DeepRob

Seminar 8
Implicit Scene-Level Representations
University of Michigan and University of Minnesota

This Week: Scene-Level Representations

- Seminar 7: Semantic Scene Graphs and Explicit Representations
  1. Image Retrieval using Scene Graphs, Johnson et al., 2015
  2. Semantic Robot Programming for Goal-Directed Manipulation in Cluttered Scenes, Zeng et al., 2018
  3. Semantic Linking Maps for Active Visual Object Search, Zeng et al., 2020
  4. Hydra: A Real-time Spatial Perception System for 3D Scene Graph Construction and Optimization, Hughes et al., 2022
- Seminar 8: Neural Radiance Fields and Implicit Representations
  1. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, Mildenhall et al., 2020
  2. iMAP: Implicit Mapping and Positioning in Real-Time, Sucar et al., 2021
  3. NeRF-SLAM: Real-Time Dense Monocular SLAM with Neural Radiance Fields, Rosinol et al., 2022
  4. NeRF-Supervision: Learning Dense Object Descriptors from Neural Radiance Fields, Yen-Chen et al., 2022
  5. NARF22: Neural Articulated Radiance Fields for Configuration-Aware Rendering, Lewis et al., 2022

Today: Implicit Representations

- Seminar 7: Semantic Scene Graphs and Explicit Representations
  1. Image Retrieval using Scene Graphs, Johnson et al., 2015
  2. Semantic Robot Programming for Goal-Directed Manipulation in Cluttered Scenes, Zeng et al., 2018
  3. Semantic Linking Maps for Active Visual Object Search, Zeng et al., 2020
  4. Hydra: A Real-time Spatial Perception System for 3D Scene Graph Construction and Optimization, Hughes et al., 2022
- Seminar 8: Neural Radiance Fields and Implicit Representations
  1. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, Mildenhall et al., 2020
  2. iMAP: Implicit Mapping and Positioning in Real-Time, Sucar et al., 2021
  3. NeRF-SLAM: Real-Time Dense Monocular SLAM with Neural Radiance Fields, Rosinol et al., 2022
  4. NeRF-Supervision: Learning Dense Object Descriptors from Neural Radiance Fields, Yen-Chen et al., 2022
  5. NARF22: Neural Articulated Radiance Fields for Configuration-Aware Rendering, Lewis et al., 2022

NeRF

Representing Scenes as Neural Radiance Fields for View Synthesis

By: Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik,
Jonathan T. Barron, Ravi Ramamoorthi, Ren Ng

Presented by: Sibo Wang, Yuxi Zhang, Yulun Zhuang

Rendering 3D scenes

Rendering 3D scenes

The Authors

Ben Mildenhall*

UC Berkeley

Pratul Srinivasan*

UC Berkeley

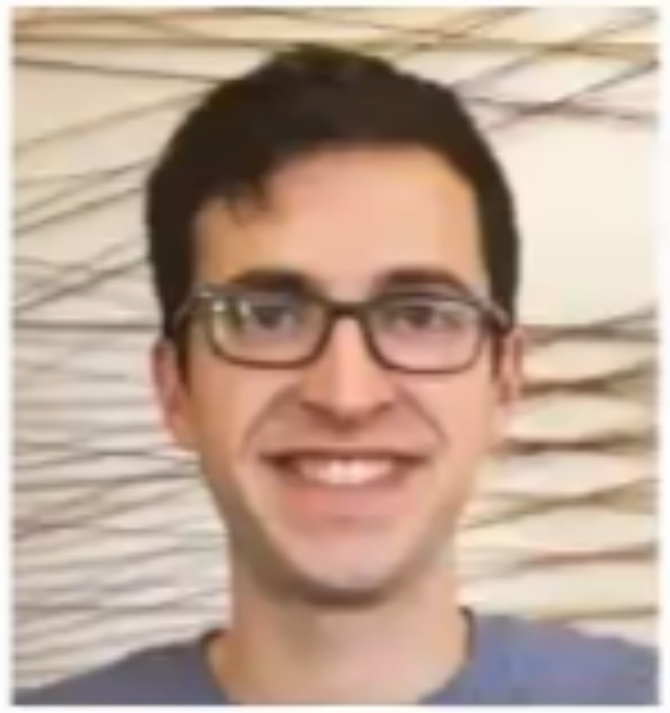
Google

UC San Diego

Ren Ng

UC Berkeley

Matt Tancik*



UC Berkeley

Jon Barron

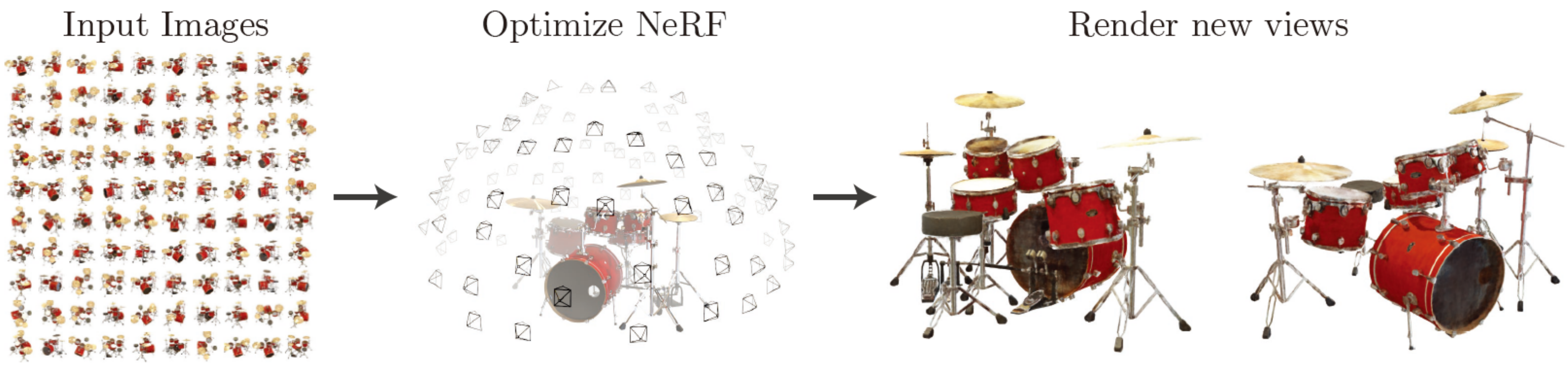
Google Research

Ravi Ramamoorthi

UC San Diego

Problem

How to reconstruct 3D scene from several 2D images inputs?



Contributions

1. An approach for representing continuous scenes with complex geometry and materials as *5D neural radiance fields*, parameterized as basic **MLP networks**.

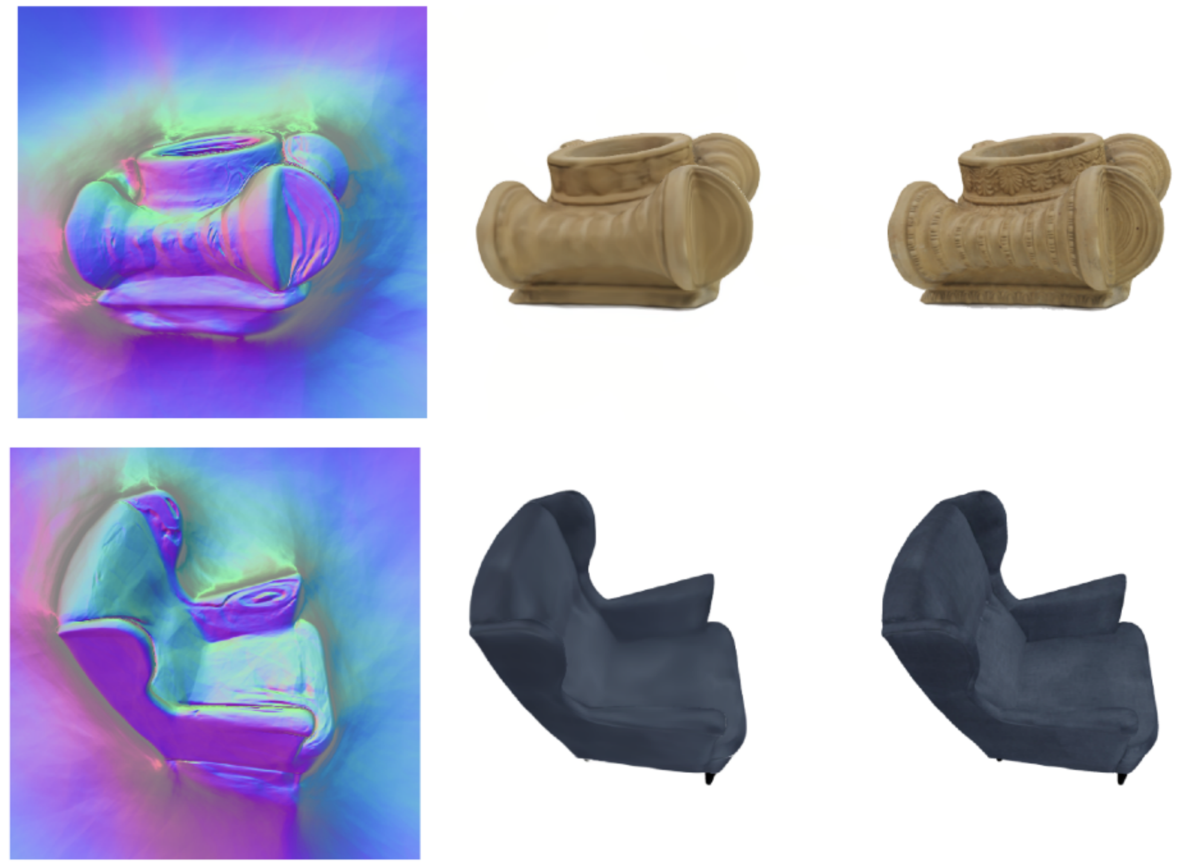
1. A differentiable rendering procedure based on classical volume rendering techniques. The procedure also includes a **hierarchical sampling strategy** to allocate the MLP's capacity towards space with visible scene content.

1. A **positional encoding** to map each input 5D coordinate into a higher dimensional space, which enables us to successfully optimize neural radiance elds to represent high-frequency scene content.

Previous Work

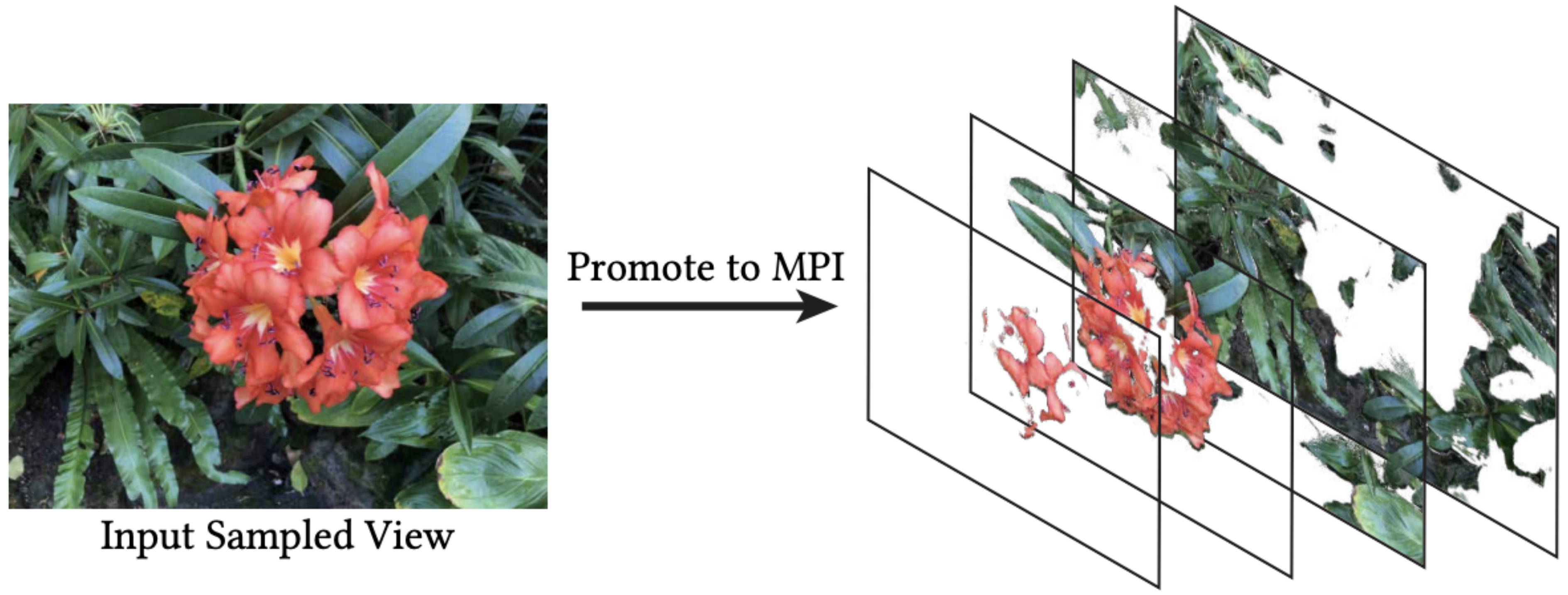
★Neural 3D shape representations

○ Scene Representation Networks (**SRN**) [1]:



★View synthesis and image-based rendering

○ Local Light Field Fusion (**LLFF**) [2]:



[1] Sitzmann, V., Zollhoefer, M., Wetzstein, G.: Scene representation networks: Continuous 3D-structure-aware neural scene representations. In: NeurIPS (2019)
[2] Mildenhall, B., Srinivasan, P.P., Ortiz-Cayon, R., Kalantari, N.K., Ramamoorthi,R., Ng, R., Kar, A.: Local light eld fusion: Practical view synthesis with prescriptive sampling guidelines. ACM Transactions on Graphics (SIGGRAPH) (2019)

NeRF Scene Representation

★Input:
- Position and Viewing Angles

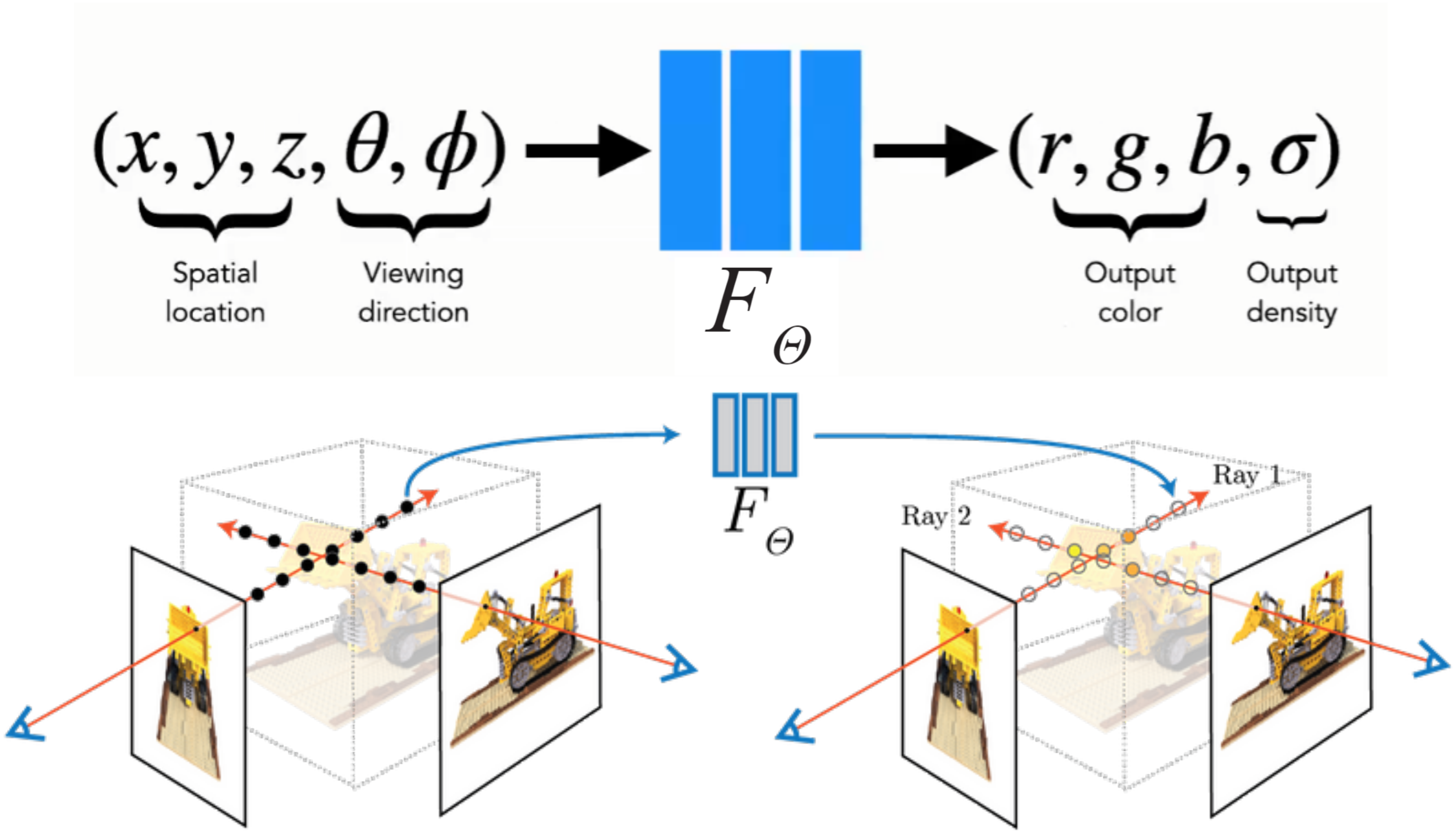
$\mathbf{x}(x, y, z)$ $\mathbf{d}(\theta, \phi)$

★Output:
- Emitted Color and Volume Density

$\mathbf{c}\,(r, g, b)$ σ ( )

★Scene Representation:

MI$F_\Theta : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma)$



ReRF MLP Network Architecture

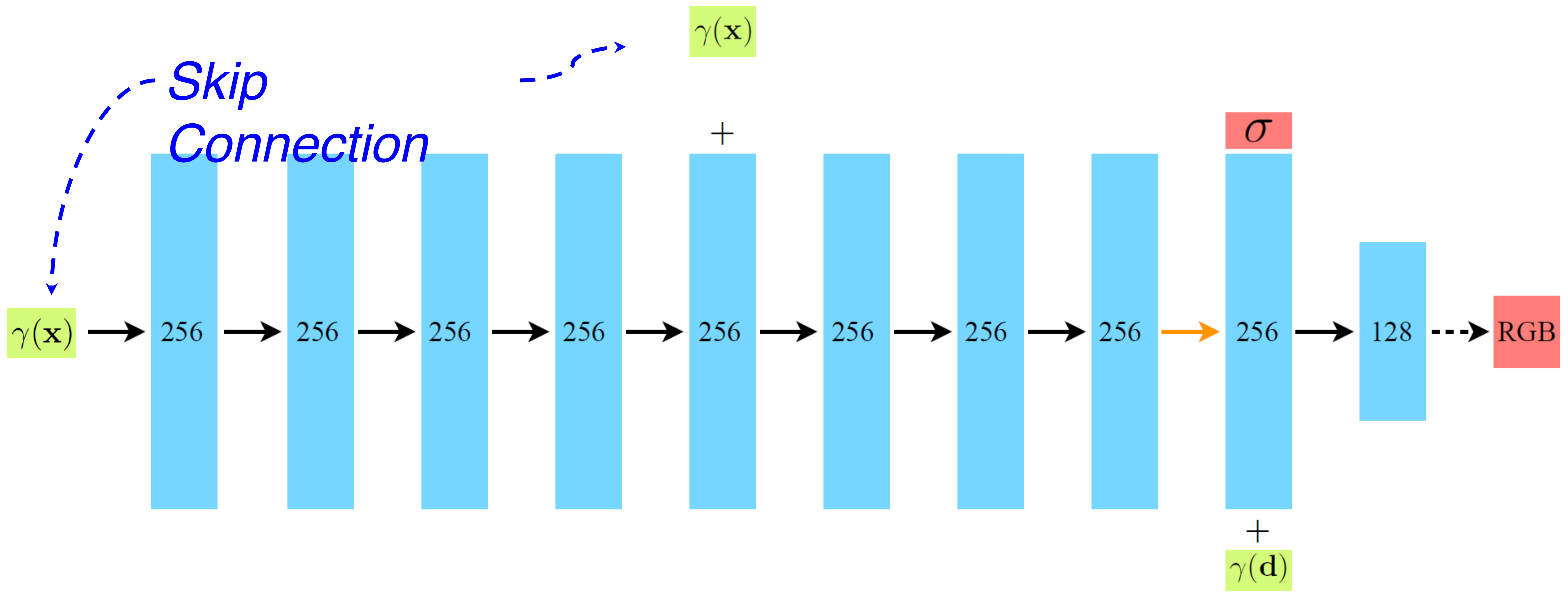
Input (*positional encoded*)

Hidden Layers

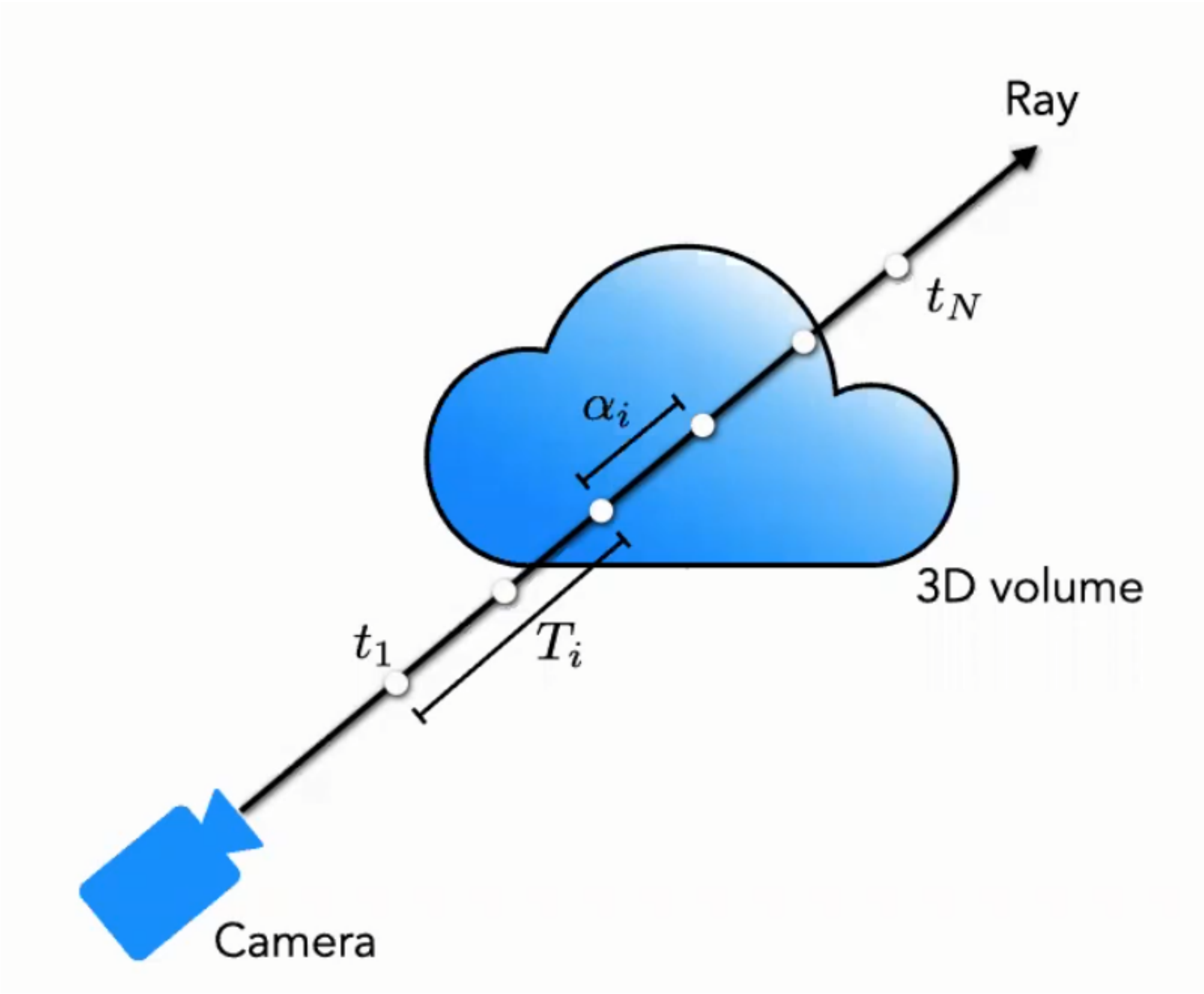
Output

ReLU

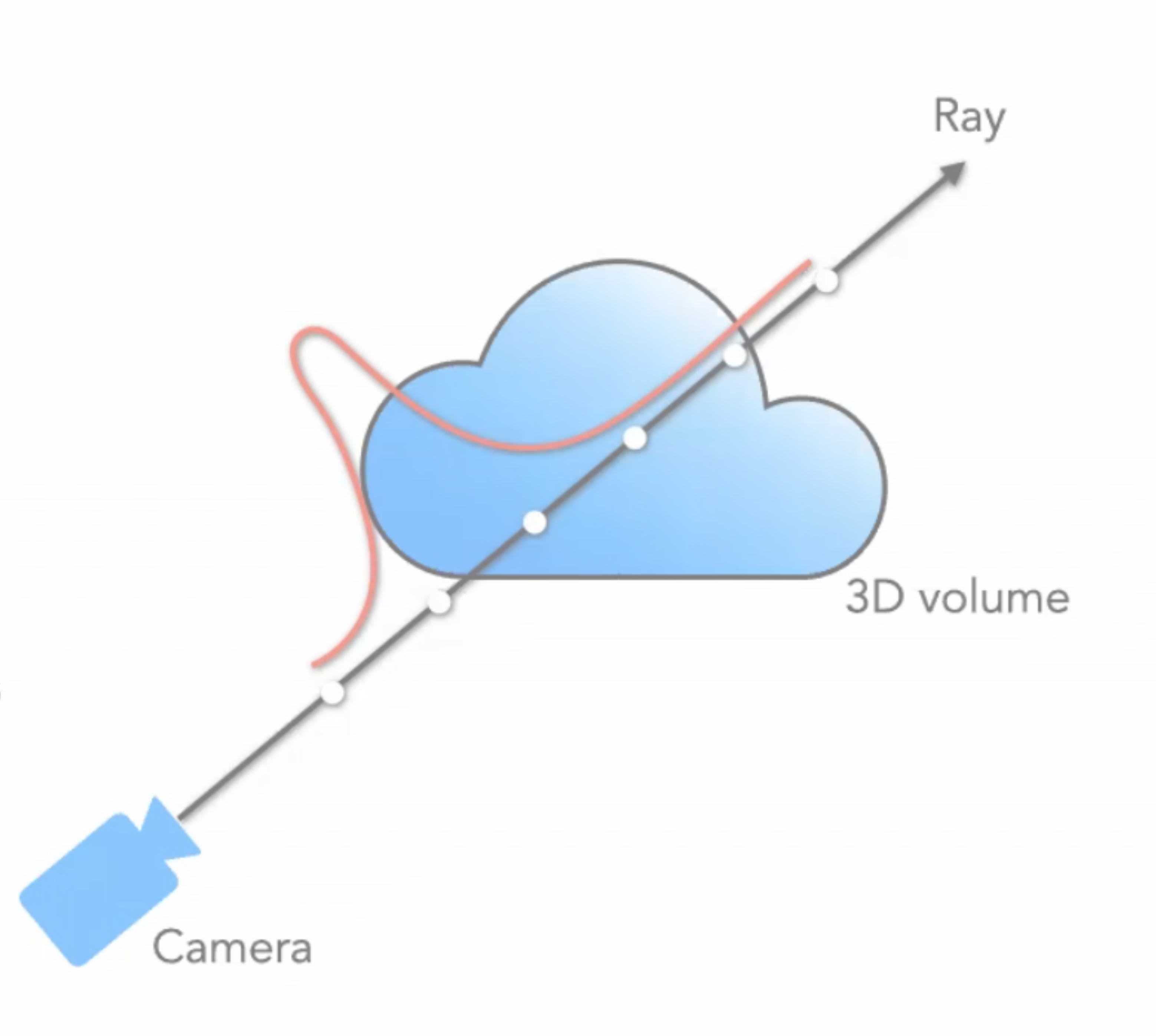
Sigmoid



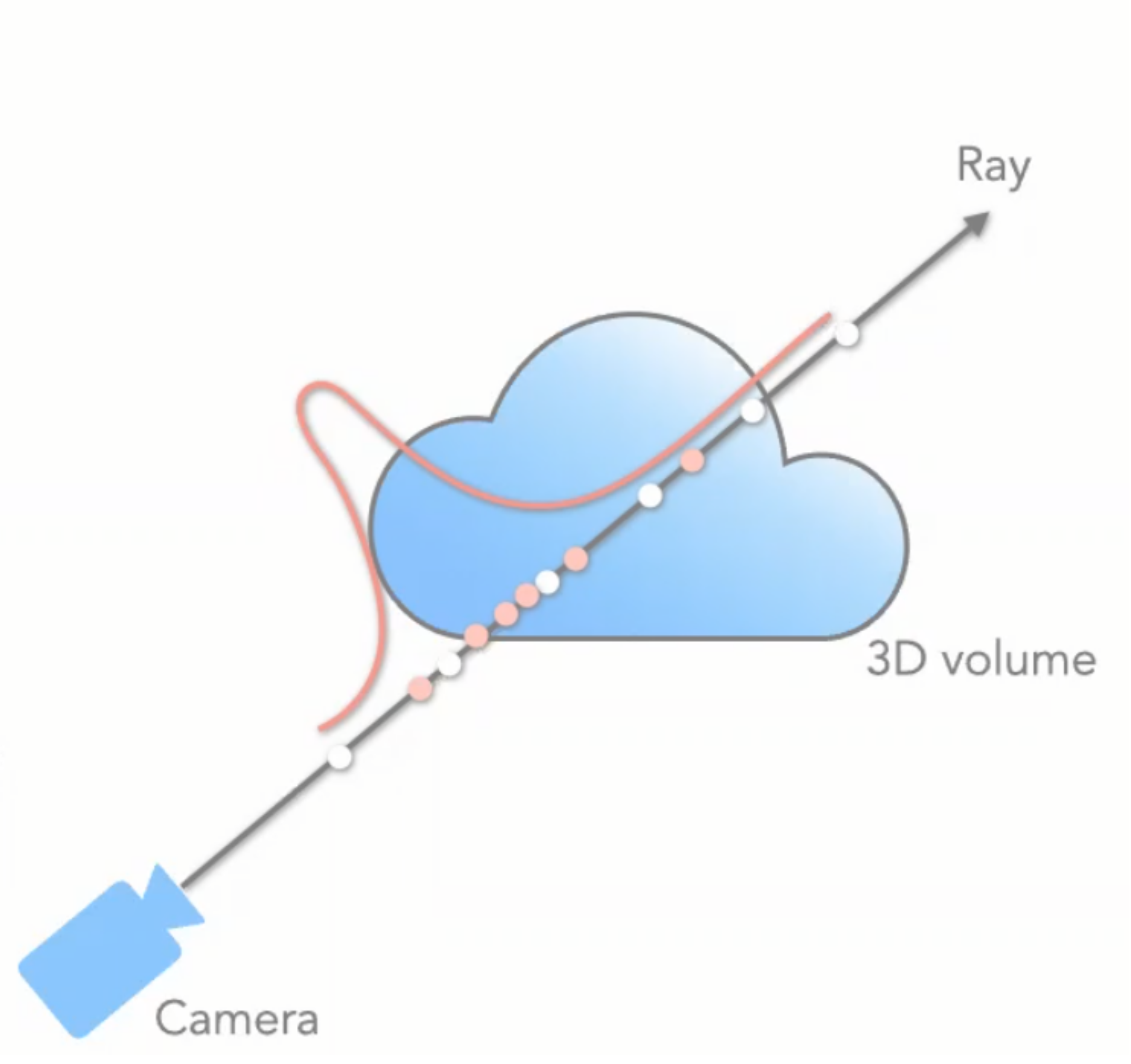
Volume Rendering and Hierarchical Sampling



Volume Rendering



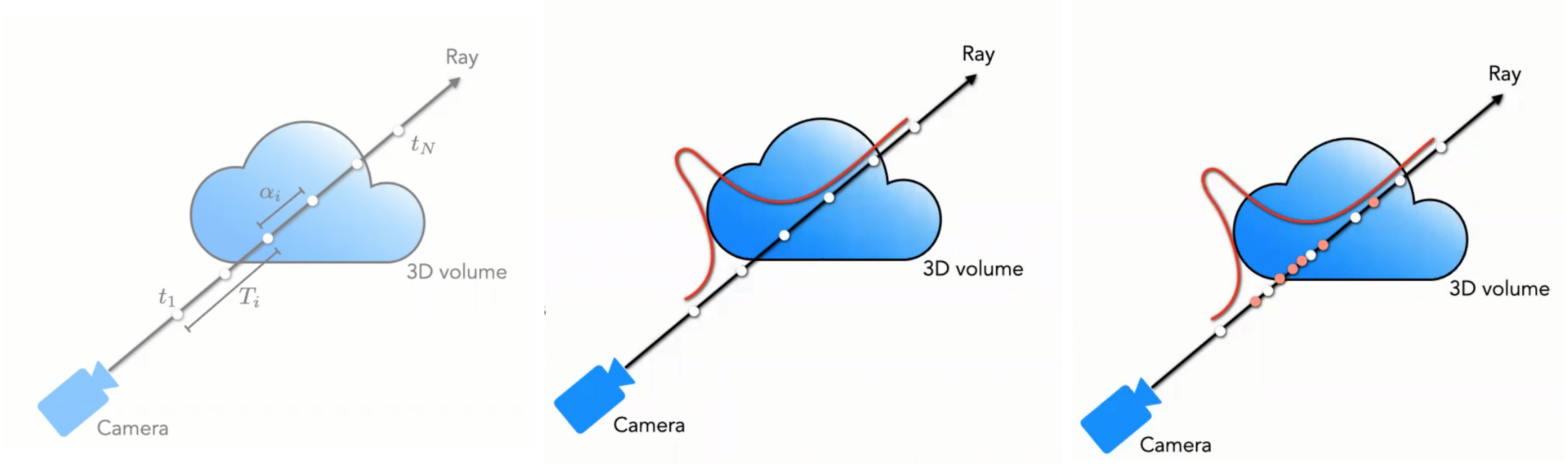
“Coarse” Sampling



“Fine” Sampling

Figures taken from the presentation *Matthew Tancik: Neural Radiance Fields for View Synthesis*

Volume Rendering and Hierarchical Sampling



Volume Rendering

"Coarse" Sampling

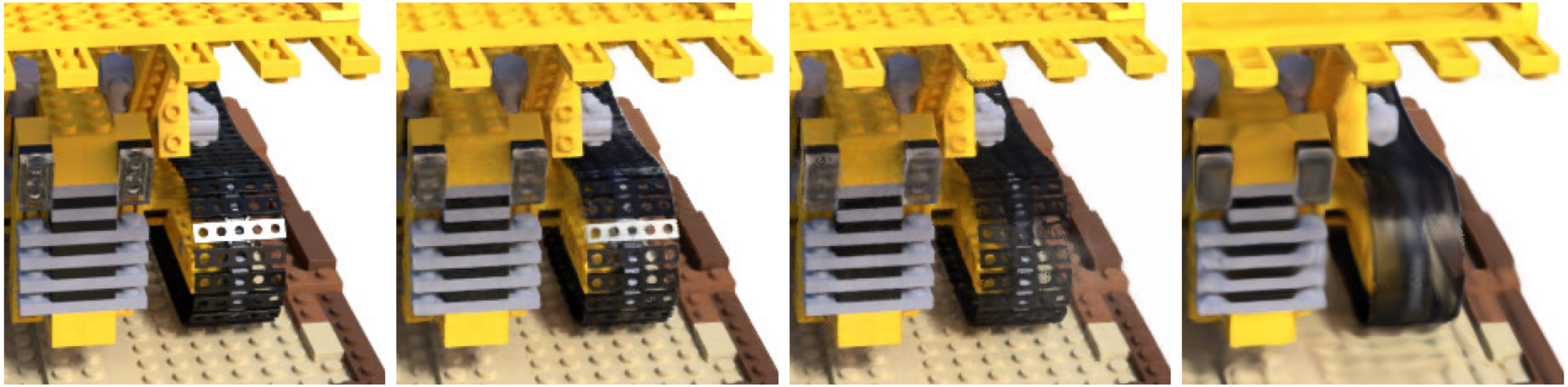
"Fine" Sampling

Figures taken from the presentation *Matthew Tancik: Neural Radiance Fields for View Synthesis*

Positional Encoding

Map inputs to a higher dimensional space such that the network can better fit the data that contains high frequency variation

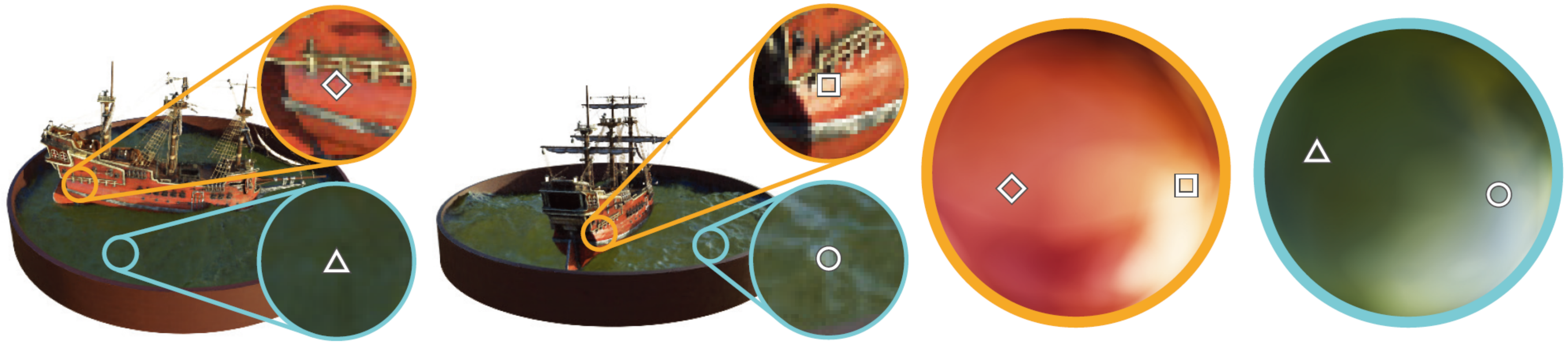
$$\gamma(p) = \left(\sin\left(2^0 \pi p\right), \cos\left(2^0 \pi p\right), \cdots, \sin\left(2^{L-1} \pi p\right), \cos\left(2^{L-1} \pi p\right) \right)$$



Ground Truth | Complete Model | No View Dependence | No Positional Encoding

Results

Method	Diffuse Synthetic 360° [41]			Realistic Synthetic 360°			Real Forward-Facing [28]		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
SRN [42]	33.20	0.963	0.073	22.26	0.846	0.170	22.84	0.668	0.378
NV [24]	29.62	0.929	0.099	26.05	0.893	0.160	-	-	-
LLFF [28]	34.38	0.985	0.048	24.88	0.911	0.114	24.13	0.798	**0.212**
Ours	**40.15**	**0.991**	**0.023**	**31.01**	**0.947**	**0.081**	**26.50**	**0.811**	0.250



(a) View 1 (b) View 2 (c) Radiance Distributions

Visual Comparisons

Visual Comparisons

Rendering Visualizations

Rendering Visualizations

Conclusions

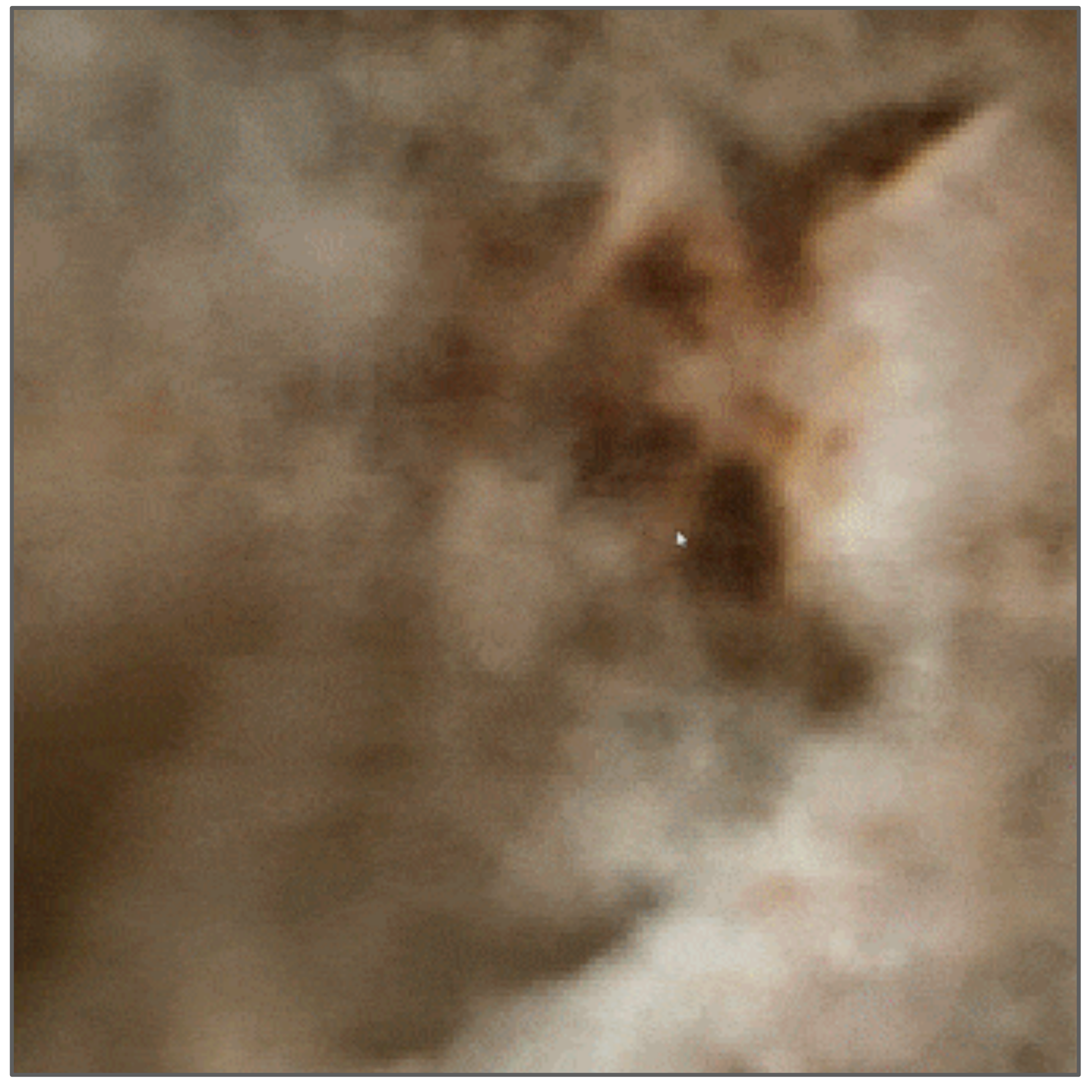
★**Introduced** NeRF, a novel method for learning and representing scenes as 5D neural radiance field based on 2D RGB images.

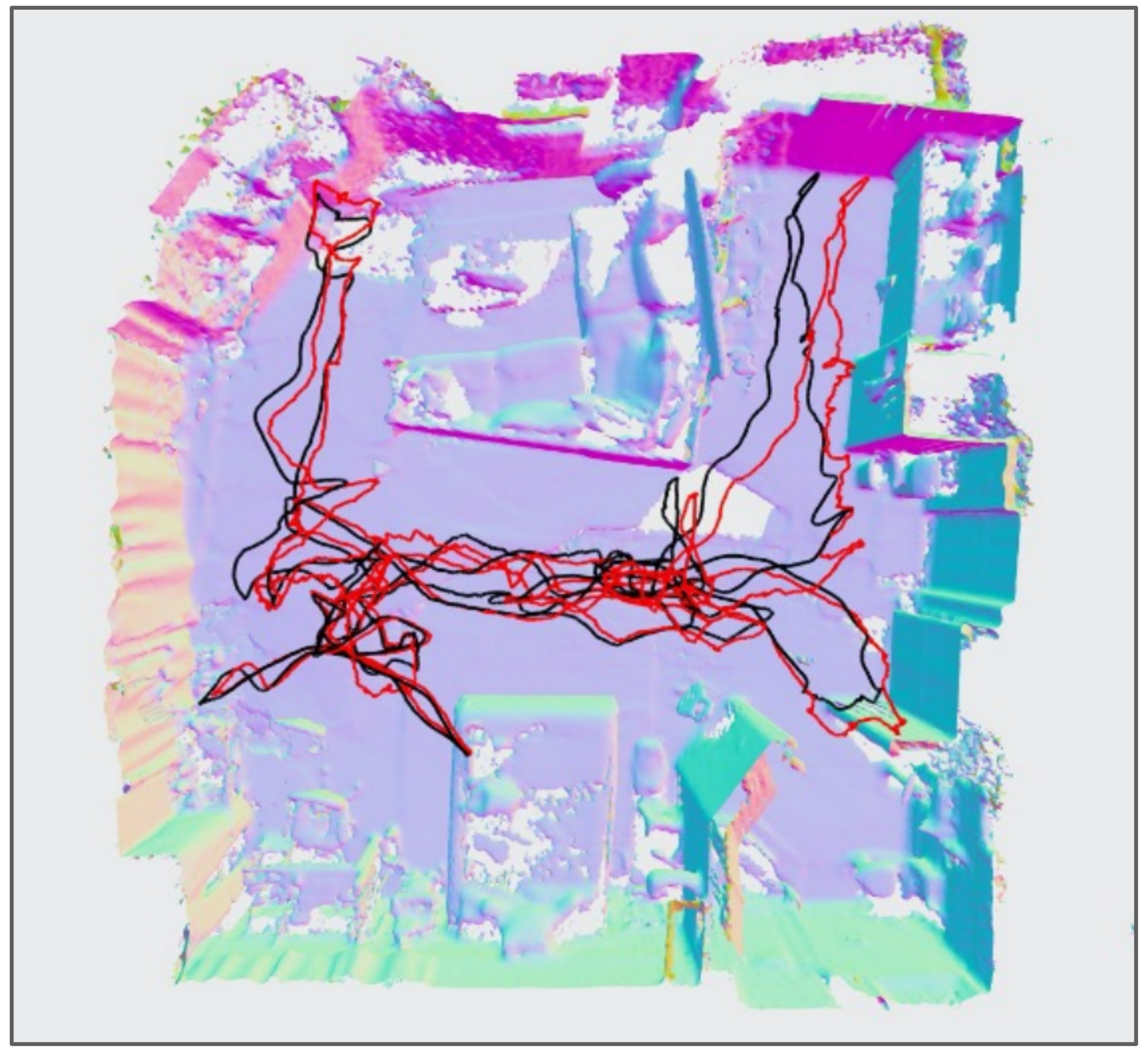
★**Outperformed** previous approach of training deep CNNs to output discretized voxel representations

Limitations and Future Work

★**Limitations**
- ○ Slow training time (1~2 days for each scene)

★**Future Directions**
- ○ Real-time rendering
- ○ Integration with SLAM

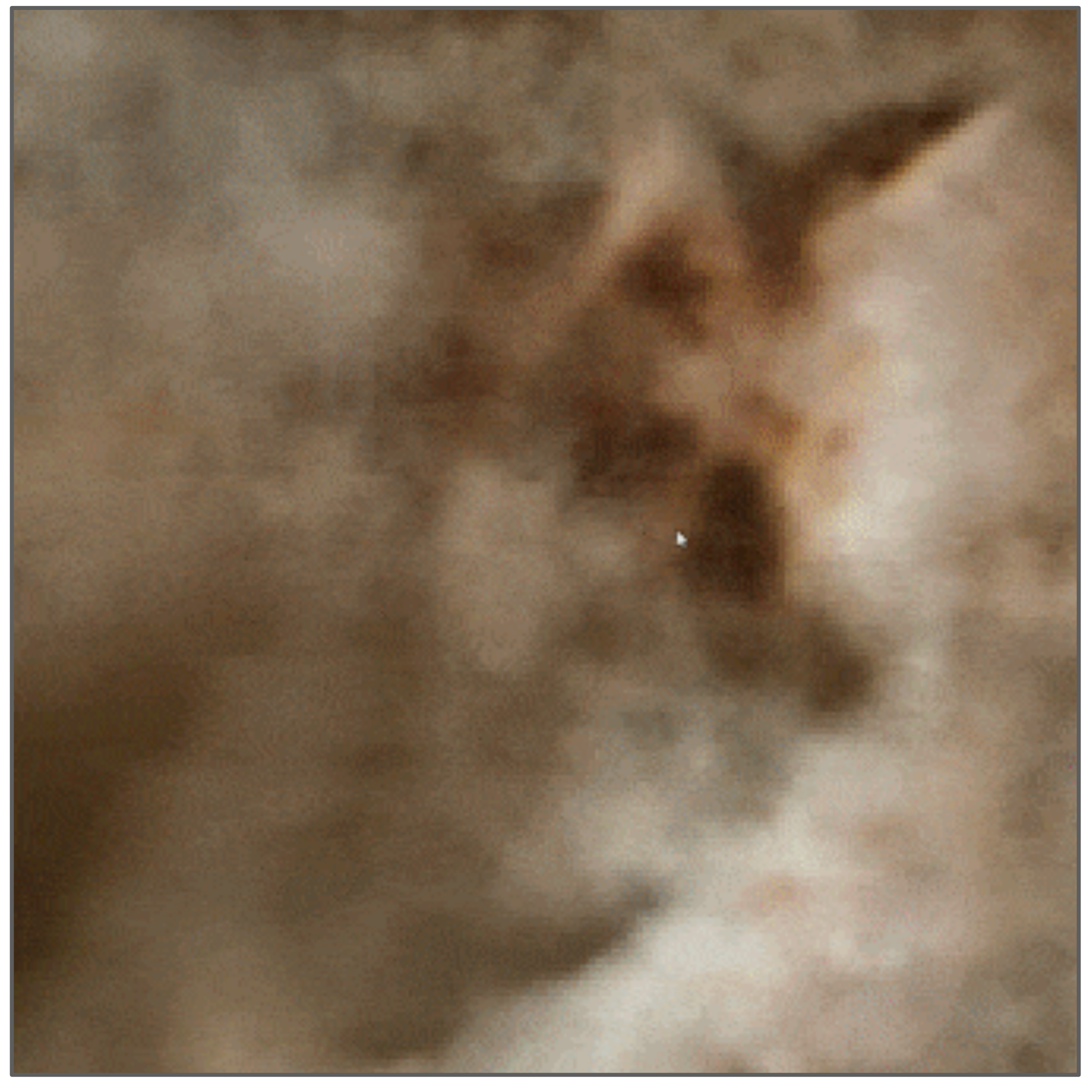


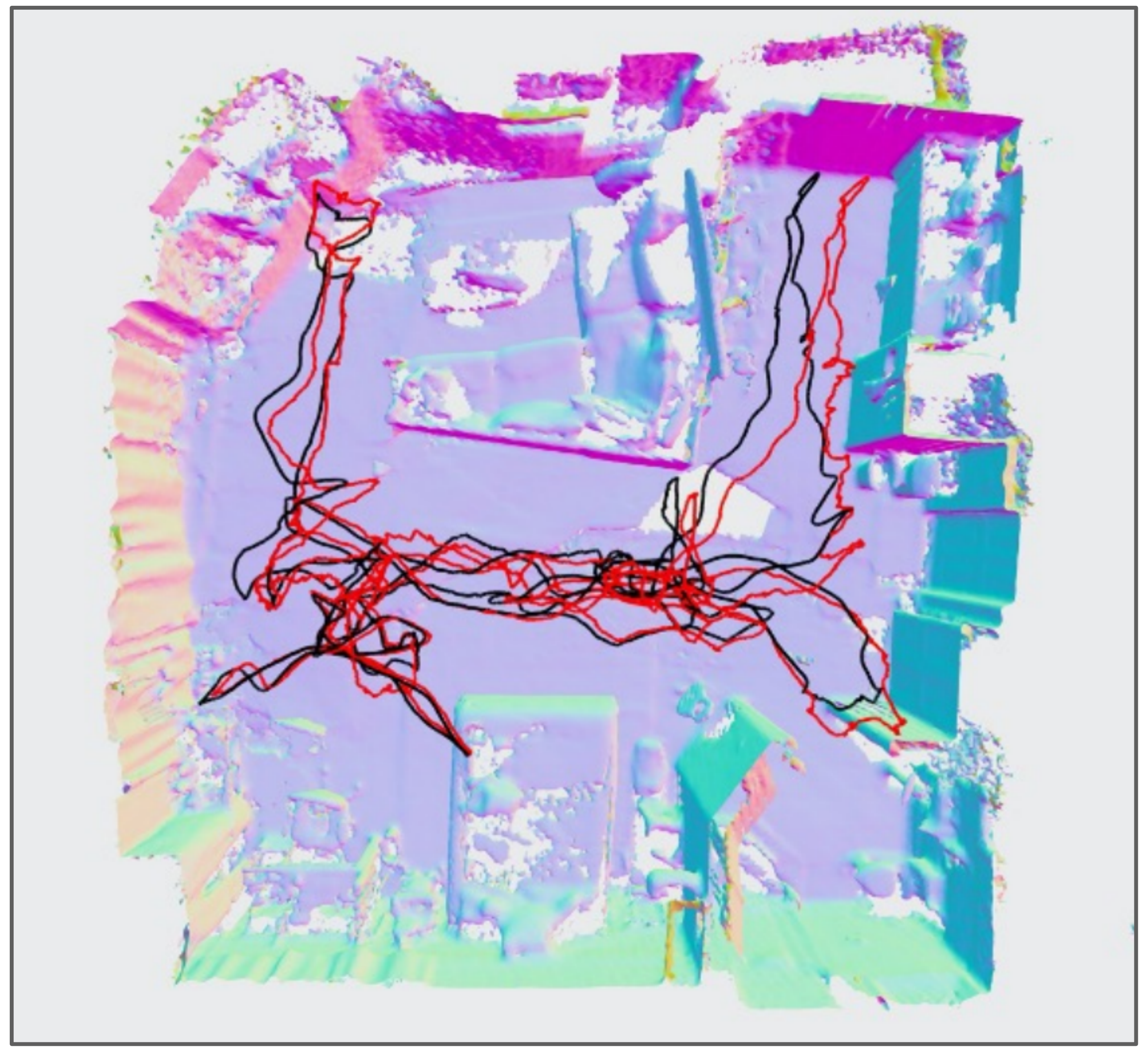


Limitations and Future Work

★**Limitations**
- Slow training time (1~2 days for each scene)

★**Future Directions**
- Real-time rendering
- Integration with SLAM





Thank you

iMAP

Implicit Mapping and Positioning in Real Time

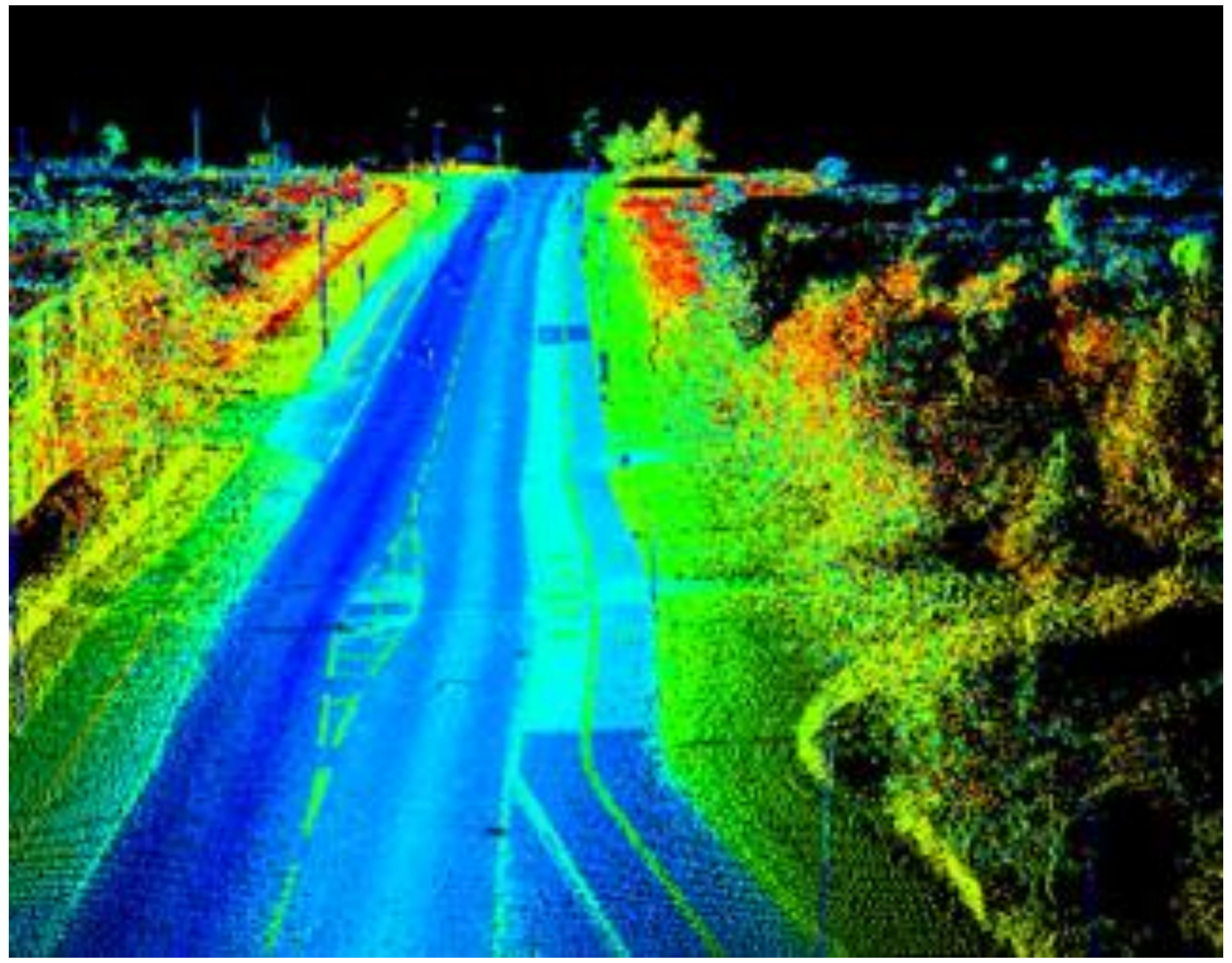
By: Edgar Sucar, Shikun Liu, Joseph Ortiz, Andrew Davidson

Presented by: Jonathan Heidegger, Seth Isaacson, Frank Kung

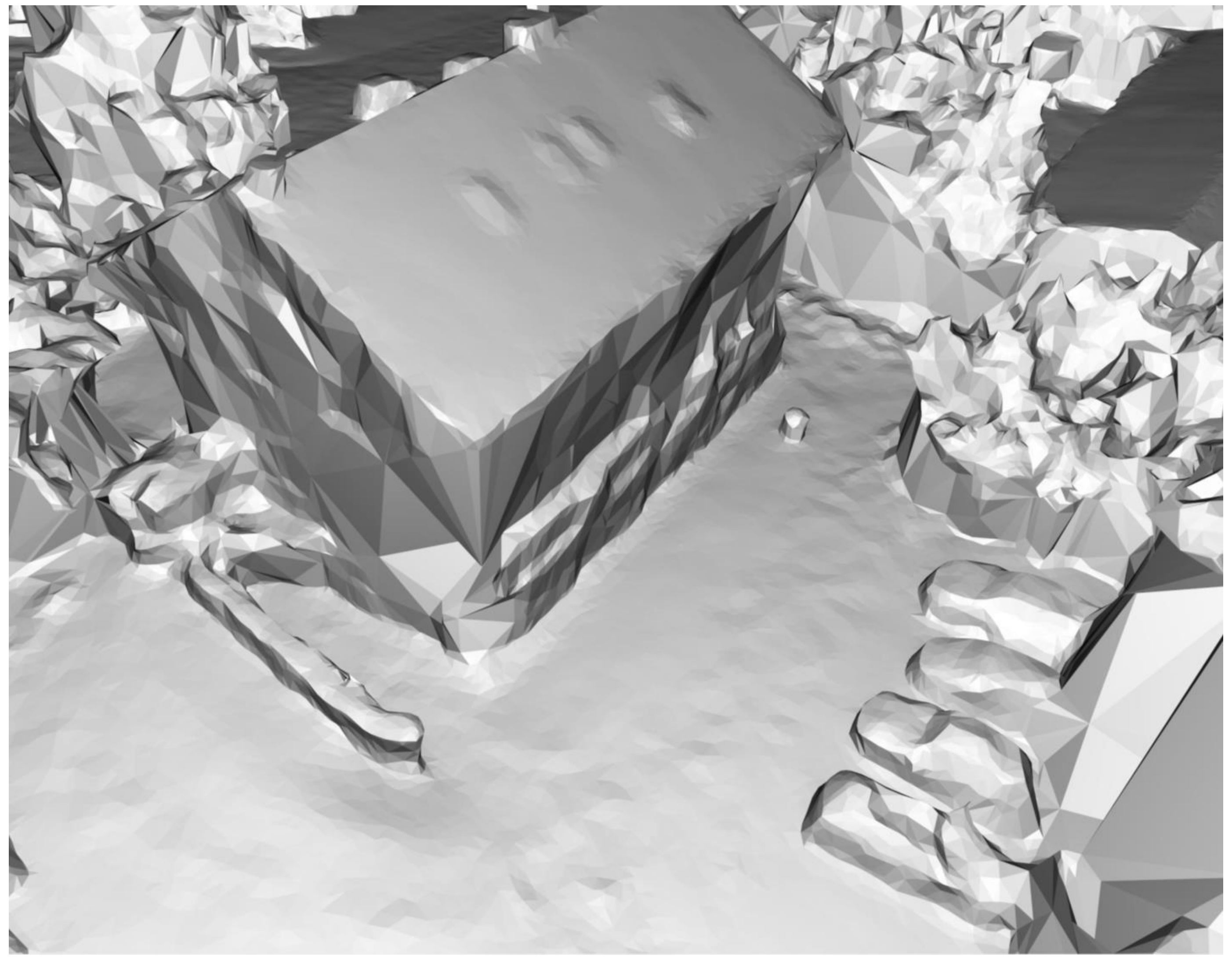
The Authors

- Edgar Sucar and Shikun Liu:
  - PhD Students in Dyson Robotics Lab
- Joseph Ortiz
  - (At the time) PhD Student at Imperial College of London
- Andrew J. Davidson
  - Professor of Robot Vision; Faculty Advisor

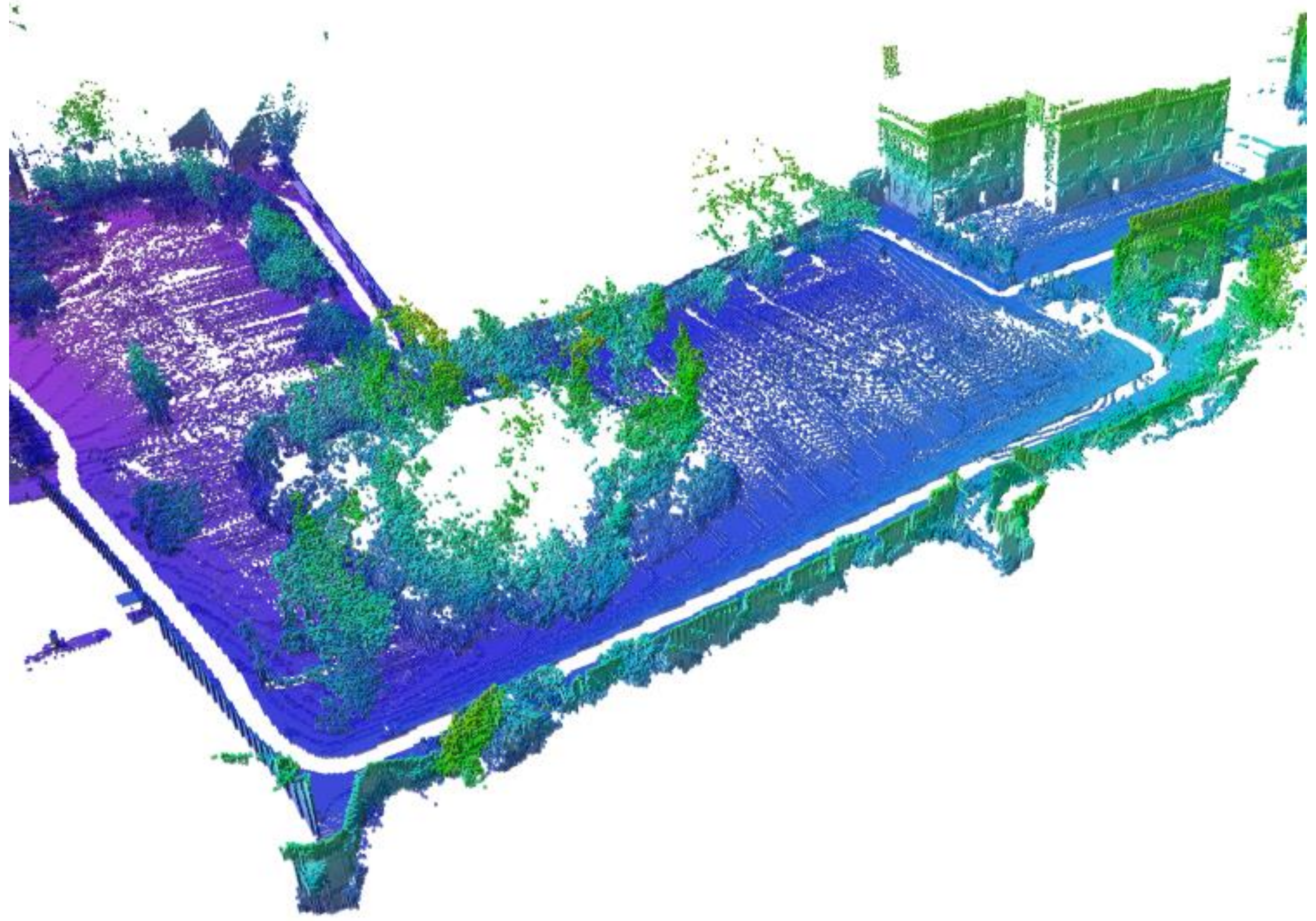
Explicit Scene Representations



https://www.anolytics.ai/blog/applications-challenges-with-3d-point-cloud-data-for-lidars/



https://docs.nframes.com/images/adc495e6b3f253ba944d2d7a29c6082f.jpg



https://octomap.github.io/newcol_big.png

Contributions

The first **neural-implicit** SLAM algorithm that estimates camera poses while training a NeRF as the map.

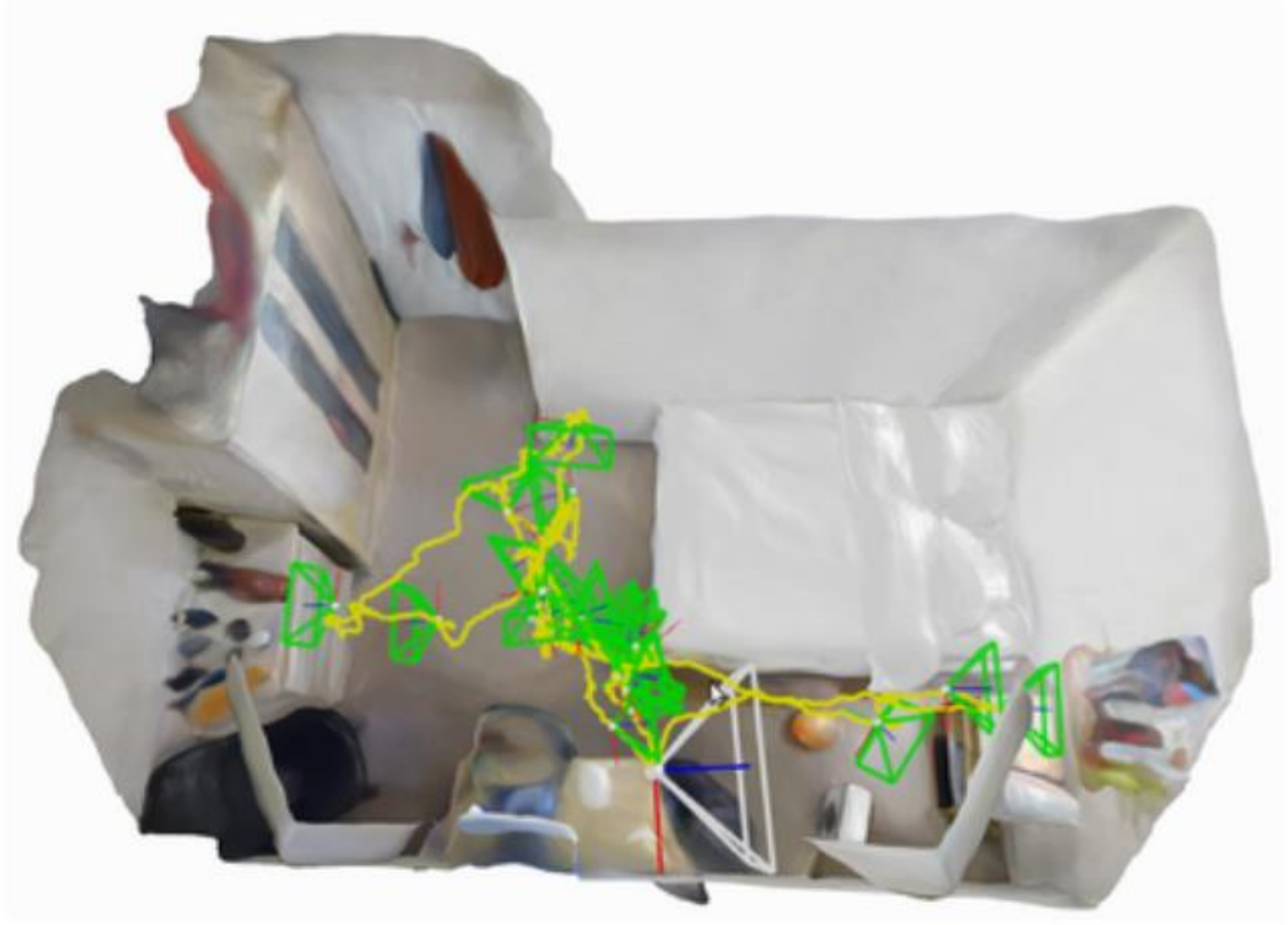
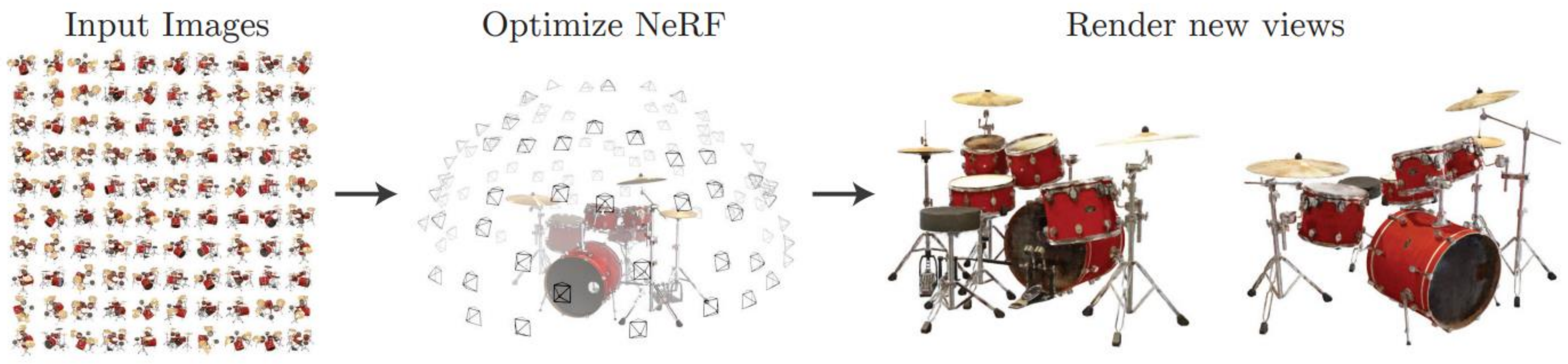


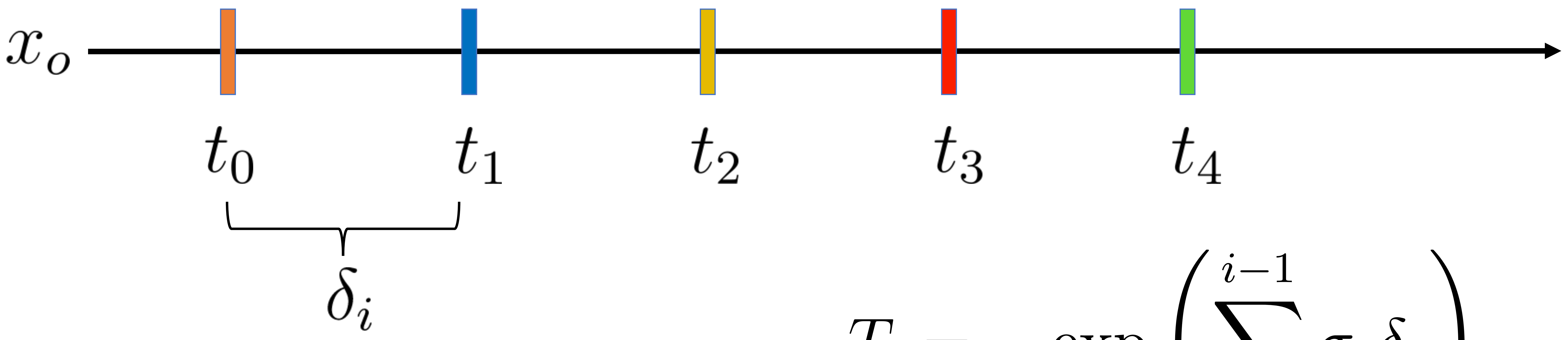
Figure 1: Room reconstruction from real-time iMAP with an Azure Kinect RGB-D camera, showing watertight scene model, camera tracking and automatic keyframe set.

Background: NeRFs



https://arxiv.org/pdf/2003.08934.pdf

Background: NeRFs



$$T_i = -\exp\left(\sum_{j=0}^{i-1} \sigma_j \delta_j\right)$$

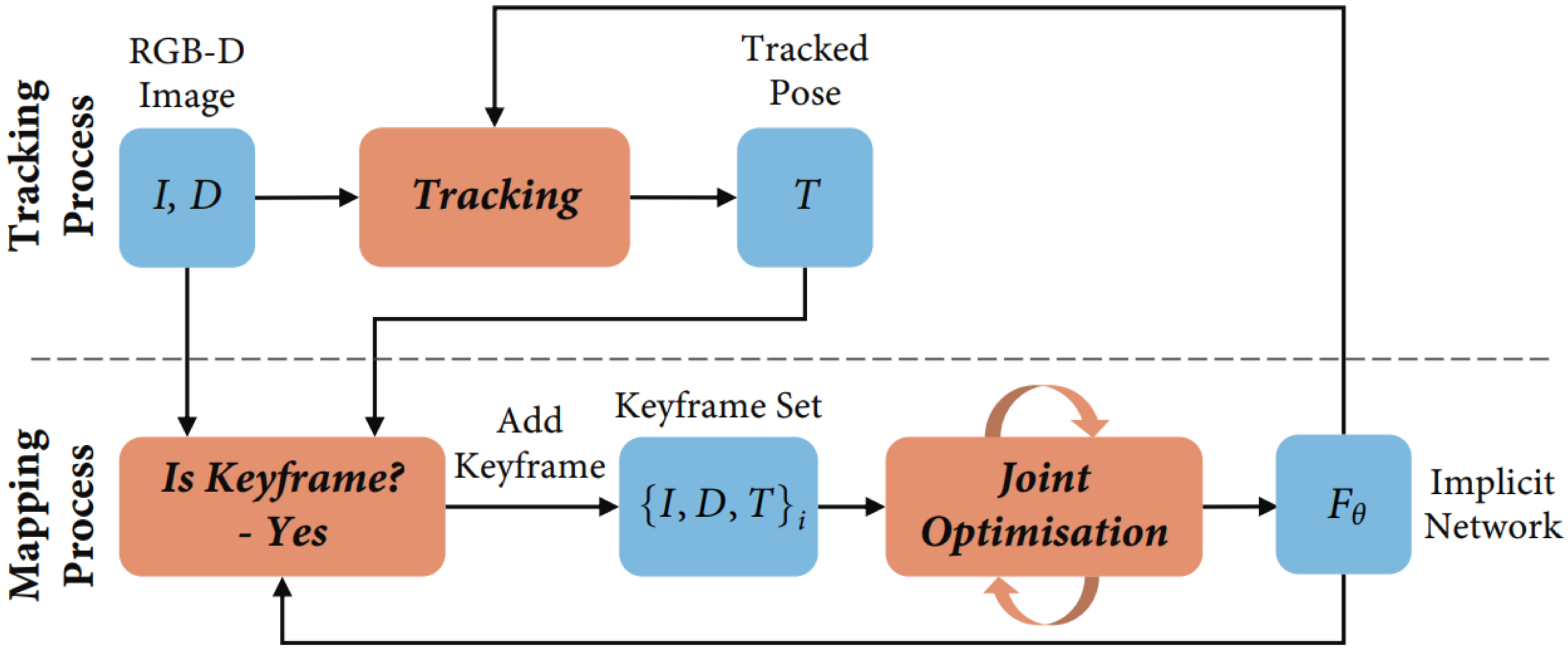
$$x_i = x_o + t_i \cdot d$$

$$\sigma_i, c_i = F(x_i, d; \Theta)$$

$$\alpha_i = T_i(1 - \exp(-\sigma_i \delta_i))$$

$$C = \sum_{i=1}^{N} \alpha_i c_i$$

System Architecture

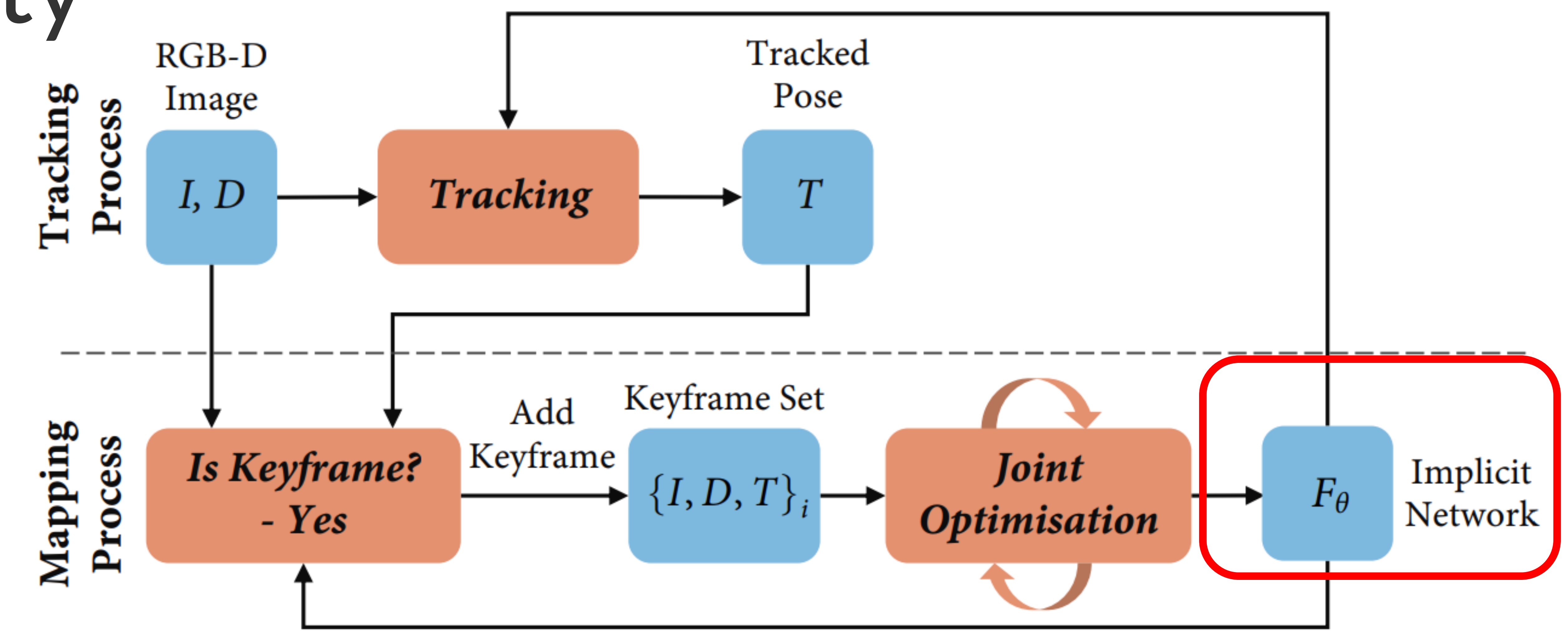


Implicit Map Network

- 4 hidden layers of size 256
- Input 3d coordinate
- 2 outputs:
  - Color and volume density

$$\mathbf{p} = (x, y, z)$$

$$F_\theta(\mathbf{p}) = (\mathbf{c}, \rho)$$



Joint Optimization

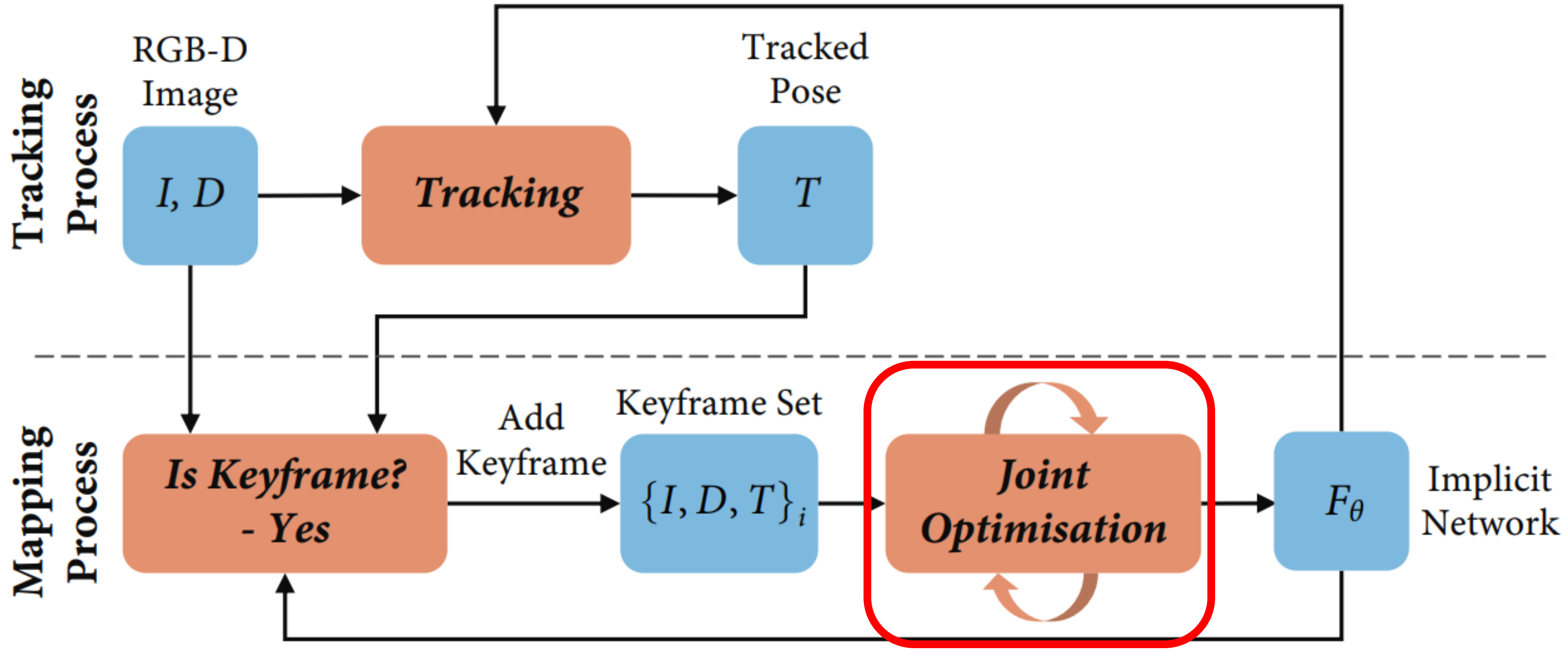
$$\min_{\theta,\{T_i\}} (L_g + \lambda_p L_p)$$

$$L_p = \frac{1}{M} \sum_{i=1}^{W} \sum_{(u,v)\in s_i} e_i^p[u,v]$$

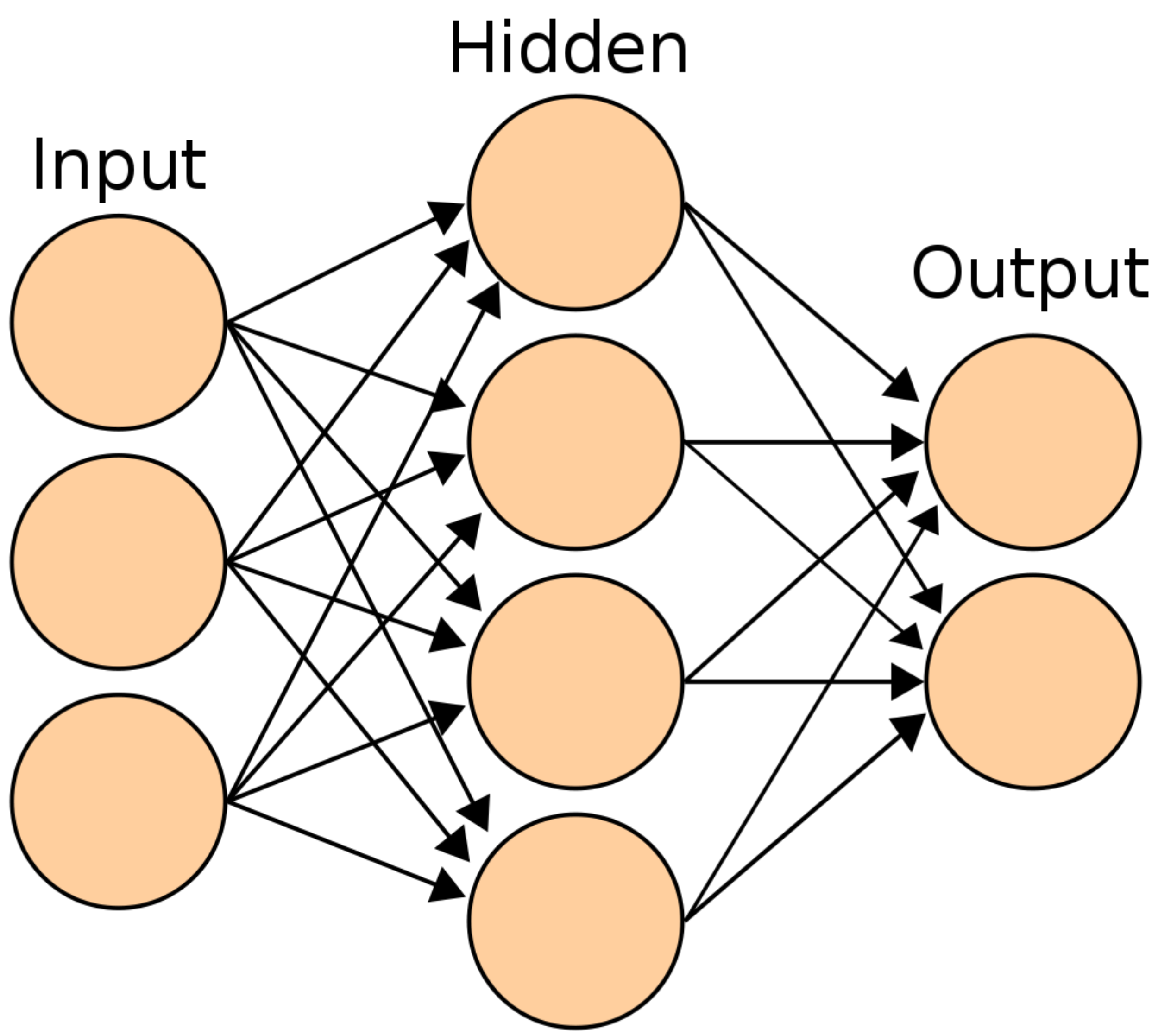
Photometric Loss (L1-norm loss)

$$L_g = \frac{1}{M} \sum_{i=1}^{W} \sum_{(u,v)\in s_i} \frac{e_i^g[u,v]}{\sqrt{\hat{D}_{var}[u,v]}}$$

Geometric Loss



Downsides of MLPs



https://en.wikipedia.org/wiki/Artificial_neural_network#/media/File:Artificial_neural_network.svg

What might happen when an MLP is optimized based off recent scene data after a long time?

Forgetting of the beginning of the scene **(Catastrophic Forgetting)**

Keyframe and Active Sampling

Check how much frame overlaps existing model
Threshold of 0.65

$$P = \frac{1}{|s|} \sum_{(u,v) \in s} \mathbb{1}\left(\frac{\left| D[u,v] - \hat{D}[u,v] \right|}{D[u,v]} < t_D \right)$$

Segment image into uniform grids
Calculate geometric loss for sample points in each grid.

$$L_i[j] = \frac{1}{|r_j|} \sum_{(u,v) \in r_j} e_i^g[u,v] + e_i^p[u,v],$$

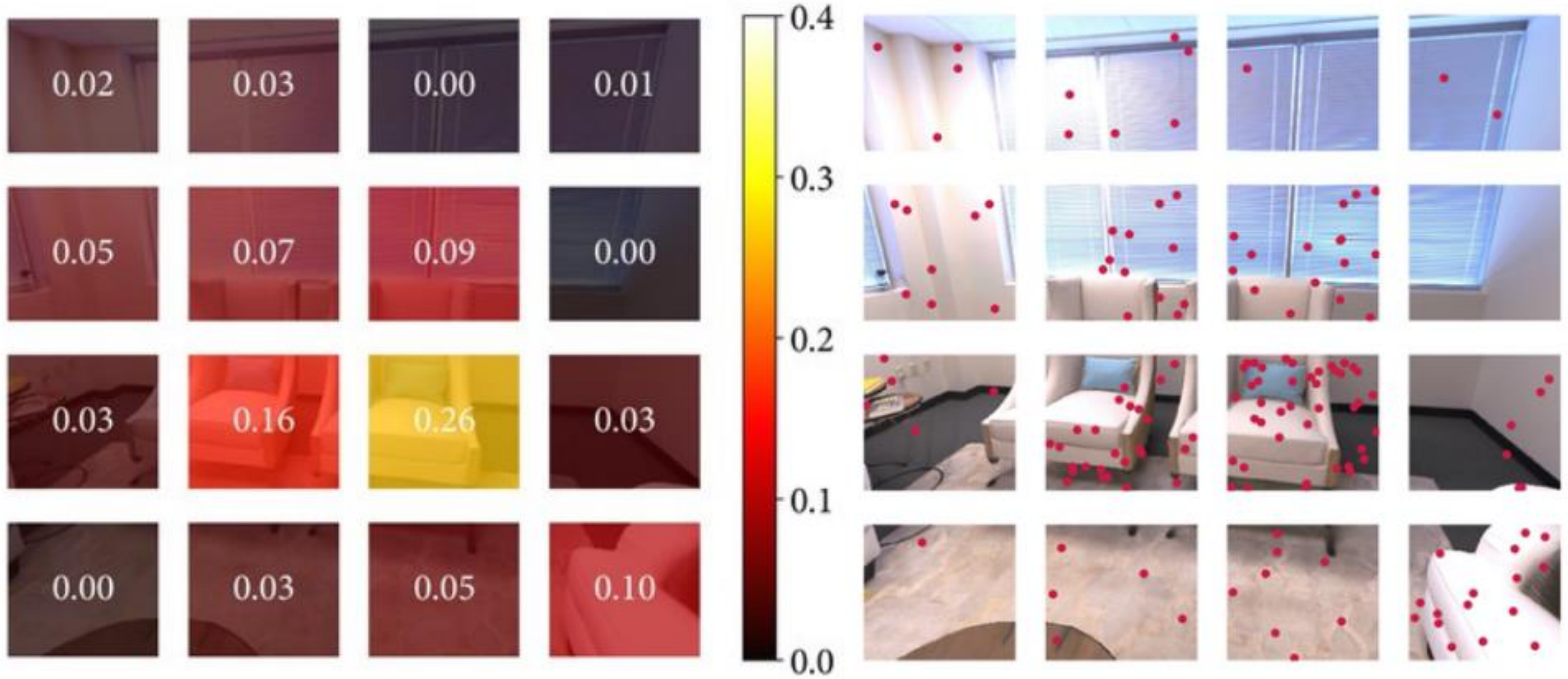
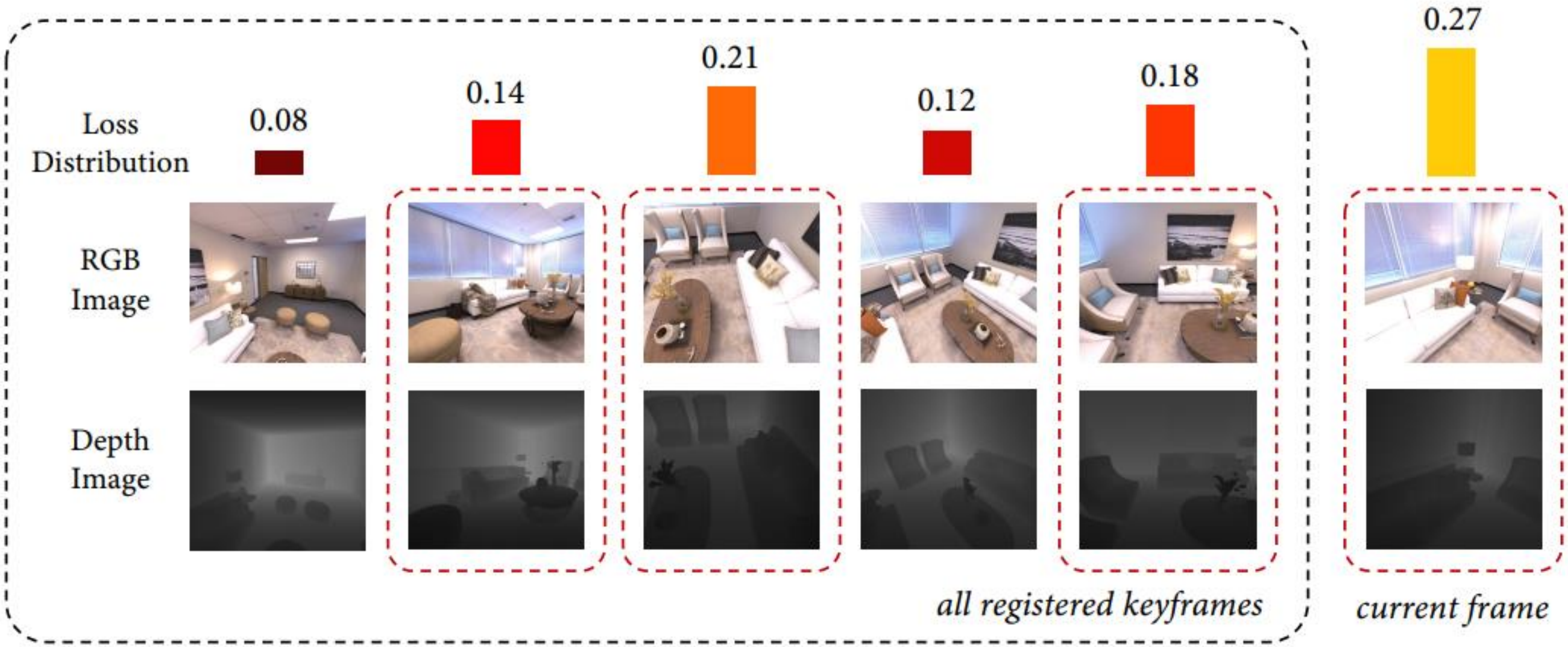


Figure 3: Image Active Sampling. Left: a loss distribution is calculated across an image grid using the geometric loss from a set of uniform samples. Right: active samples are further allocated proportional to the loss distribution.

Keyframe Buffer



Results

- Scene Reconstruction Evaluation
  - Test on Replica Dataset

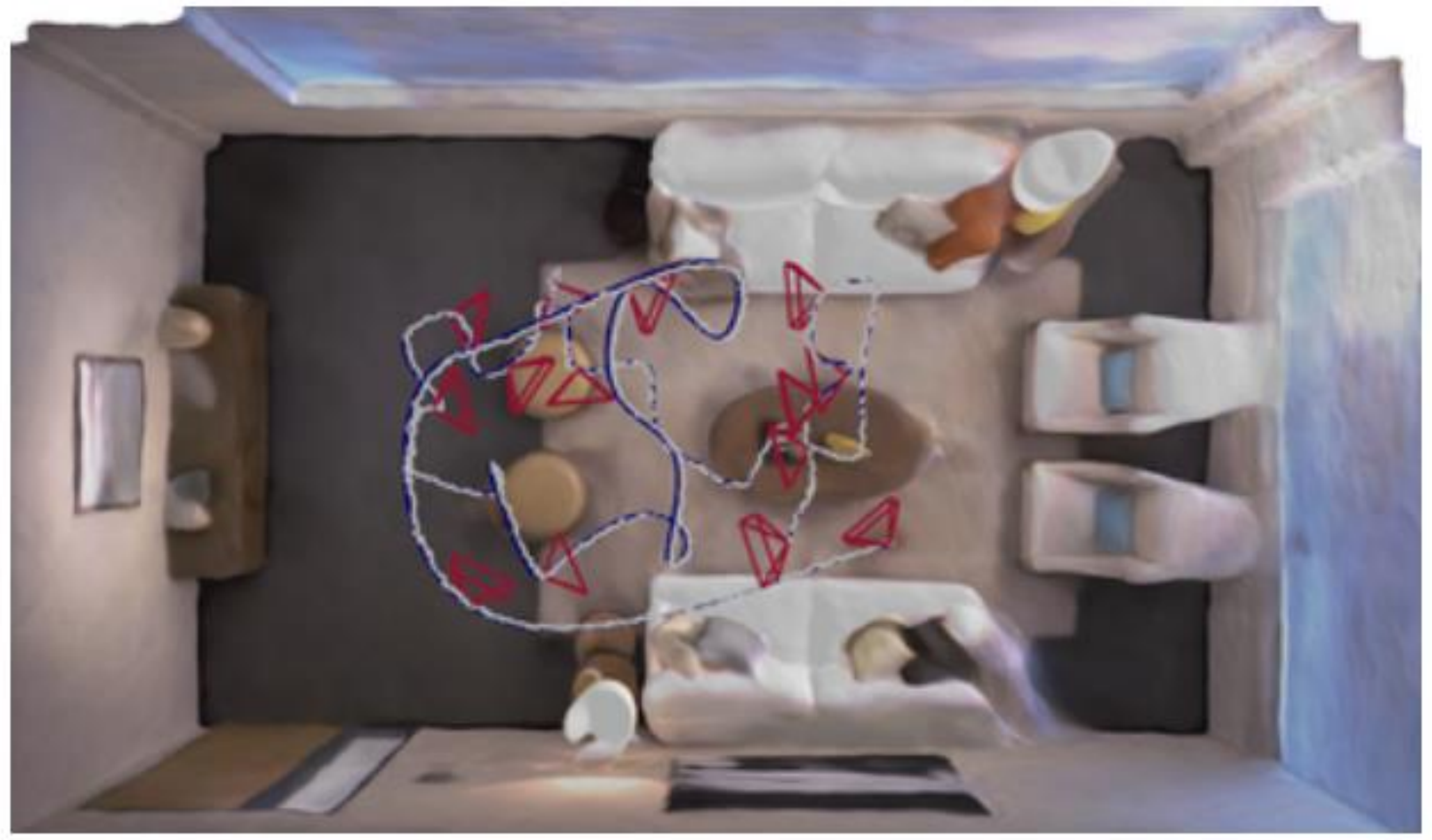
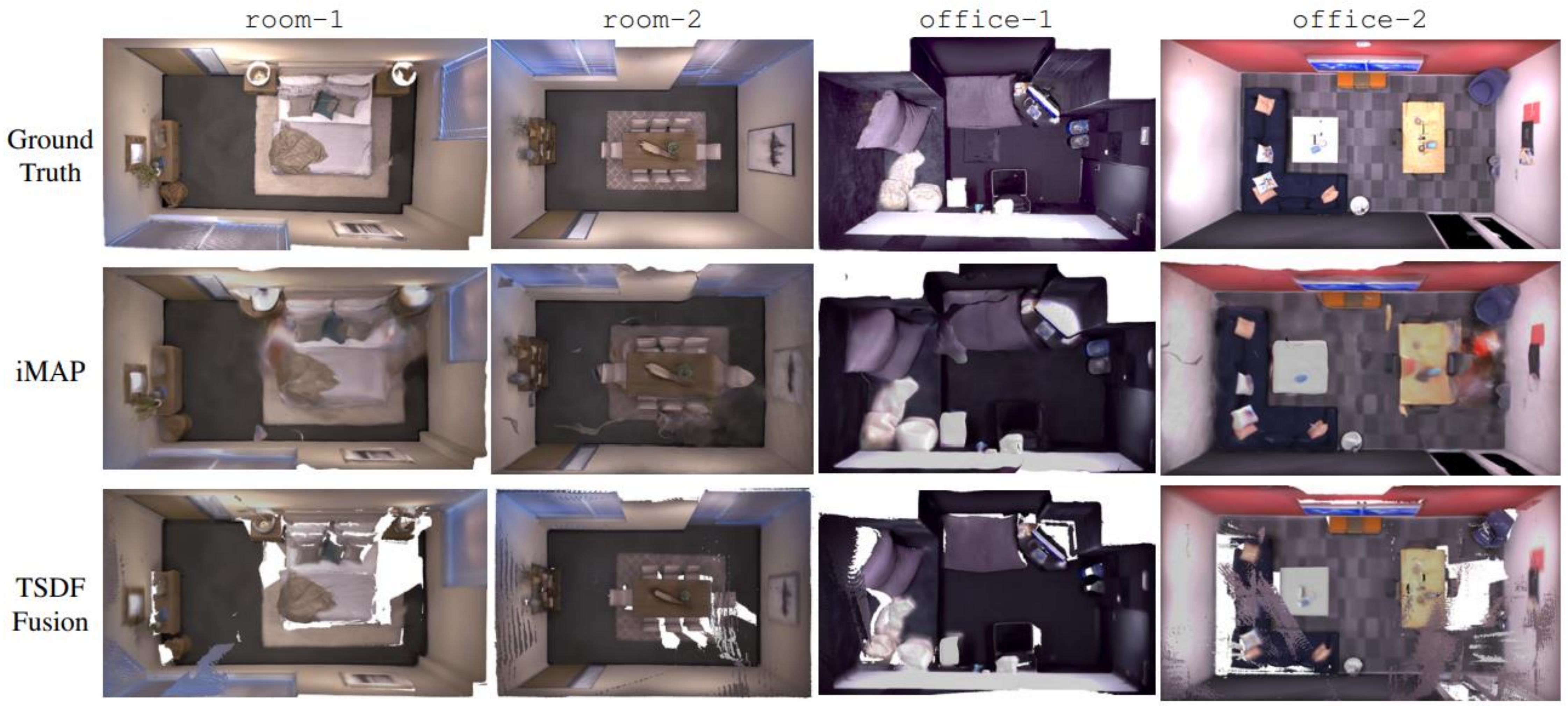


Figure 5: Reconstruction and tracking results for Replica `room-0` along with registered keyframes.

Figure 6: iMAP (left) manages to fill in unobserved regions which can be seen as holes in TSDF fusion (right).

Results

- Scene Reconstruction Evaluation
  - Test on Replica Dataset



		room-0	room-1	room-2	office-0	office-1	office-2	office-3	office-4	Avg.
iMAP	**# Keyframes**	11	12	12	10	11	10	14	11	13.37
	Acc. [cm]	3.58	3.69	4.68	5.87	3.71	4.81	4.27	4.83	4.43
	Comp. [cm]	5.06	4.87	5.51	6.11	5.26	5.65	5.45	6.59	**5.56**
	Comp. Ratio [< 5cm %]	83.91	83.45	75.53	77.71	79.64	77.22	77.34	77.63	**79.06**
TSDF Fusion	**Acc.** [cm]	4.21	3.08	2.88	2.70	2.66	4.27	4.07	3.70	**3.45**
	Comp. [cm]	5.04	4.35	5.40	10.47	10.29	6.43	6.26	4.78	6.63
	Comp. Ratio [< 5cm %]	76.90	79.87	77.79	79.60	71.93	71.66	65.87	77.11	75.09

Results

- Trajectory Evaluation
  - Test on TUM RGB-D Dataset

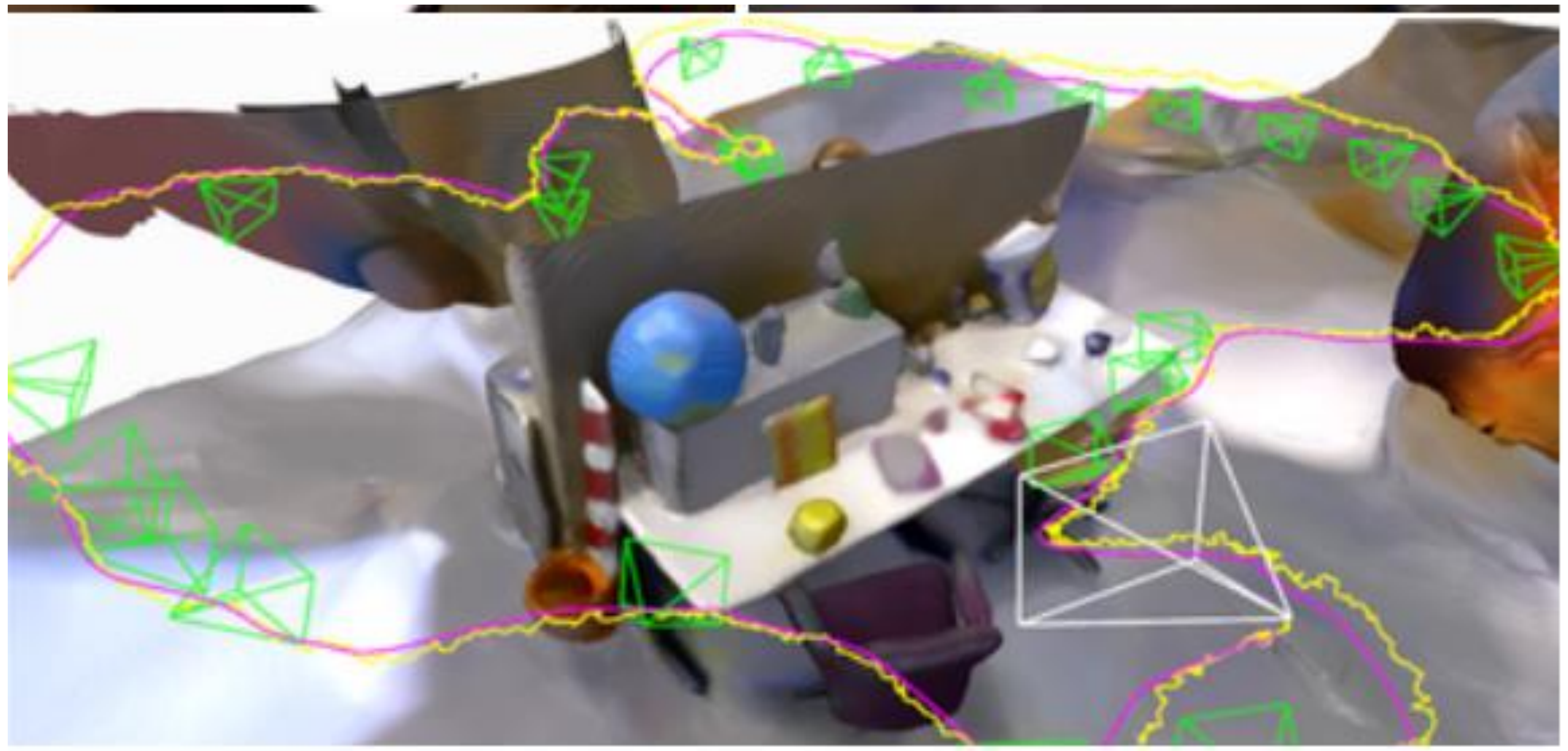


Figure 10: iMAP reconstruction results for TUM dataset.

	fr1/desk (cm)	fr2/xyz (cm)	fr3/office (cm)
iMAP	4.9	2.0	5.8
BAD-SLAM	1.7	1.1	1.73
Kintinuous	3.7	2.9	3.0
ORB-SLAM2	1.6	0.4	1.0

Table 3: ATE RMSE in cm on TUM RGB-D dataset.

Conclusions

- First real-time RGB-D SLAM based on NeRF
- Proposed loss-guided pixel sampling to achieve real-time SLAM
- Proposed intelligent keyframe selection to avoid forgetting problem in MLP

Limitations and Directions for Future Work

- Limitations
  - Only works in indoor room-scale scenes
  - Cannot handle rapid camera motion
- Future directions for iMAP include how to make more structured and compositional representations that reason explicitly about the self-similarity in scenes.

Thank you

NeRF-SLAM

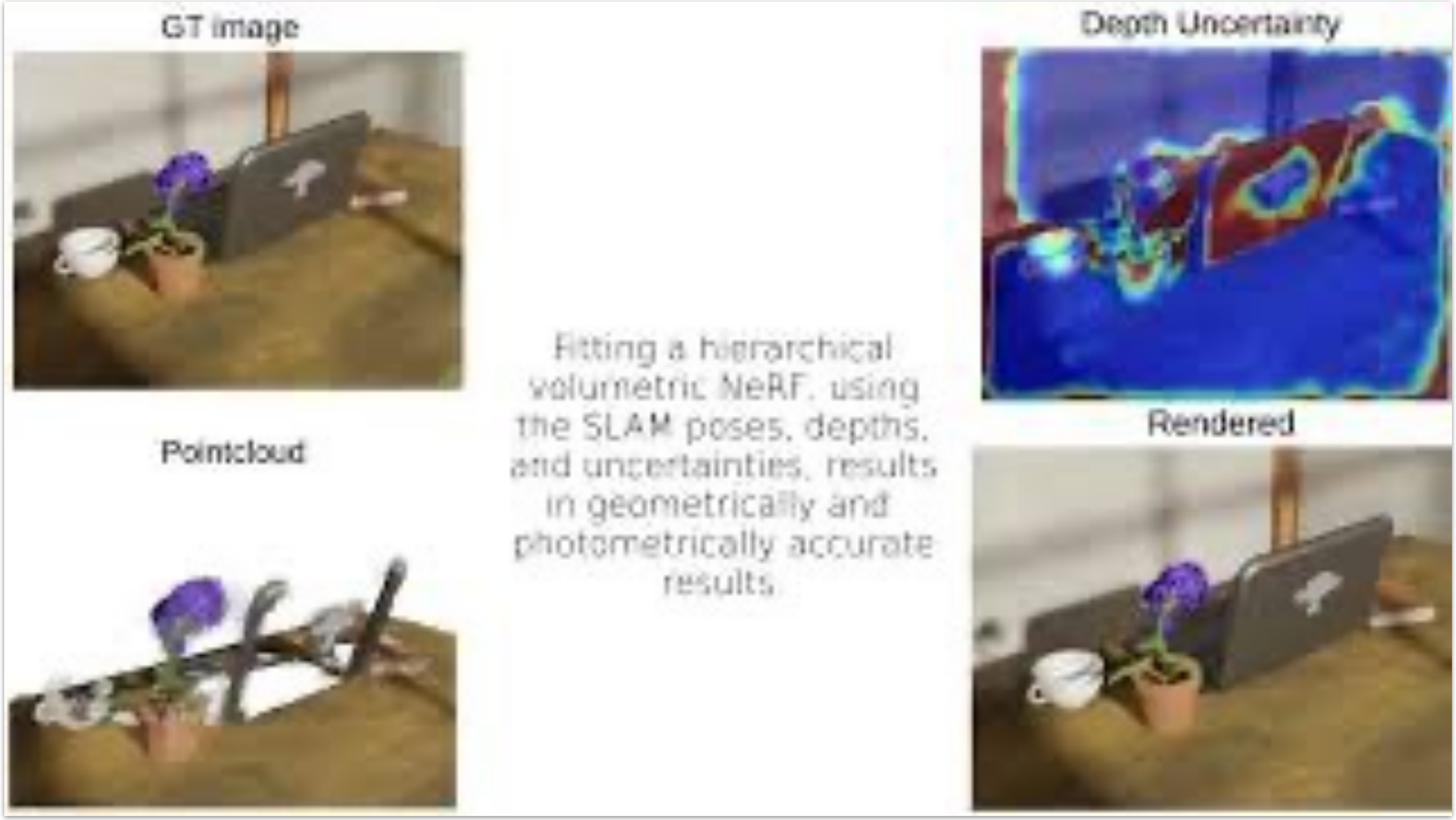
Real-Time Dense Monocular SLAM with Neural Radiance Fields

By: Antoni Rosinol, John J. Leonard, Luca Carlone

Presented by: Jack Fenton, Walter Xu

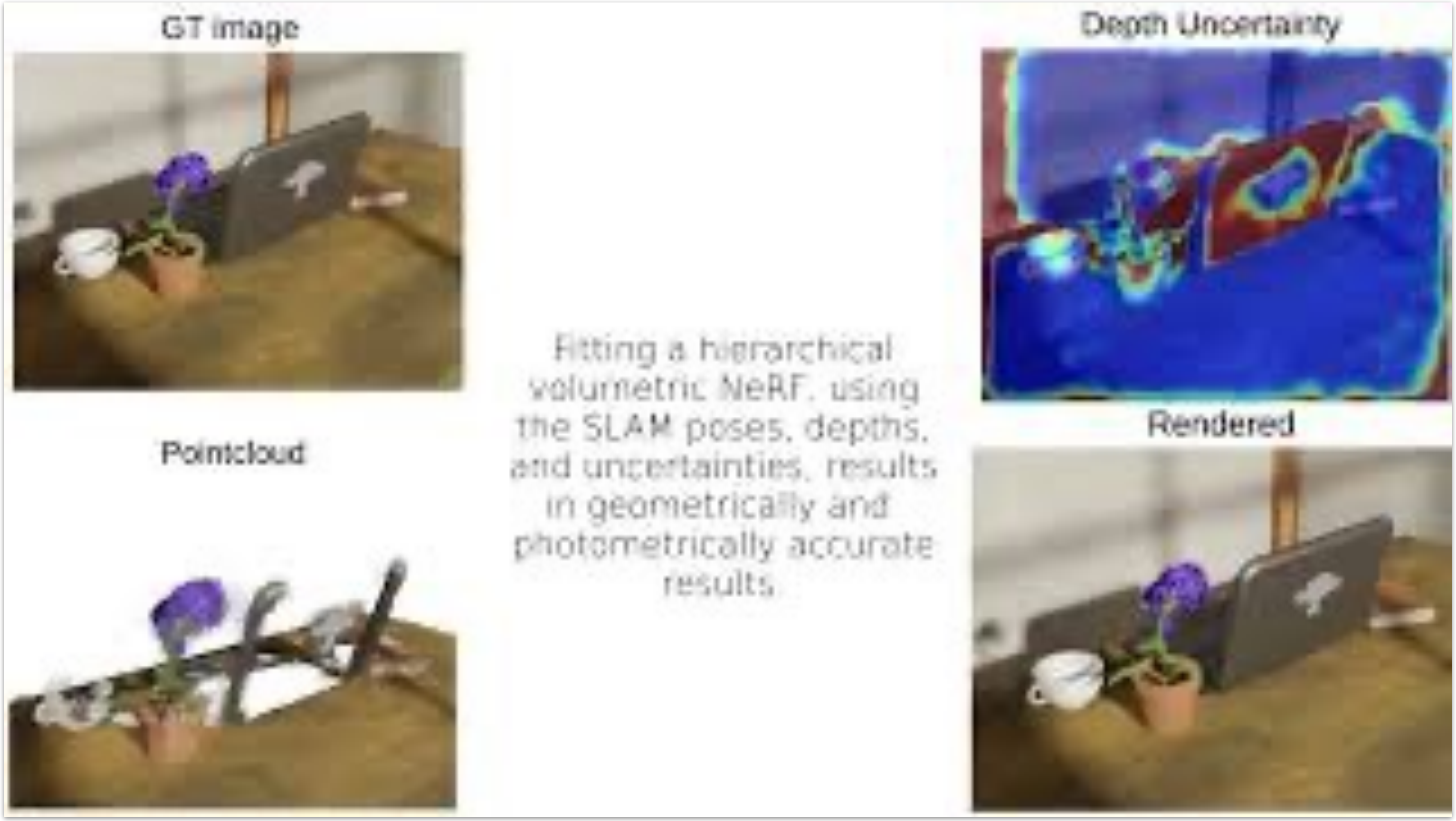
Why Do You Care?

Why Do You Care?



Why Do You Care?

Why Do You Care?



The Authors

- Antoni Rosinol
  - Ph.D. Candidate @ MIT
  - Guest Lecturer for ROB 530 in Spring 2022
  - Author of: Sigma-Fusion, Ultimate SLAM?, Kimera, 3D Dynamic Scene Gra

- John Leonard
  - MECHE Professor @ MIT

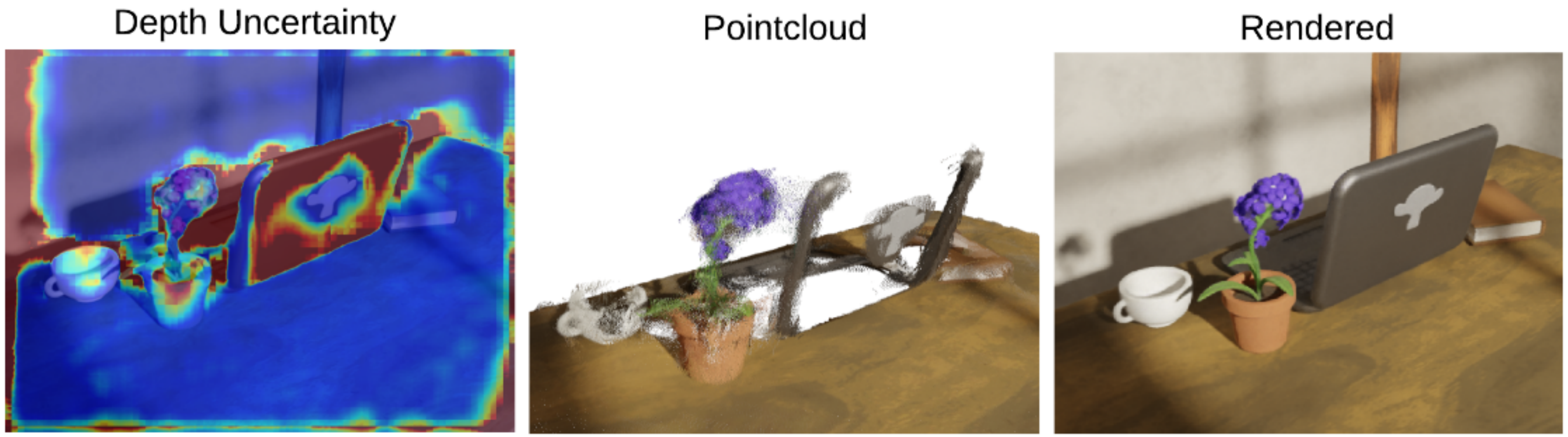
- Sergey Levine
  - AeroAstro Professor @ MIT

Motivation

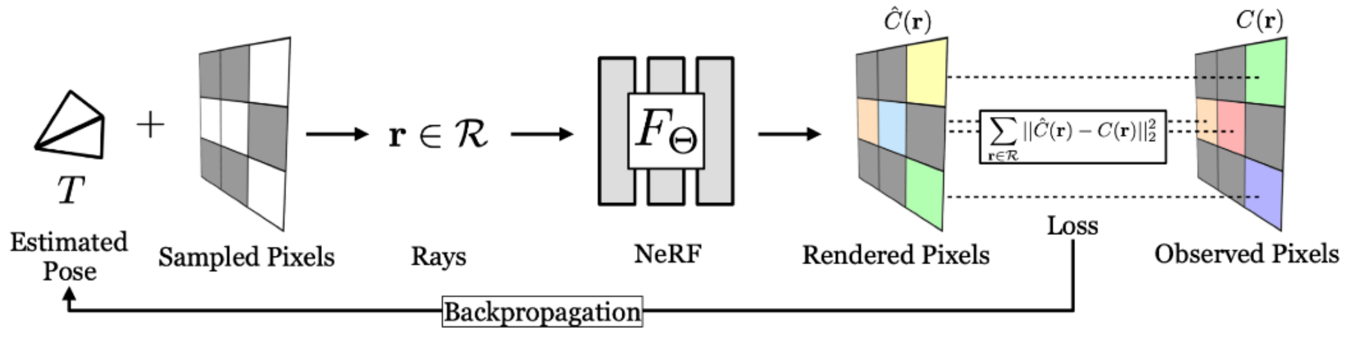
1.NeRFs allow for high-fidelity map of environment

1.NeRFs need ground truth poses and/or depth maps
→ Want to run with just monocular RGB images
→ Want to account for noisy depth maps

1. NeRFs are SLOW
→ Want to run real time

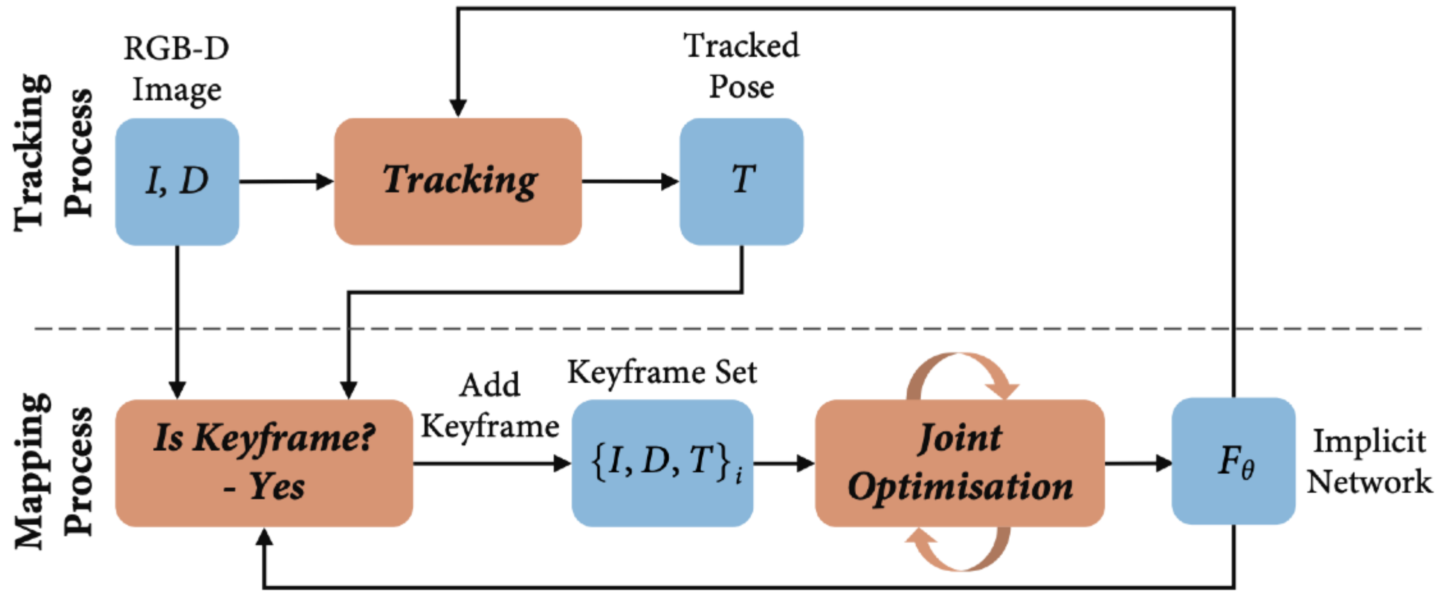


Background: SLAM with NeRFs

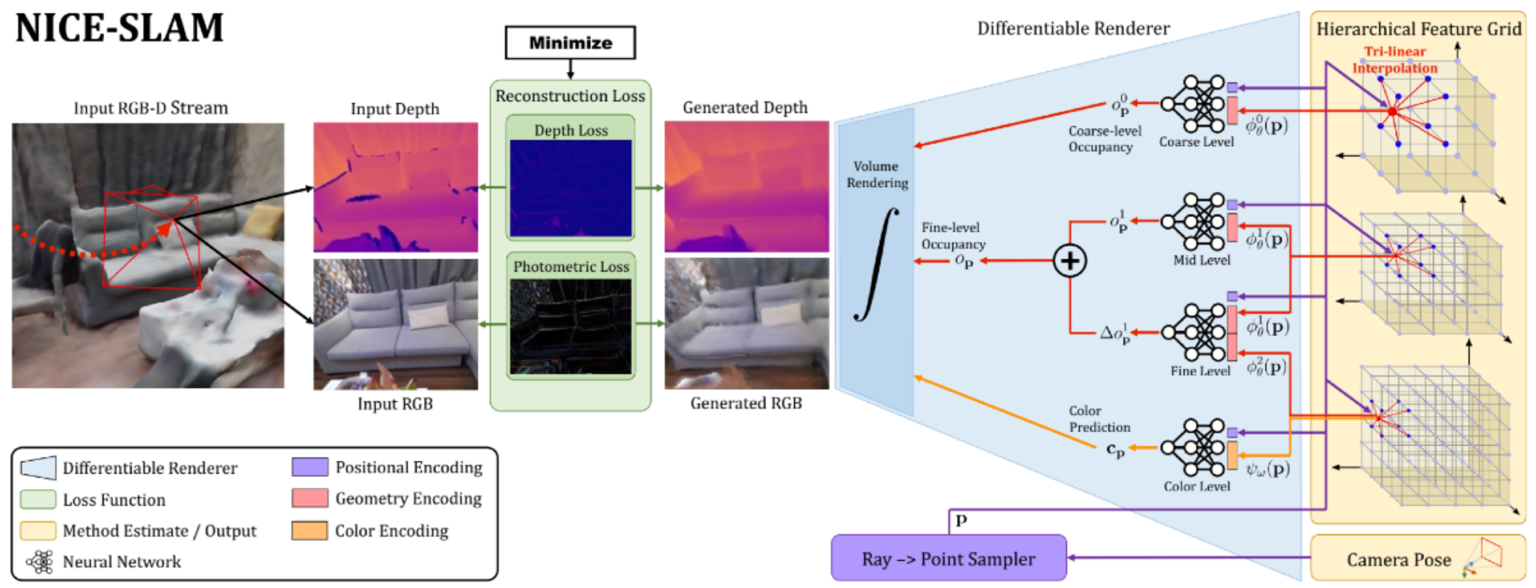


iNeRF: Inverting Neural Radiance Fields for Pose Estimation

+ Regress camera pose
- Too slow for real-time

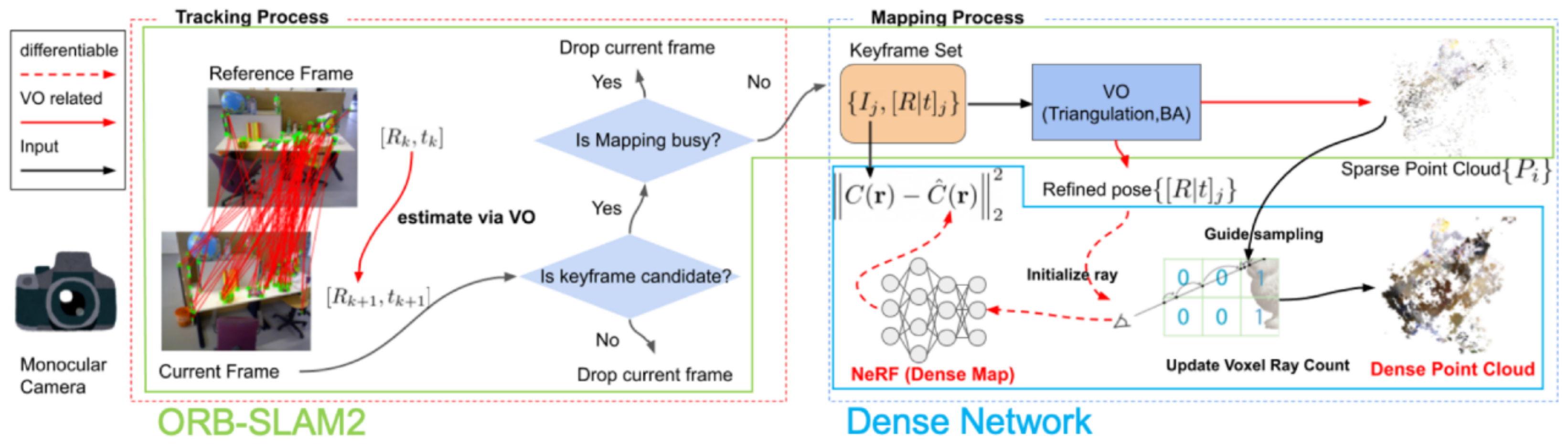


iMAP: Implicit Mapping and Positioning in Real-Time



NICE-SLAM: Neural Implicit Scalable Encoding for SLAM

+ Don’t need poses
- Need RGB-D Images



Orbeez-SLAM: A Real-time Monocular Visual SLAM with ORB Features and NeRF-realized Mapping

+ Hierarchical NeRF, RGB images
- Uncertainty in depth-map

Contributions to Problems

Contributions to Problems

1. Need Accurate NeRF Results with only RGB Images

Contributions to Problems

1. Need Accurate NeRF Results with only RGB Images
   - Dense Monocular SLAM can produce 3D poses, dense depth-maps, and probabilistic uncertainty

Contributions to Problems

1. Need Accurate NeRF Results with only RGB Images
   - Dense Monocular SLAM can produce 3D poses, dense depth-maps, and probabilistic uncertainty

Contributions to Problems

1. Need Accurate NeRF Results with only RGB Images
   - Dense Monocular SLAM can produce 3D poses, dense depth-maps, and probabilistic uncertainty

1.Need Real-Time NeRF Performance

Contributions to Problems

1. Need Accurate NeRF Results with only RGB Images
    - Dense Monocular SLAM can produce 3D poses, dense depth-maps, and probabilistic uncertainty

1. Need Real-Time NeRF Performance
    - Hash-Based Hierarchical Volumetric Neural Radiance Field

Contributions to Problems

1. Need Accurate NeRF Results with only RGB Images
   - Dense Monocular SLAM can produce 3D poses, dense depth-maps, and probabilistic uncertainty

1. Need Real-Time NeRF Performance
   - Hash-Based Hierarchical Volumetric Neural Radiance Field

Contributions to Problems

1. Need Accurate NeRF Results with only RGB Images
   - Dense Monocular SLAM can produce 3D poses, dense depth-maps, and probabilistic uncertainty

1. Need Real-Time NeRF Performance
   - Hash-Based Hierarchical Volumetric Neural Radiance Field

1. Account for uncertainty in depth map

Contributions to Problems

1. Need Accurate NeRF Results with only RGB Images
   - Dense Monocular SLAM can produce 3D poses, dense depth-maps, and probabilistic uncertainty

1. Need Real-Time NeRF Performance
   - Hash-Based Hierarchical Volumetric Neural Radiance Field

1. Account for uncertainty in depth map
   - Probabilistic Volumetric Fusion (σ-Fusion)

Contributions to Problems

1. Need Accurate NeRF Results with only RGB Images
   - Dense Monocular SLAM can produce 3D poses, dense depth-maps, and probabilistic uncertainty

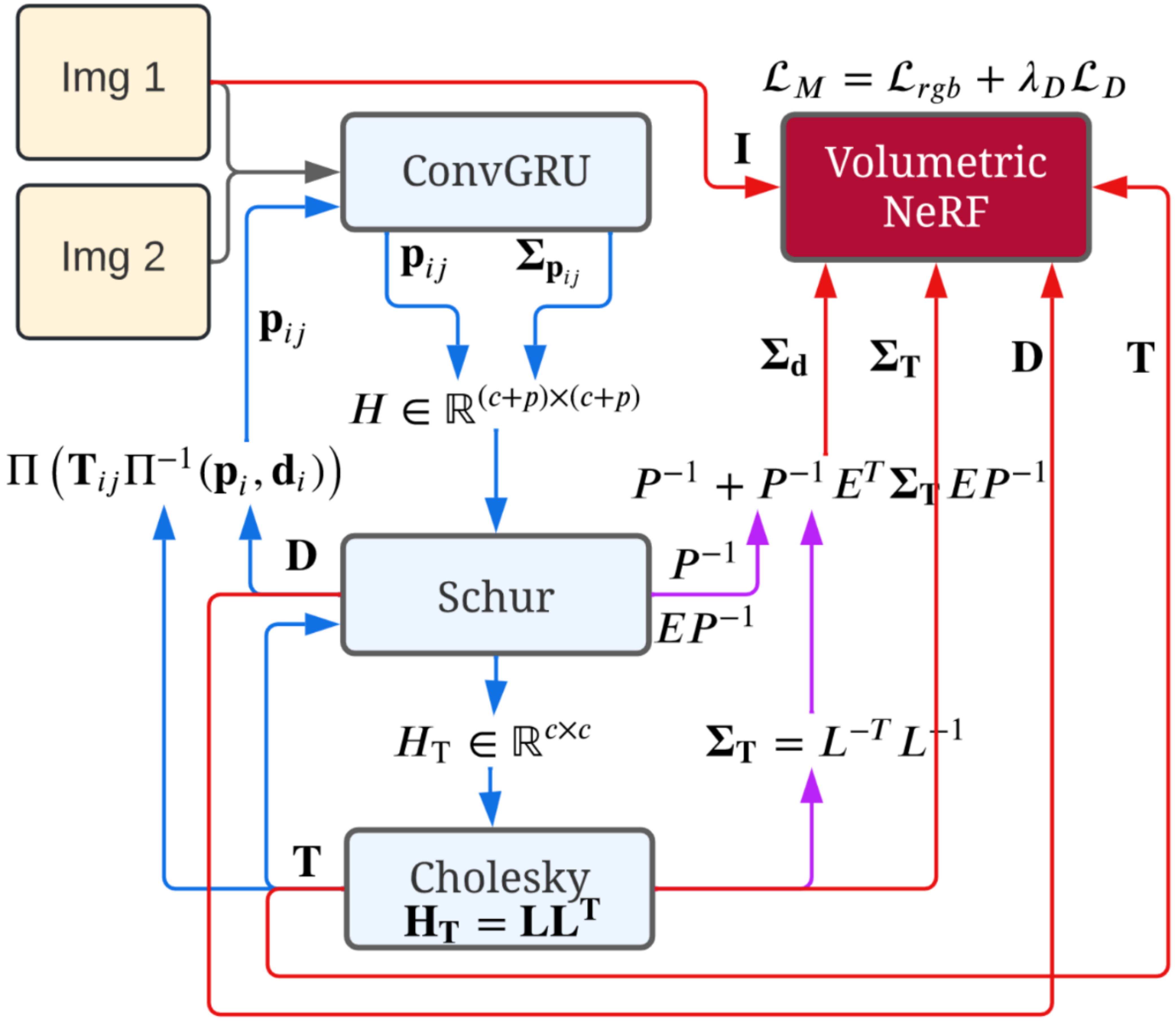
1. Need Real-Time NeRF Performance
   - Hash-Based Hierarchical Volumetric Neural Radiance Field

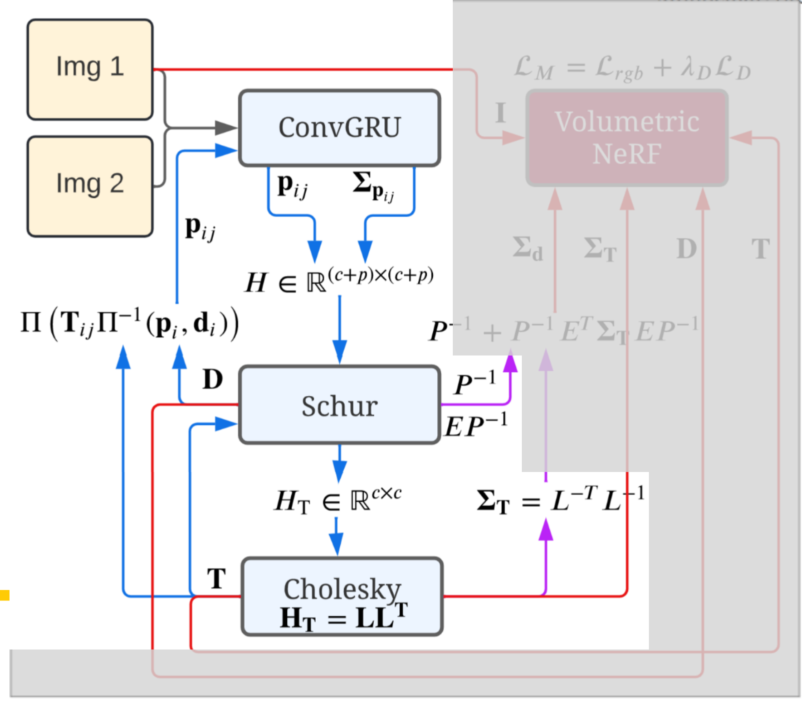
1. Account for uncertainty in depth map
   - Probabilistic Volumetric Fusion (σ-Fusion)

System Architecture

1. DROID-SLAM - tracking module
2. σ-fusion - uncertainty module
3. Instant-NGP - mapping module

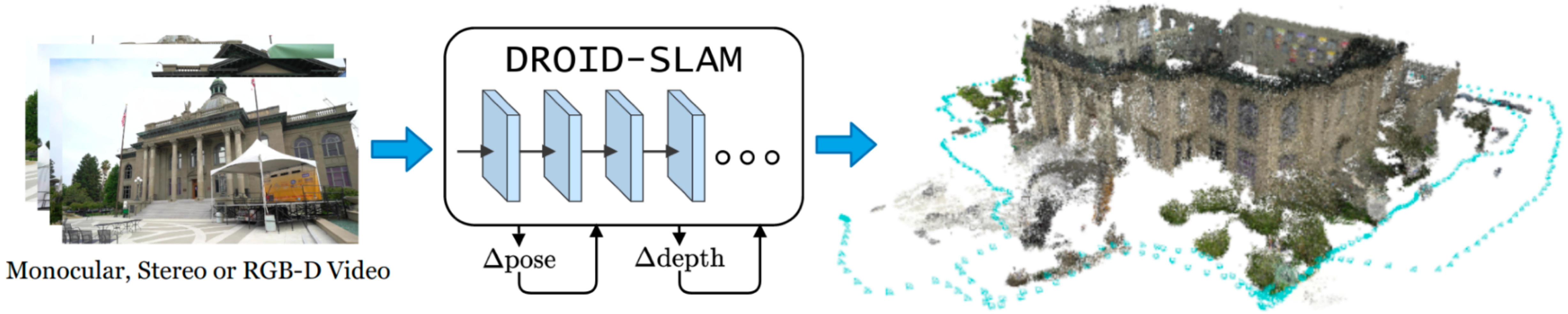
Key:
Blue – DROID-SLAM
Magenta – Sigma-Fusion
Red – Instant-NGP

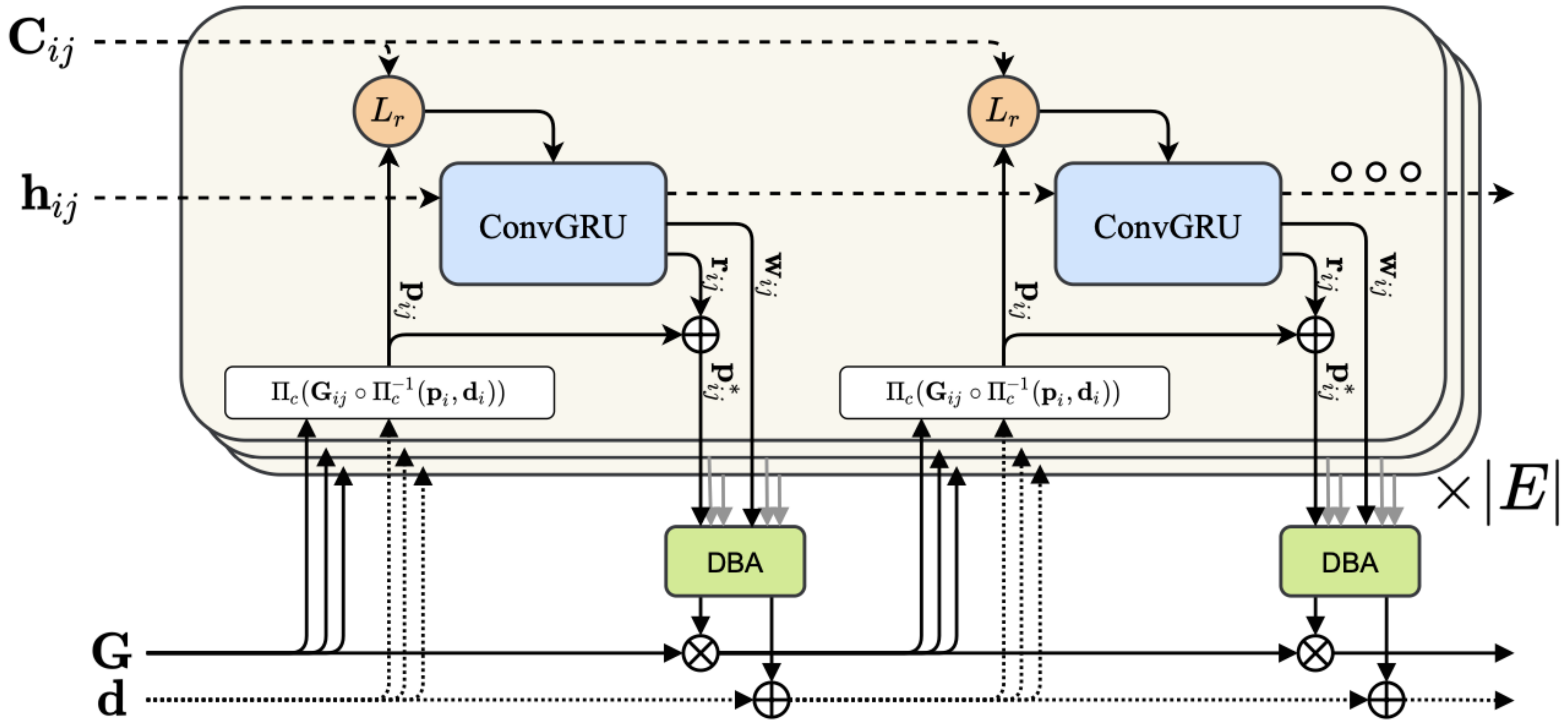




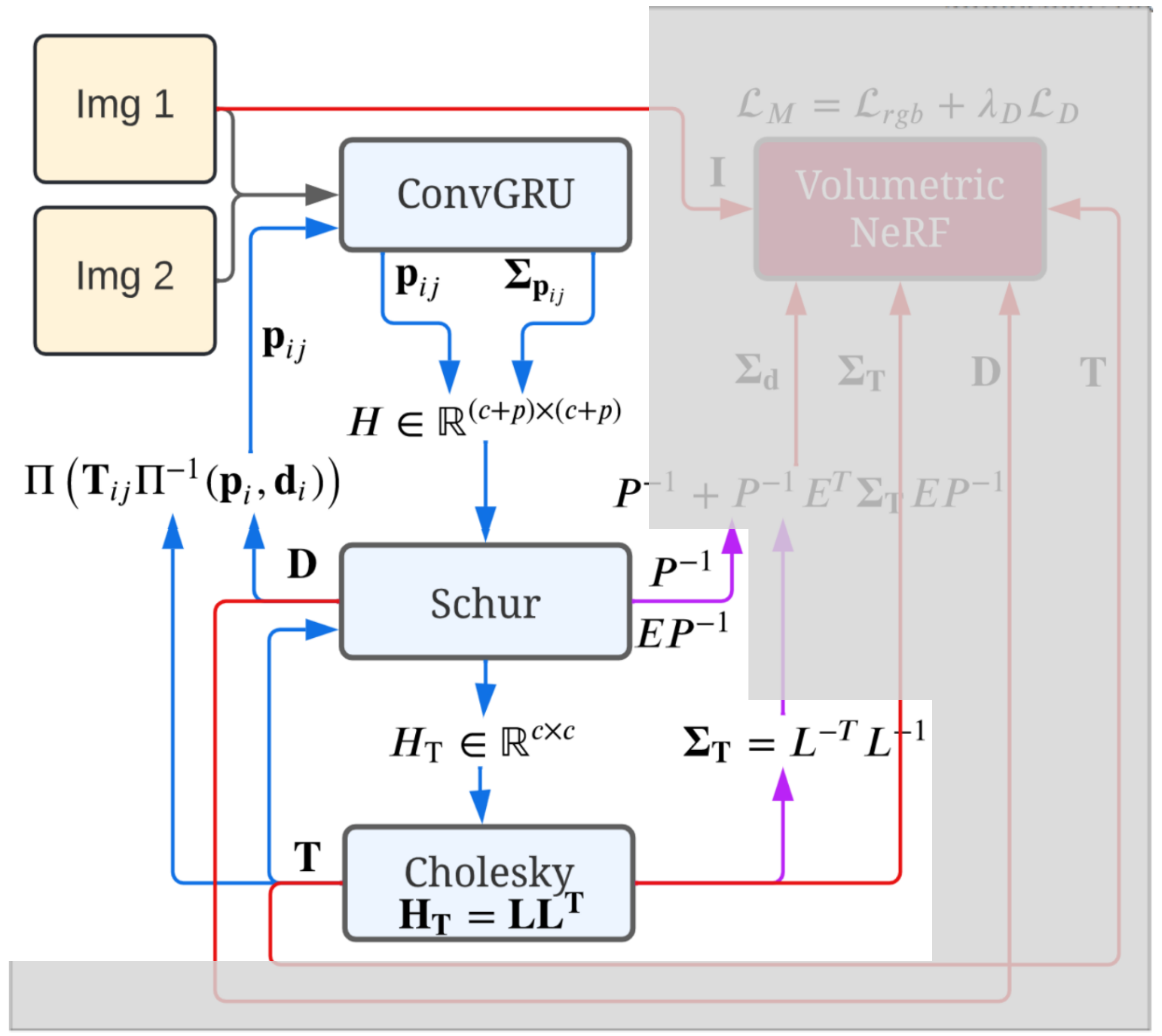
DROID-SLAM

- Dense monocular SLAM
- Track optical flow and depth map
- Hybrid of direct & indirect
  - Achieve smoother objective function
  - Greater modeling capacity





DROID-SLAM Performance



Internally

p_{ij}: correspondence to map p_i into frame j

p^*_{ij}: corrected correspondence

Produces

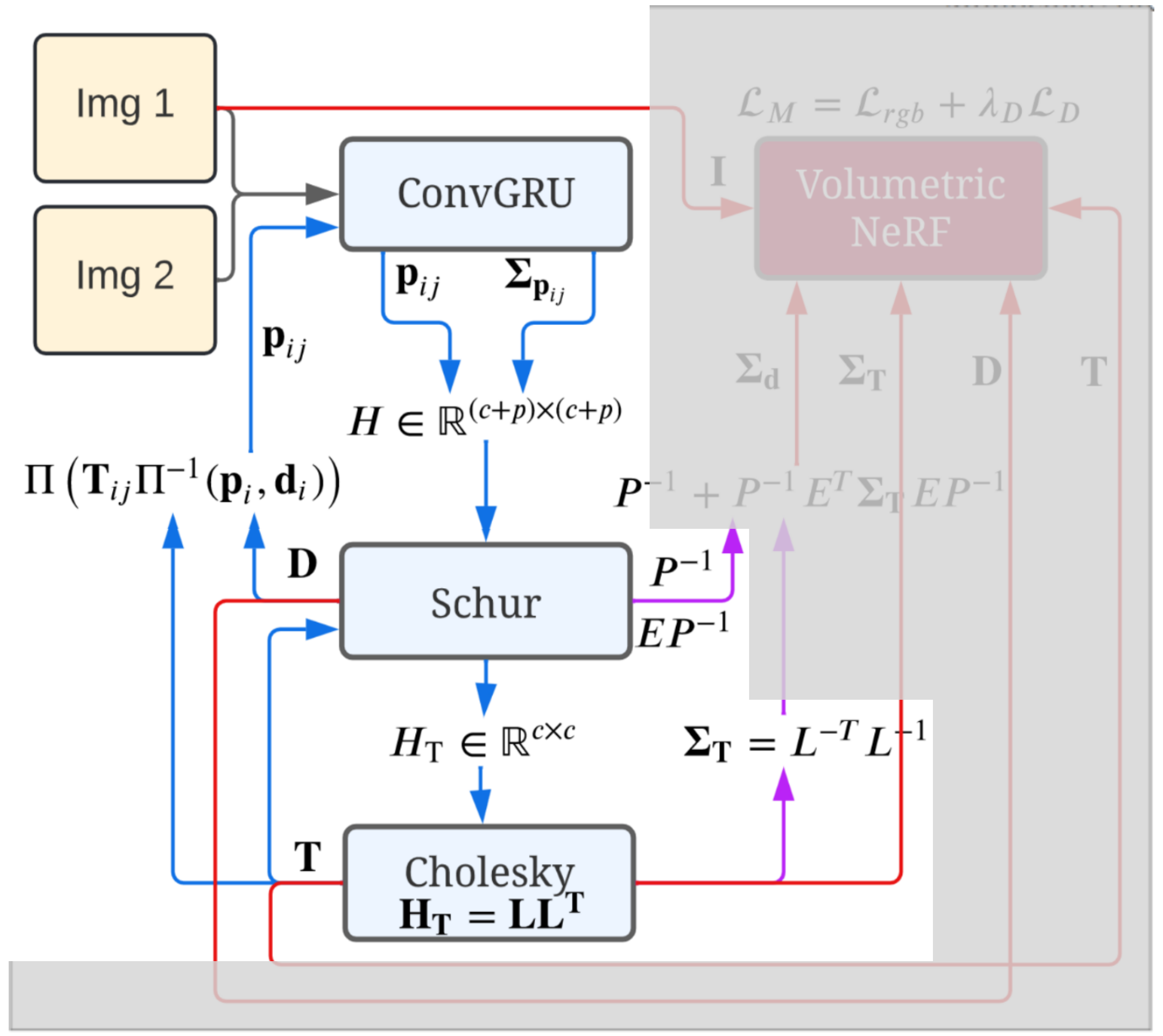
T: poses

Σ_T: pose uncertainty

D: depthmap

P: inverse depths per pixel per keyframe

DROID-SLAM Performance



Internally

p_{ij}: correspondence to map p_i into frame j

p^*_{ij}: corrected correspondence

Produces

T: poses

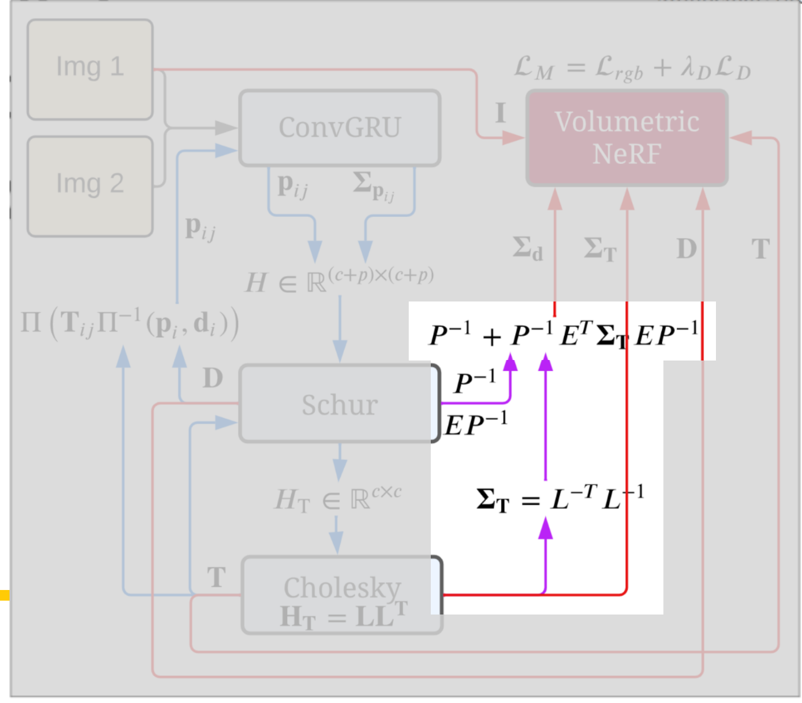
Σ_T: pose uncertainty

D: depthmap

P: inverse depths per pixel per keyframe

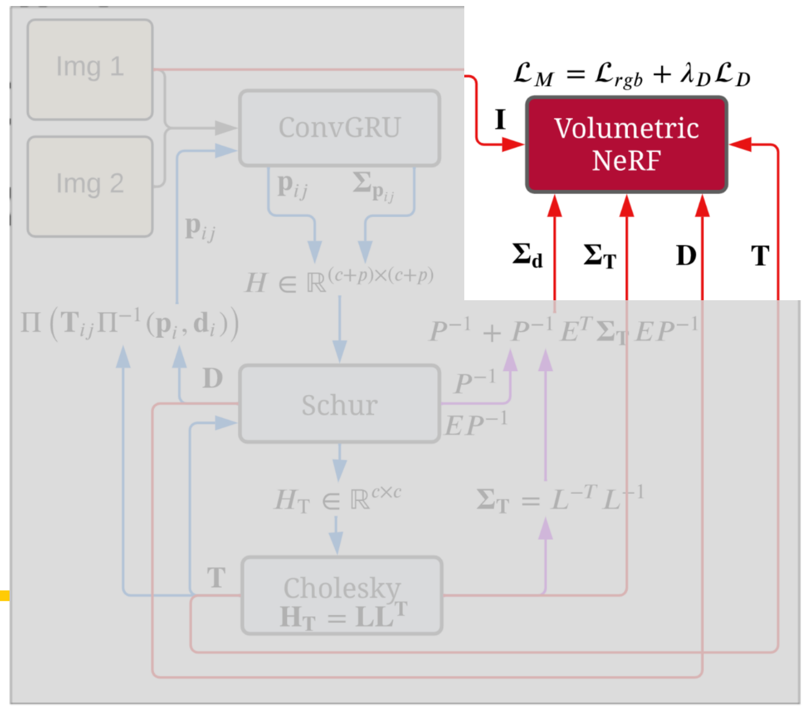
Tanks and Temples

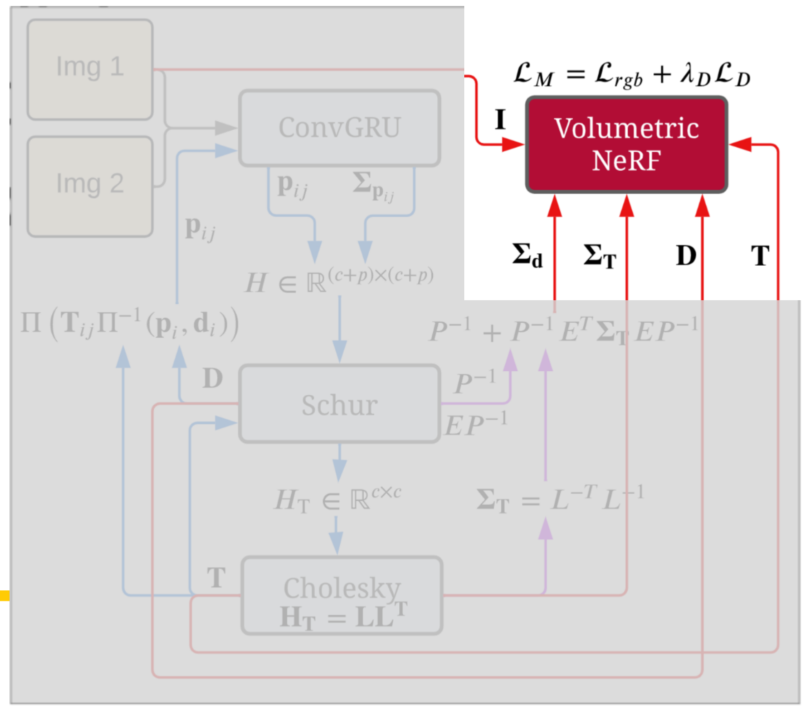
Probabilistic Volumetric Fusion
σ-Fusion



- Using DROID-SLAM's raw depth
  - Geometric error ↓
  - Photometric error ↑
- Weight depths by dense depth covariance (σ-Fusion)
  - Geometric error ↓
  - Photometric error ↓

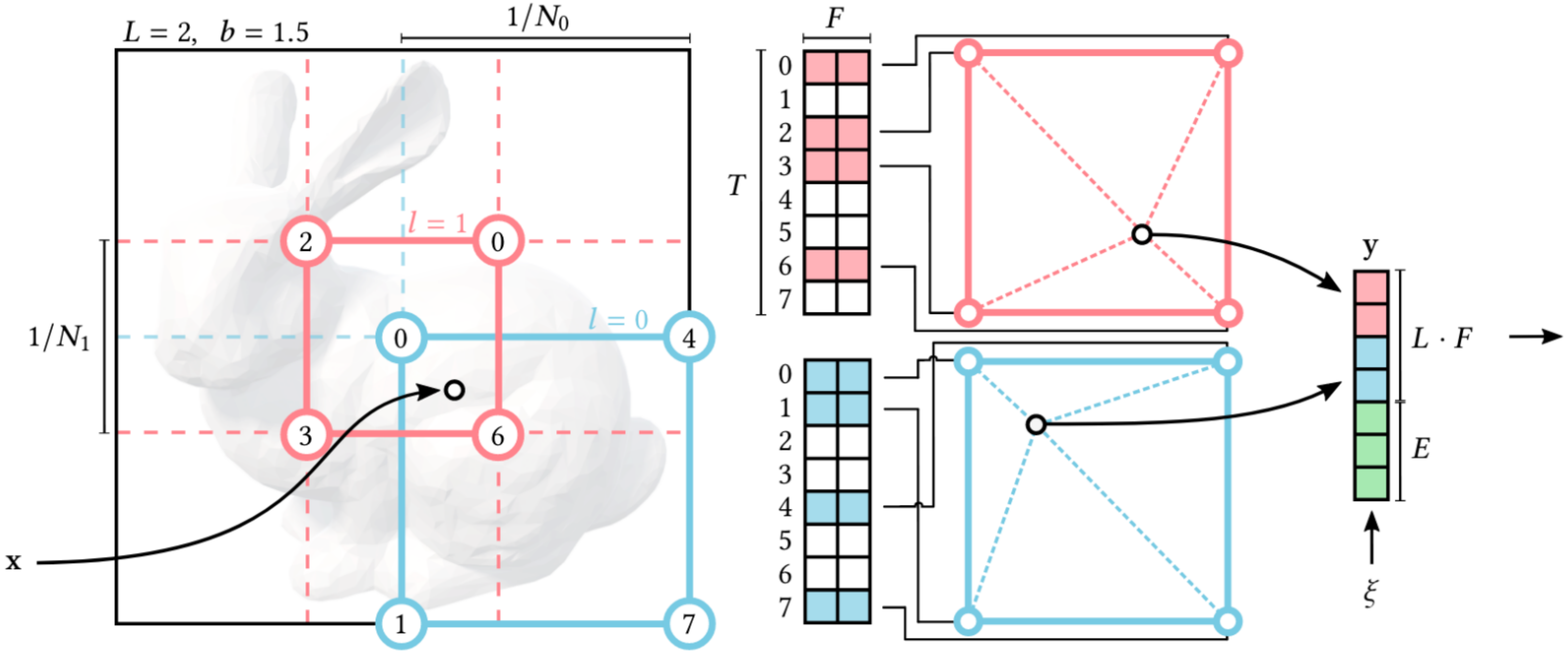
Instant NGP


$\mathcal{L}_M = \mathcal{L}_{rgb} + \lambda_D \mathcal{L}_D$
Img 1
Img 2
ConvGRU
Volumetric NeRF
I
Σ_d
Σ_T
D
T
Schur
Cholesky
$H_T = LL^T$

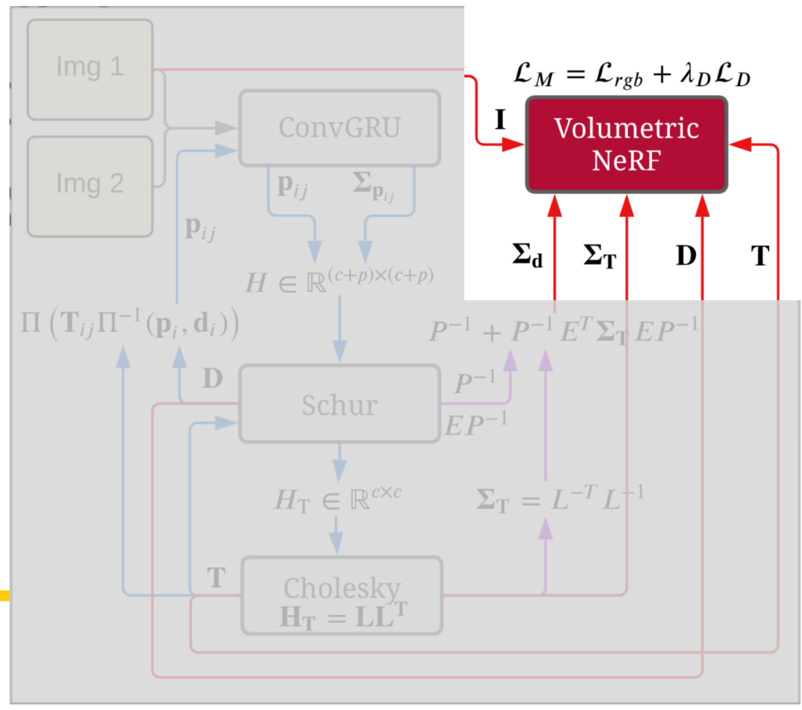


Instant NGP

Hash-based encoding



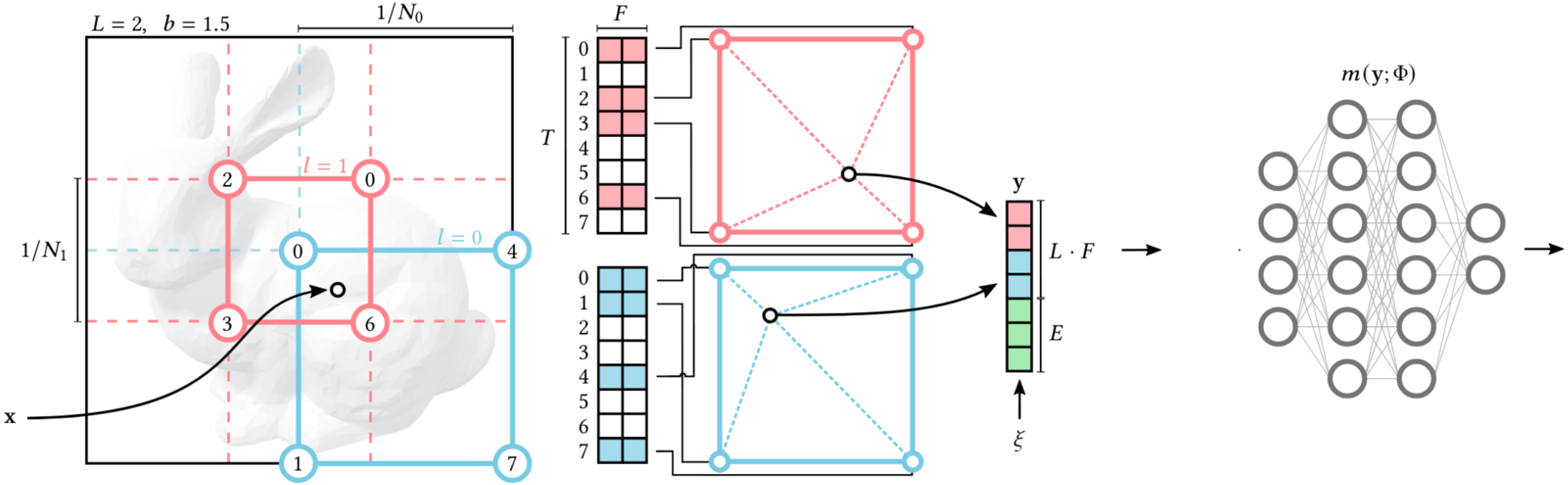
(1) Hashing of voxel vertices **(2)** Lookup **(3)** Linear interpolation **(4)** Concatenation



Instant NGP

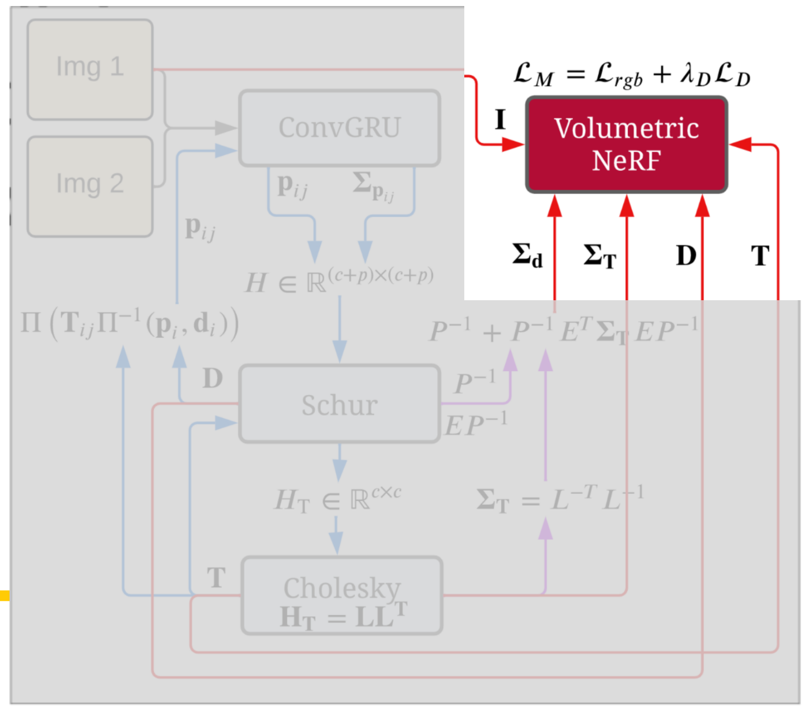
Hash-based encoding

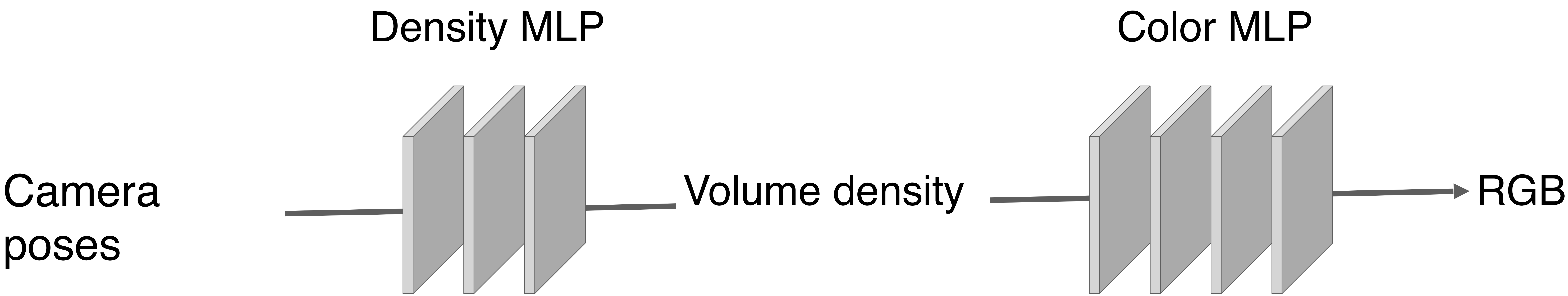
Implemented on NeRF

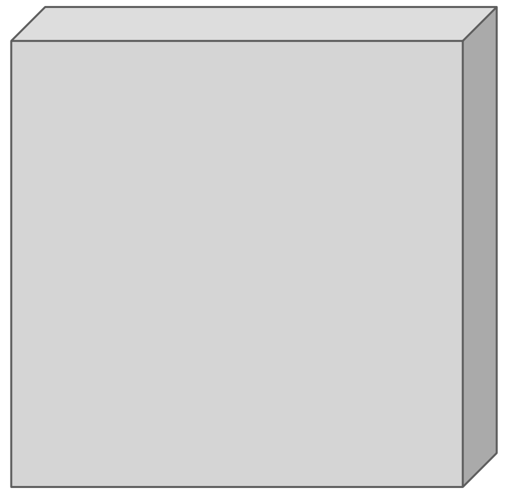


(1) Hashing of voxel vertices **(2)** Lookup **(3)** Linear interpolation **(4)** Concatenation **(5)** Neural network

NeRF

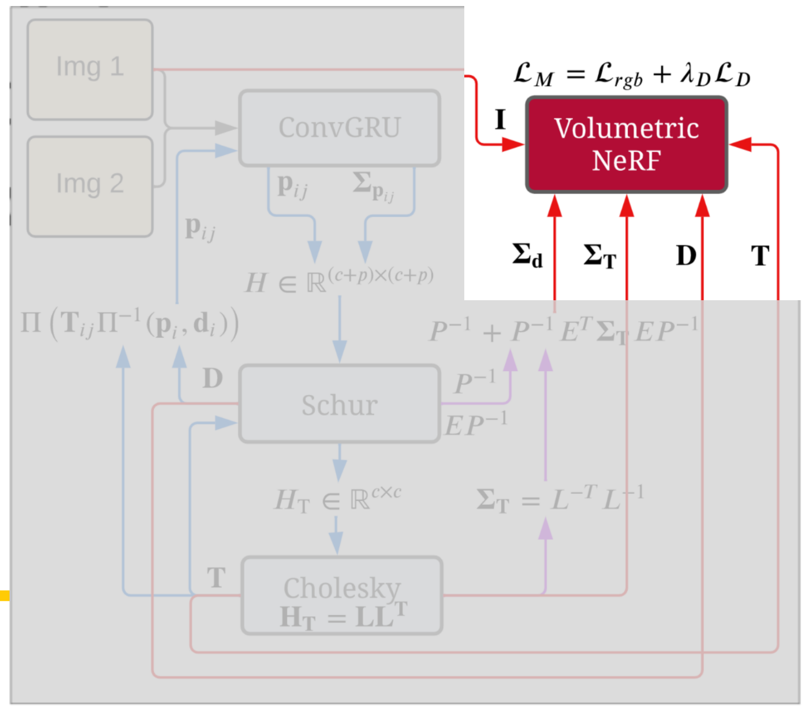






= FC layer, hidden dim 64

NeRF Loss



- Uncertainty aware mapping loss

$$\mathcal{L}_M(\mathbf{T}, \Theta) = \mathcal{L}_{\text{rgb}}(\mathbf{T}, \Theta) + \lambda_D \mathcal{L}_{\text{D}}(\mathbf{T}, \Theta)$$

Photometric loss
From original NeRF

$$\mathcal{L}_{\text{rgb}}(\mathbf{T}, \Theta) = \|I - I^\star(\mathbf{T}, \Theta)\|^2$$

Geometric loss
Mahalanobis distance

$$\mathcal{L}_{\text{D}}(\mathbf{T}, \Theta) = \|D - D^\star(\mathbf{T}, \Theta)\|^2_{\boxed{\Sigma_D}}$$

Evaluation

- Metrics
  - Geometric: Depth L1 loss (compared w/ GT depth) ↓
  - Photometric: Peak Signal / Noise Ratio (PSNR) ↑
- Methods
  - TDSF-fusion (classical)
  - σ-fusion (probabilistic)
  - Nice-SLAM, iMAP (learning-based)

Results: Replica Dataset

		room-0	room-1	room-2	office-0	office-1	office-2	office-3	office-4	Avg.
iMAP* [26] (GT depth)	**Depth L1 [cm]** ↓	5.70	4.93	6.94	6.43	7.41	14.23	8.68	6.80	7.64
	PSNR [dB] ↑	5.66	5.31	5.64	7.39	11.89	8.12	5.62	5.98	6.95
Nice-SLAM [42] (GT depth)	**Depth L1 [cm]** ↓	**2.53**	**3.45**	**2.93**	**1.51**	**0.93**	**8.41**	**10.48**	**2.43**	**4.08**
	PSNR [dB] ↑	**29.90**	**29.12**	**19.80**	**22.44**	**25.22**	**22.79**	**22.94**	**24.72**	**24.61**
TSDF-Fusion Res. = 256 (our depth)	**Depth L1 [cm]** ↓	23.51	20.94	23.34	14.11	10.50	30.89	28.92	22.83	21.88
	PSNR [dB] ↑	3.43	4.51	5.57	11.16	15.92	4.86	5.68	5.46	7.07
σ-Fusion [23] Res. = 256 (our depth)	**Depth L1 [cm]** ↓	21.92	19.28	22.40	13.80	10.21	22.27	28.70	22.21	20.10
	PSNR [dB] ↑	3.45	4.51	5.57	11.16	15.92	4.86	5.69	5.46	7.08
Nice-SLAM [42] (no depth)	**Depth L1 [cm]** ↓	11.12	9.42	19.03	11.12	10.24	16.36	21.33	14.81	14.18
	PSNR [dB] ↑	18.15	18.22	17.82	20.23	19.14	15.22	16.12	17.24	17.76
Ours (our depth)	**Depth L1 [cm]** ↓	**2.97**	**2.63**	**2.58**	**2.49**	**1.98**	**9.13**	**10.58**	**3.59**	**4.49**
	PSNR [dB] ↑	**34.90**	**36.95**	**40.75**	**48.07**	**53.44**	**39.30**	**38.63**	**39.21**	**41.40**

- Nice-SLAM, iMAP using GT depth have reasonable performance
- NeRF-SLAM outperforms all other methods (no GT depth)
  - Also outperforms GT depth methods

Pointcloud

Results: Replica Dataset

		room-0	room-1	room-2	office-0	office-1	office-2	office-3	office-4	Avg.
iMAP* [26] (GT depth)	**Depth L1** [cm] ↓	5.70	4.93	6.94	6.43	7.41	14.23	8.68	6.80	7.64
	PSNR [dB] ↑	5.66	5.31	5.64	7.39	11.89	8.12	5.62	5.98	6.95
Nice-SLAM [42] (GT depth)	**Depth L1** [cm] ↓	**2.53**	**3.45**	**2.93**	**1.51**	**0.93**	**8.41**	**10.48**	**2.43**	**4.08**
	PSNR [dB] ↑	**29.90**	**29.12**	**19.80**	**22.44**	**25.22**	**22.79**	**22.94**	**24.72**	**24.61**
TSDF-Fusion Res. = 256 (our depth)	**Depth L1** [cm] ↓	23.51	20.94	23.34	14.11	10.50	30.89	28.92	22.83	21.88
	PSNR [dB] ↑	3.43	4.51	5.57	11.16	15.92	4.86	5.68	5.46	7.07
σ-Fusion [23] Res. = 256 (our depth)	**Depth L1** [cm] ↓	21.92	19.28	22.40	13.80	10.21	22.27	28.70	22.21	20.10
	PSNR [dB] ↑	3.45	4.51	5.57	11.16	15.92	4.86	5.69	5.46	7.08
Nice-SLAM [42] (no depth)	**Depth L1** [cm] ↓	11.12	9.42	19.03	11.12	10.24	16.36	21.33	14.81	14.18
	PSNR [dB] ↑	18.15	18.22	17.82	20.23	19.14	15.22	16.12	17.24	17.76
Ours (our depth)	**Depth L1** [cm] ↓	**2.97**	**2.63**	**2.58**	**2.49**	**1.98**	**9.13**	**10.58**	**3.59**	**4.49**
	PSNR [dB] ↑	**34.90**	**36.95**	**40.75**	**48.07**	**53.44**	**39.30**	**38.63**	**39.21**	**41.40**

- Nice-SLAM, iMAP using GT depth have reasonable performance
- NeRF-SLAM outperforms all other methods (no GT depth)
  - Also outperforms GT depth methods

TSDF

Results: Replica Dataset

		room-0	room-1	room-2	office-0	office-1	office-2	office-3	office-4	Avg.
iMAP* [26] (GT depth)	Depth L1 [cm] ↓	5.70	4.93	6.94	6.43	7.41	14.23	8.68	6.80	7.64
	PSNR [dB] ↑	5.66	5.31	5.64	7.39	11.89	8.12	5.62	5.98	6.95
Nice-SLAM [42] (GT depth)	Depth L1 [cm] ↓	**2.53**	**3.45**	**2.93**	**1.51**	**0.93**	**8.41**	**10.48**	**2.43**	**4.08**
	PSNR [dB] ↑	**29.90**	**29.12**	**19.80**	**22.44**	**25.22**	**22.79**	**22.94**	**24.72**	**24.61**
TSDF-Fusion Res. = 256 (our depth)	Depth L1 [cm] ↓	23.51	20.94	23.34	14.11	10.50	30.89	28.92	22.83	21.88
	PSNR [dB] ↑	3.43	4.51	5.57	11.16	15.92	4.86	5.68	5.46	7.07
σ-Fusion [23] Res. = 256 (our depth)	Depth L1 [cm] ↓	21.92	19.28	22.40	13.80	10.21	22.27	28.70	22.21	20.10
	PSNR [dB] ↑	3.45	4.51	5.57	11.16	15.92	4.86	5.69	5.46	7.08
Nice-SLAM [42] (no depth)	Depth L1 [cm] ↓	11.12	9.42	19.03	11.12	10.24	16.36	21.33	14.81	14.18
	PSNR [dB] ↑	18.15	18.22	17.82	20.23	19.14	15.22	16.12	17.24	17.76
Ours (our depth)	Depth L1 [cm] ↓	**2.97**	**2.63**	**2.58**	**2.49**	**1.98**	**9.13**	**10.58**	**3.59**	**4.49**
	PSNR [dB] ↑	**34.90**	**36.95**	**40.75**	**48.07**	**53.44**	**39.30**	**38.63**	**39.21**	**41.40**

Nice-SLAM

- Nice-SLAM, iMAP using GT depth have reasonable performance
- NeRF-SLAM outperforms all other methods (no GT depth)
  - Also outperforms GT depth methods

Results: Replica Dataset

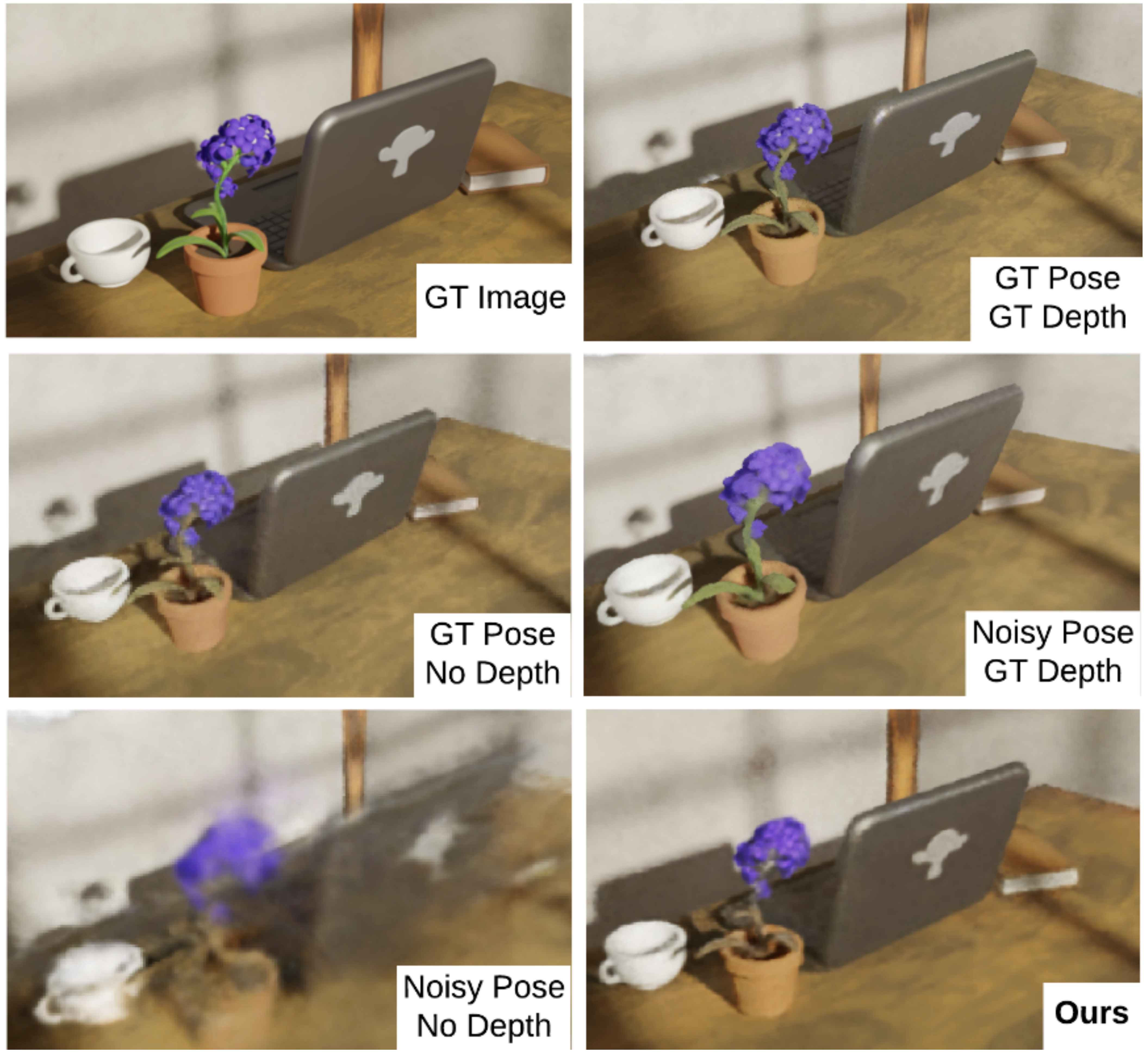
		room-0	room-1	room-2	office-0	office-1	office-2	office-3	office-4	Avg.
iMAP* [26] (GT depth)	Depth L1 [cm] ↓	5.70	4.93	6.94	6.43	7.41	14.23	8.68	6.80	7.64
	PSNR [dB] ↑	5.66	5.31	5.64	7.39	11.89	8.12	5.62	5.98	6.95
Nice-SLAM [42] (GT depth)	Depth L1 [cm] ↓	**2.53**	**3.45**	**2.93**	**1.51**	**0.93**	**8.41**	**10.48**	**2.43**	**4.08**
	PSNR [dB] ↑	**29.90**	**29.12**	**19.80**	**22.44**	**25.22**	**22.79**	**22.94**	**24.72**	**24.61**
TSDF-Fusion Res. = 256 (our depth)	Depth L1 [cm] ↓	23.51	20.94	23.34	14.11	10.50	30.89	28.92	22.83	21.88
	PSNR [dB] ↑	3.43	4.51	5.57	11.16	15.92	4.86	5.68	5.46	7.07
σ-Fusion [23] Res. = 256 (our depth)	Depth L1 [cm] ↓	21.92	19.28	22.40	13.80	10.21	22.27	28.70	22.21	20.10
	PSNR [dB] ↑	3.45	4.51	5.57	11.16	15.92	4.86	5.69	5.46	7.08
Nice-SLAM [42] (no depth)	Depth L1 [cm] ↓	11.12	9.42	19.03	11.12	10.24	16.36	21.33	14.81	14.18
	PSNR [dB] ↑	18.15	18.22	17.82	20.23	19.14	15.22	16.12	17.24	17.76
Ours (our depth)	Depth L1 [cm] ↓	**2.97**	**2.63**	**2.58**	**2.49**	**1.98**	**9.13**	**10.58**	**3.59**	**4.49**
	PSNR [dB] ↑	**34.90**	**36.95**	**40.75**	**48.07**	**53.44**	**39.30**	**38.63**	**39.21**	**41.40**

- Nice-SLAM, iMAP using GT depth have reasonable performance
- NeRF-SLAM outperforms all other methods (no GT depth)
  - Also outperforms GT depth methods

Ours

Results: Ablation Study

- Ground truth depth and poses not provided
- NeRF-SLAM is resilient to noisy poses and depths
  - Due to dense depth-map weighting



Conclusions

NeRF-SLAM = DROID-SLAM + σ-fusion + Instant-NGP

Accurate ← depth uncertainty weighting

Fast ← hash based encoding

DROID-SLAM 15fps | Instant-NGP 10fps

Limitations

- Requires 11 Gb of GPU memory
- Real-time performance is 12 FPS at 640 x 480 resolution
  - Still pretty acceptable

Directions for Future Work

- Address memory requirements
  - Correlation volumes can be computed on the fly
  - Stream “inactive” volumetric information to CPU
- Extend metric-semantic SLAM with NeRF-SLAM to have photometrically accurate representations
- Utilize NeRF-SLAM as mapping engine for high-level scene understanding

NERF

Questions? Comments? Concerns? Fears and/or Fobias?

Thank you

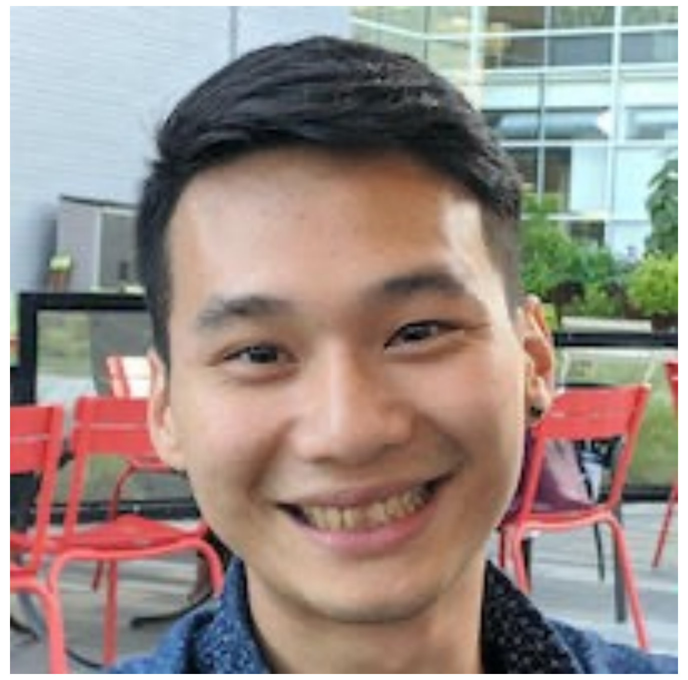
NeRF-Supervision

A Real-time Spatial Perception System for 3D Scene Graph Construction and Optimization

By: Lin Yen-Chen, Pete Florence, Jonathan T. Barron,
Tsung-Yi Lin, Alberto Rodriguez, Phillip Isola

Presented by: Aravind K, Manu Aatitya R P, Rohit B

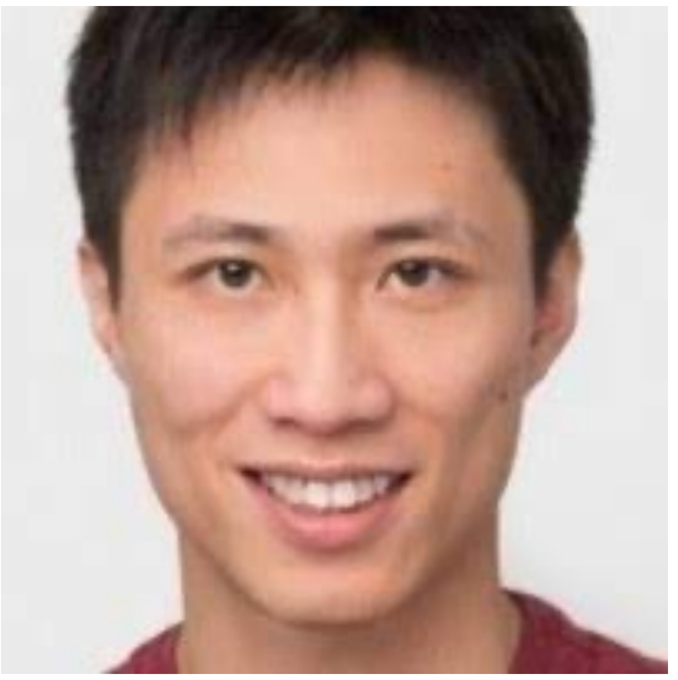
The Authors



Yen-Chen Lin,
MIT

Pete Florence,
Google

Jon Barron,
Google



Tsung-Yi Lin,
Nvidia

Alberto Rodriguez,
MIT Mech-Eng

Phillip Isola,
MIT EECS

Some attempts on 3D reconstructions

Hall Reconstruction

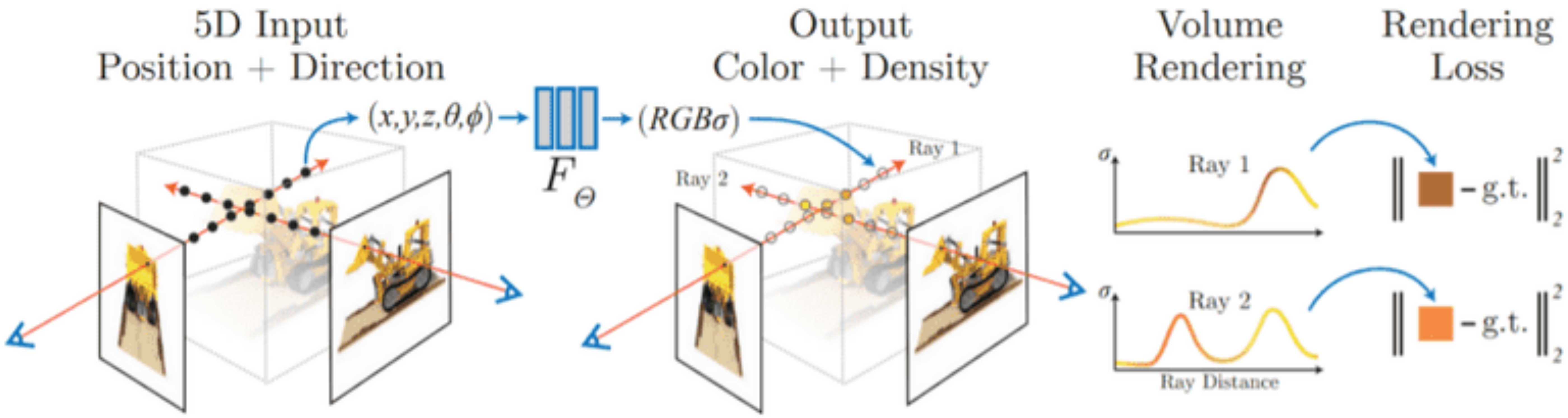
Fork Reconstruction

Some attempts on 3D reconstructions

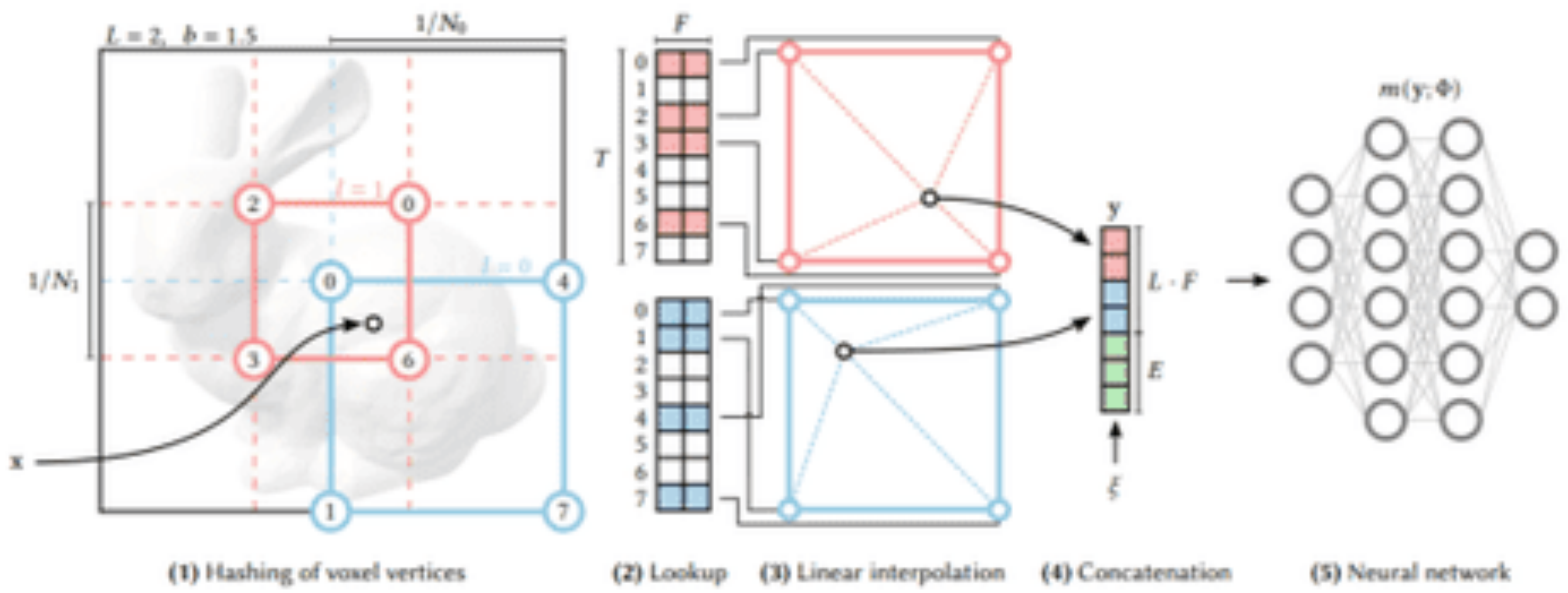
Hall Reconstruction

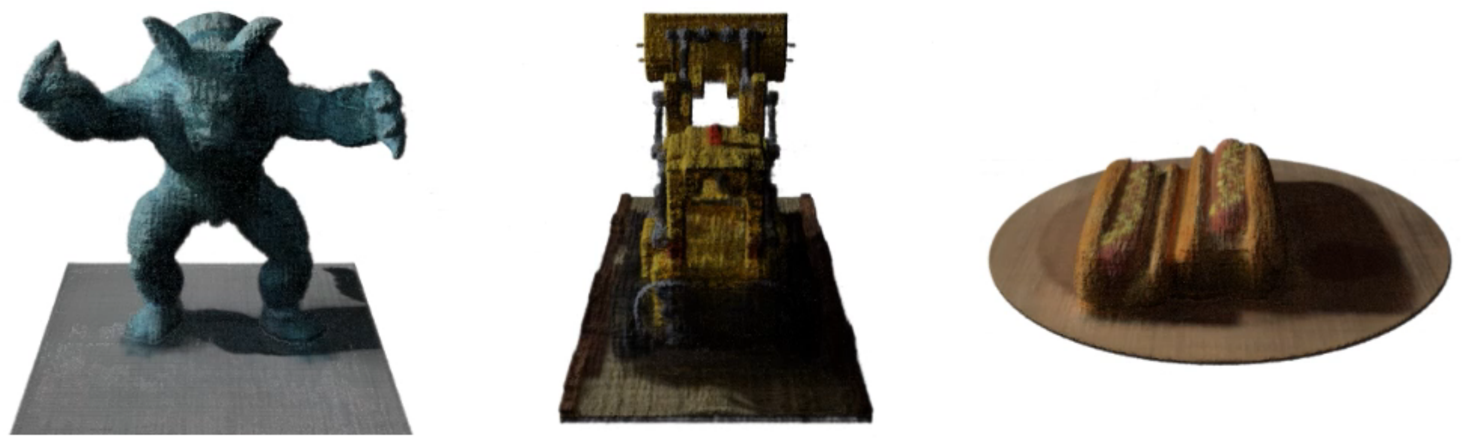
Fork Reconstruction

The Road NeRF Travelled

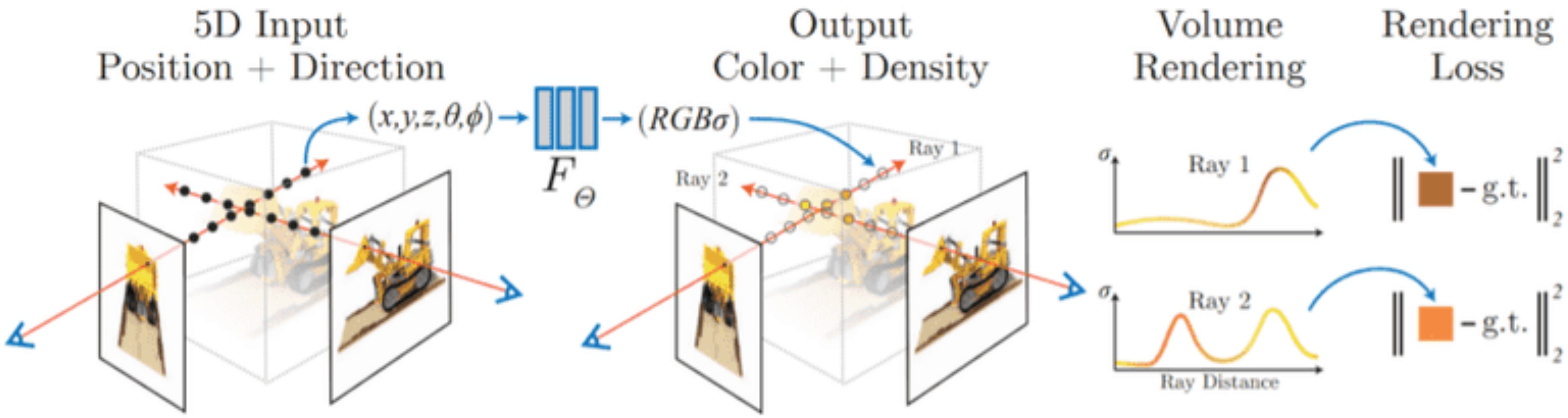


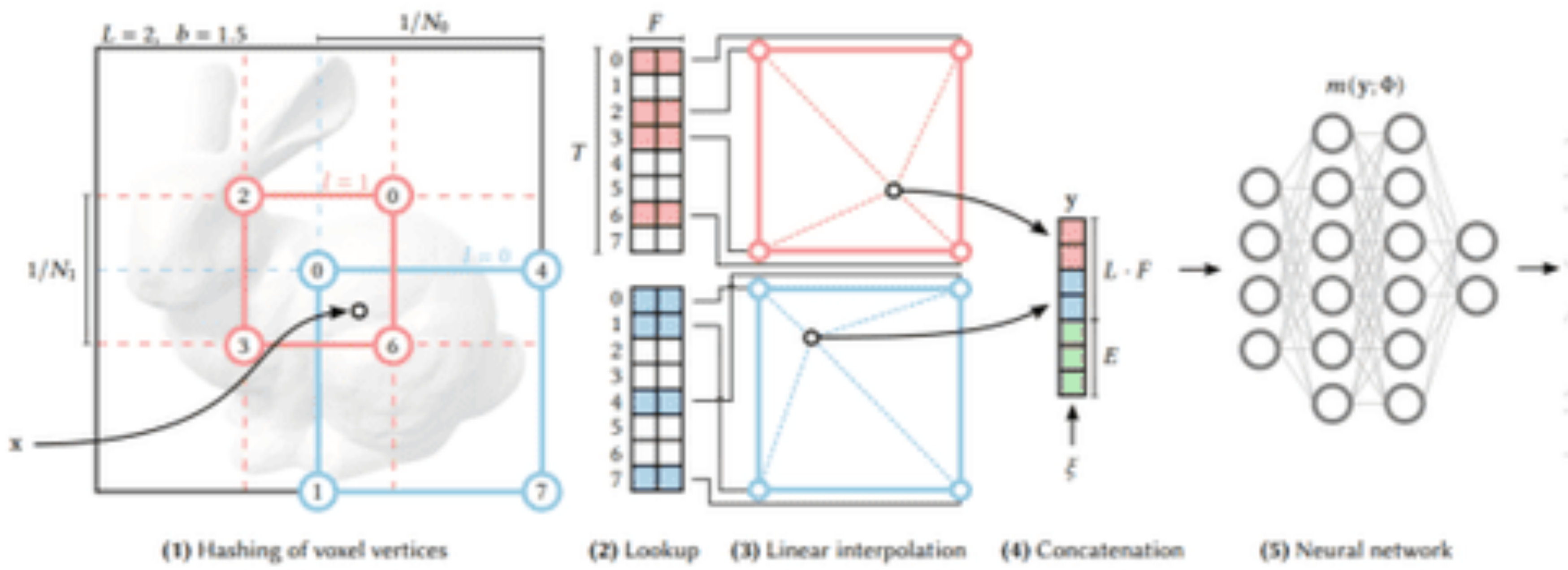
Ability to handle different lighting conditions



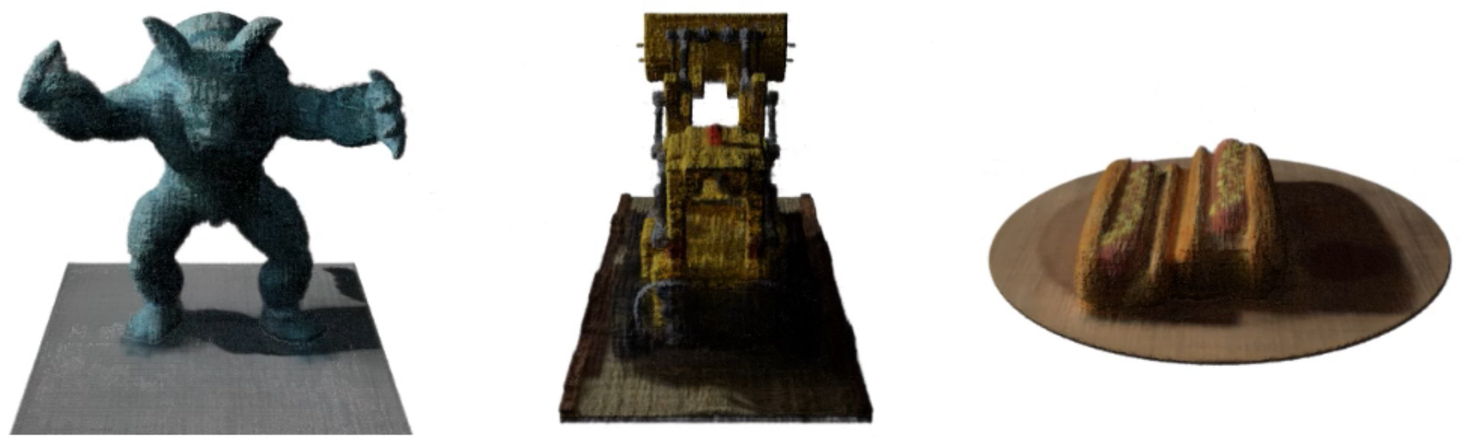


The Road NeRF Travelled

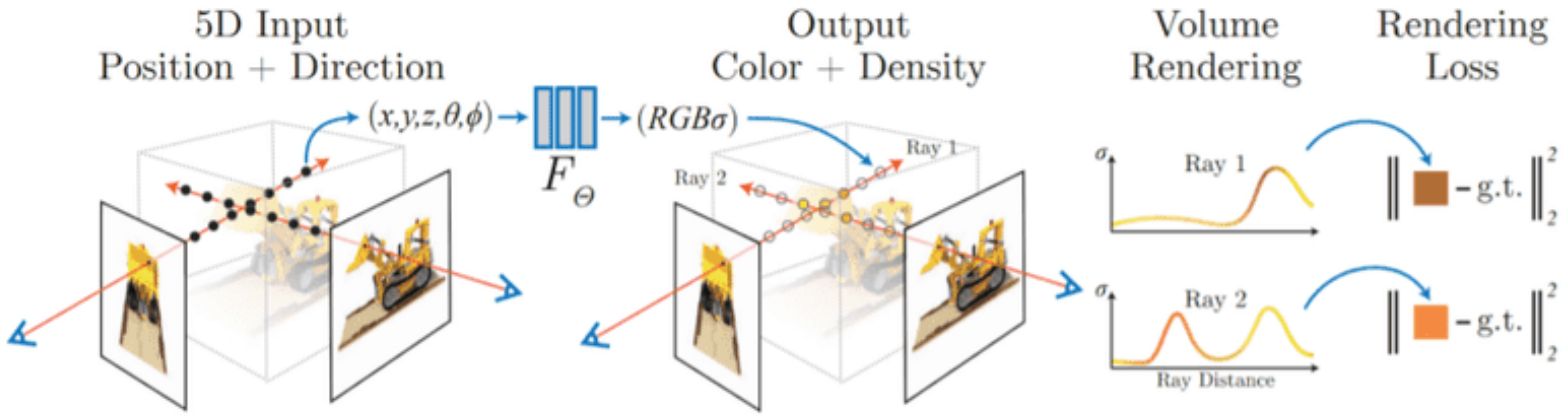


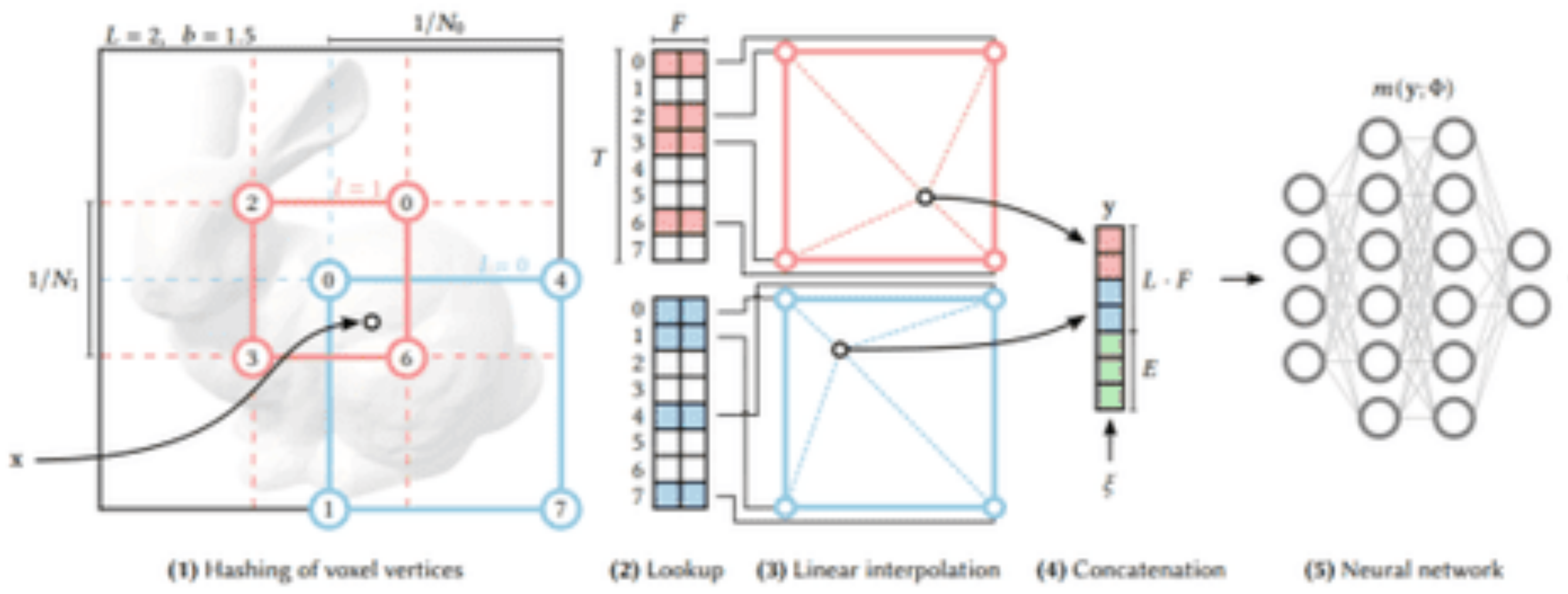


Ability to handle different lighting conditions

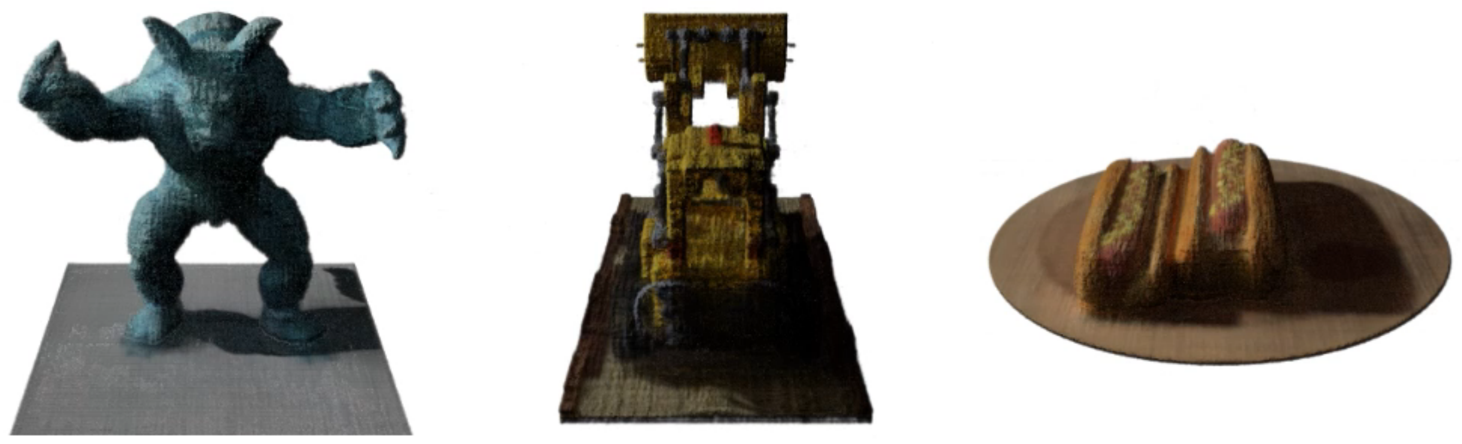


The Road NeRF Travelled



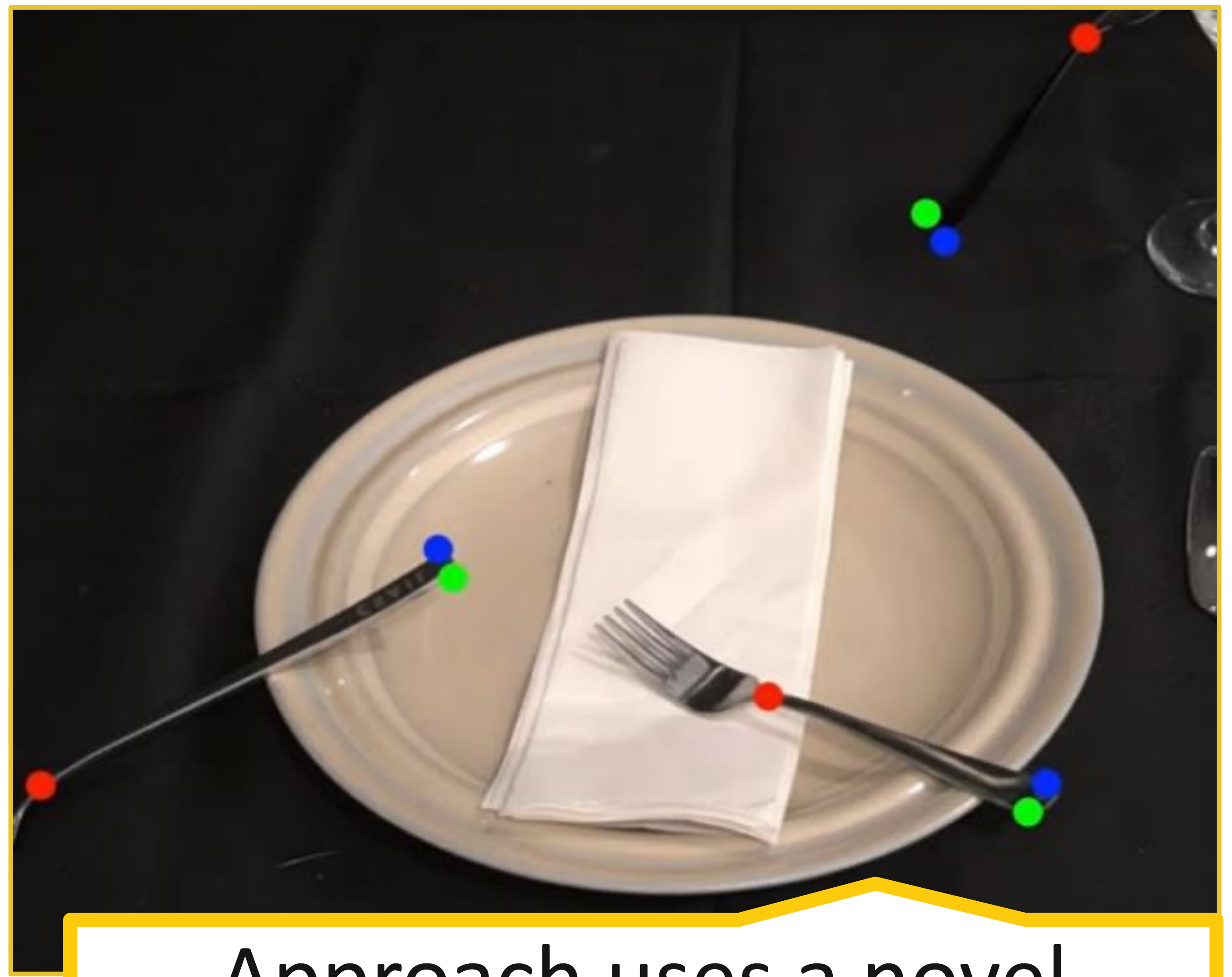


Ability to handle different lighting conditions



Overview of NeRF-Supervision

Challenging to reconstruct thin and reflective objects.

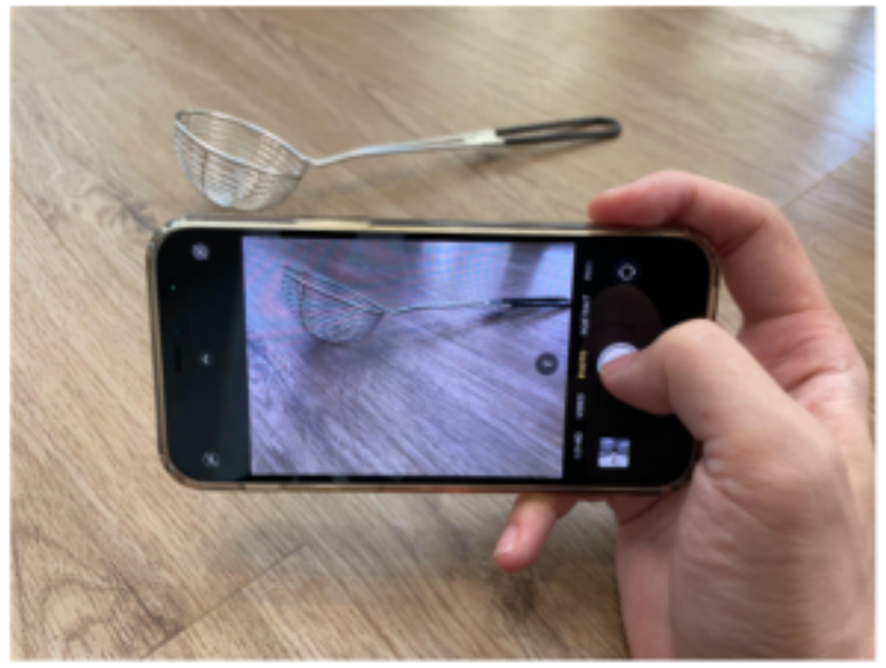


Approach uses a novel NeRF technique to solve this problem.

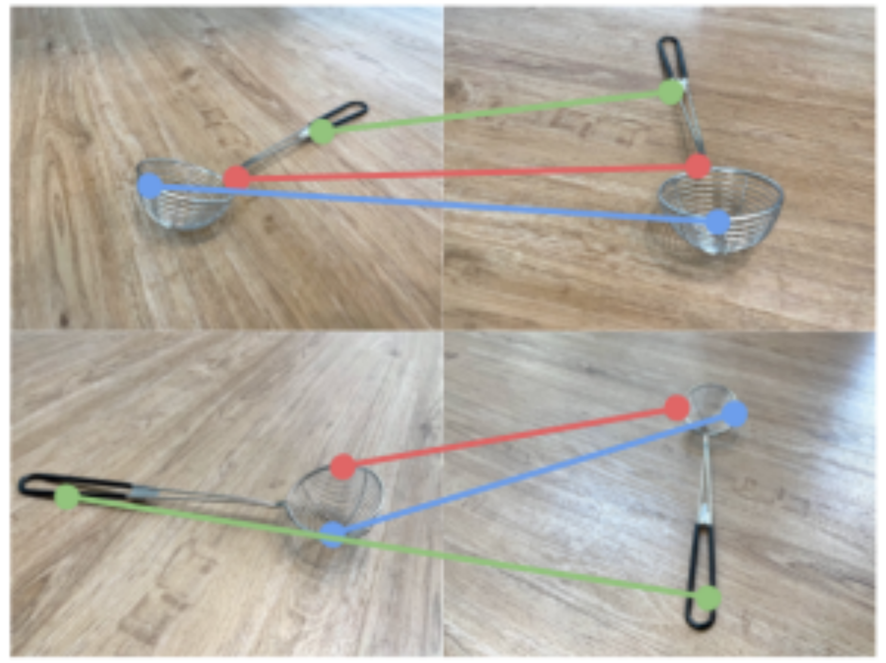
Results validates the robustness of the NeRF-Supervision pipeline.

Key Problem and Approach

This paper uses a **NeRF-Supervision pipeline** with **RGB cameras** to reconstruct **thin and highly specular objects** such as forks, knives and whisks for **robot perception**.



Collect RGB Images



Generate Dataset of Dense Correspondences

Train Dense Object Descriptors + Evaluate on new images

Pipeline of NeRF-Supervision to Reconstruct Objects

Color and Depth Estimation

Fundamental unit of **training data** is in the form of a **tuple** of the form (using only RGB data) :

RGB + H

W

$(\mathbf{I}_s, \mathbf{u}_s, \mathbf{I}_t, \mathbf{u}_t)$

Pixel-space Coordinates

Photo Loss Estimate

- NeRF uses a **MLP** to **predict** density and **RGB color** of a **3D position**.
- **Camera poses** and **true RGB** value are known from the camera.
- The photo loss is $\sum_{\mathbf{r}\in\mathcal{R}} \|\hat{\mathbf{C}}(\mathbf{r}) - \mathbf{C}(\mathbf{r})\|_2^2$

+

Depth Loss Estimate

- Modify **NeRF's photo loss** equation to **estimate depth** of a ray.
- **Ground truth depth** is obtained from **COLMAP's** partial **depth map**
- The depth loss is $\sum_{\mathbf{r}\in\mathcal{R}} \|\hat{\mathbf{D}}(\mathbf{r}) - \mathbf{D}(\mathbf{r})\|_2^2$

=

Total Loss

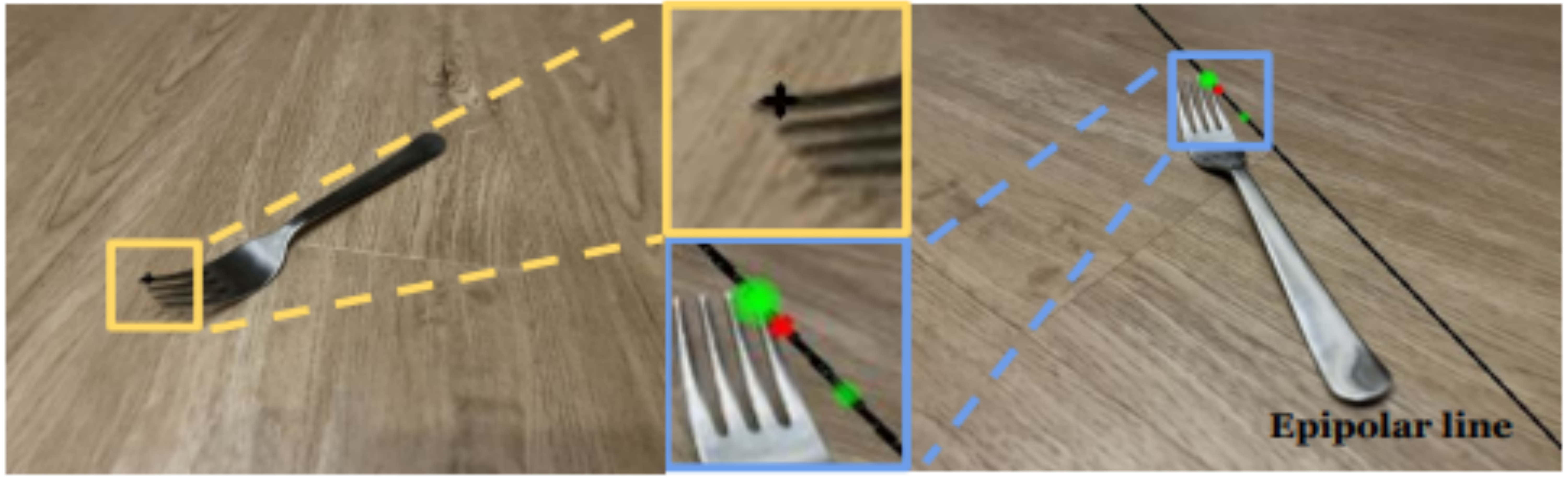
Depth-Map and Density Field

What is a depth map?

Using single-valued depth estimate at pixel $\boldsymbol{u}_s$, camera pose and intrinsic is used to generate target pixel $\boldsymbol{u}_t$ $(I_s, u_s) \rightarrow (I_t, u_t)$

What is a density field?

The correspondence generation is done via a distribution of depths rather than a single depth value

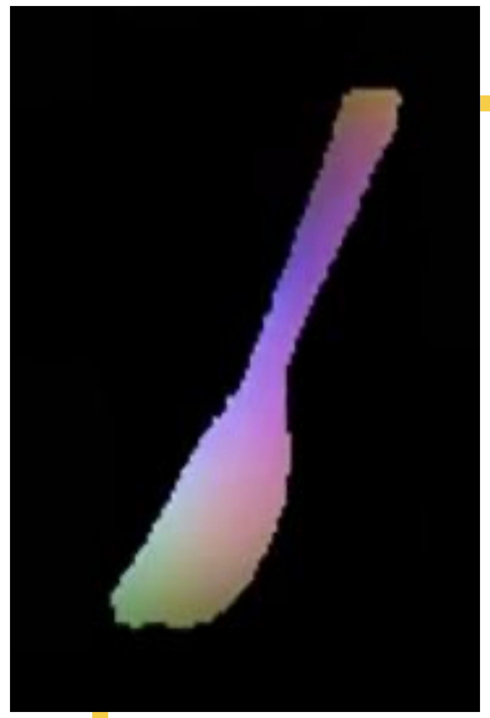


Legend

- Pixel Query u_s
- Depth Map NeRF
- Density Field NeRF

Result Objectives

Investigate whether the 3D geometry predicted by NeRF is sufficient for training precise descriptors



Compare our proposed method to existing off-the-shelf descriptors

Investigate whether the distribution-of-depth formulation is effective

Test the generalization ability of visual descriptors produced by our pipeline

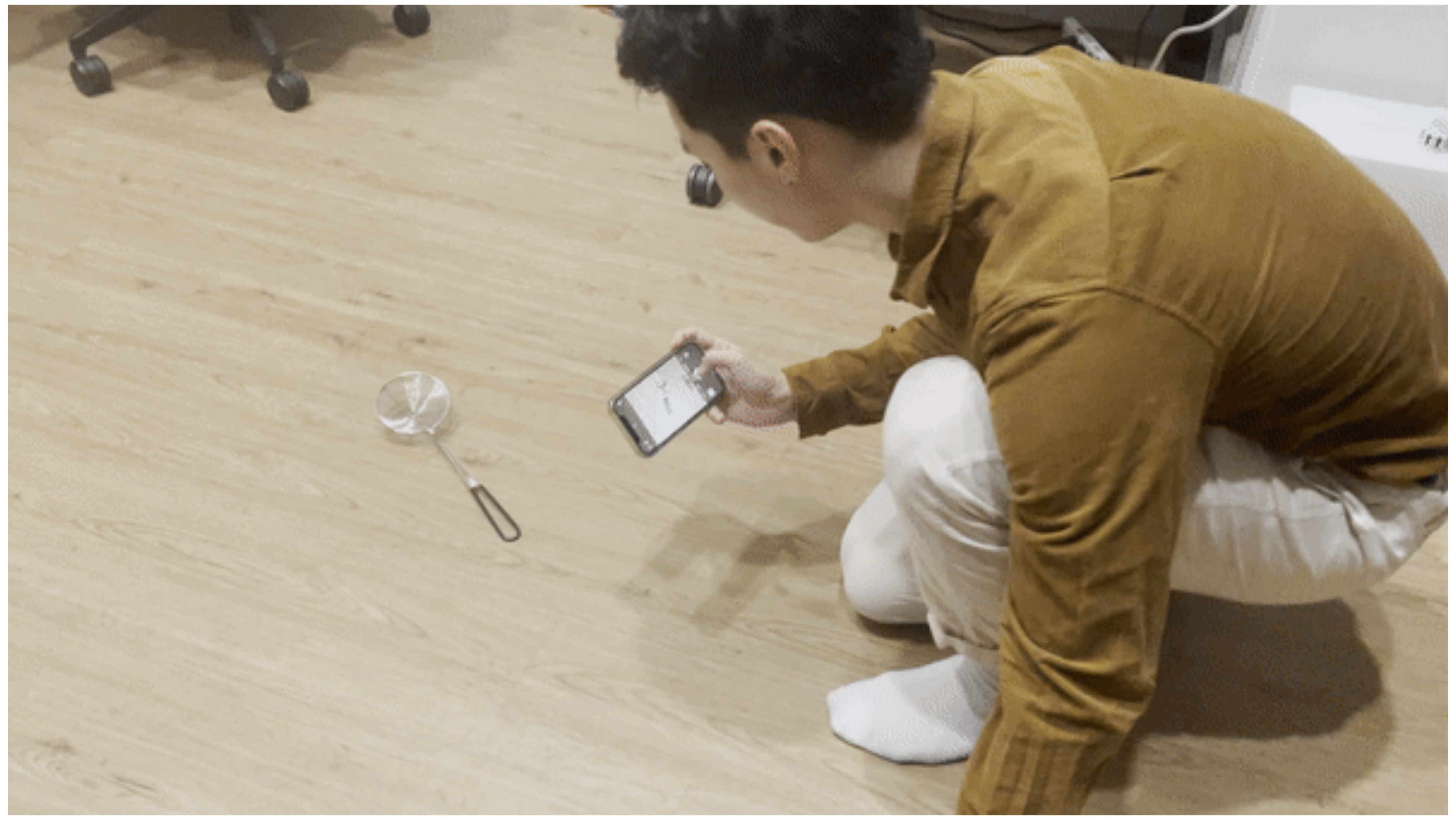
Setup & Methods

The approach and baseline methods were evaluated using **8 objects** from **3 distinct classes** (forks, whisks and strainers)

- **60 RGB input images** for each object using iPhone 12
- Camera Poses and Sparse Point Cloud estimated using **COLMAP**

Evaluation Metrics

- **Average End Point Error**
- **Percentage of Correct Keypoints**



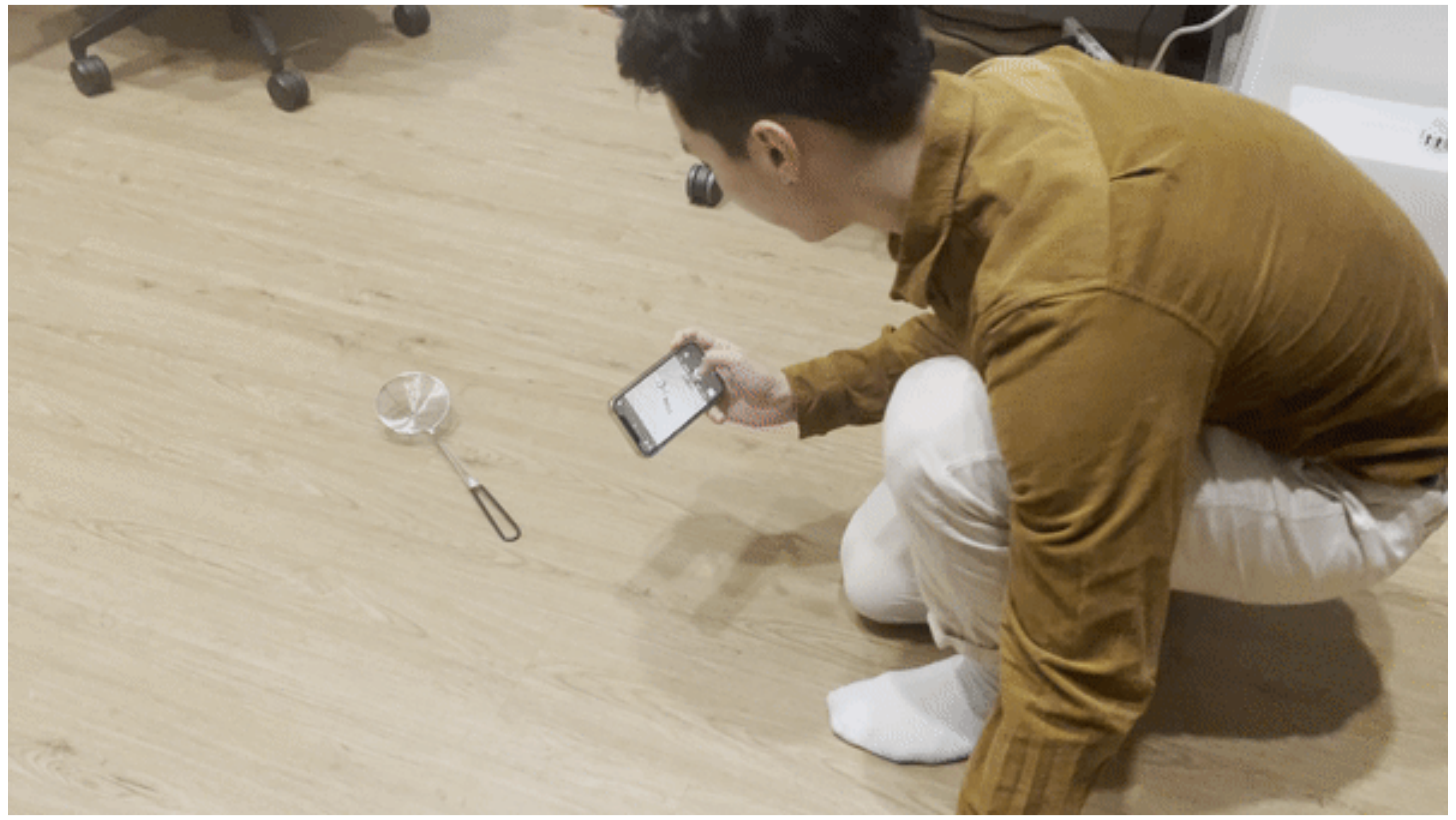
Setup & Methods

The approach and baseline methods were evaluated using **8 objects** from **3 distinct classes** (forks, whisks and strainers)

- **60 RGB input images** for each object using iPhone 12
- Camera Poses and Sparse Point Cloud estimated using **COLMAP**

Evaluation Metrics

- **Average End Point Error**
- **Percentage of Correct Keypoints**



Evaluation of Visual Descriptors

TABLE I: Average End Point Error (AEPE), ↓ lower is better.

		Strainer-S	Strainer-M	Strainer-L	Whisk-S	Whisk-M	Whisk-L	Fork-S	Fork-L	Mean
Off-the-shelf	GLU-Net [6]	33.25	28.09	28.92	16.06	15.36	39.04	17.12	18.28	24.52
	GOCor [12]	34.23	26.89	20.92	10.8	7.04	31.95	10.2	13.86	19.49
	PDC-Net [7]	32.48	13.7	23.77	7.82	5.81	19.94	8.3	8.76	15.07
DON[13] via	Depth map, COLMAP MVS	8.91	5.52	7.65	4.50	4.10	8.90	5.31	5.87	6.35
	Depth map, NeRF (ours)	5.64	4.31	5.24	3.82	3.52	6.84	3.73	4.19	4.66
	Density field, NeRF (ours)	**4.53**	**4.08**	**3.93**	**3.28**	**3.19**	**4.96**	**3.42**	**3.66**	**3.88**

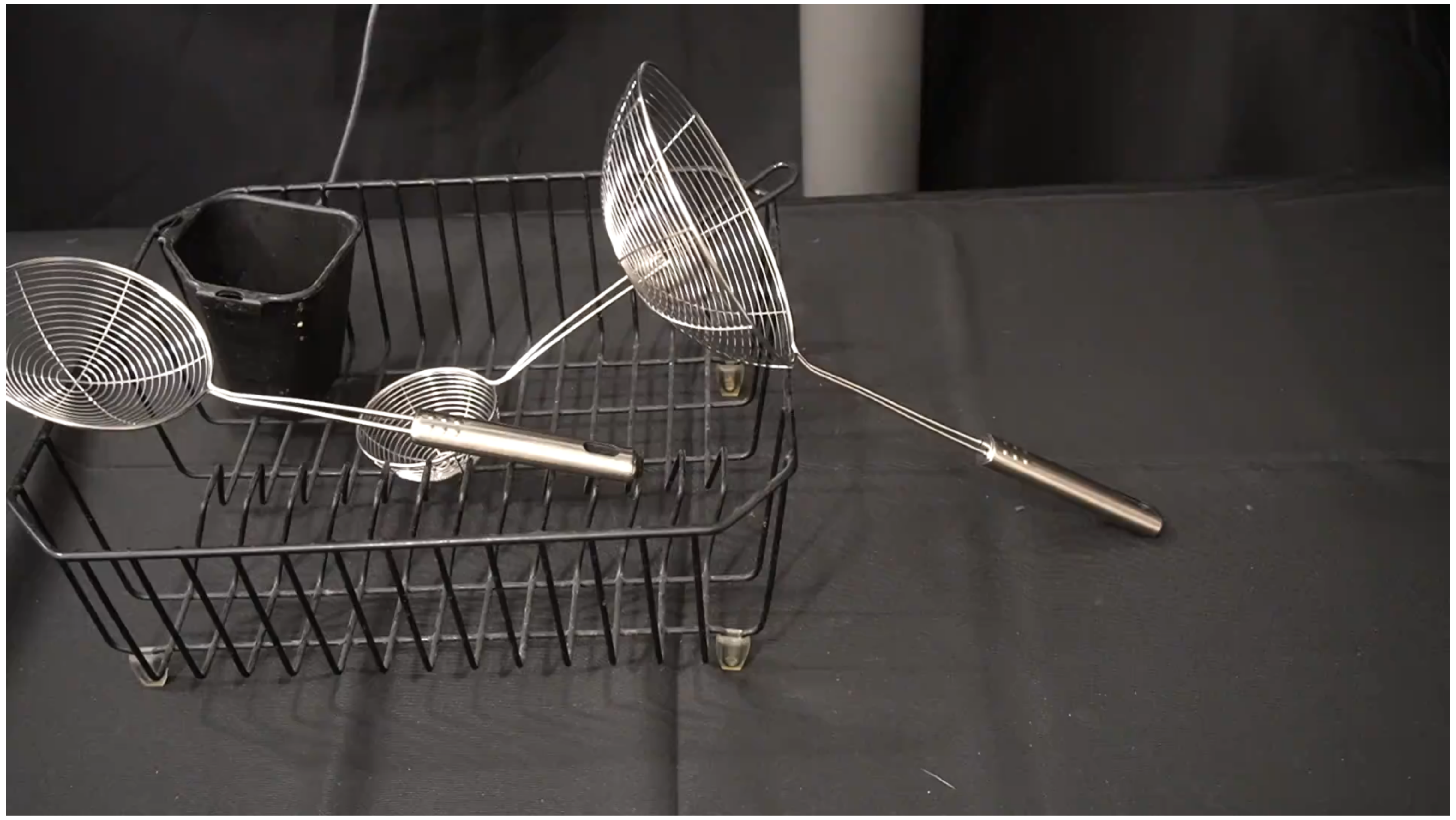
TABLE II : Percentage Correct Keypoints (PCK@5px) for 5 pixels, ↑ higher is better.

		Strainer-S	Strainer-M	Strainer-L	Whisk-S	Whisk-M	Whisk-L	Fork-S	Fork-L	Mean
Off-the-shelf	GLU-Net [6]	0.09	0.09	0.10	0.37	0.44	0.06	0.26	0.21	0.20
	GOCor [12]	0.13	0.1	0.11	0.47	0.63	0.09	0.29	0.28	0.26
	PDC-Net [7]	0.29	0.25	0.16	0.53	0.68	0.26	0.57	0.51	0.41
DON[13] via	Depth map, COLMAP MVS	0.62	0.72	0.64	0.79	0.80	0.48	0.60	0.55	0.65
	Depth map, NeRF (ours)	0.82	0.84	0.75	**0.82**	0.81	0.56	0.79	0.76	0.77
	Density field, NeRF (ours)	**0.84**	**0.87**	**0.79**	**0.82**	**0.82**	**0.64**	**0.82**	**0.78**	**0.80**

Note: Comparisons taken from **Table I and III** of the **Results** section in the paper

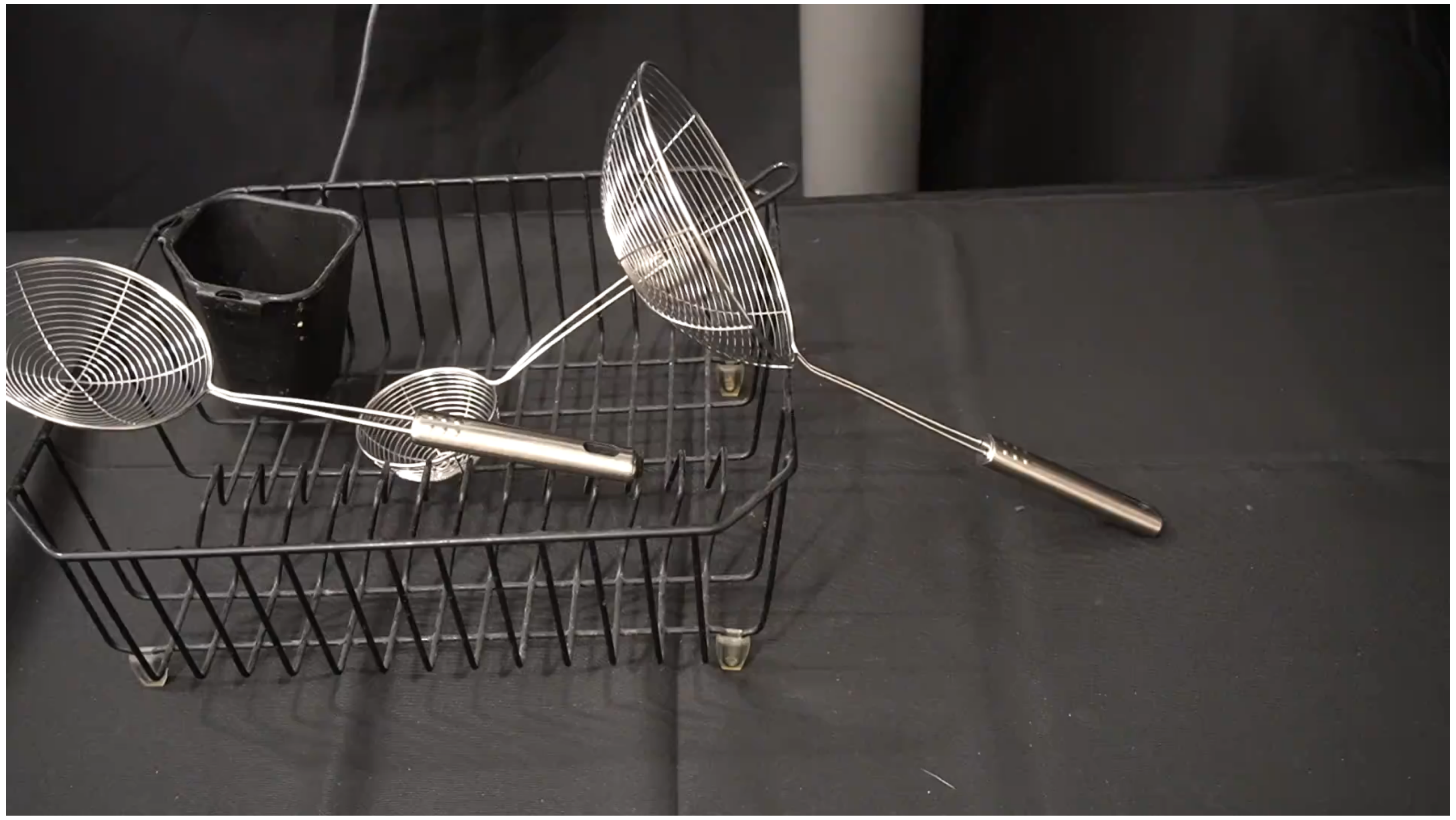
Generalization of trained DONs

The trained Dense Object Nets (DONs) were evaluated on novel scenes with **noisy background and lighting, multiple objects** and **unseen objects.**



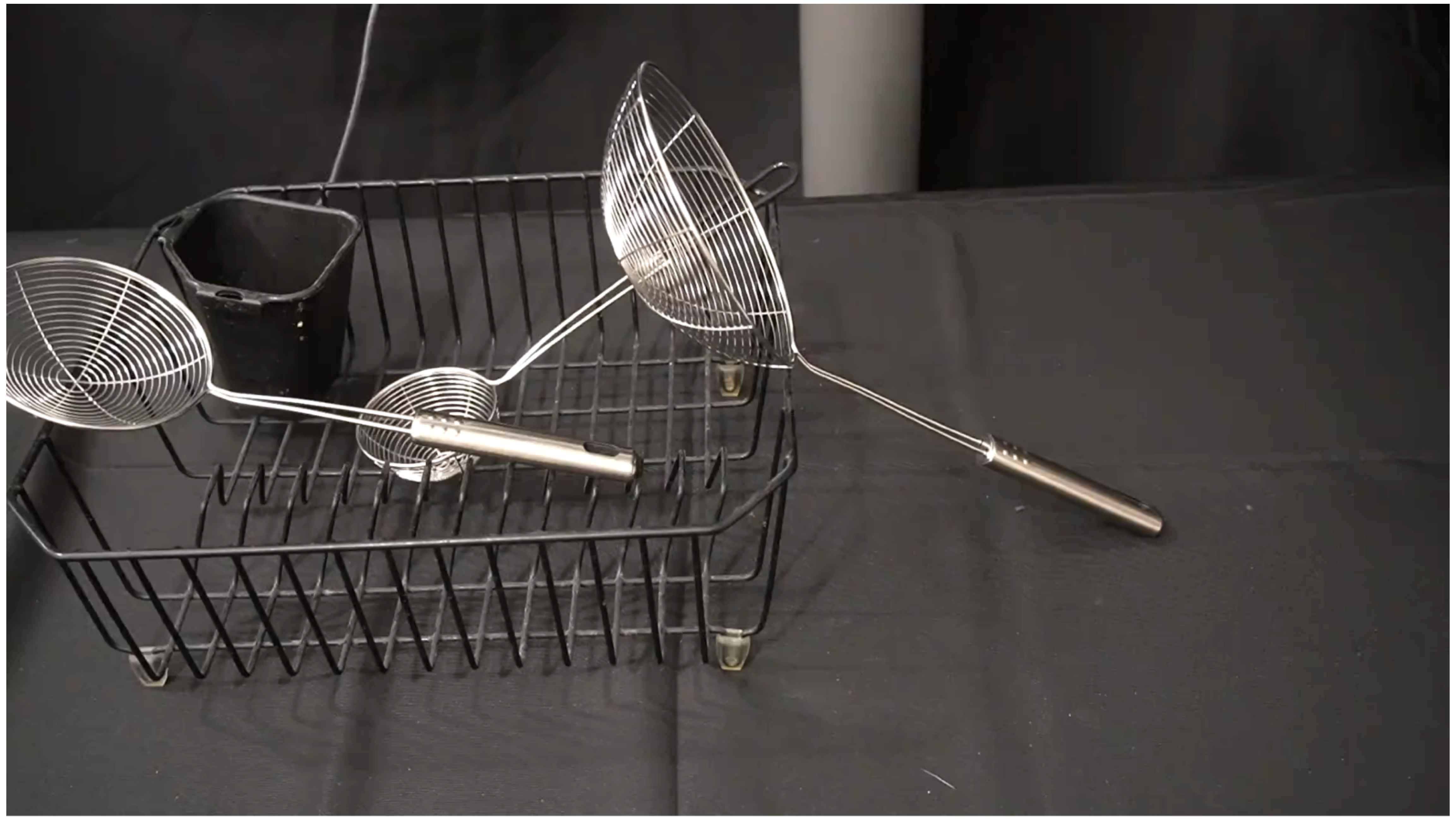
Generalization of trained DONs

The trained Dense Object Nets (DONs) were evaluated on novel scenes with **noisy background and lighting, multiple objects** and **unseen objects.**



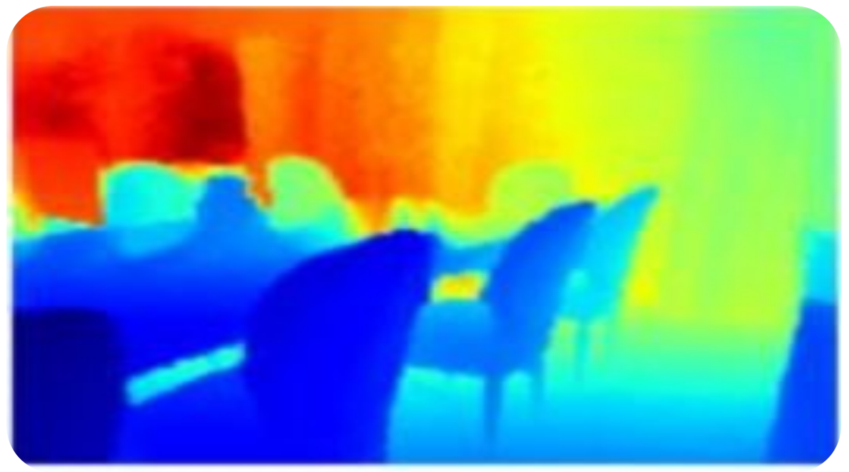
Generalization of trained DONs

The trained Dense Object Nets (DONs) were evaluated on novel scenes with **noisy background and lighting, multiple objects** and **unseen objects.**

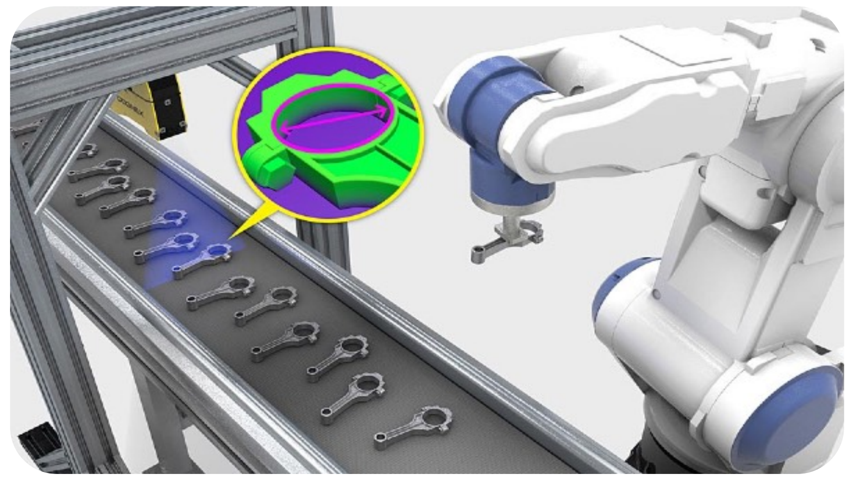


Conclusions

Introduces **NeRF-Supervision** as a state-of-the-art for learning **object-centric dense descriptors**



Proposes a pipeline using **only RGB** cameras, as opposed to **RGB-D** and **Multi-Stereo View**



Methods presented serve as a new format for **supervising robot vision systems**

Limitations and Directions for Future Work

Limitations

- The paper does not discuss the computational cost and limitations of their work.
- The pick-n-place robot manipulation done as part of this research was not well documented.

Future directions

- Test this algorithm against noisy and occluded scenes of highly specular objects.
- Extend this work to enable robots to grasp objects which are reflective and thin while they are in motion.


THANK YOU

Thank you

NARF22

Neural Articulated Radiance Fields for Configuration-Aware Rendering

By: Stanley Lewis, Jana Pavlasek, Odest Chadwicke Jenkins

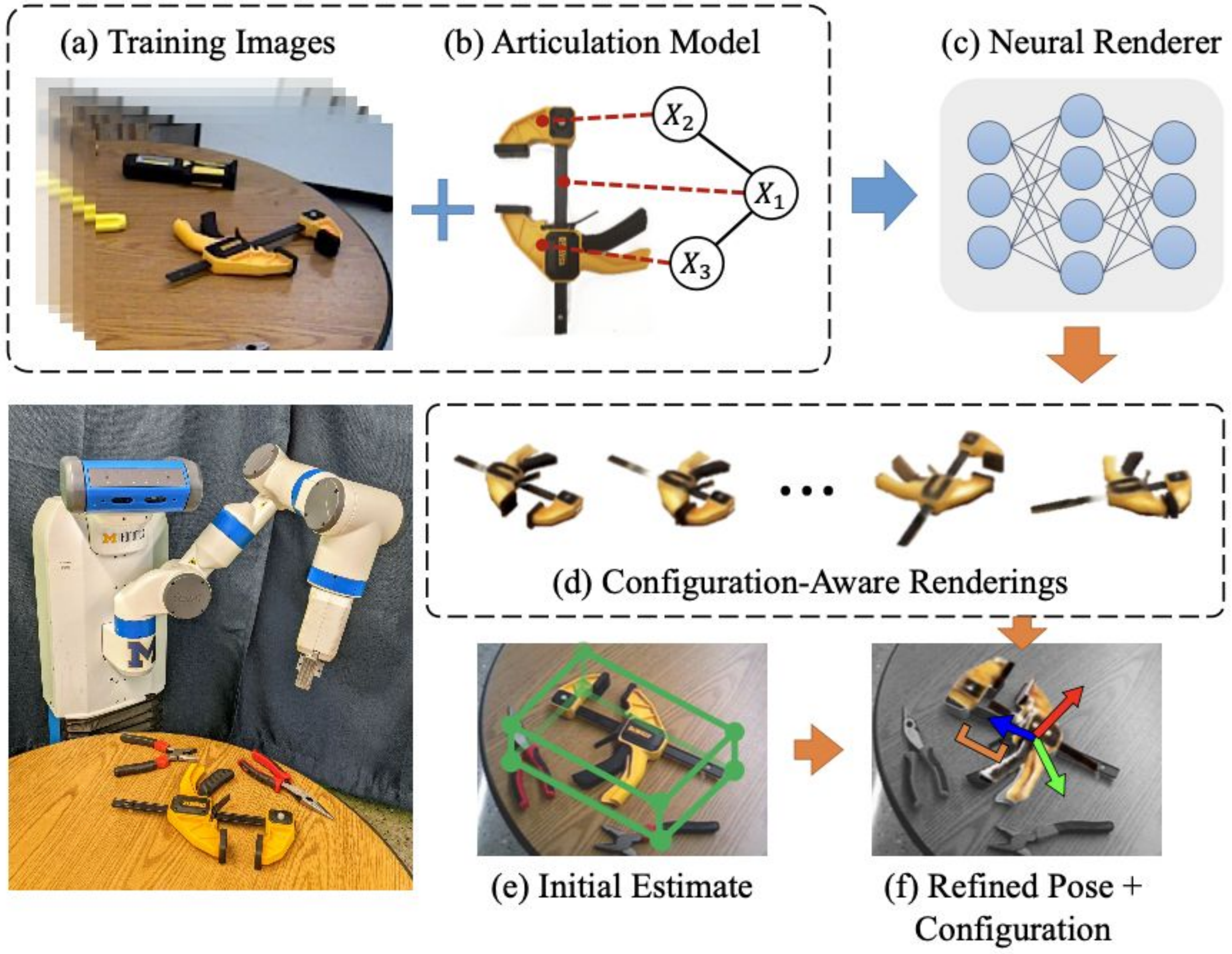
Presented by: Chetan Reddy, Wensong Hu

The Authors

- Stanley Lewis
  - PhD candidate at University of Michigan, Ann Arbor
  - Advised by: Professor Odest Chadwicke Jenkins
- Jana Pavlasek
  - PhD candidate at University of Michigan, Ann Arbor
  - Advised by: Professor Odest Chadwicke Jenkins
- Odest Chadwicke Jenkins
  - Professor at University of Michigan, Ann Arbor

Outline

- Background
- Contributions of NARF22
- Approach and Pipeline
- Results
- Conclusion



Questions

- What are the people expecting the robot do?
  - NOT just pick up things, but can USE them
- How can robot work with tools like pliers?
  - These articulated objects might change their configuration while the robot handling due to gravity or inertia
  - Robot should understand the objects and estimate its configuration in order to manipulate them

Figure 1. Articulated objects: Pliers

Background

- Difficulties: Identify the object's configuration with ***only single configuration inputted***
  - diversity of object geometries
  - high-dimensionality introduced by articulated degrees-of-freedom
- Previous work:
  - data-driven method for ***rigid body pose estimation*** is getting matured
  - data-driven ***articulated object estimation*** still a challenge, as generating large-scale datasets that cover the full range of configurations is difficult

Value Proposition

- Why ***configuration-aware rendering***?
  - Configuration-aware rendering allows fast and accurate pose estimation and motion planning
- Why ***NeRF***?
  - NeRF generate high-quality renderings quickly, making it suitable for real-time applications such as robotics.
- Why ***articulated objects***?
  - Most of object have multiple interconnected parts that can move in complex ways
  - Examples: doors, articulated tools, drawers

Contributions

1. Introduce ***NeRF*** into the articulated object detection
2. Training only with the images of ***one*** configuration inputted
3. Render objects at ***arbitrary*** configurations
4. ***Reduces real-world data required*** for downstream pose estimation

Figure 2. Different render result for clamp

High Level Approach

- Utilize training images of objects, along with their poses and articulation models to render the object at any configuration
- Able to perform pose and configuration estimation of an object given an initial guess

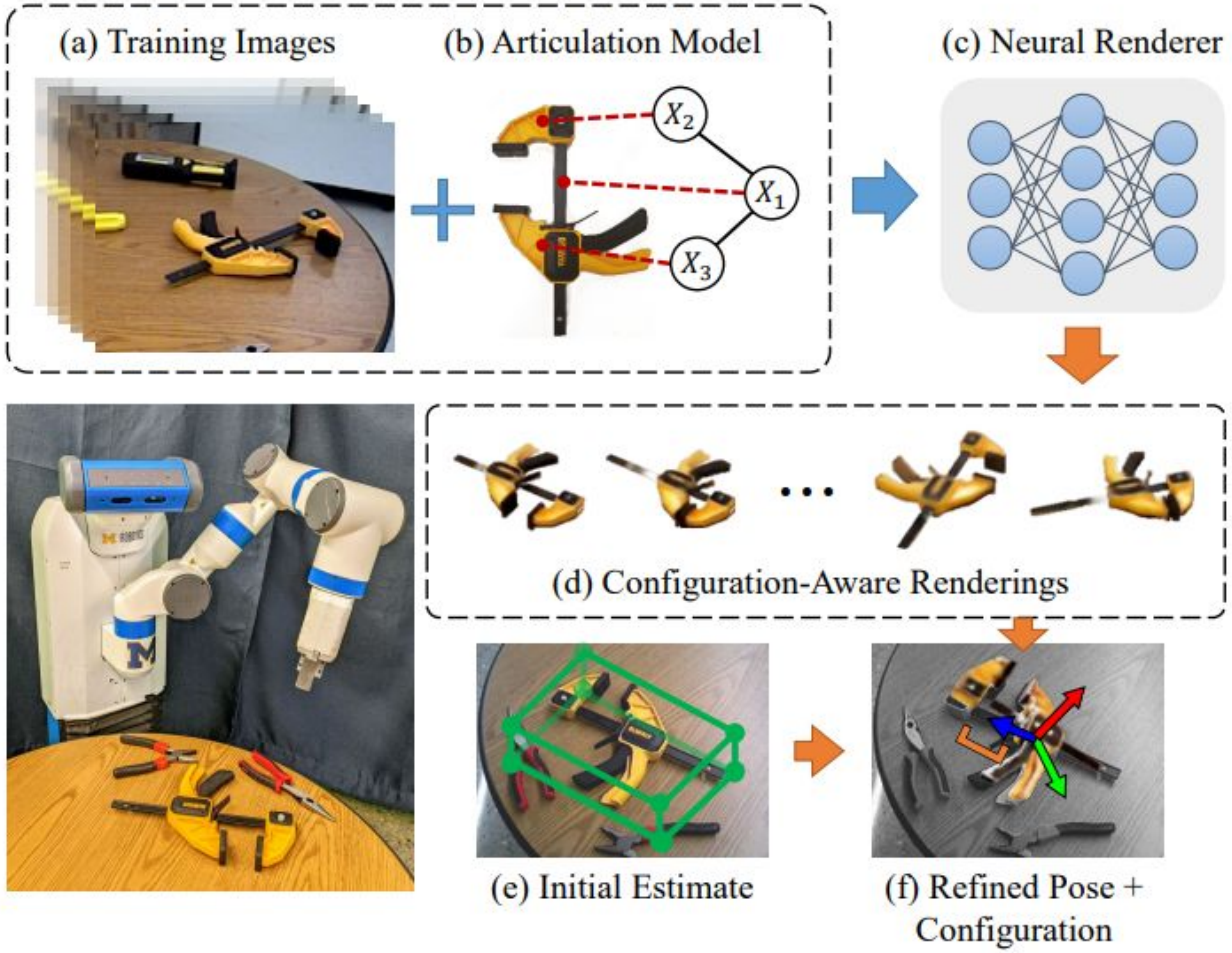


Figure 3. NARF22 Operation Procedure

Training Pipeline

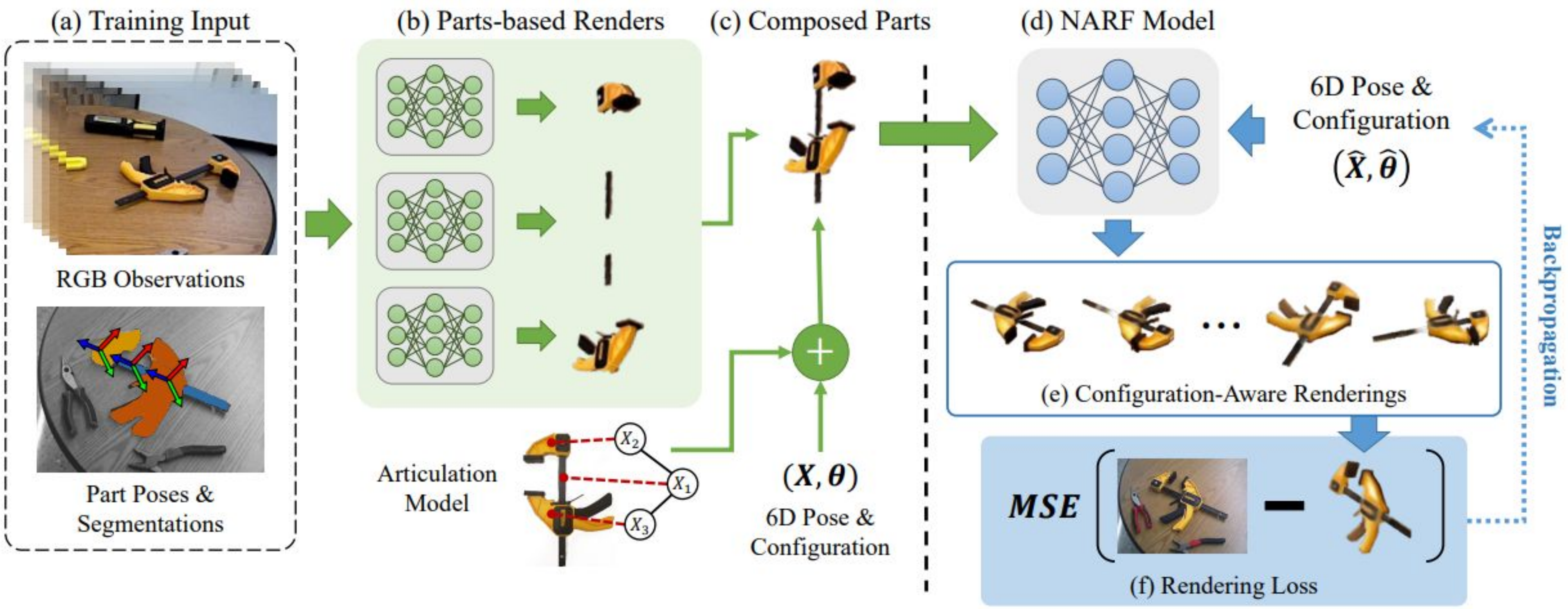


Figure 4. NARF22 Two Stage Pipeline

Training Pipeline

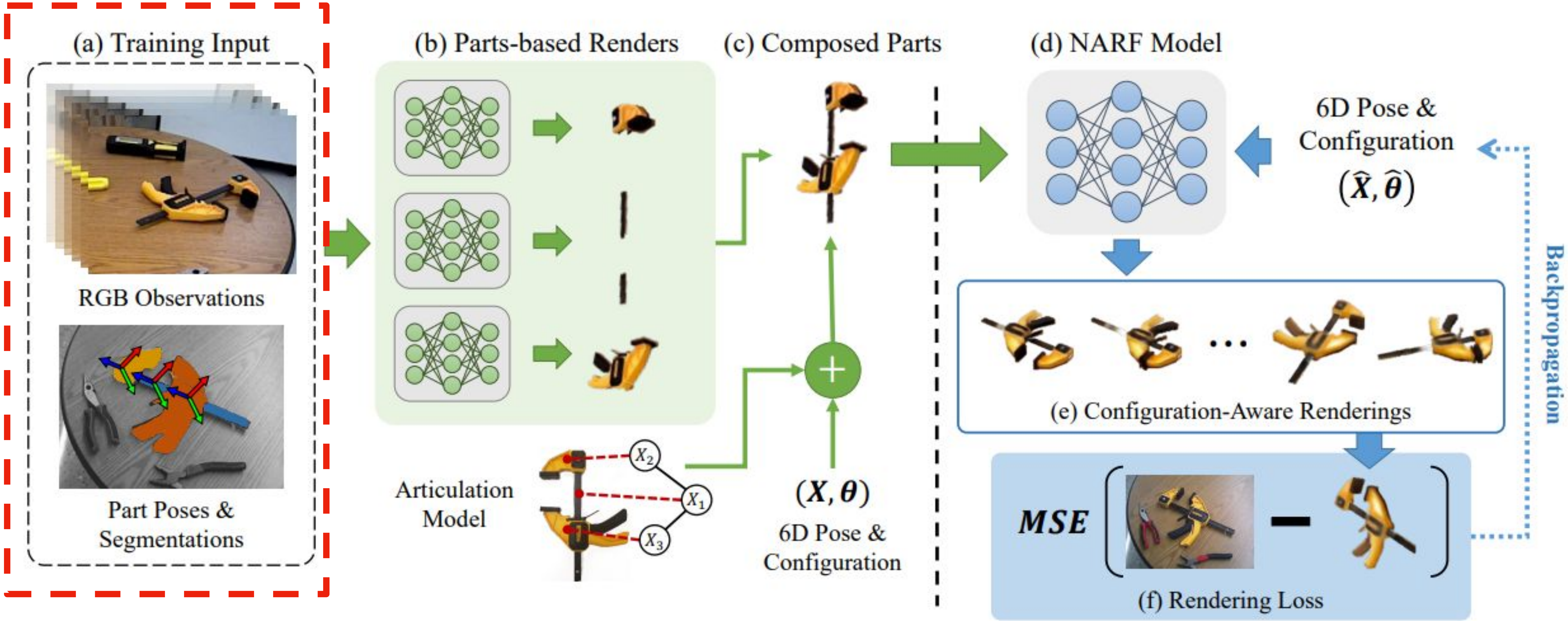


Figure 4. NARF22 Two Stage Pipeline

- Take input images with segmentation masks for each part
- Each frame is also labeled with pose and object configuration

Training Pipeline

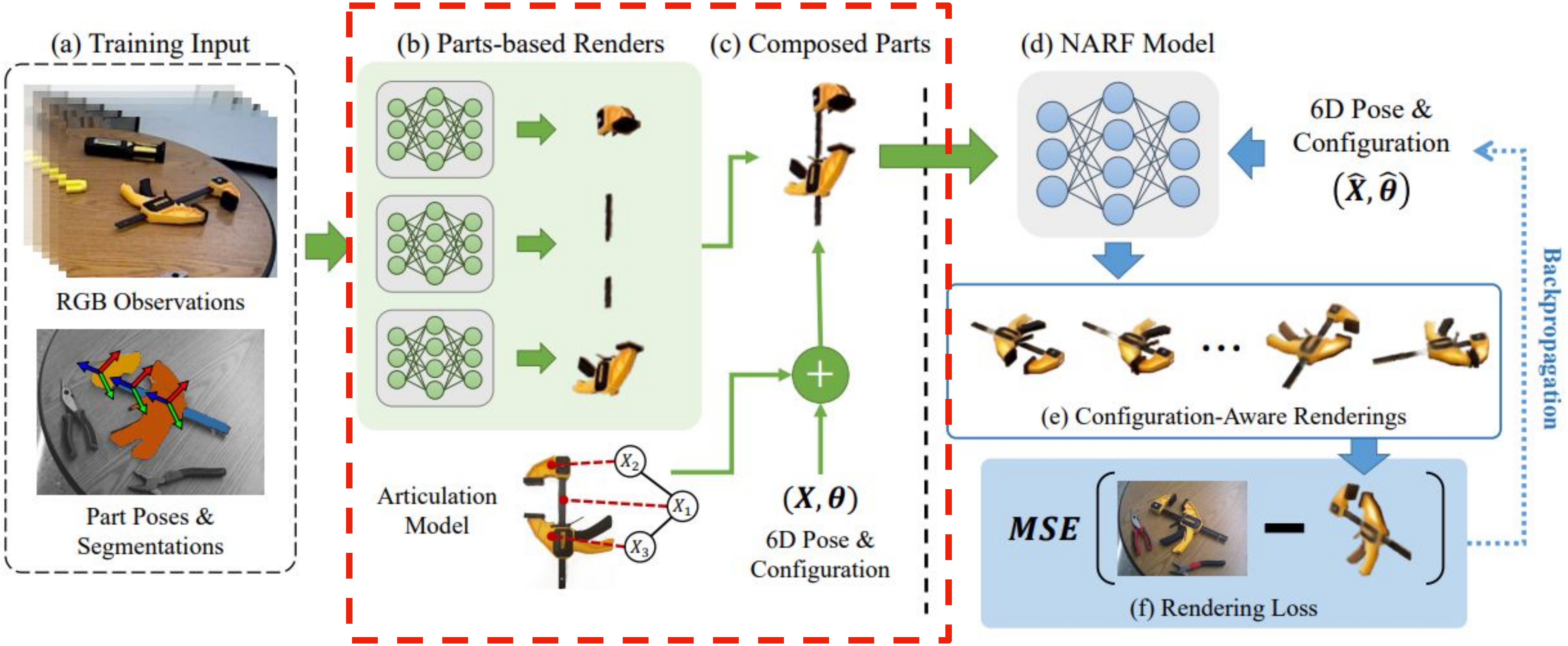


Figure 4. NARF22 Two Stage Pipeline

- Parts of the object are first individually trained without the configuration parameters
- They are then individually rendered and put together representing all of the configurations

Training Pipeline

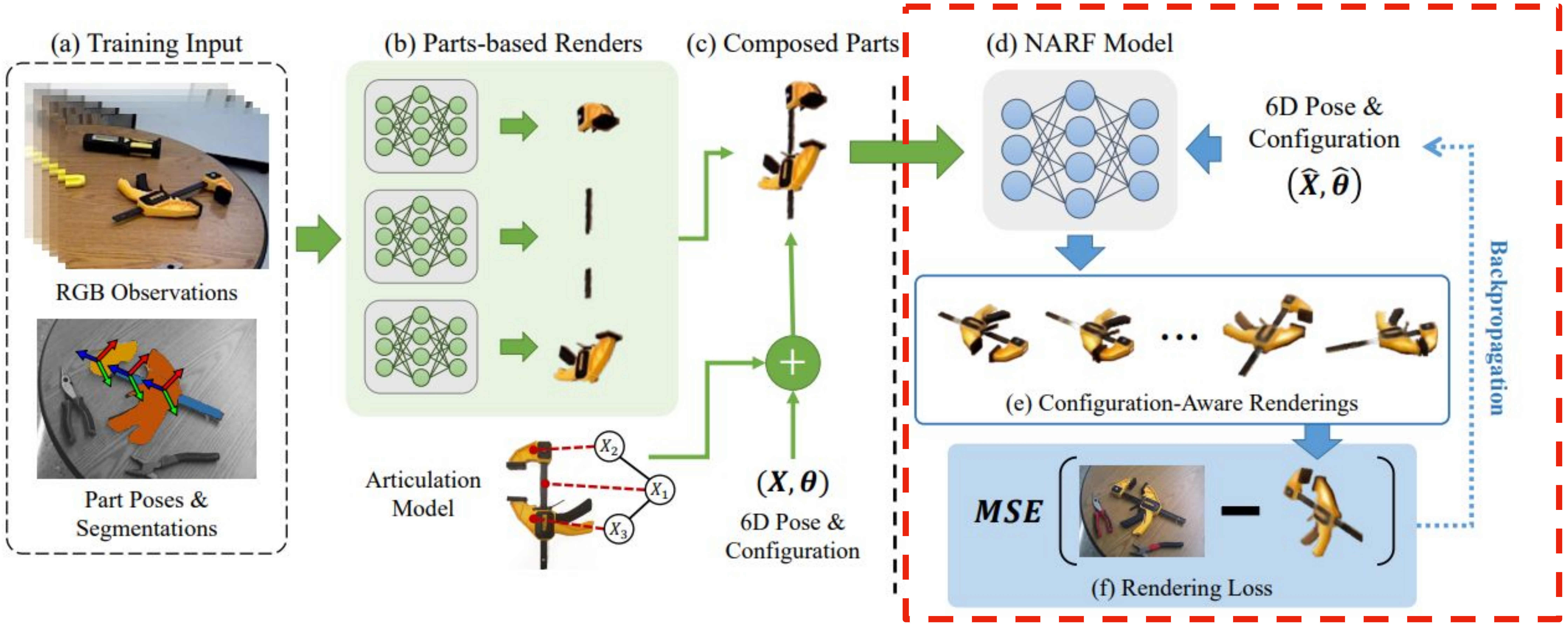


Figure 4. NARF22 Two Stage Pipeline

- The second stage consists of a neural renderer trained with the configuration-parametrized model

Renderings of Novel Configurations and Viewpoints

Figure 6. Render Result for Arbitrary Configurations

Renderings of Novel Configurations and Viewpoints

- Self obscuration of the clamp bar leads to gaps in clamp renderings

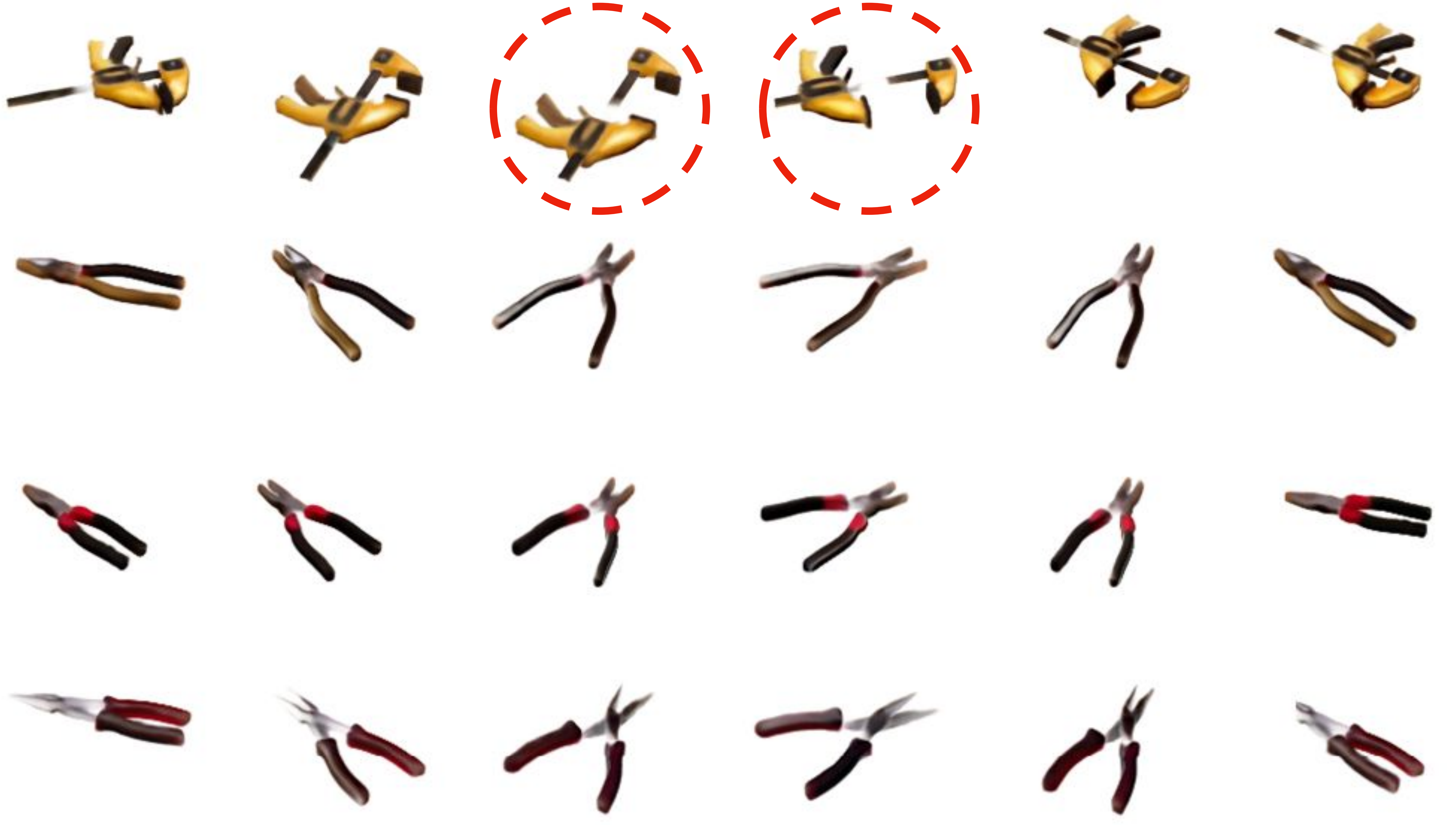


Figure 6. Render Result for Arbitrary Configurations

Renderings of Novel Configurations and Viewpoints

- Pliers were only trained on a single configuration

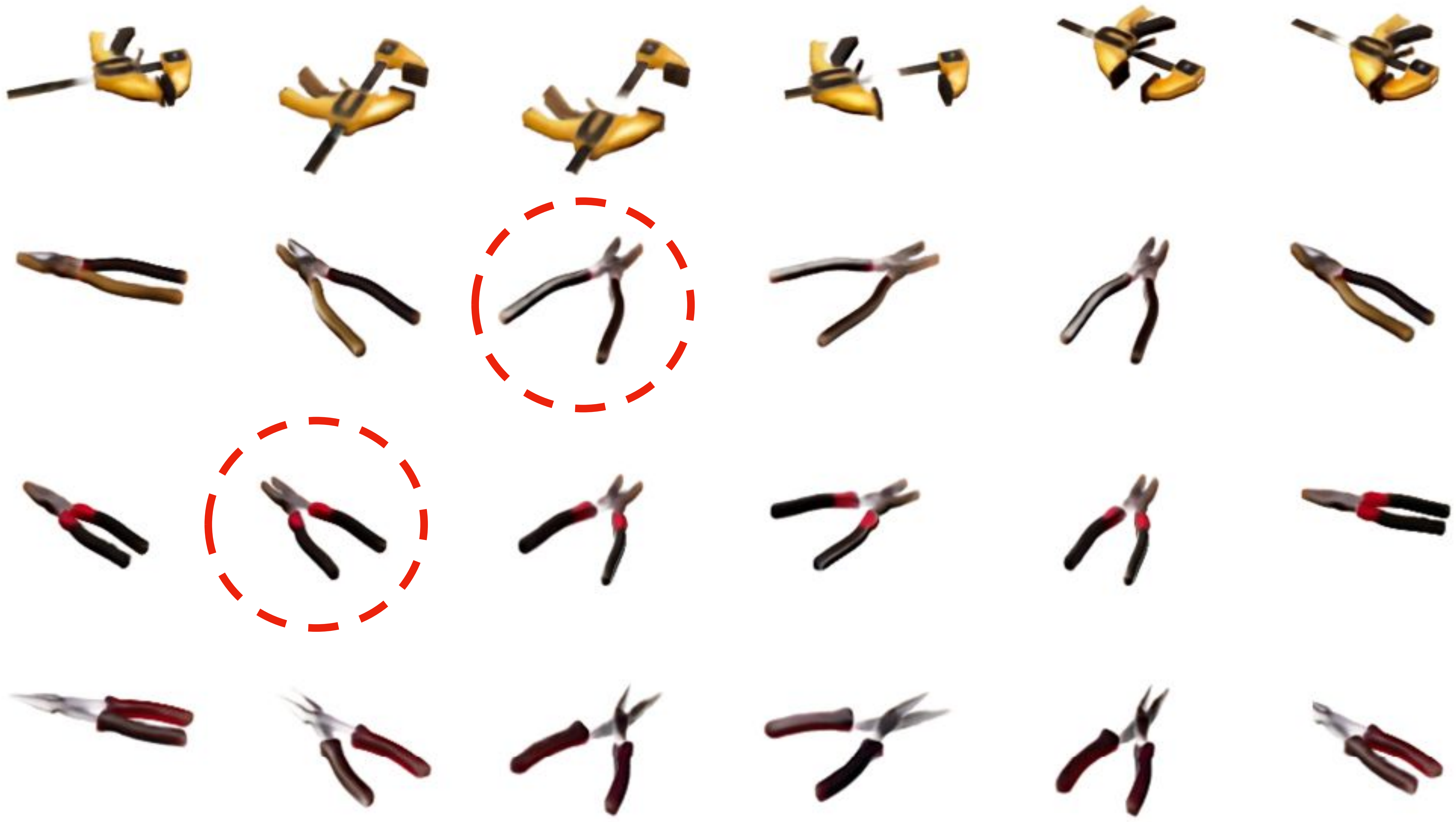


Figure 6. Render Result for Arbitrary Configurations

Renderings of Novel Configurations and Viewpoints

- Small labeling errors affect renderings for the pliers

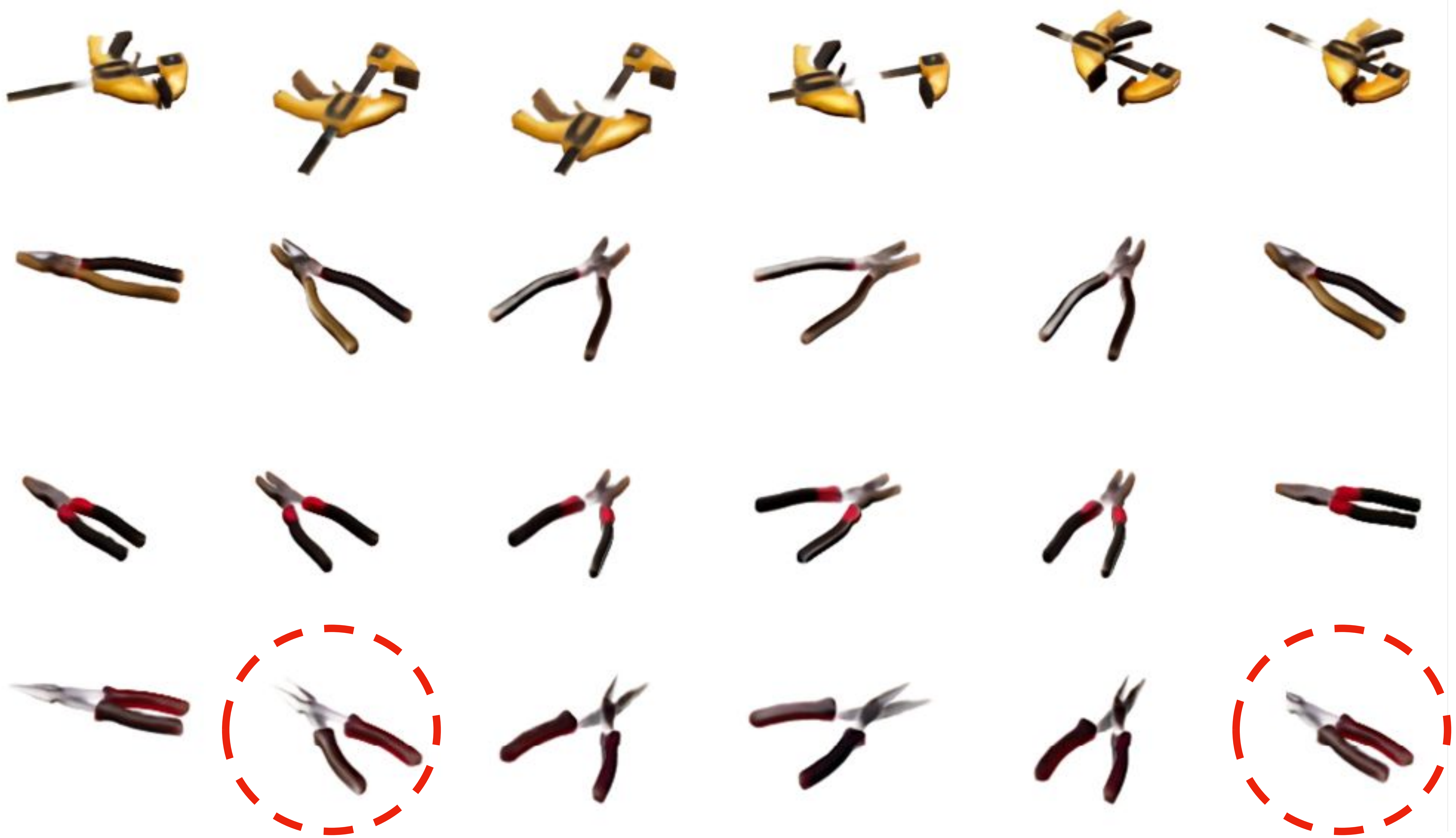


Figure 6. Render Result for Arbitrary Configurations

Articulated Object Rendering Results

- Test dataset contained same scenes and configurations, but different viewing angles
- Novel Config dataset contains additional scenes and novel configurations
- Per Pixel MSE - mean squared error between the pixel values for each R,G,B channel in the ground truth image vs rendered image

Table 1: Render Accuracy Metrics

Tool	Dataset	Per Pixel MSE	N (instances)
Clamp	Test	0.03343	272
	Train	0.03234	2483
	Novel Config	0.13245	200
Lineman's (A)	Test	0.02991	171
	Train	0.02941	1557
	Novel Config	0.10136	345
Lineman's (B)	Test	0.06348	171
	Train	0.06318	1557
	Novel Config	0.12500	326
Longnose	Test	0.08045	171
	Train	0.07900	1557
	Novel Config	0.10241	171

Articulated Object Rendering Results (cont.)

*Renderings compared to ground truth images for novel configurations

Figure 7. Qualitative vs Quantitative Results for Renderings

Articulated Object Rendering Results (cont.)

*Renderings compared to ground truth images for novel configurations

Figure 7. Qualitative vs Quantitative Results for Renderings

Poor performance due to mislabeled ground-truth mask

Articulated Object Rendering Results (cont.)

*Renderings compared to ground truth images for novel configurations

Figure 7. Qualitative vs Quantitative Results for Renderings

Poor performance due to difficult viewing angle and extreme pose

Configuration Estimation Results

- Pose refinement and configuration estimation from rigid-body pose estimation
- Perform gradient descent optimization on joint configuration and pose inputs to the renderer
- ADD - average euclidean distance between corresponding points of the object in ground truth and estimated poses
- Configuration Error: error between ground truth and estimated configuration

Table 2: Configuration Estimation Metrics

Metric	Mean	Std. Dev.
ADD (m)	0.0107	0.0043
Configuration Err. (m)	0.0073	0.0065

Conclusions

- Presents a training pipeline that allows for the renderings of articulated objects with arbitrary views and configurations
- Two-stage training process, first training on individual parts before using semi-synthetic data and tool structure for final rendering
- Able to perform gradient descent to perform pose refinement and configuration estimation

Limitations and Future Work

- Requires URDFs of articulated objects for training
- Highly sensitive to small errors in ground truth labeling
- Unable to deal with variable lighting conditions
- Currently limited to uncluttered scenes

Thank you

Next Time: Data for Deep Learning

- Seminar 9: Datasets
  1. Deep Learning for Robots: Learning from Large-Scale Interaction, Levine et al., 2016
  2. Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning, Makoviychuk et al., 2021
  3. Grounding Predicates through Actions, Migimatsu and Bohg, 2022
  4. All You Need is LUV: Unsupervised Collection of Labeled Images using Invisible UV Fluorescent Indicators, Thananjeyan et al., 2022
- Seminar 10: Self-Supervised Learning
  1. Making Sense of Vision and Touch: Self-Supervised Learning of Multimodal Representations for Contact-Rich Tasks, Lee et al., 2019
  2. VICRegL: Self-Supervised Learning of Local Visual Features, Bardes et al., 2022
  3. Fully Self-Supervised Class Awareness in Dense Object Descriptors, Hadjivelichkov and Kanoulas, 2022
  4. Self-Supervised Geometric Correspondence for Category-Level 6D Object Pose Estimation in the Wild, Zhang et al., 2022

DeepRob

Seminar 8
Implicit Scene-Level Representations
University of Michigan and University of Minnesota