Predicting 3D Shapes from 2D Images

Justin Johnson

Facebook AI Research (FAIR)

6/16/2019

Justin Johnson

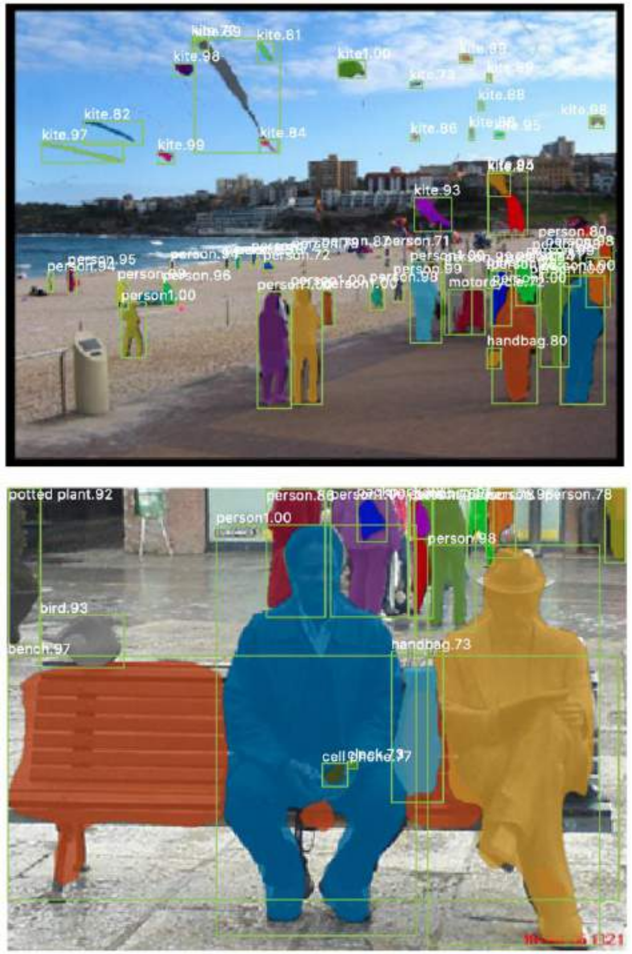
- Complete PhD at Stanford University, advised by Fei-Fei Li
- Currently a research scientist at Facebook AI Research
- Will join University of Michigan as AP from Fall 2019

Perceptual losses for real-time style transfer and super-resolution, ECCV 2016

Is computer vision solved?

Kirillov et al, “Panoptic Feature Pyramid Networks”, CVPR 2019

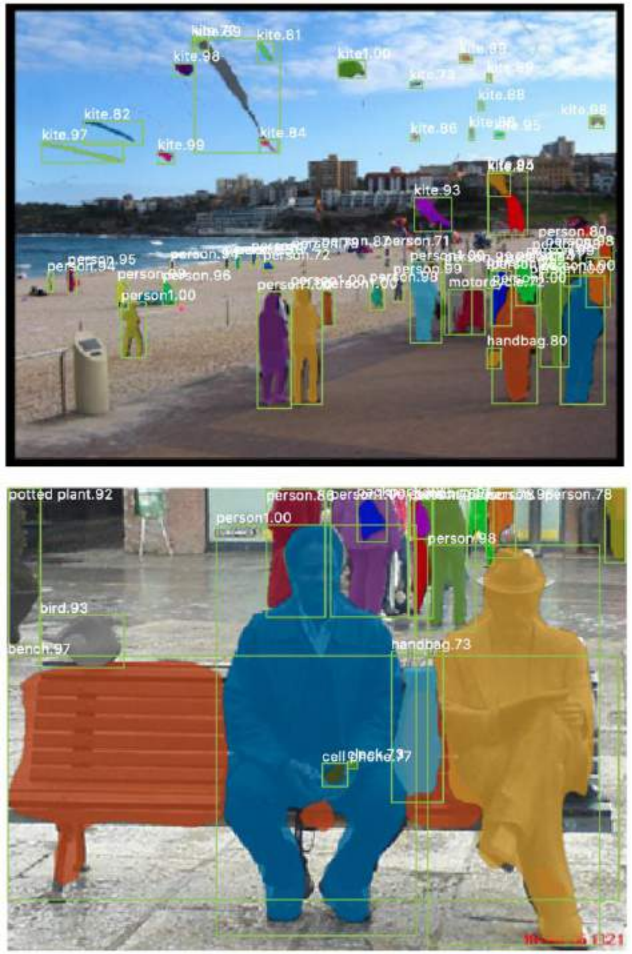
Mask R-CNN:
2D Image -> 2D shapes



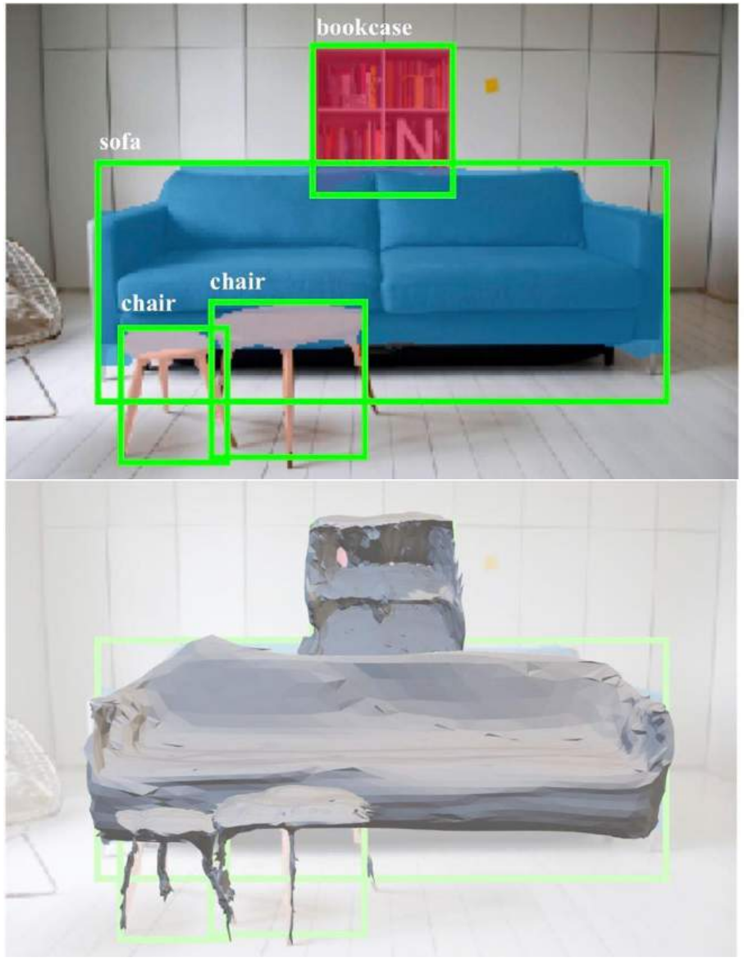
He, Gkioxari, Dollár, and Girshick, “Mask R-CNN”, ICCV 2017

A new dimension for recognition

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



Mesh R-CNN:
2D Image -> **3D** shapes



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

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

Why care about 3D perception?

Autonomous Vehicles

Why care about 3D perception?

Autonomous Vehicles

VR / AR

Why care about 3D perception?

Autonomous Vehicles

VR / AR

The world is 3D!

Predicting 3D Shapes from 2D Images

Shape Representations

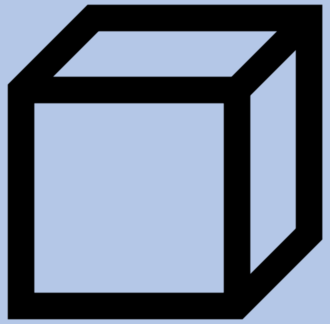
Metrics

Camera Systems

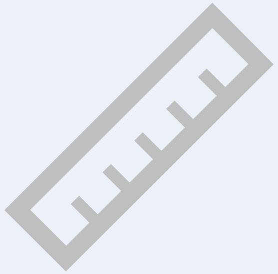
Datasets

Predicting 3D Shapes from 2D Images

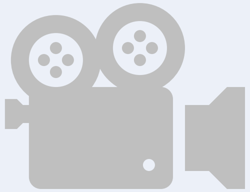
Shape Representations



Metrics

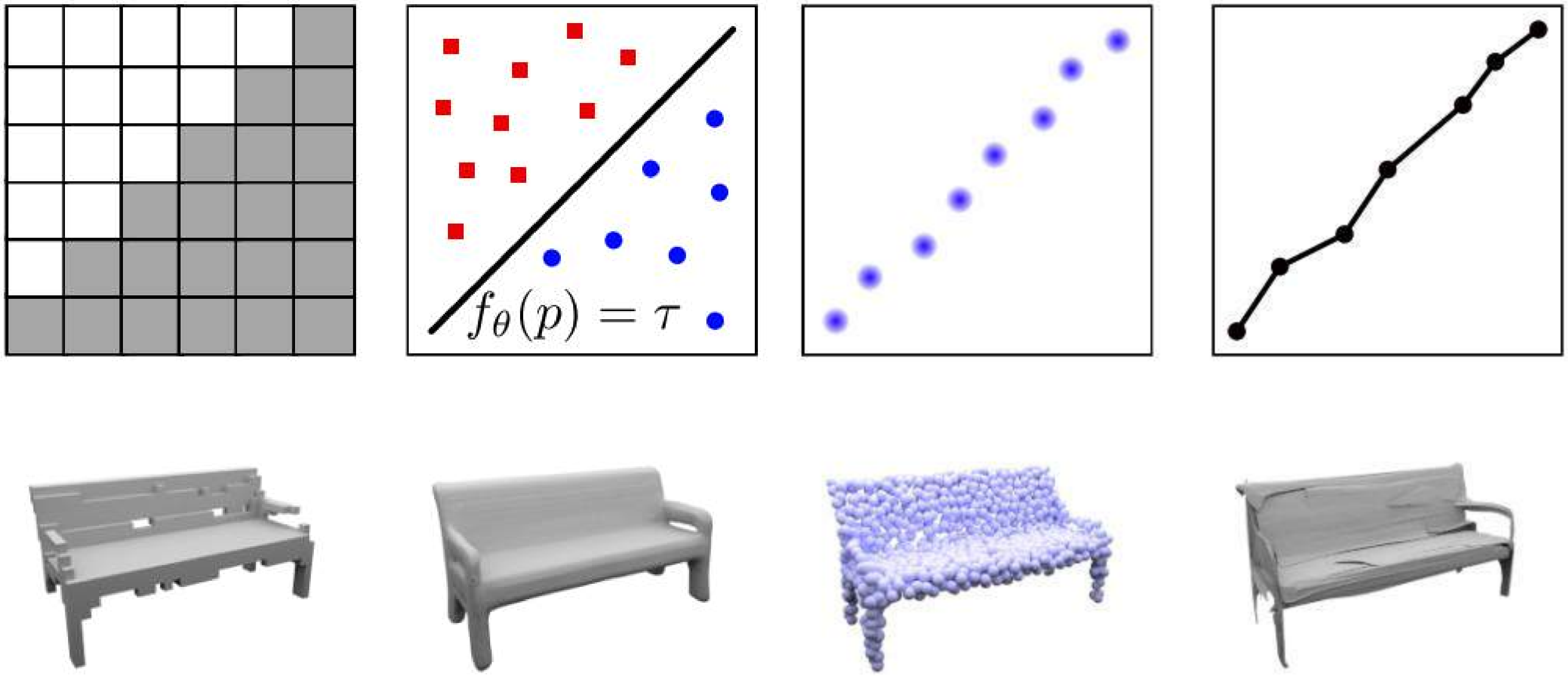


Camera Systems



Datasets

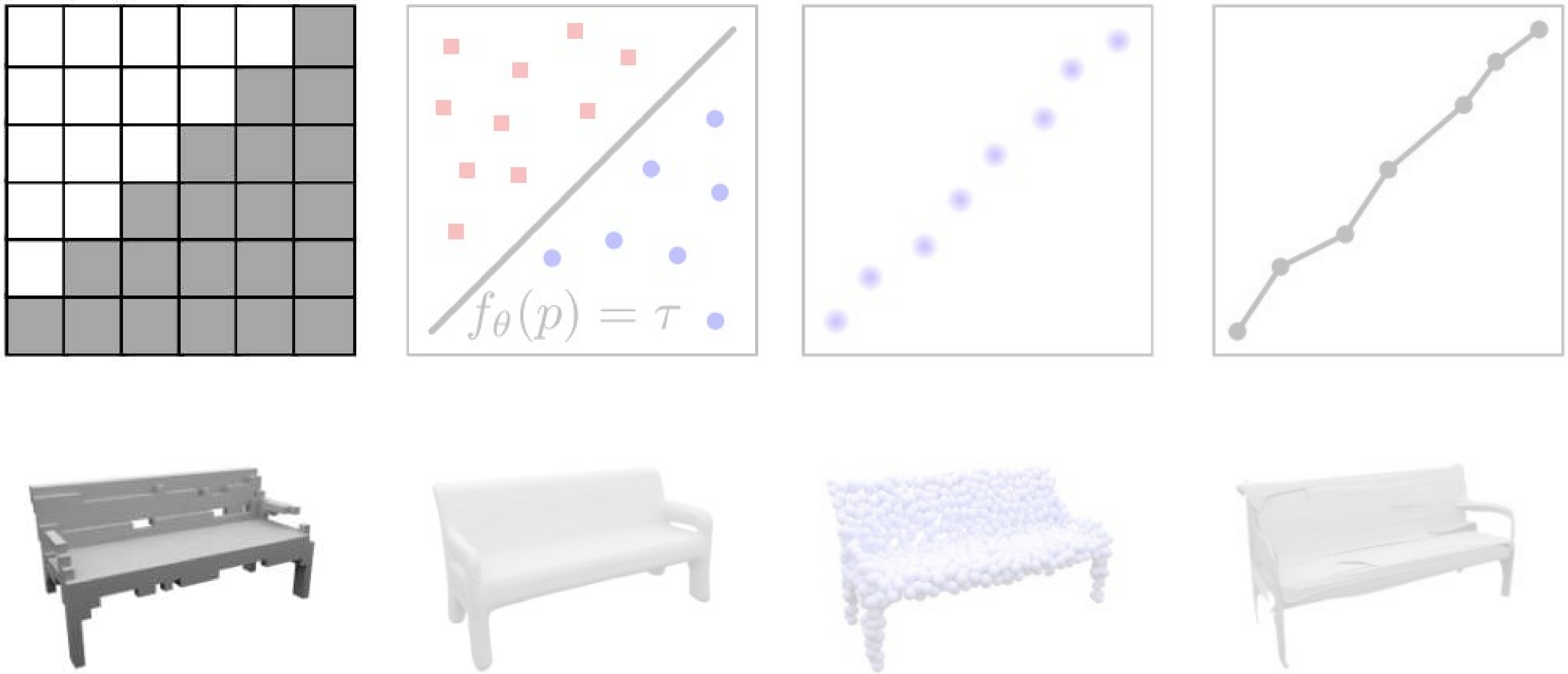
3D Shape Representations



Voxels | Implicit Surface | Pointcloud | Mesh

Figure credit: Mescheder et al, "Occupancy Networks: Learning 3D Reconstruction in Function Space", CVPR 2019

3D Shape Representations

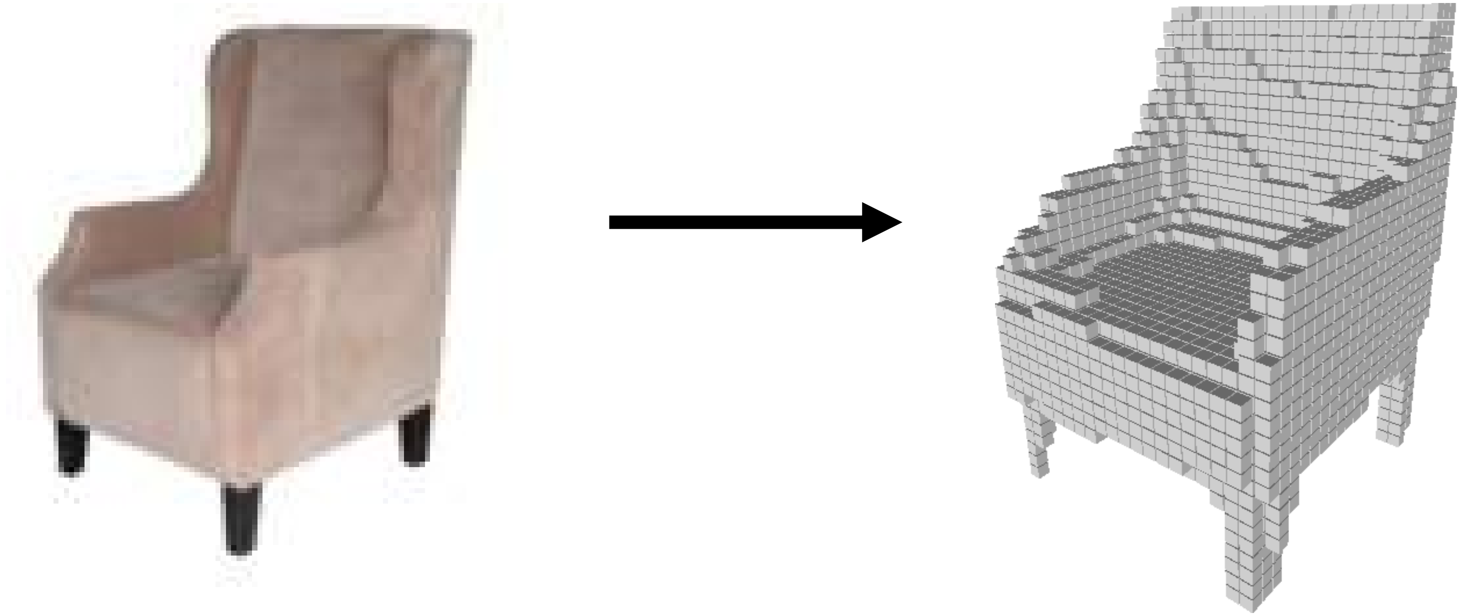


Voxels Implicit Surface Pointcloud Mesh

Figure credit: Mescheder et al, "Occupancy Networks: Learning 3D Reconstruction in Function Space", CVPR 2019

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

Predicting Voxels: 3D Convolution

Last layer:
Conv3d(H->1)

3D convolution

2D convolution

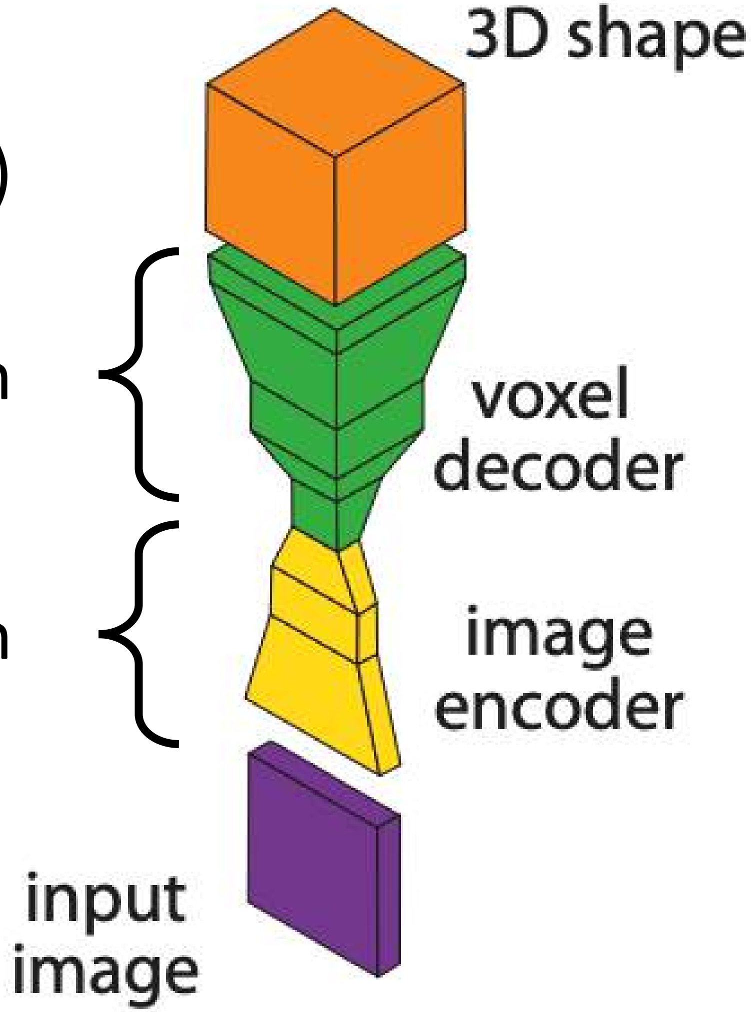


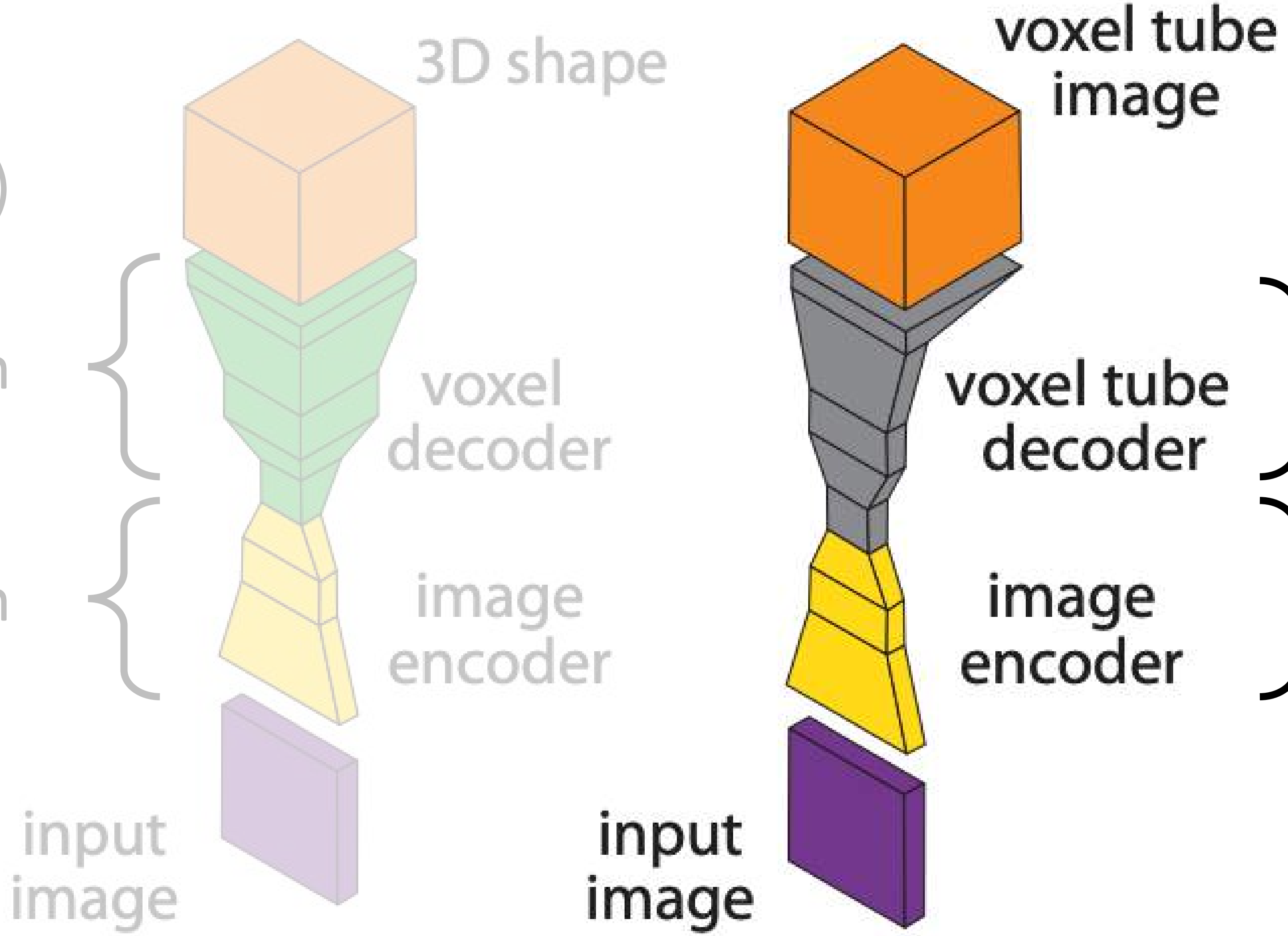
Figure Credit: Richter and Roth: “Matryoshka Networks: Predicting 3D Geometry via Nested Shape Layers”, CVPR 2018

Predicting Voxels: Voxel Tubes

Last layer:
Conv3d(H->1)

3D convolution

2D convolution



Channels ->
tube of voxels

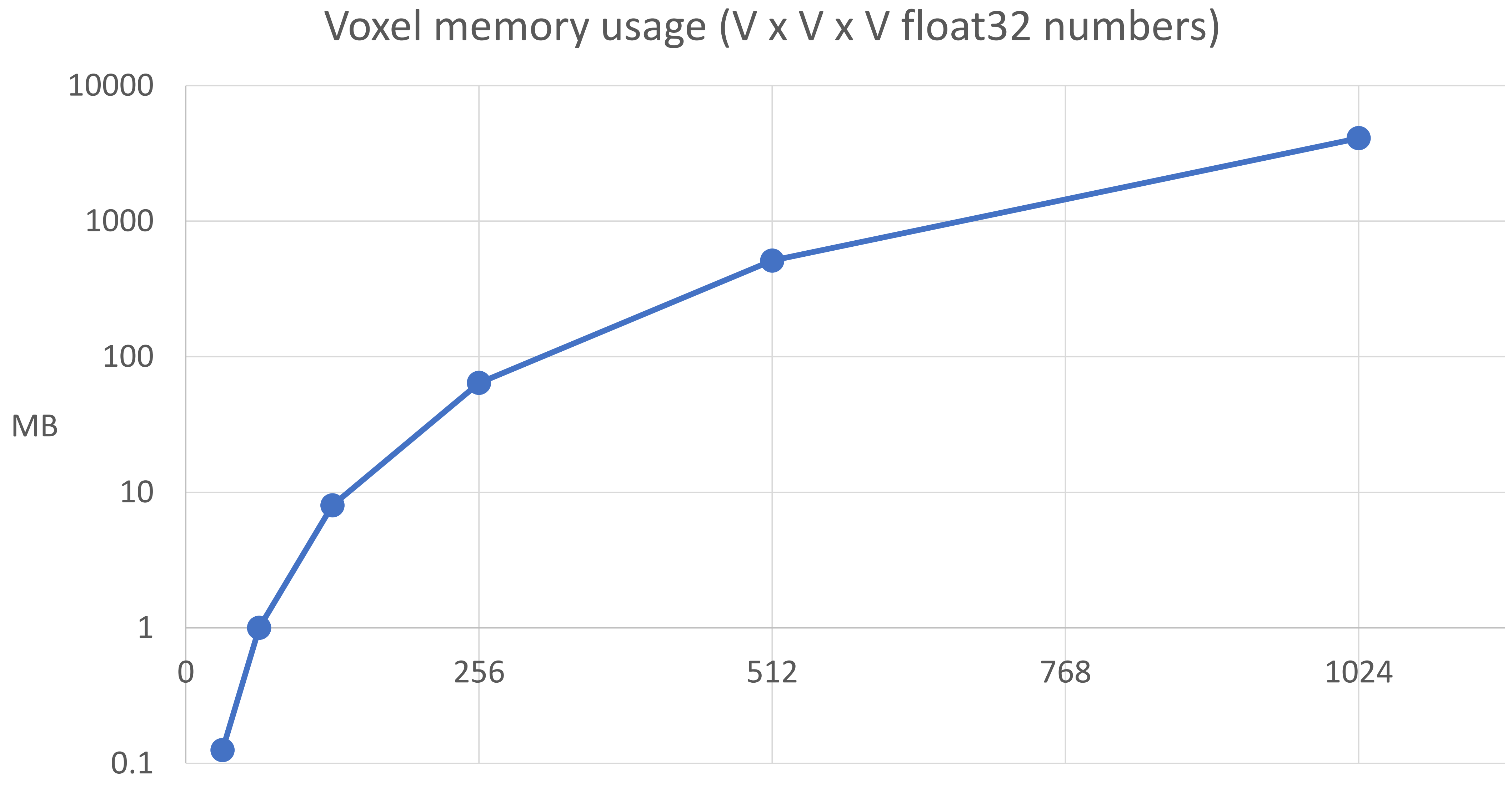
2D convolution

2D convolution

Figure Credit: Richter and Roth: “Matryoshka Networks: Predicting 3D Geometry via Nested Shape Layers”, CVPR 2018

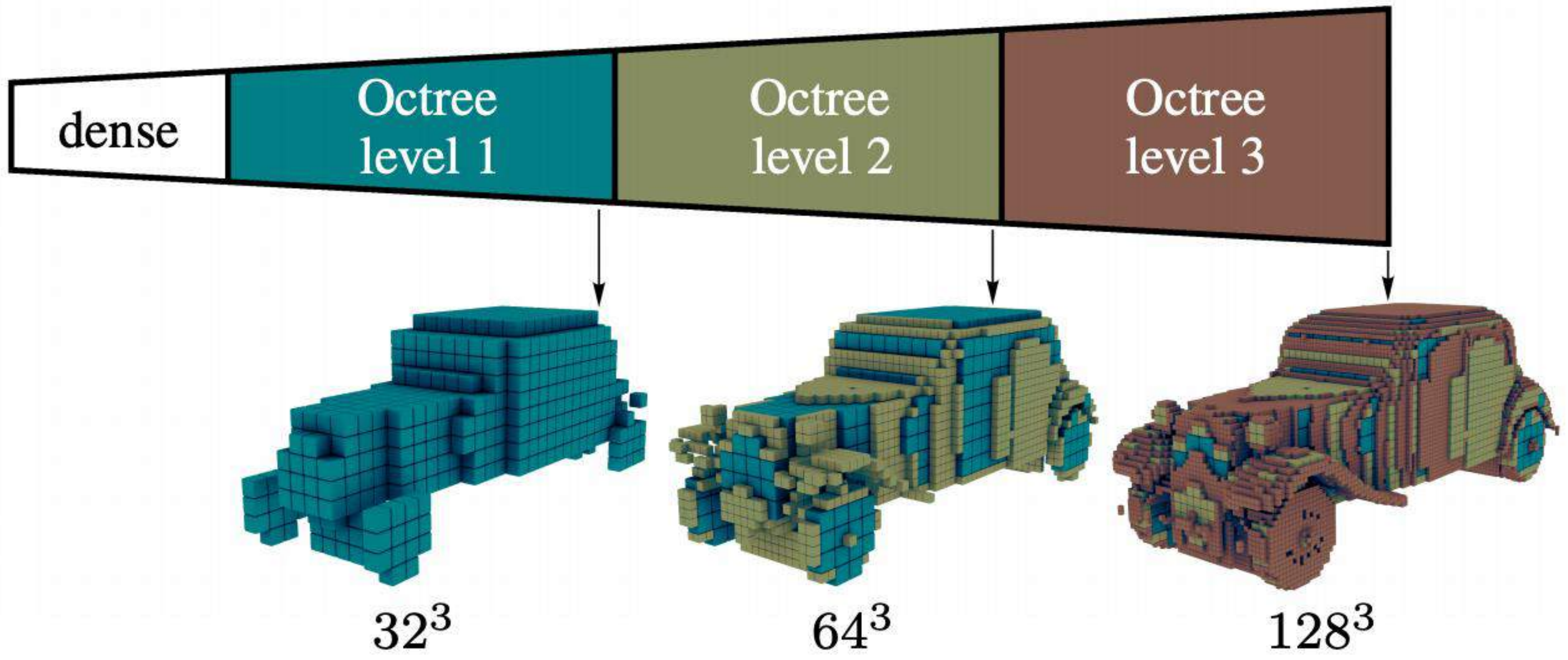
Voxel Problems: Memory Usage

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



Scaling Voxels: Oct-Trees

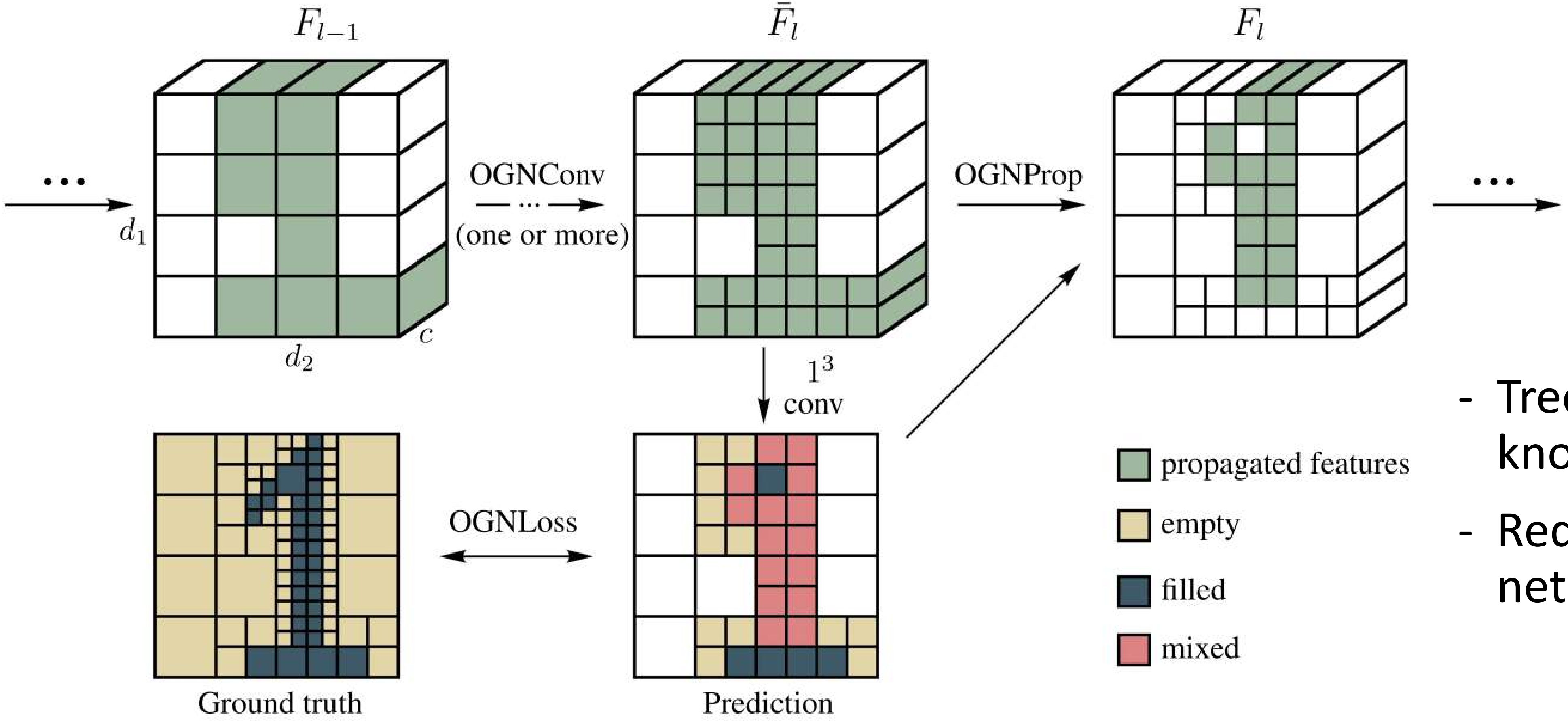
Use voxel grids with heterogenous resolution!



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

Scaling Voxels: Oct-Trees

Use voxel grids with heterogenous resolution!

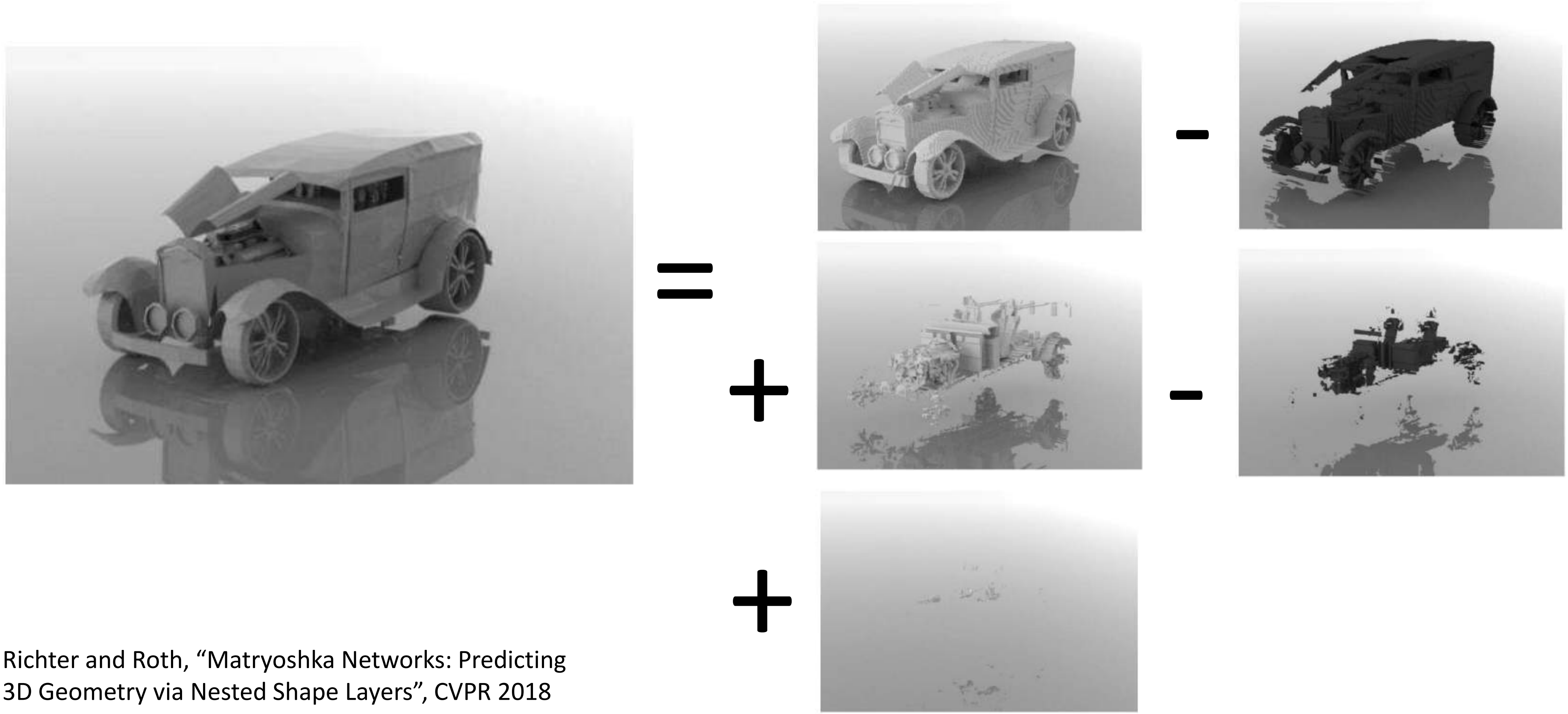


- Tree structure to be known ahead of time
- Requiring custom network layer

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

Scaling Voxels: Nested Shape Layers

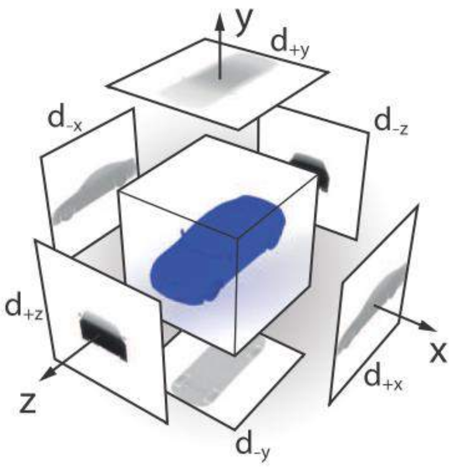
Predict shape as a composition
of positive and negative spaces

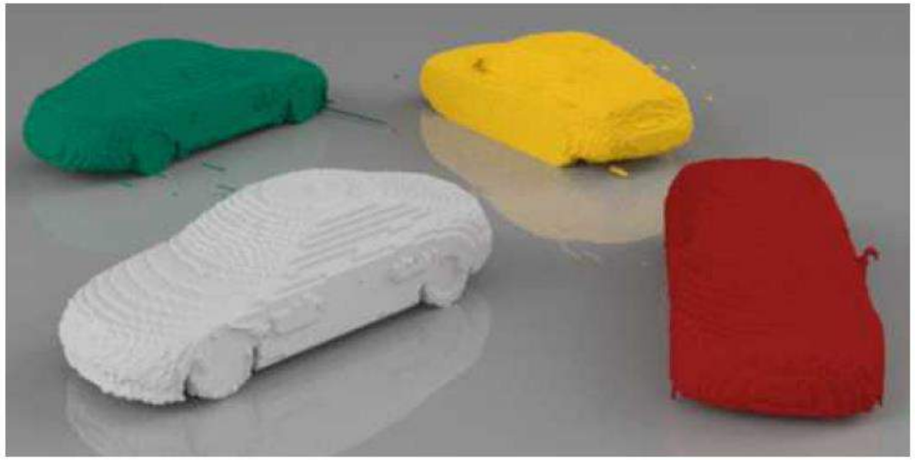


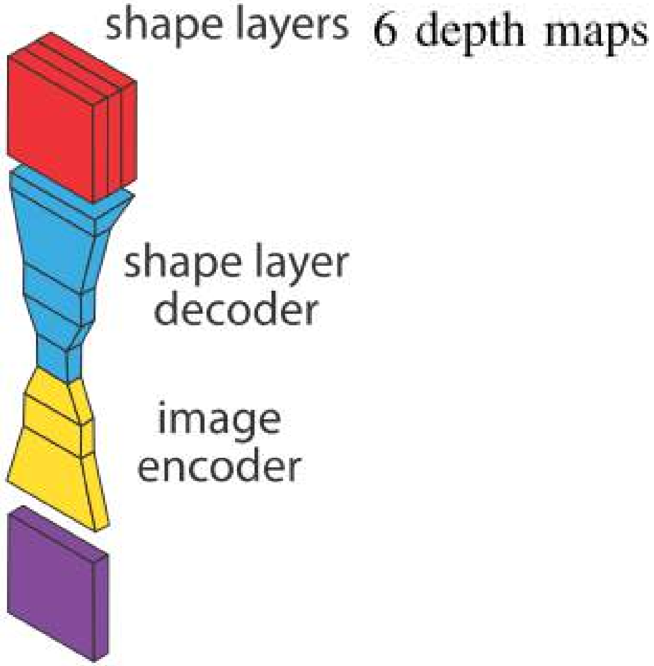
Richter and Roth, “Matryoshka Networks: Predicting
3D Geometry via Nested Shape Layers”, CVPR 2018

Scaling Voxels: Nested Shape Layers

Predict shape as a composition
of positive and negative spaces







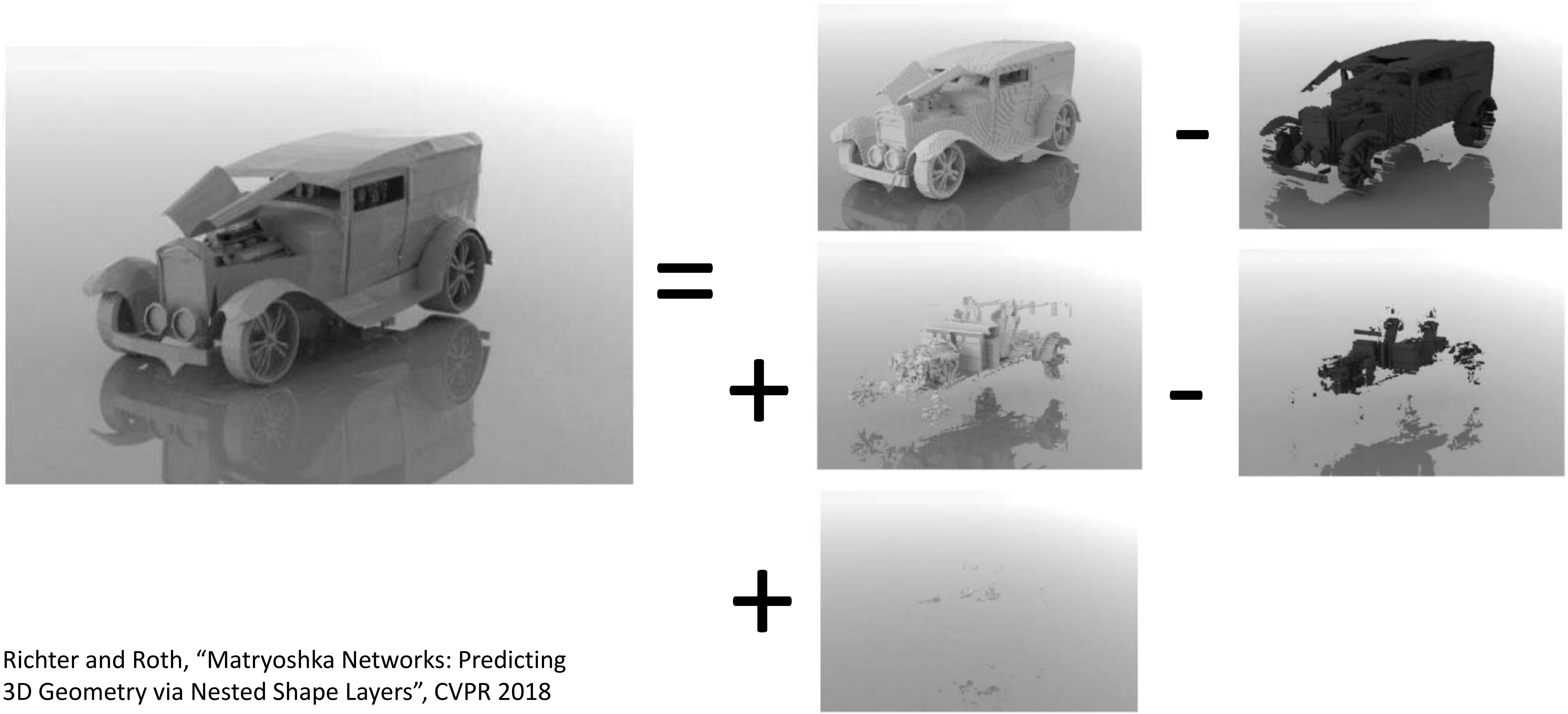
$$S_x \equiv \{(i,j,k) \mid d_{-x}(j,k) \le i \le n_o - d_{+x}(j,k)\}$$
$$S_y \equiv \{(i,j,k) \mid d_{-y}(i,k) \le j \le n_o - d_{+y}(i,k)\}$$
$$S_z \equiv \{(i,j,k) \mid d_{-z}(i,j) \le k \le n_o - d_{+z}(i,j)\}$$

$$S = \phi(\mathbf{d}) \equiv S_x \cap S_y \cap S_z \quad \text{with} \quad \phi : \mathcal{D} \to \mathcal{S}.$$

Richter and Roth, “Matryoshka Networks: Predicting 3D Geometry via Nested Shape Layers”, CVPR 2018

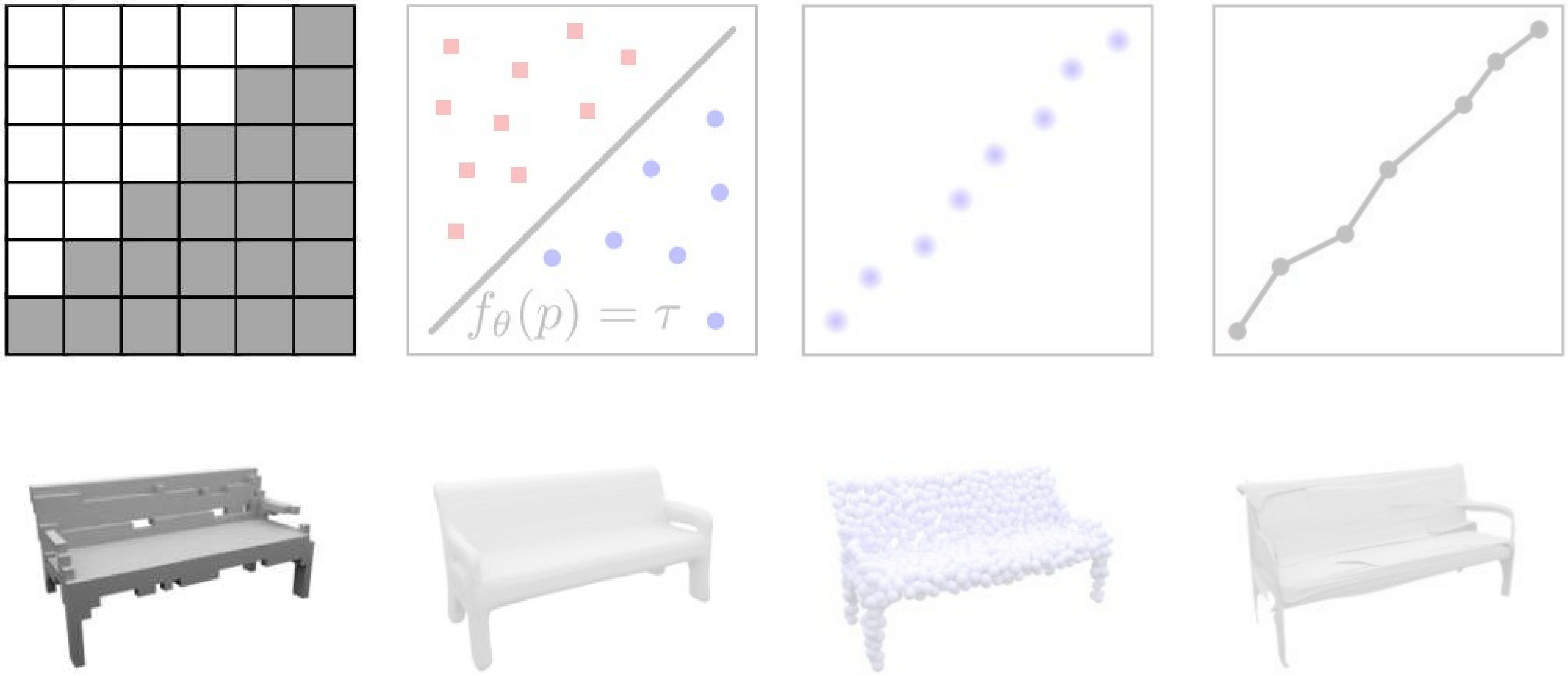
Scaling Voxels: Nested Shape Layers

Predict shape as a composition
of positive and negative spaces



Richter and Roth, “Matryoshka Networks: Predicting
3D Geometry via Nested Shape Layers”, CVPR 2018

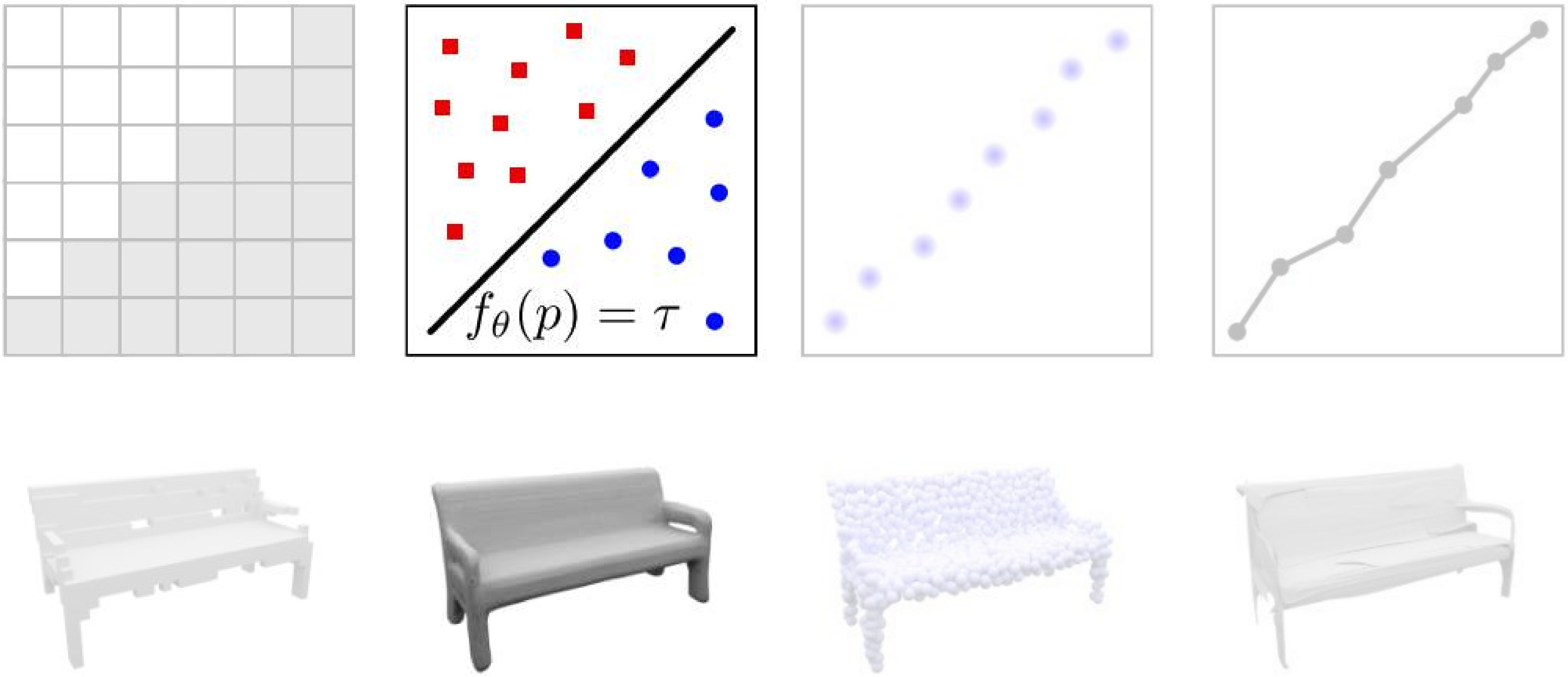
3D Shape Representations



Voxels | Implicit Surface | Pointcloud | Mesh

Figure credit: Mescheder et al, "Occupancy Networks: Learning 3D Reconstruction in Function Space", CVPR 2019

3D Shape Representations



Voxels | Implicit Surface | Pointcloud | Mesh

Figure credit: Mescheder et al, “Occupancy Networks: Learning 3D Reconstruction in Function Space”, CVPR 2019

3D Shape Representations: Implicit Function

Learn a function to classify arbitrary 3D points as inside / outside the shape

$$o : \mathbb{R}^3 \to \{0, 1\}$$

The surface of the 3D object is the level set $\{x : o(x) = \frac{1}{2}\}$

an observation $x \in \mathcal{X}$

$p \in \mathbb{R}^3$ to $\mathbb{R}$

$\longrightarrow$ $(p, x) \in \mathbb{R}^3 \times \mathcal{X}$ as input

$$f_\theta : \mathbb{R}^3 \times \mathcal{X} \to [0, 1]$$

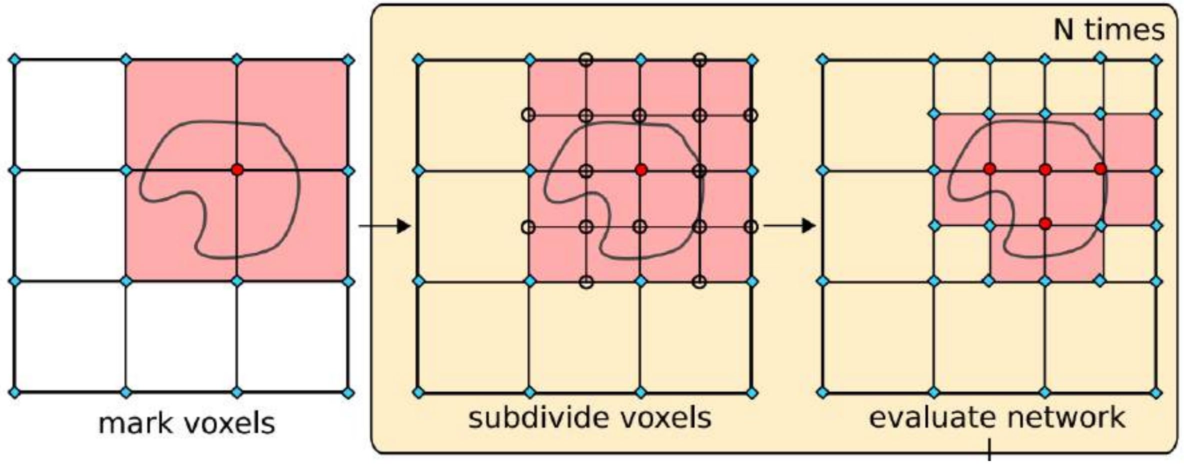
Mescheder et al, "Occupancy Networks: Learning 3D Reconstruction in Function Space", CVPR 2019

3D Shape Representations: Implicit Function

Learn a function to classify arbitrary 3D points as inside / outside the shape

$$o : \mathbb{R}^3 \to \{0, 1\}$$

The surface of the 3D object is the level set $\{x : o(x) = \tfrac{1}{2}\}$



Allows for multiscale outputs like Oct-Trees

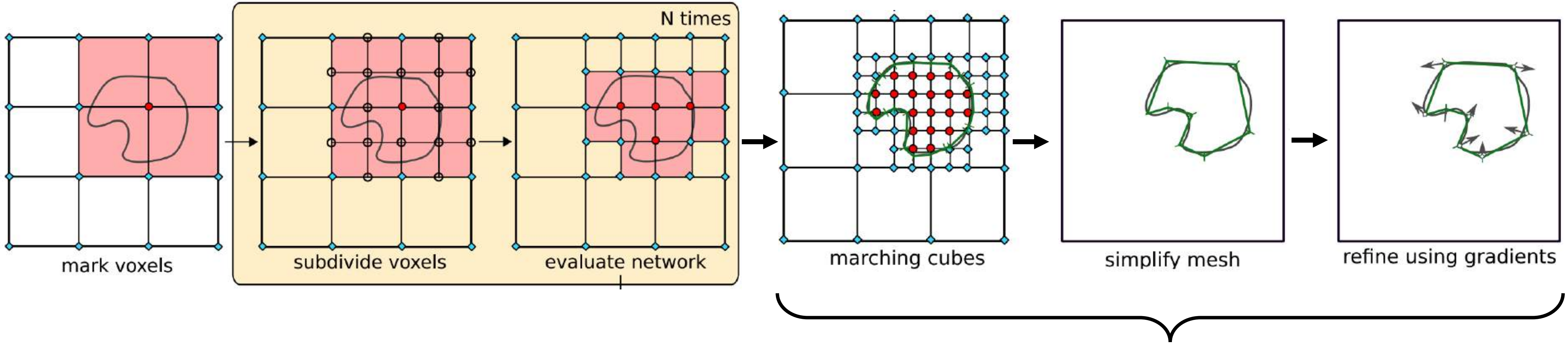
Mescheder et al, "Occupancy Networks: Learning 3D Reconstruction in Function Space", CVPR 2019

3D Shape Representations: Implicit Function

Learn a function to classify arbitrary 3D points as inside / outside the shape

$$o : \mathbb{R}^3 \to \{0, 1\}$$

The surface of the 3D object is the level set $\{x : o(x) = ½\}$



Extracting mesh outputs requires complex post-processing

Mescheder et al, "Occupancy Networks: Learning 3D Reconstruction in Function Space", CVPR 2019

3D Shape Representations

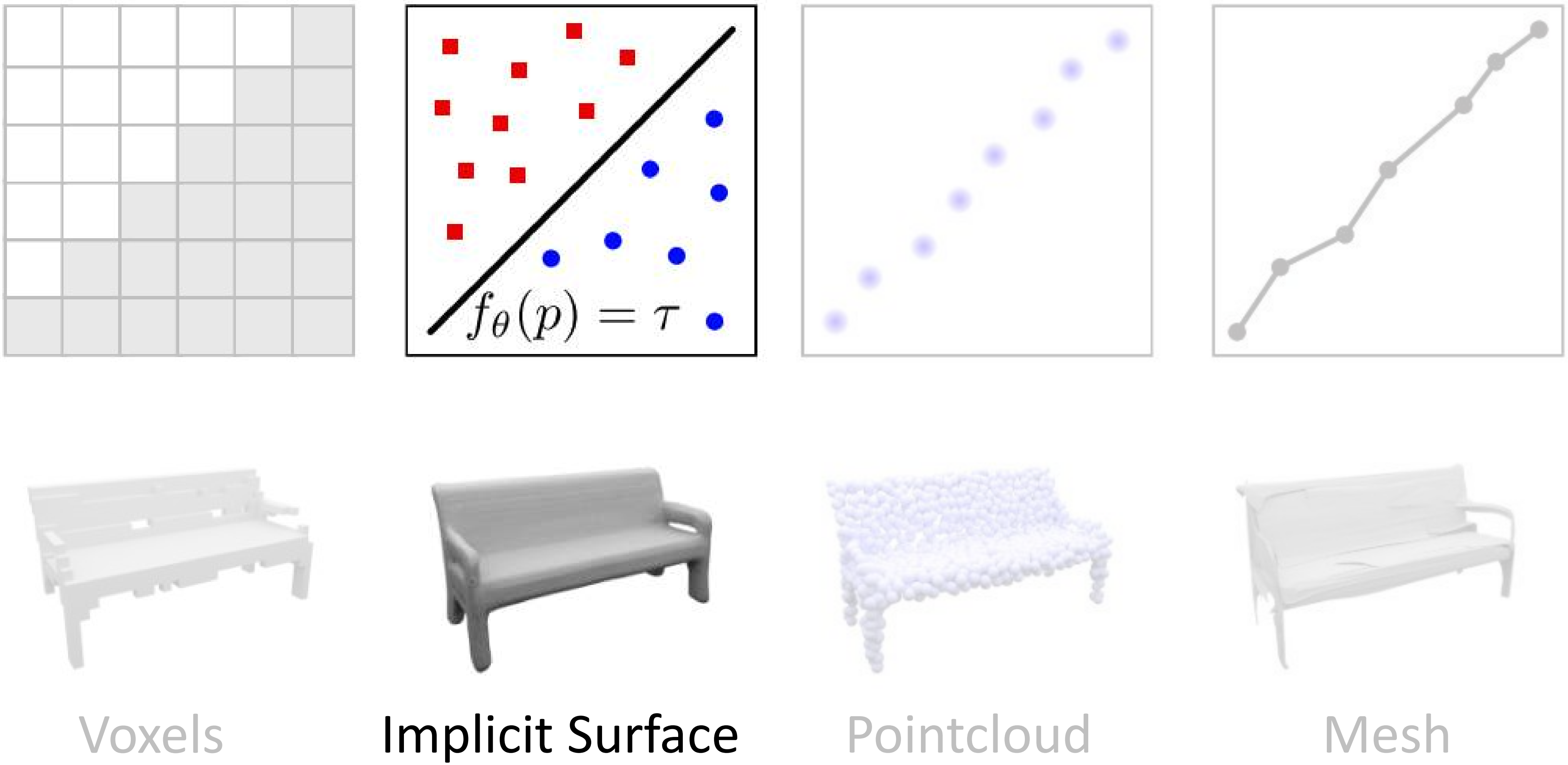


Figure credit: Mescheder et al, "Occupancy Networks: Learning 3D Reconstruction in Function Space", CVPR 2019

3D Shape Representations

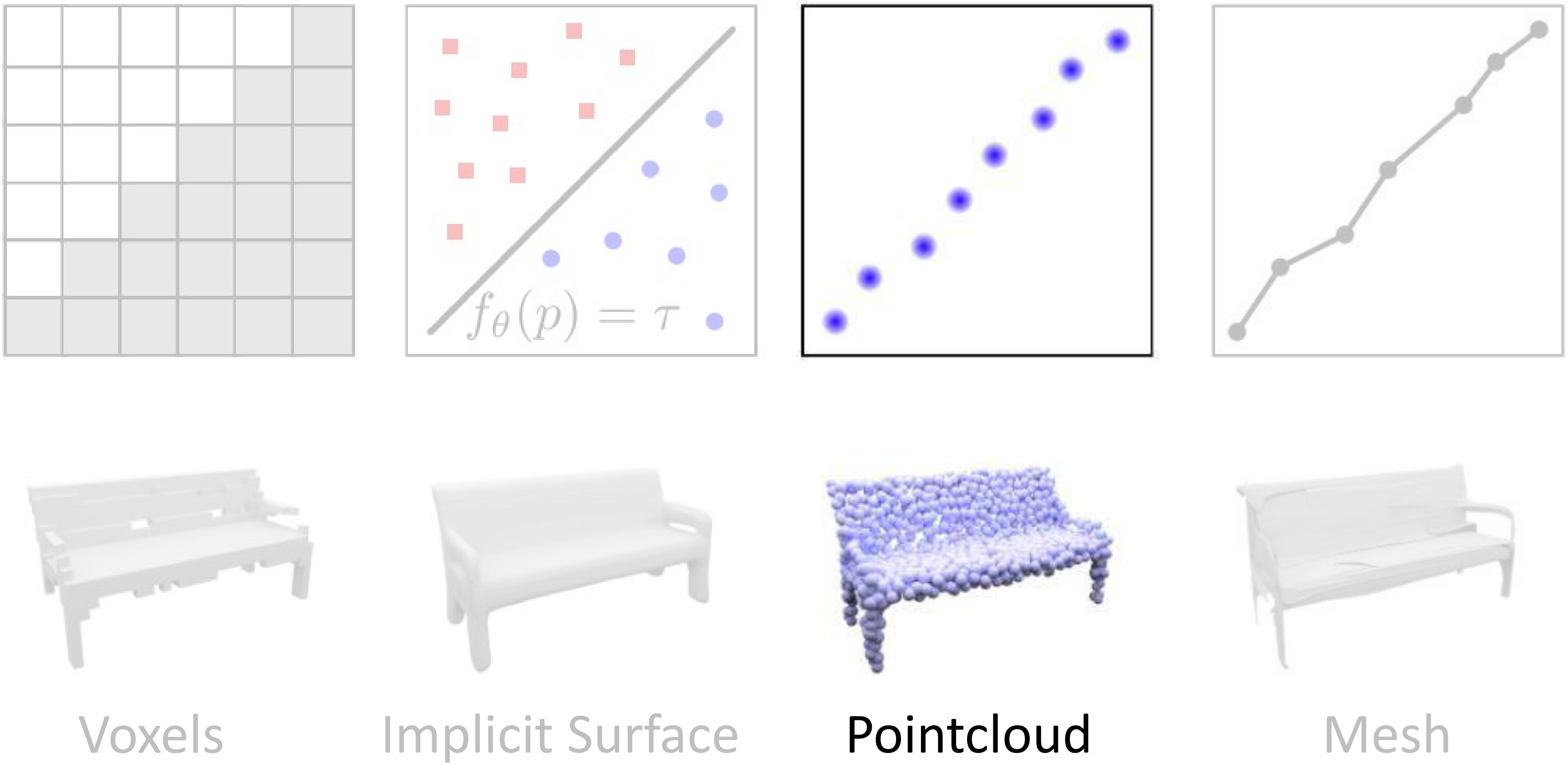
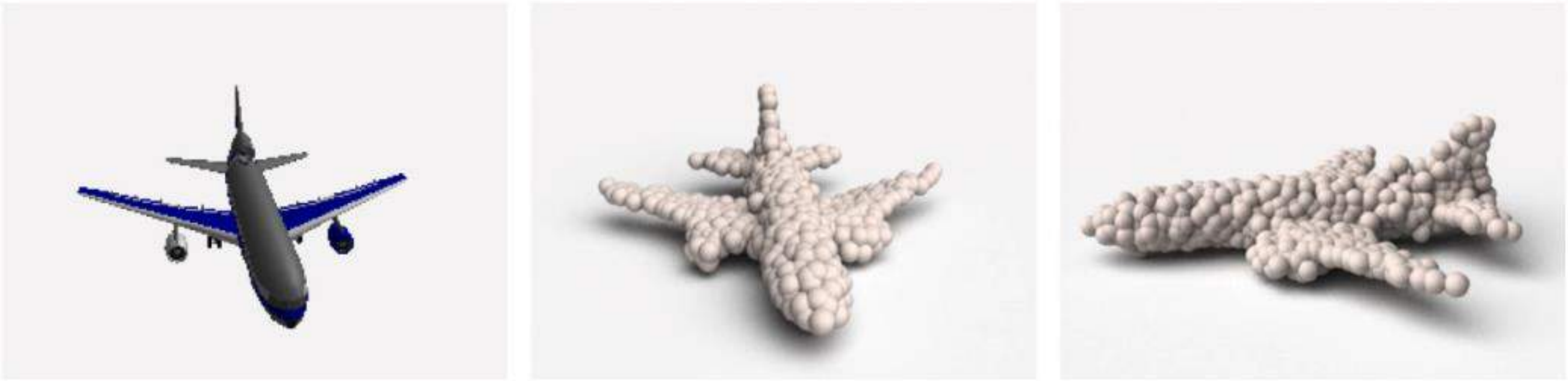


Figure credit: Mescheder et al, "Occupancy Networks: Learning 3D Reconstruction in Function Space", CVPR 2019

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



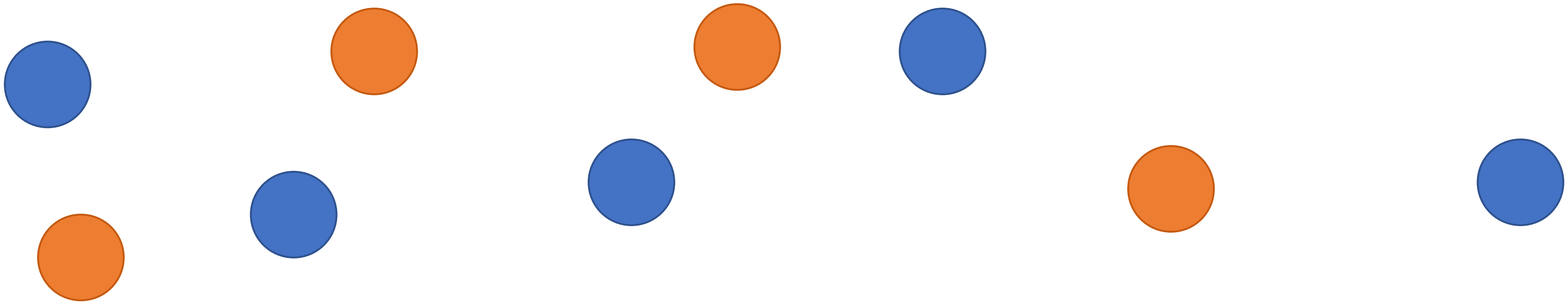
Fan et al, “A Point Set Generation Network for 3D Object Reconstruction from a Single Image”, CVPR 2017

Predicting Point Clouds: Chamfer Distance

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

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



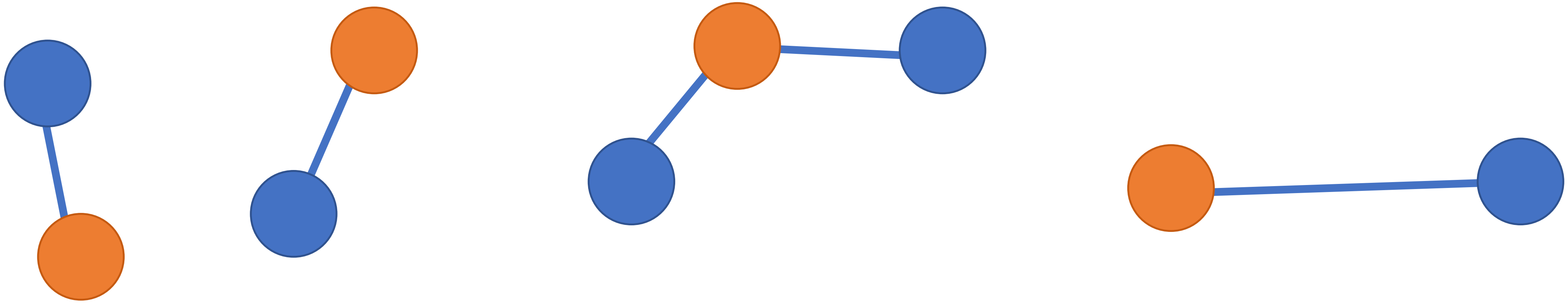
Fan et al, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image", CVPR 2017

Predicting Point Clouds: Chamfer Distance

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

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



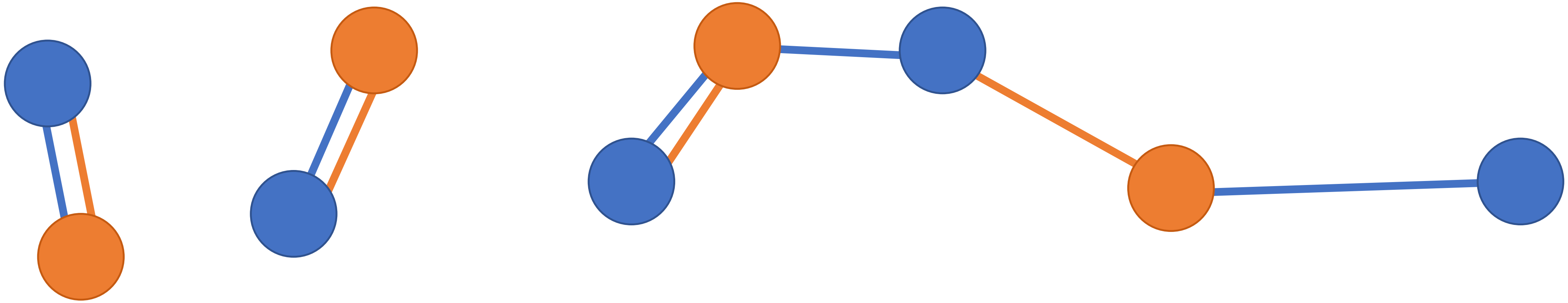
Fan et al, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image", CVPR 2017

Predicting Point Clouds: Chamfer Distance

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

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

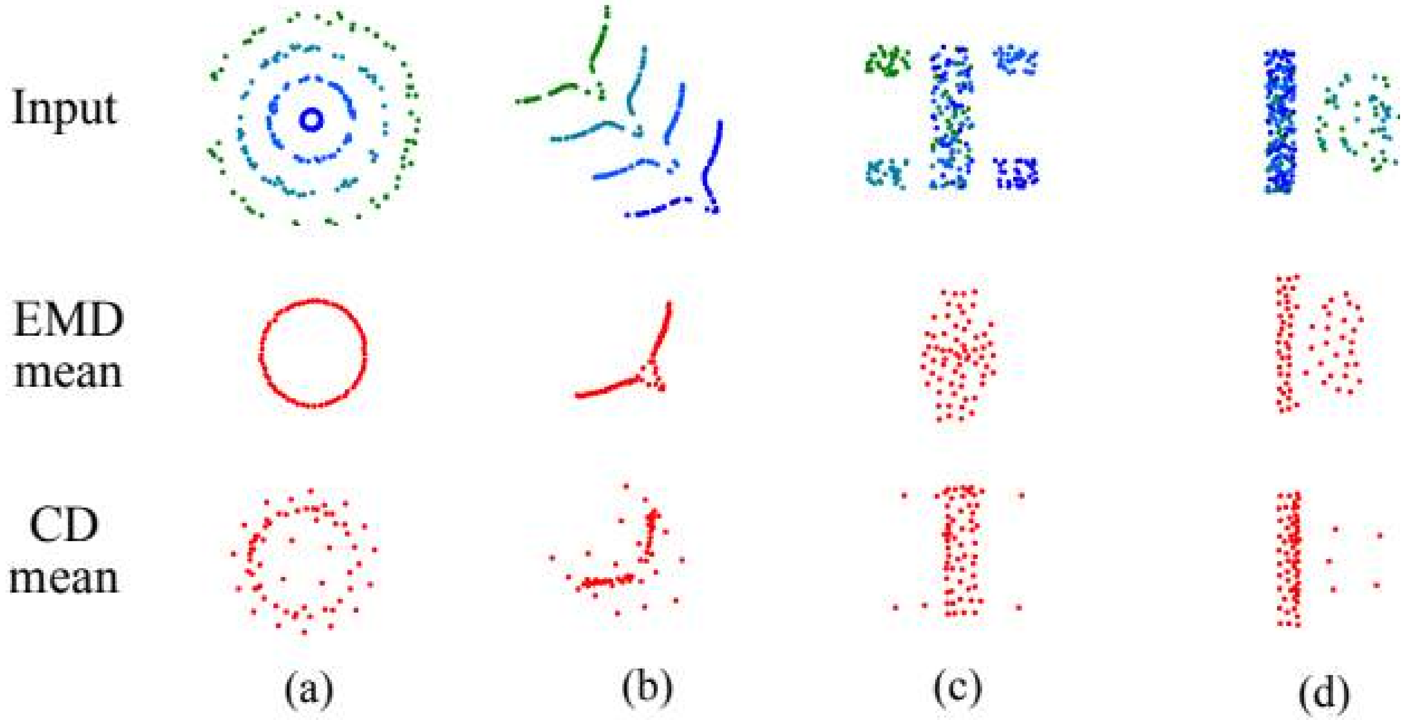


Fan et al, "A Point Set Generation Network for 3D Object Reconstruction from a Single Image", CVPR 2017

Predicting Point Clouds: Chamfer Distance

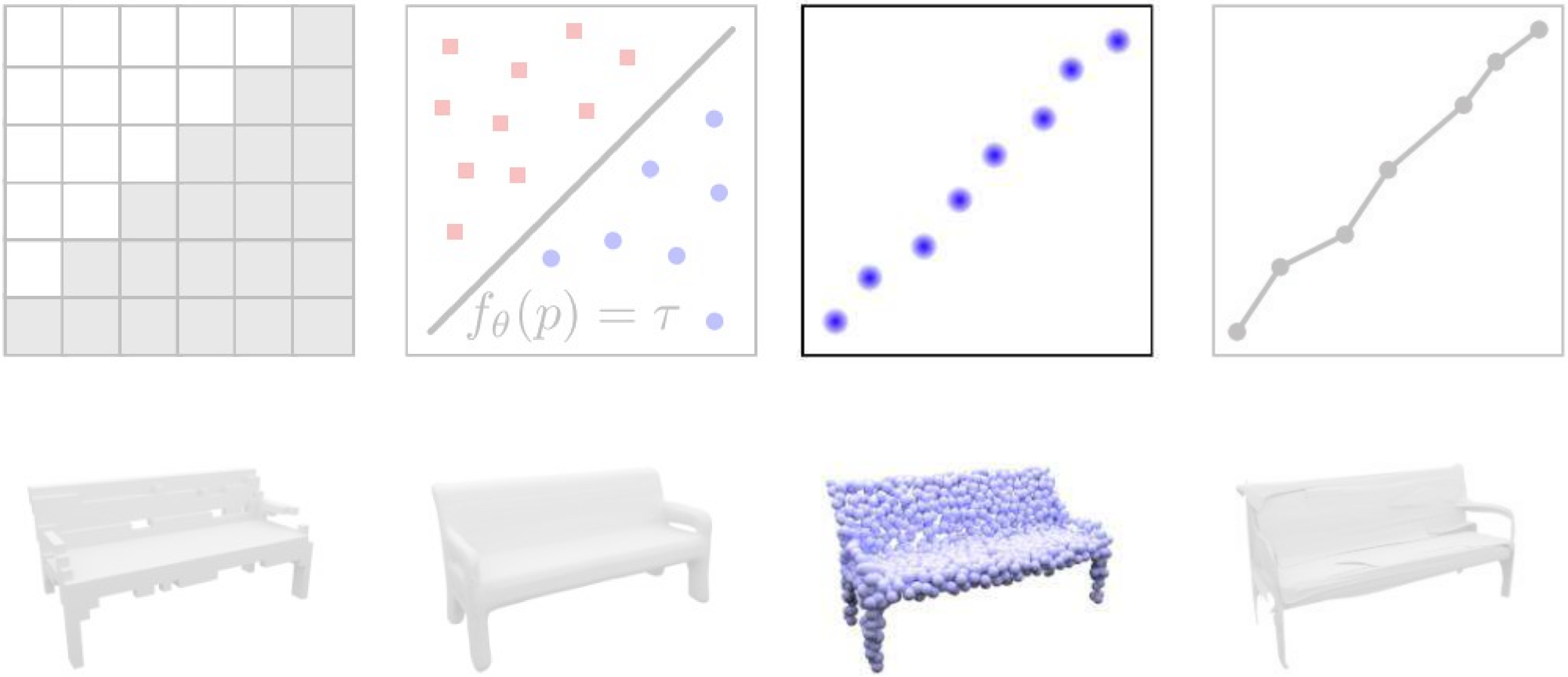
We need a (differentiable) way to compare pointclouds **as sets**!

- **Earth Mover's distance** $$d_{EMD}(S_1, S_2) = \min_{\phi: S_1 \to S_2} \sum_{x \in S_1} \|x - \phi(x)\|_2$$



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

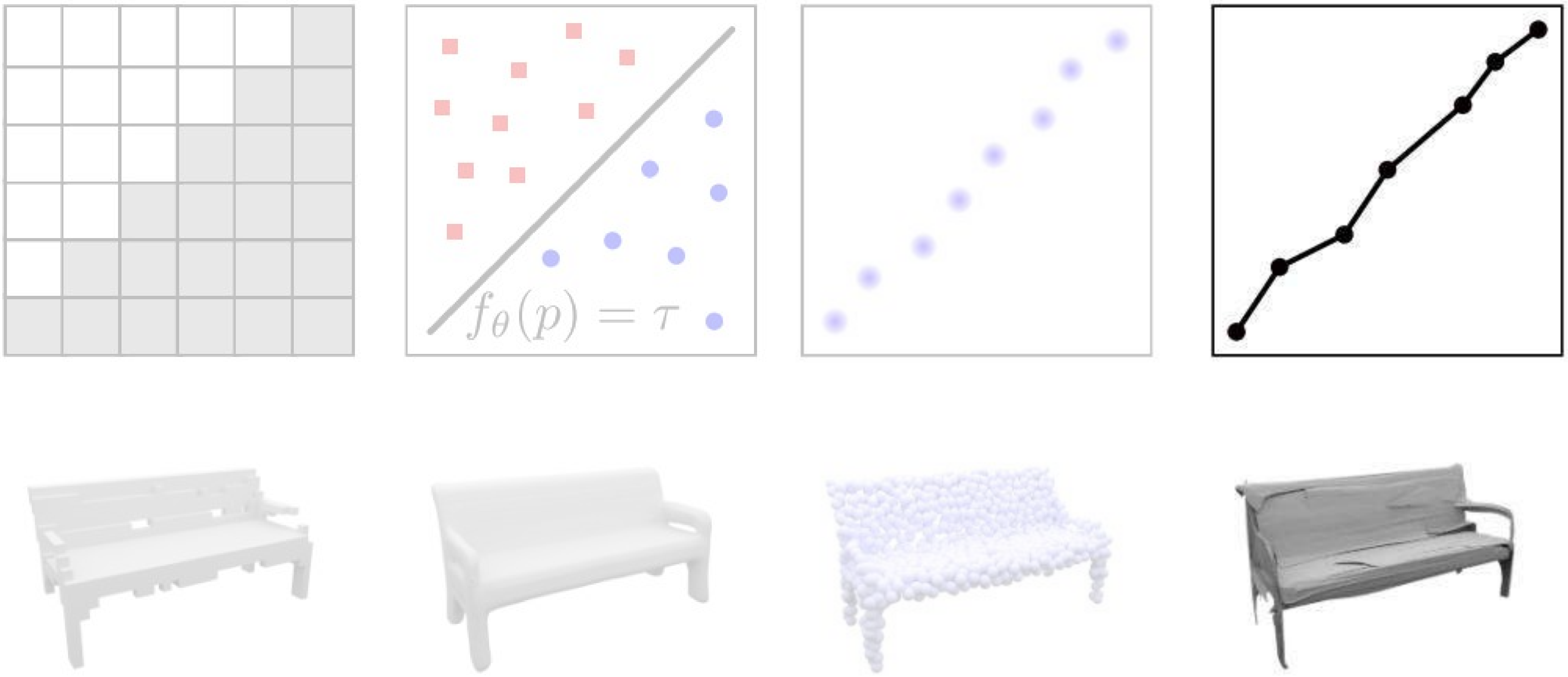
3D Shape Representations



Voxels | Implicit Surface | Pointcloud | Mesh

Figure credit: Mescheder et al, “Occupancy Networks: Learning 3D Reconstruction in Function Space”, CVPR 2019

3D Shape Representations



Voxels Implicit Surface Pointcloud **Mesh**

Figure credit: Mescheder et al, "Occupancy Networks: Learning 3D Reconstruction in Function Space", CVPR 2019

3D Shape Representations: Triangle Mesh

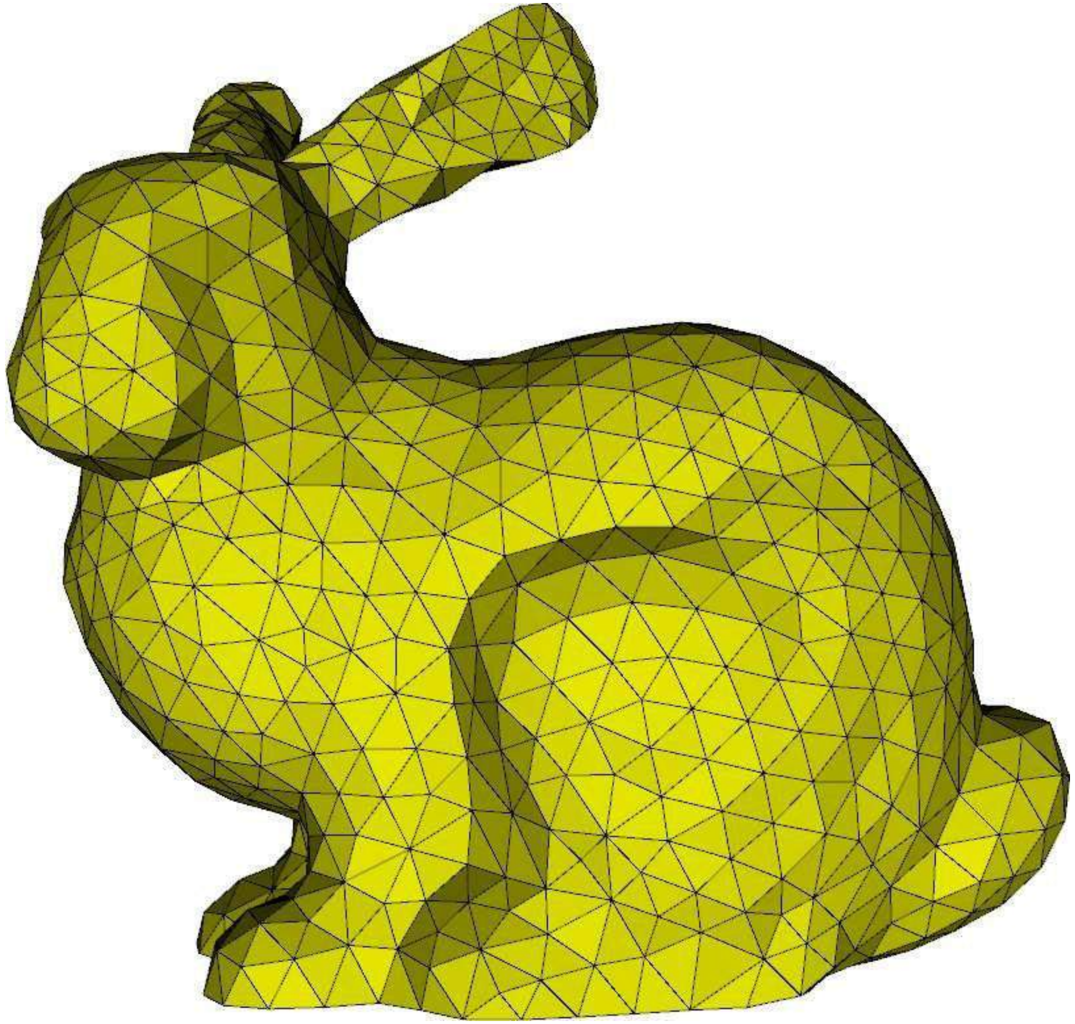
Represent a 3D shape as a set of triangles

Vertices: V x 3 matrix giving real-valued positions in 3D space

Faces: F x 3 matrix giving F triangles, each point specified as an index into 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: V x 3 matrix giving real-valued positions in 3D space

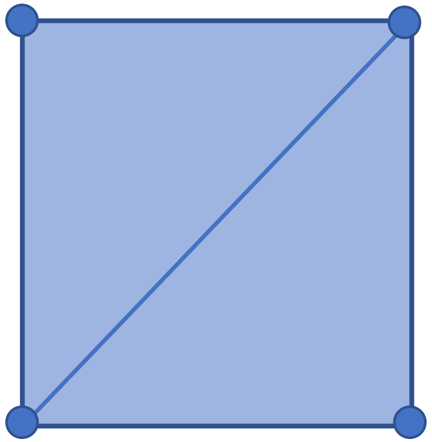
Faces: F x 3 matrix giving F triangles, each point specified as an index into the vertices

(+) Standard representation for graphics

(+) Explicitly represents 3D shapes

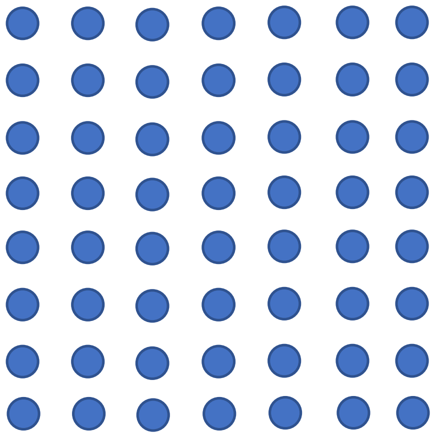
(+) Adaptive: Can represent flat surfaces very efficiently, can allocate more faces to areas with fine detail

Mesh



4 verts, 2 faces

Pointcloud



56 points

3D Shape Representations: Triangle Mesh

Represent a 3D shape as a set of triangles

Vertices: V x 3 matrix giving real-valued positions in 3D space

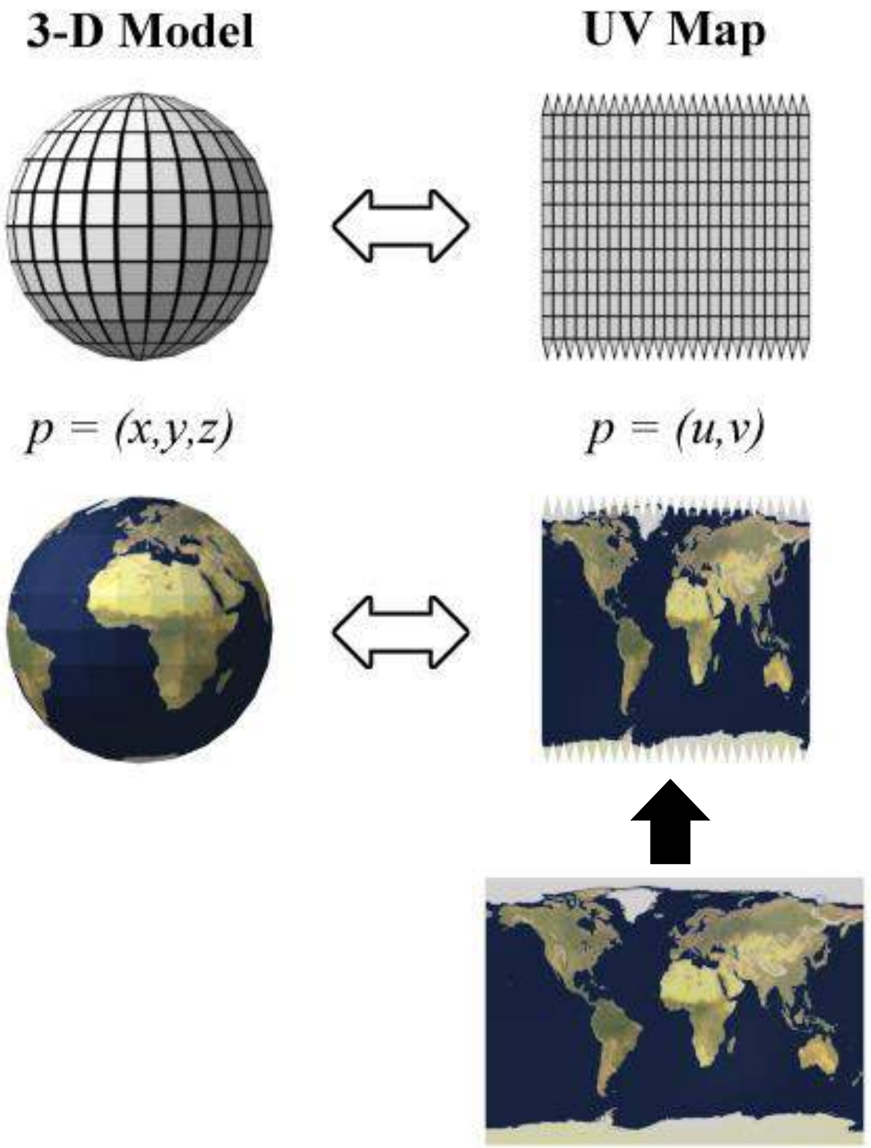
Faces: F x 3 matrix giving F triangles, each point specified as an index into the vertices

(+) 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.



3D Shape Representations: Triangle Mesh

Represent a 3D shape as a set of triangles

Vertices: V x 3 matrix giving real-valued positions in 3D space

Faces: F x 3 matrix giving F triangles, each point specified as an index into the vertices

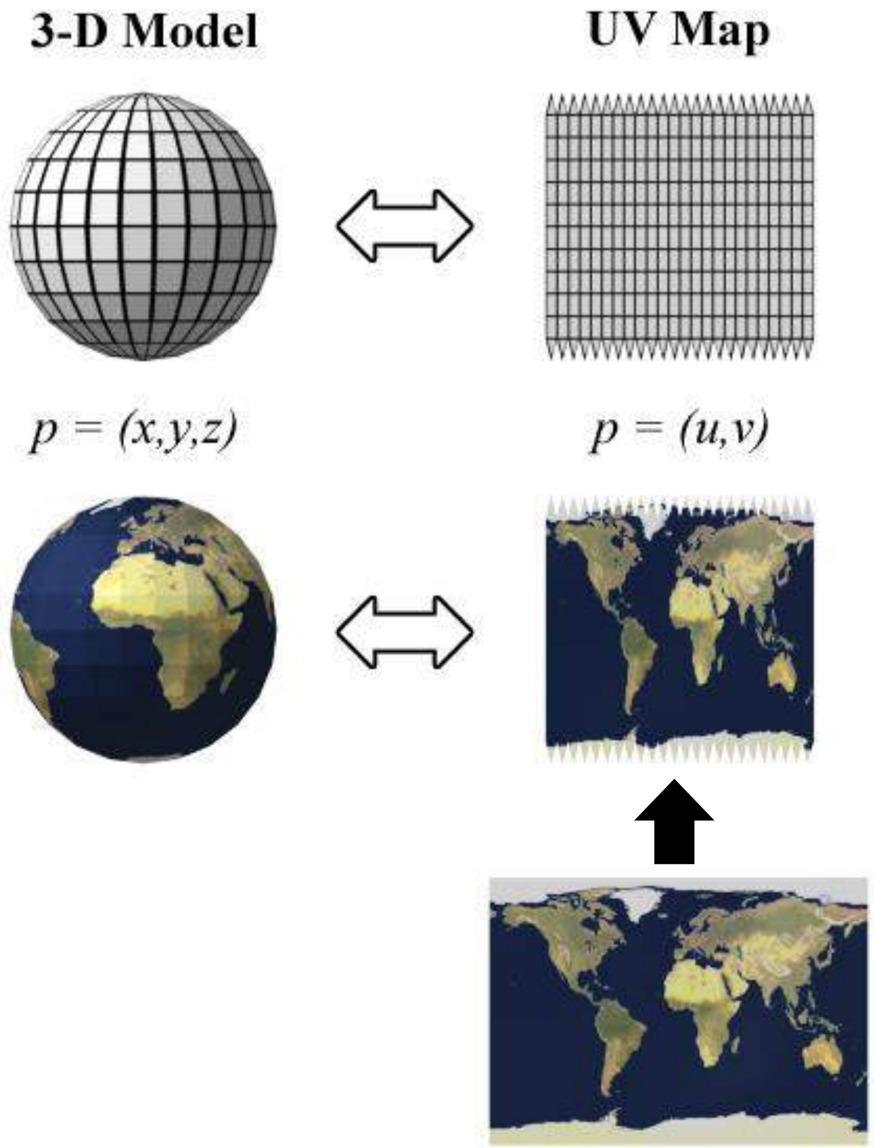
(+) Standard representation for graphics

(+) 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.

(-) Nontrivial to process with neural nets!



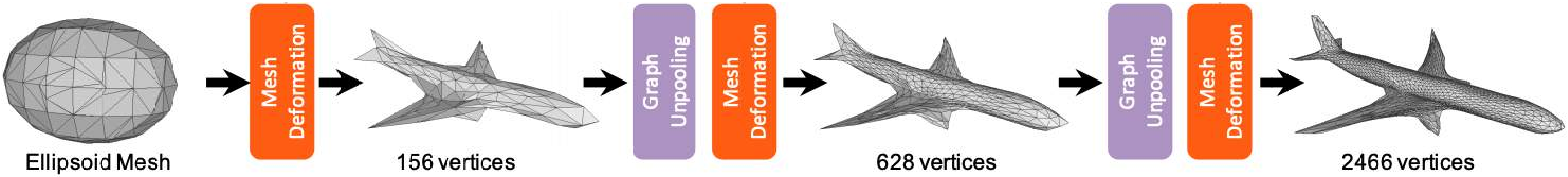
Predicting Meshes: Pixel2Mesh

Idea #1: Iterative mesh refinement

Start from initial ellipsoid mesh

Network predicts offsets for each vertex

Repeat.



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

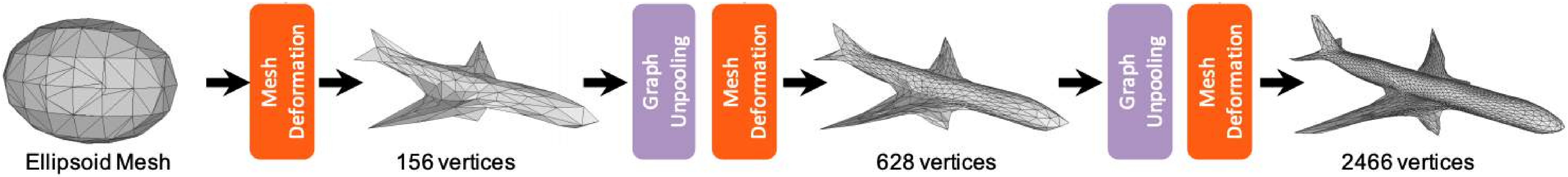
Predicting Meshes: Pixel2Mesh

Idea #1: Iterative mesh refinement

Start from initial ellipsoid mesh

Network predicts offsets for each vertex

Repeat.



Problem: Can’t model objects with holes!

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

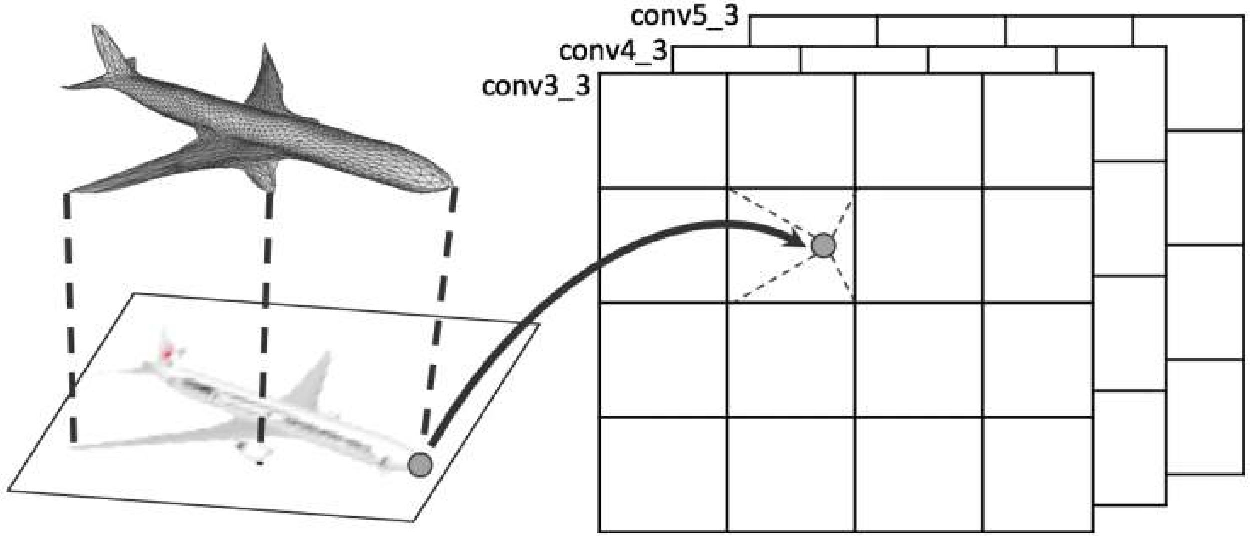
Predicting Meshes: Pixel2Mesh

Idea #2: Aligned vertex features

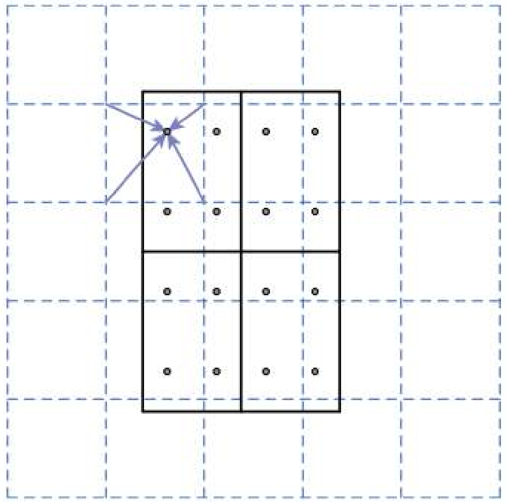
For each vertex of the mesh:

- Use camera intrinsics to project onto image plane
- Interpolate a feature from a backbone CNN

Similar to RoI-Align from Mask R-CNN: maintain alignment between image features and predictions



RoI-Align



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

Predicting Meshes: Pixel2Mesh

Idea #3: Graph Convolution

Maintain a feature vector f_i for each vertex v_i

Graph convolution computes new vectors for each vertex, propagating information along edges of the mesh

By sharing weights over all local neighborhoods, a graph conv layer can process meshes of arbitrary topology

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

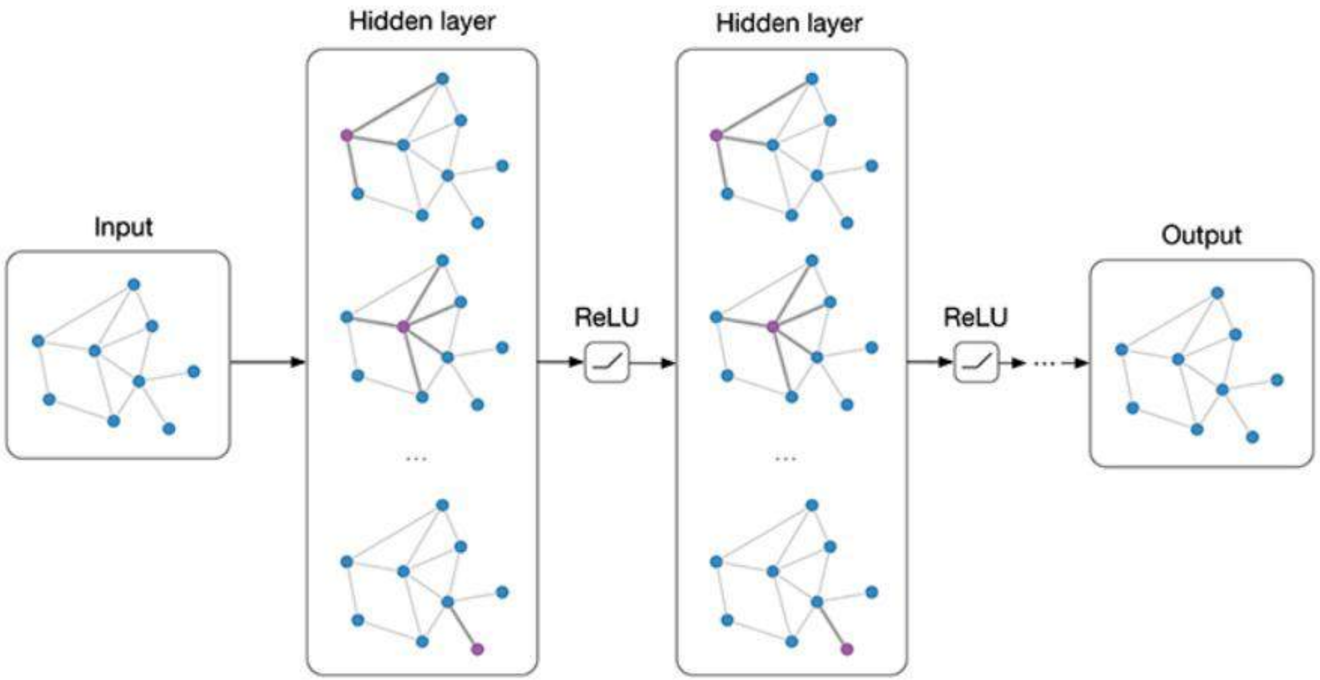
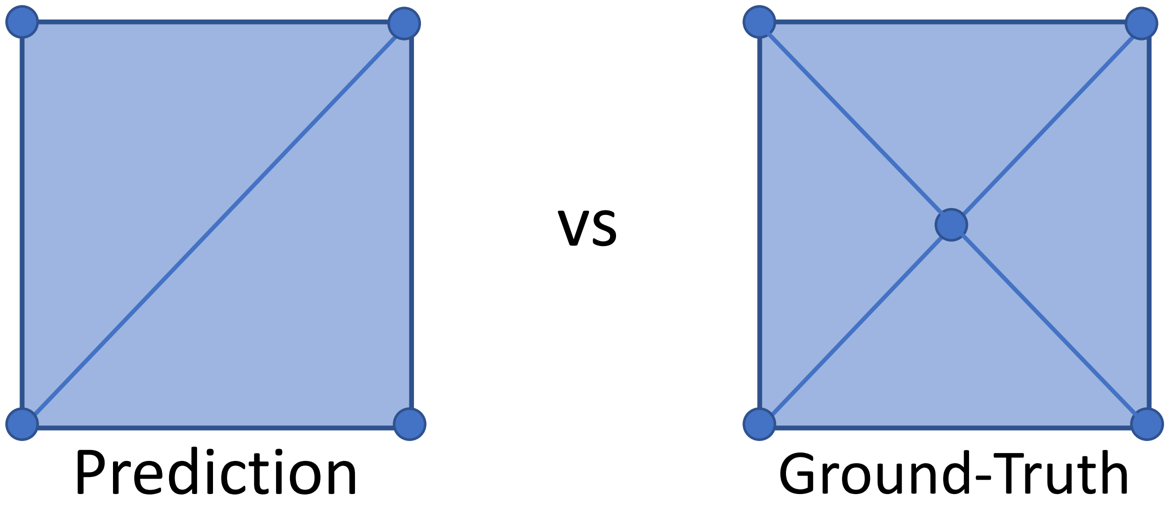


Figure credit: Thomas Kipf, https://tkipf.github.io/graph-convolutional-networks/

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?

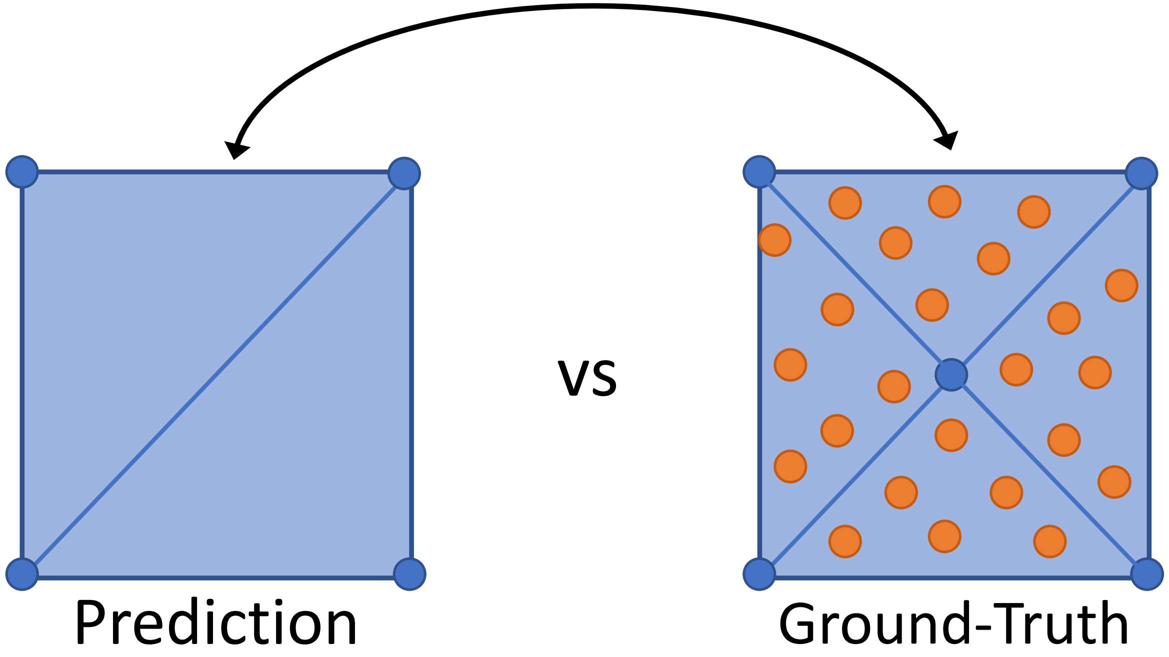


Wang et al, "Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images", ECCV 2018

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 verts and ground-truth samples



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

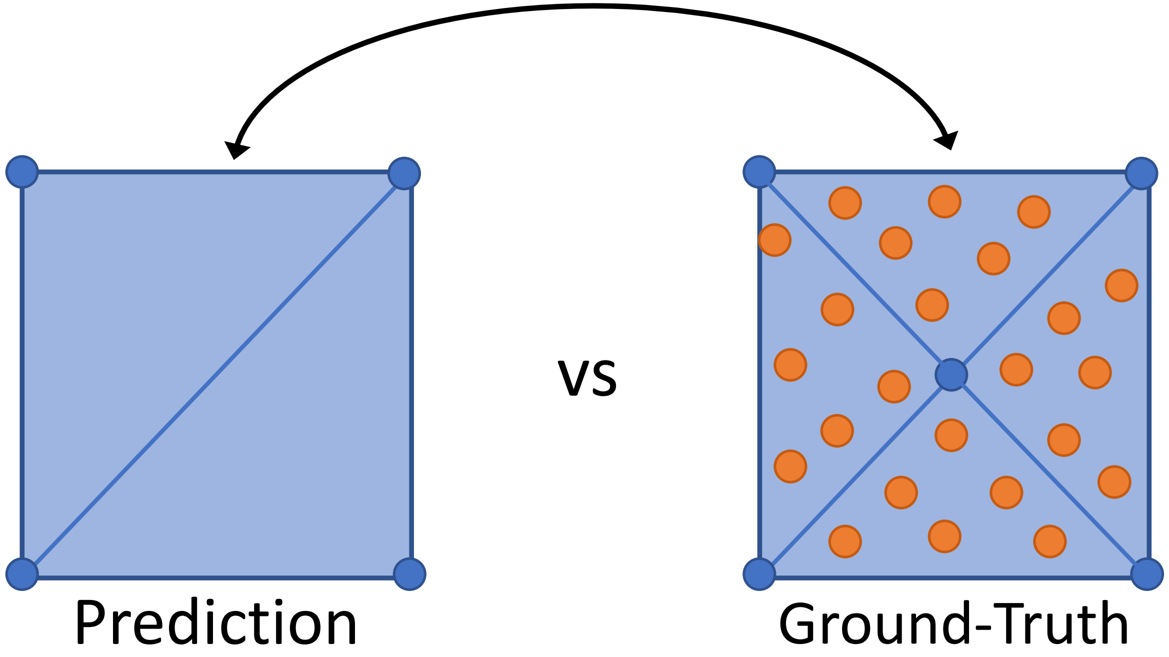
Wang et al, "Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images", ECCV 2018

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 verts and ground-truth samples

Problem: Doesn't take the interior of predicted faces into account!



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

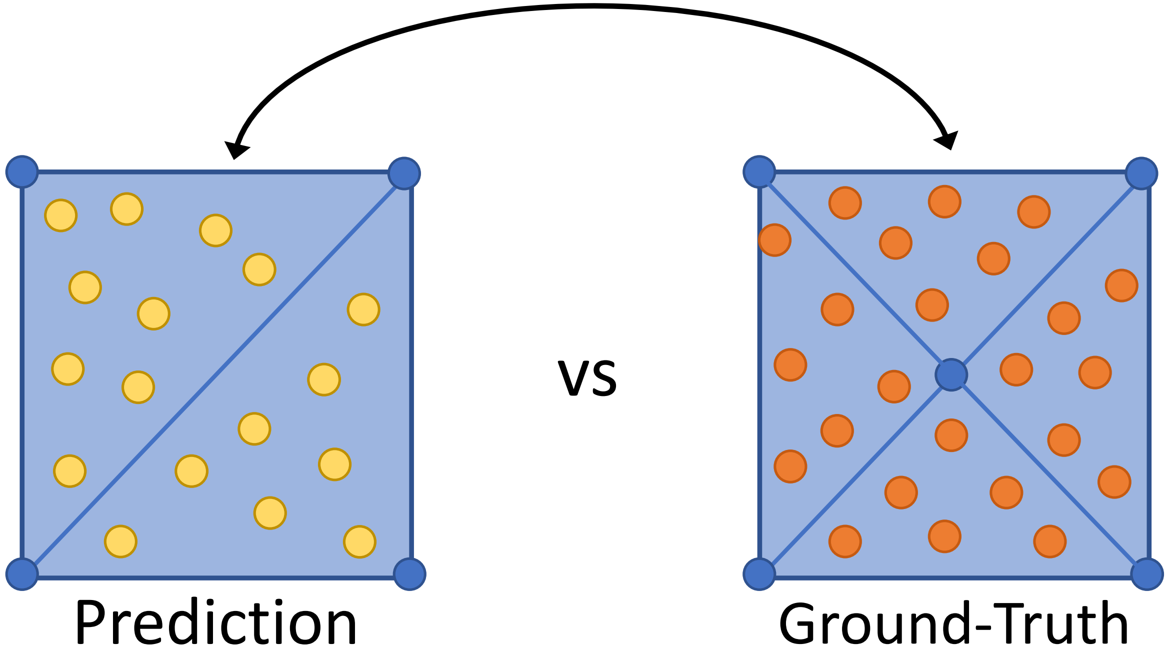
Wang et al, "Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images", ECCV 2018

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

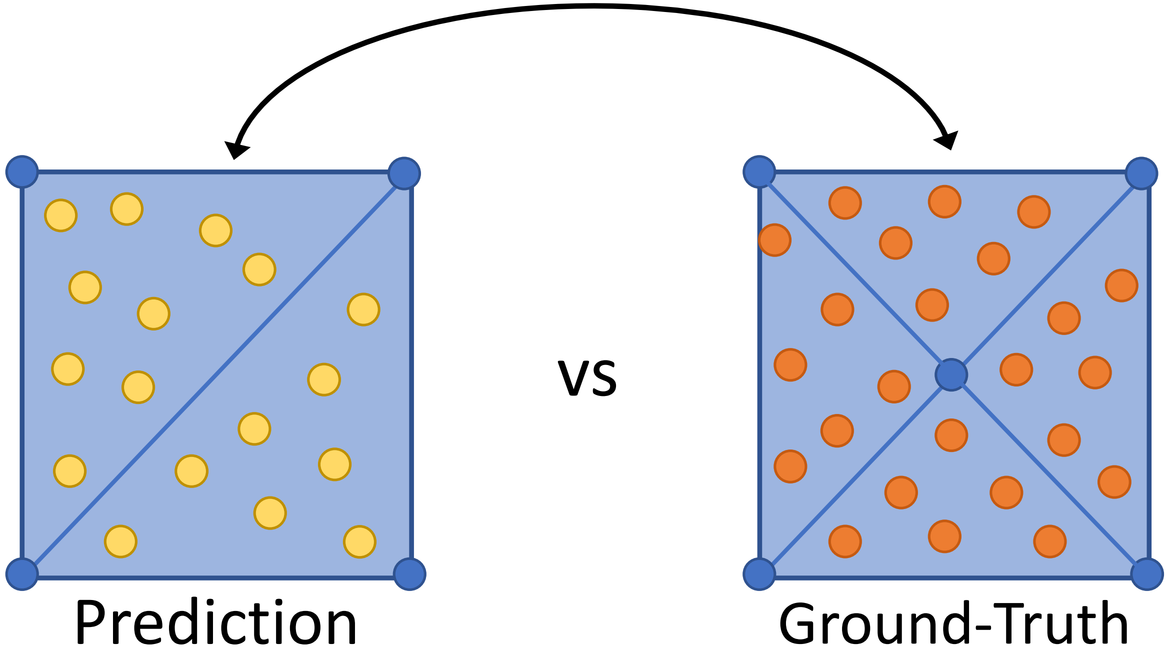
Predicting Meshes: Loss Function

Problem: Need to sample online! Must be efficient!

Problem: Need to backprop through sampling!

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

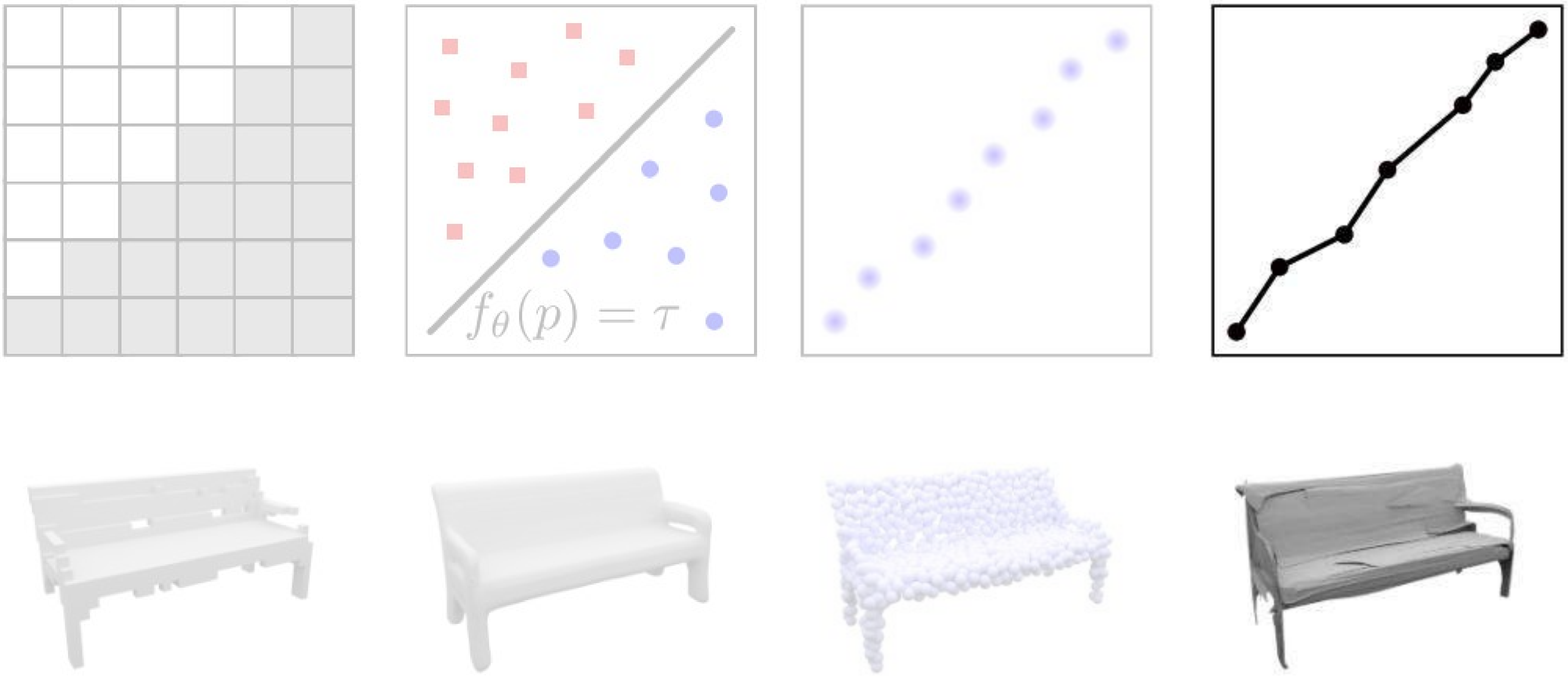
Sample points from the surface of the predicted mesh (online!)



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

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

3D Shape Representations



Voxels Implicit Surface Pointcloud Mesh

Figure credit: Mescheder et al, “Occupancy Networks: Learning 3D Reconstruction in Function Space”, CVPR 2019

Predicting 3D Shapes from 2D Images

Shape Representations

Metrics

Camera Systems

Datasets

Predicting 3D Shapes from 2D Images

Shape Representations

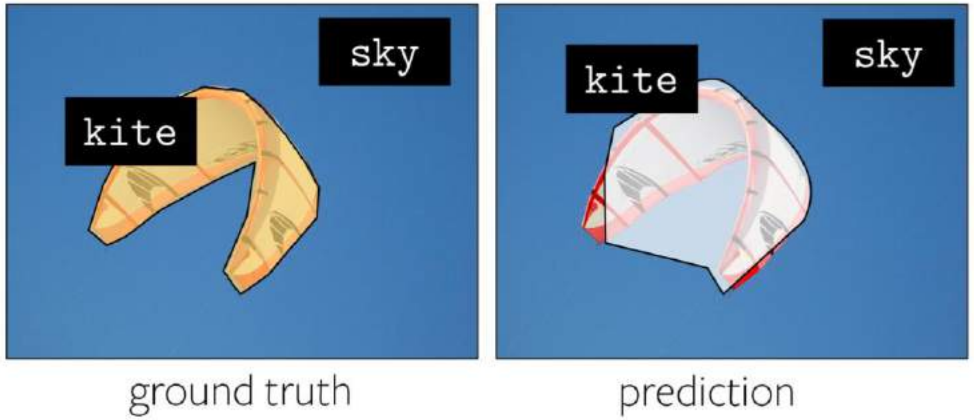
Metrics

Camera Systems

Datasets

Shape Comparison Metrics: Intersection over Union

In 2D, we evaluate segmentation masks with intersection over union (IoU): number of shared pixels over number of total pixels



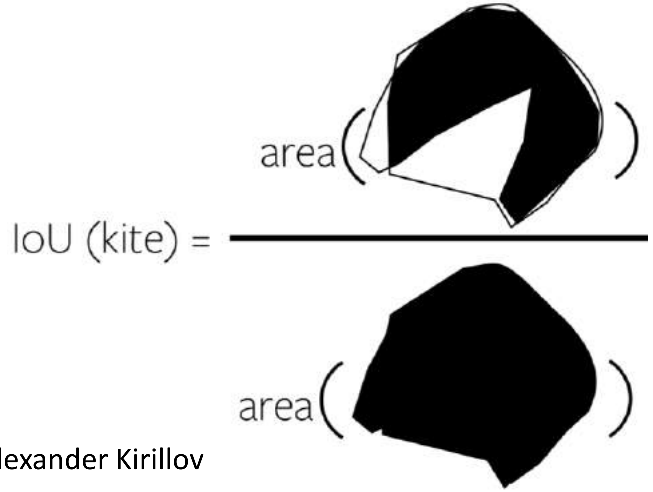
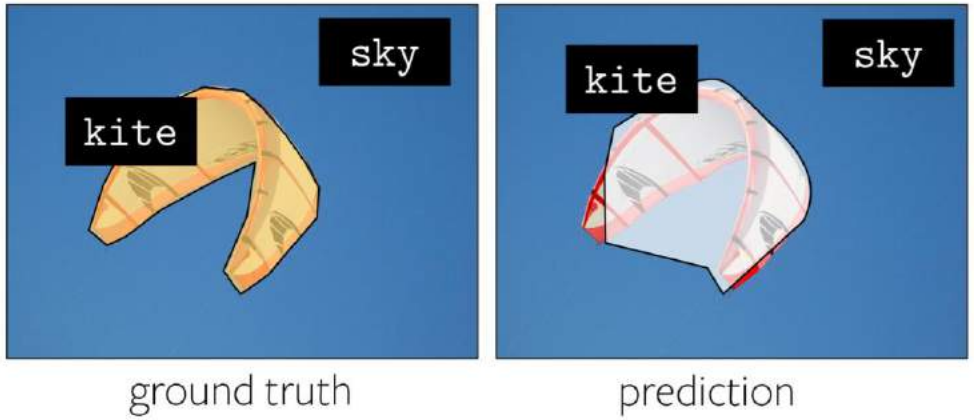


Figure credit: Alexander Kirillov

Shape Comparison Metrics: Intersection over Union

In 2D, we evaluate segmentation masks with intersection over union (IoU): number of shared pixels over number of total pixels



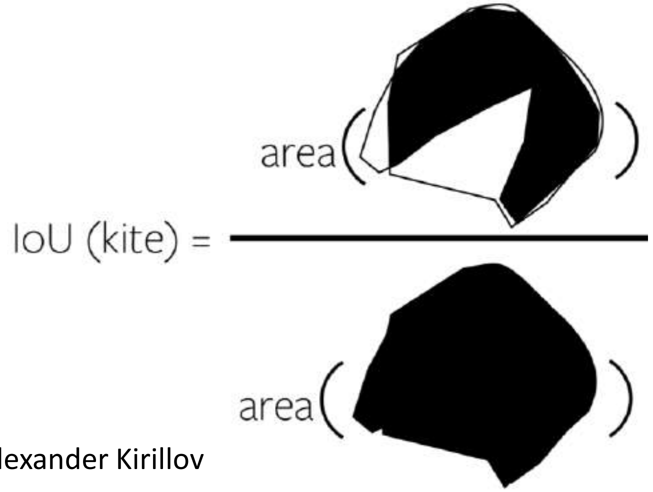


Figure credit: Alexander Kirillov

In 3D: **Voxel IoU**

Problem: Cannot capture thin structures
Problem: Cannot be applied to pointclouds
Problem: For meshes, need to voxelize or sample
Problem: Not very meaningful at low values!

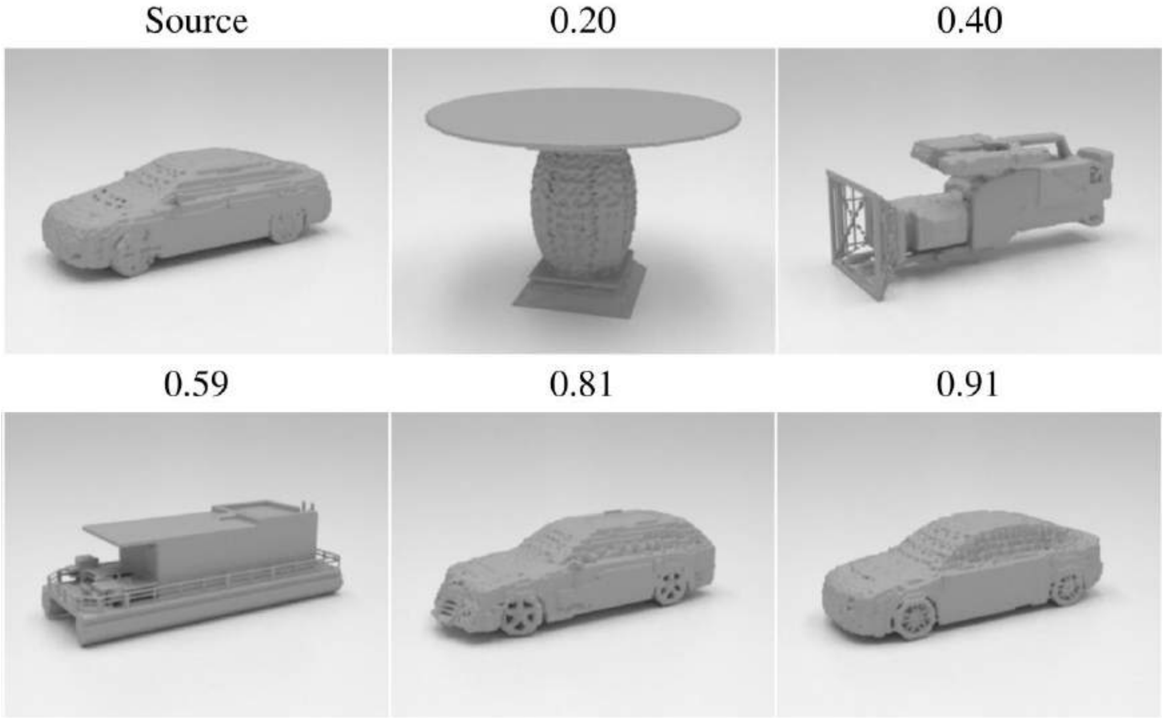
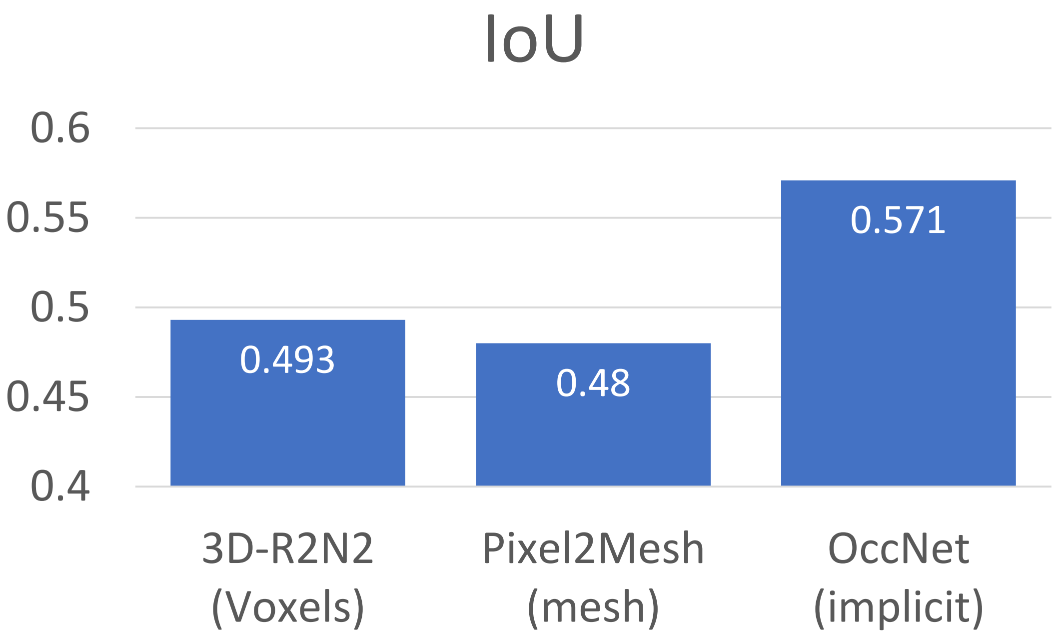


Figure credit: Tatarchenko et al, “What Do Single-view 3D Reconstruction Networks Learn?”, CVPR 2019

Shape Comparison Metrics: Intersection over Union

State–of-the-art methods achieve low IoU



Results from Mescheder et al, "Occupancy Networks: Learning 3D Reconstruction in Function Space", CVPR 2019

In 3D: **Voxel IoU**

Problem: Cannot capture thin structures
Problem: Cannot be applied to pointclouds
Problem: For meshes, need to voxelize or sample
Problem: Not very meaningful at low values!

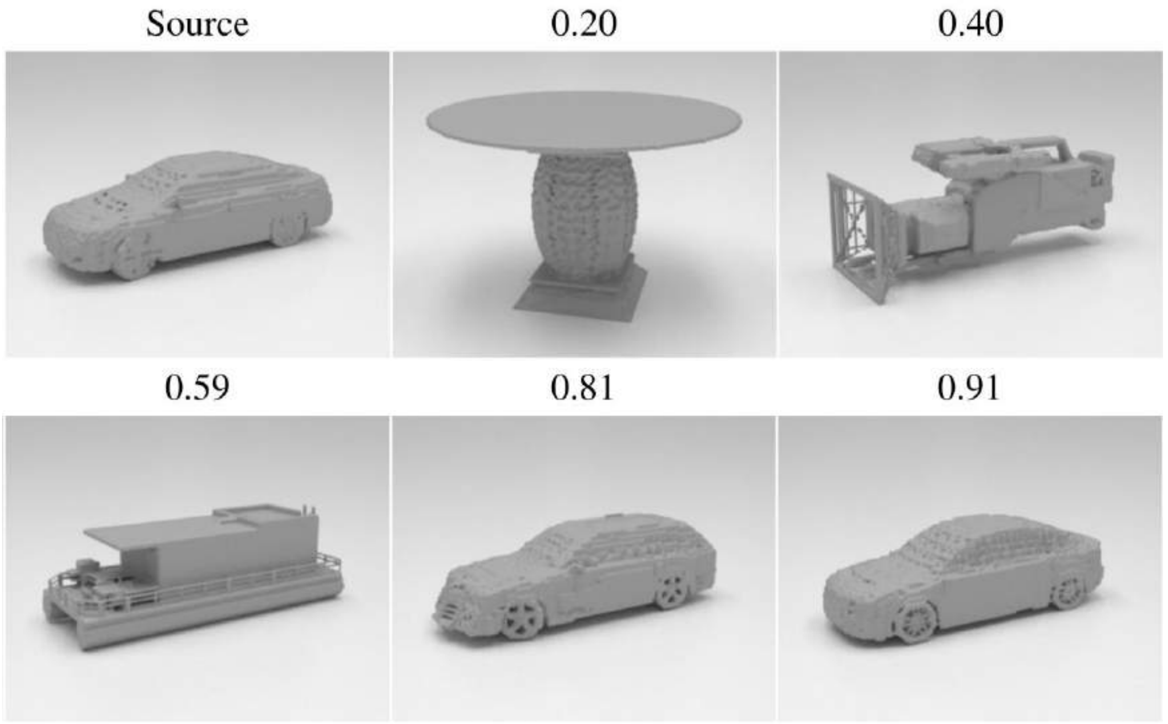
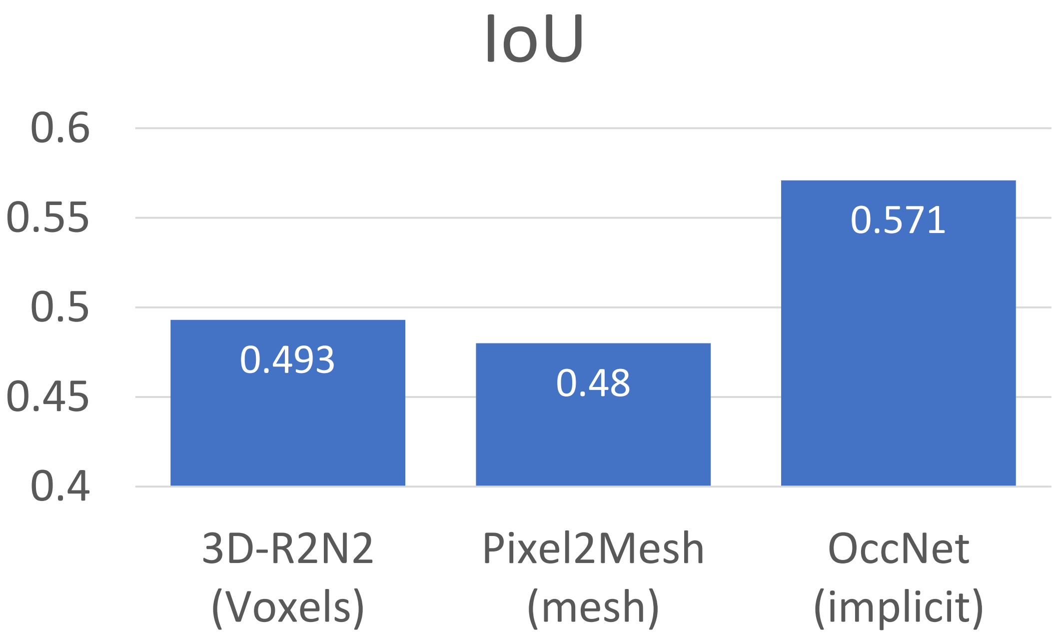


Figure credit: Tatarchenko et al, "What Do Single-view 3D Reconstruction Networks Learn?", CVPR 2019

Shape Comparison Metrics: Intersection over Union

State–of-the-art methods achieve low IoU



Results from Mescheder et al, “Occupancy Networks: Learning 3D Reconstruction in Function Space”, CVPR 2019

Conclusion: Voxel IoU not a good metric

In 3D: **Voxel IoU**

Problem: Cannot capture thin structures
Problem: Cannot be applied to pointclouds
Problem: For meshes, need to voxelize or sample
Problem: Not very meaningful at low values!

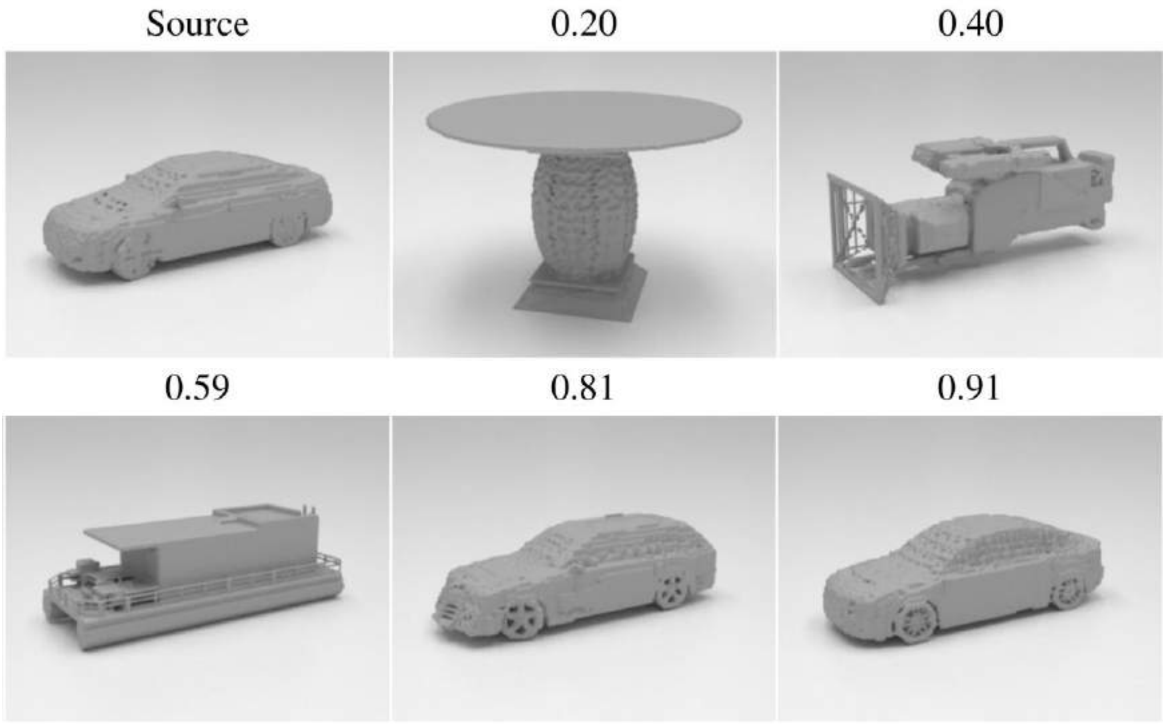


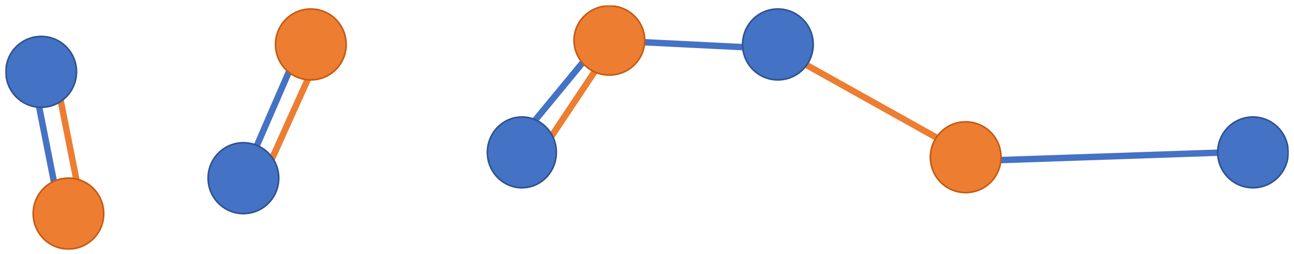
Figure credit: Tatarchenko et al, “What Do Single-view 3D Reconstruction Networks Learn?”, CVPR 2019

Shape Comparison Metrics: Chamfer Distance

We've already seen another shape comparison metric: **Chamfer distance**

1. Convert your prediction and ground-truth into pointclouds via sampling
2. Compare with Chamfer distance

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



Shape Comparison Metrics: Chamfer Distance

We've already seen another shape comparison metric: **Chamfer distance**

1. Convert your prediction and ground-truth into pointclouds via sampling
2. Compare with Chamfer distance

Problem: Chamfer is very sensitive to outliers

Compared to the source, both target shapes exhibit non-atching parts that are **equally wrong**!

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

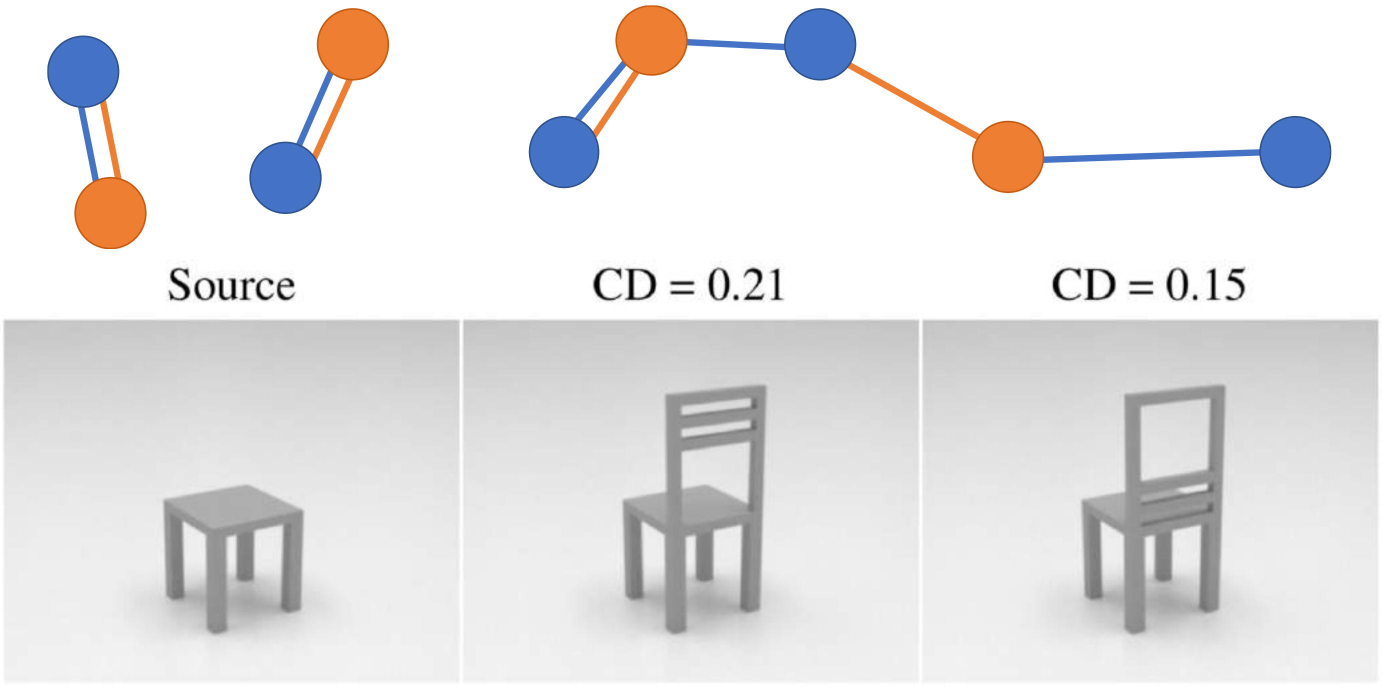
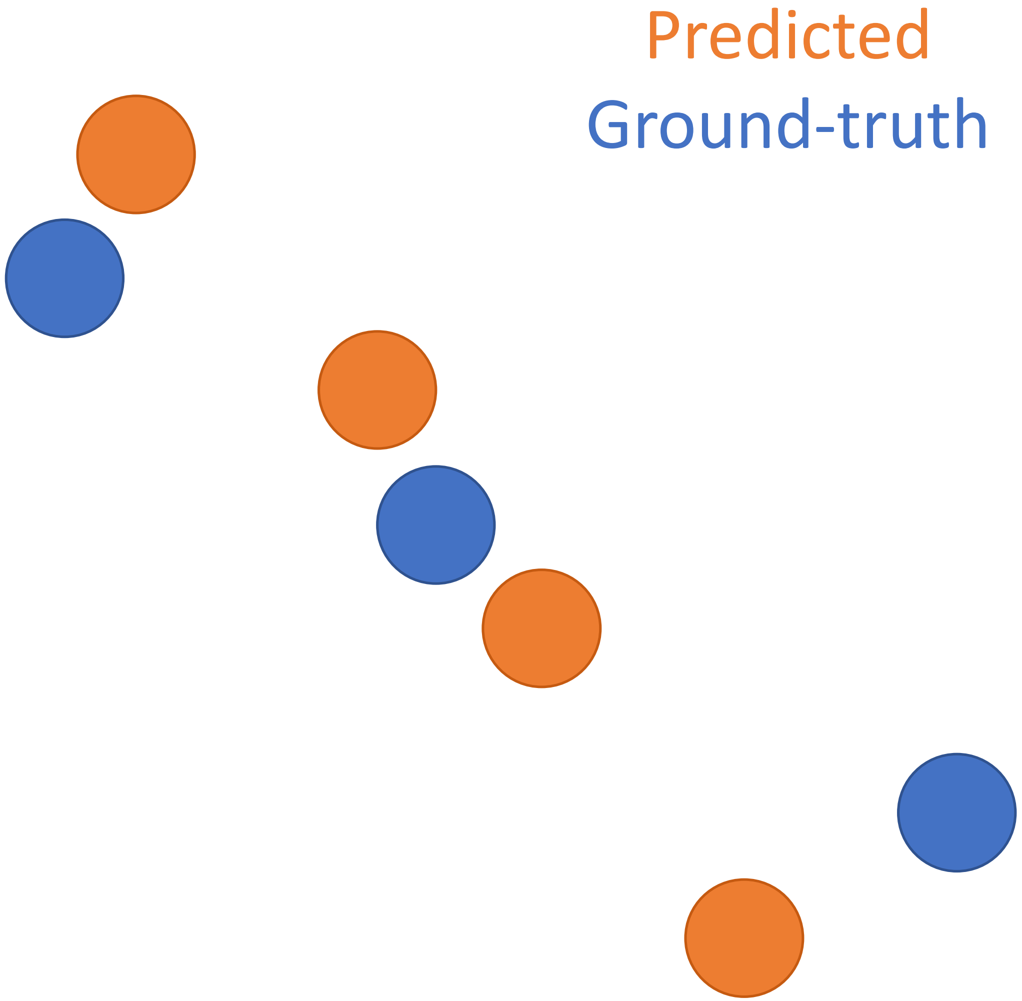


Figure credit: Tatarchenko et al, "What Do Single-view 3D Reconstruction Networks Learn?", CVPR 2019

Shape Comparison Metrics: F1 Score

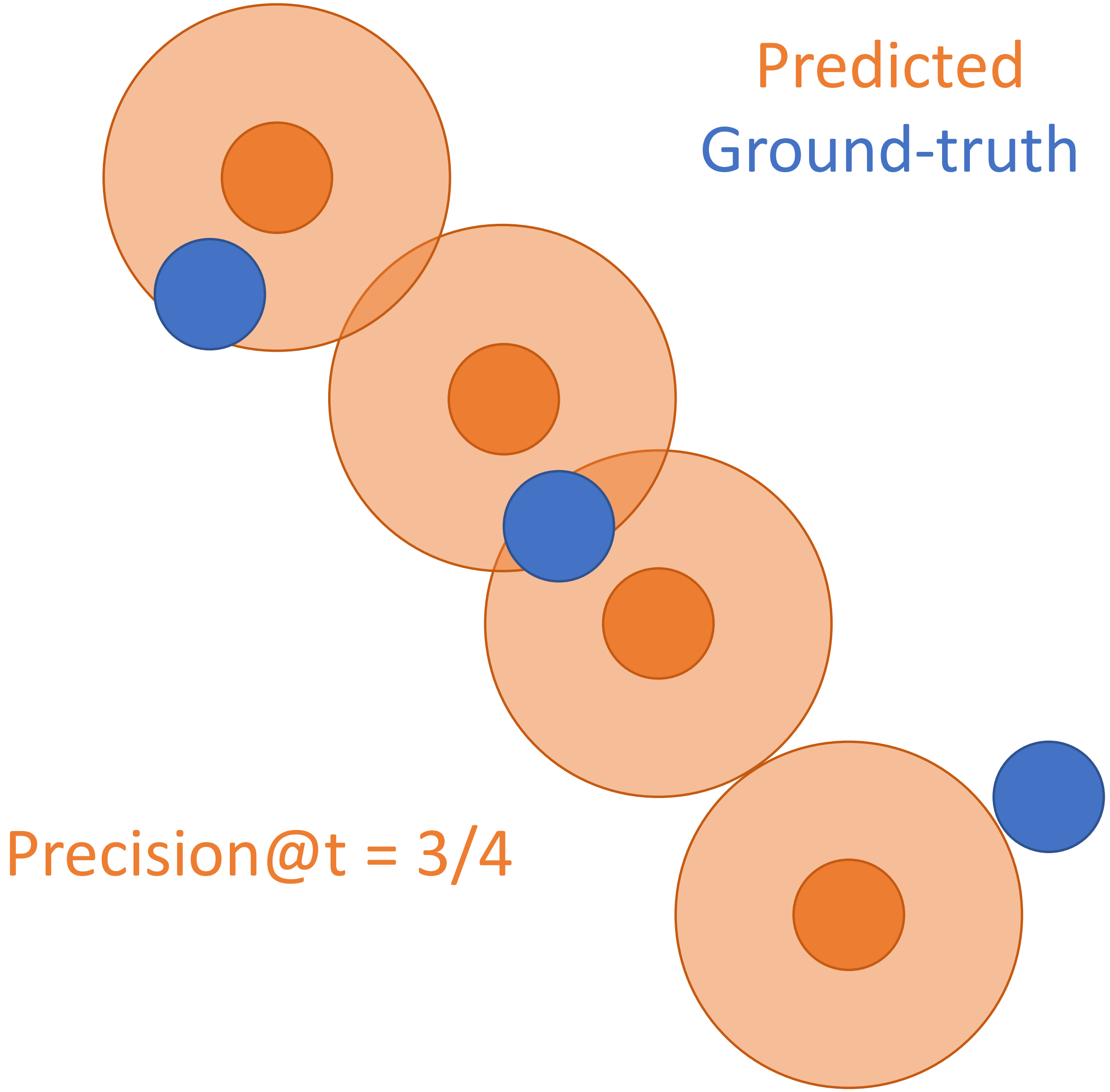
Similar to Chamfer, sample points from the surface of the prediction and the ground-truth



Shape Comparison Metrics: F1 Score

Similar to Chamfer, sample points from the surface of the prediction and the ground-truth

Precision@t = fraction of predicted points within t of some ground-truth point

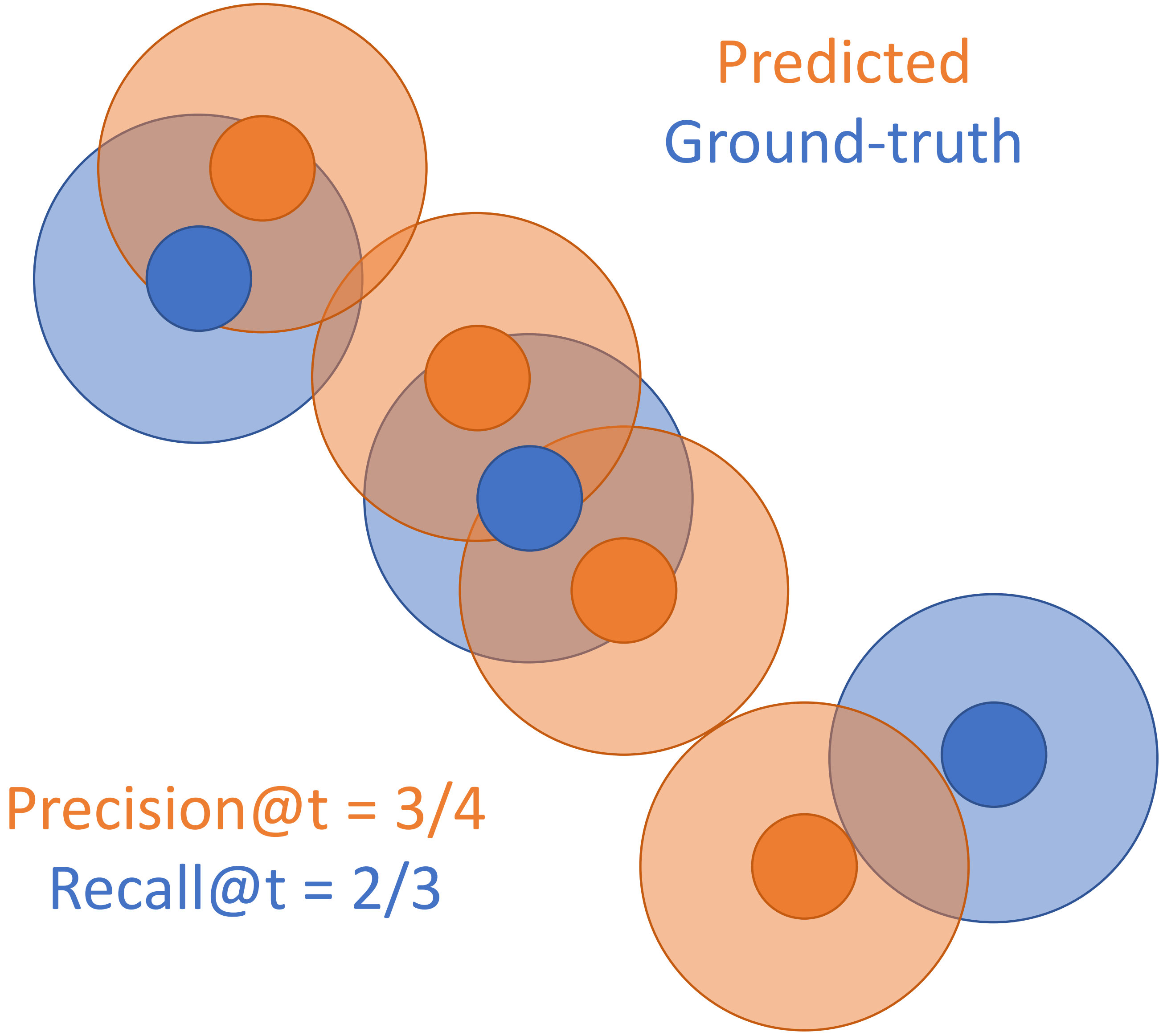


Shape Comparison Metrics: F1 Score

Similar to Chamfer, sample points from the surface of the prediction and the ground-truth

Precision@t = fraction of predicted points within t of some ground-truth point

Recall@t = fraction of ground-truth points within t of some predicted point



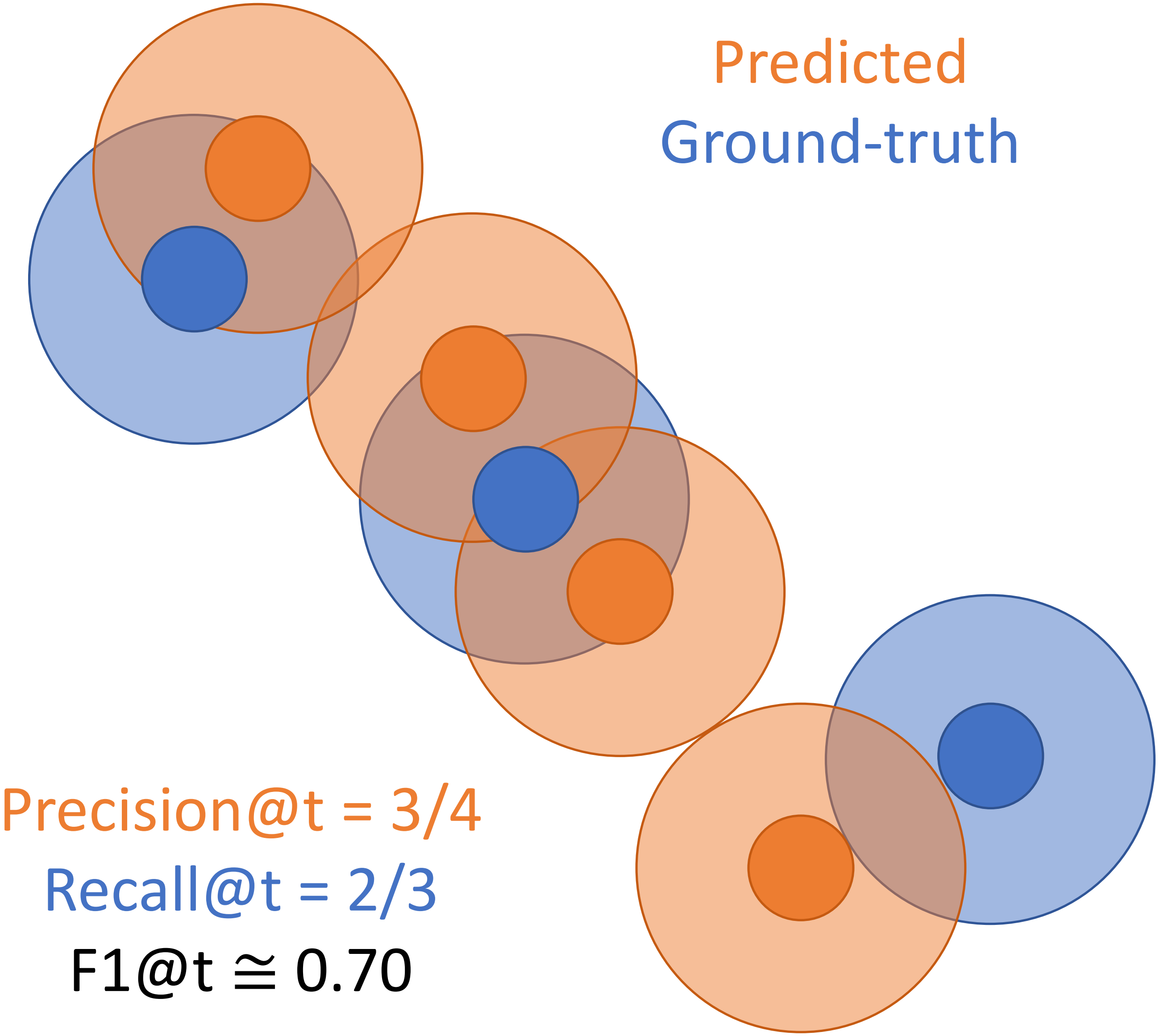
Shape Comparison Metrics: F1 Score

Similar to Chamfer, sample points from the surface of the prediction and the ground-truth

Precision@t = fraction of predicted points within t of some ground-truth point

Recall@t = fraction of ground-truth points within t of some predicted point

$$F1@t = 2 * \frac{Precision@t * Recall@t}{Precision@t + Recall@t}$$



Precision@t = 3/4

Recall@t = 2/3

F1@t ≅ 0.70

Shape Comparison Metrics: F1 Score

Similar to Chamfer, sample points from the surface of the prediction and the ground-truth

Precision@t = fraction of predicted points within t of some ground-truth point

Recall@t = fraction of ground-truth points within t of some predicted point

F1@t = $2 * \frac{Precision@t * Recall@t}{Precision@t + Recall@t}$

F1 score is robust to outliers!

Figure credit: Tatarchenko et al, “What Do Single-view 3D Reconstruction Networks Learn?”, CVPR 2019

Shape Comparison Metrics: F1 Score

Similar to Chamfer, sample points from the surface of the prediction and the ground-truth

Precision@t = fraction of predicted points within t of some ground-truth point

Recall@t = fraction of ground-truth points within t of some predicted point

F1@t = $2 * \frac{Precision@t * Recall@t}{Precision@t + Recall@t}$

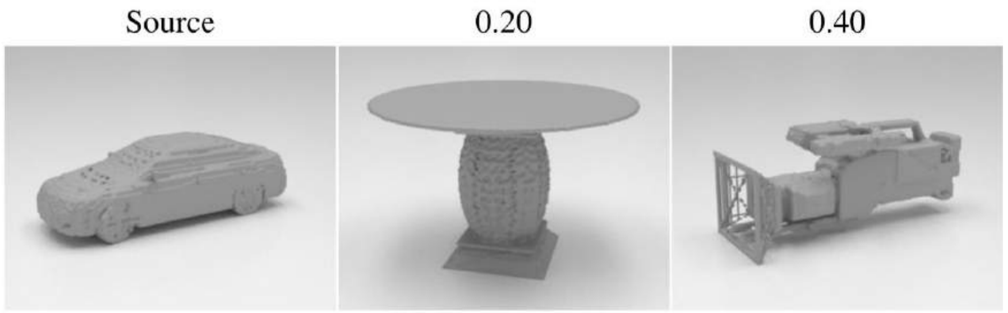
F1 score is robust to outliers!

Conclusion: F1 score is probably the best shape prediction metric in common use

Figure credit: Tatarchenko et al, "What Do Single-view 3D Reconstruction Networks Learn?", CVPR 2019

Shape Comparison Metrics Summary

Intersection over Union:
Doesn't capture fine structure,
not meaningful at low values

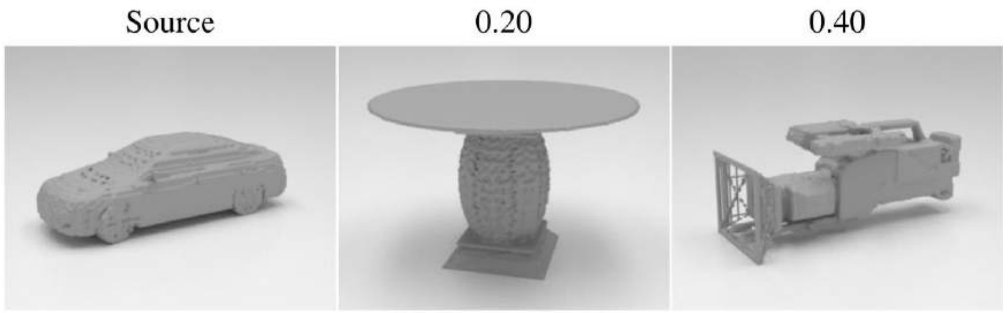


Chamfer Distance:
Very sensitive to outliers
Can be directly optimized

F1 score:
Robust to outliers, but need to
look at different threshold values
to capture details at different scales

Shape Comparison Metrics Summary

Intersection over Union:
Doesn't capture fine structure,
not meaningful at low values

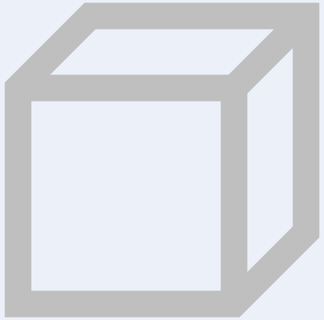


Chamfer Distance:
Very sensitive to outliers
Can be directly optimized

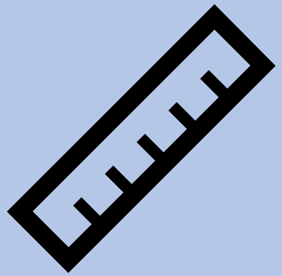
F1 score:
Robust to outliers, but need to
look at different threshold values
to capture details at different scales

Predicting 3D Shapes from 2D Images

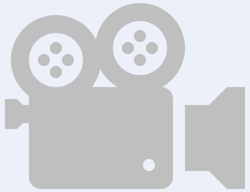
Shape Representations



Metrics



Camera Systems



Datasets

Predicting 3D Shapes from 2D Images

Shape Representations

Metrics

Camera Systems

Datasets

Cameras: Canonical vs View coordinates

Canonical Coordinates: Predict 3D shape in a canonical coordinate system (e.g. front of chair is +z) regardless of the viewpoint of the input image

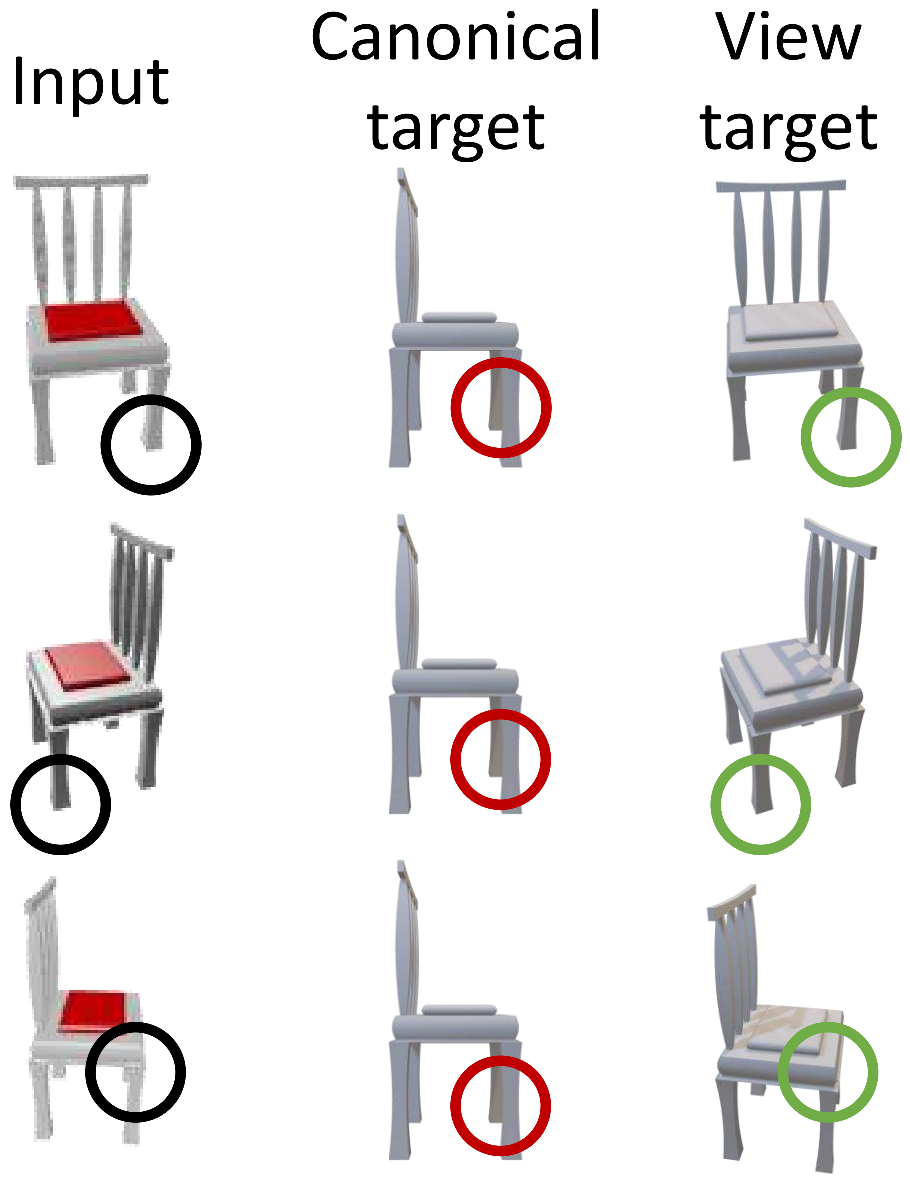
View Coordinates: Predict 3D shape aligned to the viewpoint of the camera

Many papers predict in canonical coordinates – easier to load data

Cameras: Canonical vs View coordinates

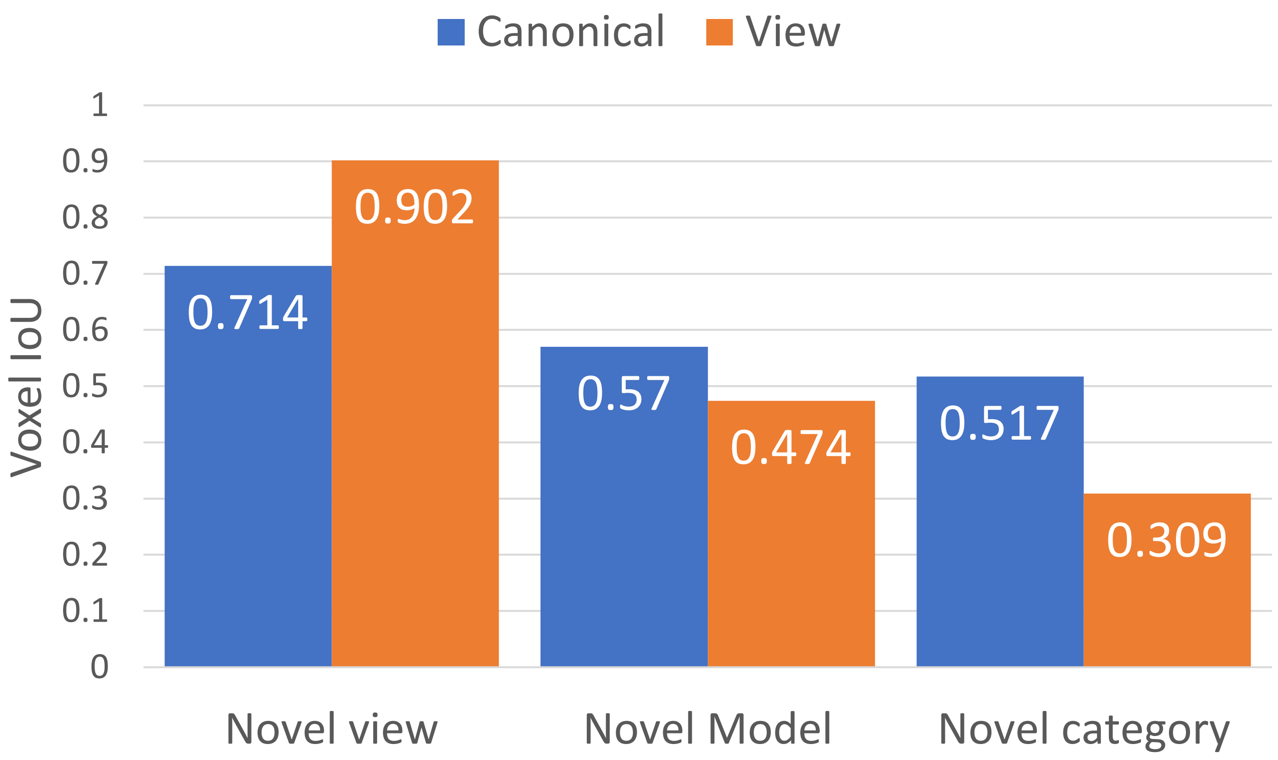
Problem: Canonical view breaks the "principle of feature alignment": Predictions should be aligned to inputs

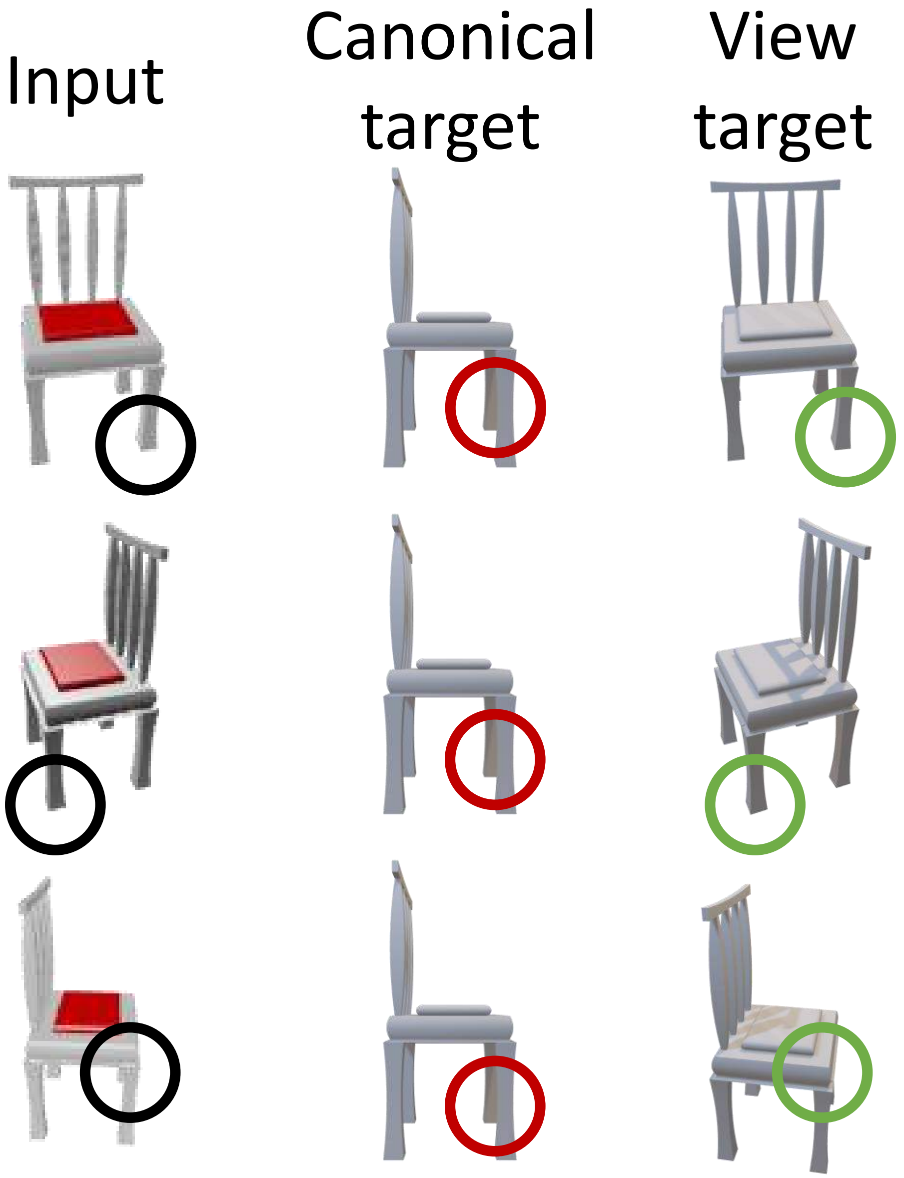
View coordinates maintain alignment between inputs and predictions!



Cameras: Canonical vs View coordinates

Problem: Canonical view overfits to training shapes:
Better generalization to new views of known shapes
Worse generalization to new shapes or new categories

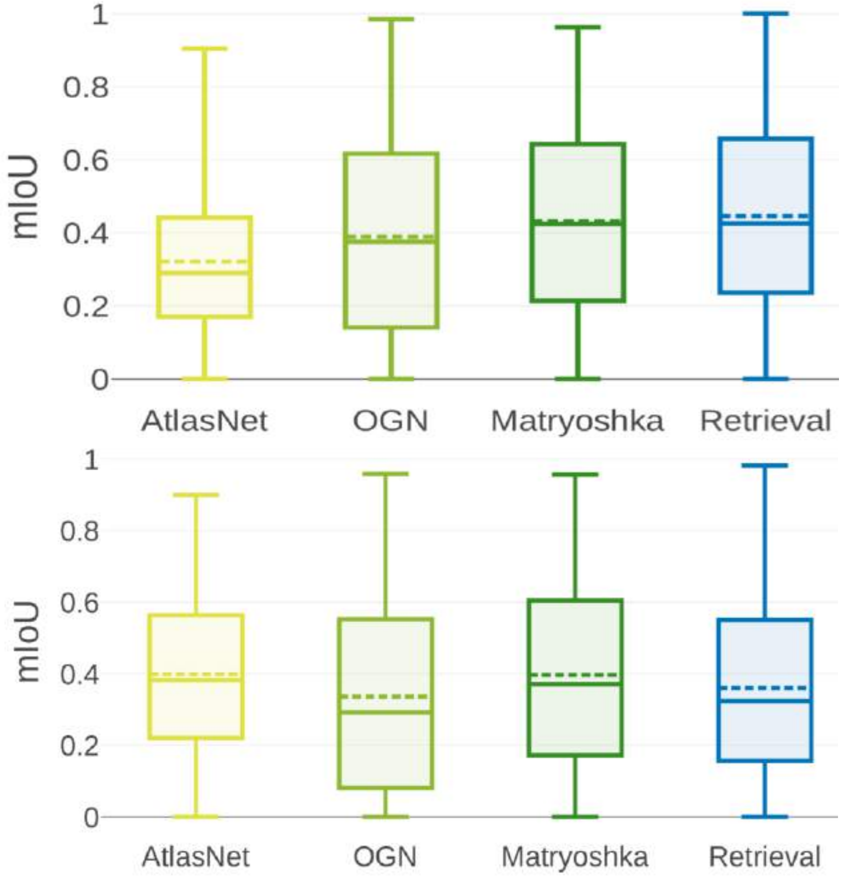


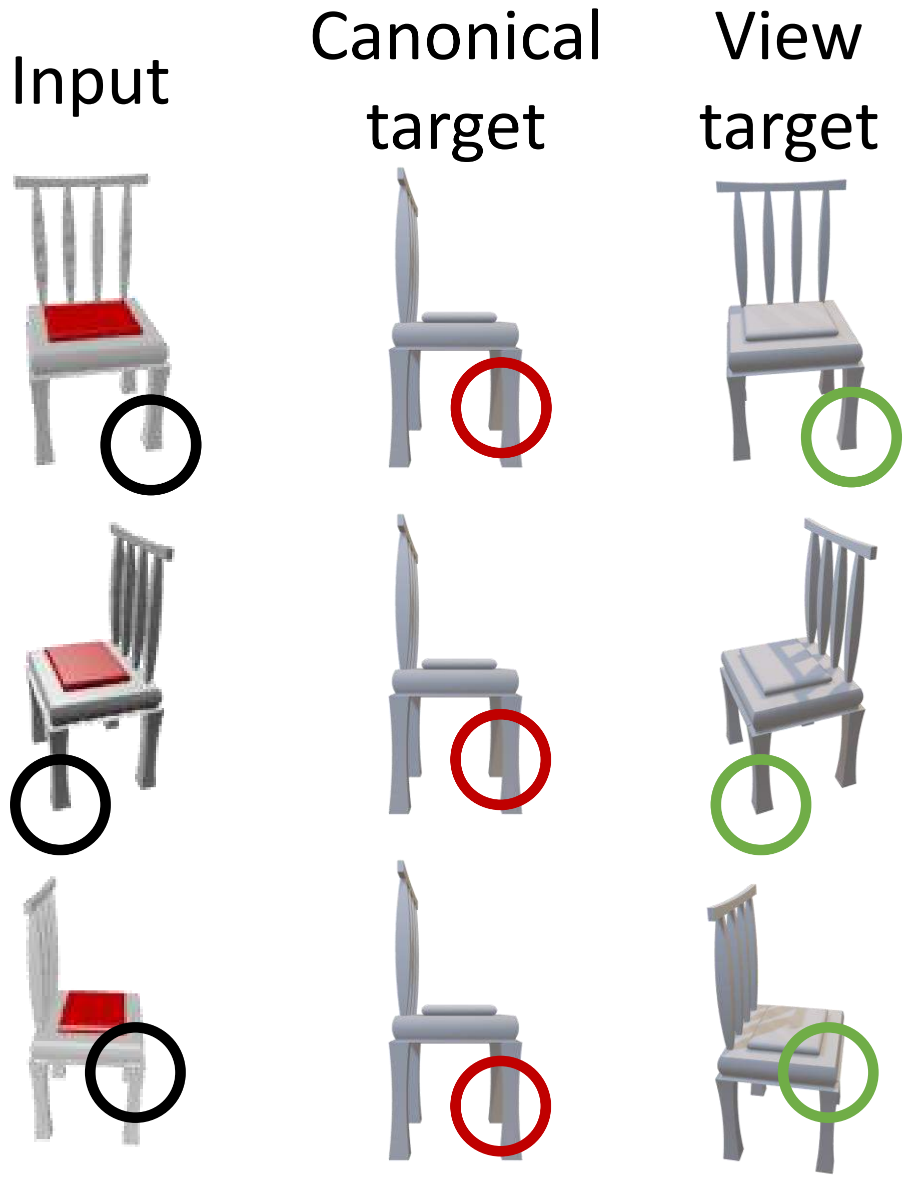


Shin et al, "Pixels, voxels, and views: A study of shape representations for single view 3D object shape prediction", CVPR 2018

Cameras: Canonical vs View coordinates

Problem: In canonical view, many methods perform on par with retrieval baseline! Not true in view coordinates

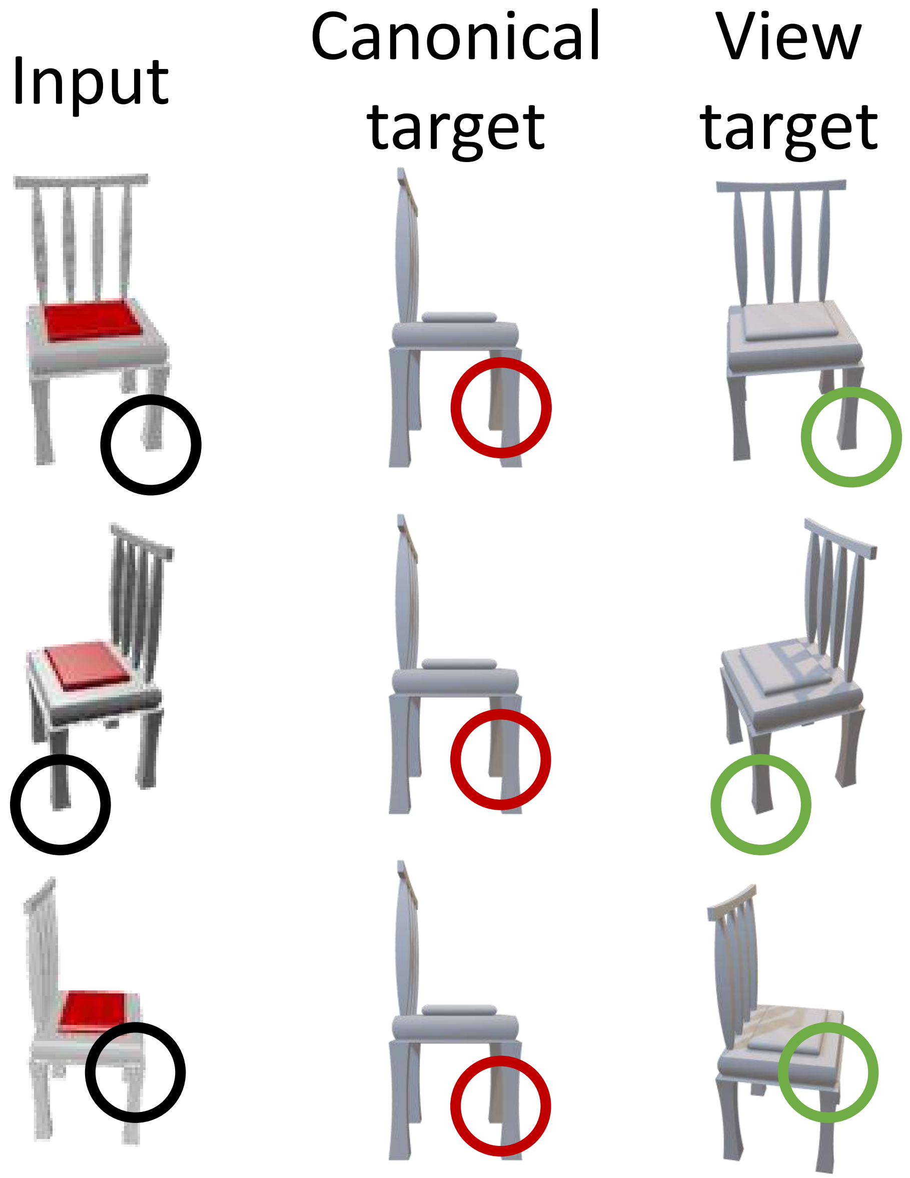




Tatarchenko et al, "What Do Single-view 3D Reconstruction Networks Learn?", CVPR 2019

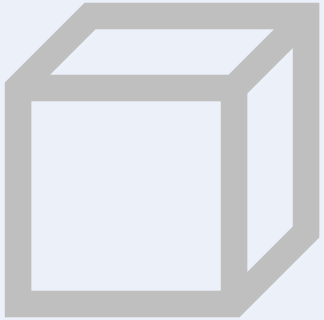
Cameras: Canonical vs View coordinates

Conclusion: We should prefer to predict in view coordinates!

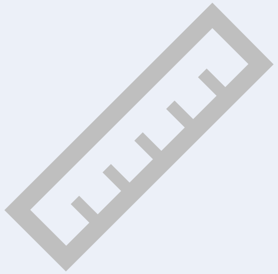


Predicting 3D Shapes from 2D Images

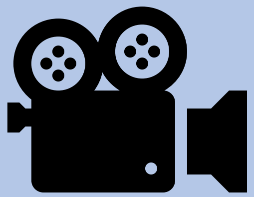
Shape Representations



Metrics



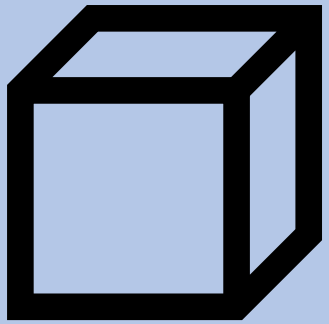
Camera Systems



Datasets

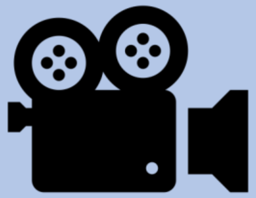
Predicting 3D Shapes from 2D Images

Shape Representations



Metrics

Camera Systems



Datasets

3D Shape 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

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

3D Shape 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

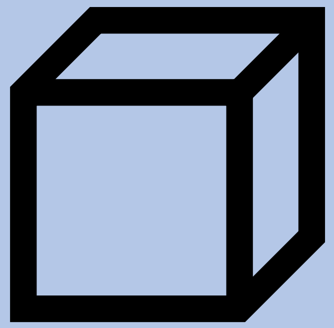
(+) Real images! Context!

(-) 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

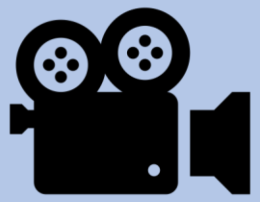
Predicting 3D Shapes from 2D Images

Shape Representations



Metrics

Camera Systems



Datasets

Mesh R-CNN

Georgia Gkioxari

Jitendra Malik

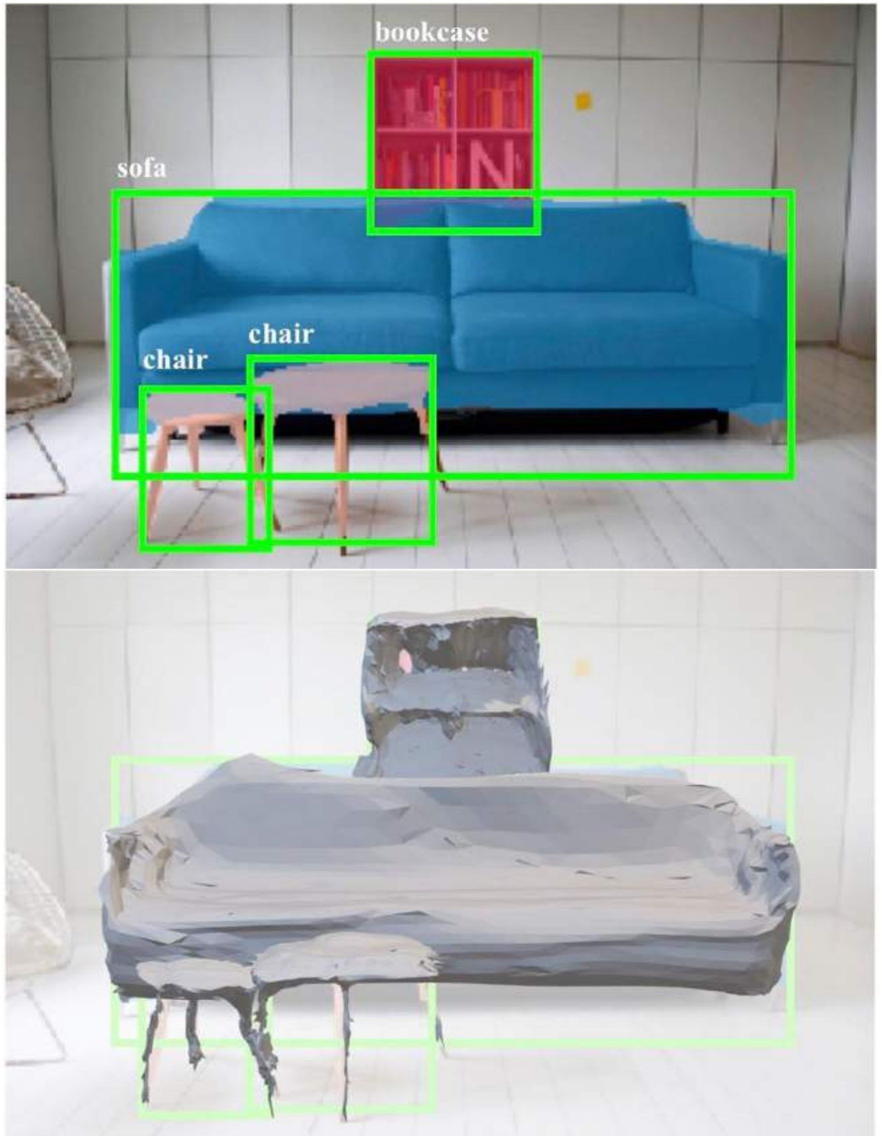
Justin Johnson

Facebook AI Research

Mesh R-CNN: Task

Input: Single RGB image

Output:
A set of detected objects
For each object:
- Bounding box
- Category label
- Instance segmentation
- 3D triangle mesh



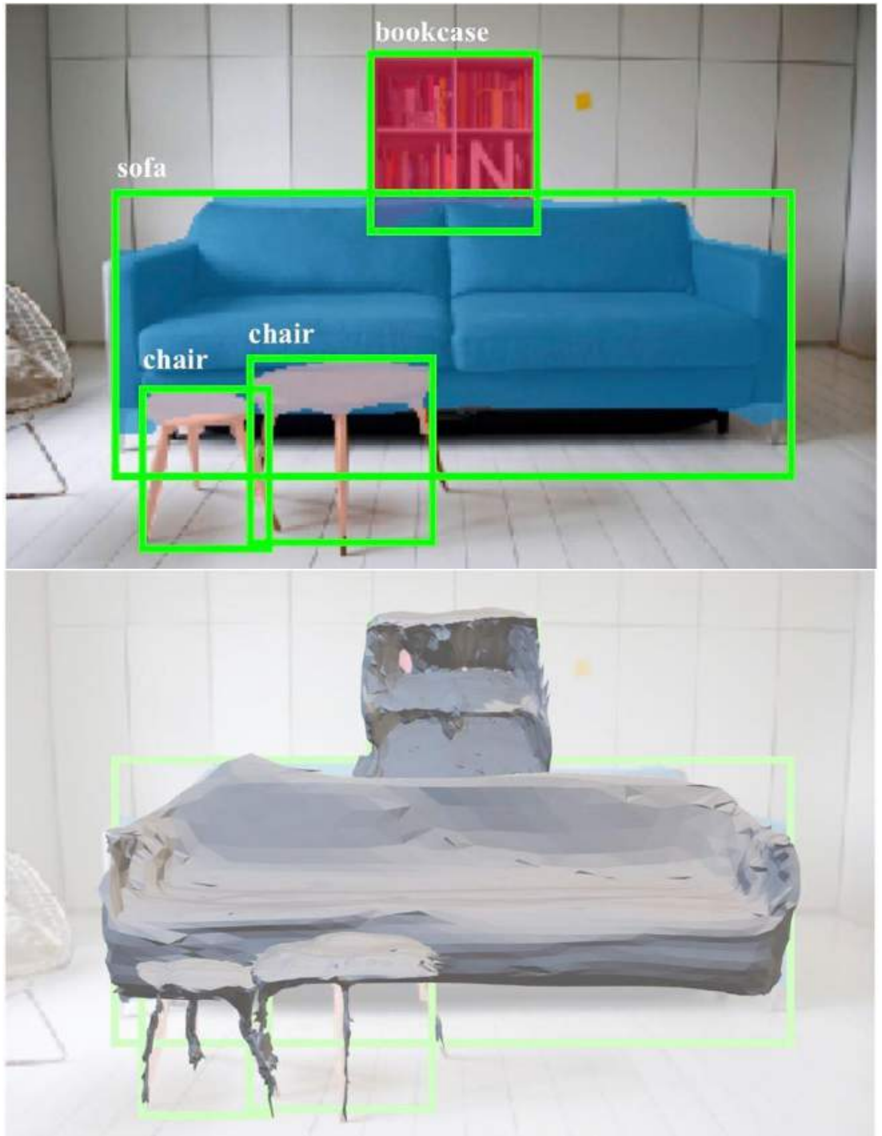
Mesh R-CNN: Task

Input: Single RGB image

Output:
A set of detected objects
For each object:
- Bounding box
- Category label
- Instance segmentation
- 3D triangle mesh

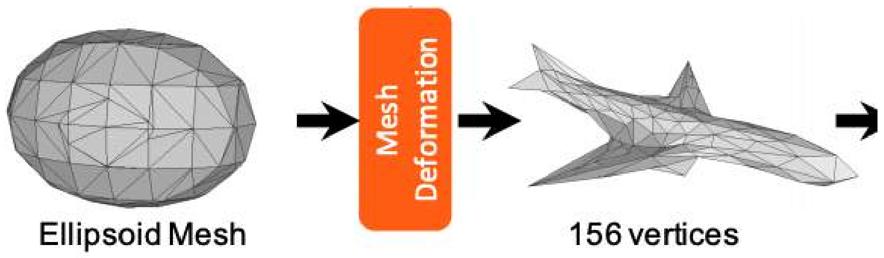
Mask R-CNN

Mesh head

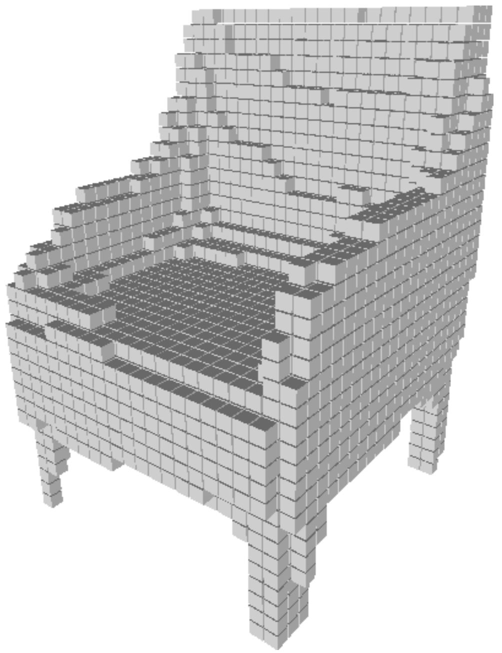


Mesh R-CNN: Hybrid 3D shape representation

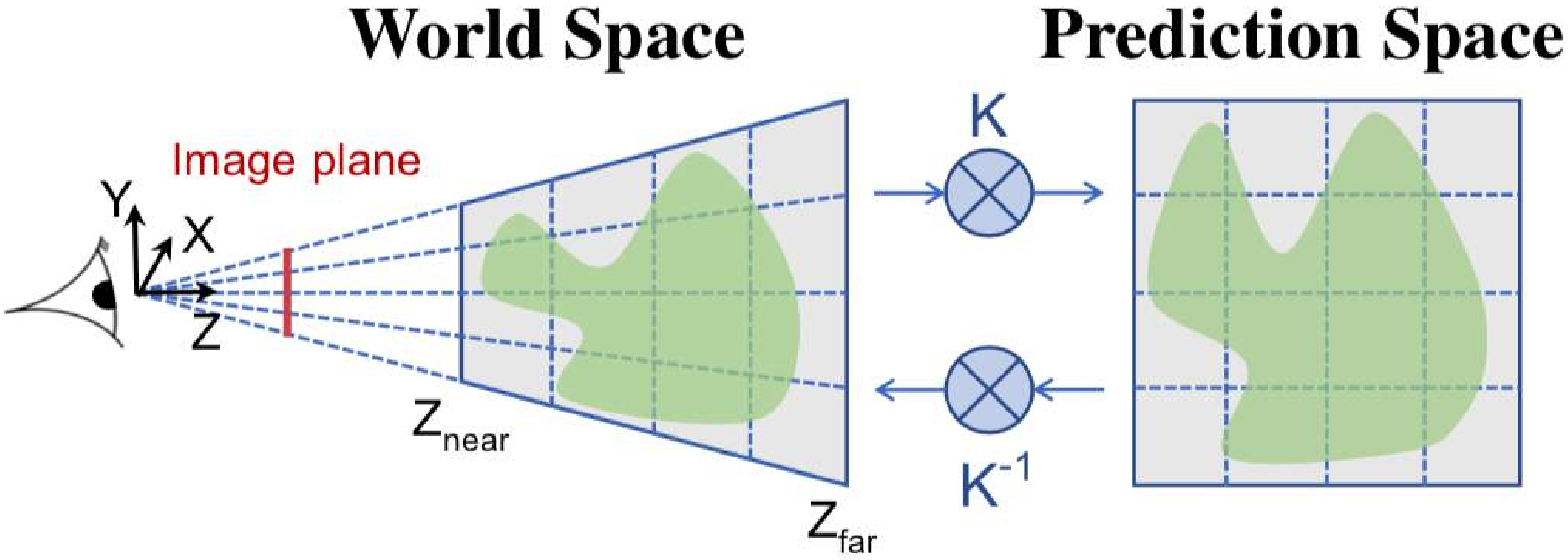
Mesh deformation gives good results, but the topology (verts, faces, genus, connected components) fixed by the initial mesh



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



Mesh R-CNN: Camera system



View-centric predictions! Voxels take perspective camera into account, so our “voxels” are actually frustums

Mesh R-CNN Pipeline

Input image

Mesh R-CNN Pipeline

Input image

→

2D object recognition

Mesh R-CNN Pipeline

Input image

2D object recognition

3D object voxels

Mesh R-CNN Pipeline

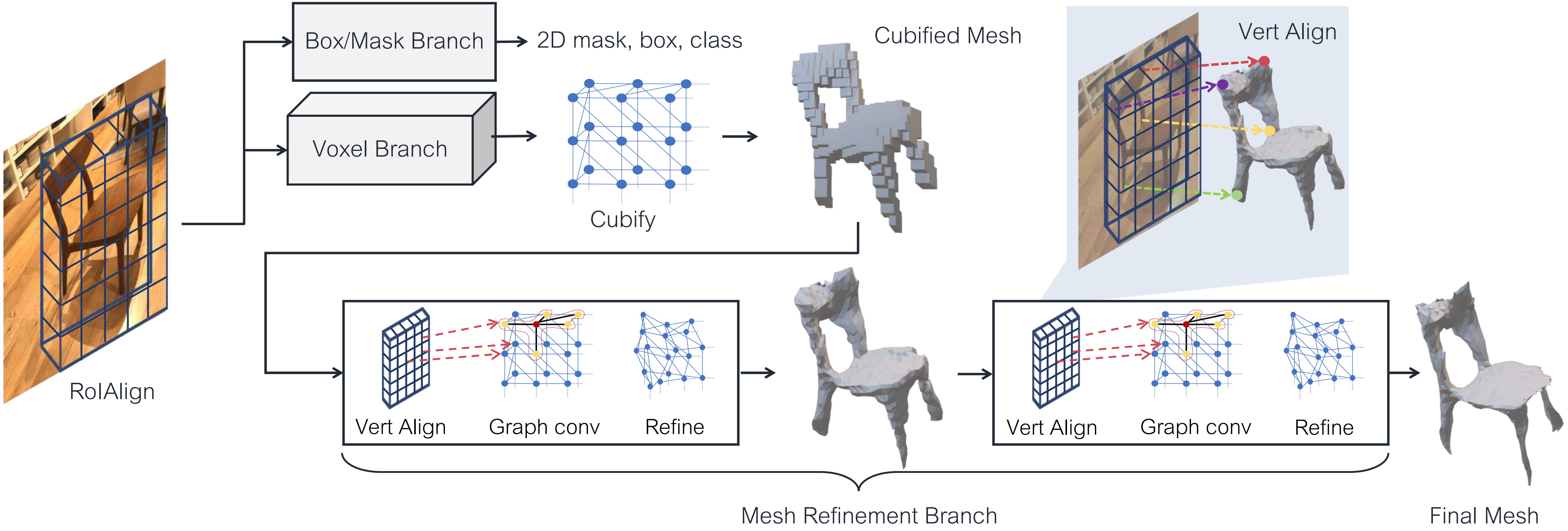
Input image

2D object recognition

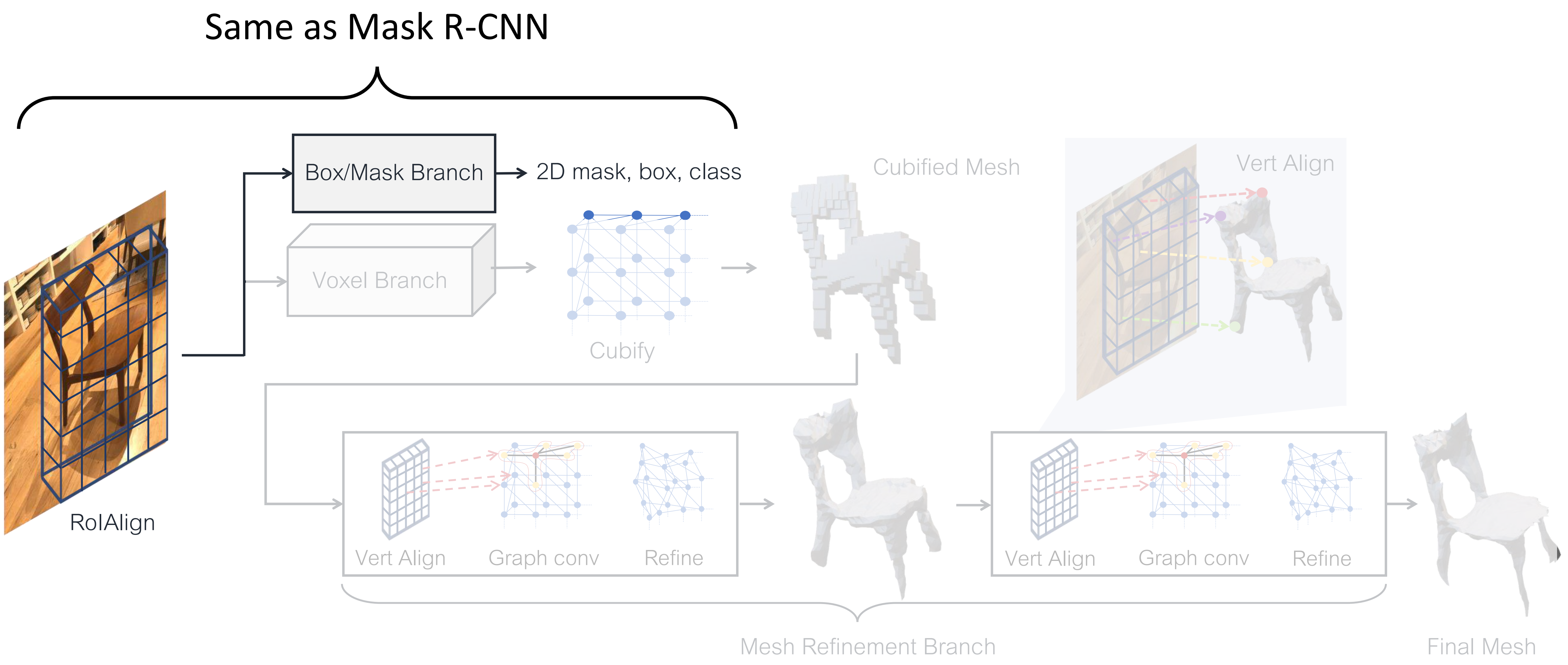
3D object voxels

3D object meshes

Mesh R-CNN: Archtecture

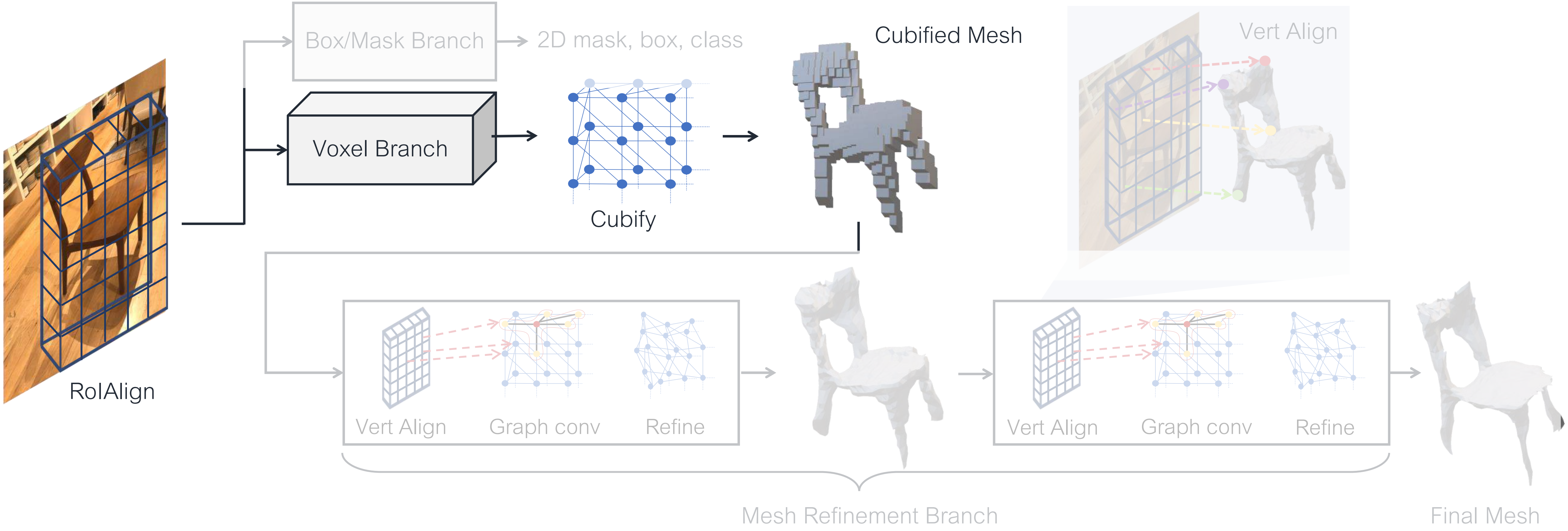


Mesh R-CNN: Architecture



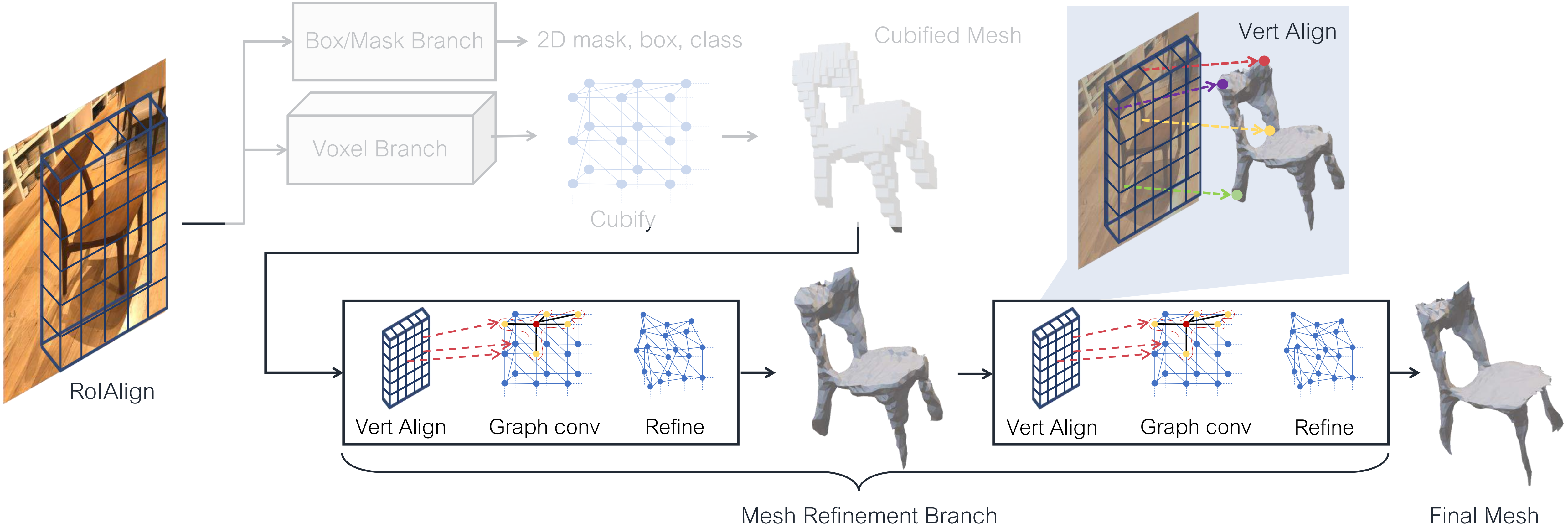
Mesh R-CNN: Architecture

Voxel branch: predict voxels, then threshold and convert to mesh



Mesh R-CNN: Architecture

Mesh refinement branch: Iteratively refine mesh through stages of vertex feature alignment, graph convolution, and vertex offset



Mesh R-CNN: Training Losses

- Instance segmentation losses: Same as Mask R-CNN
  - RPN classification
  - RPN bounding box regression
  - Per-region classification
  - Per-region bounding box regression
  - Per-region instance segmentation mask
- Voxel loss: Binary cross-entropy loss on voxel occupancy
- Mesh loss: Chamfer distance on sampled points at each stage
- Mesh regularizer: Minimize length of edges in mesh

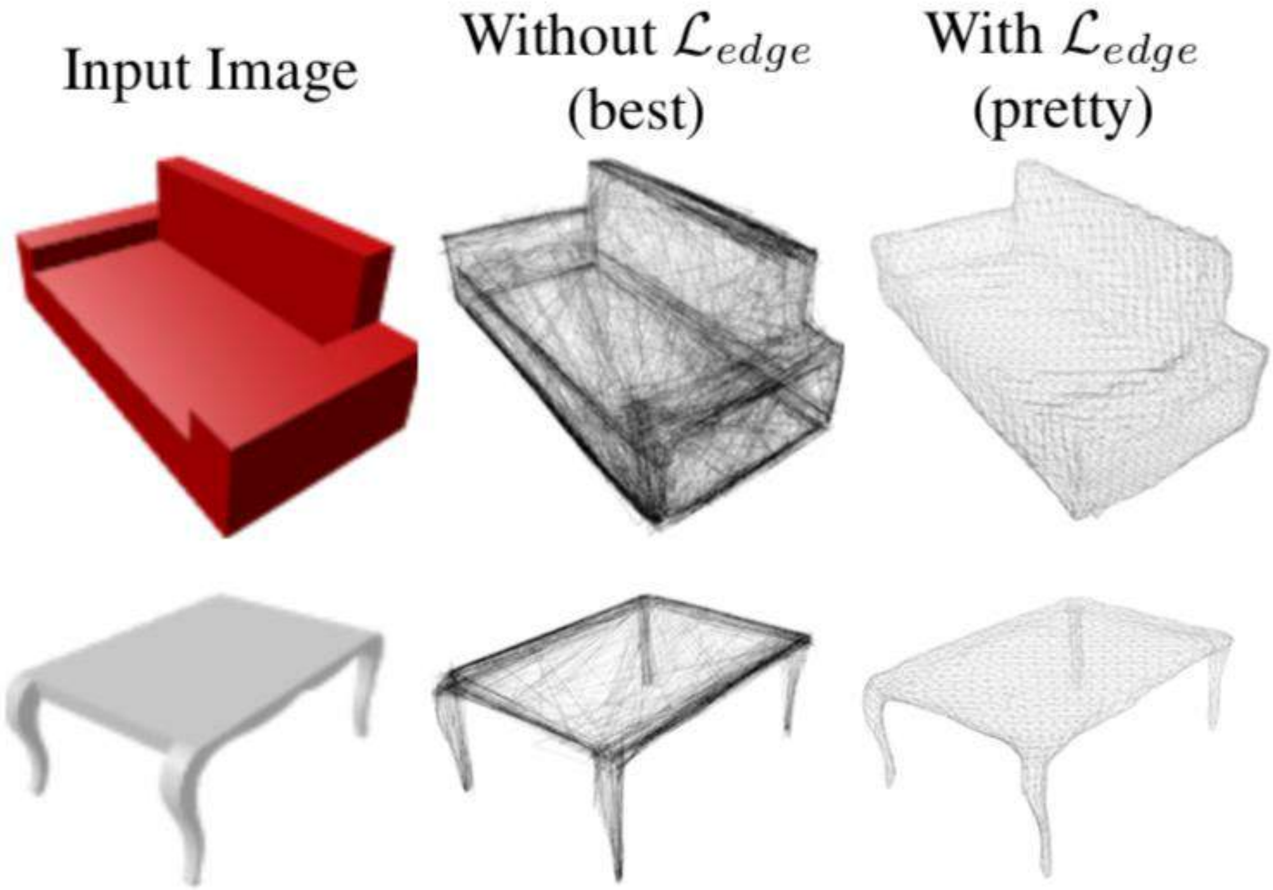
Mesh R-CNN: Batching

Each of our predicted meshes can have varying numbers of vertices and faces

Prior work that predicts meshes either shares mesh topology over the batch, or uses batch size=1

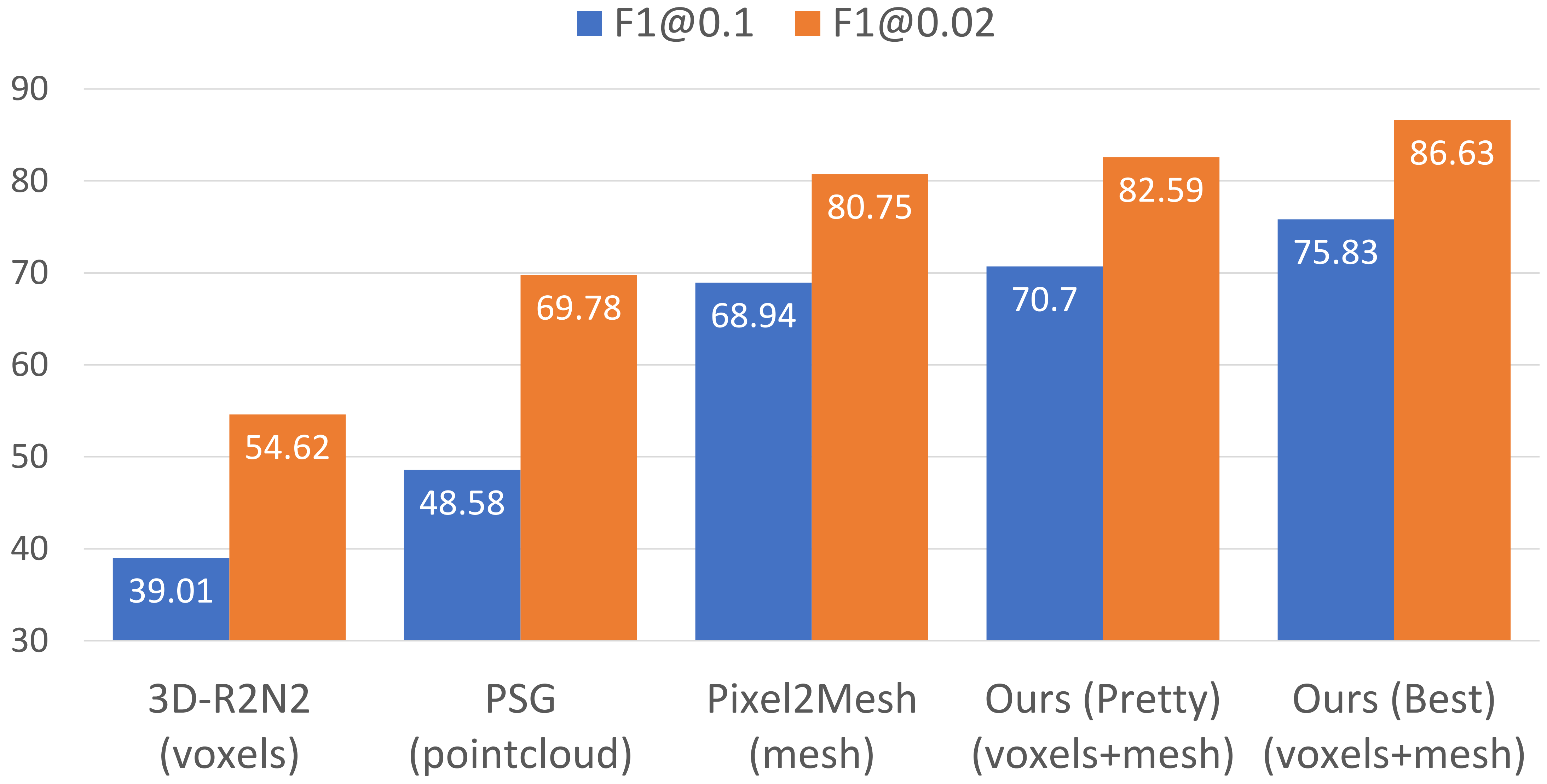
Our implementation handles batches of heterogenous meshes for all operations

Results on ShapeNet: Best vs Pretty



Using regularizers produces smooth shapes but metrics are worse. Note that the metrics do not capture “smoothness”.

Mesh R-CNN: ShapeNet results



Mesh R-CNN: ShapeNet Results

Results on Pix3D

Task & Metrics:

- Detect all objects in the image: AP^{box}
- Predict their instance mask: AP^{mask}
- Predict their 3D shape: AP^{mesh}

Definition of AP^{mesh}:

- A detection is true positive if the predicted class is correct, it is not a duplicate detection and its $F1^{0.3} > 0.5$

Results on Pix3D

Random Split S_1:

Images are split randomly into ~7500 train and ~2500 test.

Pix3D $\mathcal{S}_1$	AP$^{\text{box}}$	AP$^{\text{mask}}$	AP$^{\text{mesh}}$	*chair*	*sofa*	*table*	*bed*	*desk*	*bkcs*	*wrdrb*	*tool*	*misc*	$\|V\|$	$\|F\|$
Voxel-Only	92.1	86.6	2.1	0.0	0.3	0.7	0.1	0.5	14.3	3.1	0.0	0.0	$1548_{\pm 506}$	$2855_{\pm 917}$
Pixel2Mesh$^+$	90.4	86.5	42.6	32.5	61.8	46.6	43.1	36.3	57.5	43.9	29.5	**32.4**	$2562_{\pm 0}$	$5120_{\pm 0}$
Sphere-Init	90.4	85.4	42.1	32.8	61.4	49.1	38.4	33.4	53.7	45.7	**40.2**	23.7	$2562_{\pm 0}$	$5120_{\pm 0}$
Mesh R-CNN (ours)	92.3	87.4	**48.4**	**42.6**	**66.9**	**56.6**	**48.2**	**38.6**	**70.0**	**54.2**	36.6	21.8	$1548_{\pm 506}$	$2855_{\pm 917}$
# test instances	2440	2440	2440	1129	398	398	205	148	79	53	11	19		

Results on Pix3D

Random Split S_1:

Images are split randomly into ~7500 train and ~2500 test.

Results on Pix3D

Shape Split S_2:

Images are split such that a shape seen on test is **not** seen on train

For example, for chair

Results on Pix3D

Pix3D $\mathcal{S}_1$	AP$^{\text{box}}$	AP$^{\text{mask}}$	AP$^{\text{mesh}}$	*chair*	*sofa*	*table*	*bed*	*desk*	*bkcs*	*wrdrb*	*tool*	*misc*	$\|V\|$	$\|F\|$
Voxel-Only	92.1	86.6	2.1	0.0	0.3	0.7	0.1	0.5	14.3	3.1	0.0	0.0	$1548_{\pm 506}$	$2855_{\pm 917}$
Pixel2Mesh$^+$	90.4	86.5	42.6	32.5	61.8	46.6	43.1	36.3	57.5	43.9	29.5	**32.4**	$2562_{\pm 0}$	$5120_{\pm 0}$
Sphere-Init	90.4	85.4	42.1	32.8	61.4	49.1	38.4	33.4	53.7	45.7	**40.2**	23.7	$2562_{\pm 0}$	$5120_{\pm 0}$
Mesh R-CNN (ours)	92.3	87.4	**48.4**	**42.6**	**66.9**	**56.6**	**48.2**	**38.6**	**70.0**	**54.2**	36.6	21.8	$1548_{\pm 506}$	$2855_{\pm 917}$
# test instances	2440	2440	2440	1129	398	398	205	148	79	53	11	19		
Pix3D $\mathcal{S}_2$														
Voxel-Only	66.5	60.1	3.9	0.0	0.0	0.5	0.2	0.8	16.1	0.1	17.0	0.0	$1496_{\pm 437}$	$2758_{\pm 799}$
Pixel2Mesh$^+$	60.4	57.3	22.3	24.9	66.2	14.5	**42.6**	6.8	**20.0**	1.2	24.1	0.0	$2562_{\pm 0}$	$5120_{\pm 0}$
Sphere-Init	63.0	58.4	22.0	23.9	68.8	17.5	38.2	5.5	17.9	0.8	25.2	0.0	$2562_{\pm 0}$	$5120_{\pm 0}$
Mesh R-CNN (ours)	63.7	58.9	**25.1**	**27.0**	**74.3**	**22.6**	38.2	**8.6**	**20.0**	**1.7**	**33.2**	0.0	$1496_{\pm 437}$	$2758_{\pm 799}$
# test instances	2368	2368	2368	778	506	398	219	205	85	135	22	20		

Results on Pix3D

Pix3D $\mathcal{S}_1$	AP^{box}	AP^{mask}	AP^{mesh}	*chair*	*sofa*	*table*	*bed*	*desk*	*bkcs*	*wrdrb*	*tool*	*misc*	$\lvert V \rvert$	$\lvert F \rvert$
Voxel-Only	92.1	86.6	2.1	0.0	0.3	0.7	0.1	0.5	14.3	3.1	0.0	0.0	$1548_{\pm 506}$	$2855_{\pm 917}$
Pixel2Mesh^{+}	90.4	86.5	42.6	32.5	61.8	46.6	43.1	36.3	57.5	43.9	29.5	**32.4**	$2562_{\pm 0}$	$5120_{\pm 0}$
Sphere-Init	90.4	85.4	42.1	32.8	61.4	49.1	38.4	33.4	53.7	45.7	**40.2**	23.7	$2562_{\pm 0}$	$5120_{\pm 0}$
Mesh R-CNN (ours)	92.3	87.4	**48.4**	**42.6**	**66.9**	**56.6**	**48.2**	**38.6**	**70.0**	**54.2**	36.6	21.8	$1548_{\pm 506}$	$2855_{\pm 917}$
# test instances	2440	2440	2440	1129	398	398	205	148	79	53	11	19		
Pix3D $\mathcal{S}_2$														
Voxel-Only	66.5	60.1	3.9	0.0	0.0	0.5	0.2	0.8	16.1	0.1	17.0	0.0	$1496_{\pm 437}$	$2758_{\pm 799}$
Pixel2Mesh^{+}	60.4	57.3	22.3	24.9	66.2	14.5	**42.6**	6.8	**20.0**	1.2	24.1	0.0	$2562_{\pm 0}$	$5120_{\pm 0}$
Sphere-Init	63.0	58.4	22.0	23.9	68.8	17.5	38.2	5.5	17.9	0.8	25.2	0.0	$2562_{\pm 0}$	$5120_{\pm 0}$
Mesh R-CNN (ours)	63.7	58.9	**25.1**	**27.0**	**74.3**	**22.6**	38.2	**8.6**	**20.0**	**1.7**	**33.2**	0.0	$1496_{\pm 437}$	$2758_{\pm 799}$
# test instances	2368	2368	2368	778	506	398	219	205	85	135	22	20		

Results on Pix3D

Summary

- Predicting 3D shapes from 2D images requires rethinking shape representations, metrics, datasets
- Mesh R-CNN jointly detects objects and emits a triangle mesh for each predicted object
- We hope Mesh R-CNN can help take perception and recognition to the next dimension!
- Mesh R-CNN is built in PyTorch on top of Detectron2, we will release code and models!

FAIR Research Engineer

Menlo Park, CA
Seattle, WA

ACCELERATE AND SCALE CV RESEARCH

Familiarity with CV and ML
Ability to write high-quality and performance-critical code

wlo@fb.com