Occupancy Networks

Rodrigo Loro Schuller

28 November 2019

Introduction

- Rise in learning-based 3D reconstruction (2015-2016);
- The problem of 3D representation:
  - State-of-the-art alternatives;
  - Occupancy network;
- Results and comparisons;
- Failure cases and further work;

The problem of 3D representation

- Harder than 2D - no agreed upon standards;
- State-of-the-art alternatives:
  - Voxels;
  - Point clouds;
  - Meshes;
- Occupancy networks;

Voxels

The good, the bad and the ugly

Voxels - The good

- Natural extension to pixels;
- Simple:
  - We have marching cubes for mesh construction;
  - Works well on GPU;

Voxels - Similarity operations (the bad)

Definition (Similarity) A *similarity transformation* is an affine transformation $x \mapsto Ax + b$ in which $A = \alpha Q$ for any orthogonal matrix Q.

- Low resolution voxel-based representations don't behave well under ST!

Voxels - Similarity operations (the bad)

Let T_λ^X be a group of transformations in the space X, with correspondents T_λ^Y in the space Y.

Definition (Equivariance) A function $\phi : X \to Y$ is *equivariant* under the group T if $\forall \lambda \in \Lambda$, $\forall x \in X$, $\phi\left(T_\lambda^X x\right) = T_\lambda^Y \phi(x)$.

Voxels - Similarity operations (the bad)

Let T_λ^X be a group of transformations in the space X, with correspondents T_λ^Y in the space Y.

Definition (Equivariance) A function $\phi : X \to Y$ is *equivariant* under the group T if $\forall \lambda \in \Lambda$, $\forall x \in X$, $\phi\left(T_\lambda^X x\right) = T_\lambda^Y \phi(x)$.

Definition (Invariance) $\forall \lambda \in \Lambda$, $\forall x \in X$, $\phi\left(T_\lambda^X x\right) = \phi(x)$.

Voxels - Similarity operations (the bad)

Let T_λ^X be a group of transformations in the space X, with correspondents T_λ^Y in the space Y.

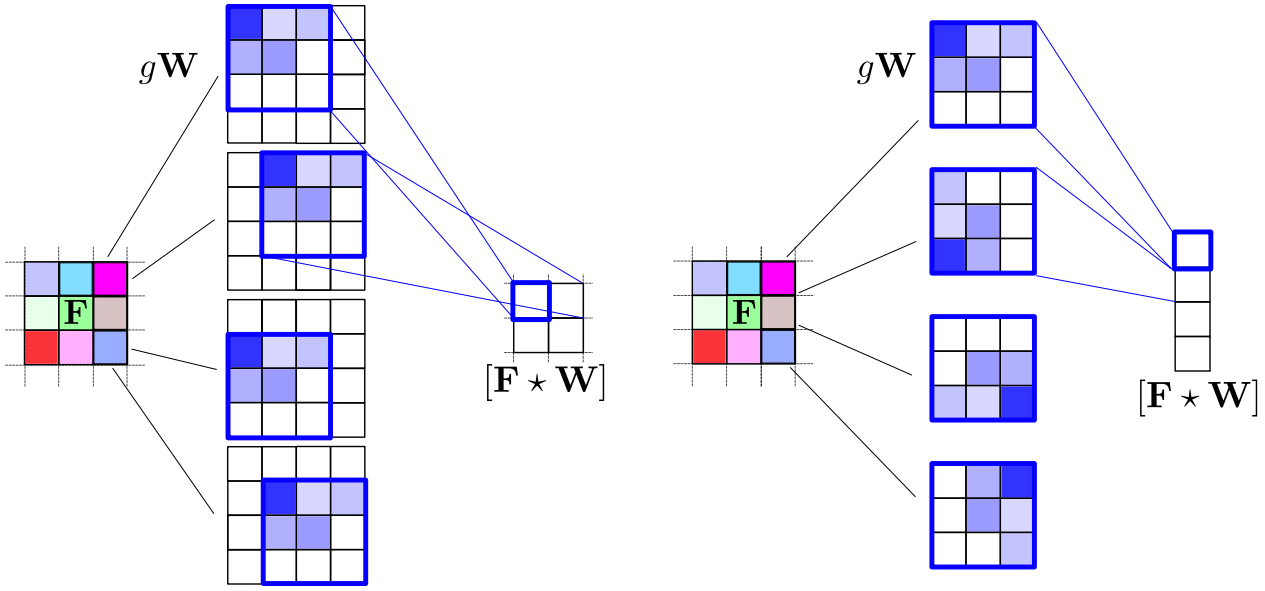
Definition (Equivariance) A function $\phi : X \to Y$ is *equivariant* under the group T if $\forall \lambda \in \Lambda$, $\forall x \in X$, $\phi\left(T_\lambda^X x\right) = T_\lambda^Y \phi(x)$.

Definition (Invariance) $\forall \lambda \in \Lambda$, $\forall x \in X$, $\phi\left(T_\lambda^X x\right) = \phi(x)$.

Observation Standard CNNs are equivariant to discrete translations!

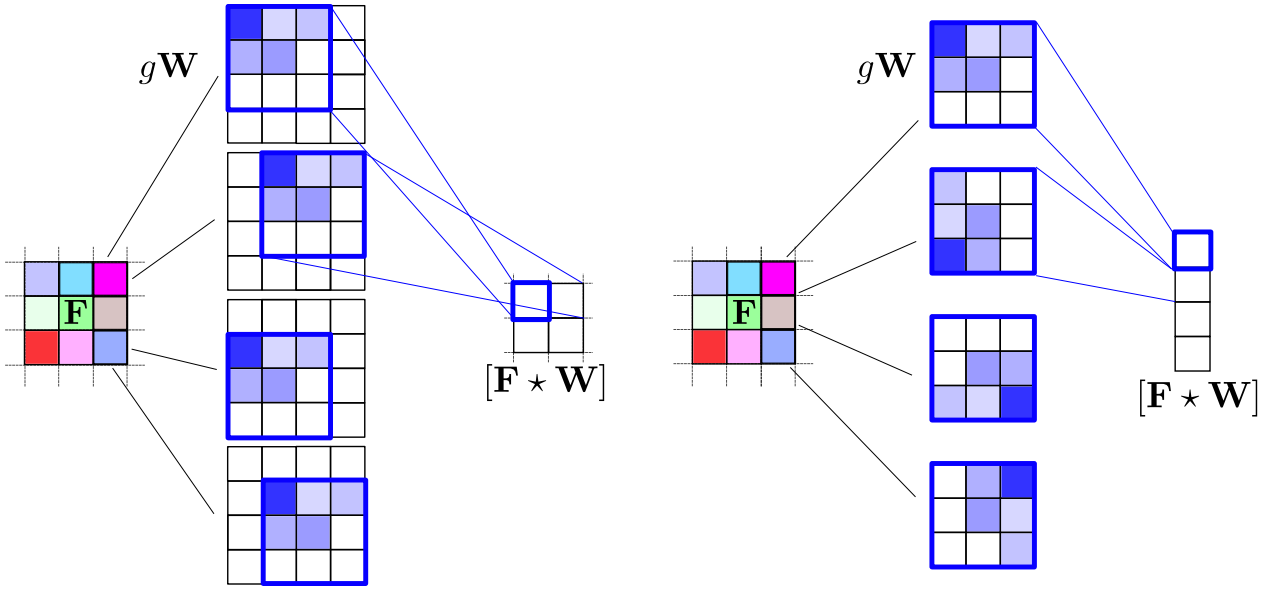
Voxels - Similarity operations (the bad)

We can make 2D CNNs equivariant to $\mathbb{Z}_4$:



Voxels - Similarity operations (the bad)

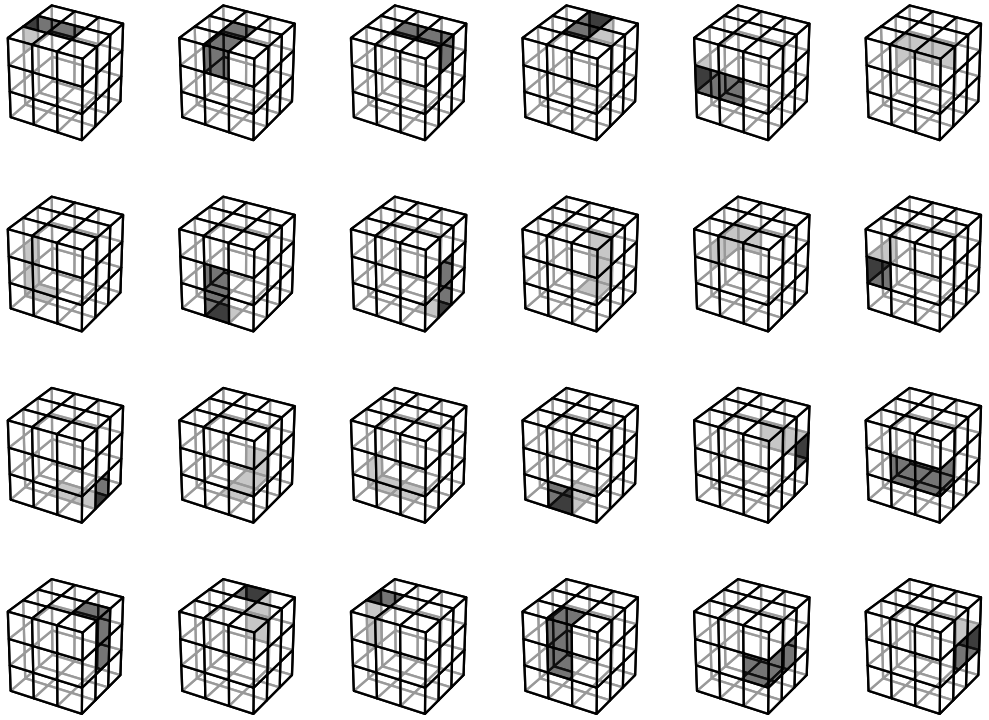
We can make 2D CNNs equivariant to $\mathbb{Z}_4$:



And to other rotations!

Voxels - Similarity operations (the bad)

In 3D we have equivariance under S_4:



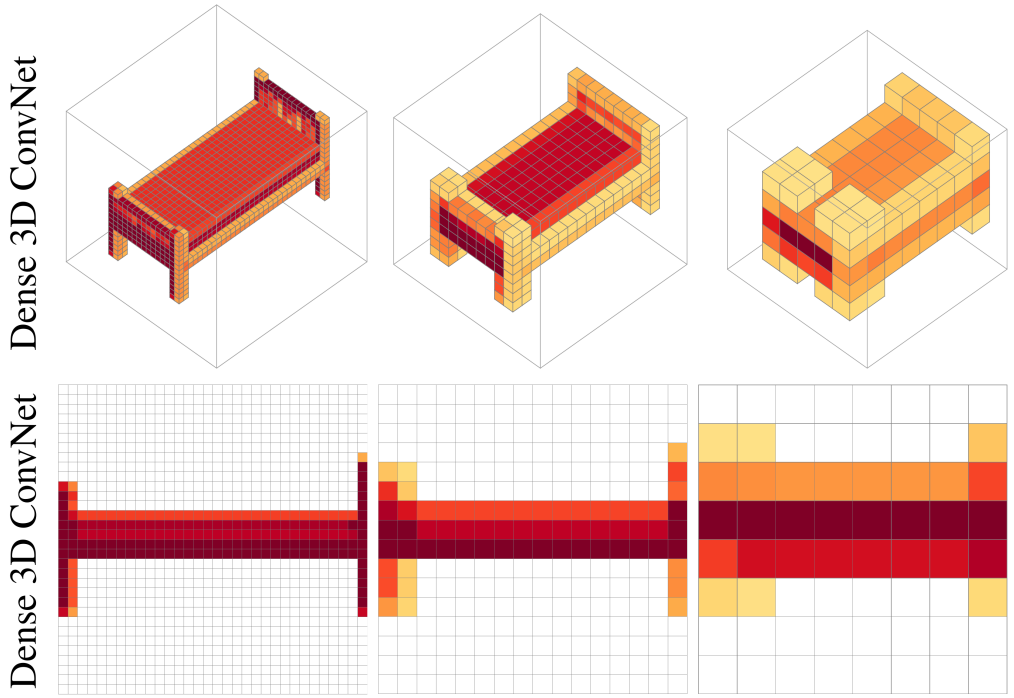
[1] CubeNet: Equiv. to 3D R. and Translation, D. Worrial and G. Brostow (2018)

Voxels - Memory complexity (the ugly)

- The elephant in the room is the $\Theta(n^3)$ complexity!
- But we can make it better;

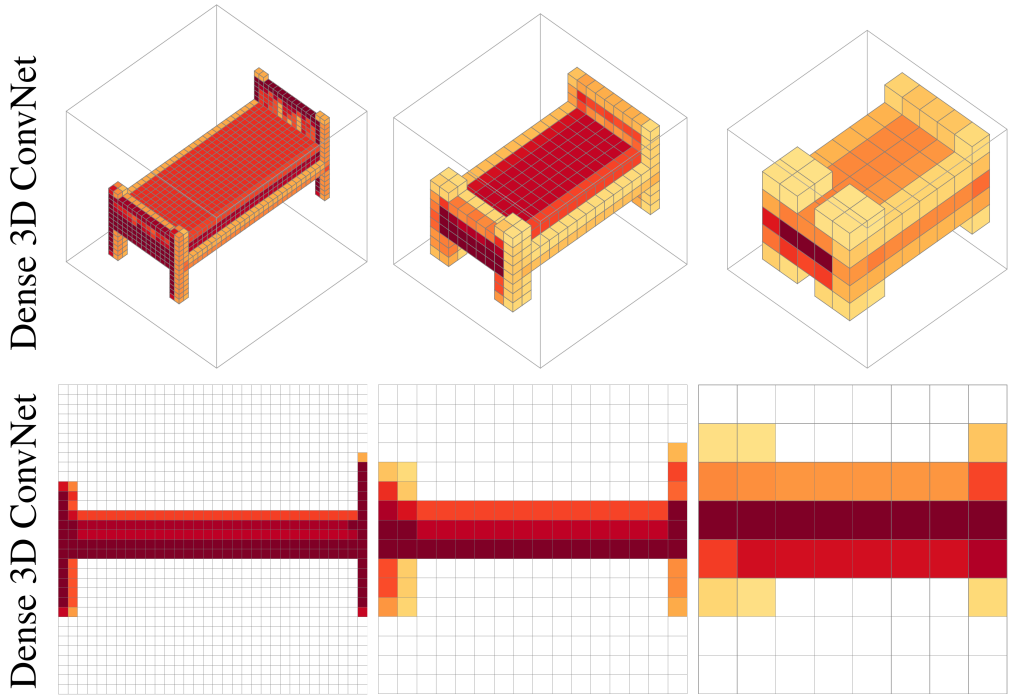
Voxels - Memory complexity (the ugly)

Activation profile for different pooling layers in ConvNet



Voxels - Memory complexity (the ugly)

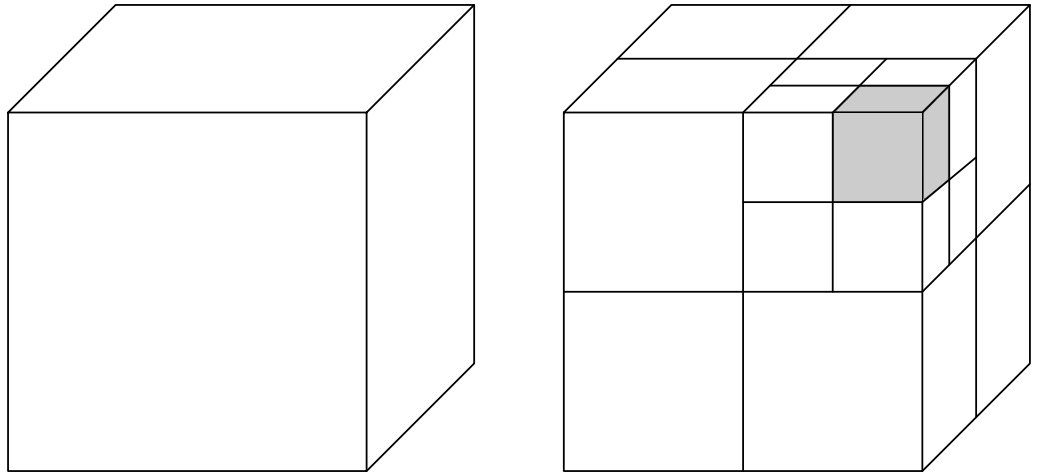
Activation profile for different pooling layers in ConvNet



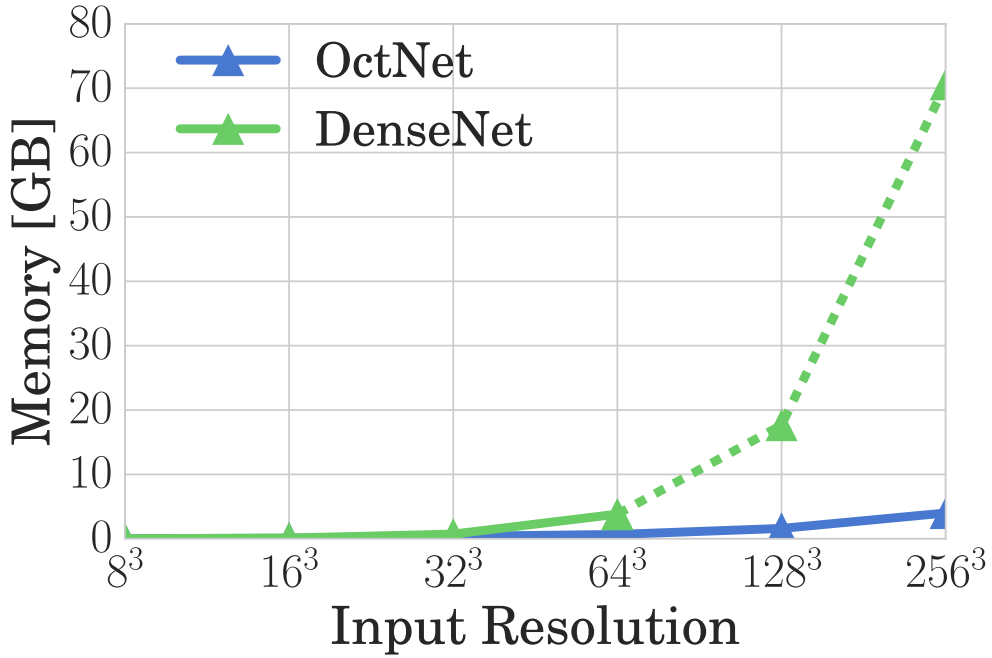
Let's try octrees!

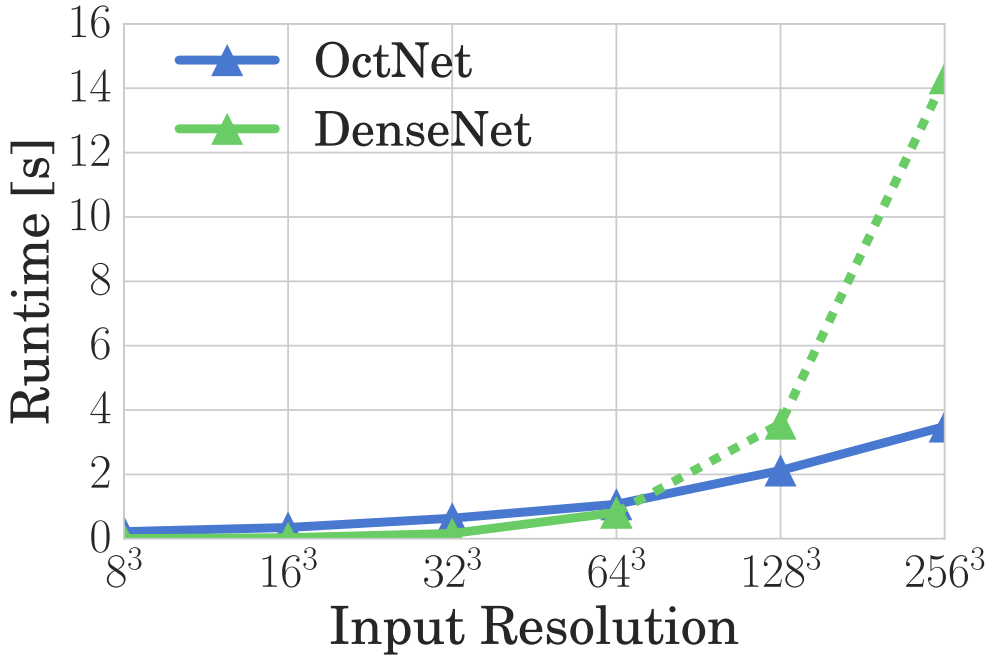
Voxels - Memory complexity (the ugly)

Octree - Adaptative data structure



Voxels - Memory complexity (the ugly)





[2] OctNet Learning 3D Repr. at High Resolutions, G. Riegler (2017)

Voxels - Memory complexity (the ugly)

Most state-of-the art voxel-based RNN use 32^3 or 64^3 voxels.

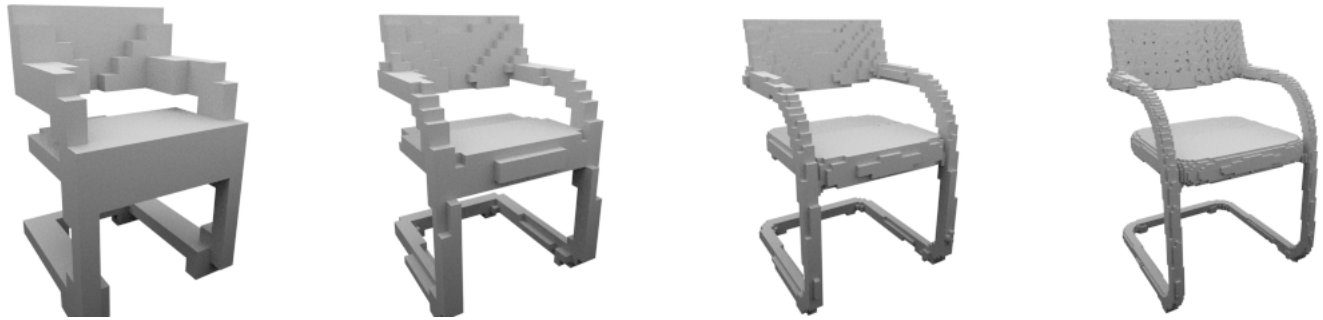


Figure: comparison between resolutions, from 16 to 128.

Voxels - Memory complexity (the ugly)

Most state-of-the art voxel-based RNN use 32^3 or 64^3 voxels.

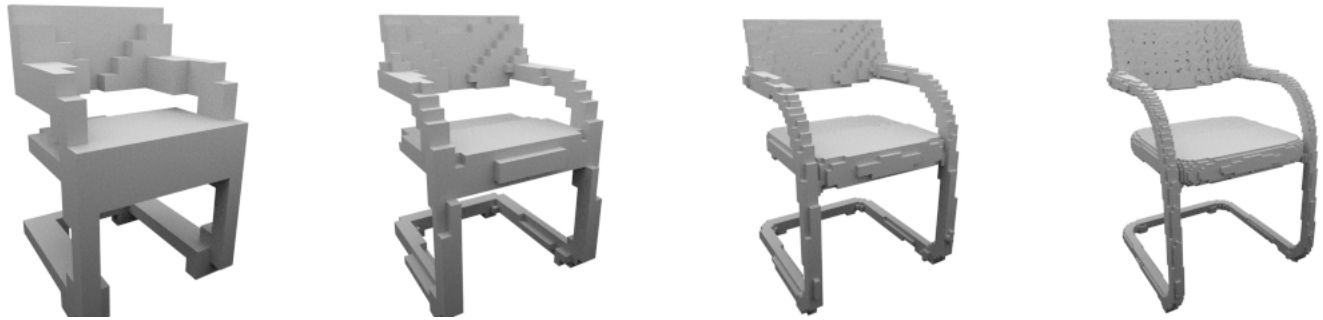


Figure: comparison between resolutions, from 16 to 128.

Observation 3D-R2N2 (Choy, 2016) uses 32^3.

Point clouds as 3D representations

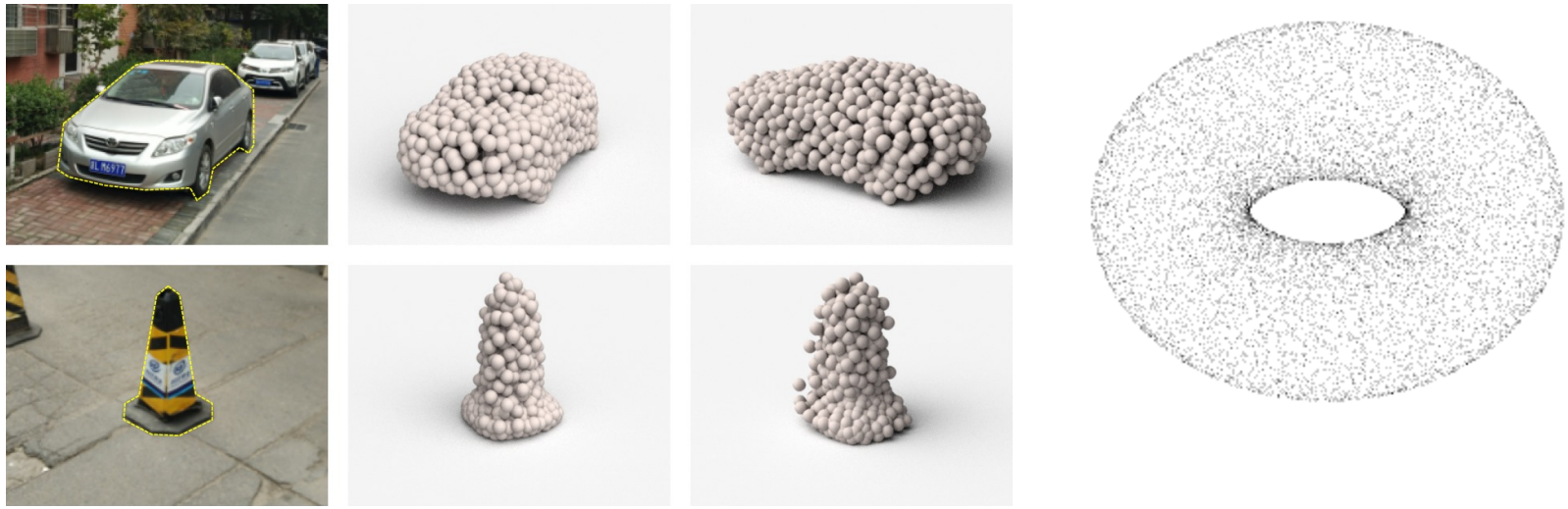
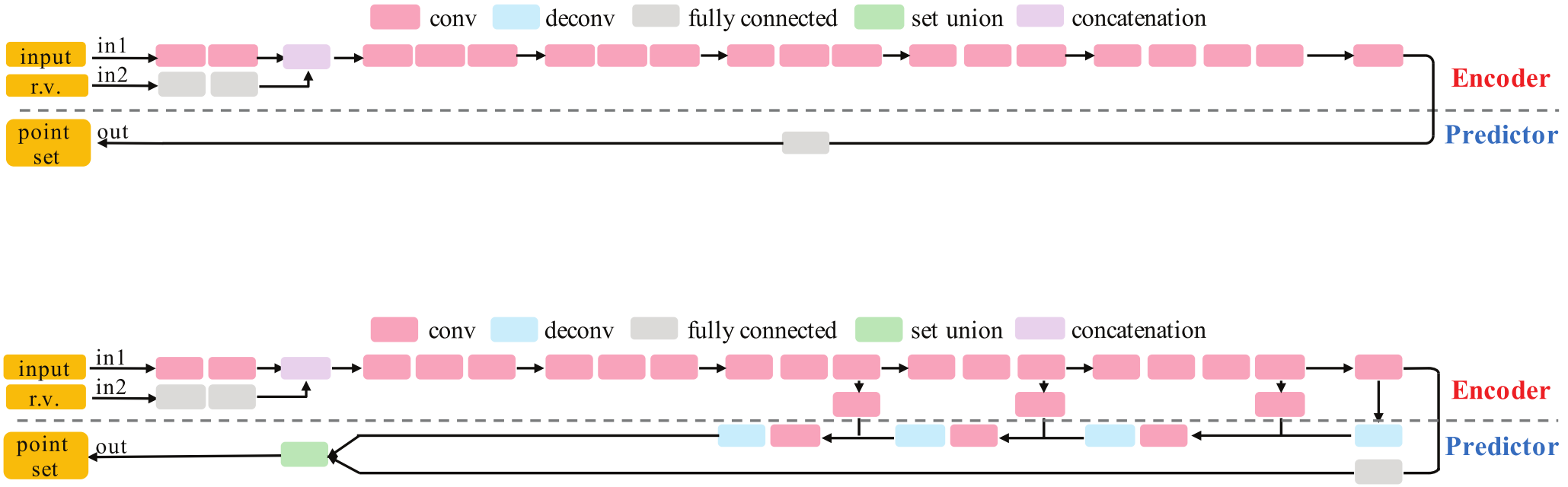


Figure: Reconstructed 3D point clouds (H. Fan, 2016) and a torus.

Point clouds as 3D representations

- Behave well under similarity and other geometric transformations;
- More efficient than voxels - adaptative;
- Produce nice 3D models in NNs:
  - Architecture divided in two parts to account for large and smooth surfaces;
  - Good for segmentation analysis;
- Extracting meshes is complicated;

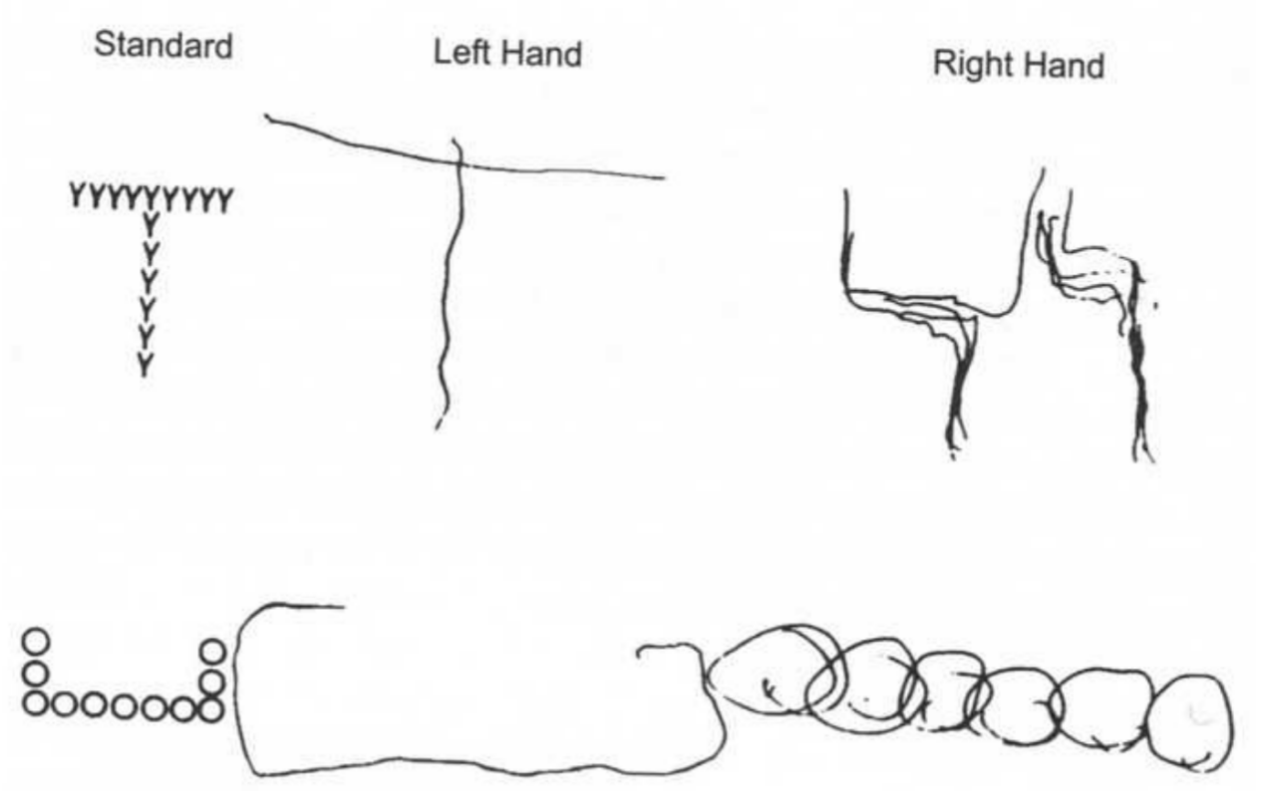
Generation network using point clouds



Observation Local and global specializations. Sounds familiar?

[3] A Point Set G. N. for 3D Object Reconst. from a Single Image, H. Fan (2016)

H. specialization - Split-brain patient



[5] Unknown source, taken from J. Gabrieli's lecture slides

H. specialization - Anatomy of the brain

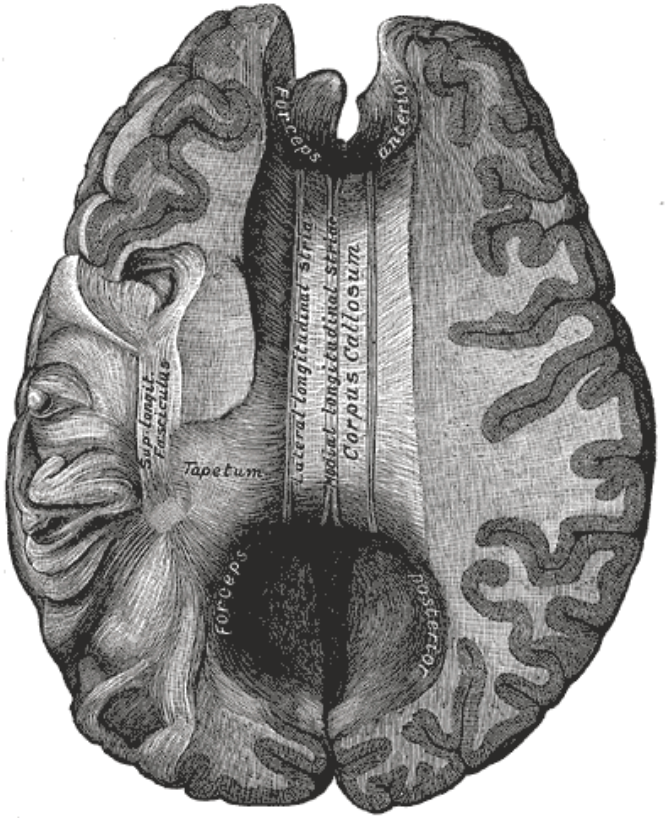
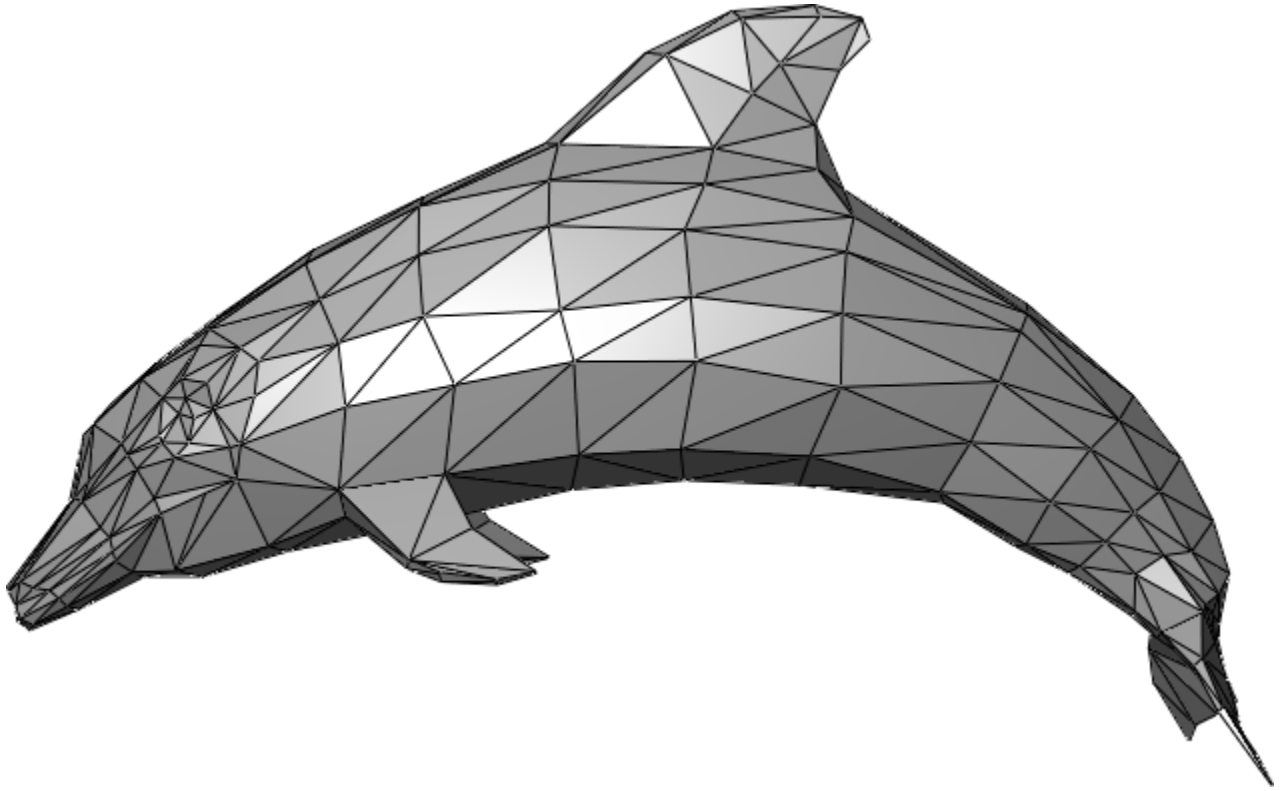


Figure Drawing of *corpus callosum*, connecting the local (LH) and the global (RH).

Meshes for 3D reconstruction

Good in theory



Meshes for 3D reconstruction

Main takeaways

- Behave well with geometric transformations;
- Small memory footprint and fast operations;
- But hard to use in practice:
  - Intersections/overlaps or non-closed geometry;
  - Topology limitations;

Meshes for 3D reconstruction - AtlasNet

[6] AtlasNet: A Papier-Mâché Approach to L. 3D Surface G., T. Groueix (2018)

Meshes for 3D reconstruction - Pixel2Mesh

[7] Pixel2Mesh: Generating 3D Mesh Models from S. RGB Images, N. Wang (2018)

Occupancy Network

Learning 3D Reconstruction in Function Space

Mescheder, Lars and Oechsle, Michael and Niemeyer, Michael and Nowozin, Sebastian and Geiger, Andreas

2019

Occupancy Network

Ideally, we would like to know the *occupancy function*

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

Occupancy Network

Ideally, we would like to know the *occupancy function*

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

Key idea Approximate with a continuous function!

Occupancy Network

Definition For a given input $x \in X$, we want a binary classification neural network: $f^x : \mathbb{R}^3 \to [0, 1]$.

Occupancy Network

Definition For a given input $x \in X$, we want a binary classification neural network: $f^x : \mathbb{R}^3 \to [0, 1]$. But we can just add x to the inputs, ie,

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

We call f_θ the *Occupancy Network*.

Occupancy Network

Definition For a given input $x \in X$, we want a binary classification neural network: $f^x : \mathbb{R}^3 \to [0, 1]$. But we can just add x to the inputs, ie,

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

We call f_θ the *Occupancy Network*.

Observation The approximated 3D surface, for a particular x_0, is given by S = $\{p \in \mathbb{R}^3 \mid f_\theta(p, x_0) = \tau\}$;

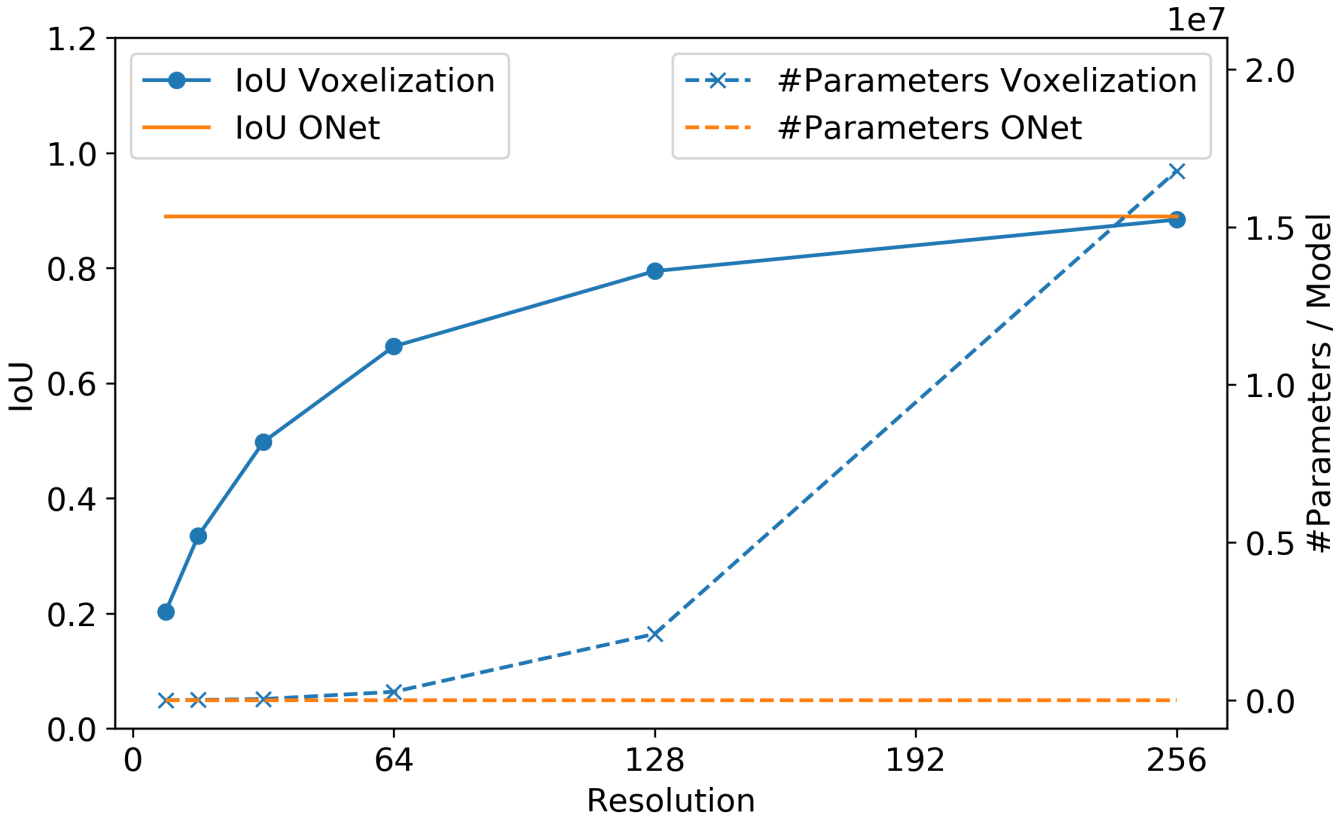
Occupancy Network

Representation capabilities

Figure: 32 to 128 voxels vs *Occupancy Network*.

Occupancy Network

Representation capabilities



Occupancy Network - Training

1. Randomly sample points in the 3D bounding volume of the object
   - with padding;

Occupancy Network - Training

1. Randomly sample points in the 3D bounding volume of the object - with padding;
2. Evaluate the mini-batch loss

$$\mathcal{L}_{\mathcal{B}}(\theta) = \frac{1}{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} \sum_{j=1}^{K} \mathcal{L}(f_\theta(p_{ij}, x_i), o_{ij})$$

in which $\mathcal{L}$ is a cross-entropy classification loss.

Occupancy Network - Training

1. Randomly sample points in the 3D bounding volume of the object - with padding;
2. Evaluate the mini-batch loss

$$\mathcal{L}_{\mathcal{B}}(\theta) = \frac{1}{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} \sum_{j=1}^{K} \mathcal{L}(f_\theta(p_{ij}, x_i), o_{ij})$$

in which $\mathcal{L}$ is a cross-entropy classification loss.

Observation Different sampling schemes were tested, random in the BB w/ padding worked best.

Occupancy Network - Architecture

- Fully connected neural network with 5 ResNet blocks using conditional batch normalization;
- Different encoders depending on the input:
  - SVI - ResNet18;
  - Point clouds - PointNet;
  - Voxelized inputs - 3D CNN;
  - Unconditional mesh generation - PointNet;

Occupancy Network - MISE

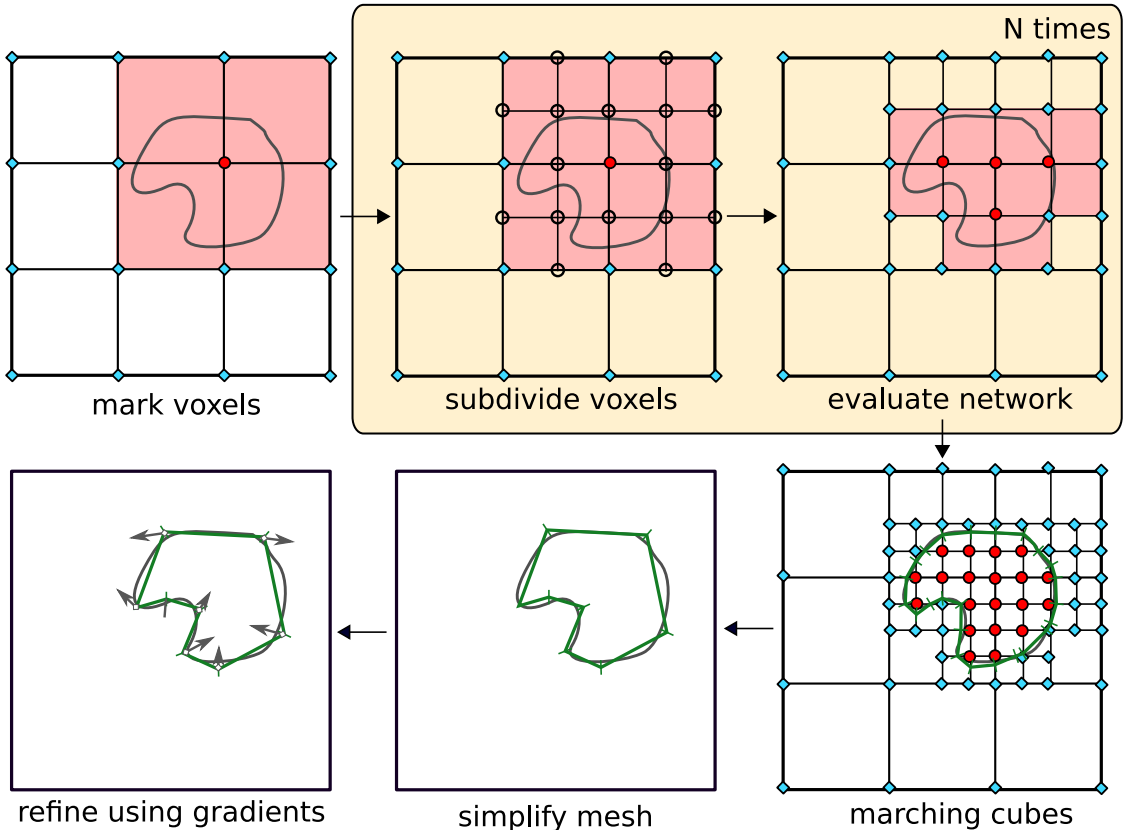
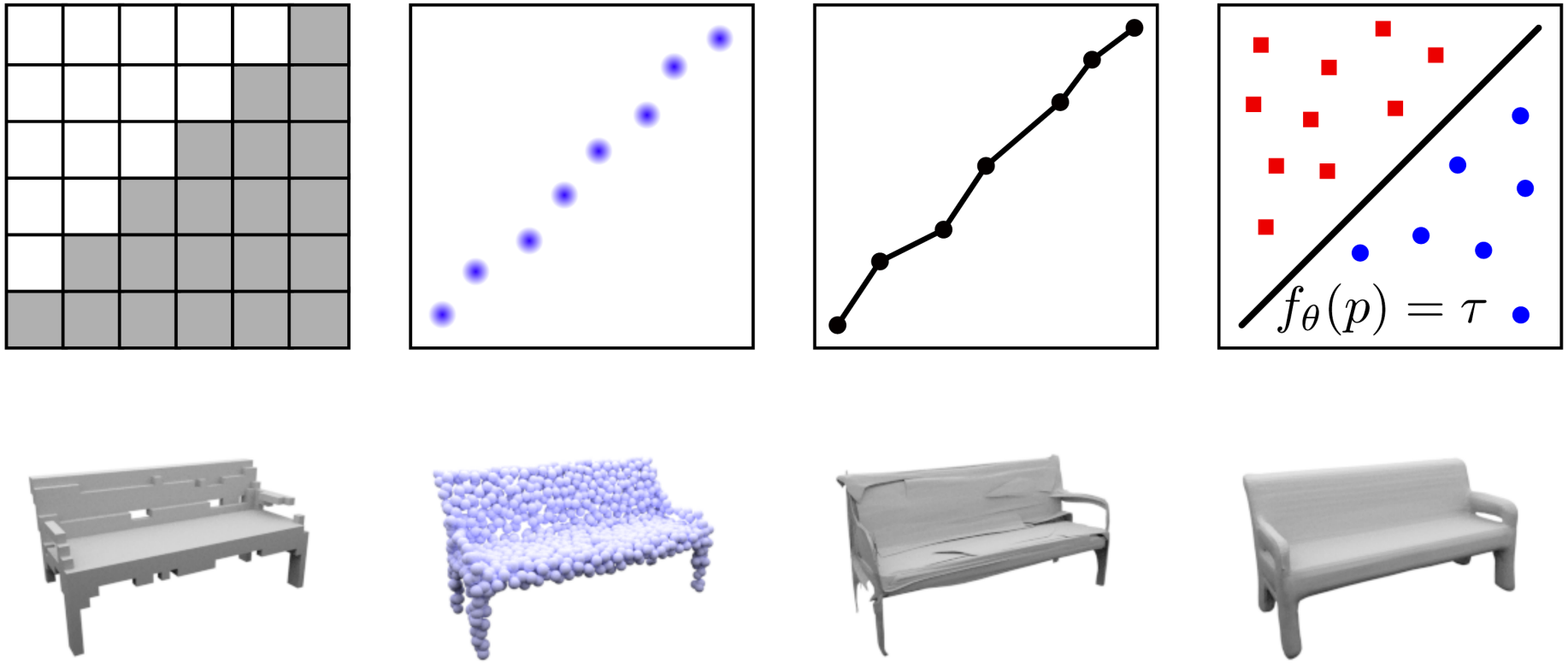
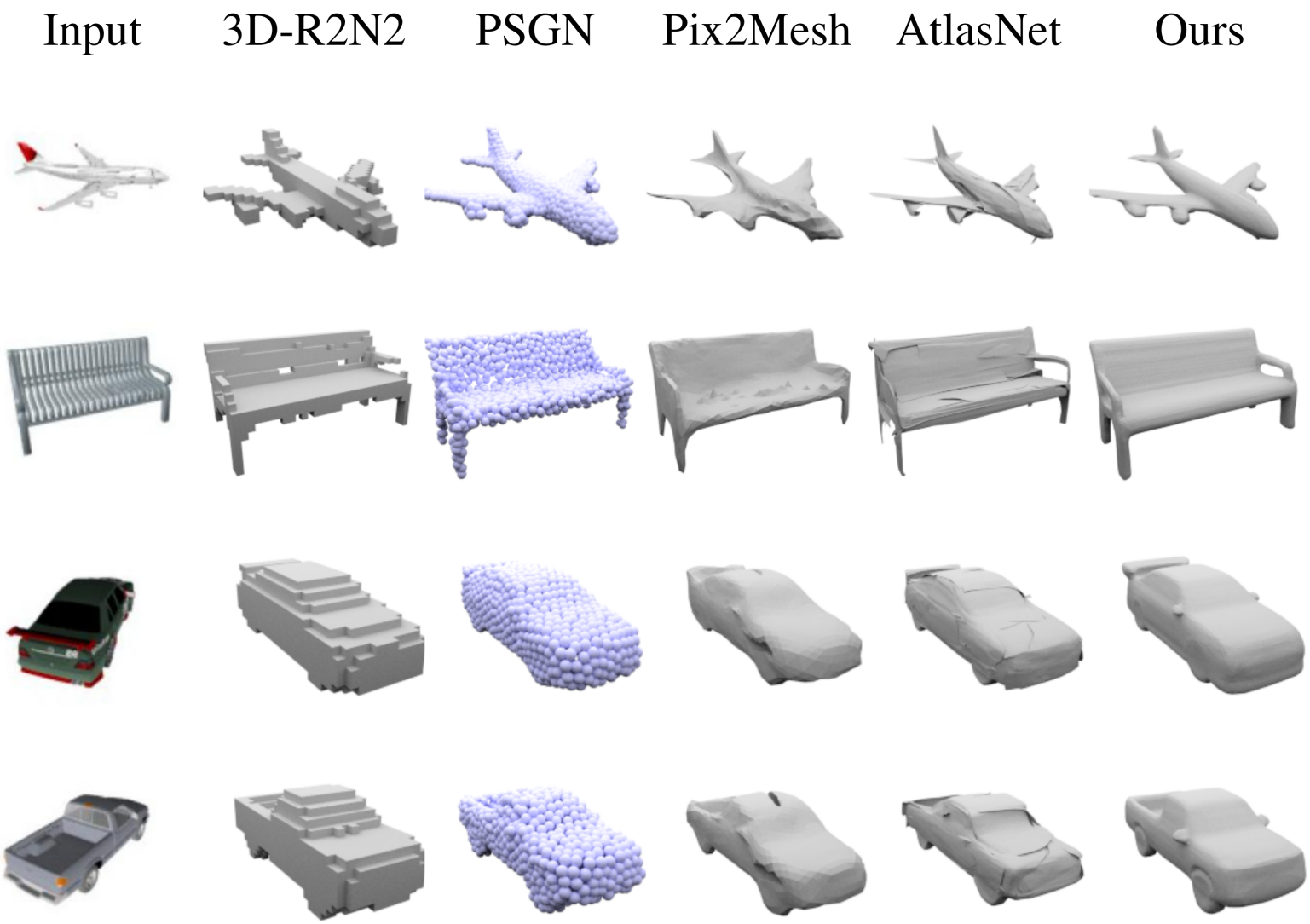


Figure Multiresolution IsoSurface Extraction Method.

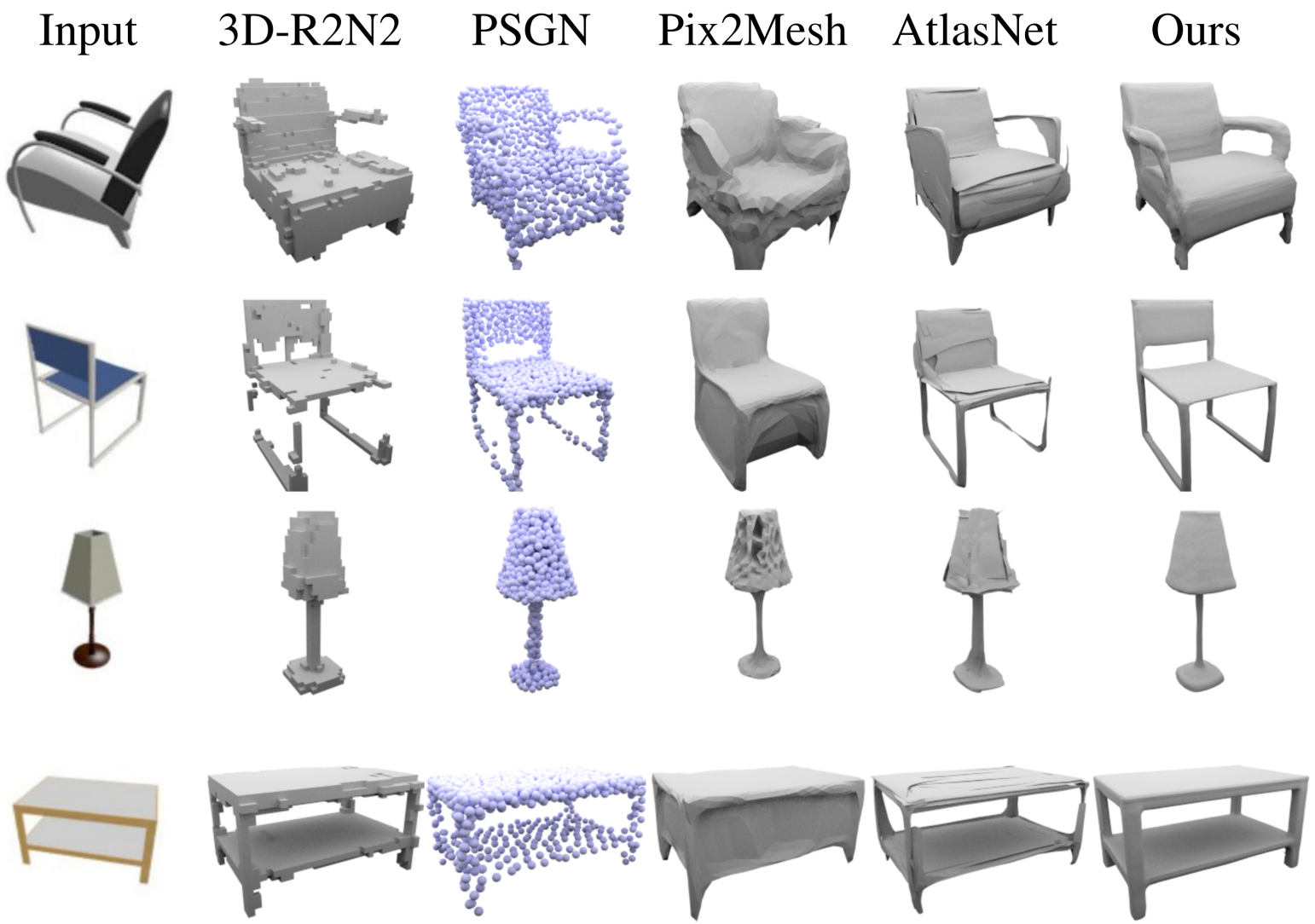
Results and comparisons



SI 3D reconstruction (ShapeNet)



SI 3D reconstruction (ShapeNet)



SI 3D reconstruction (ShapeNet)

	IoU					Chamfer-L_1					Normal Consistency				
category	3D-R2N2	PSGN	Pix2Mesh	AtlasNet	ONet	3D-R2N2	PSGN	Pix2Mesh	AtlasNet	ONet	3D-R2N2	PSGN	Pix2Mesh	AtlasNet	ONet
airplane	0.426	-	0.420	-	**0.571**	0.227	0.137	0.187	**0.104**	0.147	0.629	-	0.759	0.836	**0.840**
bench	0.373	-	0.323	-	**0.485**	0.194	0.181	0.201	**0.138**	0.155	0.678	-	0.732	0.779	**0.813**
cabinet	0.667	-	0.664	-	**0.733**	0.217	0.215	0.196	0.175	**0.167**	0.782	-	0.834	0.850	**0.879**
car	0.661	-	0.552	-	**0.737**	0.213	0.169	0.180	**0.141**	0.159	0.714	-	0.756	0.836	**0.852**
chair	0.439	-	0.396	-	**0.501**	0.270	0.247	0.265	**0.209**	0.228	0.663	-	0.746	0.791	**0.823**
display	0.440	-	**0.490**	-	0.471	0.314	0.284	0.239	**0.198**	0.278	0.720	-	0.830	**0.858**	0.854
lamp	0.281	-	0.323	-	**0.371**	0.778	0.314	0.308	**0.305**	0.479	0.560	-	0.666	0.694	**0.731**
loudspeaker	0.611	-	0.599	-	**0.647**	0.318	0.316	0.285	**0.245**	0.300	0.711	-	0.782	0.825	**0.832**
rifle	0.375	-	0.402	-	**0.474**	0.183	0.134	0.164	**0.115**	0.141	0.670	-	0.718	0.725	**0.766**
sofa	0.626	-	0.613	-	**0.680**	0.229	0.224	0.212	**0.177**	0.194	0.731	-	0.820	0.840	**0.863**
table	0.420	-	0.395	-	**0.506**	0.239	0.222	0.218	0.190	**0.189**	0.732	-	0.784	0.832	**0.858**
telephone	0.611	-	0.661	-	**0.720**	0.195	0.161	0.149	**0.128**	0.140	0.817	-	0.907	0.923	**0.935**
vessel	0.482	-	0.397	-	**0.530**	0.238	0.188	0.212	**0.151**	0.218	0.629	-	0.699	0.756	**0.794**
mean	0.493	-	0.480	-	**0.571**	0.278	0.215	0.216	**0.175**	0.215	0.695	-	0.772	0.811	**0.834**

SI 3D reconstruction (real world data)

(a) KITTI

(b) Online Products

Reconstruction from point clouds

	IoU	Chamfer-L_1 †	Normal Consistency
3D-R2N2	0.565	0.169	0.719
PSGN	-	0.144	-
DMC	0.674	0.117	0.848
ONet	**0.778**	**0.079**	**0.895**

Voxel super resolution

	IoU	Chamfer-L_1	Normal Consistency
Input	0.631	0.136	0.810
ONet	**0.703**	**0.109**	**0.879**

Figure Input resolution of 32^3 voxels.

Unconditional 3D samples

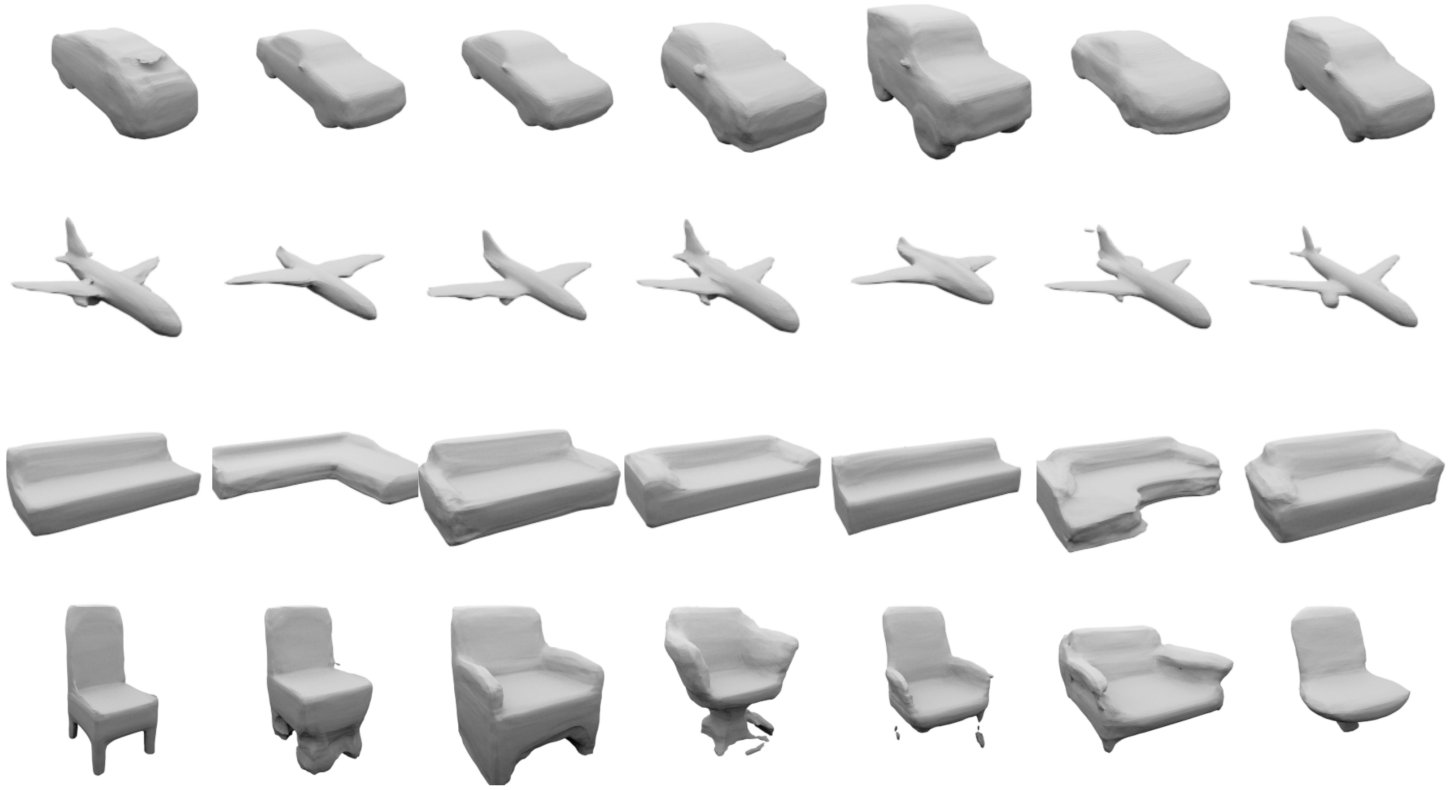
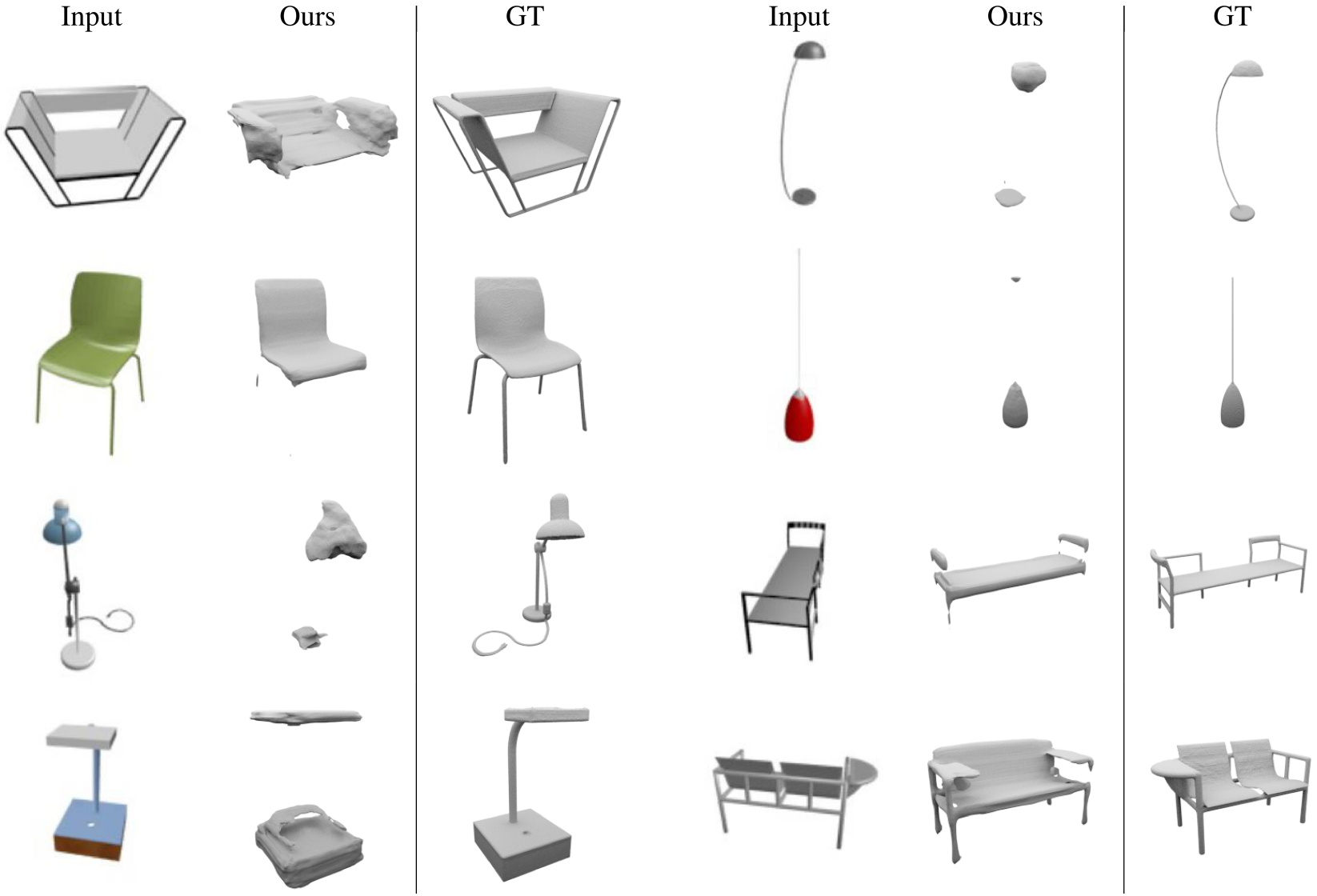


Figure Random samples of unsupervised models trained on different categories.

Failure cases and further work



Other references with similar ideas

[8] DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, Park et al. (2019)
[9] Learning Implicit Fields for Generative Shape Modeling, Chen et al. (2019)
[10] Deep Level Sets: Implicit Surface Representations for 3D Shape Inference, Michalkiewicz et al. (2019)

Thank you!