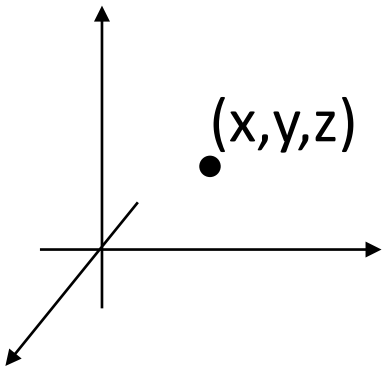
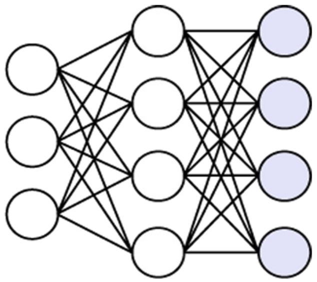
Lecture: Neural Fields 3

What are neural fields?



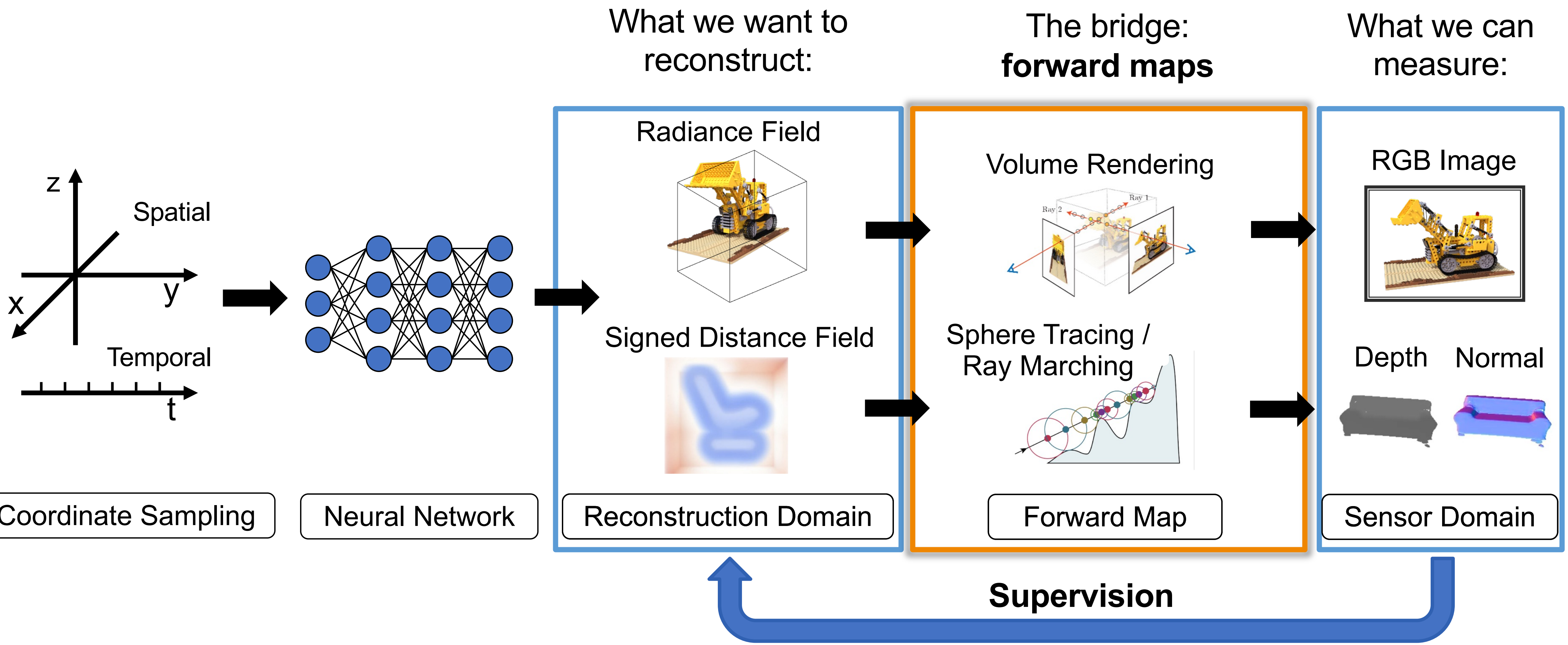
$\Phi: \mathbb{R}^n \rightarrow \mathbb{R}$



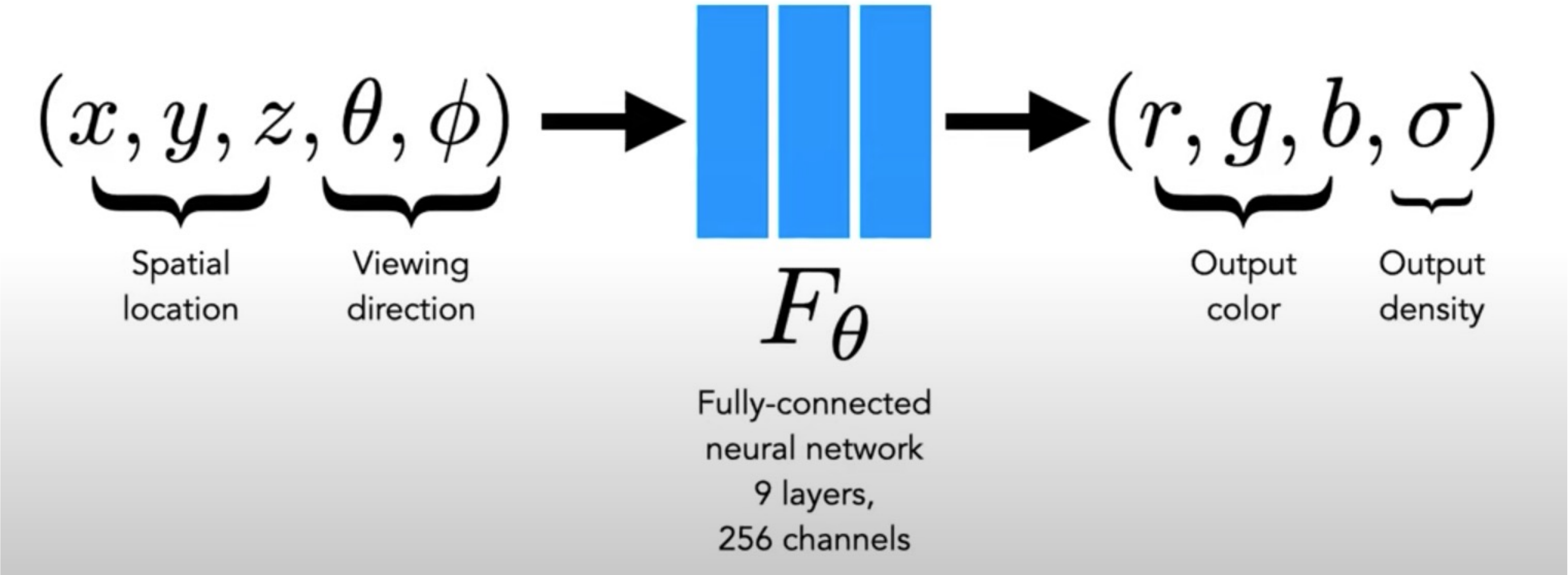
Neural Network
(Φ)

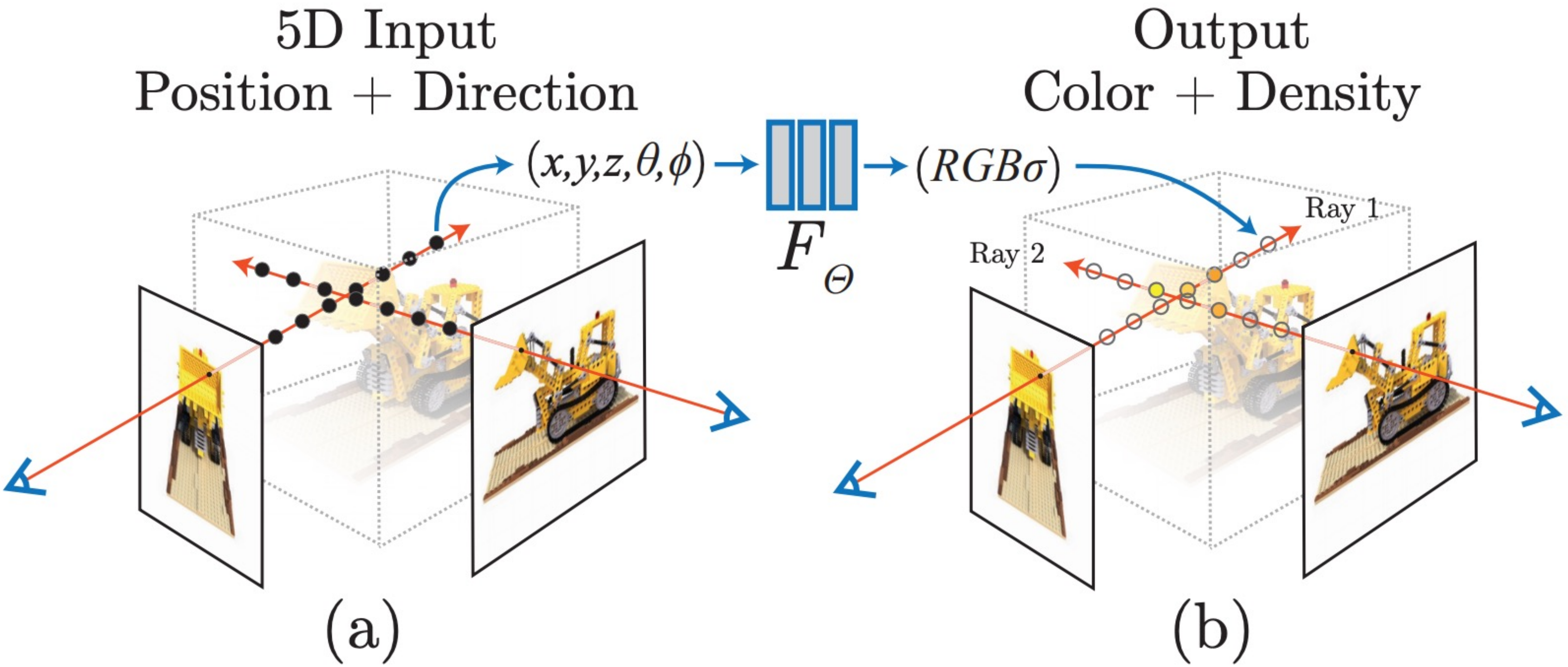
Signed Distance Function (SDF)

Neural Field General Framework



What do we learn in NeRF?





Mildenhall et al. 2020

Volume rendering estimation: integrating color along a ray

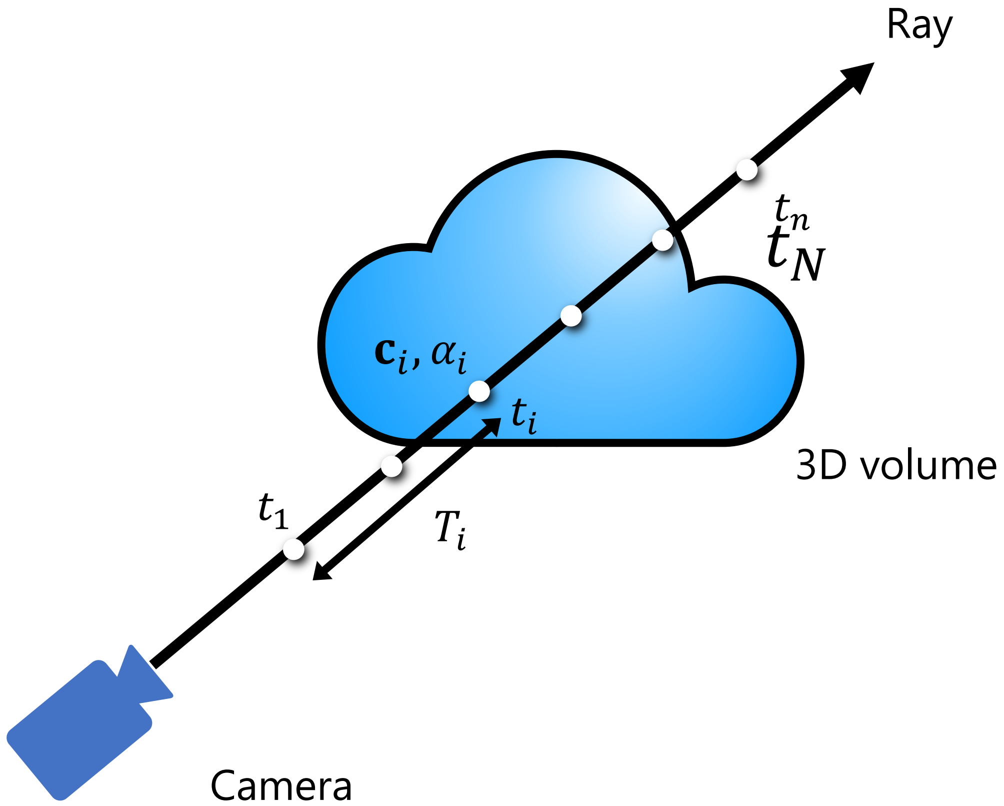
Rendering model for ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$:

$$\mathbf{c} \approx \sum_{i=1}^{n} T_i \alpha_i \mathbf{c}_i$$

final rendered color along ray

weights

colors



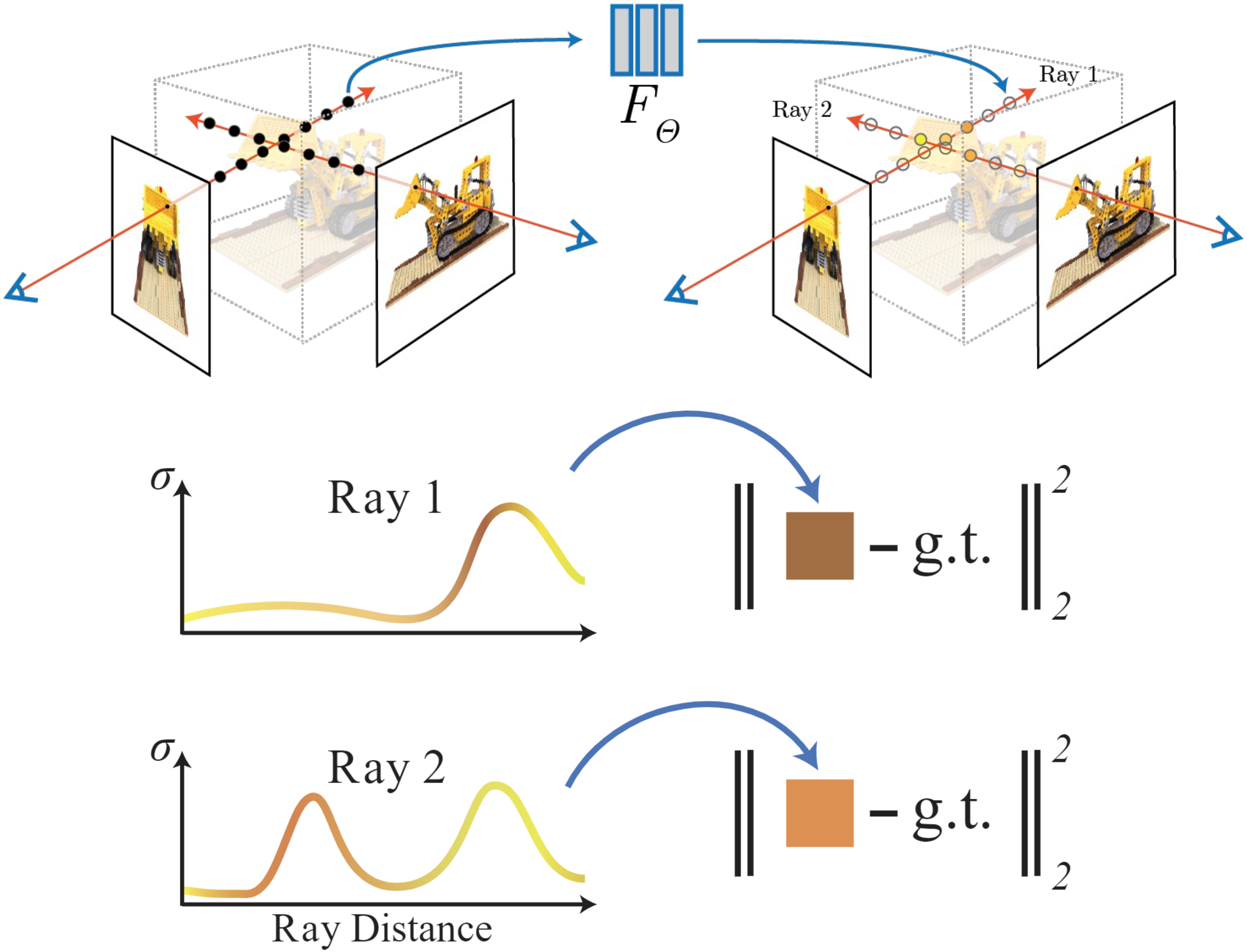
How much light is blocked earlier along ray:

$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)$$

Computing the color for a set of rays through the pixels of an image yields a rendered image

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

Slight modification: α is not directly stored in the volume, but instead is derived from a stored volume density sigma (σ) that is multiplied by the distance between samples delta (δ):



F_Θ
Ray 1
Ray 2
σ
Ray 1
$\| \square - \text{g.t.} \|_2^2$
σ
Ray 2
$\| \square - \text{g.t.} \|_2^2$
Ray Distance

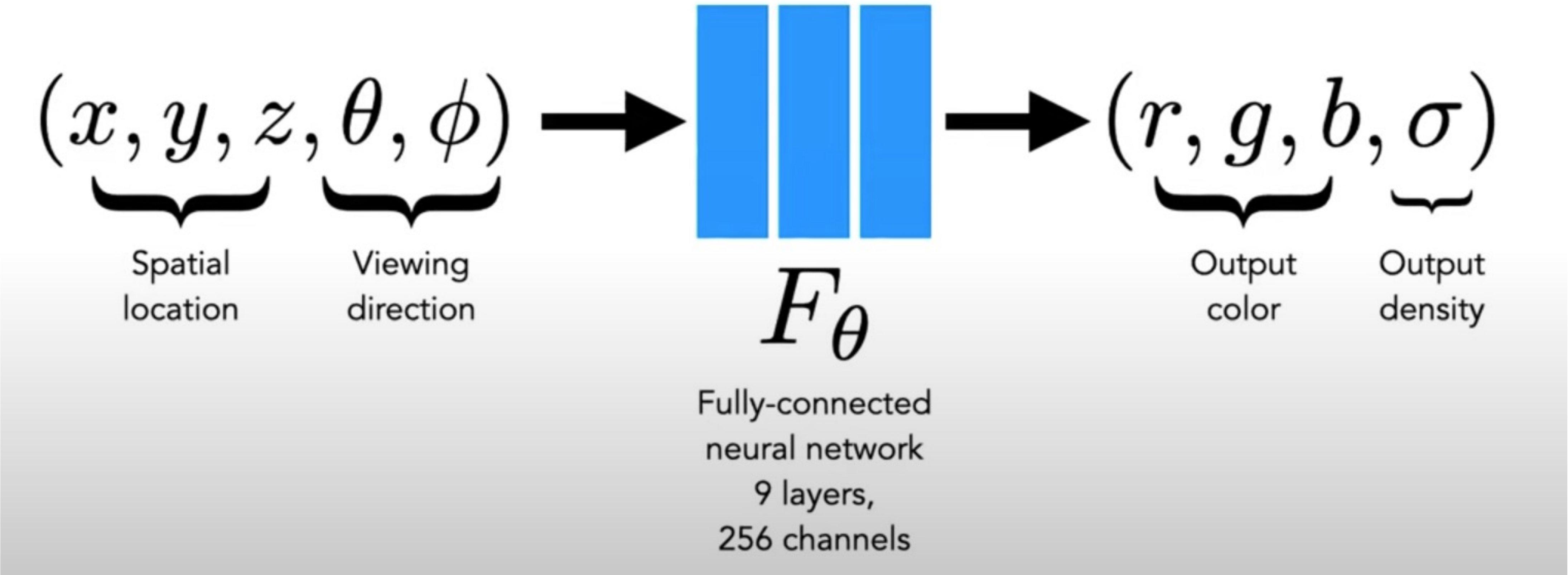
Outline

- Network Architecture
- Hybrid Representation
- Generalization
- Editing/Manipulation

Outline

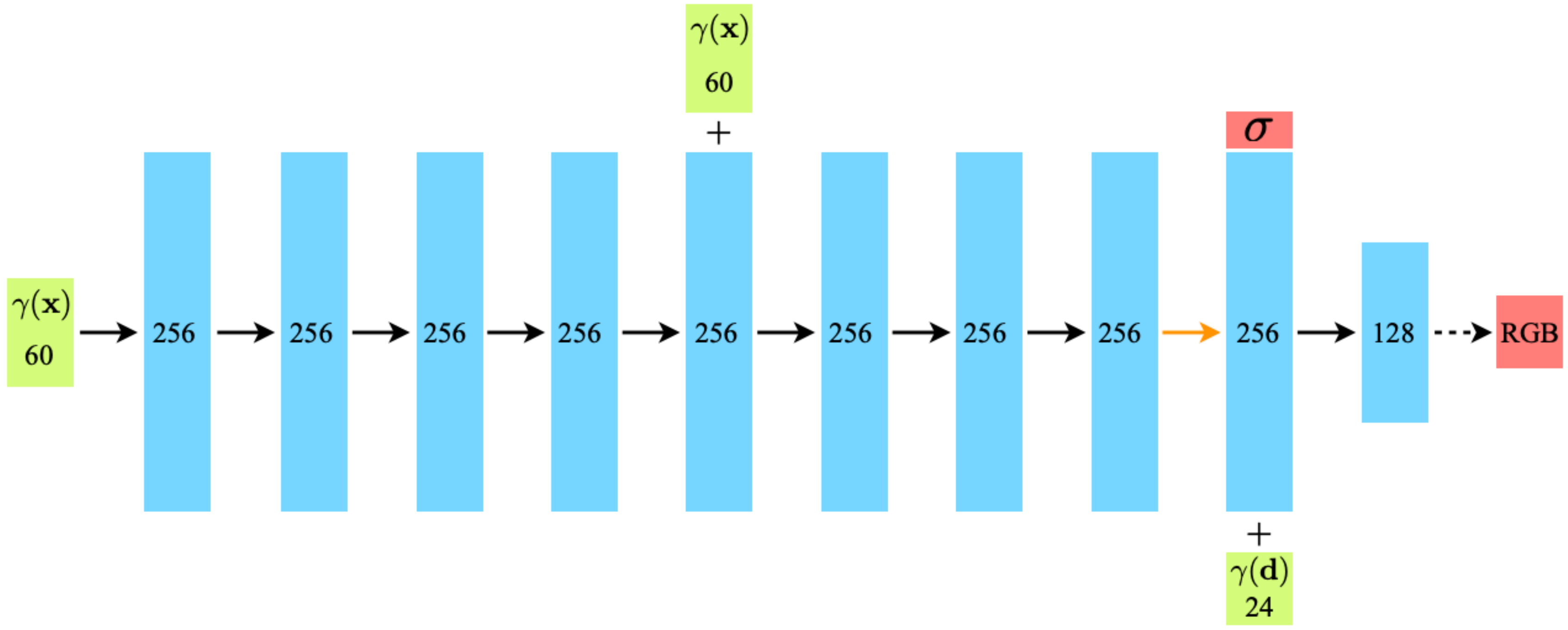
- Network Architecture
- Hybrid Representation
- Generalization
- Editing/Manipulation

What do we learn in NeRF?



DeepSDF Extensions: NeRF

- Coordinate-based modeling of RGB and Densities Instead of SDFs



Mildenhall et al. 2020

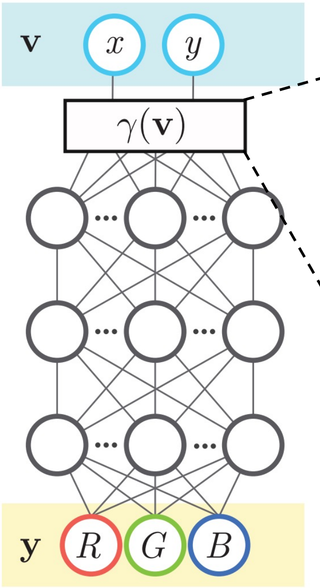
Network Architecture: Overcoming Spectral Bias

[Baatz et al. 2021]

The signals we want are high frequency!

Network Architecture: Input Encodings

Positional Encodings



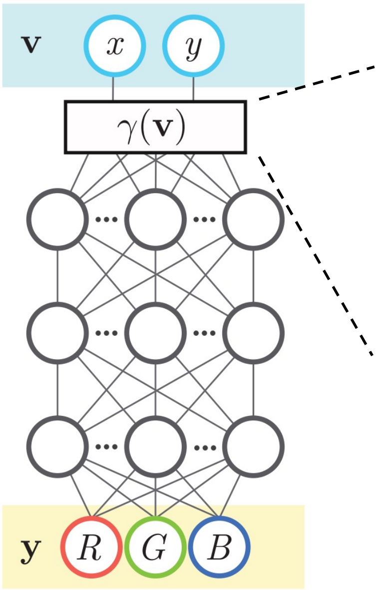
$$\gamma(\mathbf{x}) = [\gamma_1(\mathbf{x}), \gamma_2(\mathbf{x}), \ldots, \gamma_m(\mathbf{x})]$$

$$\gamma_{(2i)}(x) = sin(2^{i-1}\pi x),$$

$$\gamma_{(2i+1)}(x) = cos(2^{i-1}\pi x)$$

[Vaswani et al. 2017]

Network Architecture: Input Encodings



Random Fourier Encodings

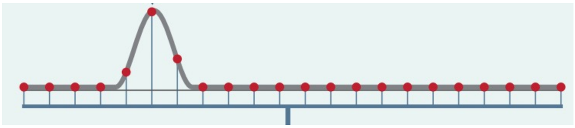
[Tancik et al. 2020]

$$\gamma(\mathbf{v}) = [\cos(2\pi\mathbf{B}\mathbf{v}), \sin(2\pi\mathbf{B}\mathbf{v})]^{\mathrm{T}}$$

Non-axis aligned sine embeddings

One-blob Encodings

[Müller et al. 2020]



Super Gaussian Encodings

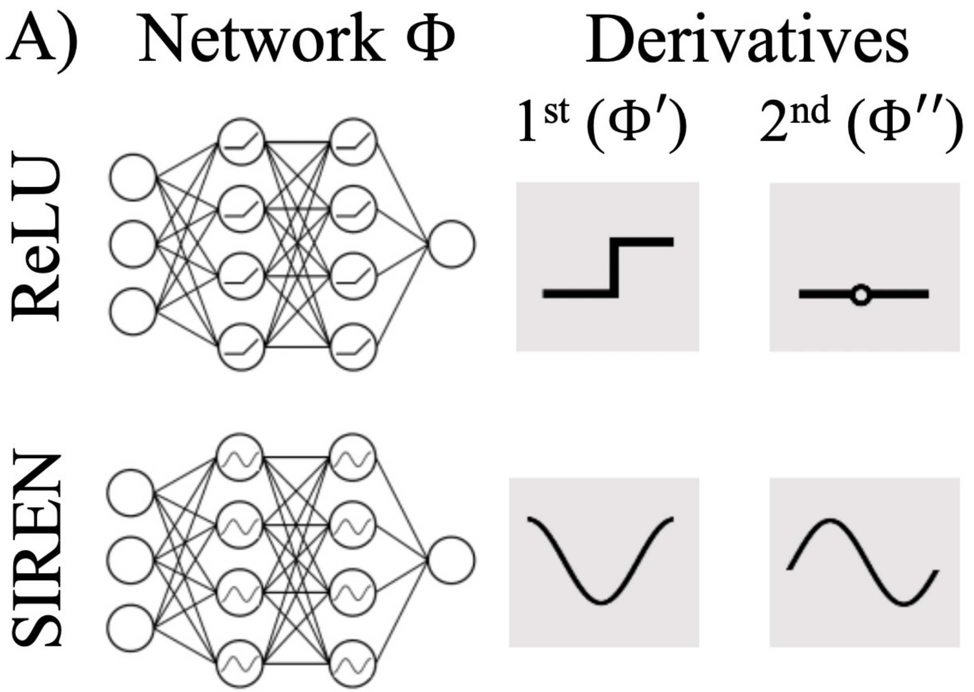
[Ramasinghe et al. 2021]

$$\Phi(\boldsymbol{x}) = [\phi_1(\boldsymbol{x}), \phi_2(\boldsymbol{x}), \ldots, \phi_D(\boldsymbol{x})]^T,$$

$$\left[e^{-\frac{(\boldsymbol{x}\cdot\alpha - t_i)^2}{2\sigma_{\boldsymbol{x}}^2}}\right]^b$$

Gaussian embeddings

Network Architecture: Activation Functions



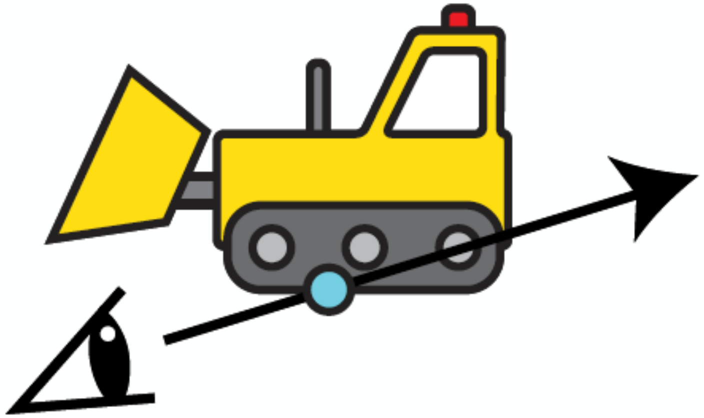
[Sitzmann et al. 2021]

Gaussian	$e^{\frac{-0.5x^2}{a^2}}$	✓
Quadratic	$\frac{1}{1+(ax)^2}$	✓
Multi Quadratic	$\frac{1}{\sqrt{1+(ax)^2}}$	✓
Laplacian	$e^{\left(\frac{-\lvert x\rvert}{a}\right)}$	✓
Super-Gaussian	$[e^{\frac{-0.5x^2}{a^2}}]^b$	✓
ExpSin	$e^{-\sin(ax)}$	✓

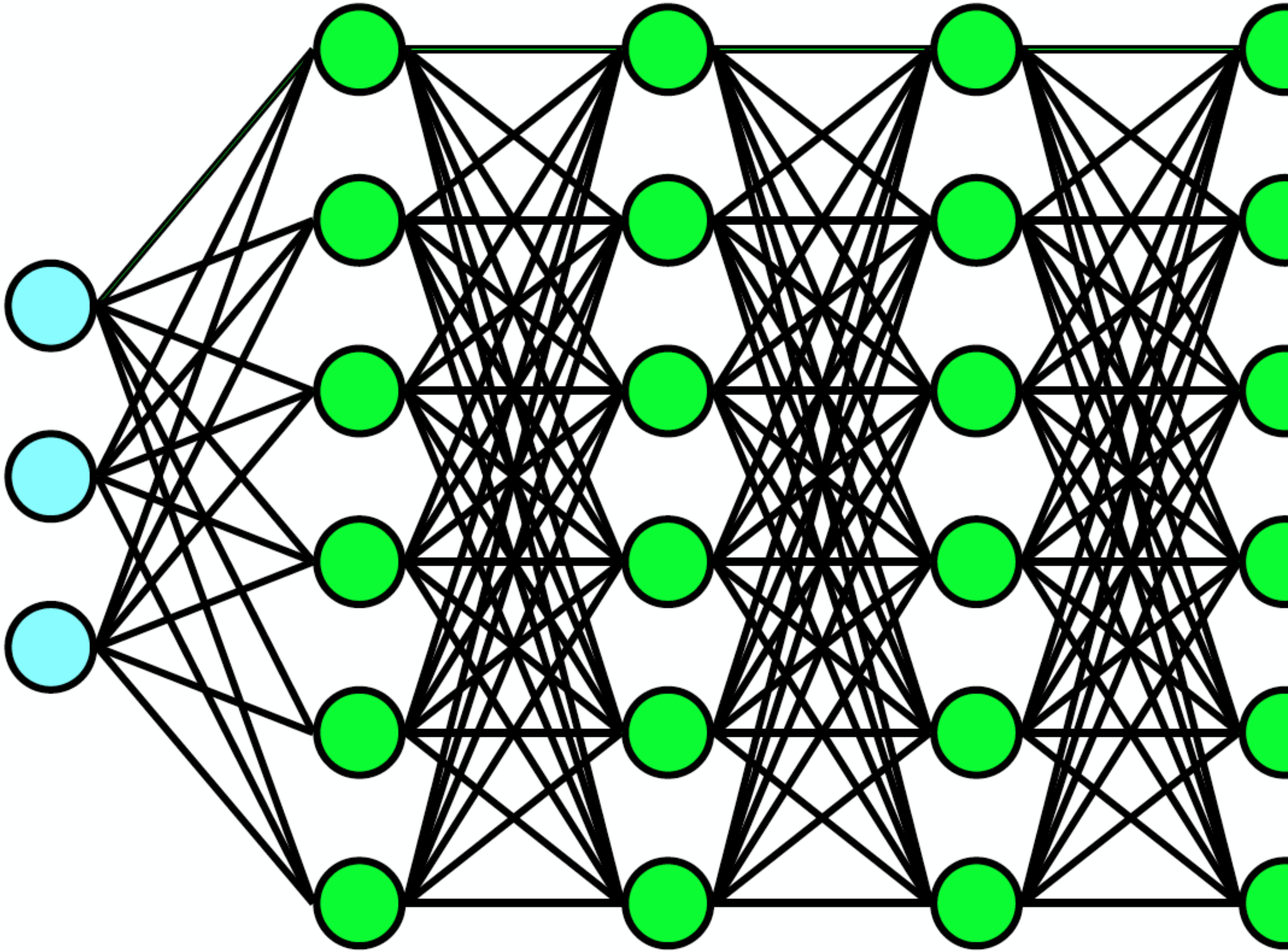
[Ramasinghe et al. 2021]

Outline

- Network Architecture
- Hybrid Representation
- Generalization
- Editing/Manipulation

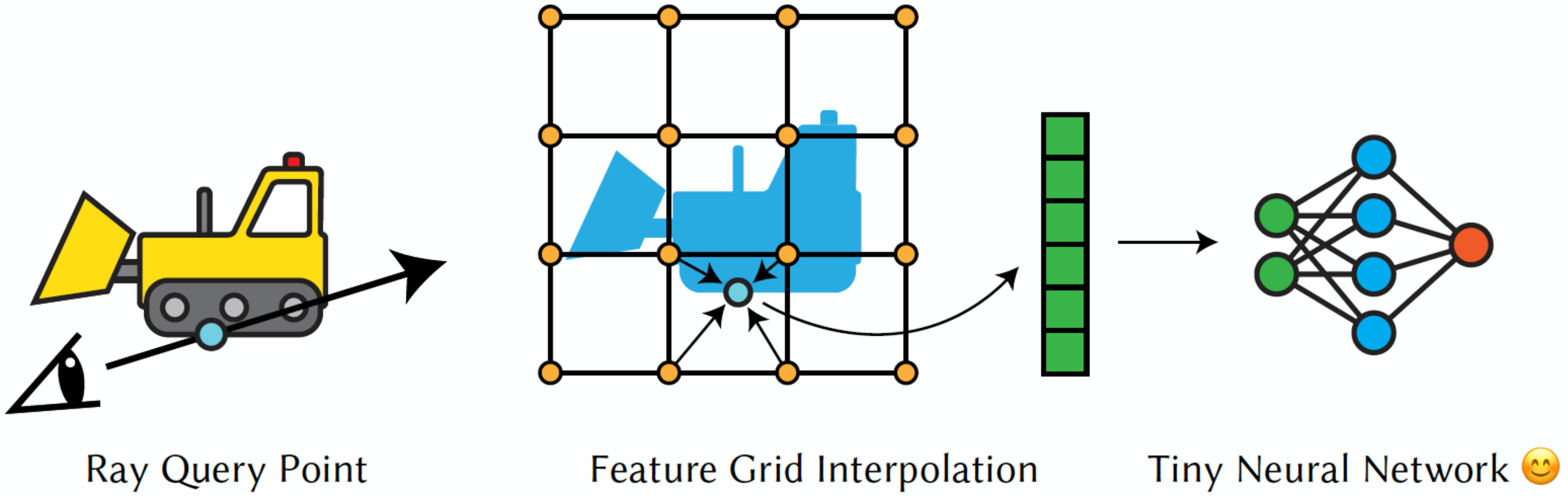


Ray Query Point



Huge Neural Network ☹

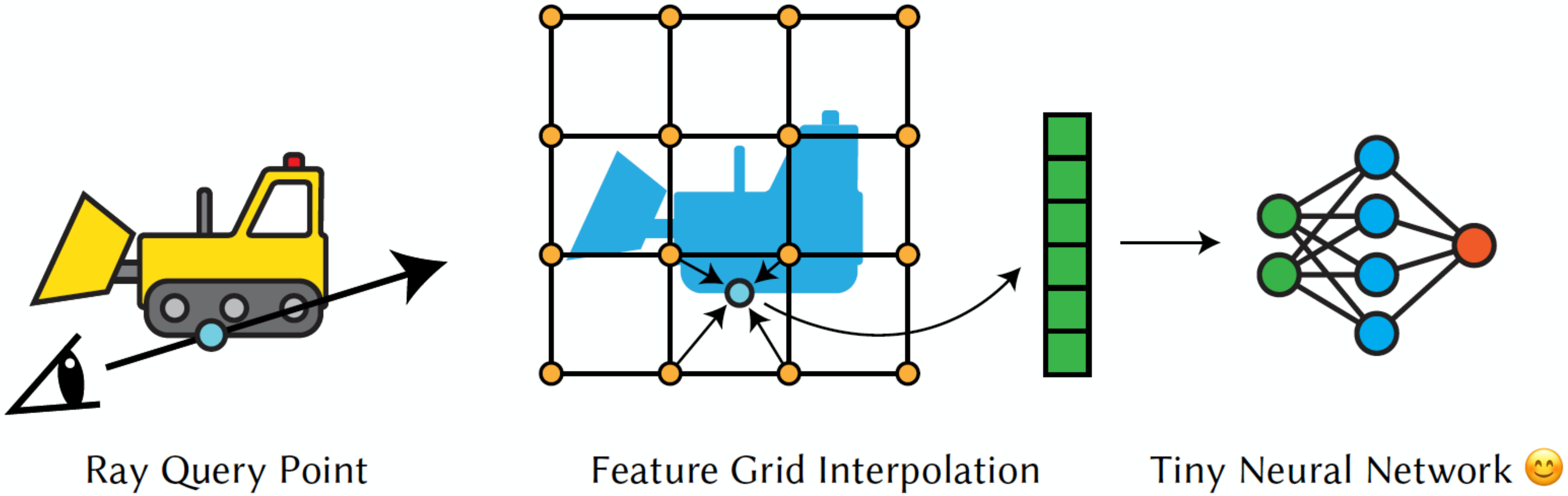
Hybrid representation



Features:

- are also parameters that can be updated while training the NeRF. (slight increase in memory, significantly faster training & inference)
- are individual NeRFs trained on a small section of a scene (for large city-size scene)
- are priors obtained from ConvNets, e.g. VGG-features (used for generalization)

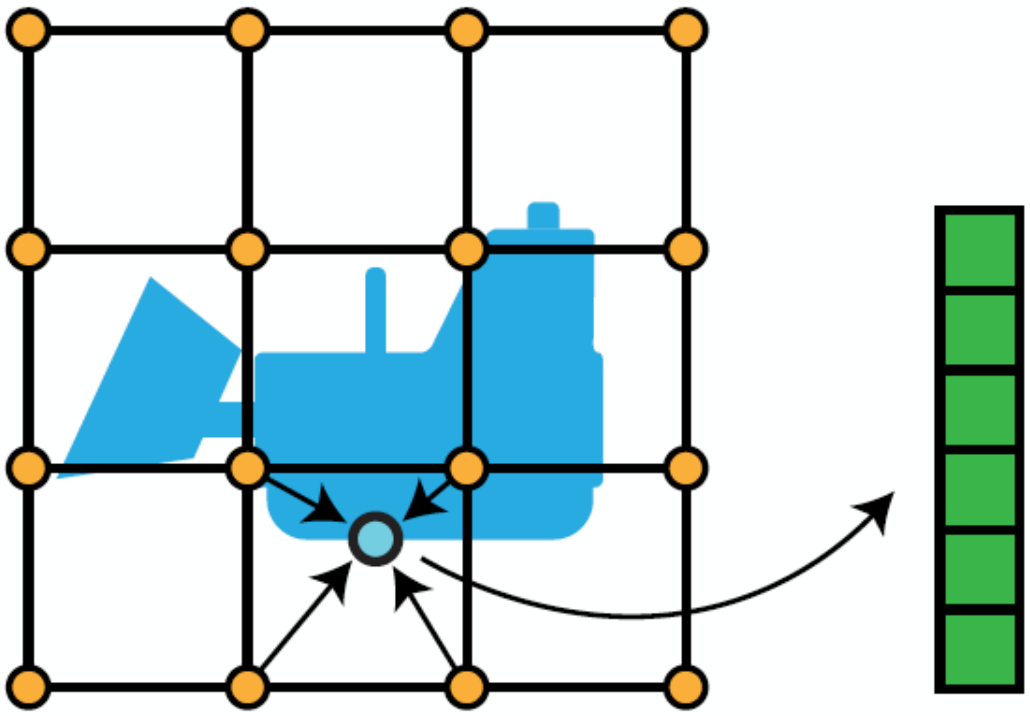
Hybrid representation: It’s all about Data Structures!



Why hybrid representation?

- Reduce the size of neural network -> fast inference & rendering.
- Helps in rendering large scale scenes.
- Helps in generalization.

Uniform Grids



[PIFu (Saito et al.), Neural Volumes (Lombardi et al.), etc]

Pros:

- Easy to implement
- Algorithmically fast access [O(1)]
- Established operations like convolutions
- Simple topology

Cons:

- Expensive in memory and bandwidth
- Limited by Nyquist

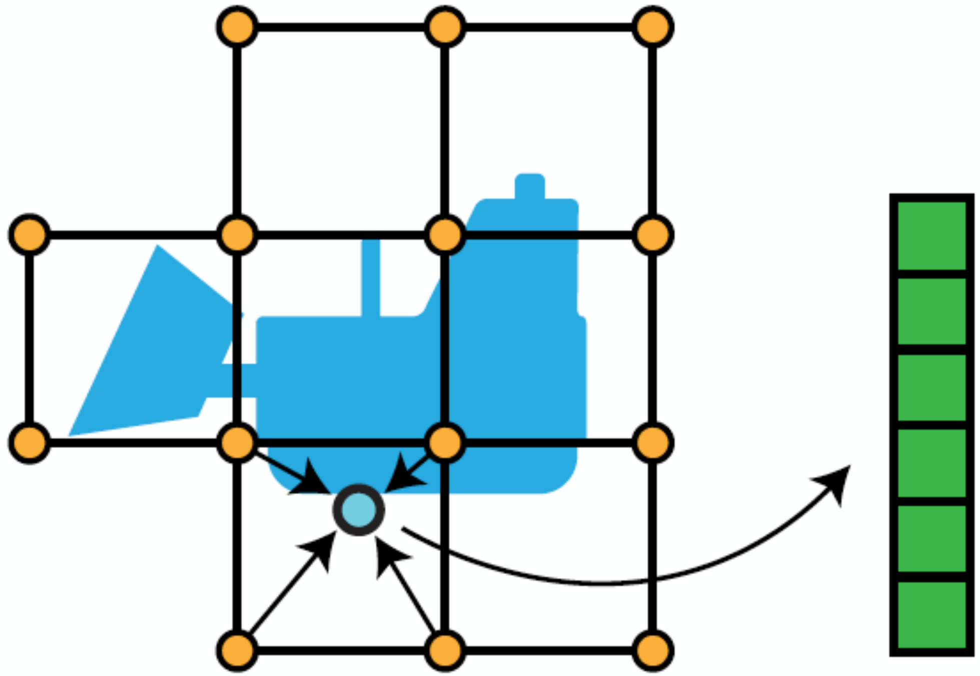
BlockNeRF (Tanick et al) – CVPR 2022

[Tancik et al.]

Train a small NeRF for each block in a city. These NeRFs are the ‘features’ in hybrid representation.

BlockNeRF (Tanick et al) – CVPR 2022

Sparse Grid



[DeepLS (Chabra et al.), NSVF (Liu et al.), NGLOD (Takikawa et al.), etc]

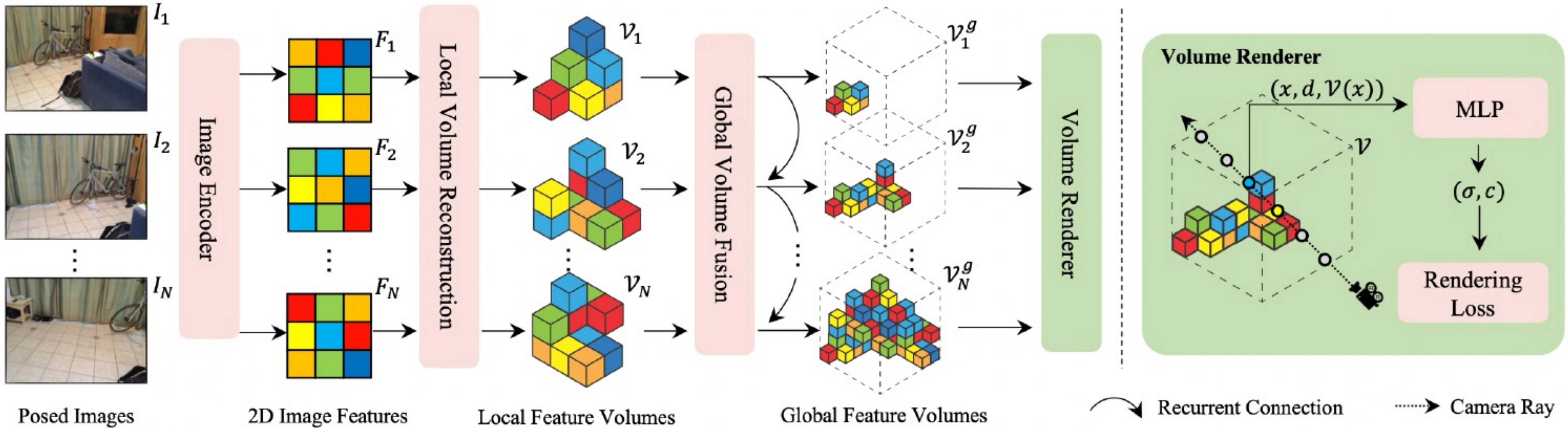
Pros:

- Memory Efficient
- Algorithmically efficient access [O(log(n))]
- GPU-compatible data structures
- Established operations like sparse 3D convs

Cons:

- Need to manage a complex data structure
- Topology hard to generate
- Still limited by Nyquist
- Sparse support region

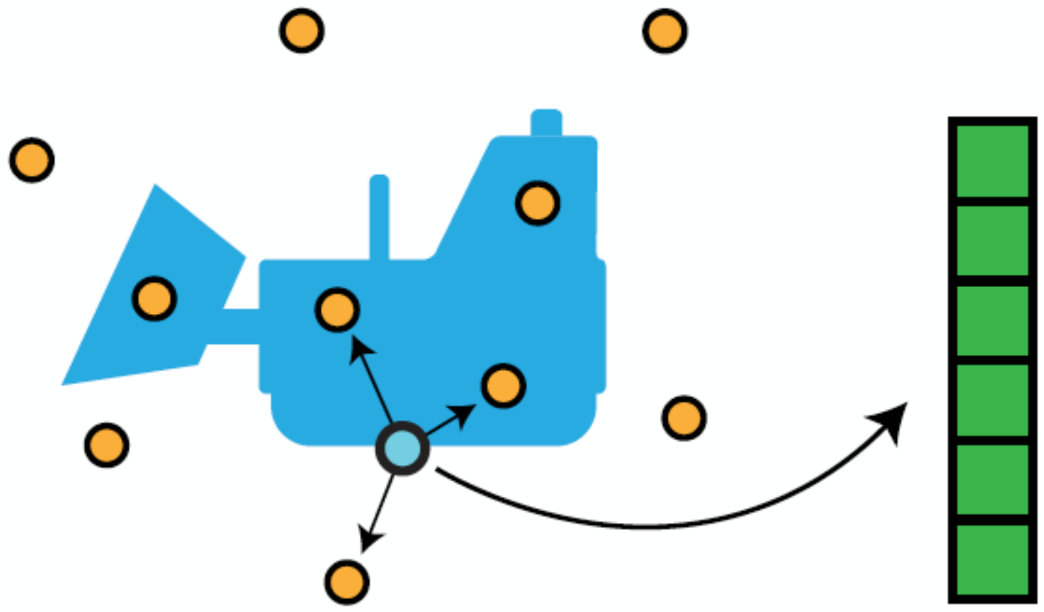
NeRFusion (Zhang et al) – CVPR 2022



[Zhang et al.]

Features = ConvNet features (from Image Encoder)

Point Clouds (Irregular Grids)



[Liu et al. 2019, LDIF (Genova et al.), 3DILG (Zhang et al.) etc]

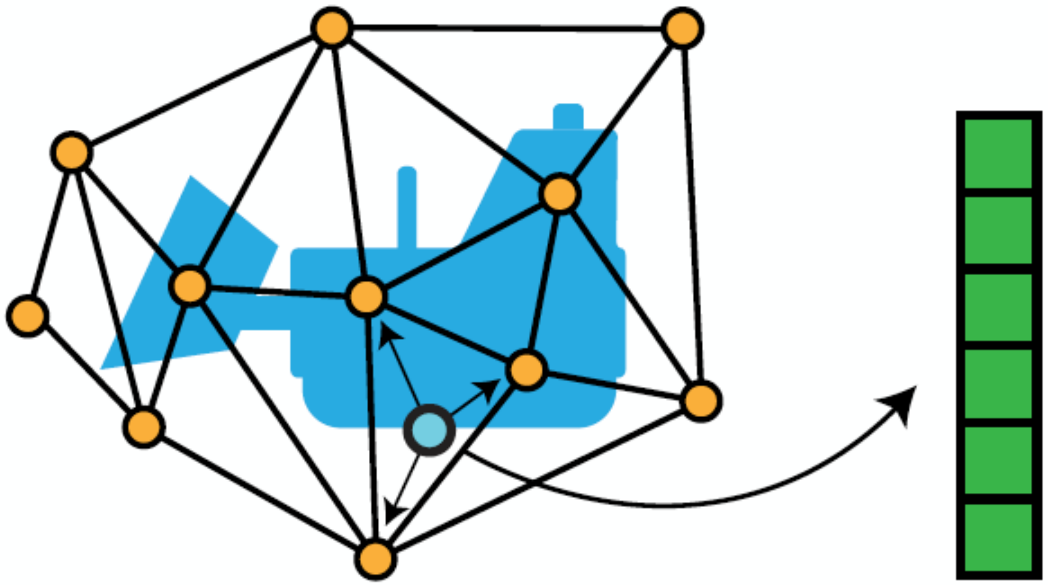
Pros:

- Not limited by Nyquist
- Can be densely supported in space
- Expressive

Cons:

- Often needs complex data structures for fast access and interpolation
- Heavily affected by choice of kernel

Mesh (Unstructured Grids)



[DefTet (Gao et al.), NeuralBody (Peng et al.), etc]

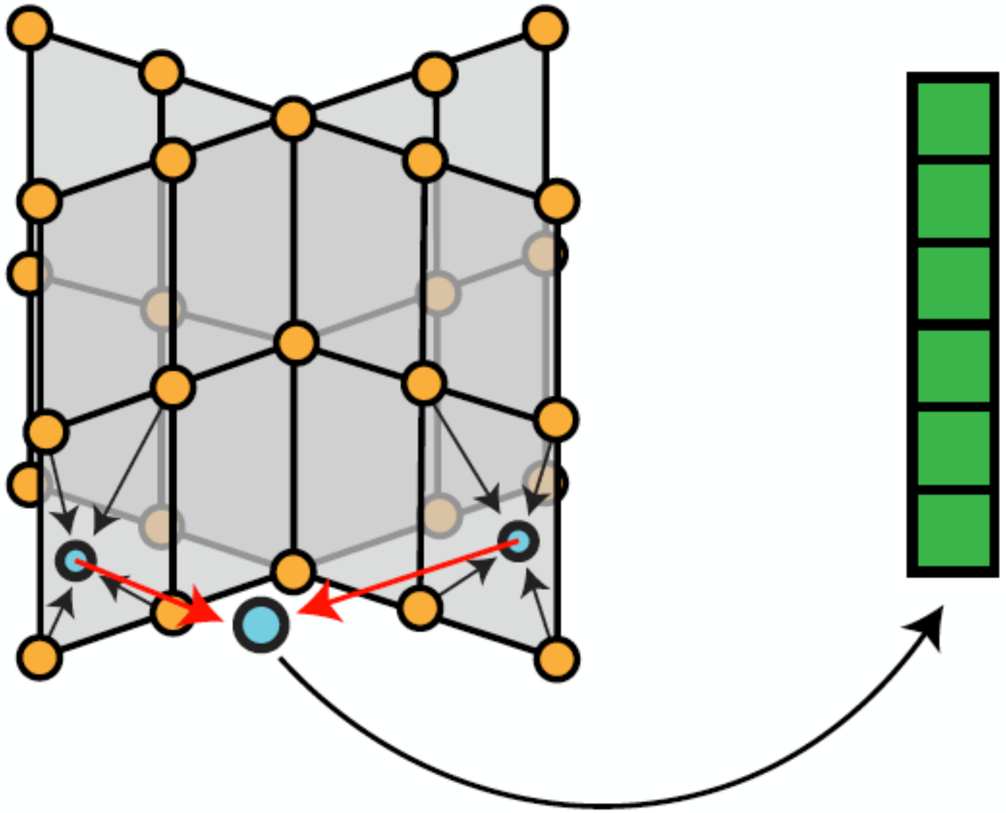
Pros:

- Not limited by Nyquist
- Can use the rich sets of tools in mesh processing

Cons:

- Is a mesh
- Non-trivial data access especially in 3D

Multiplanar Images



[Convolutional OccNet (Peng et al), EG3D (Chan et al.), etc]

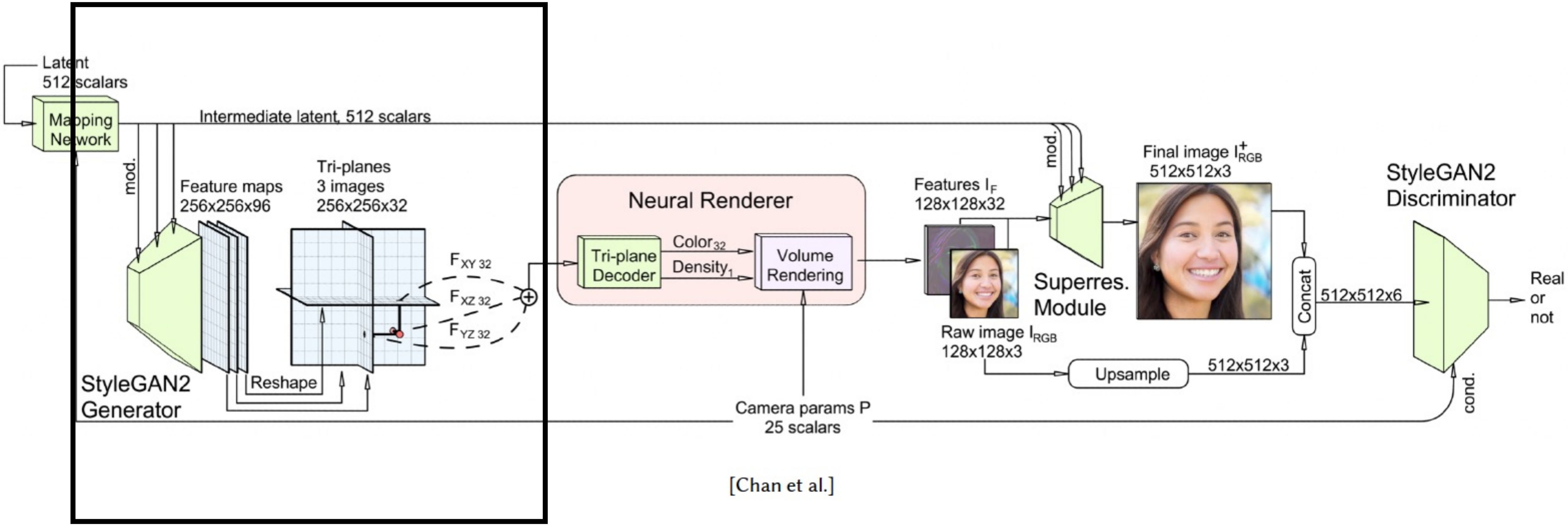
Pros:

- More compact than 3D dense grids
- Compatibility with 2D pipelines

Cons:

- Resolution bias on plane axis

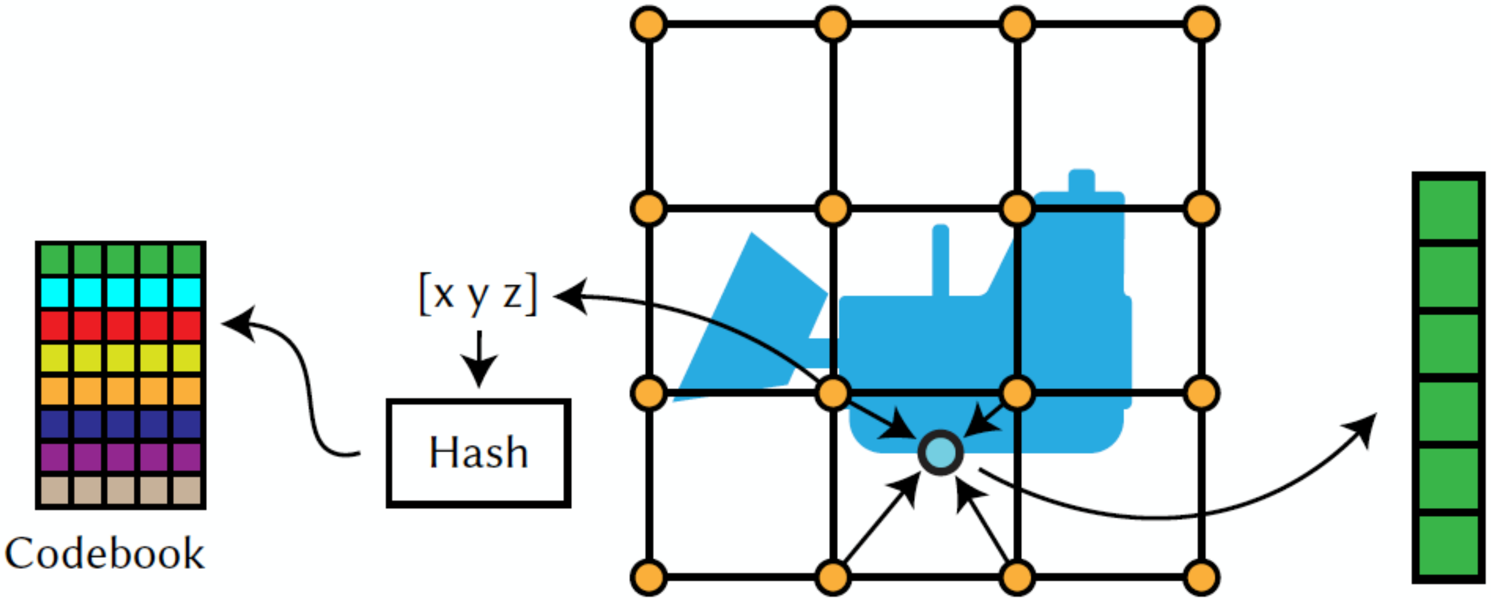
EG3D (Chan et al) – CVPR 2022



[Chan et al.]

Features = StyleGANv2 features

Hash Grids



[Instant-NGP (Muller et al.)]

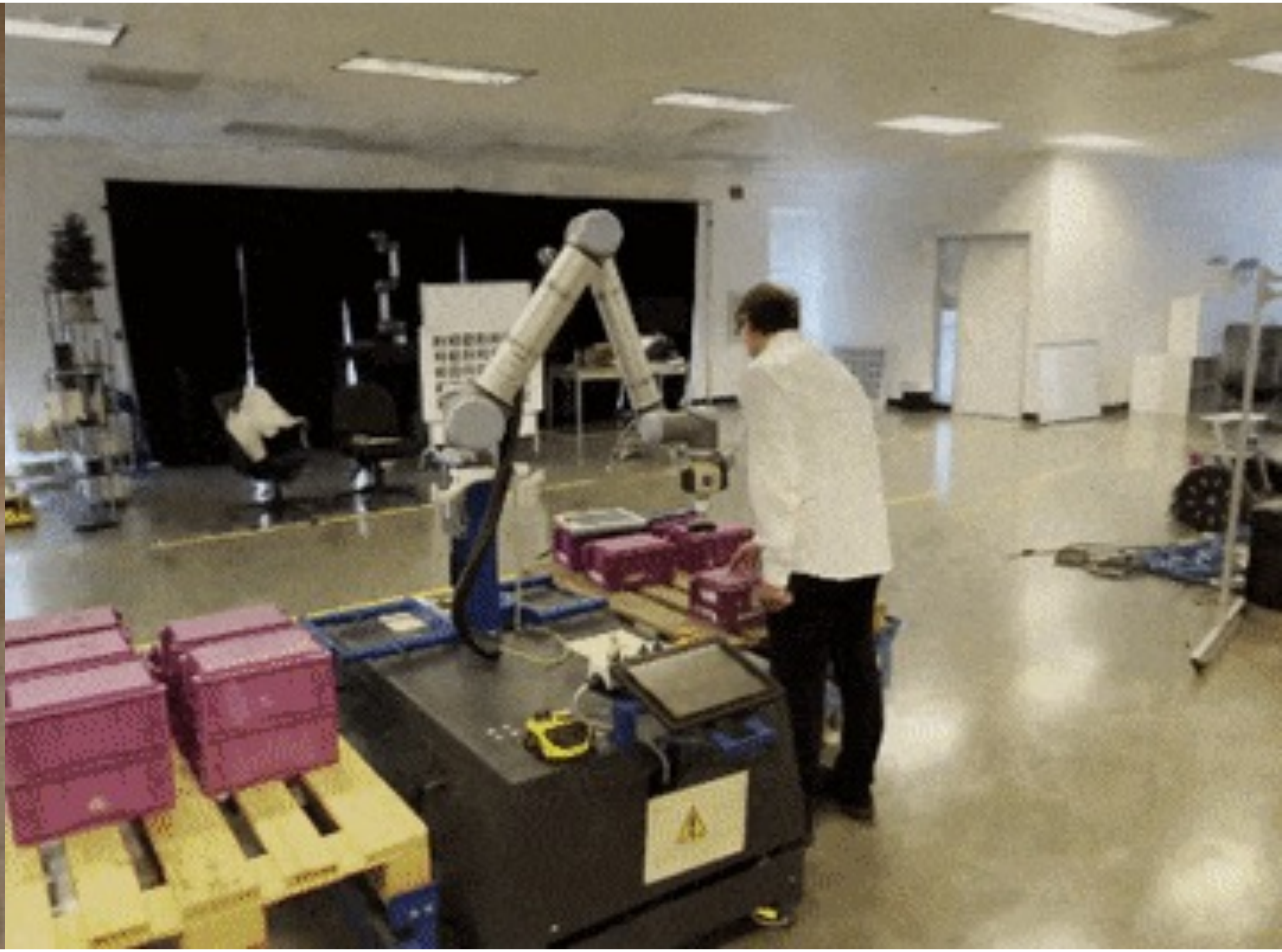
Pros:

- Densely supported
- Disaggregate resolution from memory cost
- No complex data structures
- Performant memory access if codebook is small enough

Cons:

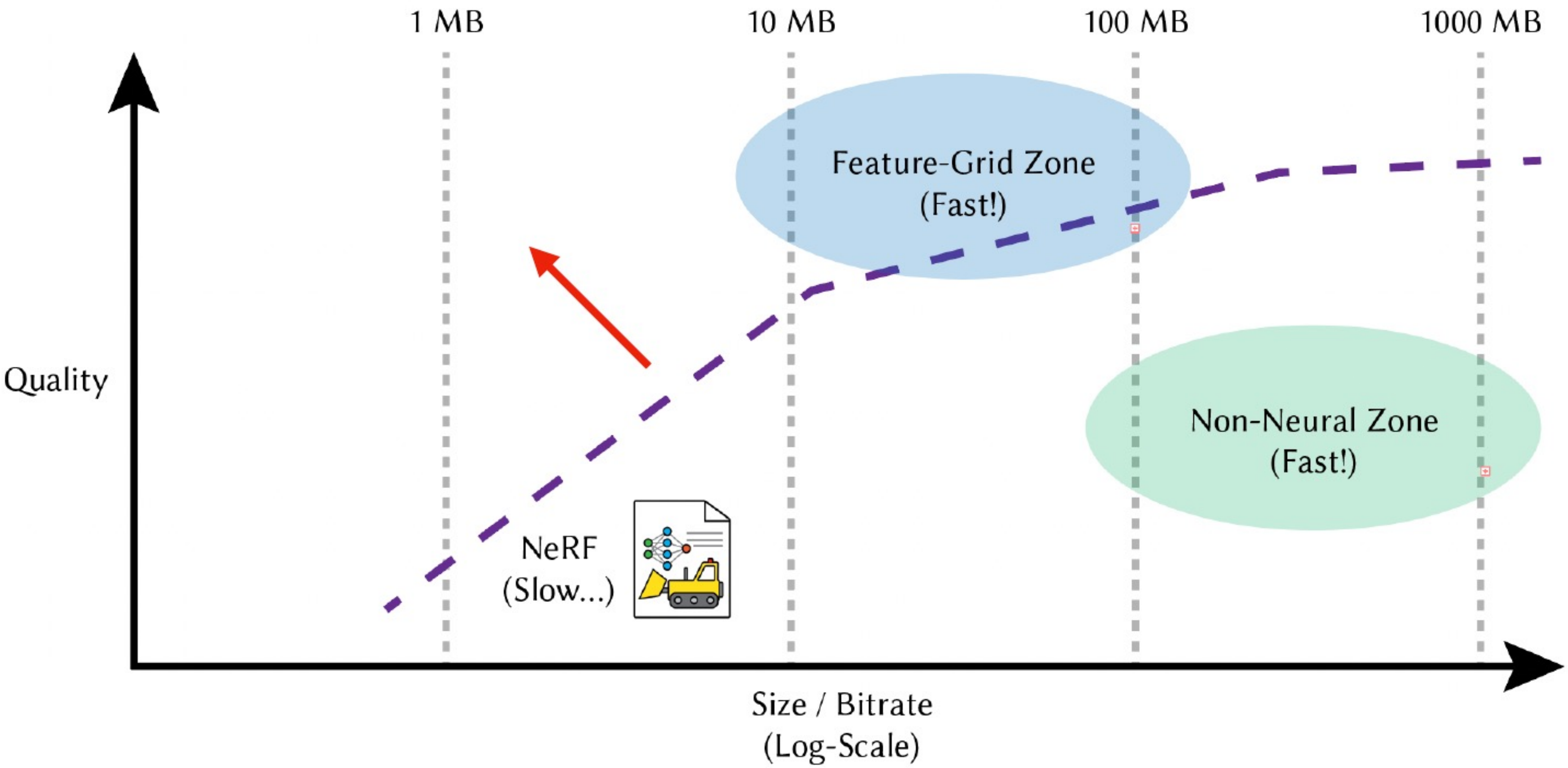
- Multiresolution and large codebooks needed for collision resolution
- Features not spatially local

Instant NGP: Lightening fast NeRF inference



Features = Trainable Parameters

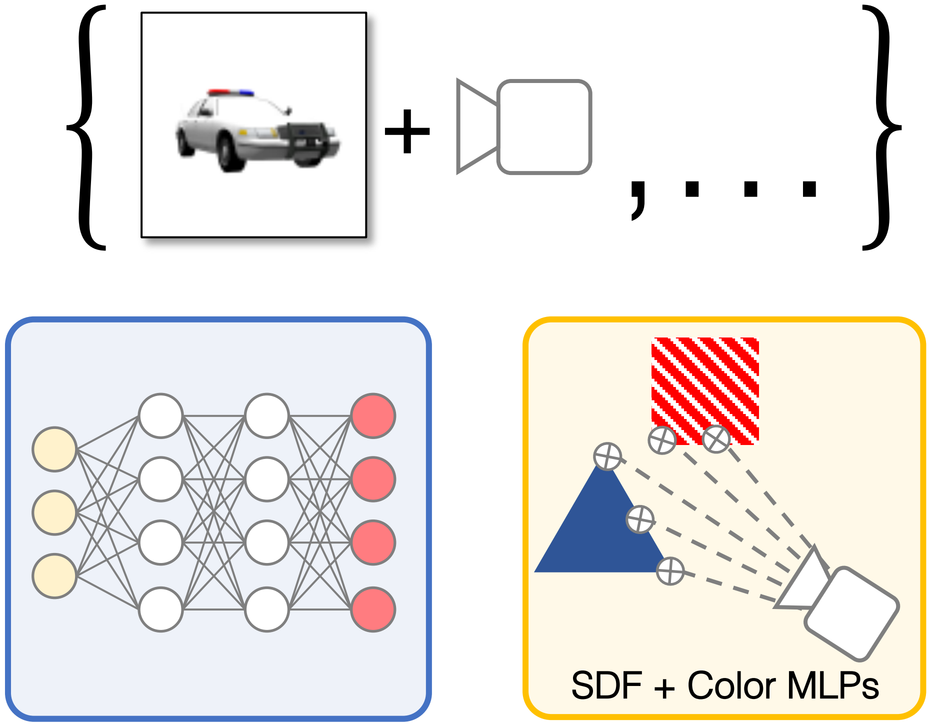
We will read this paper in details!


1 MB
10 MB
100 MB
1000 MB
Feature-Grid Zone
(Fast!)
Non-Neural Zone
(Fast!)
NeRF
(Slow...)
Quality
Size / Bitrate
(Log-Scale)

Outline

- Network Architecture
- Hybrid Representation
- Generalization
- Editing/Manipulation

Overfitting case: Inference = Fitting via Gradient Descent

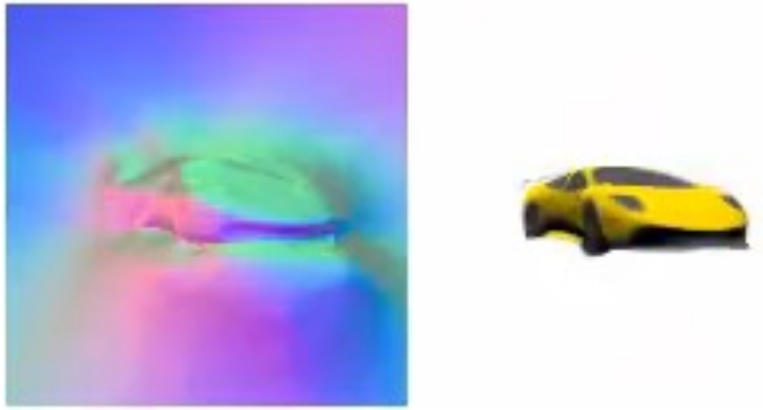


SRN_{ϕ} RENDER_{θ}

$$\min \left\| \text{RENDER}_{\theta}(\text{SRN}_{\phi}, \xi_i) - \mathcal{I}_i \right\|$$

Fitting

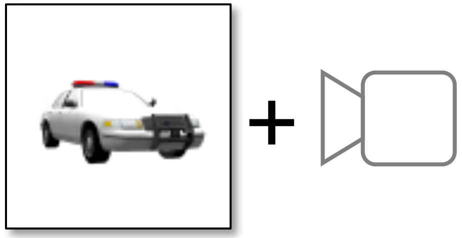
Rendering



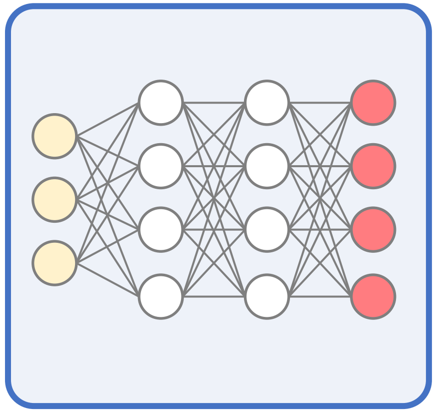
Normal map RGB

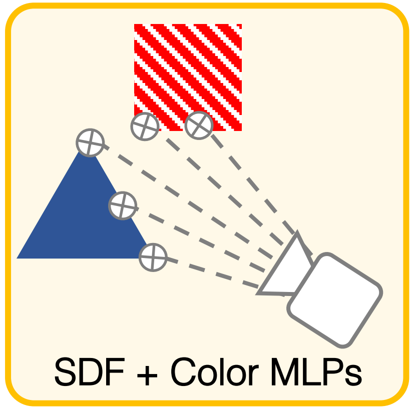
Sitzmann et al: Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations, NeurIPS 2020.

What if we have incomplete observations?



No 3D inform. $(\mathcal{J}, \xi)$





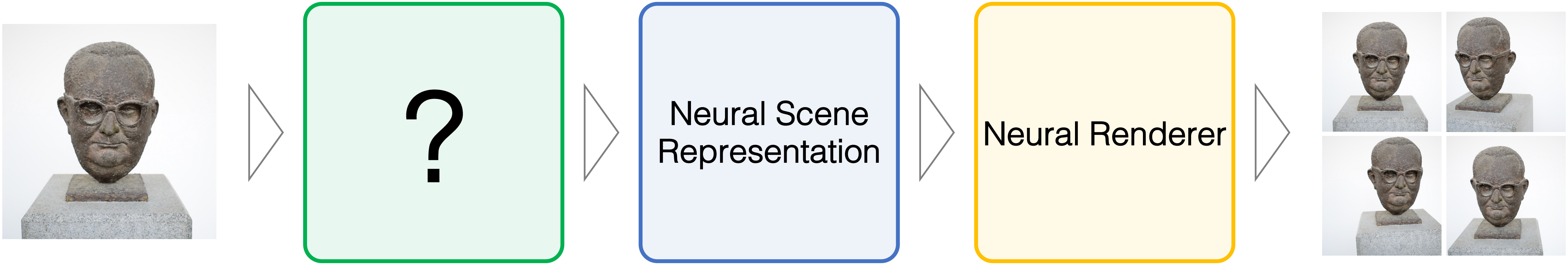
$\textsc{Srn}_{\phi}$ $\textsc{Render}_{\theta}$

$$\min \left\| \textsc{Render}_{\theta}(\textsc{Srn}_{\phi}, \xi_i) - \mathcal{J}_i \right\|$$

Normal map RGB

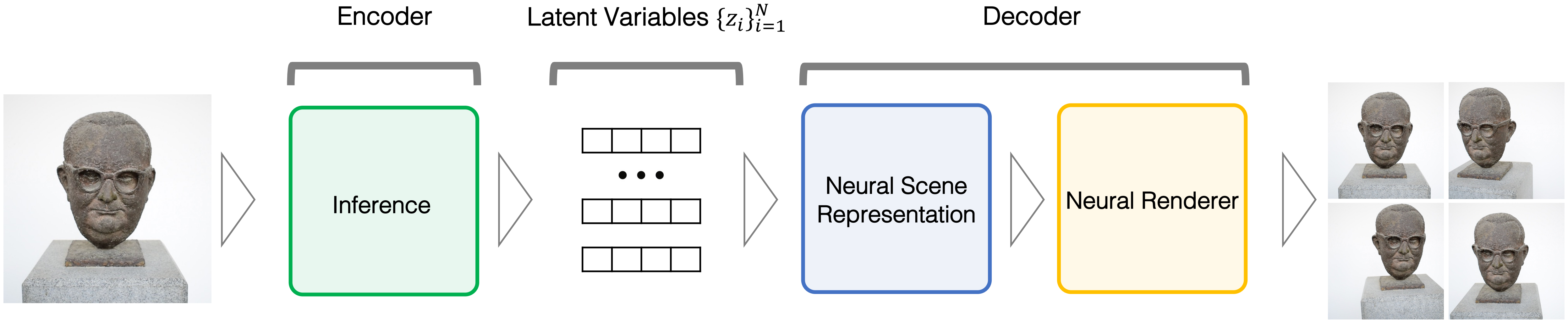
Sitzmann et al: Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations, NeurIPS 2020.

Inferring Neural Fields

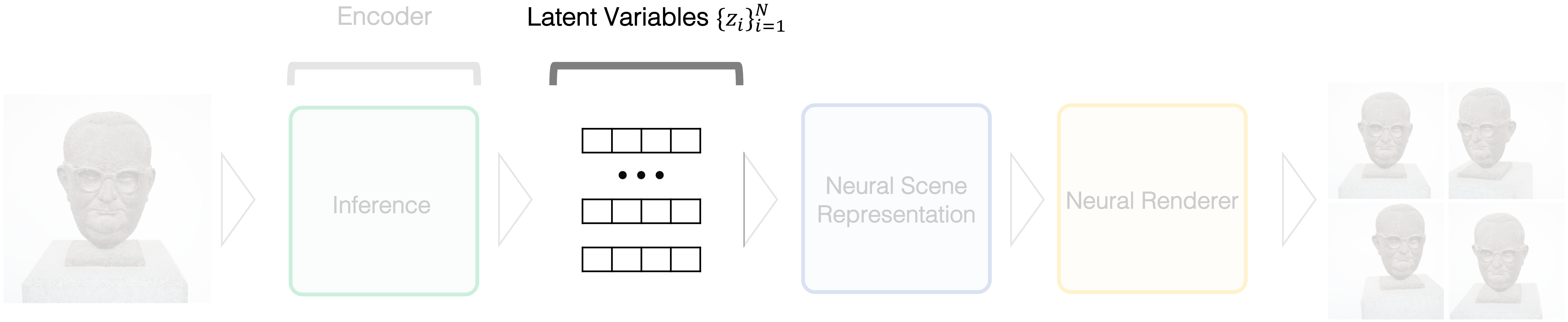


If only a single observation is available, or if only part of the scene has been observed, Inference needs to be prior-based – i.e., we need to learn to reconstruct.

General Framework: Encoder-Decoder

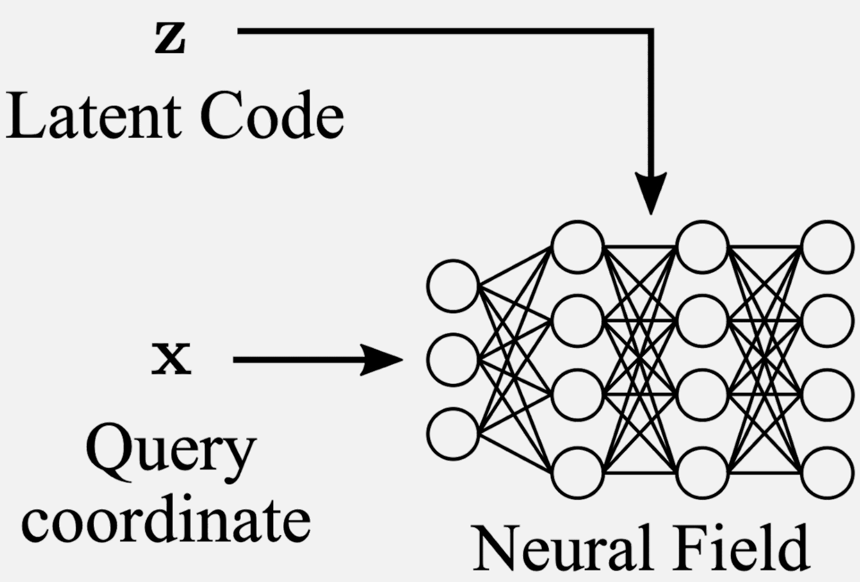


What are the latent variables?

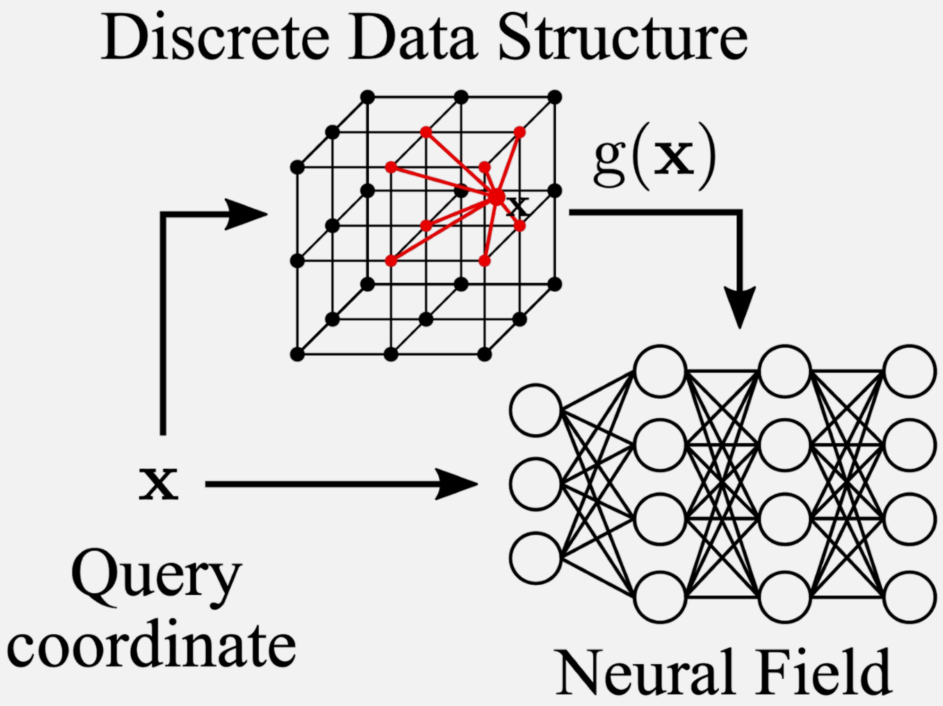


Latent Variables = hybrid representation -> helps in generalization

Key Consideration: *Locality.*



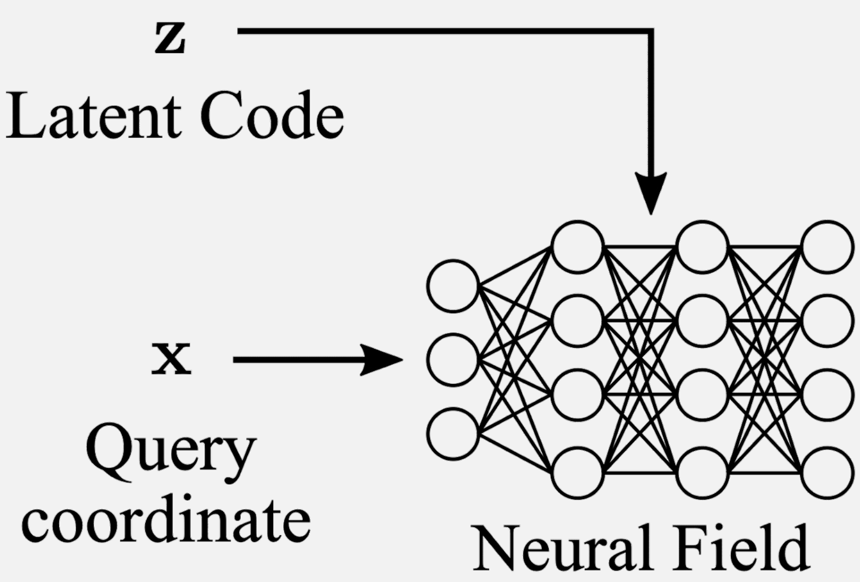
Global Conditioning



Local Conditioning

Neural Fields in Visual Computing and Beyond, Xie et al., EG STAR 2022

Global Latent Codes



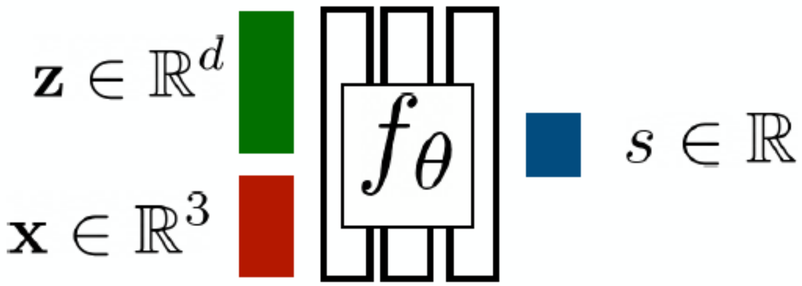
Global Conditioning

Local Conditioning

Neural Fields in Visual Computing and Beyond, Xie et al., EG STAR 2022

Global conditioning

Latent Conditioning based SDFs



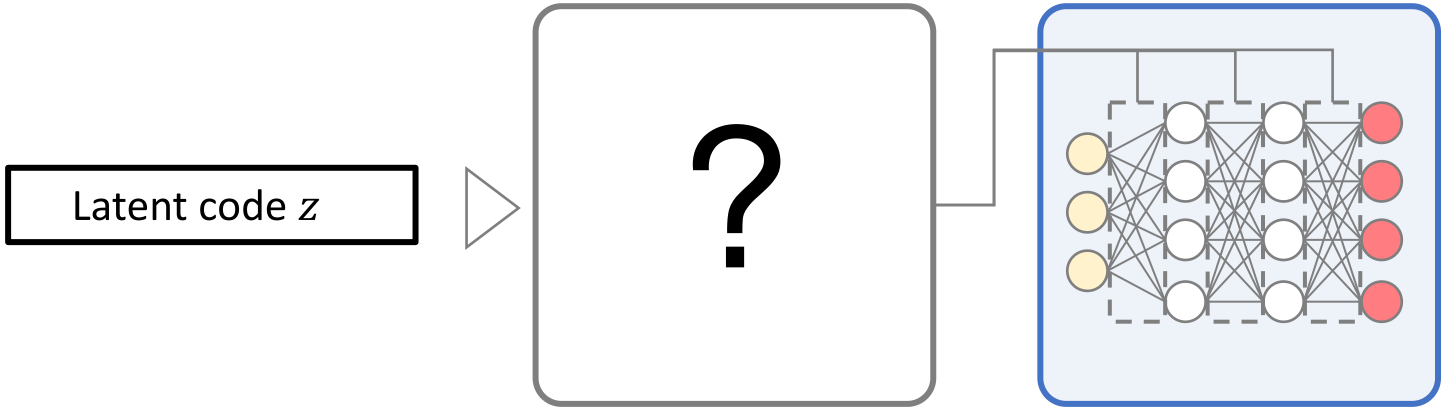
$$f_\theta : \mathbb{R}^3 \times \mathbb{R}^d \to \mathbb{R}$$

Generalizable Signed Distance Field:
(latent code, position) → (distance)

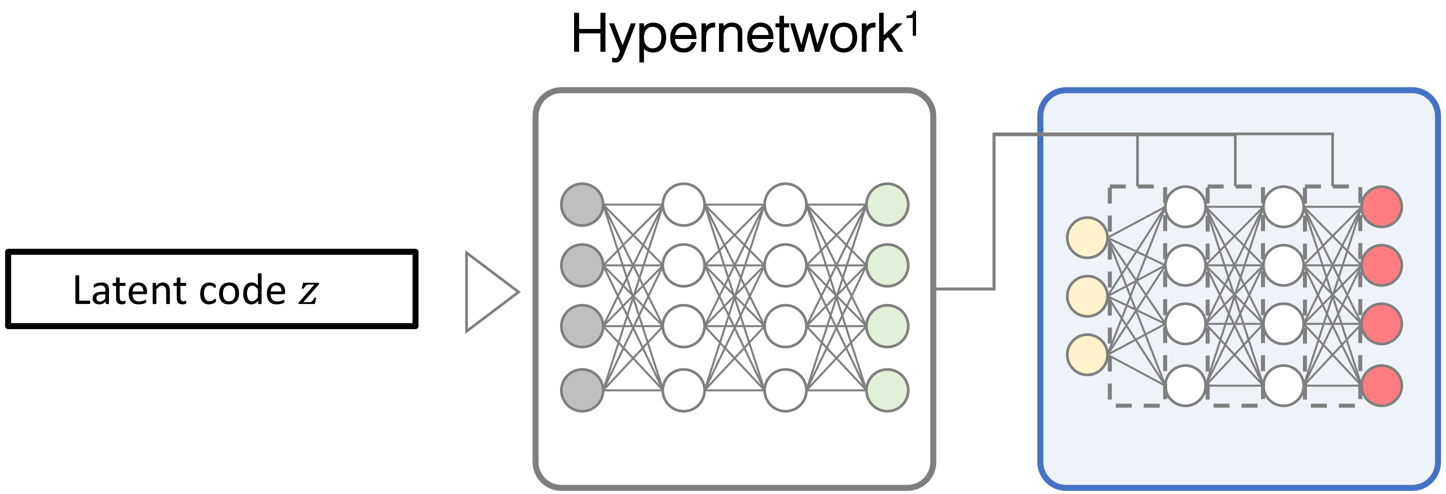
Each object is represented by a corresponding latent code (only *d* parameters per instance)

The same neural net parameters across **all** objects

Global conditioning

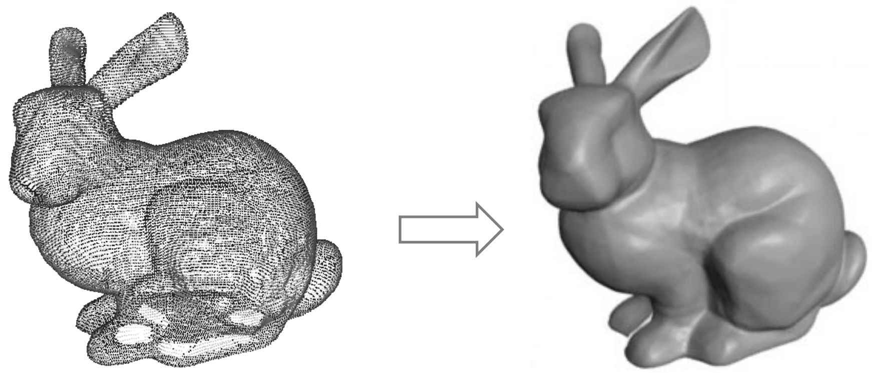


Global conditioning



[1][Schmidhuber et al. 1992, Schmidhuber et al. 1993, Stanley et al. 2009, Ha et al., 2016]

Global Latent Codes: Enables reconstruction from *partial* observations!



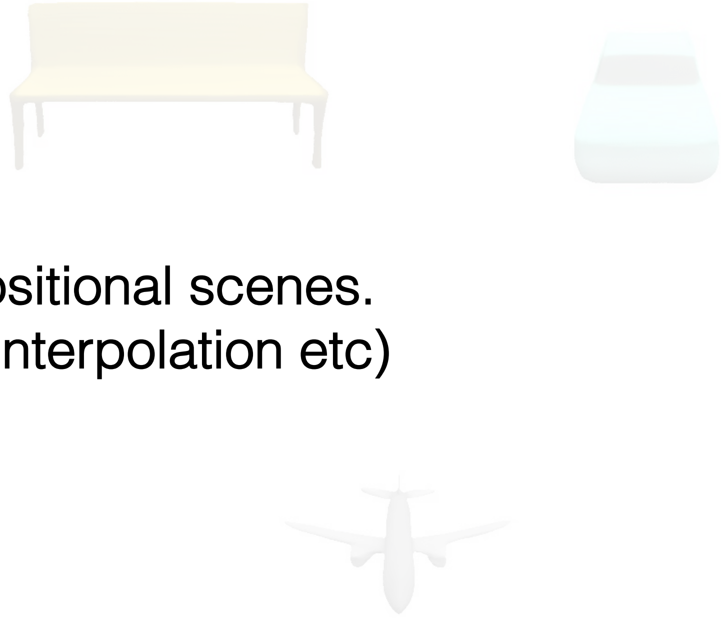
DeepSDF, Occupancy Networks, IM-Net

Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations, NeurIPS 2019.

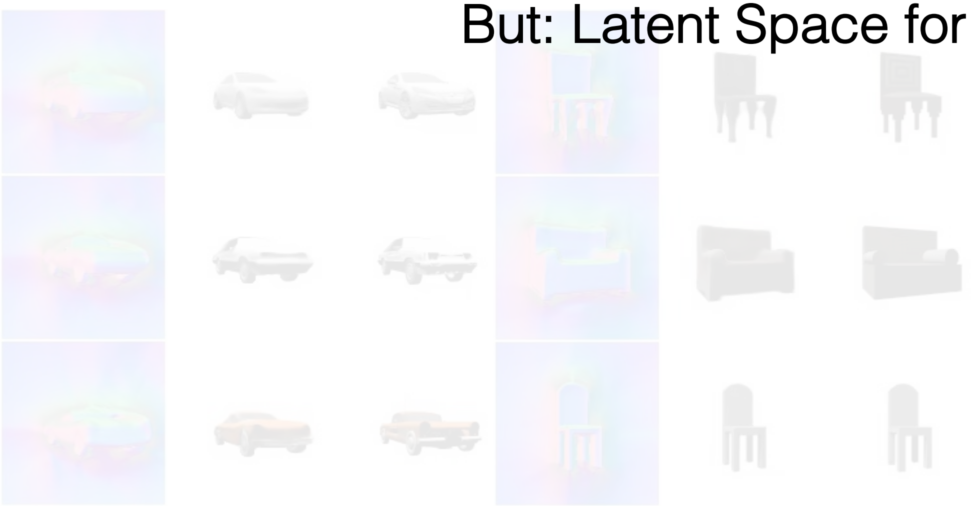
Differential Volumetric Rendering, Niemeyer et al., CVPR 2020

Global Latent Codes: Enables reconstruction from *partial* observations!

DeepSDF, Occupancy Networks, IM-Net



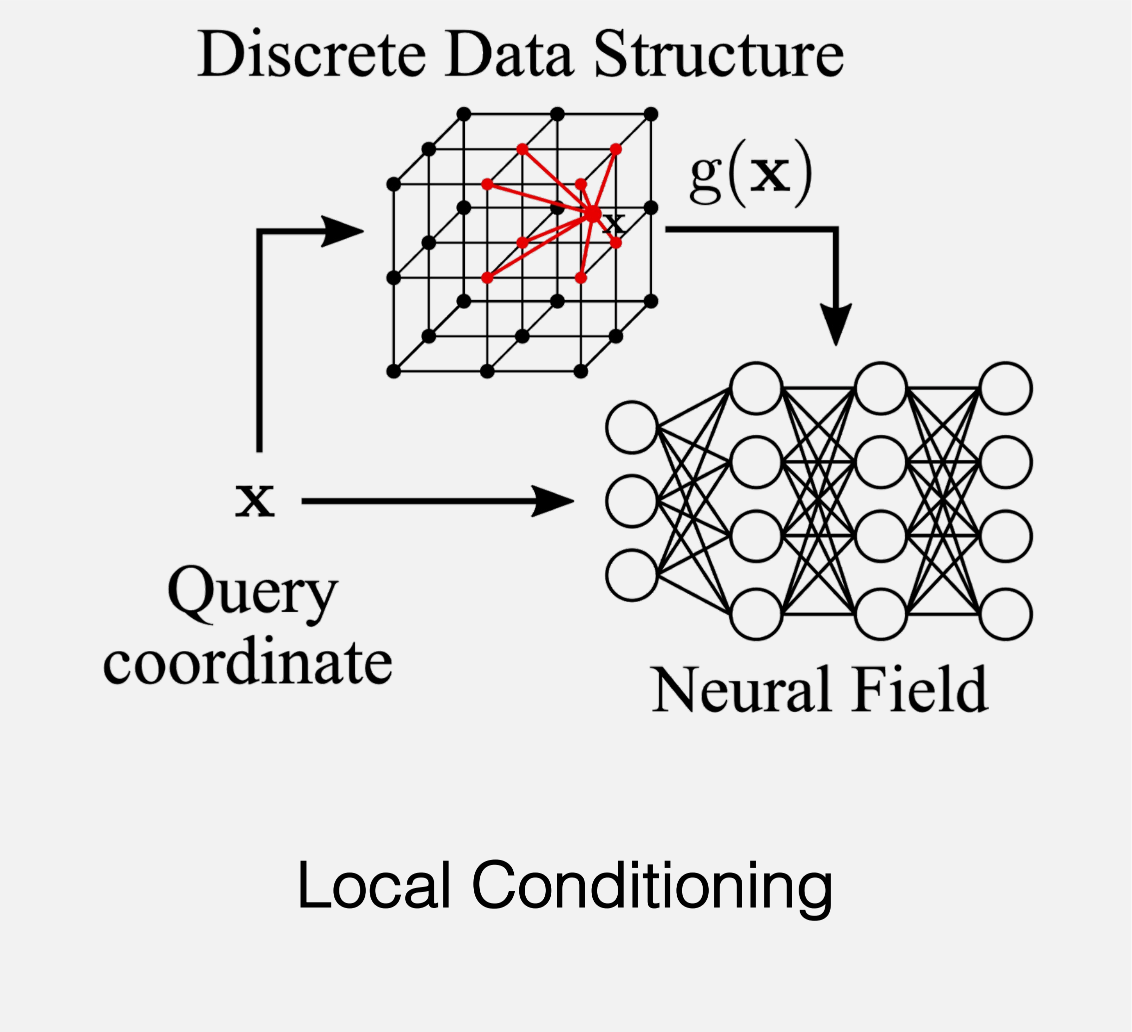
Key limitation: Simple, non-compositional scenes.
But: Latent Space for full objects (interpolation etc)



Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations, NeurIPS 2019.

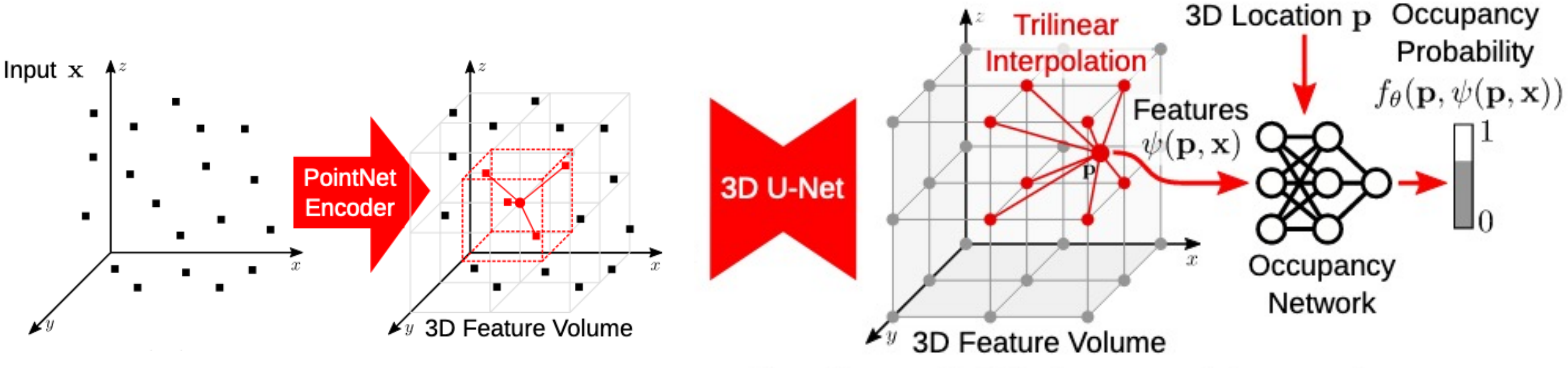
Differential Volumetric Rendering, Niemeyer et al., CVPR 2020

Local Latent Codes



Neural Fields in Visual Computing and Beyond, Xie et al., EG STAR 2022

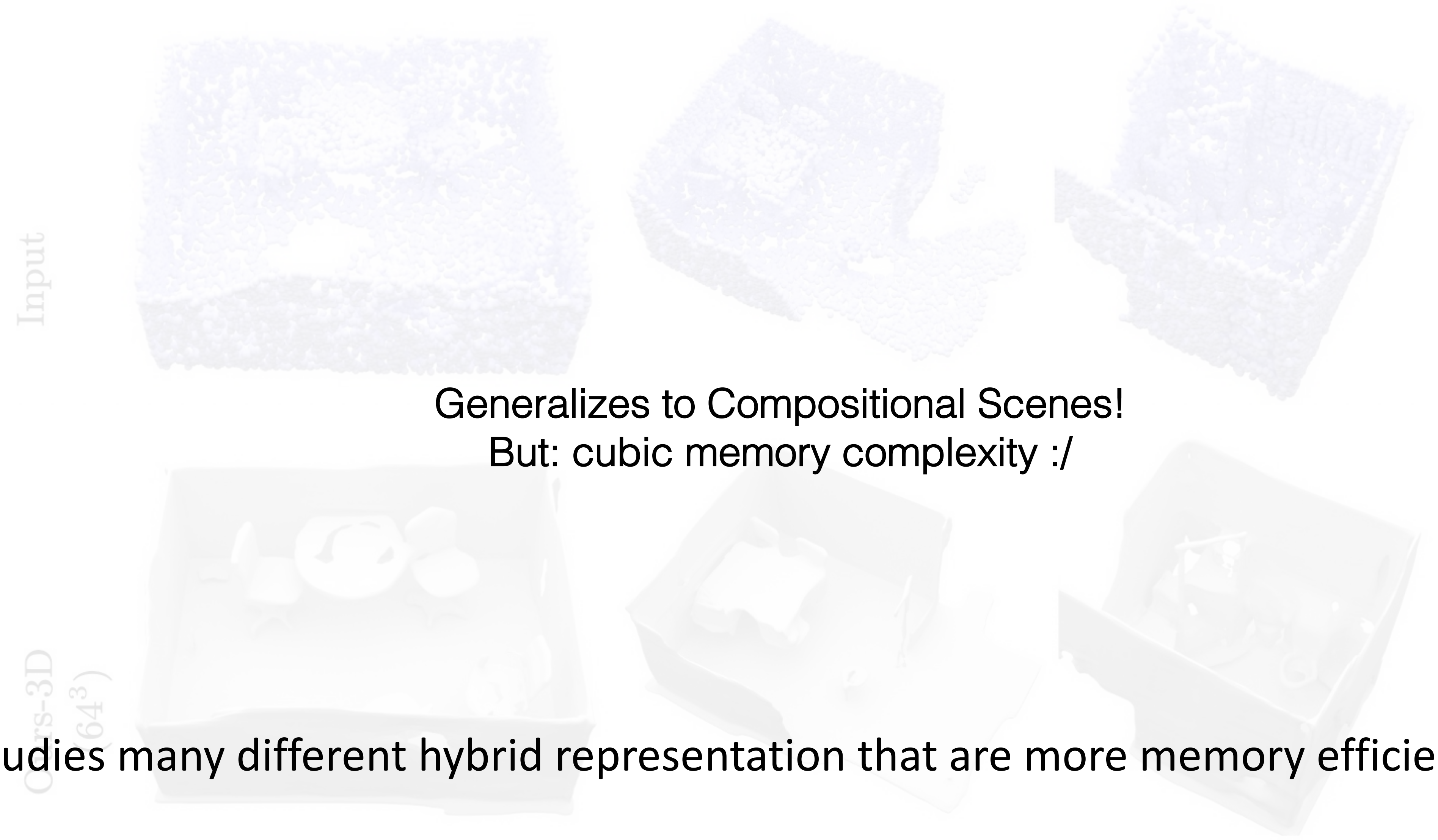
From point clouds: Conditioning on Feature Voxel grids



Local Conditioning = Hybrid Representation!

Convolutional Occupancy Networks [Peng et al. 2020]
Local Implicit Grid Representations for 3D Scenes [Jiang et al. 2020]
Implicit Functions in Feature Space for 3D Shape Reconstruction and Completion [Chabra et al. 2020]
Deep Local Shapes: Learning Local SDF Priors for Detailed 3D Reconstruction [Chibane et al. 2020]

From point clouds: Conditioning on Feature Voxel grids



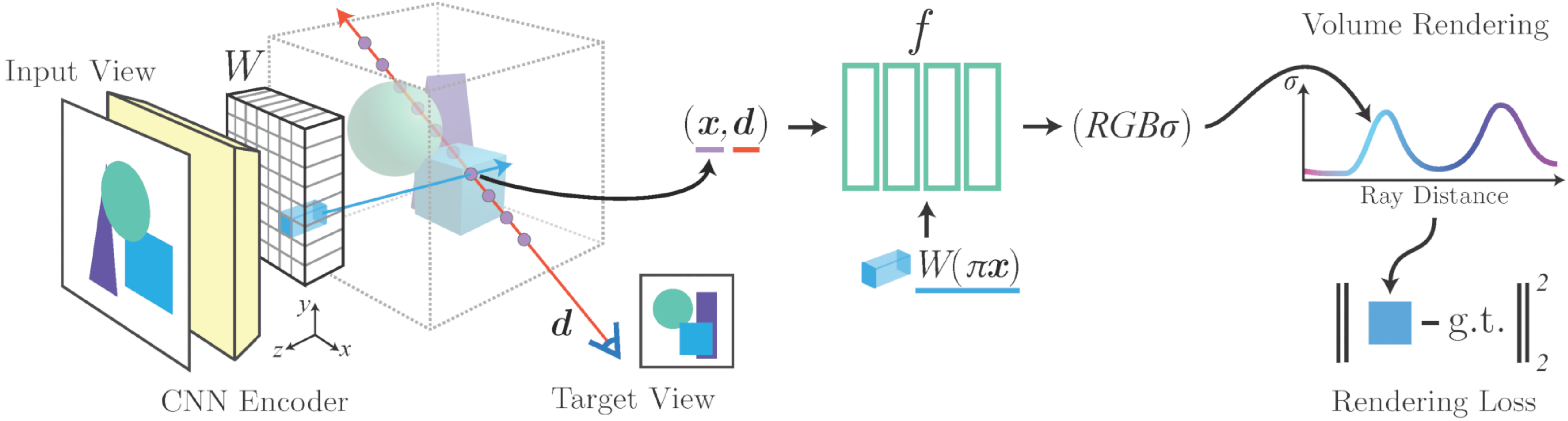
Generalizes to Compositional Scenes!
But: cubic memory complexity :/

We studies many different hybrid representation that are more memory efficient

How to locally condition if sensor domain different than field domain?

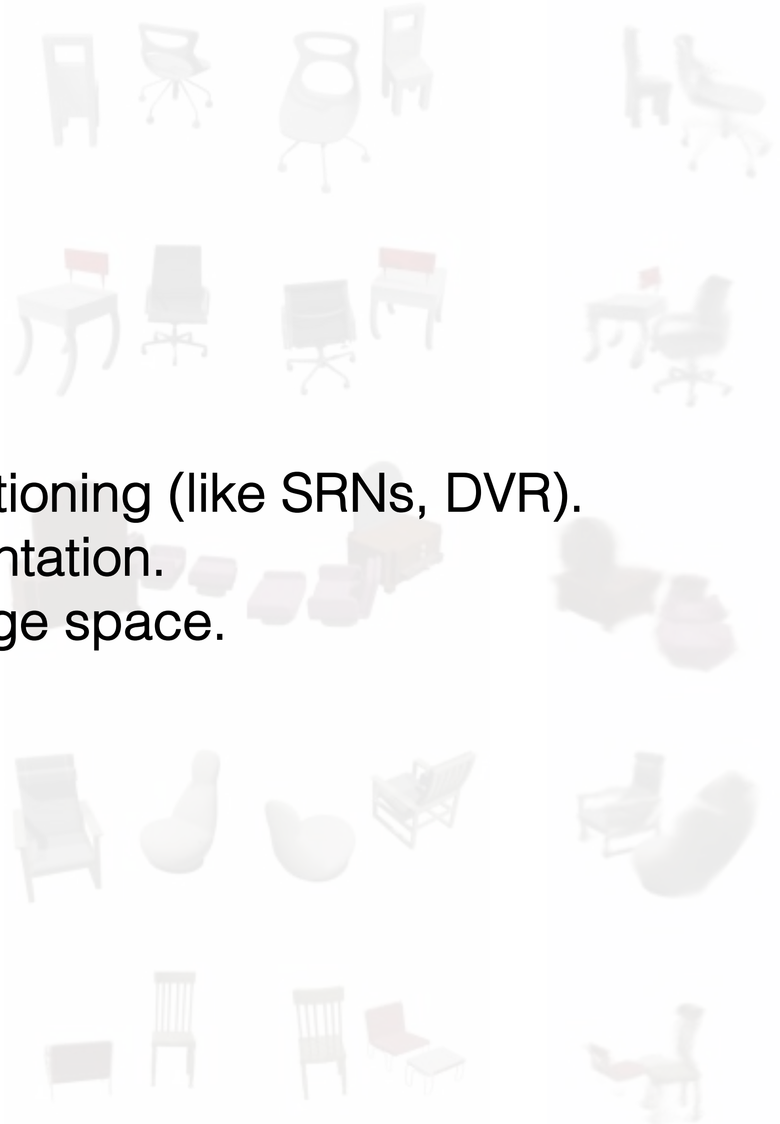
Local Conditioning: Pixel-Aligned Features.

Key idea: Project a 3D point to the 2D image and use 2D ConvNet features as the hybrid representation.

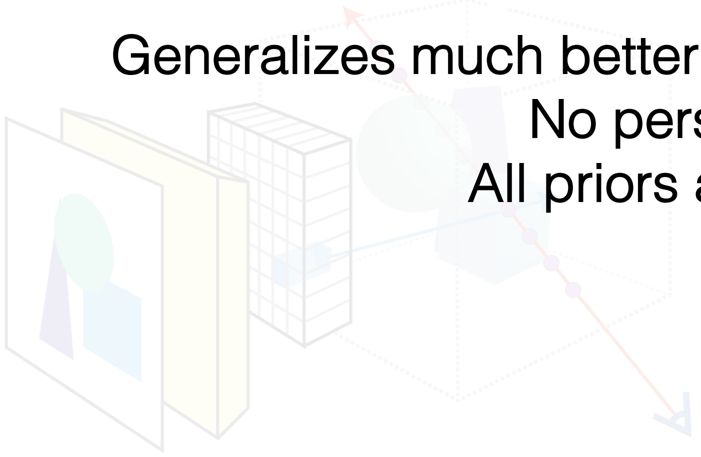


PiFU, Saito et al., ICCV 2019.
PixelNeRF, Yu et al., CVPR 2021
Grf: Learning a general radiance field…, Trevithick et al.

Local Conditioning: Pixel-Aligned Features.

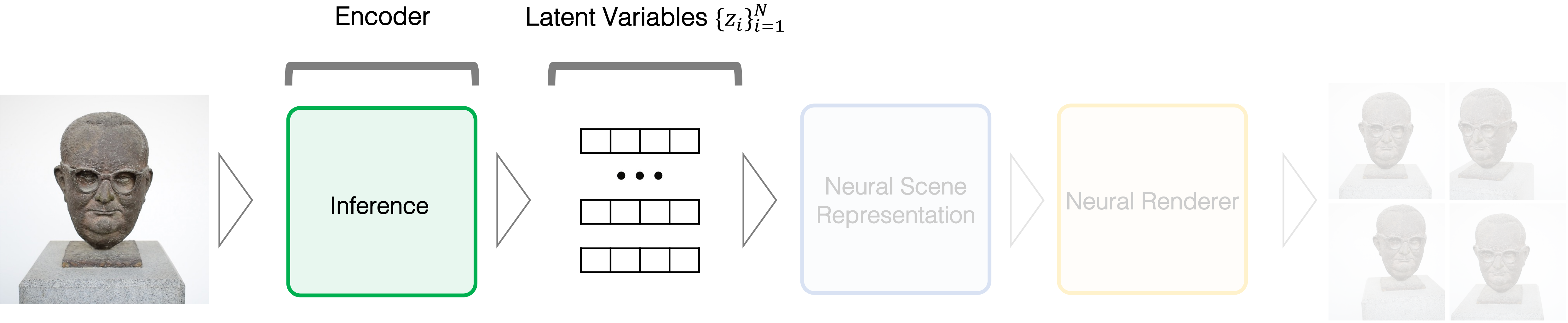


Generalizes much better than global conditioning (like SRNs, DVR).
No persistent 3D representation.
All priors are learned in image space.

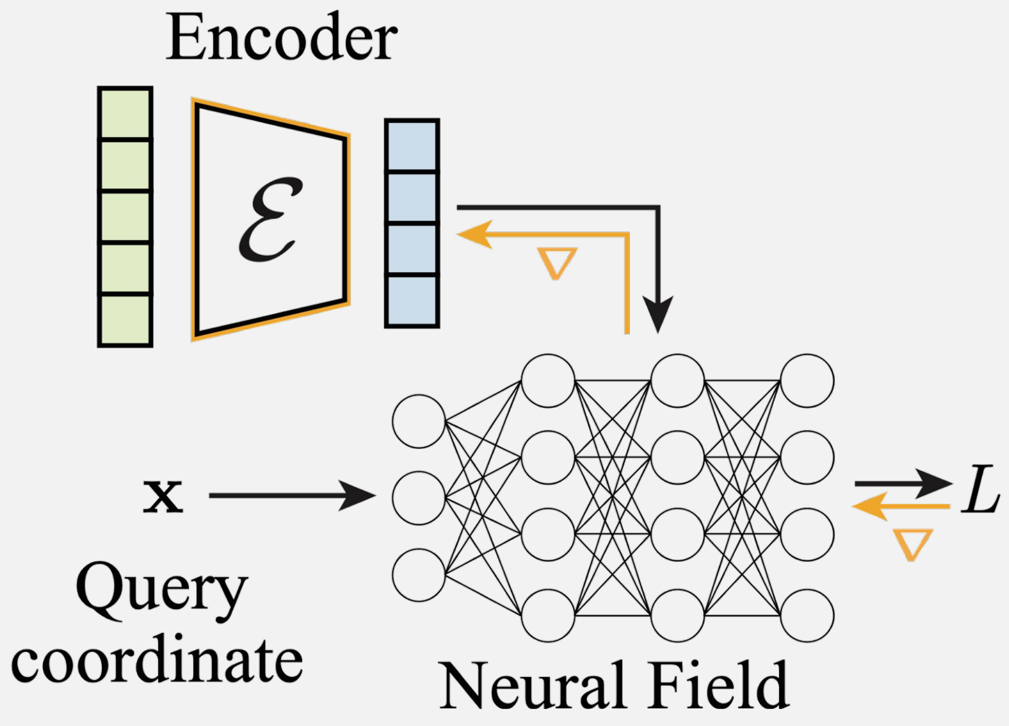


PiFU, Saito et al., ICCV 2019.
PixelNeRF, Yu et al., CVPR 2021
Grf: Learning a general radiance field…, Trevithick et al.

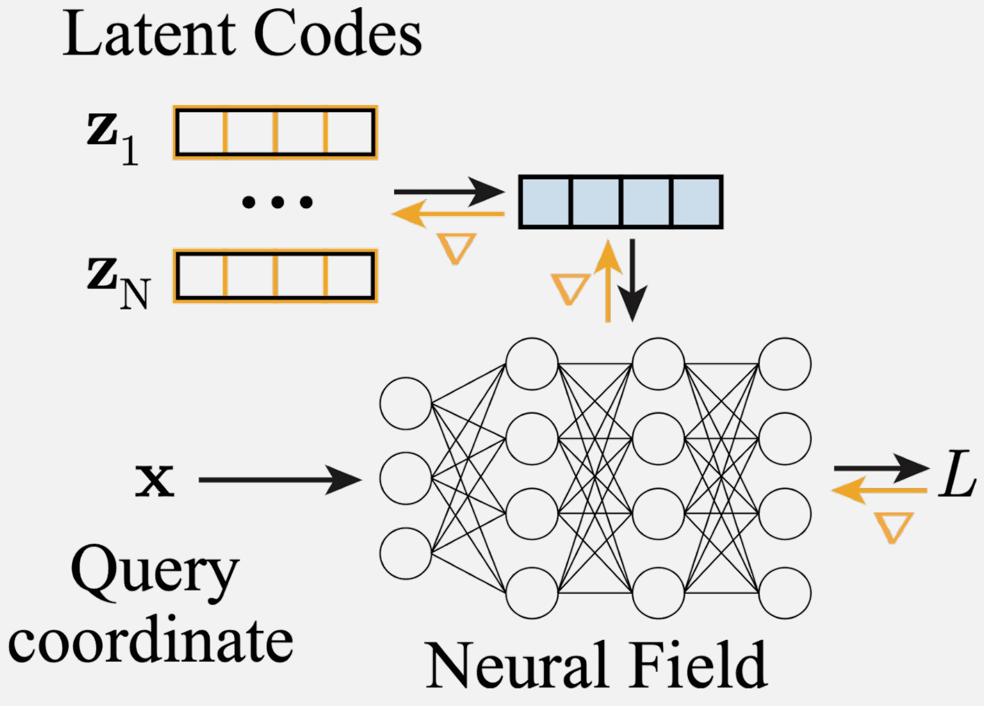
How to infer latent codes?



Encoding vs. Auto-Decoding



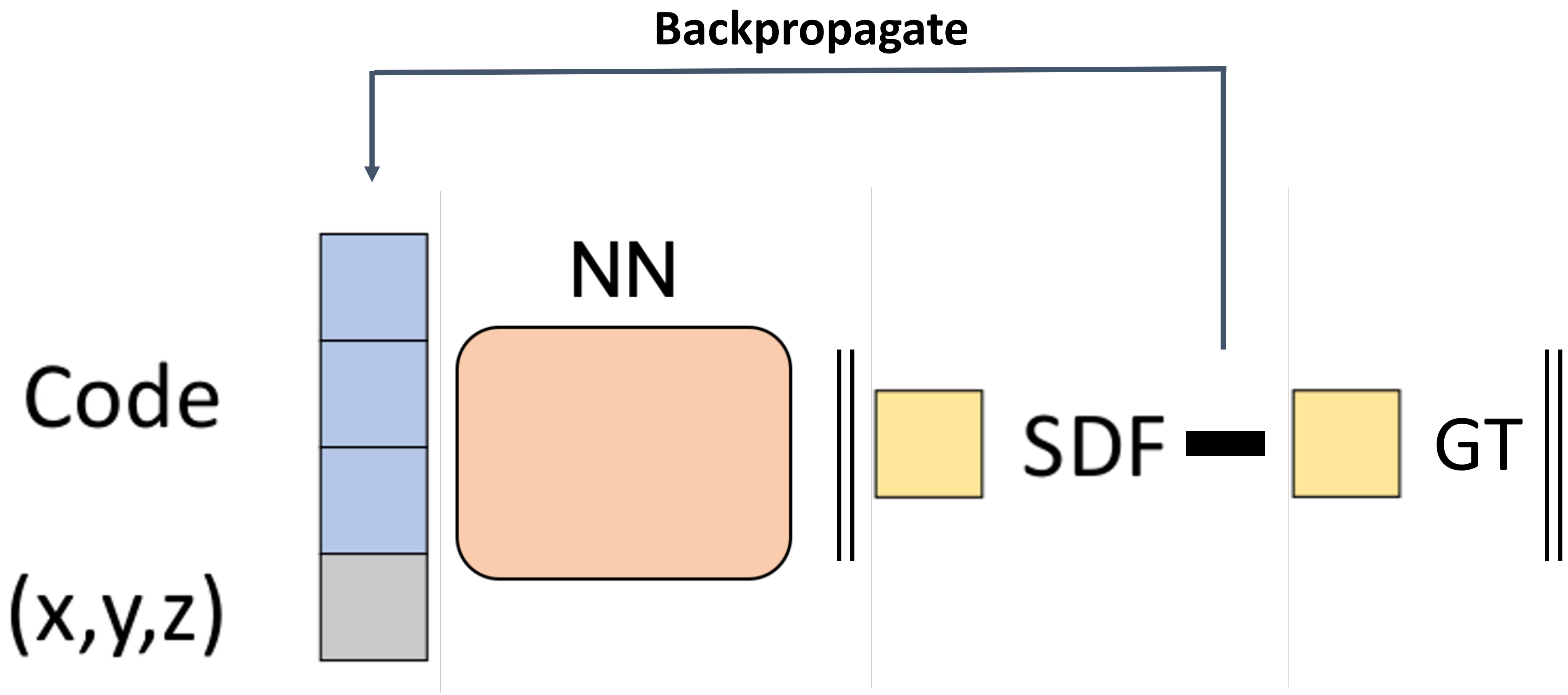
Encoding



Auto-Decoding

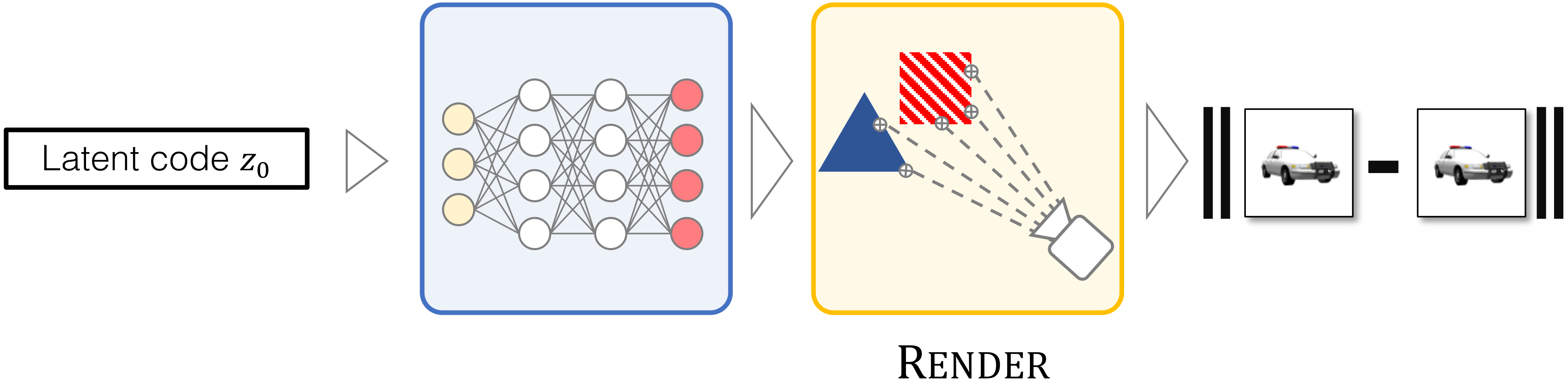
We have seen this in our DeepSDF lecture!

Auto-Decoder



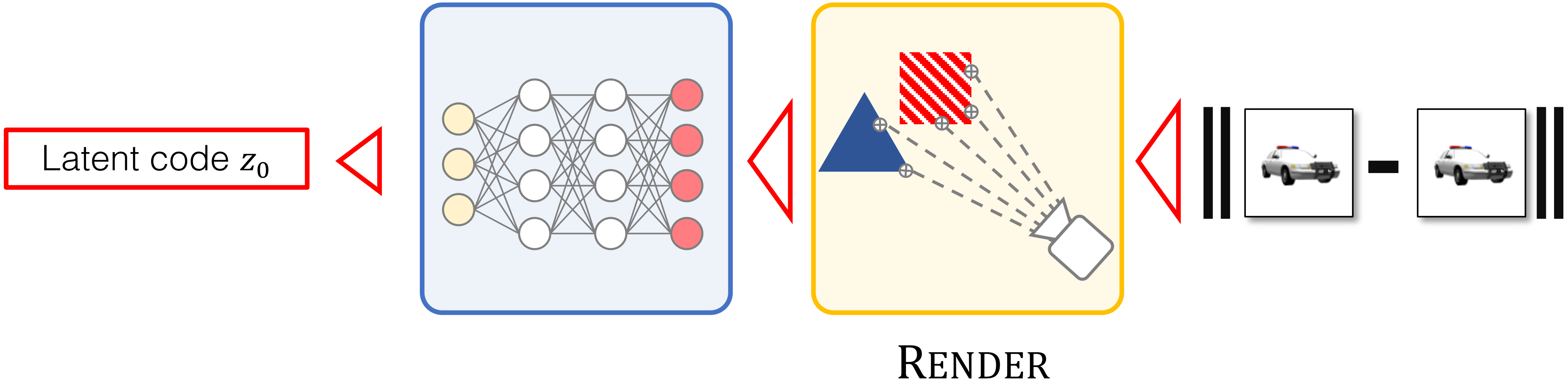
During Training: Optimize for both NN parameters and Code

Auto-Decoding for inverse graphics



Auto-Decoding for inverse graphics

3D-structured, resolution-invariant!
Samples need not lie on regular grids!



$$\hat{z} = \arg\min_{z} \|\mathrm{RENDER}(\Phi) - \mathcal{I}\|$$

Out-of-distribution generalization

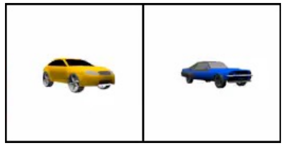
Auto-decoding often generalizes better than auto-encoding

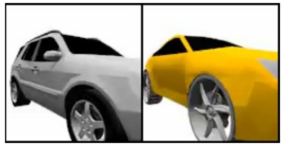
$$\hat{z} = \arg\min_{z} \left\| \mathrm{RENDER}_{\theta}(\mathrm{SRN}_{\phi=HN_{\psi}(z)}, \xi) - \mathcal{I} \right\|$$

CNN encoder

Input

Reconstruction





3D structure enables __generalization to out-of-distribution camera poses__!

Auto-encoding:

- Do not generalize well to out-of-distribution inputs, mainly due to lack of ConvNets ability to generalize.
- No optimization required at inference time, just 1 forward pass -> very fast

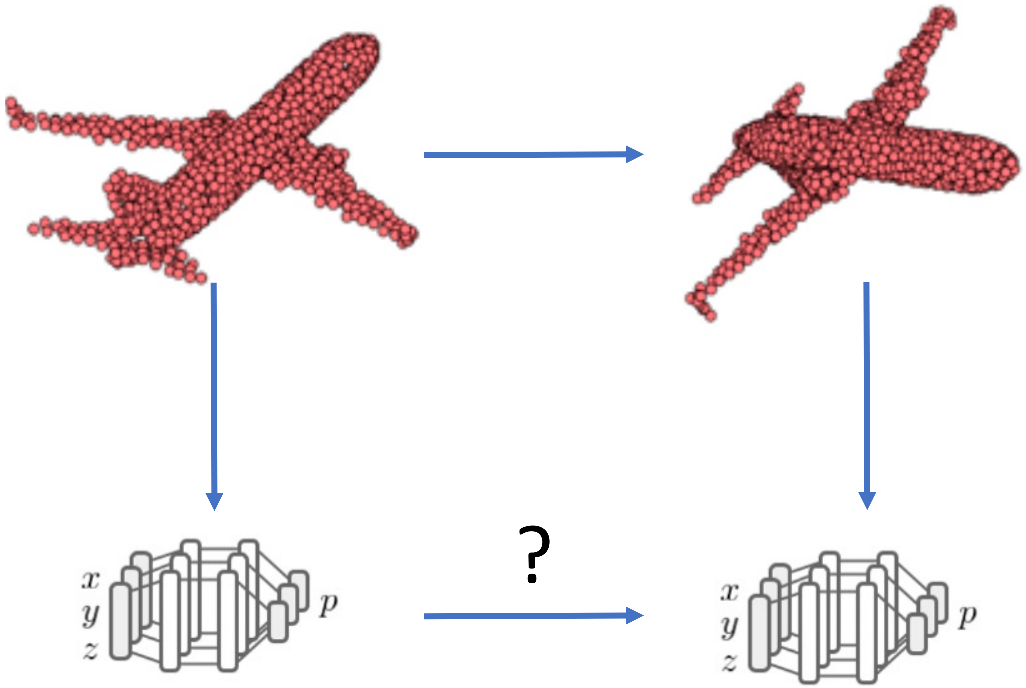
Auto-decoding:

- Generalized better to out-of-distribution inputs
- We need to run an optimization at inference time -> slow

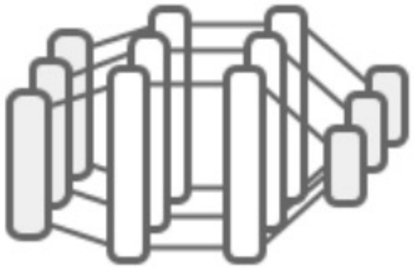
Outline

- Network Architecture
- Hybrid Representation
- Generalization
- Editing/Manipulation

Can we operate directly on Neural Fields?



Geometric Manipulation of Neural Fields

x  $F_\theta(x)$

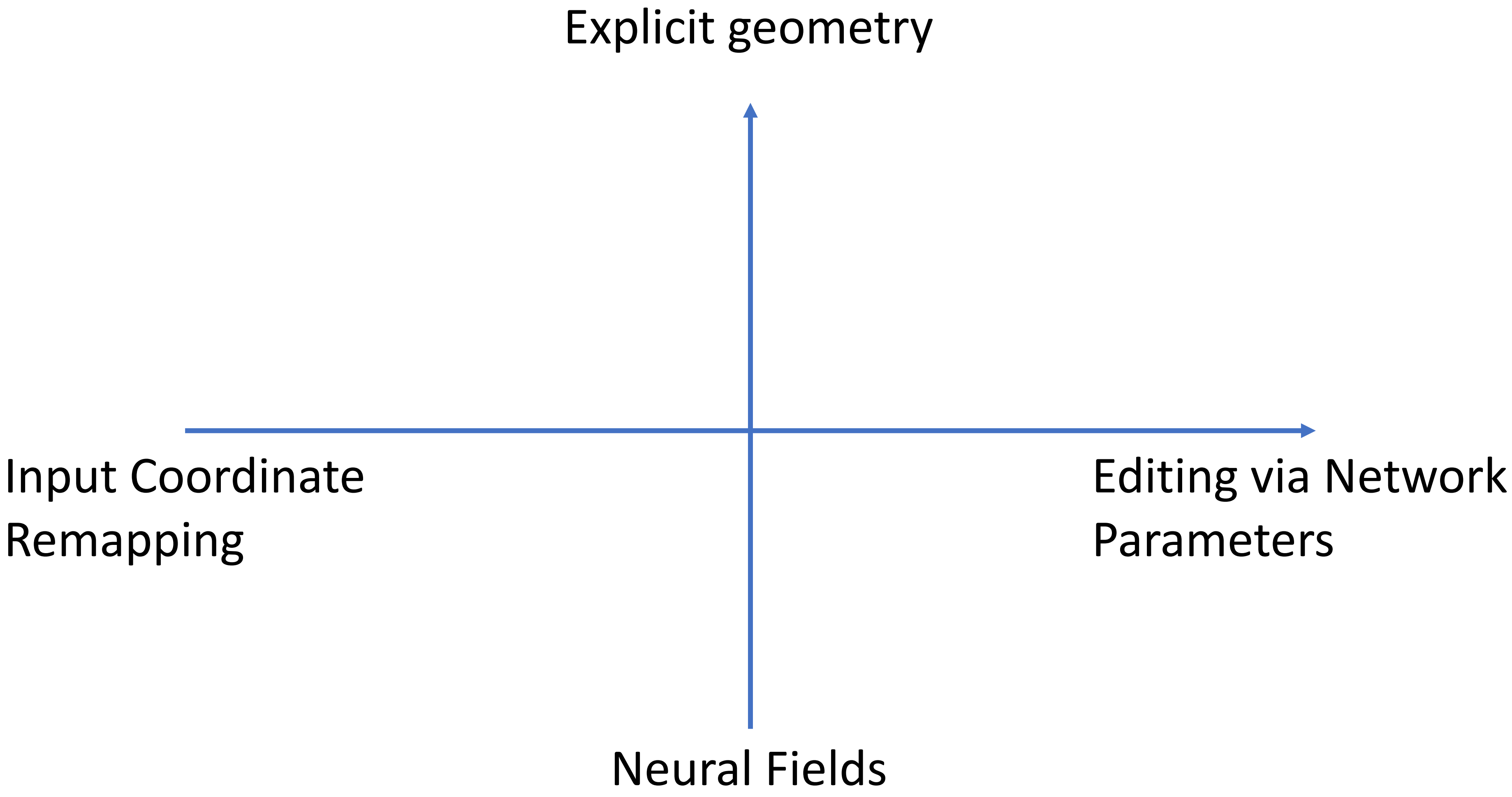
$$x' = \Phi(x)$$

$$\theta' = \Psi(\theta)$$

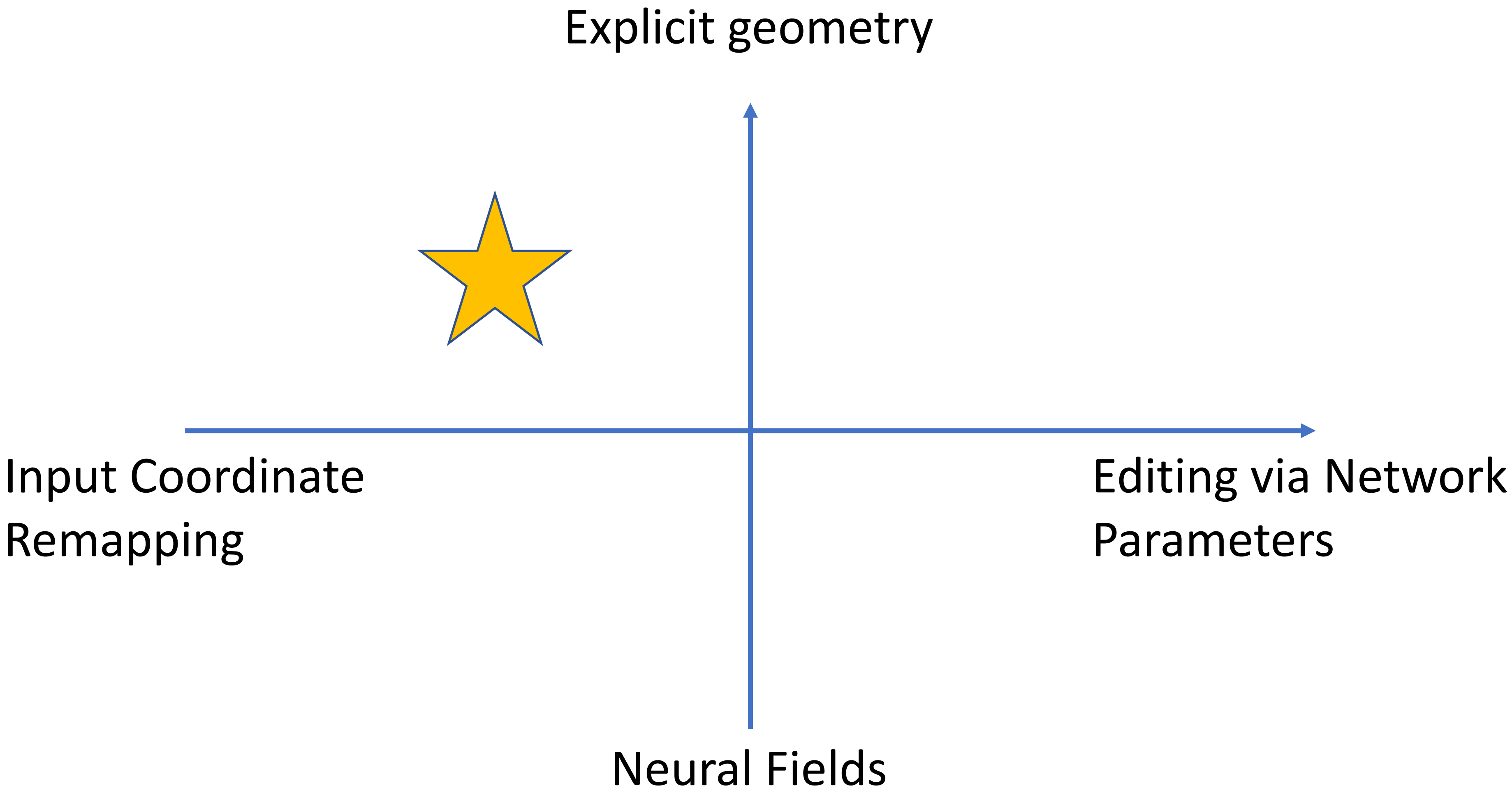
Input Coordinate Remapping

Editing via Network Parameters

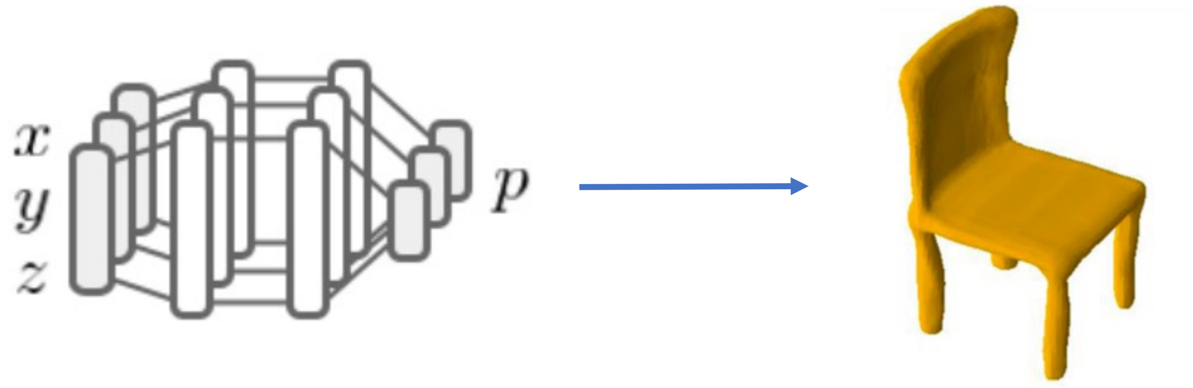
Geometric Manipulation of Neural Fields

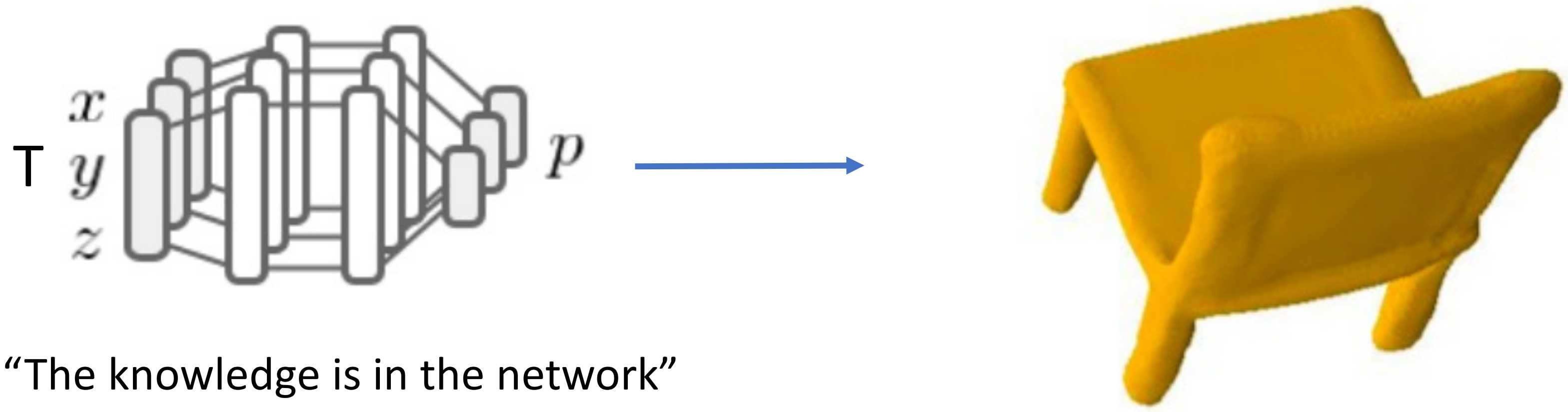


Geometric Manipulation of Neural Fields



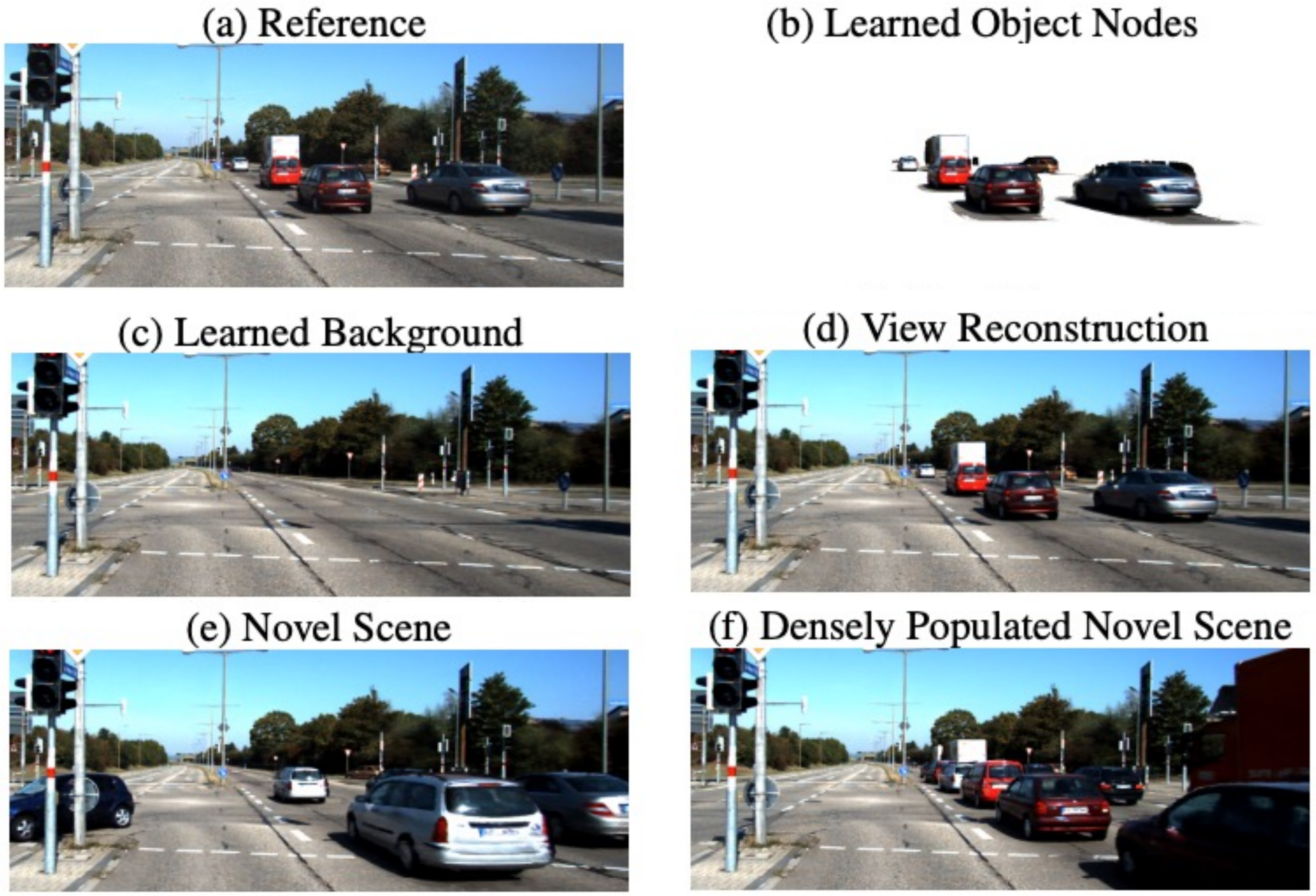
Input Coordinate Remapping through Explicit Geometry

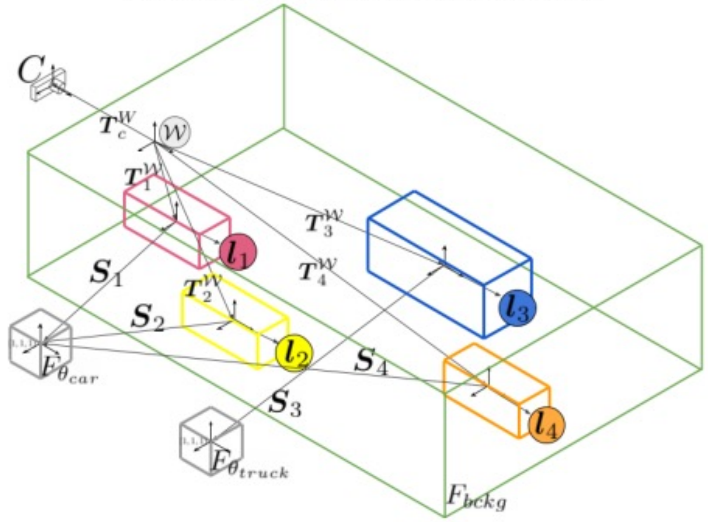




“The knowledge is in the network”

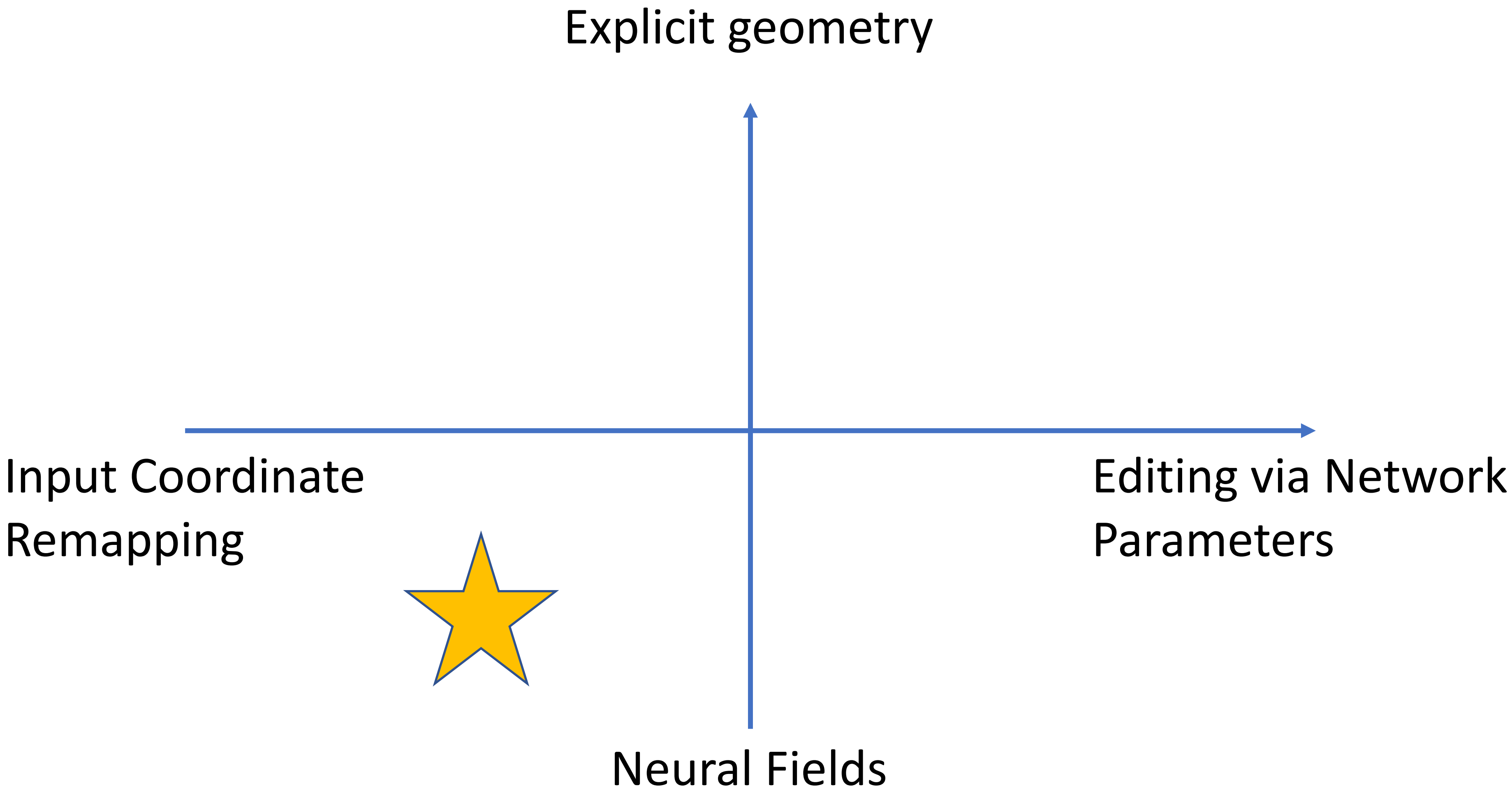
Dynamic Scene Manipulation through Local Frames



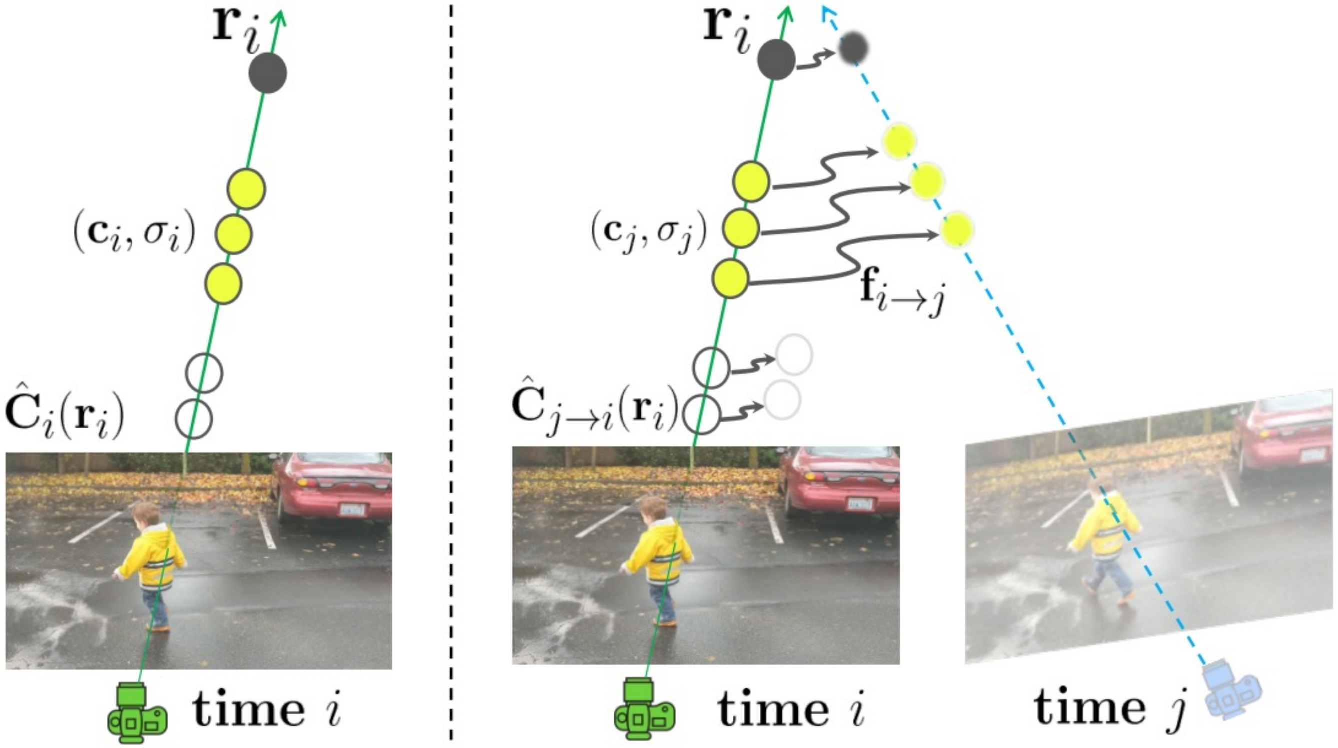


[Ost et al., CVPR'21]

Geometric Manipulation of Neural Fields



Scene Manipulation via Neural Flow Fields



$$\hat{\mathbf{C}}_{j\to i}(\mathbf{r}_i) = \int_{t_n}^{t_f} T_j(t)\,\sigma_j(\mathbf{r}_{i\to j}(t))\,\mathbf{c}_j(\mathbf{r}_{i\to j}(t), \mathbf{d}_i)dt$$

$$\text{where } \mathbf{r}_{i\to j}(t) = \mathbf{r}_i(t) + \mathbf{f}_{i\to j}(\mathbf{r}_i(t)).$$

Li et al., CVPR’21

Scene Manipulation via Neural Flow Fields

- Temporal photometric consistency

$$\mathcal{L}_{\text{pho}} = \sum_{\mathbf{r}_i} \sum_{j \in \mathcal{N}(i)} ||\hat{\mathbf{C}}_{j \rightarrow i}(\mathbf{r}_i) - \mathbf{C}_i(\mathbf{r}_i)||_2^2$$

- Data prior (2D flow prediction net)

$$\mathcal{L}_{\text{geo}} = \sum_{\mathbf{r}_i} \sum_{j \in \{i \pm 1\}} ||\hat{\mathbf{p}}_{i \rightarrow j}(\mathbf{r}_i) - \mathbf{p}_{i \rightarrow j}(\mathbf{r}_i))||_1.$$

Li et al., CVPR'21

Scene Manipulation via Neural Flow Fields

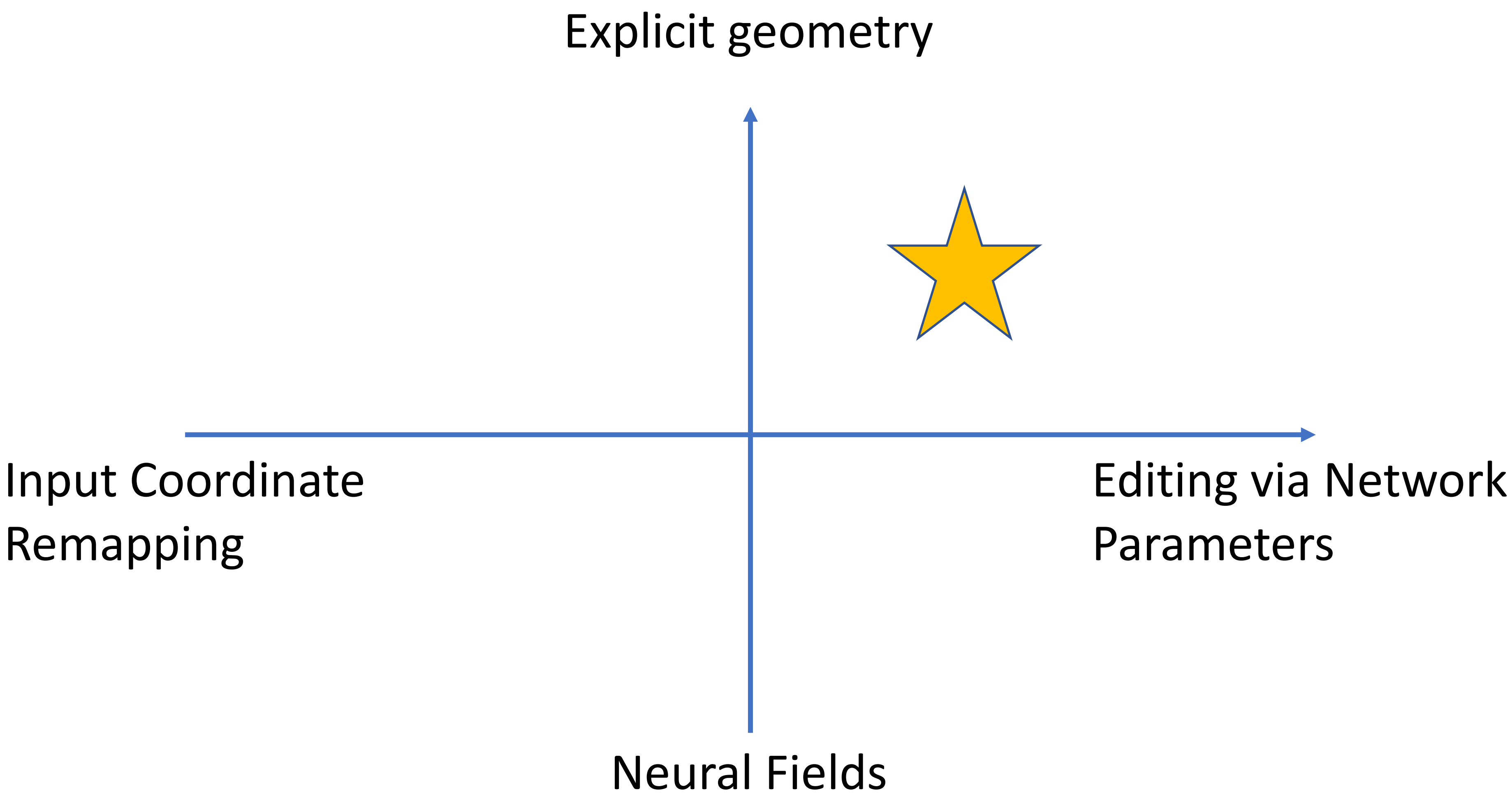
Input

Fixed time

Fixed view

Space-time interpolation

Geometric Manipulation of Neural Fields



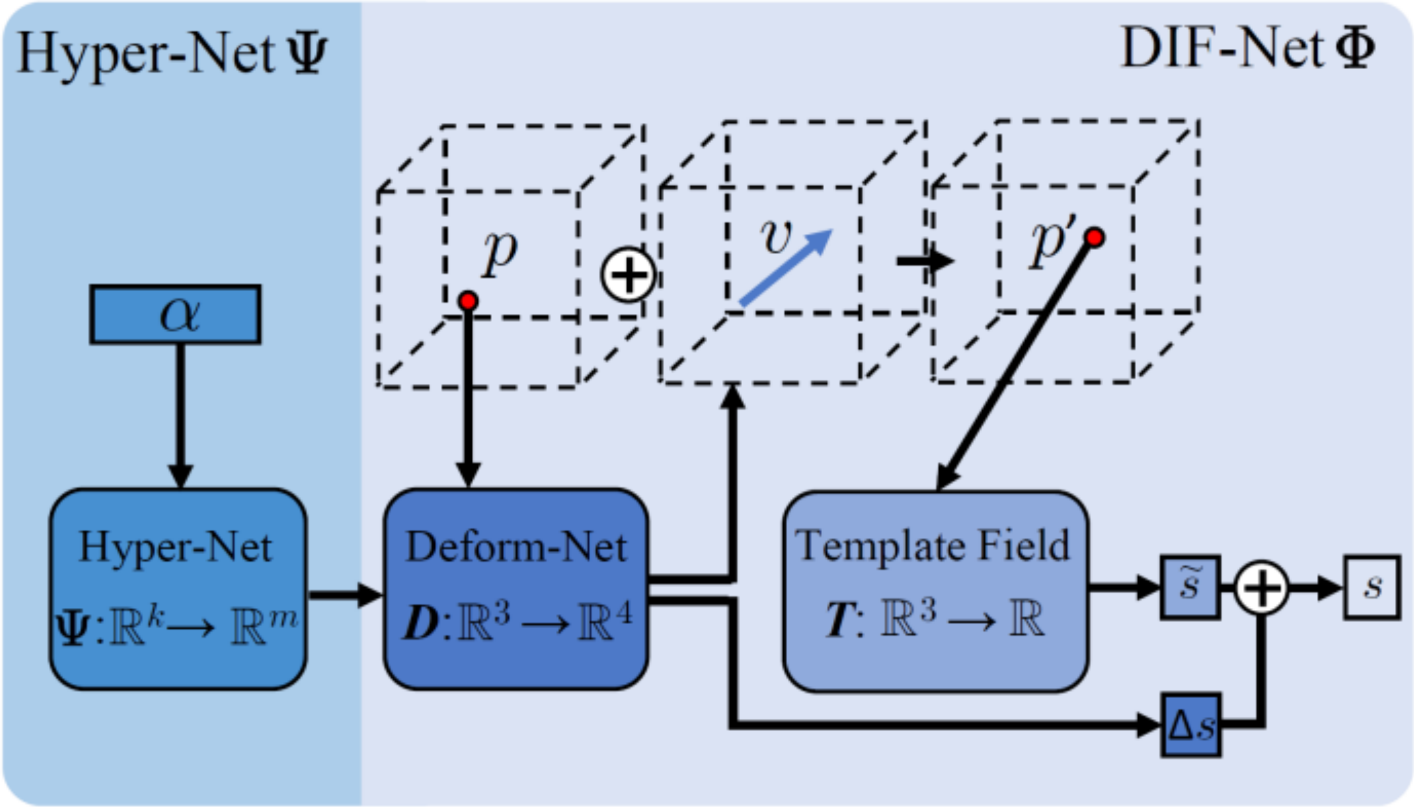


Figure 2. Overview of our proposed method. For a shape code α, Hyper-Net Ψ predicts (a part of) the weights of DIF-Net Φ, which further predicts the SDF for the shape. DIF-Net Φ consists of Deform-Net D which predicts a 3D deformation field and a correction field for the shape, and network T for generating a template implicit field shared across all shapes.

Vary parameters of the neural network to deform the final 3D shape represented by SDF.

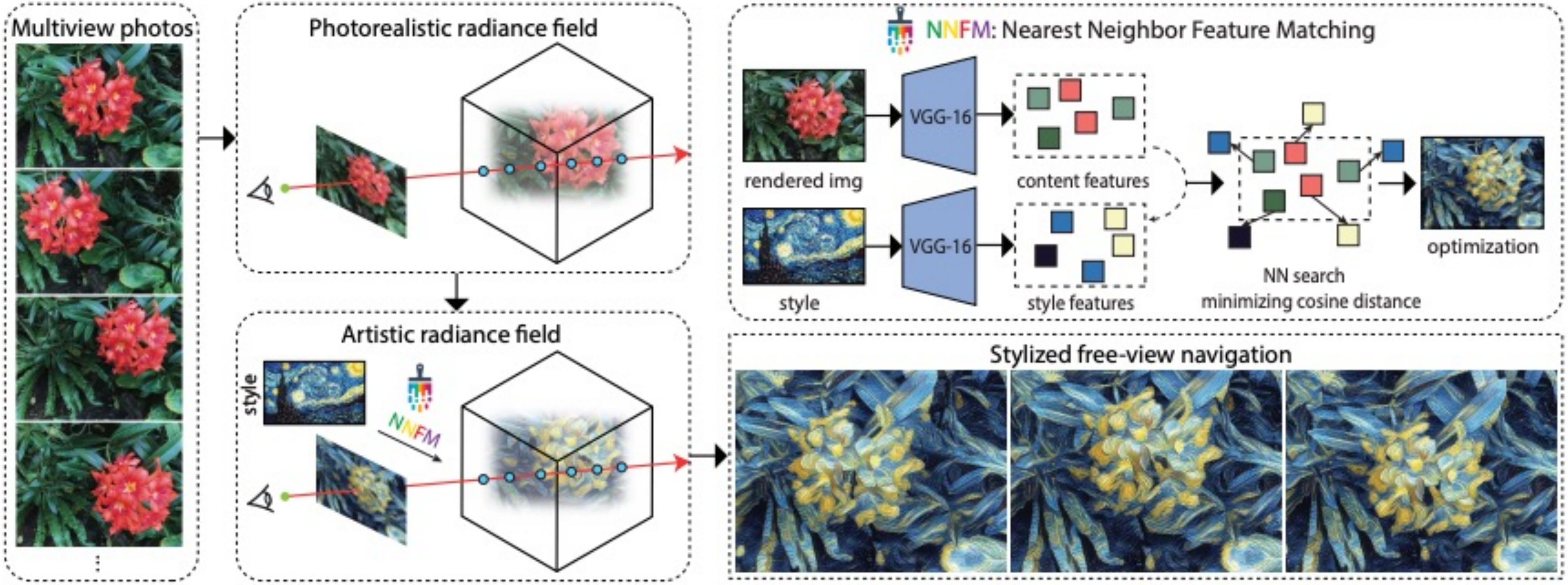
"Deformed Implicit Field: Modeling 3D Shapes with Learned Dense Correspondence", Deng et al. CVPR 2021.

Beyond Geometry

Optimization-based Editing: Style

Zhang et al., 2022

Optimization-based Editing: Style



Zhang et al., 2022

WEEK 12				
Tue Nov 1	Faster Inference	[1]Plenoxels: Radiance Fields without Neural Networks. [2]Instant Neural Graphics Primitives with a Multiresolution Hash Encoding.	[1] Raul Chun-Hung Chao [2] Ian Thomas	Smart data structure to enable fast rendering
Thrs Nov 3	Generalization	[1]GRF: Learning a General Radiance Field for 3D Scene Representation and Rendering. [2]pixelNeRF: Neural Radiance Fields from One or Few Images.	[1] Chin Tseng [2] Austin Hale	Generalization, rather than overfitting/optimization.
WEEK 13				
Tue Nov 8	Multi-View	[1]IBRNet: Learning Multi-View Image-Based Rendering. [2]MVSNeRF: Fast Generalizable Radiance Field Reconstruction from Multi-View Stereo.	[1] Jun Myeong Choi [2] Pierre-Nicolas Perrin	NeRF for multi-view stereo: 3D reconstruction from multiple images.
Thrs Nov 10	Inverse Rendering	[1]NeROIC: Neural Object Capture and Rendering from Online Image Collections . [2]SAMURAI: Shape And Material from Unconstrained Real-world Arbitrary Image collections.	[1] Basar Demir [2] Chin Tseng	NeRF for Inverse Rendering: Predicting shape + material.
WEEK 14				
Tue Nov 15	Misc.	[1]SparseNeuS: Fast Generalizable Neural Surface Reconstruction from Sparse views. [2]Dex-NeRF: Using a Neural Radiance field to Grasp Transparent Objects.	[1] Austin Hale [2] Ian Thomas	Misc papers: • From sparse views • Robotics Application • Deformable objects (faces) • Compositionality
Thrs Nov 17	Misc.	[1]Nerfies: Deformable Neural Radiance Fields. [2]GIRAFFE: Representing Scenes as Compositional Generative Neural Feature Fields.	[1] Basar Demir [2] Pierre-Nicolas Perrin	

Slide Credits

- ”Introduction to Computer Vision”, Noah Snavely, Cornell Tech, Spring 2022
- “Understanding and Extending Neural Radiance Field”, Jon Barron MIT & Tu Munich Lecture.
- “Neural Fields in Computer Vision”, CVRP 2022 Tutorial.
- Shubham Tulsiani, “Learning for 3D Vision”, Spring 2022, CMU
- Leo Guibas, JJ Park, “Neural Models for 3D geometry”, Spring 2022, Stanford.