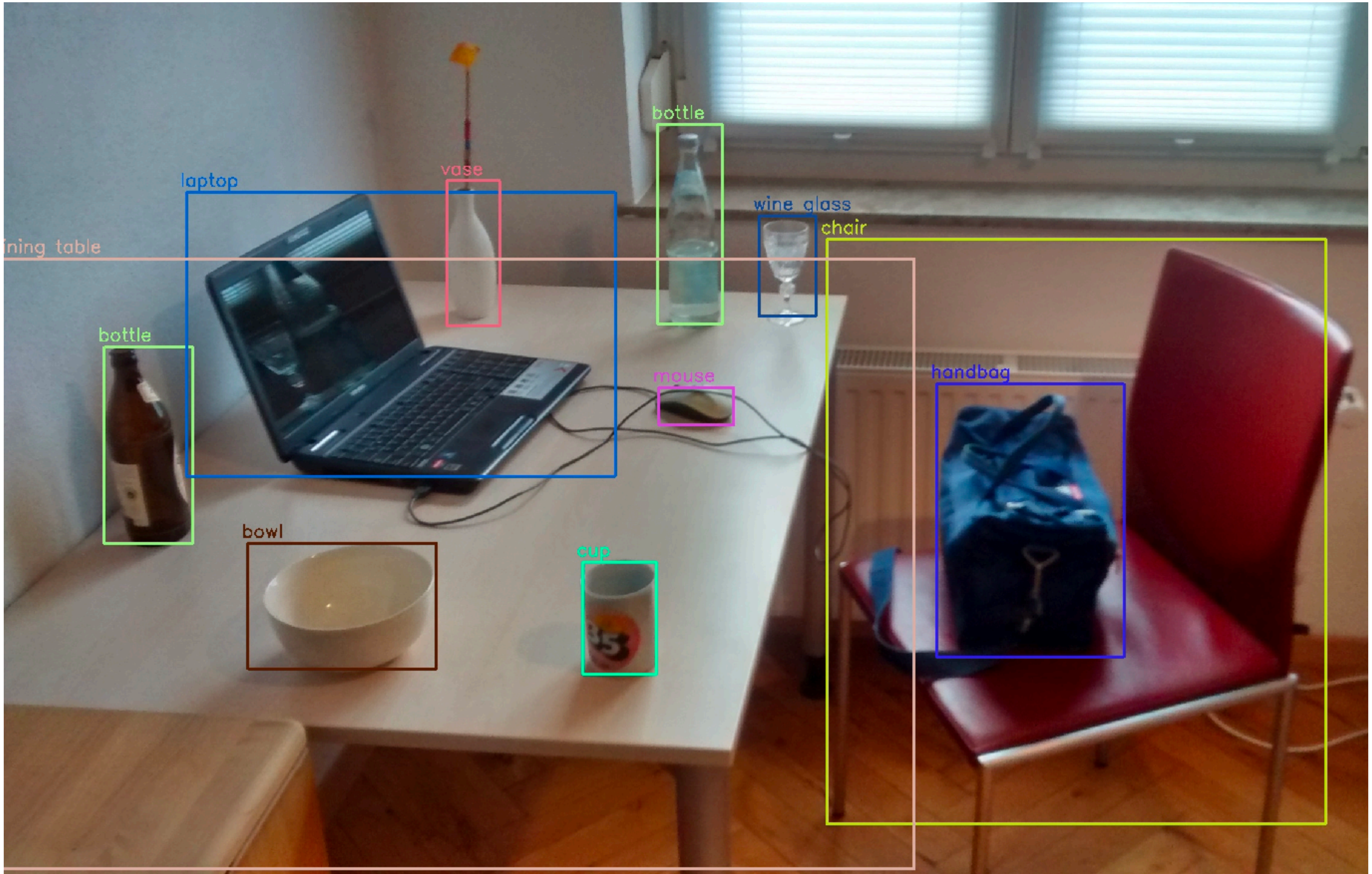
3D Perception: PointNet and NERFs

Sanjiban Choudhury

Cornell Bowers CIS
Computer Science

Last Class: How does a robot identify objects?



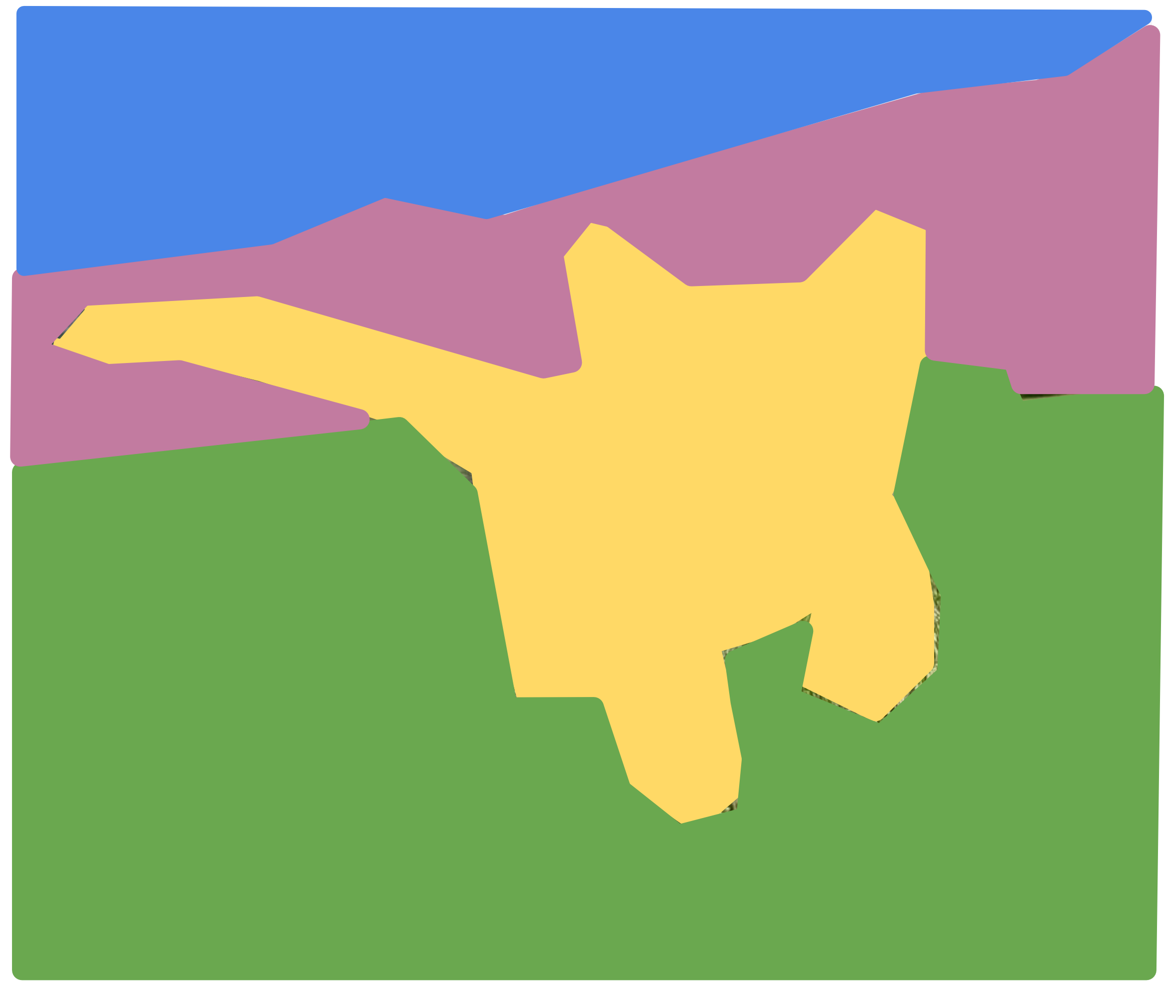
Last Class: How does a robot identify objects?

Classification

CAT

No spatial extent

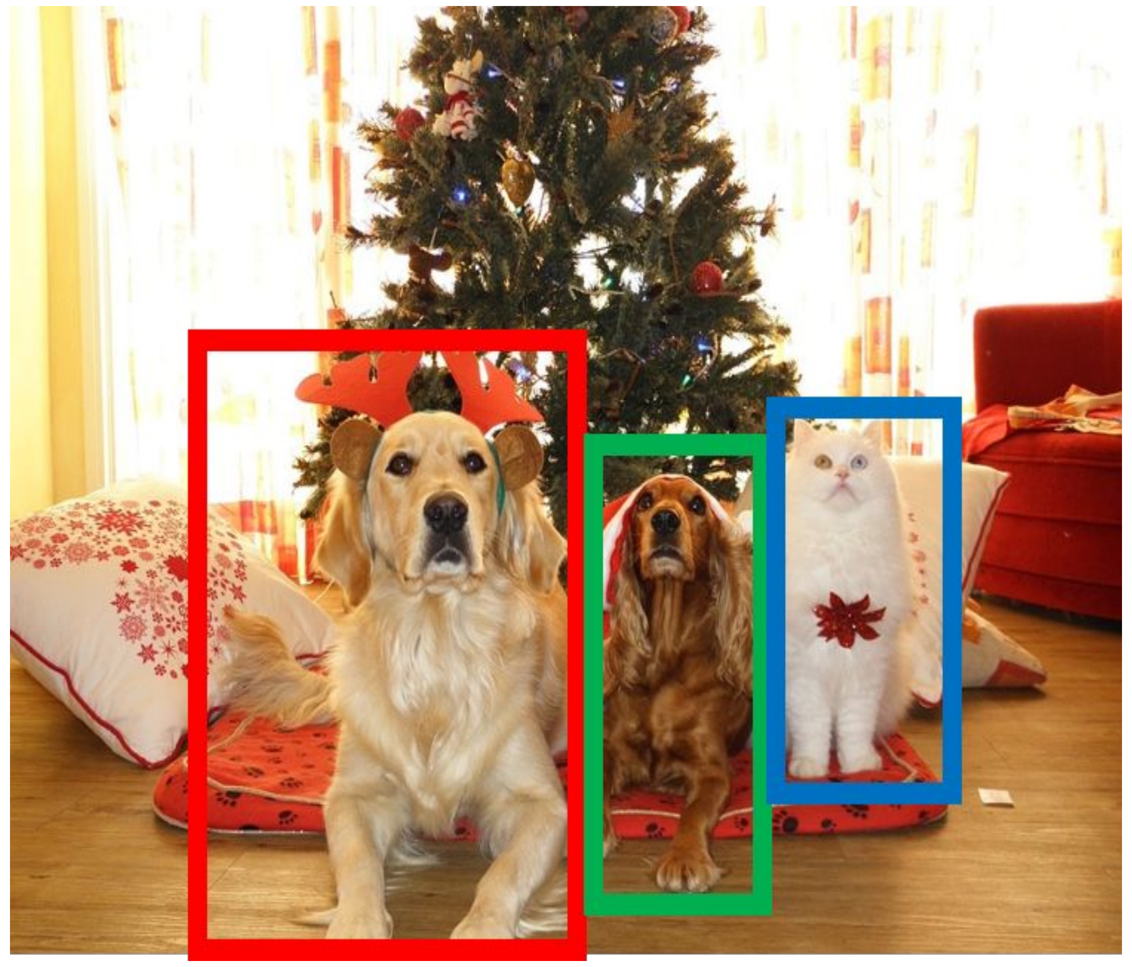
Semantic Segmentation



GRASS, CAT, TREE, SKY

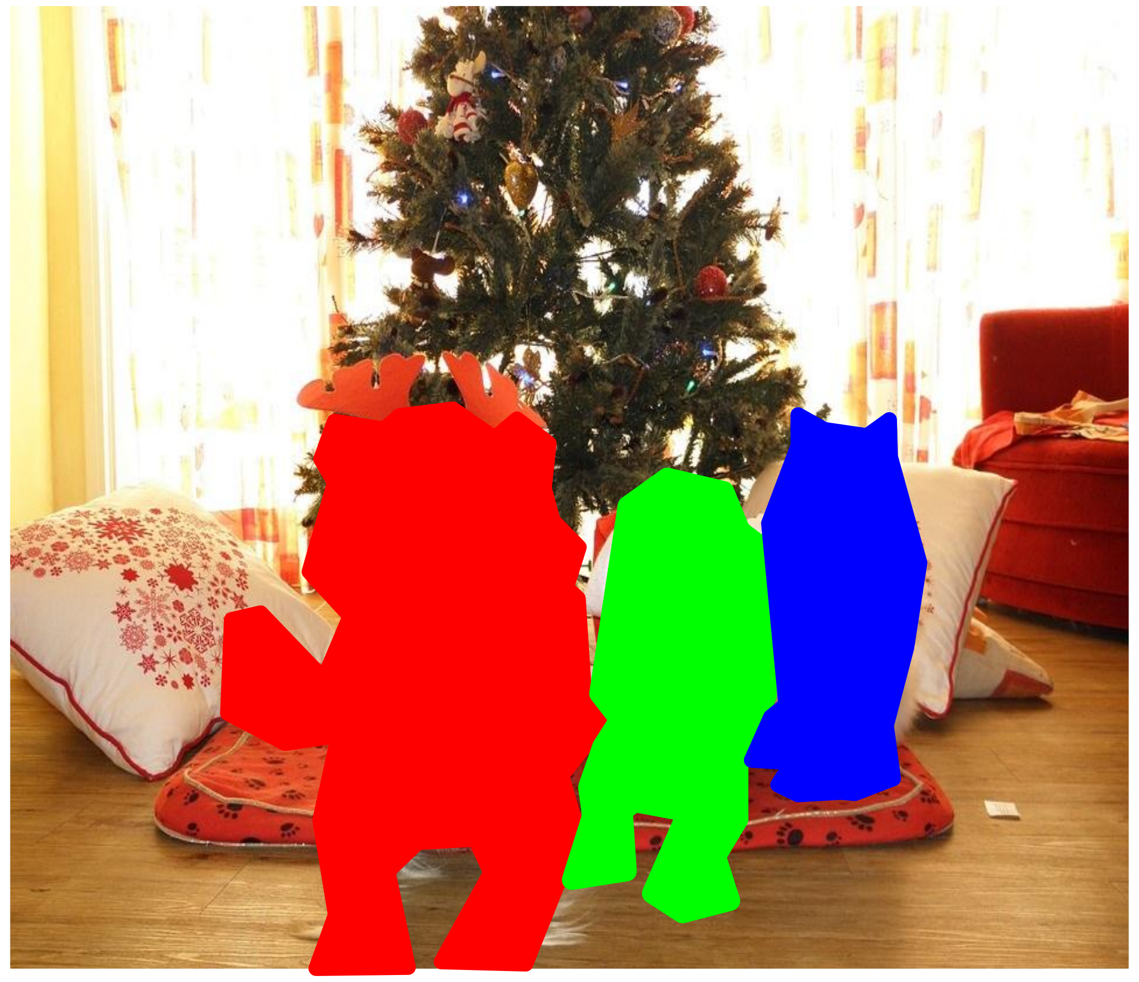
No objects, just pixels

Object Detection



DOG, DOG, CAT

Instance Segmentation



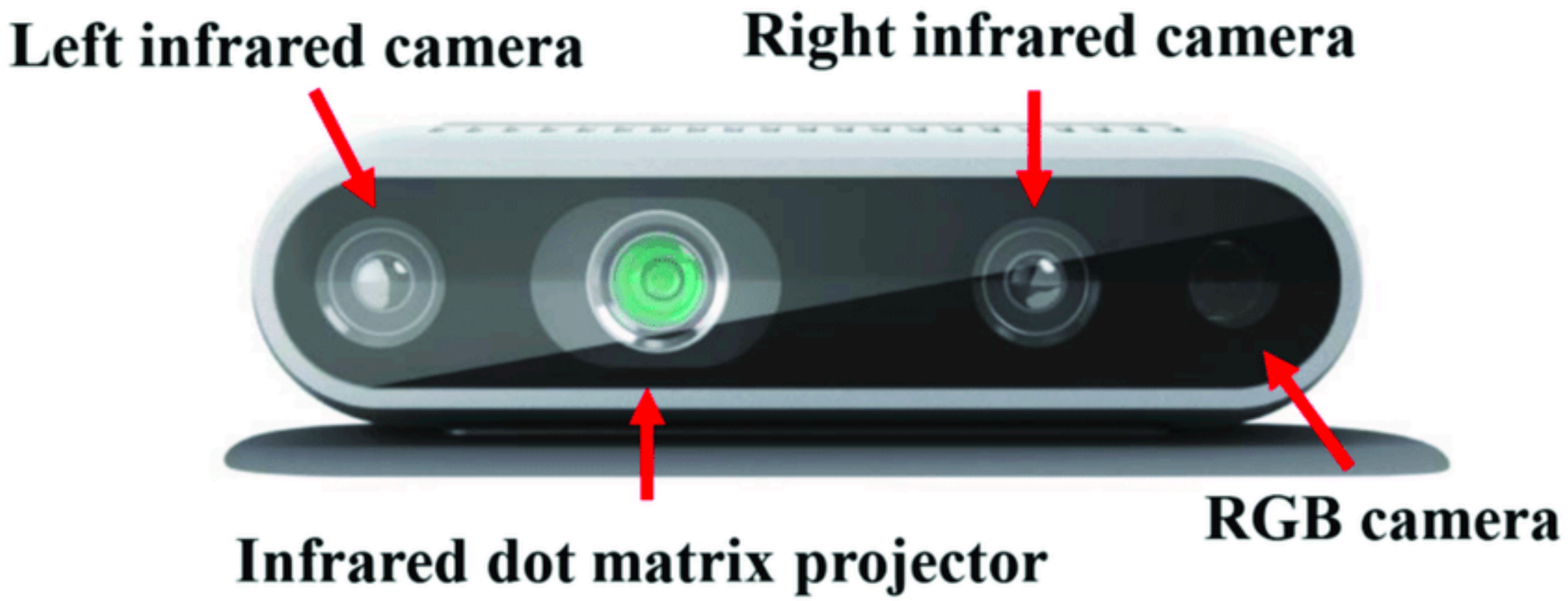
DOG, DOG, CAT

Multiple Object

Slides from Stanford CS231N: Object Detection and Image Segmentation

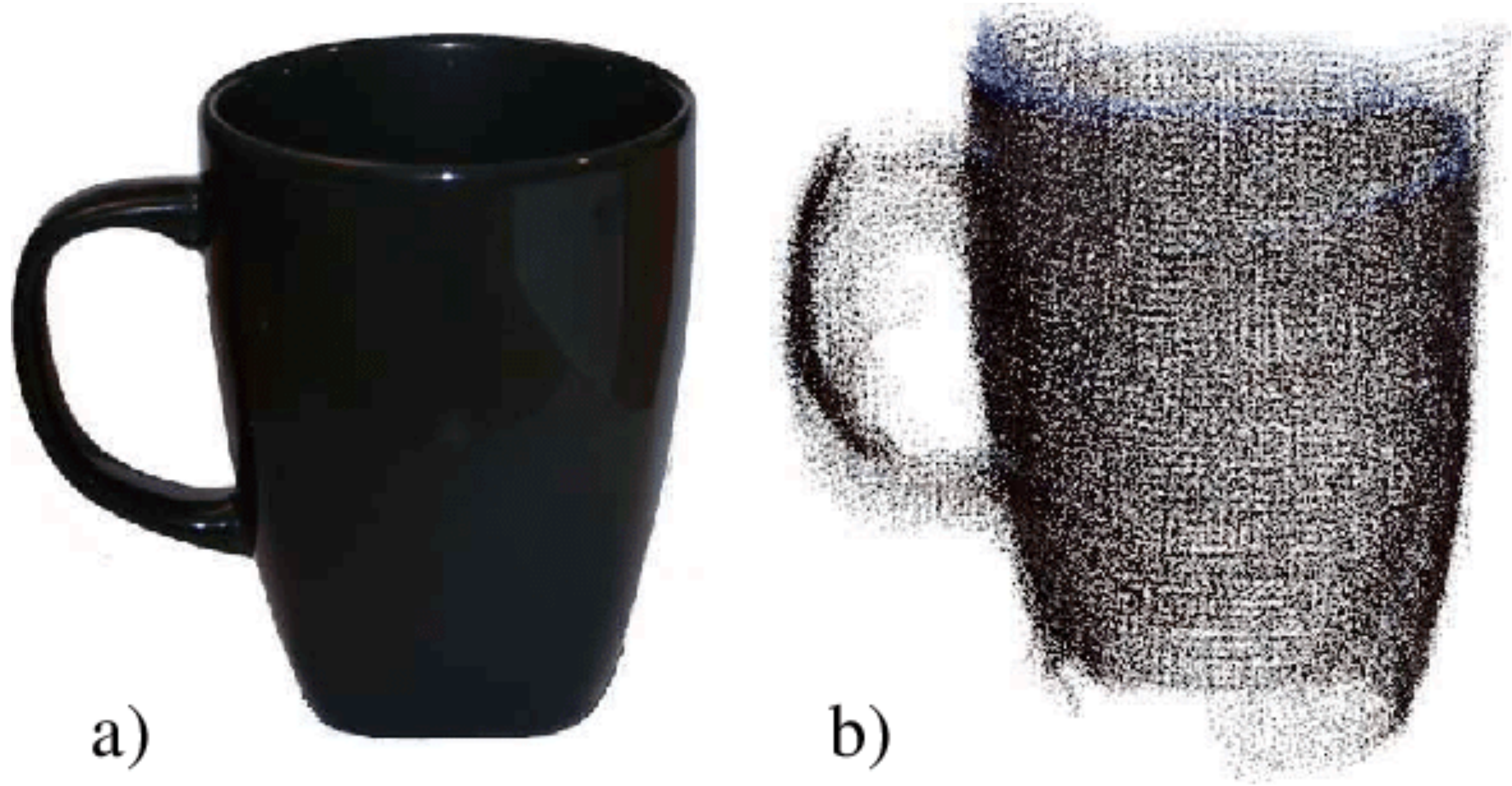
But manipulating objects require 3D reasoning!

Depth cameras give us 3D information!



Add Source
2D
3D
Intel RealSense D435I
3.2
File: "20210622_173905.bag"
Speed: x1
0:00:03
0:00:26
Pause
Reset
Lock
Source
Texture
Shading
Measure
Export
Stereo Module
on
Resolution: 848 x 480
Frame Rate (FPS): 30
Infrared 2 Y8
Infrared 1 Y8
Depth Z16
Visual Preset:
0
Emitter Enabled:
1
Enable Auto Exposure:
1
Controls
Depth Visualization
Post-Processing
RGB Camera
on
Resolution: 1280 x 720
Frame Rate (FPS): 30
Color RGB8
Enable Auto Exposure:
1
Controls
Post-Processing
Motion Module
on
Gyro stream:
Format: MOTION_XYZ32F
Frame Rate (FPS): 200
Accel stream:
Format: MOTION_XYZ32F
Frame Rate (FPS): 63
Controls

Masked Depth Image -> Point Cloud



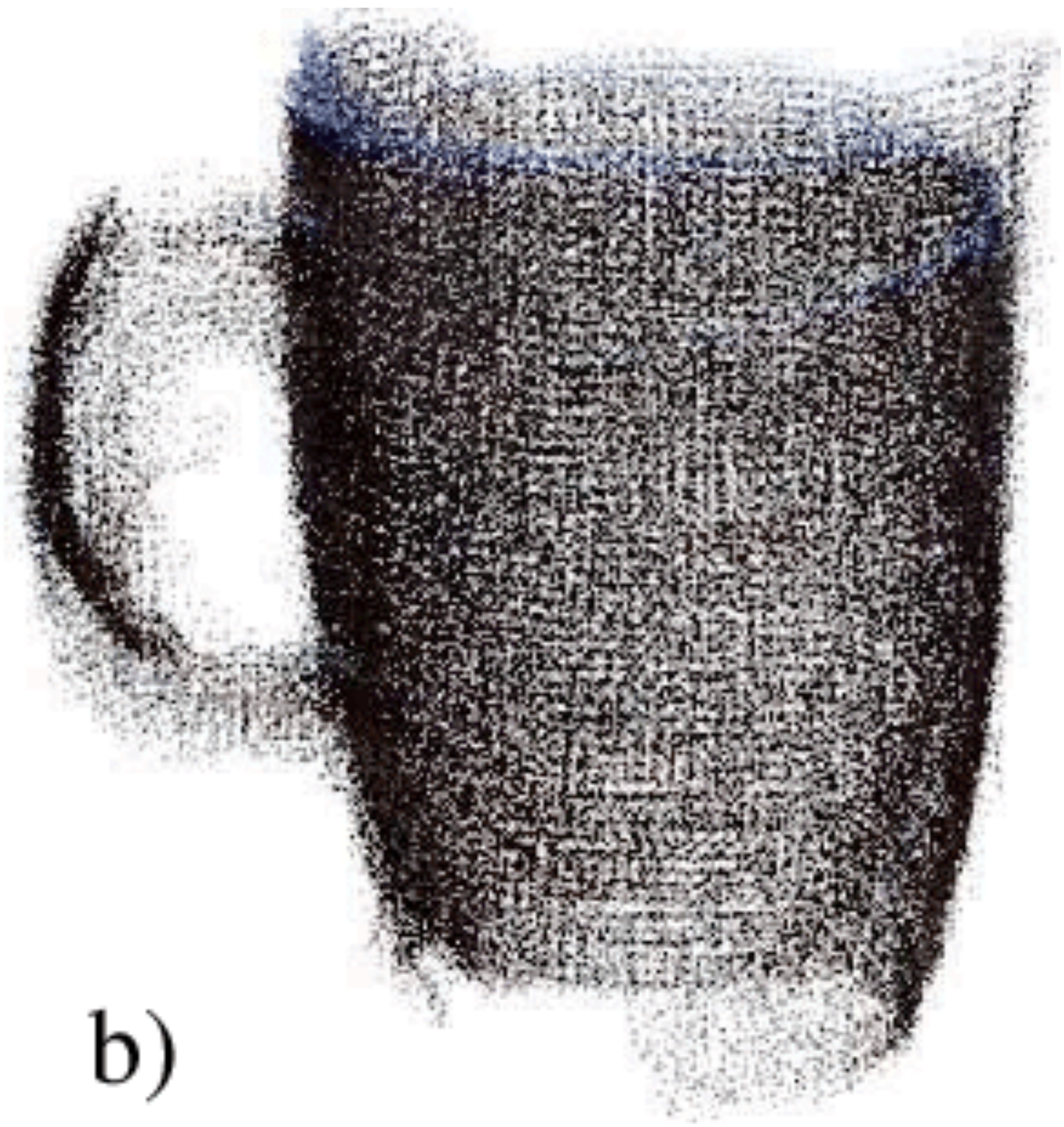
Activity!

Think-Pair-Share!

Think (30 sec): Given a point cloud of an object, how would you learn where to grasp it? What are some informative features?

Pair: Find a partner

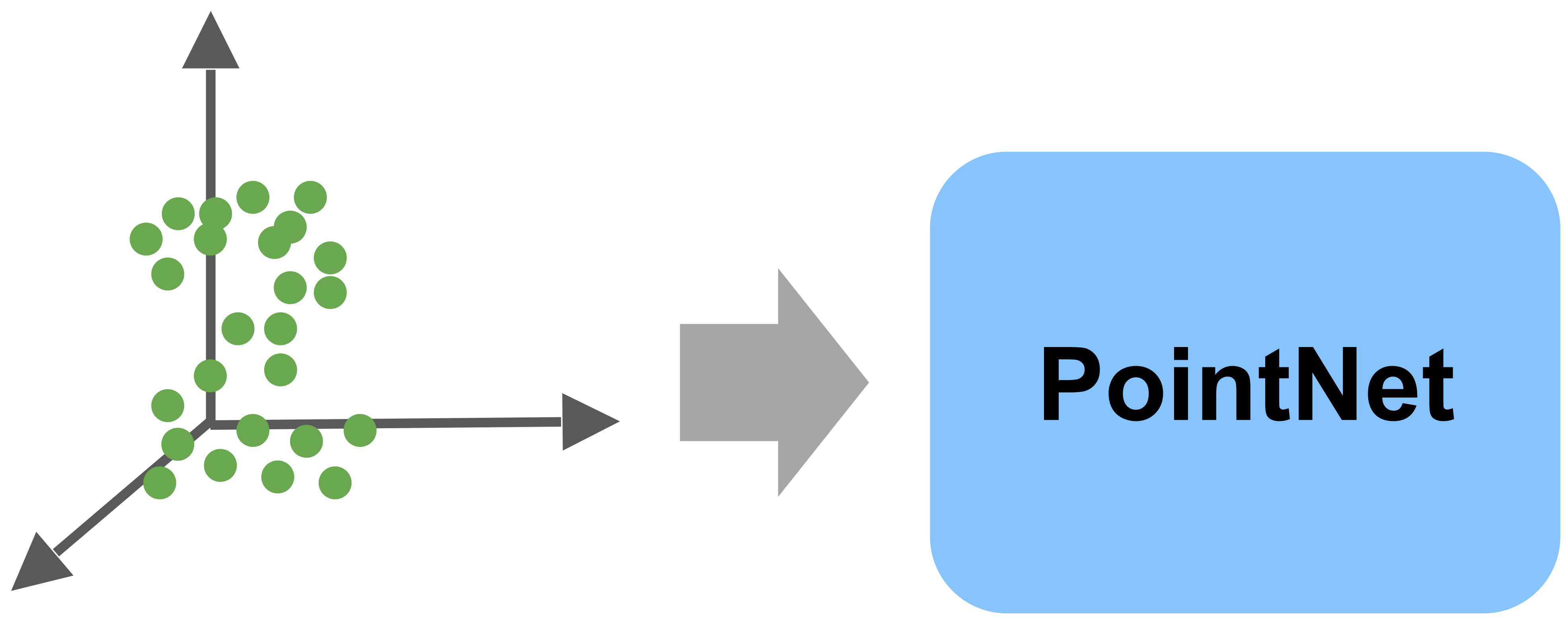
Share (45 sec): Partners exchange ideas


b)

Can we learn effective feature learning directly on point clouds?

PointNet

End-to-end learning for **scattered, unordered** point data

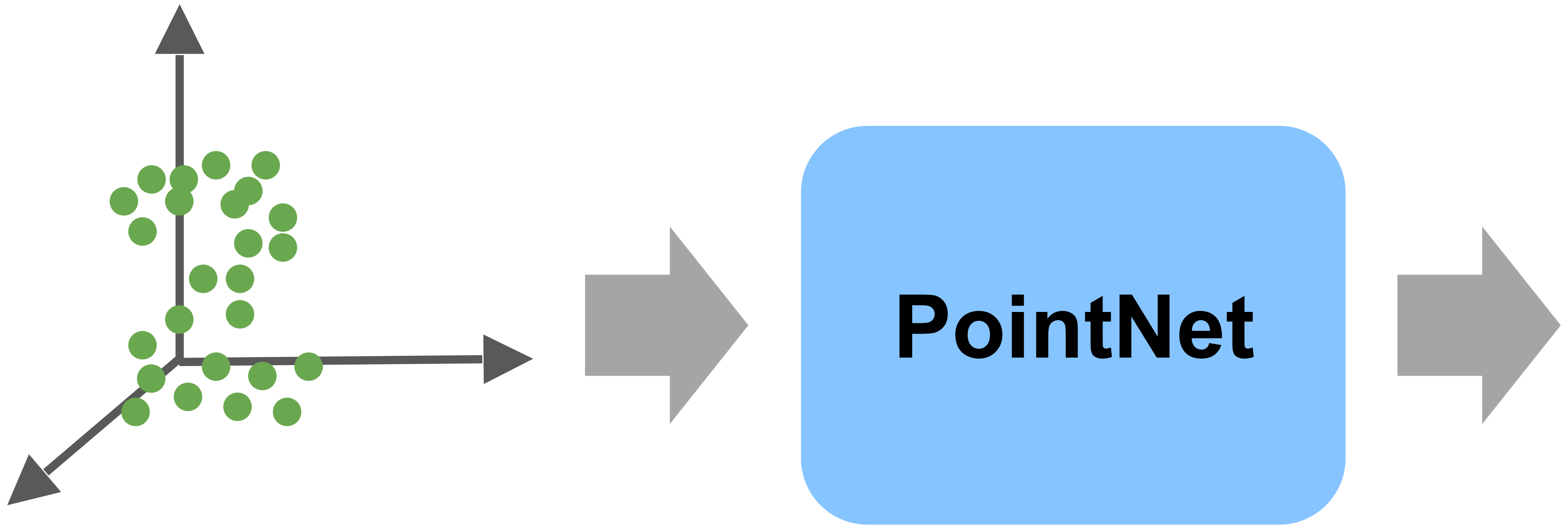


Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

PointNet

End-to-end learning for **scattered, unordered** point data

Unified framework for various tasks



Object Classification

Object Part Segmentation

Semantic Scene Parsing

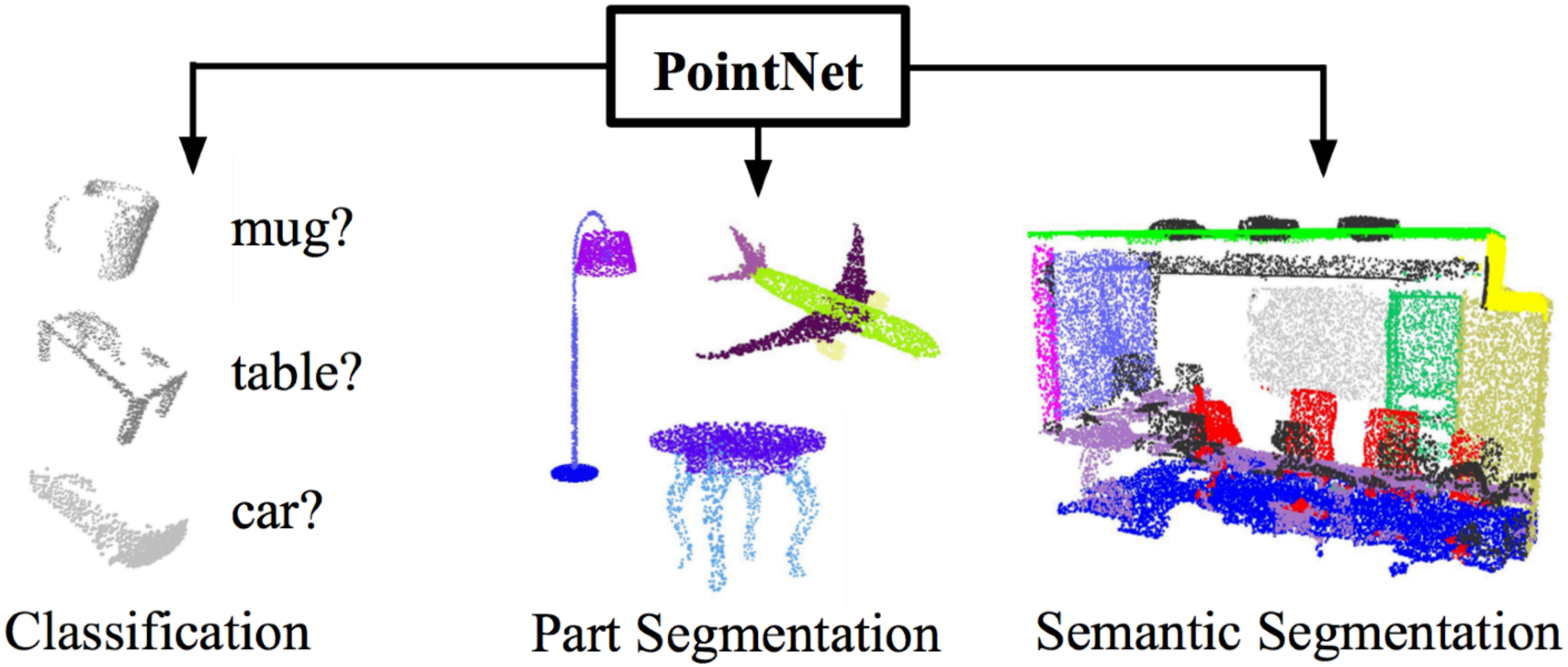
...

Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

PointNet

End-to-end learning for **scattered, unordered** point data

Unified framework for various tasks



Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

Why is learning with point clouds challenging?

Two Challenges

Challenge 1: **Unordered** point set as input

Model needs to be invariant to N! permutation

Challenge 2: Invariance under **geometric** transformations

Point cloud rotation should not alter classification results

Two Challenges

Challenge 1: **Unordered** point set as input

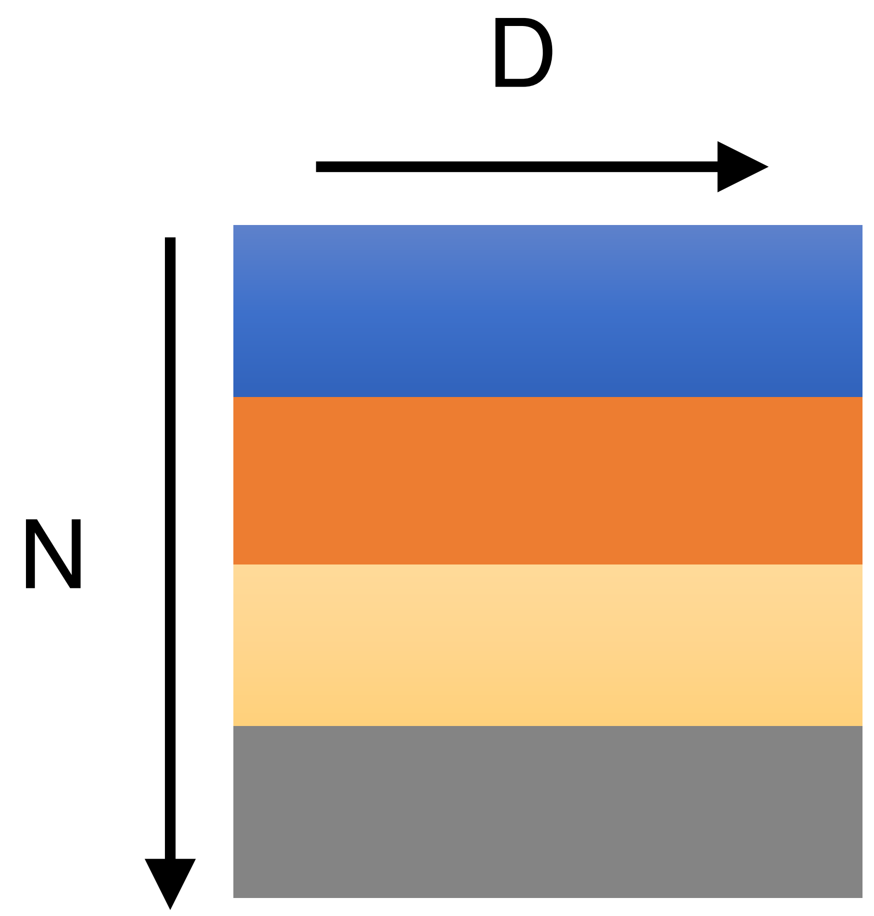
Model needs to be invariant to N! permutation

Challenge 2: Invariance under **geometric** transformations

Point cloud rotation should not alter classification results

Unordered Input

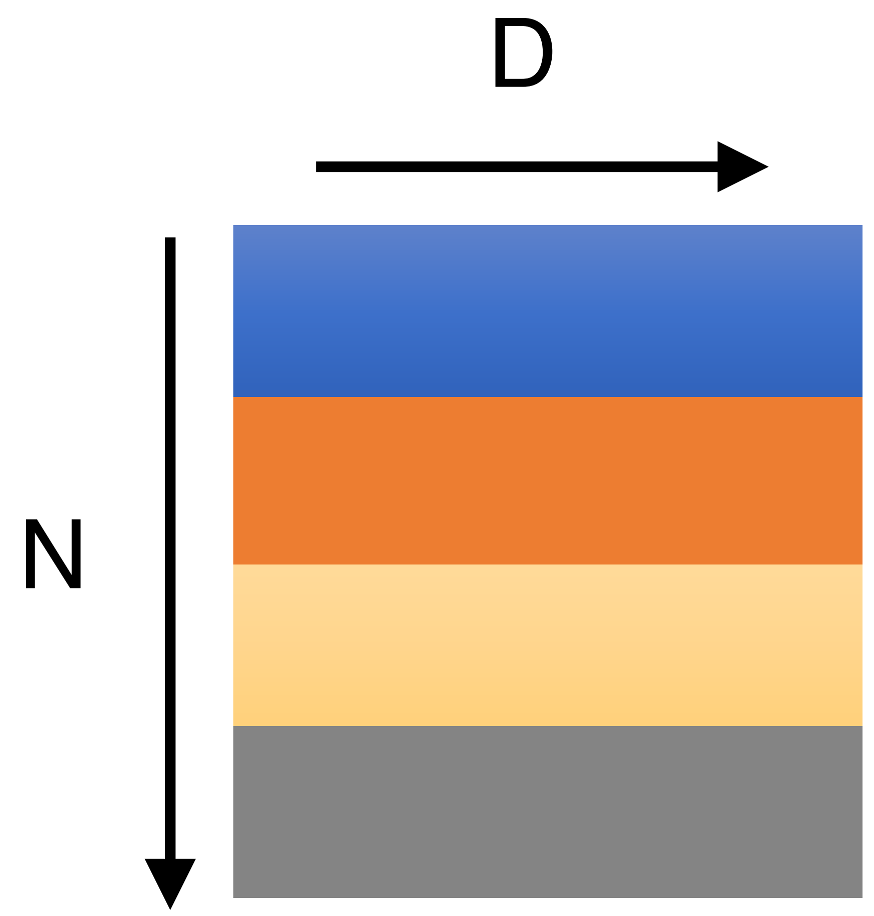
Point cloud: N **orderless** points, each represented by a D dim vector



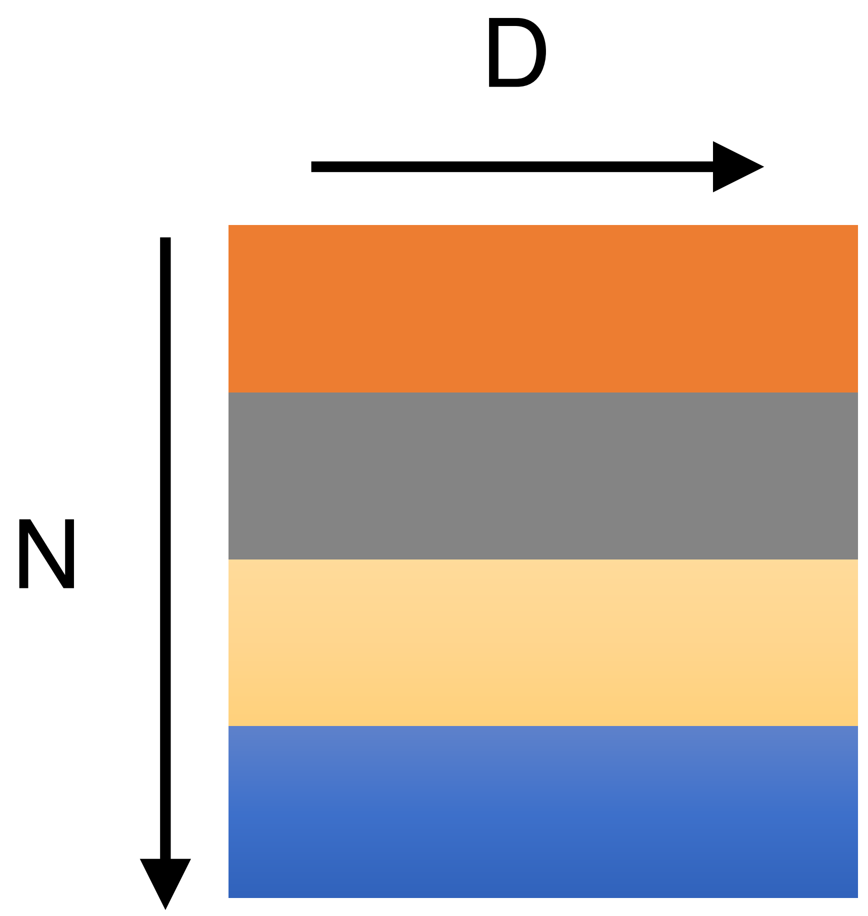
Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

Unordered Input

Point cloud: N **orderless** points, each represented by a D dim vector



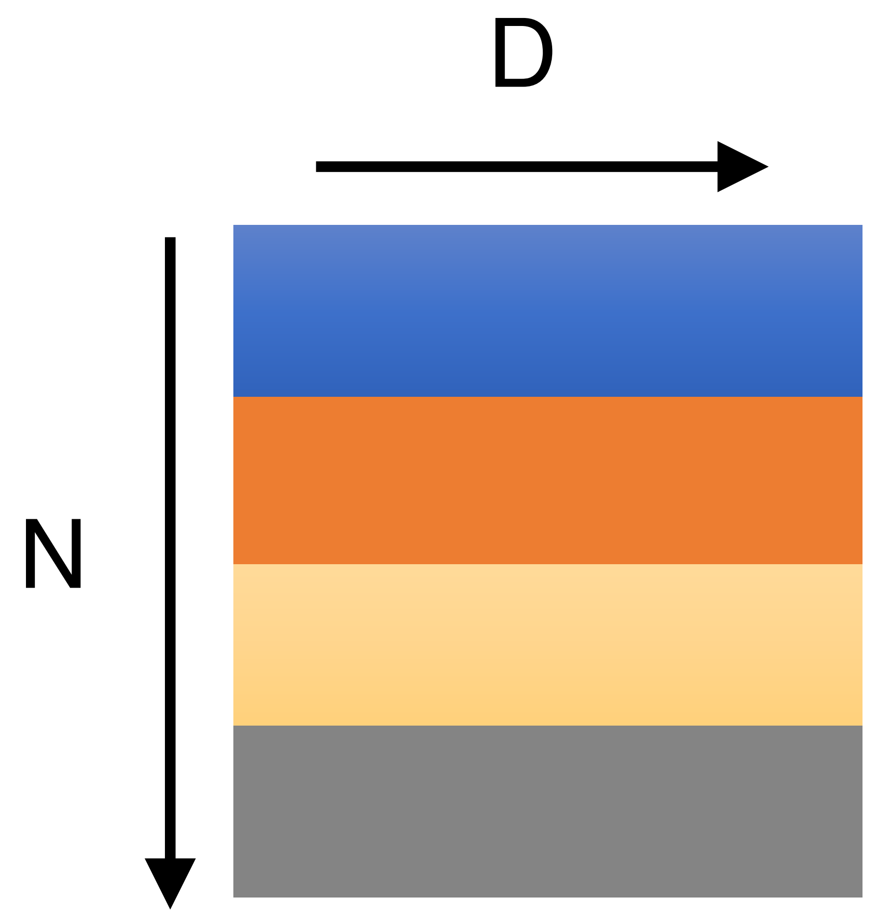
represents the same **set** as



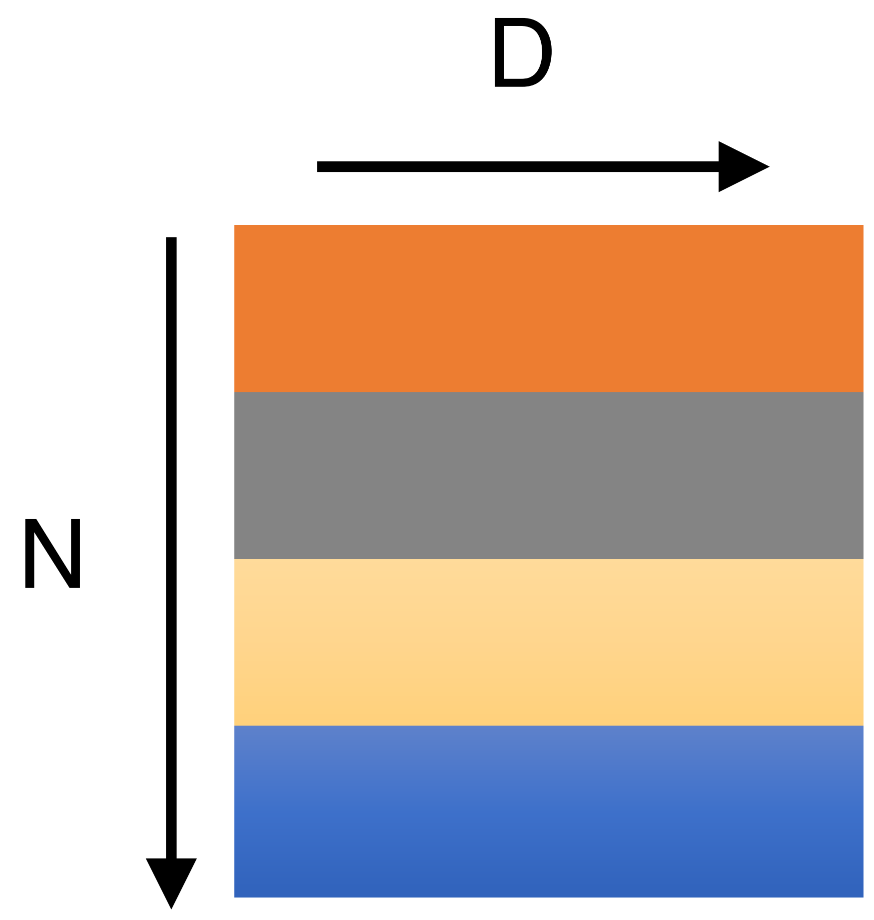
Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

Unordered Input

Point cloud: N **orderless** points, each represented by a D dim vector



represents the same **set** as



Model needs to be invariant to N! permutations

Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

Permutation Invariance: Symmetric Function

$$f(x_1, x_2, \ldots, x_n) \equiv f(x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n}), \quad x_i \in \mathbb{R}^D$$

Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

Permutation Invariance: Symmetric Function

$$f(x_1, x_2, \ldots, x_n) \equiv f(x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n}), \quad x_i \in \mathbb{R}^D$$

Examples:

$$f(x_1, x_2, \ldots, x_n) = \max\{x_1, x_2, \ldots, x_n\}$$

$$f(x_1, x_2, \ldots, x_n) = x_1 + x_2 + \ldots + x_n$$

…

Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

Permutation Invariance: Symmetric Function

$$f(x_1, x_2, \ldots, x_n) \equiv f(x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n}), \quad x_i \in \mathbb{R}^D$$

Examples:

$$f(x_1, x_2, \ldots, x_n) = \max\{x_1, x_2, \ldots, x_n\}$$

$$f(x_1, x_2, \ldots, x_n) = x_1 + x_2 + \ldots + x_n$$

...

How can we construct a family of symmetric functions by neural networks?

Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

Permutation Invariance: Symmetric Function

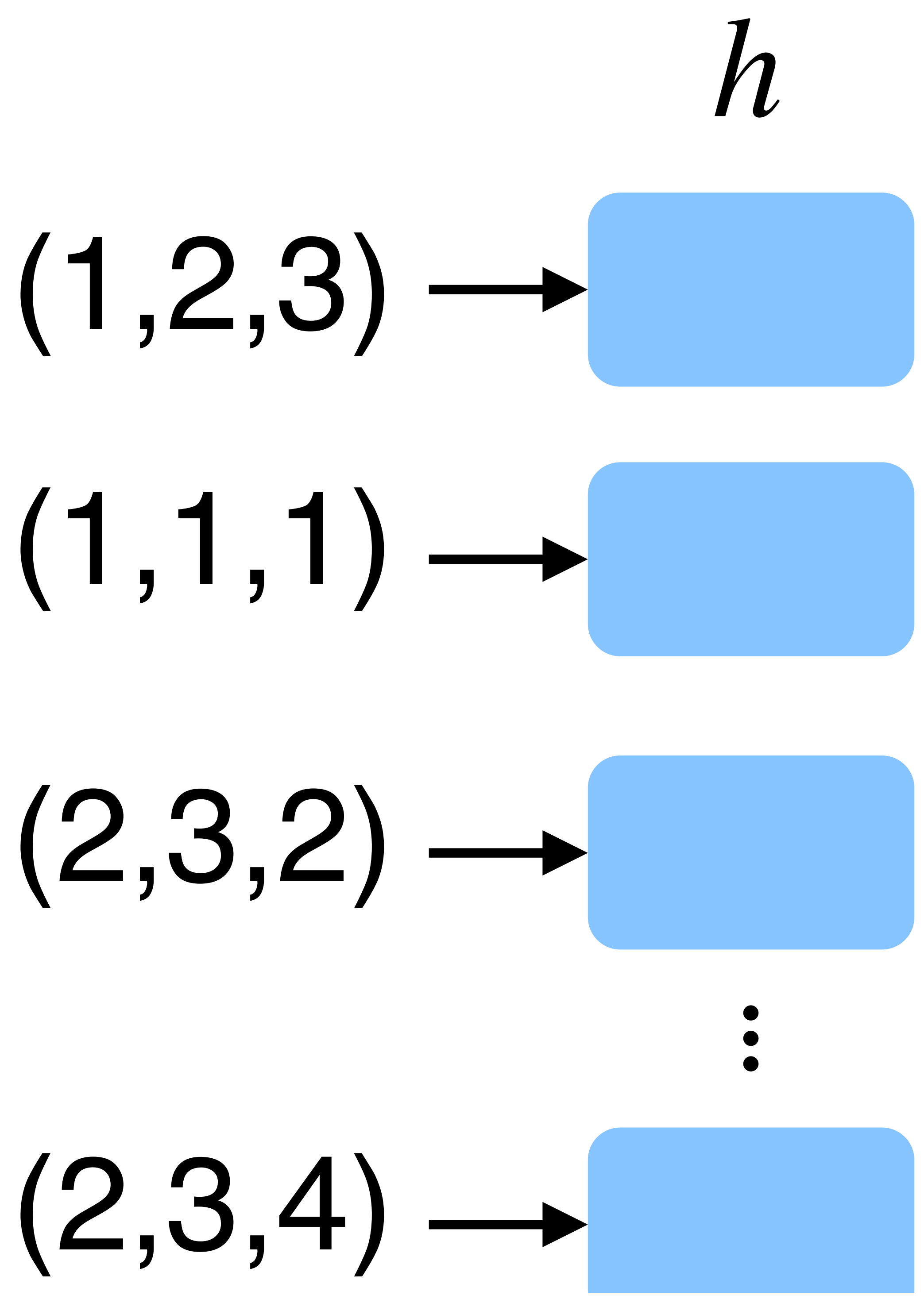
Observe:

$f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n))$ is symmetric if g is symmetric

Permutation Invariance: Symmetric Function

Observe:

$f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n))$ is symmetric if g is symmetric

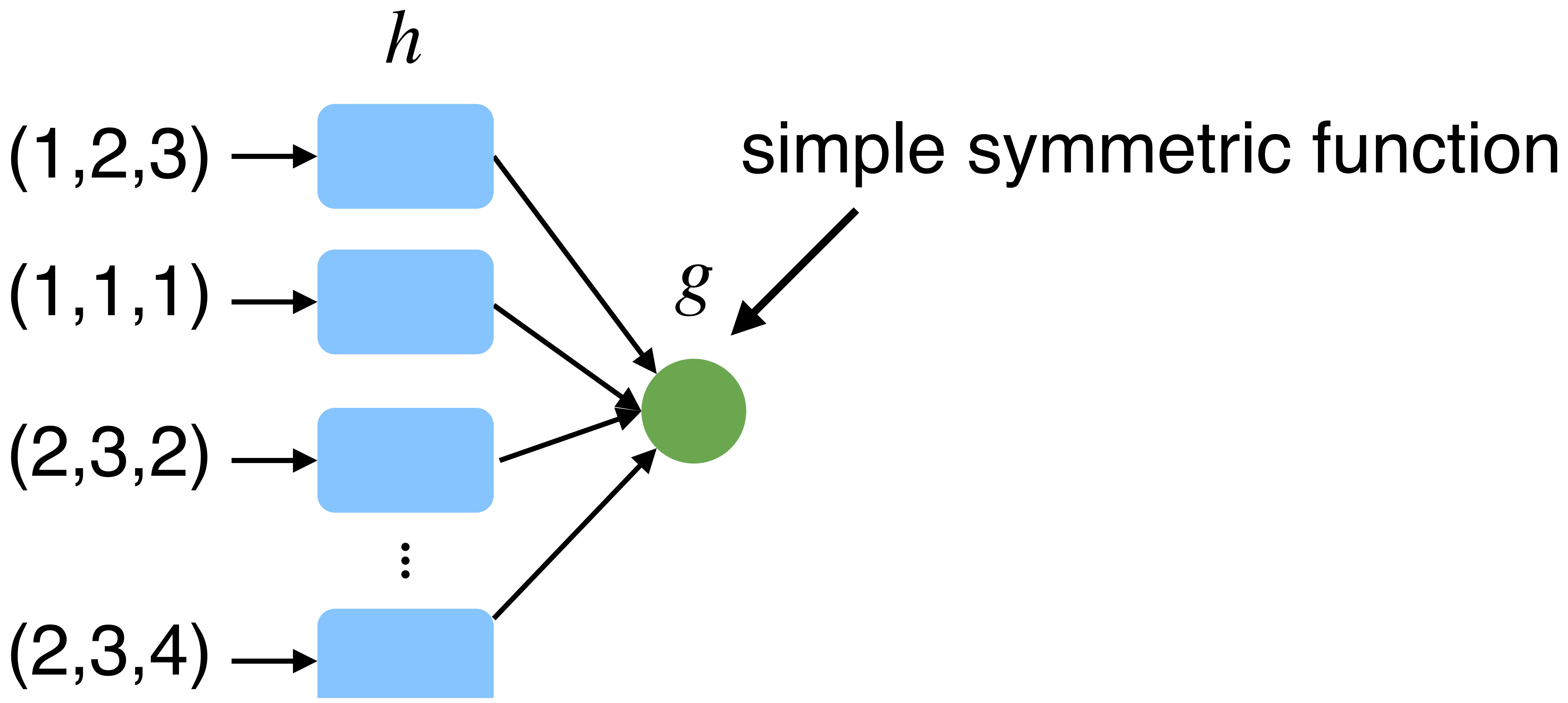


Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

Permutation Invariance: Symmetric Function

Observe:

$f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n))$ is symmetric if g is symmetric

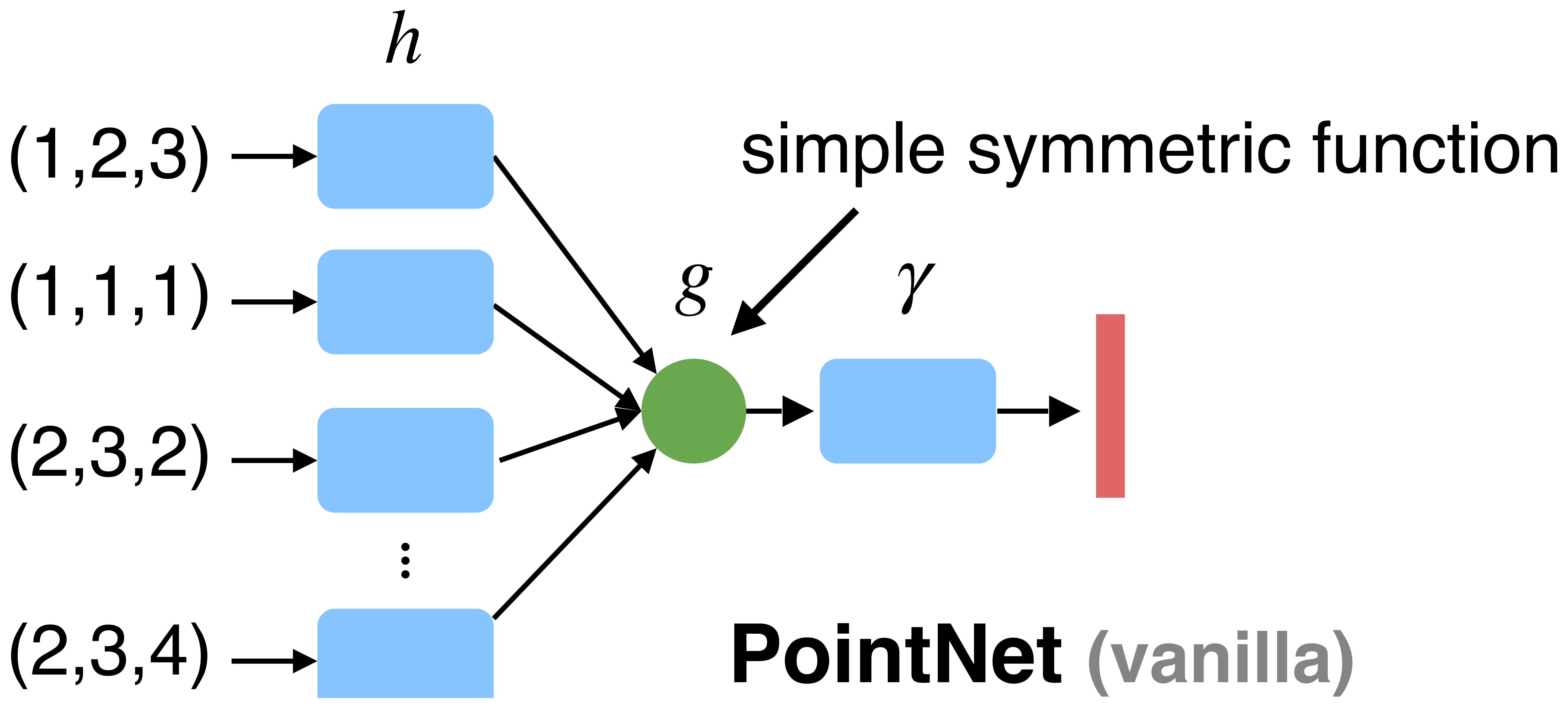


Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

Permutation Invariance: Symmetric Function

Observe:

$f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n))$ is symmetric if g is symmetric

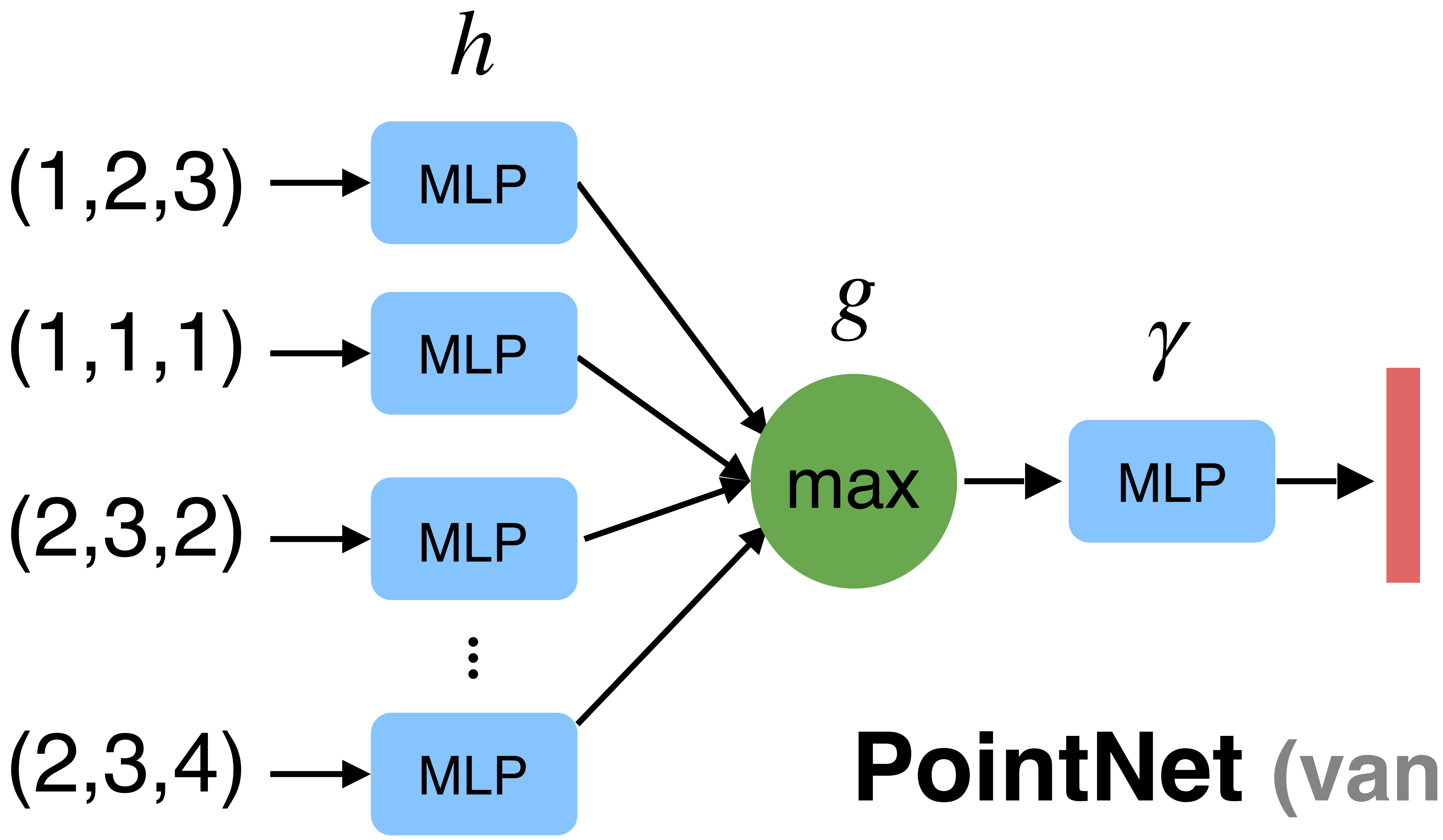


PointNet **(vanilla)**

Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

Basic PointNet Architecture

Empirically, we use **multi-layer perceptron (MLP)** and **max pooling**:



PointNet (vanilla)

Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

Two Challenges

Challenge 1: **Unordered** point set as input

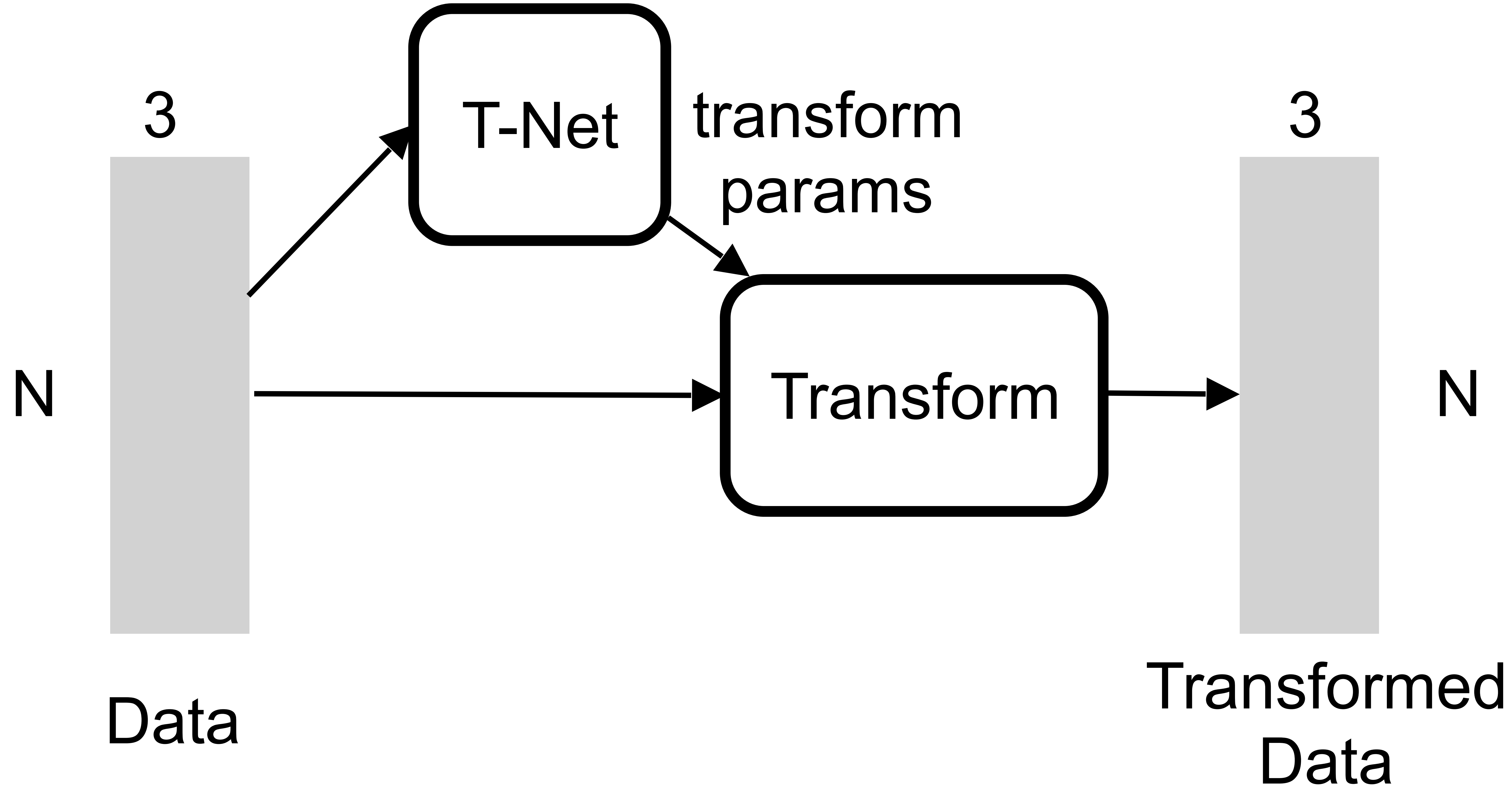
Model needs to be invariant to N! permutation

Challenge 2: Invariance under **geometric** transformations

Point cloud rotation should not alter classification results

Input Alignment by Transformer Network

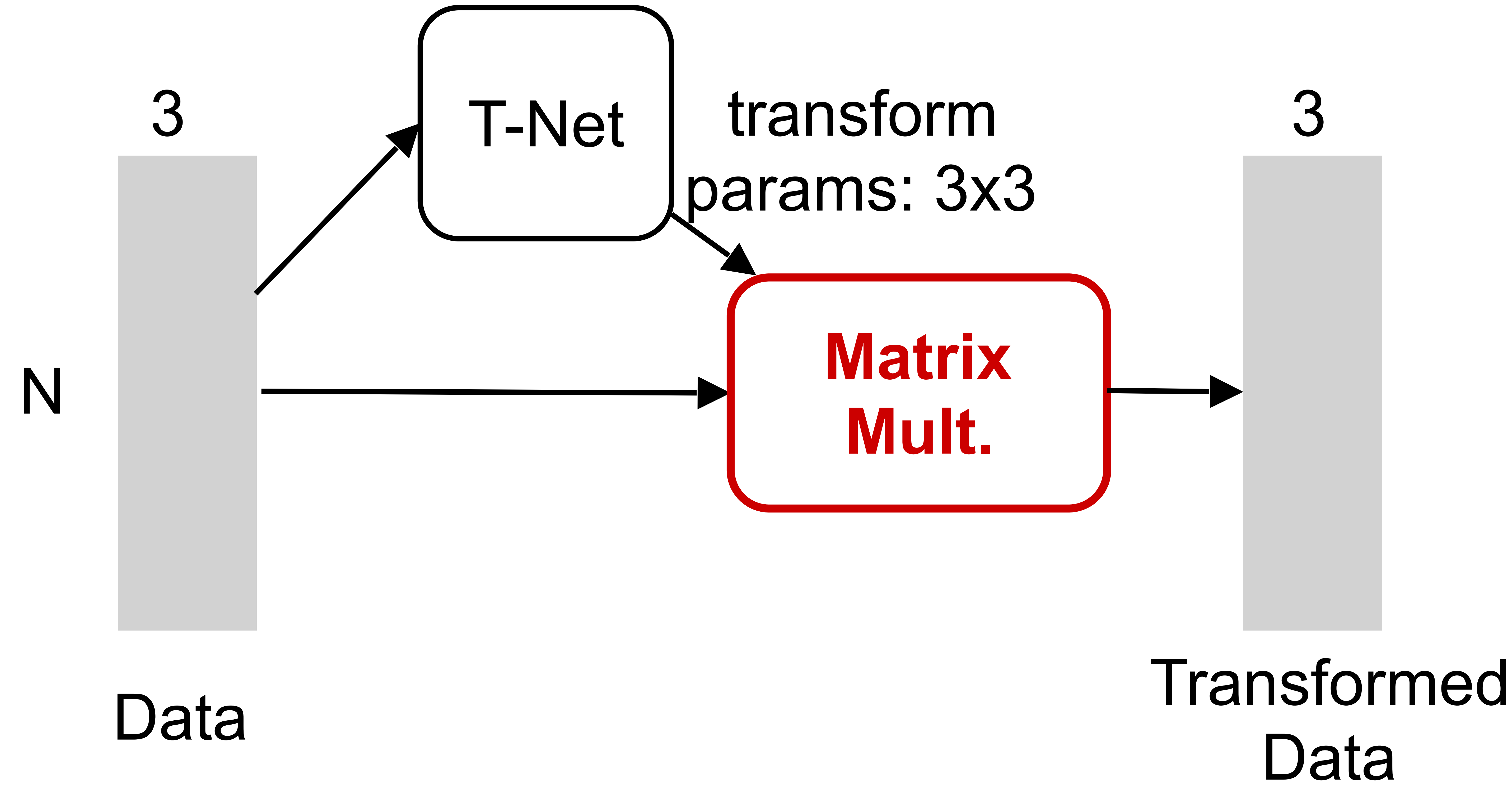
Idea: Data dependent transformation for automatic alignment



Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

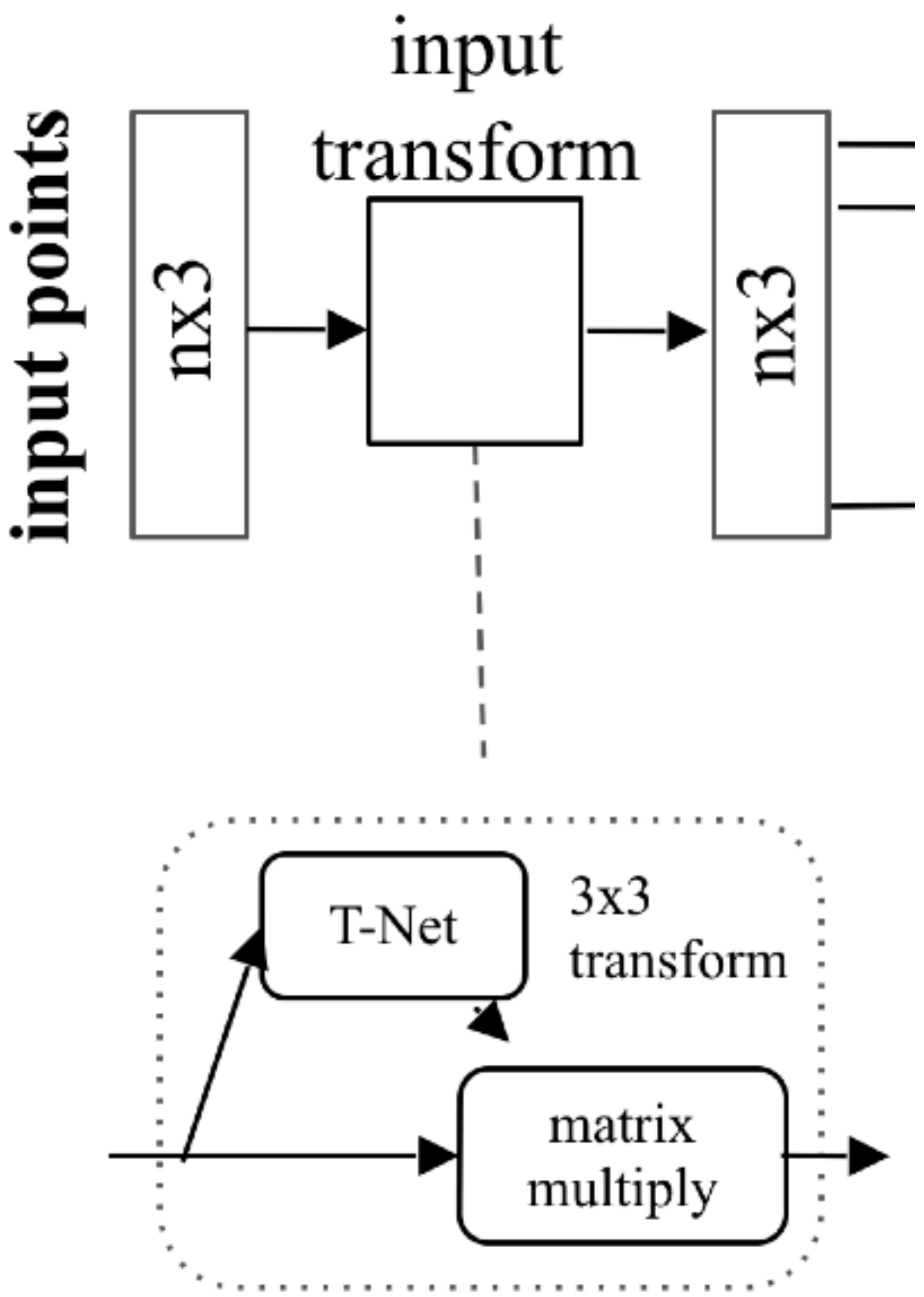
Input Alignment by Transformer Network

The transformation is just matrix multiplication!



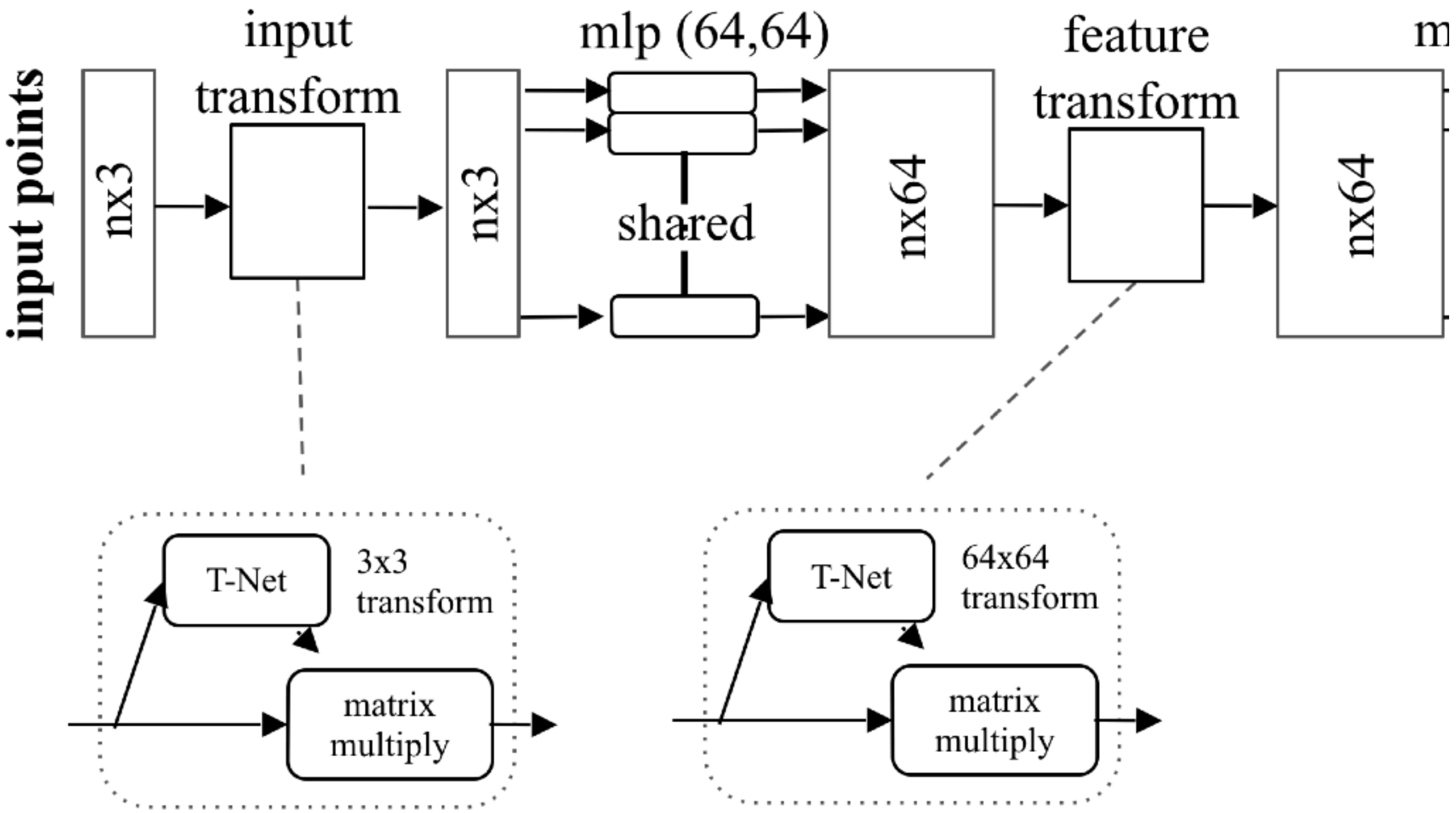
Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

PointNet Classification Network



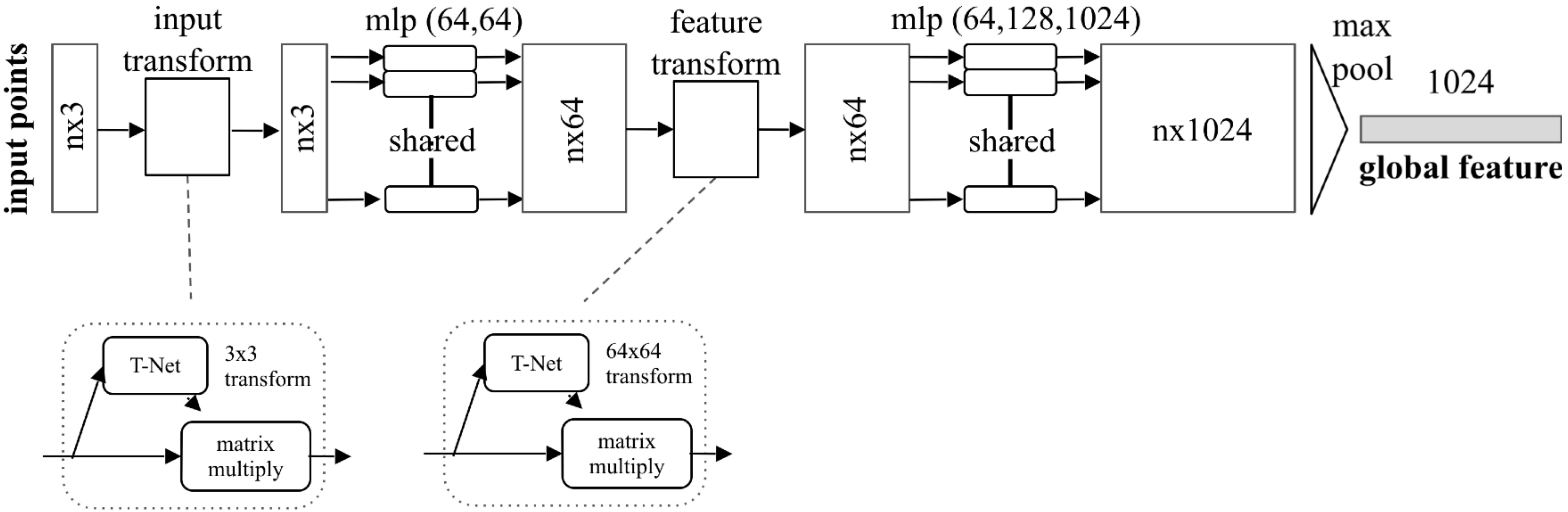
Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

PointNet Classification Network



Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

PointNet Classification Network

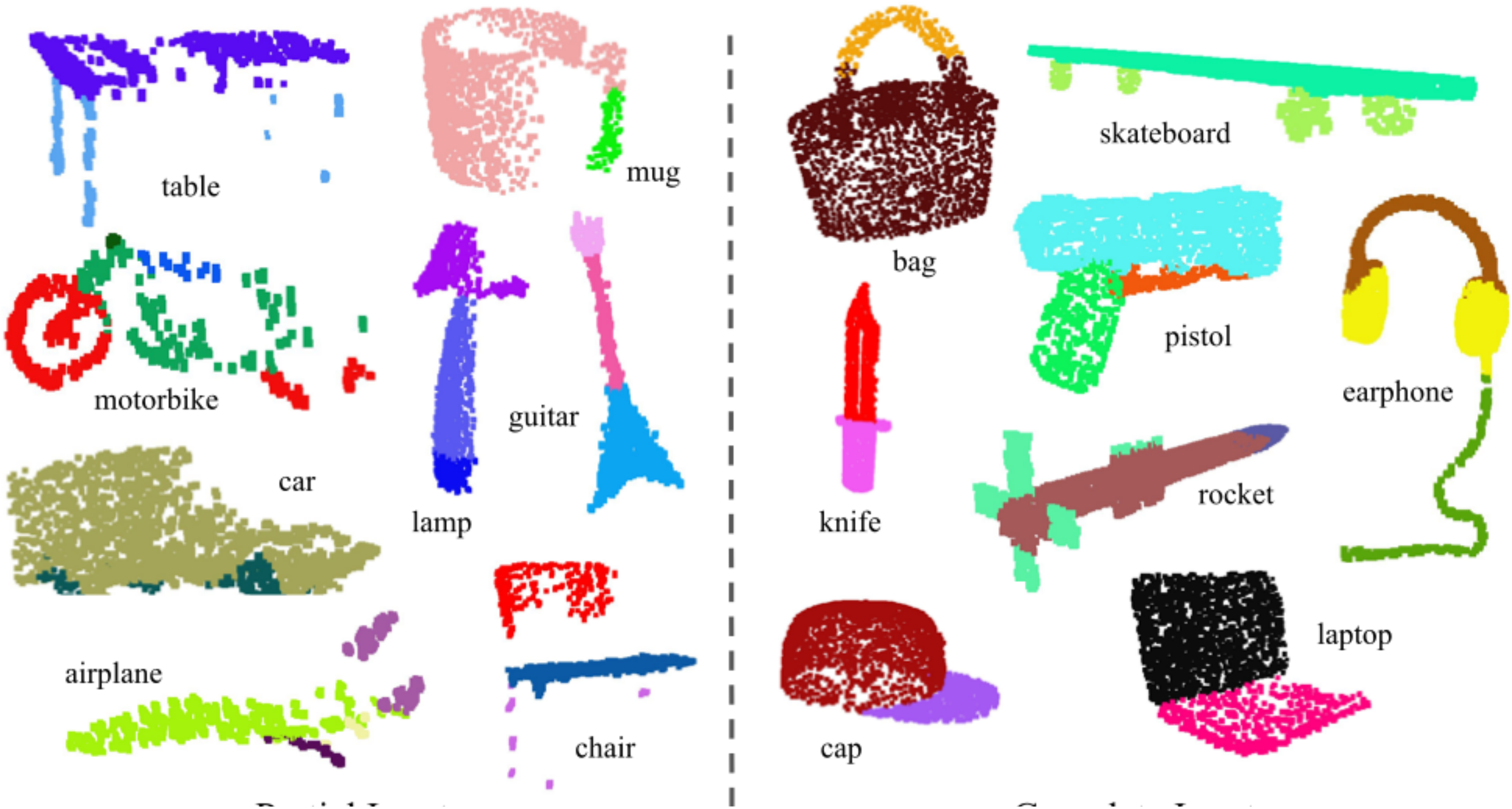


Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

PointNet Classification Network

Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

Results on Object Part Segmentation



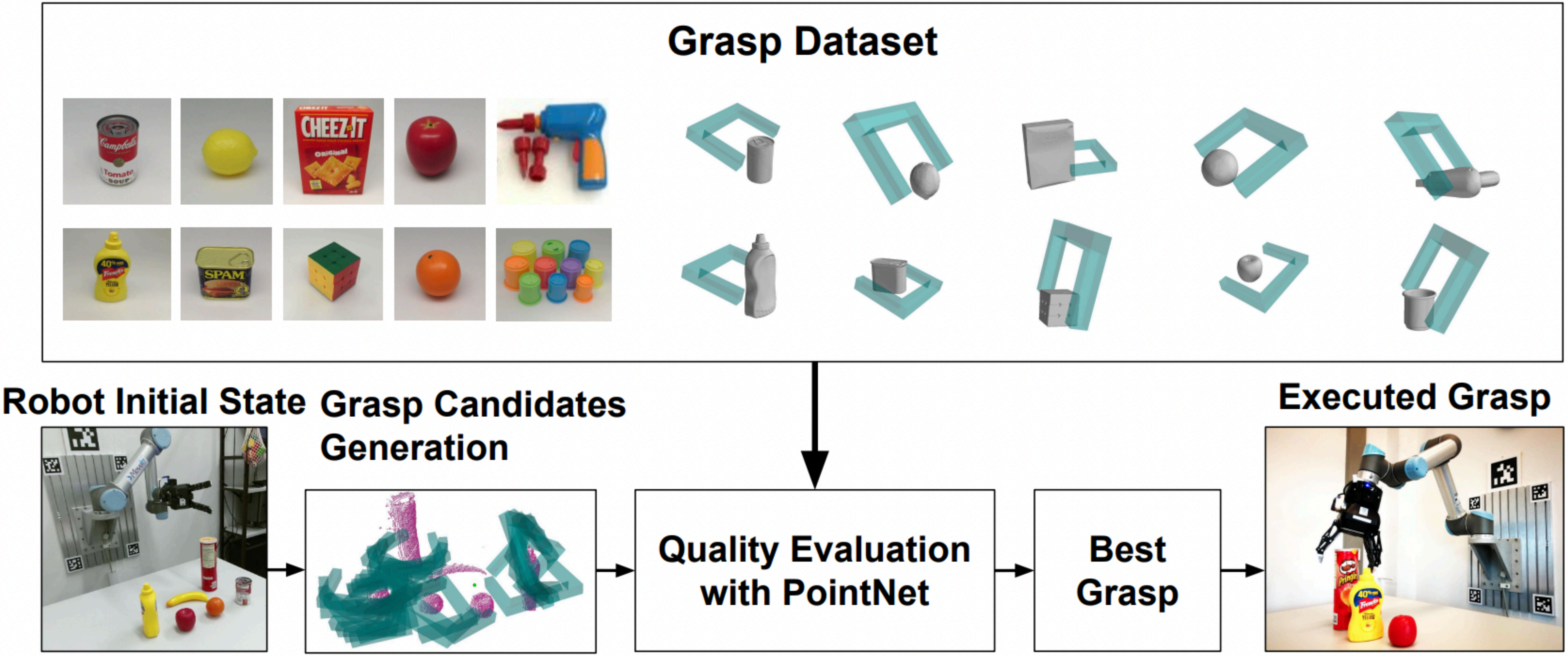
Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

Results on Semantic Scene Parsing

Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

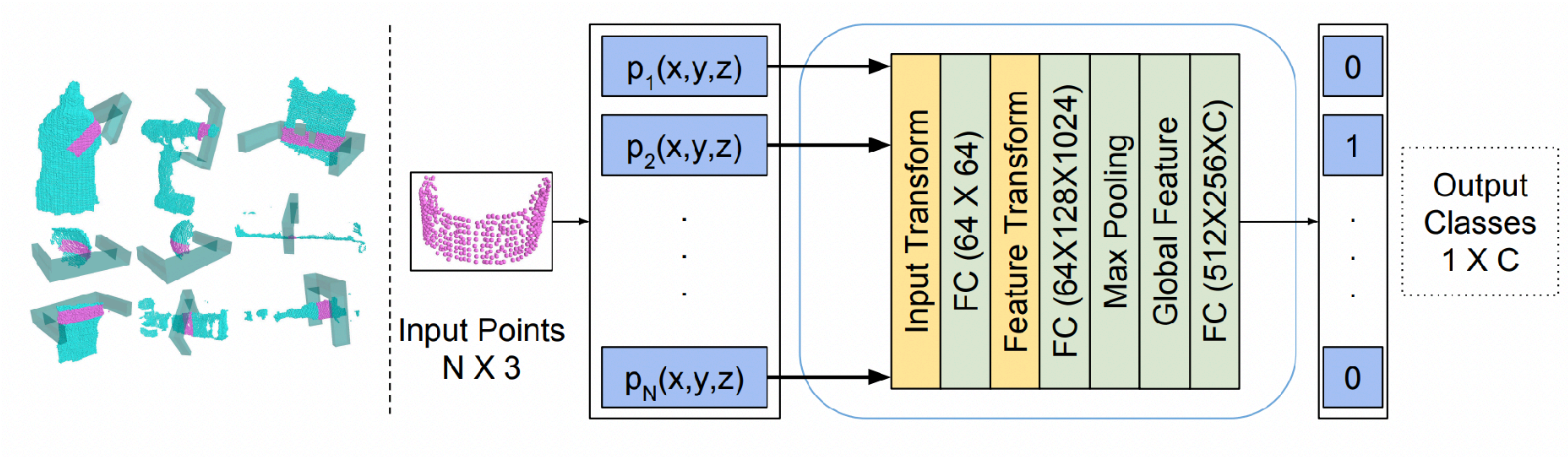
How do we use this for learning grasping?

PointNetGPD: Detecting Grasp Configurations from Point Sets



Liang et al.

PointNetGPD: Detecting Grasp Configurations from Point Sets



Liang et al.

But … what if we don't have RGBD data?

Reasons for not having depth data

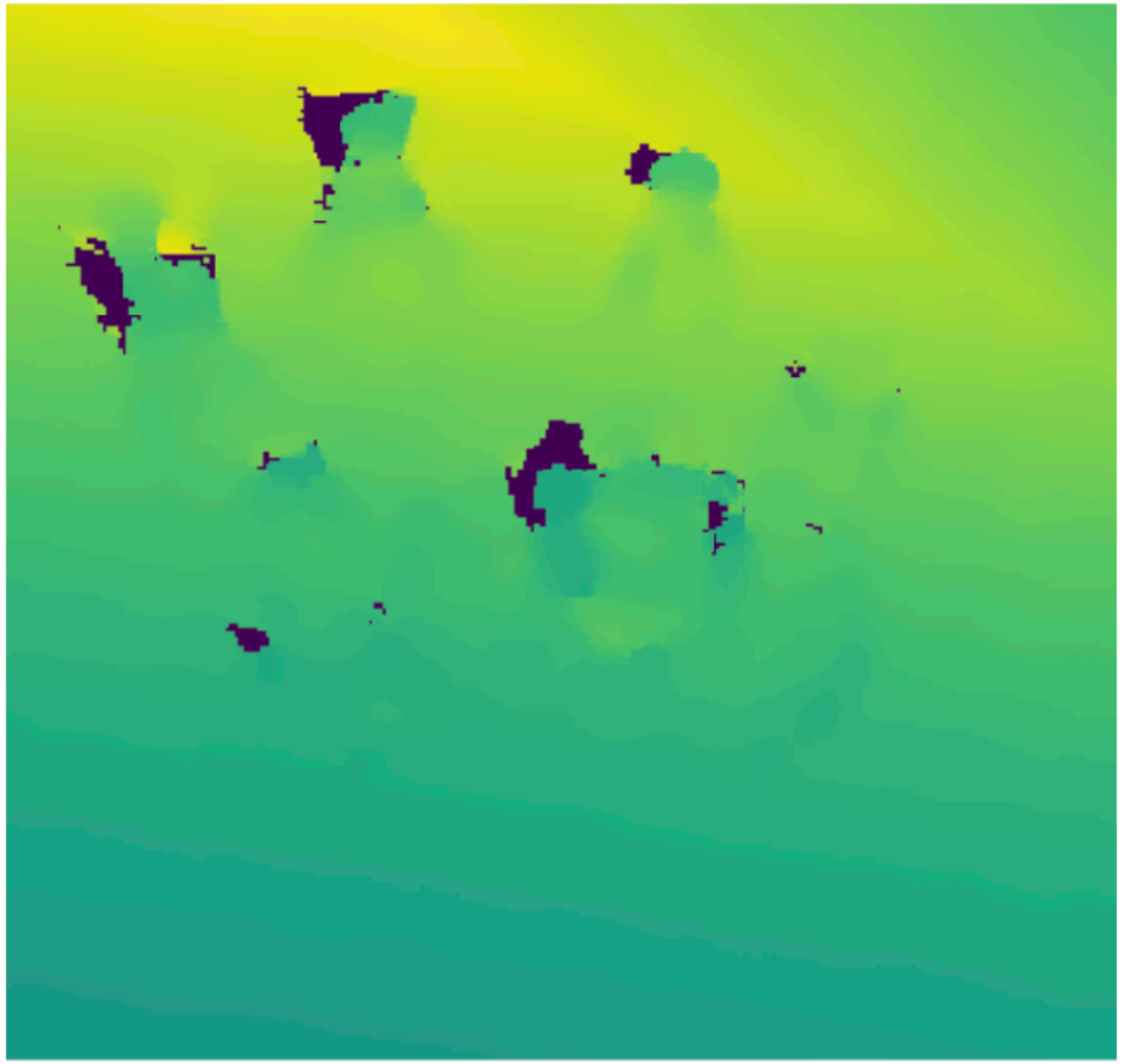
Don't have a depth sensor!

Outside in the sun / beyond maximum range

Glass or transparent objects

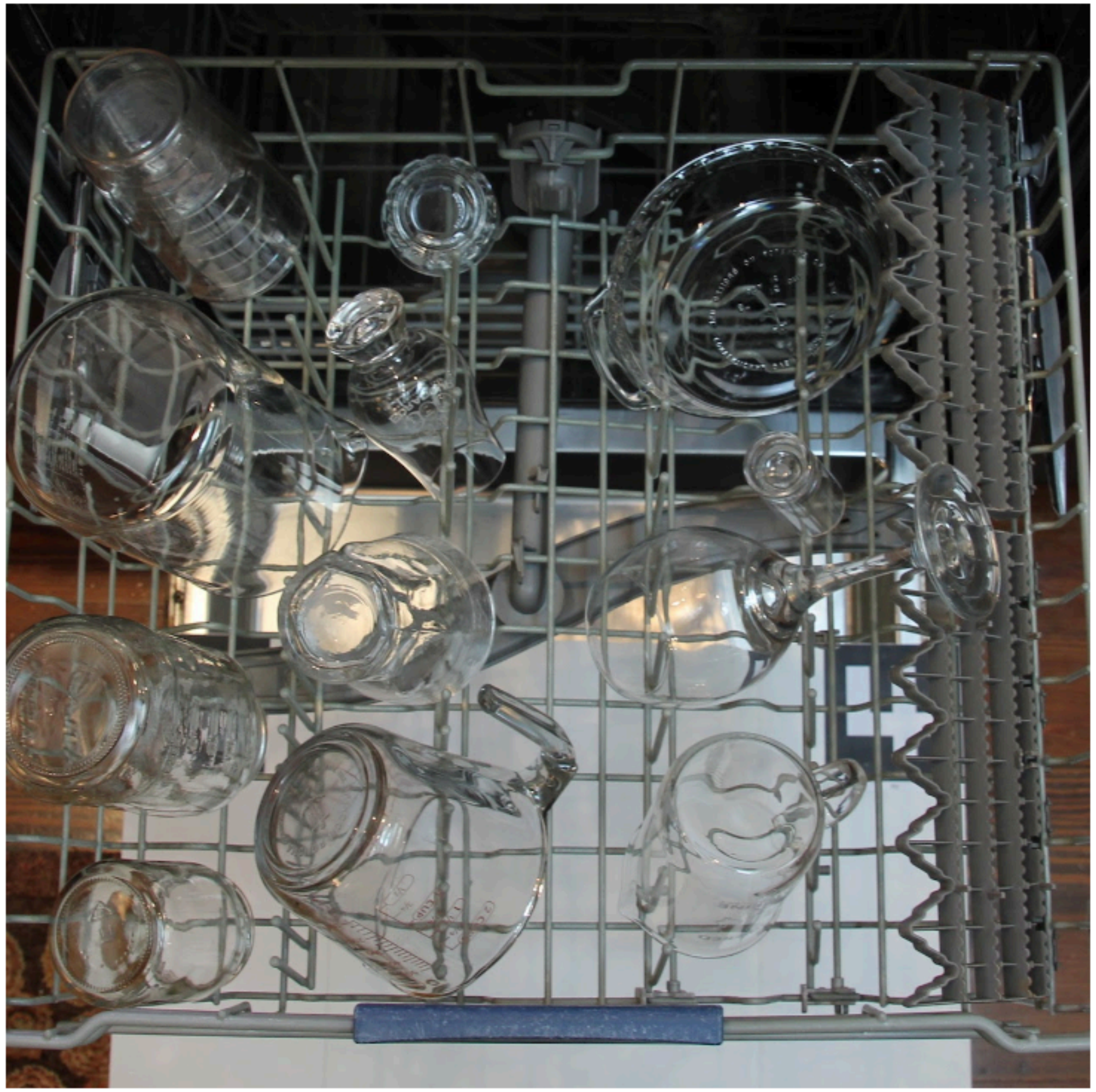
Glass or transparent objects

Real-world Scene

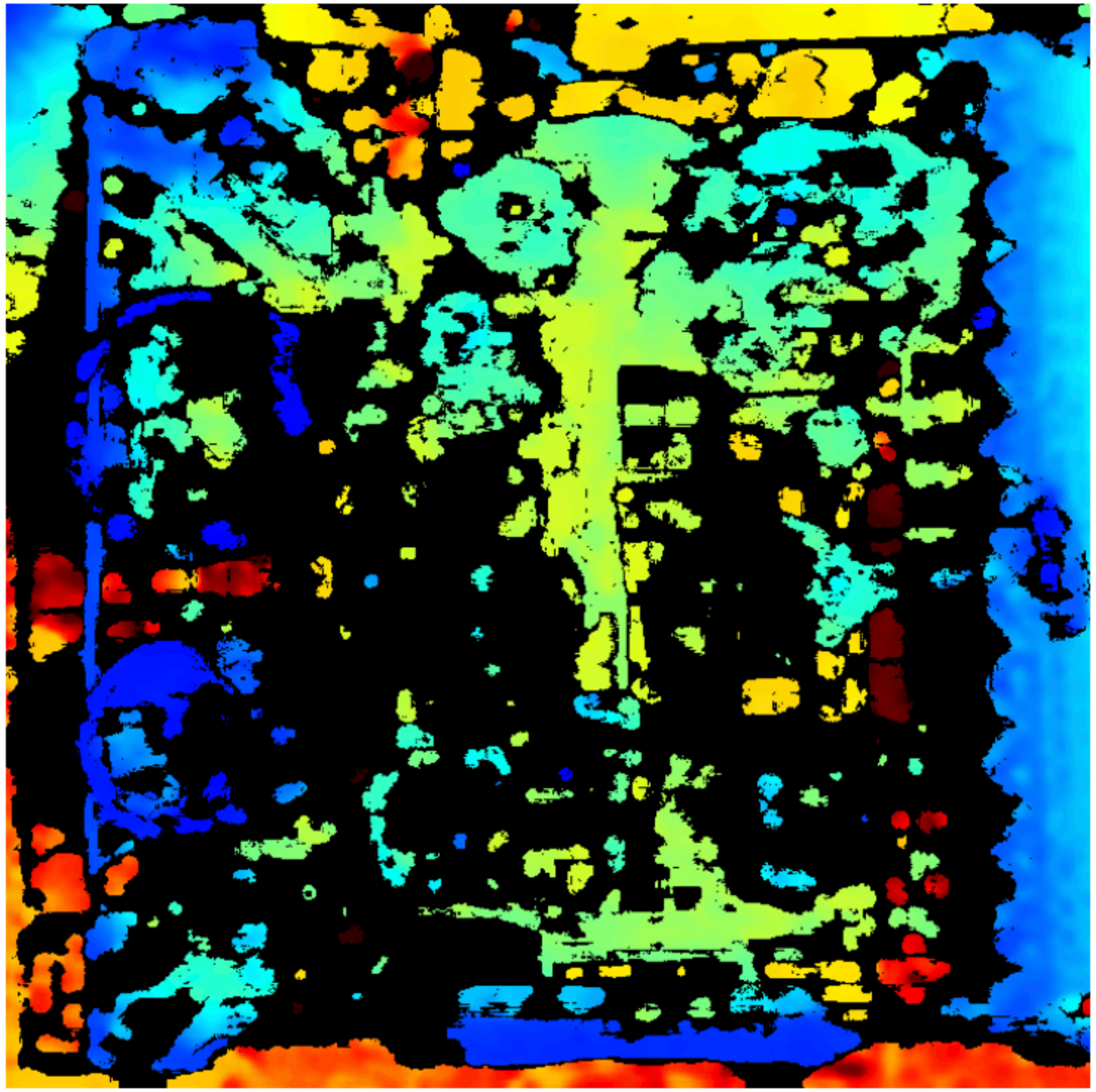


RealSense D410 Depth Image

Glass or transparent objects



Dishwasher Real-world Scene



RealSense D410 Depth Image

Let's say I just have a set of images & camera poses

$x_1, y_1, z_1, \theta_1, \phi_1$

$x_2, y_2, z_2, \theta_2, \phi_2$

$x_3, y_3, z_3, \theta_3, \phi_3$

How do we predict a 3D structure?

2D images

Camera Poses

$x_1, y_1, z_1, \theta_1, \phi_1$ $x_2, y_2, z_2, \theta_2, \phi_2$ $x_3, y_3, z_3, \theta_3, \phi_3$

3D structure

If you can predict how the object will

look from a *novel viewpoint*,

you have *implicitly* modeled the

3D structure

Let's setup a learning problem

x, y, z, θ, ϕ

Novel viewpoint

Neural
Network

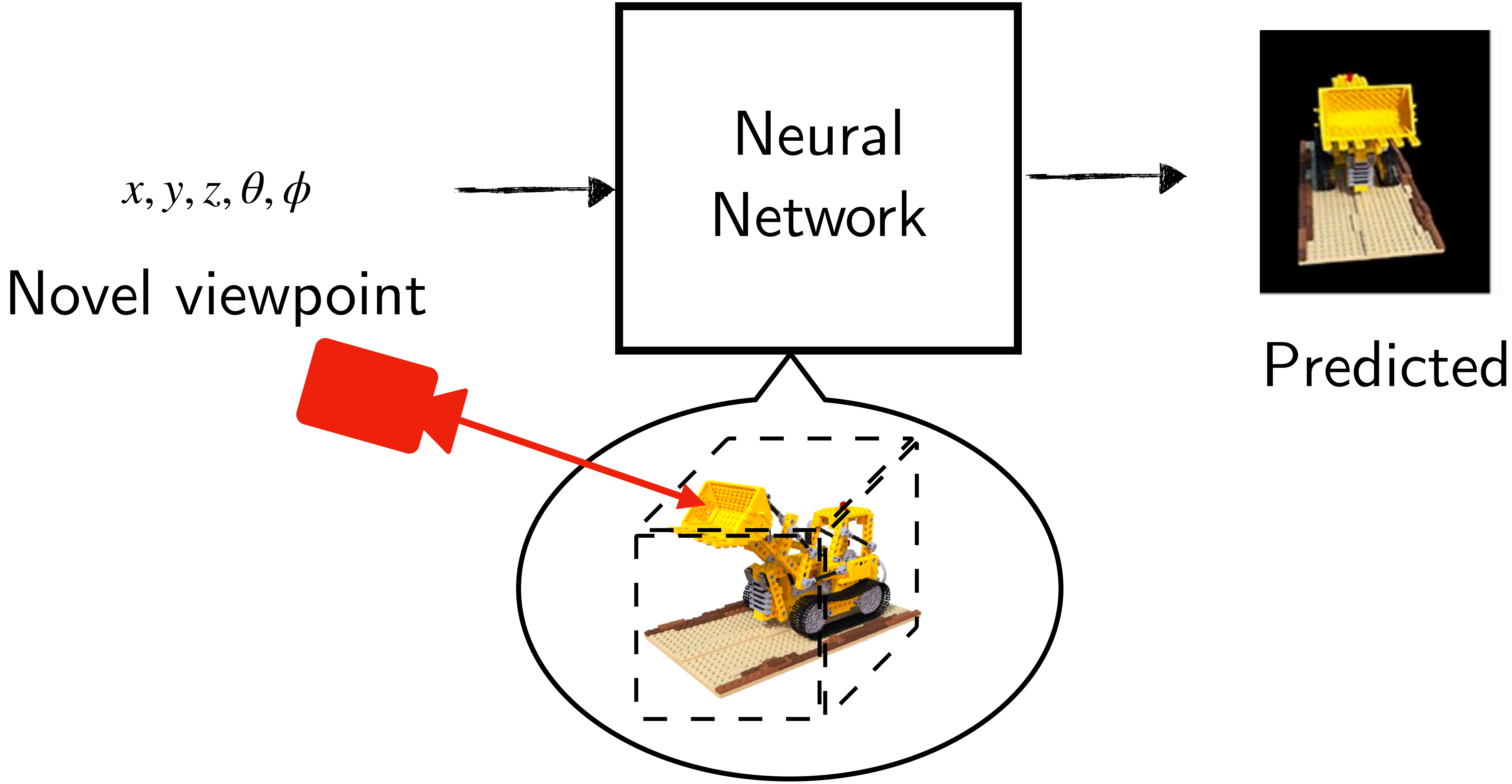
Predicted

Let's setup a learning problem

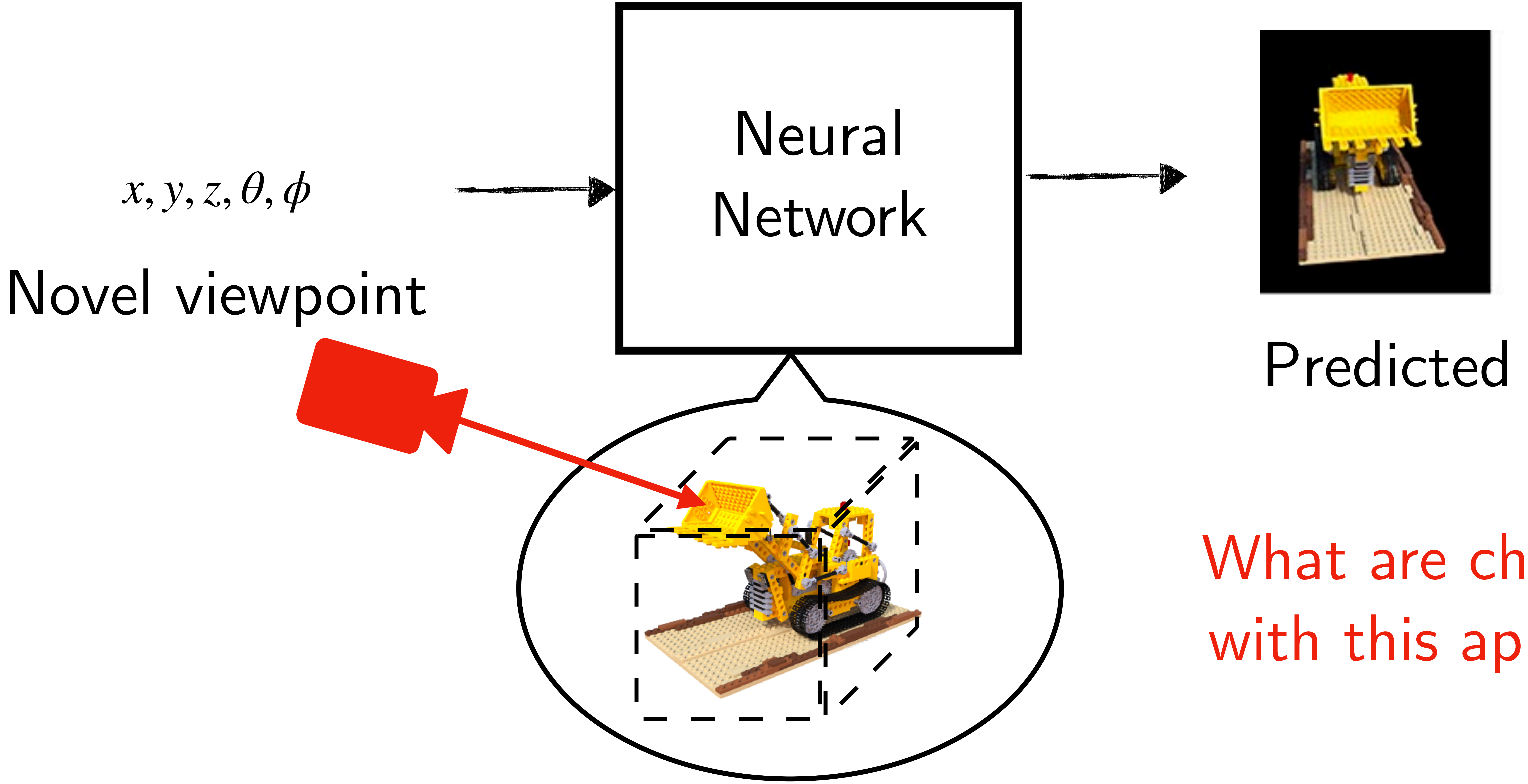
Simple idea:

Can't we just make the neural network predict a 3D voxel grid of RGB values?

Predict a 3D grid, render image from viewpoint



Predict a 3D grid, render image from viewpoint



What are challenges with this approach?

Challenges

Discretization loses information!

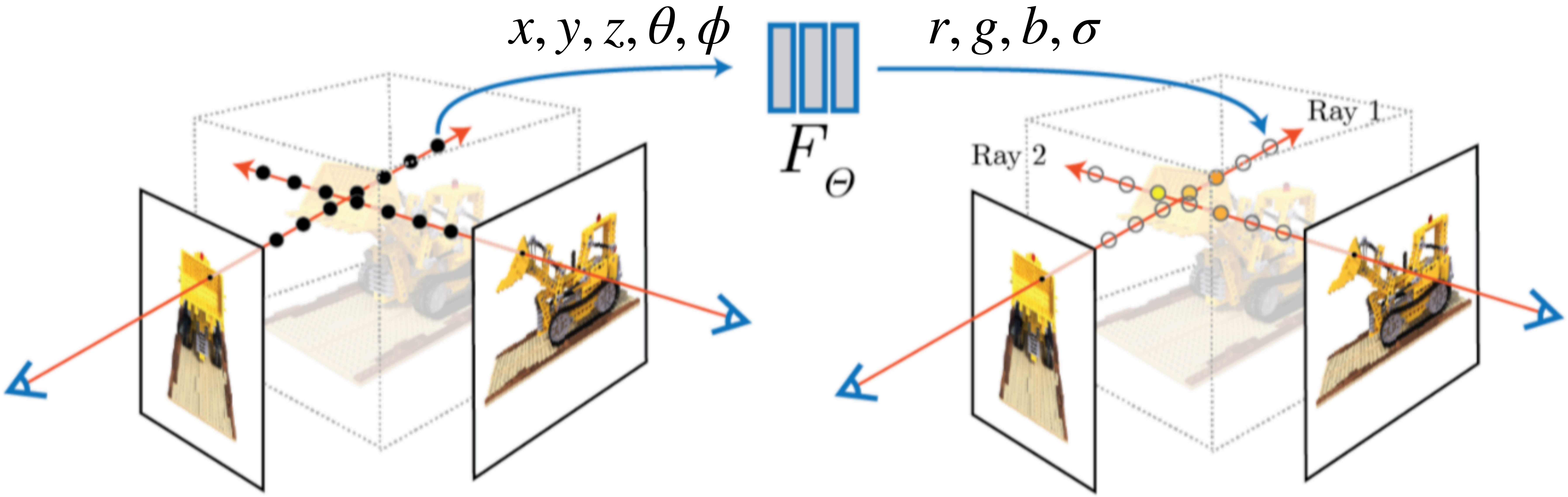
Memory Inefficient!

Not differentiable! (Not a continuous projection)

NERF to the rescue!

What are Neural Radiance Fields (NeRFs)?

Idea: Use a neural network to *implicitly* represent 3D volume!



✓ No Discretization ✓ Compressible ✓ Differentiable

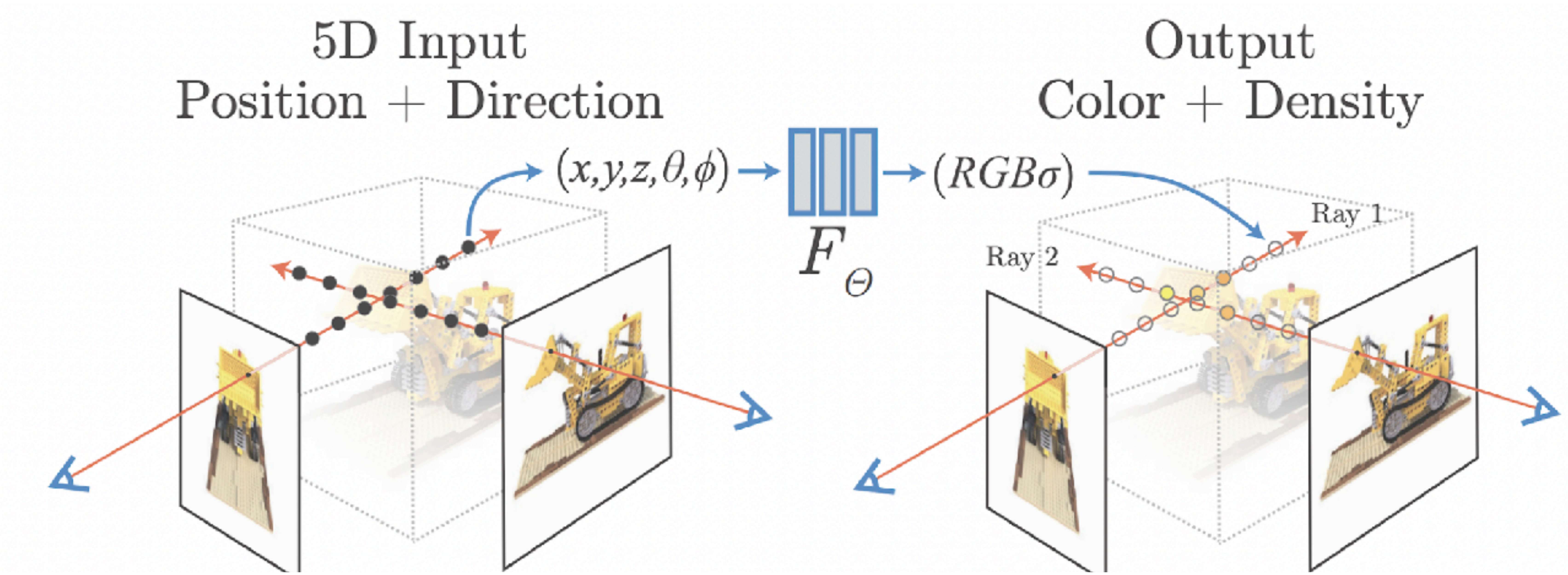
NeRF scene representation

$$\underbrace{(x, y, z,}_{\text{Spatial location}}\ \underbrace{\theta, \phi)}_{\text{Viewing direction}} \longrightarrow F_{\Omega} \longrightarrow \underbrace{(r, g, b,}_{\text{Output color}}\ \underbrace{\sigma)}_{\text{Output density}}$$

F_{Ω}

Fully-connected
neural network
9 layers,
256 channels

Differentiable Loss Function

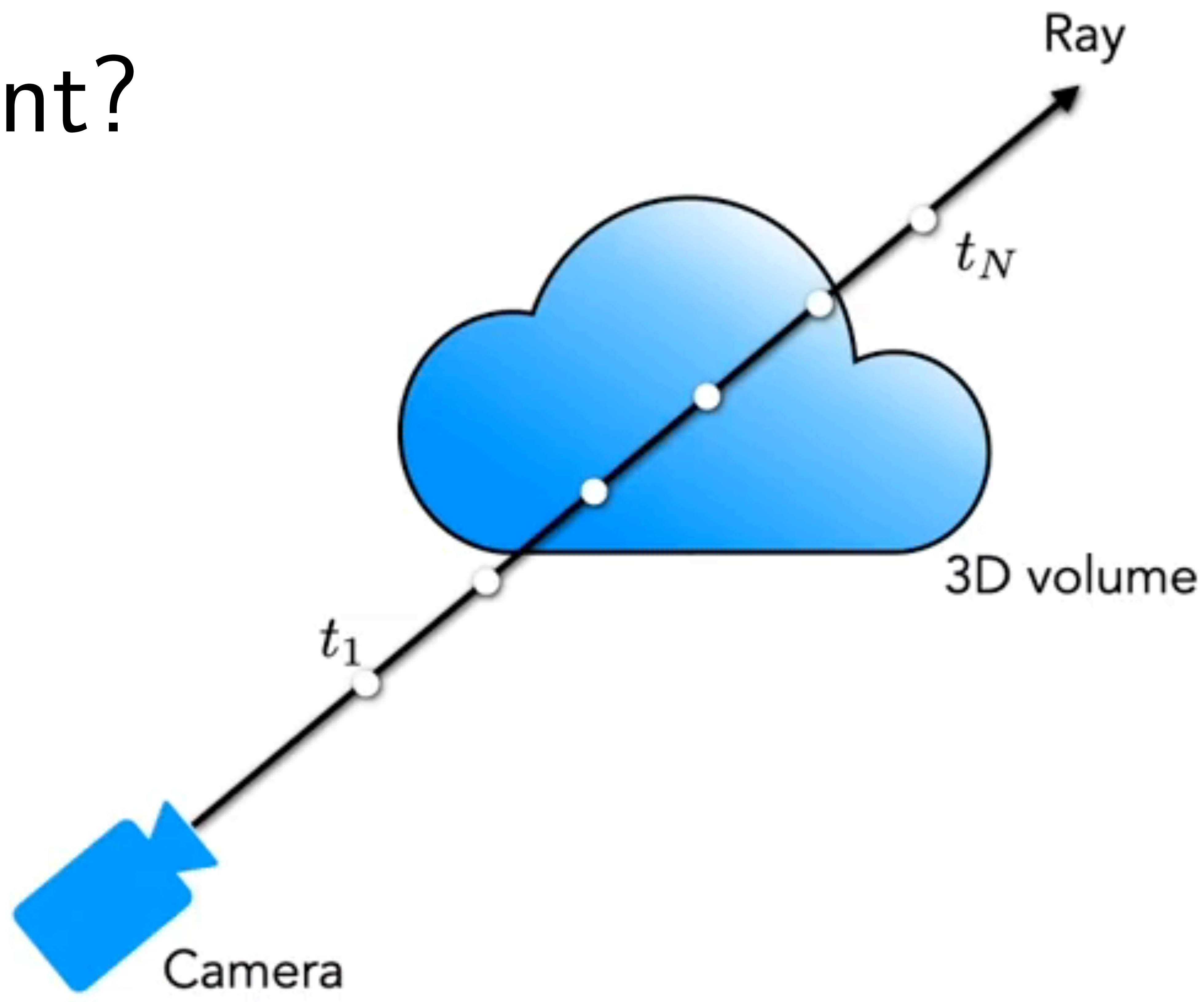


$$\min_{\Theta} \sum_i ||\mathbf{render}_i(F_\Theta) - I_i^{gt}||^2$$

What is the render() function?
How is it differentiable?

Volume rendering model

What is the color from this viewpoint?



Volume rendering model

Rendering model for ray r(t) = o + td:

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

colors

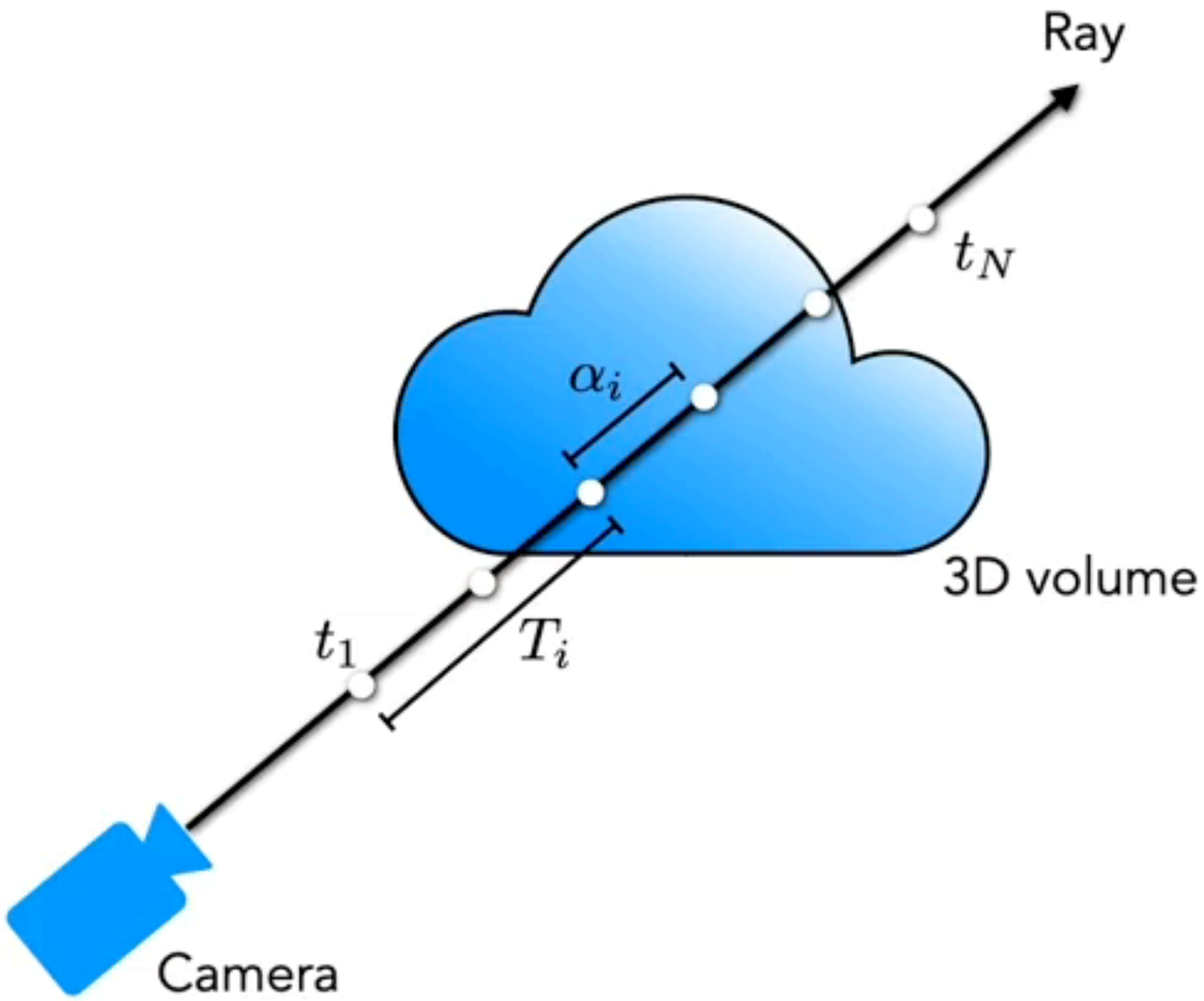
weights

How much light is blocked earlier along ray:

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

How much light is contributed by ray segment *i*:

$$\alpha_i = 1 - e^{-\sigma_i \delta t_i}$$



Numerical integration step easily differentiable

Rendering model for ray r(t) = o + td:

$$C \approx \sum_{i=1}^{N} \underbrace{T_i \alpha_i}_{\text{weights}} \underbrace{c_i}_{\text{colors}}$$

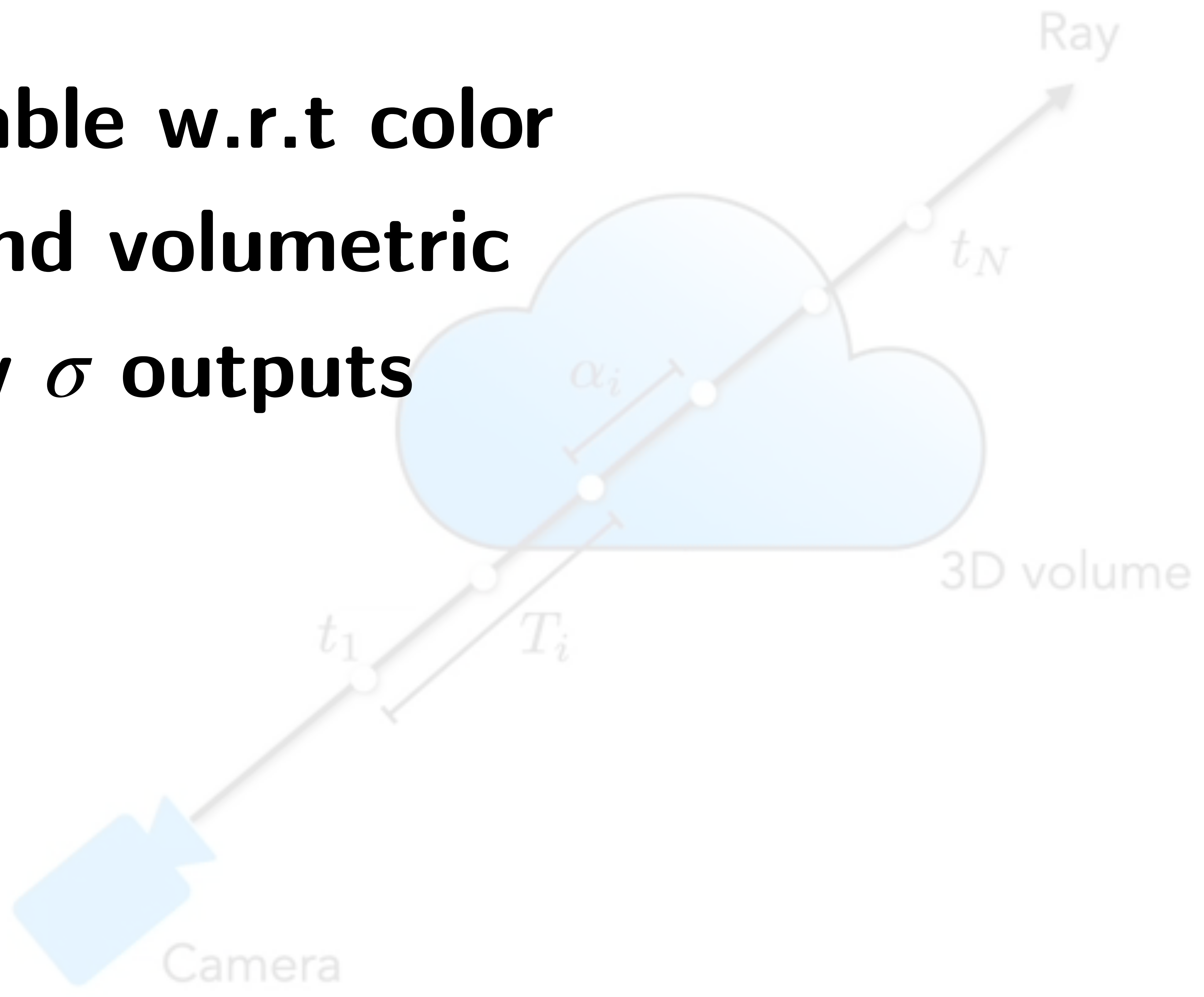
Differentiable w.r.t color (r, g, b) and volumetric density σ outputs

How much light is blocked earlier along ray:

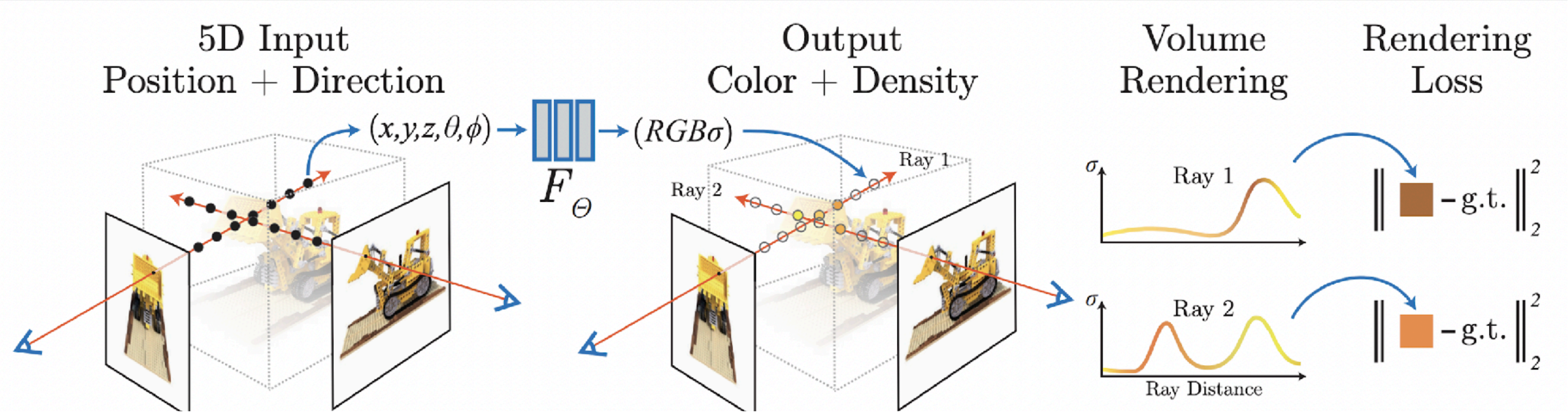
$$T_i = \prod_{j=1}^{i-1} (1 - \alpha_j)$$

How much light is contributed by ray segment *i*:

$$\alpha_i = 1 - e^{-\sigma_i \delta t_i}$$



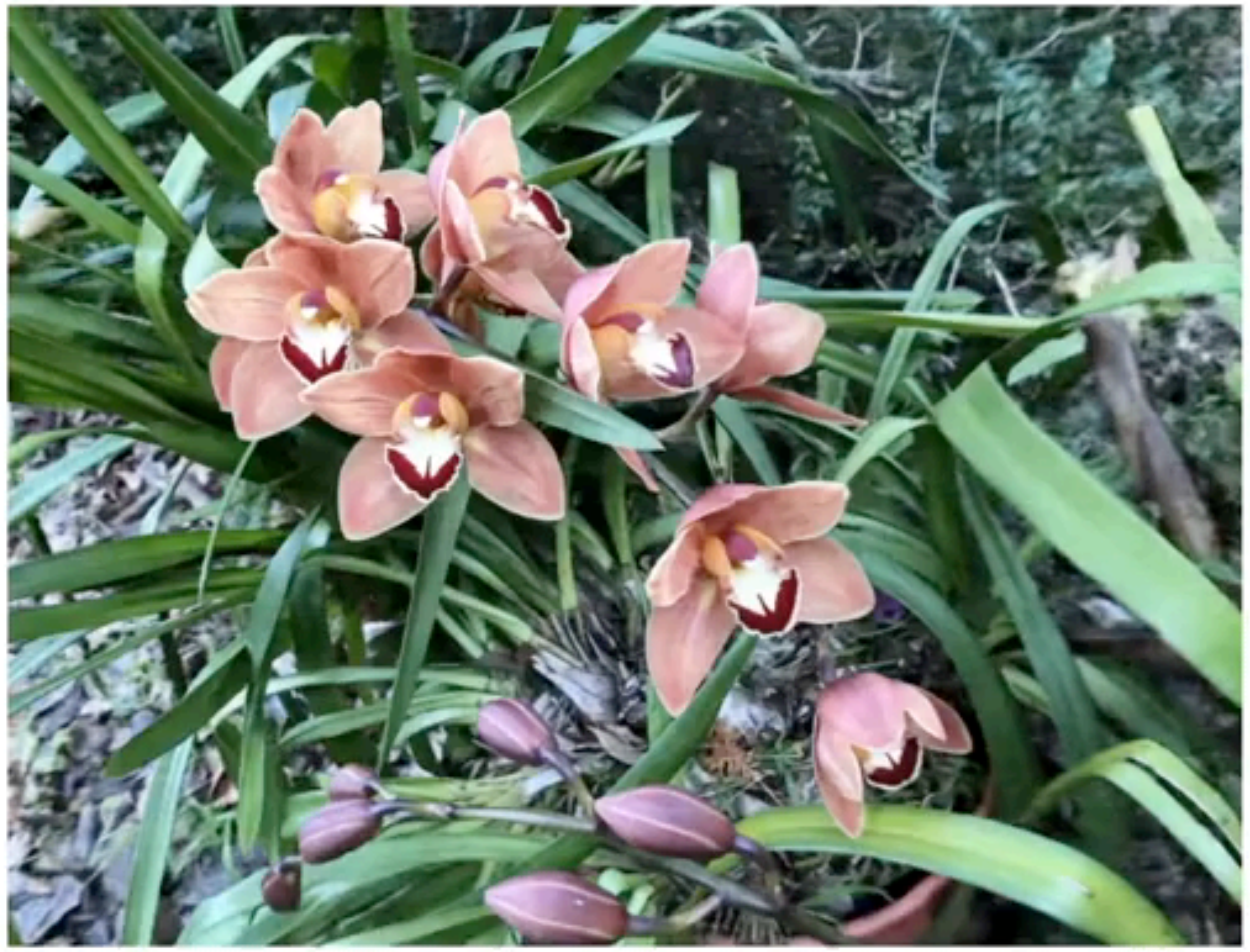
NeRF Summary



Results

Novel View Synthesis

Inputs: sparsely sampled images of scene



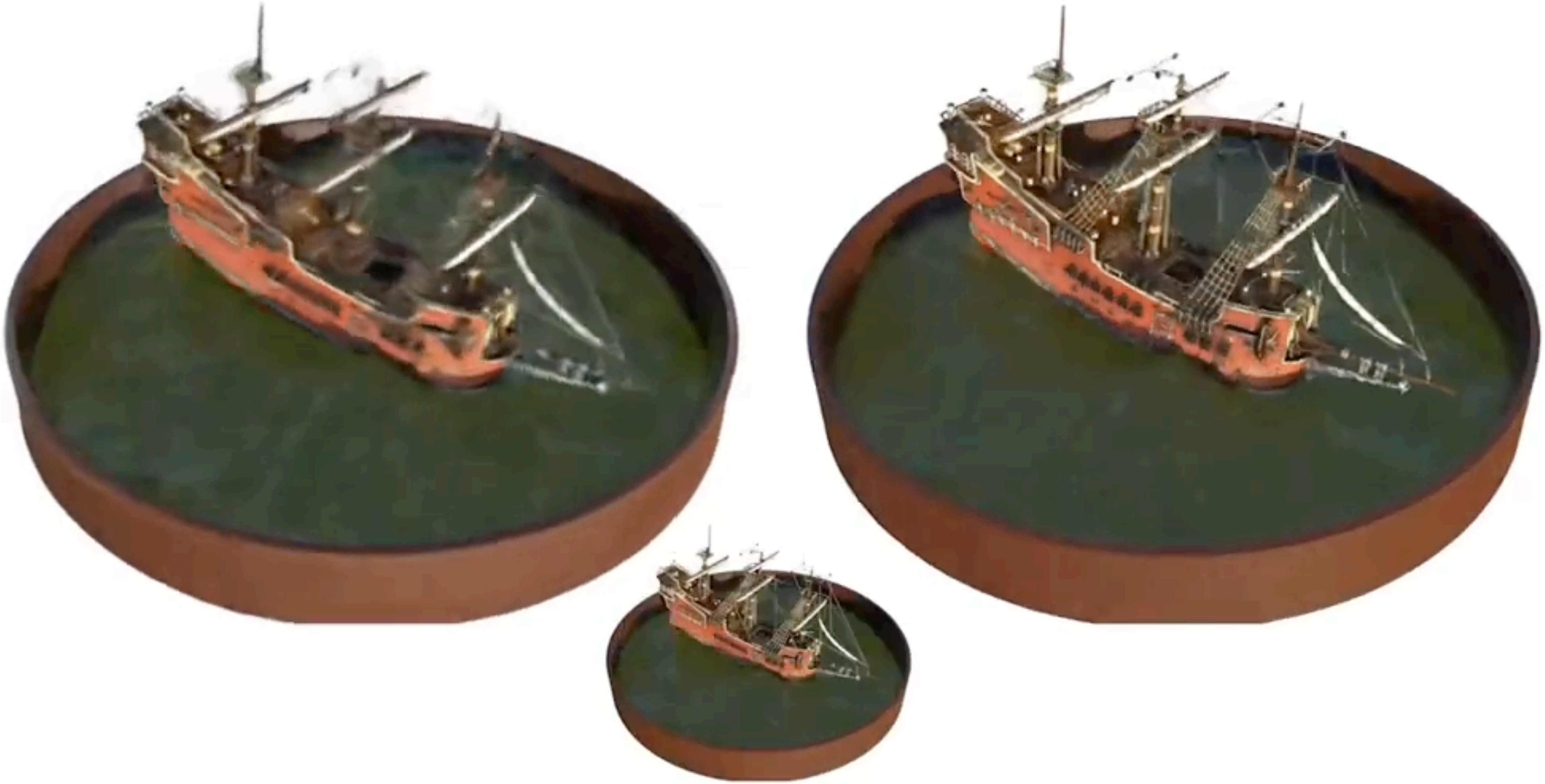
Outputs: *new* views of same scene
(rendered by our method)

2


More detailed and consistent than prior work that represents scene as discrete voxel grid

Neural Volumes [Lombardi 2019]

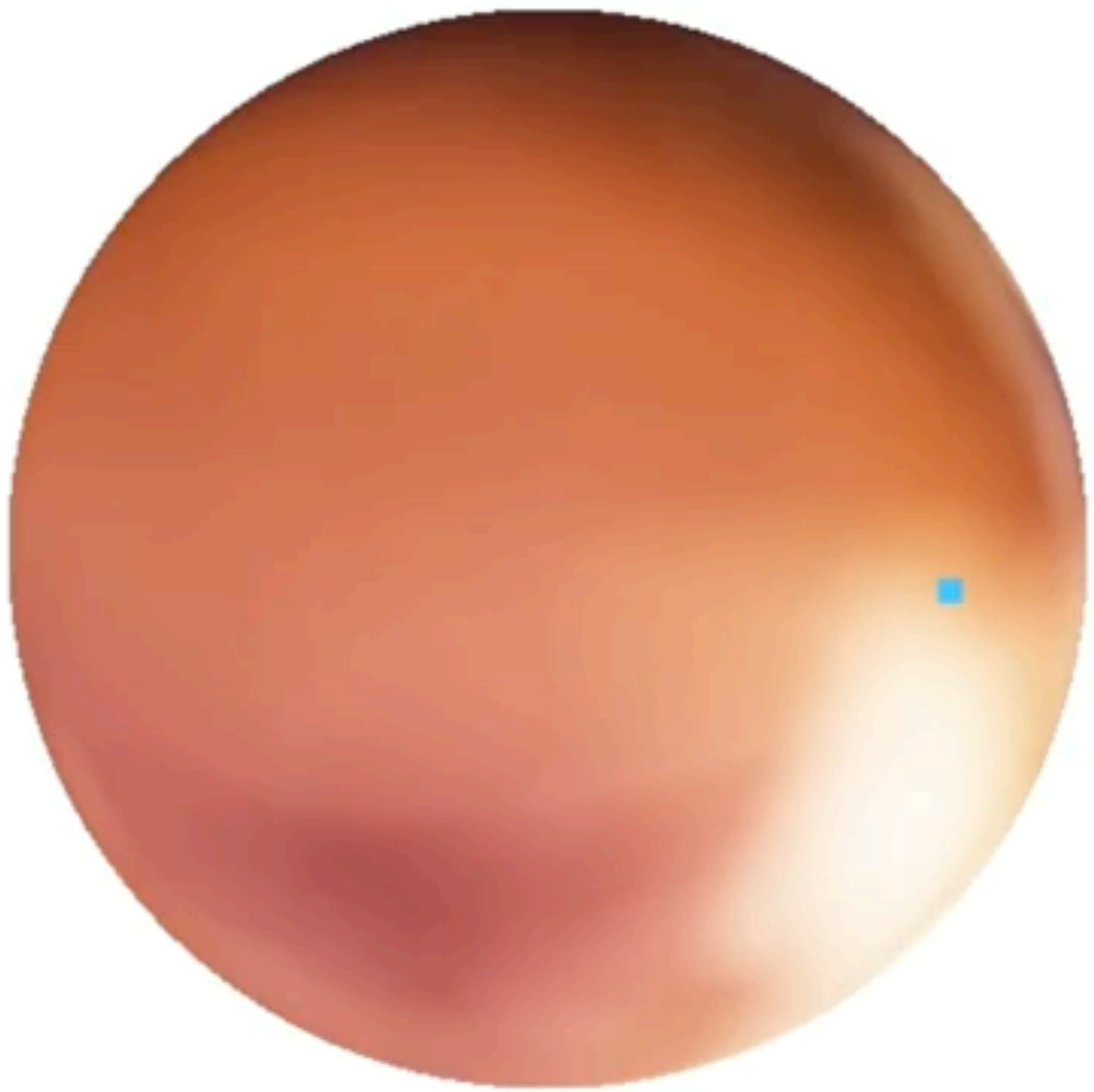
NeRF



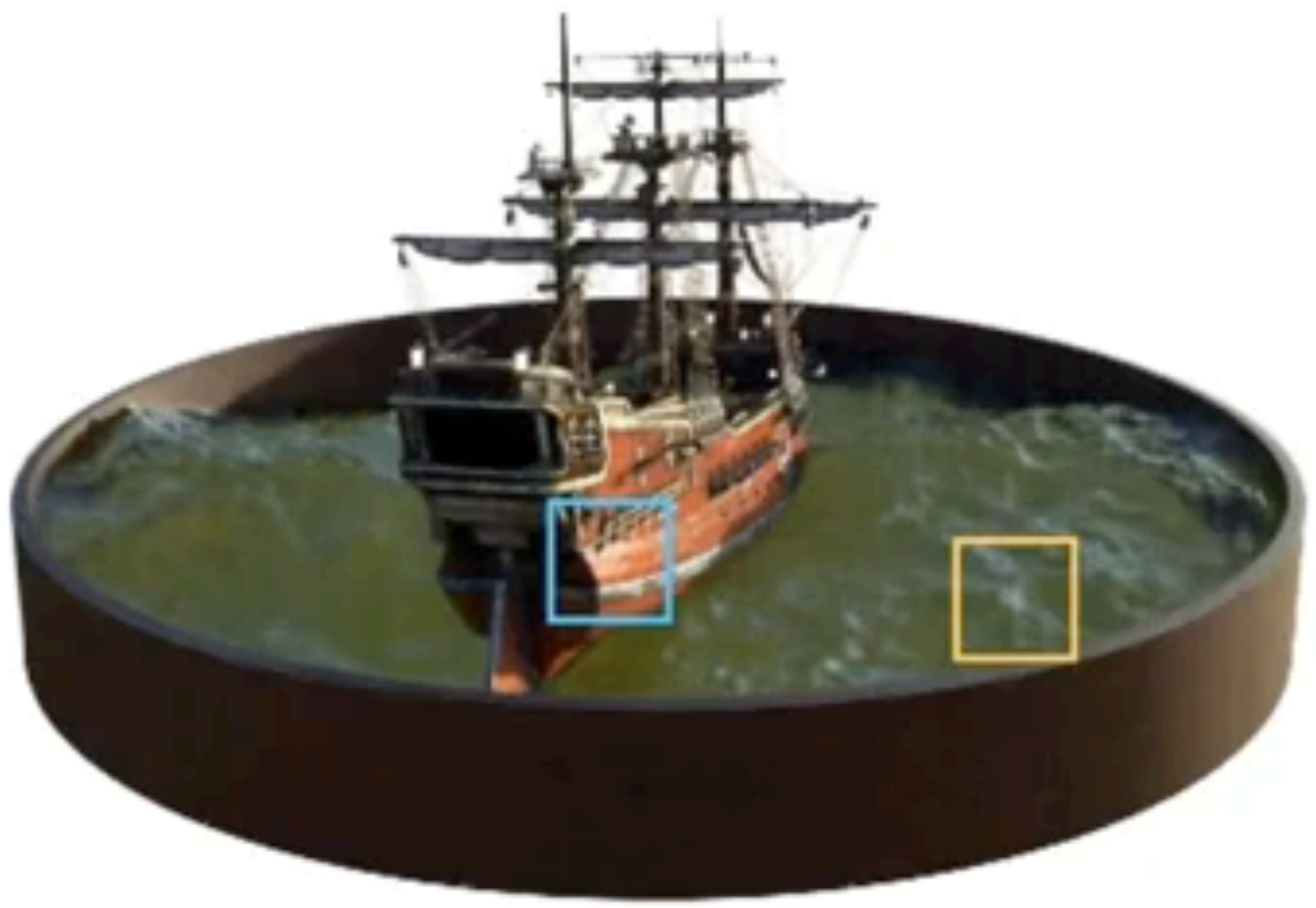
36

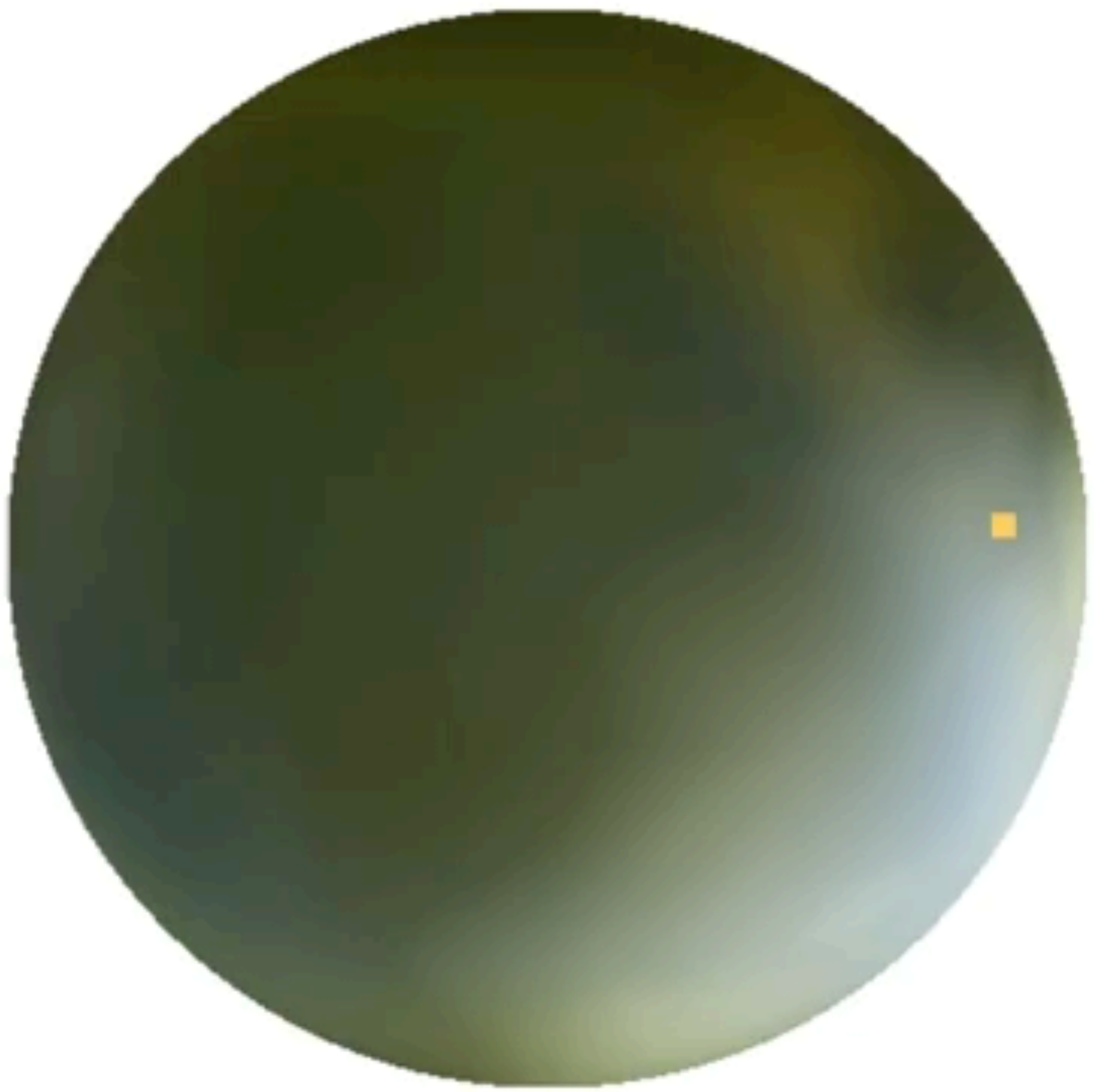
Why do we need r,g,b to be a function of viewpoint?

Viewing directions as input



Radiance distribution for point on side of ship





Radiance distribution for point on water's surface

One key trick to make it work ...

Naively passing in position creates blurry images!

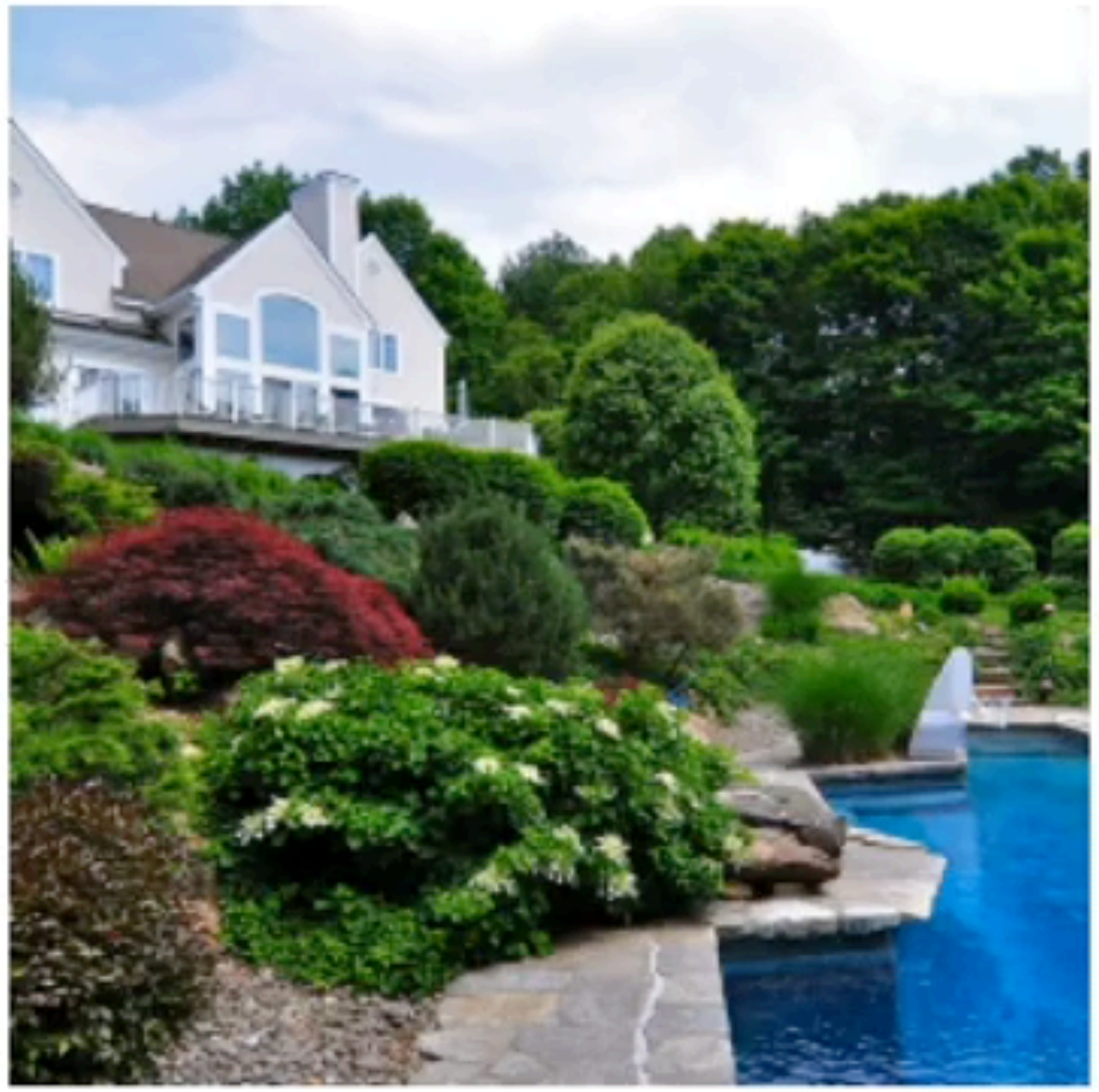
$$\underbrace{(x, y, z,}_{\text{Spatial location}} \underbrace{\theta, \phi)}_{\text{Viewing direction}} \rightarrow F_{\Omega} \rightarrow \underbrace{(r, g, b,}_{\text{Output color}} \underbrace{\sigma)}_{\text{Output density}}$$

Why?

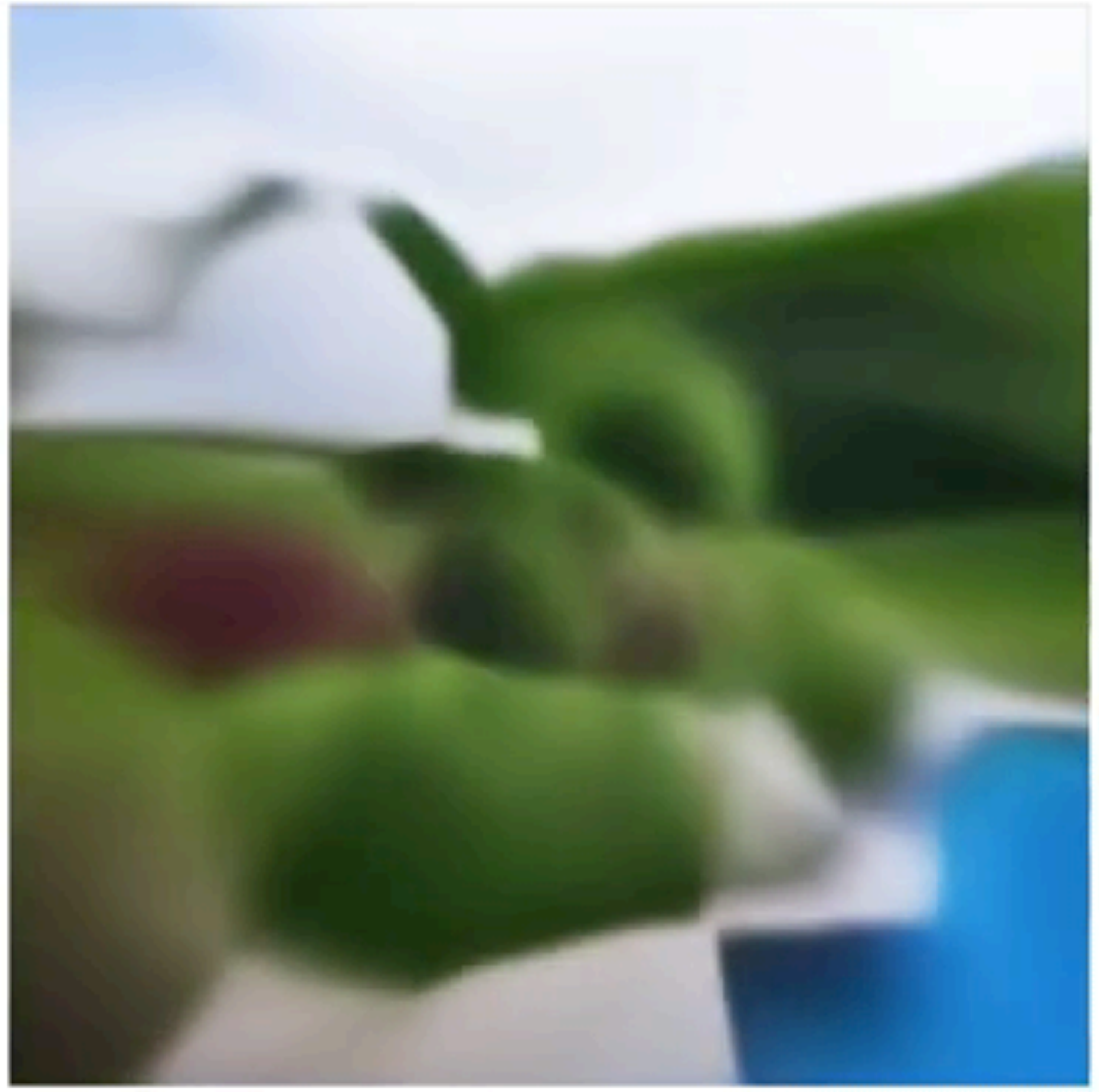
Activity!

Let's say we train a network to memorize an image

Ground truth image



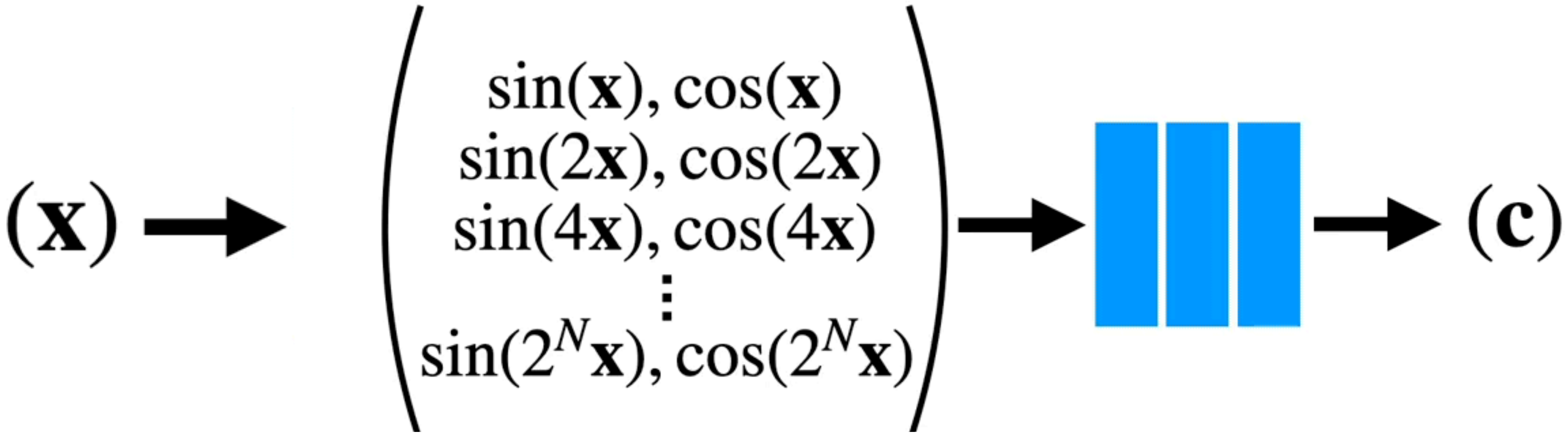
Standard fully-connected net



How do we make the image look sharper?

Idea: Encode low-dim coordinates to high-dim features

High-frequency embedding of input coordinates

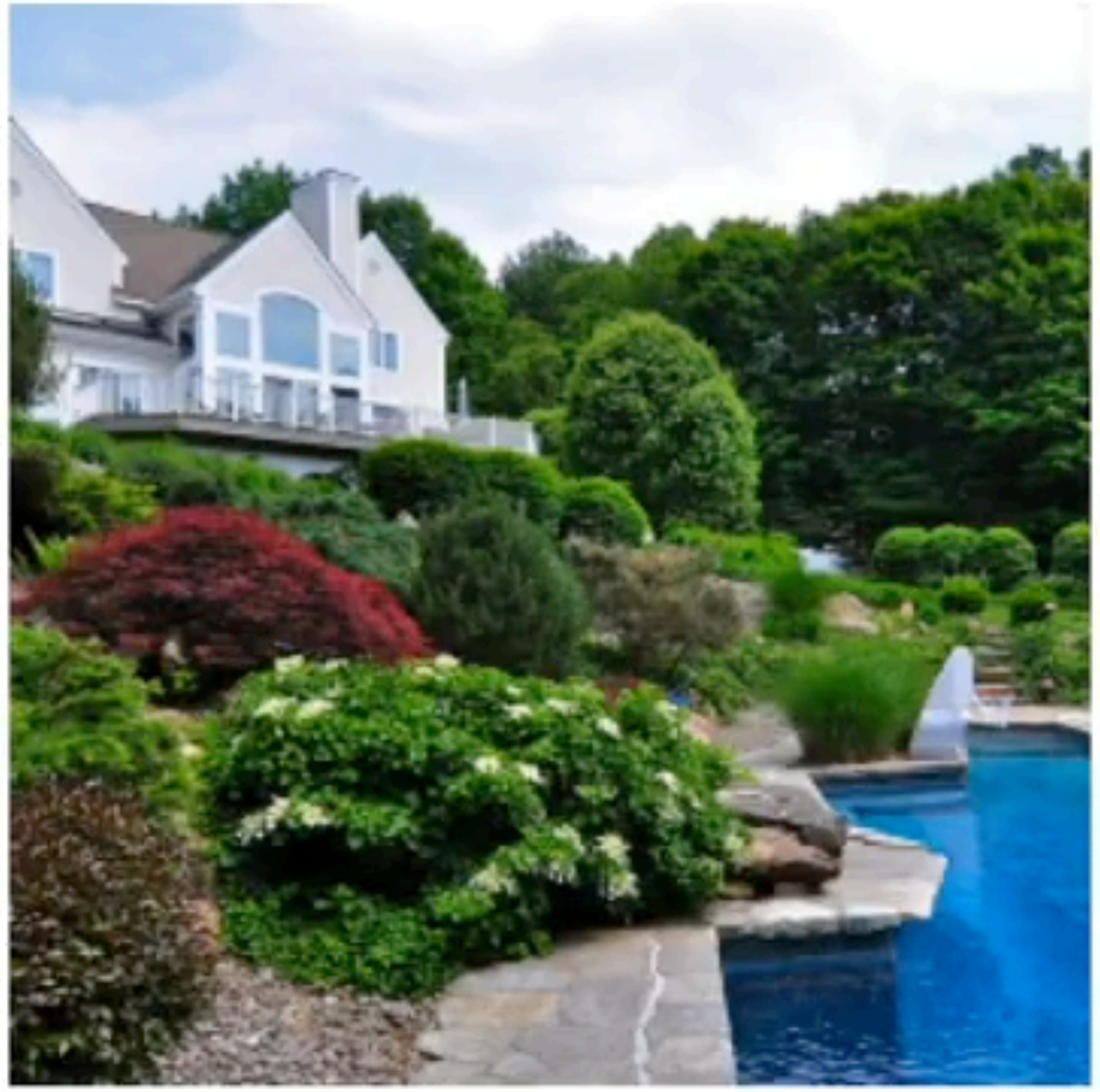


<u>Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains, Tancik et al.</u>

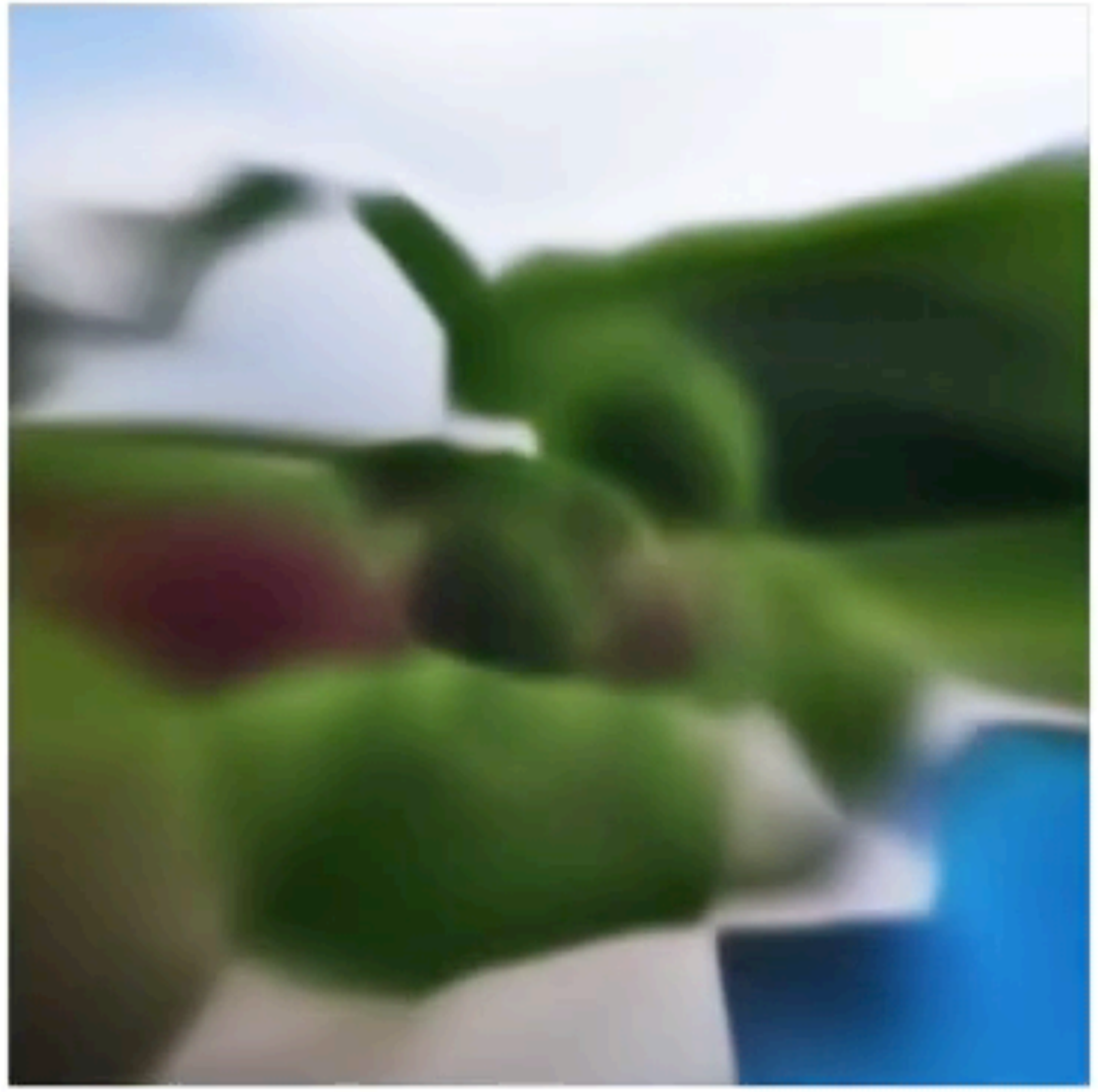
Use positional encoding

$$\begin{pmatrix} \sin(\mathbf{x}), \cos(\mathbf{x}) \\ \sin(2\mathbf{x}), \cos(2\mathbf{x}) \\ \sin(4\mathbf{x}), \cos(4\mathbf{x}) \\ \vdots \\ \sin(2^N\mathbf{x}), \cos(2^N\mathbf{x}) \end{pmatrix}$$

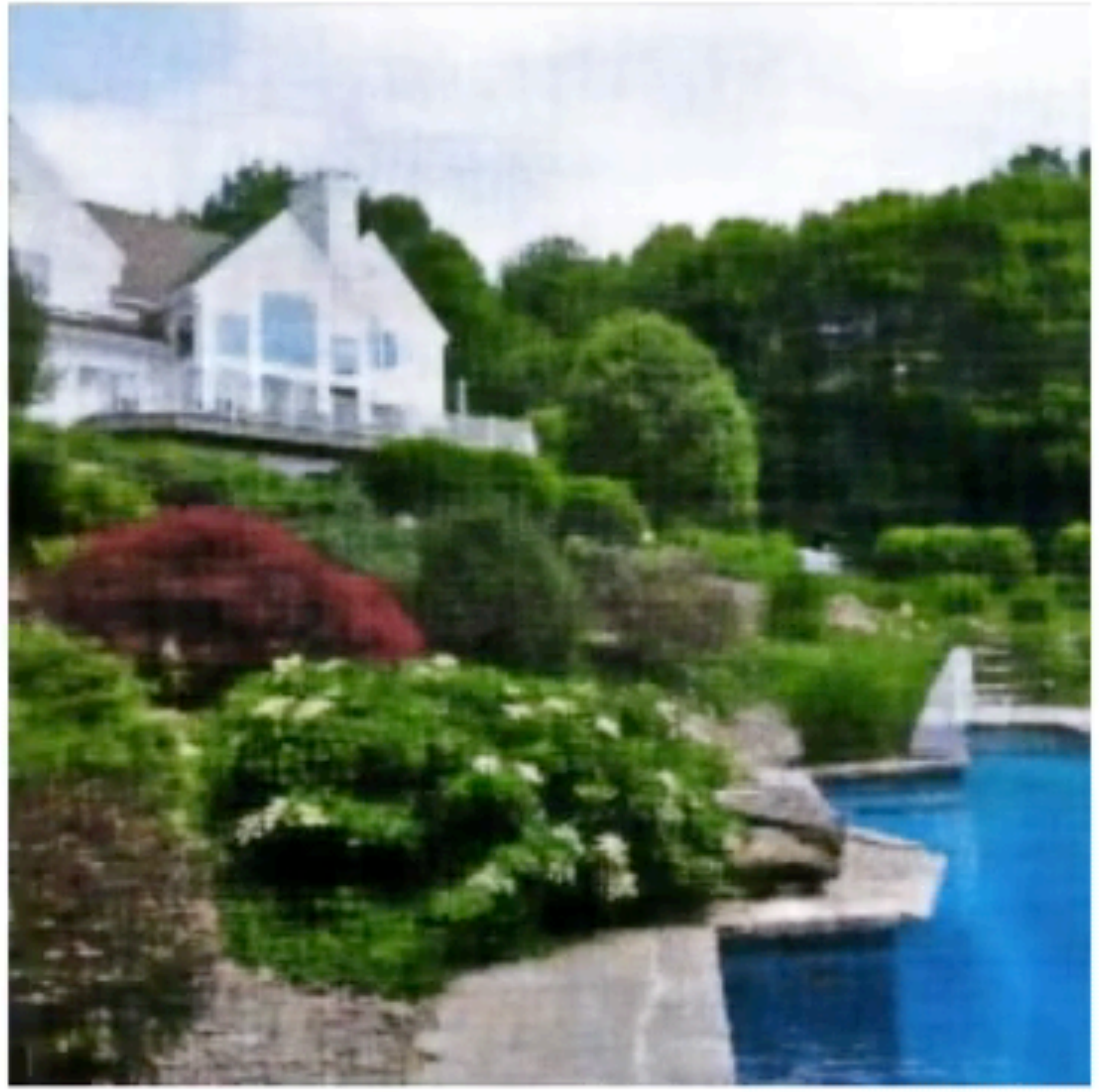
Ground truth image

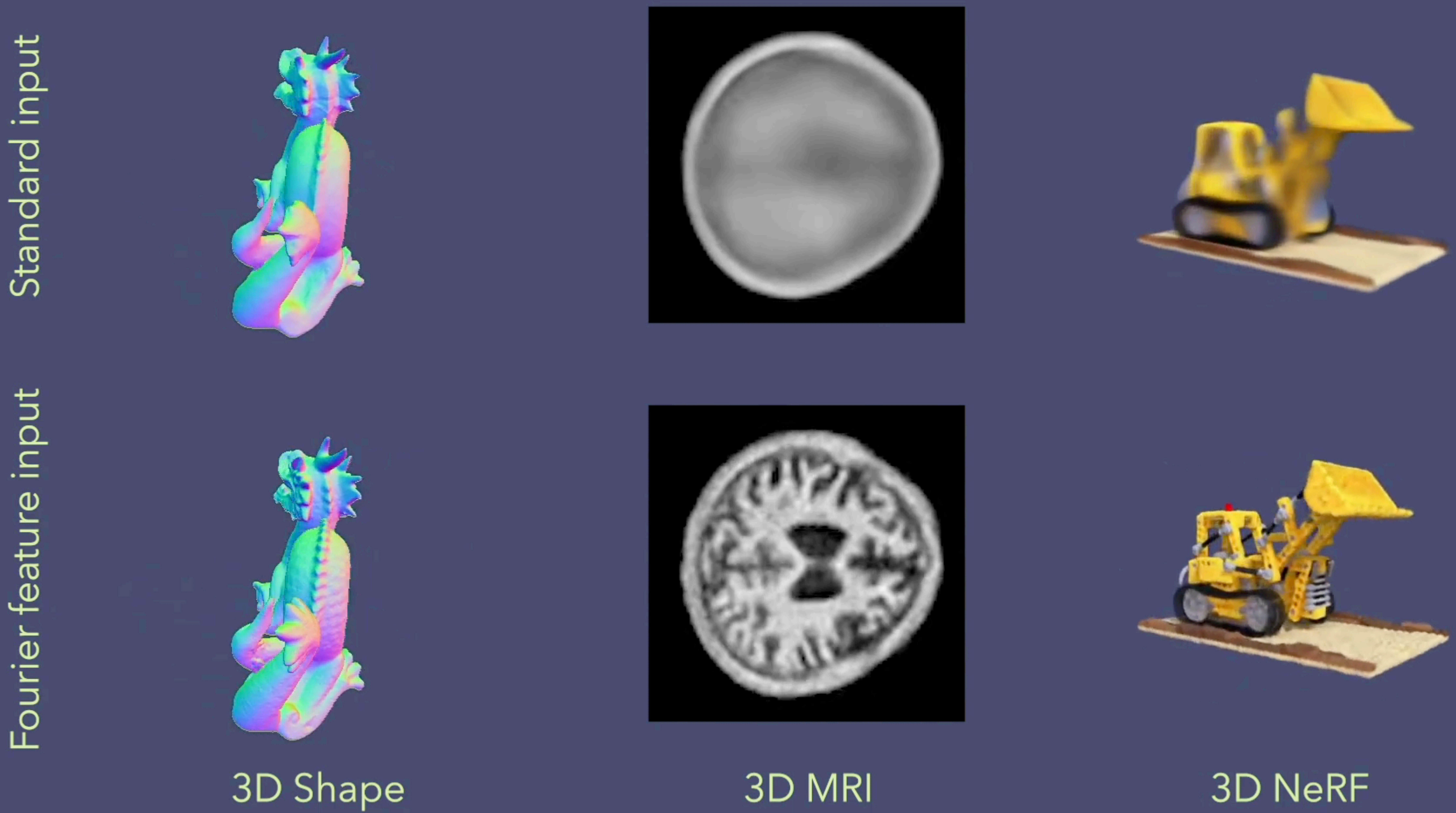


Standard fully-connected net



With "positional encoding"

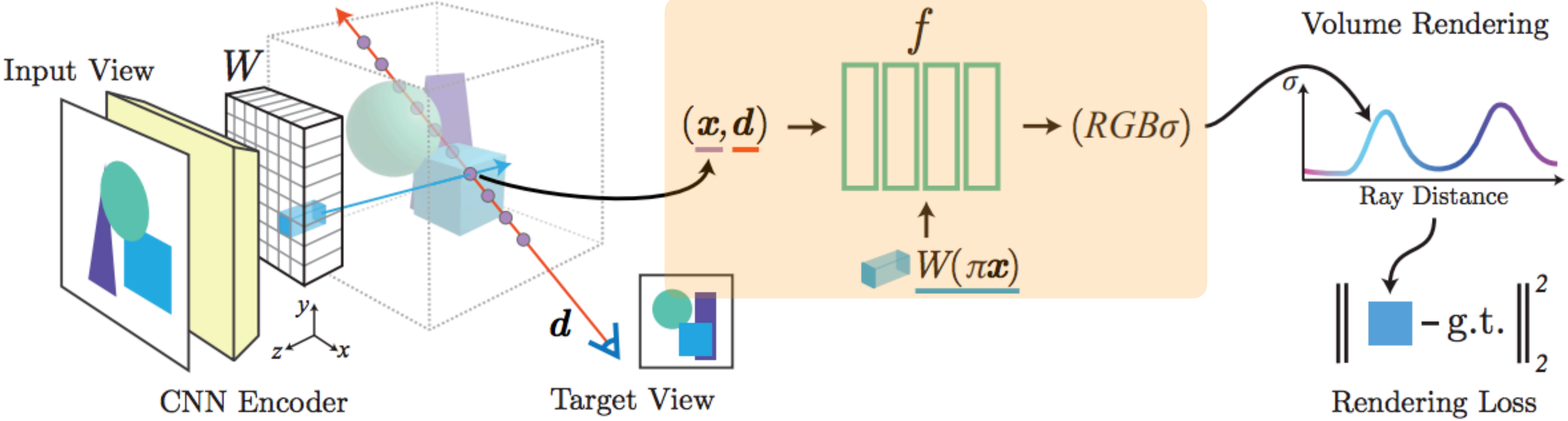



Standard input
Fourier feature input
3D Shape
3D MRI
3D NeRF

Lots of extension and applications!

Generalization

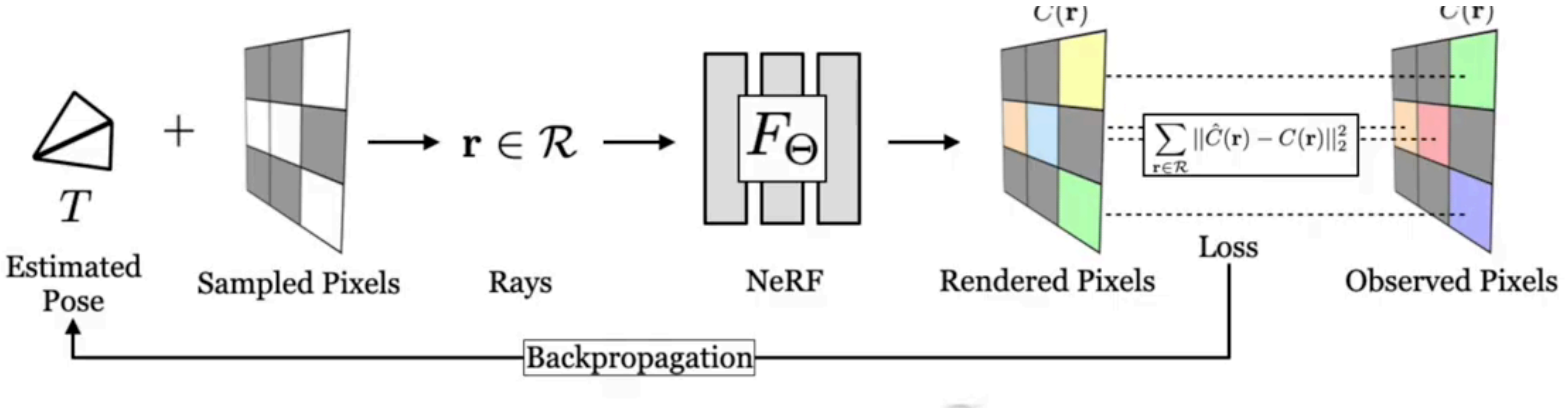
Goal: Train a NeRF for arbitrary new scenes with fewer images



pixelNeRF: Neural Radiance Fields from One or Few Images

Unknown camera poses

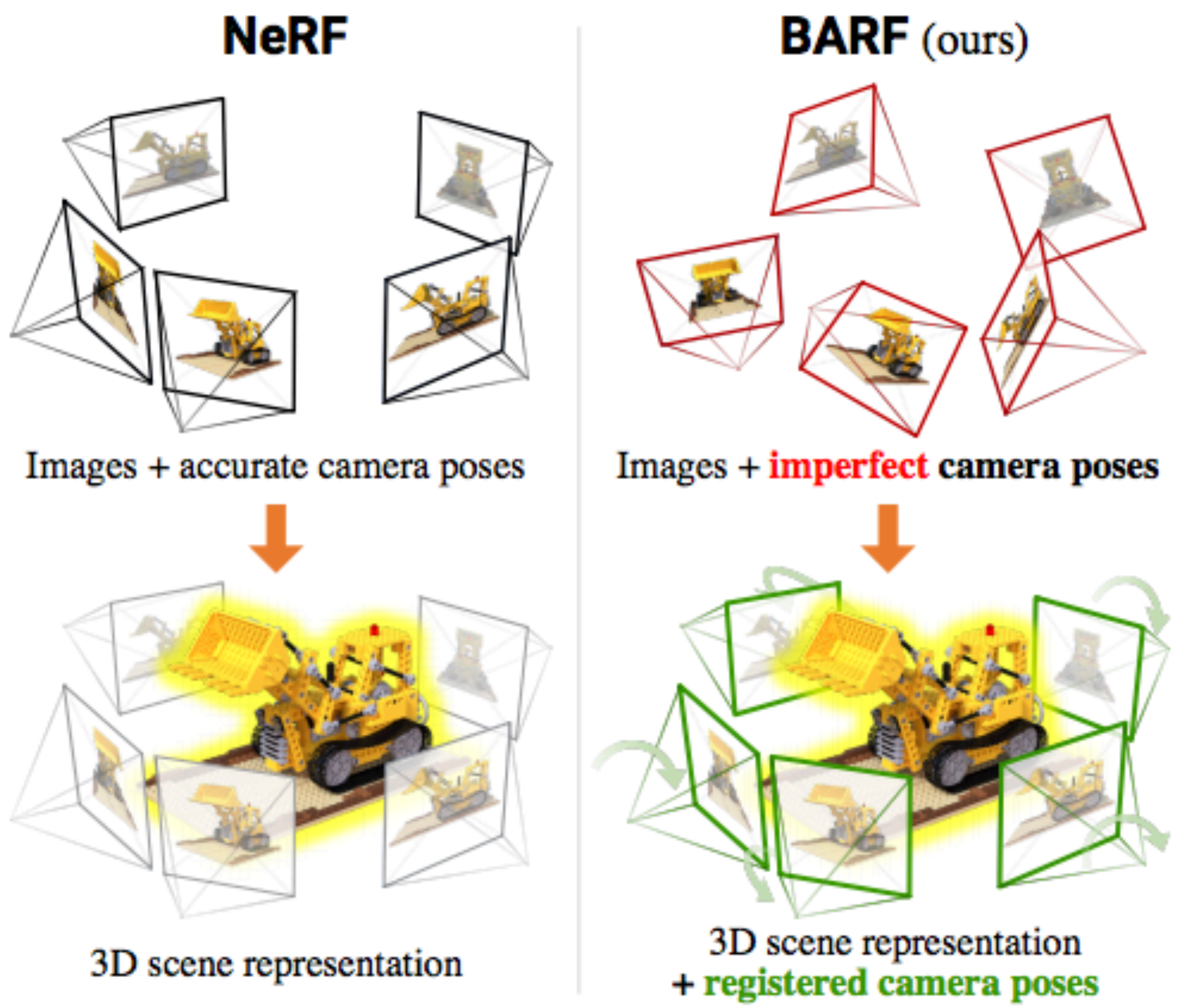
Goal: Estimate poses given a trained NeRF

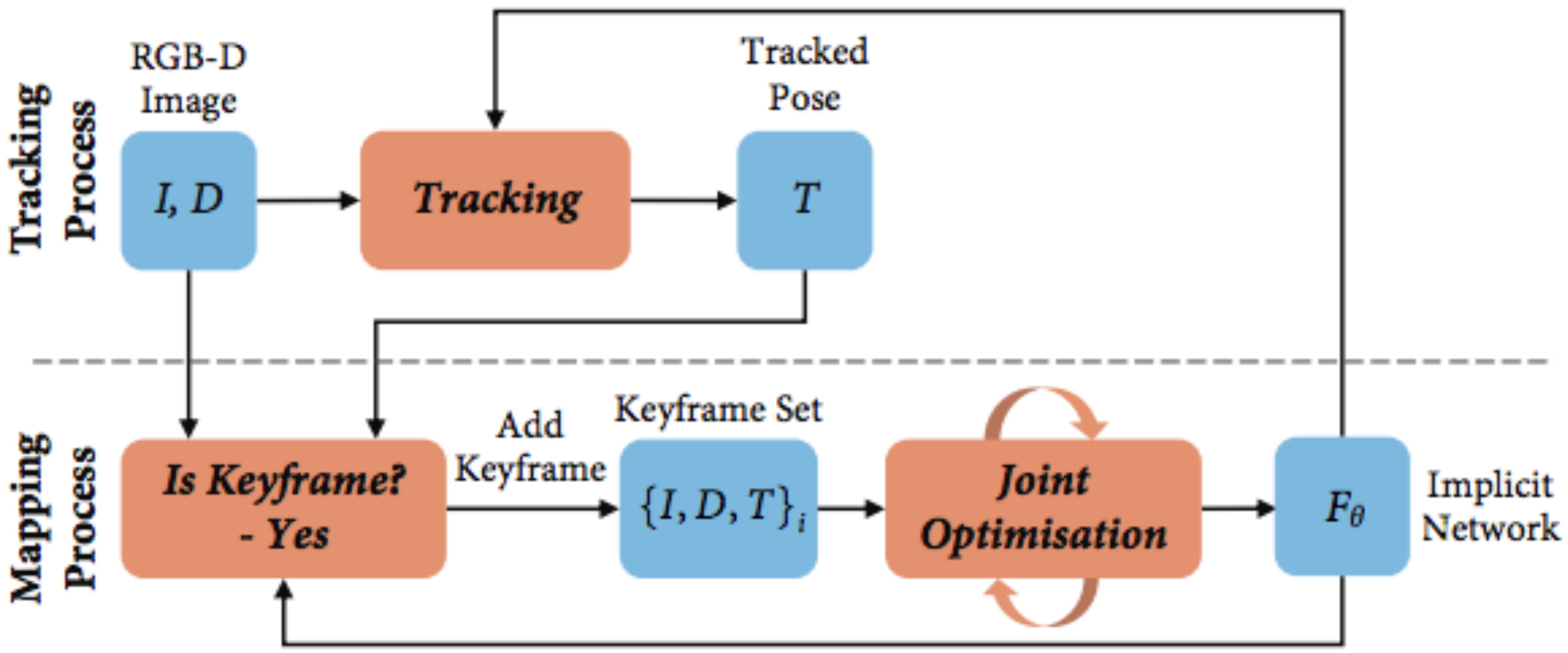


[iNeRF: Inverting Neural Radiance Fields for Pose Estimation]

Unknown camera poses + Scene

Goal: Simultaneously estimate pose and scene representation





[BARF : Bundle-Adjusting Neural Radiance Fields]

[iMAP: Implicit Mapping and Positioning in Real-Time]

NERF for Grasping

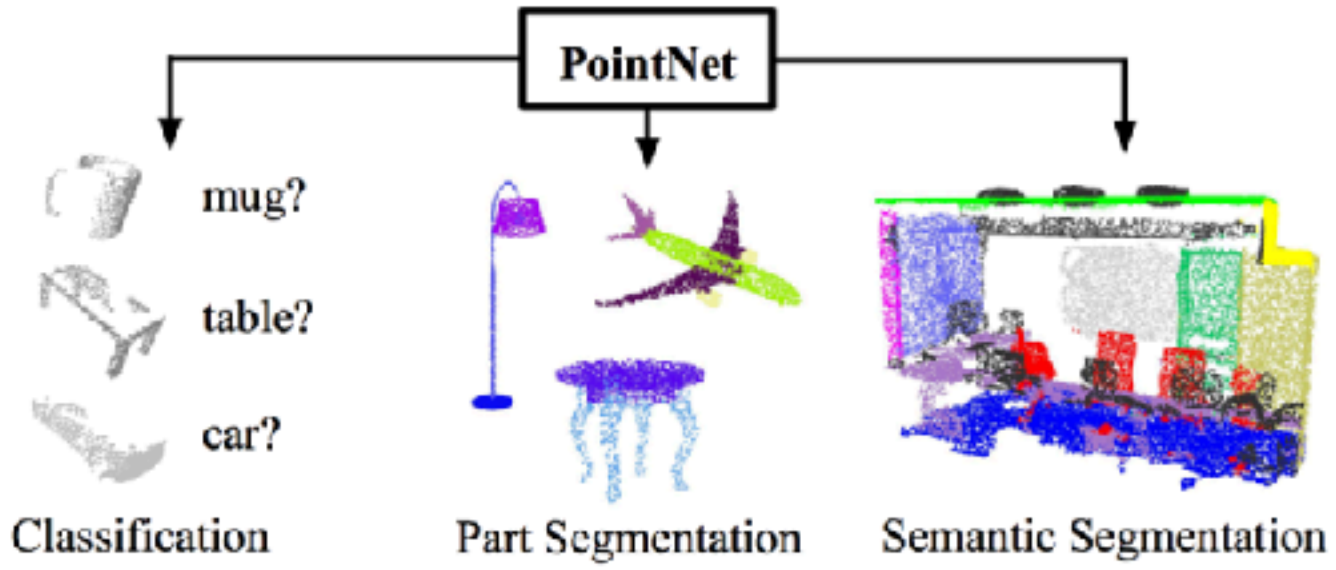
tl;dr

But manipulating objects require 3D reasoning!

PointNet

End-to-end learning for **scattered, unordered** point data

Unified framework for various tasks

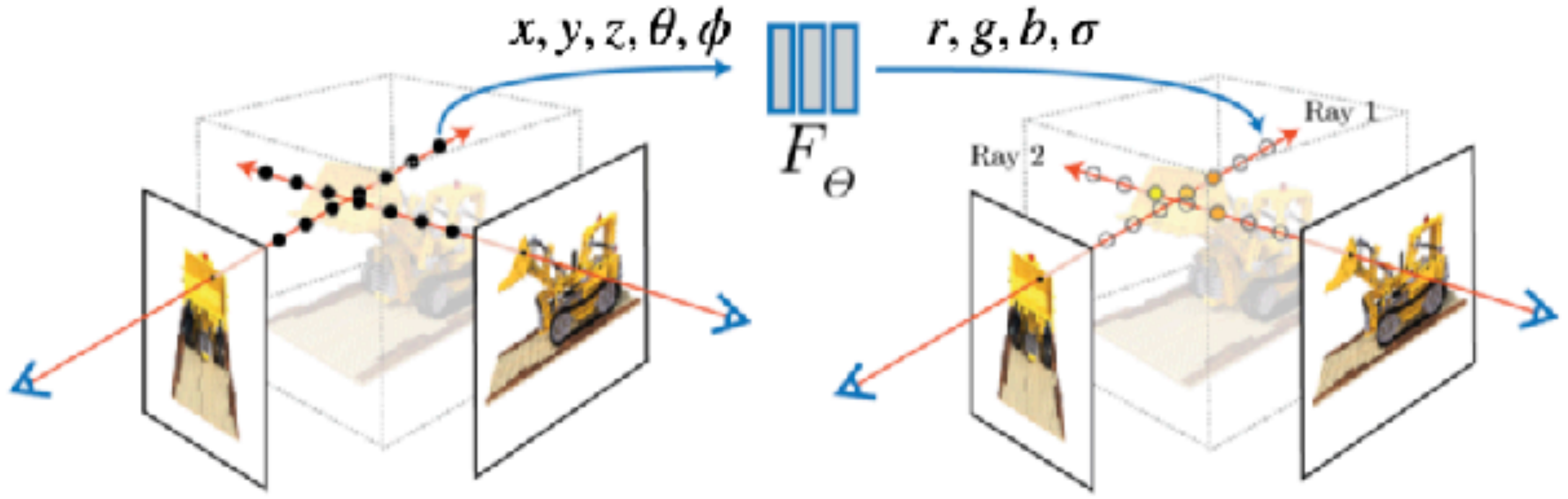


Slides from Qi et al, CVP 2017 http://stanford.edu/~rqi/pointnet/docs/cvpr17_pointnet_slides.pdf

13

What are Neural Radiance Fields (NeRFs)?

Idea: Use a neural network to *implicitly* represent 3D volume!



✓ No Discretization ✓ Compressible ✓ Differentiable

54