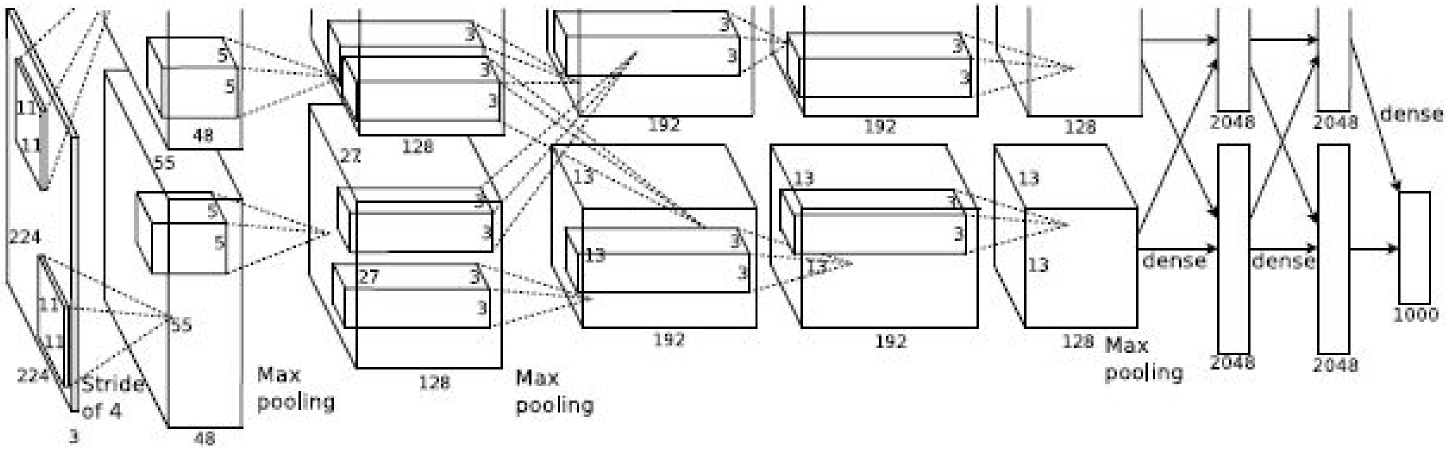
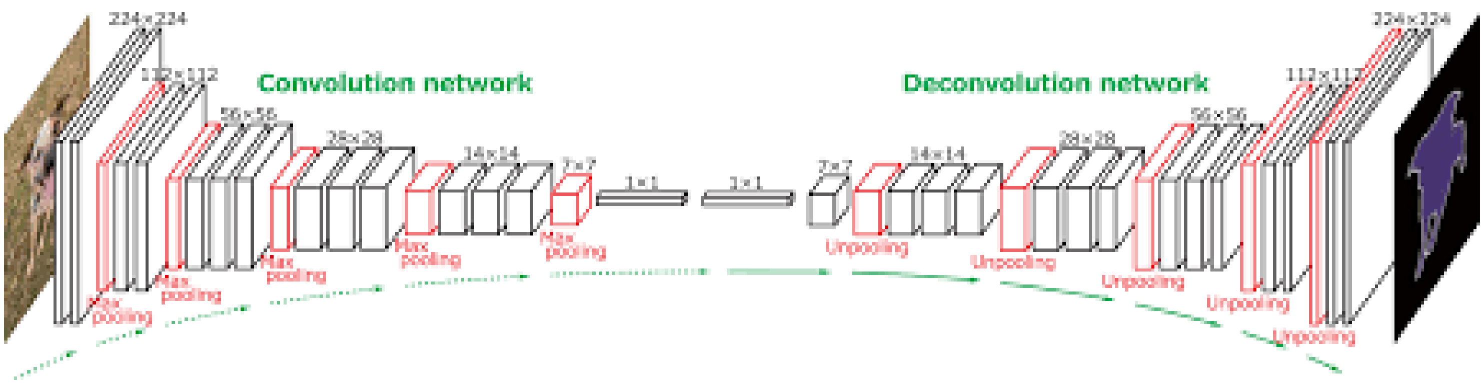
GAMES
3D Deep Learning

Qixing Huang
September 2th 2021

AlexNet

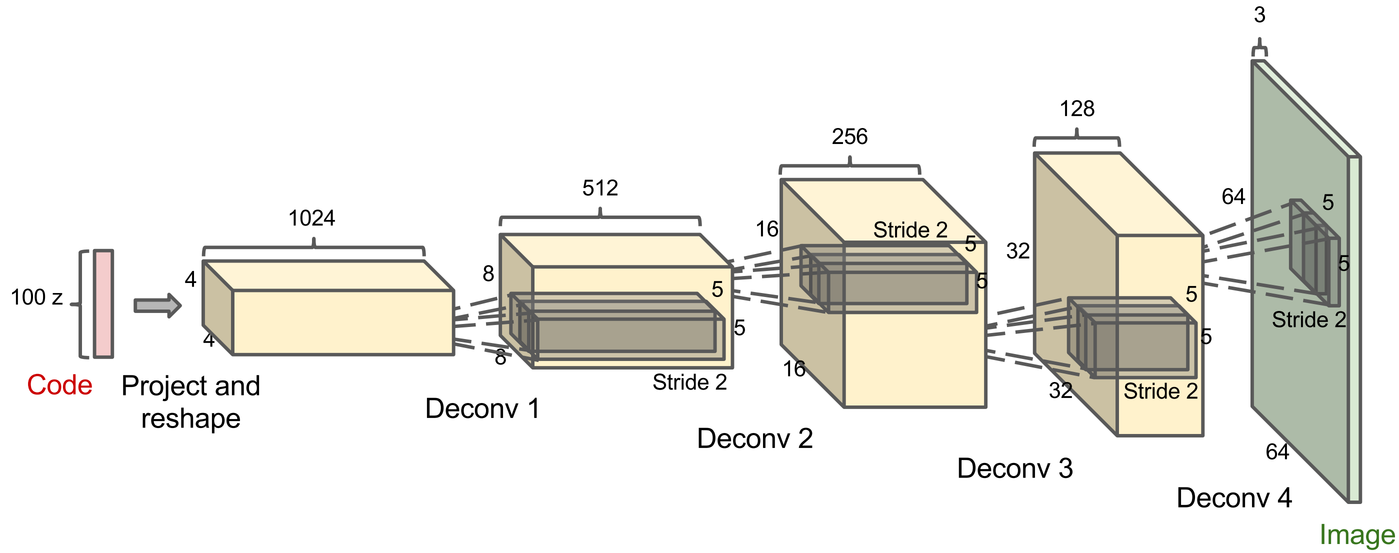


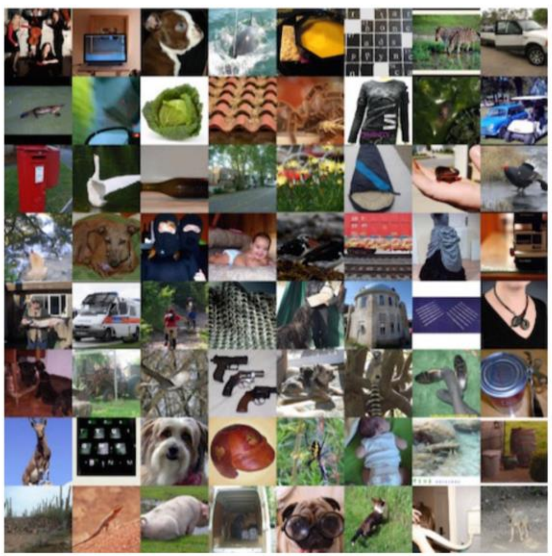
224
11
11
224
3
Stride of 4
11
11
55
55
48
48
5
5
5
5
Max pooling
27
27
128
128
3
3
Max pooling
13
13
192
192
128
Max pooling
dense
dense
dense
2048
2048
2048
2048
1000



Convolution network
Deconvolution network
224×224
112×112
56×56
28×28
14×14
7×7
1×1
Max pooling
Unpooling

Image Generation

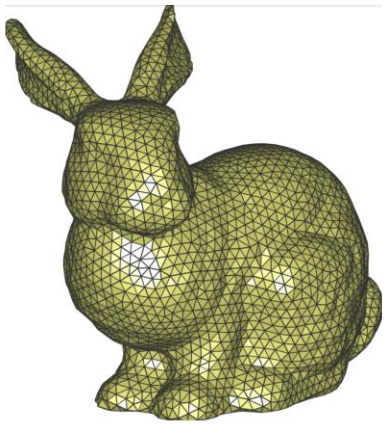




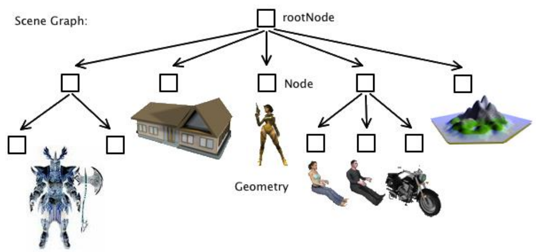
Real images (ImageNet)

Generated images

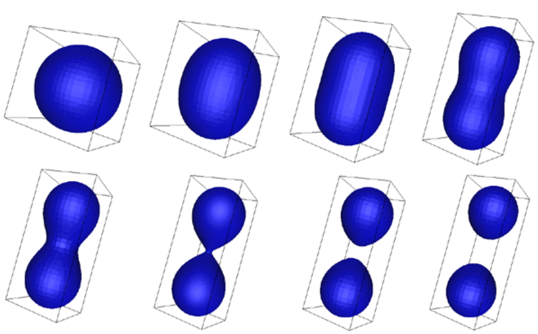
3D Surface Representations



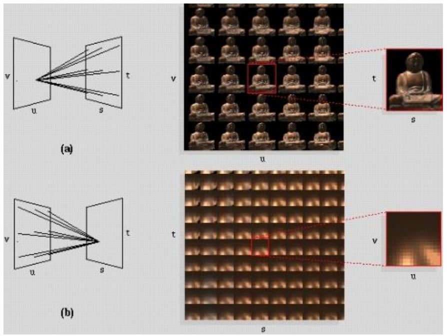
Triangular mesh



Part-based models



Implicit surface



Light Field Representation

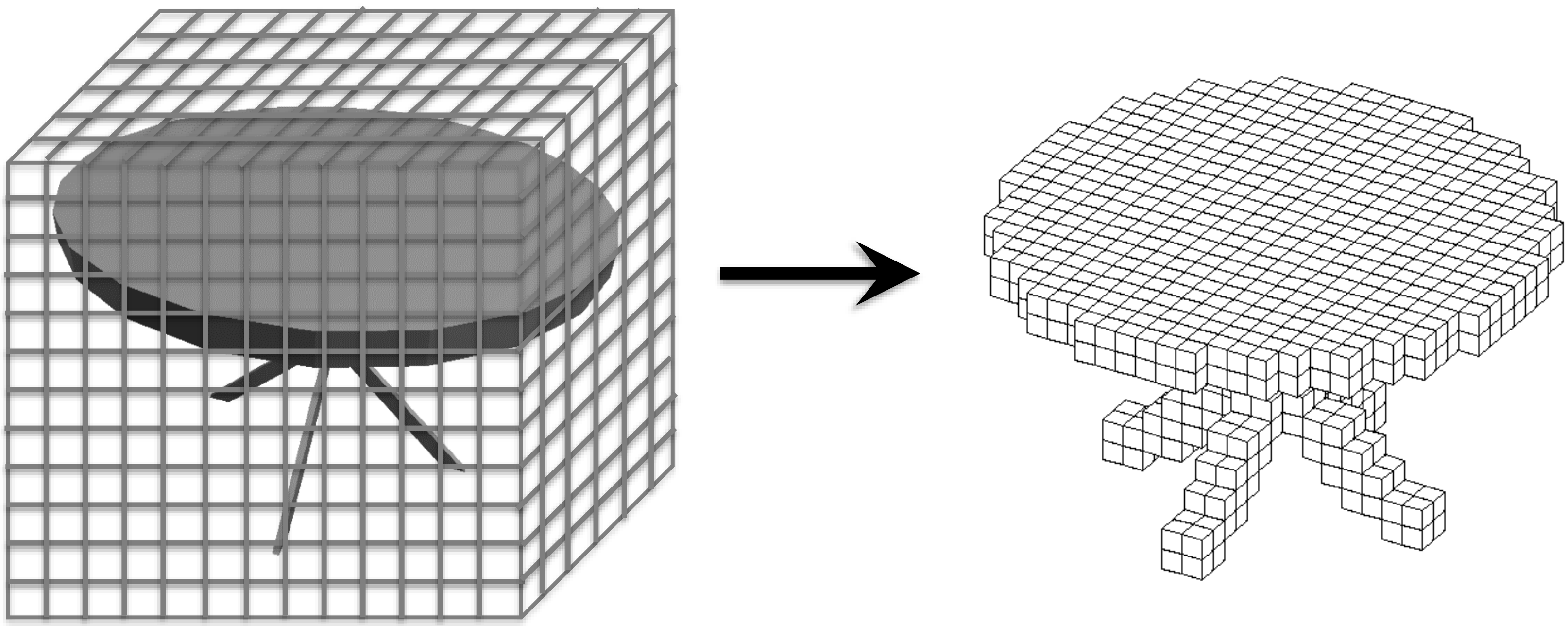
Point cloud

3D Voxel Grids

[Wu et al. 15]

3D Deep Learning

3D Shape as Volumetric Representation

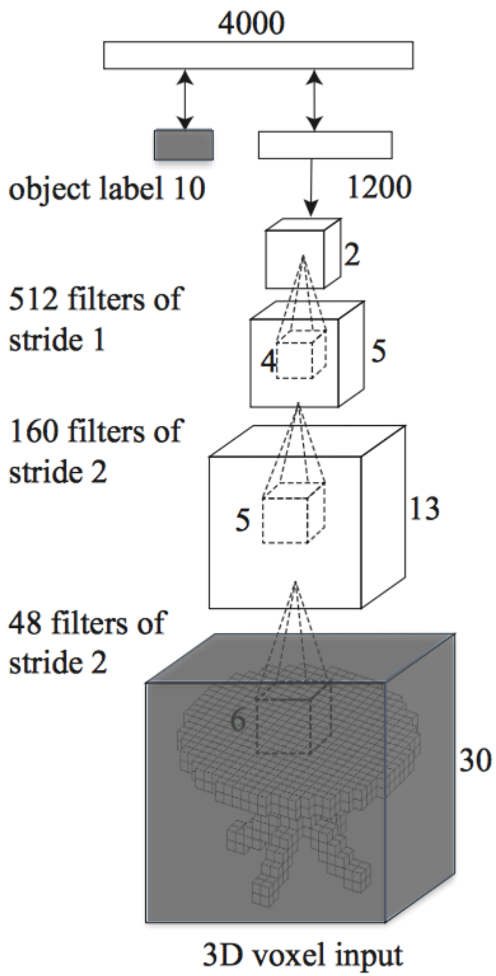


mesh

binary voxel

[Wu et al. 15]

3D ShapeNets



Convolutional Deep Belief Network $p(\mathbf{x}, y)$

A **Deep Belief Network** is a generative graphical model that describes the distribution of input x over class y.

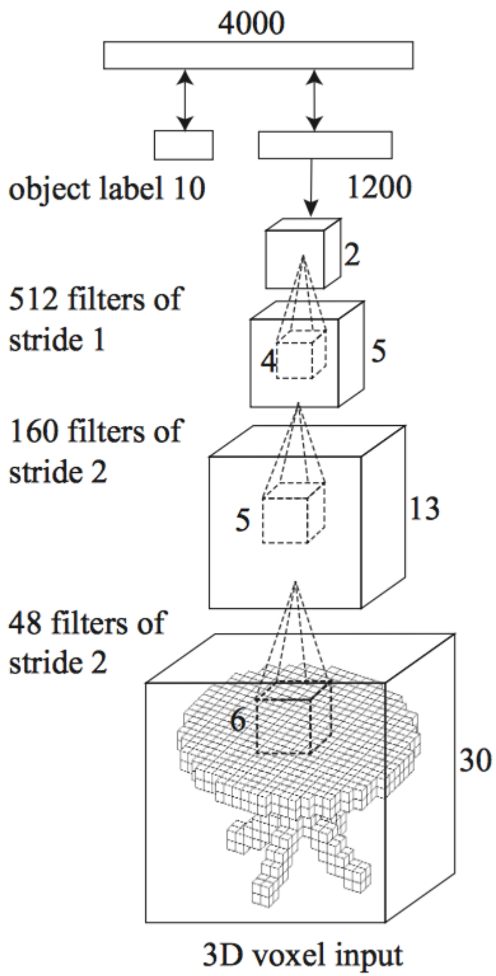
- Convolution to enable compositionality
- No pooling to reduce reconstruction error

configurations

Layer 1-3	convolutional RBM
Layer 4	fully connected RBM
Layer 5	multinomial label + Bernoulli feature form an associate memory

[Wu et al. 15]

3D ShapeNets



Convolutional Deep Belief Network $p(\mathbf{x}, y)$

3D ShapeNets ≠ CNNs

$$p(x, y) \qquad p(y|x)$$

$$\downarrow$$

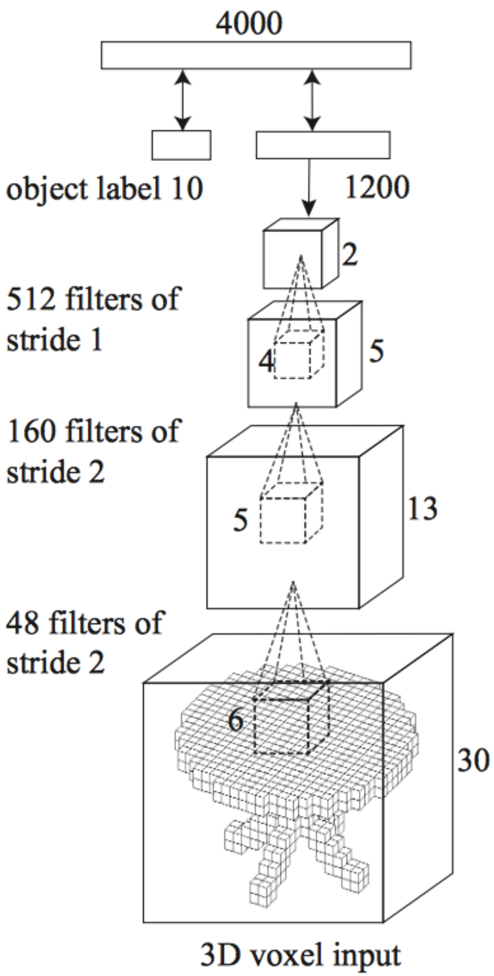
$p(y|x)$ discriminative process

$p(x|y)$ generative process

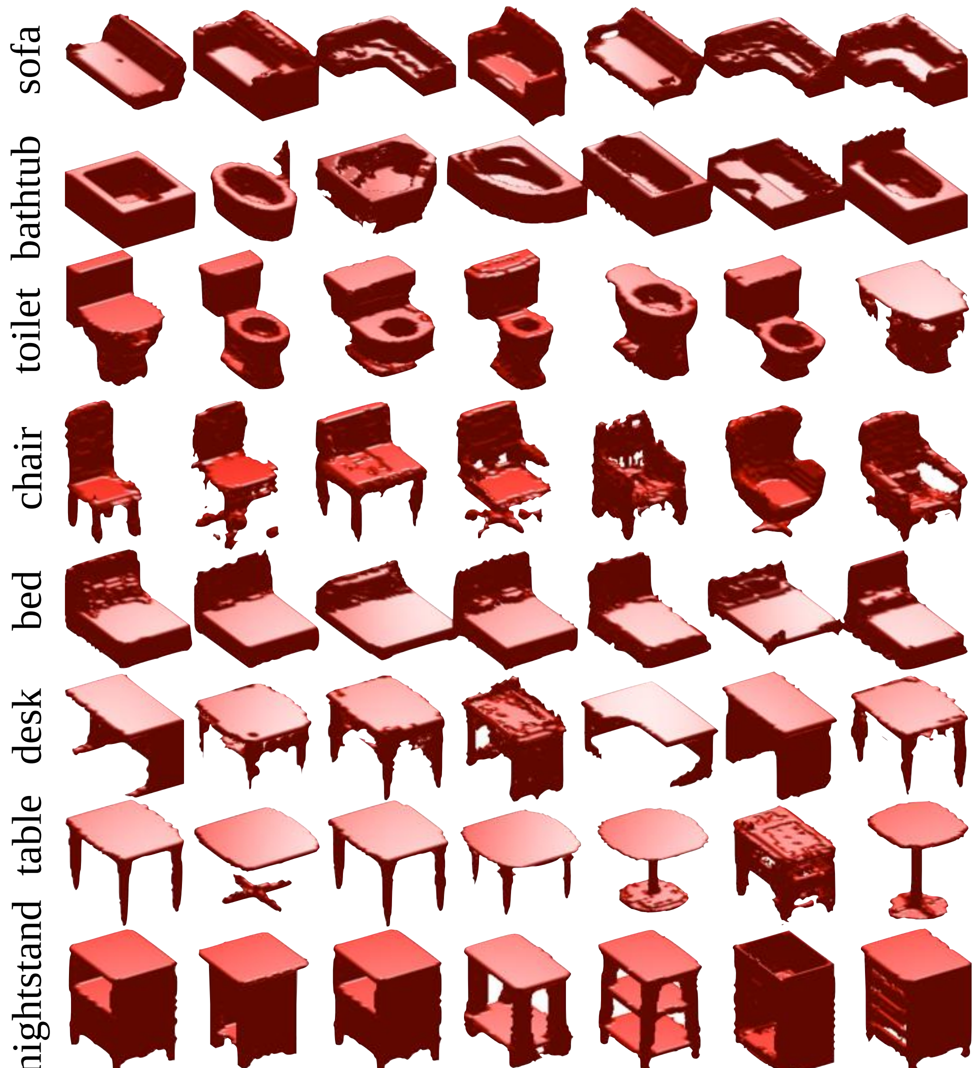
* 3D ShapeNets can be converted into a CNN,
and discriminatively trained with back-propagation.

[Wu et al. 15]

As a 3D Shape Prior

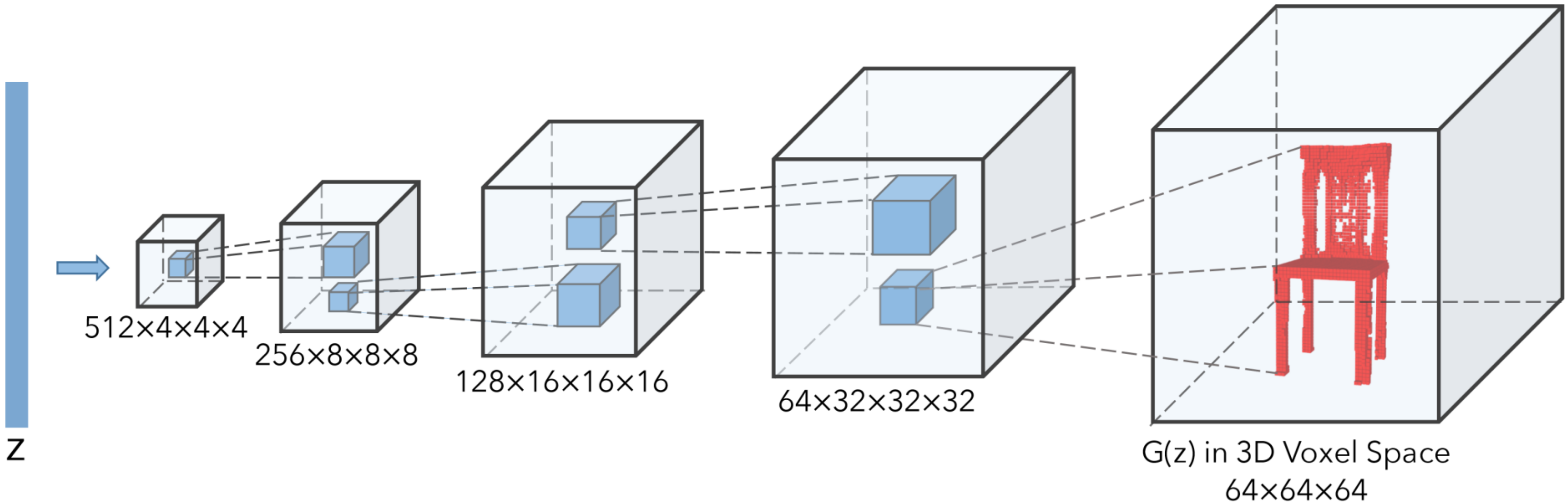


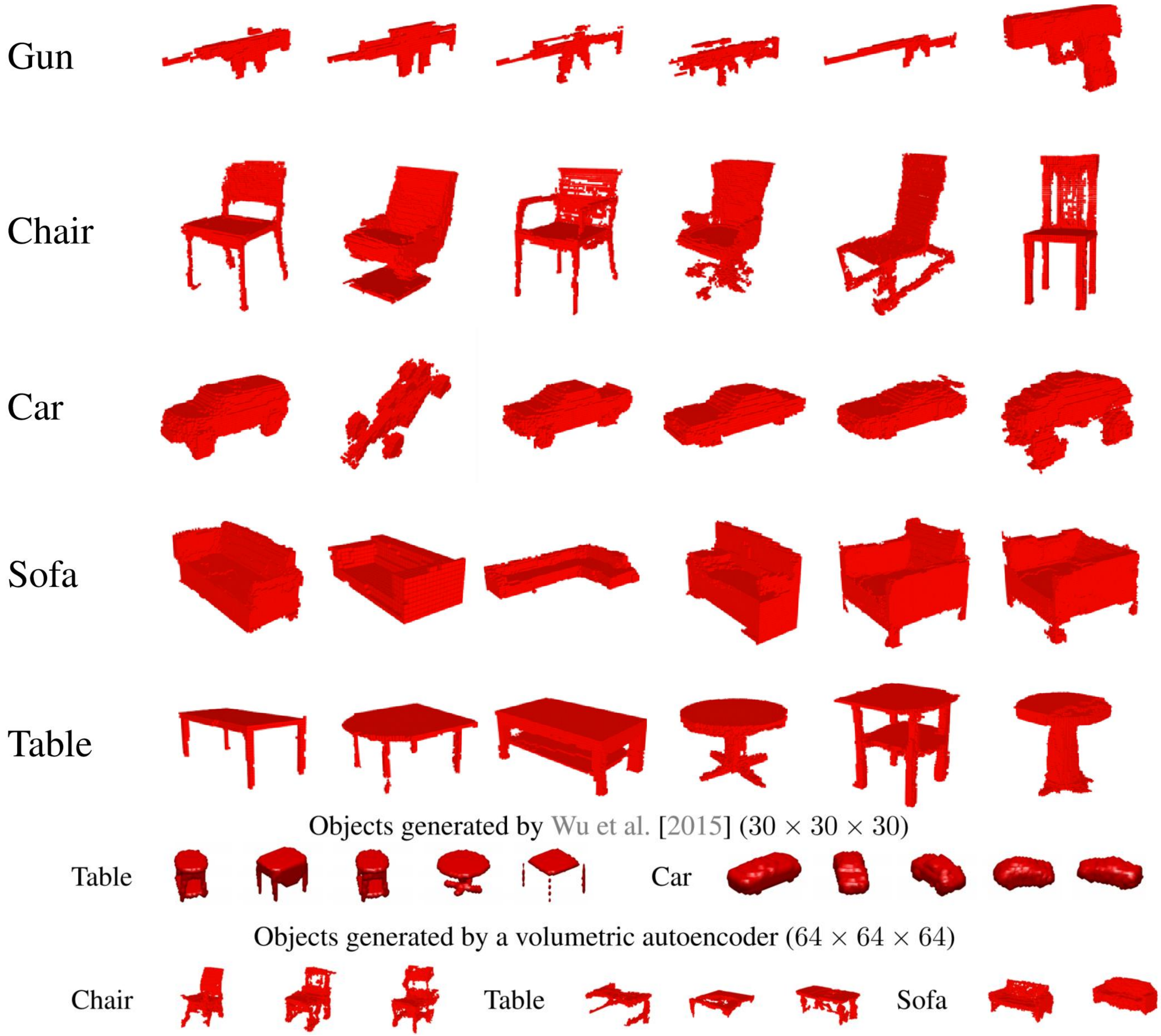
Convolutional Deep Belief Network $p(\mathbf{x}, y)$



Sampled Models

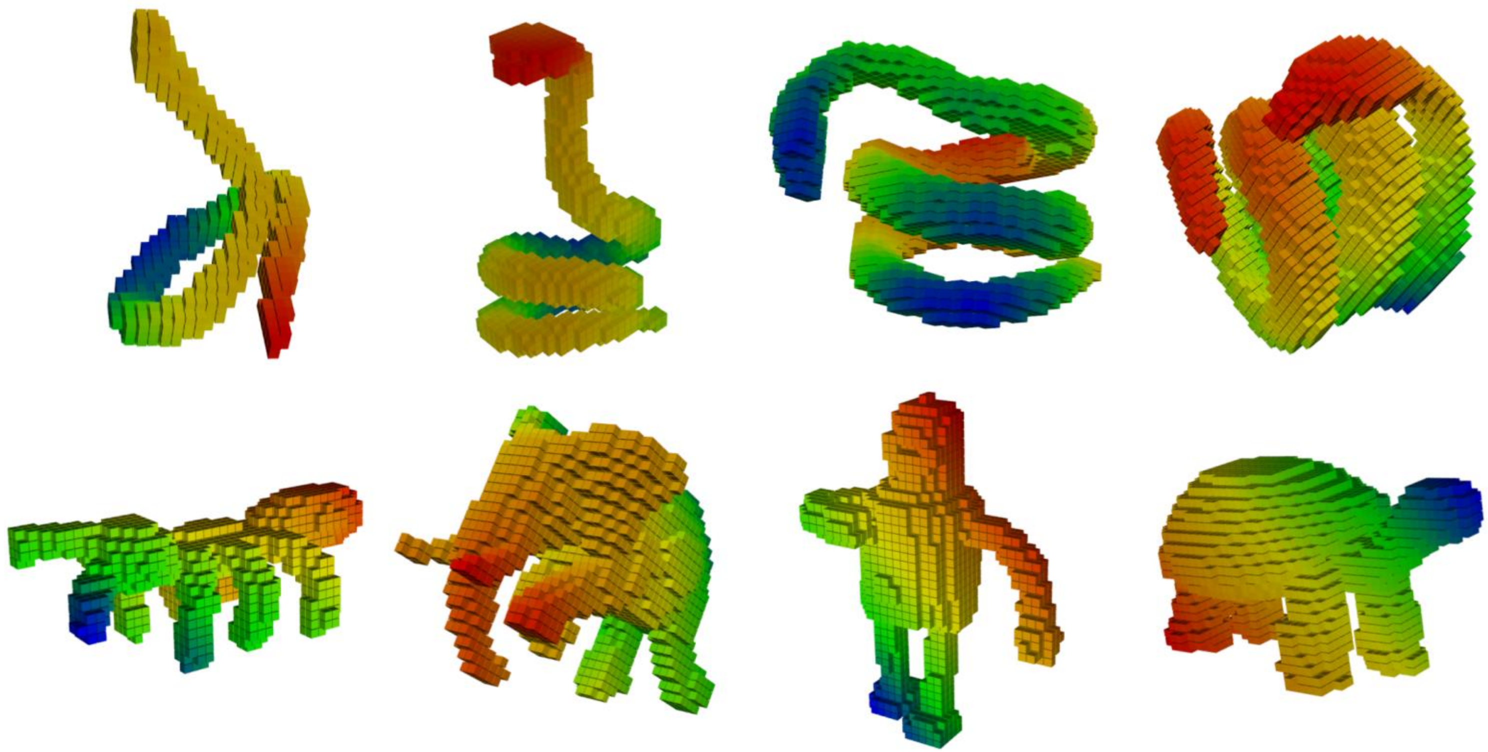
3D Generative Adversarial Network [Wu et al. 16]




Gun
Chair
Car
Sofa
Table
Objects generated by Wu et al. [2015] (30 × 30 × 30)
Table
Car
Objects generated by a volumetric autoencoder (64 × 64 × 64)
Chair
Table
Sofa

Sparse 3D Convolutional Networks

[Ben Graham 2016]



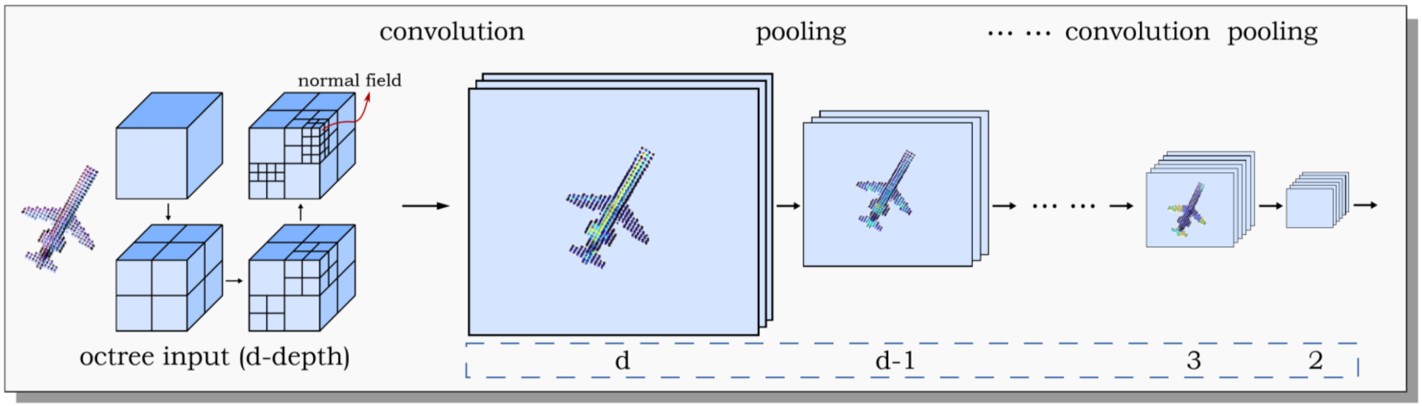
40x40x40 Grid

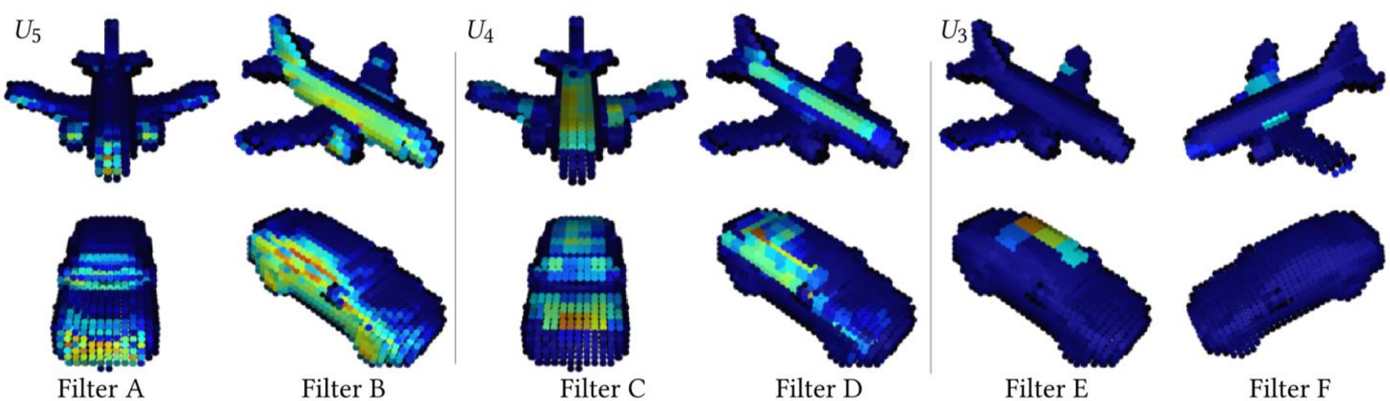
Sparsity for lower layers

Low resolution for upper layers

Octree classification networks

[Wang et al. 18]





The responses of some convolutional filters at different levels on two models are rendered. Red represents a large response and blue a low response.

Discussion

\+ Easy to implement

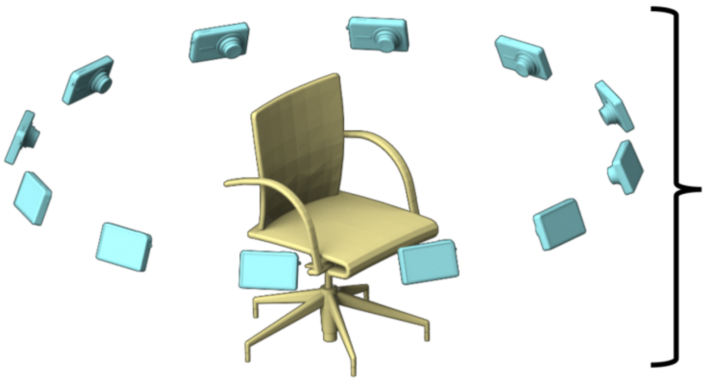
\+ Hardware friendly

\- Low resolution

\- No structural information

\- Cannot utilize 2D training data

Light Field Representation



3D shape model
rendered with
different virtual cameras

Method	Training Config.			Test Config.	Classification	Retrieval
	Pre-train	Fine-tune	#Views	#Views	(Accuracy)	(mAP)
(1) SPH [16]	-	-	-	-	68.2%	33.3%
(2) LFD [5]	-	-	-	-	75.5%	40.9%
(3) 3D ShapeNets [37]	ModelNet40	ModelNet40	-	-	77.3%	49.2%
(4) FV	-	ModelNet40	12	1	78.8%	37.5%
(5) FV, 12×	-	ModelNet40	12	12	84.8%	43.9%
(6) CNN	ImageNet1K	-	-	1	83.0%	44.1%
(7) CNN, f.t.	ImageNet1K	ModelNet40	12	1	85.1%	61.7%
(8) CNN, 12×	ImageNet1K	-	-	12	87.5%	49.6%
(9) CNN, f.t.,12×	ImageNet1K	ModelNet40	12	12	88.6%	62.8%
(10) MVCNN, 12×	ImageNet1K	-	-	12	88.1%	49.4%
(11) MVCNN, f.t., 12×	ImageNet1K	ModelNet40	12	12	89.9%	70.1%
(12) MVCNN, f.t.+metric, 12×	ImageNet1K	ModelNet40	12	12	89.5%	**80.2**%
(13) MVCNN, 80×	ImageNet1K	-	80	80	84.3%	36.8%
(14) MVCNN, f.t., 80×	ImageNet1K	ModelNet40	80	80	**90.1**%	70.4%
(15) MVCNN, f.t.+metric, 80×	ImageNet1K	ModelNet40	80	80	**90.1**%	79.5%

* f.t.=fine-tuning, metric=low-rank Mahalanobis metric learning

Discussion

+ Can utilize 2D training data
+ Efficient since using 2D convolutions
+ Top-performing algorithms

-- Redundancy
-- Loss of information per view
-- How to pick views?

? Convolutions on Spheres

Point cloud Representation

[Su et al. 17a, Su et al. 17b]

Object Classification on Partial Scans

Input:
Partial scan
(XYZ)

Output:
Category
classification

Car

Car

Airplan
e

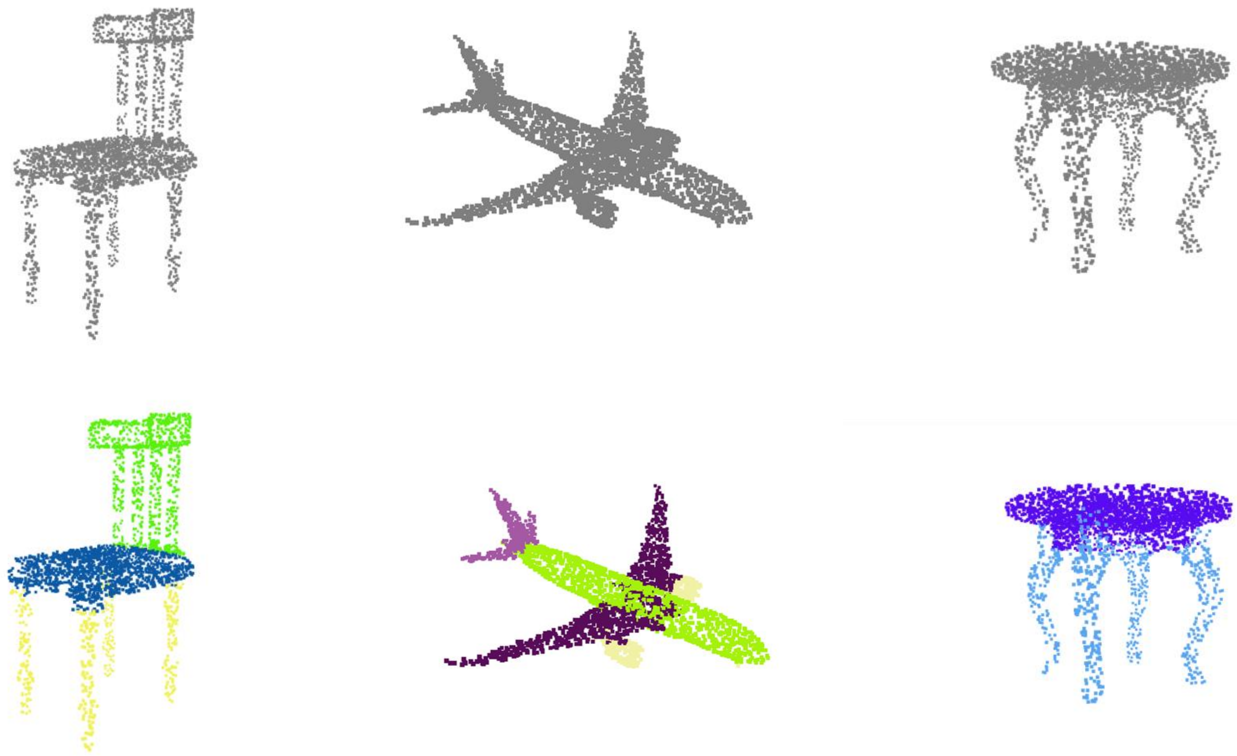
Tabl
e

Mu
g

Object Part Segmentation

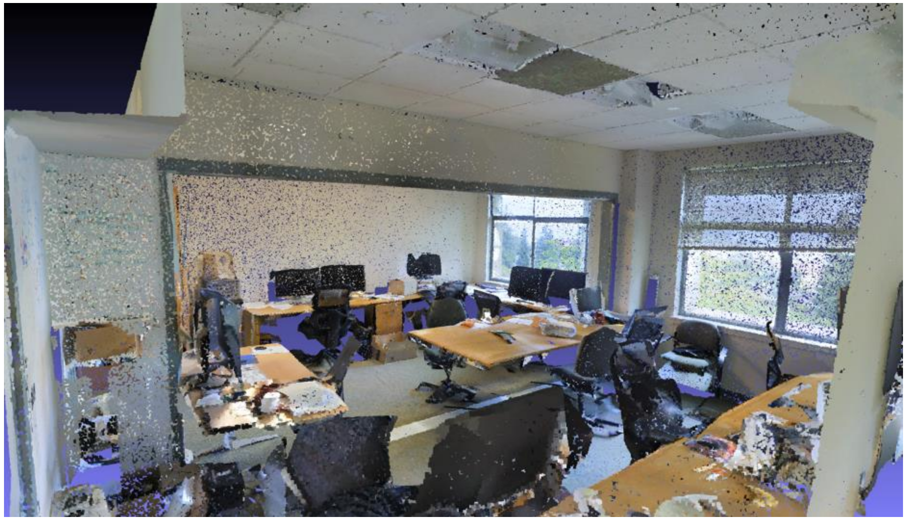
Input:
Point cloud (XYZ)

Output:
Per point label

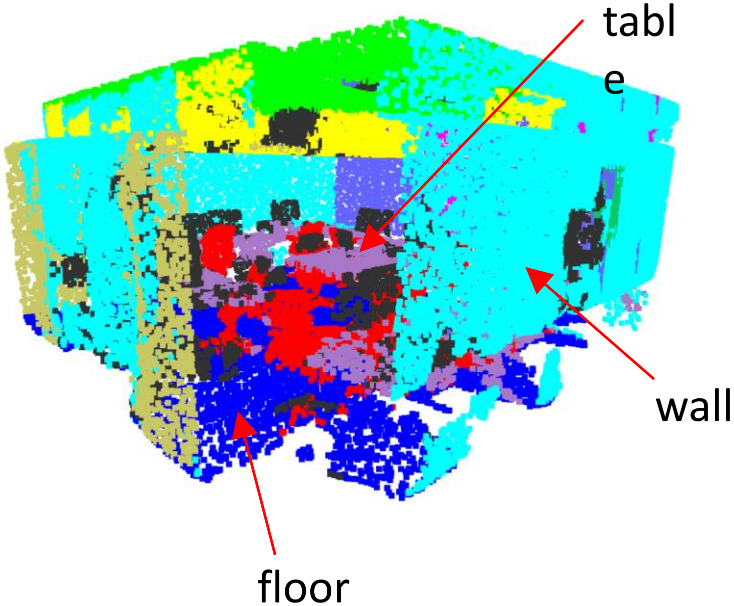


Semantic Segmentation for Indoor Scenes

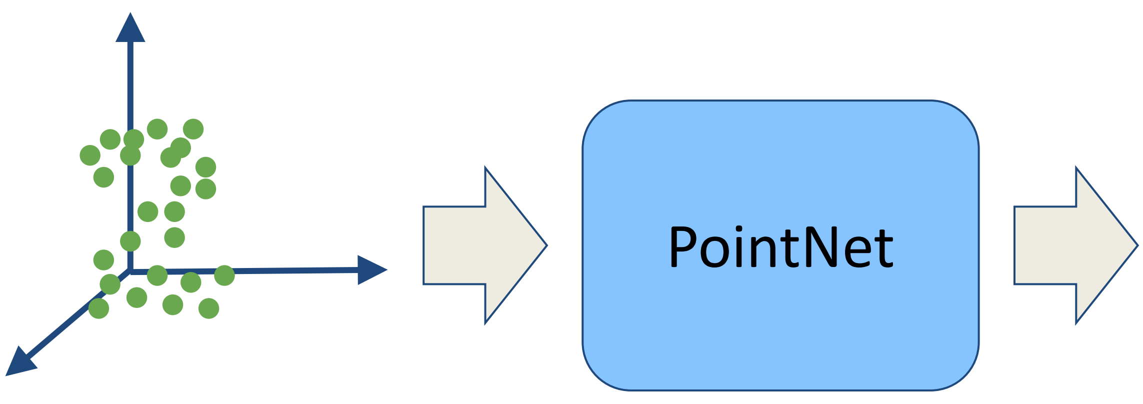
Input:
Point cloud (XYZRGB) of a room



Output (*current performance*):
Semantic segmentation of the room



Uniform Framework: PointNet



Point Set

Object Classification

Part Segmentation

Semantic Segmentation in scenes

Point Feature Learning

...

Theorem 1. *Suppose* $f : \mathcal{X} \to \mathbb{R}$ *is a continuous set function w.r.t Hausdorff distance* $d_H(\cdot,\cdot)$. $\forall \epsilon > 0$, $\exists$ *a continuous function* h *and a symmetric function* $g(x_1, \ldots, x_n) = \gamma \circ MAX$, *such that for any* $S \in \mathcal{X}$,

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

where $x_1, \ldots, x_n$ *is the full list of elements in* S *ordered arbitrarily,* γ *is a continuous function, and MAX is a vector max operator that takes* n *vectors as input and returns a new vector of the element-wise maximum.*

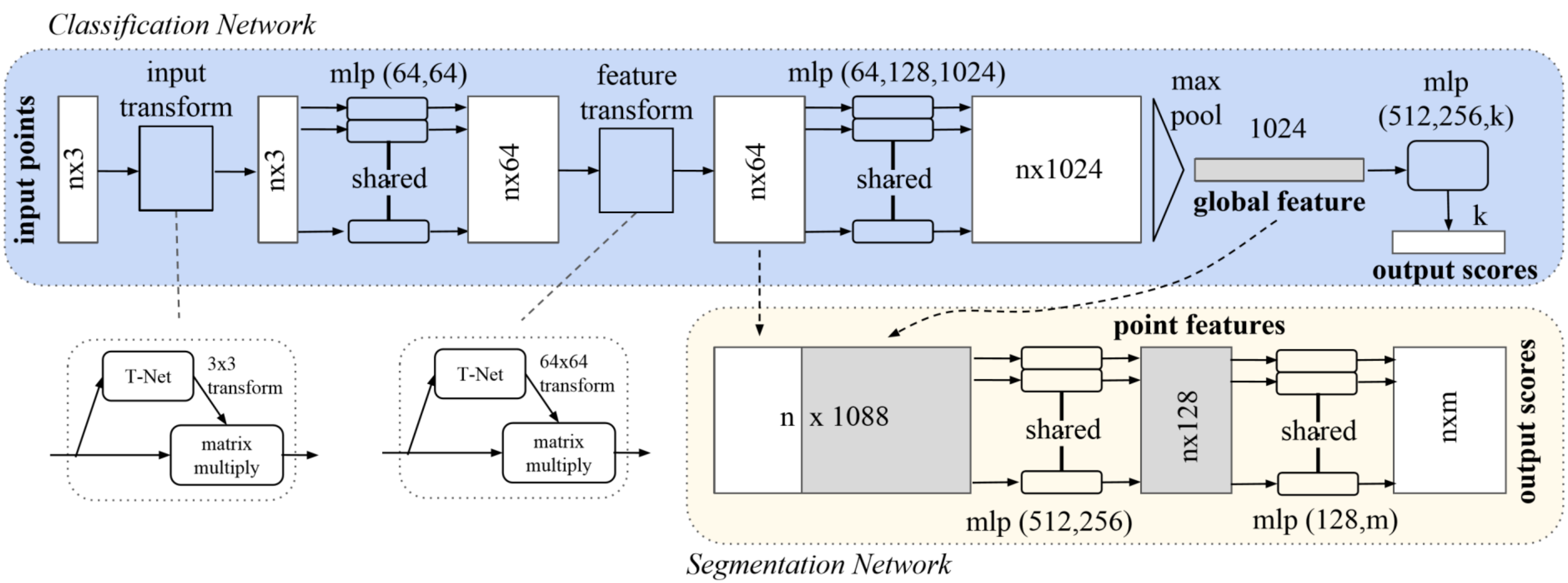


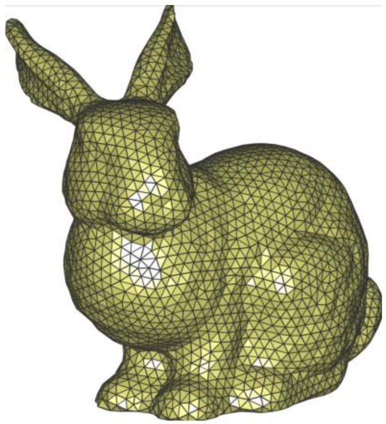
Figure 2. PointNet Architecture. The classification network takes n points as input, applies input and feature transformations, and then aggregates point features by max pooling. The output is classification score for k classes. The segmentation network is an extension to the classification net. It concatenates global and local features and outputs per point scores. "mlp" stands for multi-layer perceptron, the numbers in brackets are its layer sizes. Batchnorm is used for all layers with ReLU. Dropout layers are used for the last mlp in classification net.

ModelNet shape 40-class classification

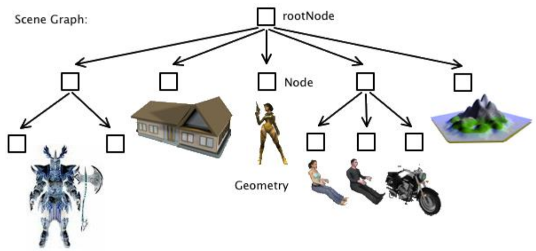
Model	Accuracy
MLP	40%
LSTM	75%
Conv-Max-FC (1 max)	84%
Conv-Max-FC (2 max)	86%
Conv-Max-FC (2 max) + Input Transform	87.8%
Conv-Max-FC (2 max) + Feature Transform	86.8%
Conv-Max-FC (2 max) + Feature Transform + orthogonal regularization	87.4%
Conv-Max-FC (2 max) + Input Transform + Feature Transform + orthogonal regularization	**88.9%**

Best Volumetric CNN: 89.1%
However, PointNet is around 5x - 10x faster than Volumetric CNN

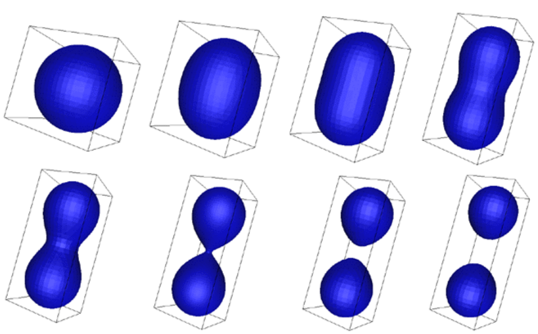
3D Surface Representations



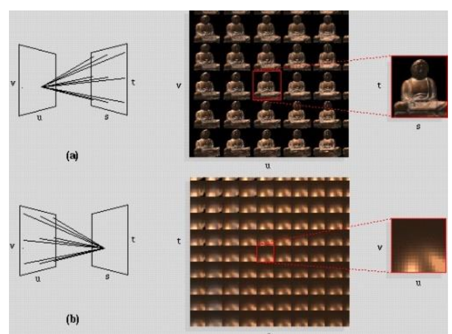
Triangular mesh



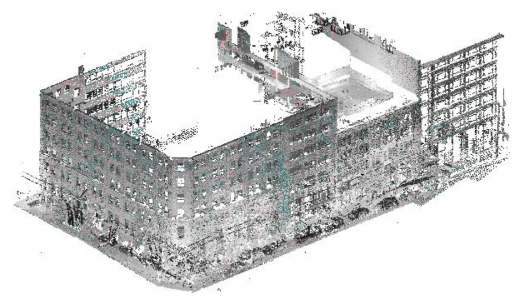
Part-based models



Implicit surface



Light Field Representation



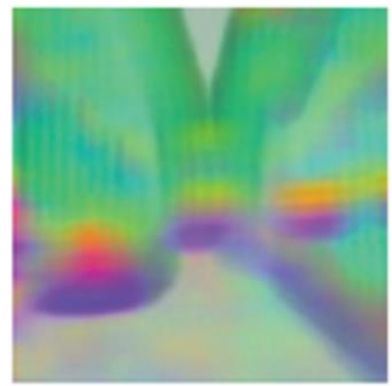
Point cloud

Matching in Embedding Spaces

[Wei, Huang, Ceylan, Vouga, Li 16]

Existing methods usually follow a two-step approach (e.g., SIFT flow)

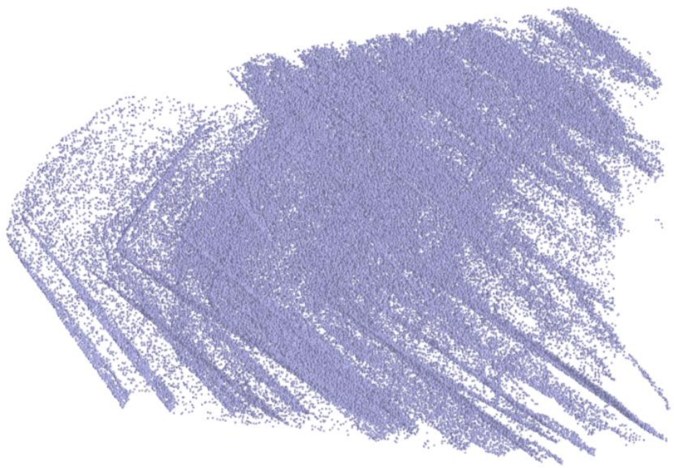
- Local descriptor computation



- Dense pixel labeling via MRF inference
  - Preserve descriptors
  - Preserve smoothness

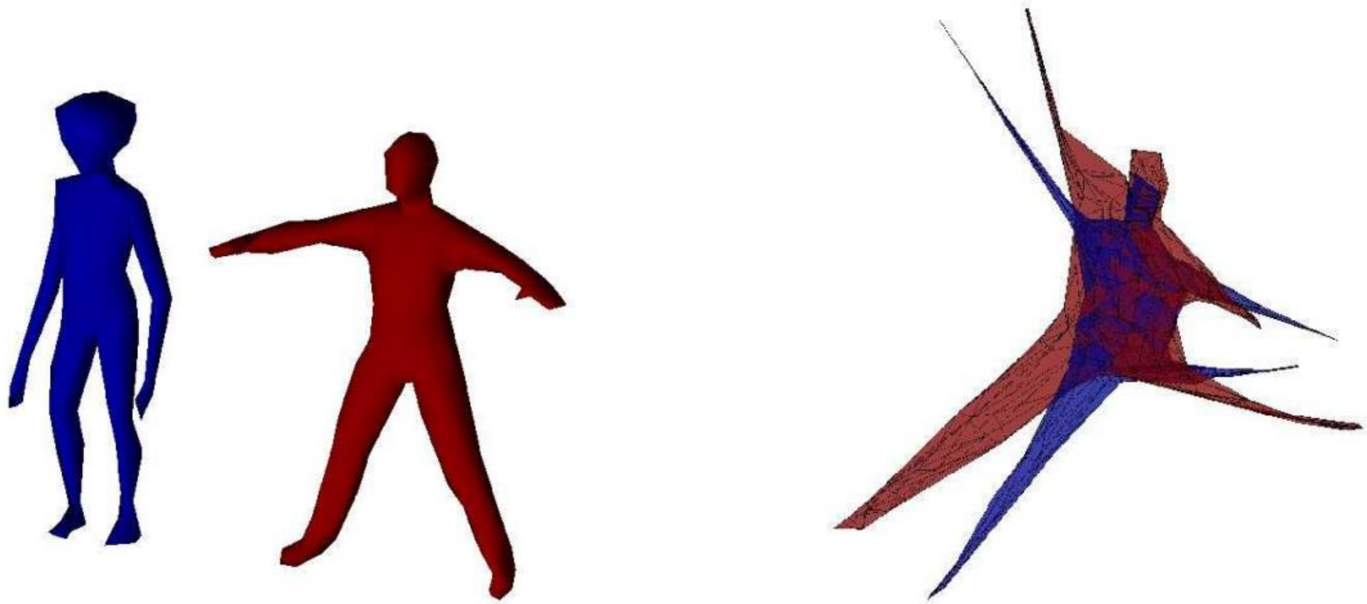
Issues of such two-step approach

Partial similarity



Inefficient when
matching multiple objects

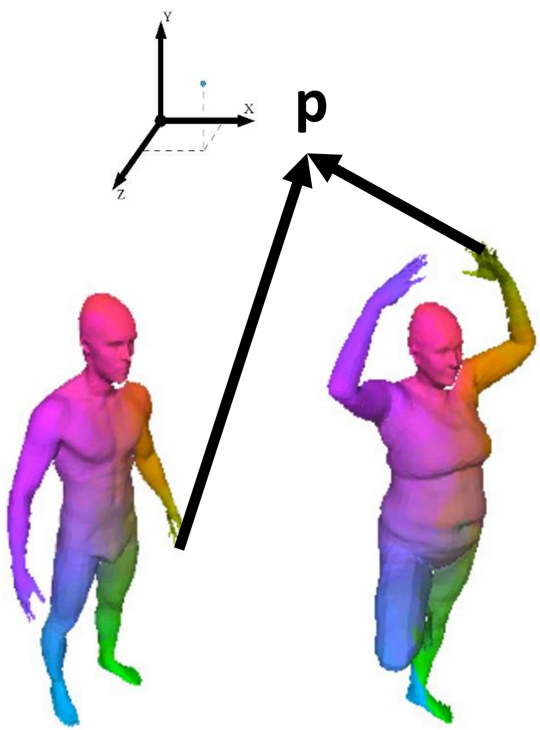
Embedding --- establishing correspondences in the embedding space



Spectral embedding [Liu et al. 06]

Sensitive to 1) partial similarity, and 2) geometric and topological changes

Properties of the desired embedding space



Corresponding points are matched in the embedding space

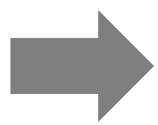
Embedding preserves continuity

The benefits of object embedding

- Correspondences become nearest neighbor query
  - Efficiency for multiple object matching
    $O(n)$ embeddings + $O(n^2)$ queries
  - Partial similarity
  - Fuzzy correspondences

The biggest message of deep neural networks

- Approximate any function given sufficient data



Focus on depth images

- Scanning devices generate depth images

- Complete shape embedding are aggregated from depth image embeddings
  - 3D convolution is not ready yet

Architecture

	0	1	2	3	4	5	6	7	8	9	10
layer	**image**	**conv**	**max**	**conv**	**max**	**2×conv**	**conv**	**max**	**2×conv**	**int**	**conv**
filter-stride	-	11-4	3-2	5-1	3-2	3-1	3-1	3-2	1-1	-	3-1
channel	1	96	96	256	256	384	256	256	4096	4096	16
activation	-	relu	lrn	relu	lrn	relu	relu	idn	relu	idn	relu
size	512	128	64	64	32	32	32	16	16	128	512
num	1	1	4	4	16	16	16	64	64	1	1

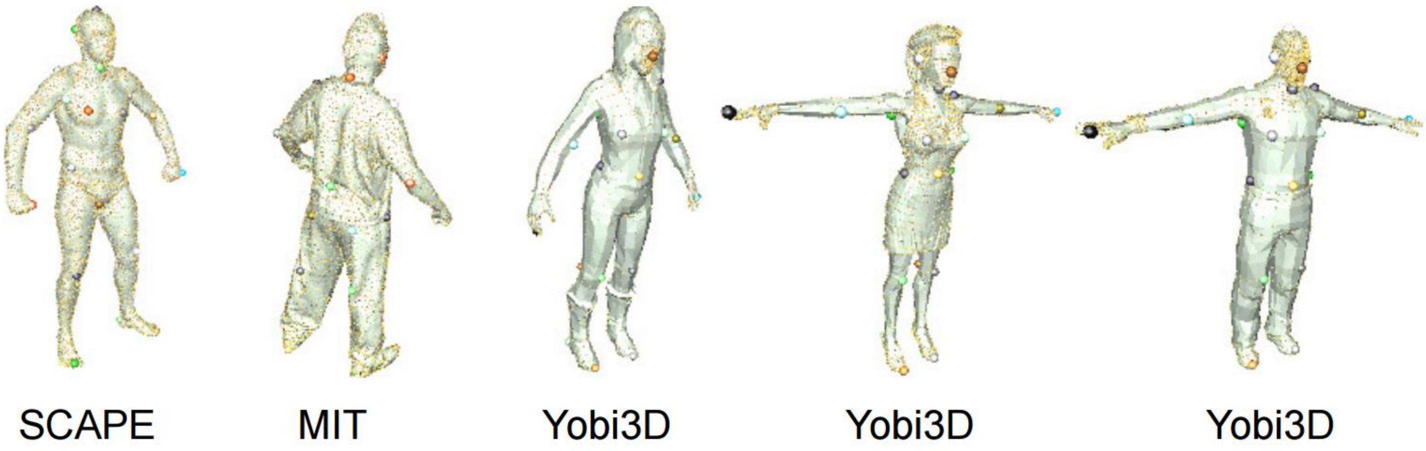
The input is a depth image

The output is a per-pixel descriptor (dim 16)

Convolution + Deconvolution

Training data

- 4 animation sequences (dense correspondences)
- 2500 shapes from Yobi3D (33 feature points)

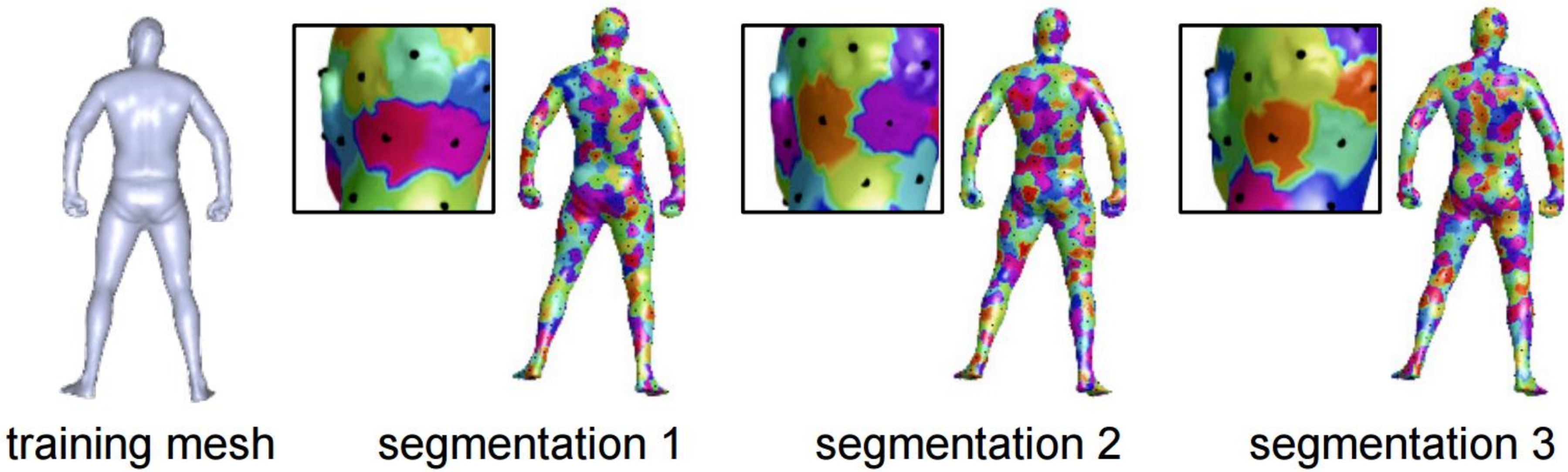


Direct versus Indirect

- Descriptor learning (e.g., triplet loss [Schroff et al. 15])

- Classification loss (e.g., the second last layer of AlexNet)

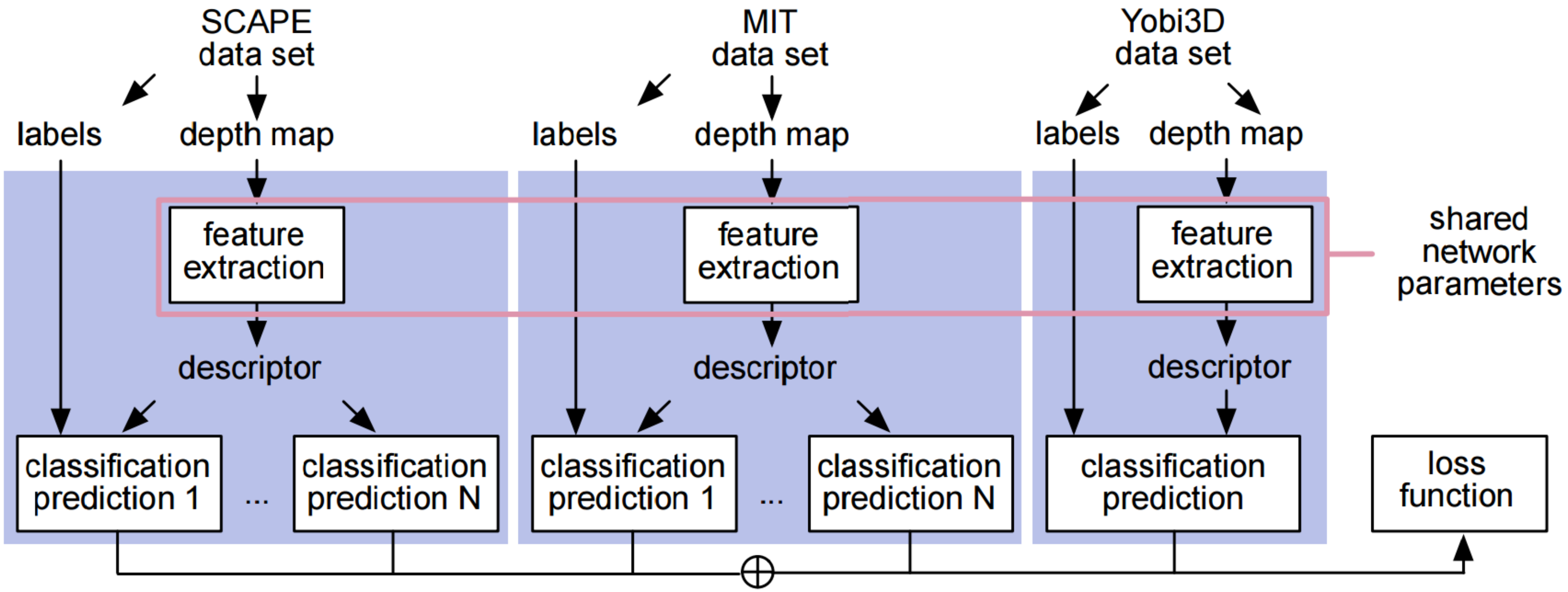
We employ a classification loss



Classes are defined in terms of super-patches

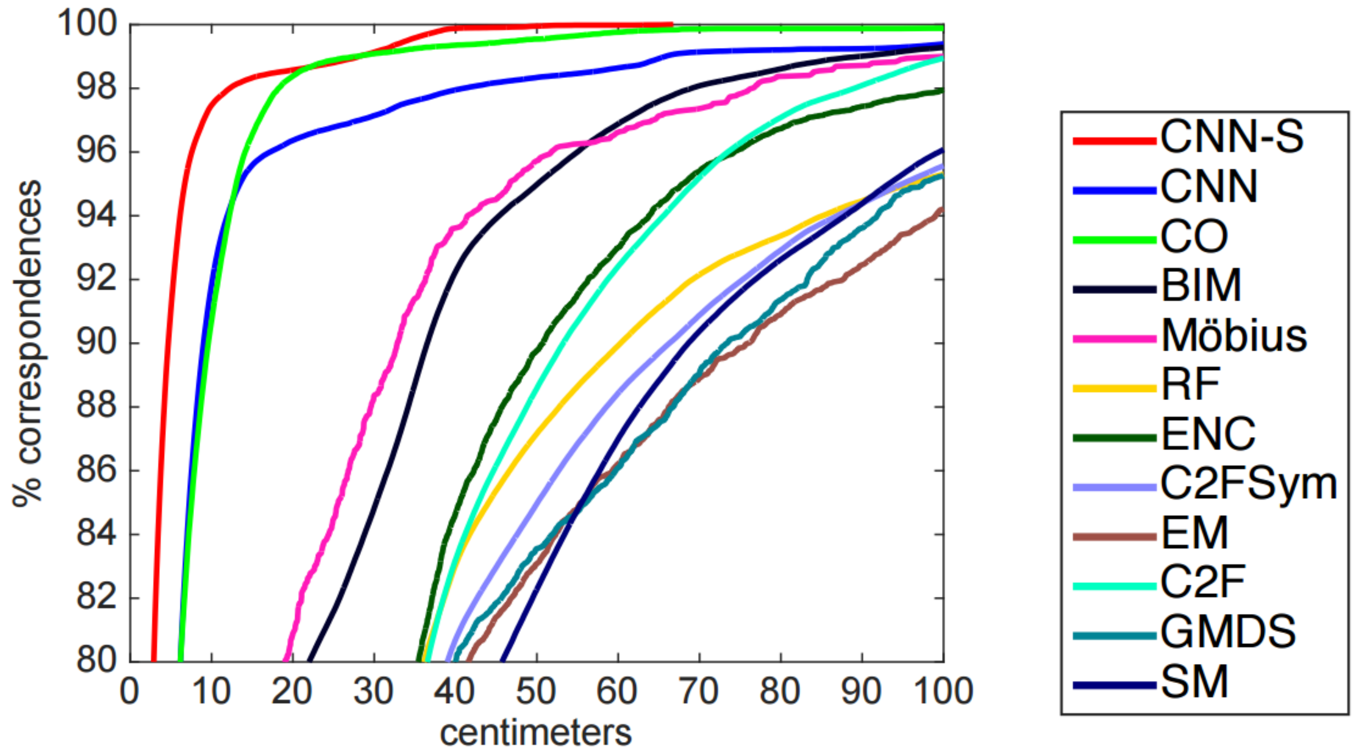
We use multiple segmentations --- so the probability of two points belong to the same segment is related to their distance

We employ the classification loss



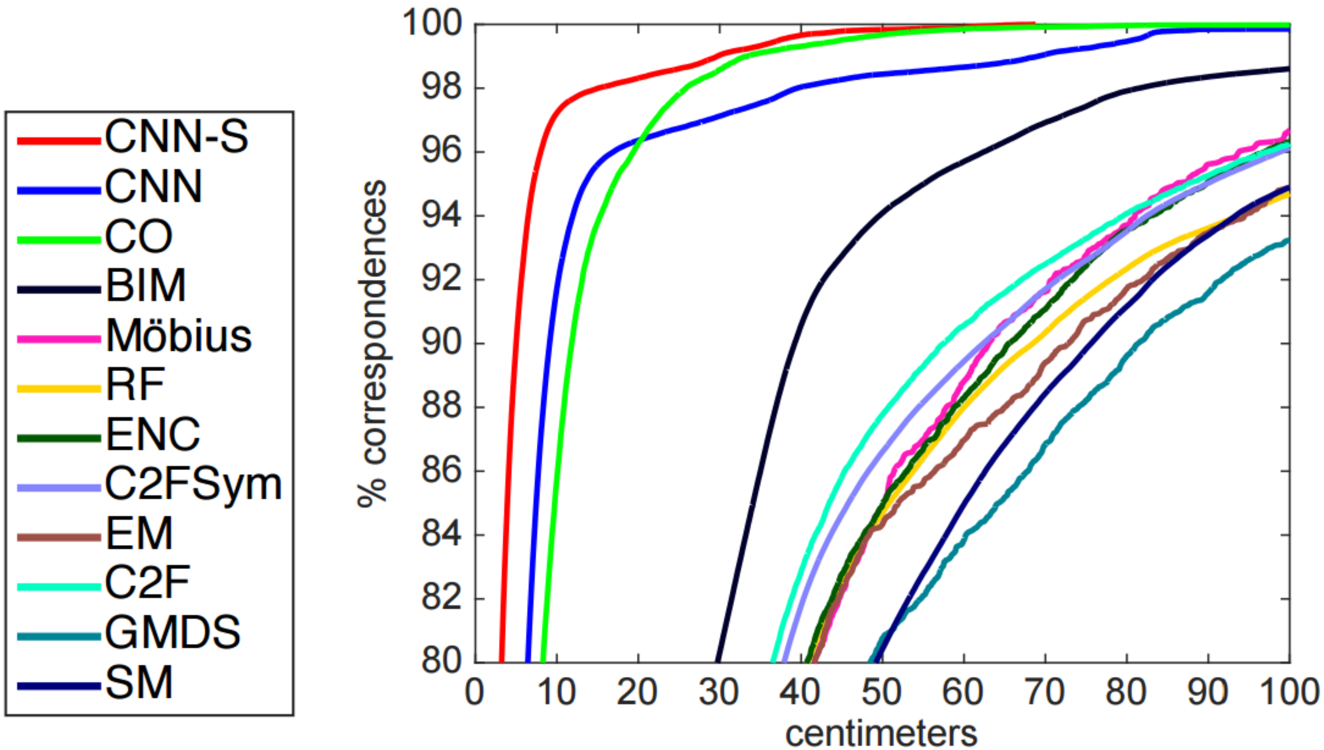
$$\{\mathbf{w}_i^\star\}, \mathbf{w}^\star = \arg\min_{\{\mathbf{w}_i\},\mathbf{w}} \sum_{i=1}^{M} l(\mathbf{w}_i, \mathbf{w})$$

Evaluation on the FAUST dataset



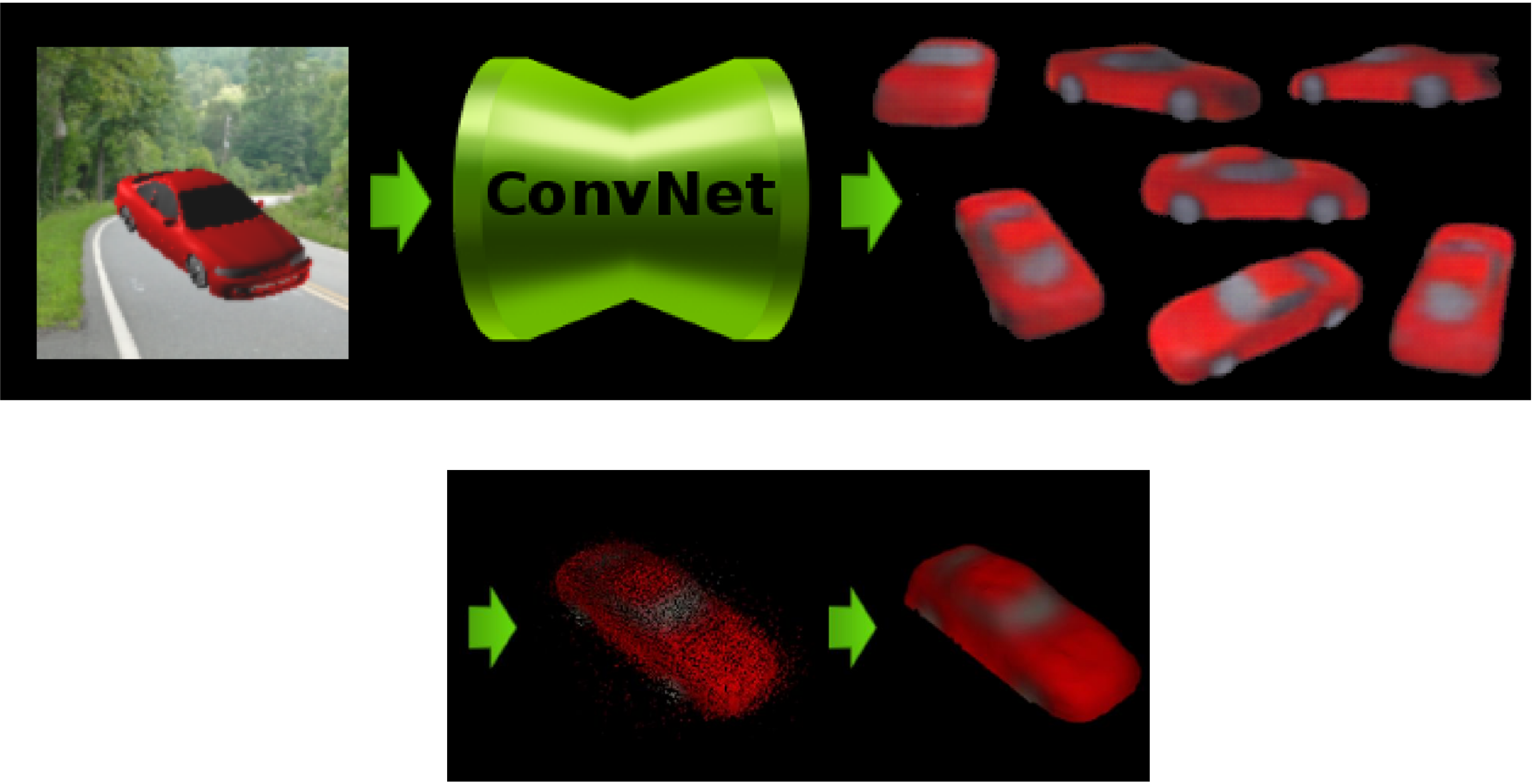
Cumulative error distribution, intra-subject

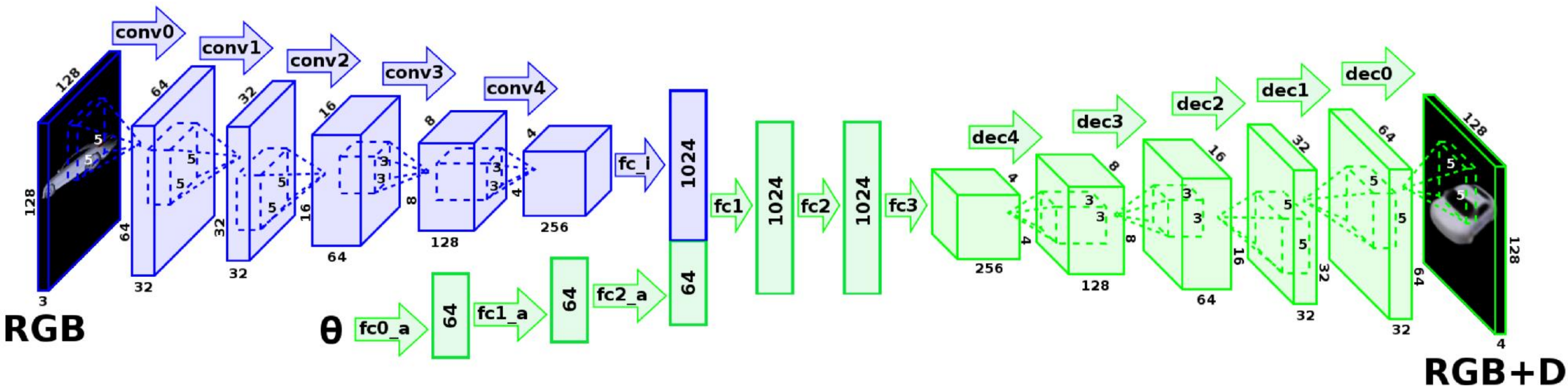
Evaluation on the FAUST dataset



Cumulative error distribution, inter-subject

Multi-view 3D Models from Single Images With a Convolutional Network [ECCV' 16]


ConvNet


RGB
conv0
conv1
conv2
conv3
conv4
fc_i
1024
θ
fc0_a
64
fc1_a
64
fc2_a
64
fc1
1024
fc2
1024
fc3
dec4
dec3
dec2
dec1
dec0
RGB+D

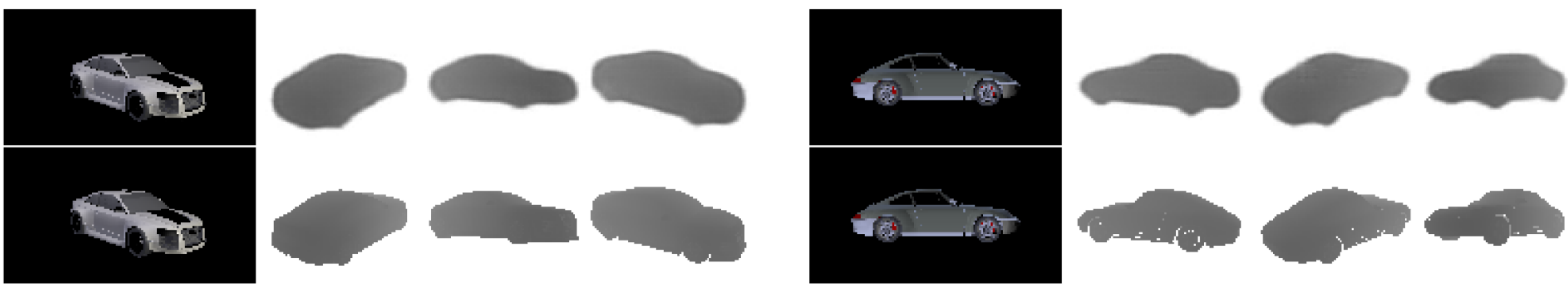


Fig. 5. Depth map predictions (**top row**) and the corresponding ground truth (**bottom row**). The network correctly estimates the shape.

Multi-view 3D Models from Single Images with a Convolutional Network

Maxim Tatarchenko, Alexey Dosovitskiy, Thomas Brox

Department of Computer Science
University of Freiburg
{tatarchm, dosovits, brox}@cs.uni-freiburg.de

ECCV 2016

Perspective Transformer Nets: Learning Single-View 3D Object Reconstruction without 3D Supervision [Yan et al. 16]

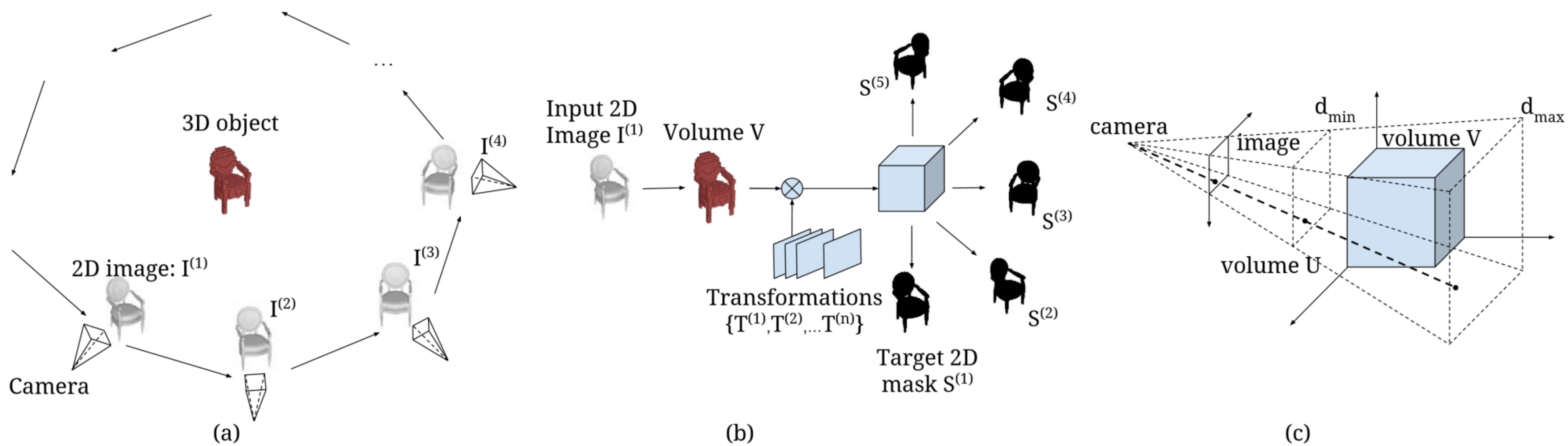
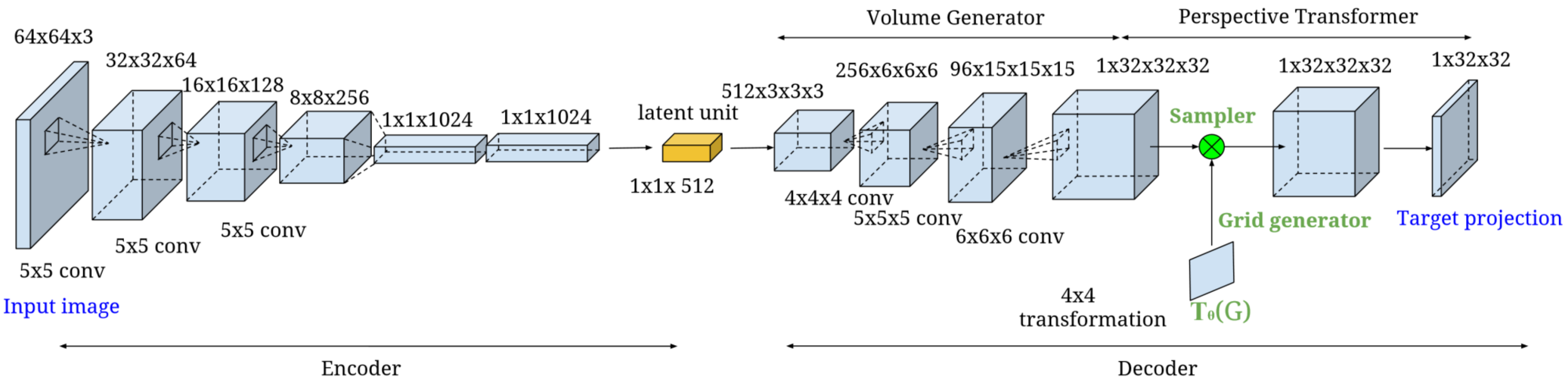


Figure 1: (a) Understanding 3D object from learning agent's perspective; (b) Single-view 3D volume reconstruction with perspective transformation. (c) Illustration of perspective projection. The minimum and maximum disparity in the screen coordinates are denoted as d_{min} and d_{max}.

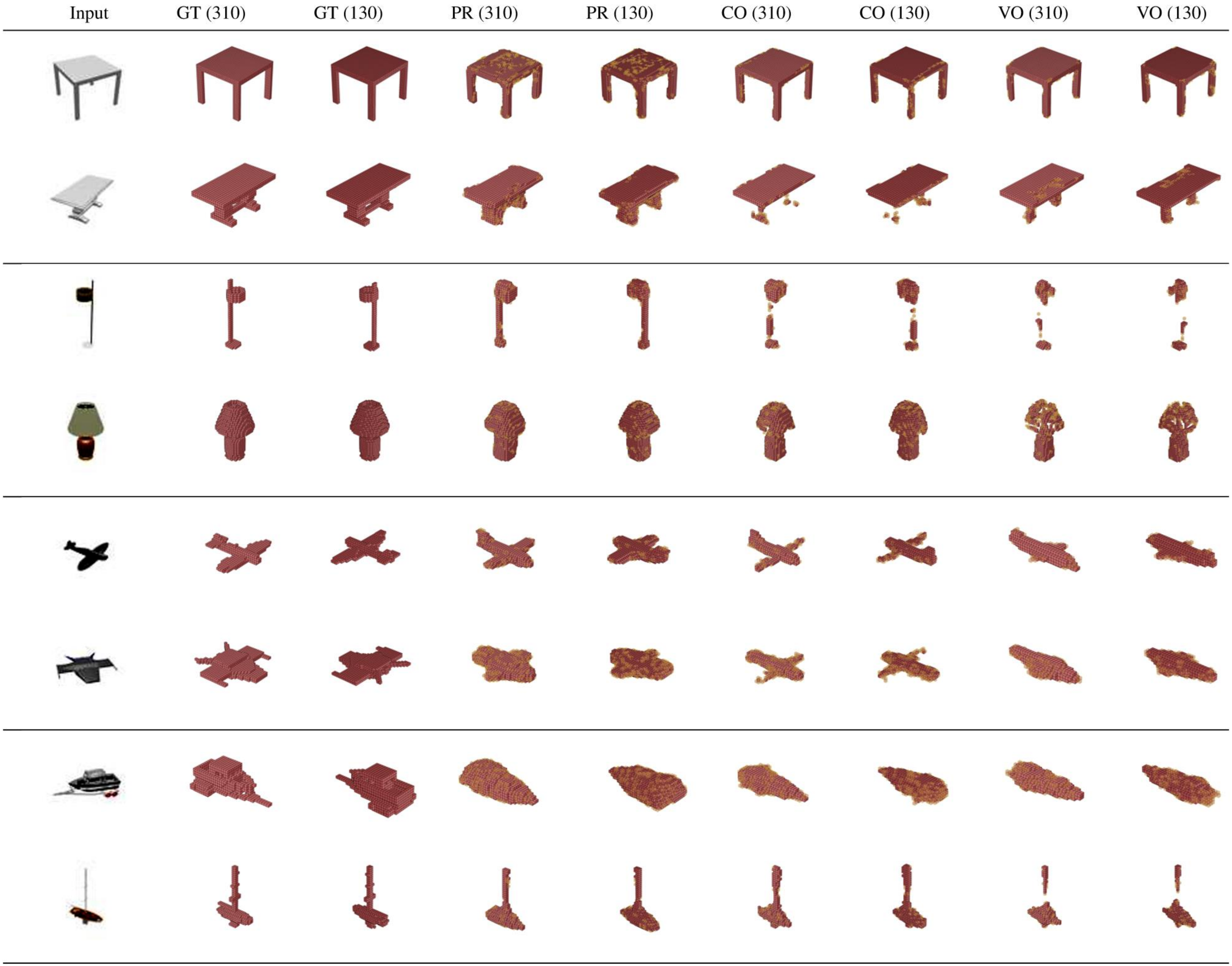
$$\mathcal{L}_{vol}(I^{(k)}) = ||f(I^{(k)}) - \mathbf{V}||_2^2$$

$$\mathcal{L}_{proj}(I^{(k)}) = \sum_{j=1}^{n} \mathcal{L}_{proj}^{(j)}(I^{(k)}; S^{(j)}, \alpha^{(j)}) = \frac{1}{n} \sum_{j=1}^{n} ||P(f(I^{(k)}); \alpha^{(j)}) - S^{(j)}||_2^2$$

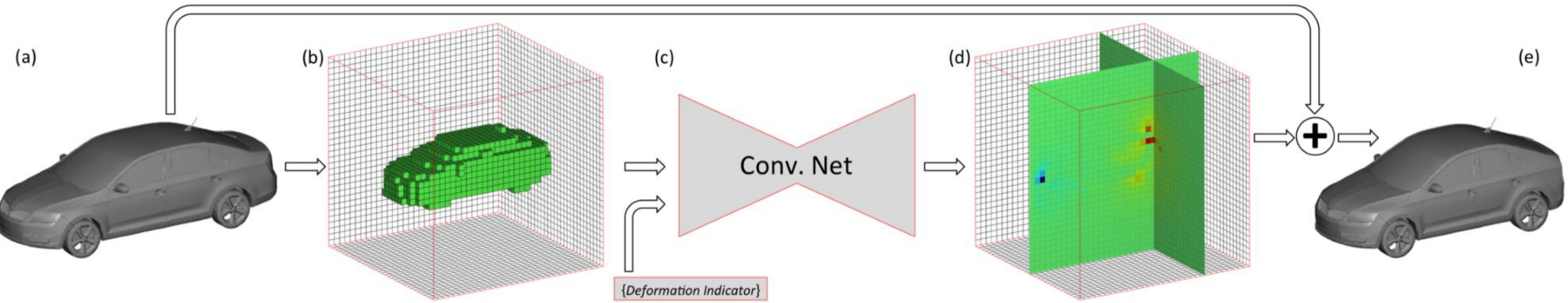
$$\mathcal{L}_{comb}(I^{(k)}) = \lambda_{proj}\mathcal{L}_{proj}(I^{(k)}) + \lambda_{vol}\mathcal{L}_{vol}(I^{(k)})$$



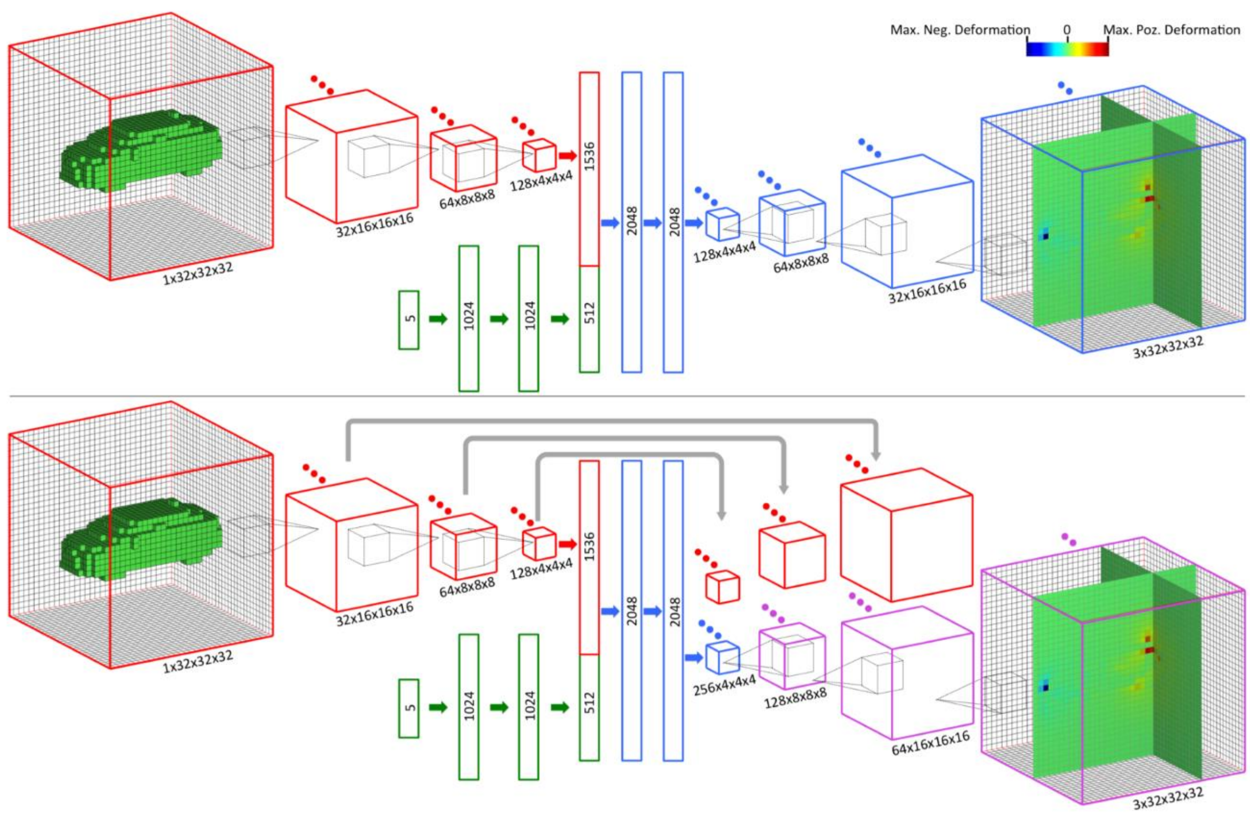
Volume Generator
Perspective Transformer
64x64x3
32x32x64
16x16x128
8x8x256
1x1x1024
1x1x1024
latent unit
1x1x 512
512x3x3x3
256x6x6x6
96x15x15x15
1x32x32x32
1x32x32x32
1x32x32
Sampler
Grid generator
Target projection
4x4x4 conv
5x5x5 conv
6x6x6 conv
5x5 conv
5x5 conv
5x5 conv
Input image
4x4
transformation
Tθ(G)
Encoder
Decoder


Input
GT (310)
GT (130)
PR (310)
PR (130)
CO (310)
CO (130)
VO (310)
VO (130)

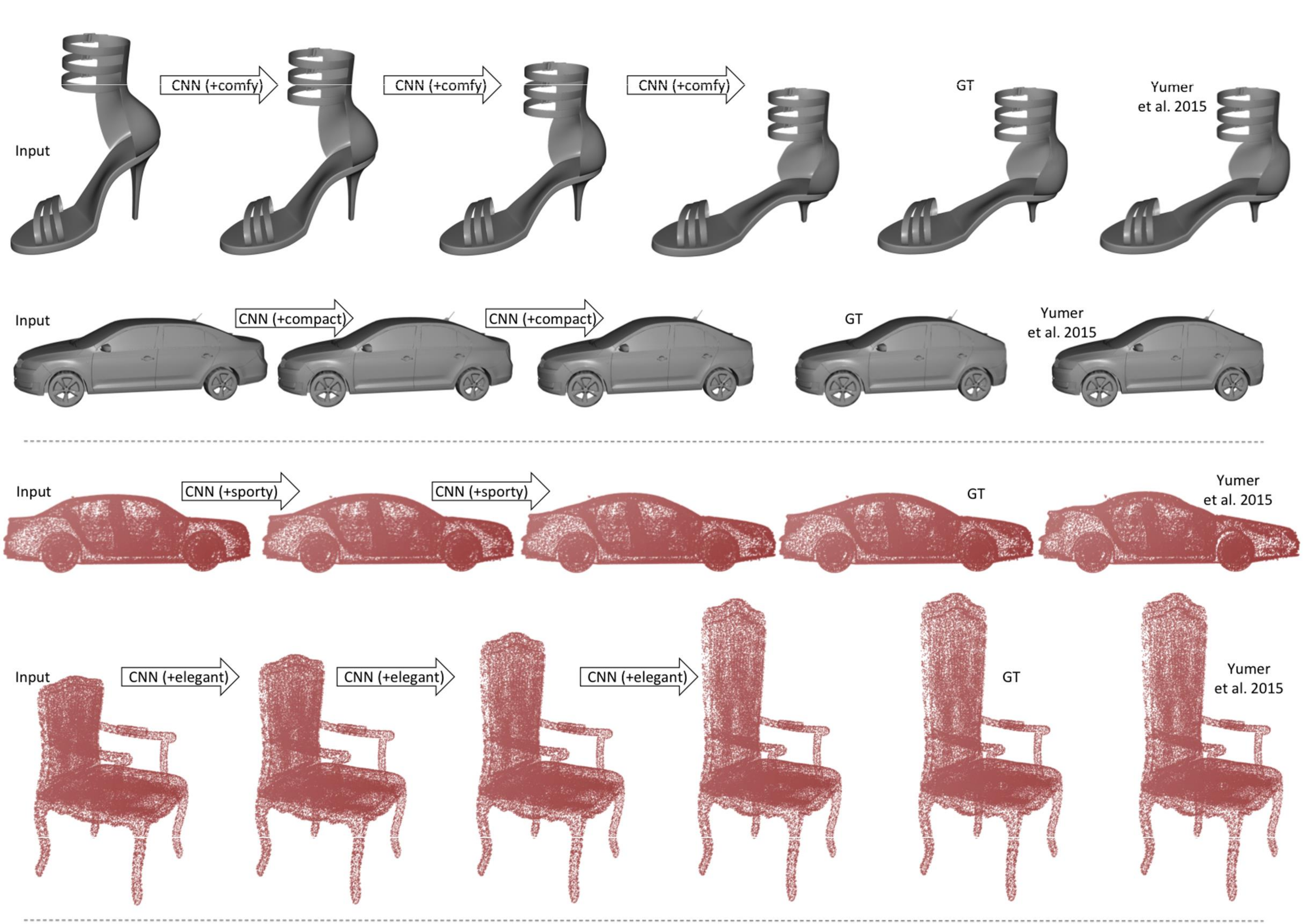
Learning Semantic Deformation Flows with 3D Convolutional Networks [Yumer and Mitra 2016]



(a)
(b)
(c)
(d)
(e)
Conv. Net
{Deformation Indicator}

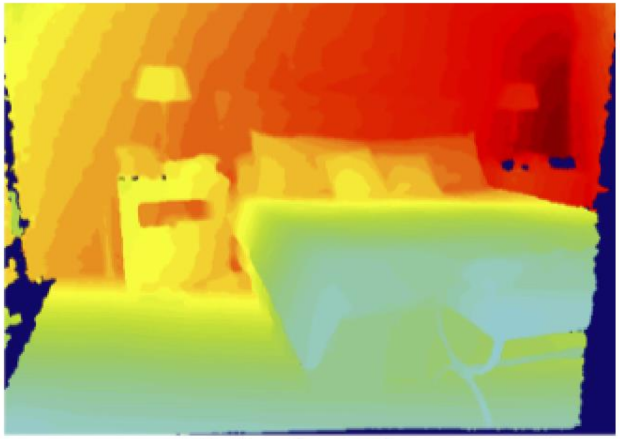


Max. Neg. Deformation
0
Max. Poz. Deformation
1x32x32x32
32x16x16x16
64x8x8x8
128x4x4x4
1536
512
5
1024
1024
2048
2048
128x4x4x4
64x8x8x8
32x16x16x16
3x32x32x32
1x32x32x32
32x16x16x16
64x8x8x8
128x4x4x4
1536
512
5
1024
1024
2048
2048
256x4x4x4
128x8x8x8
64x16x16x16
3x32x32x32



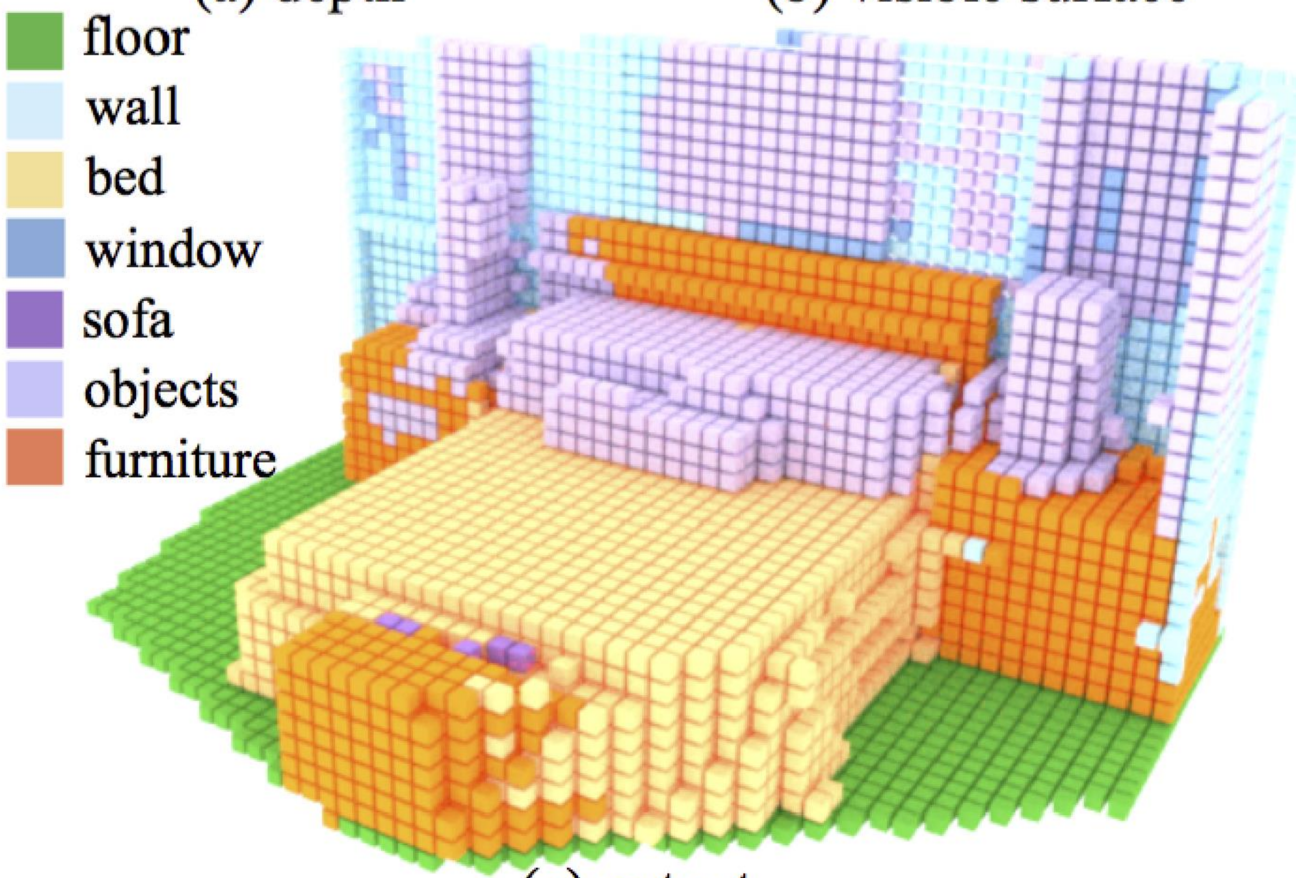
Input
CNN (+comfy)
CNN (+comfy)
CNN (+comfy)
GT
Yumer et al. 2015
Input
CNN (+compact)
CNN (+compact)
GT
Yumer et al. 2015
Input
CNN (+sporty)
CNN (+sporty)
GT
Yumer et al. 2015
Input
CNN (+elegant)
CNN (+elegant)
CNN (+elegant)
GT
Yumer et al. 2015

Semantic Scene Completion from a Single Depth Image [Song et al. 17]

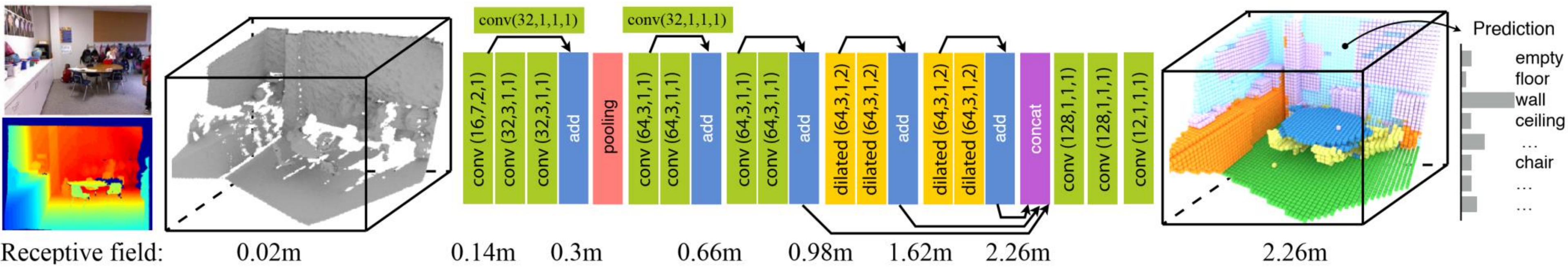


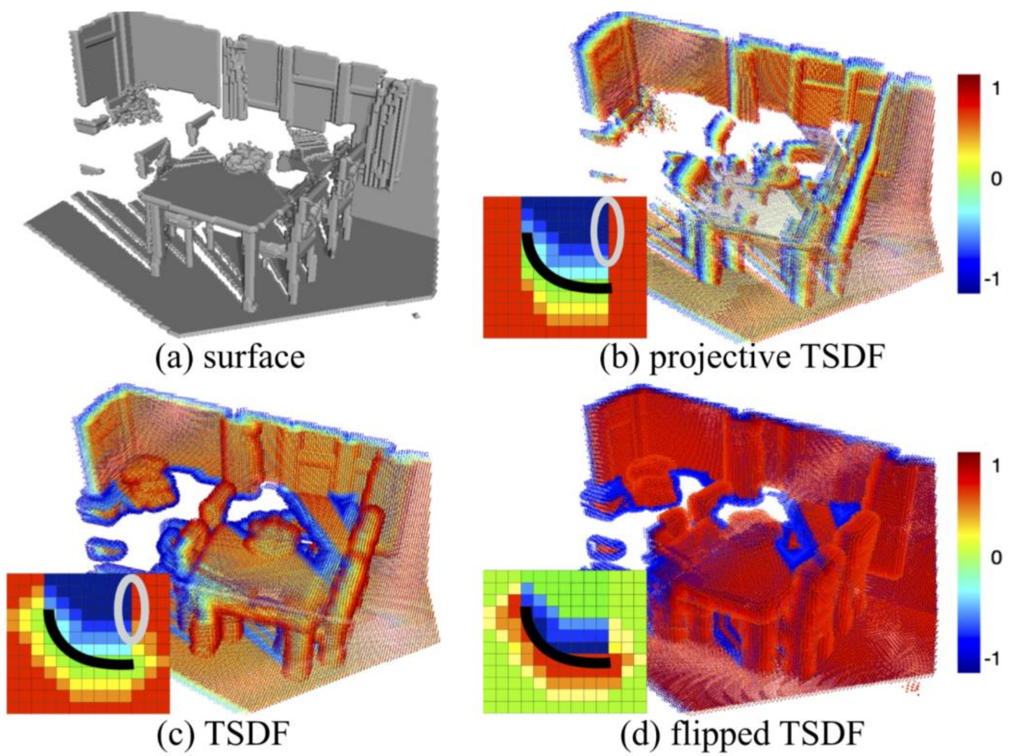
(a) depth

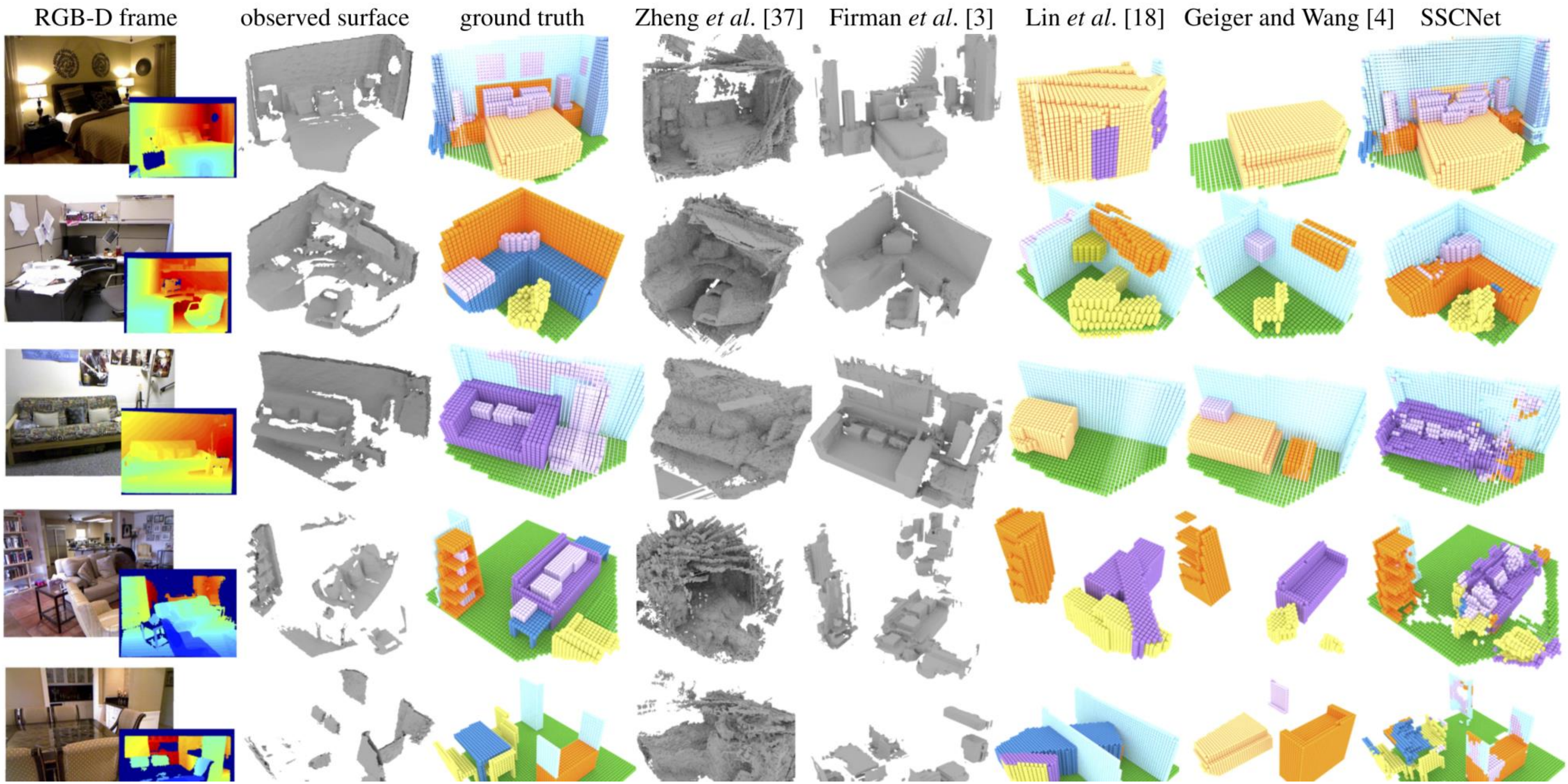
(b) visible surface



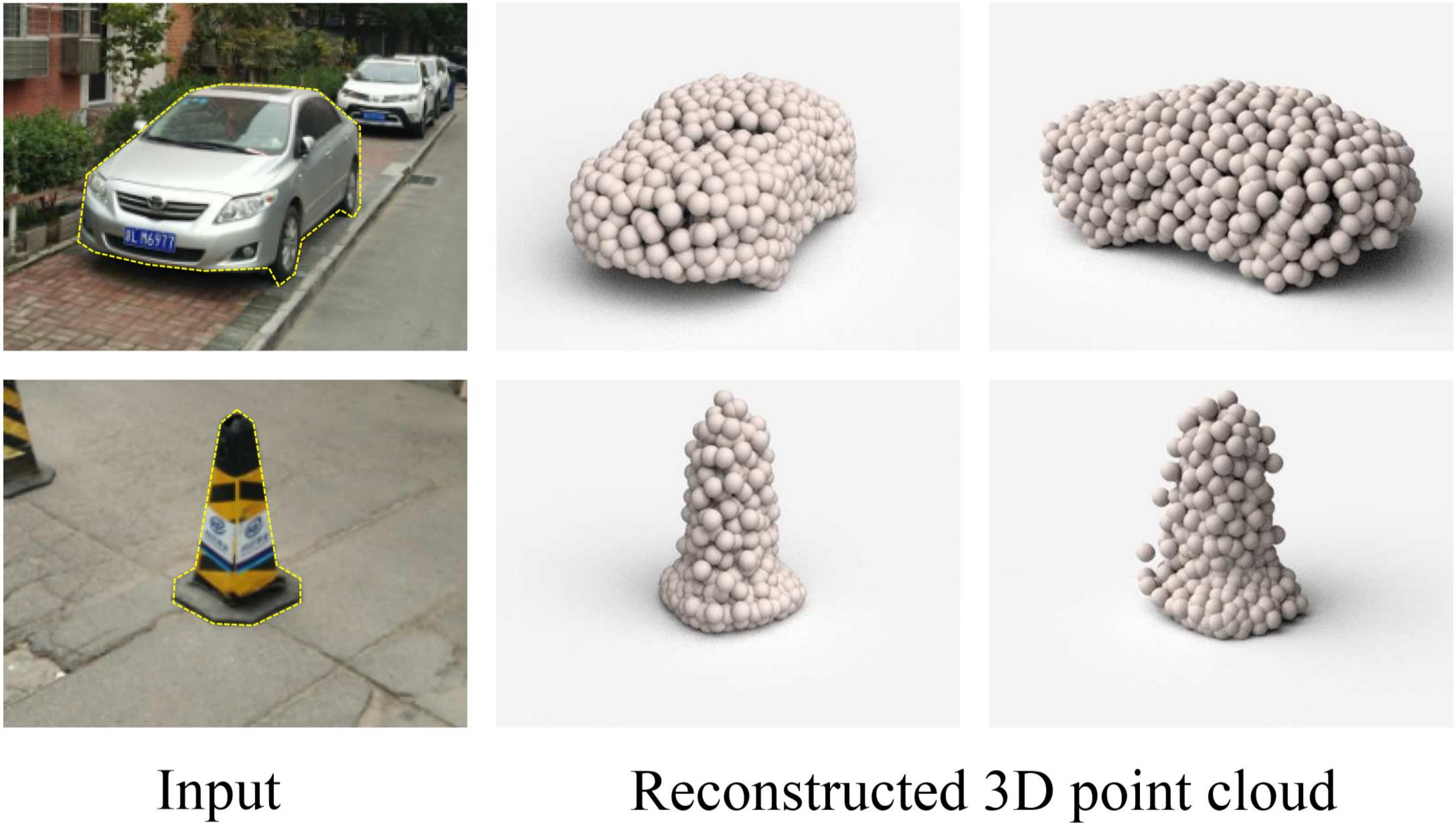
(c) output


conv(32,1,1,1)
conv(32,1,1,1)
conv (16,7,2,1)
conv (32,3,1,1)
conv (32,3,1,1)
add
pooling
conv (64,3,1,1)
conv (64,3,1,1)
add
conv (64,3,1,1)
conv (64,3,1,1)
add
dilated (64,3,1,2)
dilated (64,3,1,2)
add
dilated (64,3,1,2)
dilated (64,3,1,2)
add
concat
conv (128,1,1,1)
conv (128,1,1,1)
conv (12,1,1,1)
Prediction
empty
floor
wall
ceiling
…
chair
…
…
Receptive field:
0.02m
0.14m
0.3m
0.66m
0.98m
1.62m
2.26m
2.26m

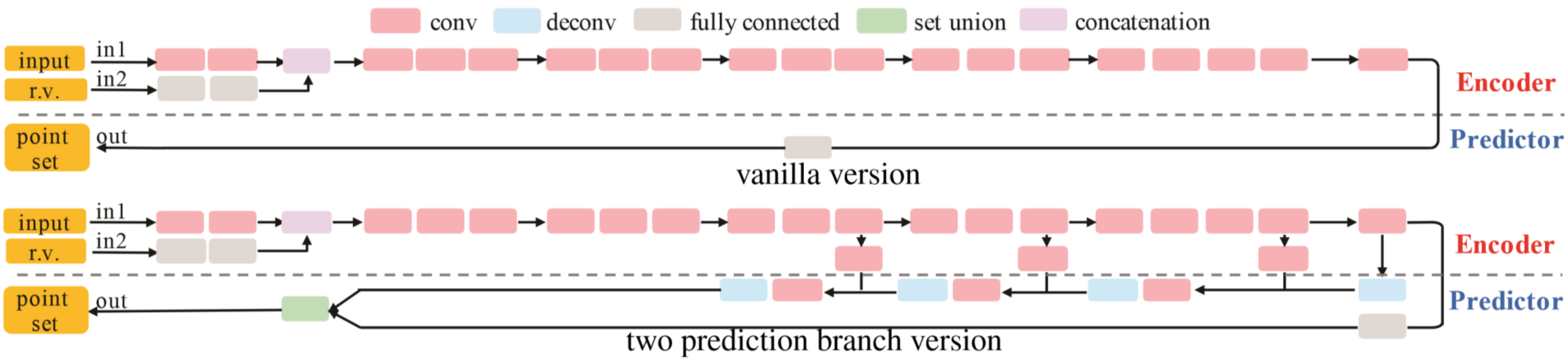

1
0
-1
1
0
-1
(a) surface
(b) projective TSDF
(c) TSDF
(d) flipped TSDF


RGB-D frame
observed surface
ground truth
Zheng *et al*. [37]
Firman *et al*. [3]
Lin *et al*. [18]
Geiger and Wang [4]
SSCNet

A Point Set Generation Network for 3D Object Reconstruction from a Single Image [Fan, Su, Guibas, 2017]


Input
Reconstructed 3D point cloud

Network Architecture



Distance Metrics

- Chamfer distance

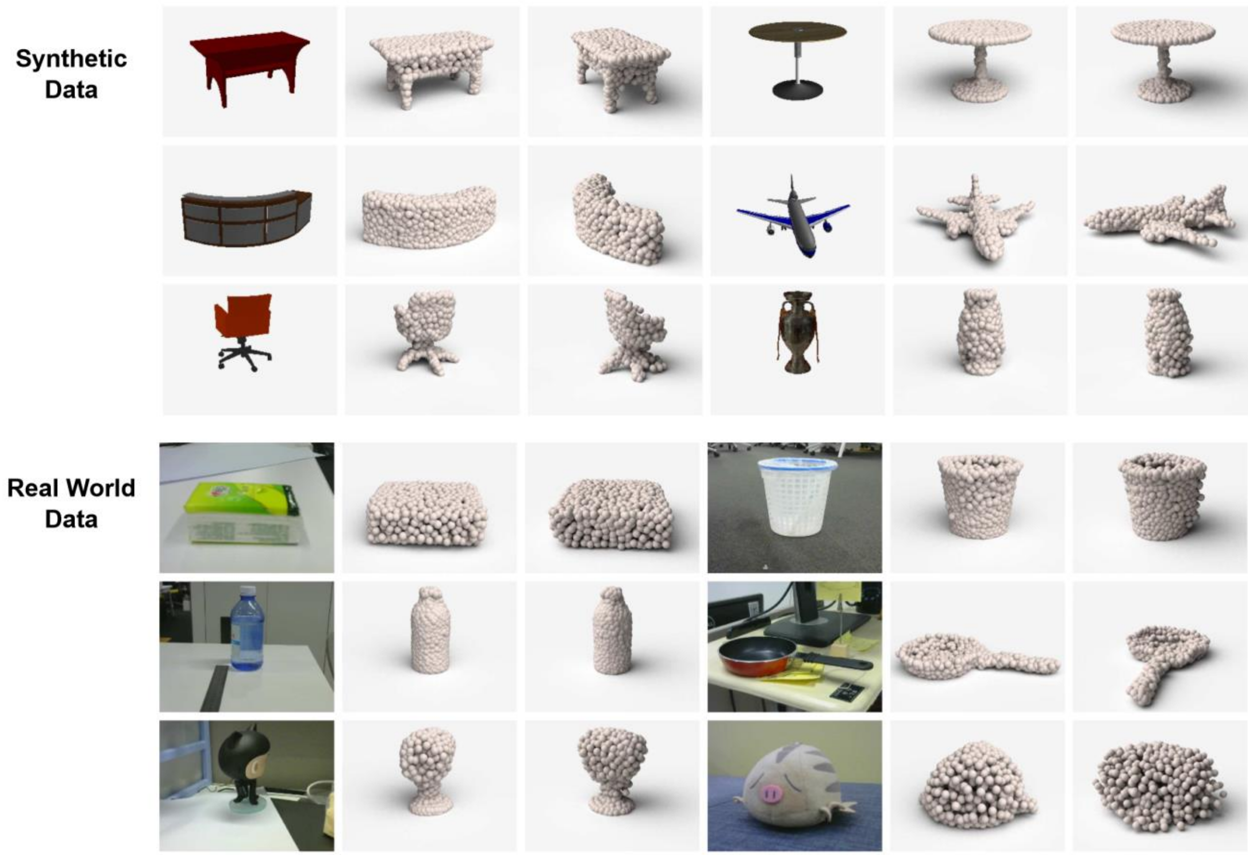
$$d_{CD}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} \|x - y\|_2^2 + \sum_{y \in S_2} \min_{x \in S_1} \|x - y\|_2^2$$

- Earth Mover's distance

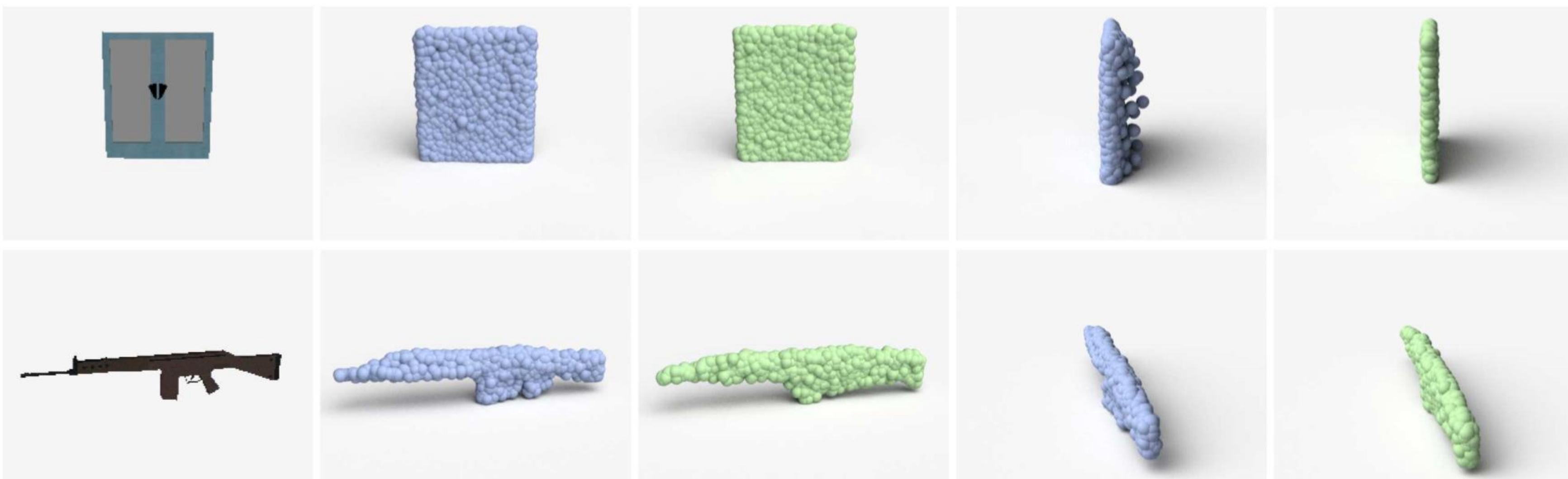
$$d_{EMD}(S_1, S_2) = \min_{\phi: S_1 \to S_2} \sum_{x \in S_1} \|x - \phi(x)\|_2$$

$\phi : S_1 \to S_2$ is a bijection

Visual results



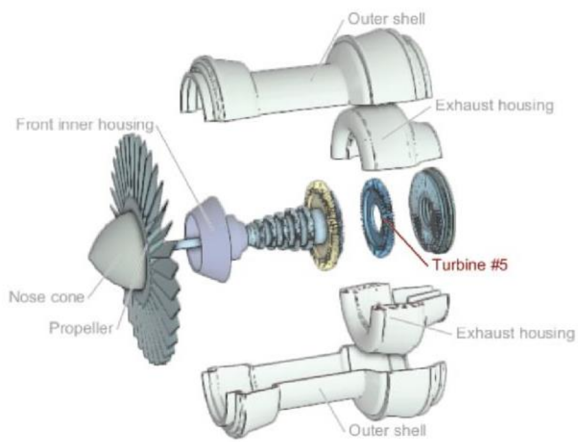
CD (Left) versus EMD (Right)

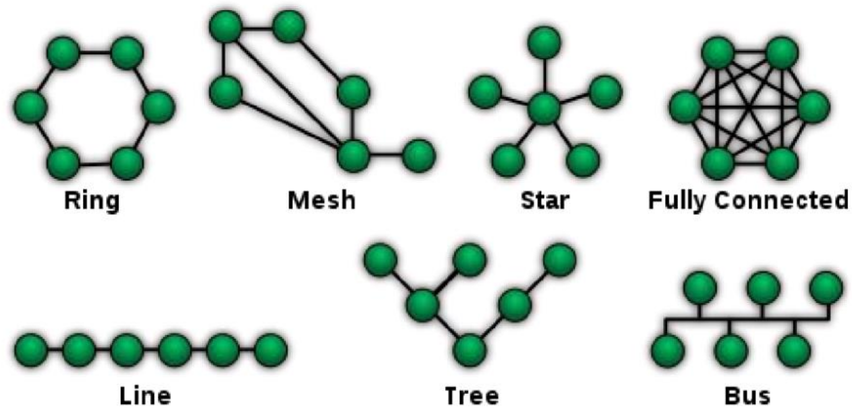


GRASS: Generative Recursive Autoencoders for Shape Structures
[Li, Xu, Chaudhuri, Yumer, Zhang, Guibas, SIGGRAPH' 17]

Huge Variety of (Attributed) Graphs

- Arbitrary numbers/types of vertices (parts), arbitrary numbers of connections (adjacencies/symmetries)

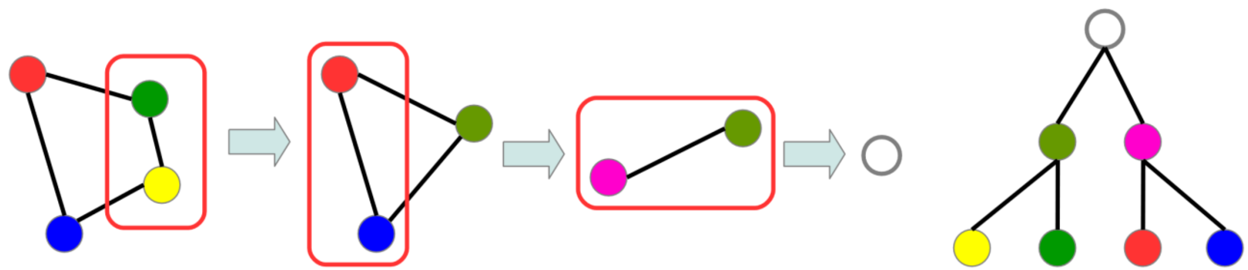




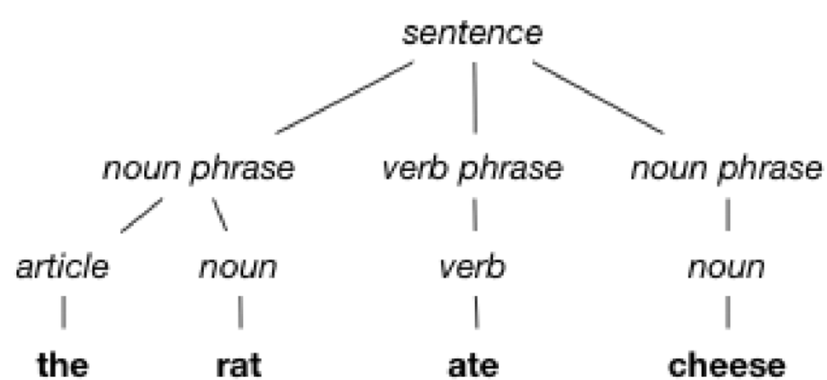
- For linear graphs (chains) of arbitrary length, we can use a recurrent neural network (RNN/LSTM)

Key Insight

- Edges of a graph can be collapsed sequentially to yield a hierarchical structure



- Looks like a parse tree for a sentence!

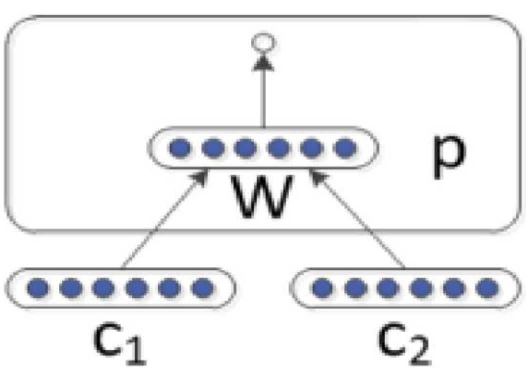


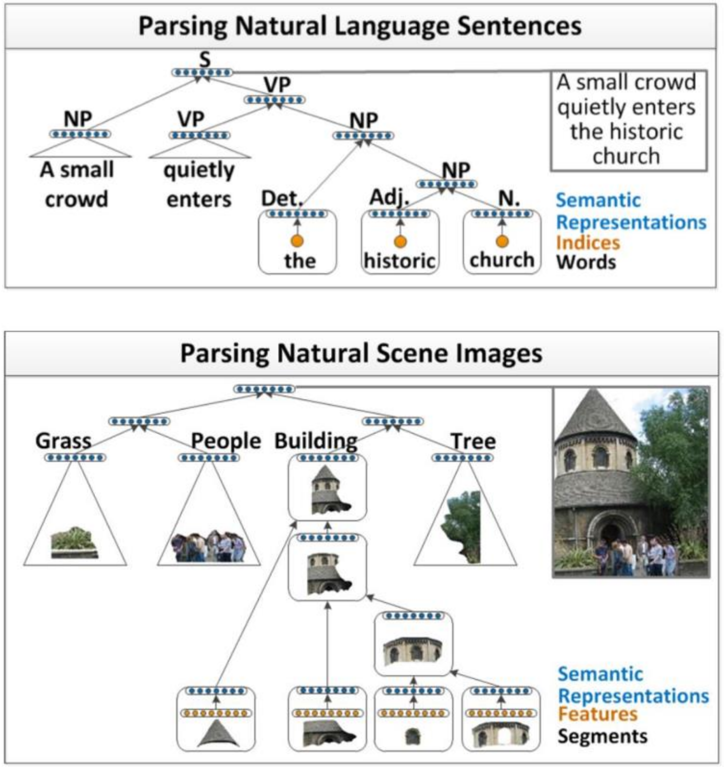
Recursive Neural Network (RvNN)

- Repeatedly merge two nodes into one

- Each node has an n-D feature vector, computed recursively

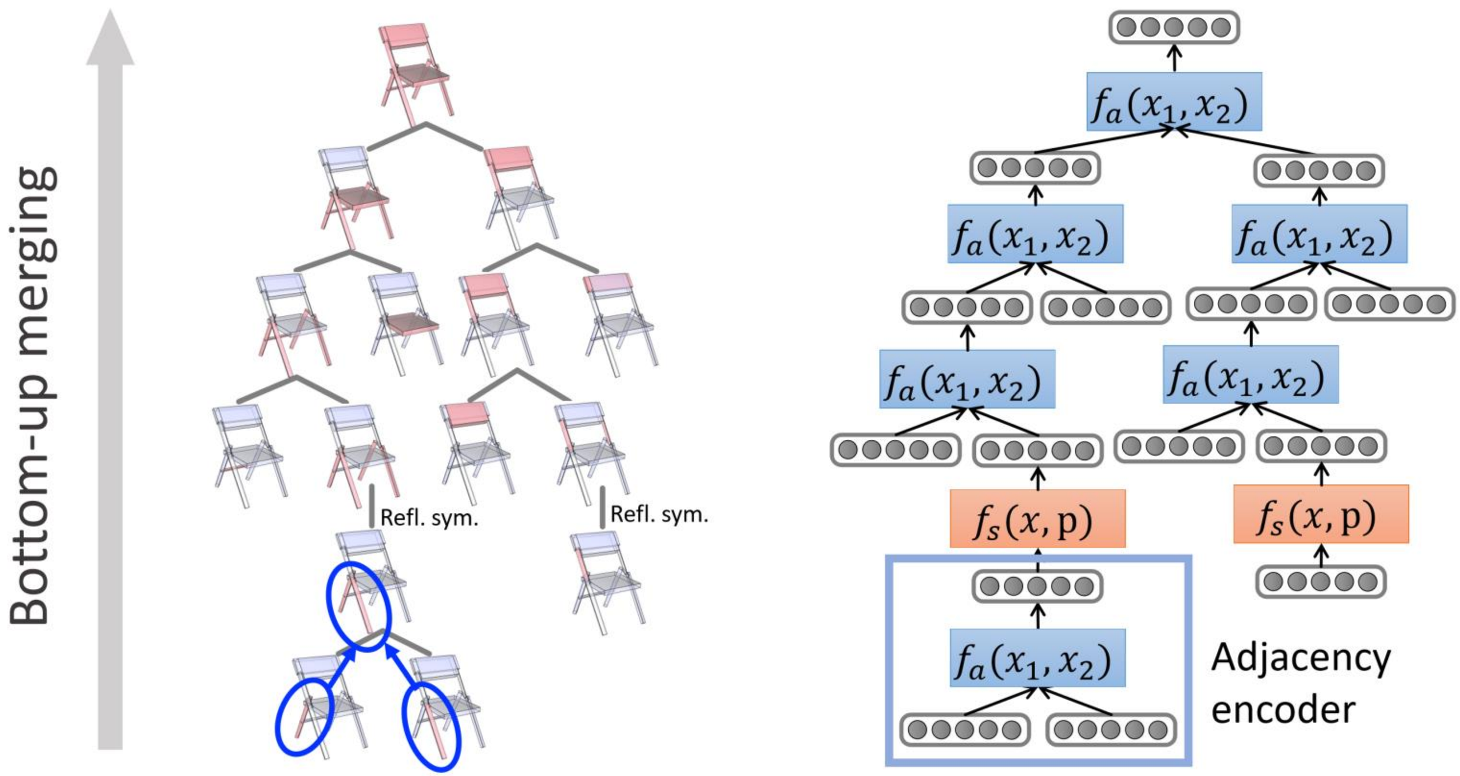
$$p = f(W[c_1;c_2] + b)$$



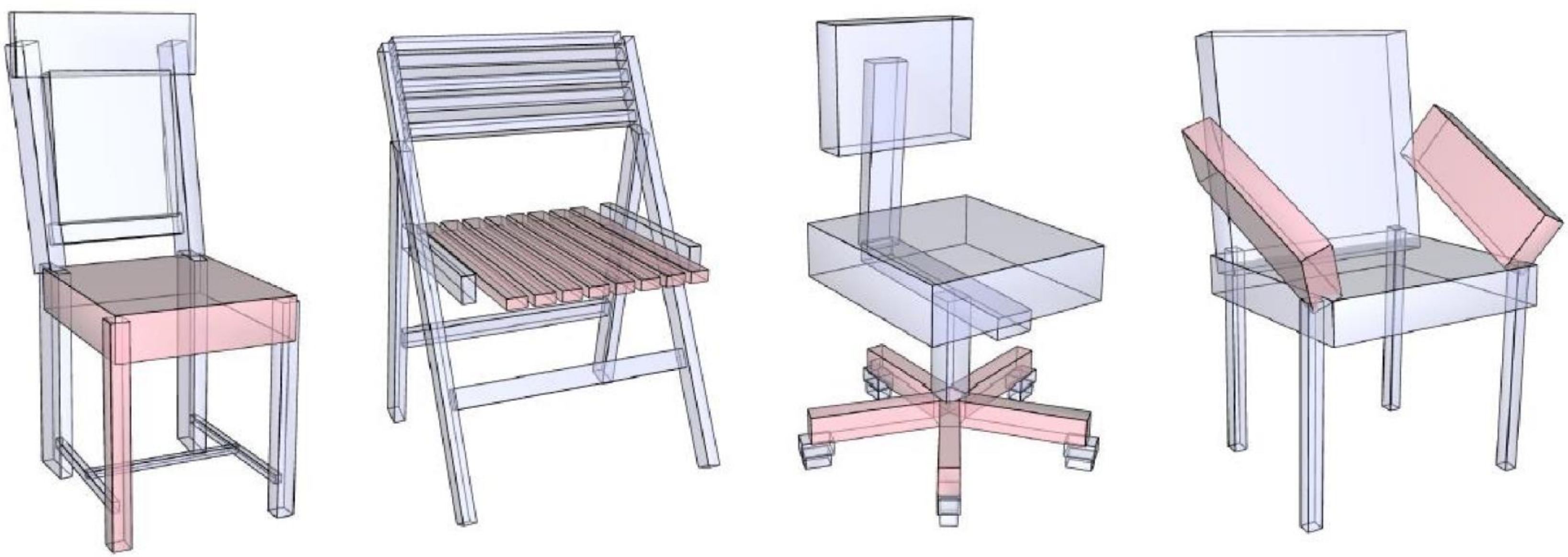


Socher et al. 2011

Recursively Merging Parts



Different types of merges, varying cardinalities!



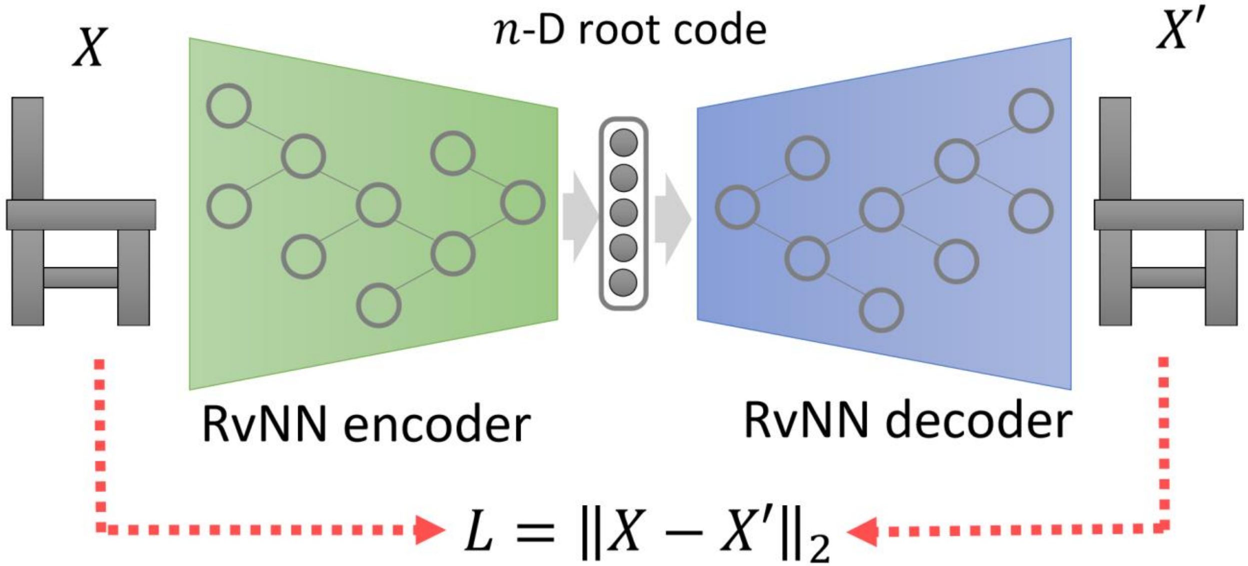
Adjacency

Translational symmetry

Rotational symmetry

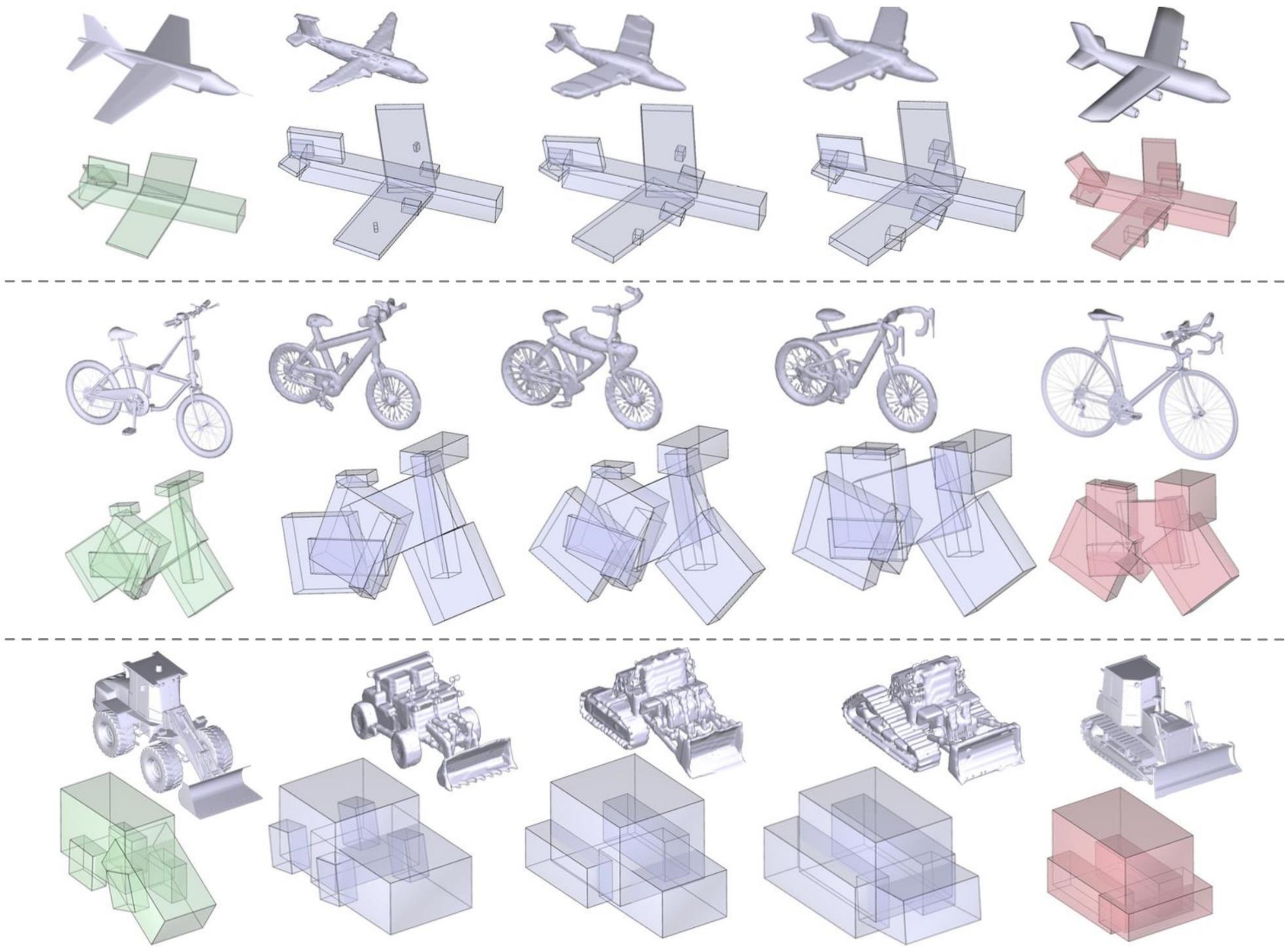
Reflectional symmetry

Training with Reconstruction Loss



- Learn weights from a variety of randomly sampled merge orders for each box structure

Results: Shape interpolation



Discussion