

Self-Calibrating Neural Radiance Fields

Yoonwoo Jeong, Seokjun Ahn, Christopher Choy, Animashree
Anandkumar, Minsu Cho, Jaesik Park

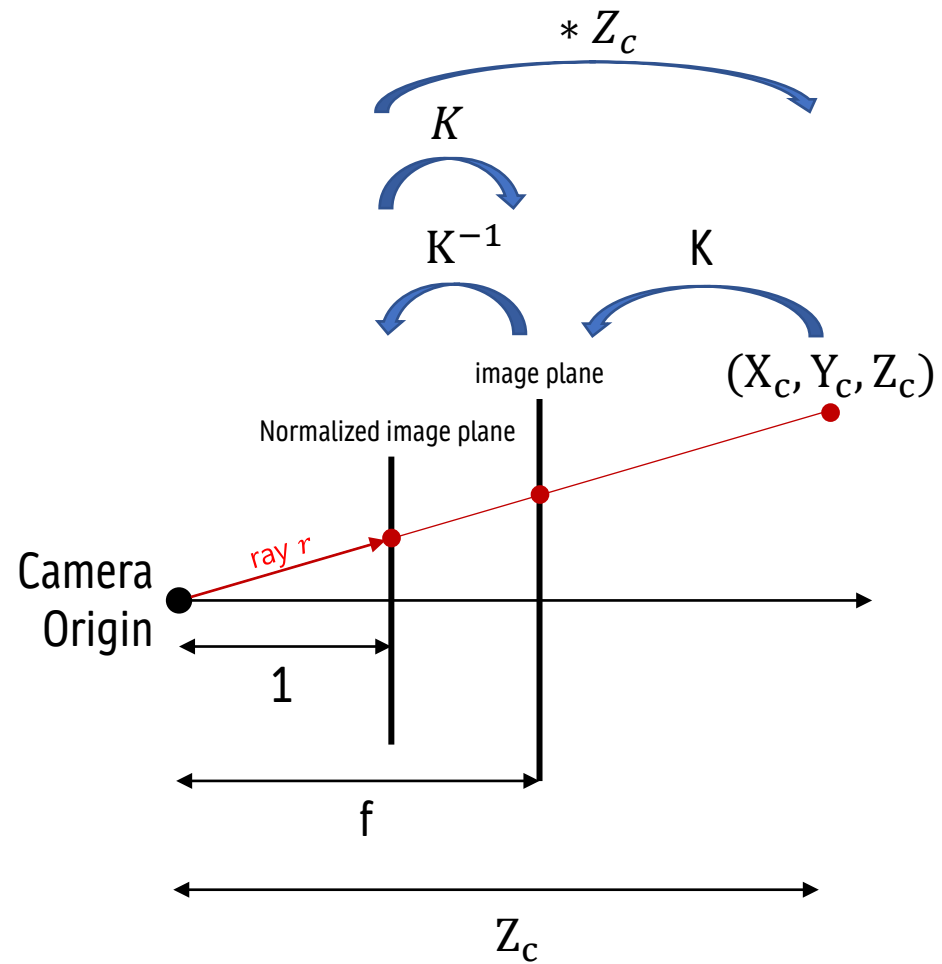
발표자 : 김민정

Contents

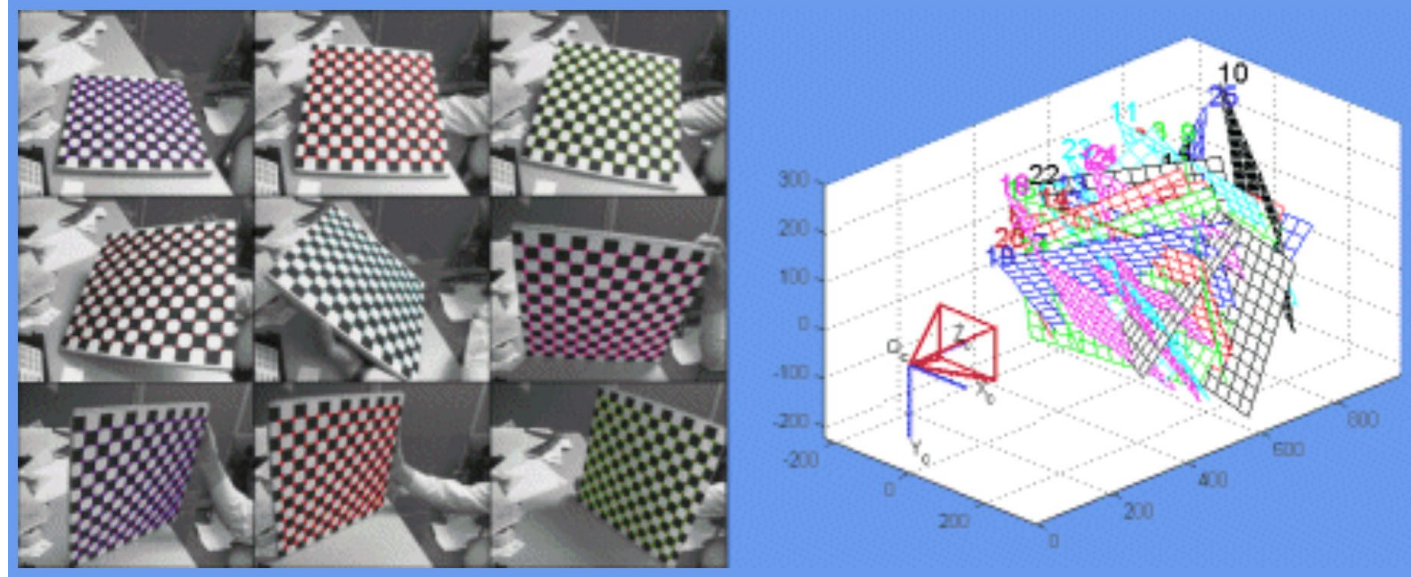
- Introduction
 - Camera Calibration.. Why?
 - Related Work
 - Problem Definition
- Method
 - Differentiable Camera Model
 - Geometric/Photometric Consistency Loss
 - Curriculum Learning
- Experiment & Result
- Conclusion
- Reference

Introduction

Camera Calibration.. Why?



Camera Calibration Tool



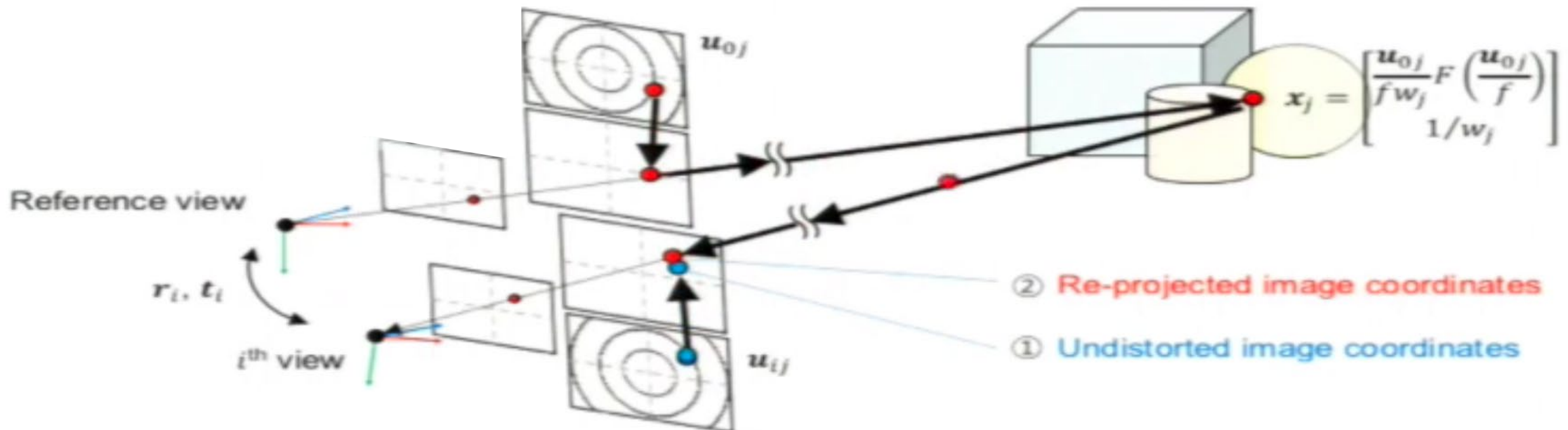
Related Work | Self-Calibration Algorithms

Pros calibrates camera parameters without an external calibration object (e.g., a checkerboard pattern)

Cons Use only a sparse set of image correspondences

=> Not enough interest points -> diverging results with extreme sensitivity to noise

Do not improve the 3D geometry of the objects



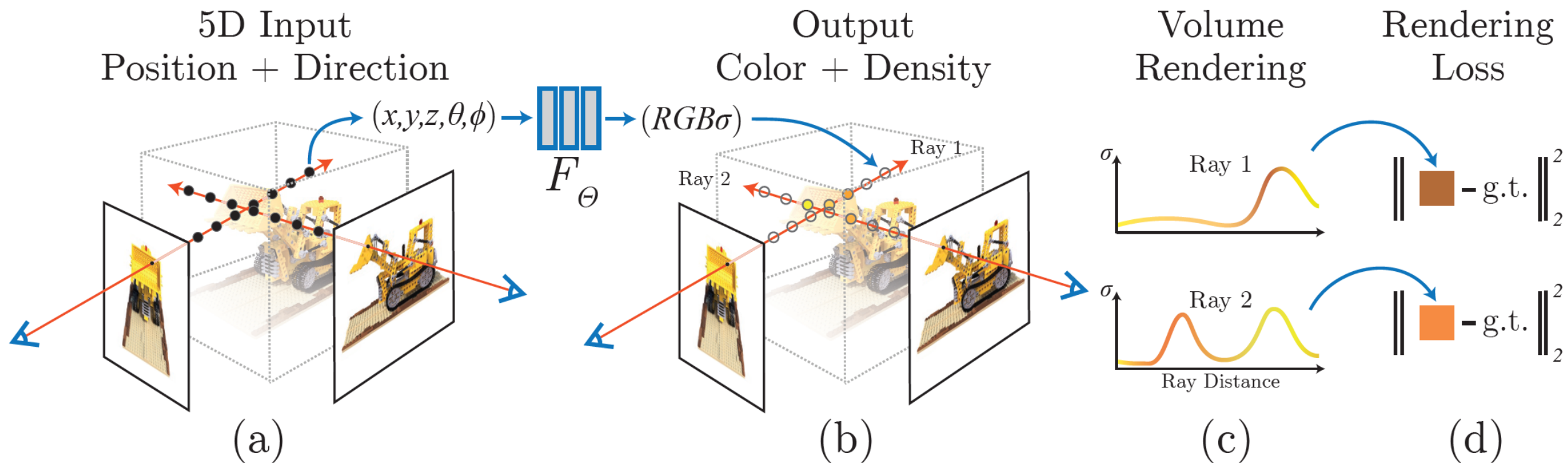
Related Work | NeRF

Pros Synthesize novel views by learning volumetric scene function with multi-layer perceptron

Cons Assumes known camera poses, intrinsics

Example

$$\hat{\mathbf{C}}(\mathbf{r}) \approx \sum_i^N \left(\prod_{j=1}^{i-1} \alpha(\mathbf{r}(t_j), \Delta_j) \right) (1 - \alpha(t_i, \Delta_i)) \mathbf{c}(\mathbf{r}(t_i), \mathbf{v})$$



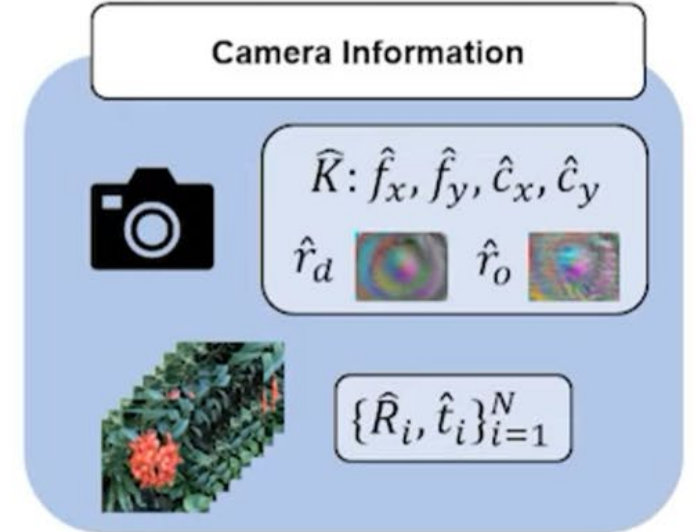
Problem Definition

Goal : Remove the necessity of precomputed camera information

Input Images



Our model



Jointly optimizes
camera parameters &
NeRF parameters

Method

Method

- Differentiable Camera Model
 - Pinhole Camera model
 - fourth order radial distortion
 - Generic non-linear camera distortion
 - Computational graph of ray direction & origin
- Loss
- Curriculum Learning

Differentiable Camera Model | Pinhole Camera Model

$$P'_{3 \times 1} = M_{3 \times 4} P_w = \boxed{K_{3 \times 3} \begin{bmatrix} R & T \end{bmatrix}_{3 \times 4}} P_{w4 \times 1}$$

Find!

- Camera Intrinsic

- $K = \begin{bmatrix} f_x + \Delta f_x & 0 & c_x + \Delta c_x \\ 0 & f_y + \Delta f_y & c_y + \Delta c_y \\ 0 & 0 & 1 \end{bmatrix} = K_0 + \Delta K \in \mathbb{R}^{3 \times 3}$

Initial Values

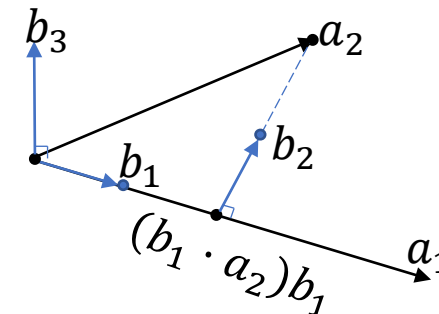
$f_x, f_y = W/2, H/2$
 $c_x, c_y = W/2, H/2$
 $R = I$
 $t = 0$

- Camera Extrinsic

- $\mathbf{t} = \mathbf{t}_0 + \Delta \mathbf{t}$

- $R = f(\mathbf{a}_0 + \Delta \mathbf{a})$

- $f\left(\begin{bmatrix} | & | \\ \mathbf{a}_1 & \mathbf{a}_2 \\ | & | \end{bmatrix}\right) = \begin{bmatrix} | & | & | \\ \mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 \\ | & | & | \end{bmatrix}_{3 \times 3}$

**Gram-Schmidt-like process**

$$\mathbf{b}_1 = N(\mathbf{a}_1)$$

$$\mathbf{b}_2 = N(\mathbf{a}_2 - (\mathbf{b}_1 \cdot \mathbf{a}_2)\mathbf{b}_1)$$

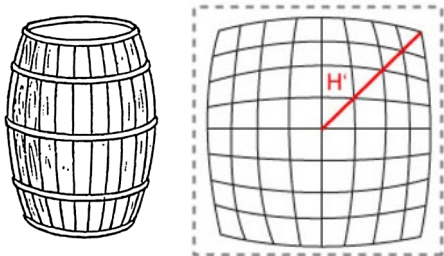
$$\mathbf{b}_3 = \mathbf{b}_1 \times \mathbf{b}_2$$

Differentiable Camera Model | 4th order radial distortion

Radial Distortion

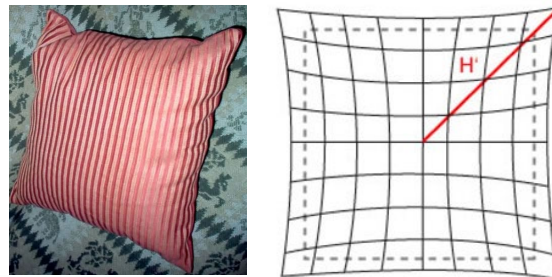
① Barrel (negative)

Concave lens가
일반적으로
barrel distortion가짐

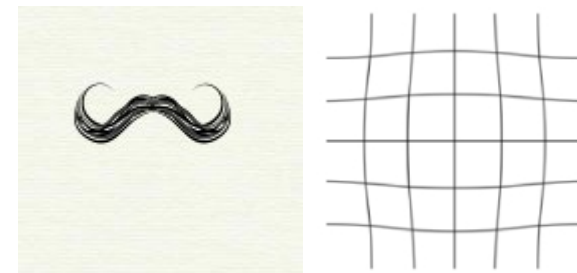


② Pincushion(Positive)

Convex lens가
일반적으로
pincushion
distortion가짐



③ Mustache (Complex)



Modeling

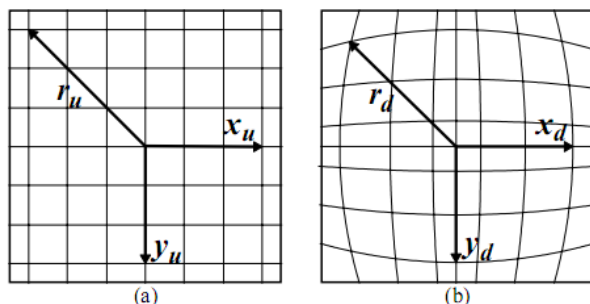


Figure 1: Illustration of barrel distortion model

$$\begin{bmatrix} x_{n_d} \\ y_{n_d} \end{bmatrix} = (1 + k_1 r_u^2 + k_2 r_u^4 + k_3 r_u^6) \begin{bmatrix} x_{n_u} \\ y_{n_u} \end{bmatrix} \quad \text{In general}$$

$r_u^2 = x_{n_u}^2 + y_{n_u}^2$

$$\begin{bmatrix} x_{n_u} \\ y_{n_u} \end{bmatrix} = (1 + k_1 r_d^2 + k_2 r_d^4 + k_3 r_d^6) \begin{bmatrix} x_{n_d} \\ y_{n_d} \end{bmatrix} \quad \text{Makes Sense}$$

$r_d^2 = x_{n_d}^2 + y_{n_d}^2$

Find
residual!

$$\mathbf{k} = (k_1 + z_{k_1}, k_2 + z_{k_2})$$

Differentiable Camera Model | Pinhole + Radial Distortion

What We Want To Find

- Pixel Coordinate $(p_x, p_y) \rightarrow$ Distorted Normalized coordinate (n_x, n_y)

$$(n_x, n_y) = \left(\frac{p_x - c_x}{f_x}, \frac{p_y - c_y}{f_y} \right), r = \sqrt{n_x^2 + n_y^2}$$

논문과 다름. 자체 수정

- Distorted Normalized Coordinate $(n_x, n_y) \rightarrow$ Undistorted Normalized Coordinate (n'_x, n'_y)

$$(n'_x, n'_y) = (n_x(1 + k_1 r^2 + k_2 r^4), n_y(1 + k_1 r^2 + k_2 r^4))$$

$$= K^{-1} [p_x(1 + k_1 r^2 + k_2 r^4), p_y(1 + k_1 r^2 + k_2 r^4), 1]^T$$

- Undistorted Normalized Coordinate $(n'_x, n'_y) \rightarrow$ Undistorted Normalized Coordinate in the World $(n^{w'}, n^{w'})$

$$(n^{w'}, n^{w'}) = R \cdot [n'_x, n'_y, 1]^T + t$$

- Ray direction & ray origin

$$r_d = N(R \cdot [n'_x, n'_y, 1]^T)$$

$$r_o = t$$

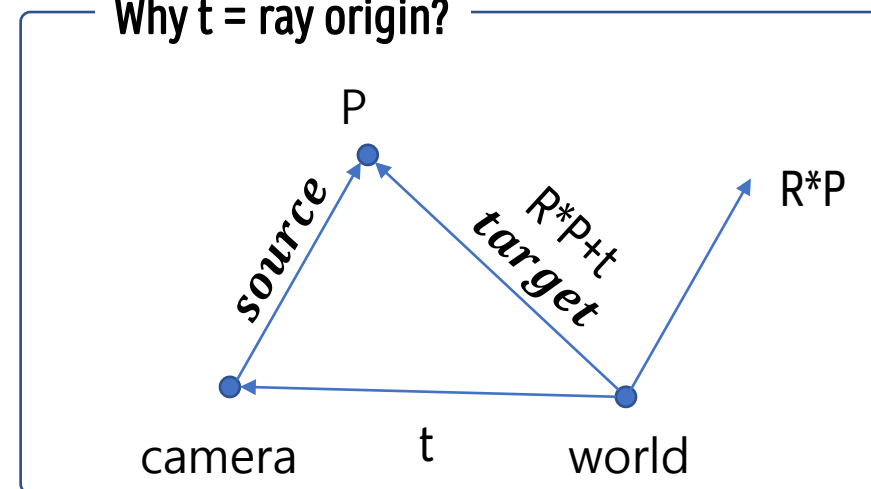
- Sampled Point

$$(r_o + t * r_d)$$

NeRF INPUT!

We can pass the gradients to the residuals!

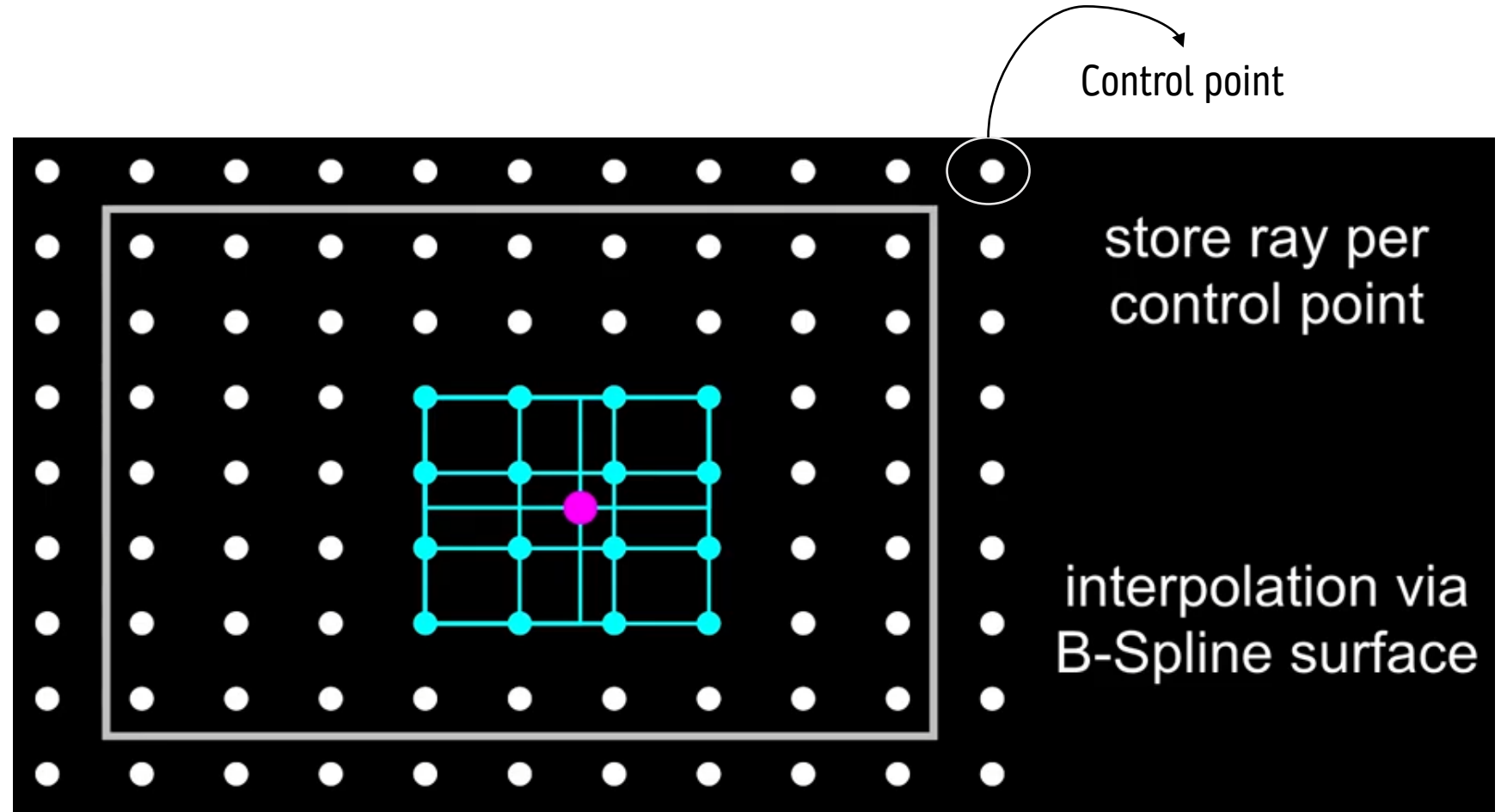
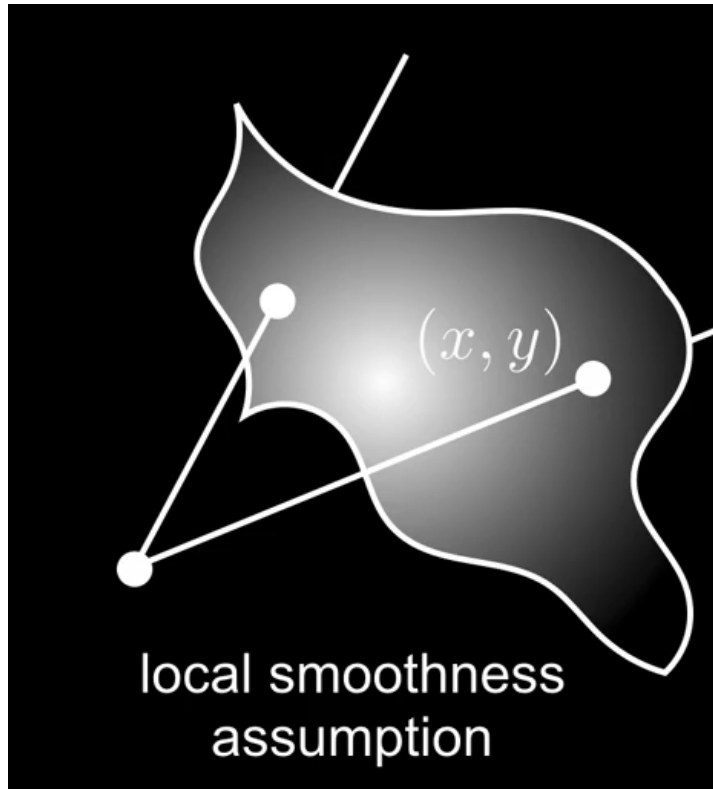
Why t = ray origin?



Differentiable Camera Model | Generic non-linear camera distortion

[2020 CVPR] Why having 10,000 parameters in your camera model is better than twelve

Camera parameters $\sim O(\# \text{ pixels})$



Differentiable Camera Model | Generic non-linear camera distortion

$$\mathbf{r}'_d = \mathbf{r}_d + \mathbf{z}_d, \quad \mathbf{r}'_o = \mathbf{r}_o + \mathbf{z}_o$$

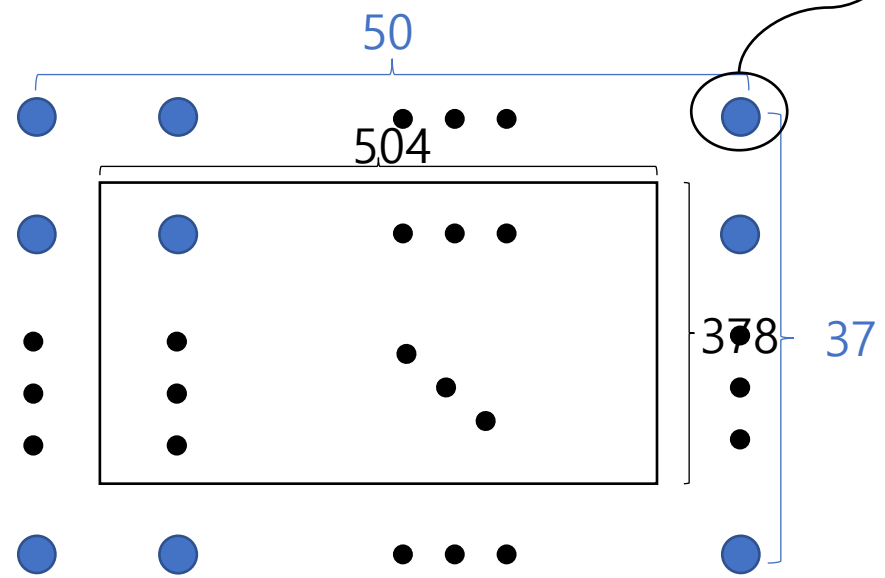
What We Want To Find

$$\mathbf{z}_d(\mathbf{p})_{3 \times 1} = \sum_{x=\lfloor \mathbf{p}_x \rfloor}^{\lfloor \mathbf{p}_x \rfloor + 1} \sum_{y=\lfloor \mathbf{p}_y \rfloor}^{\lfloor \mathbf{p}_y \rfloor + 1} (1 - |x - \mathbf{p}_x|)(1 - |y - \mathbf{p}_y|) \mathbf{z}_d[x, y]$$

Neighbor Control point

← 논문과 다름. 자체 수정

Control point



Dual comes for free

$$\mathbf{z}_o(\mathbf{p})_{3 \times 1} = \sum_{x=\lfloor \mathbf{p}_x \rfloor}^{\lfloor \mathbf{p}_x \rfloor + 1} \sum_{y=\lfloor \mathbf{p}_y \rfloor}^{\lfloor \mathbf{p}_y \rfloor + 1} (1 - |x - \mathbf{p}_x|)(1 - |y - \mathbf{p}_y|) \mathbf{z}_o[x, y]$$

Neighbor Control point

Differentiable Camera Model | Computation Graph of Ray Direction & Origin

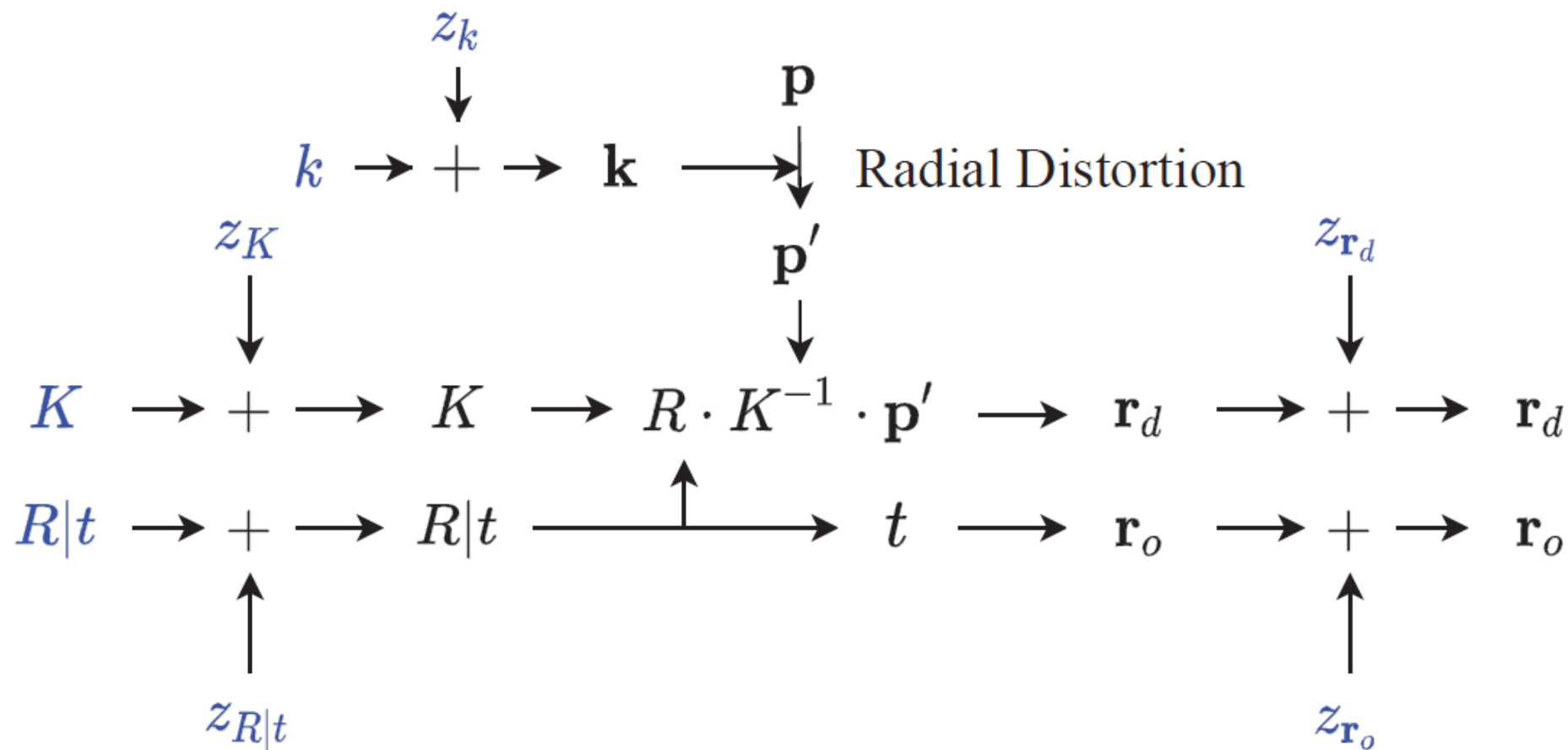
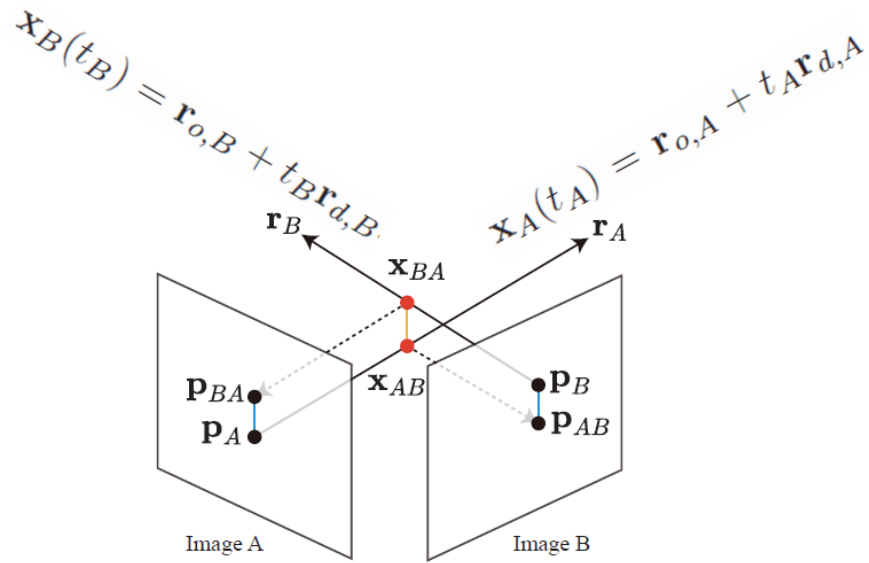


Figure 2. Computation graph of $\mathbf{r}_o, \mathbf{r}_d$ from camera parameters and camera parameter noise.

Loss | Geometric & Photometric Consistency

What We Want To Minimize

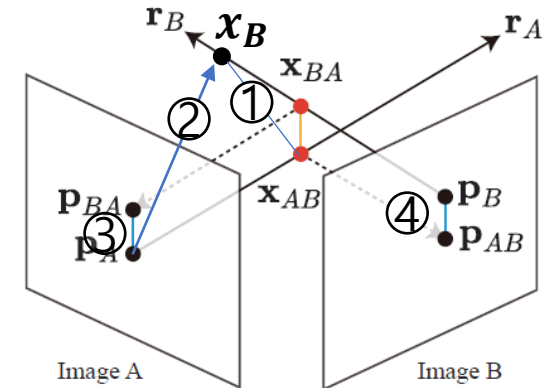


- $\mathbf{p}_A, \mathbf{p}_B$: matching points
- the ray \mathbf{r}_A and \mathbf{r}_B should intersect at the 3D point
- we can measure the deviation by computing the shortest distance between rays

$$\textcircled{1} d = \frac{\textcircled{2} |(\mathbf{r}_{o,B} + t_B \mathbf{r}_{d,B} - \mathbf{r}_{o,A}) \times \mathbf{r}_{A,d}|}{\|\mathbf{r}_{A,d}\|}$$

- If we solve for $\frac{dd^2}{dt_B} \big|_{\hat{t}_B} = 0$, we get

$$\hat{t}_B = \frac{(\mathbf{r}_{A,o} - \mathbf{r}_{B,o}) \times \mathbf{r}_{A,d} \cdot (\mathbf{r}_{A,d} \times \mathbf{r}_{B,d})}{(\mathbf{r}_{A,d} \times \mathbf{r}_{B,d})^2}$$



- $\mathbf{x}_B = \mathbf{x}_B(\hat{t}_B), \mathbf{x}_A = \mathbf{x}_A(\hat{t}_A)$

$$\mathcal{L} = \sum_{\mathbf{p} \in \mathcal{I}} \|C(\mathbf{p}) - \hat{C}(\mathbf{r}(\mathbf{p}))\|_2^2$$

Photometric Consistency Loss

$$d_\pi = \frac{\textcircled{3} \|\pi_A(\mathbf{x}_B) - \mathbf{p}_A\| + \textcircled{4} \|\pi_B(\mathbf{x}_A) - \mathbf{p}_B\|}{2}$$

Geometric Consistency Loss

When $R_A \mathbf{x}_B[z] > 0, R_B \mathbf{x}_A[z] > 0$

Curriculum Learning

Algorithm 1 Joint Optimization of Color Consistency Loss and Ray Distance Loss using Curriculum Learning

Initialize NeRF parameter Θ

Initialize camera parameter $z_K, z_{R|t}, z_{ray_o}, z_{ray_d}, z_k$

Learnable Parameters $\mathcal{S} = \{\Theta\}$

for iter=1,2,... **do**

$S' = \text{get_params}(\text{iter})$ ▷ Curriculum learning

$\mathbf{r}_d, \mathbf{r}_o \leftarrow \text{camera model}(K, \mathbf{z})$ Sec. 4

$\mathcal{L} \leftarrow \text{volumetric rendering}(\mathbf{r}_d, \mathbf{r}_o, \Theta)$ Eq. 1

if iter % $n == 0$ and iter $\geq n_{prd}$ **then**

$I' \leftarrow \text{random}(R_I, t_I, \mathcal{I})$

$\mathcal{C} \leftarrow \text{Correspondence}(I, I')$

$\mathcal{L}_{prd} \leftarrow \text{Projected Ray Distance}(\mathcal{C})$ Sec. 5.1

$\mathcal{L} \leftarrow \mathcal{L} + \lambda \mathcal{L}_{prd}$

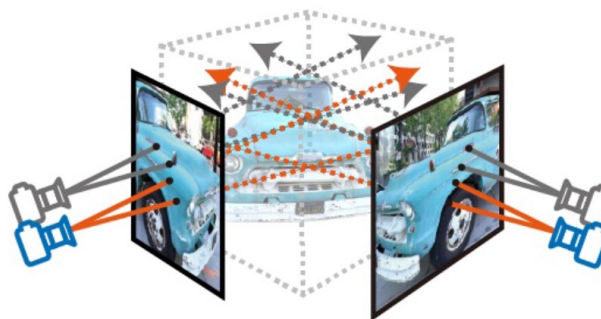
end if

for $s \in S'$ **do**

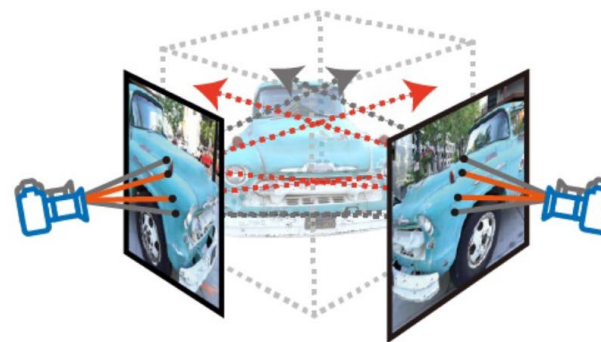
$s \leftarrow s + \nabla_s \mathcal{L}$

end for

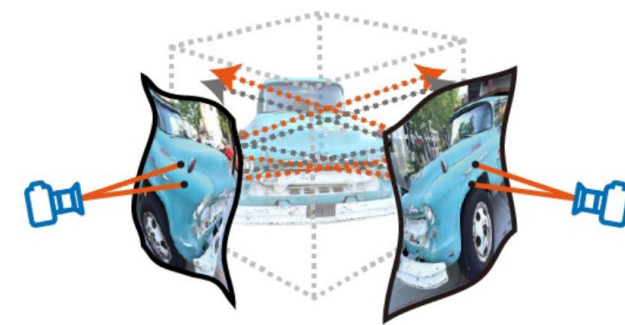
end for







(a) Calibrating extrinsic camera parameters



(b) Calibrating intrinsic camera parameters



(c) Calibrating non-linear distortion parameters

-  Init. cam. pose
-  Calib. cam. pose
-  Initial ray
-  Calibrated ray

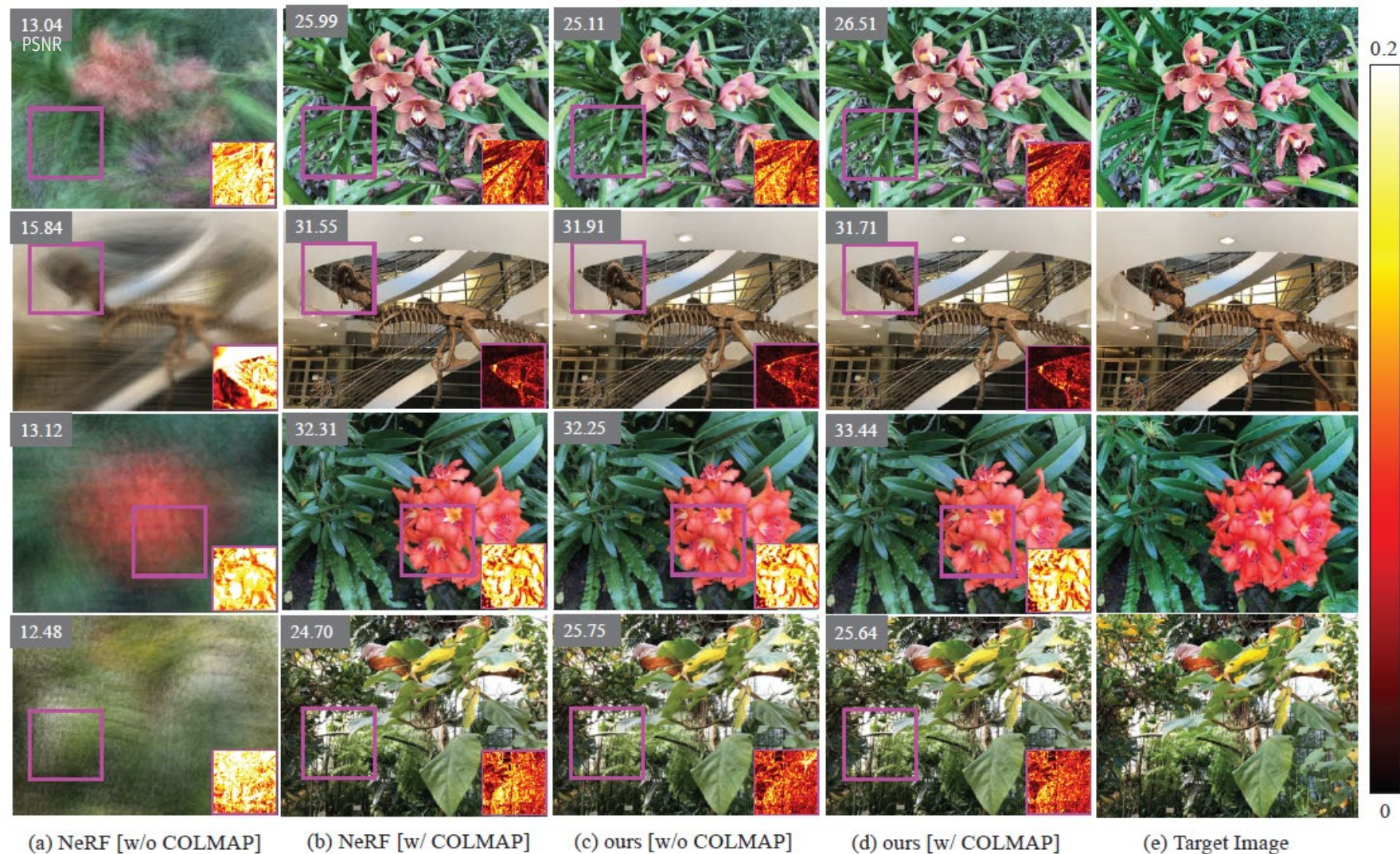
Experiment

Experiment | Dataset

- LLFF
 - 8 scenes
 - Estimate camera parameters using COLMAP
- Tanks and Temples
 - 4 scenes
 - Estimate camera parameters using COLMAP
- Author collected data
 - 6 scenes
 - fish-eye camera
 - Estimate camera parameters using COLMAP

Experiment | SCNeRF vs NeRF

LLFF dataset



Exp. On Train set

Table 1. Comparison of NeRF and our model when no calibrated camera information is given. "nan" denotes the case when no inlier matches are acquired due to the wrong camera information.

Scene	Model	PSNR(\uparrow) / SSIM(\uparrow) / LPIPS(\downarrow) / PRD(\downarrow)
Flower	NeRF	13.8 / 0.302 / 0.716 / nan
	ours	33.2 / 0.945 / 0.060 / 0.911
Fortress	NeRF	16.3 / 0.524 / 0.445 / nan
	ours	35.7 / 0.945 / 0.069 / 0.833
Leaves	NeRF	13.01 / 0.180 / 0.687 / nan
	ours	25.75 / 0.878 / 0.146 / 0.885
Trex	NeRF	15.70 / 0.409 / 0.575 / nan
	ours	31.75 / 0.954 / 0.104 / 1.002

Experiment | SCNeRF vs NeRF with COLMAP

LLFF dataset

target
Learned
ray offset

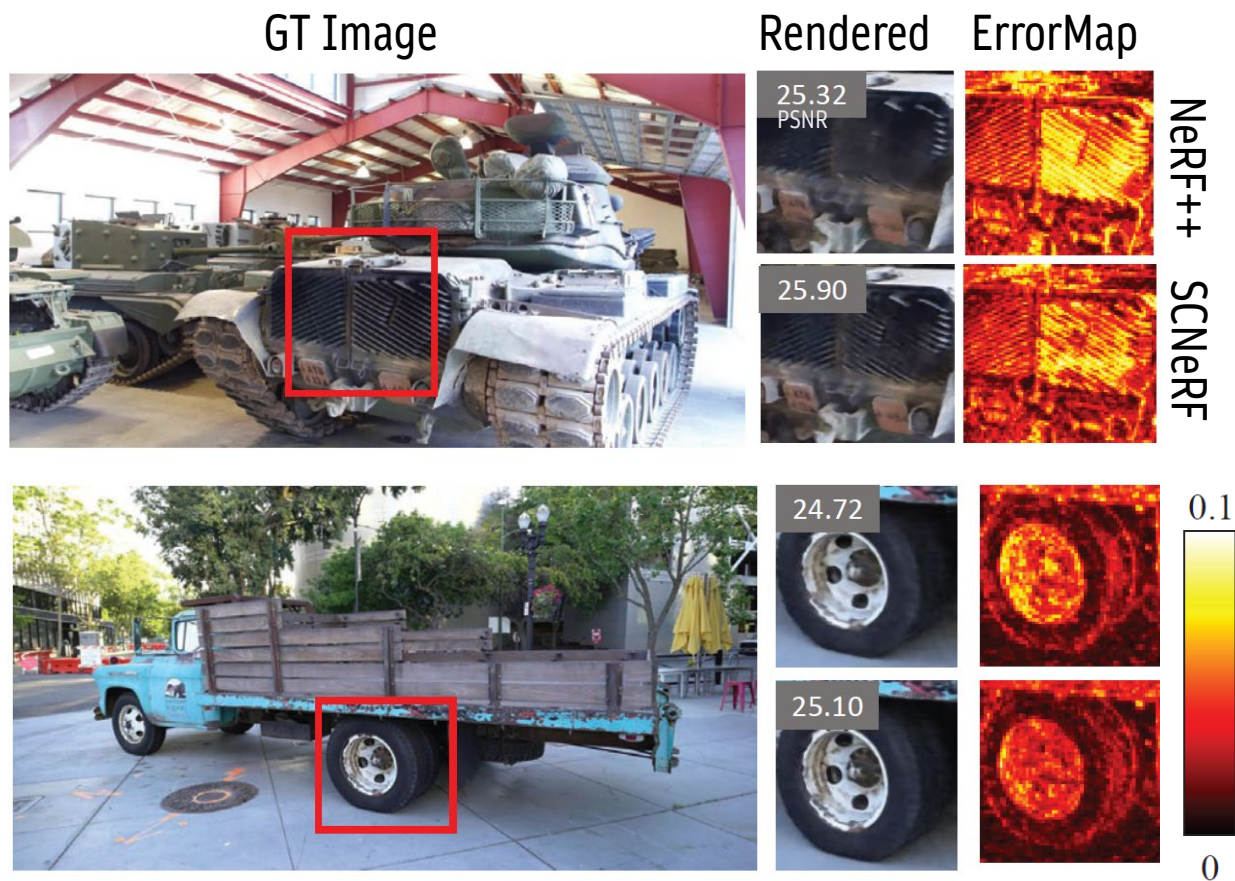


Table 2. Comparison of NeRF and our model when the camera parameters are initialized with COLMAP [16] in LLFF [12] dataset.

Scene	Model	PSNR(\uparrow) / SSIM(\uparrow) / LPIPS(\downarrow) / PRD(\downarrow)
Flower	NeRF	32.2 / 0.937 / 0.067 / 2.440
	ours	33.3 / 0.946 / 0.058 / 0.895
Fortress	NeRF	35.3 / 0.947 / 0.056 / 2.475
	ours	36.6 / 0.960 / 0.049 / 0.724
Leaves	NeRF	25.3 / 0.874 / 0.149 / 2.709
	ours	25.9 / 0.886 / 0.136 / 0.854
Trex	NeRF	31.4 / 0.955 / 0.099 / 2.368
	ours	32.0 / 0.959 / 0.095 / 0.953

Experiment | SCNeRF vs NeRF++

Tanks and Temples



Exp. On Train set

Table 3. Rendering qualities of NeRF++ and our model in tanks and temples [8] dataset.

Scene	Model	PSNR(↑) / SSIM(↑) / LPIPS(↓) / PRD(↓)
M60	NeRF++	25.62 / 0.772 / 0.395 / 1.335
	ours	26.99 / 0.805 / 0.359 / 1.326
Playground	NeRF++	25.14 / 0.681 / 0.434 / 1.302
	ours	26.17 / 0.715 / 0.396 / 1.299
Train	NeRF++	21.80 / 0.619 / 0.479 / 1.261
	ours	22.71 / 0.651 / 0.450 / 1.255
Truck	NeRF++	24.13 / 0.730 / 0.392 / 1.248
	ours	25.22 / 0.763 / 0.352 / 1.240

Experiment | SCNeRF vs NeRF++ with COLMAP (Fish-eye Lens)

- Conventional feature matching algorithms fail → skip the projected ray distance loss
- Since the NeRF++ camera model does not incorporate the radial distortion, it's modified to incorporate the fish-eye distortion in ray computation.

Images captured using a fish-eye camera



globe



cube

Table 4. Rendering qualities of scenes captured on fish-eye cameras. "RD" denotes the modified implementation to reflect radial distortions.

Scene	Model	PSNR(↑) / SSIM(↑) / LPIPS(↓)
Globe	NeRF++[RD]	21.97 / 0.572 / 0.659
	ours	23.76 / 0.598 / 0.633
Cube	NeRF++[RD]	21.30 / 0.574 / 0.643
	ours	23.17 / 0.605 / 0.616

Experiment | Ablation Study

- The performance for each phase in curriculum learning.
- 200K iterations for each phase.
- extending model is more potential in rendering clearer images
- for some scenes, adopting projected ray distance increases the overall projected ray distance.

Table 5. Ablation studies about components of our model. "IE", "OD", and "PRD" denote learnable intrinsic and extrinsic parameters, learnable non-linear distortion, and projected ray distance loss, respectively.

Scene		PSNR(\uparrow) / SSIM(\uparrow) / LPIPS(\downarrow) / PRD(\downarrow)
Fortress	NeRF	30.5 / 0.866 / 0.096 / 0.856
	+ IE	35.3 / 0.948 / 0.058 / 0.729
	+ IE + OD	36.4 / 0.957 / 0.051 / 0.724
	+ IE + OD + PRD	36.6 / 0.96 / 0.049 / 0.724
Room	NeRF	31.5 / 0.950 / 0.096 / 0.883
	+ IE	38.3 / 0.978 / 0.070 / 0.806
	+ IE + OD	39.4 / 0.980 / 0.065 / 0.805
	+ IE + OD + PRD	39.7 / 0.981 / 0.063 / 0.805

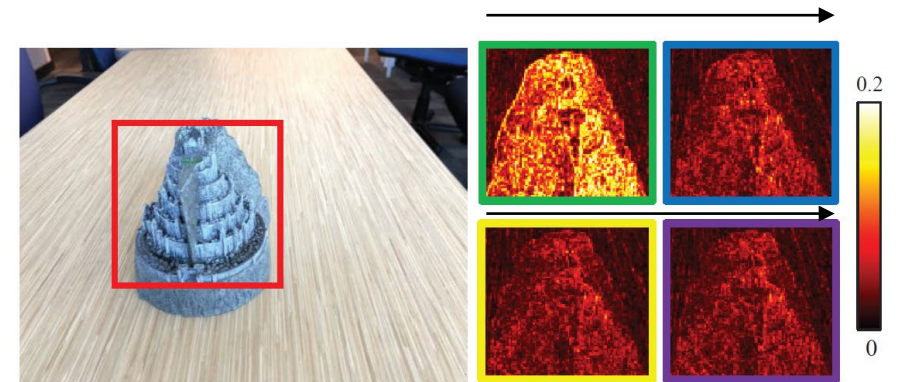


Figure 8. Visualization of rendered images for each configurations shown in Table 5. The green, blue, yellow, and purple box are the error map of NeRF, NeRF + IE, NeRF + IE + OD, and NeRF + IE + OD + PRD, respectively.

Conclusion

Take Home Message | Main

- SCNeRF proposed a self-calibration algorithm that learns geometry and camera parameters jointly end-to-end.
- SCNeRF learns geometry and camera parameters from scratch w/o poses
- SCNeRF Improves both NeRF and NeRF++ to be more robust with given camera poses.
- The camera model of SCNeRF consists of a pinhole model, radial distortion, and non-linear distortion
- SCNeRF proposed projected ray distance to improve accuracy

Take Home Message | Sub

- Initialization matters
- Curriculum learning is required to exploit learned geometry for PRD
- PRD loss is proposed but not appropriate for fisheye images
- PRD loss is not always successful
 - For some scenes, adopting projected ray distance increases the overall projected ray distance

References

- Project Page: <https://postech-cvlab.github.io/SCNeRF/>
- Code: <https://github.com/POSTECHCVLab/SCNeRF>
- Paper: <https://arxiv.org/abs/2108.13826>
- Video: <https://www.youtube.com/watch?v=wsjx6geduvk>

- Generic Camera Model:
<https://arxiv.org/abs/1912.02908#:~:text=Why%20Having%2010%2C000%20Parameters%20in%20Your%20Camera%20Model%20is%20Better%20Than%20Twelve,-Thomas%20Sch%C3%B6ps%2C%20Viktor&text=Camera%20calibration%20is%20an%20essential,to%20complex%20real%20lens%20distortion.>
- What is Camera Calibration : https://www.mathworks.com/help/vision/ug/camera-calibration.html#mw_901dad02-dbeb-4418-a316-6631bda4b844
- Camera Calibration Toolbox for Matlab : <http://robots.stanford.edu/cs223b04/JeanYvesCalib/>
- Camera Model : https://cvgl.stanford.edu/teaching/cs231a_winter1415/lecture/lecture2_camera_models_note.pdf
- Point-Line Distance--3-Dimensional : <https://mathworld.wolfram.com/Point-LineDistance3-Dimensional.html>