Deep Learning in 3D

CSE P576

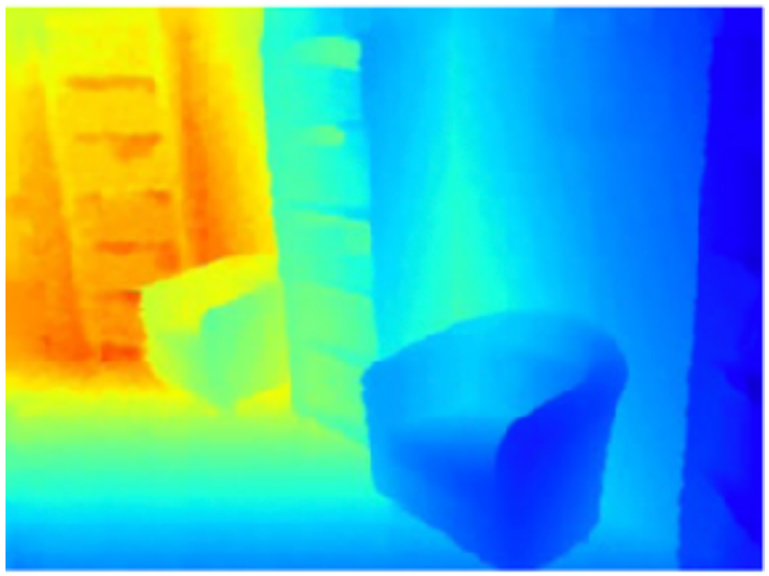
Dr. Matthew Brown

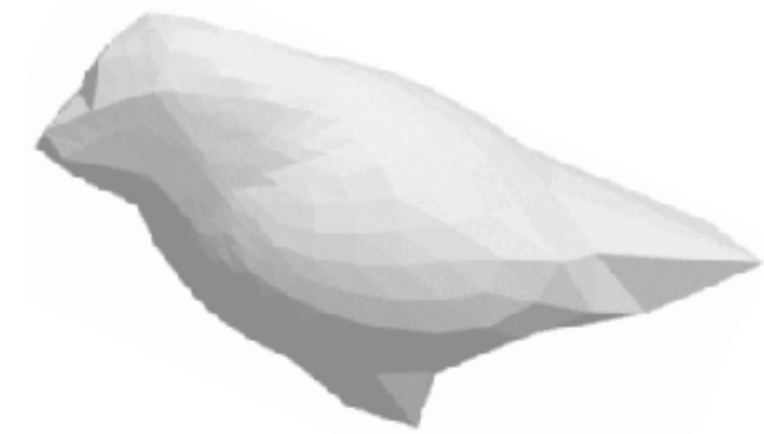
Deep Learning in 3D

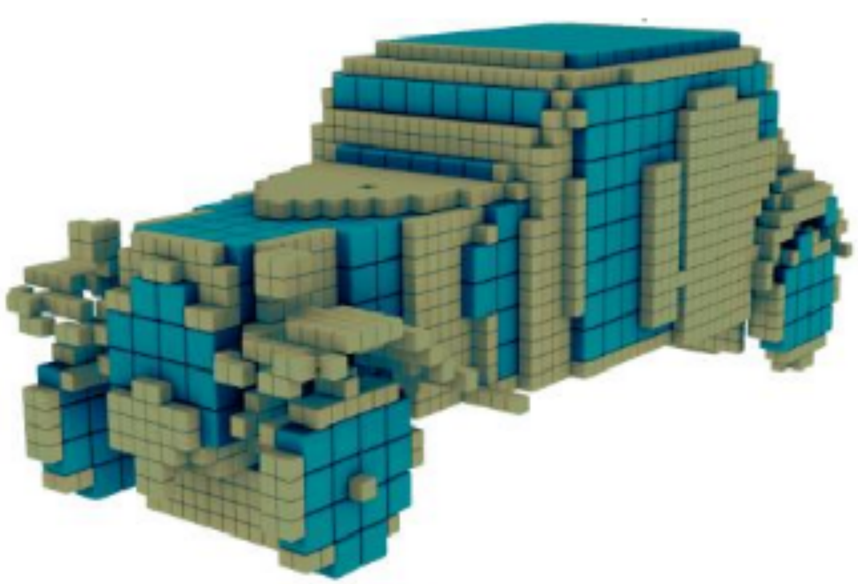
- We'll focus on predicting 3D from one or more image
- Supervision: depth, mesh, silhouettes, view supervision
- Representations: Depth, Points, Meshes, Voxels, SDFs
- Neural Scene Representation and Rendering

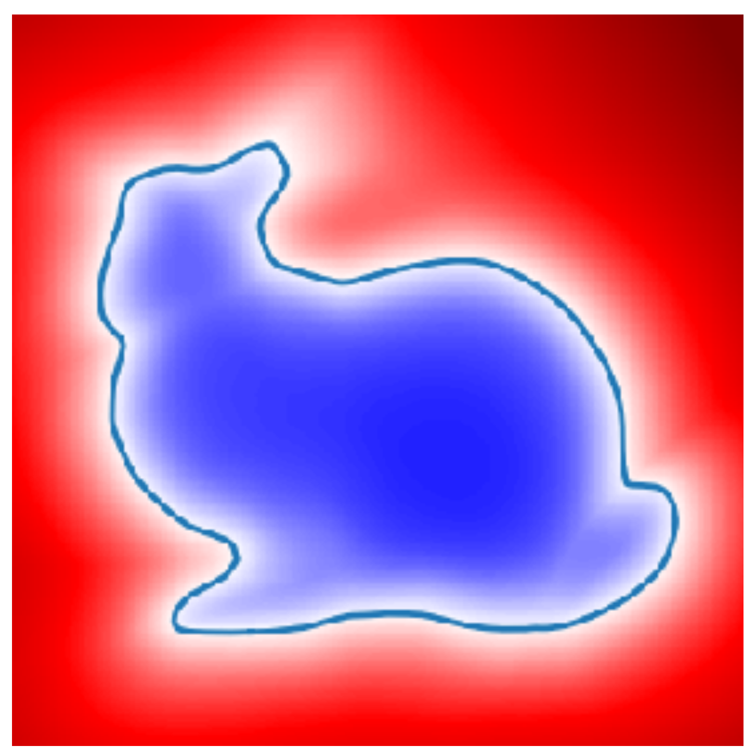
3D Representation

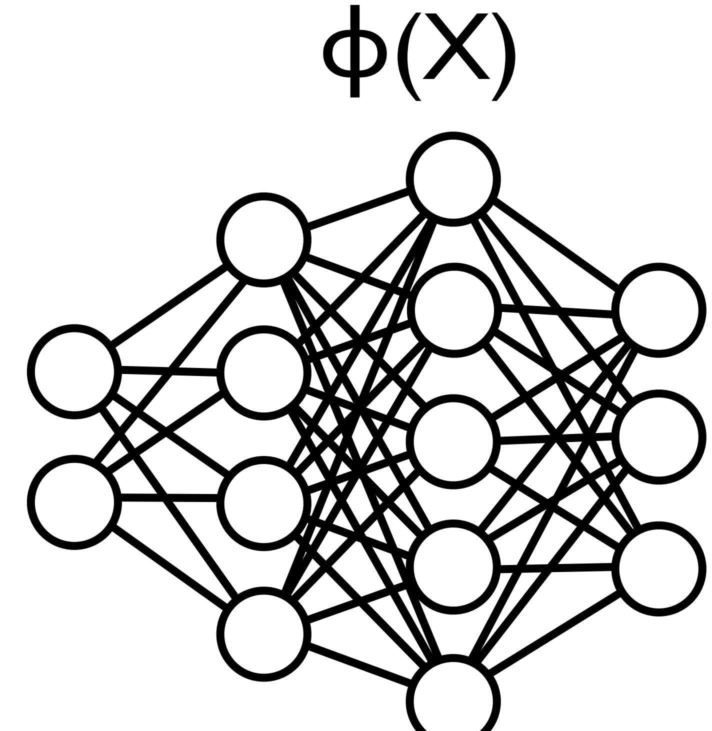
- Many ways to represent objects in 3D



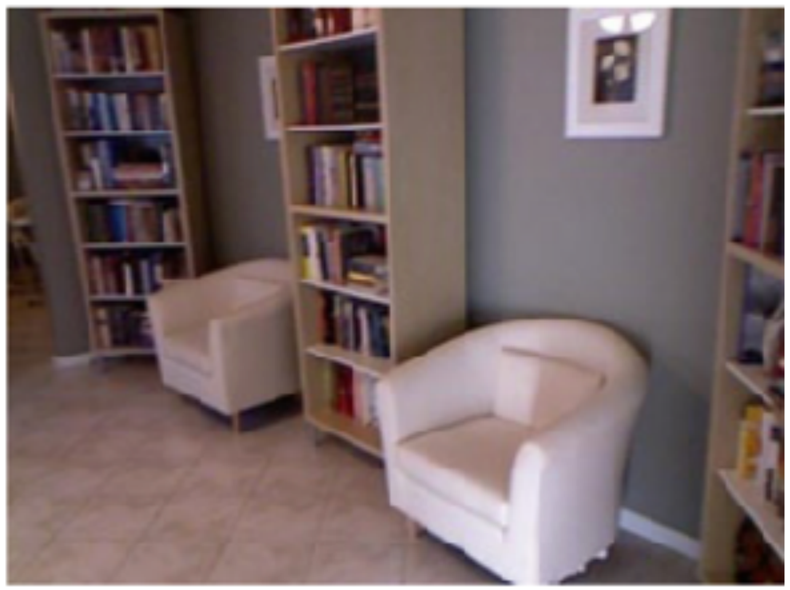


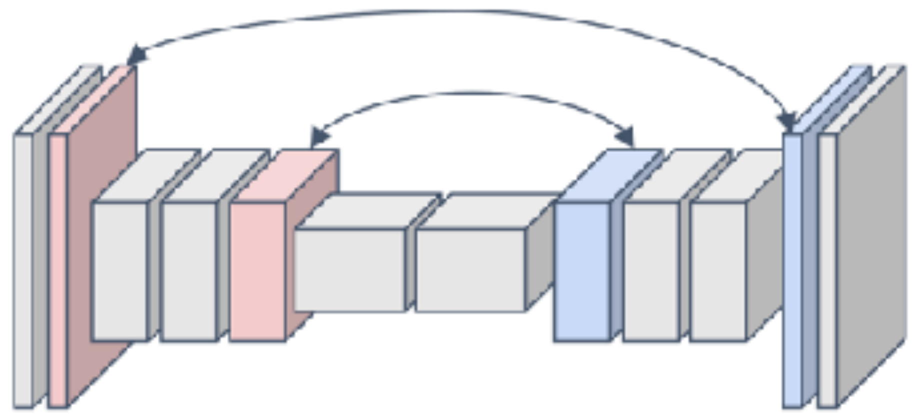




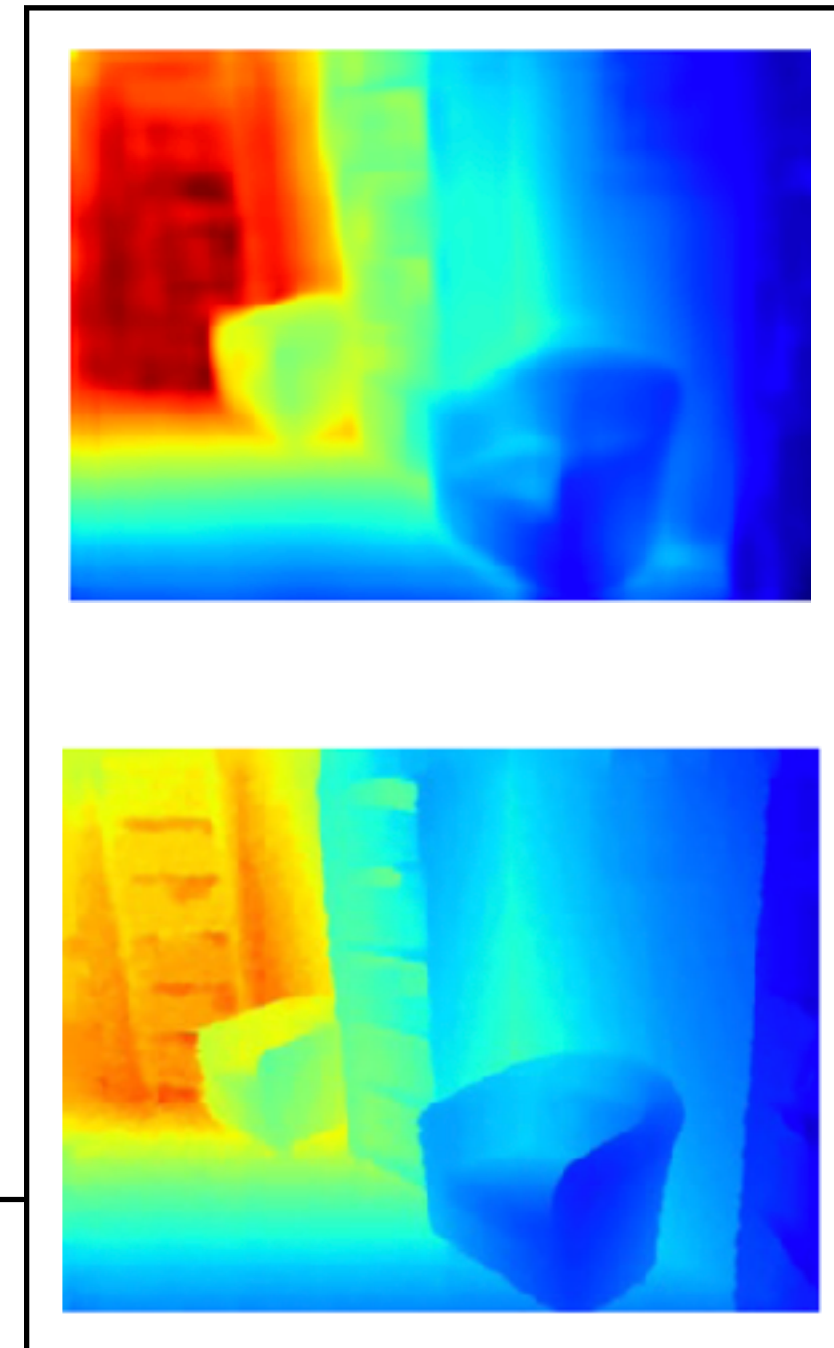


Single-View Depth Estimation





U-Net with skip connections

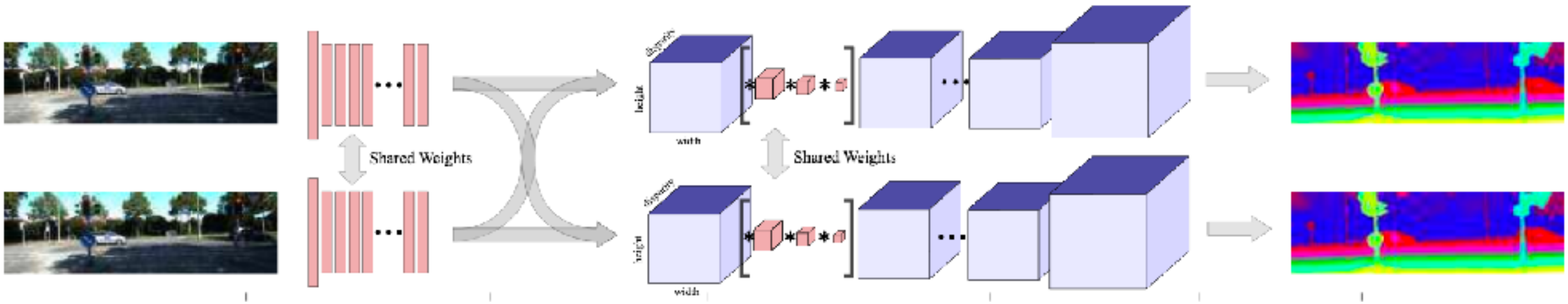


Direct supervision via Kinect RGB+D

Loss, e.g., L2

2-view Stereo

- Form HxWxD=disparity volume and use 3D convolution



Extract features at each pixel using 2D CNN

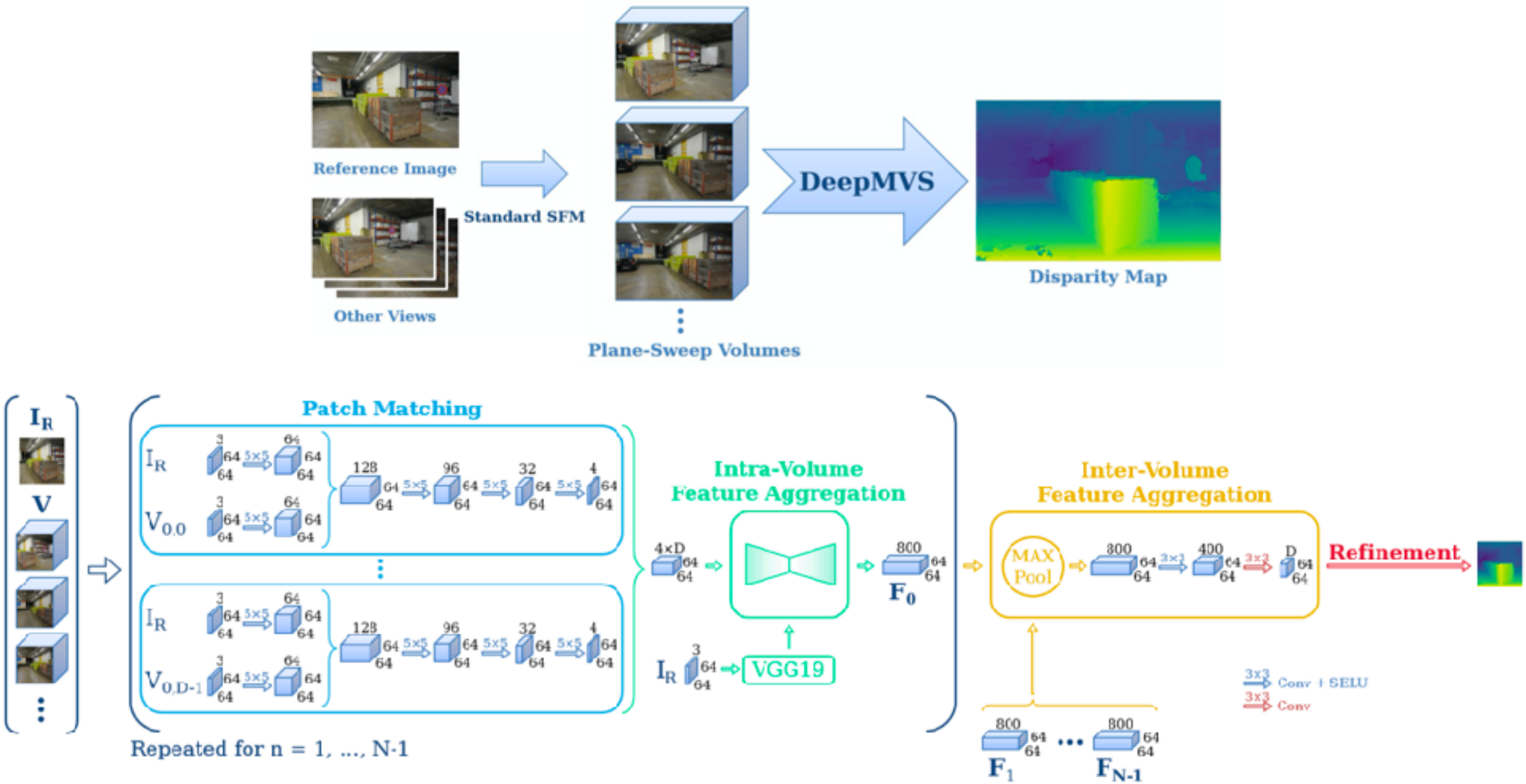
Form volume by sliding features from 2nd image at D disparities

Perform 3D convolution on feature volume

Treat output as disparity cost volume and perform soft argmax

https://www.youtube.com/watch?v=VtAzDS1NLmo [Kendall et al. 2017]

Multi-view Stereo

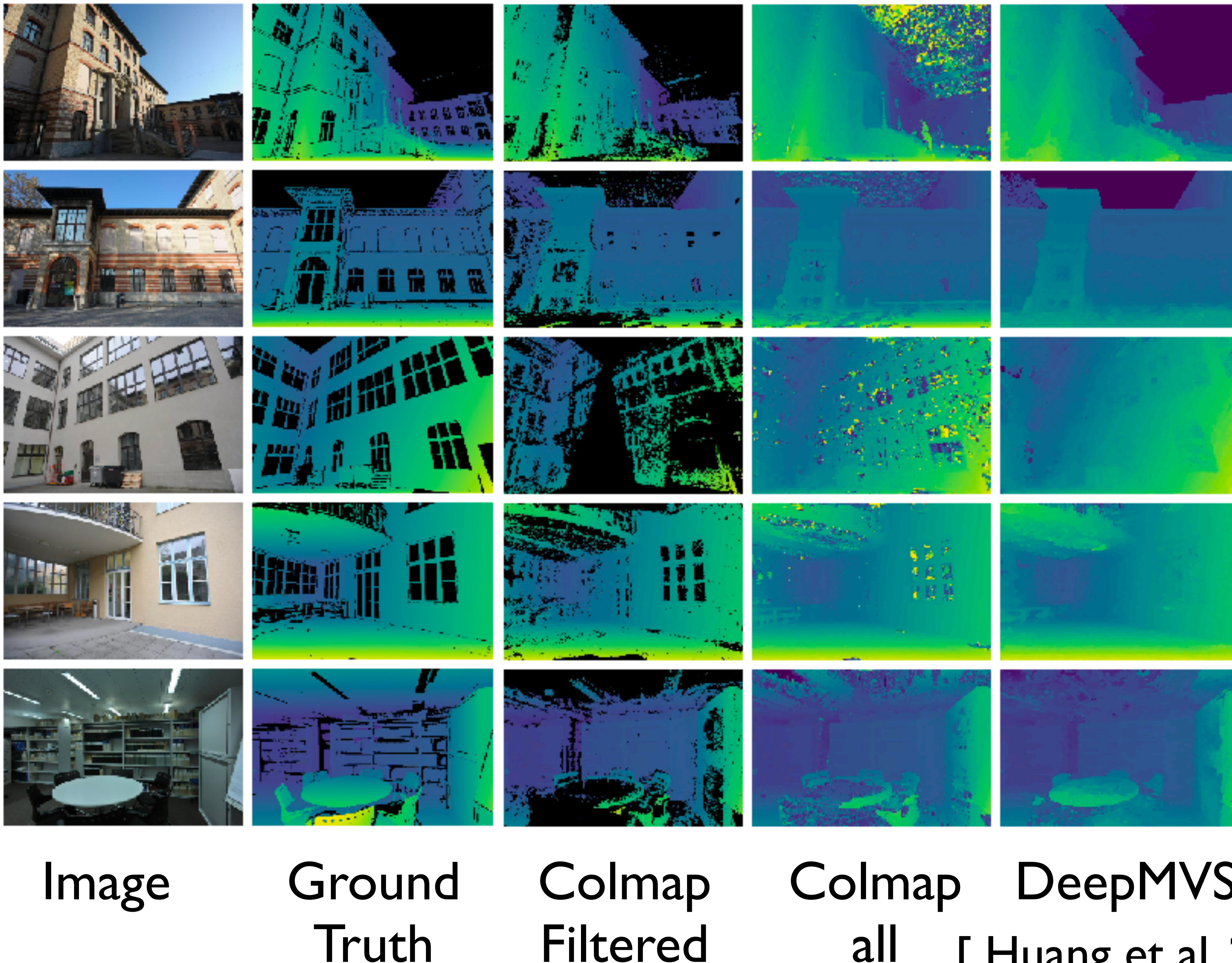


Compare patches in ref image to plane sweep volumes from other images

Perform intra and inter-volume aggregation of features

[DeepMVS, Huang et al. 2018]

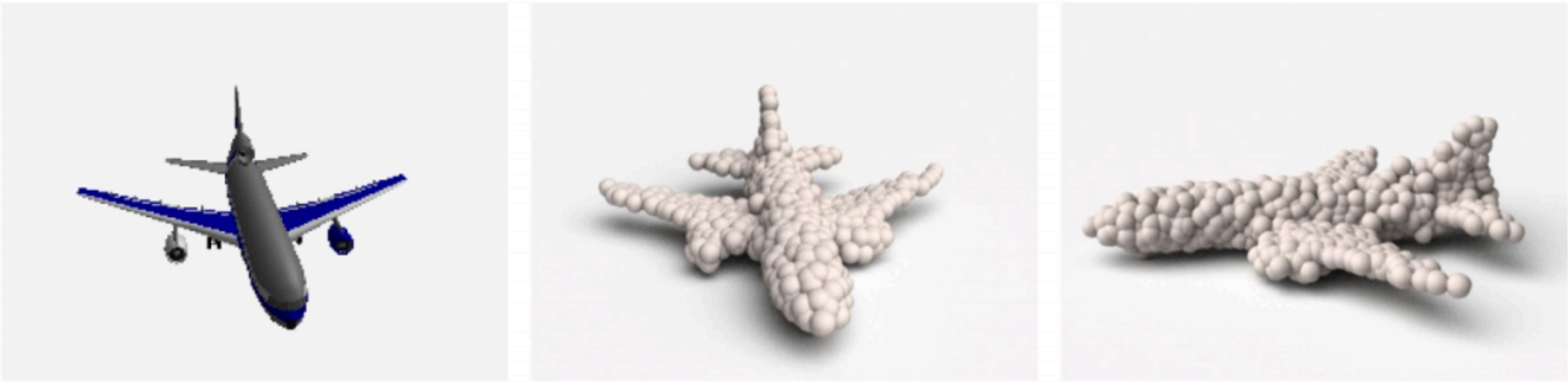
DeepMVS: Results



[Huang et al. 2018]

3D Shape Representations: Point Cloud

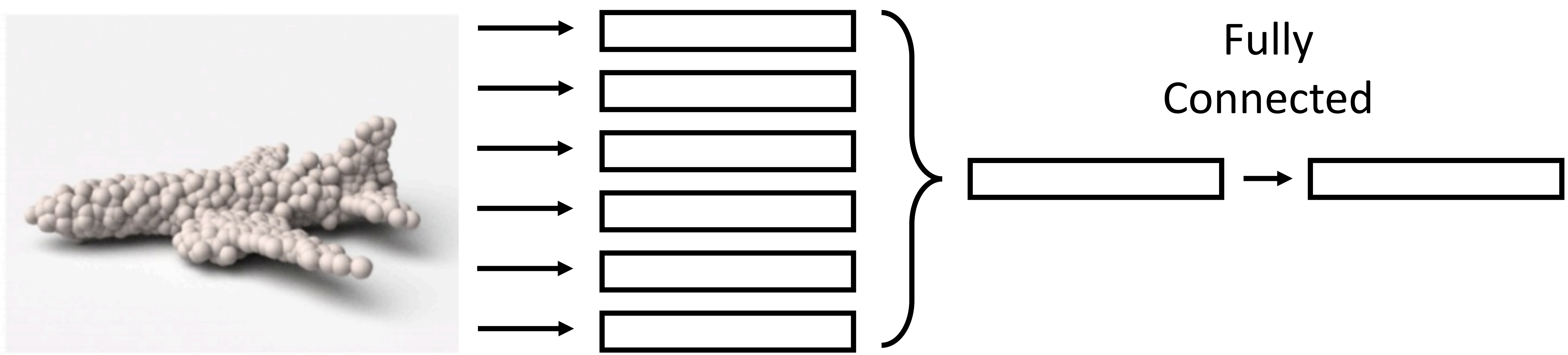
- Represent shape as a set of P points in 3D space
- (+) Can represent fine structures without huge numbers of points
- ( ) Requires new architecture, losses, etc
- (-) Doesn't explicitly represent the surface of the shape: extracting a mesh for rendering or other applications requires post-processing



Fan et al, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image", CVPR 2017

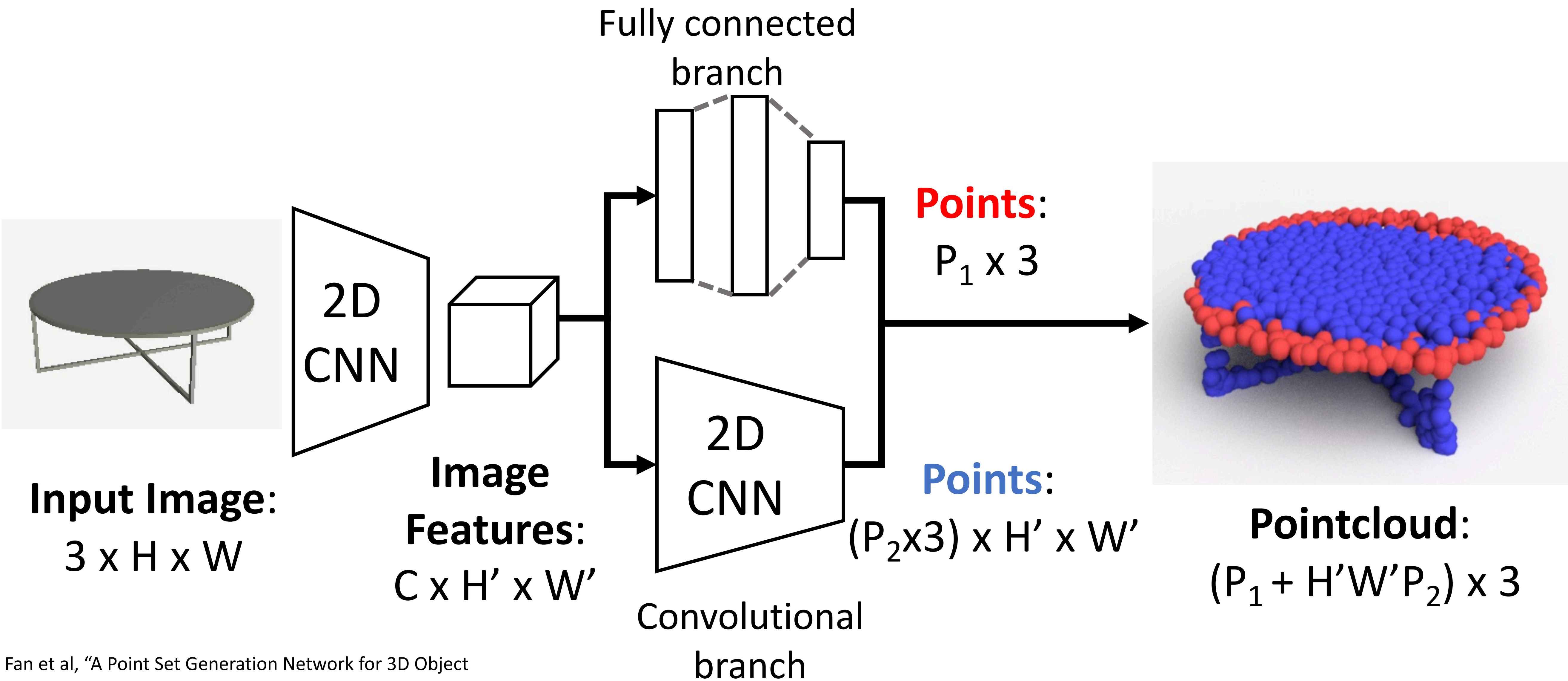
Processing Pointcloud Inputs: PointNet

Want to process pointclouds as **sets**: order should not matter



Qi et al, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation", CVPR 2017
Qi et al, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space", NeurIPS 2017

Generating Pointcloud Outputs



Fan et al, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image", CVPR 2017

Predicting Point Clouds: Loss Function

We need a (differentiable) way to compare pointclouds **as sets**!

Chamfer distance is the sum of L2 distance to each point's nearest neighbor in the other set

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2$$

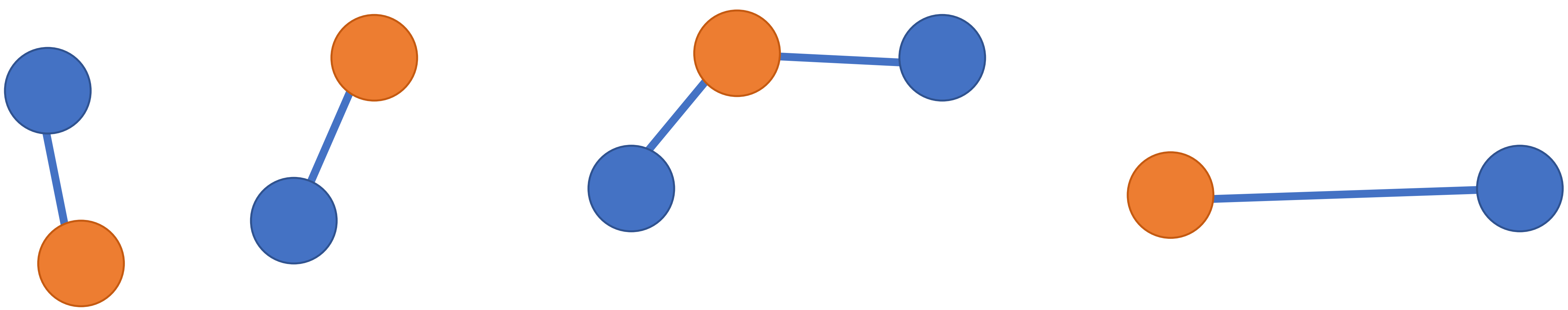
Fan et al, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image", CVPR 2017

Predicting Point Clouds: Loss Function

We need a (differentiable) way to compare pointclouds **as sets**!

Chamfer distance is the sum of L2 distance to each point's nearest neighbor in the other set

$$d_{CD}(\boxed{S_1}, \boxed{S_2}) = \boxed{\sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2} + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2$$



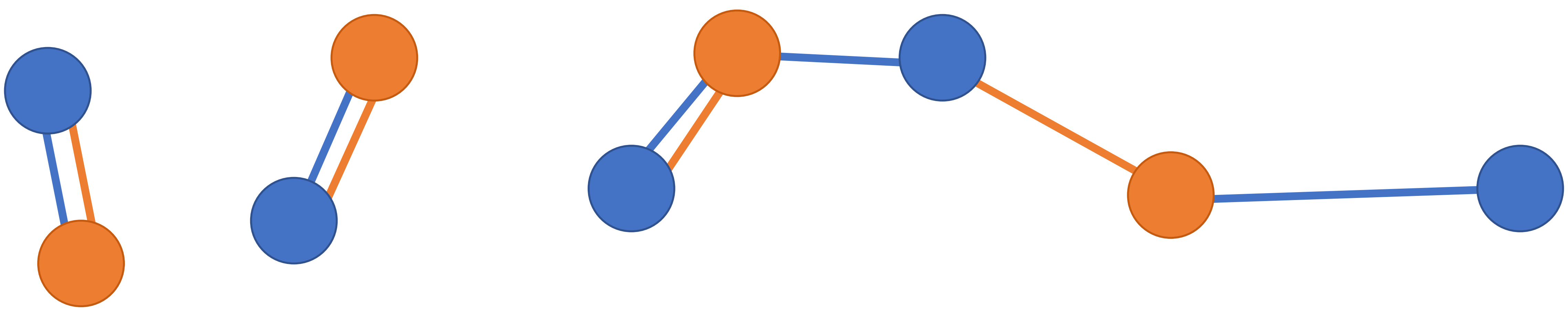
Fan et al, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image", CVPR 2017

Predicting Point Clouds: Loss Function

We need a (differentiable) way to compare pointclouds **as sets**!

Chamfer distance is the sum of L2 distance to each point's nearest neighbor in the other set

$$d_{CD}(\boxed{S_1}, \boxed{S_2}) = \boxed{\sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2} + \boxed{\sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2}$$



Fan et al, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image", CVPR 2017

ICP-like distance function

3D Shape Representations: Triangle Mesh

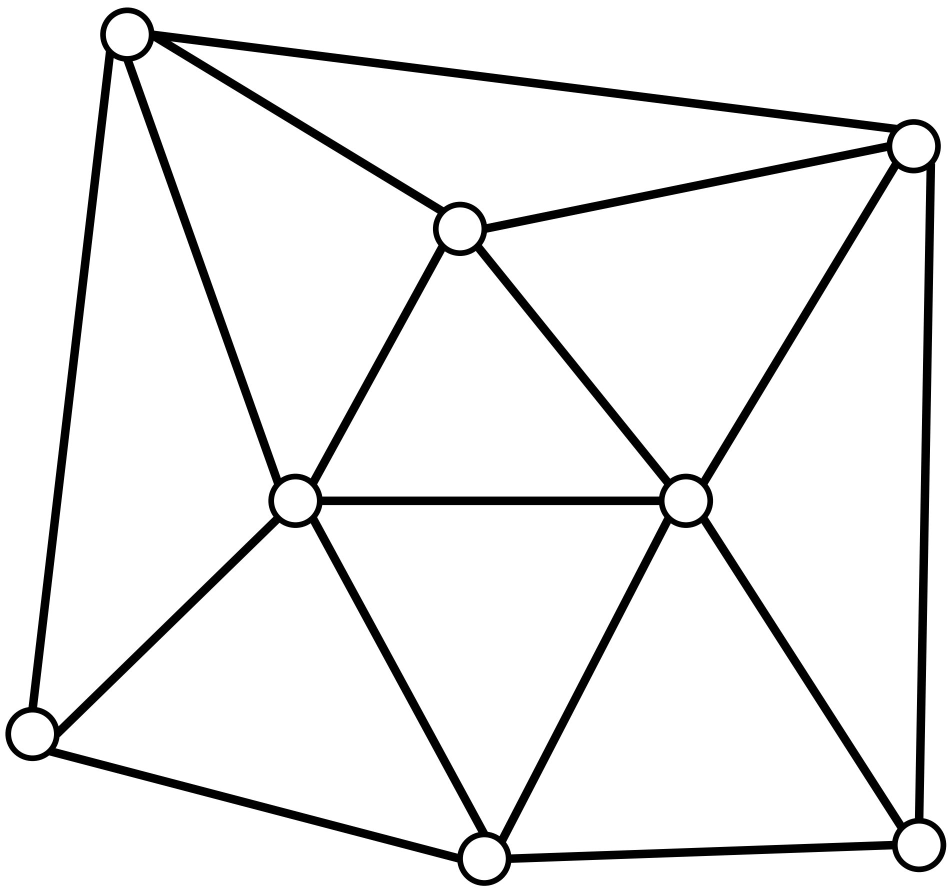
Represent a 3D shape as a set of triangles

Vertices: Set of V points in 3D space

Faces: Set of triangles over the vertices

(+) Standard representation for graphics

(+) Explicitly represents 3D shapes



3D Shape Representations: Triangle Mesh

Represent a 3D shape as a set of triangles

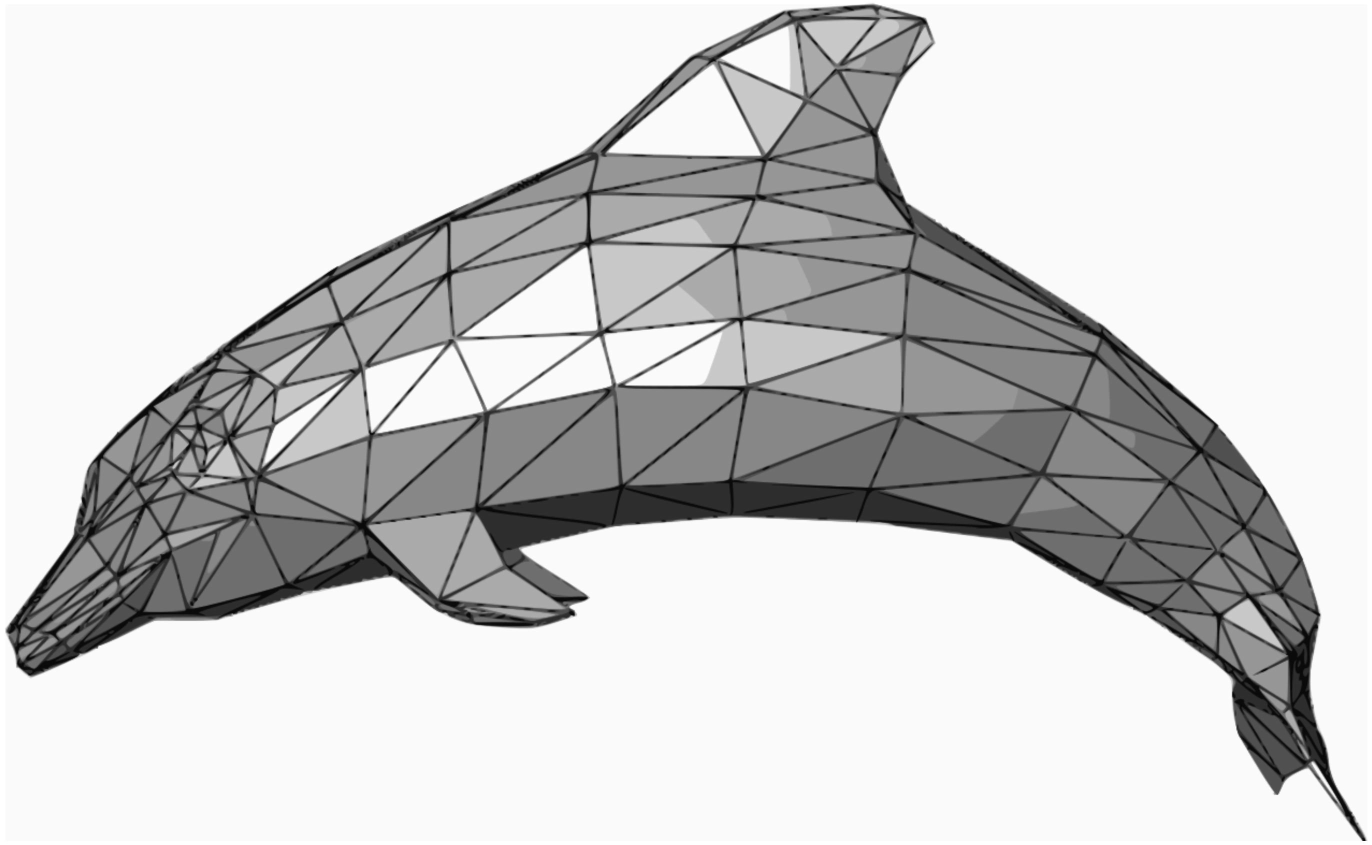
Vertices: Set of V points in 3D space

Faces: Set of triangles over the vertices

(+) Standard representation for graphics

(+) Explicitly represents 3D shapes

(+) Adaptive: Can represent flat surfaces very efficiently, can allocate more faces to areas with fine detail



Dolphin image is in the public domain

3D Shape Representations: Triangle Mesh

Represent a 3D shape as a set of triangles

Vertices: Set of V points in 3D space

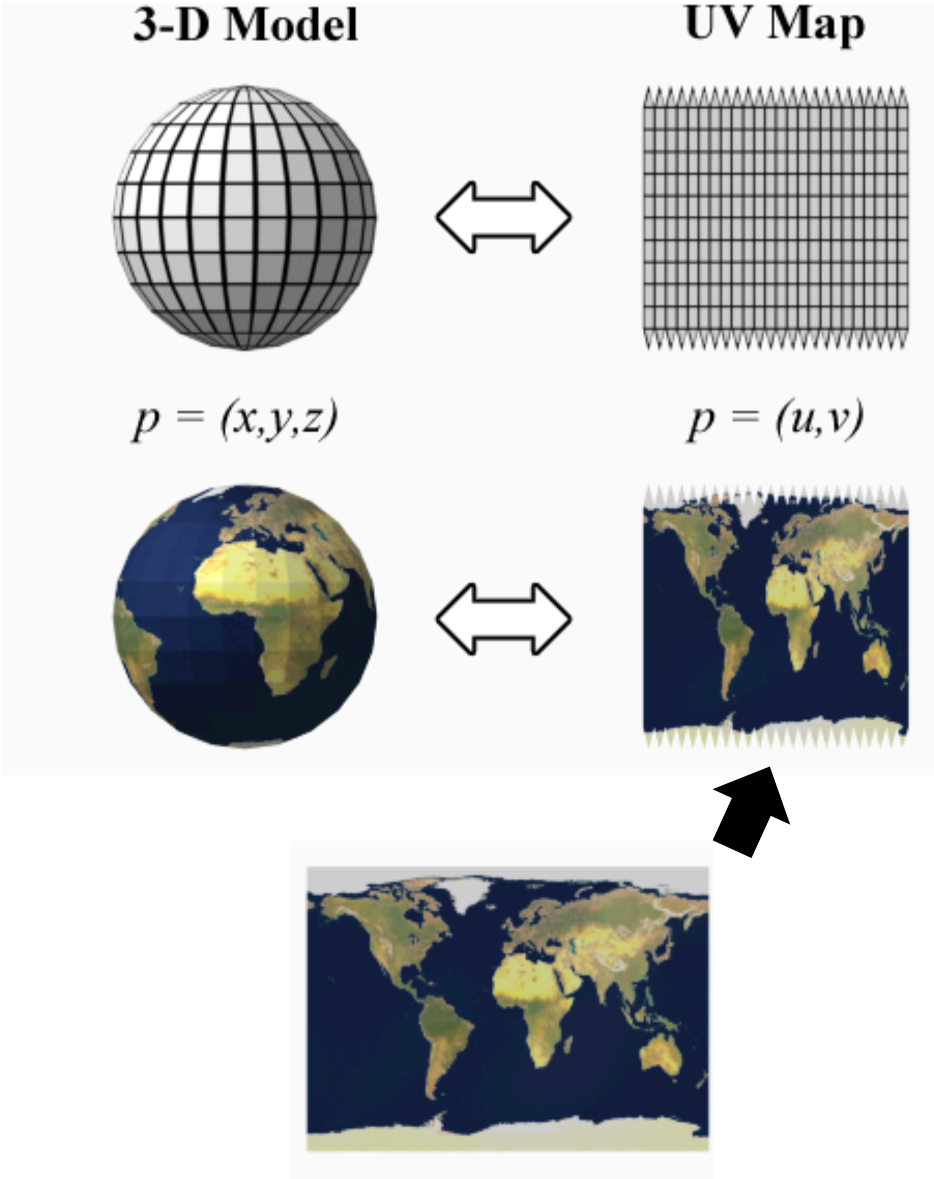
Faces: Set of triangles over the vertices

(+) Standard representation for graphics

(+) Explicitly represents 3D shapes

(+) Adaptive: Can represent flat surfaces very efficiently, can allocate more faces to areas with fine detail

(+) Can attach data on verts and interpolate over the whole surface: RGB colors, texture coordinates, normal vectors, etc.

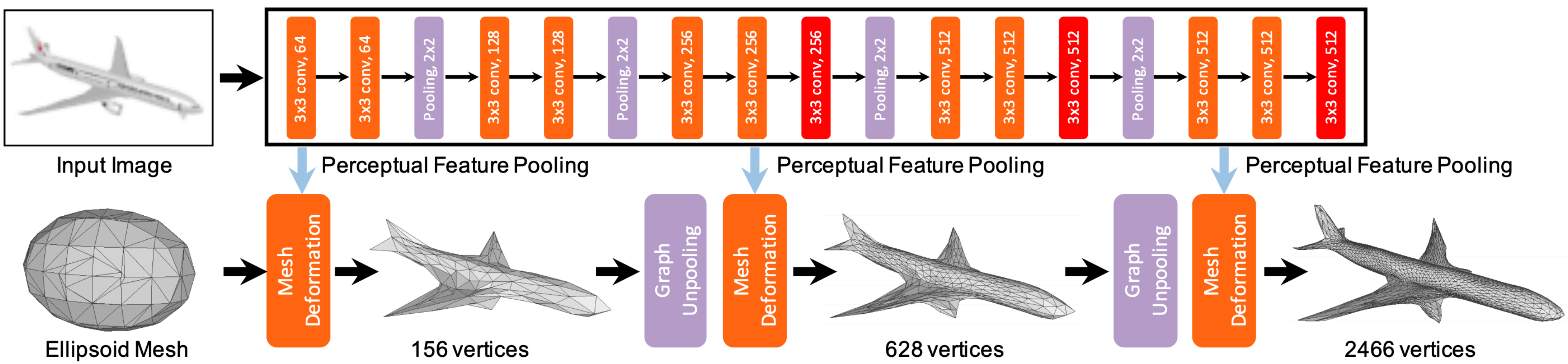


Predicting Meshes: Pixel2Mesh

Input: Single RGB Image of an object

Key ideas:
Iterative Refinement
Graph Convolution
Vertex Aligned-Features
Chamfer Loss Function

Output: Triangle mesh for the object



Wang et al, "Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images", ECCV 2018

Supervised with ground truth meshes

Predicting Triangle Meshes: Iterative Refinement

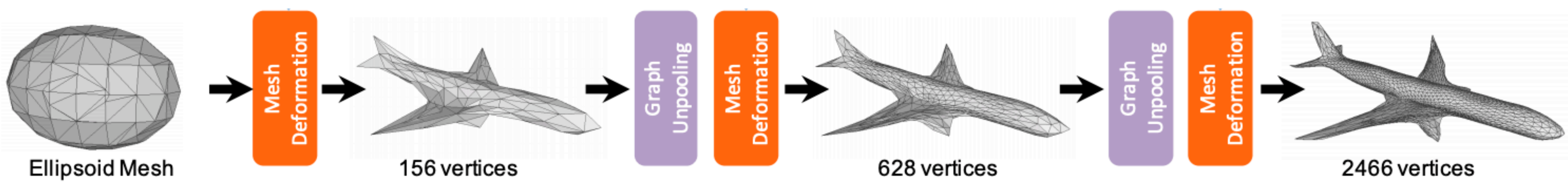
Idea #1: Iterative mesh refinement

Start from initial ellipsoid mesh

Network predicts offsets for each vertex

Repeat.

Fixed mesh structure



Wang et al, "Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images", ECCV 2018

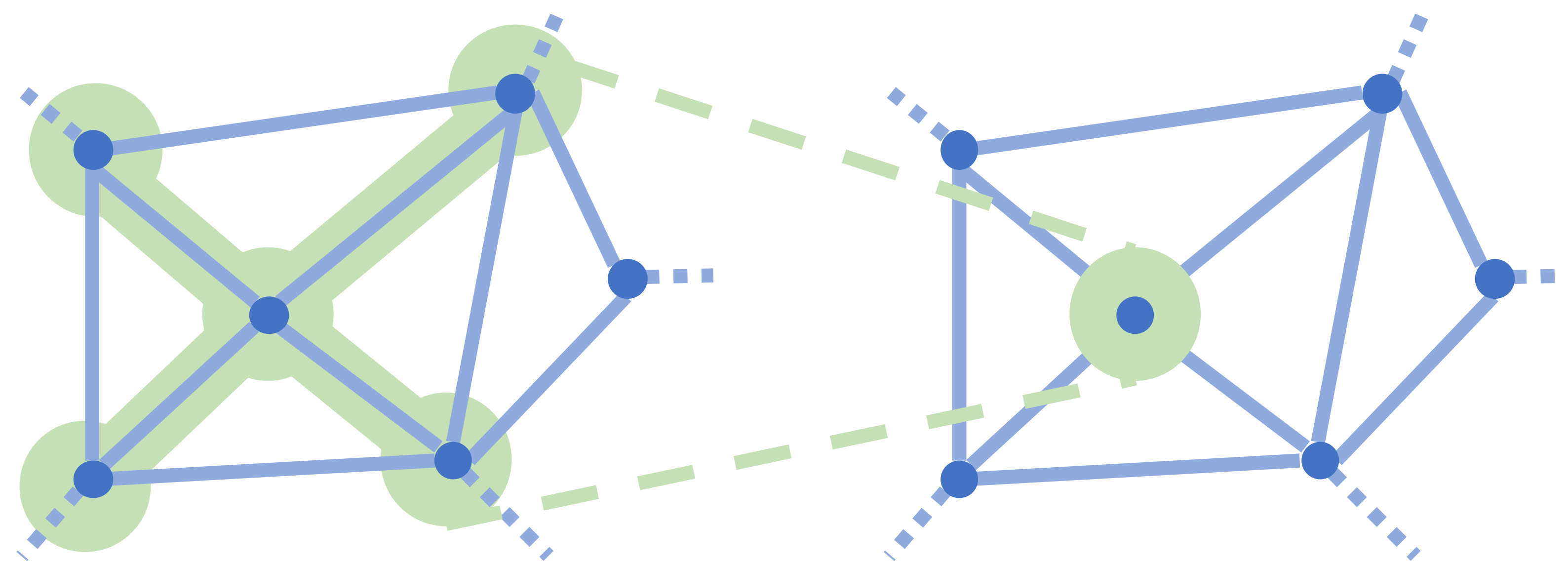
Predicting Triangle Meshes: Graph Convolution

$$f'_i = W_0 f_i + \sum_{j \in \mathcal{N}(i)} W_1 f_j$$

Vertex v_i has feature f_i

New feature f'_i for vertex vi depends on feature of neighboring vertices N(i)

Use same weights W0 and W1 to compute all outputs

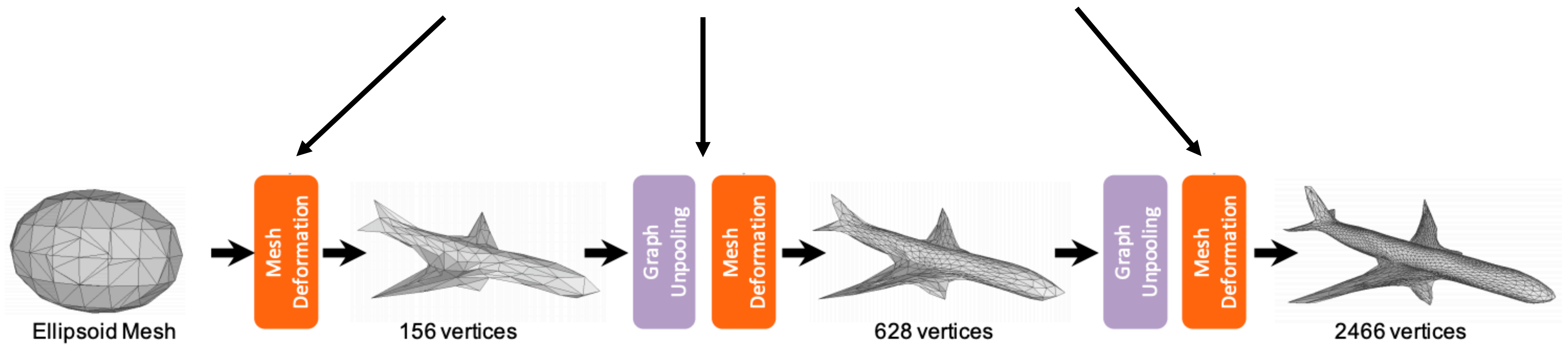


Input: Graph with a feature vector at each vertex

Output: New feature vector for each vertex

Predicting Triangle Meshes: Graph Convolution

Each of these blocks consists of a stack of **graph convolution layers** operating on edges of the mesh



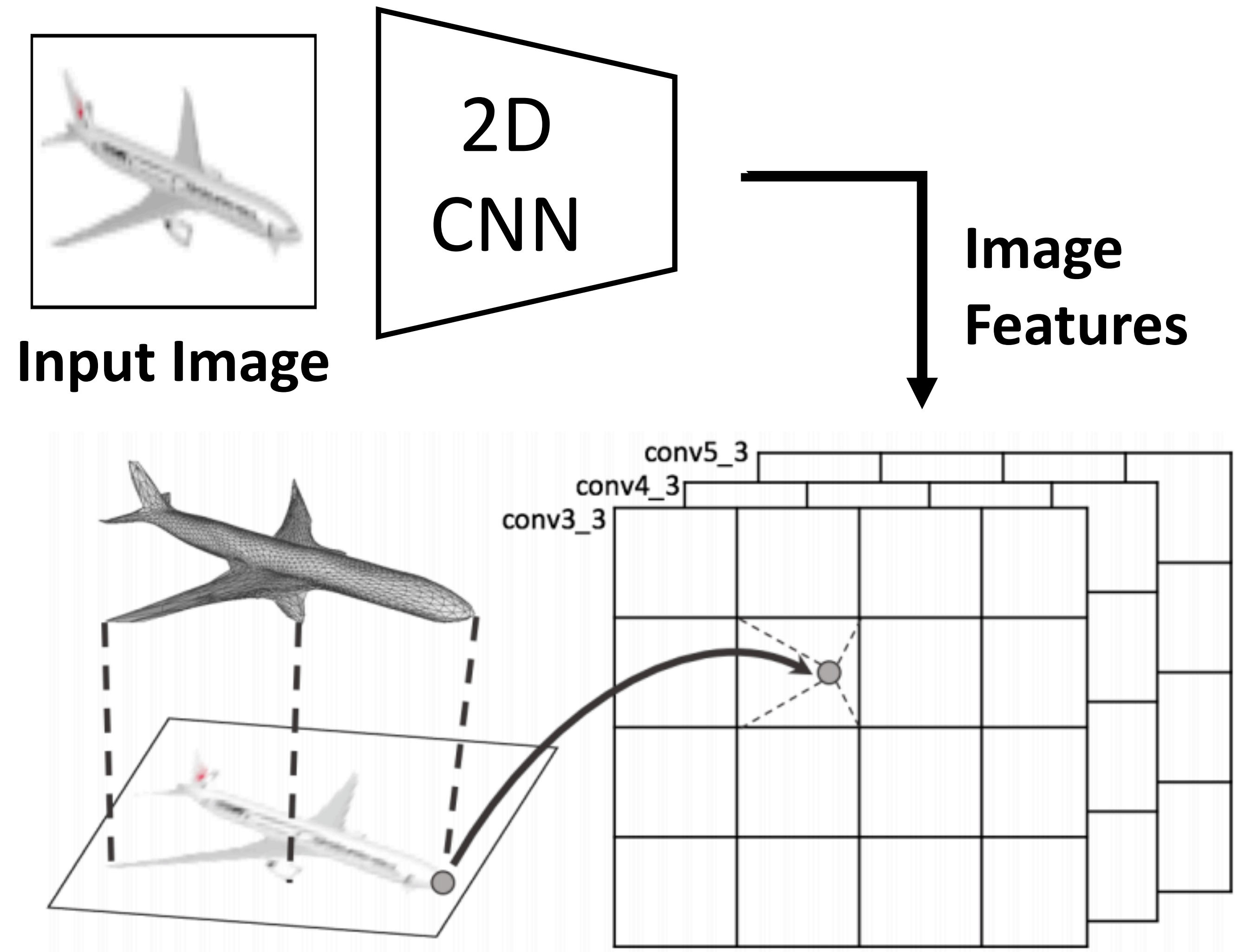
Wang et al, "Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images", ECCV 2018

Predicting Triangle Meshes: Vertex-Aligned Features

Idea #2: Aligned vertex features

For each vertex of the mesh:

- Use camera information to project onto image plane
- Use bilinear interpolation to sample a CNN feature

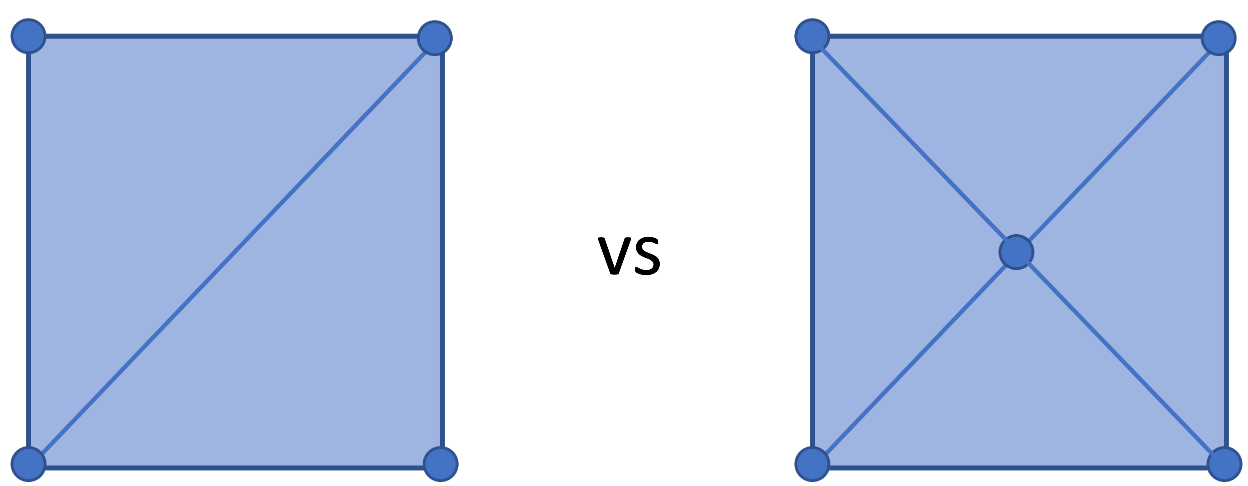


Wang et al, "Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images", ECCV 2018

Predicting Meshes: Loss Function

The same shape can be represented with different meshes – how can we define a loss between predicted and ground-truth mesh?

Idea: Convert meshes to pointclouds, then compute loss



Prediction

Ground-Truth

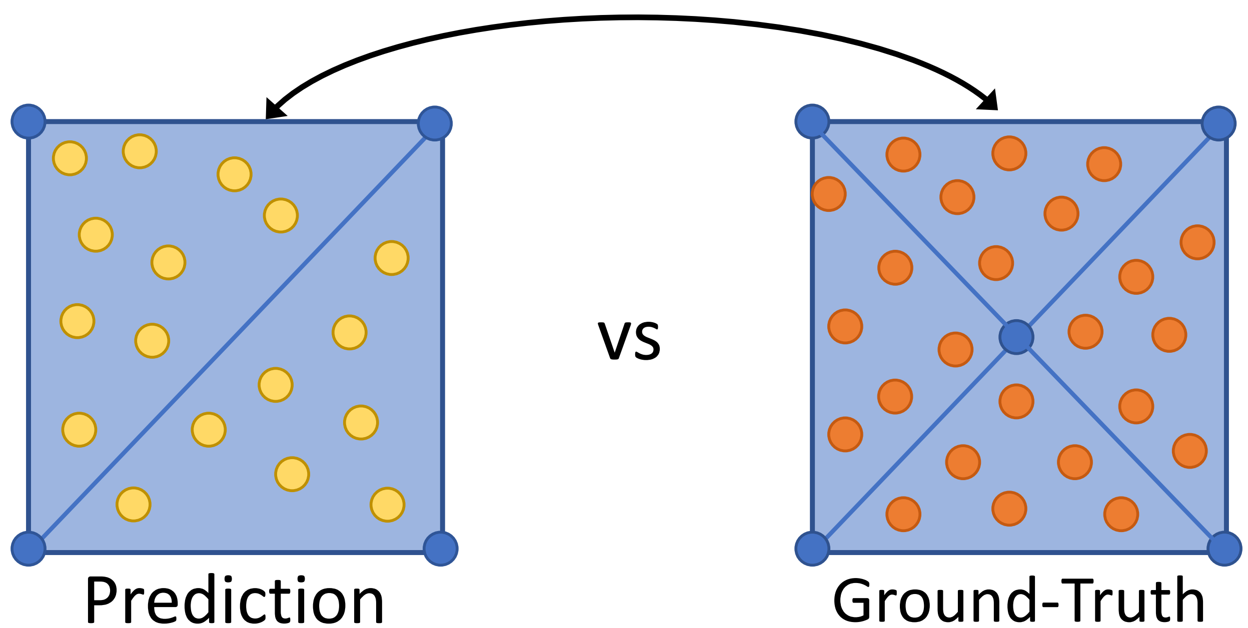
Wang et al, "Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images", ECCV 2018

Predicting Meshes: Loss Function

The same shape can be represented with different meshes – how can we define a loss between predicted and ground-truth mesh?

Loss = Chamfer distance between **predicted samples** and **ground-truth samples**

Sample points from the surface of the predicted mesh (online!)



Sample points from the surface of the ground-truth mesh (offline)

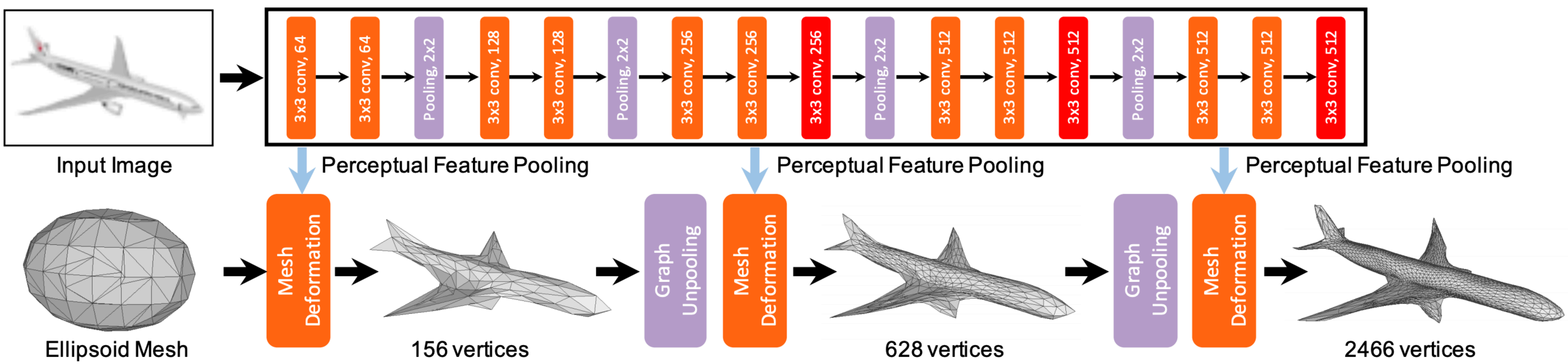
Smith et al, "GEOMetrics: Exploiting Geometric Structure for Graph-Encoded Objects", ICML 2019

Predicting Meshes: Pixel2Mesh

Input: Single RGB Image of an object

Key ideas:
Iterative Refinement
Graph Convolution
Vertex Aligned-Features
Chamfer Loss Function

Output: Triangle mesh for the object

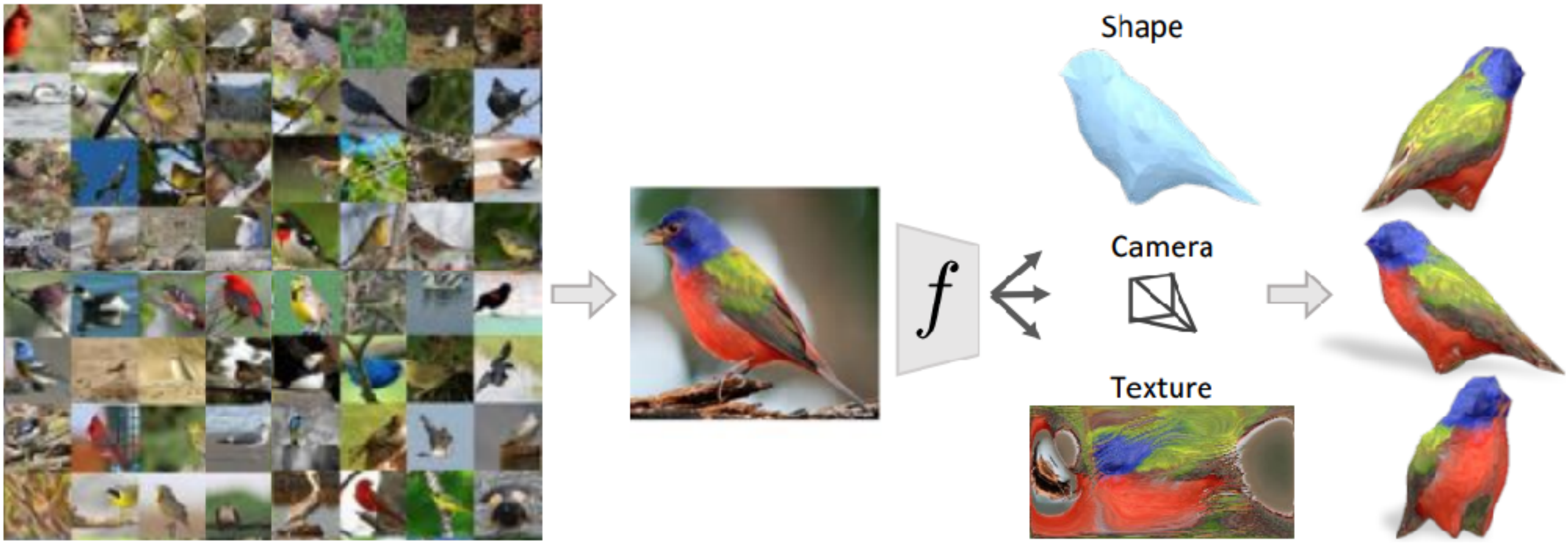


Wang et al, "Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images", ECCV 2018

Supervised with ground truth meshes

Category Specific Mesh Reconstruction

- Can we learn without ground truth meshes?



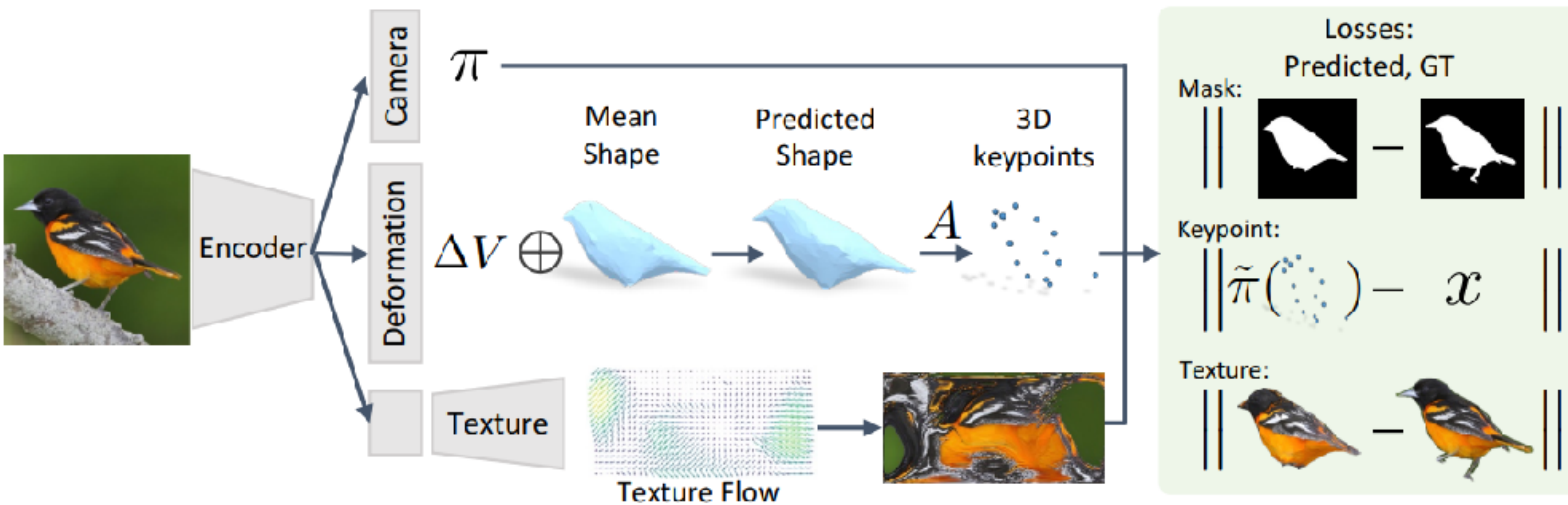
Given an image, infer mesh, camera, texture

[Kanazawa et al. 2018]

Data = Caltech-UCSD birds CUB-200-2011, 6000 images of 200 bird species, + segmentation, 14 semantic keypoints, remove 300 images where num visible keypoints <= 6

Category Specific Mesh Reconstruction

- Train a model to predict object mesh (deformation of mean category shape) + camera pose

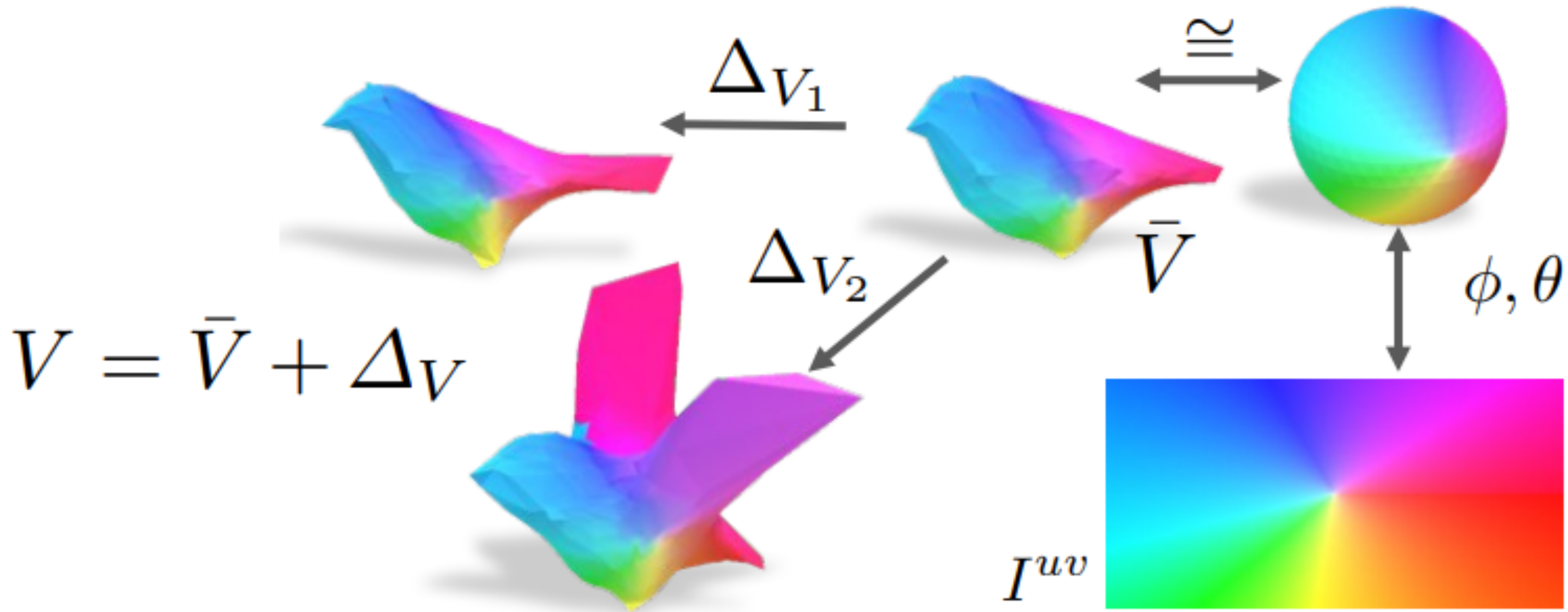


- Use semantic keypoints and object masks to learn shape (texture not used to learn shape in this implementation)

[Kanazawa et al. 2018]

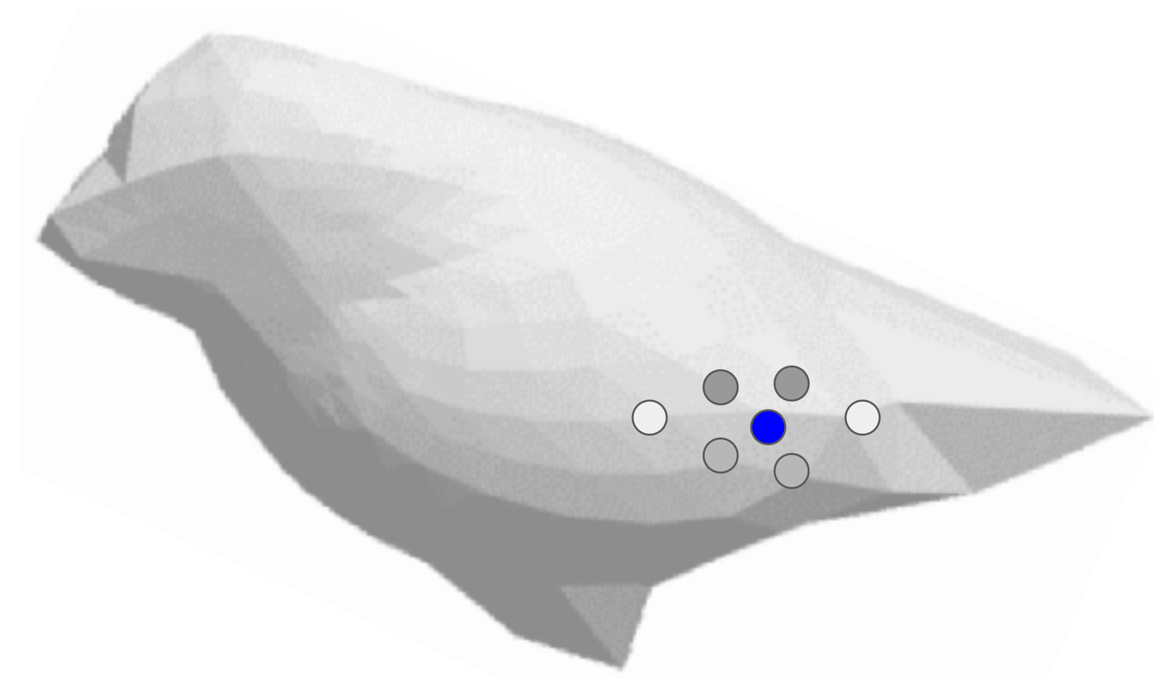
Mesh Parametrization

- Fixed spherical mesh (subdivided icosahedron) 642 vertices V 1280 faces
- Instances are deformations ΔV of a mean class shape V
- Texture is modelled as RGB colour in spherical coordinates



Keypoints and Projection

- Semantic keypoint positions are modelled as weighted vertex positions
- Matrix A is learned per-class, can be viewed as per-vertex probabilities, with keypoints as the expected value
- Projection **π** is modelled by a camera with translation t, rotation q (quaternion) and scale s



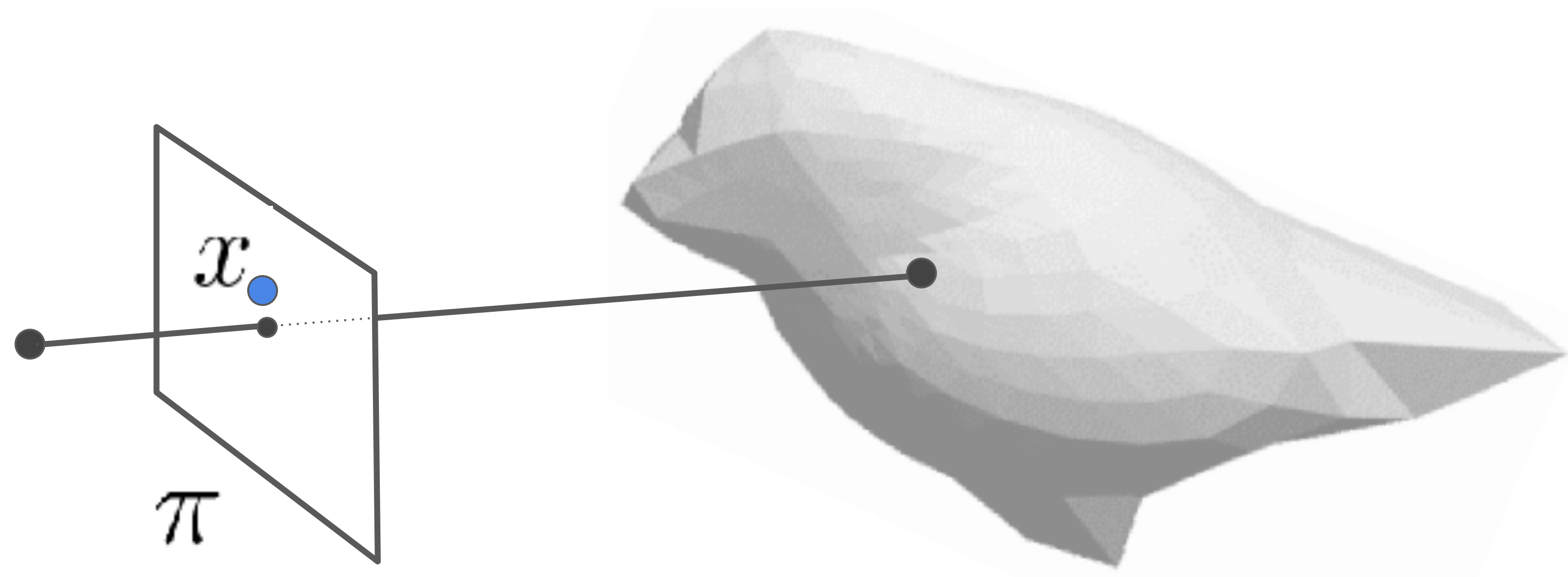
$$v_k = \sum_v A_{k,v} v$$

$$A \in \mathcal{R}_+^{|K| \times |V|}$$

$A \cdot V$ = set of keypoint positions

Keypoint Projection Loss

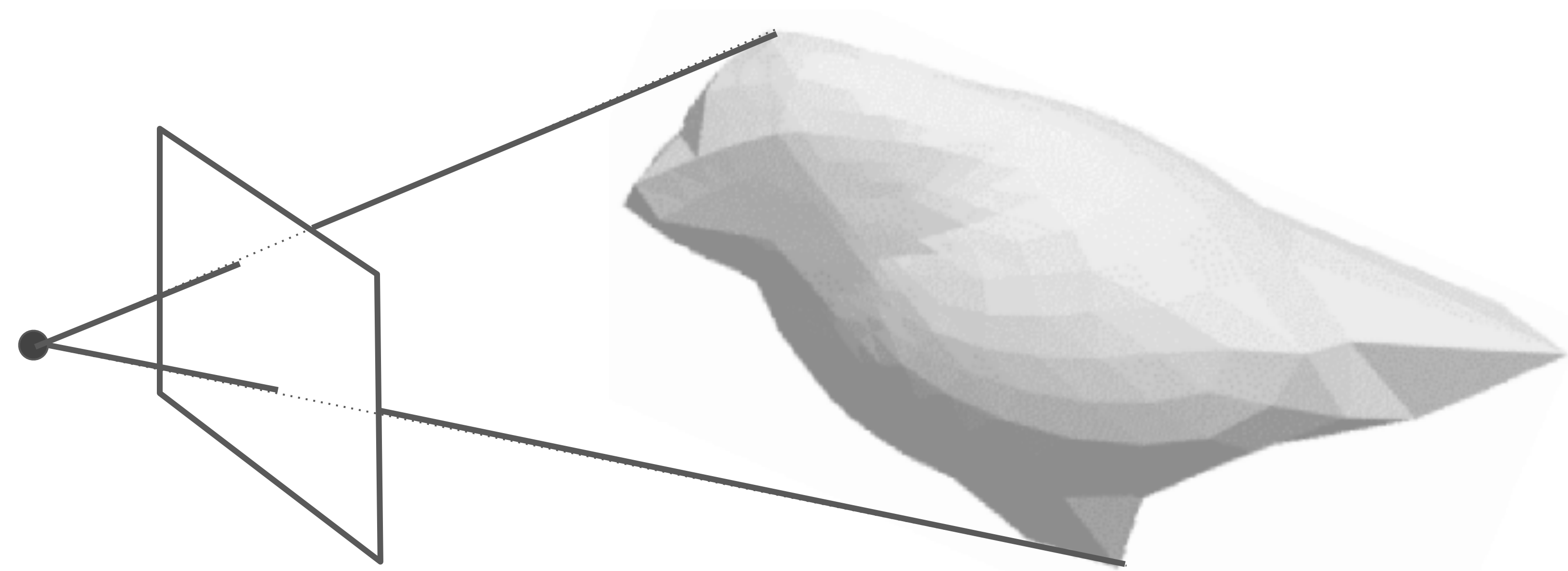
- Ensure that keypoints (parametrized as weighted vertex positions) map to the known positions xi. Note: weightings A are per class, vertices V per instance



$$L_{\texttt{reproj}} = \sum_i ||x_i - \tilde{\pi}_i(AV_i)||_2$$

Mask Projection Loss

- Ensure that the mesh maps to the known silhouette. Note: gradient depends on rendering the mesh

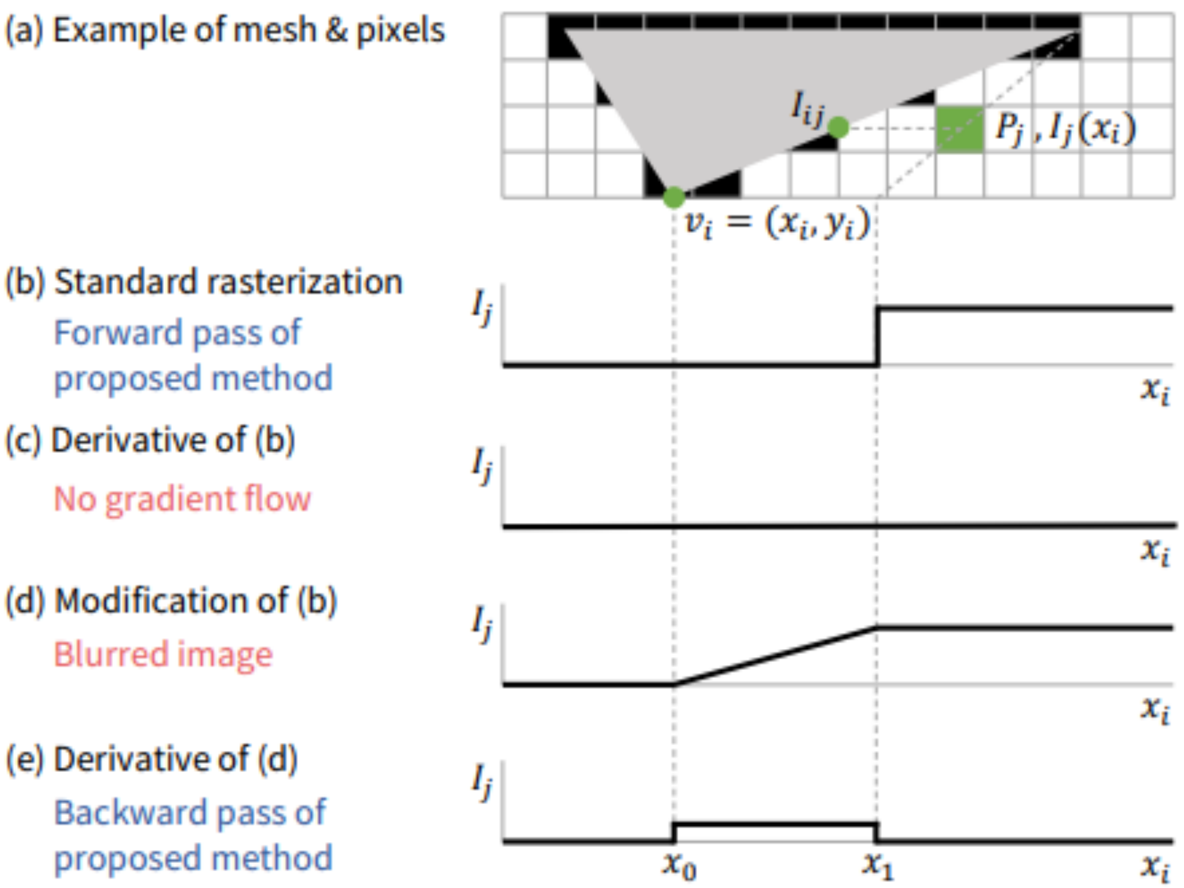


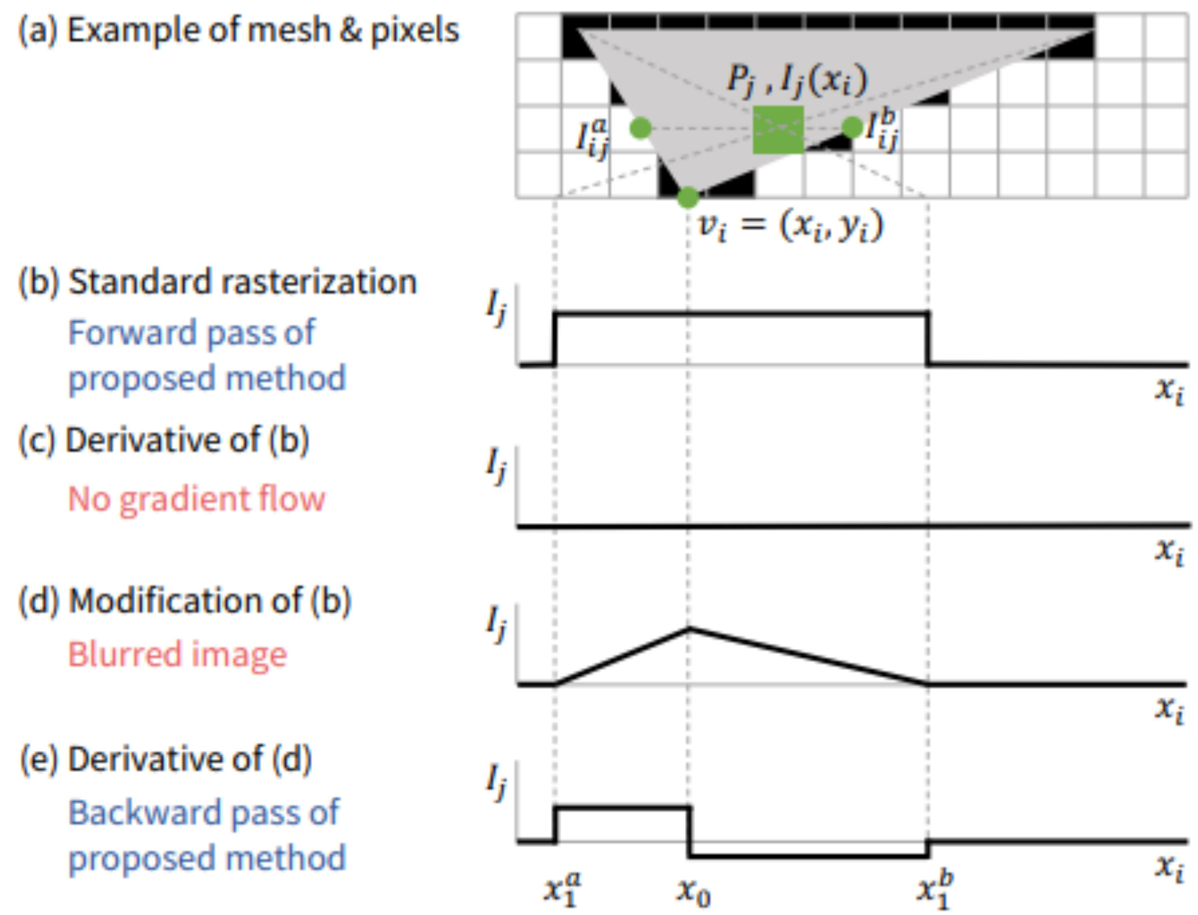
$$L_{\texttt{mask}} = \sum_i ||S_i - \mathcal{R}(V_i, F, \tilde{\pi}_i)||_2$$

S = silhouette, R(.) mesh rendering

Gradient of Mesh Render

Extend gradient for each pixel inside/ outside triangles with linear ramp

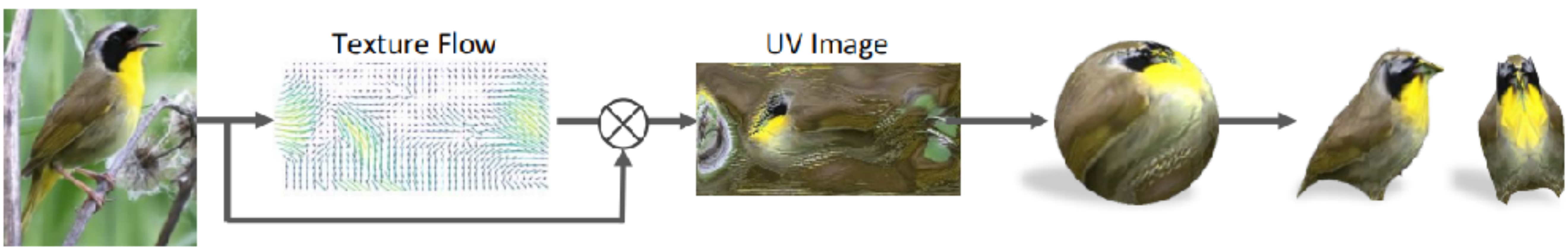




[Neural 3D Mesh Renderer, Kato et al. CVPR 2018]

Texture Representation

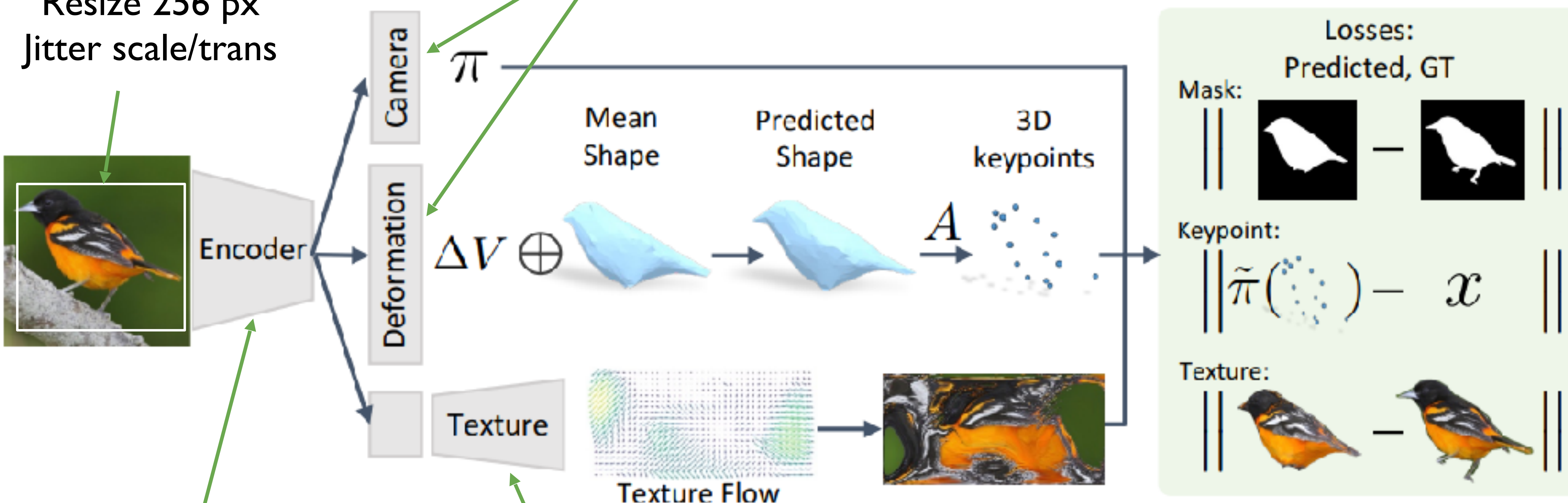
- Texture is parametrized as coordinates (flow) of the input image I(u,v)
- → each point on the reference sphere is given a coordinate in the input image
- Latent representation is upconvolved to generate flow I(u,v)
- Loss is Zhang et al. perceptual loss [1] of projected texture
- Note: texture loss is not used to learn shape!



[1] The unreasonable effectiveness of deep networks as a perceptual metric. R. Zhang et al. CVPR 2018

Crop to BBox
Resize 256 px
Jitter scale/trans

Linear layers



ResNet18 (ImageNet)
Conv ↓2 space/channel
Vectorize → 4096D
2 x fully conn → 200D

5 upconv layers, tanh
activation to constrain
to [-1,1] coords

(not used to learn shape!)

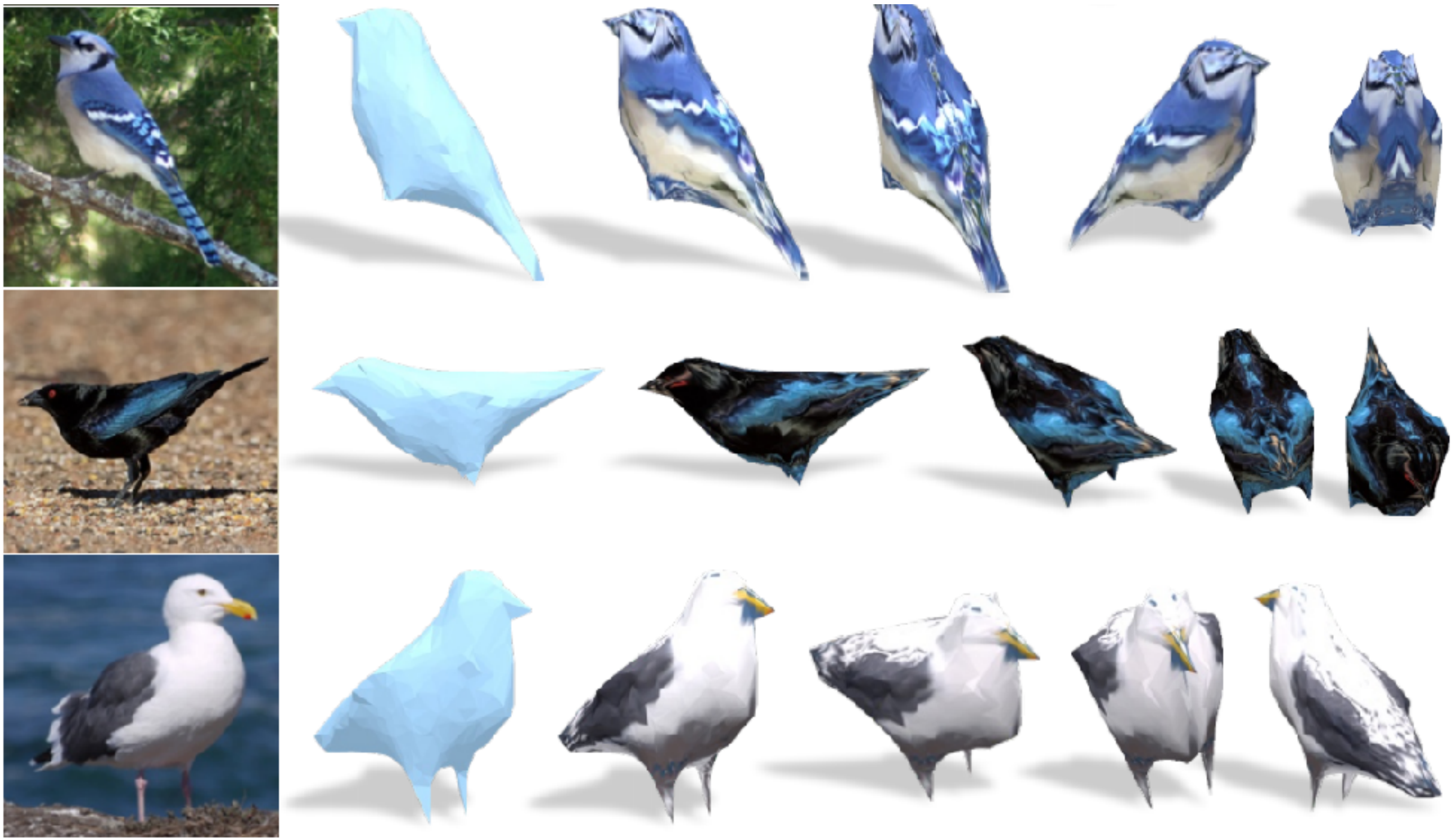
SFM Initialization

- In principle, camera π, mean shape V, instance shape ΔV, keypoint weightings A could be learned from supervised keypoint and silhouette losses
- In practice, the authors initialize cameras π and mean shape V via SFM
- Note this involves bundle adjustment / optimization over different birds, so results in fitting an "average" bird model
- The mesh is initialized as the convex hull of keypoint positions, and camera solutions π_hat are recorded

Initial Mean Shape

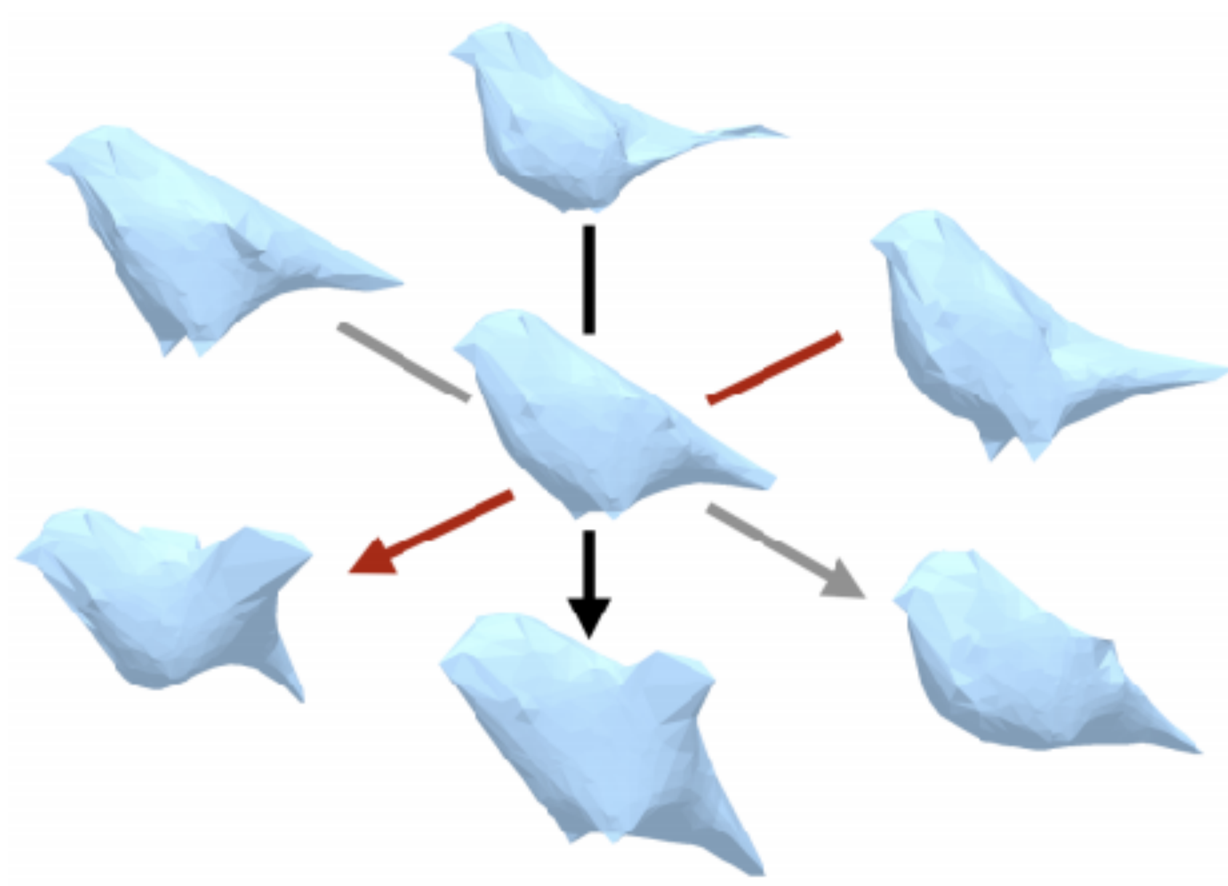
Results

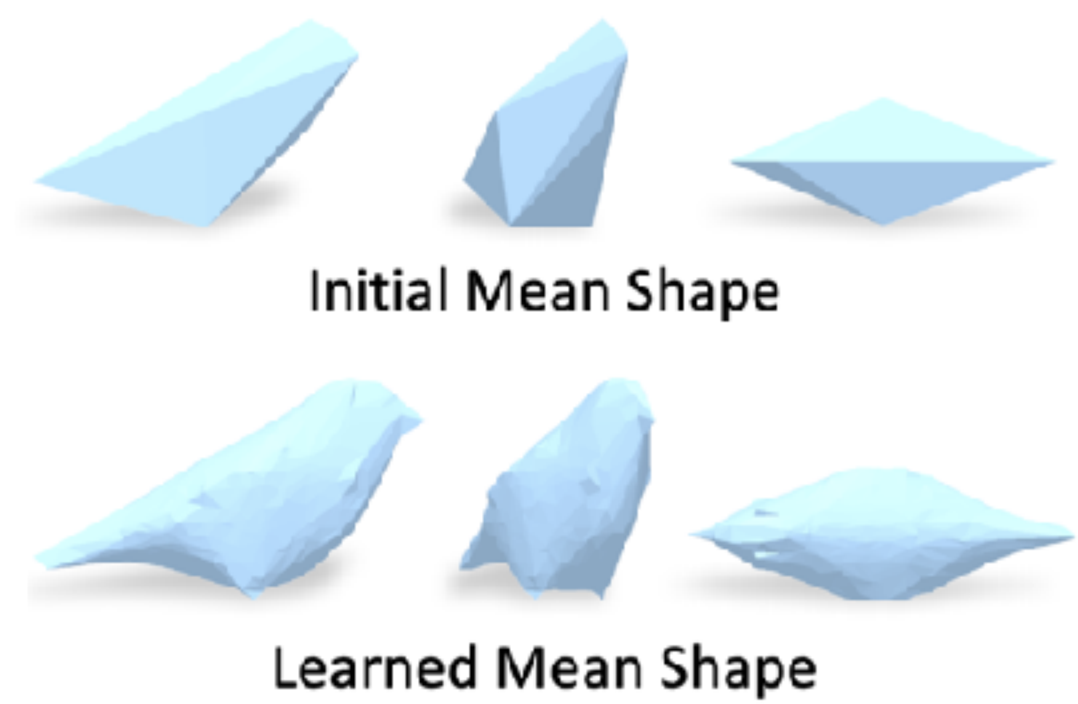
Results



Deformation Modes

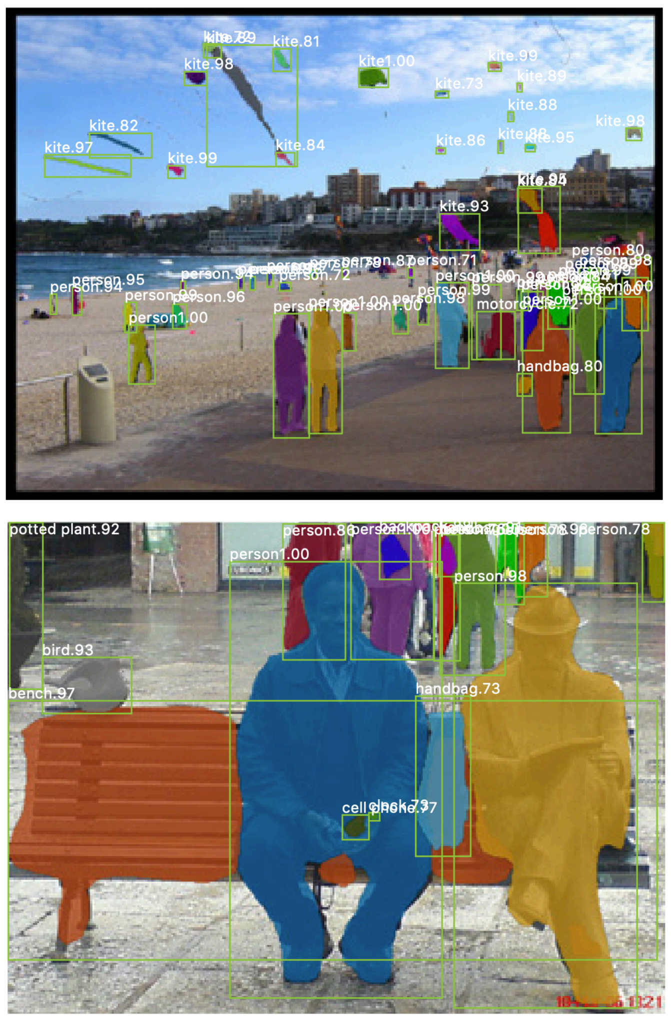
- Mean and first 3 PCA components of bird shapes





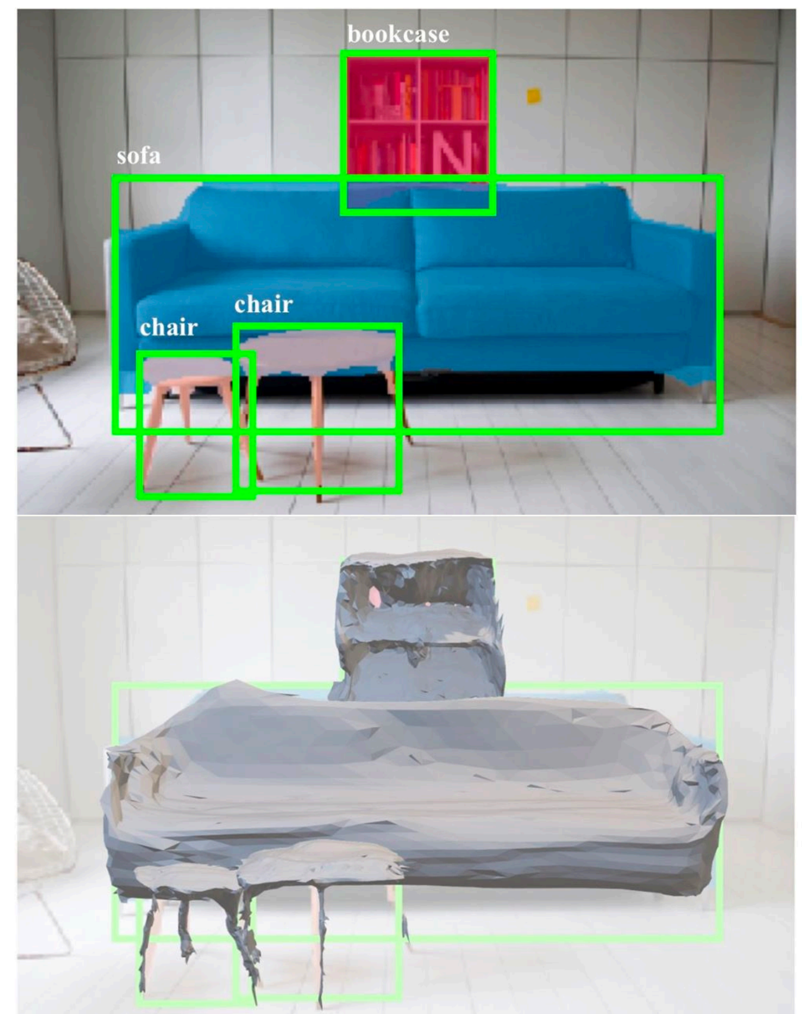
3D Shape Prediction: Mesh R-CNN

Mask R-CNN:
2D Image -> 2D shapes



He, Gkioxari, Dollár, and Girshick, "Mask R-CNN", ICCV 2017

Mesh R-CNN:
2D Image -> **Triangle Meshes**



Gkioxari, Malik, and Johnson, "Mesh R-CNN", ICCV 2019

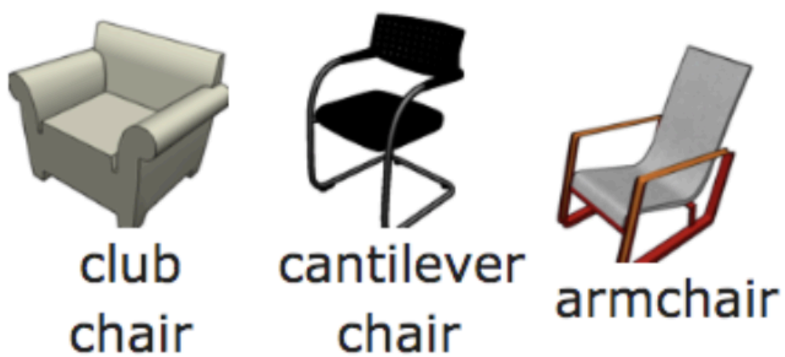
Detect objects and extract silhouettes

Estimate 3D mesh

uses 3D mesh models from IKEA

3D Datasets: Object-Centric

ShapeNet



~50 categories, ~50k 3D CAD models

Standard split has 13 categories, ~44k models, 25 rendered images per model

Many papers show results here

(-) Synthetic, isolated objects; no context

(-) Lots of chairs, cars, airplanes

Pix3D

9 categories, 219 3D models of IKEA furniture aligned to ~17k real images

Some papers train on ShapeNet and show qualitative results here, but use ground-truth segmentation masks

(+) Real images! Context!

(-) Small, partial annotations – only 1 obj/image

Chang et al, “ShapeNet: An Information-Rich 3D Model Repository”, arXiv 2015
Choy et al, “3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction”, ECCV 2016
Sun et al, “Pix3D: Dataset and Methods for Single-Image 3D Shape Modeling”, CVPR 2018

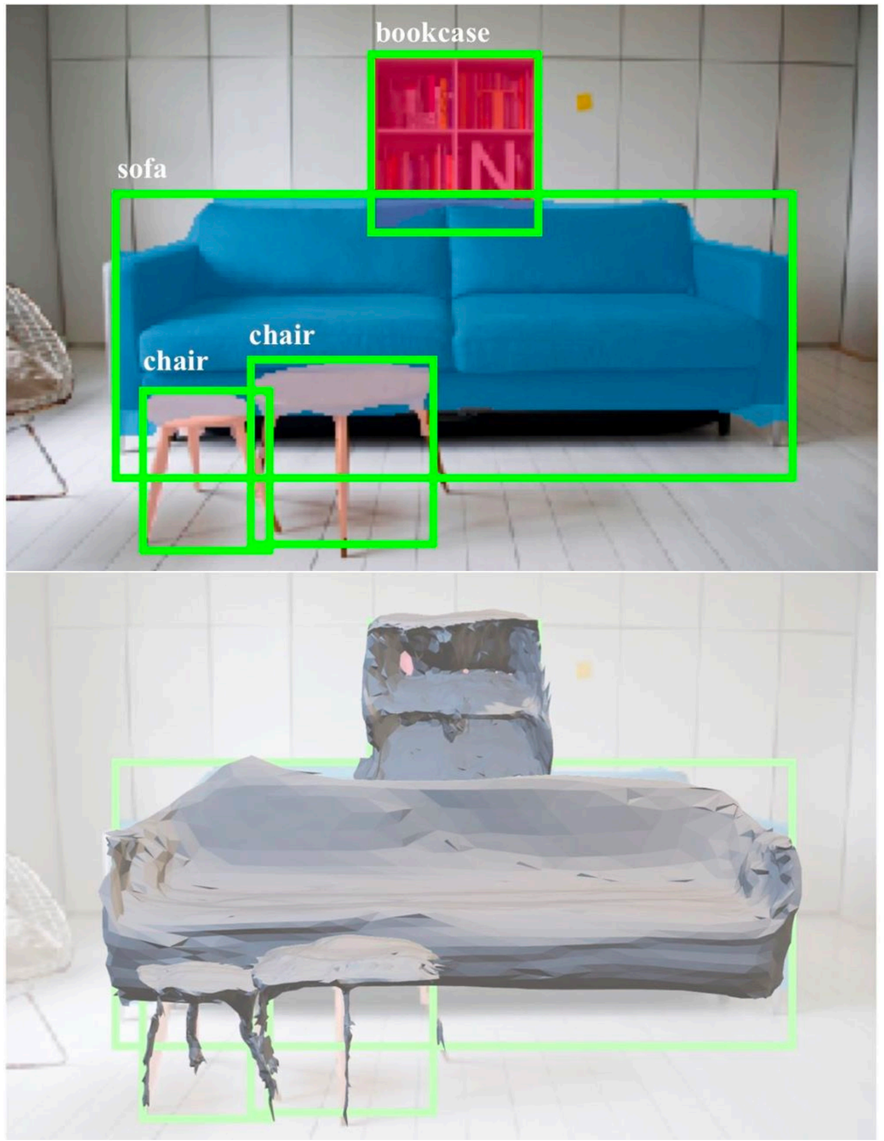
Mesh R-CNN: Task

Input: Single RGB image

Output:
A set of detected objects
For each object:

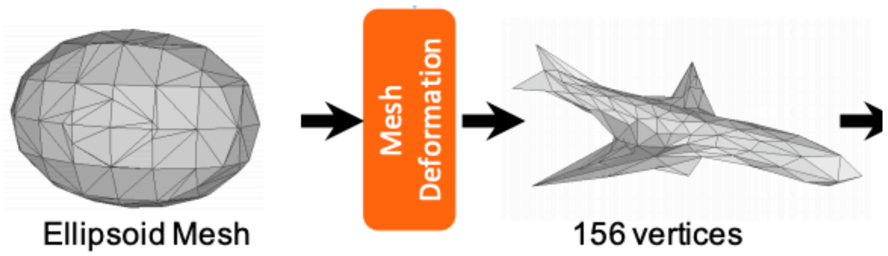
Mask R-CNN
- Bounding box
- Category label
- Instance segmentation

Mesh head
- 3D triangle mesh

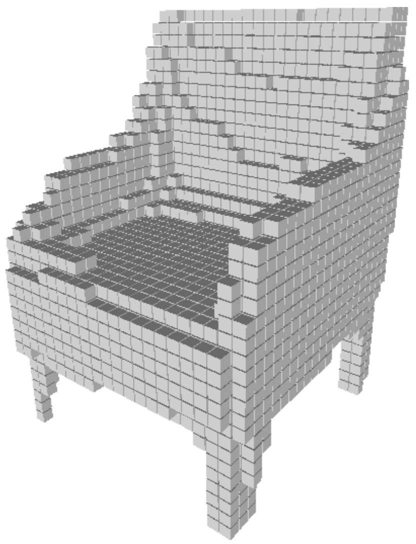


Mesh R-CNN: Hybrid 3D shape representation

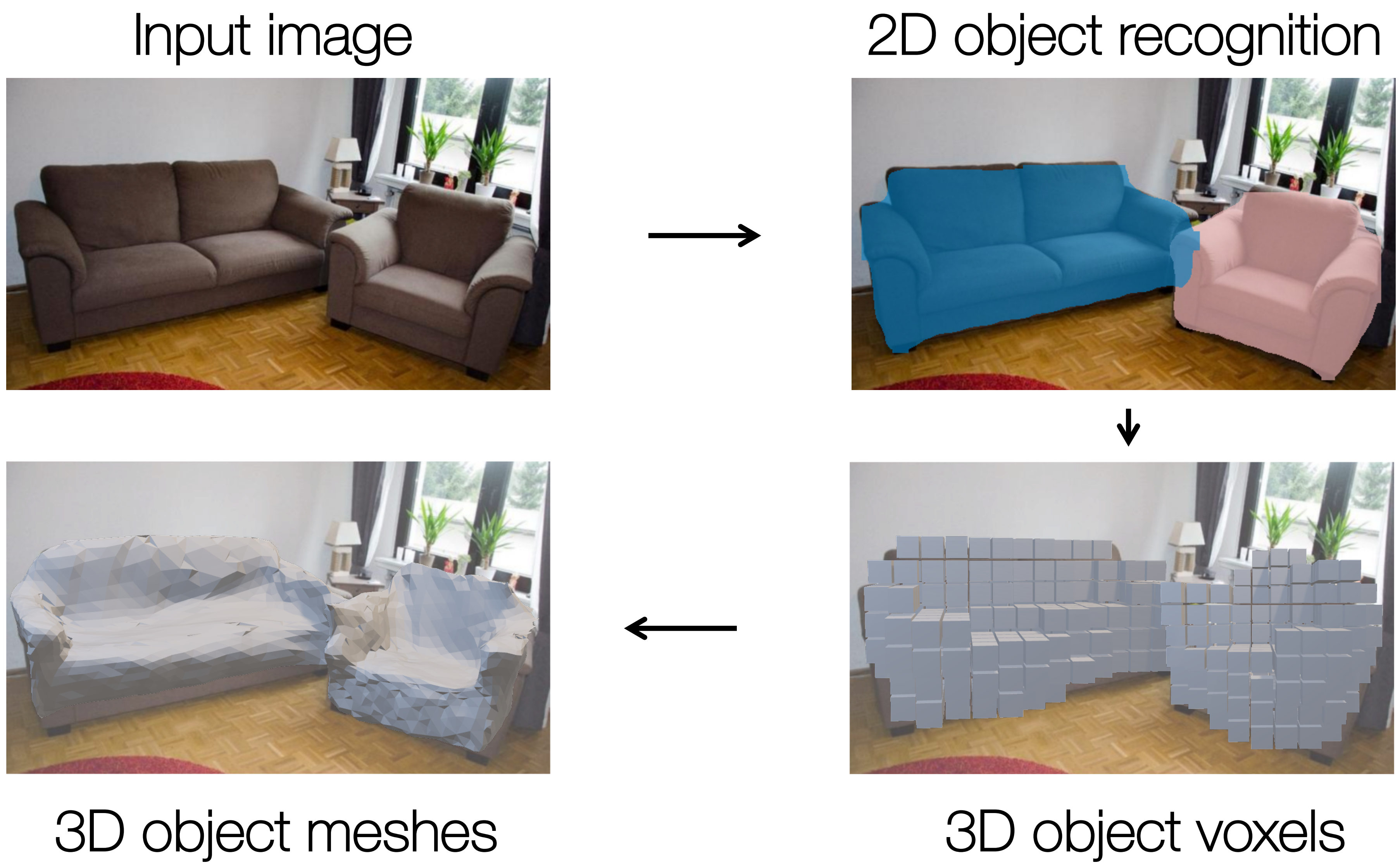
Mesh deformation gives good results, but the topology (verts, faces, genus, connected components) fixed by the initial mesh



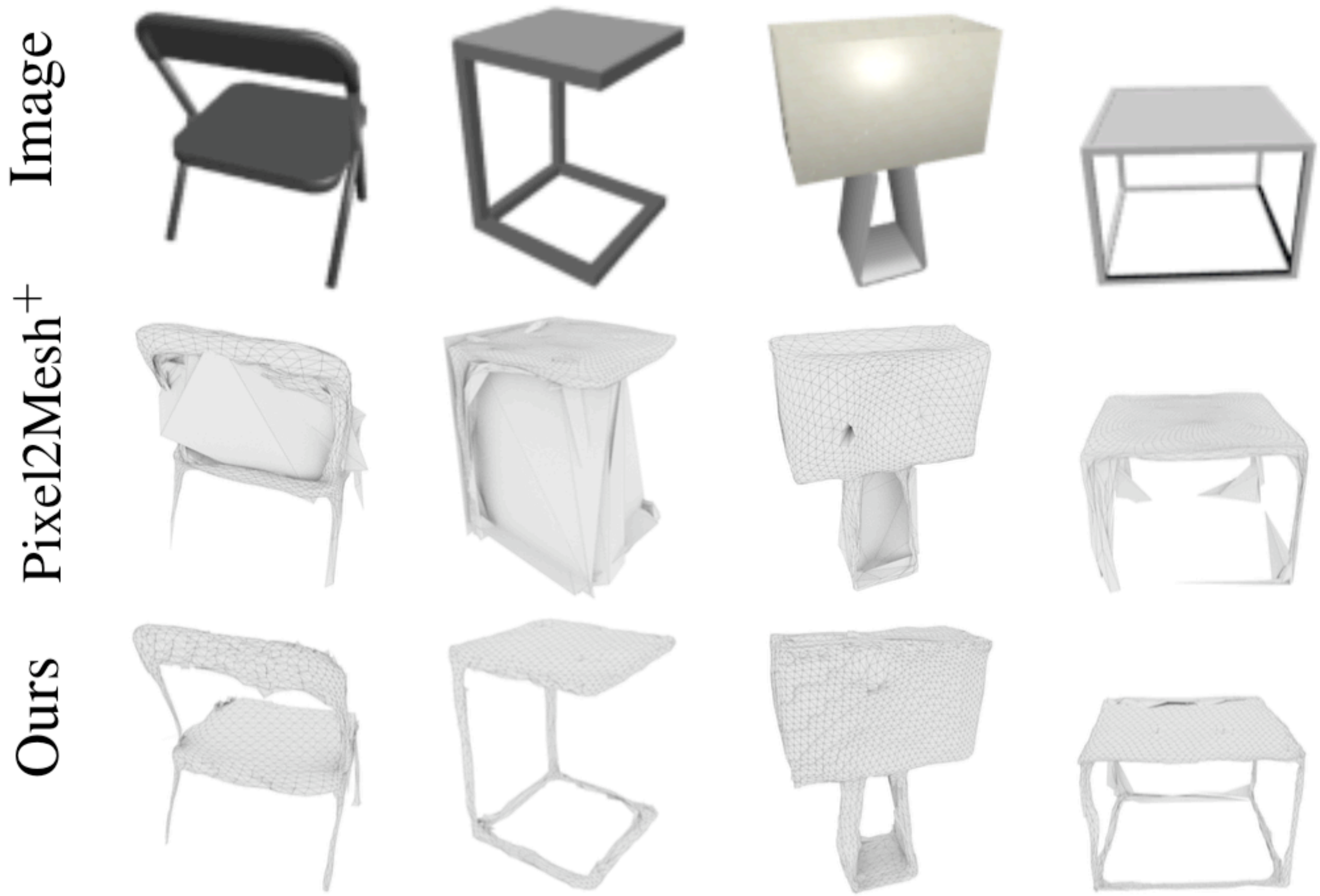
Our approach: Use voxel predictions to create initial mesh prediction!



Mesh R-CNN Pipeline

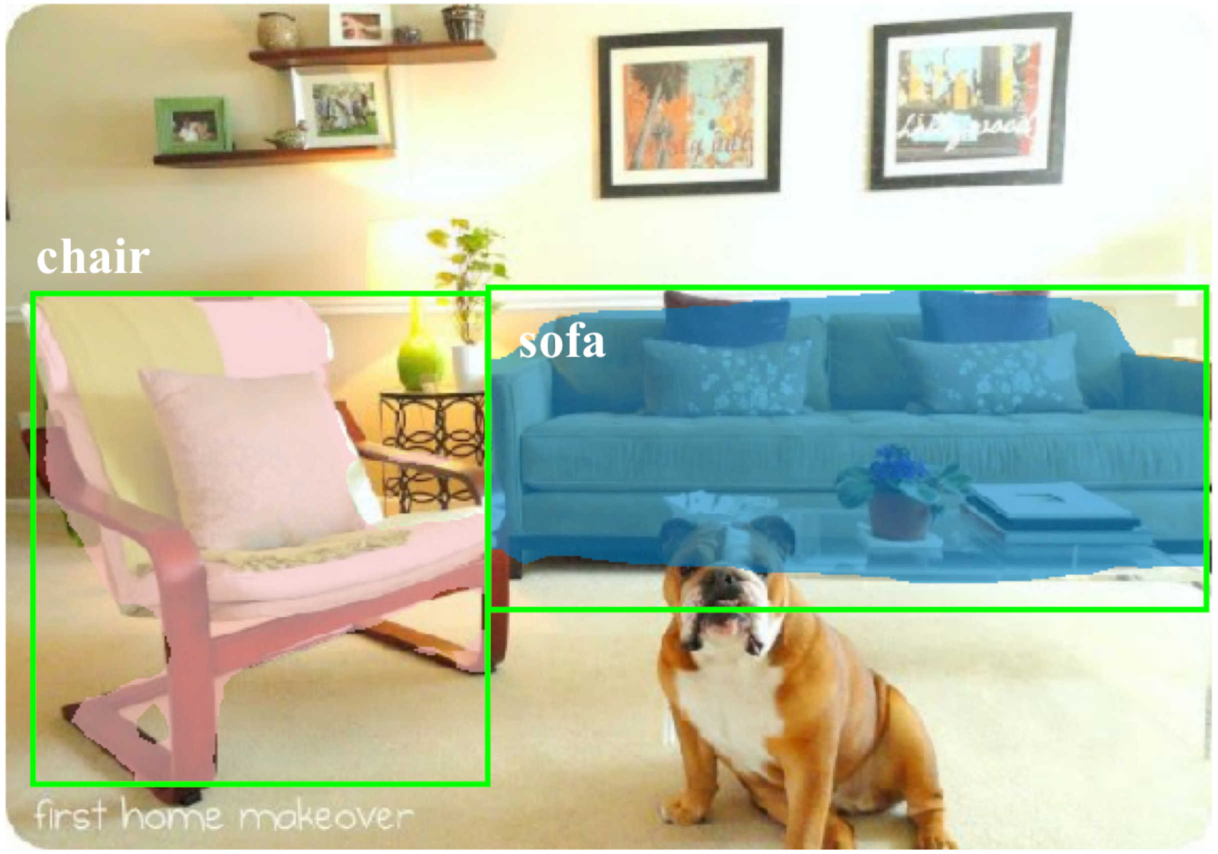


Mesh R-CNN: ShapeNet Results

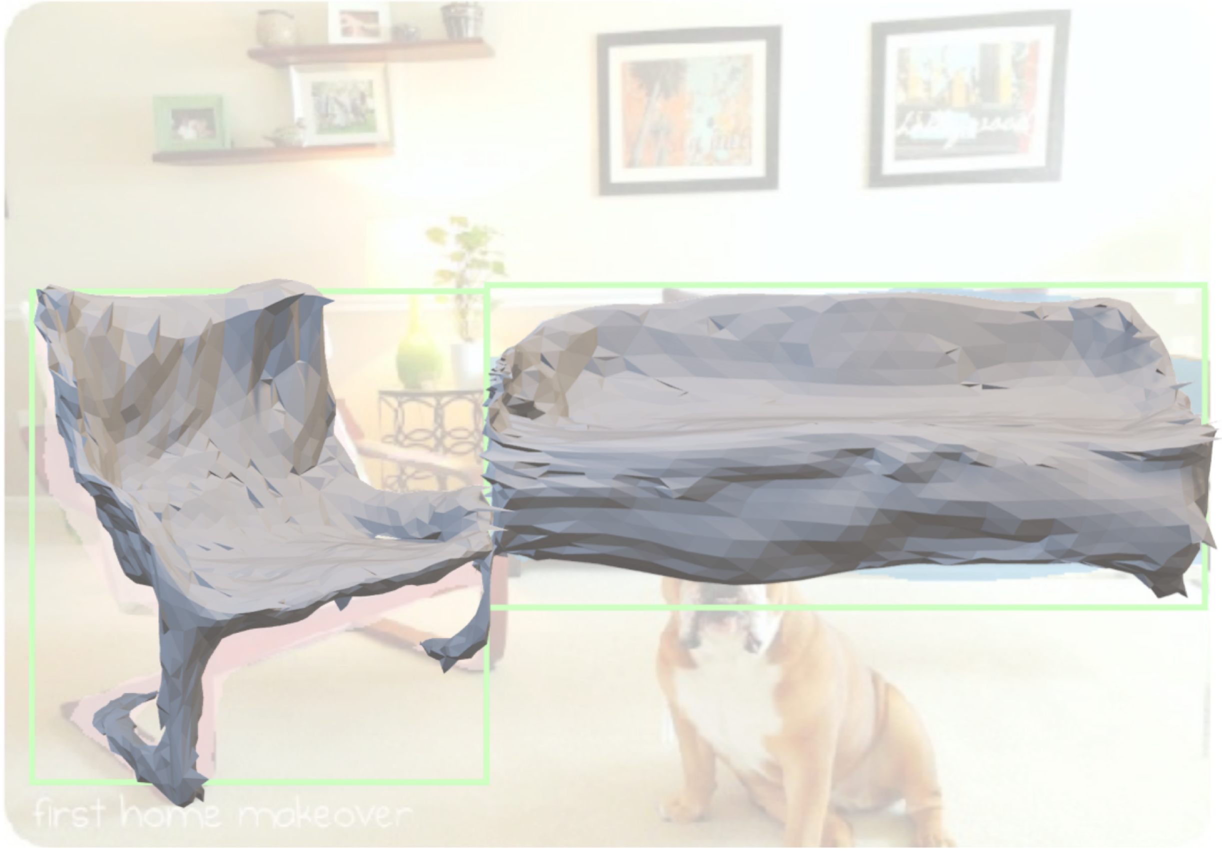


Mesh R-CNN: Pix3D Results

Amodal completion: predict occluded parts of objects



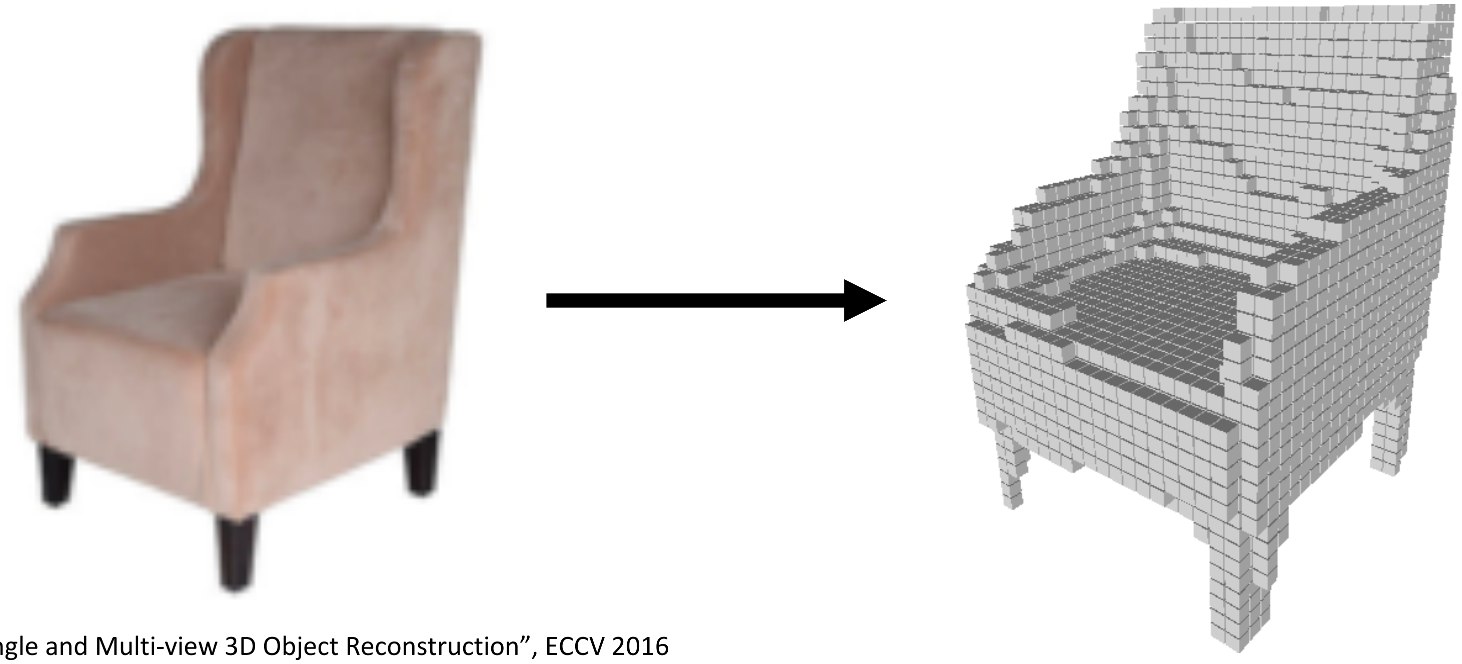
Box & Mask Predictions



Mesh Predictions

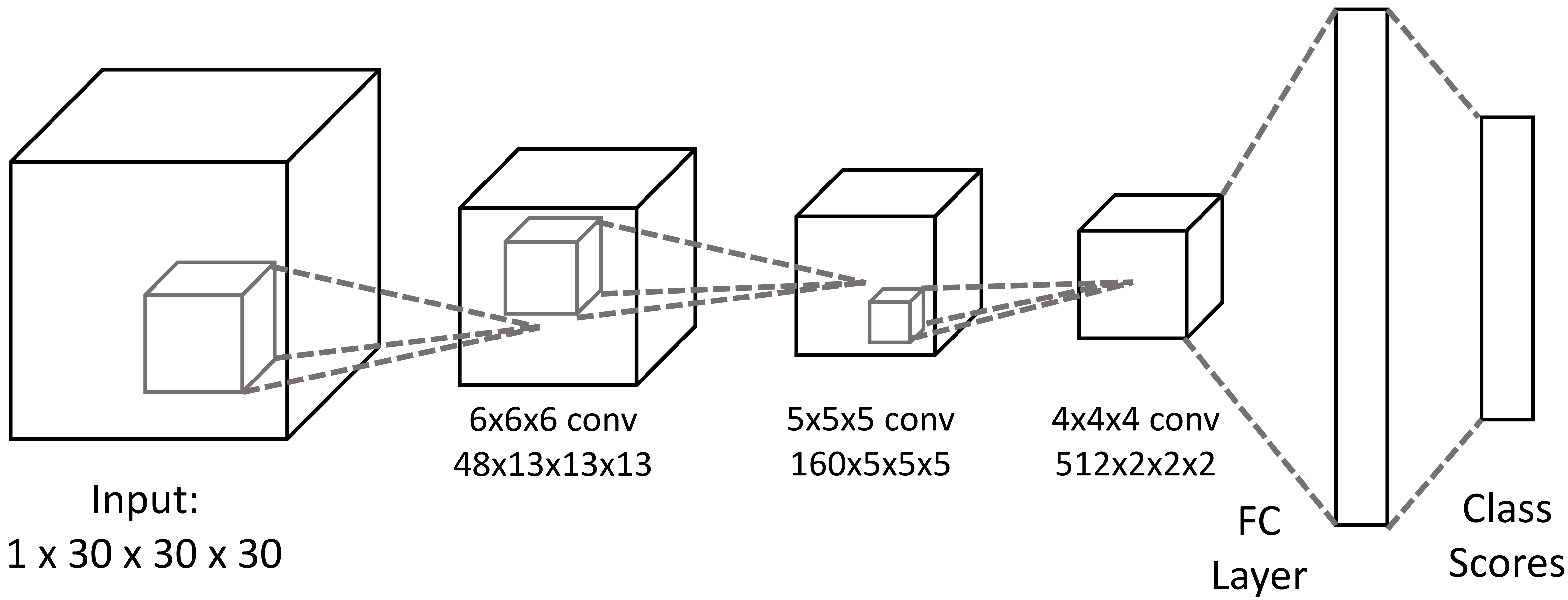
3D Shape Representations: Voxels

- Represent a shape with a V x V x V grid of occupancies
- Just like segmentation masks in Mask R-CNN, but in 3D!
- (+) Conceptually simple: just a 3D grid!
- (-) Need high spatial resolution to capture fine structures
- (-) Scaling to high resolutions is nontrivial!



Choy et al, "3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction", ECCV 2016

Processing Voxel Inputs: 3D Convolution

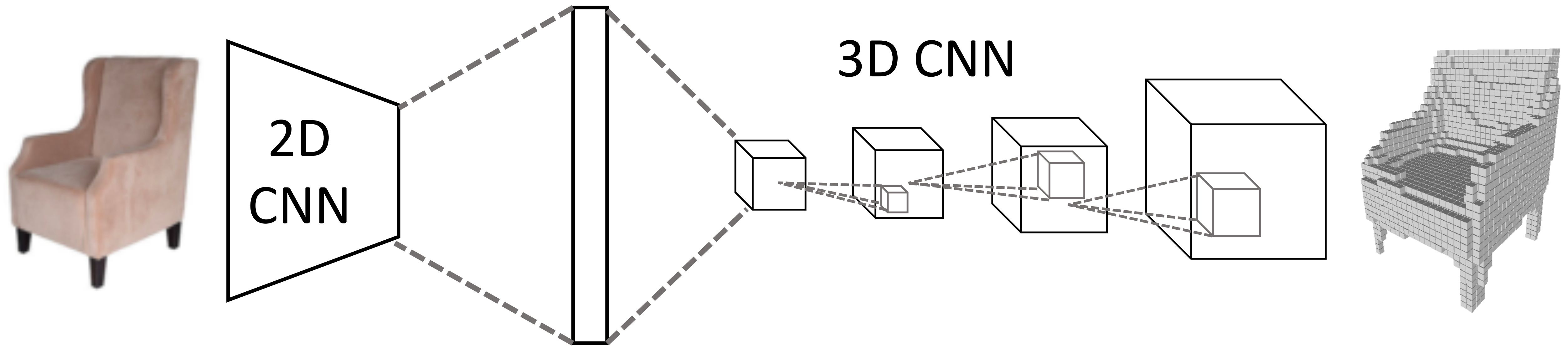


Train with classification loss

Wu et al, "3D ShapeNets: A Deep Representation for Volumetric Shapes", CVPR 2015

(for classification of a voxel grid)

Generating Voxel Shapes: 3D Convolution



Input image:
3 x 112 x 112

2D Features:
C x H x W

3D Features:
C' x D' x H' x W'

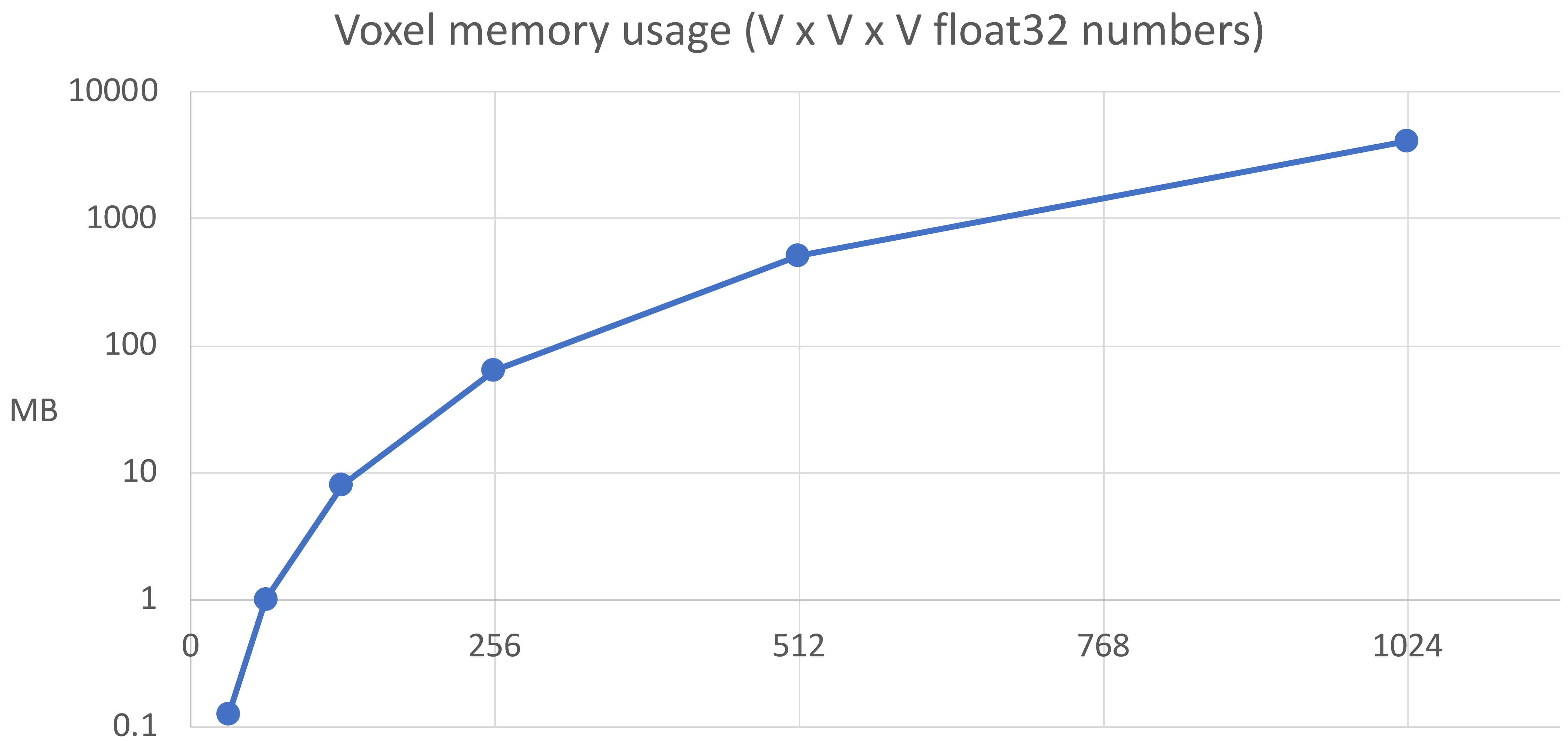
Voxels:
1 x V x V x V

Train with per-voxel cross-entropy loss

Choy et al, "3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction", ECCV 2016

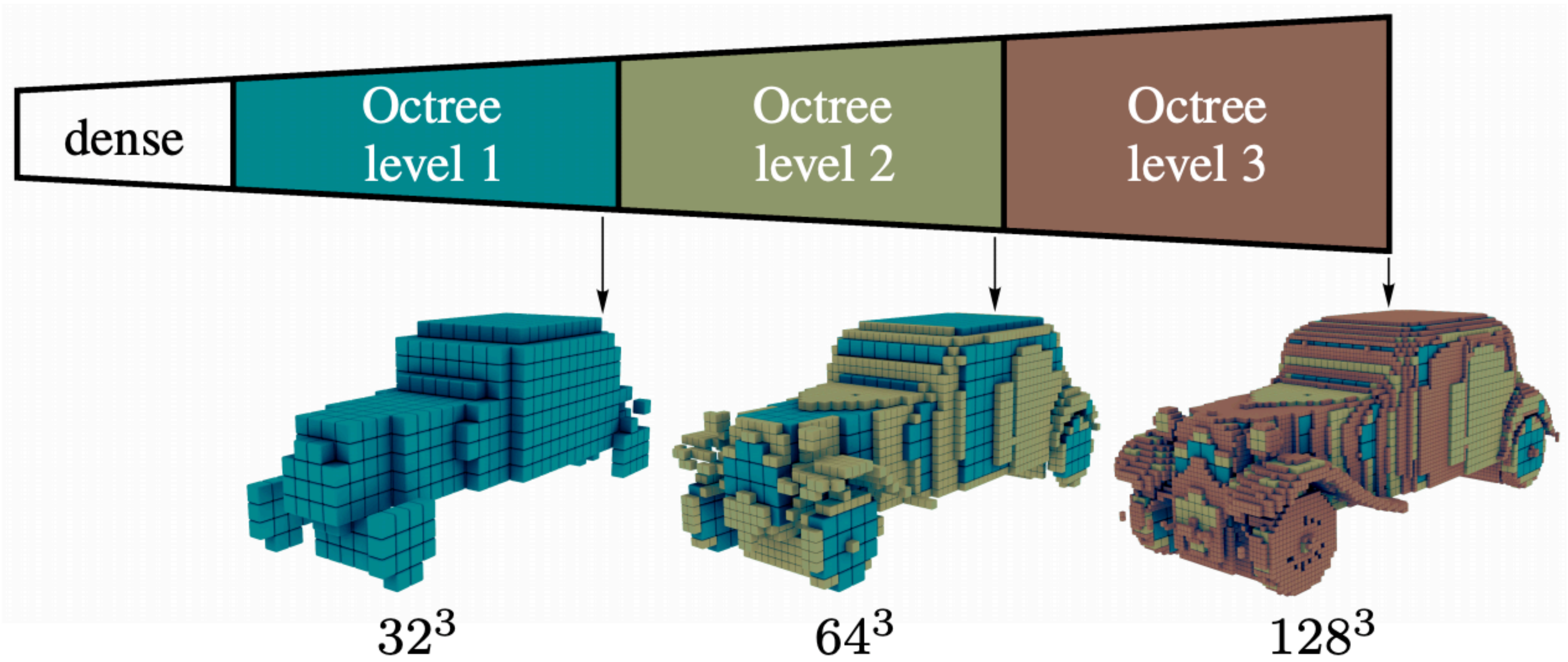
Voxel Problems: Memory Usage

Storing 1024^3 voxel grid takes 4GB of memory!



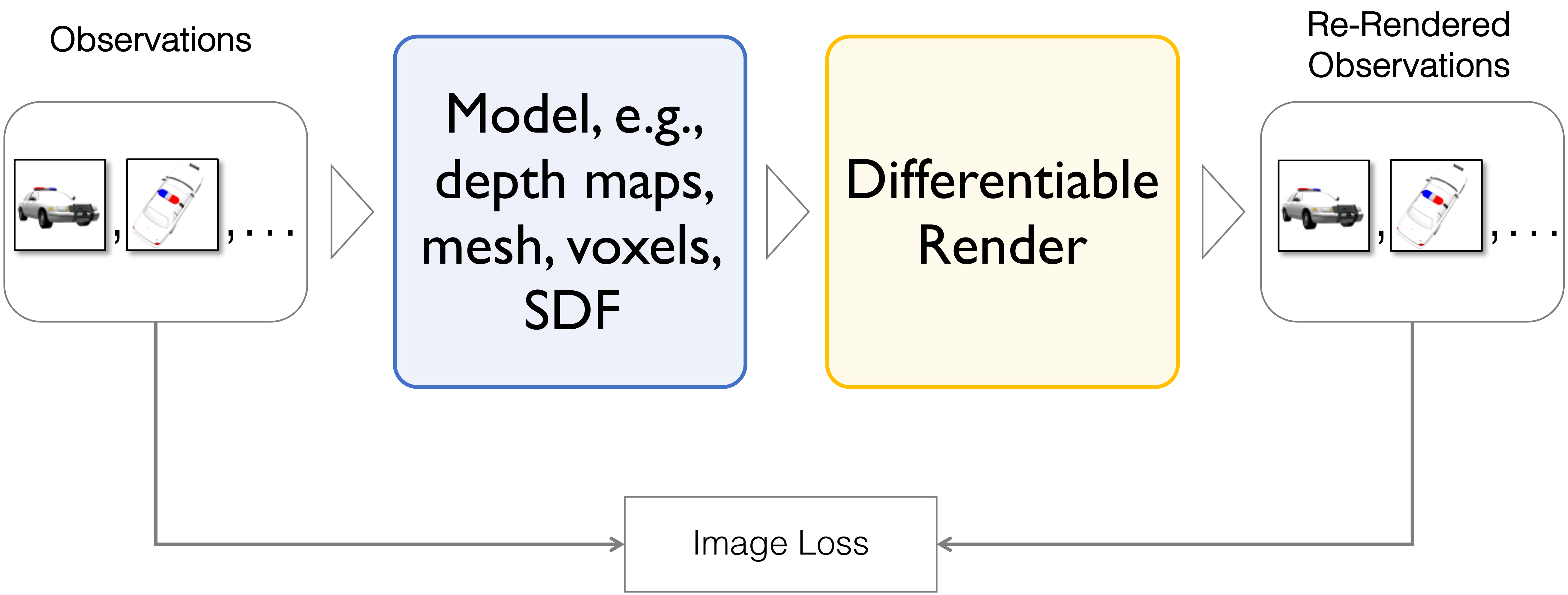
Scaling Voxels: Oct-Trees

Use voxel grids with heterogenous resolution!

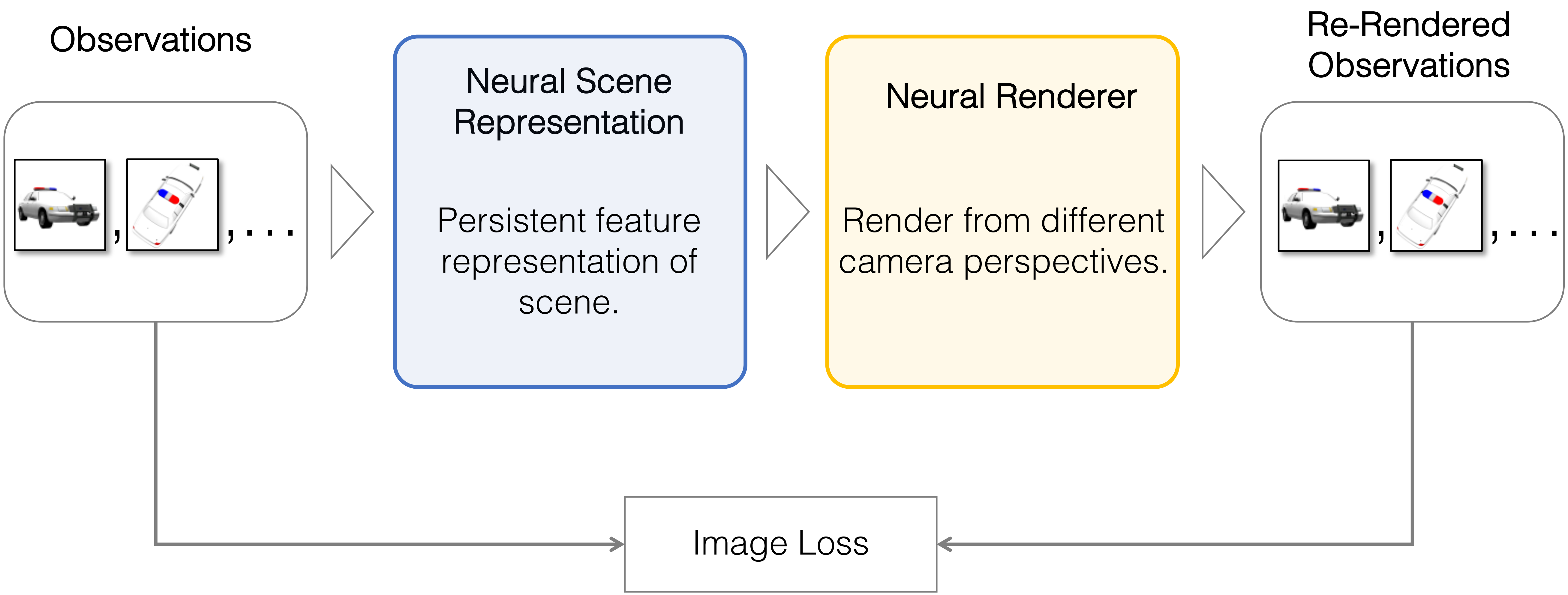


Tatarchenko et al, "Octree Generating Networks: Efficient Convolutional Architectures for High-resolution 3D Outputs", ICCV 2017

Self-supervised Scene Representation Learning

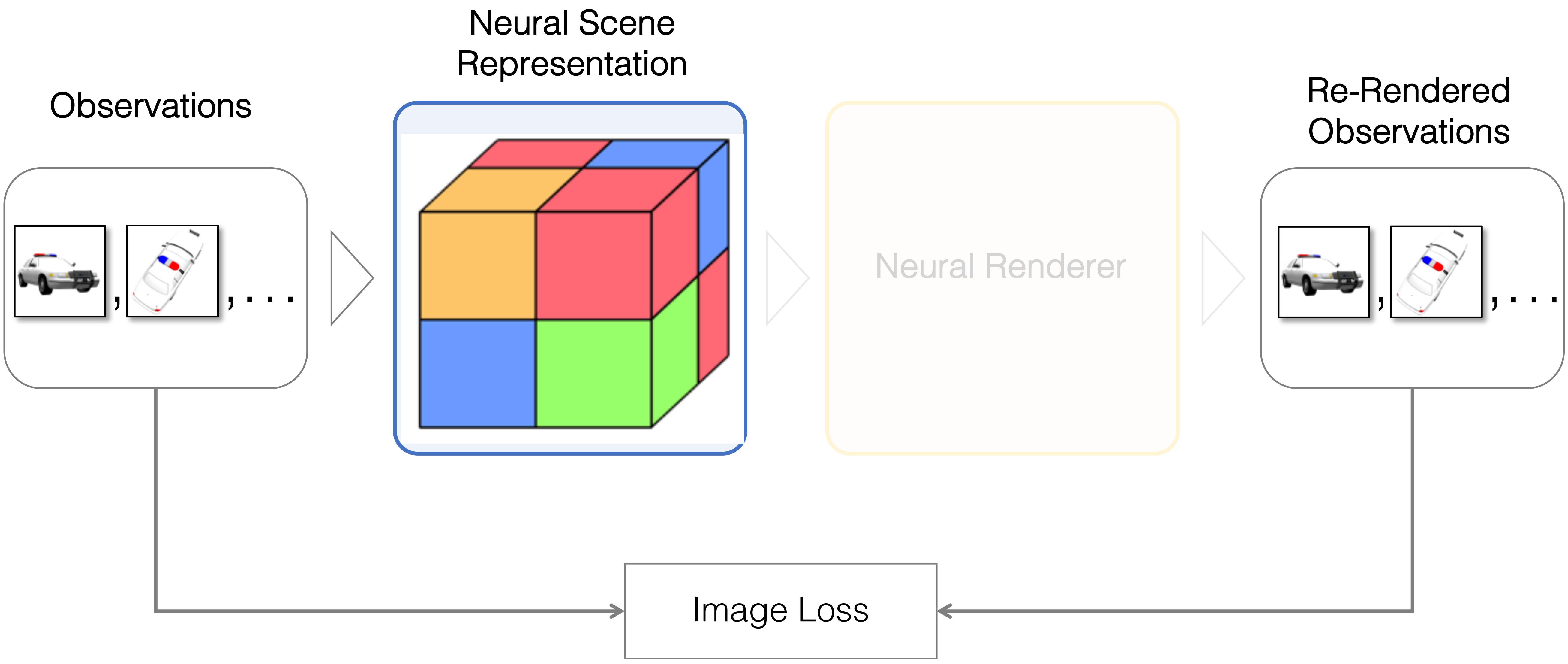


Self-supervised Scene Representation Learning



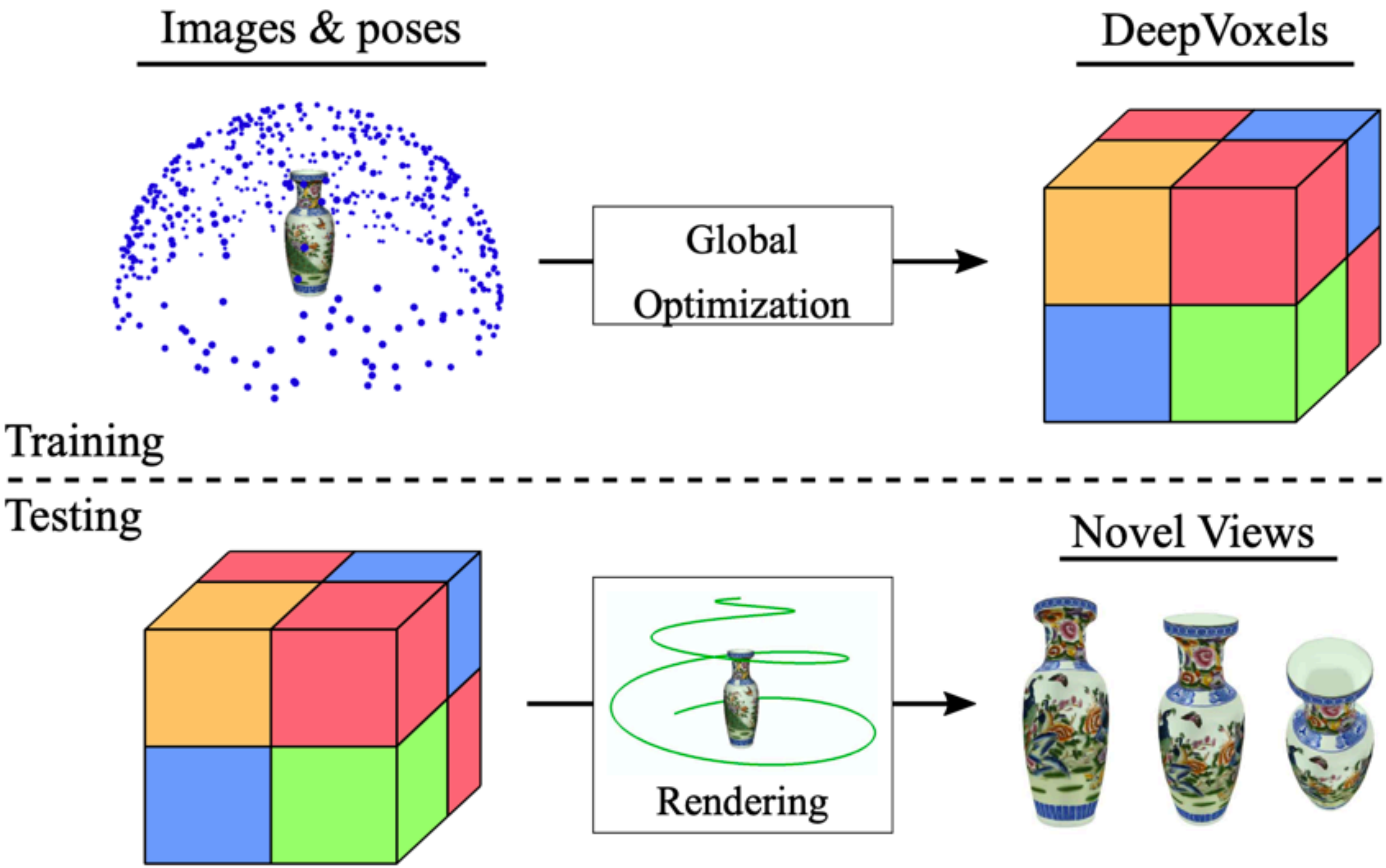
DeepVoxels

Embedding vector per voxel

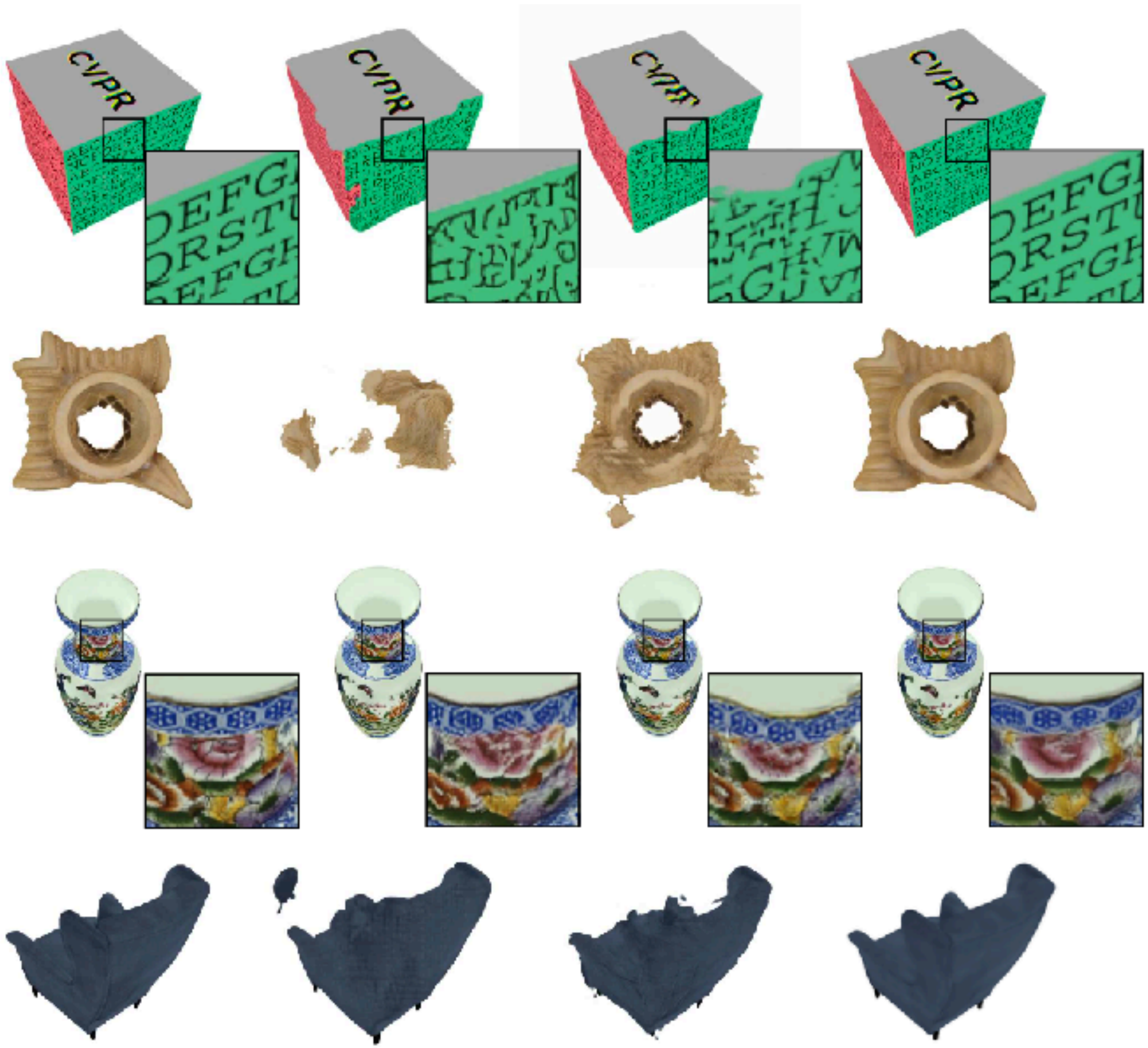


Scene represented as an embedding vector per 3D point

DeepVoxels



[Sitzmann et al. 2019]



Ground Truth

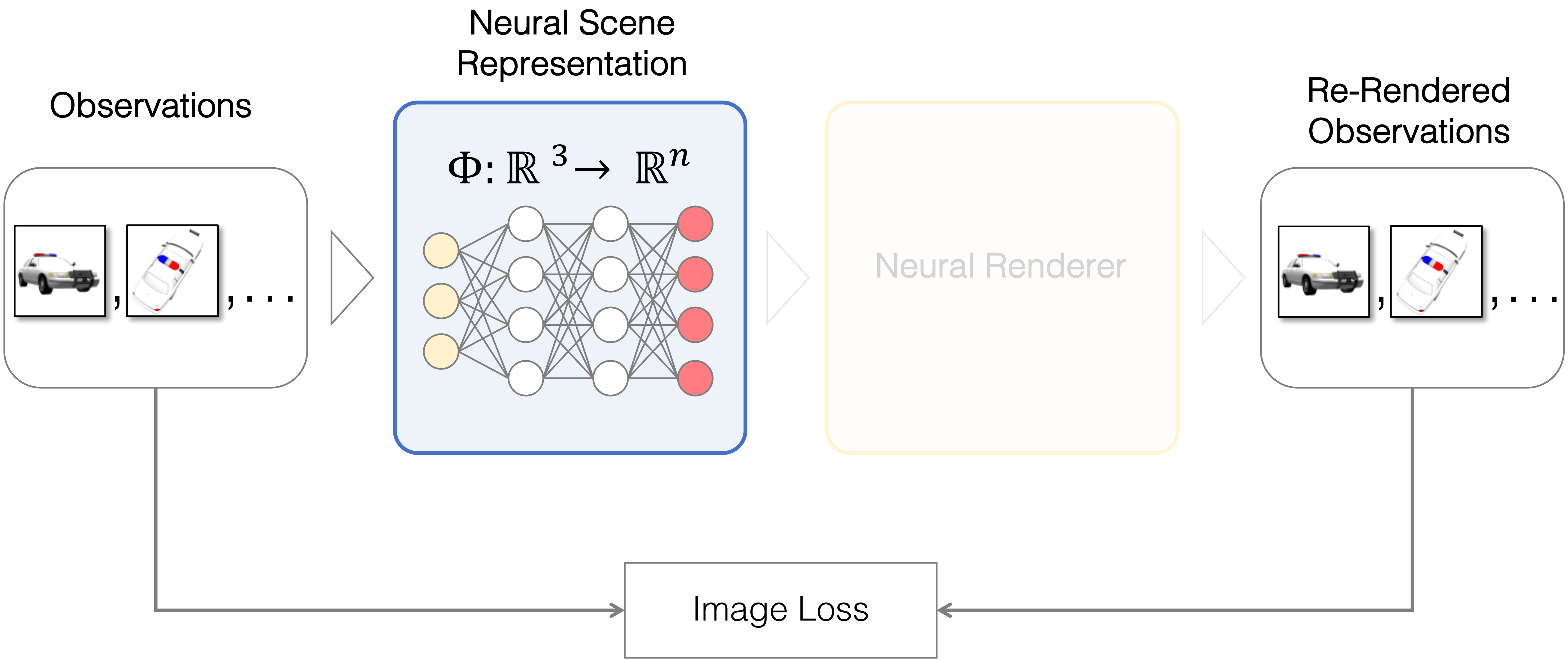
Worral et al

pix2pix

DeepVoxels

Scene Representation Networks

Fully connected network ϕ=f(X)

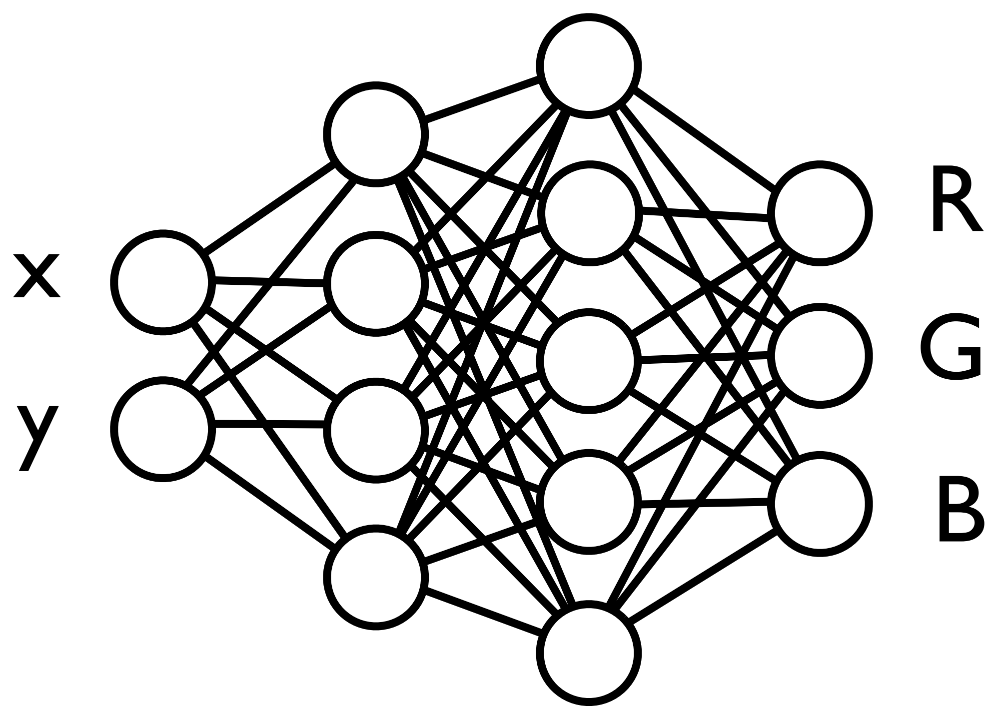


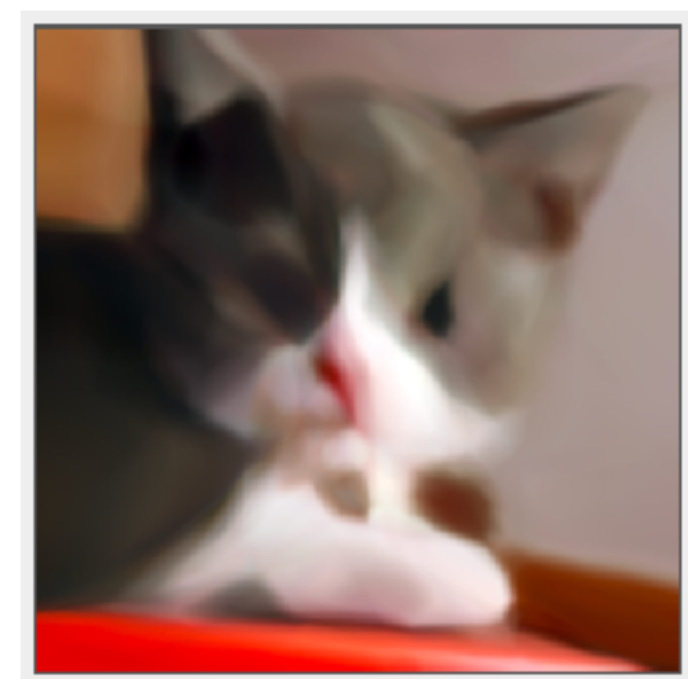
Scene represented as an embedding vector per 3D point

Image Regression

- Networks that operate on coordinates to generate image representations are sometimes called "Compositional Pattern Producing Networks" (CPPNs)

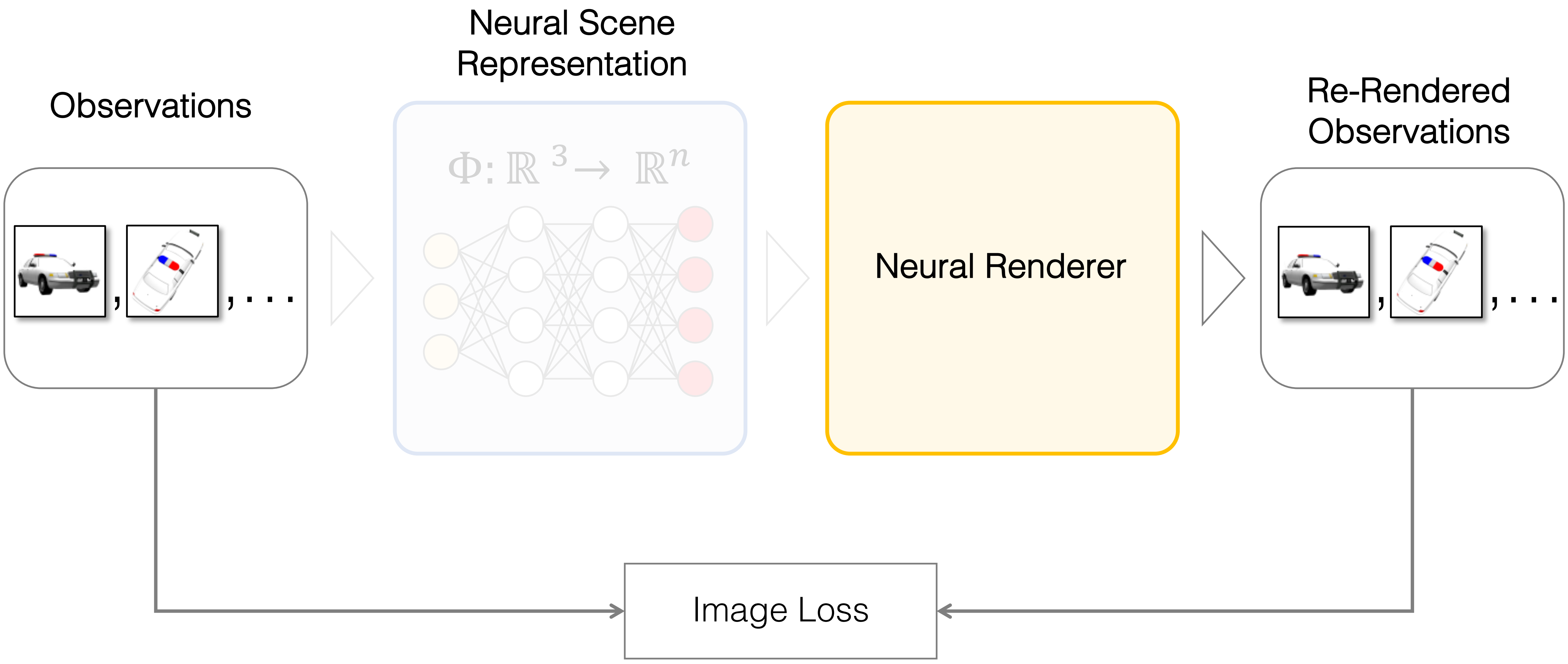
$$(R, G, B) = \phi(x, y)$$



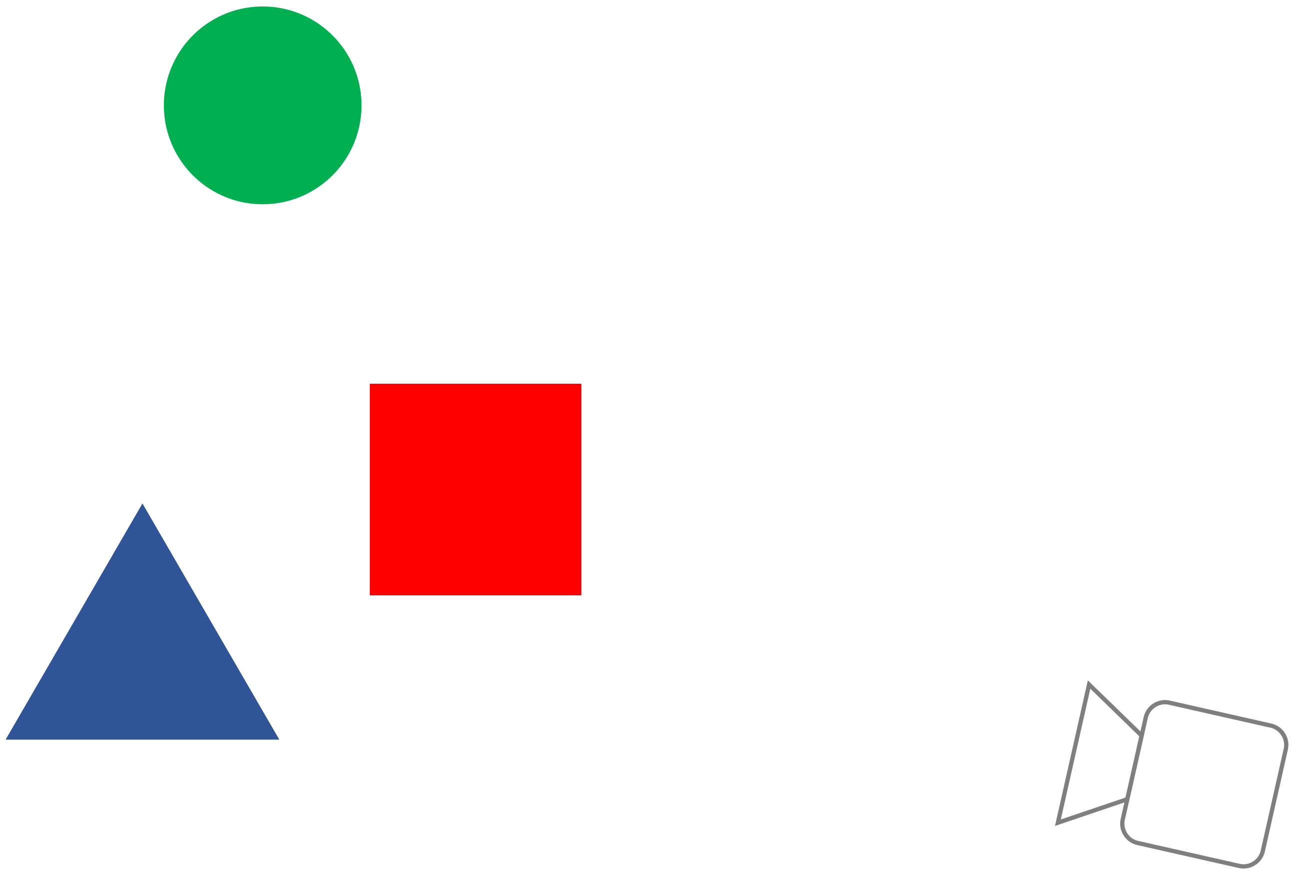


[A. Karpathy ConvNetJS Image Regression demo]

Scene Representation Networks

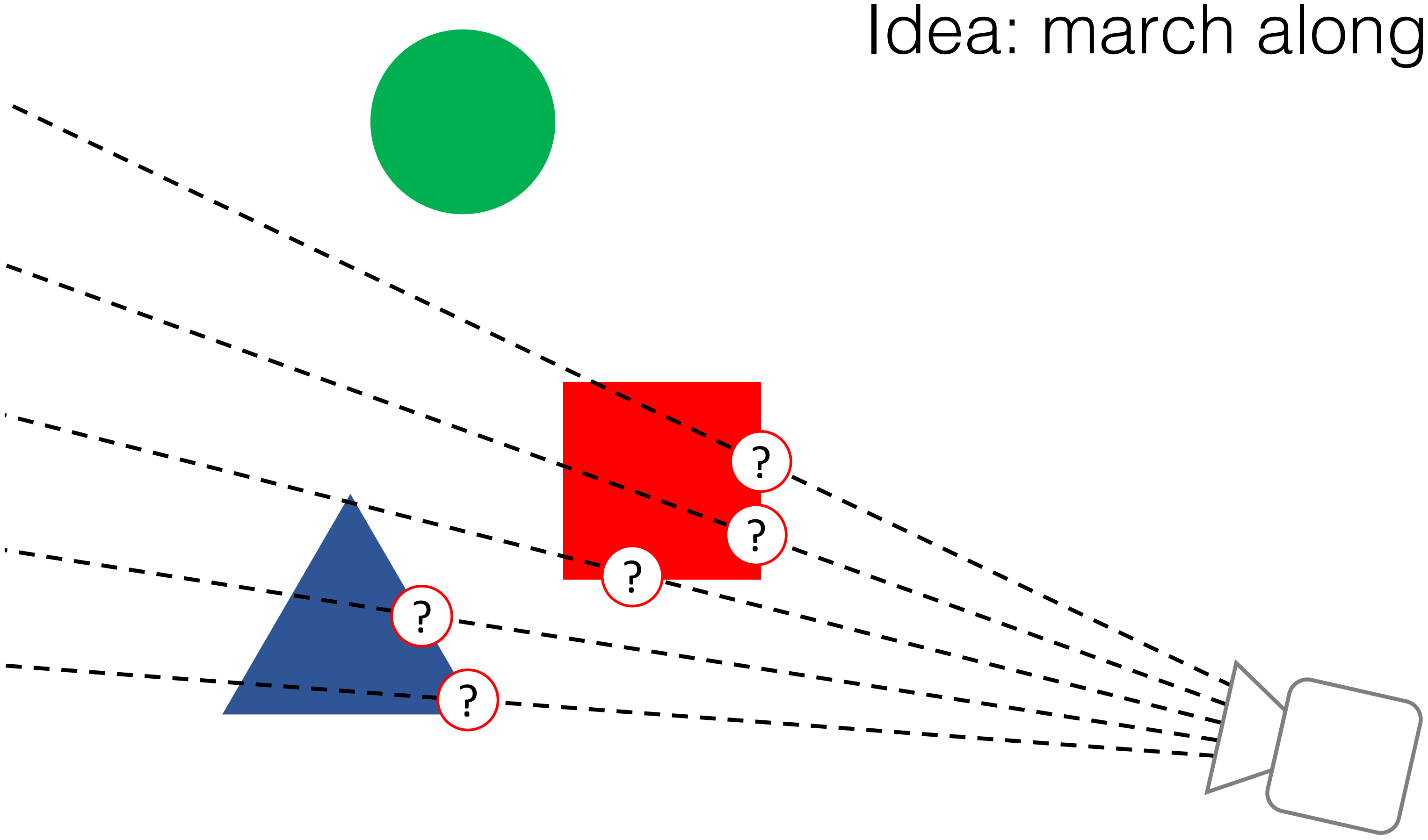


Neural Renderer.

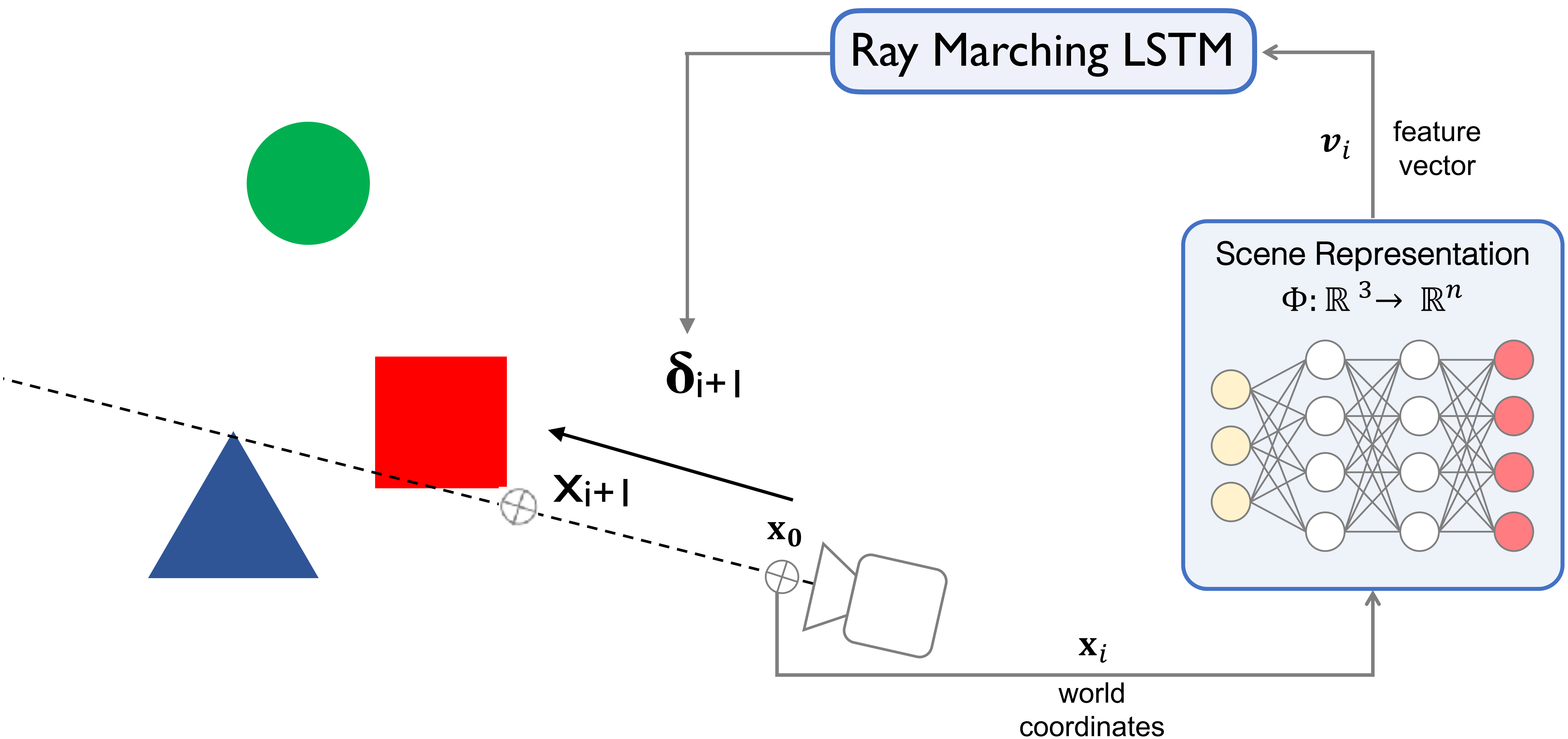


Neural Renderer Step 1: Intersection Testing.

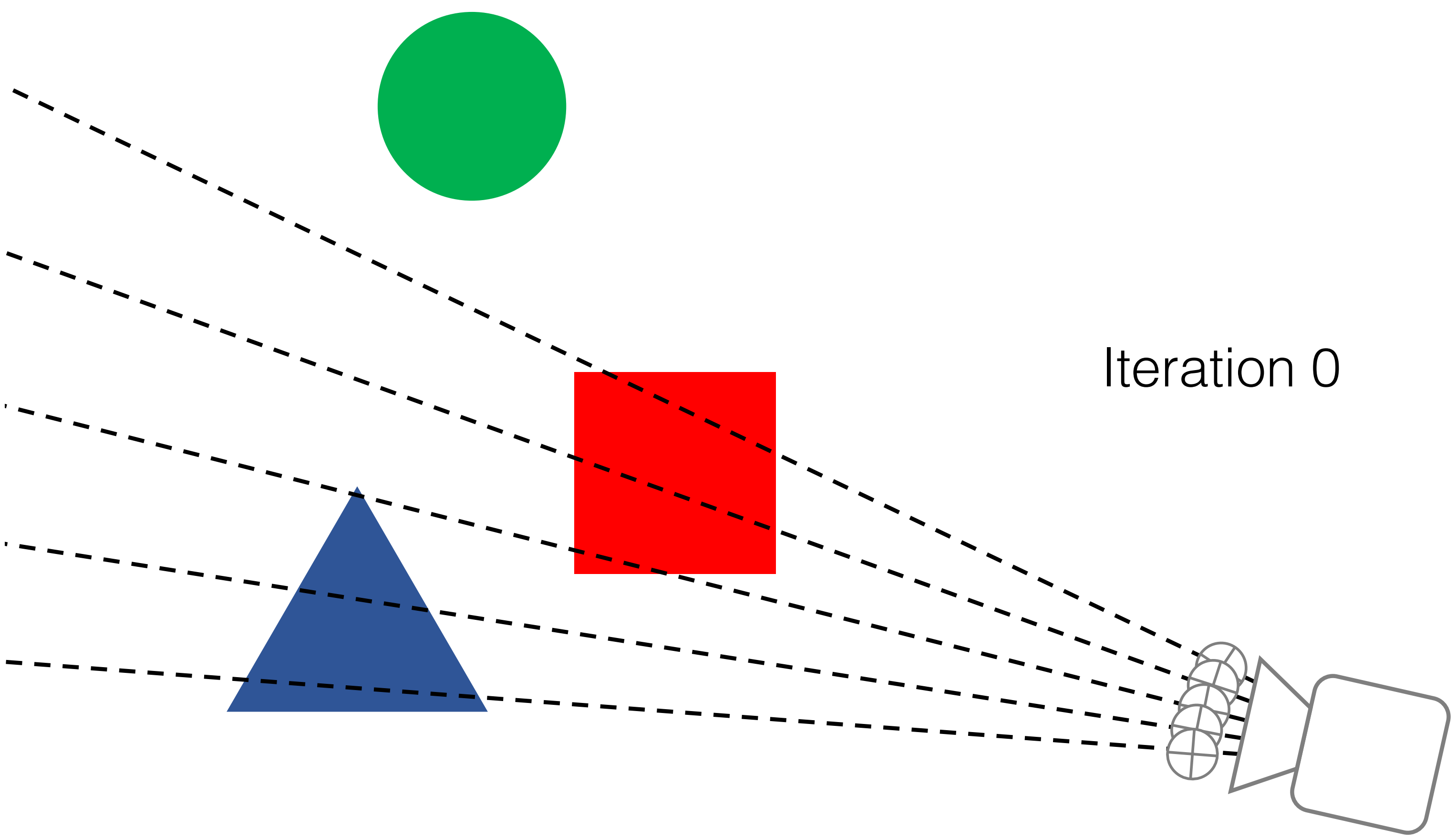
Idea: march along ray until arrived at surface.



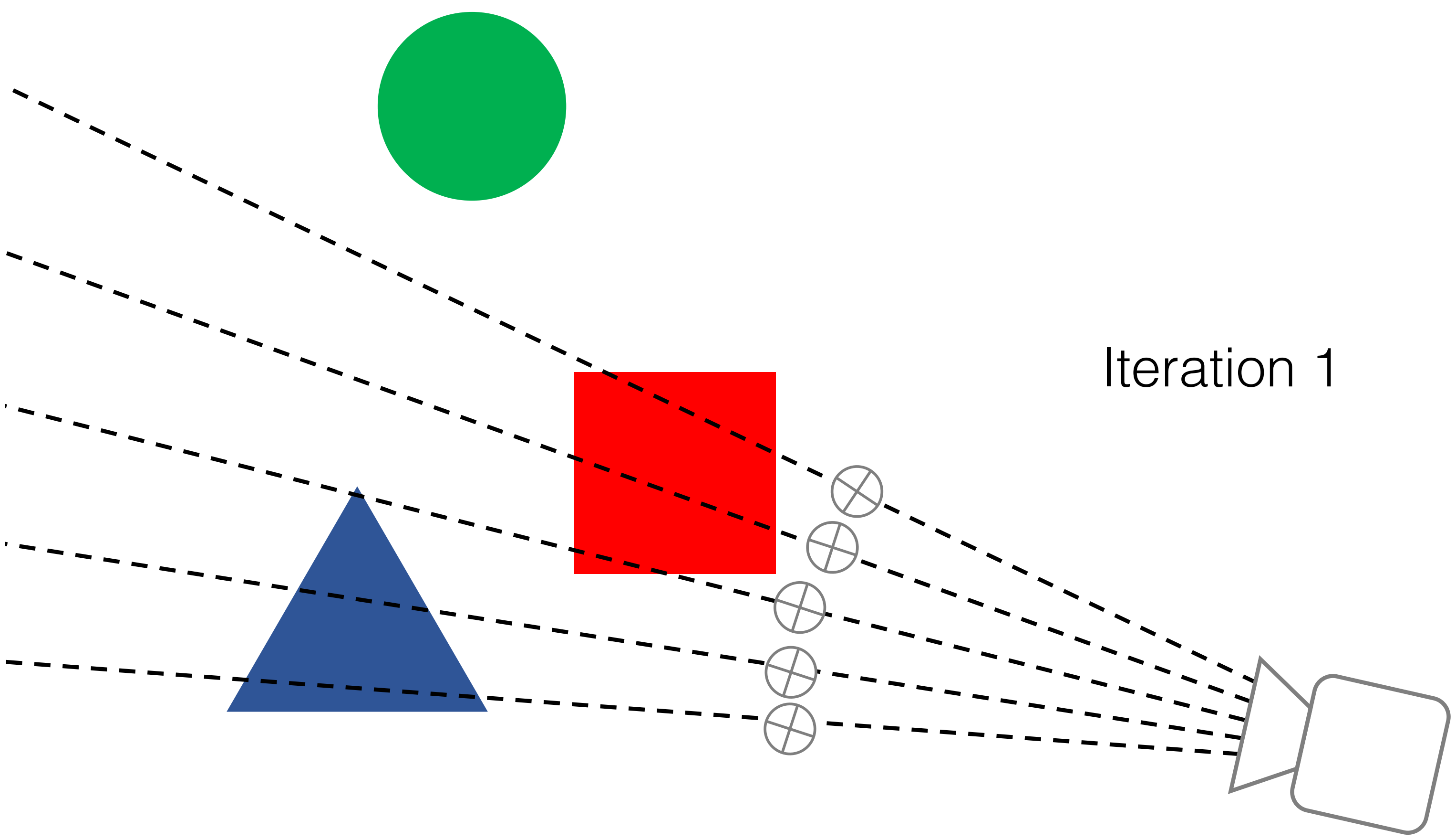
Neural Renderer Step 1: Intersection Testing.



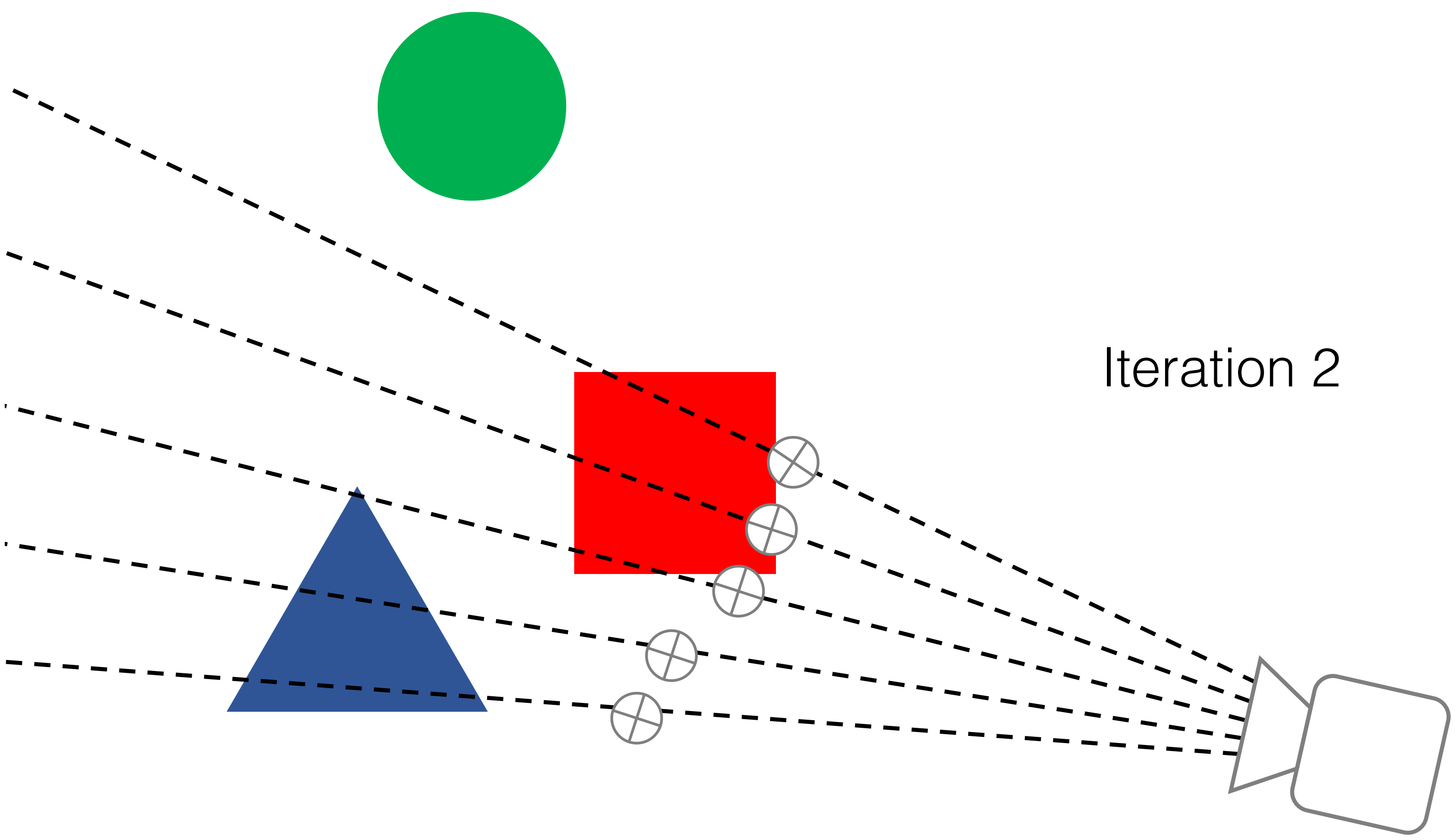
Neural Renderer Step 1: Intersection Testing.



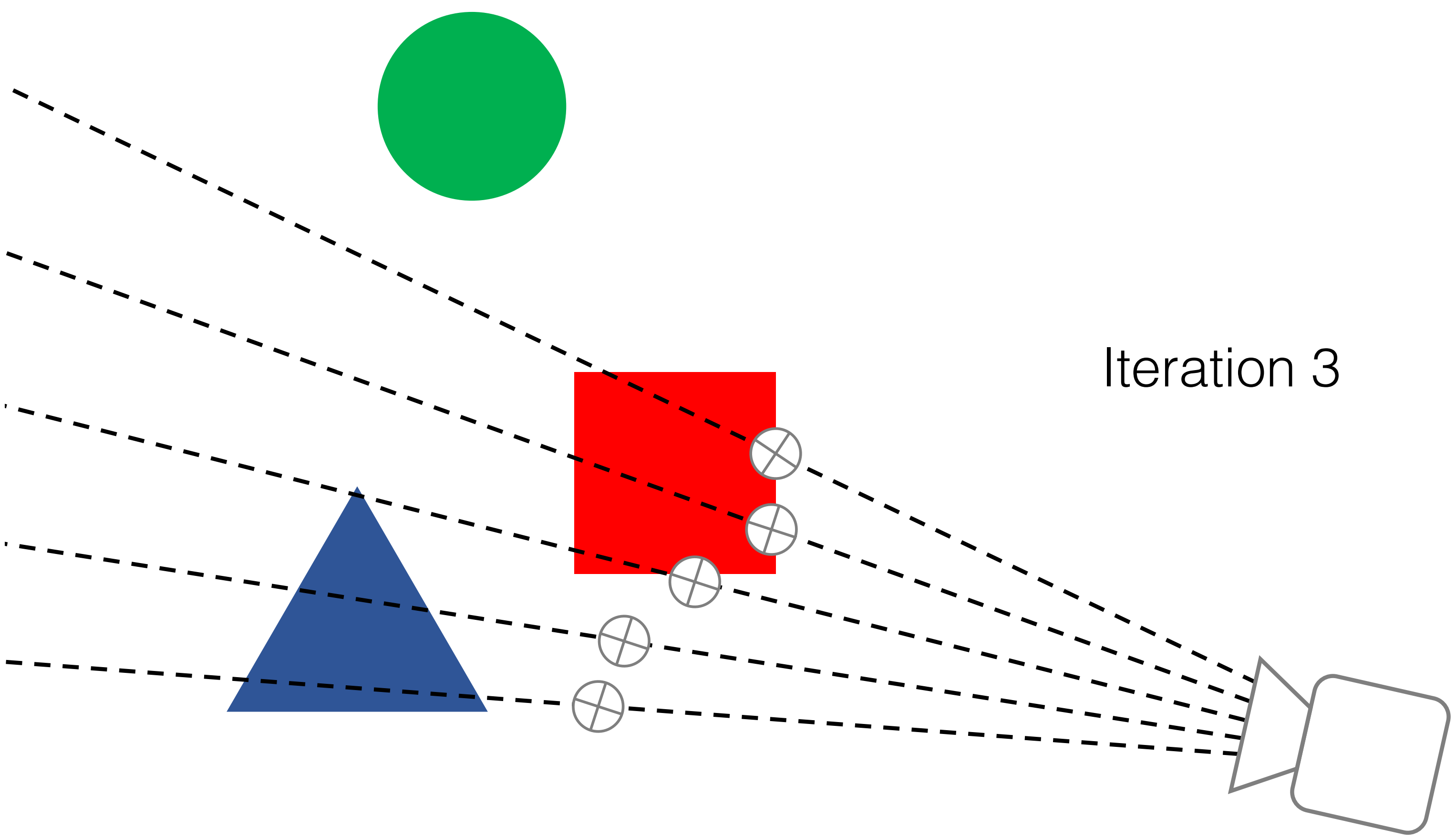
Neural Renderer Step 1: Intersection Testing.



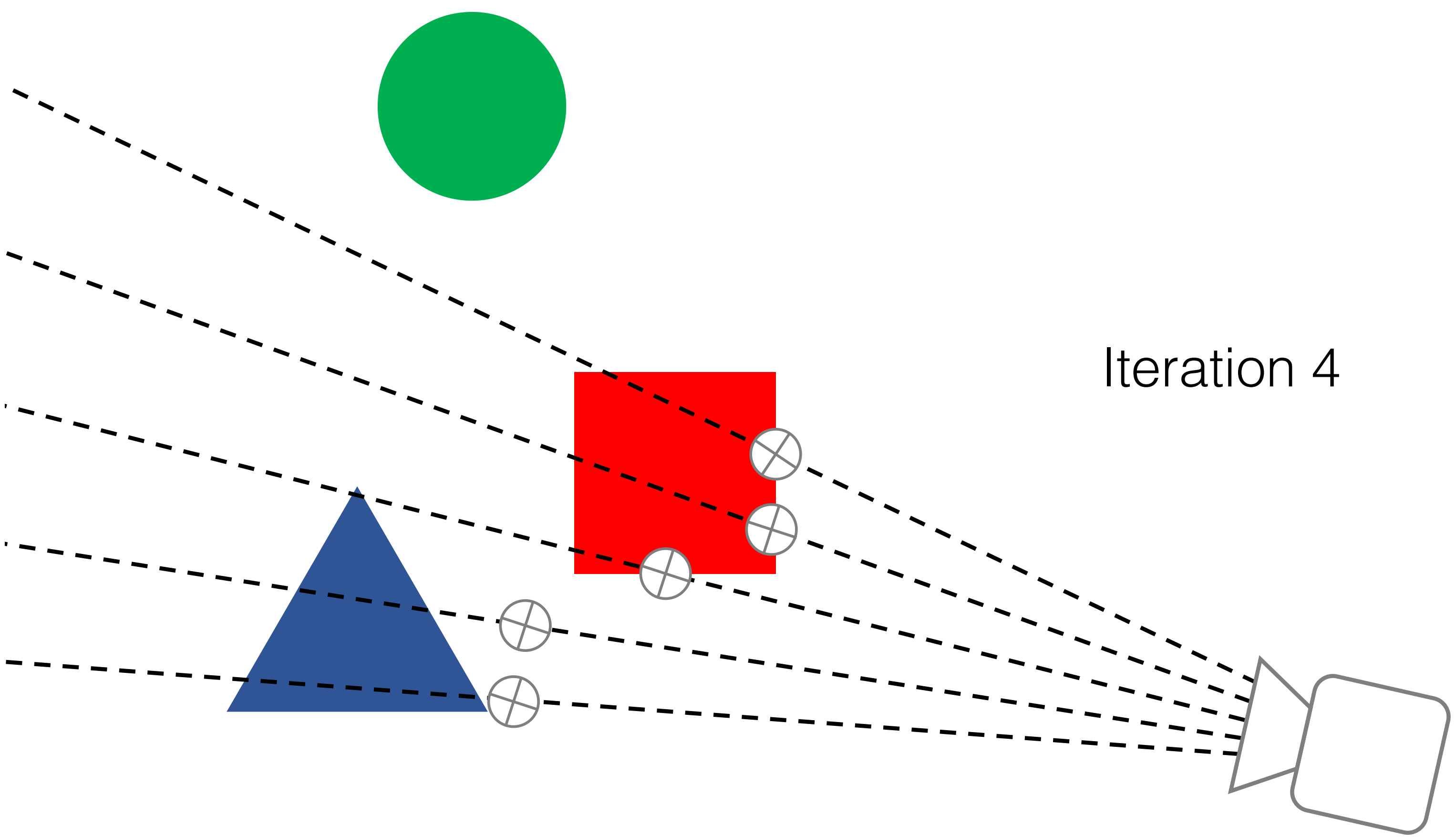
Neural Renderer Step 1: Intersection Testing.



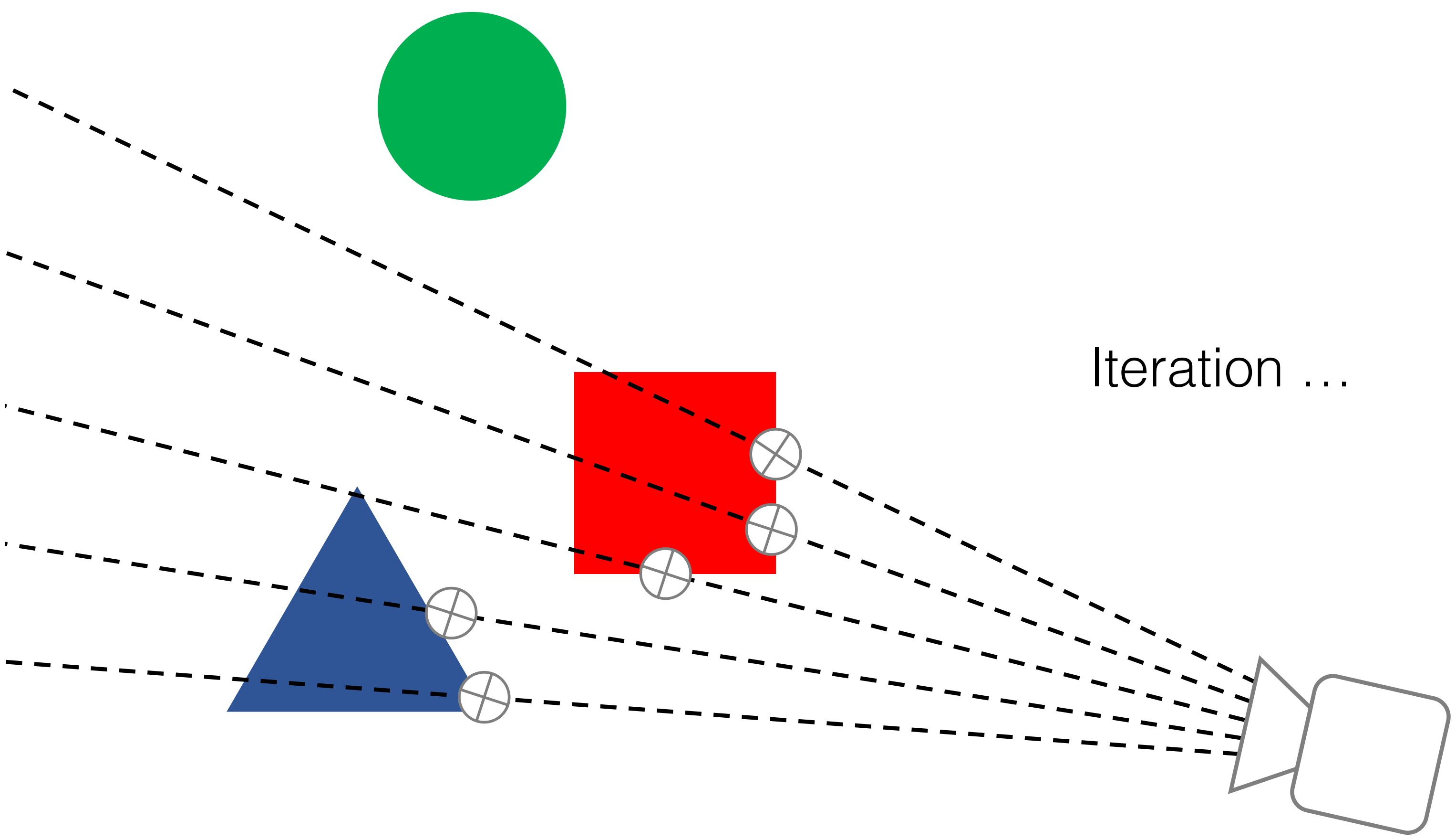
Neural Renderer Step 1: Intersection Testing.



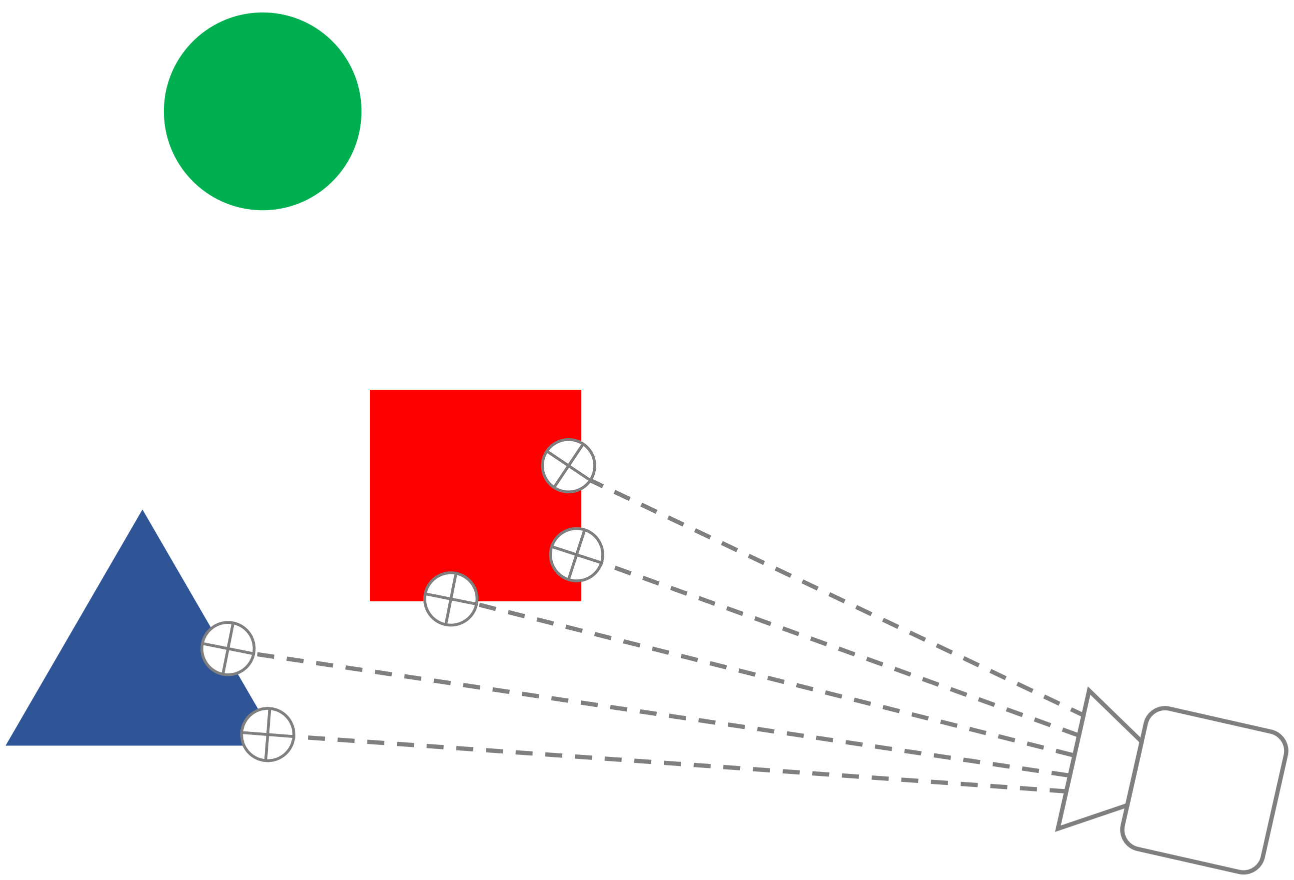
Neural Renderer Step 2: Color Generation



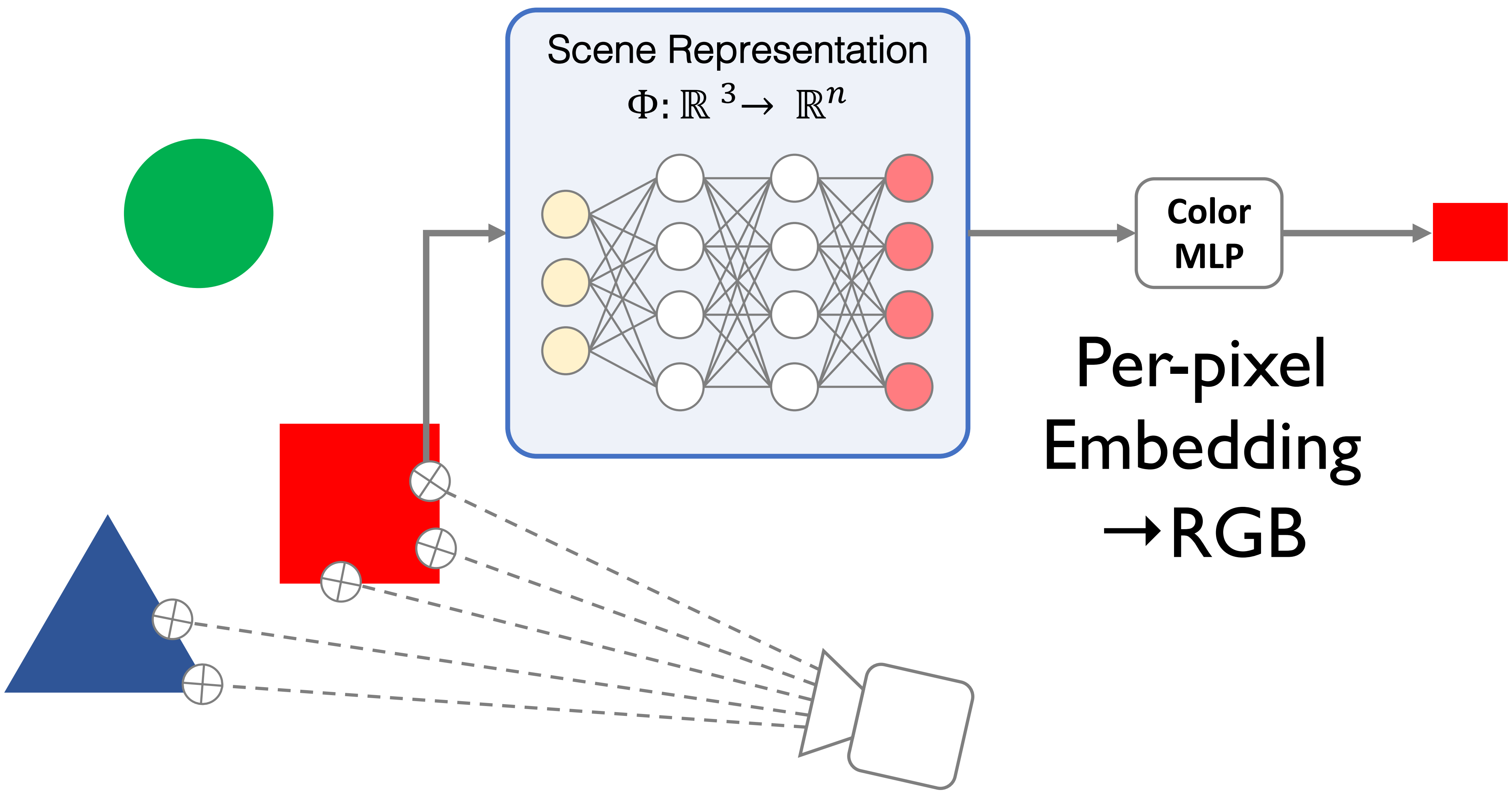
Neural Renderer Step 1: Intersection Testing.



Neural Renderer Step 1: Intersection Testing.

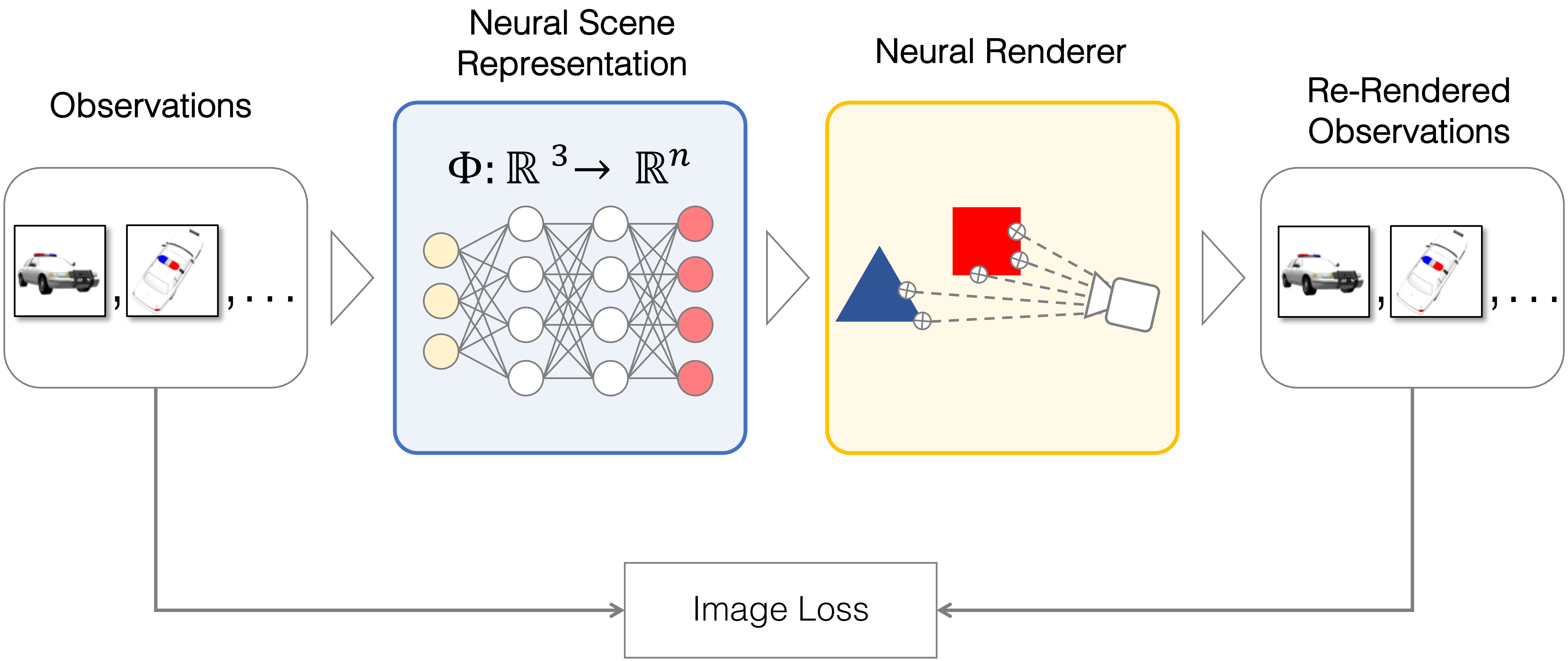


Neural Renderer Step 2: Color Generation



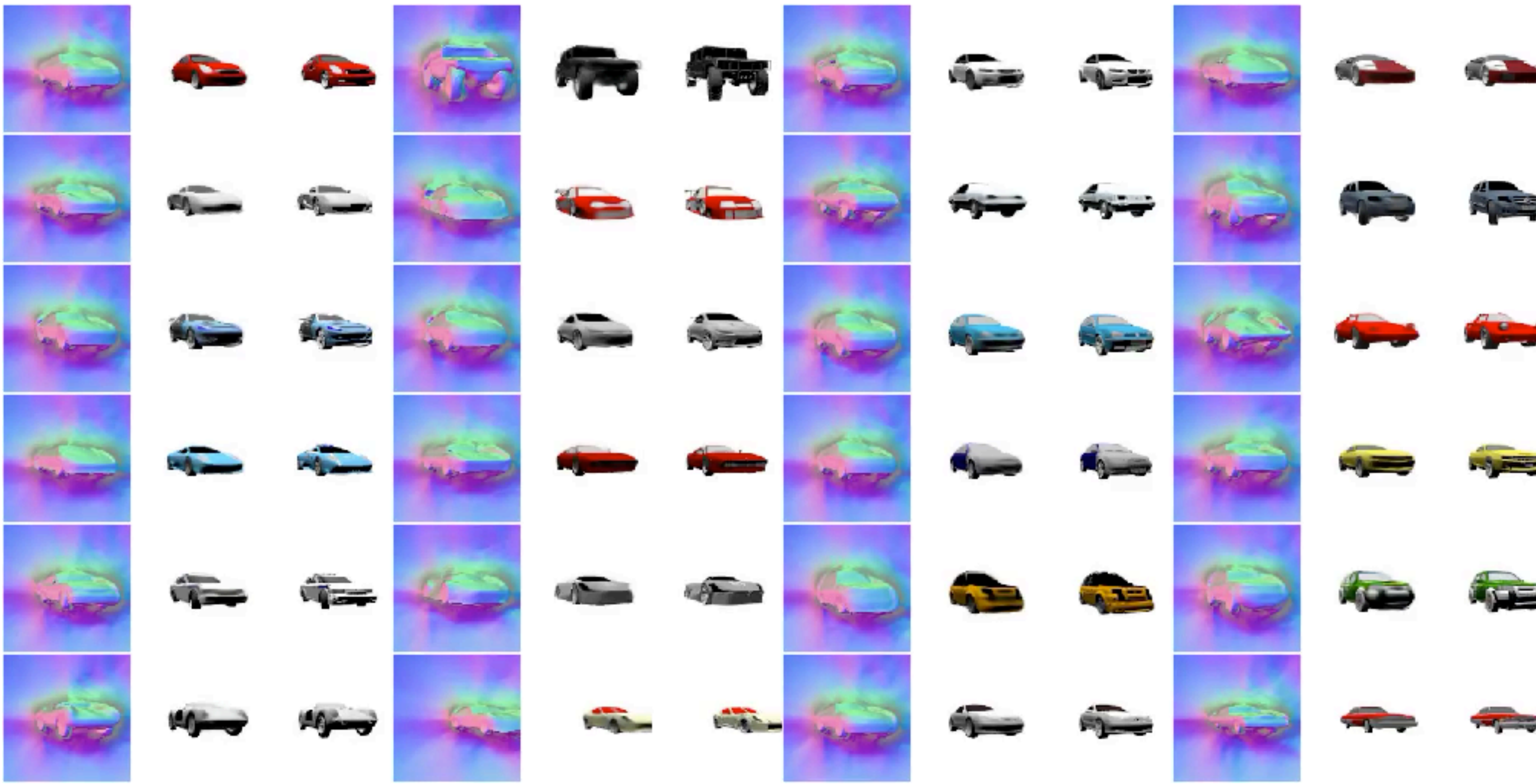
Per-pixel Embedding →RGB

Can now train end-to-end with posed images only!



View Synthesis: Shapenet Cars

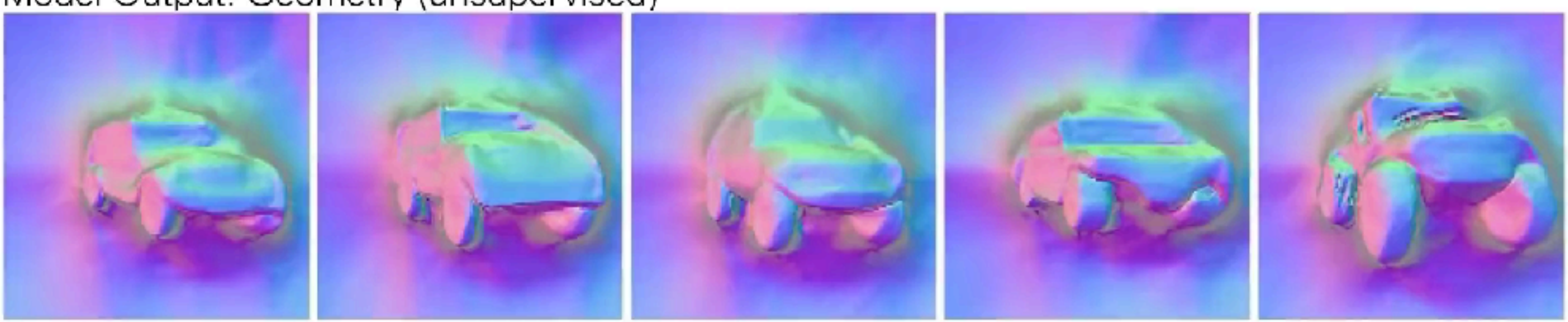
- Train using 50 observations per object, known cameras



Observation: <u>Single</u> Image

Model Output: Novel Views

Model Output: Geometry (unsupervised)



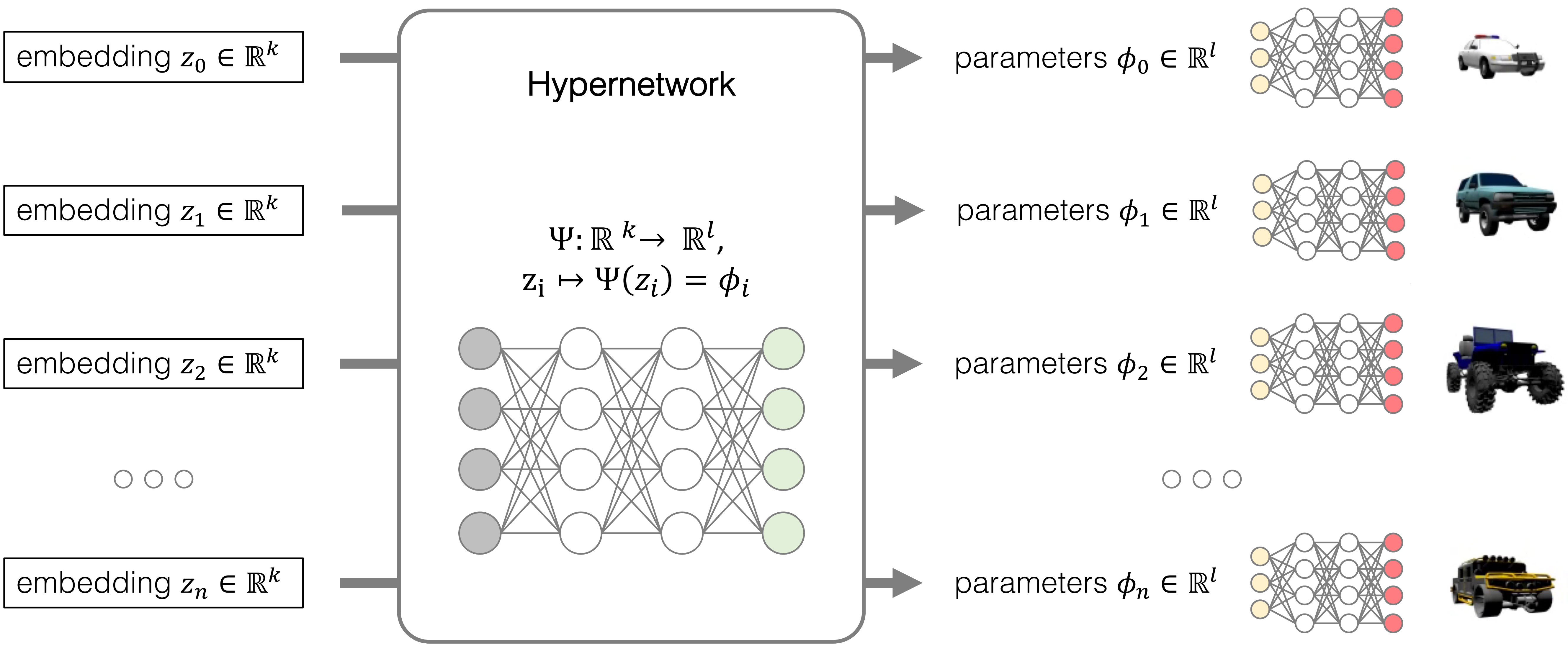
Sampling at arbitrary resolutions

Surface Normals

RGB

Can render scene at any resolution ϕ=f(X)

Each scene represented by its own SRN.



Latent Code Interpolation

- Interpolated latent codes give meaningful scenes

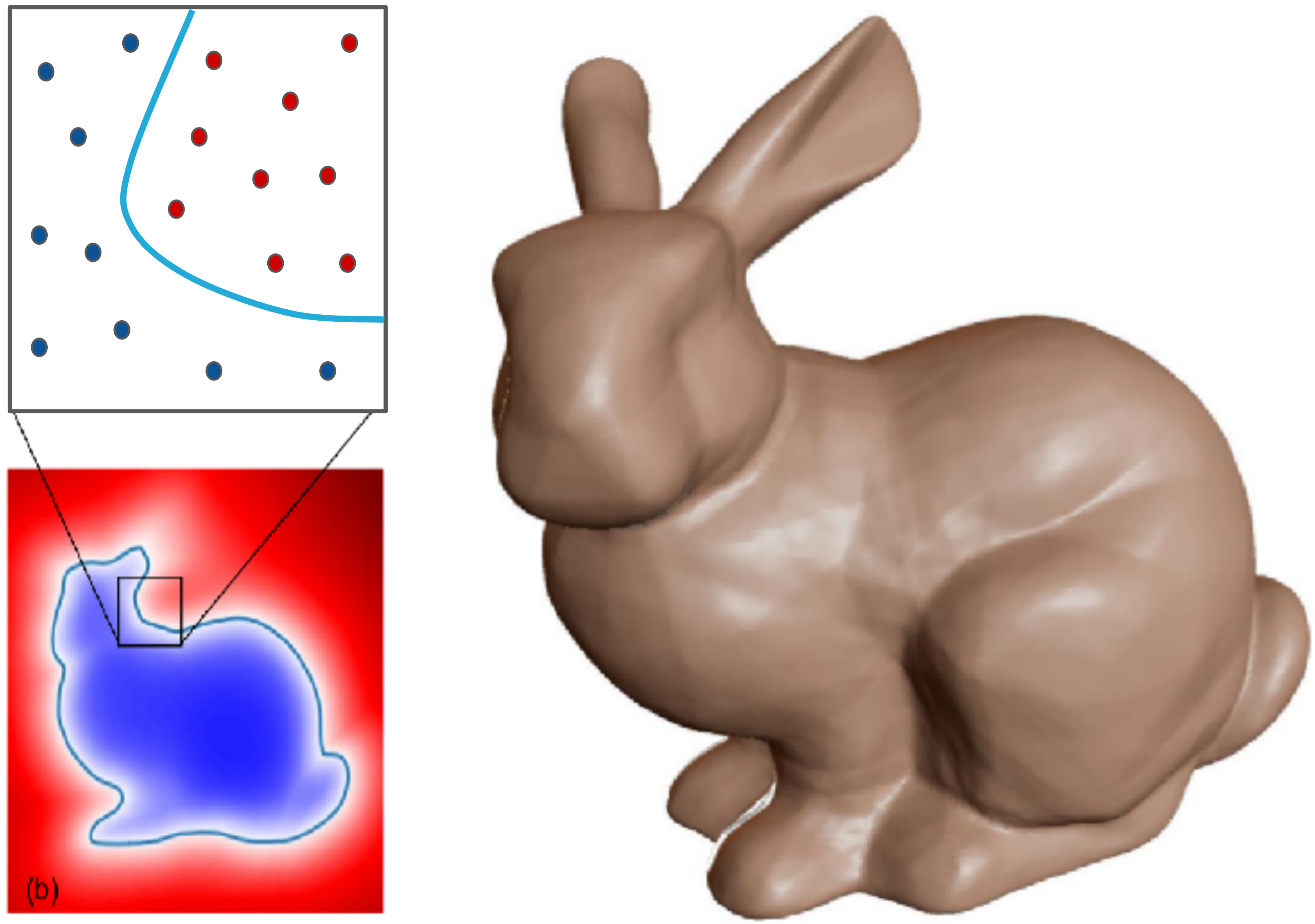
DeepSDF: Learning Continuous SDFs for Shape Representation

Jeong Joon Park, Peter Florence, Julian Straub,
Richard Newcombe, Steven Lovegrove

CVPR 2019

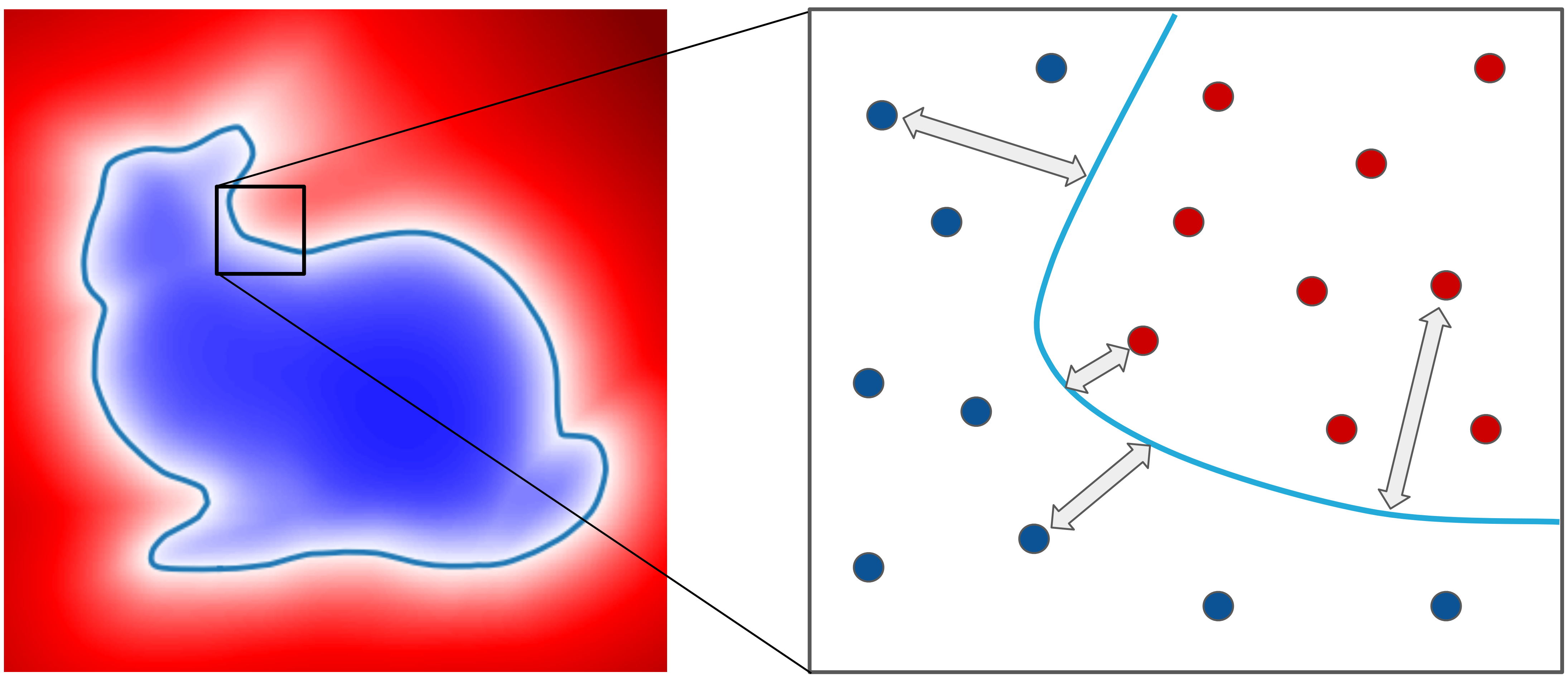
DeepSDF

- CPPN for signed distance function, SDF=f(X)



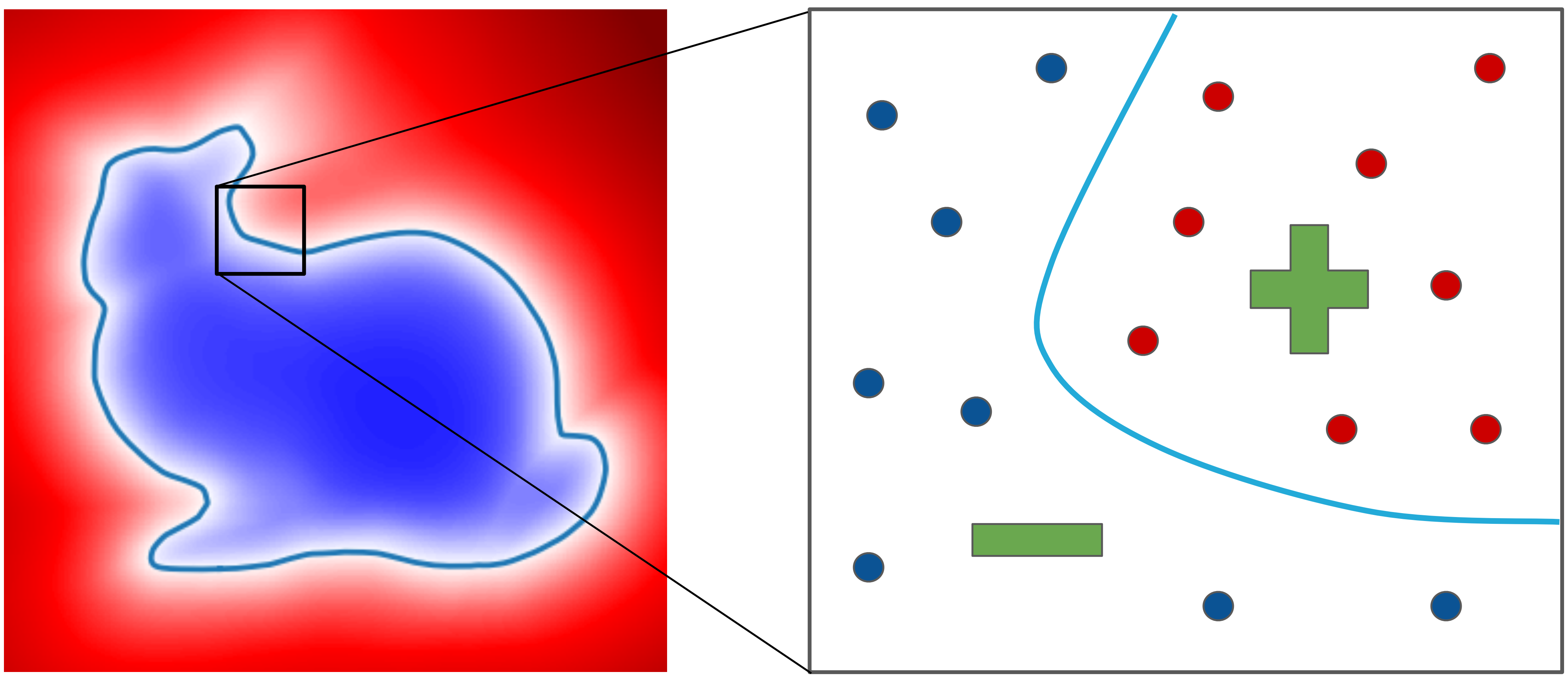
[Slides: Jeong Joon Park]

Signed Distance Function



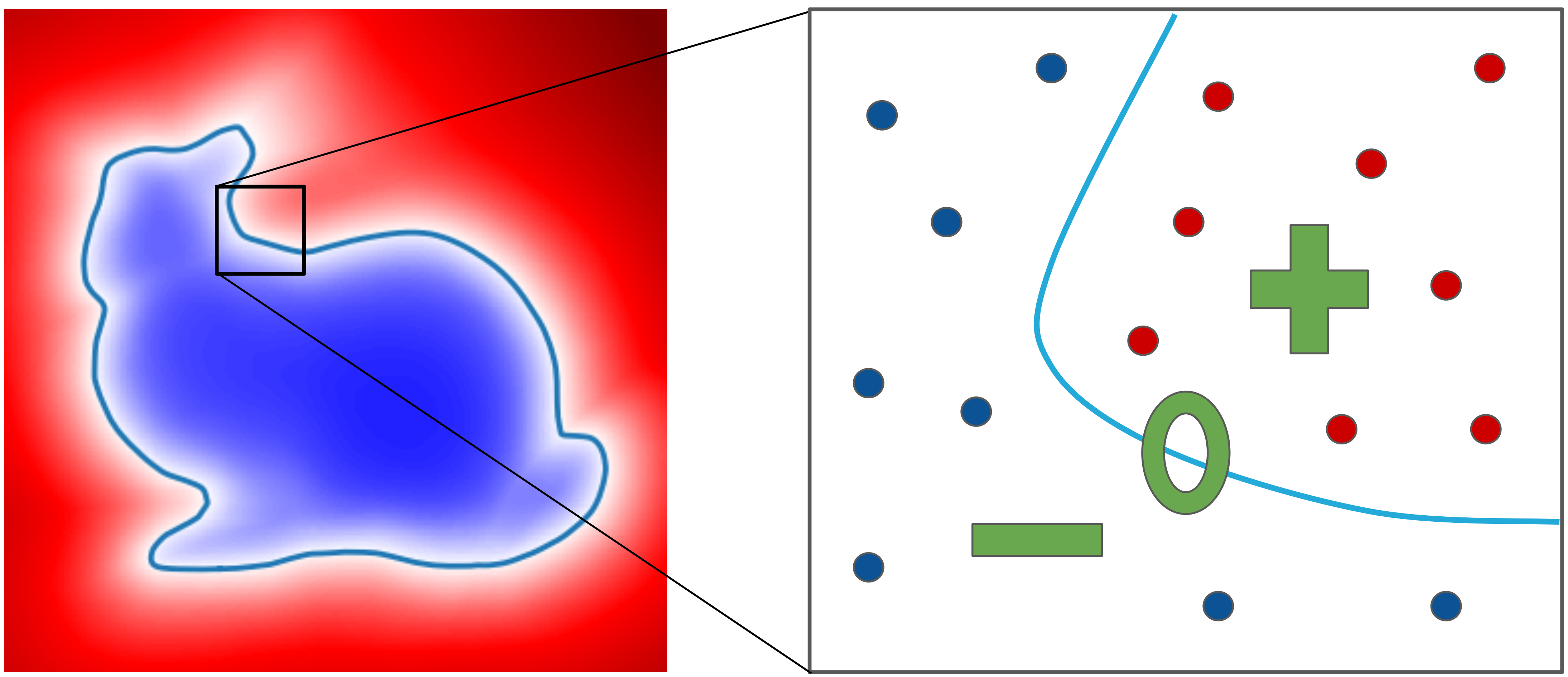
[Slides: Jeong Joon Park]

Signed Distance Function



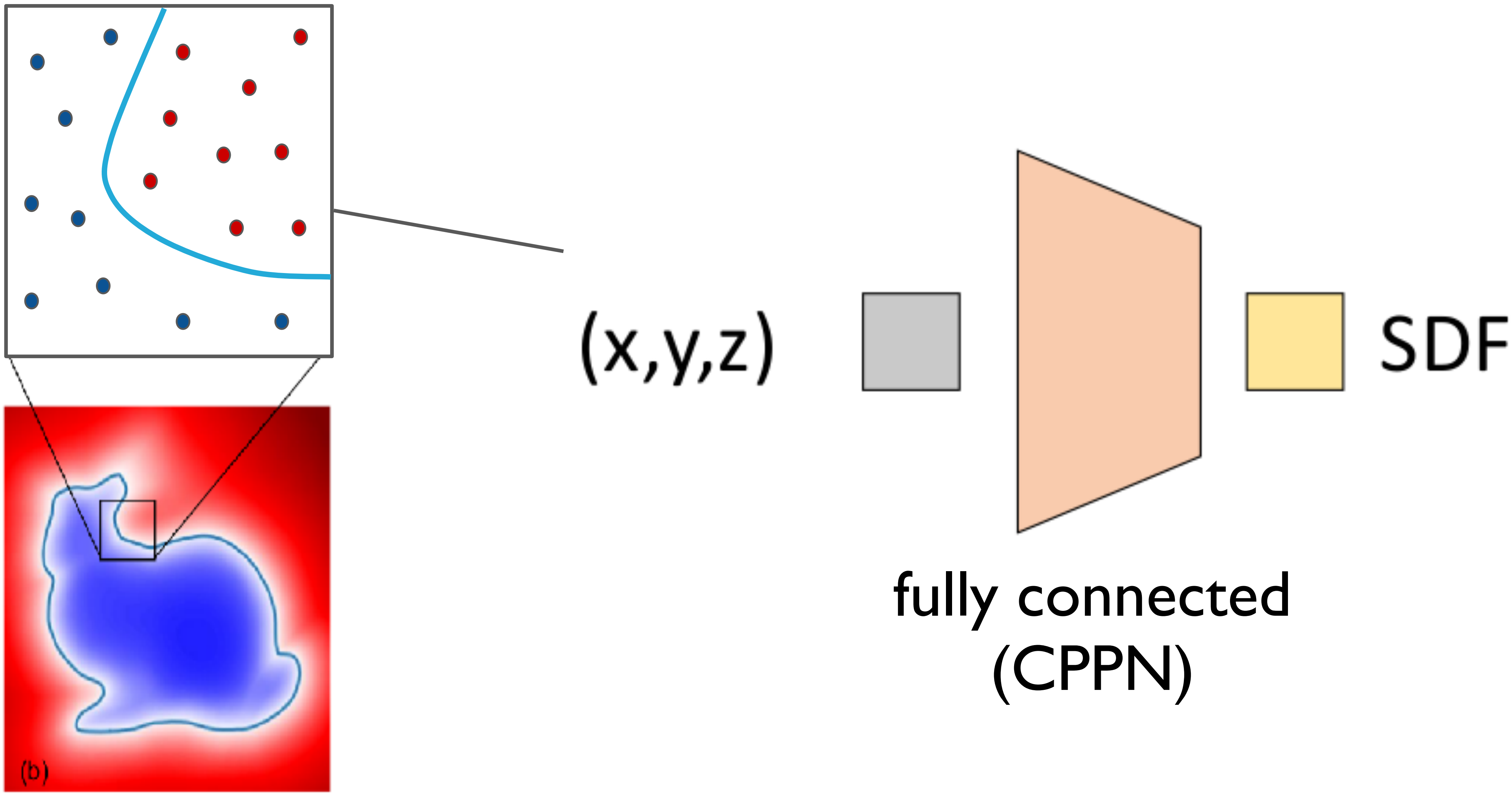
[Slides: Jeong Joon Park]

Signed Distance Function



[Slides: Jeong Joon Park]

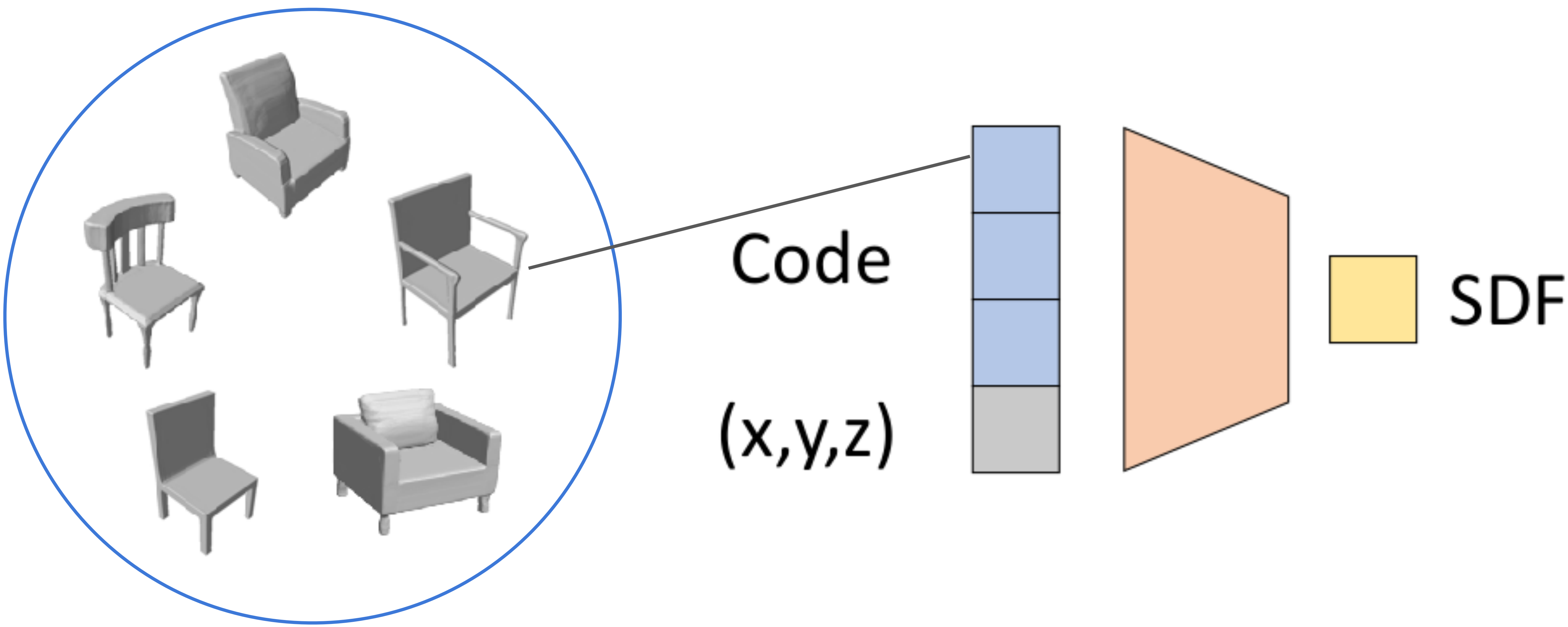
SDF Regression



Estimate parameters of fully connected net f(X) to fit known SDF

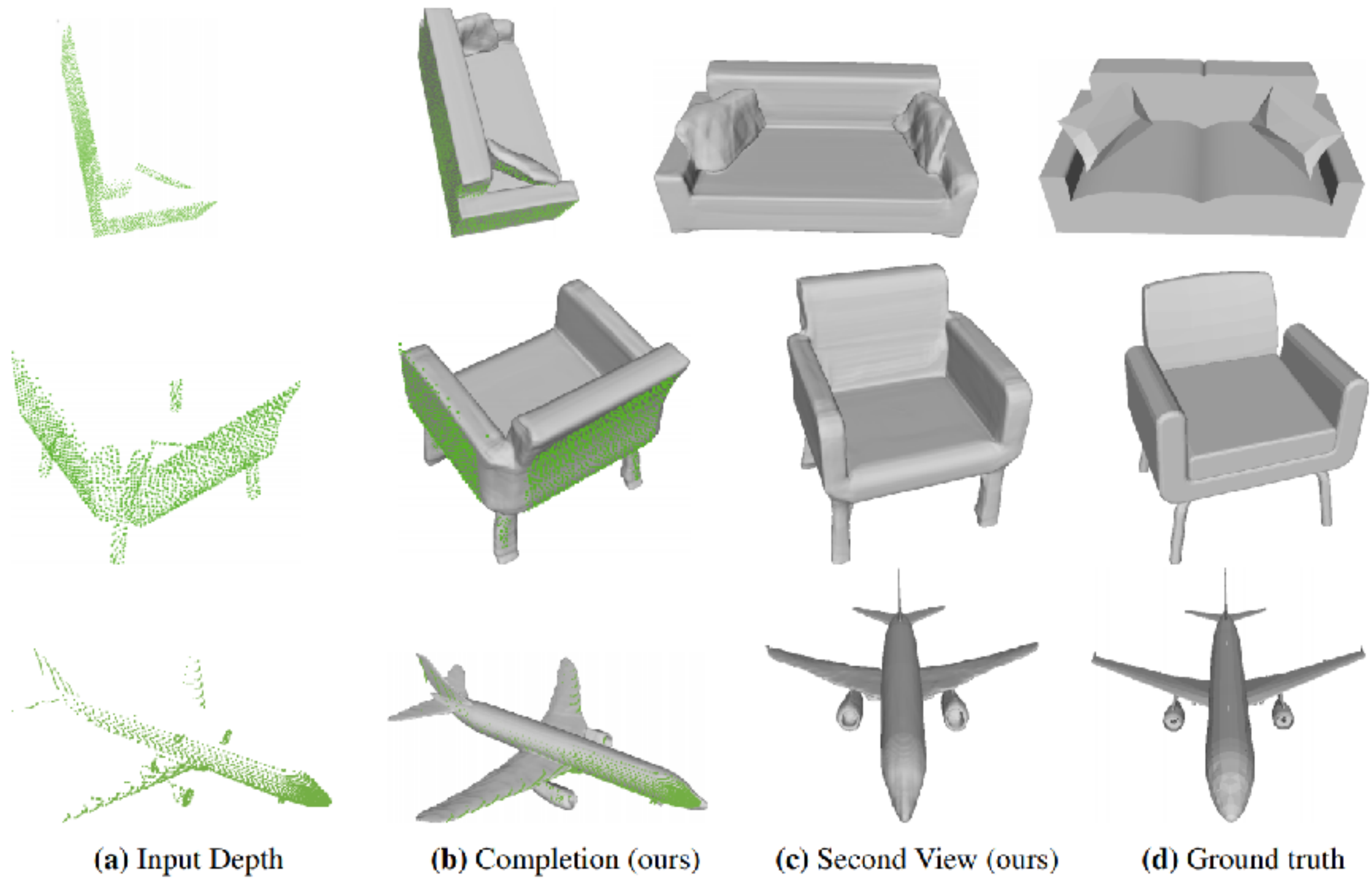
Shape Modelling

Coding Multiple Shapes



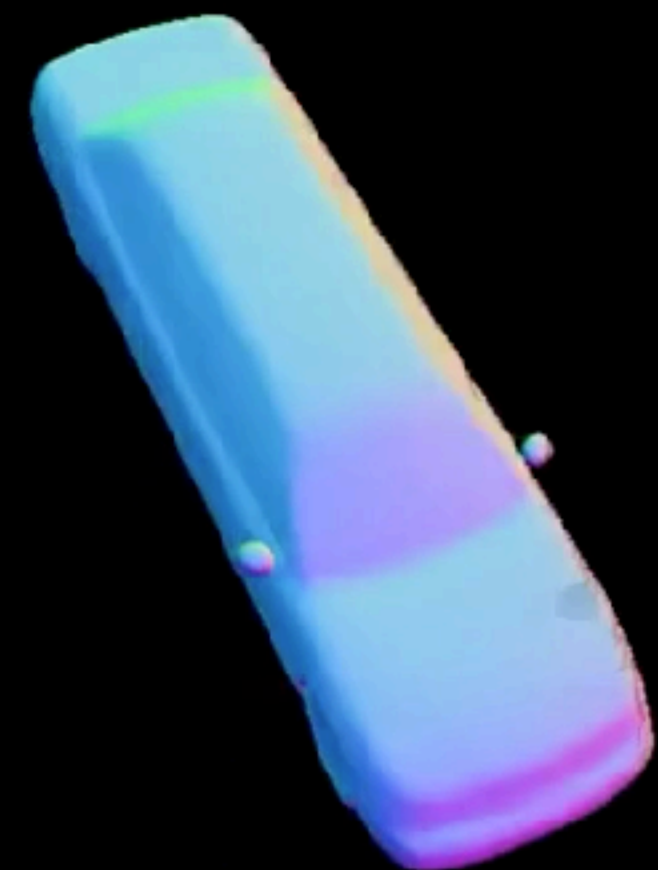
Assign random codes to each training object,
optimise network parameters to fit known 3D

Shape Completion



Optimise latent code given partial SDF by backprop to input

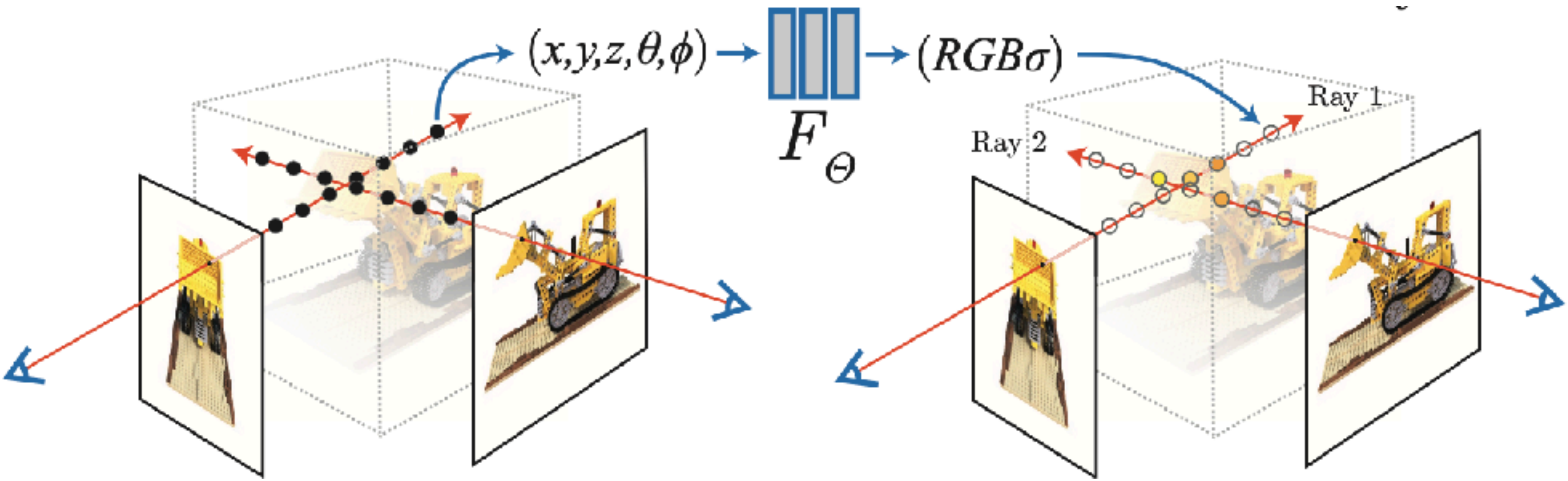
Learned Chair Shape Space

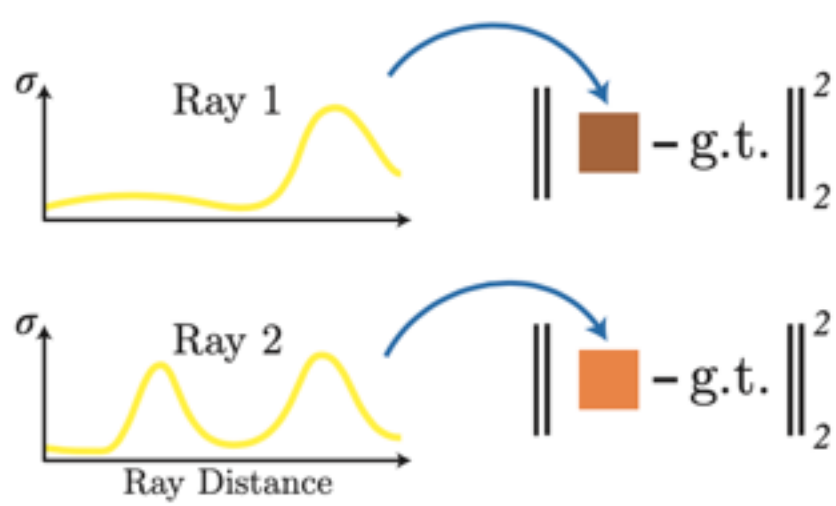


Learned Car Shape Space

Neural Radiance Fields

- Another continuous scene representation using a FCN

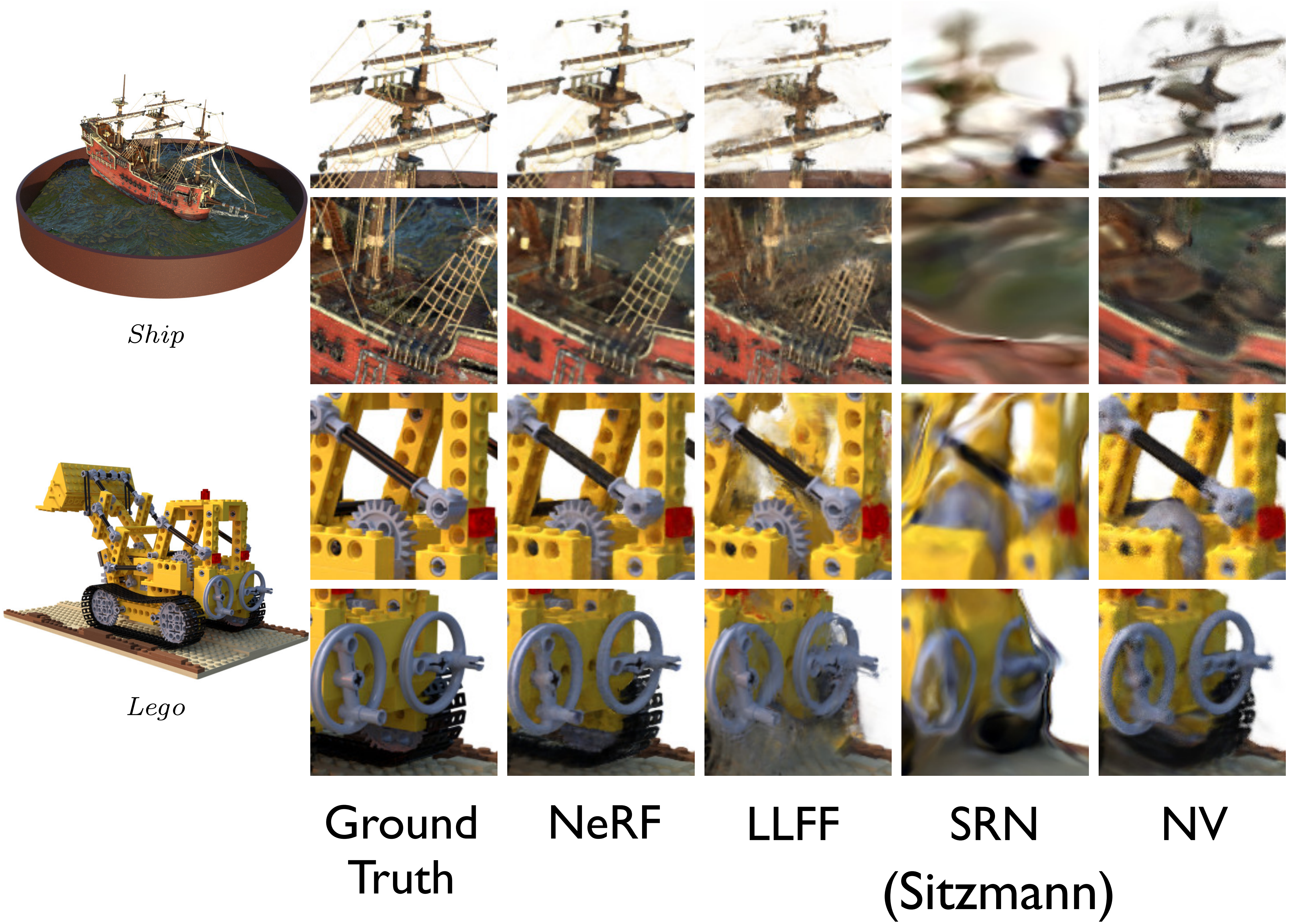




Predict density at each location, integrate along ray to get color (volume rendering)

[NeRF, Mildenhall, Srinivasan, Tancik et al. 2020]

Results



Neural Radiance Fields

- Neural Radiance Fields, ~10s of input views

matthewtancik.com/nerf

Next Lecture

- Image Generation, Generative Adversarial Networks