Neural Fields

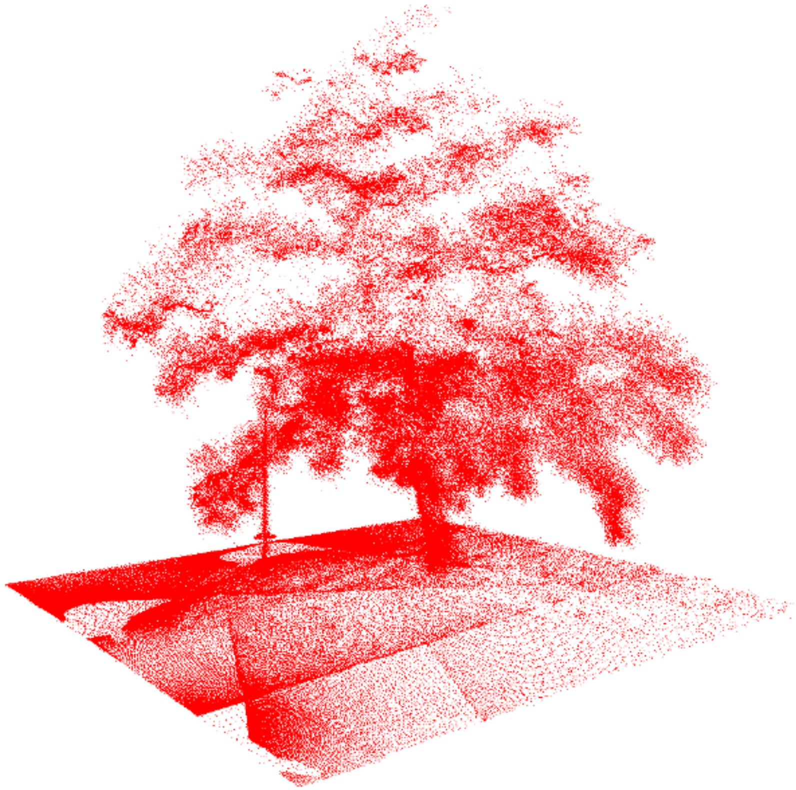
Maren Bennewitz, Sicong Pan
Humanoid Robots Lab, University of Bonn

Goal of This Chapter

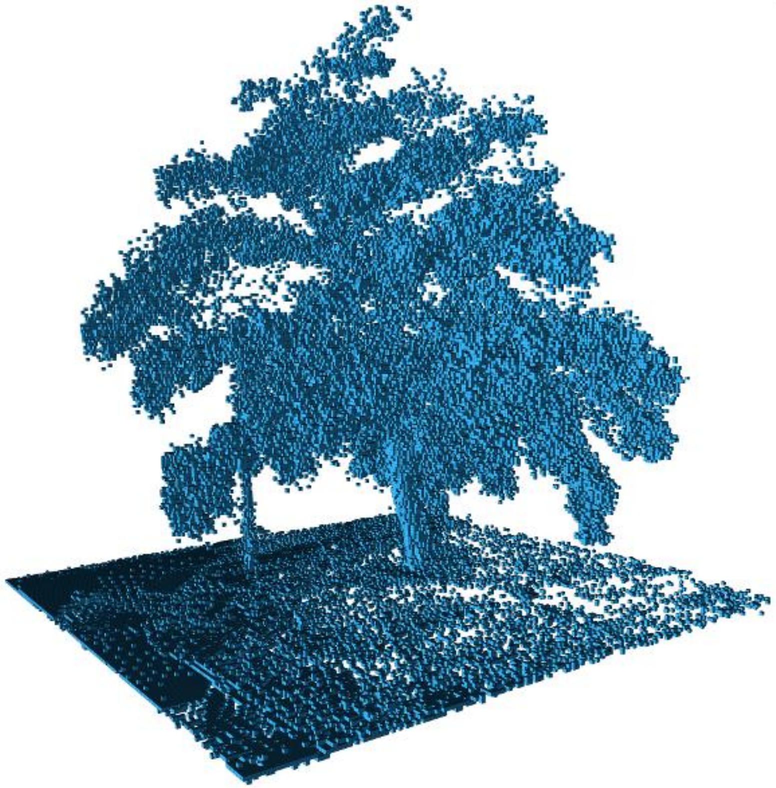
- Overview of different types of neural field representations with milestones in their development
- Discuss how neural fields can be integrated into robotic systems for improved perception and scene understanding
- Learn how to implement simple differentiable rendering techniques for optimization directly from images

Recap of Conventional 3D Representation

- Point clouds
- Voxel grids
- Meshes
- Distance fields
- …



Point cloud



Voxel grid

Limits of Conventional 3D Representation

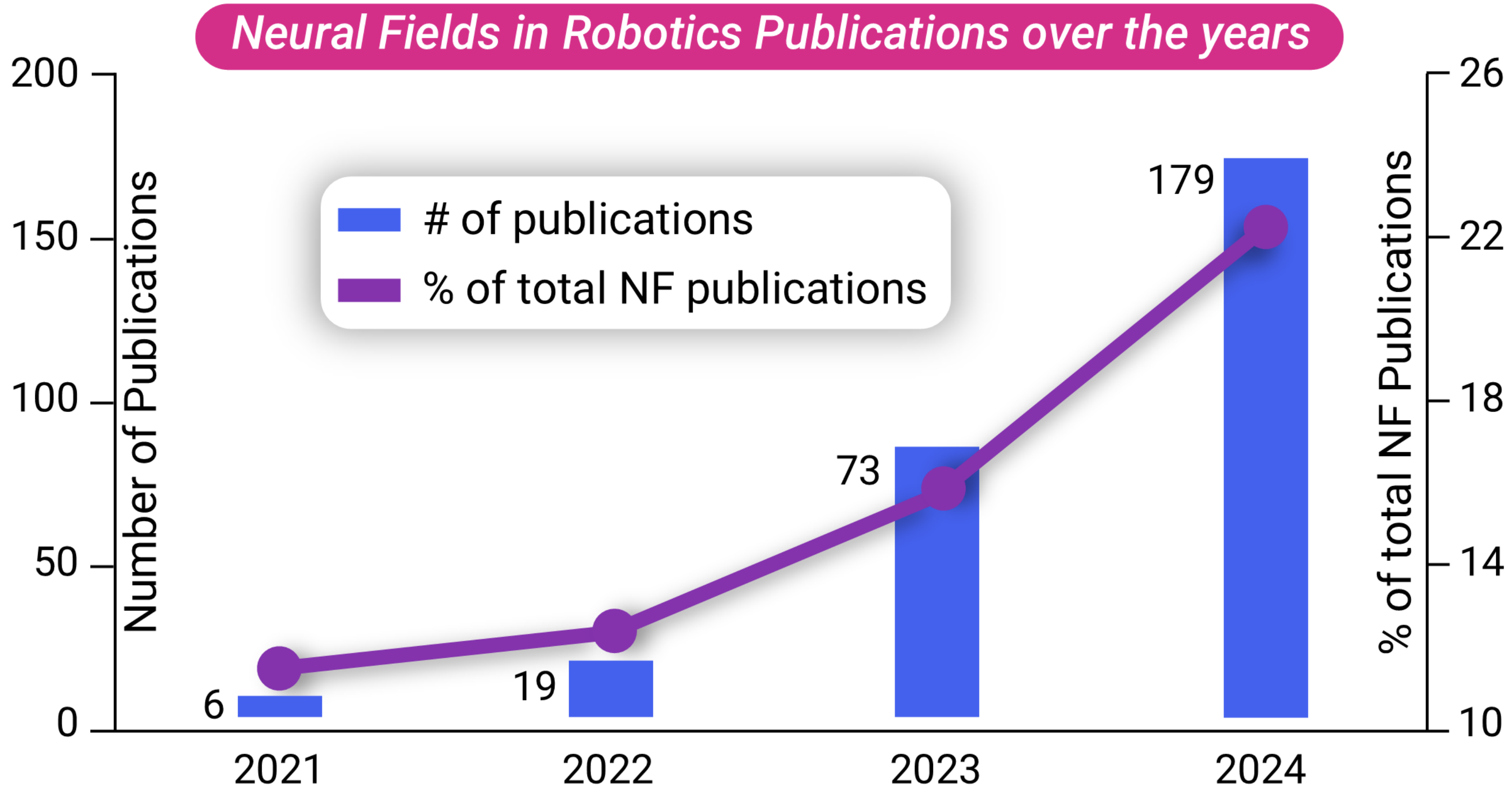
- Explicit representation often lacks detail due to resolution
- Limited continuity and smoothness
- Difficulty handling complex geometry
- High memory and storage requirements
- Often limited scalability for large and dynamic scenes

Motivation

- Achieve high-fidelity 3D reconstructions with virtually unlimited resolution
- Provide continuous and compact representations
- Learn to handle complex geometries
- Reduce memory and storage needs via efficient encoding
- Scale and generalize well to large and dynamic scenes

Growth of Neural Fields in Robots

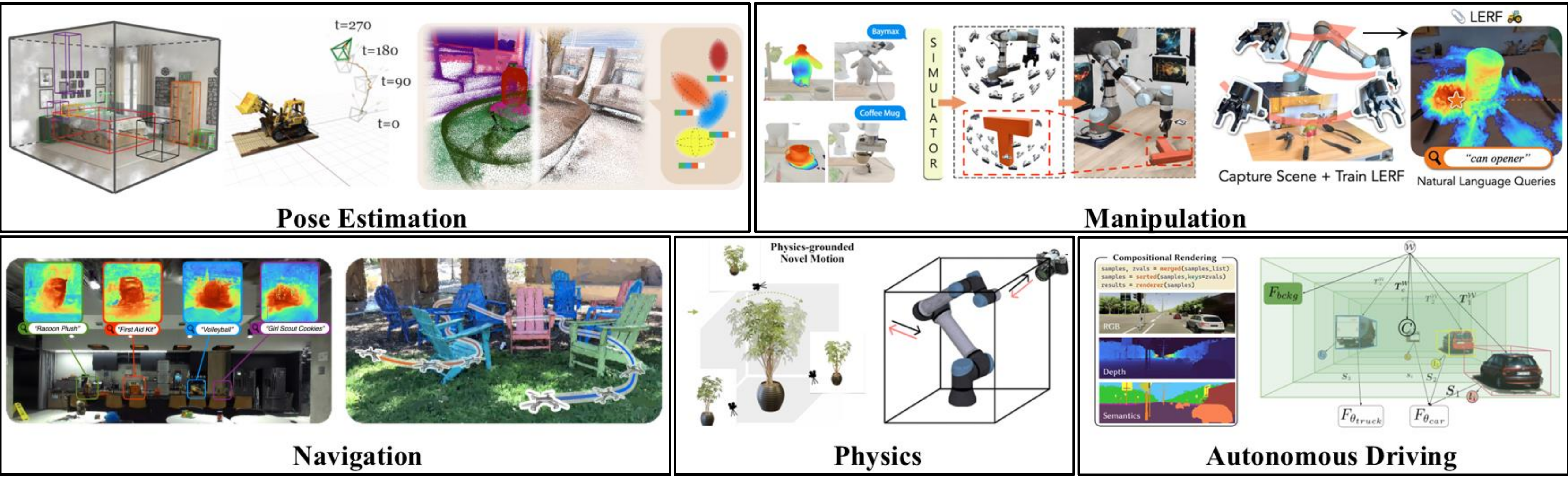
- Number of Publications: 6 to 179
- Percentage of Total Neural Field Publications: 11% to 22%



Growth of Neural Fields in Robotics, Irshad et al., ArXiv preprint, 2024

Robotics Applications of Neural Fields

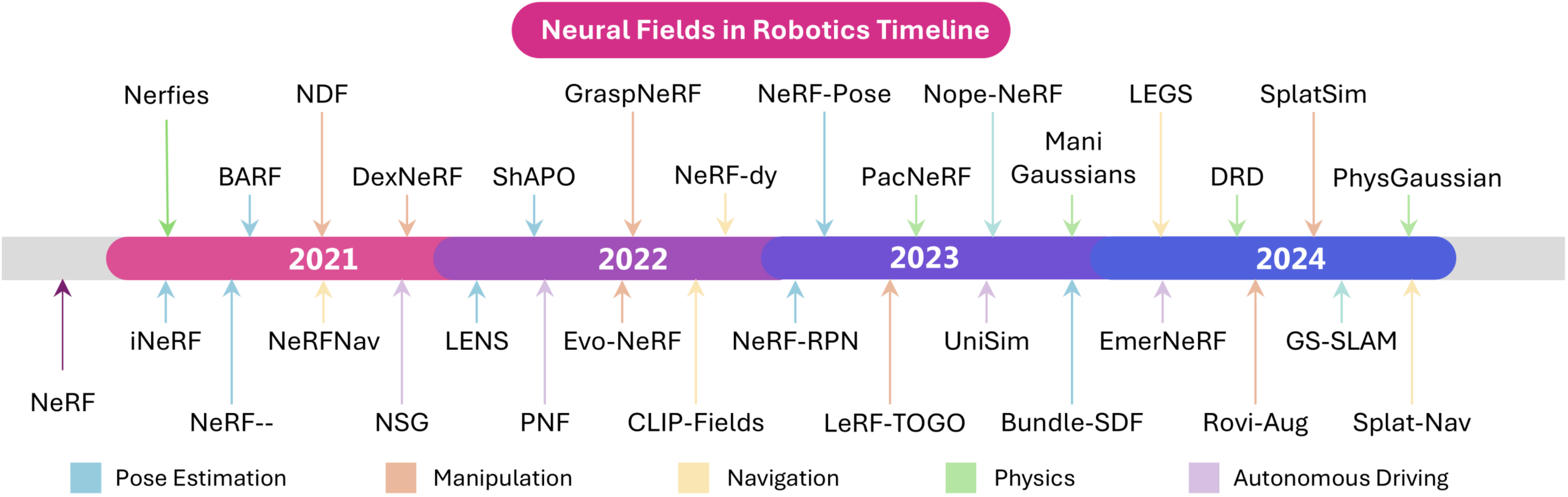
- Five major robotics application areas



Overview of Robotics Applications, Irshad et al., ArXiv preprint, 2024

Timeline of Papers

- Key papers divided into 5 major application areas



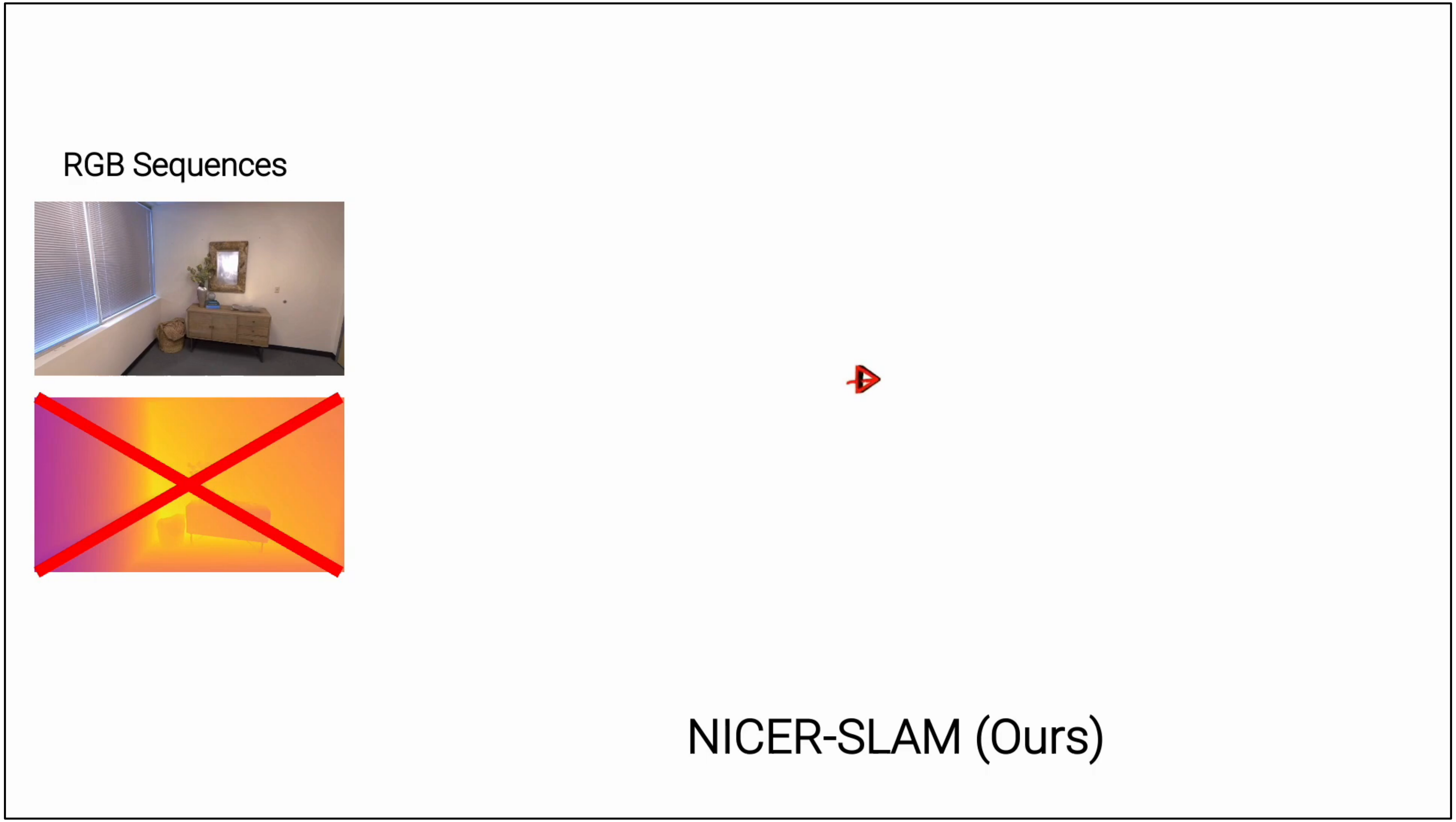
Timeline of Neural Fields in Robotics, Irshad et al., ArXiv preprint, 2024

PhysGaussian (CVPR 2024)

- Physically grounded **dynamics** for novel motion synthesis

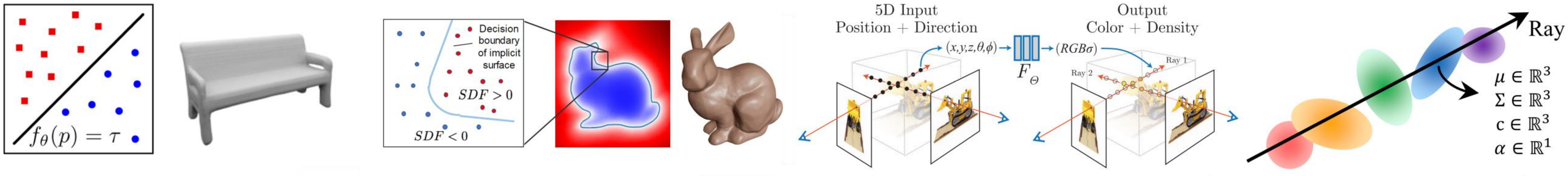
NICER-SLAM (3DV 2024)

- Dense **RGB** SLAM and high-quality novel view synthesis



Neural Fields - Property

- Four core neural field representations
  - A) Occupancy Networks vs. OctoMap
  - B) DeepSDF, NeuS vs. SDF
  - C) Neural (I, NeRF) and Explicit (II, GS) Radiance Fields



A) Occupancy Networks B) Signed Distance Fields C-I) Neural Radiance Fields C-II) Gaussian Splatting

Neural Field Representations, Irshad et al., ArXiv preprint, 2024

Neural Fields - Input Dependency

- View-independent fields
  - Examples: Occupancy Networks, DeepSDF
  - Field value: occupancy, signed distance, etc.
- View-dependent fields
  - Examples: NeuS, NeRF, GS
  - Field value: RGB color, density, etc. (conditioned on viewing direction)

Formulation of View-Independent Fields

- Given a 3D point $p(x, y, z) \in \mathbb{R}^3$
- A view-independent neural field defines a scalar function $f(p): \mathbb{R}^3 \rightarrow \mathbb{R}$ that returns the field value at point p
- Field value $f(p)$ represents a physical or geometric property, such as color, occupancy, SDF, or density
- **Note:** Similar to conventional 3D representations, but using a neural network as the **continuous** field function

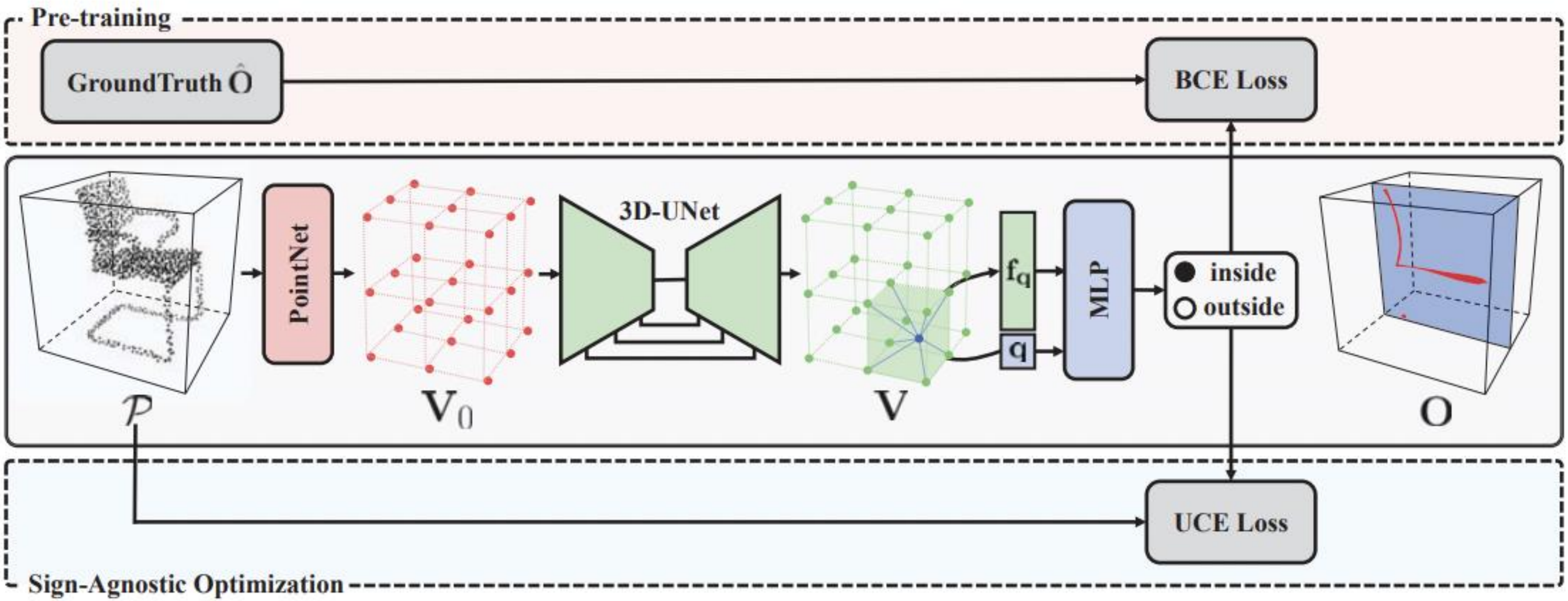
Occupancy Networks

- $f(p)$ predicts the occupancy probability indicating whether the point lies inside or outside a surface
- The key variation across occupancy-based methods lies in the network inputs (e.g., image, point cloud, voxel) and the architecture used to learn $f(p)$

Demo results, Mescheder et al. CVPR 2019

SA-ConvONet

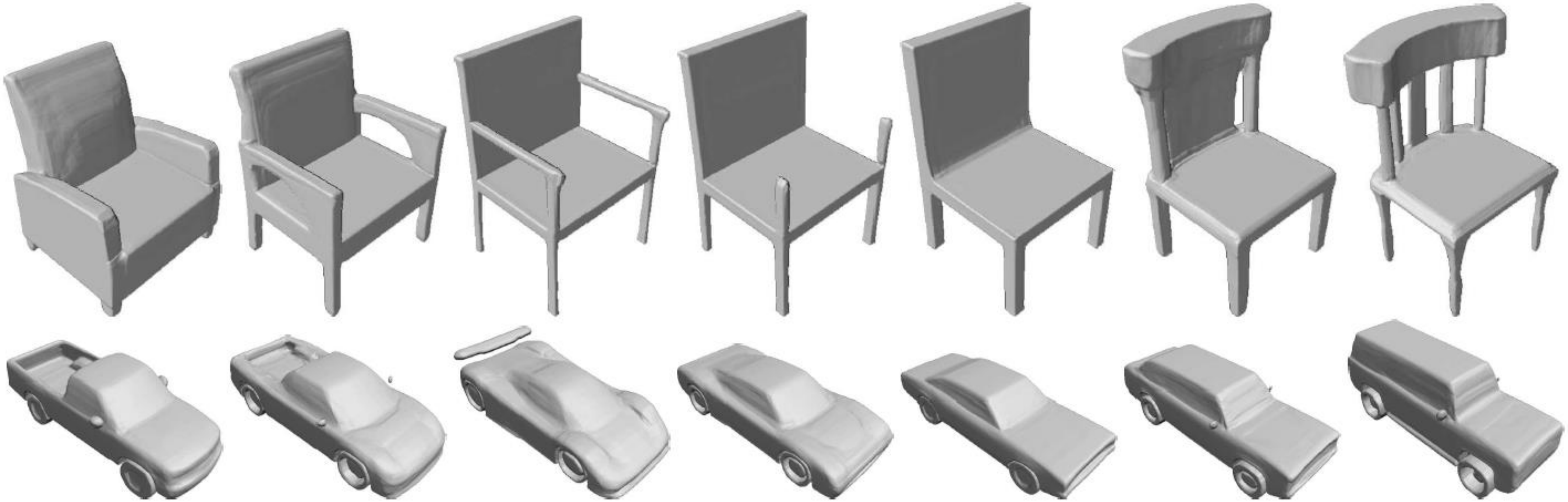
- Voxel Input
- Architecture: PointNet + 3D-UNet



SA-ConvONet, Tang et al. CVPR 2021

Neural SDF

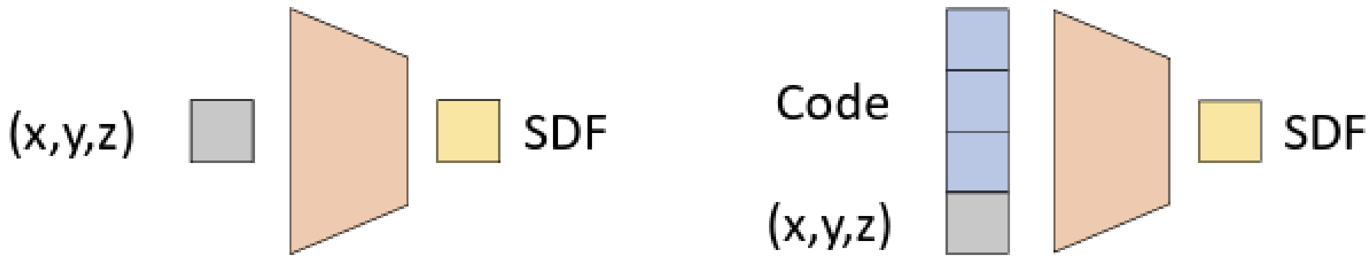
- $f(p)$ predicts the signed distance value from the point p to its nearest surface
- The key variation still lies in the architecture



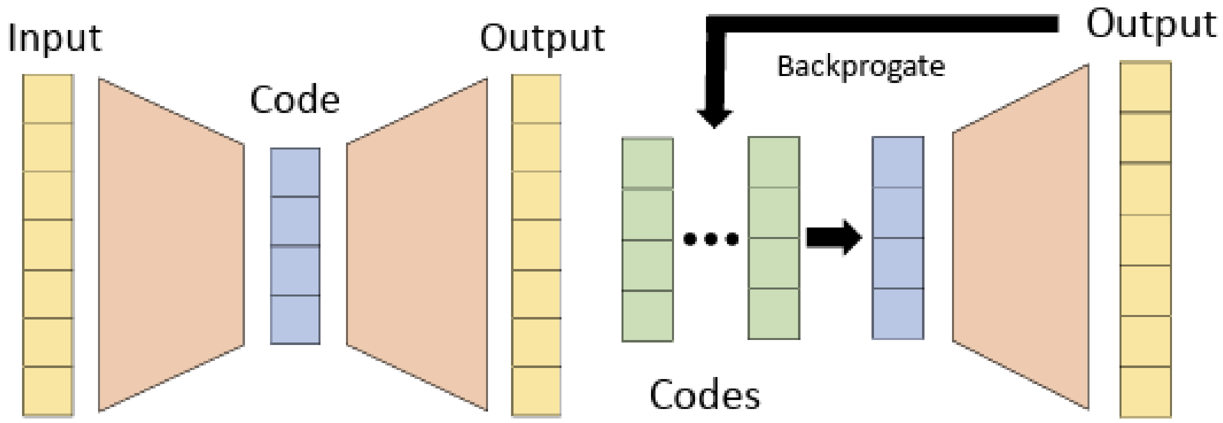
Demo results, Park et al. CVPR 2019

DeepSDF

- Auto-encoder and auto-decoder shape DeepSDF



(a) Single Shape DeepSDF (b) Coded Shape DeepSDF



(a) Auto-encoder (b) Auto-decoder

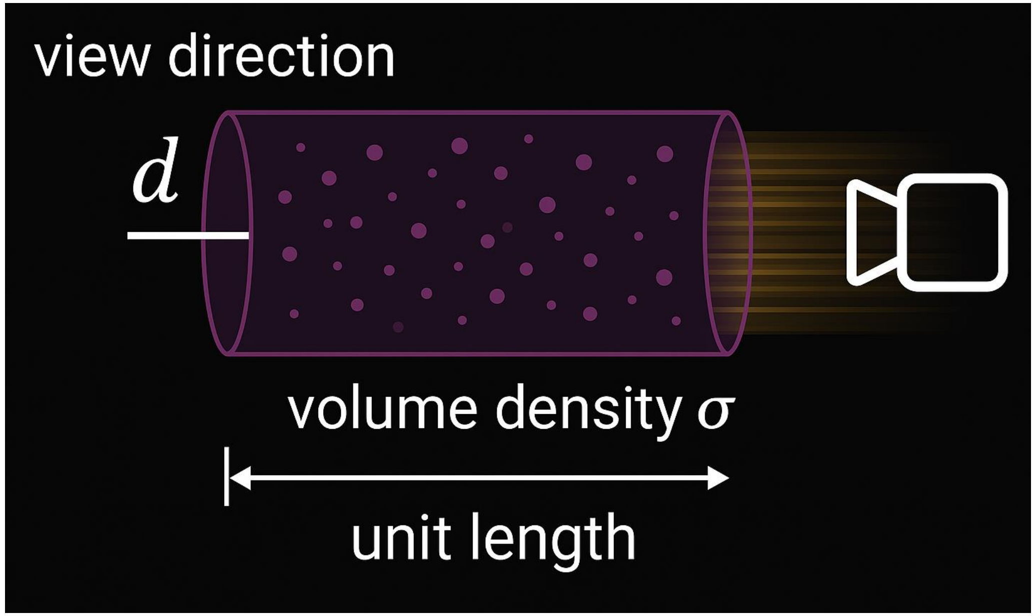
DeepSDF, Park et al. CVPR 2019

From View-independent to View-dependent

- Early neural fields focus on geometry only, such as SDF or occupancy
- These models are typically view-independent and cannot produce rendered RGB images
- **Volume rendering** enables projecting a 3D field into a 2D image by integrating along each pixel’s viewing direction

Volume Rendering Theory

- Opaque (solid) regions block light — less light passes through
- Transparent or empty regions allow more light to pass
- **Volume density** $\boldsymbol{\sigma}$ defines how much light is absorbed per unit length
- Integration along the viewing direction determines the final image intensity



Formulation of View-Dependent Fields

- Given a 3D point $p(x, y, z) \in \mathbb{R}^3$ and a viewing direction $d \in \mathbb{R}^3$
- A view-dependent neural field defines a scalar or vector-valued function $f(p, d): \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^n$ that returns the field value at point p conditioned on the viewing direction d
- Note: Given a viewpoint 3D position $o \in \mathbb{R}^3$, the viewing direction is typically computed as a unit vector $d = \frac{p-o}{||p-o||}$

Differentiable Volume Rendering (RGB)

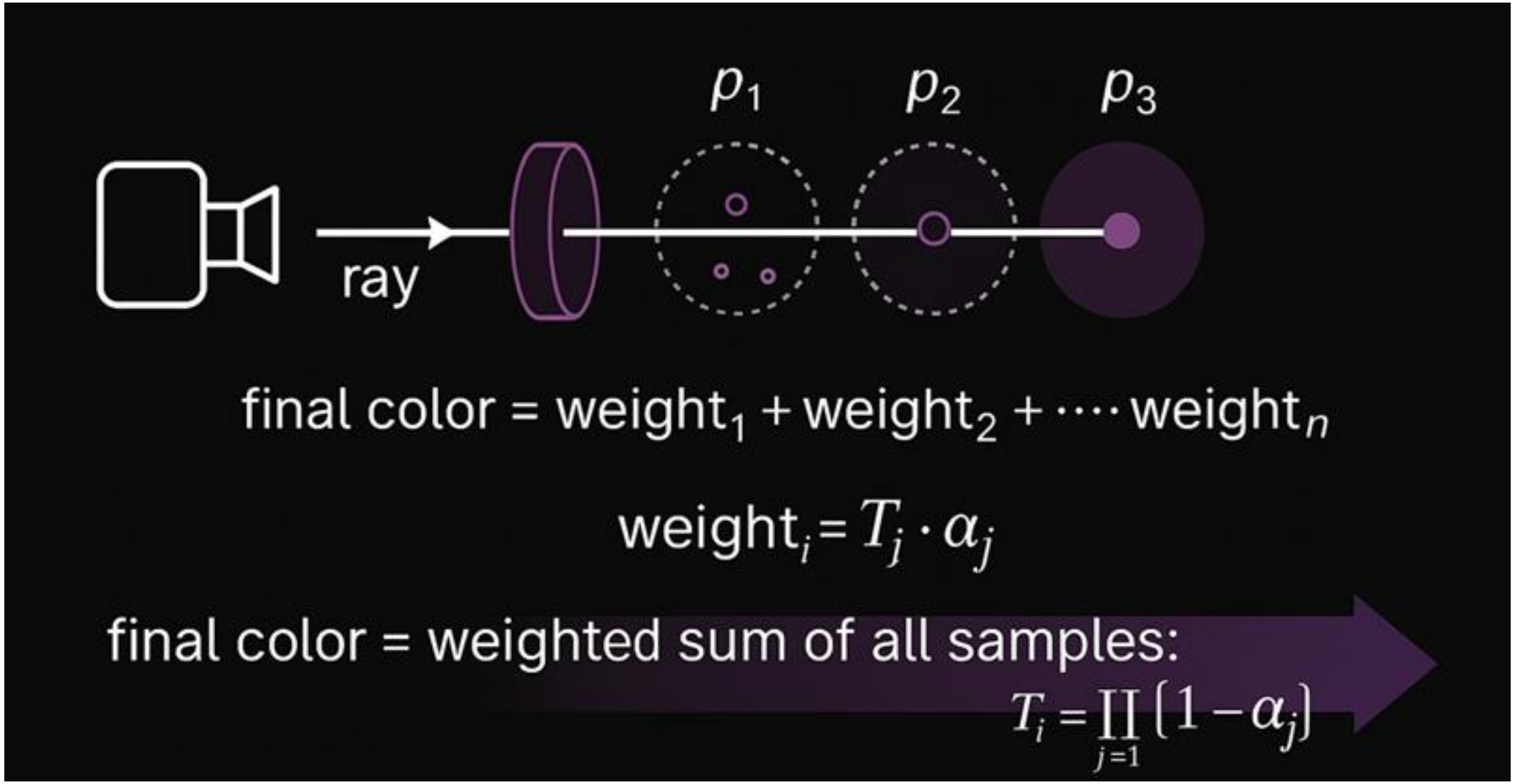
- We want to render the color of a pixel by accumulating color and transparency along a ray
- For a view-pixel ray $r(t) = o + td$, we sample N points p_i along the ray, where t is the step size
- At each point p_i, the neural field $f(p, d)\colon \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^4$ predicts σ_i volume density and $c_i \in \mathbb{R}^3$ RGB color
- The final rendered pixel color $\hat{C}$ is computed as:

$$\hat{C}(r) = \sum_{i=1}^{N} T_i * \alpha_i * c_i\,,$$

where $\alpha_i = 1 - \exp(-\sigma_i(t_{i+1} - t_i))$ and $T_i = \prod_{j=1}^{i-1}(1 - \alpha_j)$

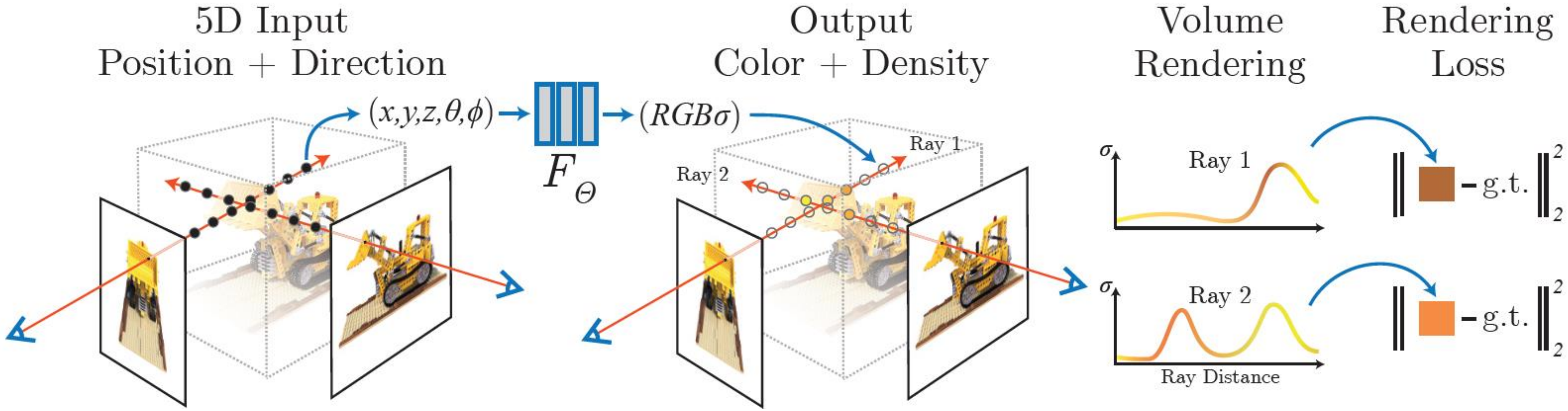
Visualizing Volume Rendering

- Early opaque samples contribute **more**; later samples are **faded** by accumulated transmittance



Neural Radiance Field (NeRF)

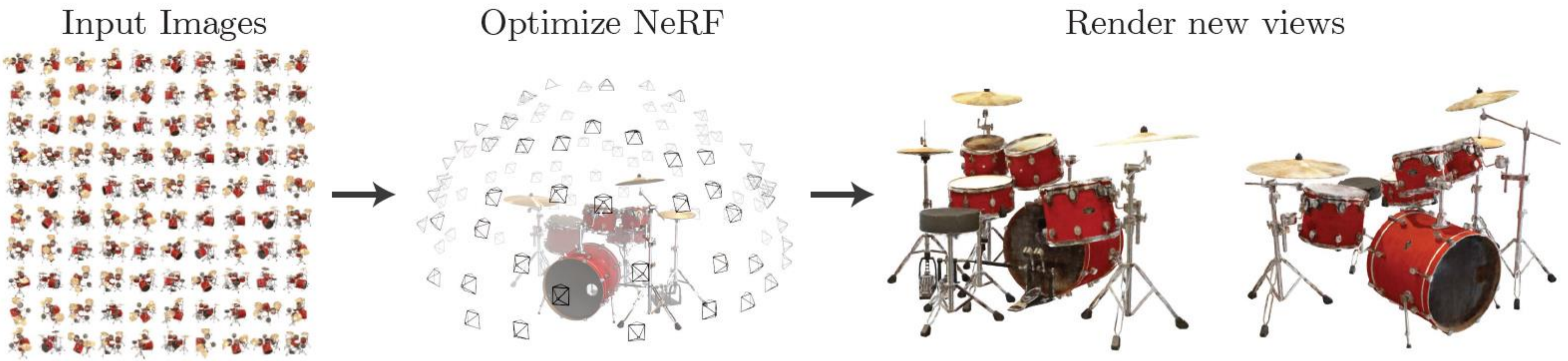
- Sampled points along each ray (as spherical coordinates $\mathbb{S}^2$) are passed through the network
- Final color is computed via differentiable volume rendering
- Supervised by comparing rendered pixel colors to ground-truth RGB images



NeRF, Mildenhall et al. ECCV 2020

Neural Radiance Field (NeRF)

- Learning from a set of posed RGB images
- Unlimited resolution for novel view rendering
- Much less memory: 15GB 3D voxel grid vs. 5 MB NeRF



Practical Usage of NeRF, Mildenhall et al. ECCV 2020

Neural Radiance Field (NeRF)

NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis

Ben Mildenhall* UC Berkeley
Pratul P. Srinivasan* UC Berkeley
Matthew Tancik* UC Berkeley
Jonathan T. Barron Google Research
Ravi Ramamoorthi UC San Diego
Ren Ng UC Berkeley

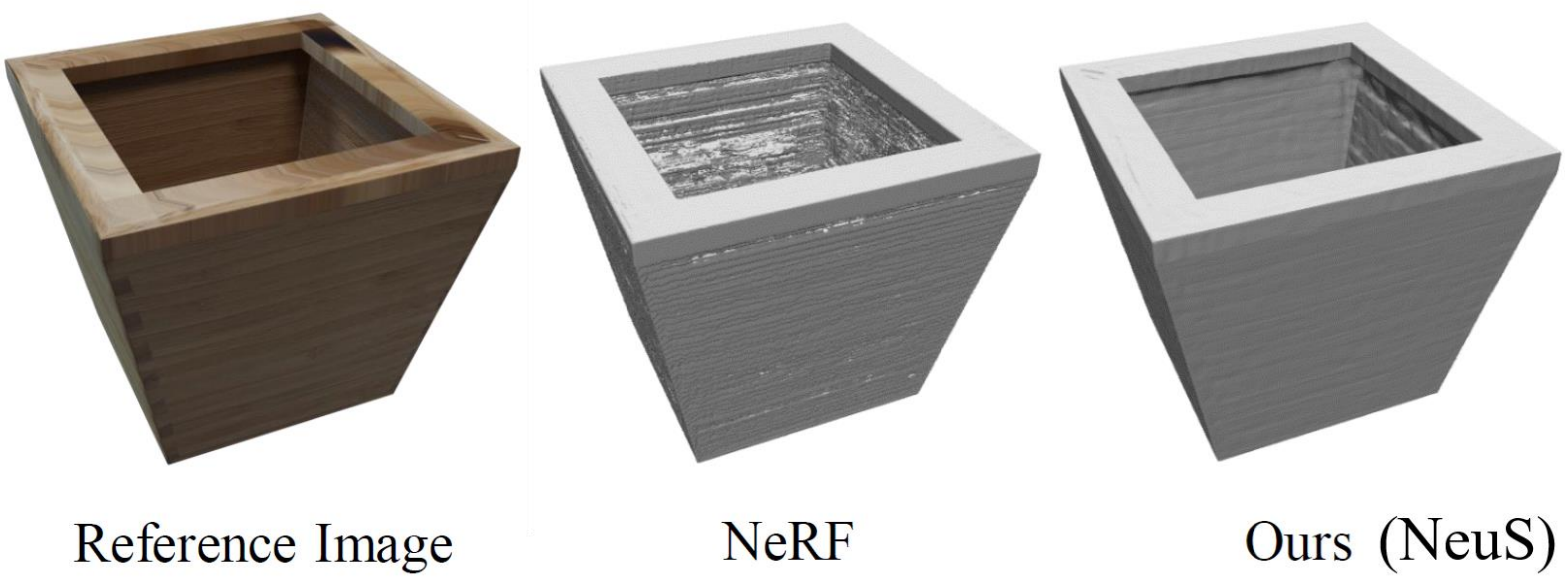
* Denotes Equal Contribution

Building Mini NeRF via PyTorch3D

NeRF Component	PyTorch & PyTorch3D API	Description
Ray representation	pytorch3d.renderer.RayBundle	Stores ray origins, directions, and sample intervals
Ray sampling	pytorch3d.renderer.MonteCarlo Raysampler	Samples rays for training
Neural field $f(p, d)$	Custom nn.Module	Maps 3D point p and view direction d to RGB and density
Field evaluation	your_mlp(points, directions)	Forward pass through the neural network
Volume rendering	pytorch3d.renderer.EmissionAbsorption Renderer	Composites color along the ray using alpha-weighted sum
Renderer wrapper	pytorch3d.renderer.ImplicitRenderer	Wraps ray sampling and rendering into a single module
Loss function	torch.nn.functional.mse_loss() or huber (smooth-l1) loss	Computes loss between rendered and ground-truth RGB

SDF Meets Volume Rendering

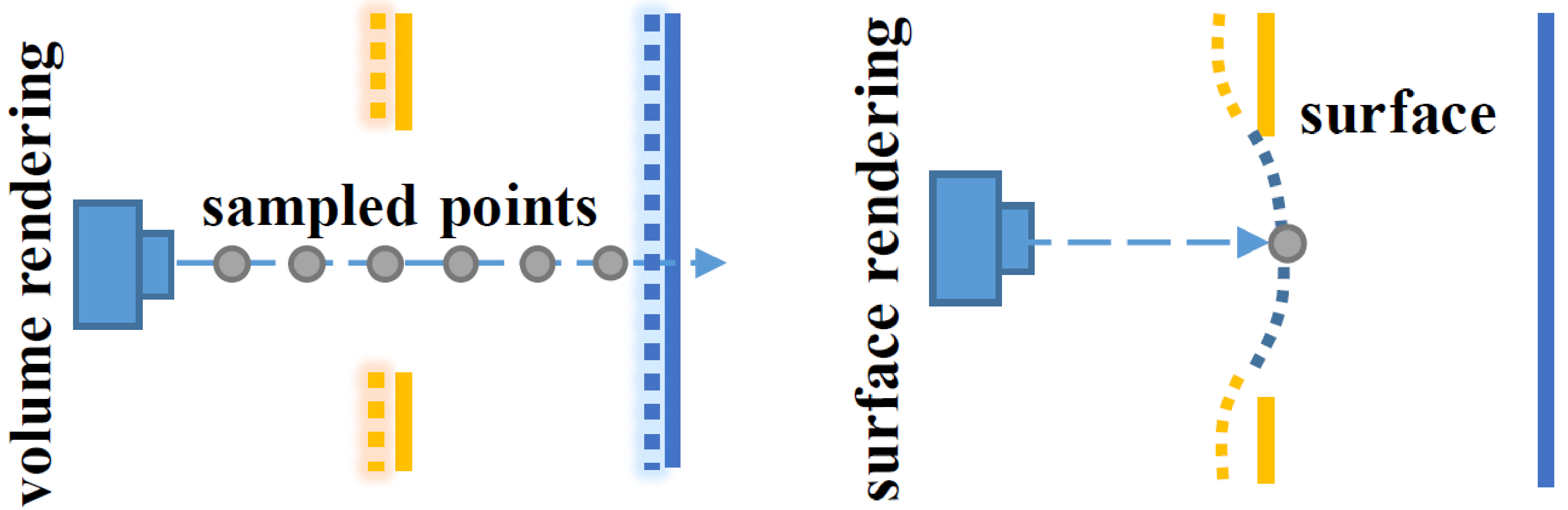
- Surface extraction from NeRF via marching cubes is noisy and resolution-limited
- Combining SDF with volume rendering enables photorealistic view synthesis and accurate surface reconstruction



Demo meshing results, Wang et al. NeurIPS 2021

Neural Implicit Surface (NeuS)

- Two different fields: $SDF(p): \mathbb{R}^3 \rightarrow \mathbb{R}$ and $C(p,d): \mathbb{R}^3 \times \mathbb{S}^2 \rightarrow \mathbb{R}^3$
- NeRF struggles with color ambiguity along rays with complex geometry due to depth discontinuity (e.g. a hole)
- NeuS uses surface-aware rendering to avoid such mistakes



Volume rendering vs. Surface rendering, Wang et al. NeurIPS 2021

Why S-density? Surfaces Smoothly Glow

- We want only the surface (i.e. SDF ≈ 0) to glow
- So NeuS defines a smooth function that peaks at surface and drops off nearby
- This S-density is actually the derivative of the sigmoid function (logistic):

$$\phi_s(x) = \frac{d}{dx}\mathbf{\Phi}_s(x) = \frac{se^{-sx}}{(1+e^{-sx})^2}$$

where $\mathbf{\Phi}_s(x) = \frac{1}{1+e^{-sx}}$, x is SDF value, and s controls sharpness (higher s means sharper peak at surface)

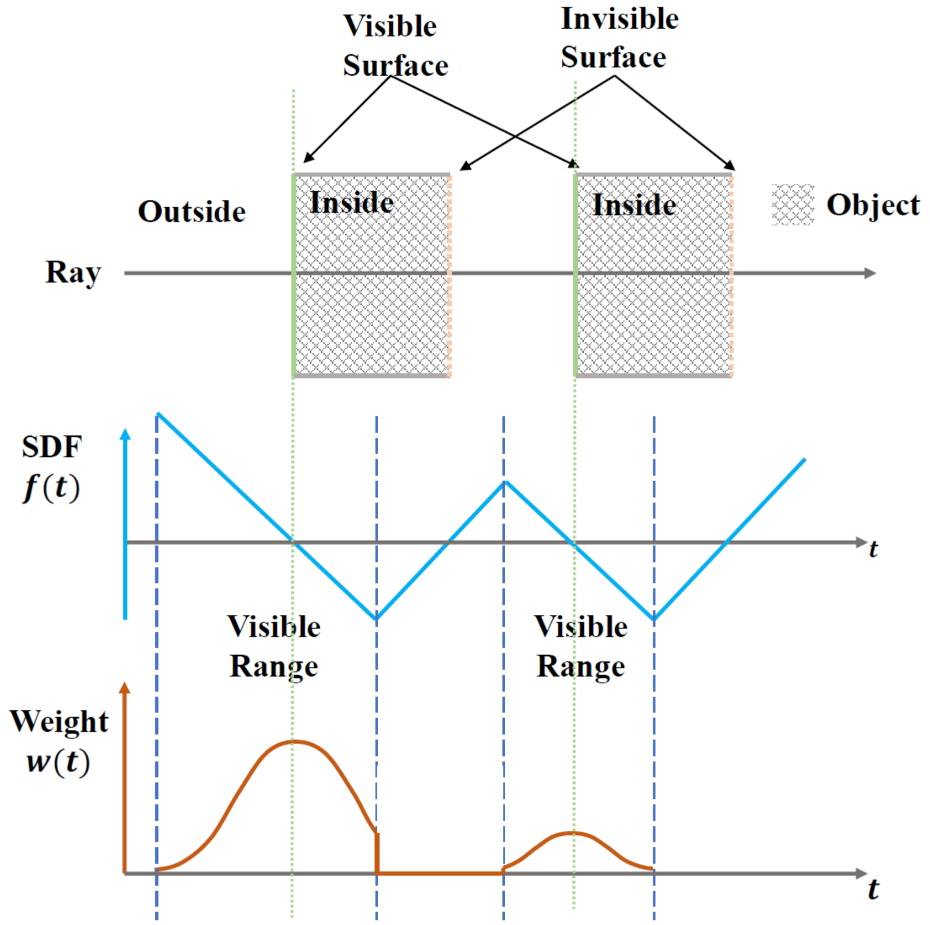
Surface Rendering with S-density

- Compute color along a ray by integrating surface-based weights:

$$C(r) = \sum_{i=1}^{N} T_i * \phi_s(SDF(p_i)) * c_i$$

$$T_i = \prod_{j=1}^{i-1} \left(1 - \phi_s\left(SDF\left(p_j\right)\right) * (t_{i+1} - t_i)\right)$$

- Here T_i still means accumulated transparency as in NeRF
- Only visible surfaces glow!



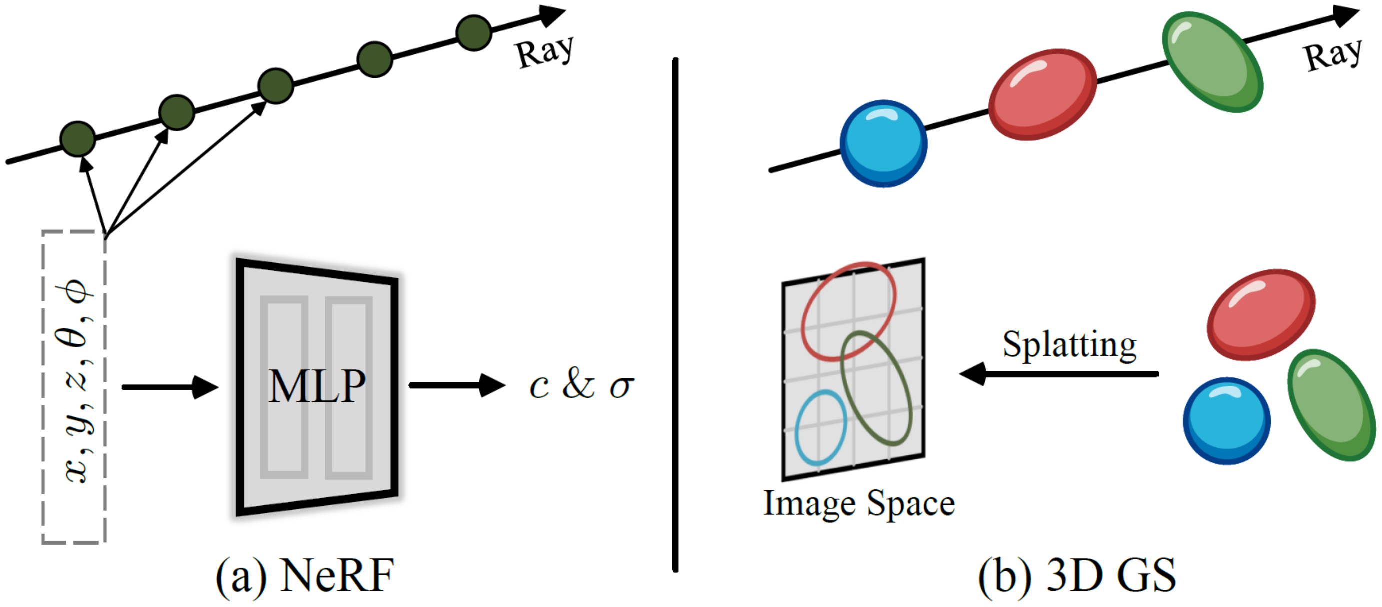
Multiple surface intersection, Wang et al. NeurIPS 2021

Limitations of NeuS and NeRF

- Less efficient in interactive or real-time applications
- Computationally expensive (slow training and rendering)
  - Require lots of samples along rays → costly integration
  - Not friendly to hardware acceleration (e.g. rasterization)

3D Gaussian Splatting (3DGS)

- NeRF uses ray tracing (backward mapping): sample along rays and query an MLP
- 3DGS uses rasterization (forward mapping): project 3D Gaussians onto the image plane and splat in parallel



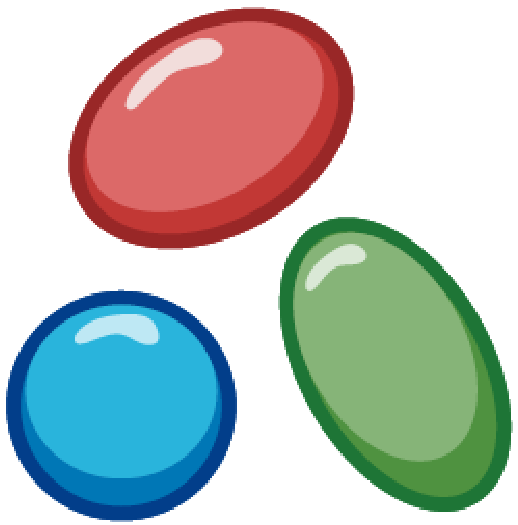
NeRF vs. 3D GS, Chen et al., ArXiv preprint, 2024

3D Gaussian Representation

- Think of each Gaussian as a soft, elliptical point in space
- Each 3D Gaussian has position $\mu \in \mathbb{R}^3$, covariance $\Sigma \in \mathbb{R}^{3\times 3}$ (shape & orientation), color: $c \in \mathbb{R}^3$, and opacity $\alpha \in [0,1]$

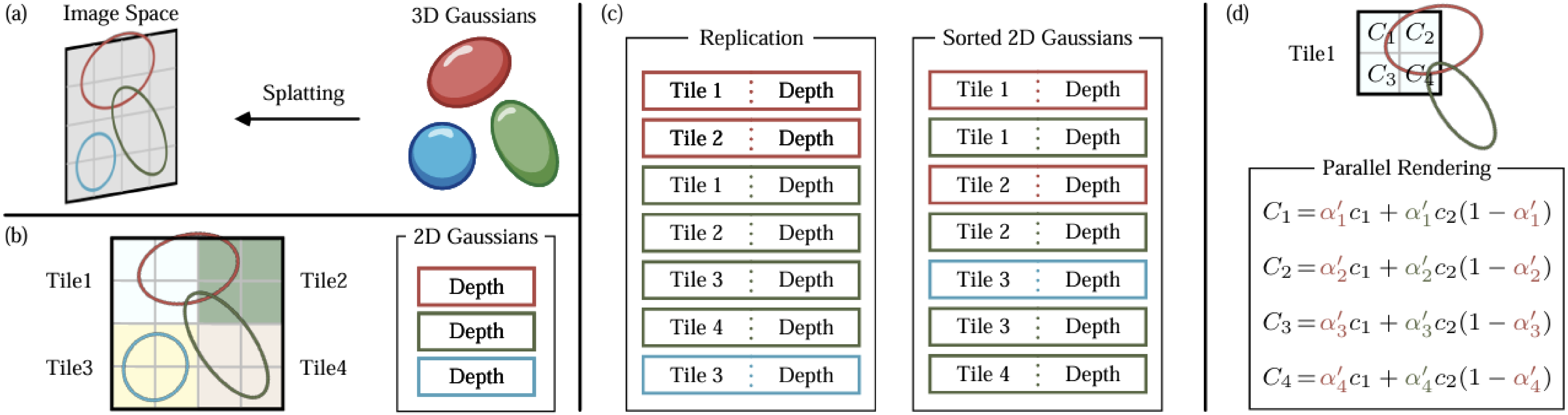
$$C(x) = \sum_i \alpha_i * g_i(x) * c_i$$

$$g_i(x) = \exp(-\frac{1}{2}(x-\mu)^{\mathrm{T}}\Sigma^{-1}(x-\mu))$$



Why 3DGS is Fast and Parallelizable

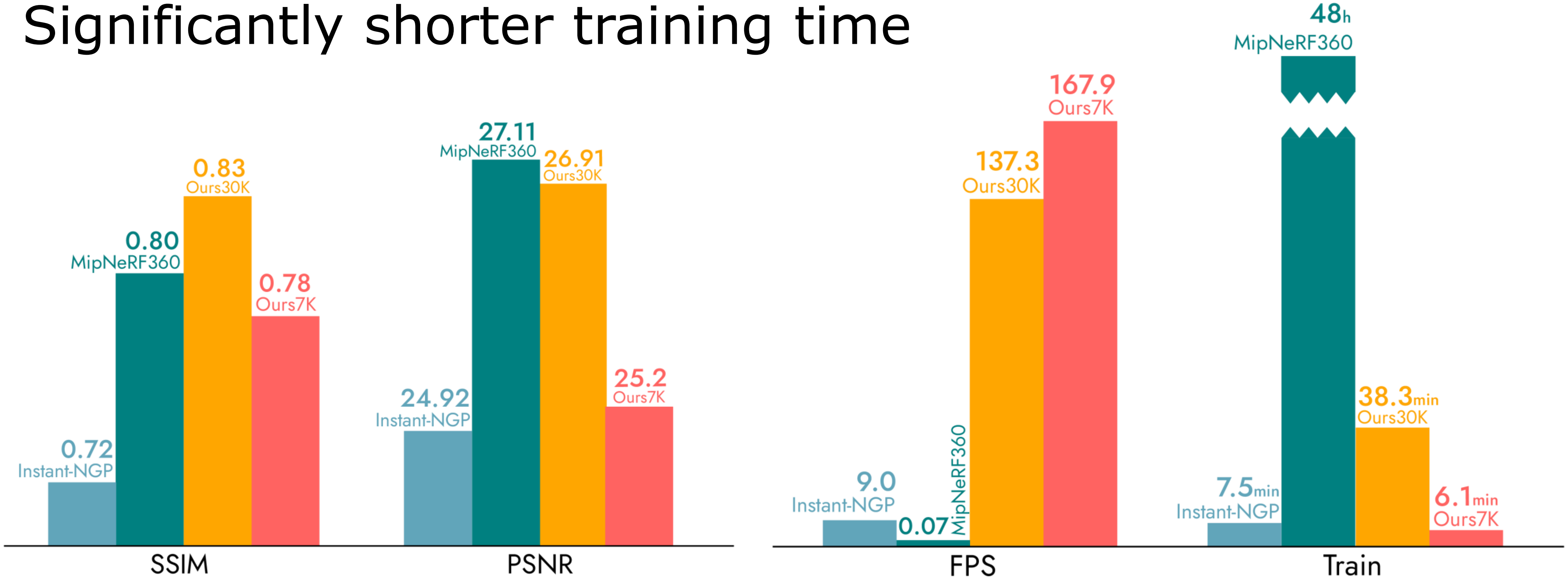
- No neural field: uses explicit 3D Gaussians instead of MLPs
- Optimize position, scale, color, and opacity directly
- Project Gaussians → tiles → sort by depth → render in parallel



Forward process of 3DGS, Chen et al., ArXiv preprint, 2024

Fast and Accurate 3DGS

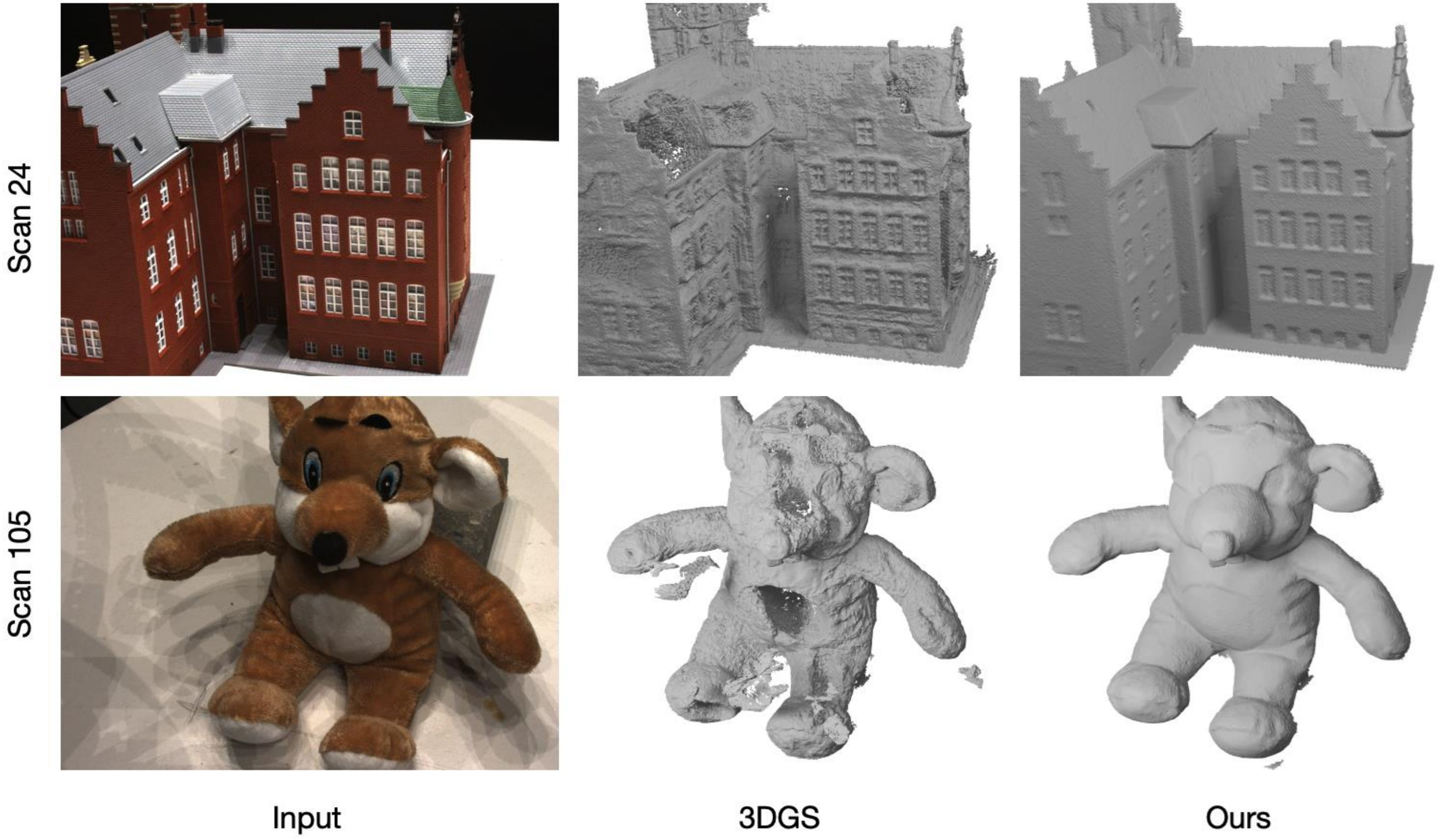
- Comparable or better image quality (SSIM/PSNR)
- 10x–1000x faster rendering (FPS)
- Significantly shorter training time



Comparison with NeRFs, Kerbl et al., ACM Trans. Graph., 2023

2D Gaussian Splatting (2DGS)

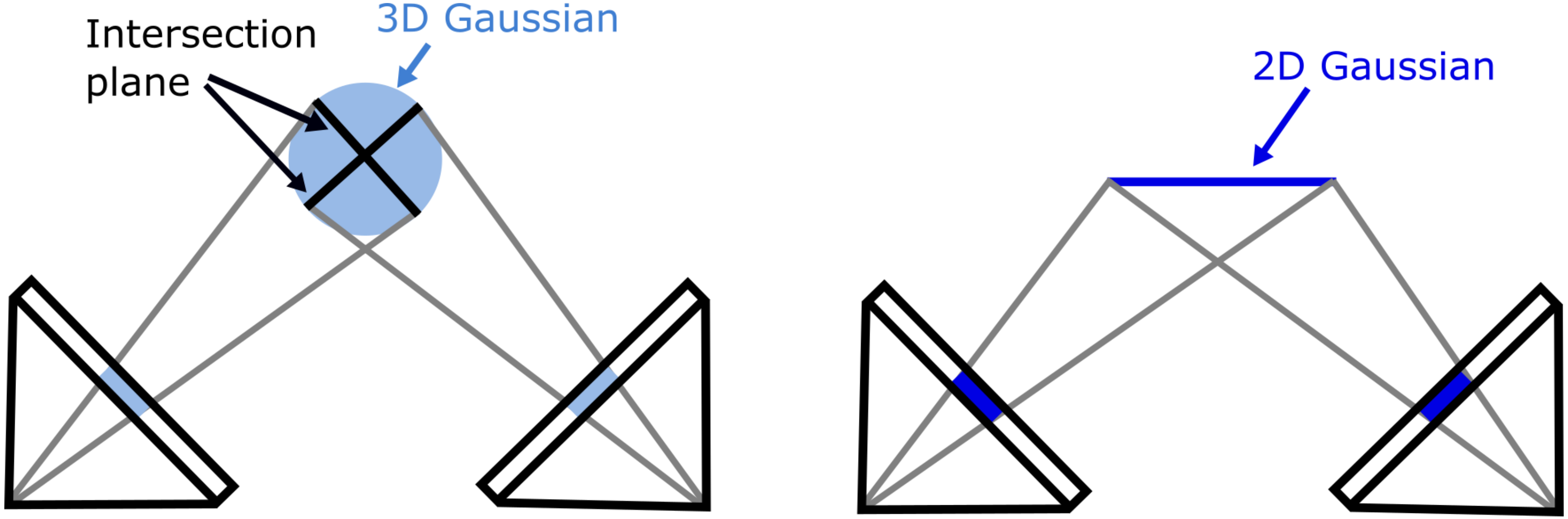
- 3DGS is fast, but lacks accurate surface geometry
- 2DGS adds mesh-aware splatting for better geometry



Demo meshing results, Huang et al., SIGGRAPH, 2024

From Volumes to Planar Disks

- Collapse each 3D Gaussian to a planar disk to the surface
- The 2D disk is oriented along the surface normal, but floats above the surface
- Extract the surface by aggregating the disks to a point cloud



Multi-view consistency, Huang et al., SIGGRAPH, 2024

Summary: Visual Neural Fields

- Visual neural fields enable **rich perception and interaction** for robotic applications
- Traditional methods (meshes, voxels): **fast** but limited in detail and flexibility
- View-independent fields (occupancy networks, DeepSDF): **fine geometry** but lack realistic appearance and efficiency
- View-dependent fields (NeRF, NeuS): both **fine geometry and appearance** but are slow and not real-time
- Gaussian Splatting (3DGS, 2DGS): efficient, explicit fields with real-time rendering—bridging **fidelity and speed**

Beyond Vision: Tactile Sensing

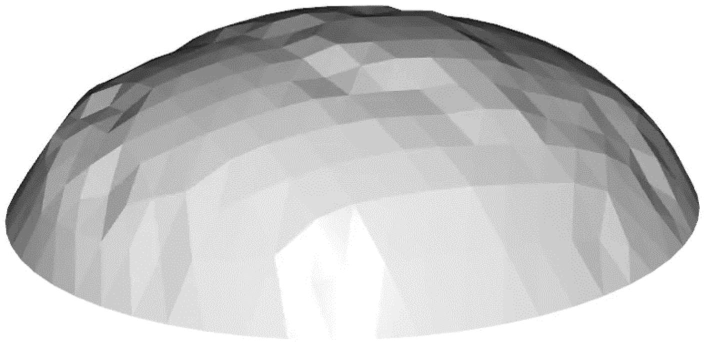
- Limitations of vision: occlusion, missing fine surface details, transparency issues
- Tactile sensing provides complementary physical feedback during interaction
- Allows direct measurements of surface geometry, texture, and force distribution
- Emerging trend: combining tactile and visual inputs via **neural fields** for better 3D reconstruction and material estimation

Integration of Vision and Tactile Data

- Complementary strengths: Vision for global shape and structure
- Tactile for fine-grained details and hidden areas
- Enhanced reconstruction accuracy and robustness
- Enables richer geometric and material property extraction

Punyo Visuotactile Sensor

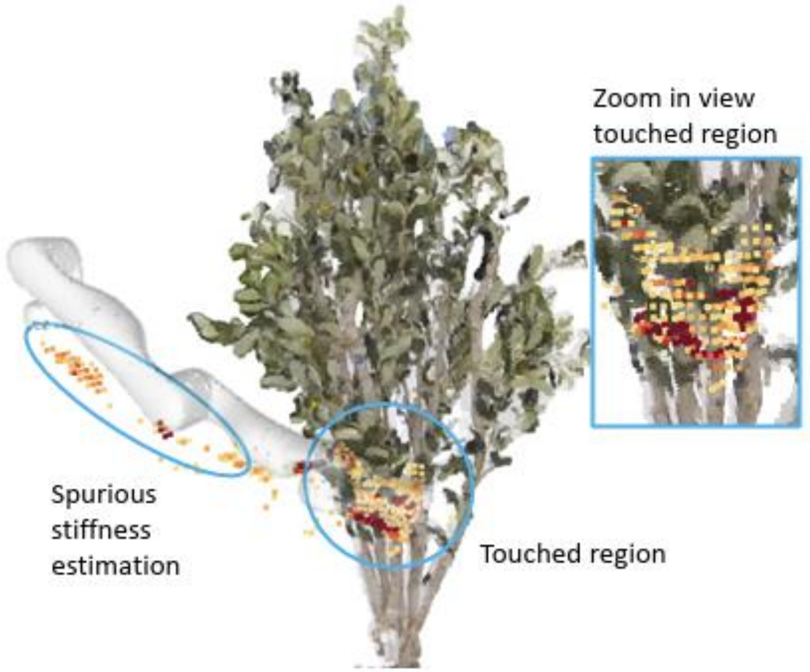
- Soft bubble visuotactile sensor with built-in depth sensing
- High-resolution contact geometry for robust manipulation

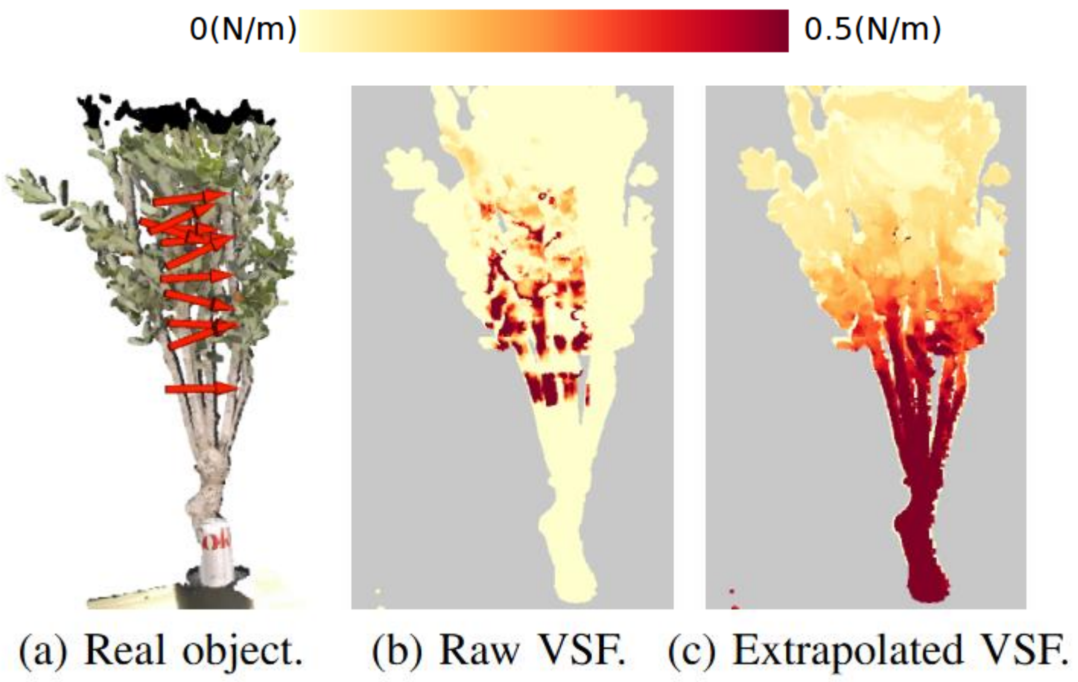


Soft bubble sensor, Alspach et al., RoboSoft, 2019

Volumetric Stiffness Field (VSF)

- Tactile interaction enables spatially-varying stiffness estimation from Punyo sensing
- Extrapolation yields full volumetric stiffness field from sparse touch frames

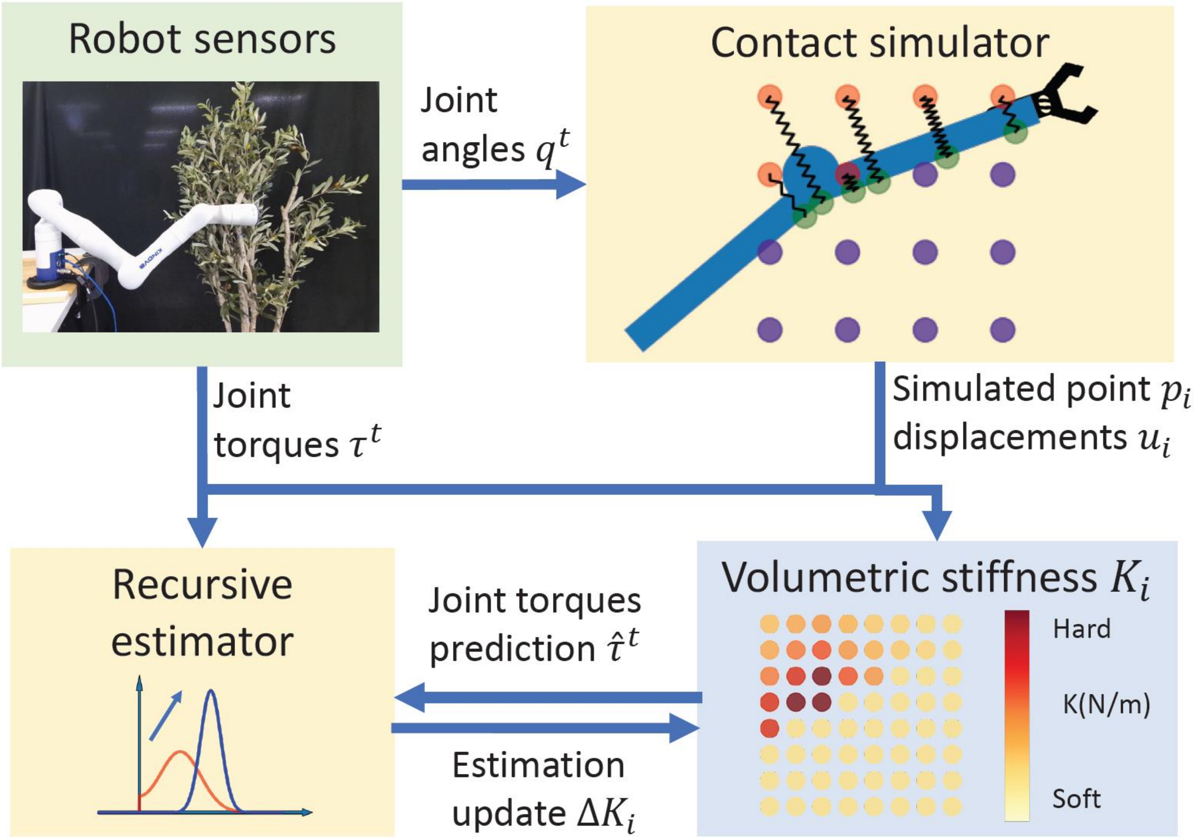




VSF results, Yao, et al., ICRA, 2023

Point-Based VSF Estimation

- Combine sensing and contact simulation to estimate stiffness
- Recursive update fuses evidence over time to refine the VSF



VSF estimation, Yao, et al., ICRA, 2023

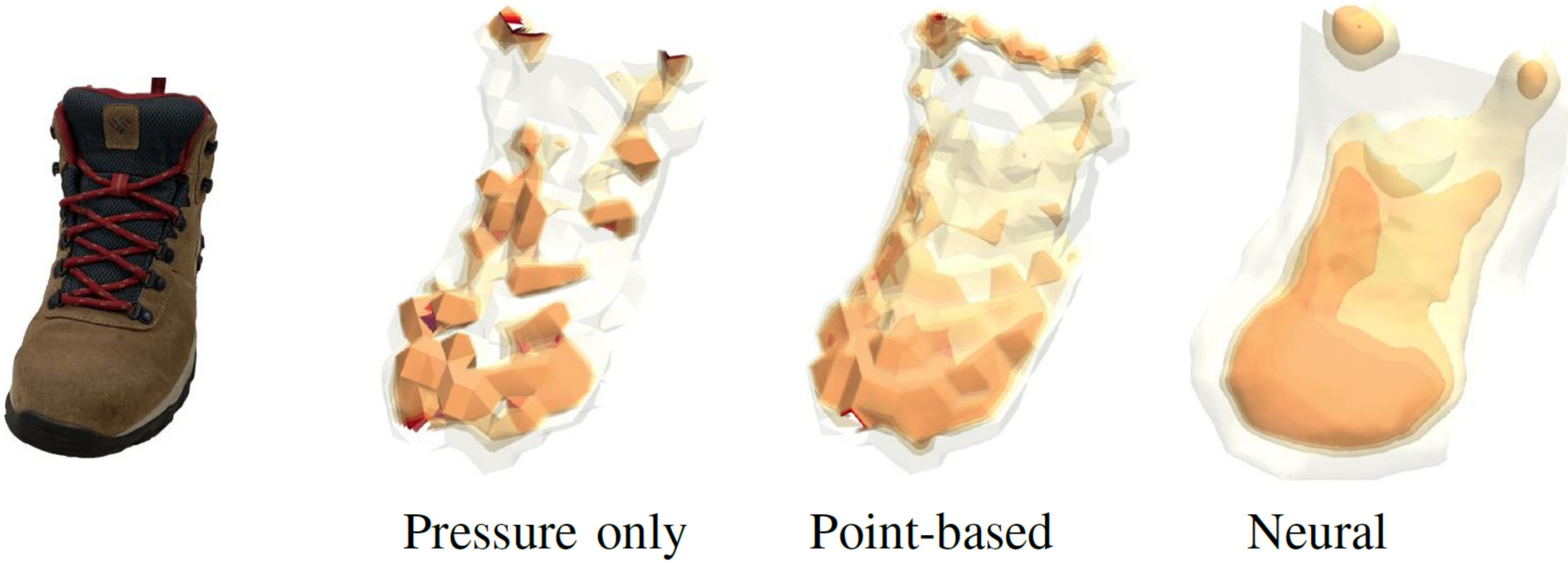
Neural VSF

- Tactile exploration collects sparse force interaction data
- Neural field interpolates in a smooth continuous VSF

Neural VSF optimization, Han, et al., ICRA, 2025

Improved Estimation with Neural VSF

- Reduce artifacts and noise in stiffness maps
- Produce smoother and more consistent field estimates than pressure-only and point-based methods



Blind localization of hidden objects, Han, et al., ICRA, 2025

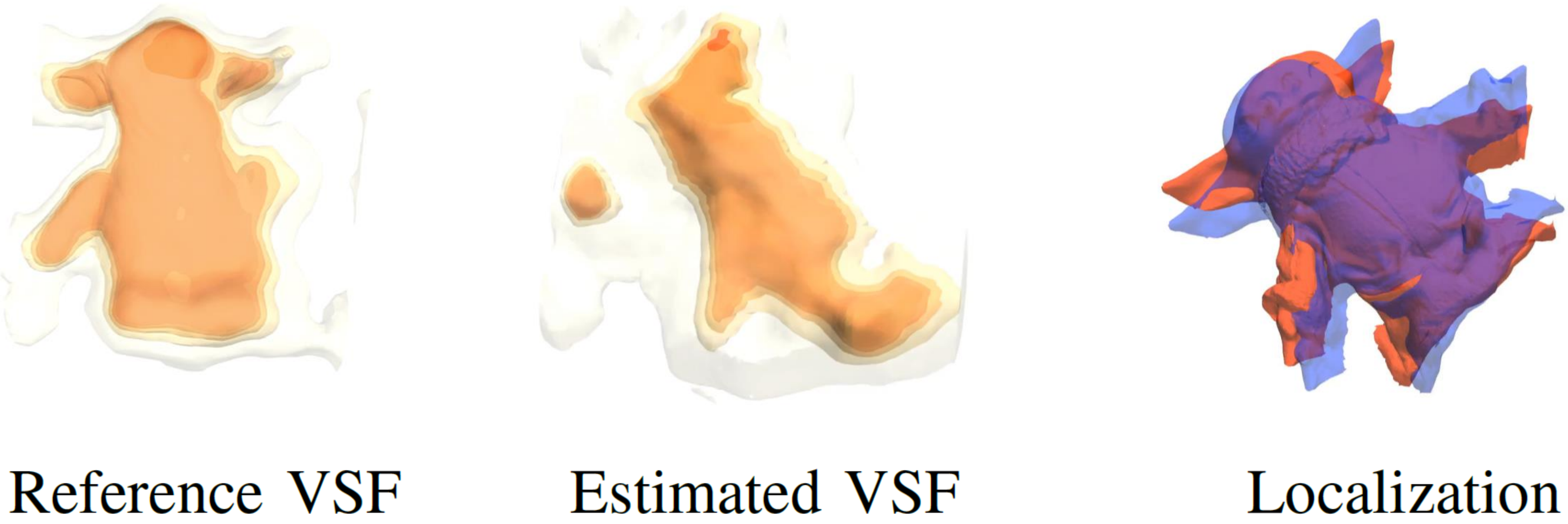
Tactile-Based Localization under Occlusion

- Occluded objects covered by deformable plastic
- Vision is blocked—robot relies on touch to infer object shape and position

Blind localization of hidden objects, Han, et al., ICRA, 2025

Estimating Object Pose via Given VSF

- Localize occluded object using only tactile stiffness field
- Match estimated VSF to reference for accurate object pose estimation



Blind localization of hidden objects, Han, et al., ICRA, 2025

Summary: Neural Fields

- Visual neural fields: photorealistic shape & appearance (NeRF, NeuS, 3DGS)
- Tactile neural fields: material-aware sensing (VSF, contact geometry)
- Combining modalities: toward robust 3D understanding and interaction for robotics

Literature Neural Fields (1)

- Irshad, Muhammad Zubair, et al. "Neural Fields in Robotics: A Survey." arXiv preprint arXiv:2410.20220 (2024).
- Xie, Tianyi, et al. "Physgaussian: Physics-integrated 3d gaussians for generative dynamics." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2024.
- Zhu, Zihan, et al. "Nicer-slam: Neural implicit scene encoding for rgb slam." 2024 International Conference on 3D Vision (3DV). IEEE, 2024.
- Mescheder, Lars, et al. "Occupancy networks: Learning 3d reconstruction in function space." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019.
- Tang, Jiapeng, et al. "Sa-convonet: Sign-agnostic optimization of convolutional occupancy networks." Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021.

Literature Neural Fields (2)

- Park, Jeong Joon, et al. "Deepsdf: Learning continuous signed distance functions for shape representation." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019.
- Mildenhall, Ben, et al. "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis." European Conference on Computer Vision. 2020.
- Wang, Peng, et al. "NeuS: learning neural implicit surfaces by volume rendering for multi-view reconstruction." Proceedings of the 35th International Conference on Neural Information Processing Systems. 2021.
- Chen, Guikun, and Wenguan Wang. "A survey on 3d gaussian splatting." arXiv preprint arXiv:2401.03890 (2024).

Literature Neural Fields (3)

- Kerbl, Bernhard, et al. "3d gaussian splatting for real-time radiance field rendering." ACM Trans. Graph. 42.4 (2023): 139-1.
- Huang, Binbin, et al. "2d gaussian splatting for geometrically accurate radiance fields." ACM SIGGRAPH conference papers. 2024.
- Alspach, Alex, et al. "Soft-bubble: A highly compliant dense geometry tactile sensor for robot manipulation." 2nd IEEE International Conference on Soft Robotics (RoboSoft). 2019.
- Yao, Shaoxiong, and Kris Hauser. "Estimating tactile models of heterogeneous deformable objects in real time." IEEE International Conference on Robotics and Automation (ICRA). 2023.
- Han, Jiaheng, et al. "Estimating High-Resolution Neural Stiffness Fields using Visuotactile Sensors." IEEE International Conference on Robotics and Automation (ICRA). 2025.