Agenda

- Deep learning on regular structures
- Deep learning on meshes
- **Deep learning on point cloud and parametric models**
  - Point cloud analysis
  - Point cloud synthesis
  - Primitive-based shapes

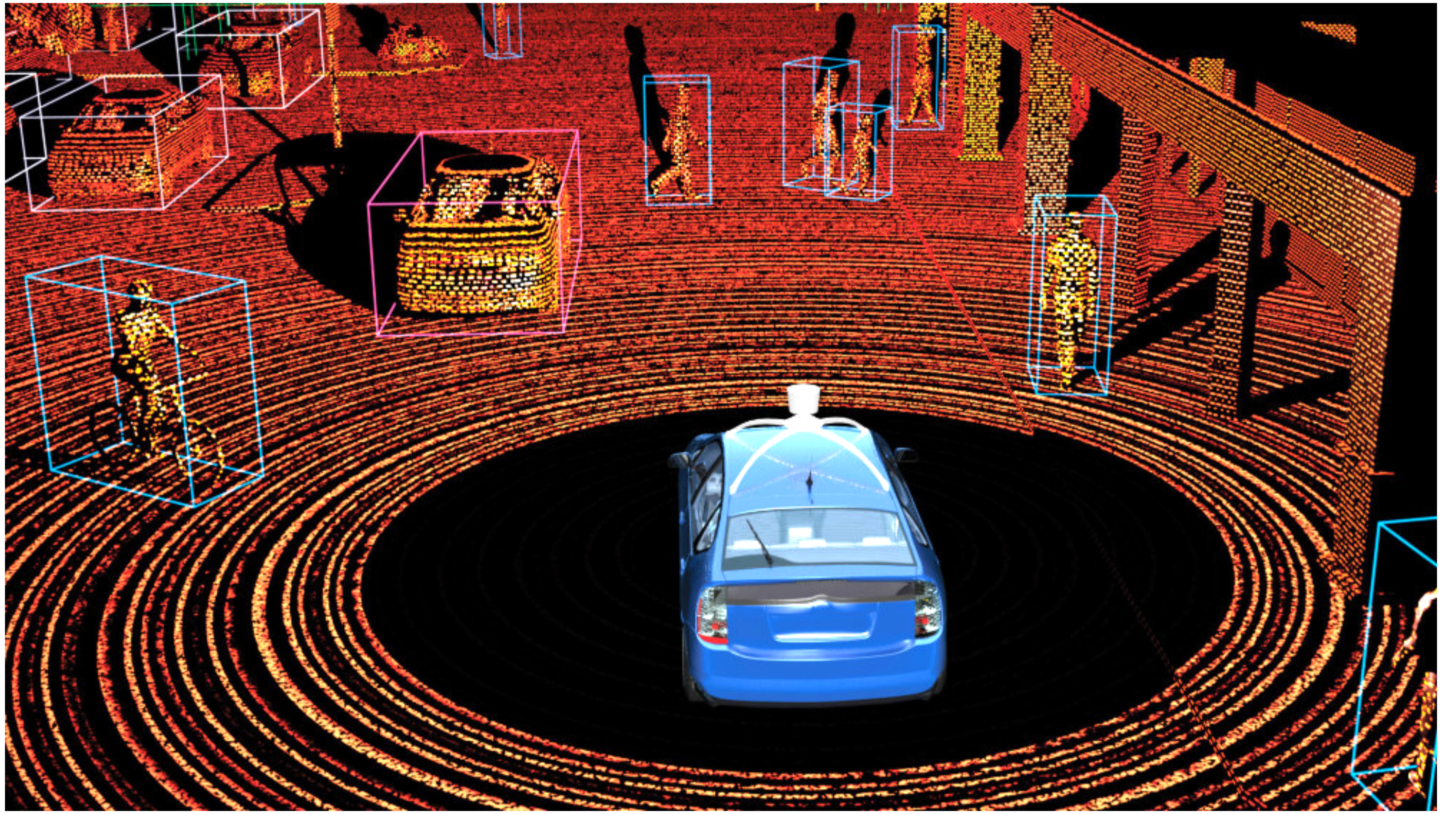
Agenda

- Deep learning on regular structures
- Deep learning on meshes
- **Deep learning on point cloud and parametric models**
  - **Point cloud analysis**
  - Point cloud synthesis
  - Primitive-based shapes

Applications of Point Set Analysis

- **Robot Perception**

What and where are the objects in a LiDAR scanned scene?

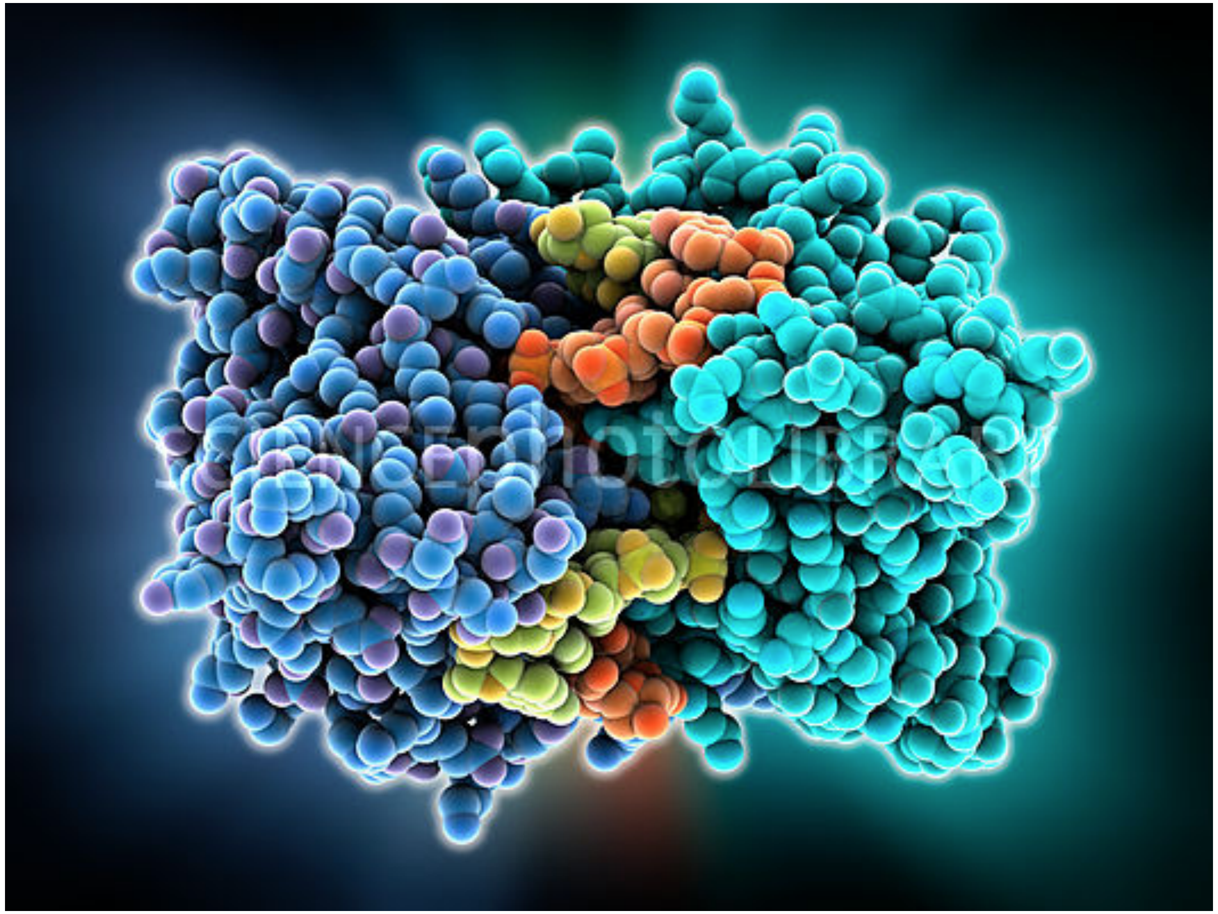


https://3dprint.com/116569/self-driving-cars-privacy/

Applications of Point Set Analysis

- **Molecular Biology**

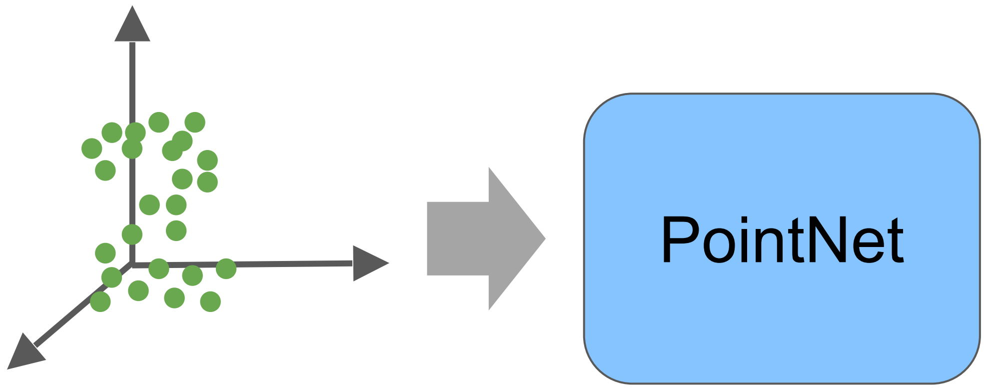
Can we infer an enzyme's category (reactions they catalyze) from its structure?



EcoRV restriction enzyme molecule, LAGUNA DESIGN/SCIENCE PHOTO LIBRARY

Directly process point cloud data

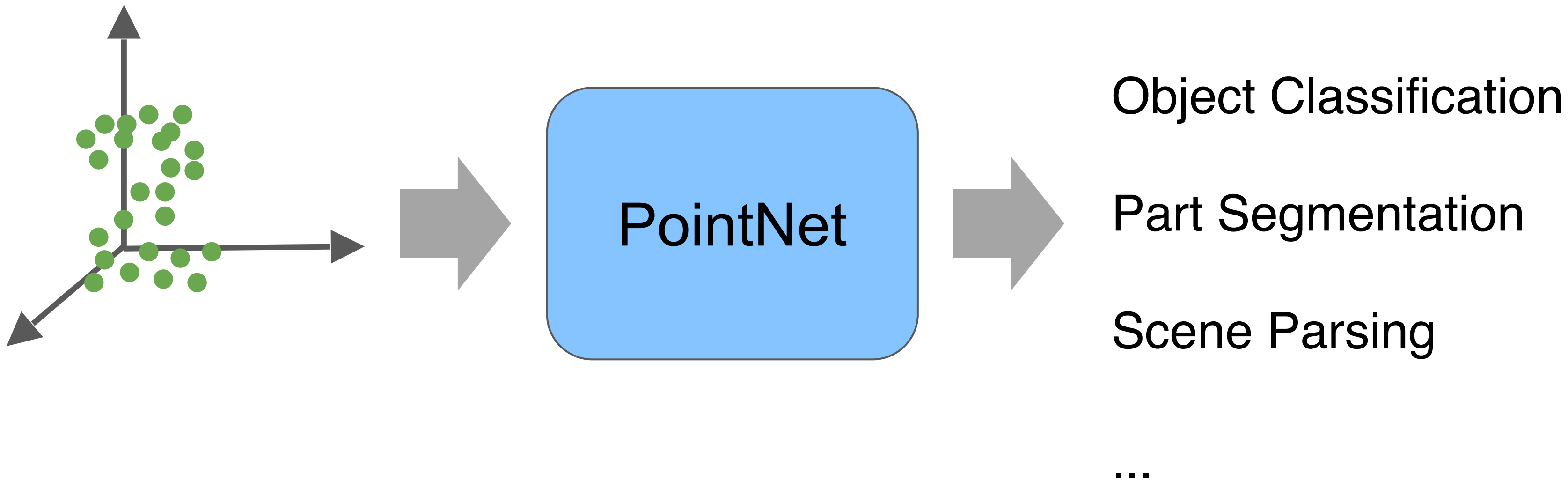
End-to-end learning for **unstructured, unordered** point data



Directly process point cloud data

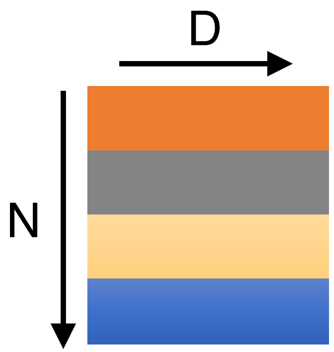
End-to-end learning for **unstructured, unordered** point data

Unified framework for various tasks



Properties of a desired neural network on point clouds

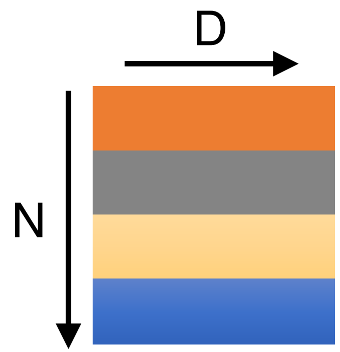
Point cloud: N **orderless** points, each represented by a D dim coordinate



2D array representation

Properties of a desired neural network on point clouds

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



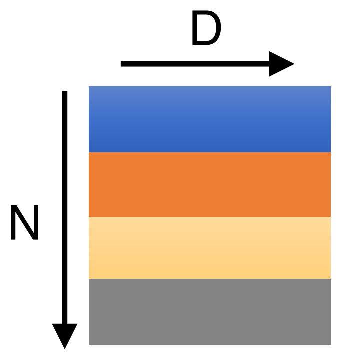
2D array representation

Permutation invariance

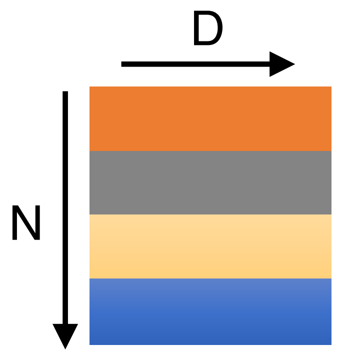
Transformation invariance

Properties of a desired neural network on point clouds

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



represents the same **set** as



2D array representation

Permutation invariance

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:

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

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

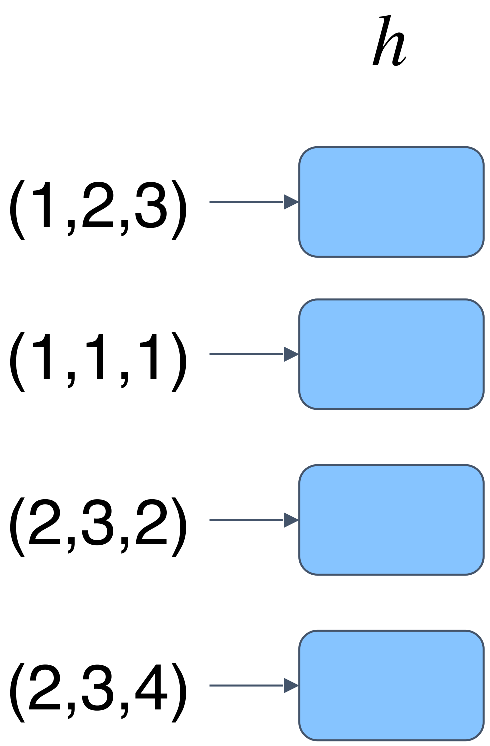
$\ldots$

Construct symmetric function family

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

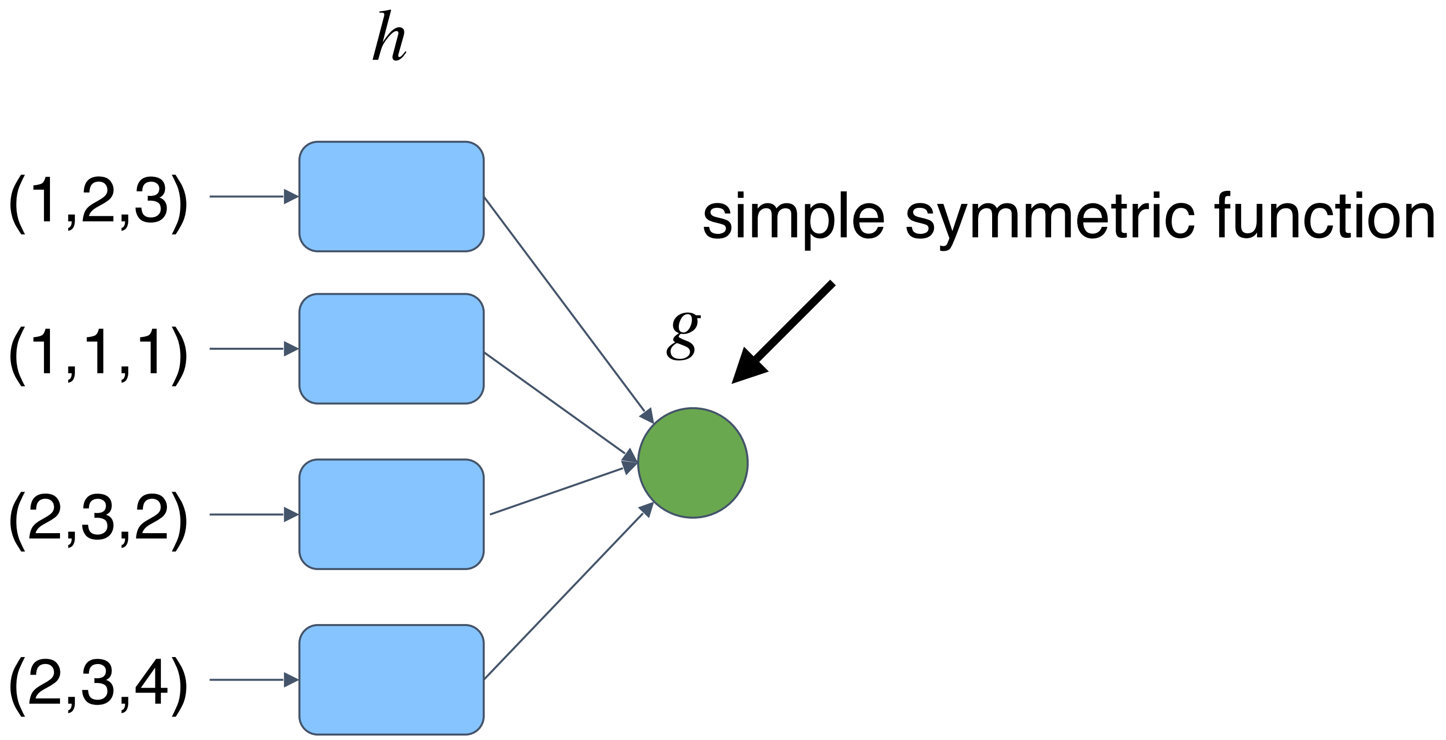
Construct symmetric function family

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



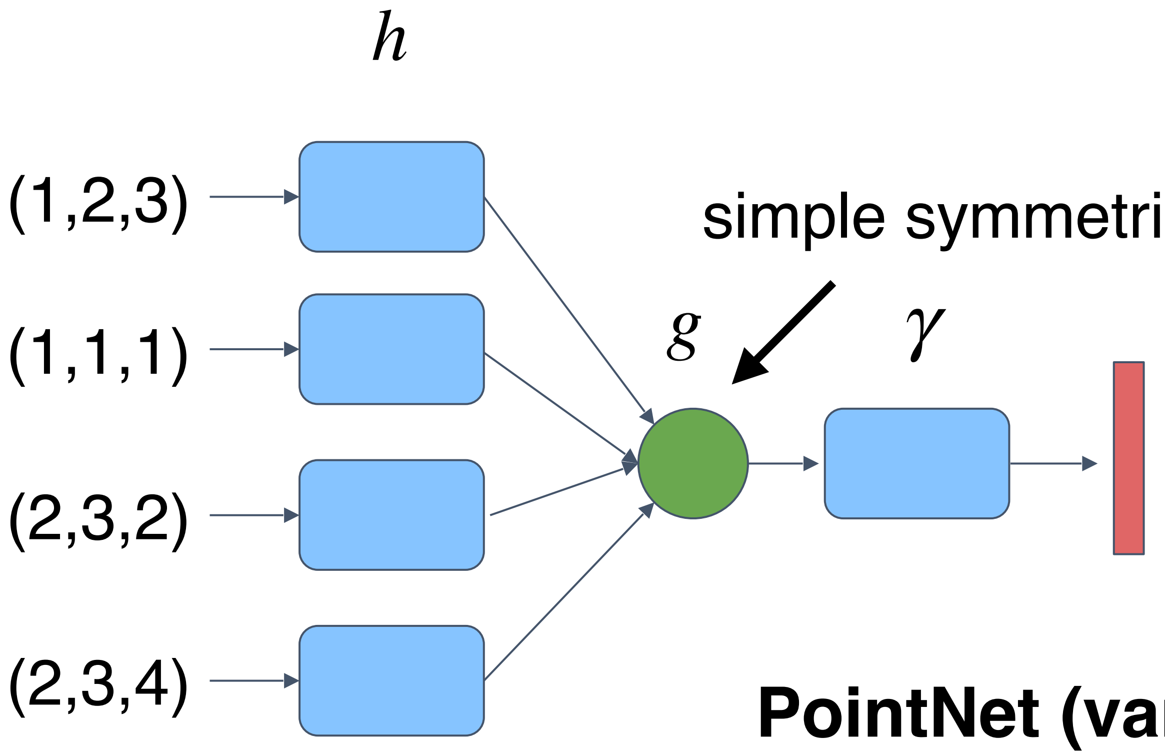
Construct symmetric function family

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



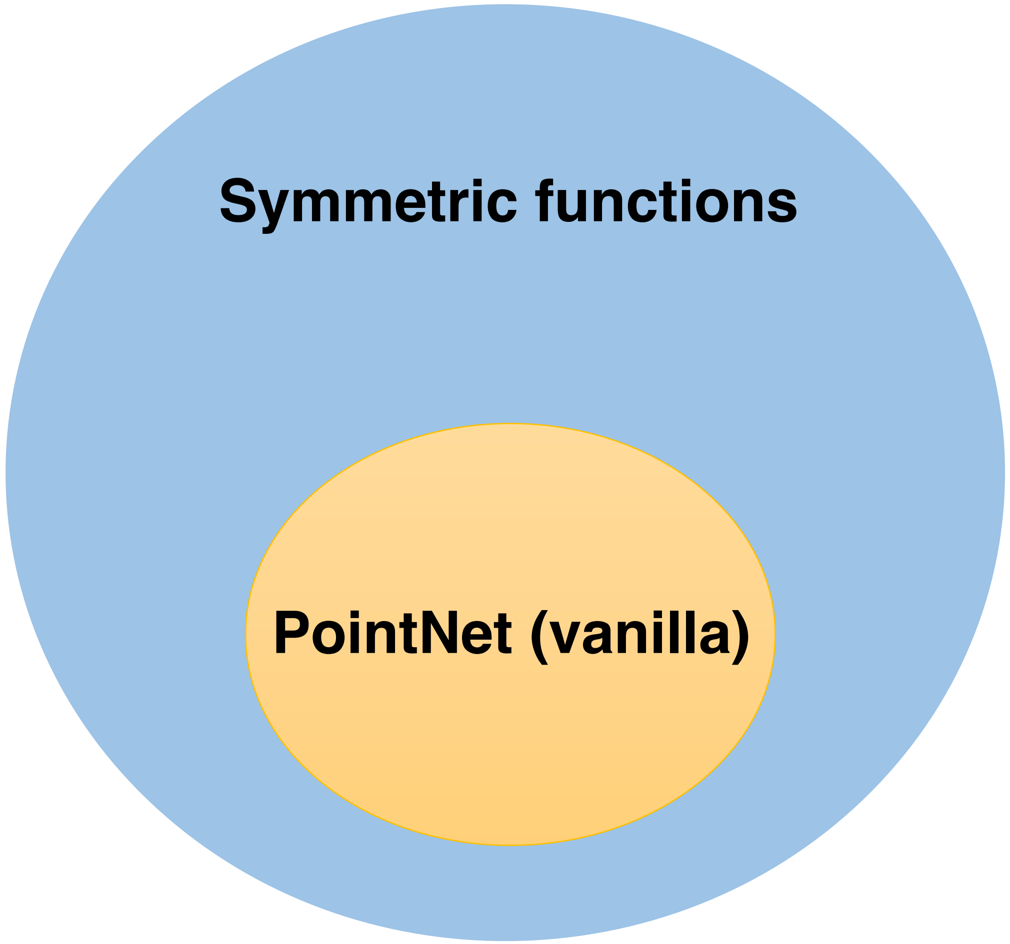
Construct symmetric function family

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)

Q: What symmetric functions can be constructed by PointNet?



A: Universal approximation to **continuous** symmetric functions

Theorem:

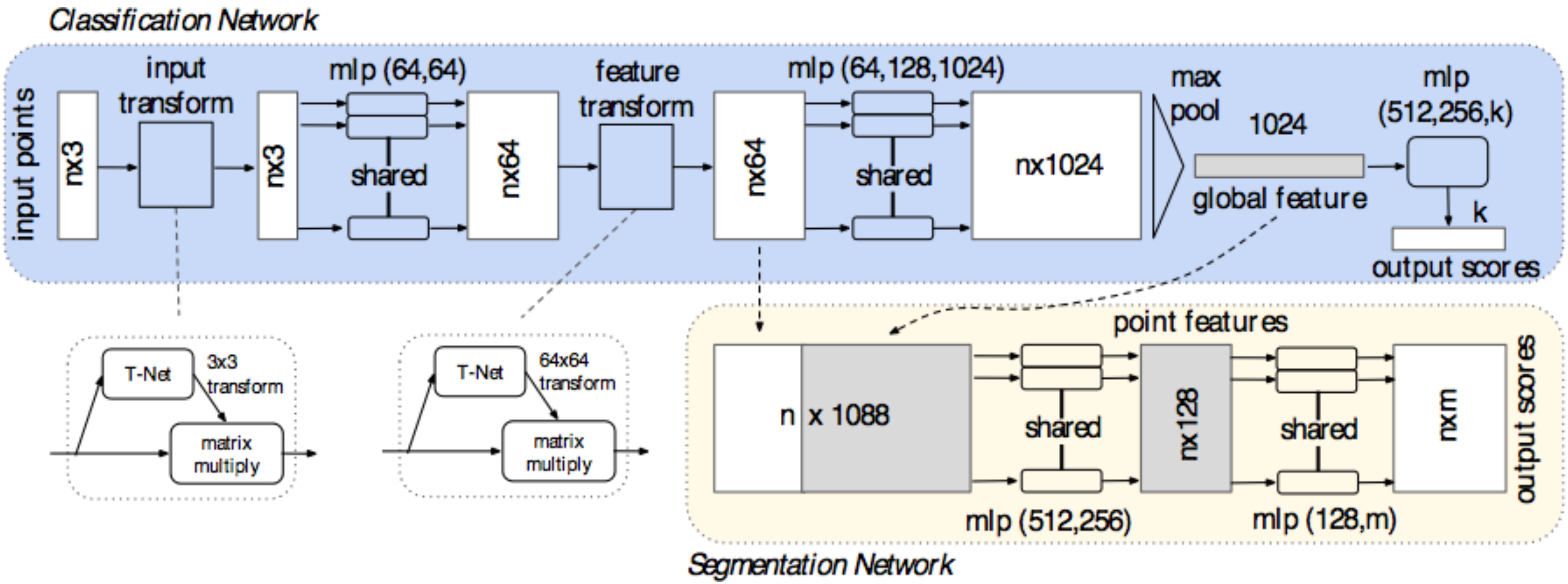
A Hausdorff continuous symmetric function $f : 2^{\mathcal{X}} \rightarrow \mathbb{R}$ can be arbitrarily approximated by PointNet.

$$\left| f(S) - \boxed{\gamma \left(\underset{x_i \in S}{\mathrm{MAX}} \{h(x_i)\} \right)} \right| < \epsilon$$

$S \subseteq \mathbb{R}^d$,

PointNet (vanilla)

PointNet Architecture

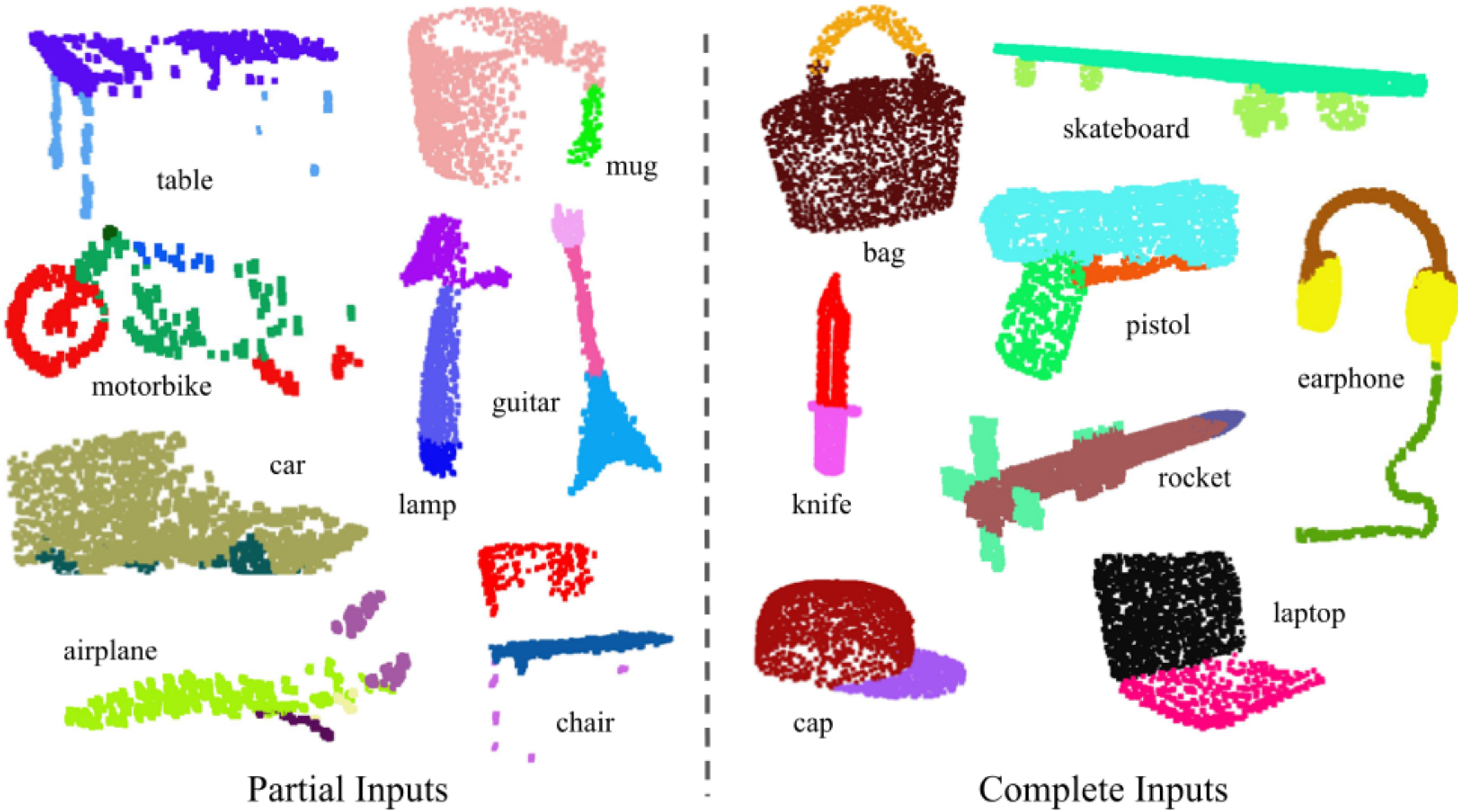


Results on Object Classification

Object Classification Accuracy on ModelNet40

	input	#views	accuracy avg. class	accuracy overall
SPH [12]	mesh	-	68.2	-
3DShapeNets [29]	volume	1	77.3	84.7
VoxNet [18]	volume	12	83.0	85.9
Subvolume [19]	volume	20	86.0	**89.2**
LFD [29]	image	10	75.5	-
MVCNN [24]	image	80	**90.1**	-
Ours baseline	point	-	72.6	77.4
Ours PointNet	point	1	86.2	**89.2**

Results on Object Part Segmentation

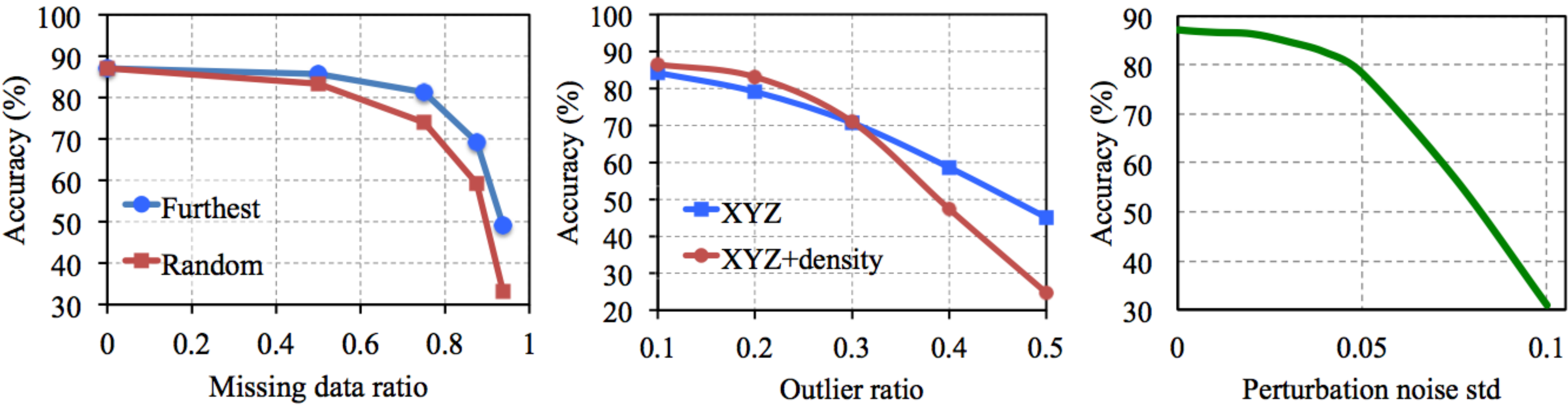


Results on Object Part Segmentation

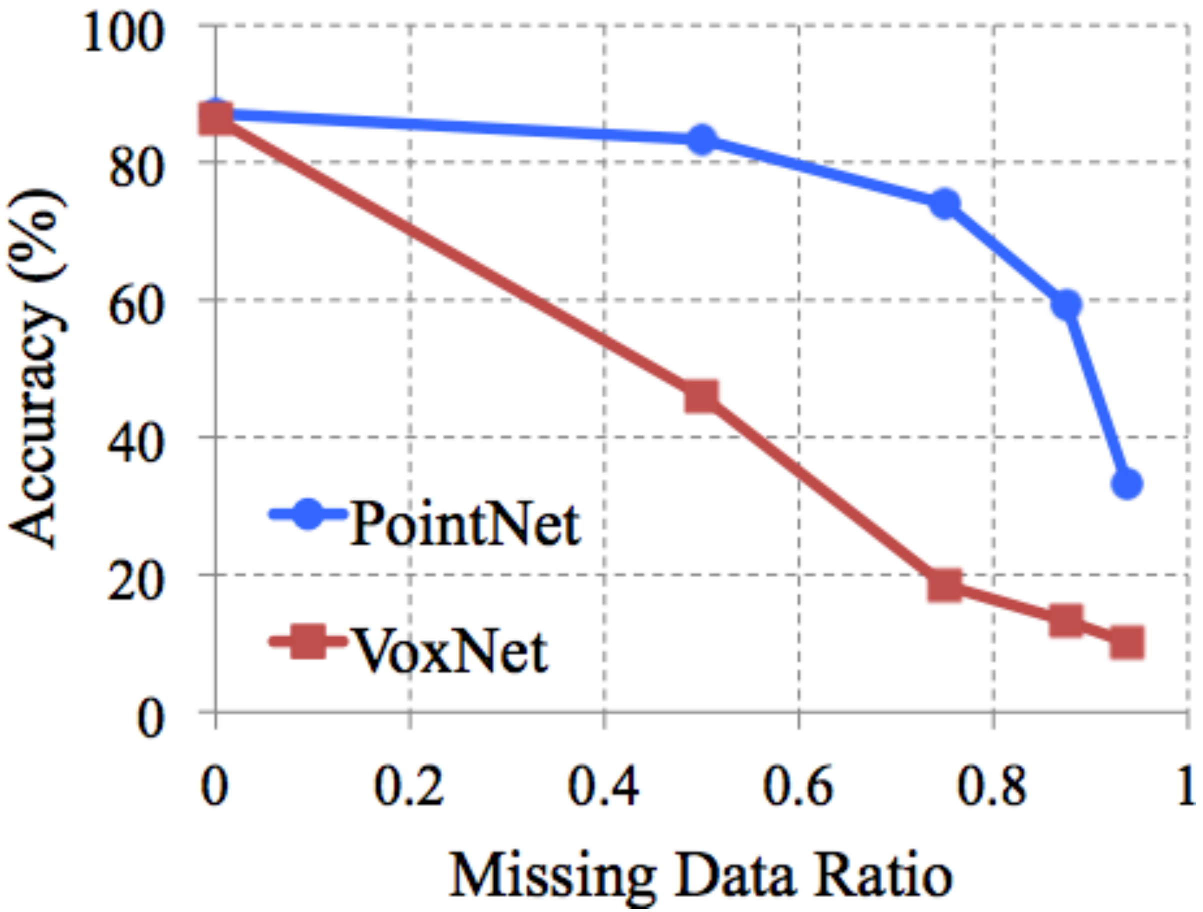
Part Segmentation mIoU on ShapeNet Part Dataset

	mean	aero	bag	cap	car	chair	ear phone	guitar	knife	lamp	laptop	motor	mug	pistol	rocket	skate board	table
# shapes		2690	76	55	898	3758	69	787	392	1547	451	202	184	283	66	152	5271
Wu [28]	-	63.2	-	-	-	73.5	-	-	-	74.4	-	-	-	-	-	-	74.8
Yi [30]	81.4	81.0	78.4	77.7	**75.7**	87.6	61.9	**92.0**	85.4	**82.5**	**95.7**	**70.6**	91.9	**85.9**	53.1	69.8	75.3
3DCNN	79.4	75.1	72.8	73.3	70.0	87.2	63.5	88.4	79.6	74.4	93.9	58.7	91.8	76.4	51.2	65.3	77.1
Ours	**83.7**	**83.4**	**78.7**	**82.5**	74.9	**89.6**	**73.0**	91.5	**85.9**	80.8	95.3	65.2	**93.0**	81.2	**57.9**	**72.8**	**80.6**

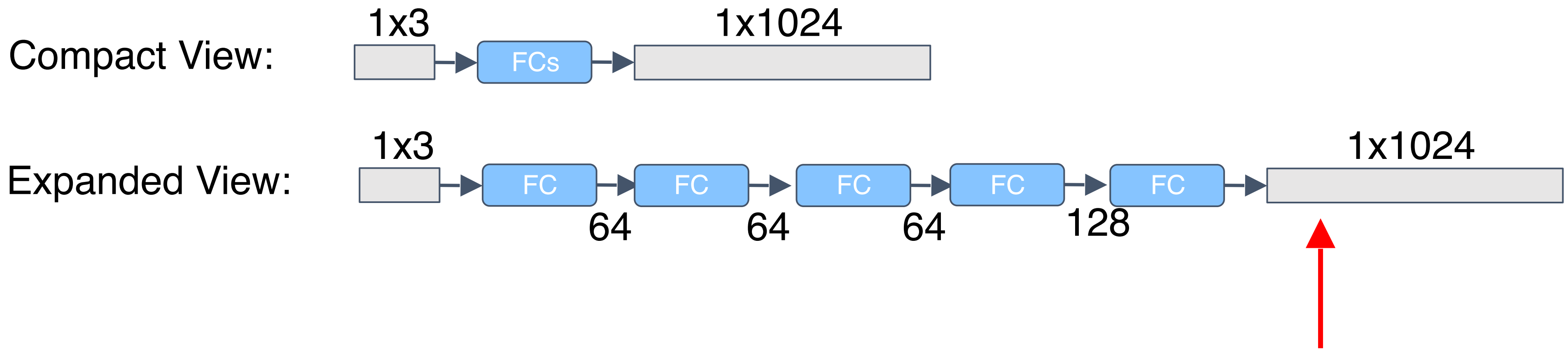
Robustness to Data Corruption



Robustness to Data Corruption



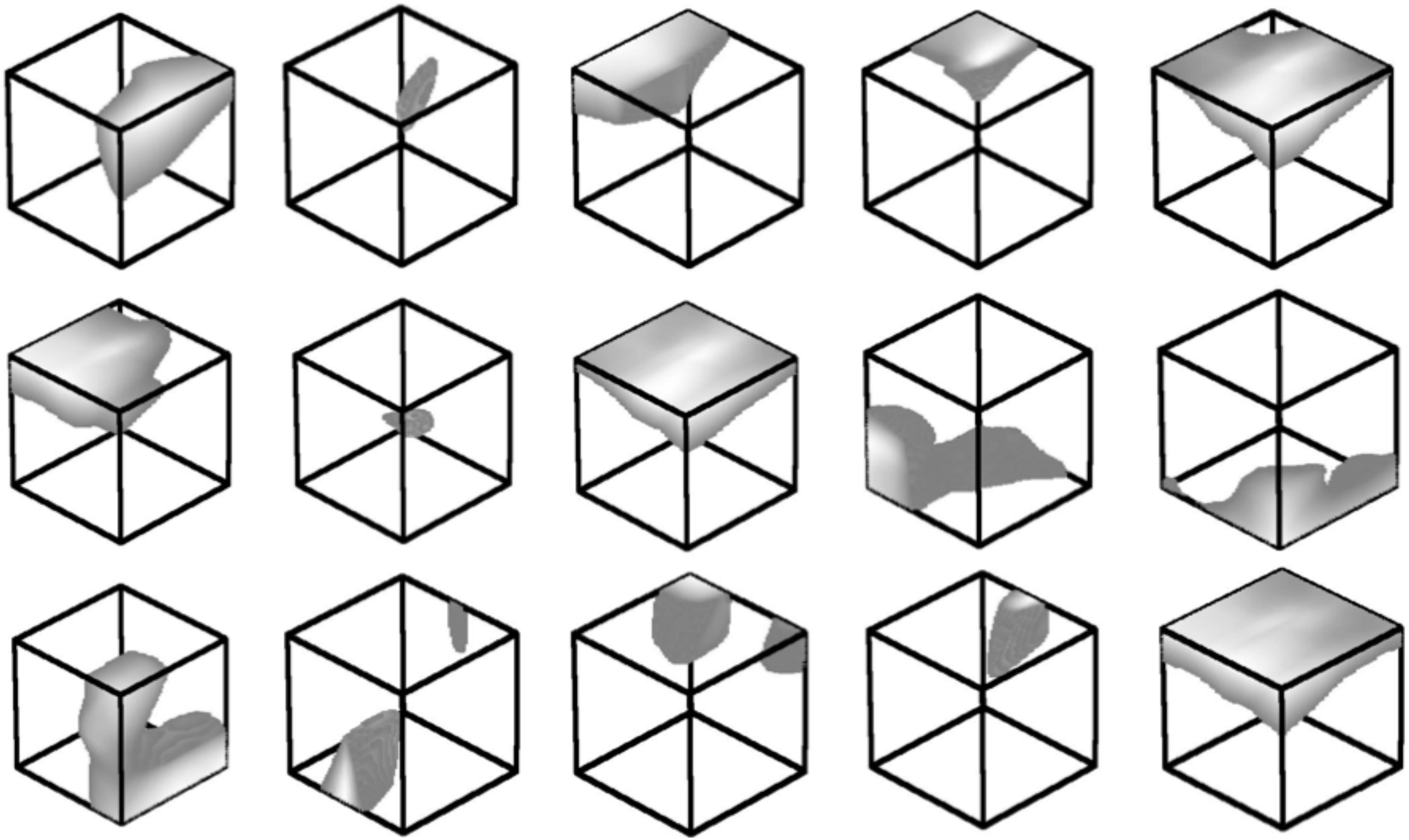
Visualizing Point Functions



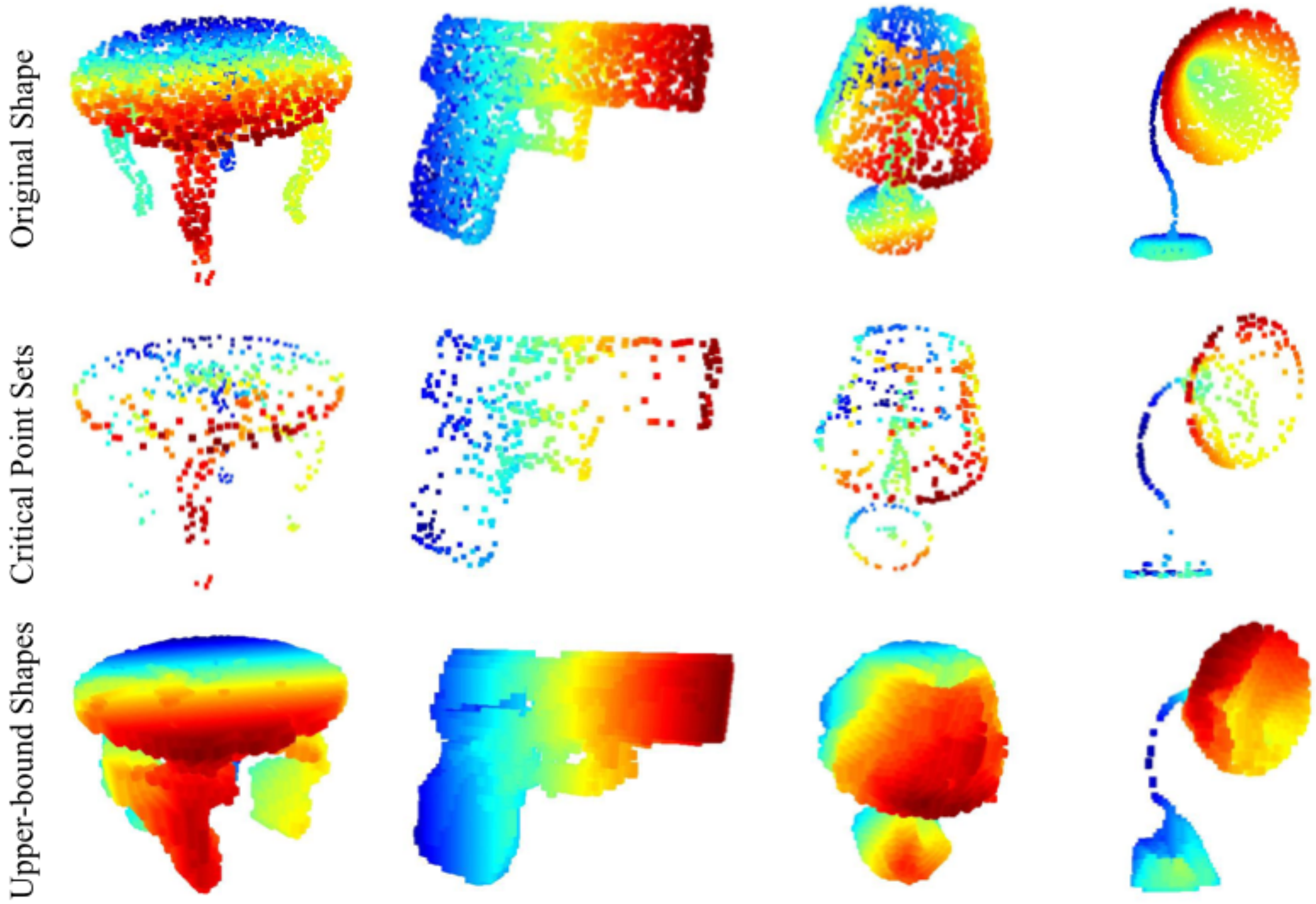
Which input point will activate neuron j?

Find the top-K points in a dense volumetric grid that activates neuron j.

Visualizing Point Functions



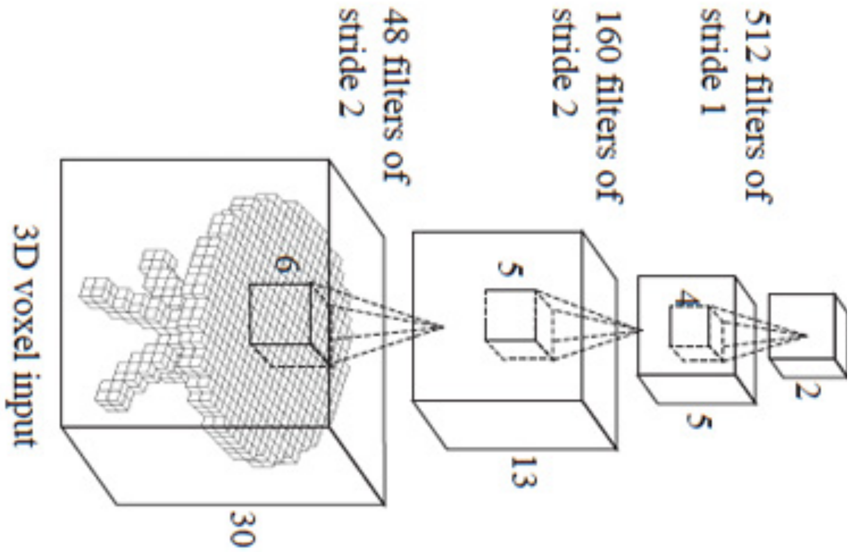
Visualizing Global Point Cloud Features



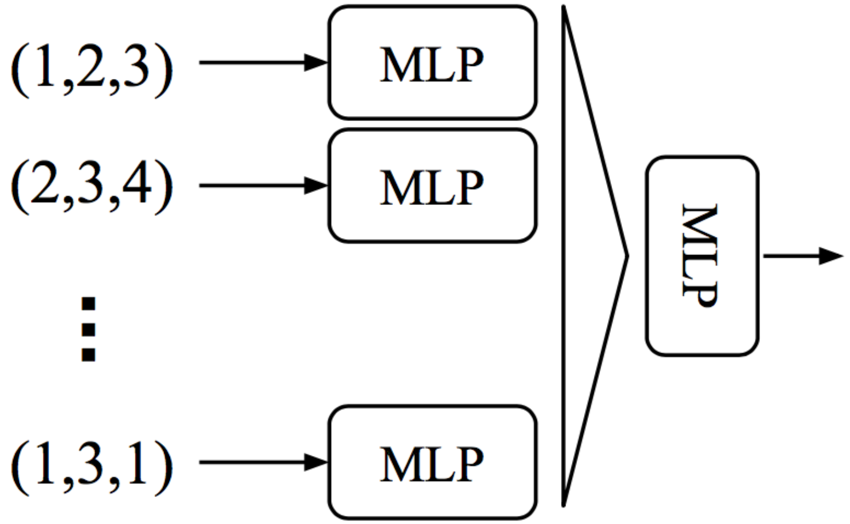
Limitations of PointNet

<u>Hierarchical</u> feature learning
<u>Multiple levels</u> of abstraction

<u>Global</u> feature learning
Either <u>one</u> point or <u>all</u> points



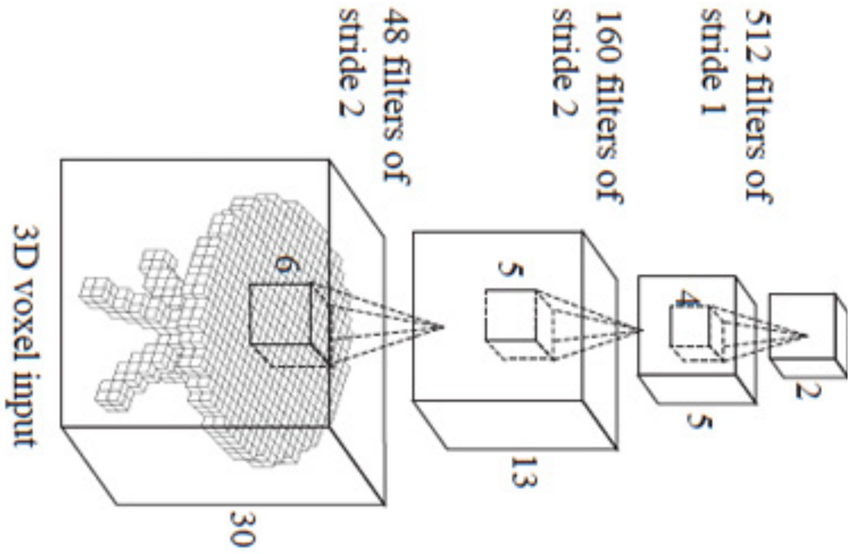
3D CNN (Wu et al.)



PointNet (vanilla) (Qi et al.)

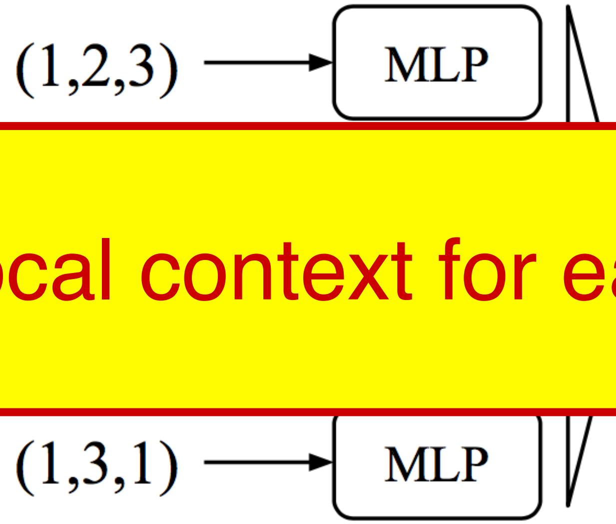
Limitations of PointNet

<u>Hierarchical</u> feature learning
<u>Multiple levels</u> of abstraction



3D CNN (Wu et al.)

<u>Global</u> feature learning
Either <u>one</u> point or <u>all</u> points

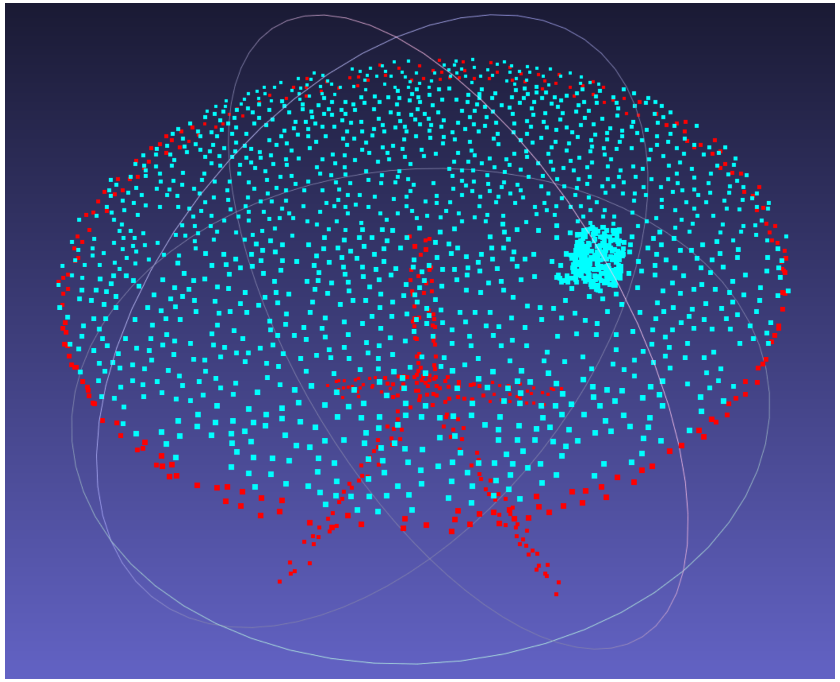


No local context for each point!

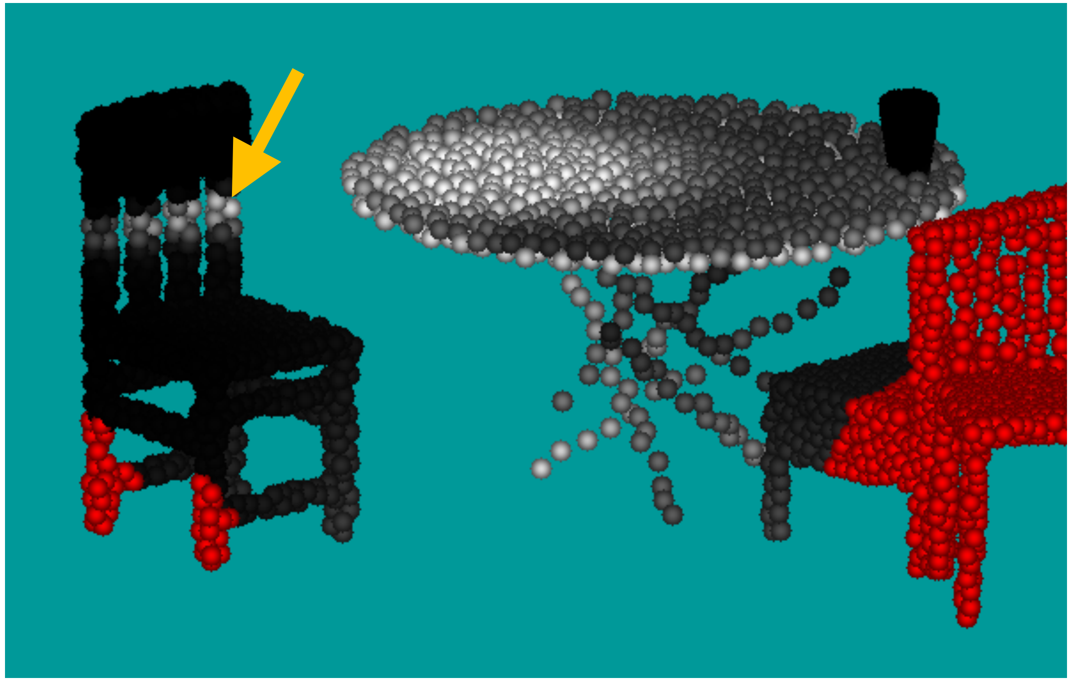
PointNet (vanilla) (Qi et al.)

Limitation of PointNet

Lack of local context => artifacts in segmentation tasks



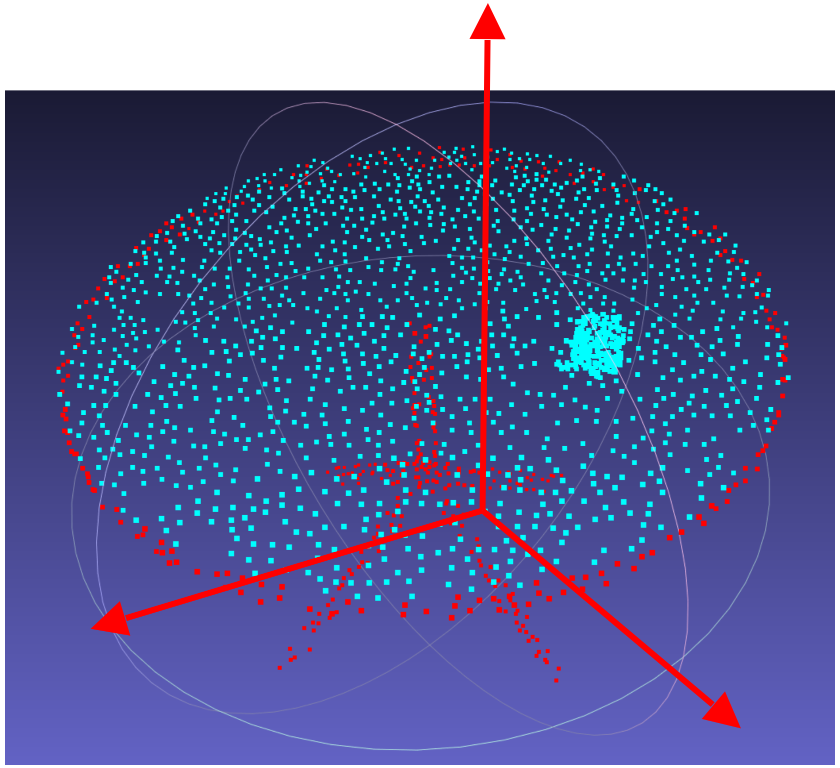
Semantic segmentation of randomly translated table-cup scene

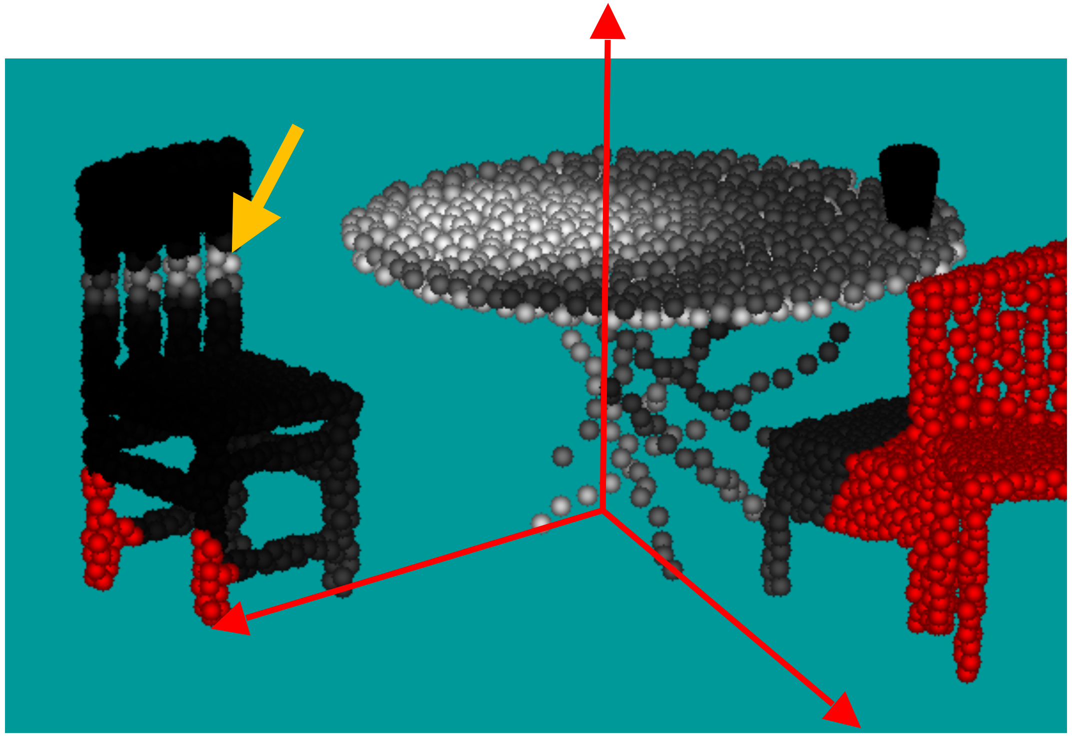


Instance mask prediction in table-chair-cup scene

Limitation of PointNet

Lack of local context => artifacts in segmentation tasks



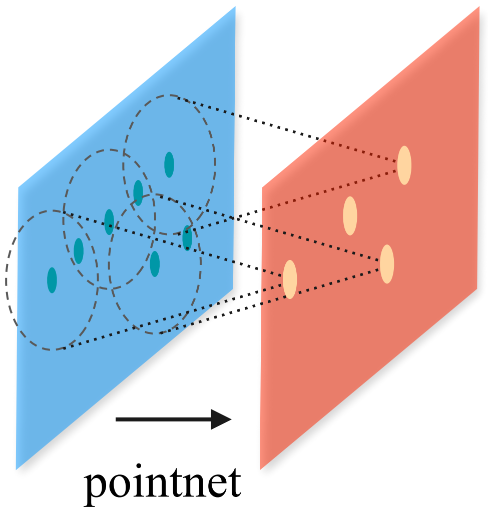


Global feature depends on absolute coordinate. Hard to generalize to unseen scene configurations!

PointNet++

Use pointnet in local regions, aggregate local features by pointnet again

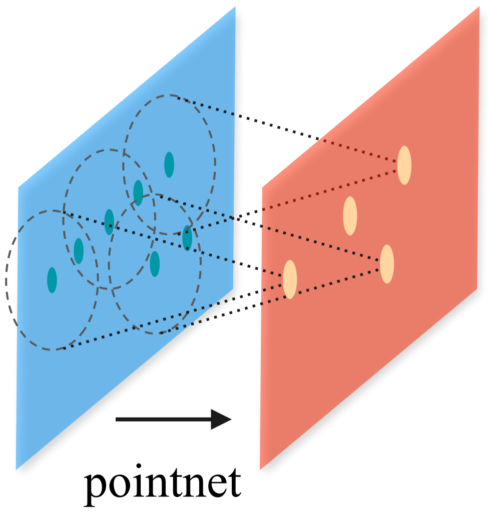
=> Hierarchical feature learning



PointNet++

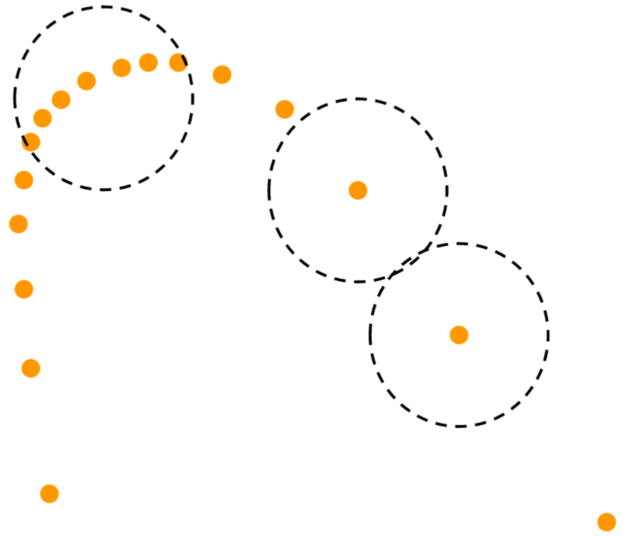
Use pointnet in local regions, aggregate local features by pointnet again

=> Hierarchical feature learning

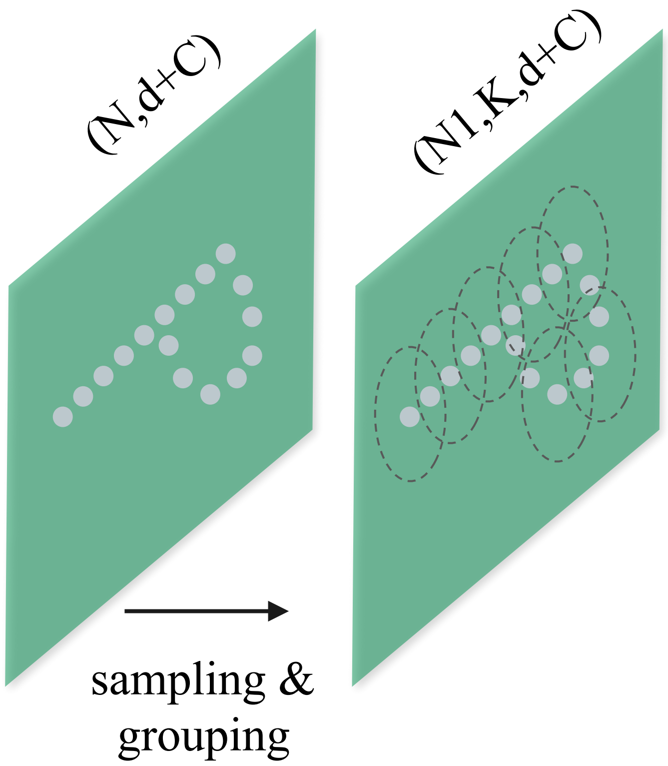


Common issue: inconsistent sampling density

=> Robust to non-uniform sampling density



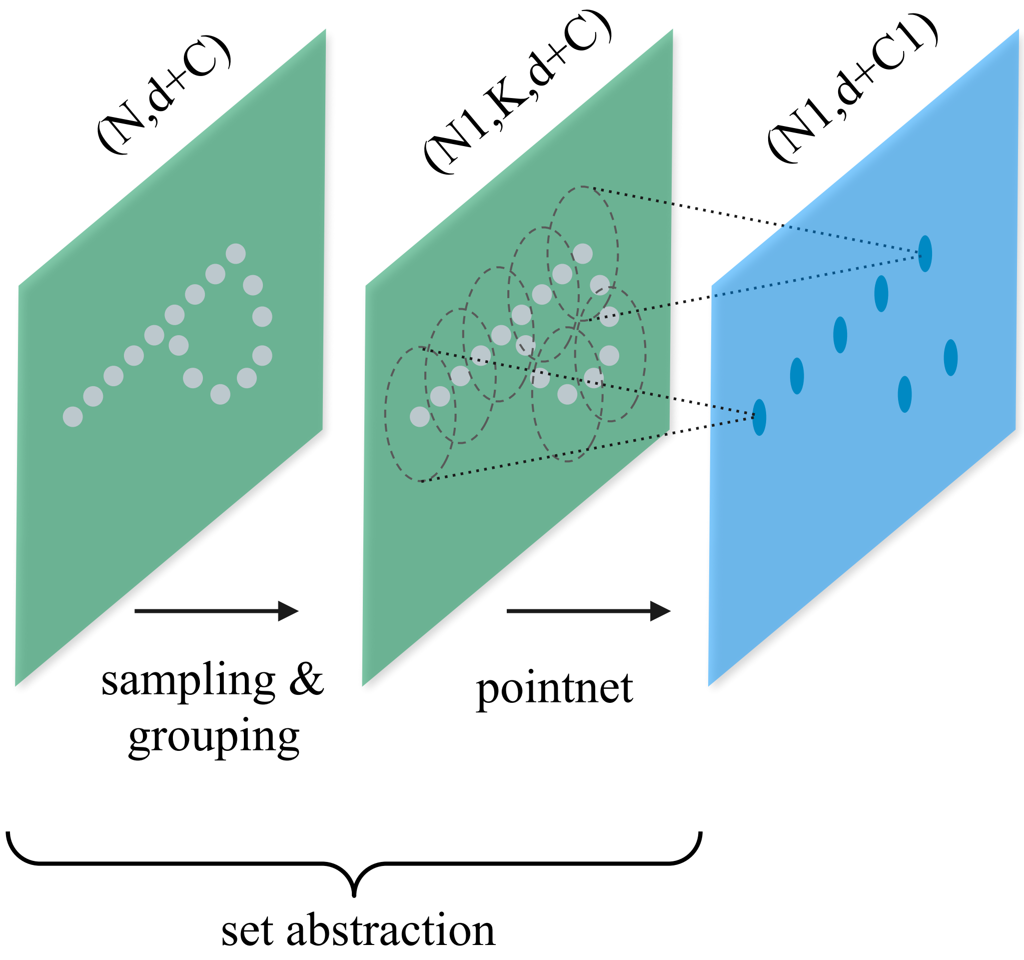
Hierarchical Point Set Feature Learning



Sampling: Farthest Point Sampling (FPS)

Grouping: radius based ball query

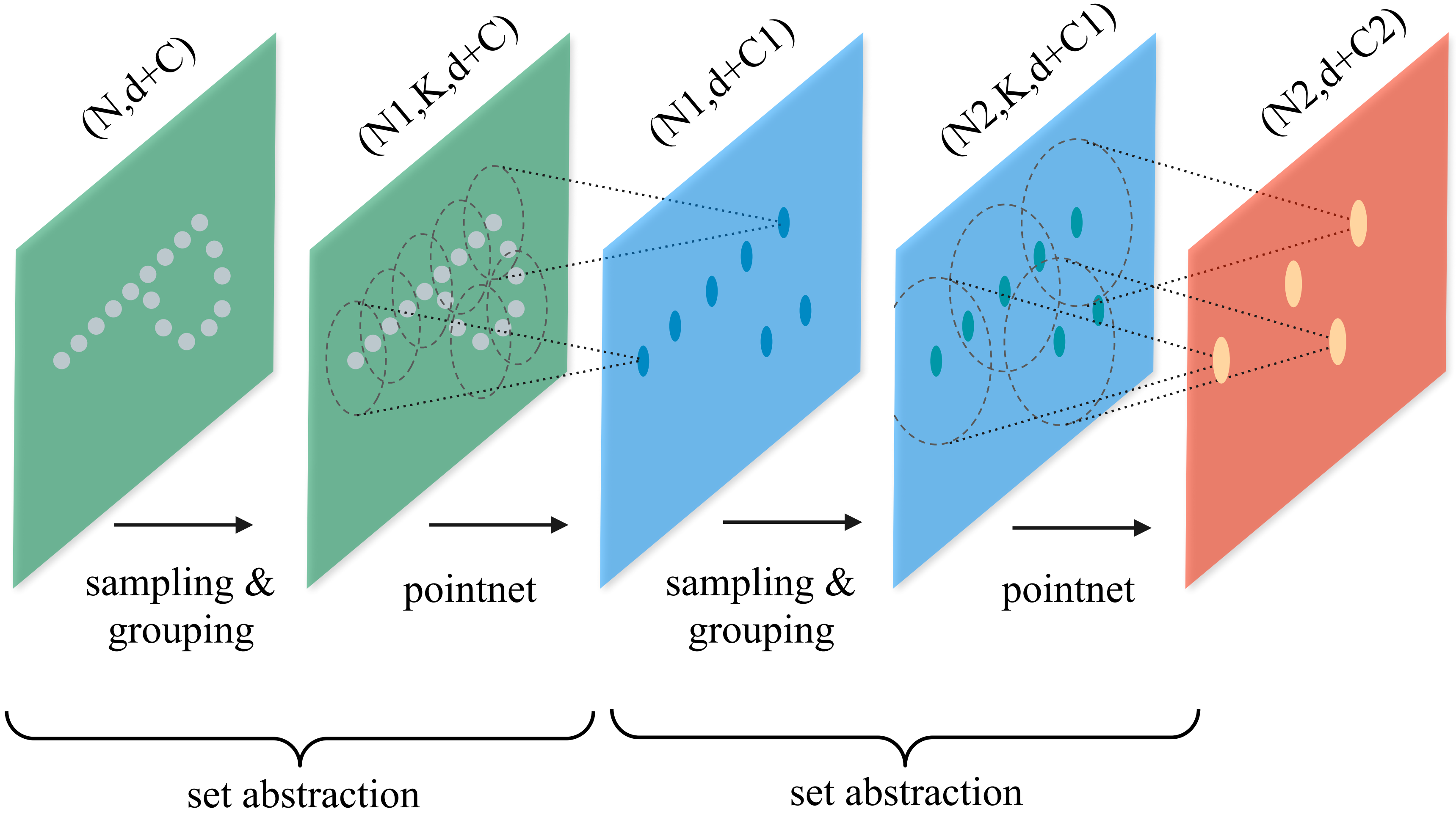
Hierarchical Point Set Feature Learning



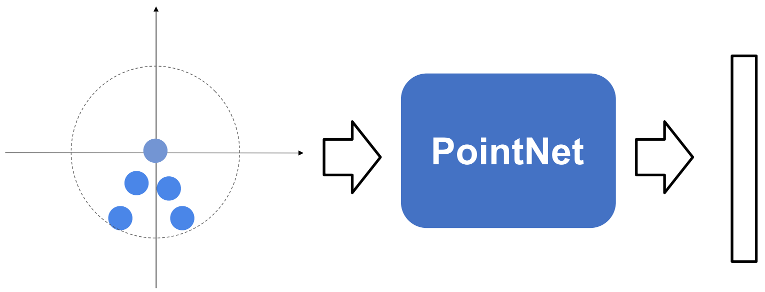
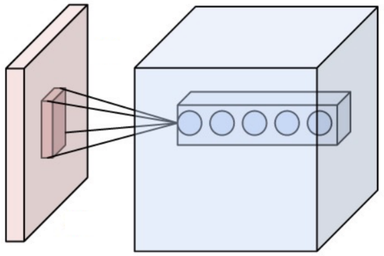
Shared pointnet applied in each local region using local coord.

Hierarchical Point Set Feature Learning

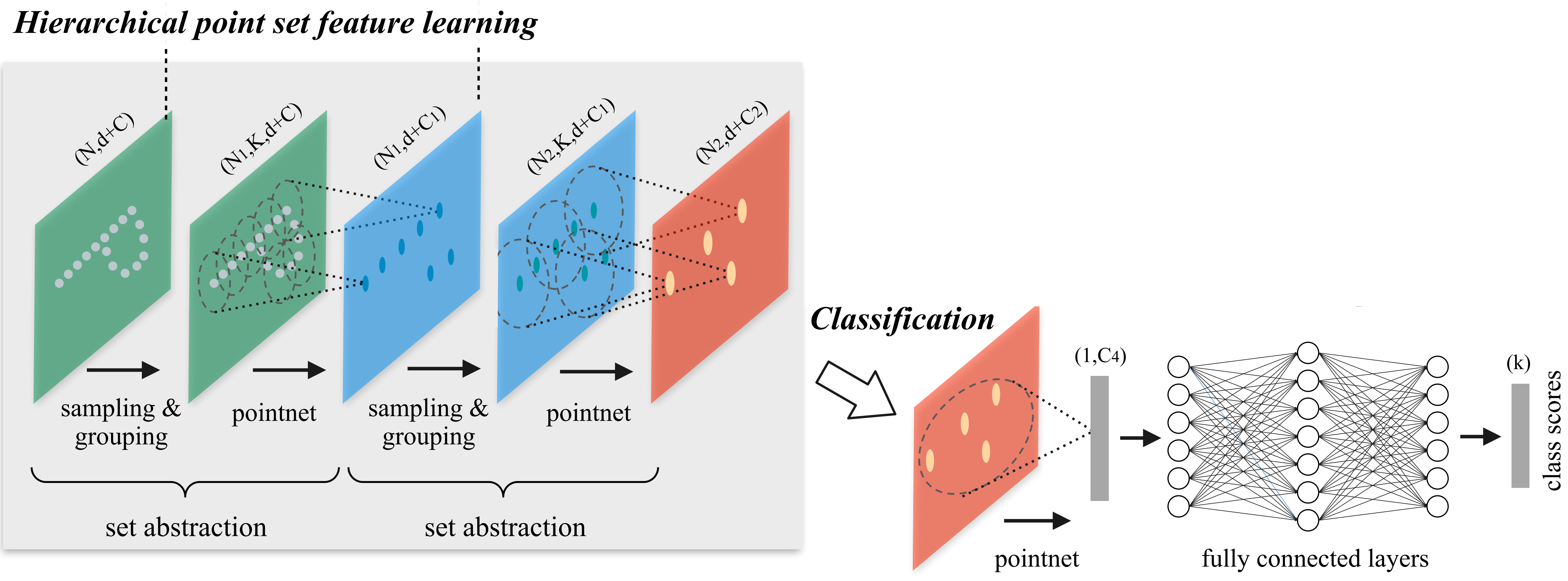
Recursively apply pointnet:



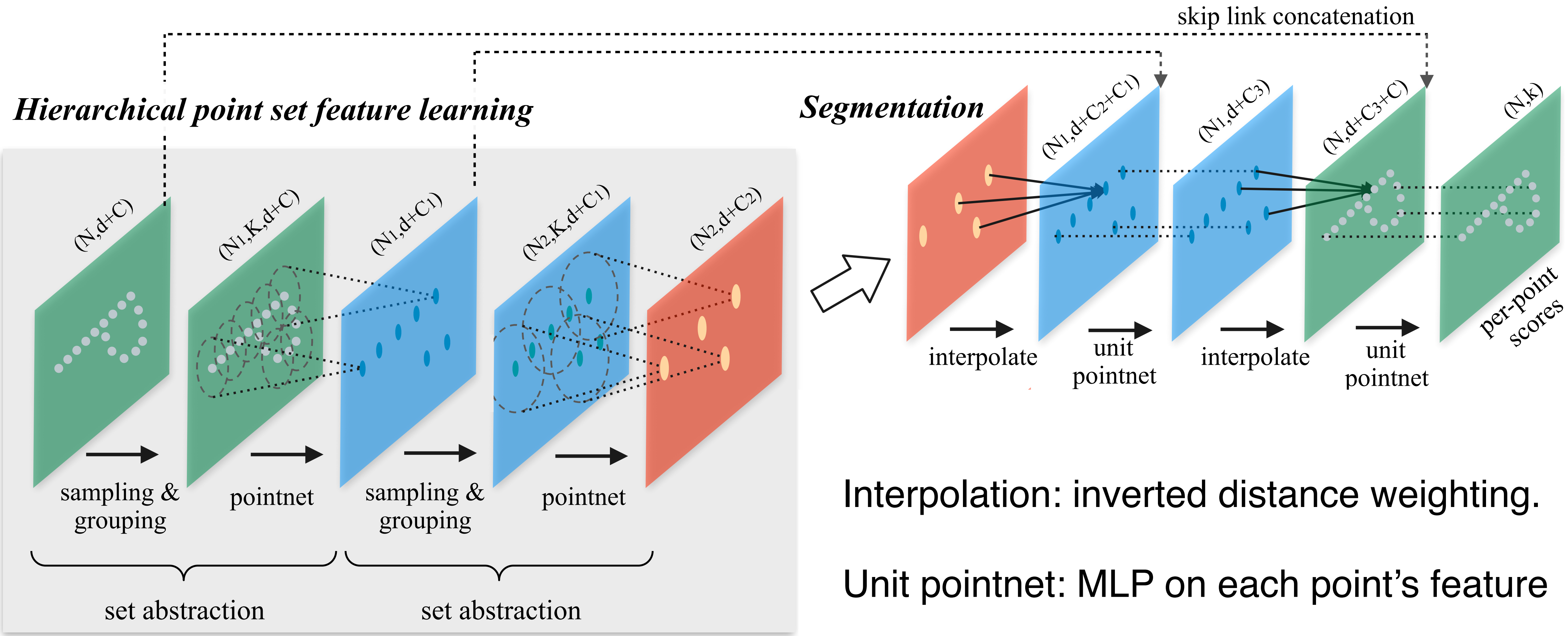
PointNet layer v.s. Convolution layer

	PointNet layer	Convolution layer
		
Input:	Point set	Dense array
Operation:	PointNet (order invariant)	Convolution (index-ordered)
Neighborhood:	Radius ball query (varying #points)	Array index (fixed #pixel/voxel)

PointNet++ for Classification and Segmentation



PointNet++ for Classification and Segmentation



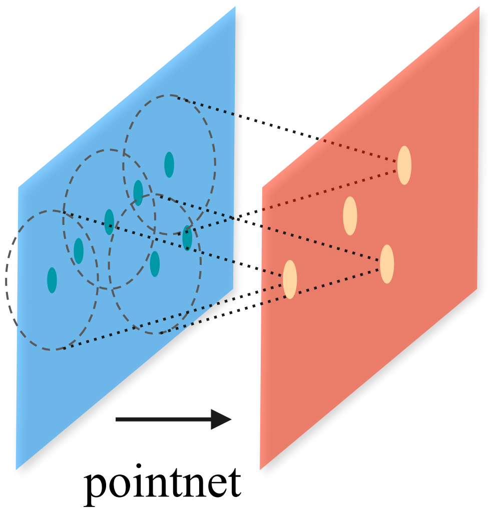
Interpolation: inverted distance weighting.

Unit pointnet: MLP on each point's feature

PointNet++

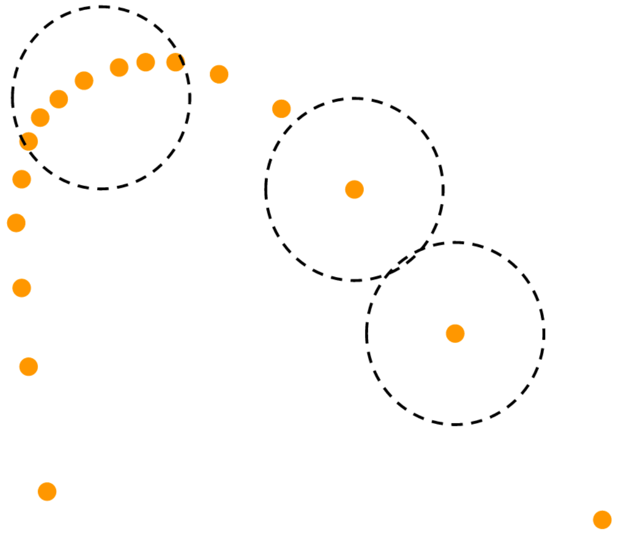
Use pointnet in local regions, aggregate local features by pointnet again

=> Hierarchical feature learning



Common issue: inconsistent sampling density

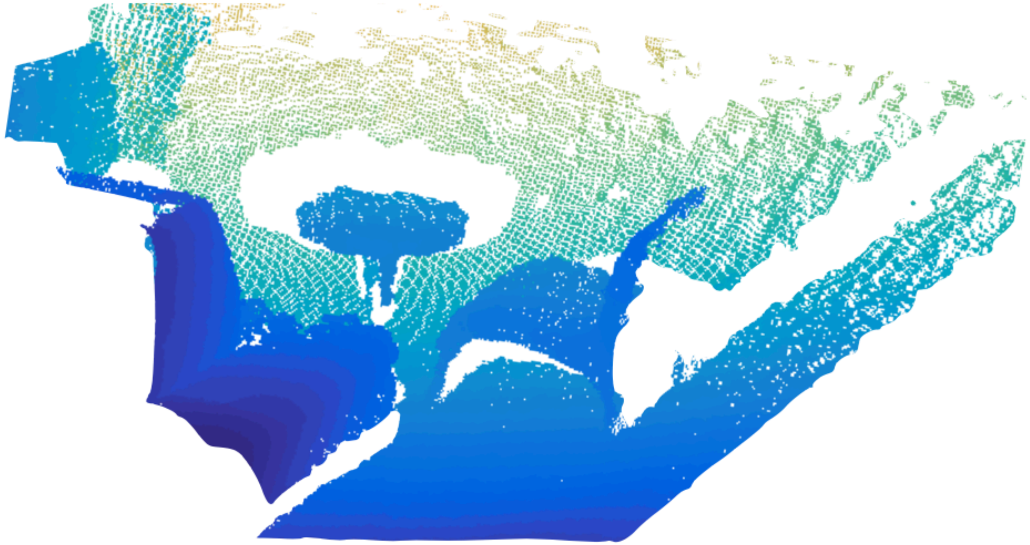
=> Robust to non-uniform sampling density



Non-uniform Sampling Density

Density variation is a common issue of 3D point cloud

- perspective effect, radial density variation, motion etc.



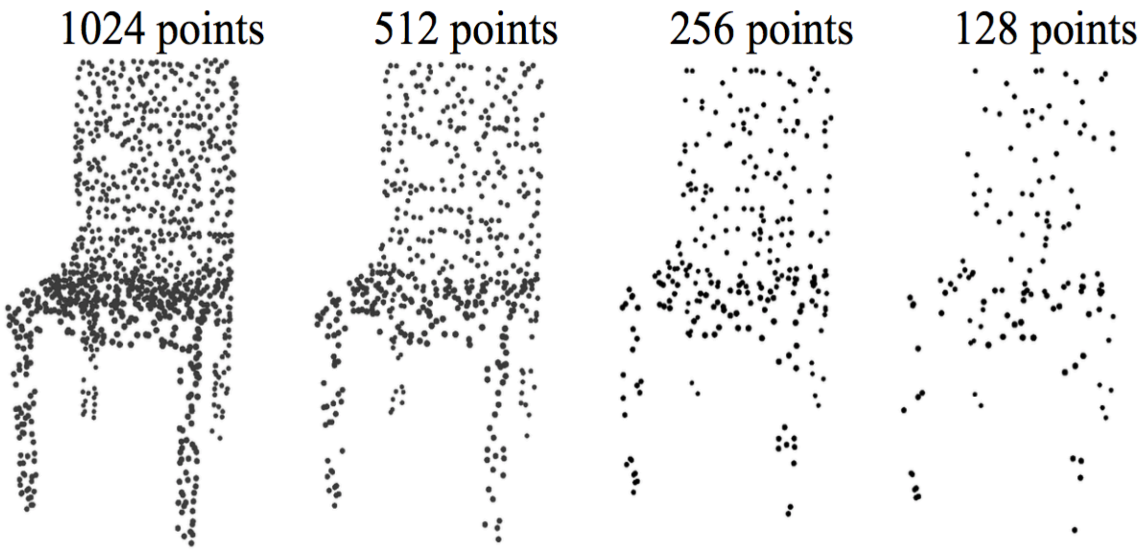
Density variation affects hierarchy

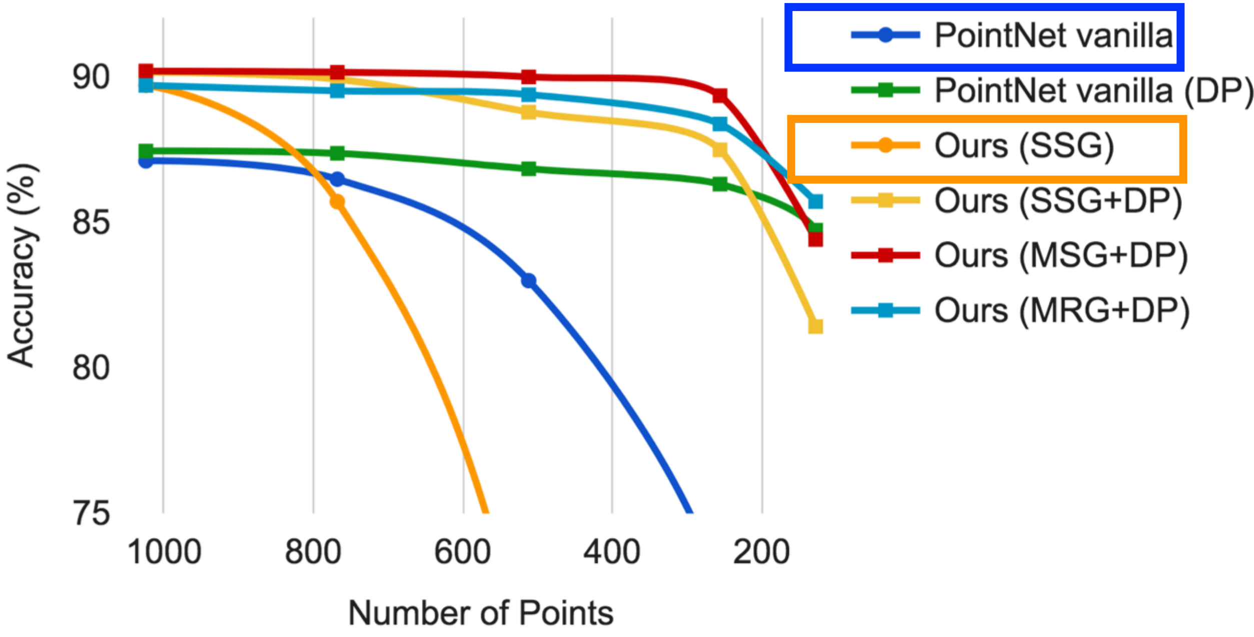
- In CNN, small kernels are usually better

Karen Simonyan & Andrew Zisserman, Very Deep Convolutional Networks for Large-scale Image Recognition, ICLR2015

- Is it also true for point cloud learning?

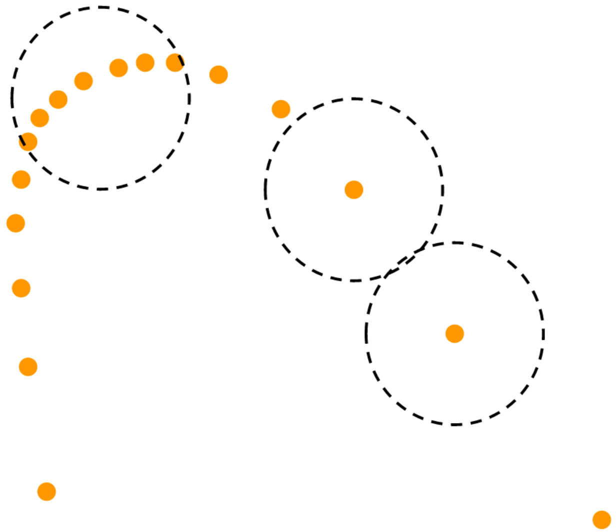
Density variation affects hierarchy

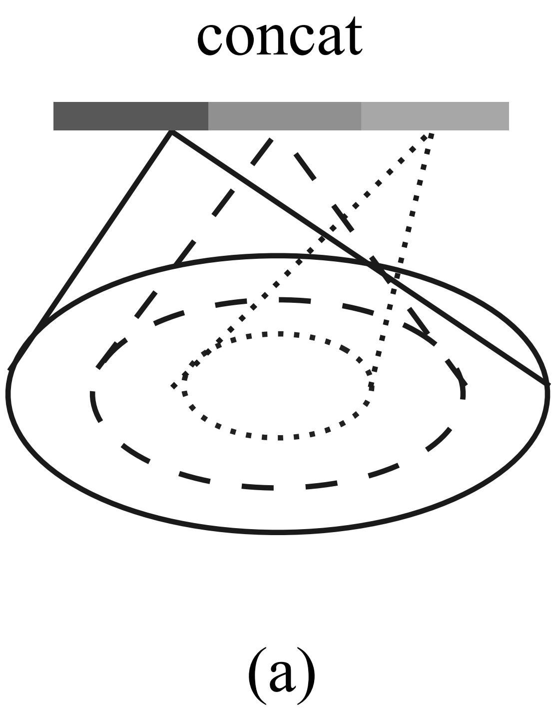


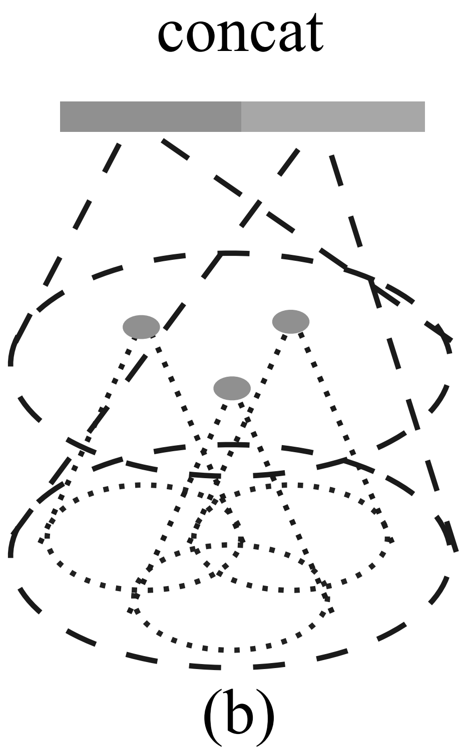


We expect to trust regions with high sampling density; use larger field of view for sparse regions.

Non-uniform Sampling Density



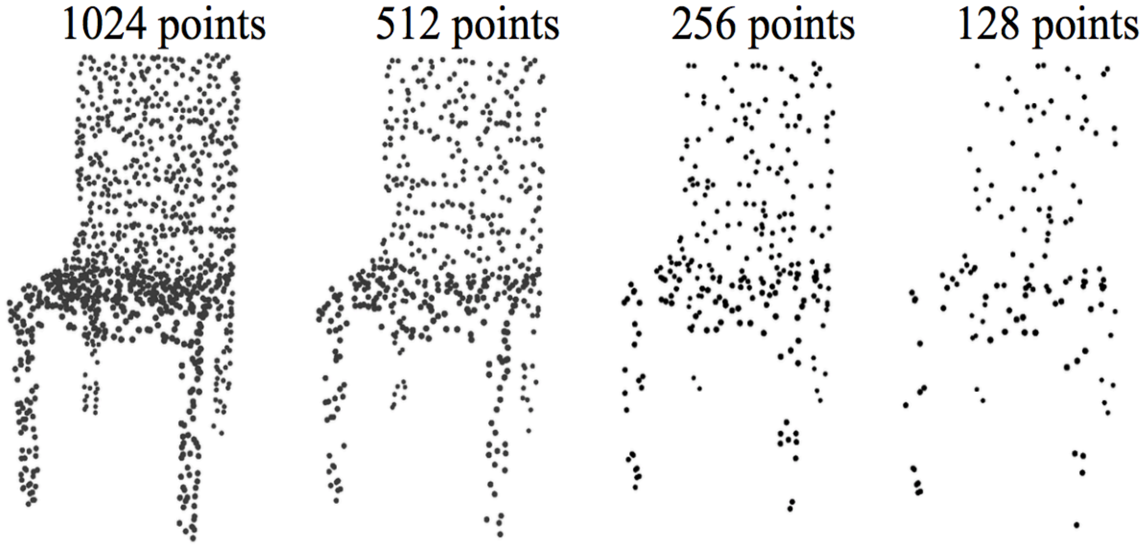


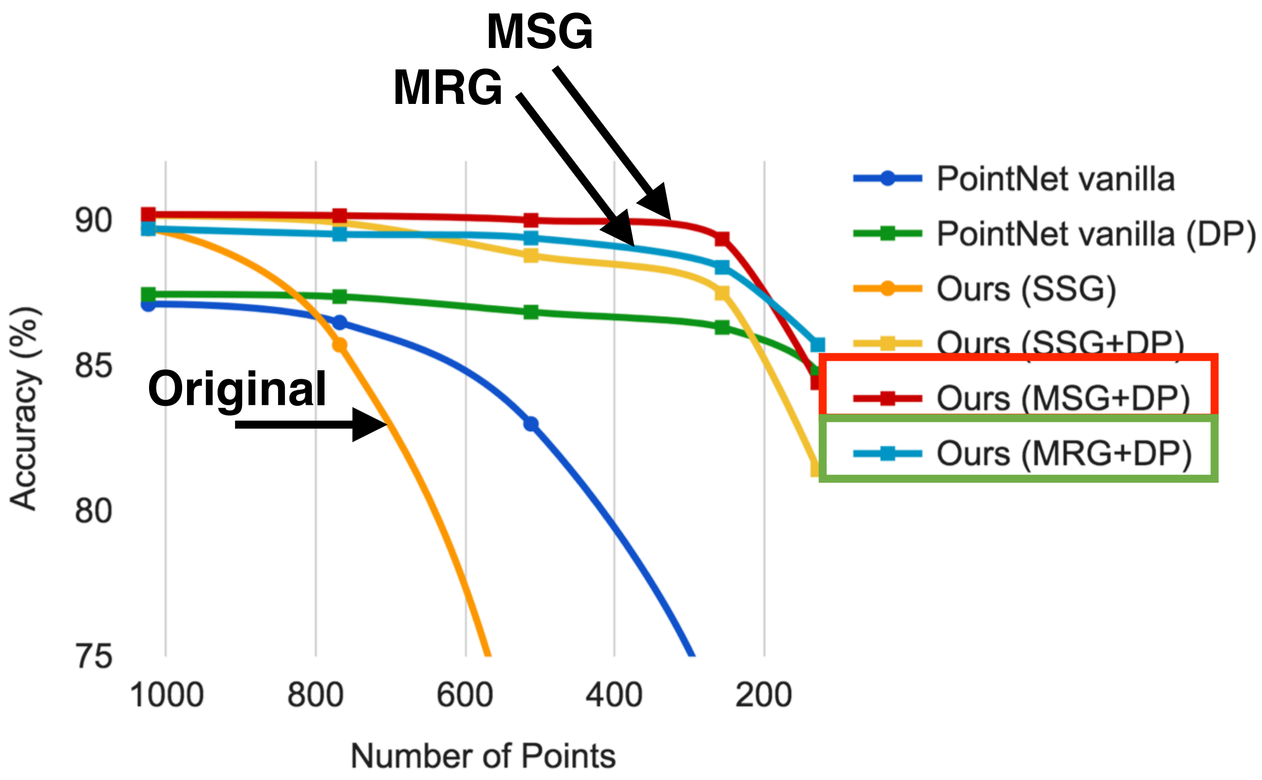


Multi-scale grouping (MSG) Multi-res grouping (MRG)

During Training: input point dropout with random dropout ratio

Robust learning under varying sampling density





PointNet++ Results: Classification

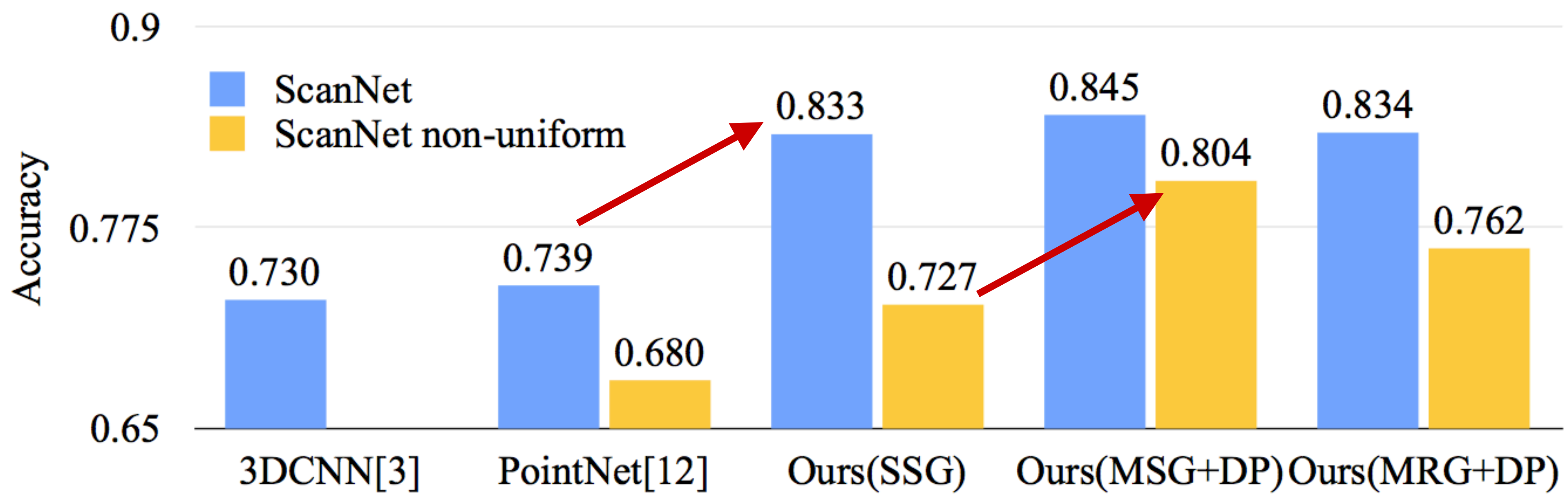
By hierarchical feature learning, we can achieve even better results than the original PointNet. By using normals, we can even beat best view-CNN based method.

Method	Input	Accuracy (%)
Subvolume [20]	vox	89.2
MVCNN [25]	img	90.1
PointNet (vanilla) [19]	pc	87.2
PointNet [19]	pc	89.2
Ours	pc	90.7
Ours (with normal)	pc	**91.9**

Table 2: ModelNet40 shape classification.

PointNet++ Results: Scene Parsing

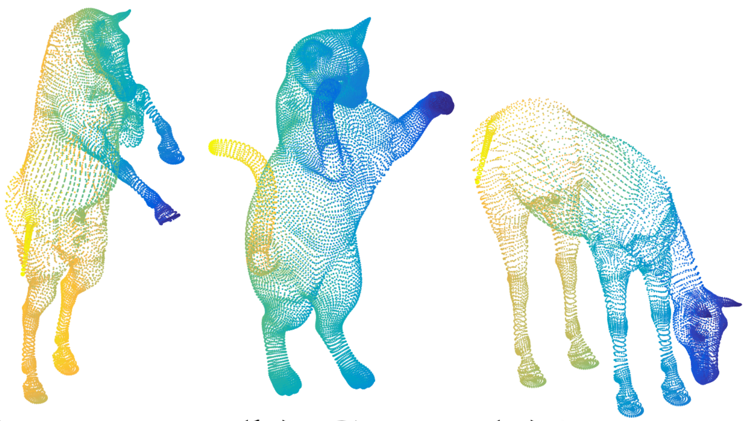
Much better generalizability in scene parsing. MSG layer very helpful for point cloud with non-uniform densities.



PointNet++ Results: Non-Euclidean Space

For organic shape recognition, PointNet++ can generalize to non-Euclidean space:

- intrinsic point features (HKS, WKS, Gaussian curvature)
- intrinsic distance metric (geodesic)



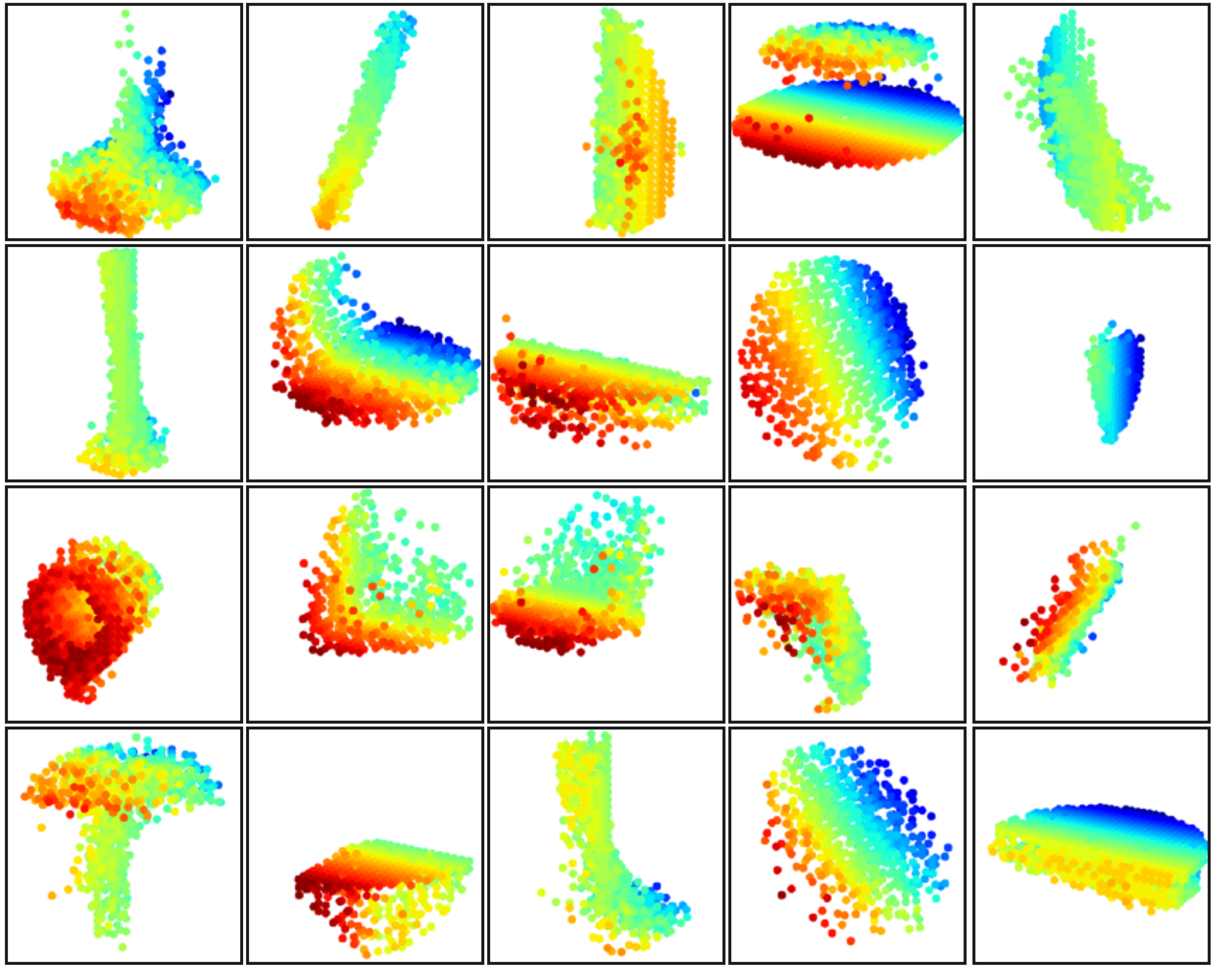
(a) Horse (b) Cat (c) Horse

	Metric space	Input feature	Accuracy (%)
DeepGM [13]	-	Intrinsic features	93.03
Ours	Euclidean	XYZ	60.18
	Euclidean	Intrinsic features	94.49
	Non-Euclidean	Intrinsic features	**96.09**

Table 3: SHREC15 Non-rigid shape classification.

PointNet++ Feature Visualization

Trained on ModelNet40 (mostly furniture) we see structures of planes, double planes, lines, corners etc.



First layer patterns

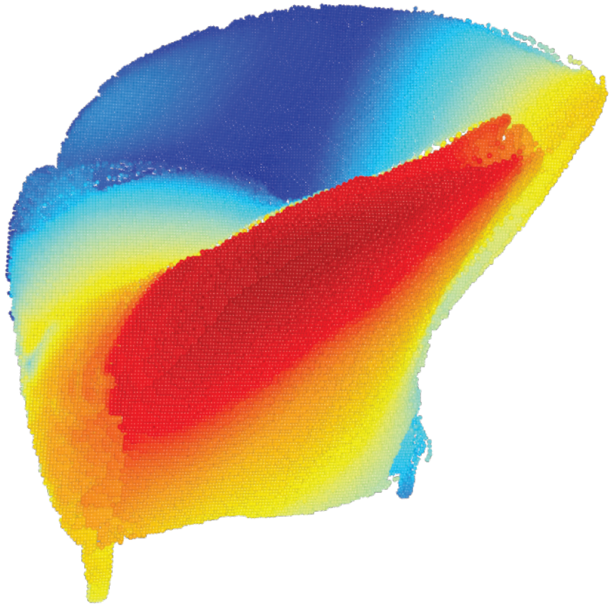
Agenda

- Deep learning on regular structures

- Deep learning on meshes

- **Deep learning on point cloud and parametric models**
  - Point cloud analysis
  - **Point cloud synthesis**
  - Primitive-based shapes

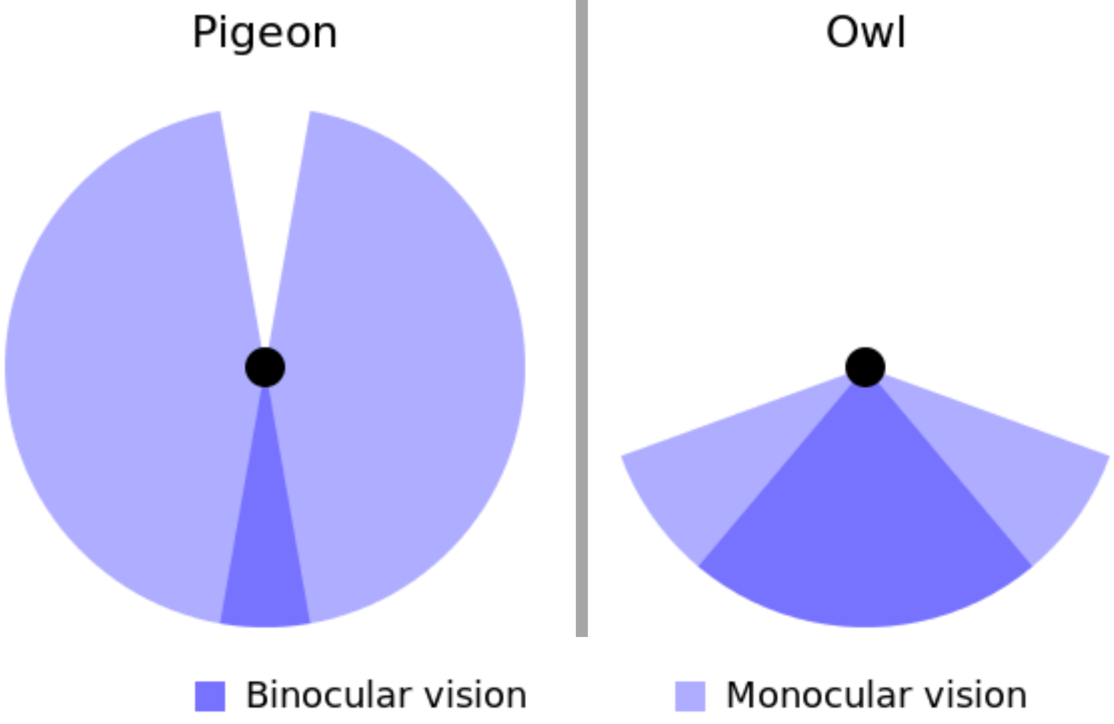
Task: 3D reconstruction from a single image



Monocular vision

a typical prey

a typical predator

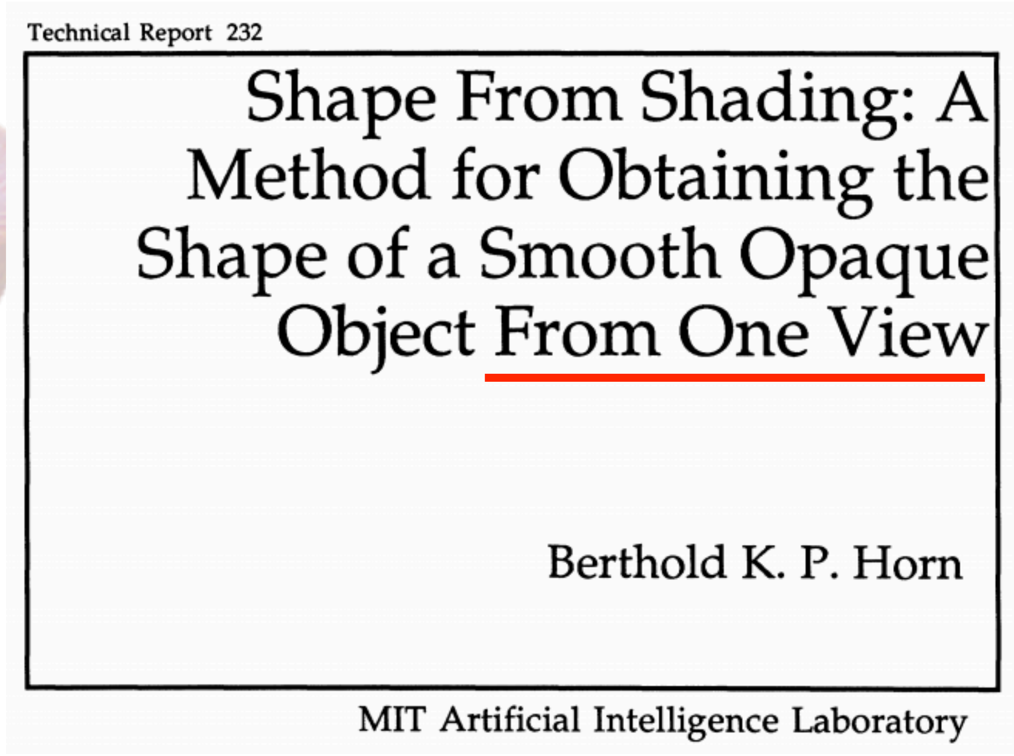


Cited from https://en.wikipedia.org/wiki/Binocular_vision

Monocular 3D perception by human

A core problem of computer vision

’1970



Technical Report 232

Shape From Shading: A Method for Obtaining the Shape of a Smooth Opaque Object From One View

Berthold K. P. Horn

MIT Artificial Intelligence Laboratory

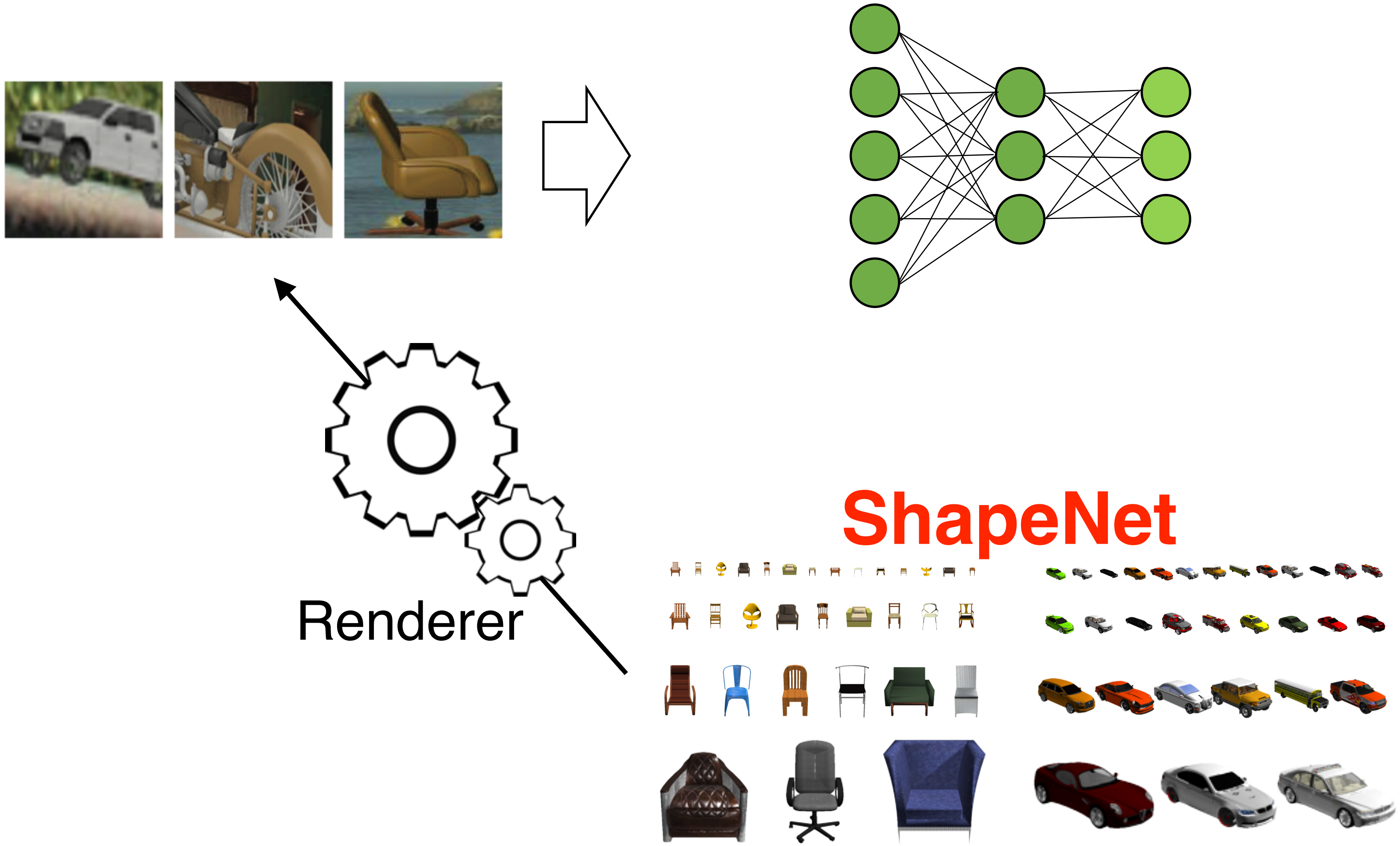
One of Earliest problems of computer vision.
Over 40 years of history!

A data-driven solution: Geometric prior learning

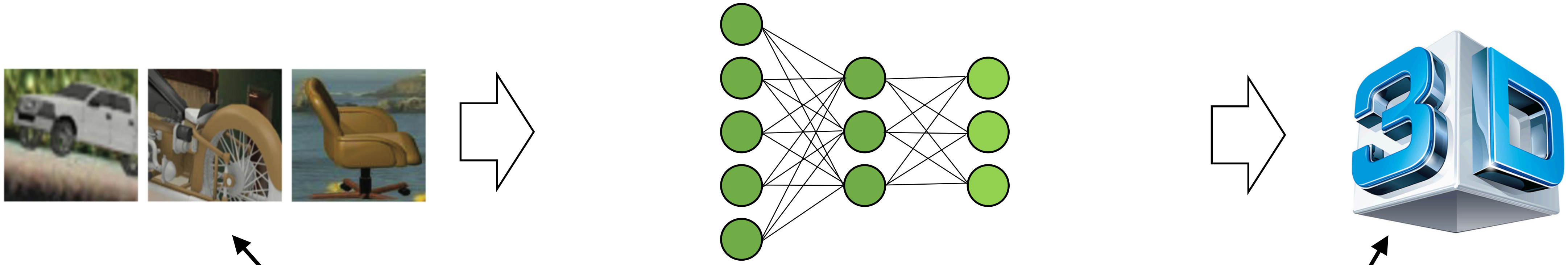
Many 3D objects

Apriori knowledge of 3D geometry

Supervision from “Synthesize for Learning”



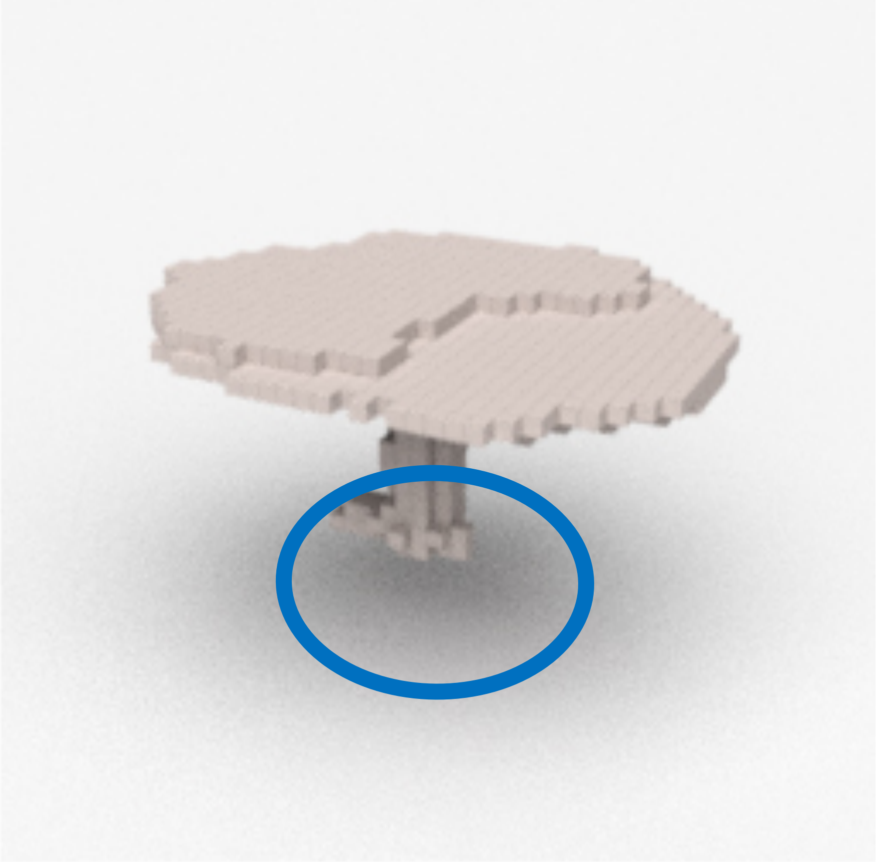
Supervision from “Synthesize for Learning”



- **200K shapes from 2K categories**
- **10M images with groundtruth**

Volumetric upconvolution?

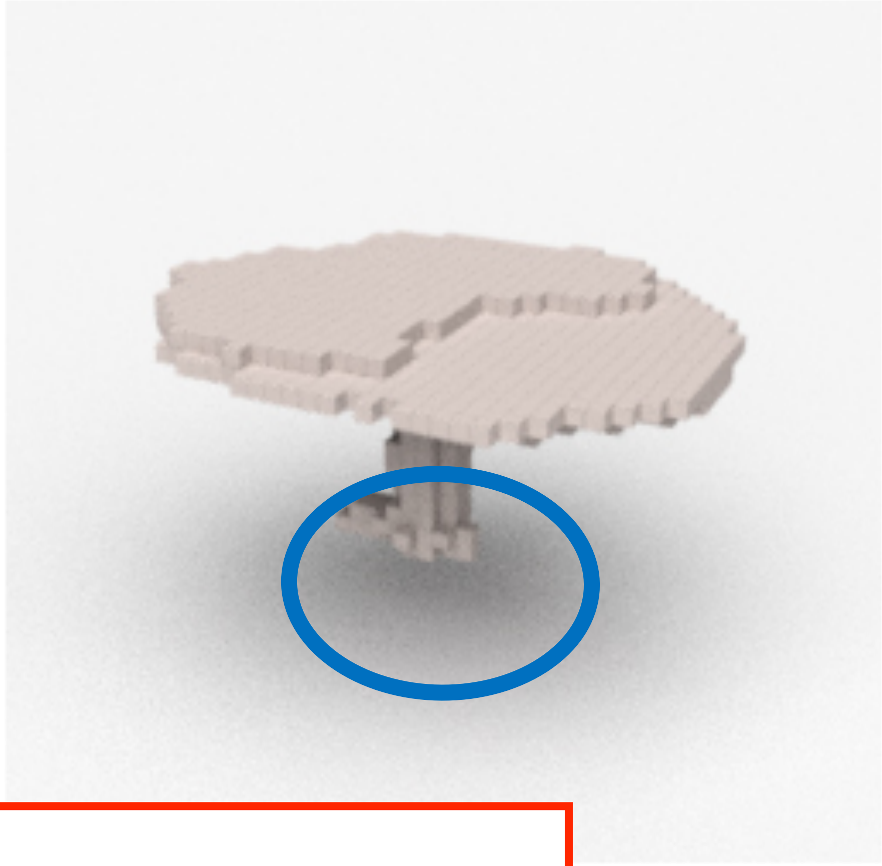
Input image



Cons: Low resolution Error in structure

Volumetric upconvolution?

Input image



Reason:

- Geometric transformation is hard for upconv

Error in structure

Another representation possibility: Point clouds

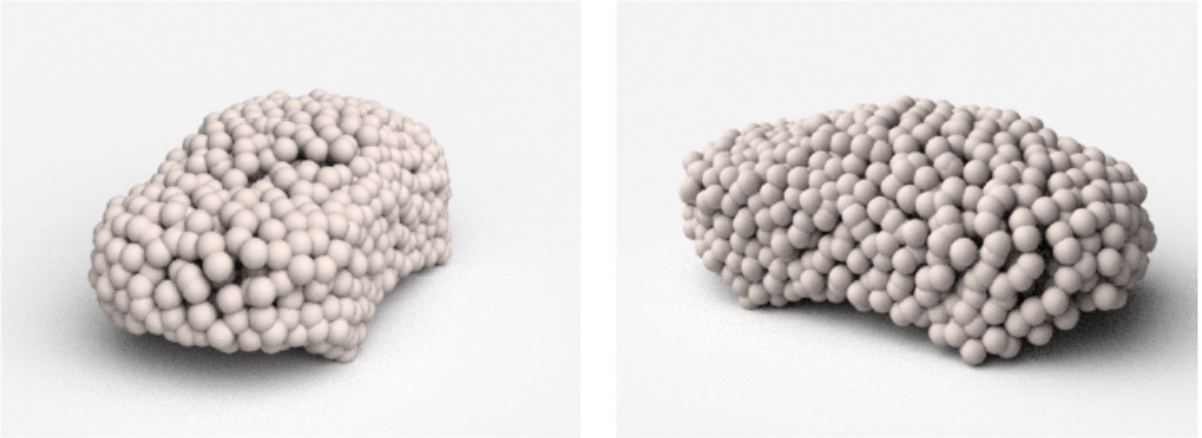
✓ Transformation friendly for networks

? Usable as **network output?**

No prior works in deep learning community!

CVPR '17, Point Set Generation

3D prediction by **point clouds**

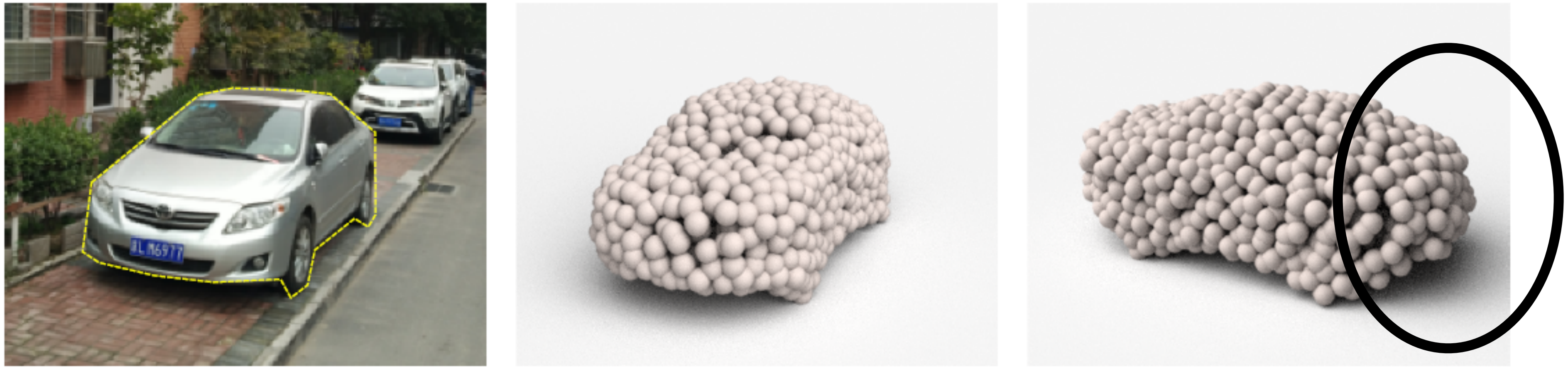


Input

Reconstructed 3D point cloud

Pioneer work on set generation in deep learning

3D prediction by **point clouds**

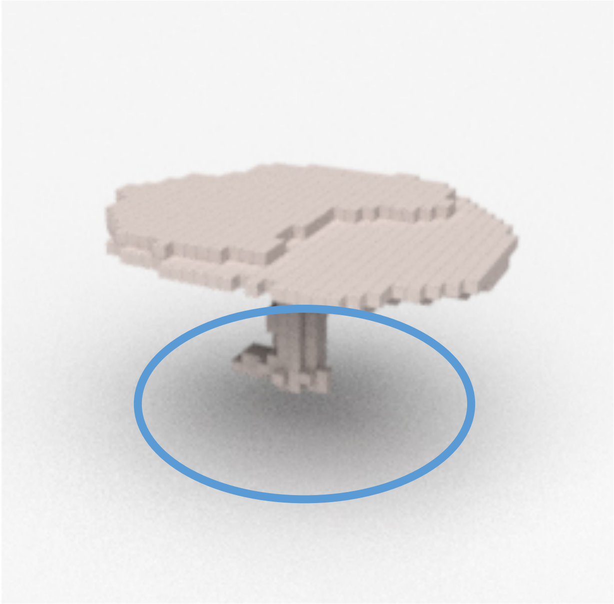


Input Reconstructed 3D point cloud

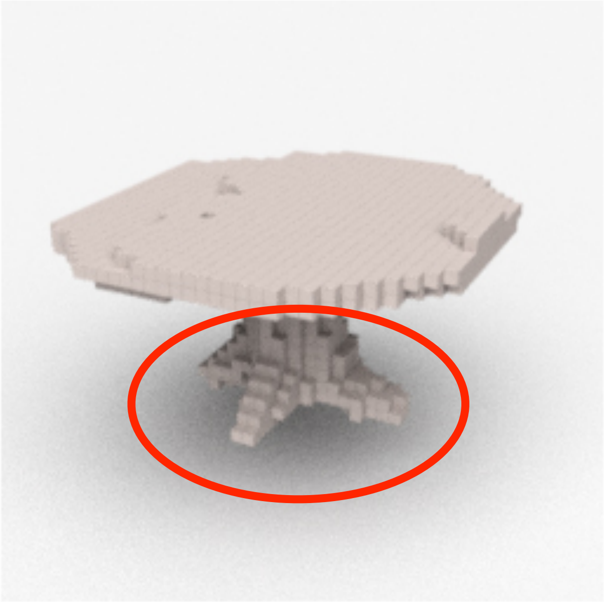
Pioneer work on set generation in deep learning

Comparison to direct 3D volumetric upconvolution

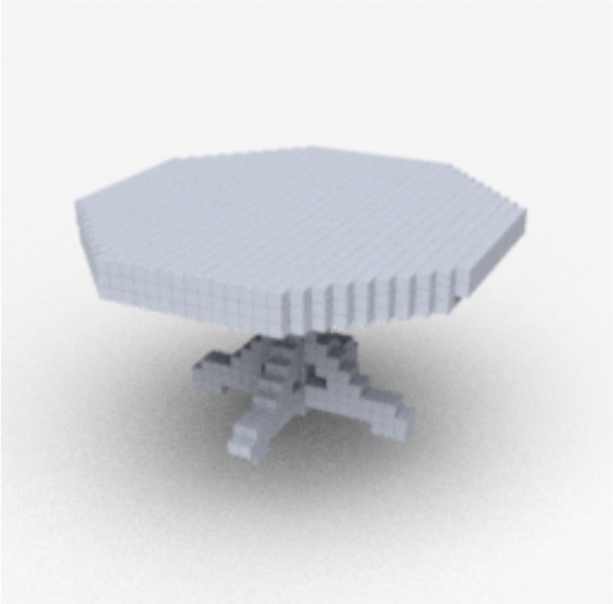
Input



Volumetric upconv
(ECCV 2016, 3D-R2N2)

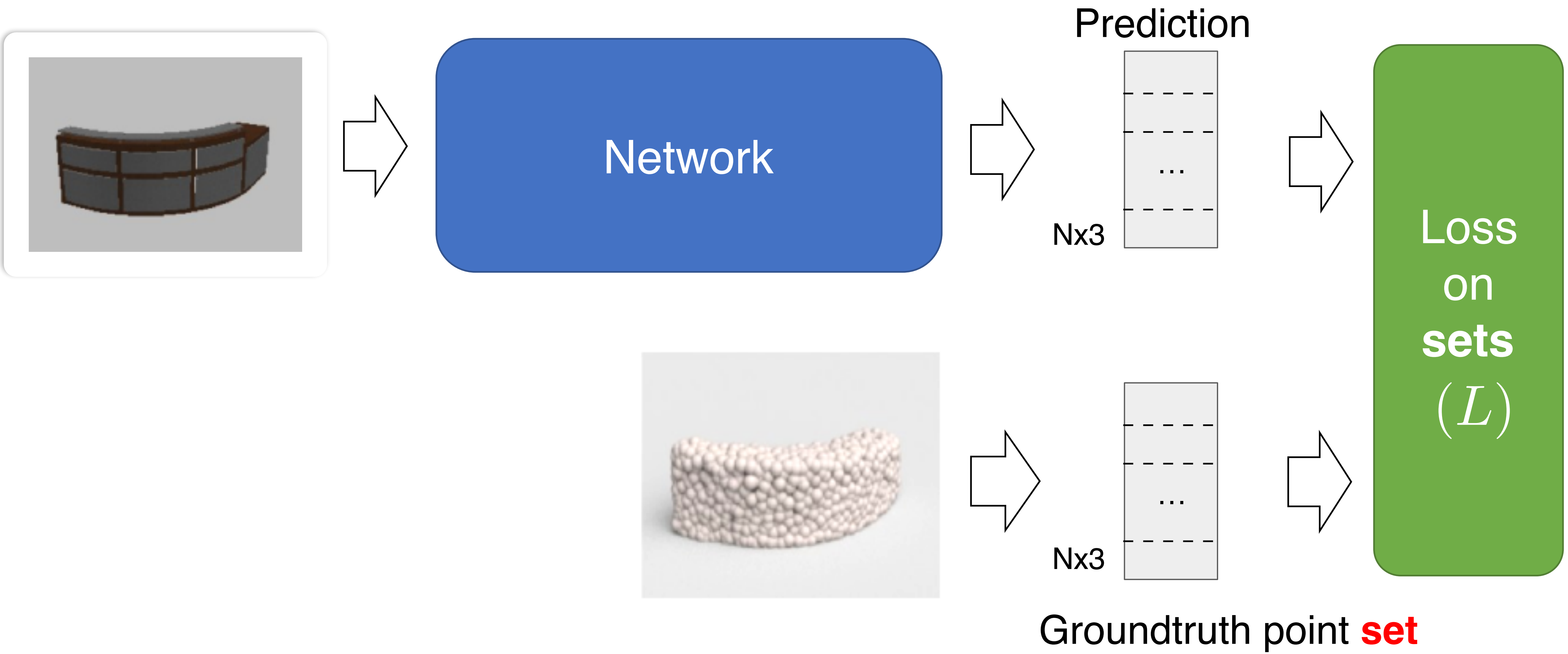


Ours
(post-processed to volumetric)



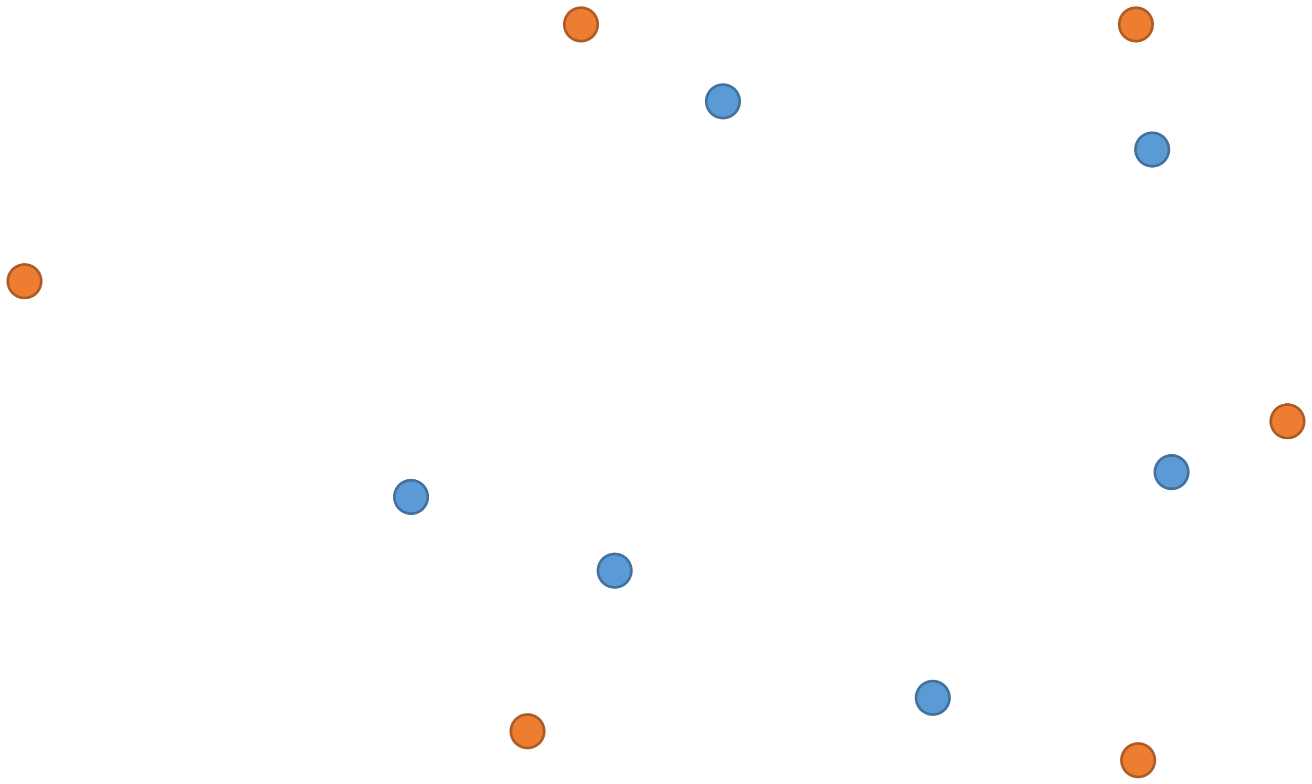
Groundtruth

Pipeline


Network
Prediction
...
Nx3
...
Nx3
Loss on **sets** (L)
Groundtruth point **set**

Set comparison

Given two sets of points, measure their discrepancy



Set comparison

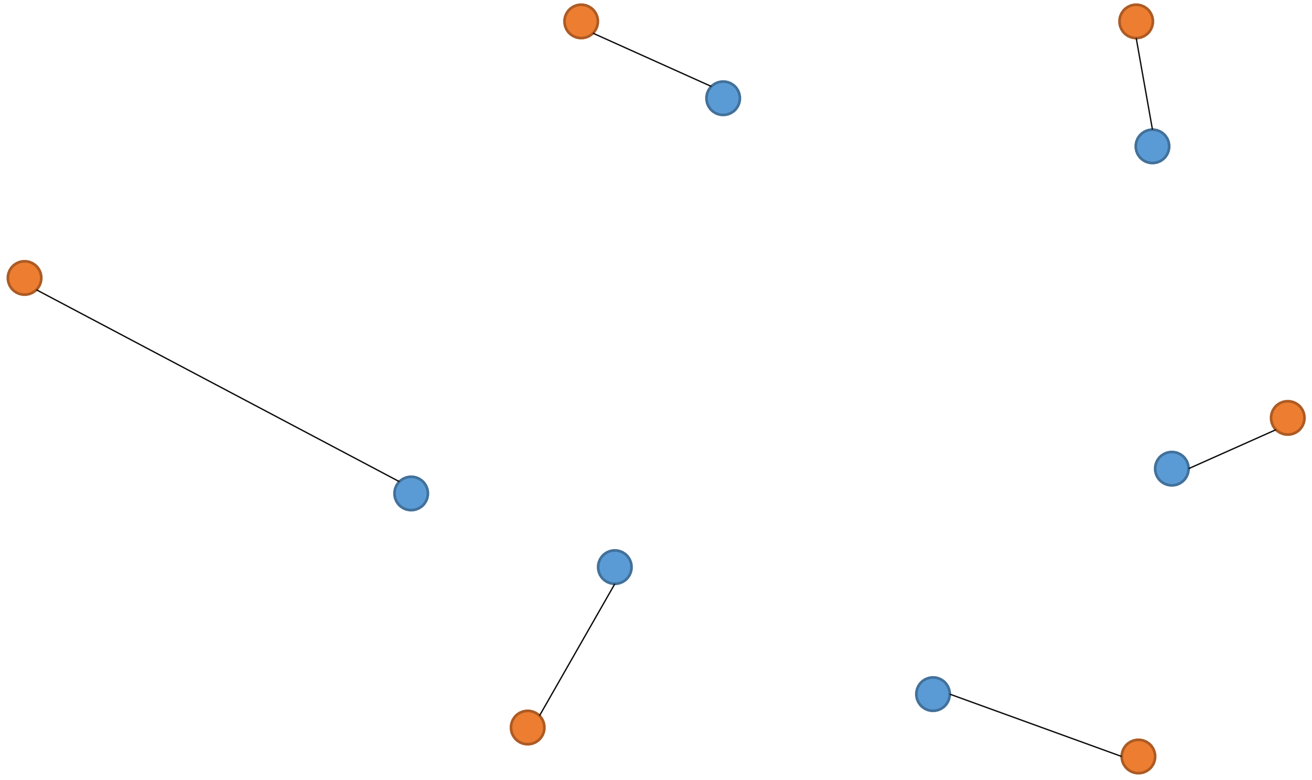
Given two sets of points, measure their discrepancy

Key challenge:

correspondence problem

Correspondence (I): optimal assignment

Given two sets of points, measure their discrepancy

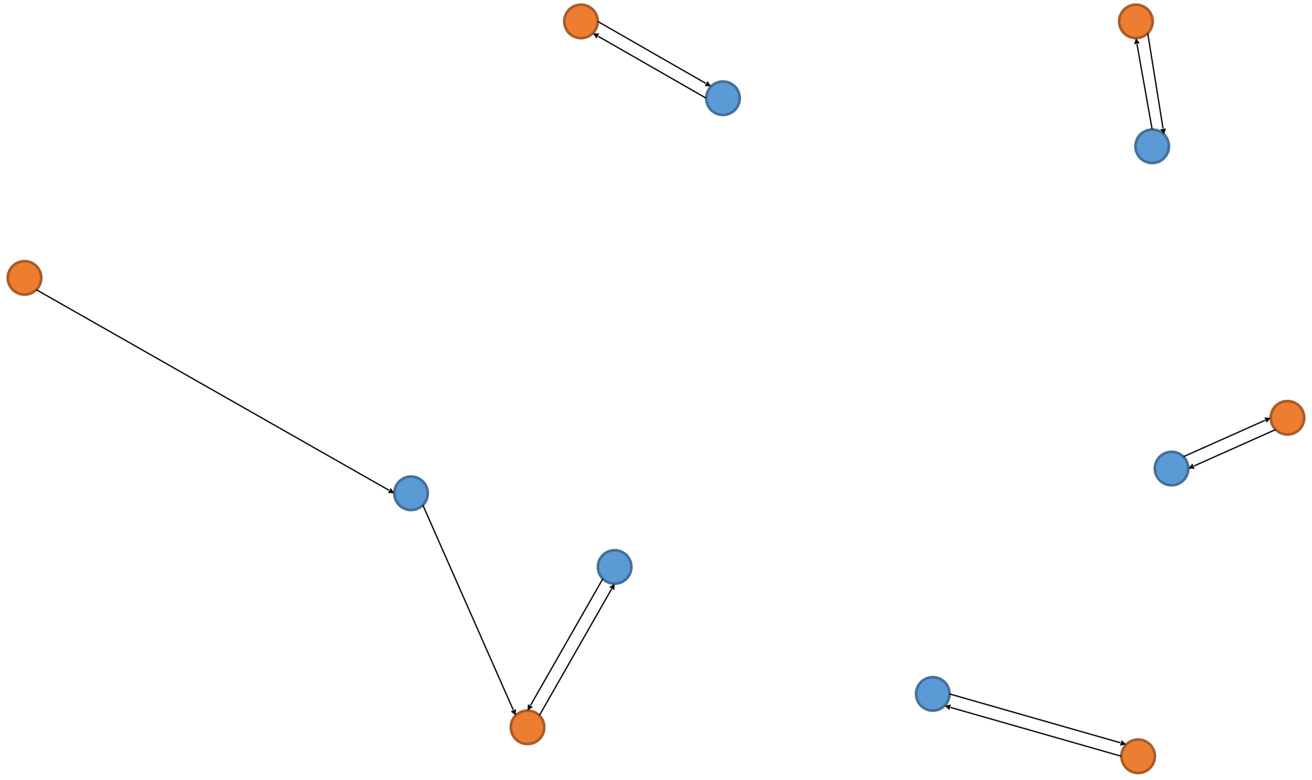


a.k.a Earth Mover's distance (EMD)

$$d_{EMD}(S_1, S_2) = \min_{\phi: S_1 \to S_2} \sum_{x \in S_1} \|x - \phi(x)\|_2 \quad \text{where } \phi : S_1 \to S_2 \text{ is a bijection.}$$

Correspondence (II): closest point

Given two sets of points, measure their discrepancy



a.k.a Chamfer distance (CD)

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

Required properties of distance metrics

Geometric requirement

Computational requirement

Required properties of distance metrics

Geometric requirement

- Reflects natural shape differences
- Induce a nice space for ***shape interpolations***

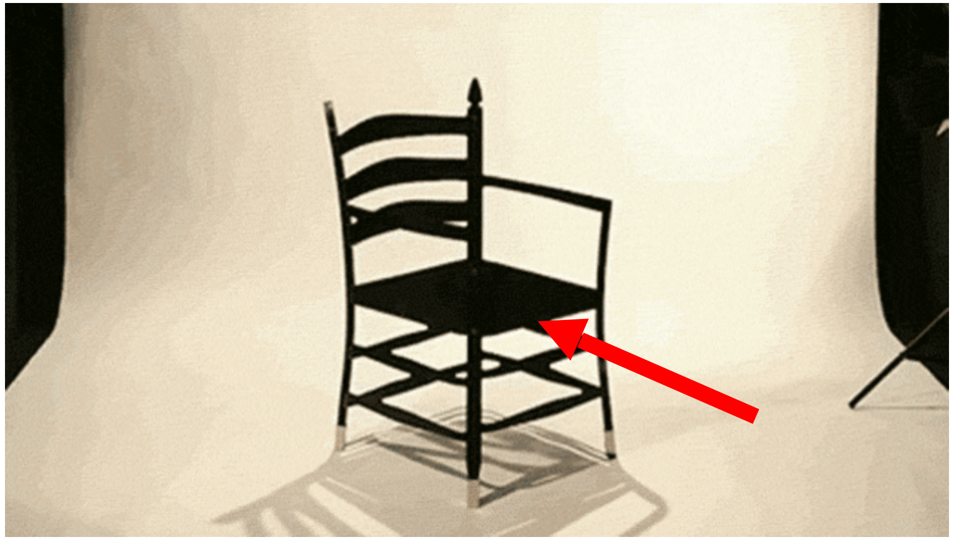
Computational requirement

How distance metric affects learning?

A fundamental issue: inherent ambiguity in prediction

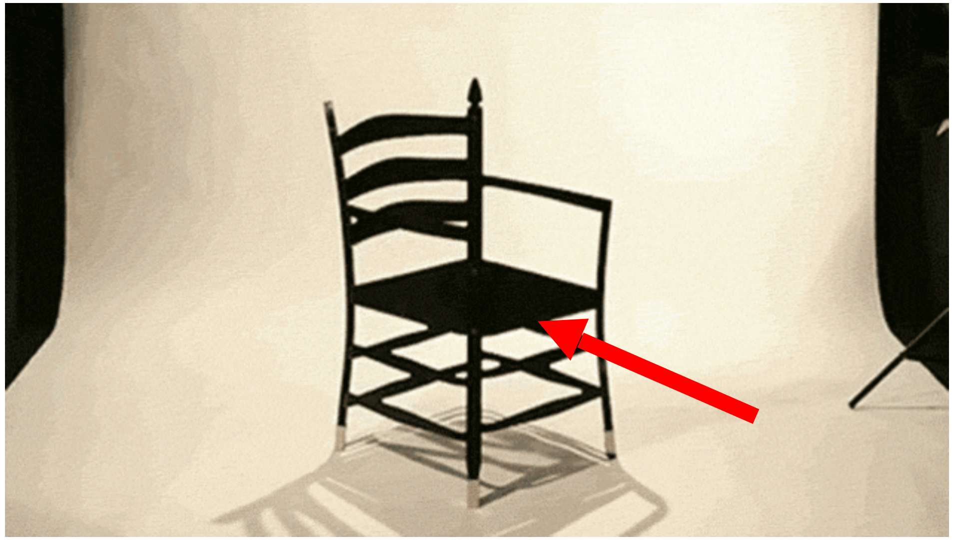
How distance metric affects learning?

A fundamental issue: inherent ambiguity in prediction



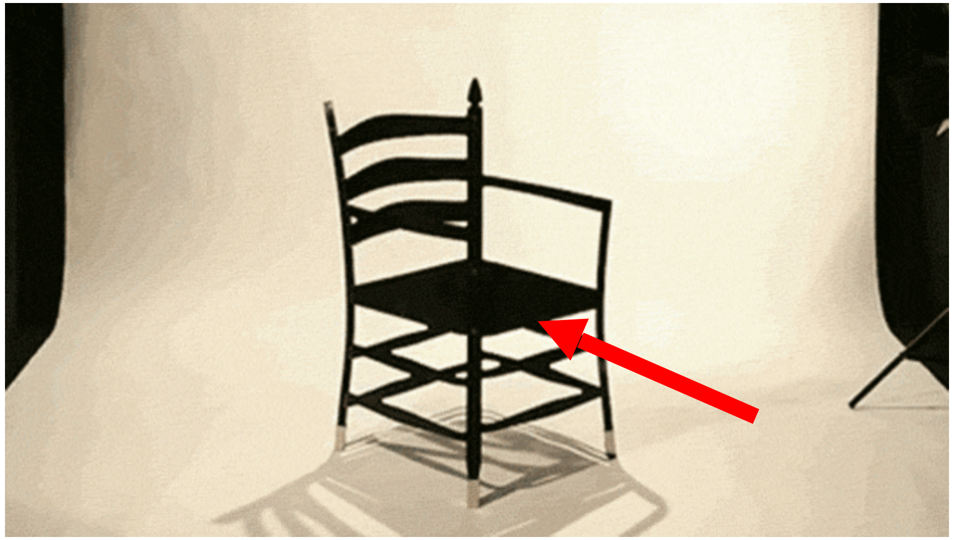
How distance metric affects learning?

A fundamental issue: inherent ambiguity in prediction



How distance metric affects learning?

A fundamental issue: inherent ambiguity in prediction



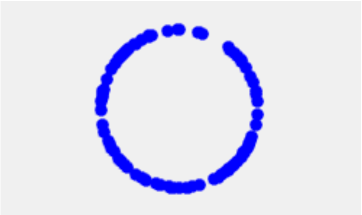
- By loss minimization, the network tends to predict a “**mean shape**” that **averages out** uncertainty

Distance metrics affect mean shapes

The mean shape carries characteristics of the distance metric

$$\bar{x} = \underset{x}{\operatorname{argmin}}\ \mathbb{E}_{s\sim\mathbb{S}}[d(x,s)]$$

continuous hidden variable (radius)



Input

EMD mean

Chamfer mean

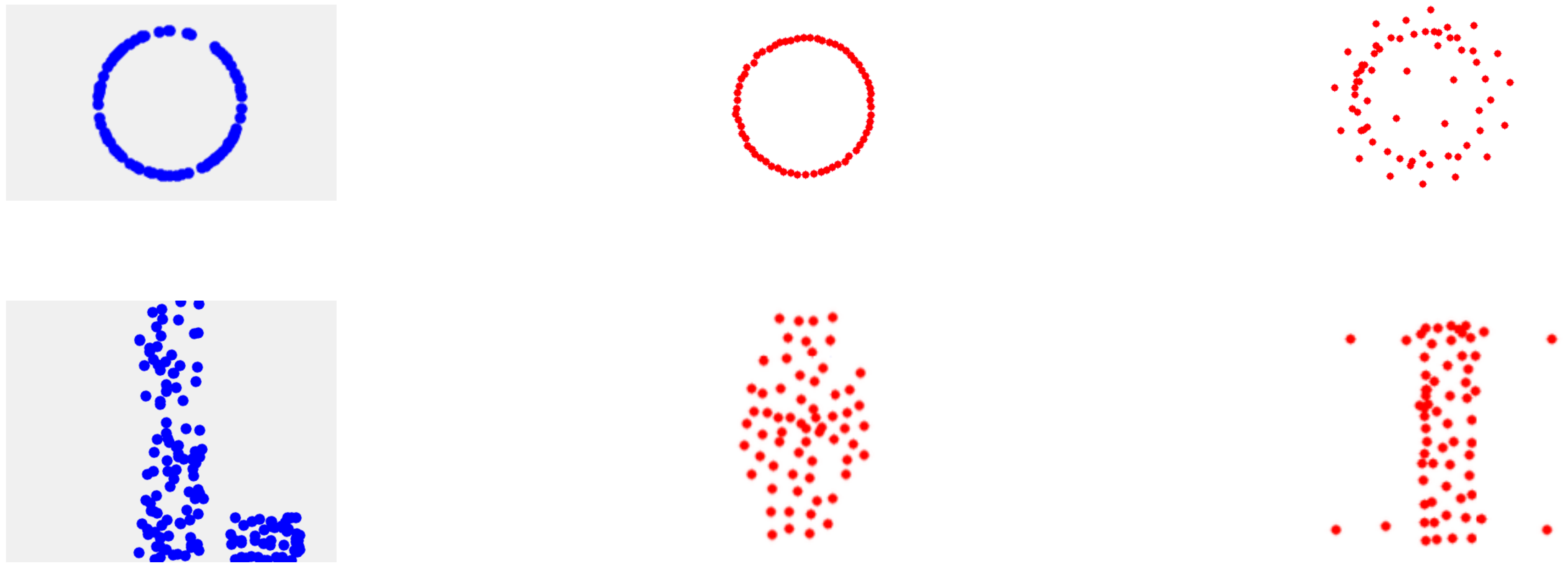
Mean shapes from distance metrics

The mean shape carries characteristics of the distance metric

$$\bar{x} = \underset{x}{\operatorname{argmin}}\ \mathbb{E}_{s\sim\mathbb{S}}[d(x, s)]$$

continuous
hidden variable
(radius)

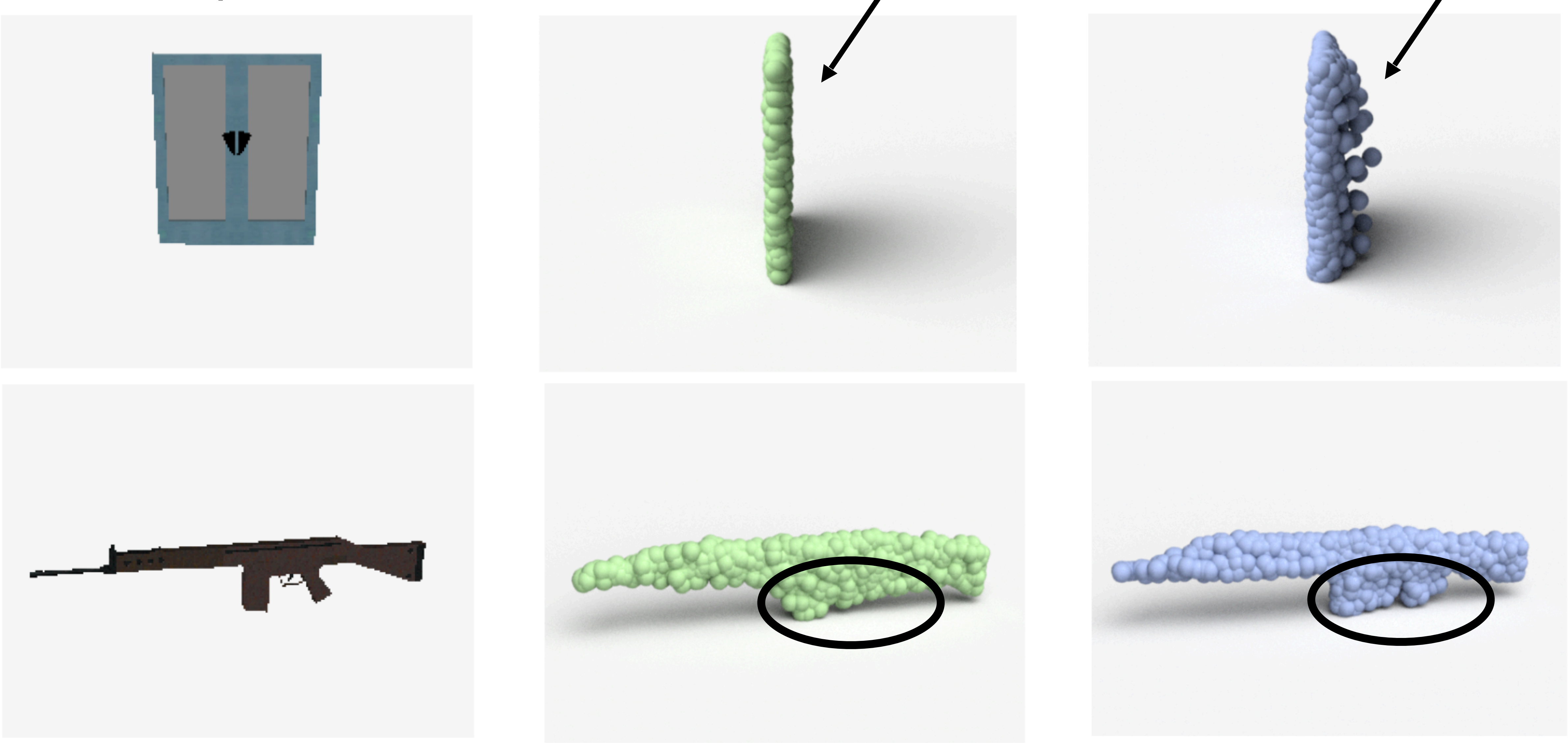
discrete
hidden variable
(add-on location)



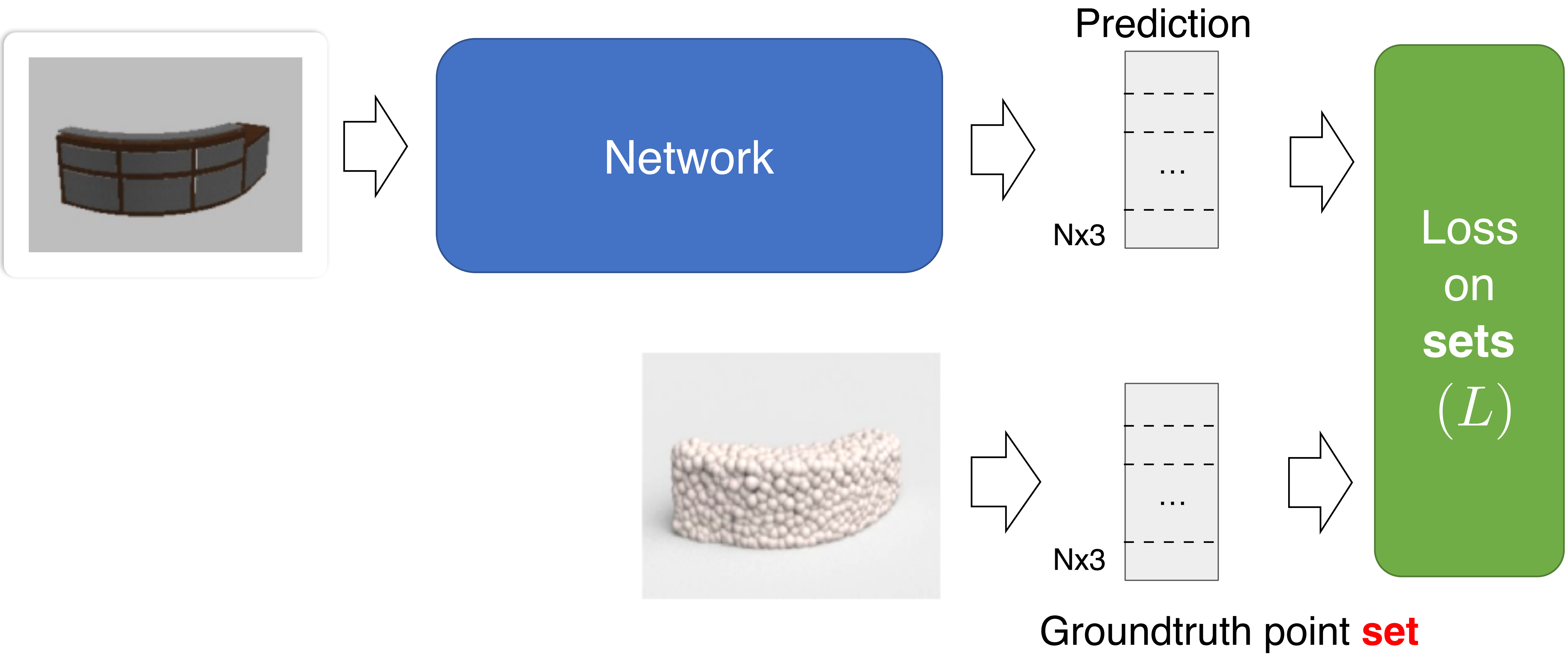
Input | EMD mean | Chamfer mean

Comparison of predictions by EMD versus CD

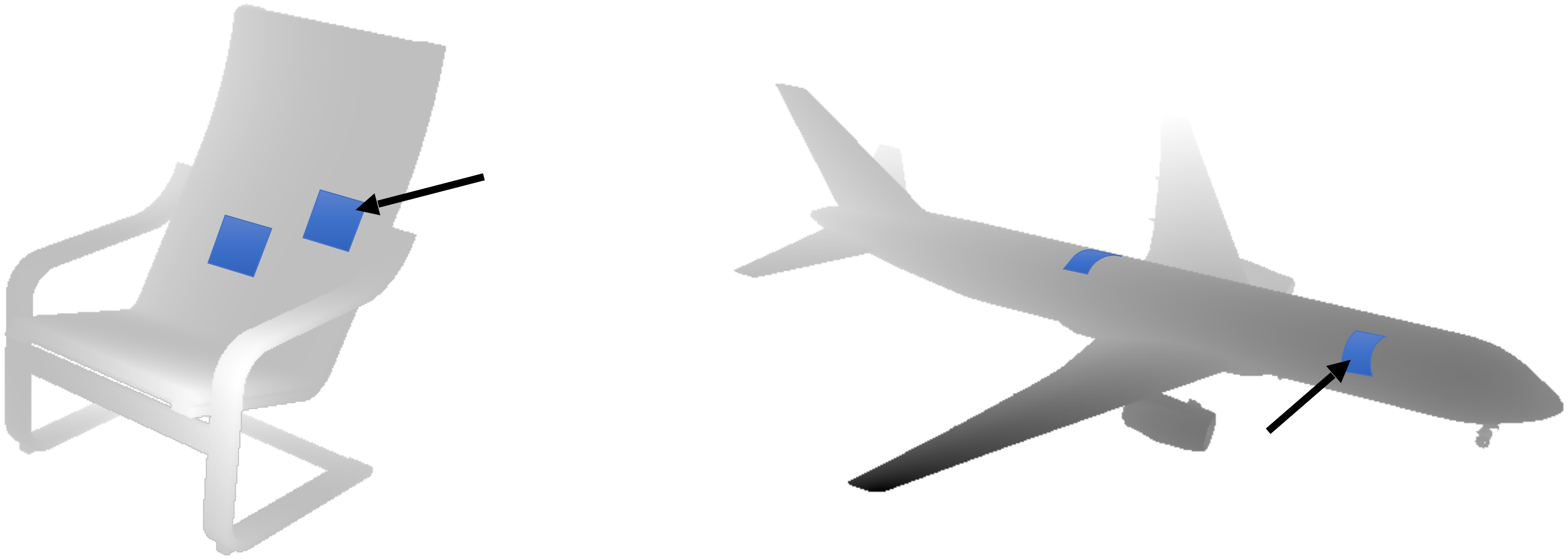
Input EMD Chamfer



Pipeline

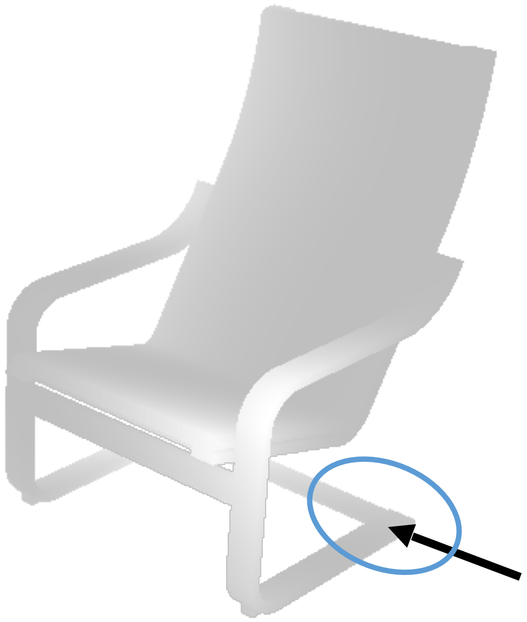

Network
Prediction
...
Nx3
...
Nx3
Loss
on
sets
(L)
Groundtruth point **set**

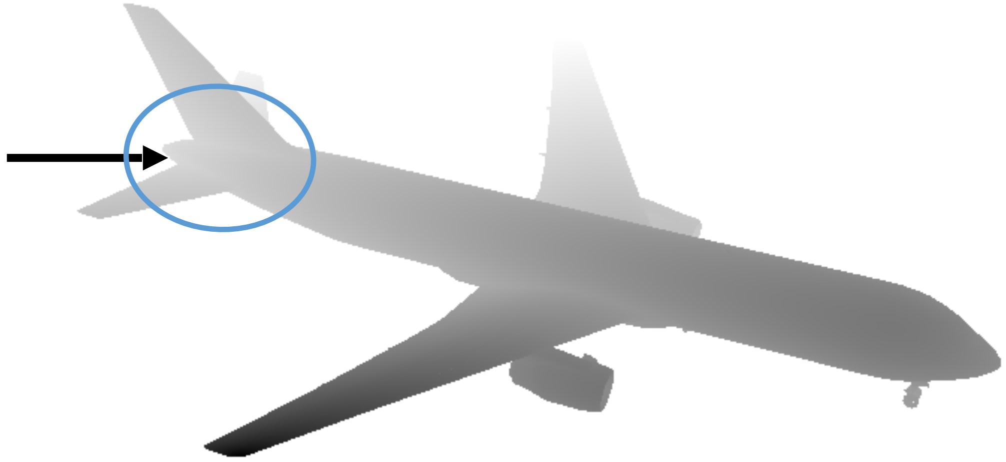
Network design: Respect natural statistics of geometry



- Many local structures are common

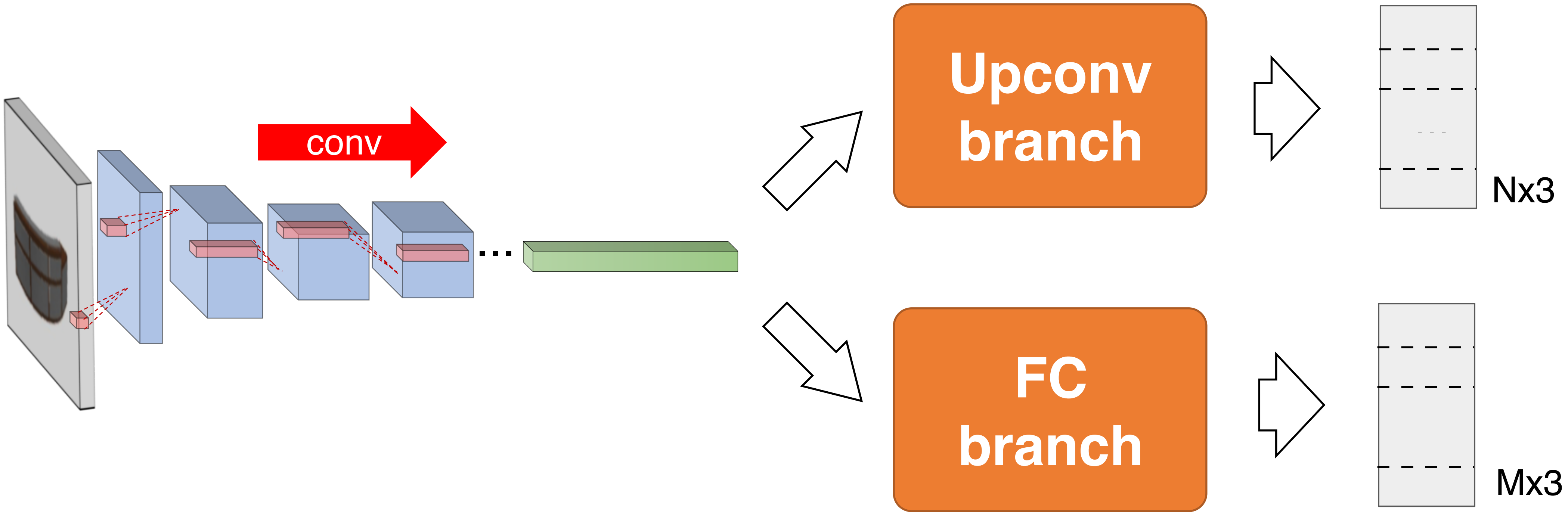
Network design: Respect natural statistics of geometry



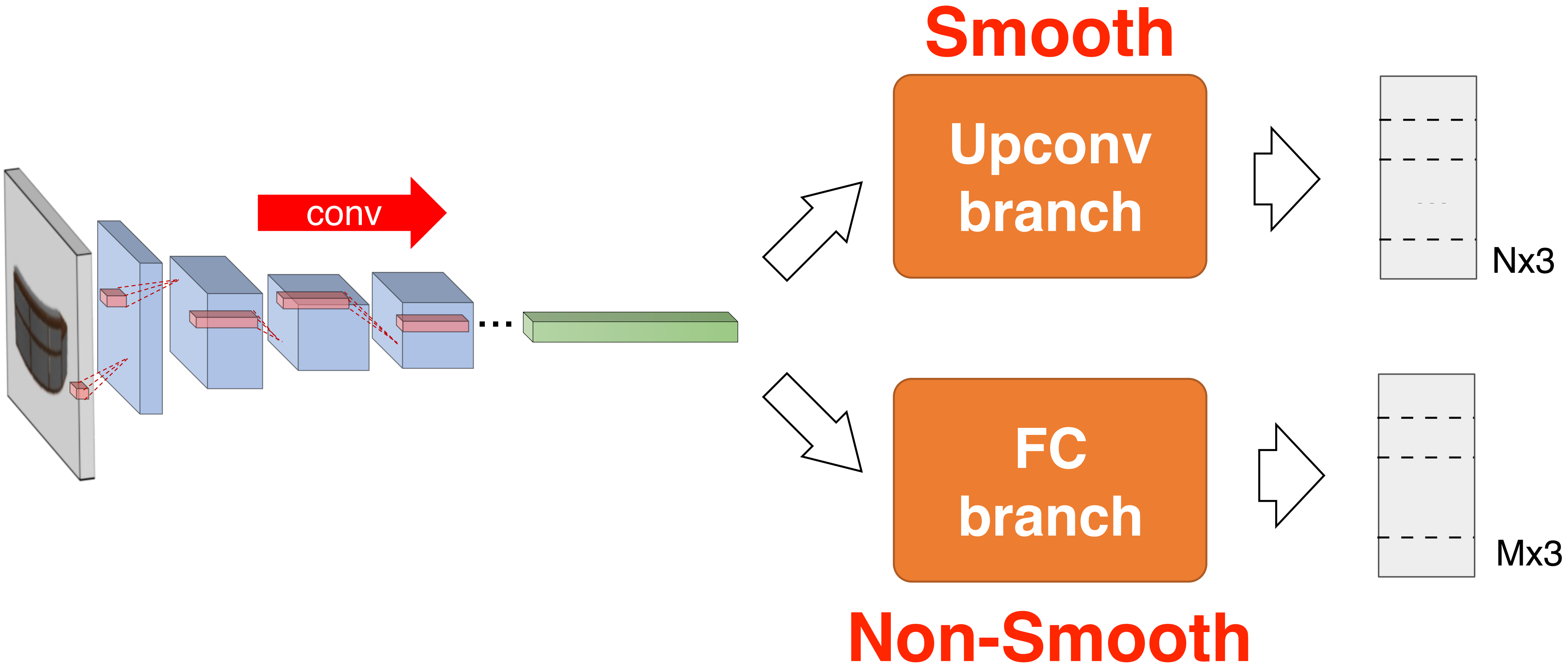


- Many local structures are common
- Also some intricate structures

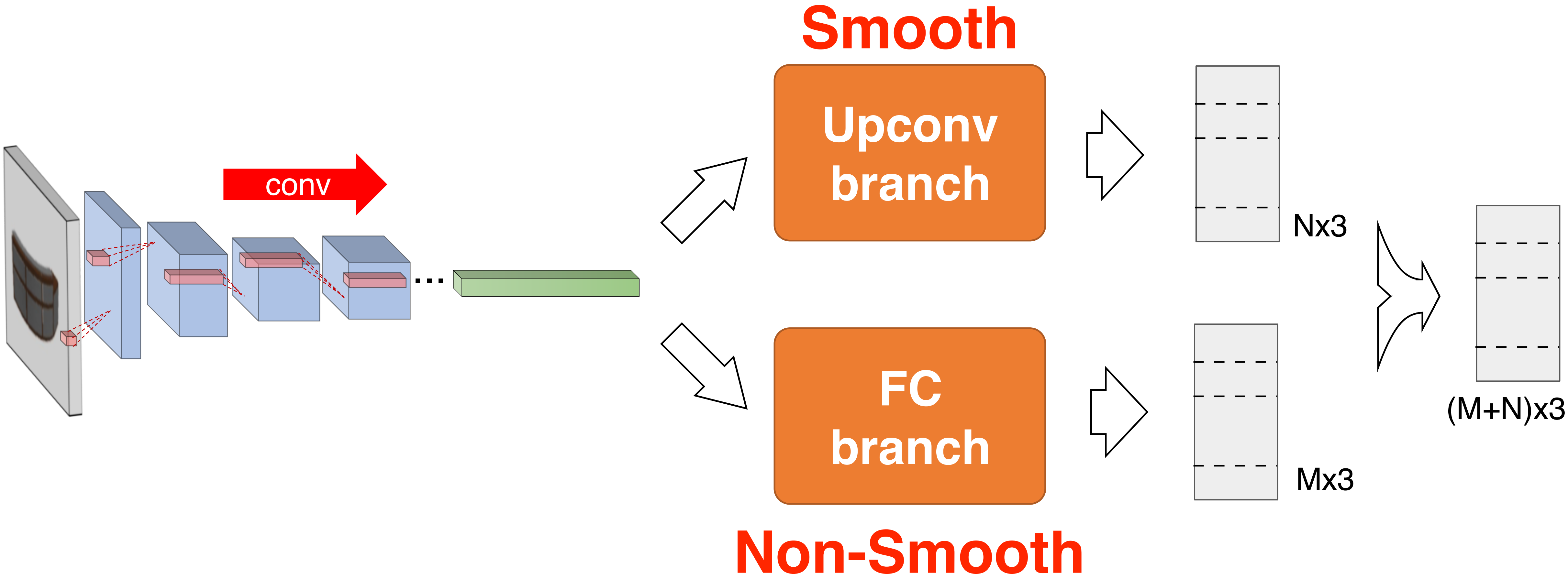
Two-branch architecture



Two-branch architecture



Two-branch architecture



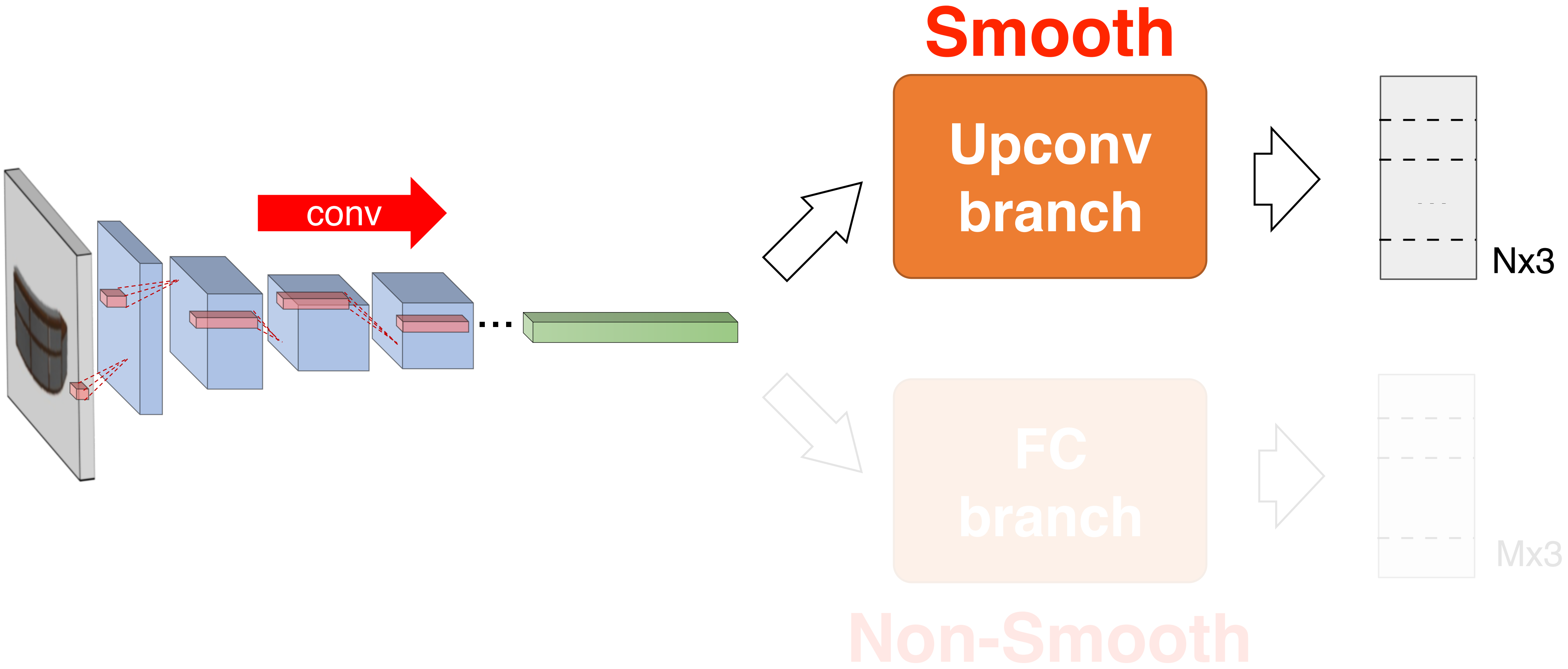
Set union by array concatenation

Which color corresponds to the upconv branch? FC branch?

Input

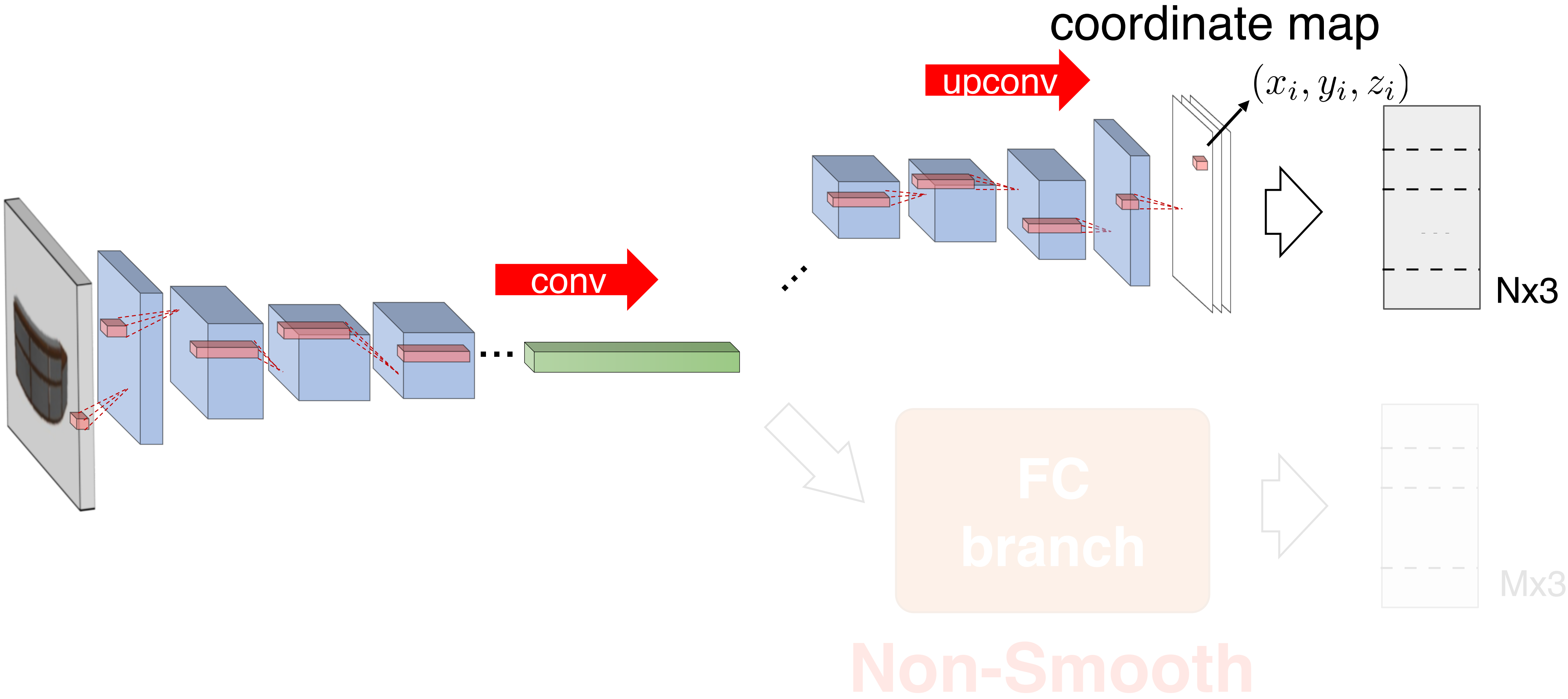
Prediction

Design of upconvolution branch



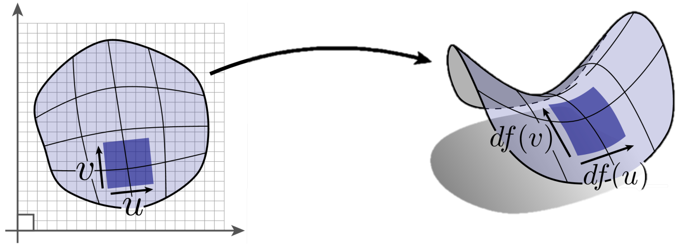
Set union by array concatenation

Design of upconvolution branch

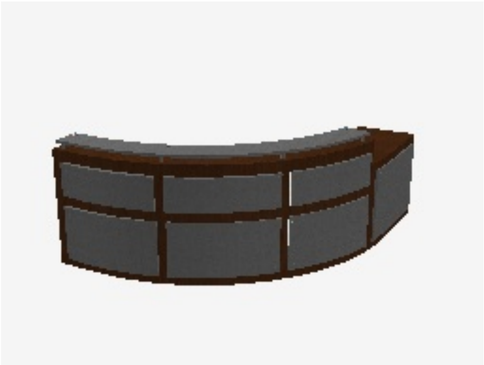


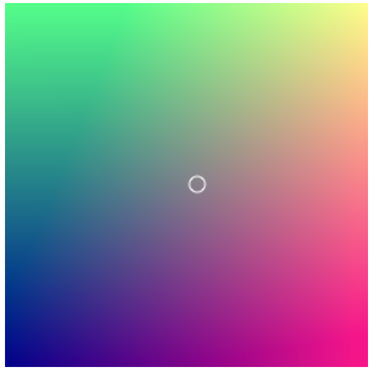
Learns a **surface parameterization**

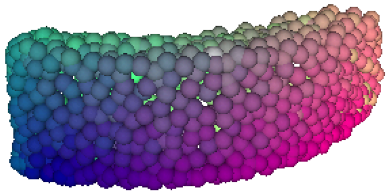
Smooth parameterization from 2D to 3D



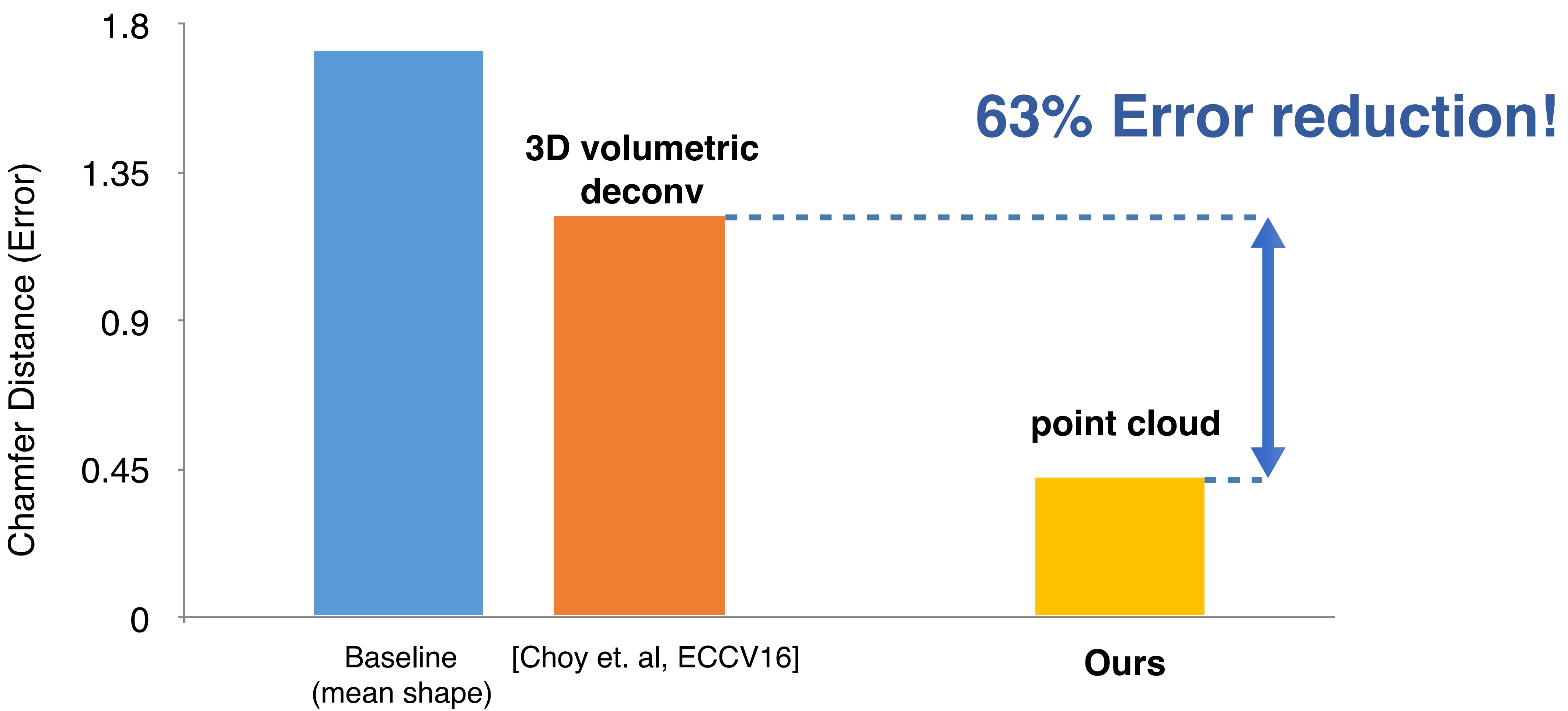
[image credit: Keenan Crane]





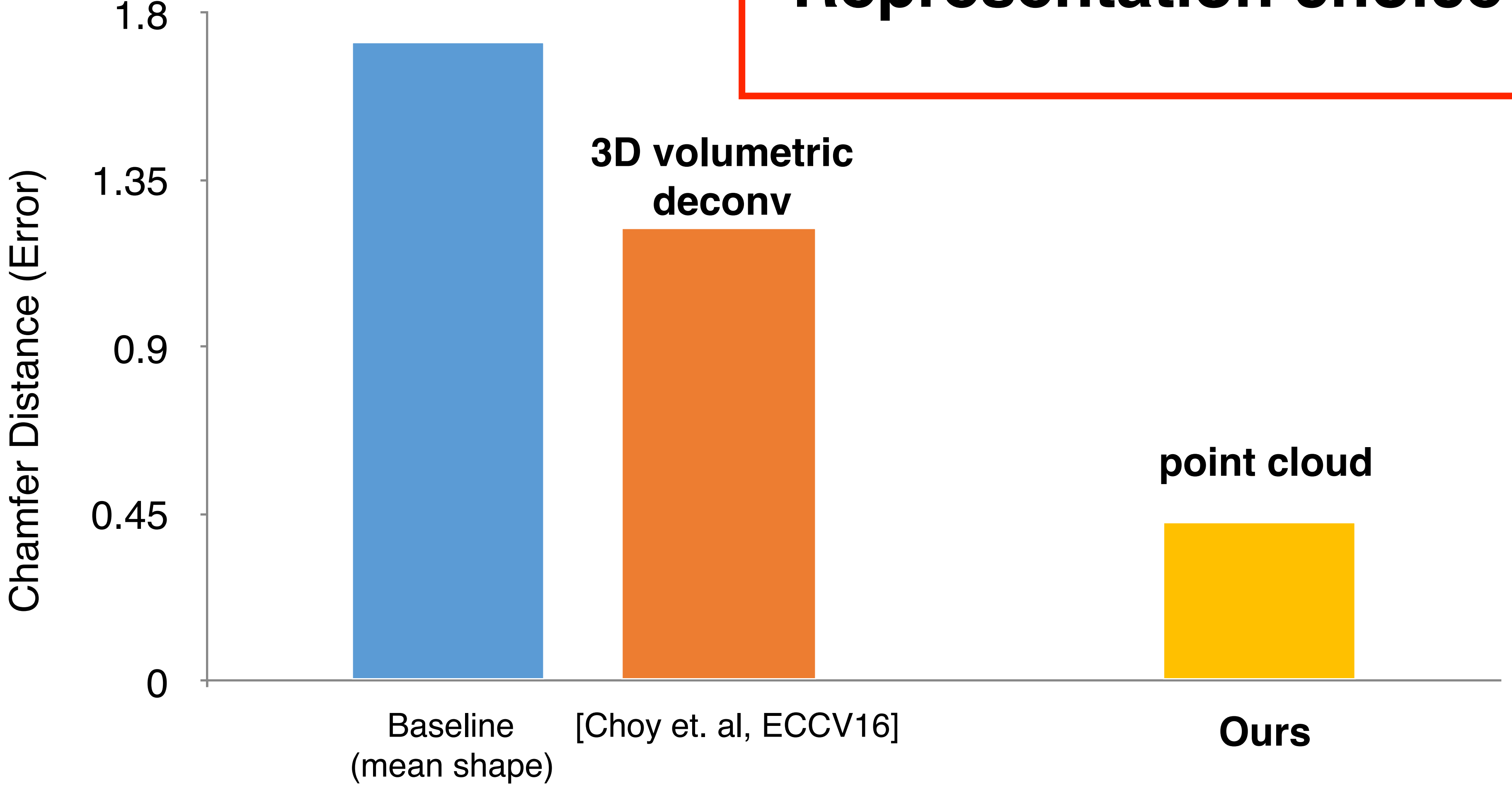


Quantitative evaluation



Quantitative evaluation

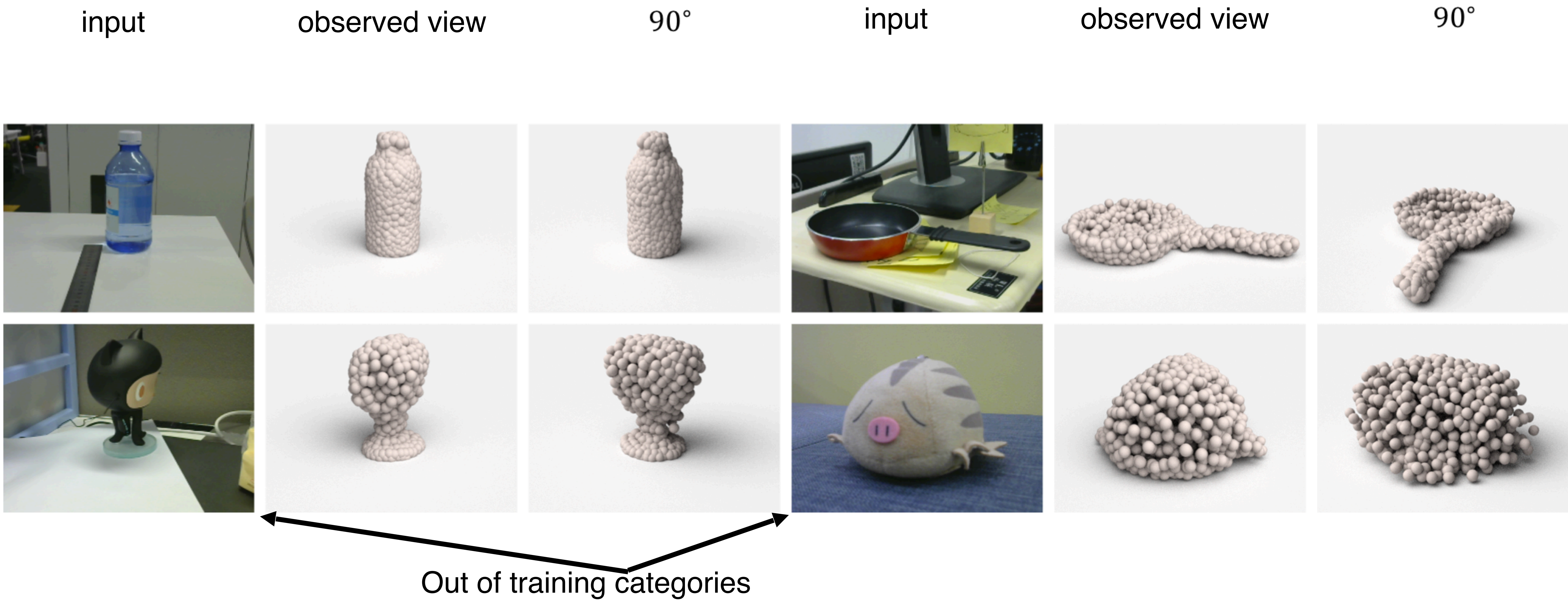
Representation choice matters!



Real-world results

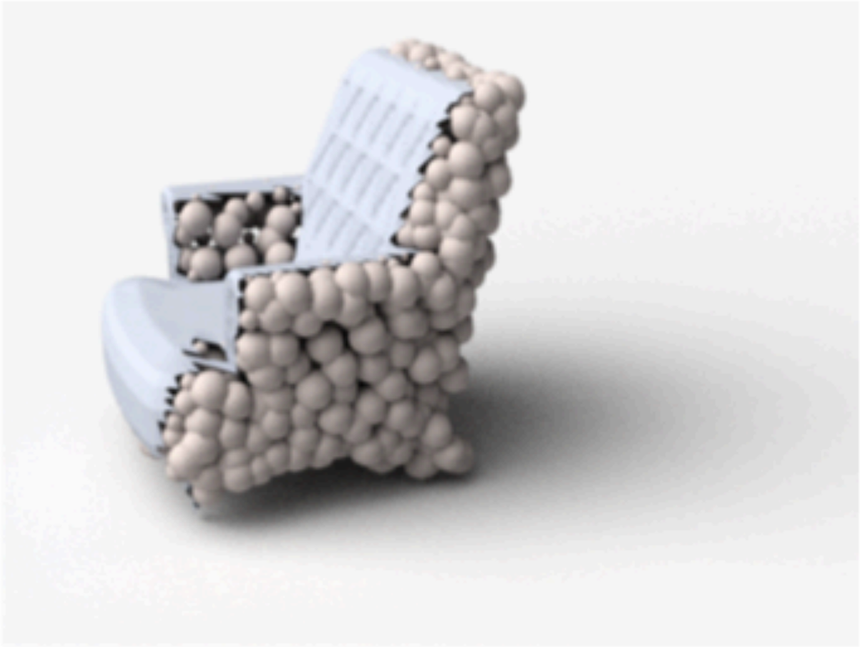
input | observed view | 90° | input | observed view | 90°

Generalization to unseen categories



Extension: Shape completion

Input: Incomplete shape



Output: Completed shape

Open problems

A better metric that takes the best of Chamfer and EMD?

How to add further structure constraints?

How to extend the pipeline to scene level?

How generalizable the method is?

In principle, what is the generalizability of a geometry estimator? To what extend is 3D perception ability innate or learned?

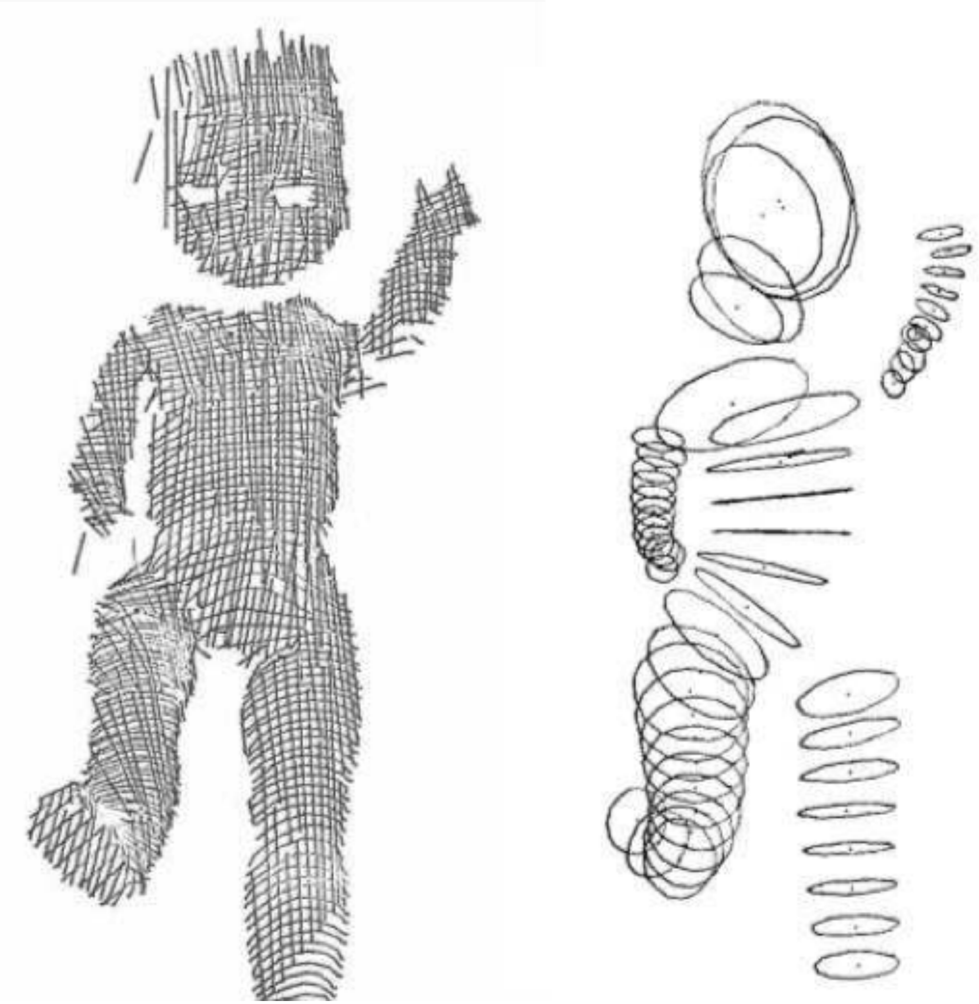
Agenda

- Deep learning on regular structures

- Deep learning on meshes

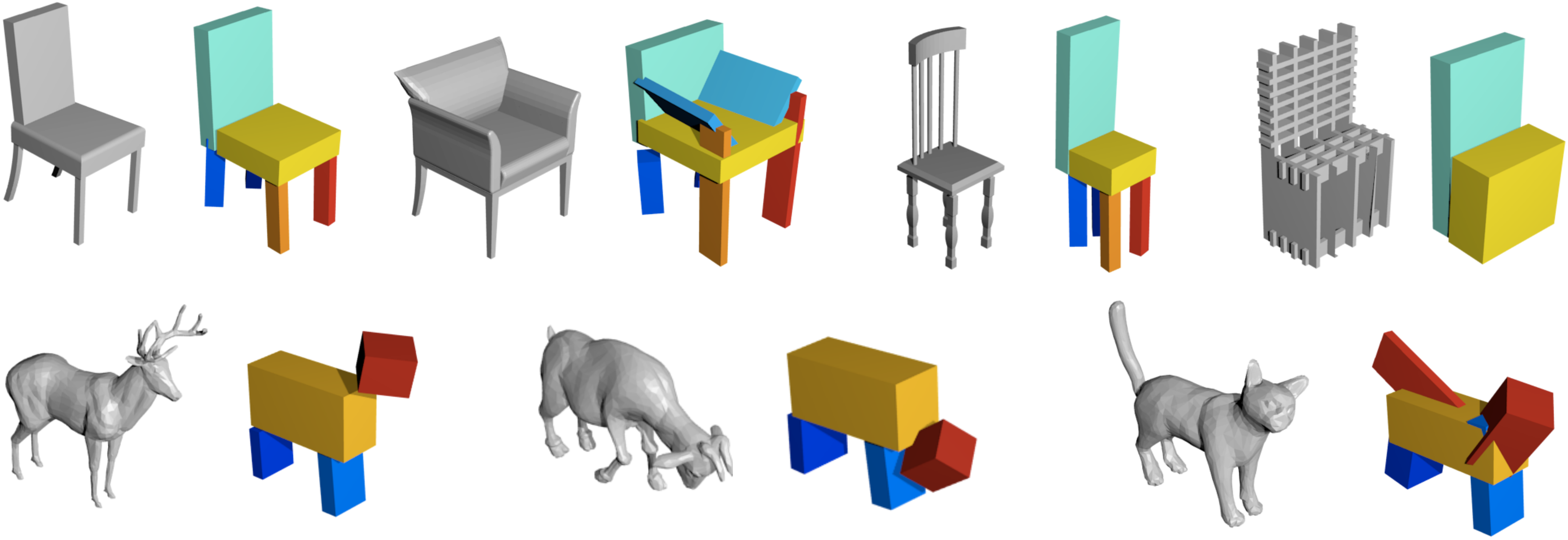
- **Deep learning on point cloud and parametric models**
  - Point cloud analysis
  - Point cloud synthesis
  - **Primitive-based shapes**

A historical overview



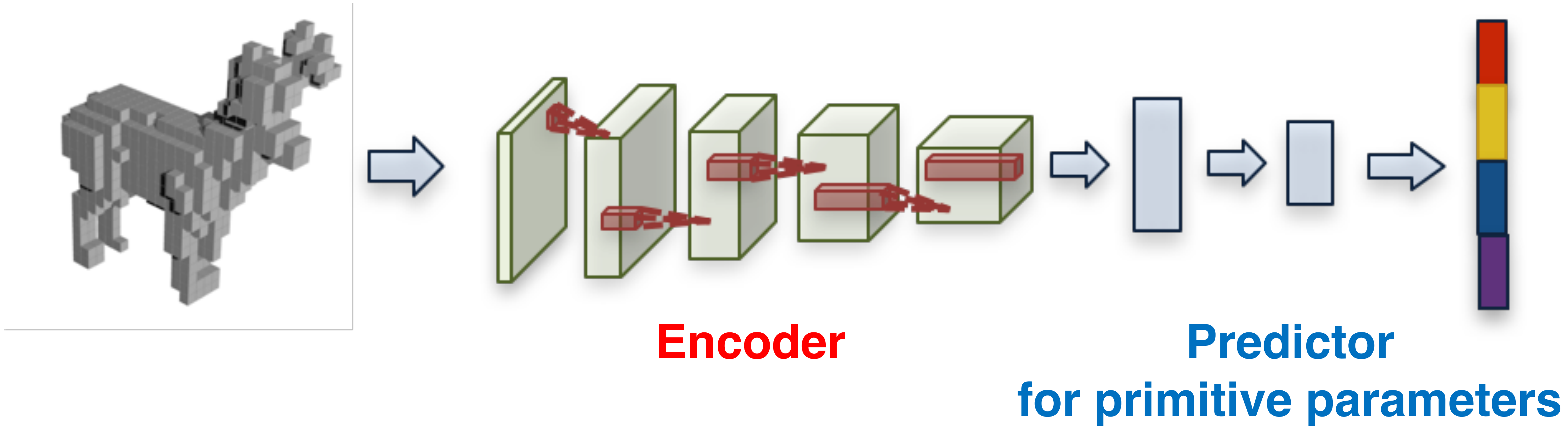
Generalized Cylinders, Binford (1971)

Primitive-based assembly



Learn to predict a corresponding shape composed by primitives.
It allows us to predict **consistent** compositions across objects.

Approach

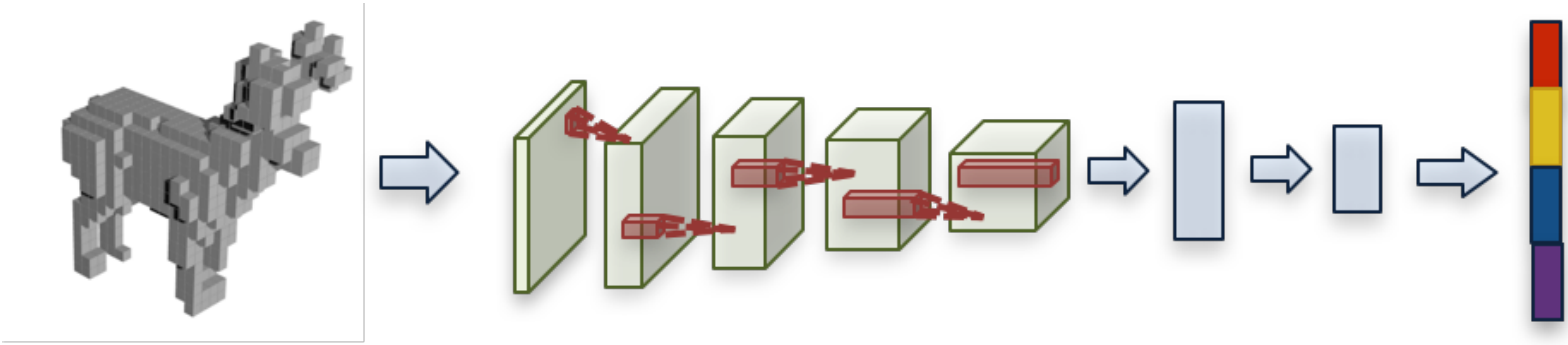


Predict primitive parameters: size, rotation, translation of M cuboids.

Unsupervised parsing

Each point is colored according to the assigned primitive

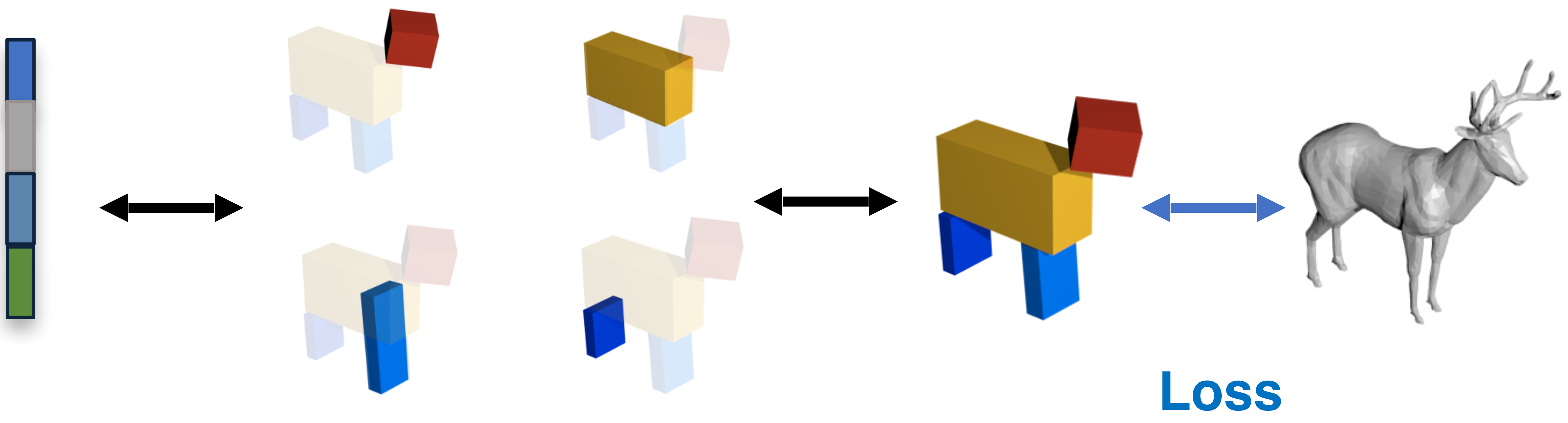
Approach



Predict primitive parameters: size, rotation, translation of M cuboids.

Variable number of parts? We predict “primitive existence probability”

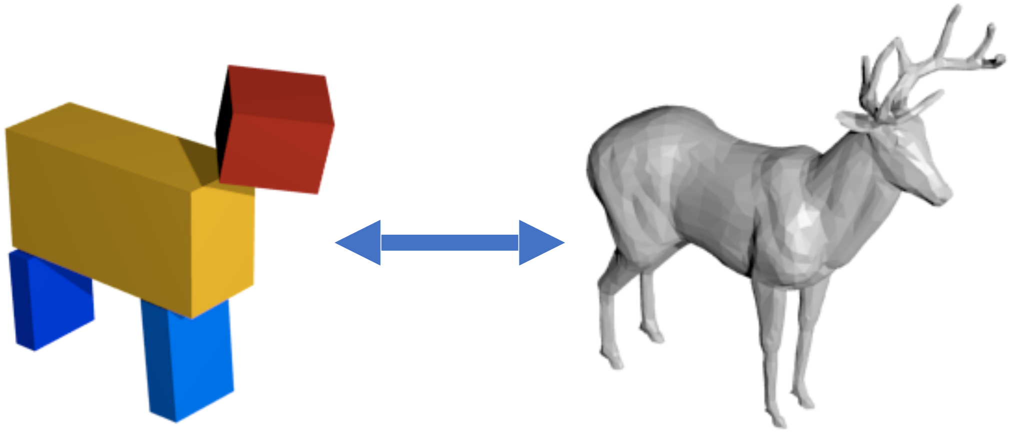
Loss function



Loss function construction

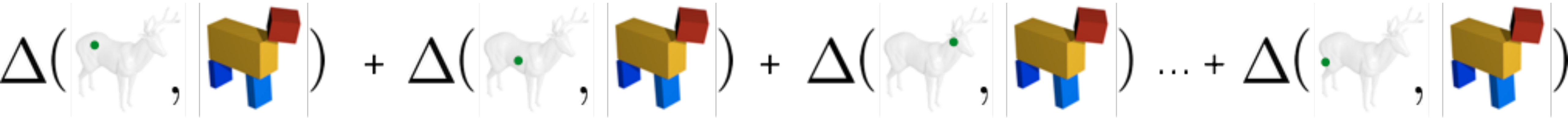
Basic idea: **Chamfer distance!**

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



Loss function construction

Sample points on the groundtruth mesh and predicted assembly

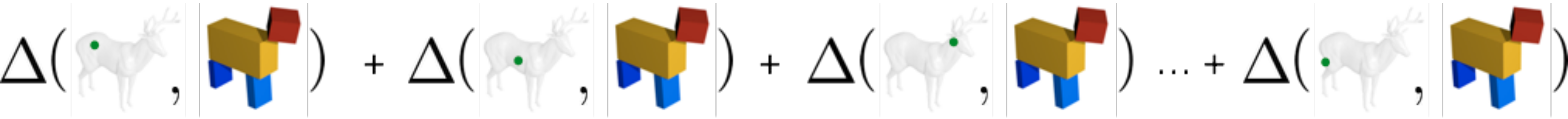


Each point is a **linear function** of mesh/primitive vertex coordinates

Differentiable!

Loss function construction

Sample points on the groundtruth mesh and predicted assembly

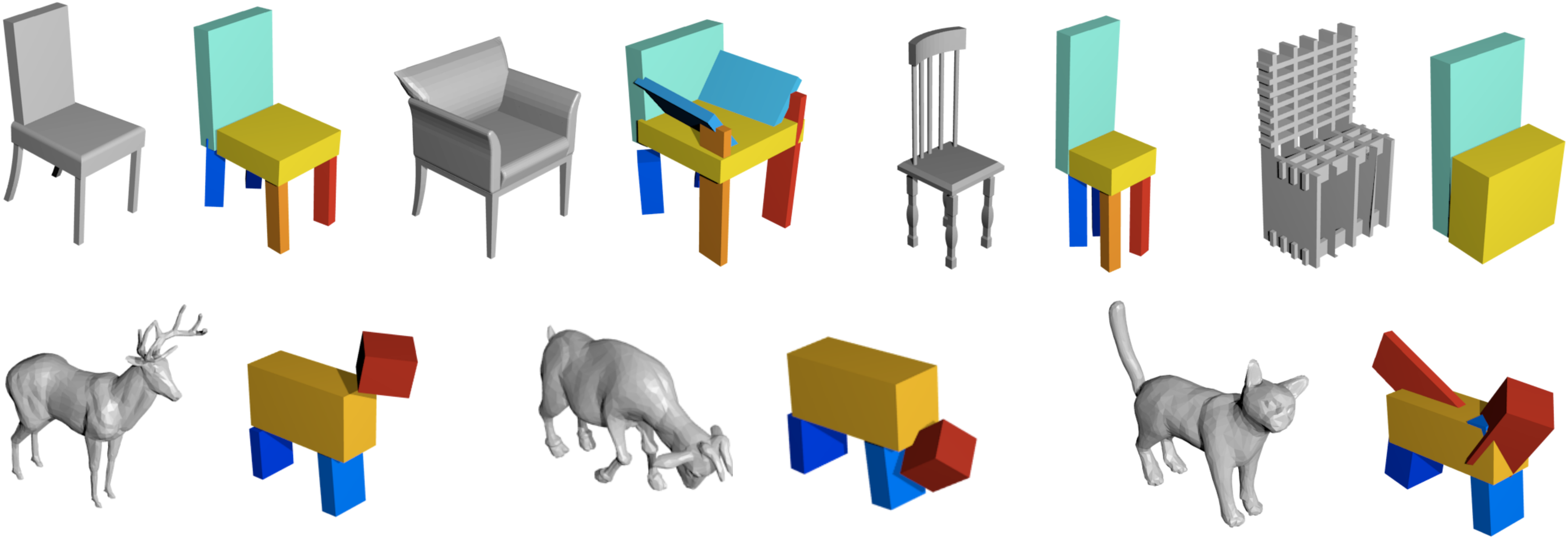


Each point is a **linear function** of mesh/primitive vertex coordinates

Differentiable!

Speed up the computation leveraging parameterization of primitives

Consistent primitive configurations



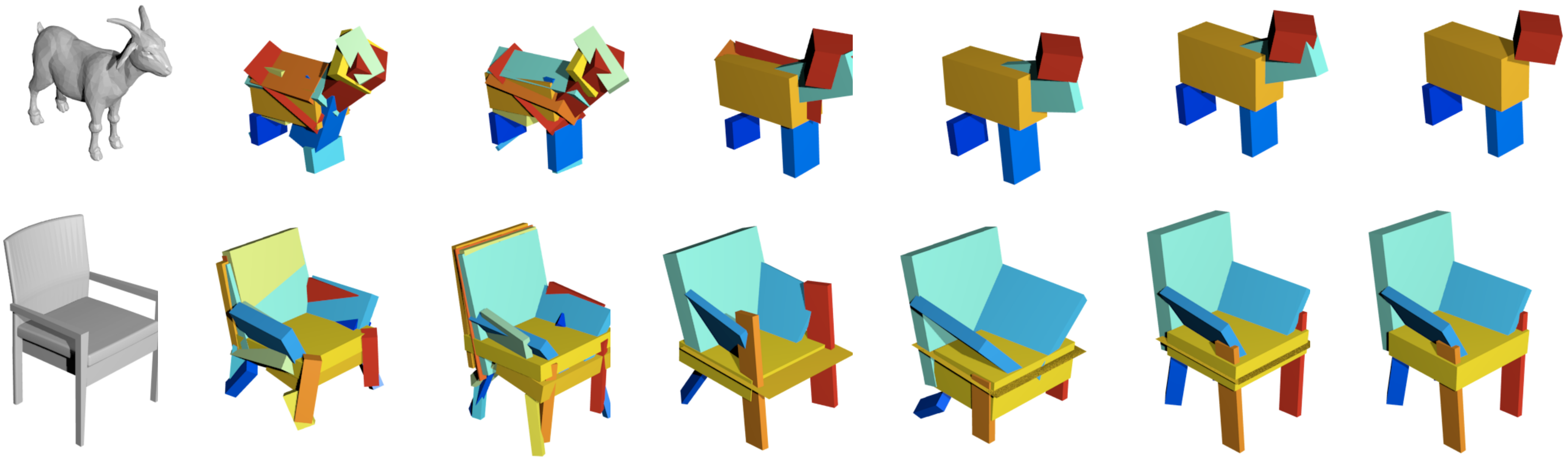
Primitive locations are **consistent** due to
the **smoothness** of primitive prediction network

Unsupervised parsing

Method	[31] (initial)	[31] (refined)	Ours
Accuracy	78.6	84.8	89.0

Mean accuracy (face area) on Shape COSEG chairs.

Analysis



Shapes become more parsimonious as training progresses (due to our parsimony reward)

Image-based modeling

Open problems

How to introduce other primitives types?

Towards image based modeling, how to add more operations to edit those primitives?

- e.g., Deform? Extrude? Loop cut?

How to use it for design purposes? For example, with certain structural and functional constraints.

Ultimately, we expect to automate the modeling process from images, as artists do.