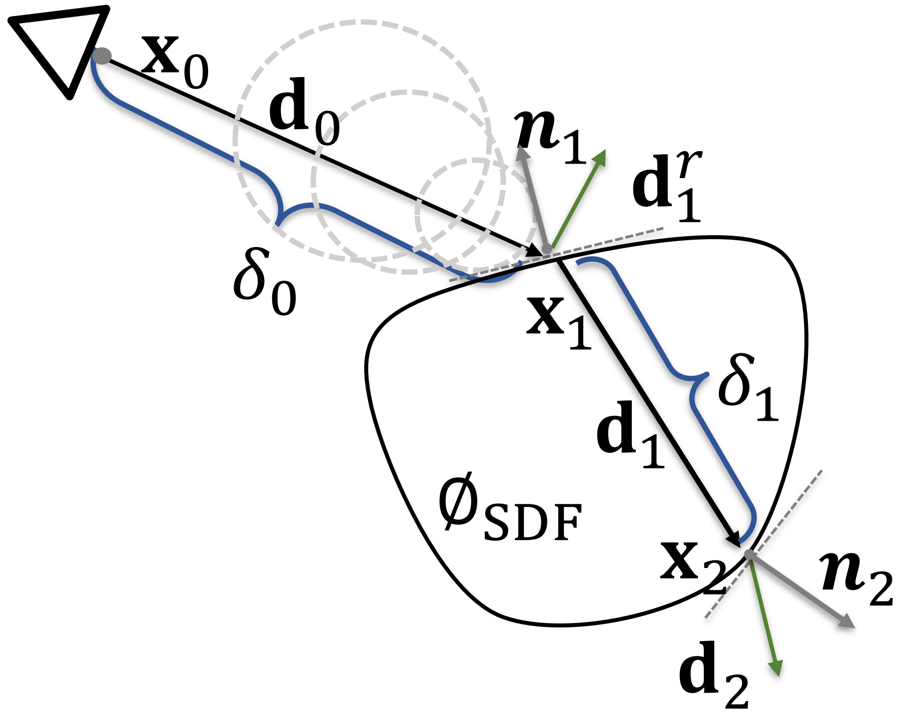
$$\mathbf{n}_{i+1} = \frac{\mathrm{d}}{\mathrm{d}\mathbf{x}} \phi_{\mathrm{SDF}}(\mathbf{x}_{i+1})$$

# NEURAL SURFACE REFIMENT

## Analytic Derivatives

### Intersection node




$$\mathbf{x}_{i+1} = \mathbf{x}_i + \delta_i \mathbf{d}_i$$

$$\mathbf{n}_{i+1} = \frac{\mathrm{d}}{\mathrm{d}\mathbf{x}} \phi_{\mathrm{SDF}}(\mathbf{x}_{i+1})$$

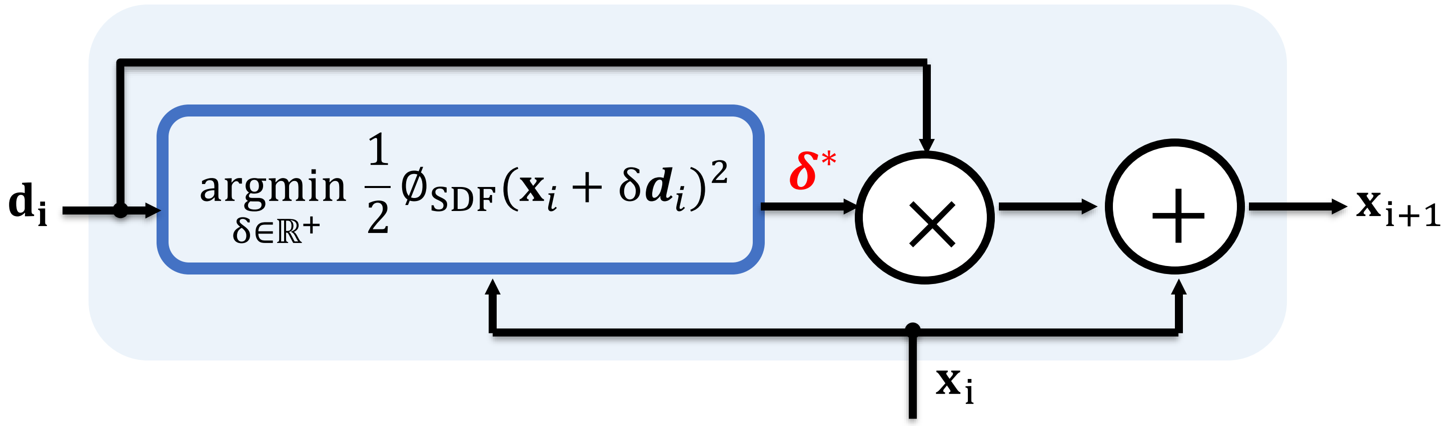
## Proposition 4.6 from Gould et al. (2021)†

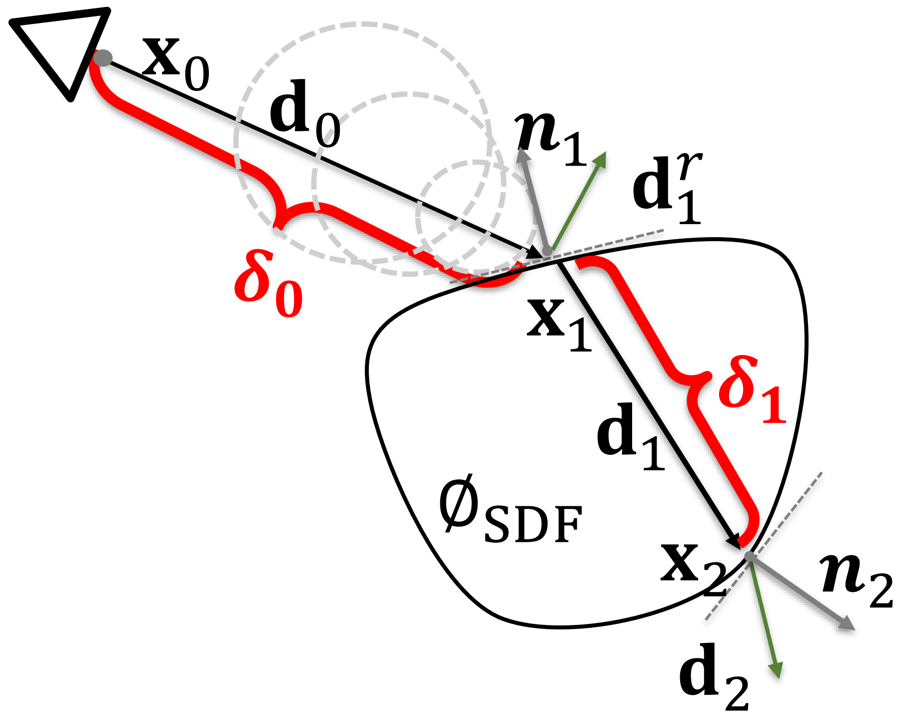
†*Gould etal., "Deep declarative networks." In TPAMI, 2021*

# NEURAL SURFACE REFIMENT

## Analytic Derivatives

### Intersection node




## Proposition 4.6 from Gould et al. (2021)†

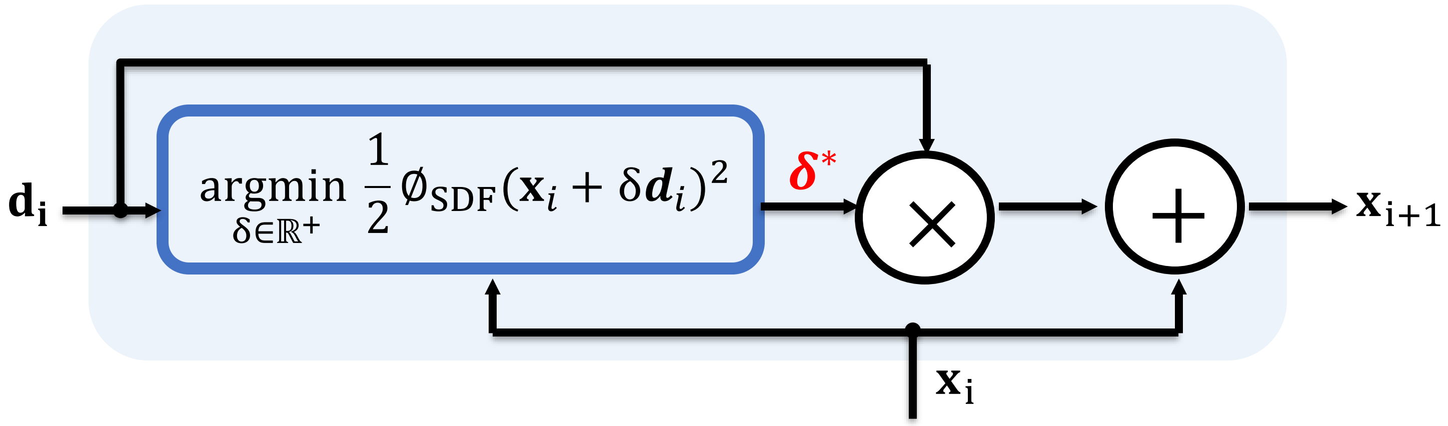
- The derivative of distance on the SDF parameters

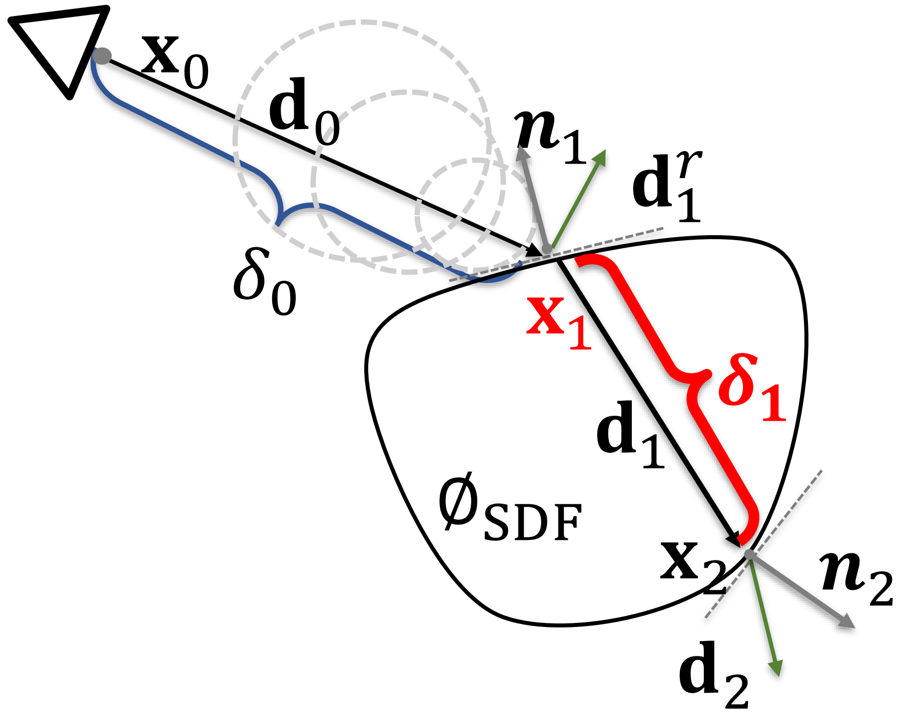
$$\frac{\mathrm{d}\delta_i}{\mathrm{d}\boldsymbol{\theta}} = -\frac{1}{\mathbf{n}_{i+1}^{\top}\mathbf{d}_i} \frac{\partial}{\partial \boldsymbol{\theta}} \phi_{\mathrm{SDF}}(\mathbf{x}_{i+1}; \boldsymbol{\theta})$$

# NEURAL SURFACE REFIMENT

## Analytic Derivatives

### Intersection node




## Proposition 4.6 from Gould et al. (2021)†

- The derivative of distance on the SDF parameters

$$\frac{\mathrm{d}\delta_i}{\mathrm{d}\boldsymbol{\theta}} = -\frac{1}{\mathbf{n}_{i+1}^\mathsf{T}\mathbf{d}_i}\frac{\partial}{\partial\boldsymbol{\theta}}\phi_{\mathrm{SDF}}(\mathbf{x}_{i+1};\boldsymbol{\theta})$$

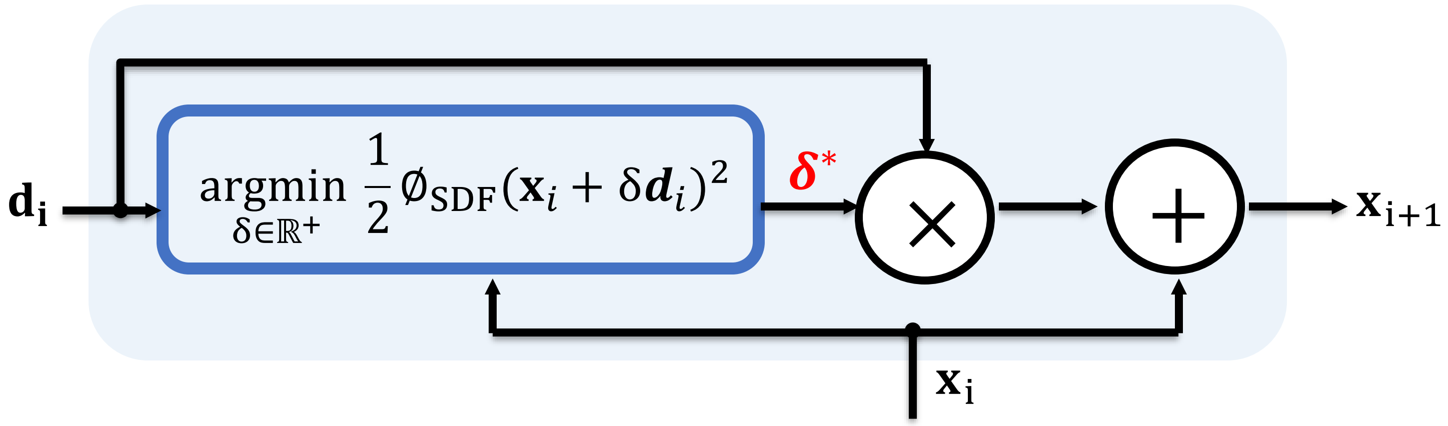
- <u>The impact of starting point on the distance</u>

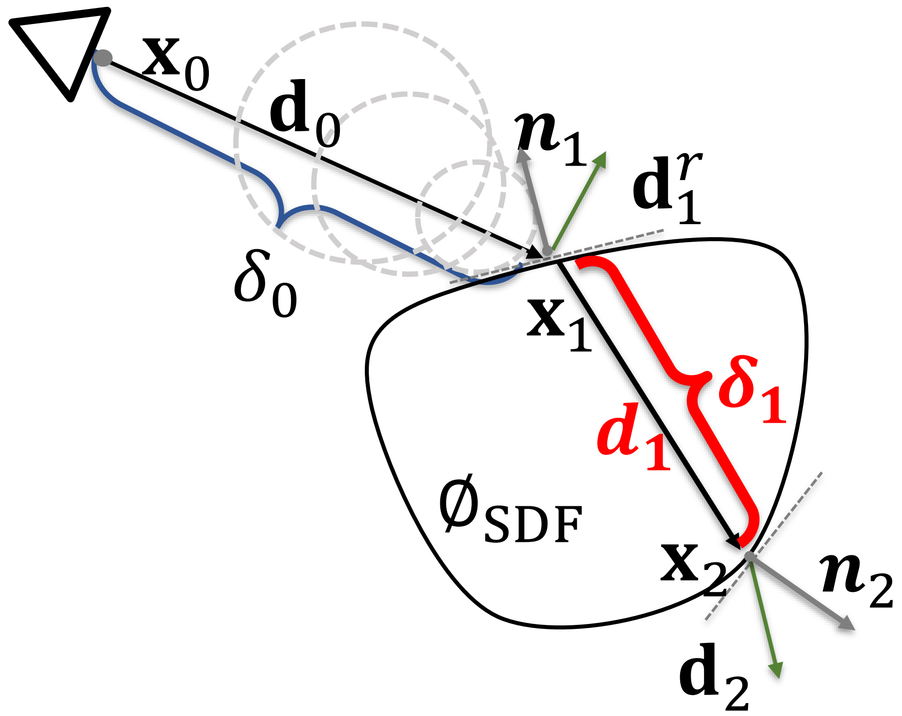
$$\frac{\mathrm{d}\delta_i}{\mathrm{d}\mathbf{x}_i} = -\frac{\mathbf{n}_{i+1}}{\mathbf{n}_{i+1}^\mathsf{T}\mathbf{d}_i}$$

# NEURAL SURFACE REFIMENT

## Analytic Derivatives

### Intersection node




## Proposition 4.6 from Gould et al. (2021)†

- The derivative of distance on the SDF parameters

$$\frac{\mathrm{d}\delta_i}{\mathrm{d}\boldsymbol{\theta}} = -\frac{1}{\mathbf{n}_{i+1}^\top \mathbf{d}_i}\frac{\partial}{\partial\boldsymbol{\theta}}\phi_{\mathrm{SDF}}(\mathbf{x}_{i+1};\boldsymbol{\theta})$$

- The impact of starting point on the distance

$$\frac{\mathrm{d}\delta_i}{\mathrm{d}\mathbf{x}_i} = -\frac{\mathbf{n}_{i+1}}{\mathbf{n}_{i+1}^\top \mathbf{d}_i}$$

- <u>The impact of starting direction on the distance</u>

$$\frac{\mathrm{d}\delta_i}{\mathrm{d}\mathbf{d}_i} = -\frac{\delta_i \mathbf{n}_{i+1}}{\mathbf{n}_{i+1}^\top \mathbf{d}_i}$$

# NEURAL SURFACE REFIMENT

## Analytic Derivatives

### Python implementation

Python Code

```
class DistToIntersection(torch.autograd.Function):
 def forward(ctx, si, xi, di, nj, deltai):
  ctx.save_for_backward(di, nj, deltai)
  return deltai
 def backward(ctx, grad_output):
  di, nj, deltai = ctx.saved_tensors
  return -grad_output / dot(nj, di),
   -grad_output * nj / dot(nj, di),
   -grad_output * deltai * nj / dot(nj, di),
   None, None
```

End-to-end optimization solely on multi-view RGB images

## Proposition 4.6 from Gould et al. (2021)†

- The derivative of distance on the SDF parameters

$$\frac{\mathrm{d}\delta_i}{\mathrm{d}\boldsymbol{\theta}} = -\frac{1}{\mathbf{n}_{i+1}^{\mathsf{T}}\mathbf{d}_i}\frac{\partial}{\partial\boldsymbol{\theta}}\phi_{\mathrm{SDF}}(\mathbf{x}_{i+1};\boldsymbol{\theta})$$

- The impact of starting point on the distance

$$\frac{\mathrm{d}\delta_i}{\mathrm{d}\mathbf{x}_i} = -\frac{\mathbf{n}_{i+1}}{\mathbf{n}_{i+1}^{\mathsf{T}}\mathbf{d}_i}$$

- The impact of starting direction on the distance

$$\frac{\mathrm{d}\delta_i}{\mathrm{d}\mathbf{d}_i} = -\frac{\delta_i\mathbf{n}_{i+1}}{\mathbf{n}_{i+1}^{\mathsf{T}}\mathbf{d}_i}$$

# M E R I T S

√ Uncontrolled setup

√ Unknown Refractive index

√ Unknown geometry

√ Unknown and close background

# MERITS OVER PRIOR WORK

Controlled Setup[1,4]

Known refractive index[2,3]

Known geometry[2]

Infinitely distant background[5]


*1. Lyu et al., Differentiable refraction-tracing for mesh reconstruction of transparent objects. In ACM Trans. Graph. 2020*

*2. Pan et al., Sampling neural radiance fields for refractive objects. In SIGGRAPH Asia 2022 Technical Communications, 2022*

*3. Li, et al., Through the looking glass: Neural 3d reconstruction of transparent shapes. In CVPR 2020*

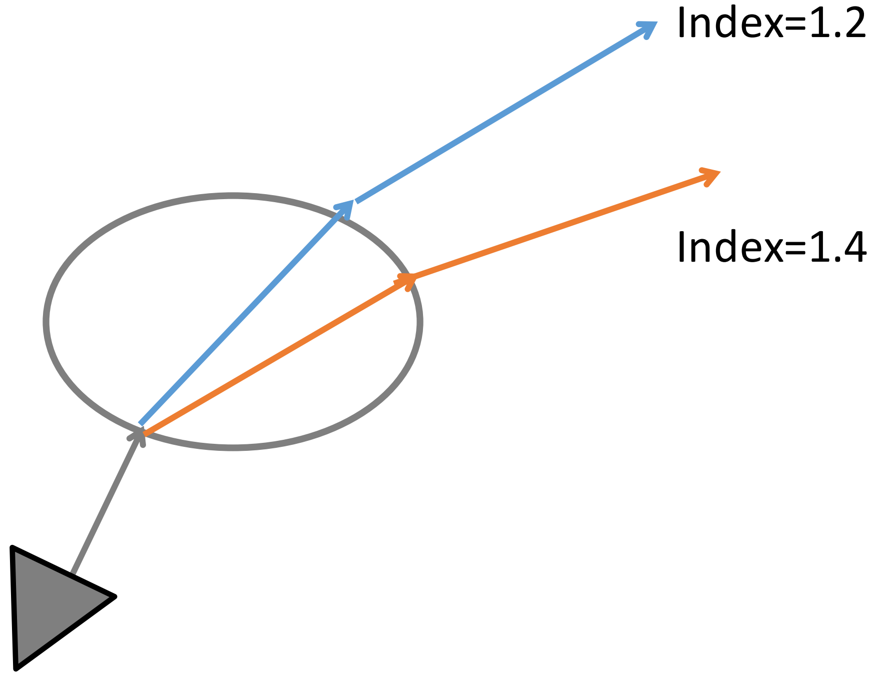
*4. Li et al., Neto: Neural reconstruction of transparent objects with self-occlusion aware refraction-tracing. ICCV 2023*

*5. Wang et al., Nemto: Neural environment matting for novel view and relighting synthesis of transparent objects. In ICCV 2023*

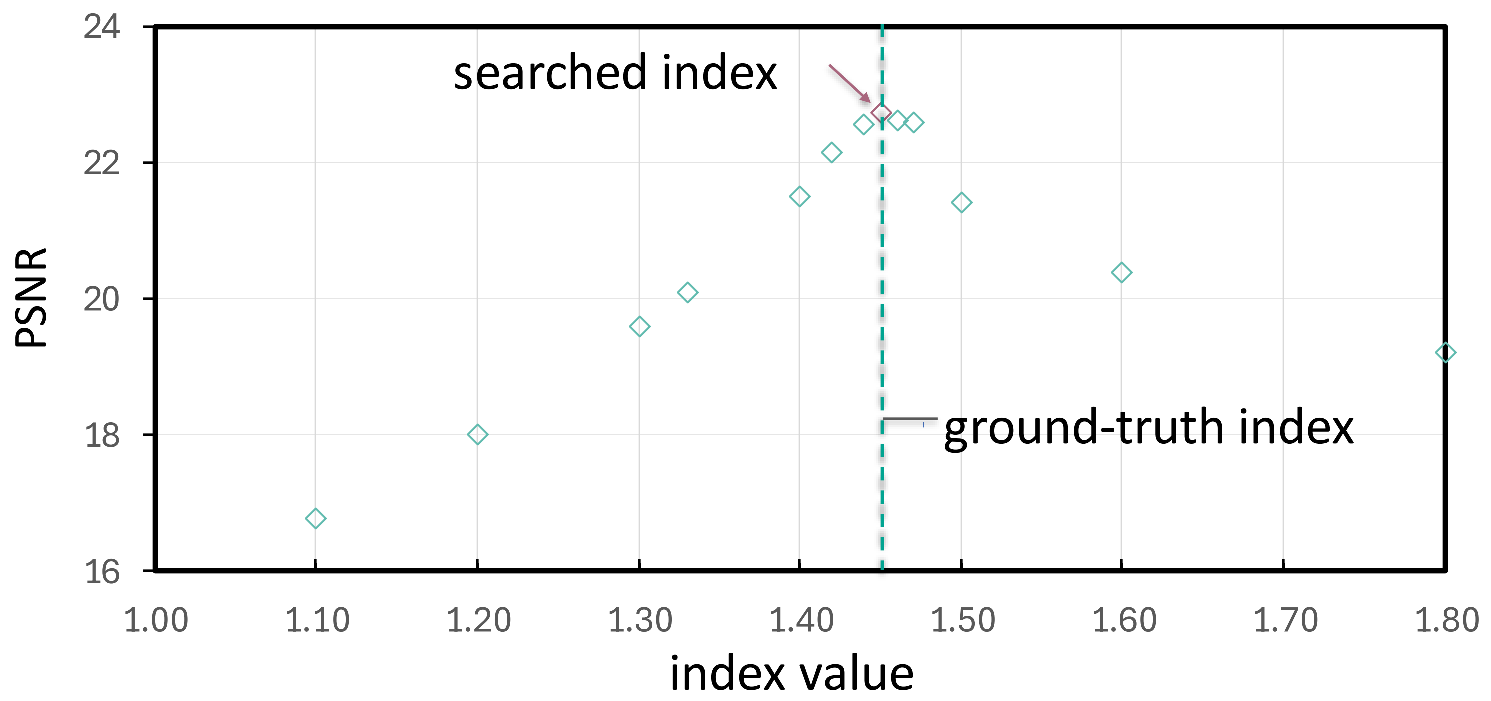
# E X P E R I M E N T S

## Refractive Index Search

Given initial surface, search an index that gives the highest PSNR



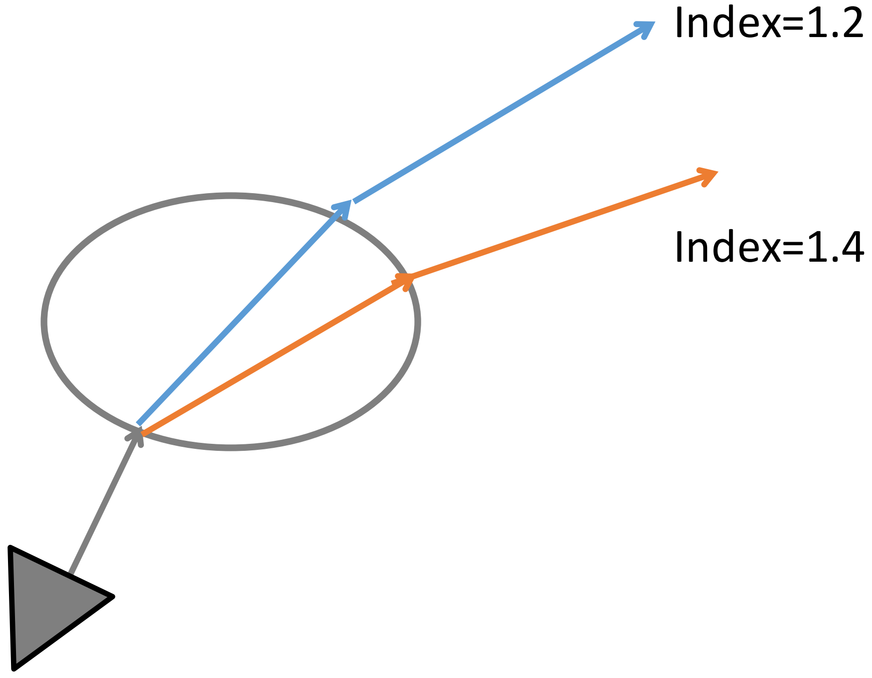

Optical Ball



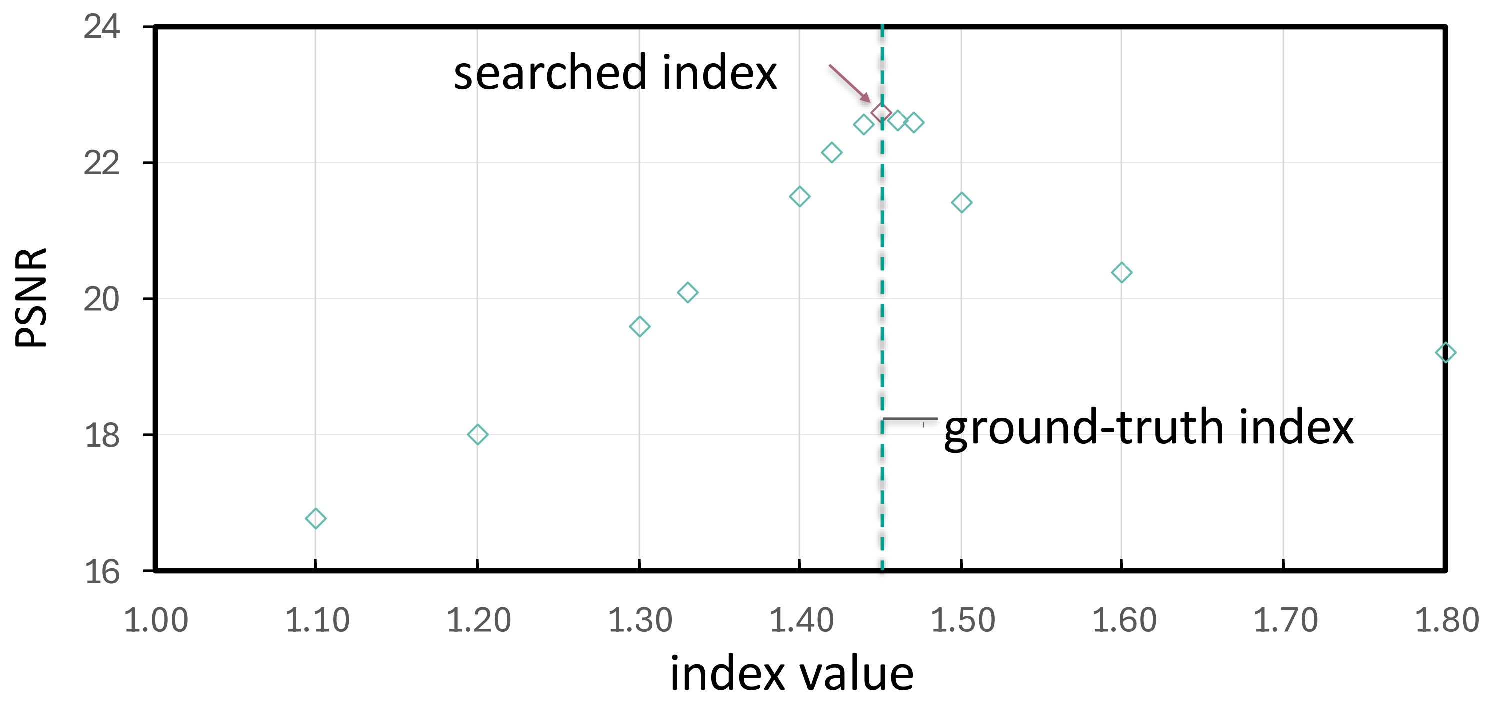
# EXPERIMENTS

## Refractive Index Search

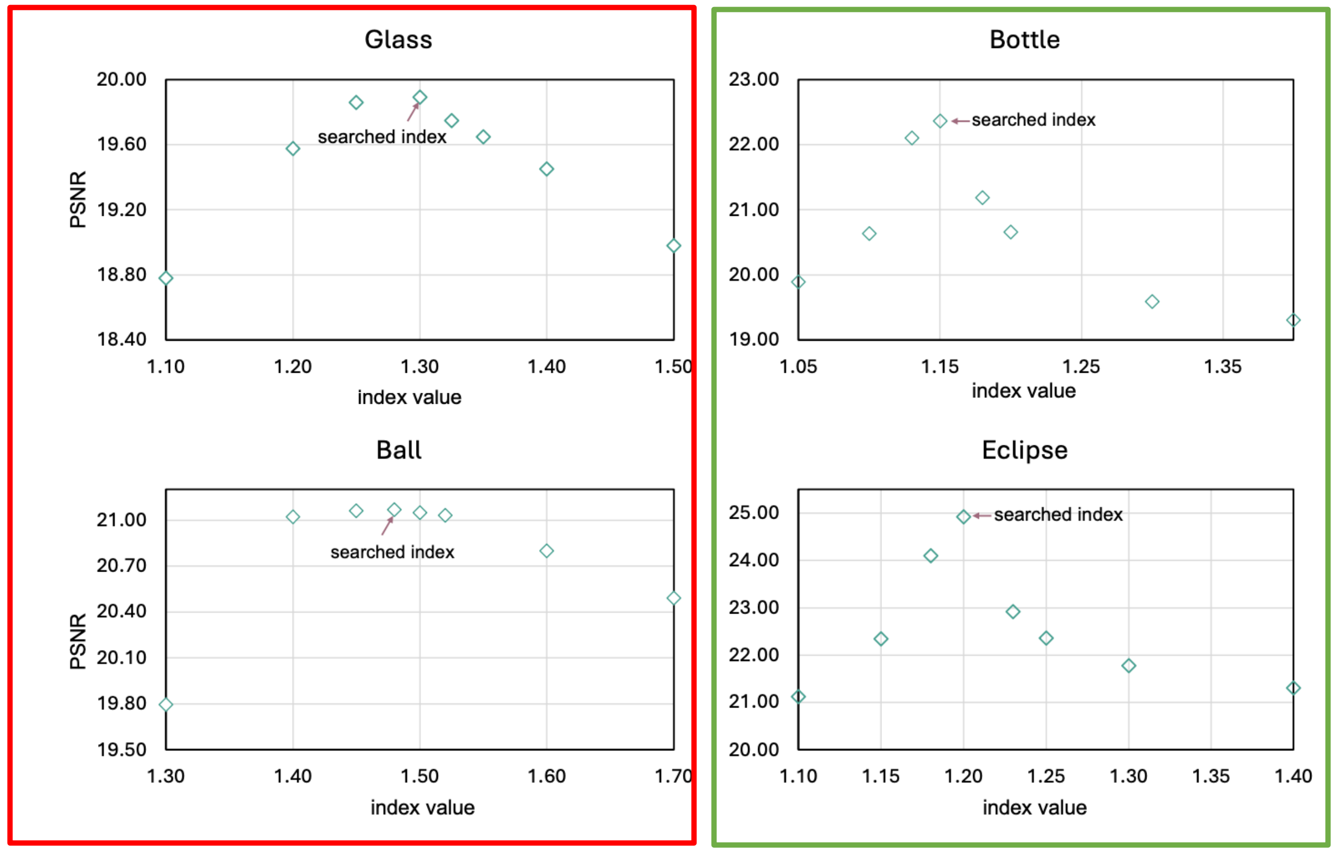
Given initial surface, search an index that gives the highest PSNR




Optical Ball




Real-world datasets (unknown refractive index)




Synthetic datasets (estimated refractive indices match the ground truths)

# NOVEL VIEW SYNTHESIS

| Model | Ball [1] | | | Glass [1] | | | Optical Ball | | | Bottle | | | Kitty | | | Ellipsoid | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS |
| TensoRF [5] | 21.41 | 0.735 | 0.187 | 20.49 | 0.695 | 0.226 | 22.42 | 0.806 | 0.327 | 20.76 | 0.786 | 0.267 | 19.37 | 0.782 | 0.384 | 22.45 | 0.850 | 0.195 |
| Instant-NGP [22] | 21.56 | 0.790 | 0.121 | 21.42 | 0.748 | 0.148 | 20.93 | 0.800 | 0.330 | 20.48 | 0.784 | 0.296 | 19.51 | 0.783 | 0.362 | 23.61 | 0.886 | 0.146 |
| Nerfacto [30] | 21.67 | 0.797 | 0.113 | **22.14** | 0.774 | 0.121 | 20.73 | 0.794 | 0.300 | 20.94 | 0.828 | 0.196 | 19.14 | 0.782 | 0.309 | 23.39 | 0.894 | 0.128 |
| MS-NeRF [41] | **22.35** | **0.810** | **0.105** | 21.83 | **0.781** | **0.119** | 21.36 | 0.822 | 0.281 | 21.35 | 0.856 | 0.161 | 19.57 | 0.800 | 0.240 | 23.01 | 0.906 | 0.110 |
| Eikonal Fields [1] | 21.64 | 0.699 | 0.217 | 20.92 | 0.663 | 0.262 | 13.17 | 0.511 | 0.507 | 13.82 | 0.404 | 0.485 | 13.82 | 0.649 | 0.529 | 13.76 | 0.617 | 0.521 |
| NeuS [35] | 22.24 | 0.780 | 0.129 | 21.95 | 0.754 | 0.136 | 22.84 | 0.812 | 0.249 | 21.13 | 0.859 | 0.166 | 19.43 | 0.805 | 0.271 | 23.36 | 0.894 | 0.128 |
| NeuS+Ref. | 21.15 | 0.773 | 0.128 | 20.15 | 0.745 | 0.141 | 22.57 | 0.815 | 0.127 | 22.59 | 0.854 | 0.115 | 18.76 | 0.805 | 0.170 | 25.21 | 0.902 | 0.073 |
| Ours | 21.80 | 0.785 | **0.105** | 21.30 | 0.754 | 0.120 | **24.07** | **0.826** | **0.112** | **23.20** | **0.866** | **0.084** | **19.61** | **0.812** | **0.163** | **25.24** | **0.915** | **0.061** |

*A higher PSNR/ SSIM denotes higher performance*

*A Lower LPIPS denotes higher performance*

*Please see our paper for references*

Our method gains competitive results compared with other methods

# NOVEL VIEW SYNTHESIS

| Model | Ball [1] | | | Glass [1] | | | Optical Ball | | | Bottle | | | Kitty | | | Ellipsoid | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS | PSNR | SSIM | LPIPS |
| TensoRF [5] | 21.41 | 0.735 | 0.187 | 20.49 | 0.695 | 0.226 | 22.42 | 0.806 | 0.327 | 20.76 | 0.786 | 0.267 | 19.37 | 0.782 | 0.384 | 22.45 | 0.850 | 0.195 |
| Instant-NGP [22] | 21.56 | 0.790 | 0.121 | 21.42 | 0.748 | 0.148 | 20.93 | 0.800 | 0.330 | 20.48 | 0.784 | 0.296 | 19.51 | 0.783 | 0.362 | 23.61 | 0.886 | 0.146 |
| Nerfacto [30] | 21.67 | 0.797 | 0.113 | **22.14** | 0.774 | 0.121 | 20.73 | 0.794 | 0.300 | 20.94 | 0.828 | 0.196 | 19.14 | 0.782 | 0.309 | 23.39 | 0.894 | 0.128 |
| MS-NeRF [41] | **22.35** | **0.810** | **0.105** | 21.83 | **0.781** | **0.119** | 21.36 | 0.822 | 0.281 | 21.35 | 0.856 | 0.161 | 19.57 | 0.800 | 0.240 | 23.01 | 0.906 | 0.110 |
| Eikonal Fields [1] | 21.64 | 0.699 | 0.217 | 20.92 | 0.663 | 0.262 | 13.17 | 0.511 | 0.507 | 13.82 | 0.404 | 0.485 | 13.82 | 0.649 | 0.529 | 13.76 | 0.617 | 0.521 |
| NeuS [35] | 22.24 | 0.780 | 0.129 | 21.95 | 0.754 | 0.136 | 22.84 | 0.812 | 0.249 | 21.13 | 0.859 | 0.166 | 19.43 | 0.805 | 0.271 | 23.36 | 0.894 | 0.128 |
| NeuS+Ref. | 21.15 | 0.773 | 0.128 | 20.15 | 0.745 | 0.141 | 22.57 | 0.815 | 0.127 | 22.59 | 0.854 | 0.115 | 18.76 | 0.805 | 0.170 | 25.21 | 0.902 | 0.073 |
| Ours | 21.80 | 0.785 | **0.105** | 21.30 | 0.754 | 0.120 | **24.07** | **0.826** | **0.112** | **23.20** | **0.866** | **0.084** | **19.61** | **0.812** | **0.163** | **25.24** | **0.915** | **0.061** |

Our method gains competitive results compared with other methods

- *A higher PSNR/ SSIM denotes higher performance*
- *A Lower LPIPS denotes higher performance*
- *Please see our paper for references*

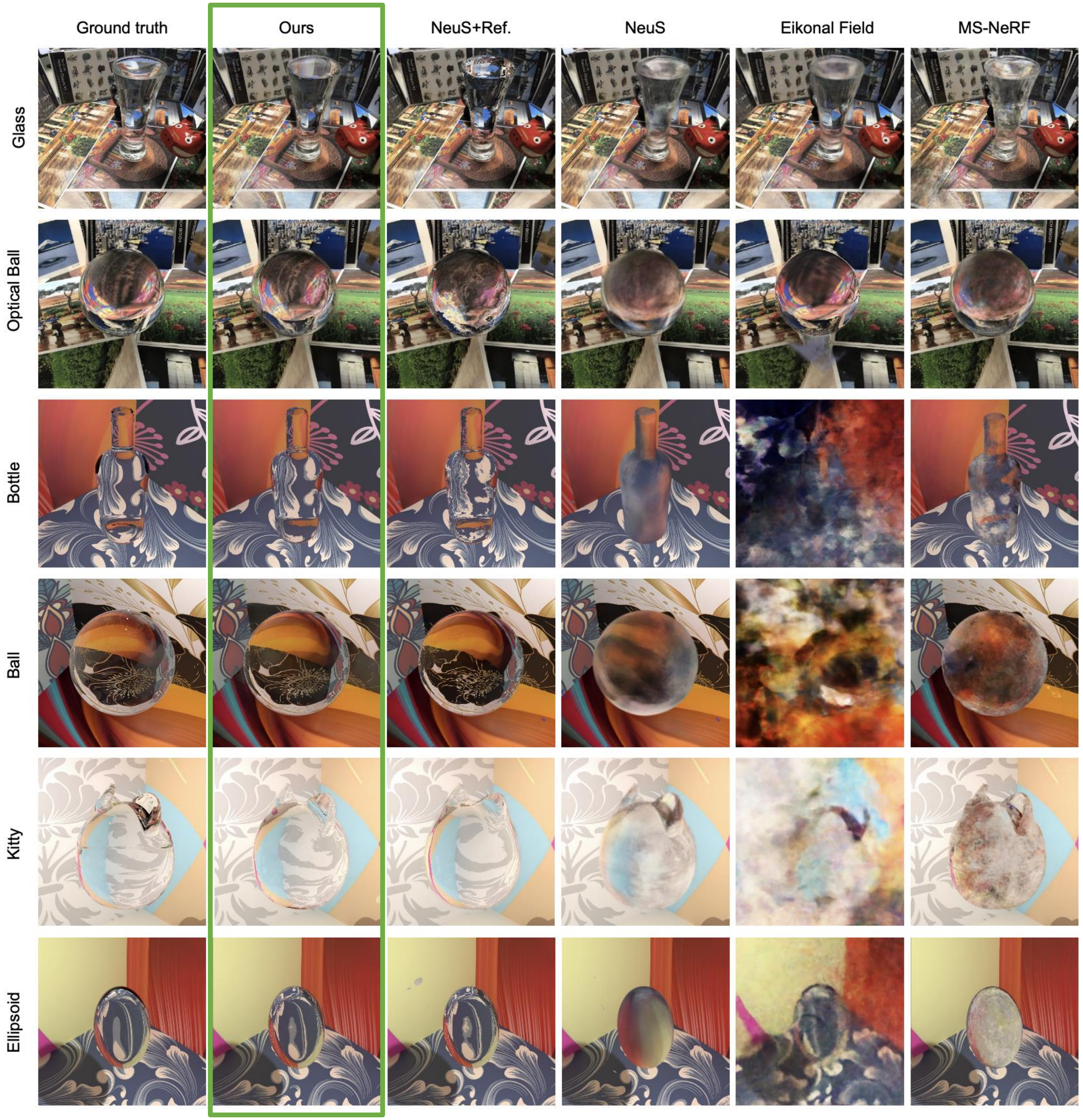
# 3D RECONSTRUCTION

Our method achieves better performance

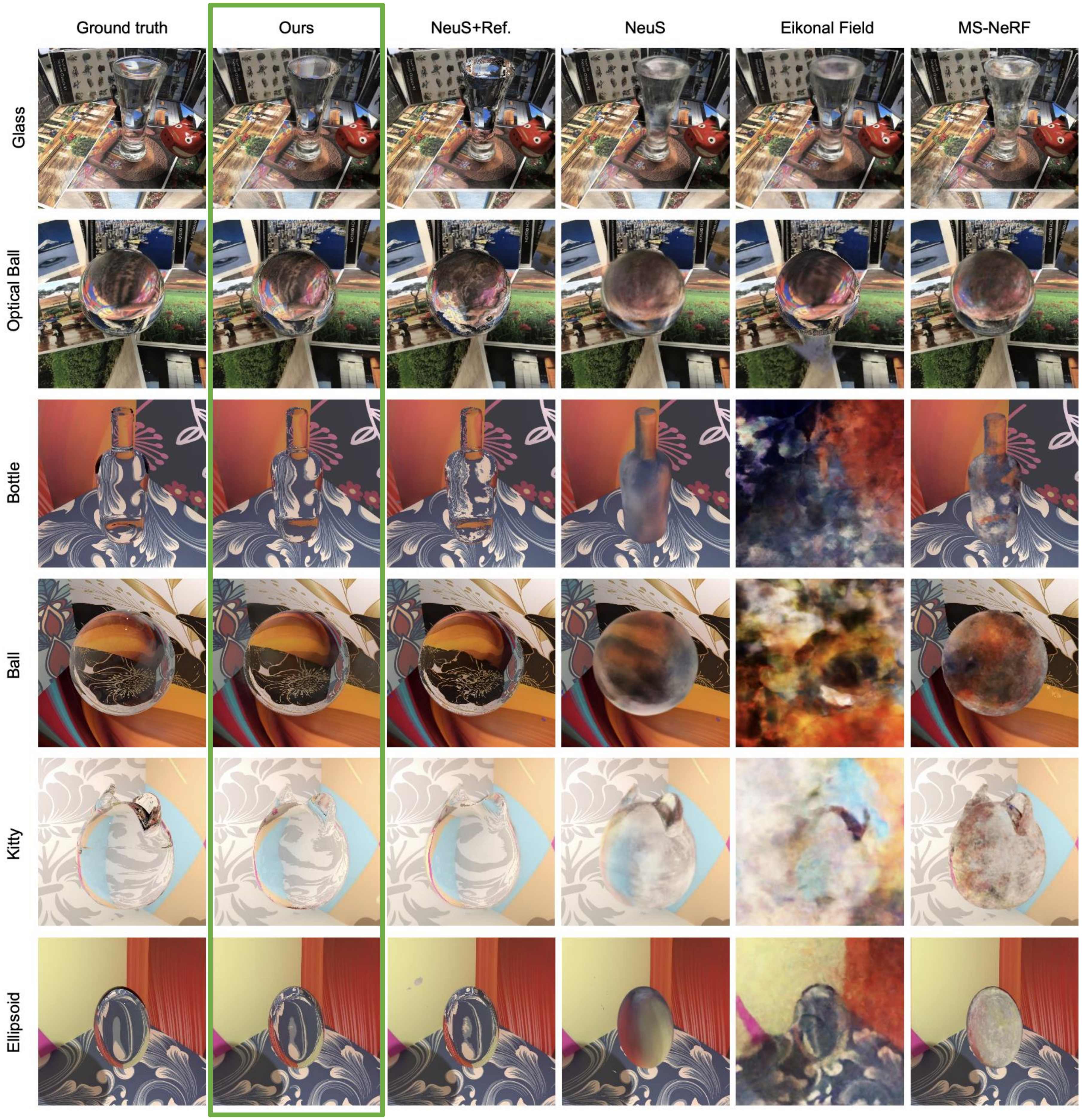
| Model | ↓Chamfer $L_1 (\times 10^{-3})$ Optical Ball | Bottle | Kitty | Ellipsoid |
|---|---|---|---|---|
| Nerfacto [30] | 99.67 | 56.37 | 31.79 | 5.16 |
| UNISURF [23] | 2.08 | 33.10 | – | 1.24 |
| VolSDF [40] | 1.98 | 32.01 | – | 1.15 |
| NeuS [35] | 1.96 | 31.05 | 19.28 | 1.05 |
| Ours | **1.91** | **29.54** | **18.05** | **0.80** |

- *A Lower Chamfer $L_1$ denotes higher performance*

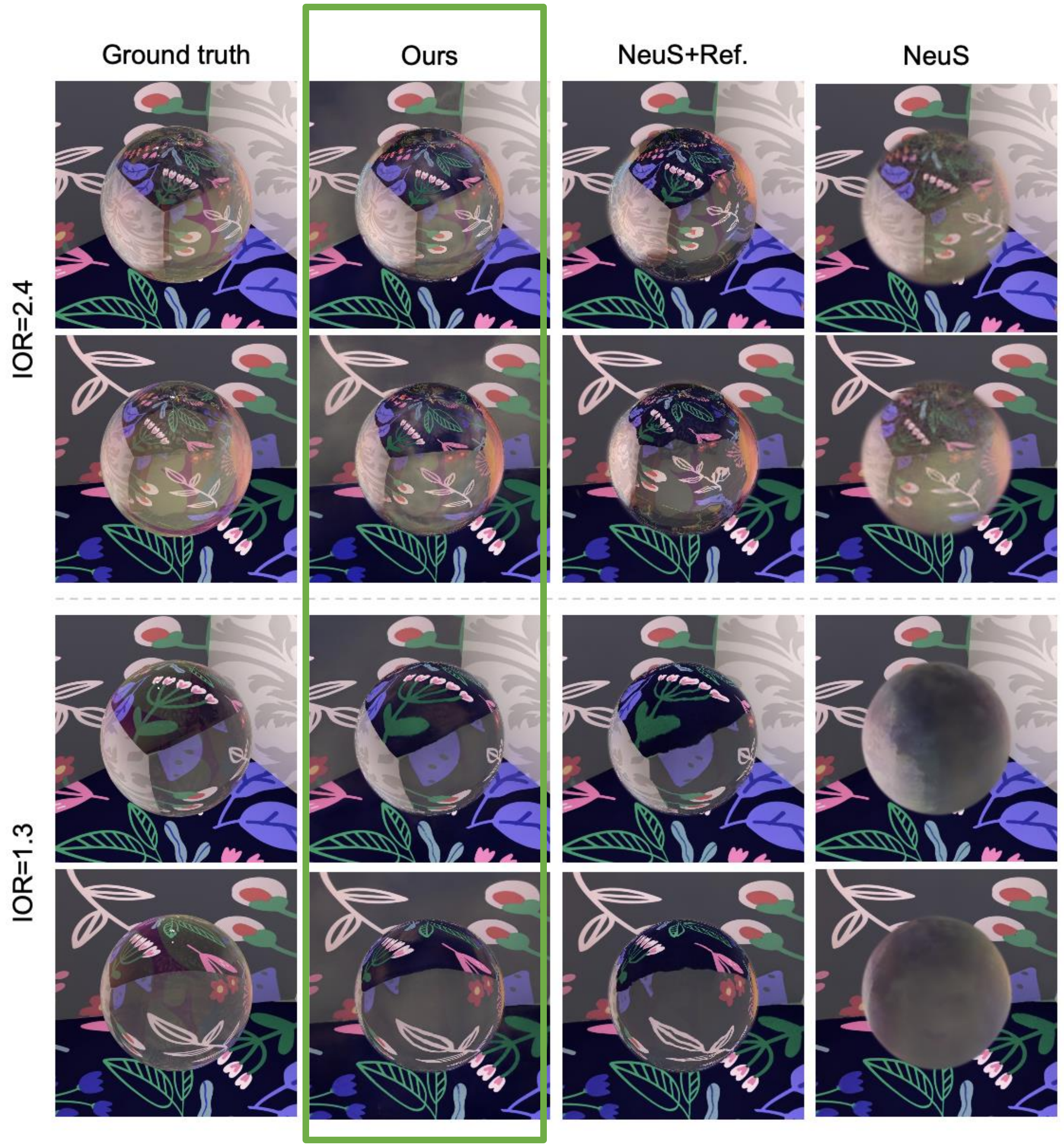
# ENHANCED NOVEL VIEW SYNTHESIS



# ENHANCED NOVEL VIEW SYNTHESIS

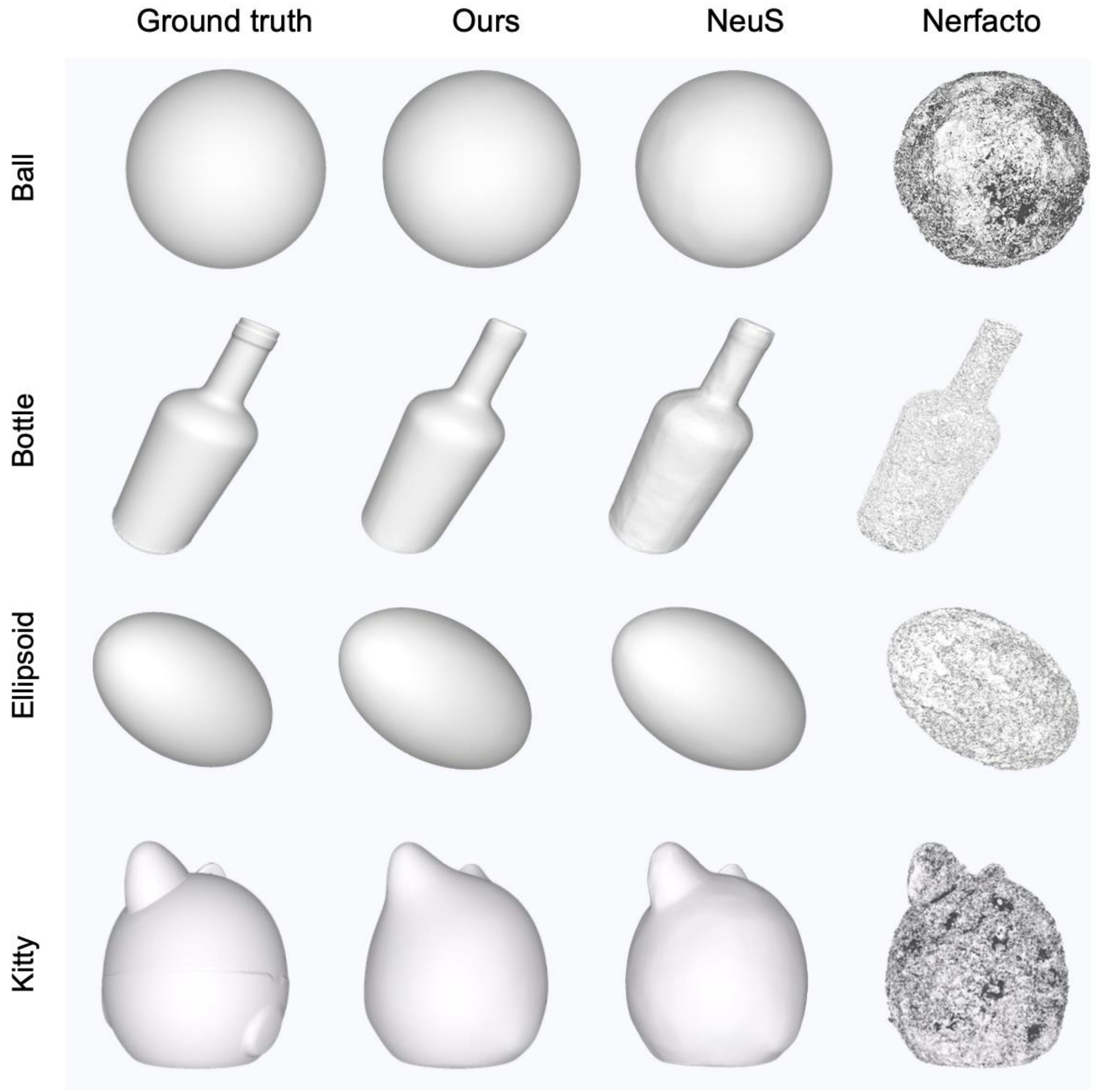



Our method handles various refractive indices




Deng et al, CVPR 2024

# IMPROVED 3D RECONSTRUCTION




Our method offers smoother reconstructions compared to NeuS, suggesting improved modeling

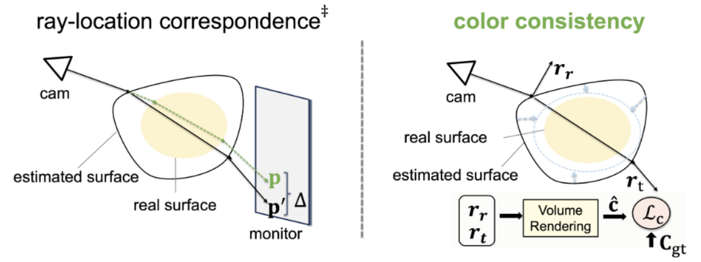
# PROJECT PAGE

## Neural Surface Refinement for Modeling Transparent Objects

Weijian Deng[1], Dylan Campbell[1], Chunyi Sun[1],
Shubham Kanitkar[2], Matthew E. Shaffer[2], Stephen Gould[1],
[1]Australian National University, [2]RIOS Intelligent Machines

**CVPR'24** [Paper] [GitHub]

❑ **Our Solution** Given an initial surface learned by NeuS*, use color consistency for refinement




❑ **Merits**
- ✅ Uncontrolled setup
- ✅ Unknown geometry
- ✅ Unknown Refractive index
- ✅ Unknown and close background

We present Transparent Neural Surface Refinement (TNSR), which reconstructs transparent surfaces using only color consistency from multi-view RGB images, significantly improving geometry estimation and view synthesis.

## Abstract

Neural implicit surface reconstruction leveraging volume rendering has led to significant advances in multi-view reconstruction. However, results for transparent objects can be very poor, primarily because the rendering function fails to account for the intricate light transport induced by refraction and reflection. In this study, we introduce transparent neural surface refinement (TNSR), a novel surface reconstruction framework that explicitly incorporates physical refraction and reflection tracing. Beginning with an initial, approximate surface, our method employs sphere tracing combined with Snell's law to cast both reflected and refracted rays. Central to our proposal is an innovative differentiable technique devised to allow signals from the photometric evidence to propagate back to the surface model by considering how the surface bends and reflects light rays. This allows us to connect surface refinement with volume rendering, enabling end-to-end optimization solely on multi-view RGB images. In our experiments, TNSR demonstrates significant improvements in novel view synthesis and geometry estimation of transparent objects, without prior knowledge of the refractive index.

https://weijiandeng.xyz/nsr/

Paper, code, and datasets

# THANK YOU!