IV: Point Clouds

3D CV

Kirill Struminsky

In the Previous Episode

- Depth Estimation
  - Stereo Depth Estimation
  - Monocular Depth Estimation
  - 2.5D Data

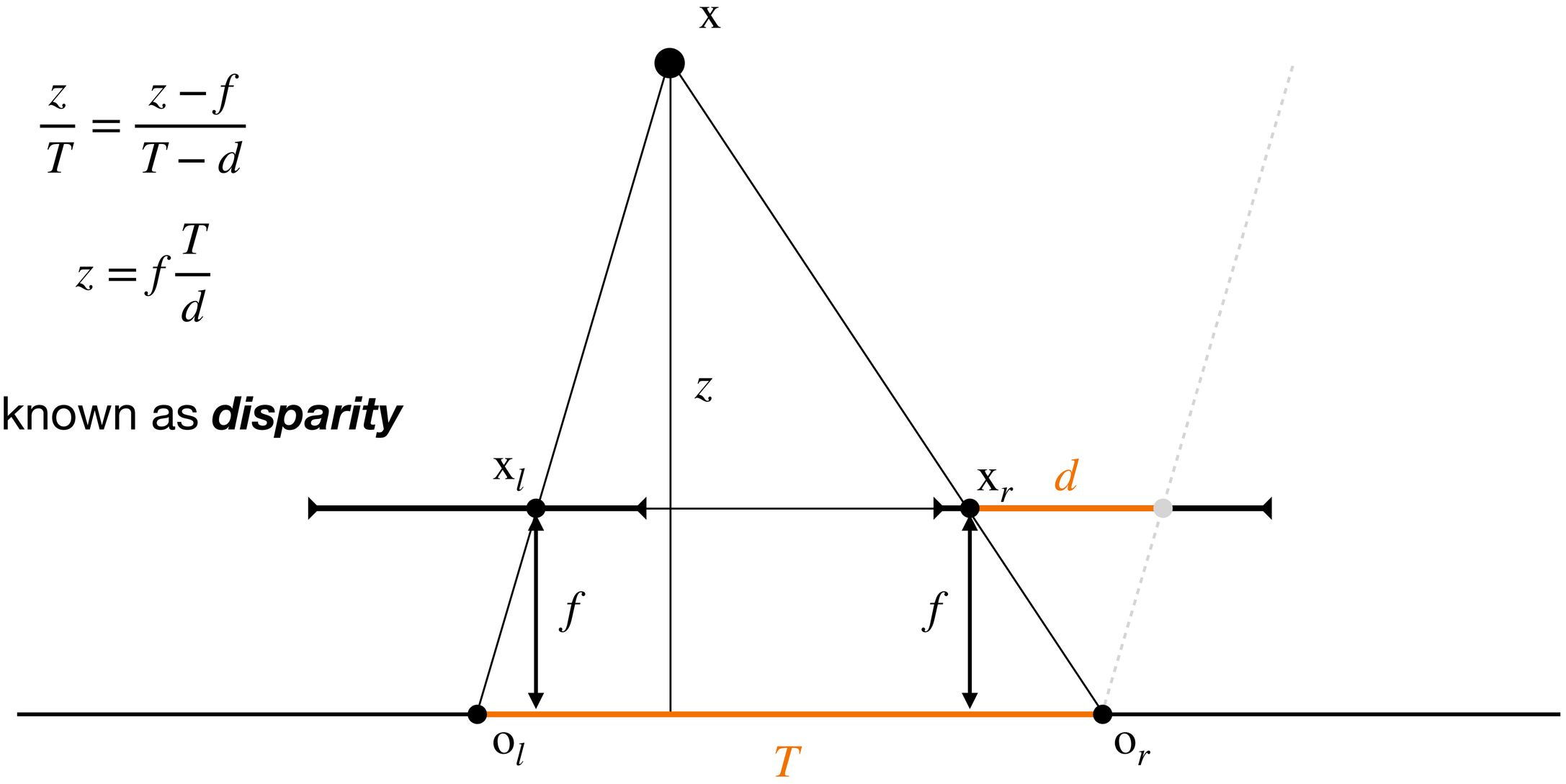
Binocular Vision

Depth Estimation in Two-View Geometry

$$\frac{z}{T} = \frac{z-f}{T-d}$$

$$z = f\frac{T}{d}$$

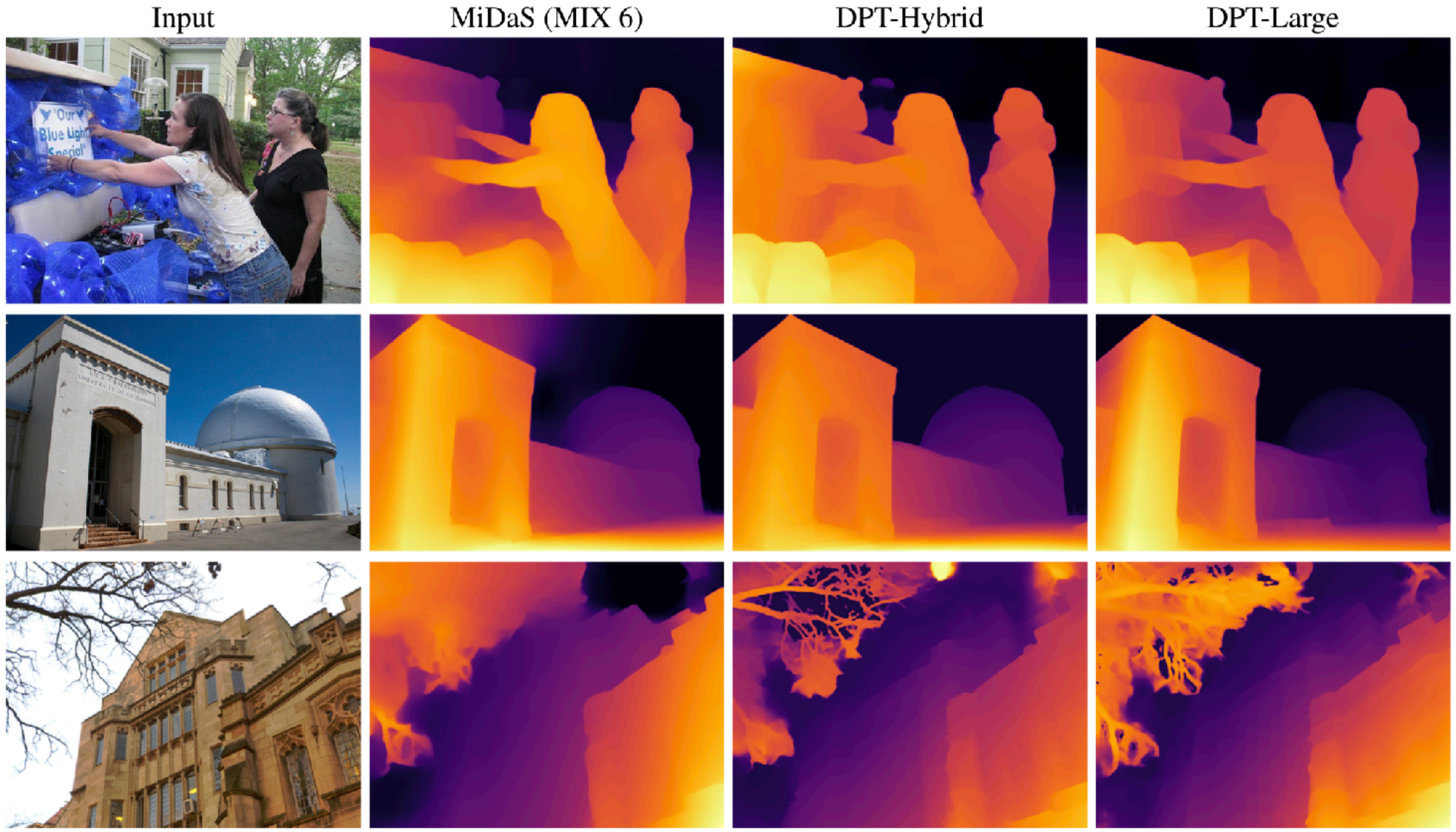
d is known as ***disparity***



Depth Map Prediction

from a Single Image using a Multi-Scale Deep Network

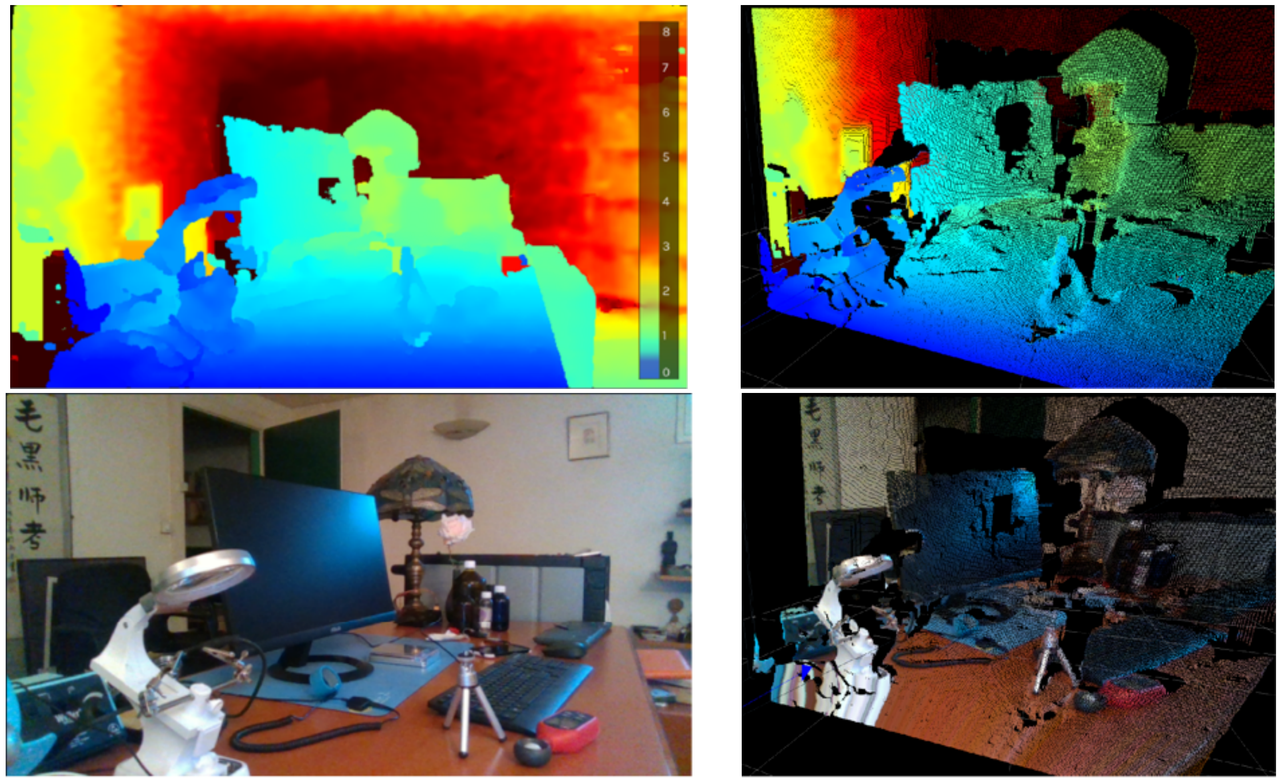
Recent Progress in Monocular Depth Estimation



From 2D to 2.5D

- Recover 3D points using camera intrinsics K and depth map

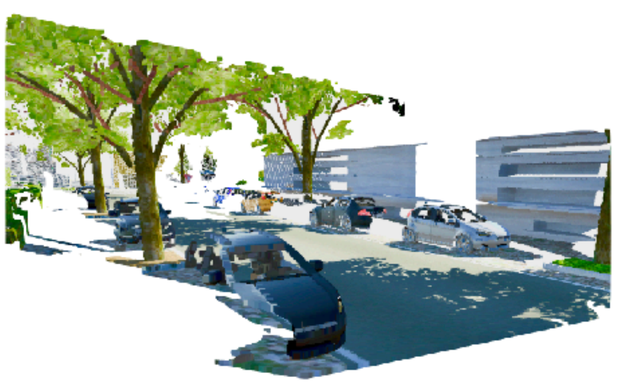
$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \text{depth} \cdot K^{-1} \begin{pmatrix} x_{pix} \\ y_{pix} \\ 1 \end{pmatrix}$$

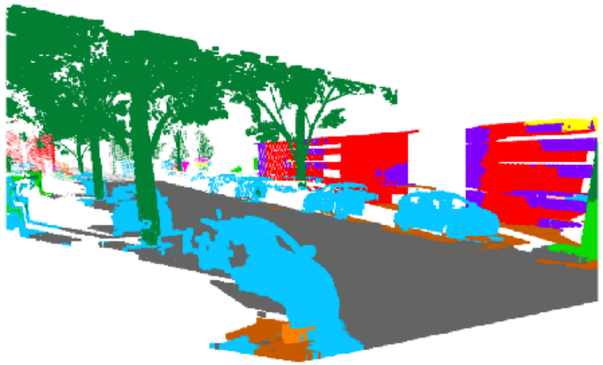


https://www.calvert.ch/maurice/2018/06/12/intel-realsense-d435-review/

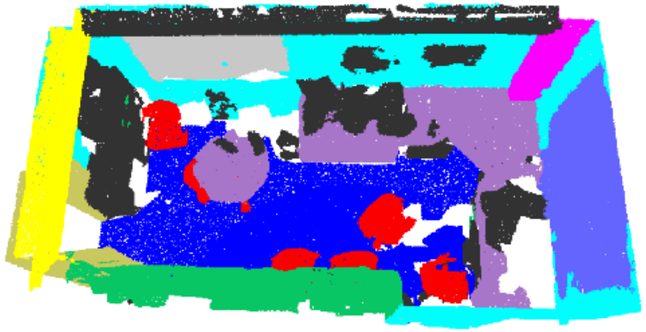
Today's Lecture: Point Cloud Processing

Outdoor Scene





Indoor Scene



Input: 3D Point Cloud

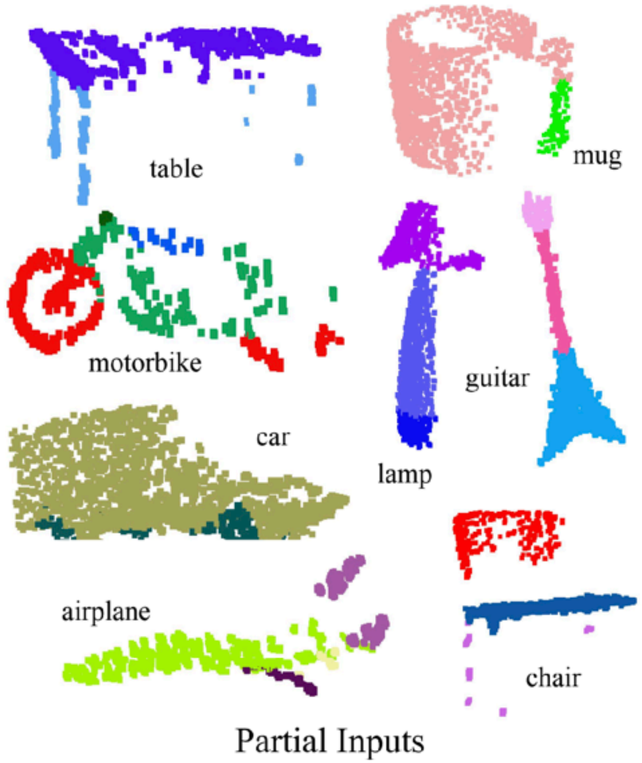
Output: Semantic Segmentation

Engelmann F. et al. Know what your neighbors do: 3D semantic segmentation of point clouds // ECCV Workshops

Point Cloud Architectures

Problem Setup

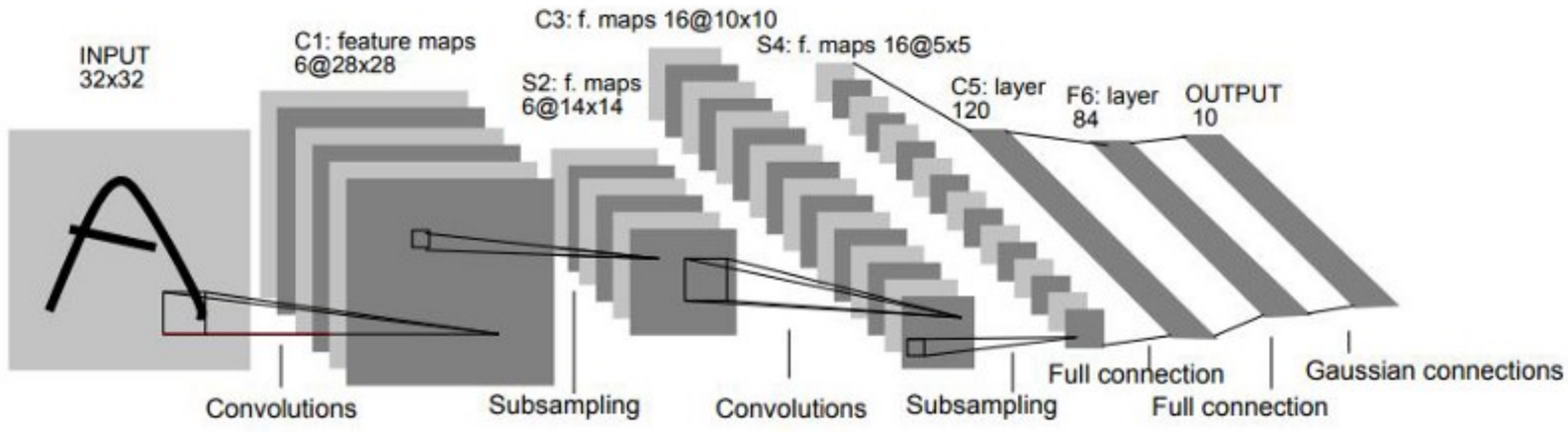
- Point cloud
  - Input $x \in \mathbb{R}^{n \times 3}$
  - Number of points n is not fixed
  - May include additional features (e.g. normals, colours)
- Target variable
  - Class label $y \in \{1, \ldots, k\}$
  - Segmentation mask $y \in \{1, \ldots, k\}^n$



Data Symmetries

CNNs

- Taking data symmetries into account simplifies learning
- E.g. convolutional neural networks allow shifting inputs



Gradient-based learning applied to document recognition Yann Lecun, Léon Bottou, Yoshua Bengio, Patrick Haffner

Data Symmetries

In general

- Consider function f and a group G acting on $\mathrm{dom}\, f$
- **Equivariance:** $\forall g \in G \;\; f(g \cdot x) = g \cdot f(x)$
  - E.g. convolutional layer, ReLU activation
- **Invariance:** $\forall g \in G \;\; f(g \cdot x) = f(g)$
  - E.g. global pooling operations
- Deep architectures exploit symmetries by stacking equivariant layers

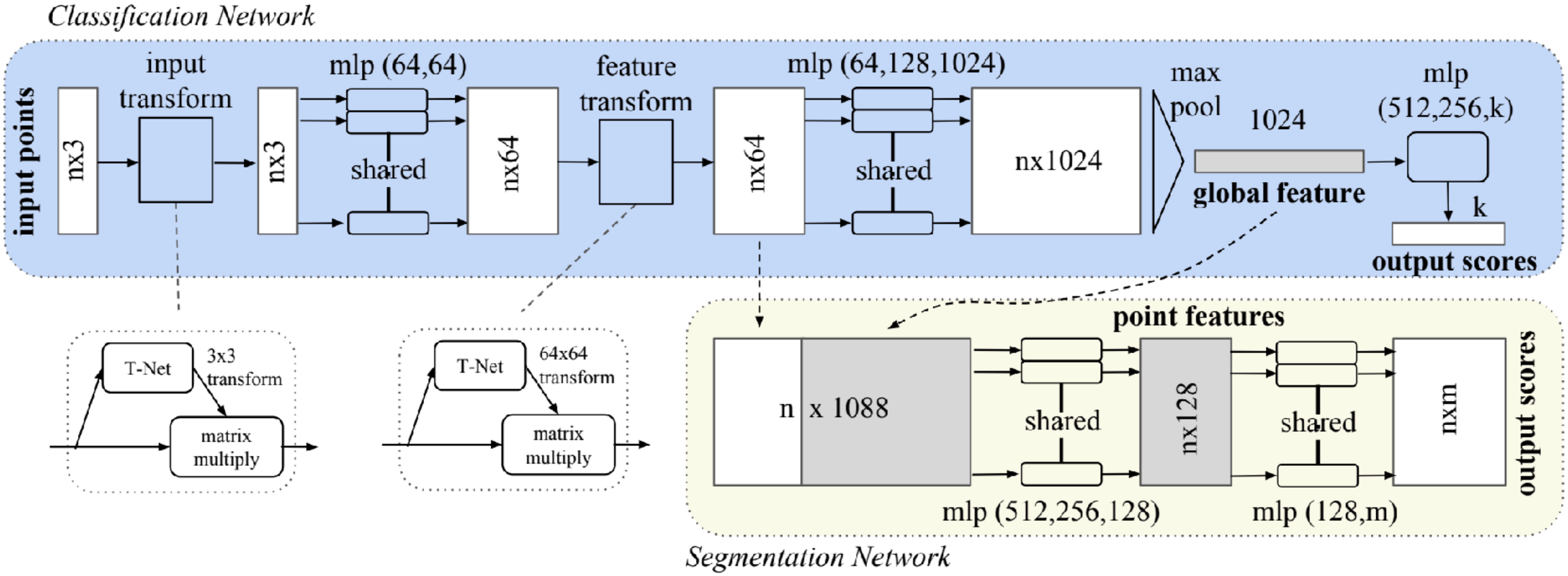
Point Cloud Symmetries

- Consider point cloud $x \in \mathbb{R}^{n \times d}$
- Permutation invariance
  - Prediction is robust w.r.t. ordering of n points
- Space symmetries
  - Rotation & shift invariance for d features of each point

Permutation Invariant Networks

- Two seminal papers proposed similar solutions for deep learning on sets
  - PointNet (December 2016)
    - $f(x_1, \ldots, x_n) = g\left(\mathrm{MAX}(h(x_1), \ldots h(x_n))\right)$
  - DeepSets (March 2017)
    - $f(x_1, \ldots, x_n) = g\left(\sum_{i=1}^{n} h(x_i)\right)$
  - In practice, h and g are MLPs
- Both works formulate universal approximation results

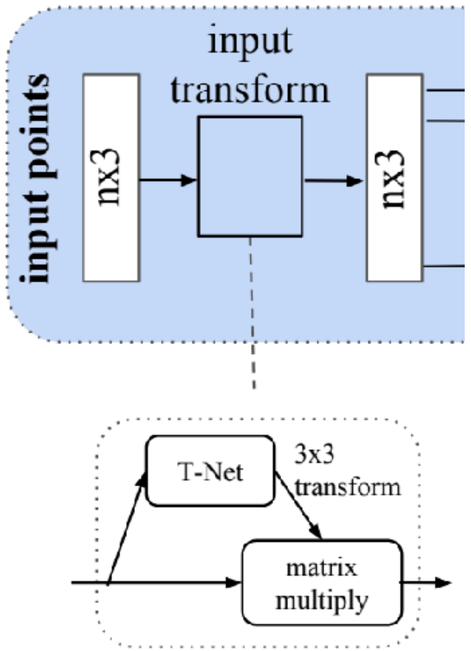
PointNet



Qi C. R. et al. Pointnet: Deep learning on point sets for 3d classification and segmentation

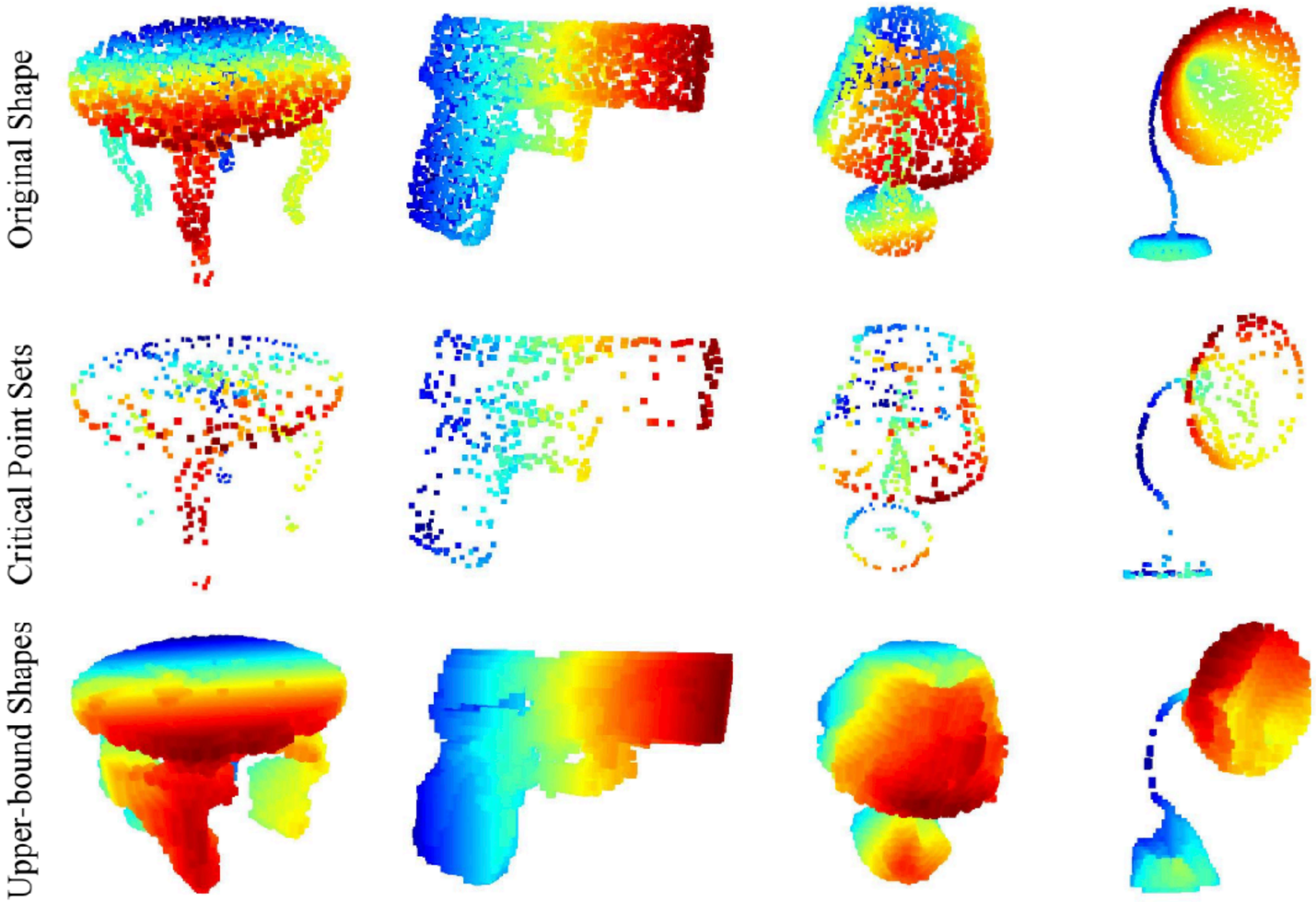
PointNet: Transform Blocks

- Idea: align input to canonical coordinates before processing
- T-Net is a tiny permutation-invariant network
  - Takes point-cloud as input
  - PointNet in miniautre: $g(\mathrm{MAX}(h(x_1), \ldots, h(x_n)))$
  - Outputs $n \times n$ matrix A
  - Regularize A to be orthogonal $L_{reg} = \|I - AA^T\|_F^2$



Qi C. R. et al. Pointnet: Deep learning on point sets for 3d classification and segmentation

PointNet: Critical Sets



PointNet Experiments

	input	#views	accuracy avg. class	accuracy overall
SPH [11]	mesh	-	68.2	-
3DShapeNets [28]	volume	1	77.3	84.7
VoxNet [17]	volume	12	83.0	85.9
Subvolume [18]	volume	20	86.0	**89.2**
LFD [28]	image	10	75.5	-
MVCNN [23]	image	80	**90.1**	-
Ours baseline	point	-	72.6	77.4
Ours PointNet	point	1	86.2	**89.2**

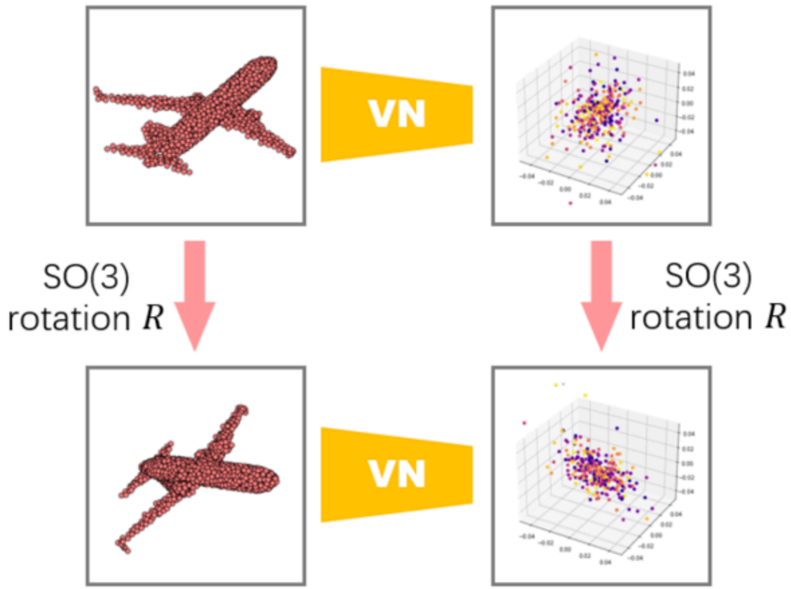
Table 1. **Classification results on ModelNet40.** Our net achieves state-of-the-art among deep nets on 3D input.

Transform	accuracy
none	87.1
input (3x3)	87.9
feature (64x64)	86.9
feature (64x64) + reg.	87.4
both	**89.2**

	#params	FLOPs/sample
PointNet (vanilla)	0.8M	148M
PointNet	3.5M	440M
Subvolume [18]	16.6M	3633M
MVCNN [23]	60.0M	62057M

Vector Neurons (VN)

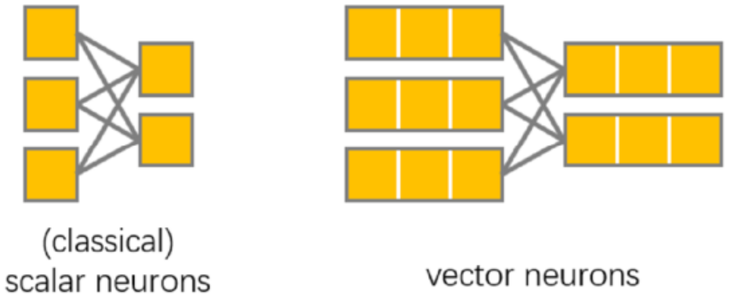
- PointNet augments data during training to learn rotation invariance
- Is there a way to design a rotation-invariant architecture?
- VN propose rotation equivariant
  - linear layers
  - activations
  - normalizations
  - pooling



Deng C. et al. Vector Neurons: A general framework for so (3)-equivariant networks // ICCV 2021

Vector Neurons: Linear Layers

- Idea: treat each neuron as a 3D point rather than scalar
- Replace scalar feature $v \in \mathbb{R}^{c \times 1}$ with a vector $v \in \mathbb{R}^{c \times 3}$
- Linear layer $W \in \mathbb{R}^{c' \times c}$: $v \to Wv$
- Rotation equivariance is trivial: $(Wv)R^T = W(vR^T)$
- Note: can't use bias term

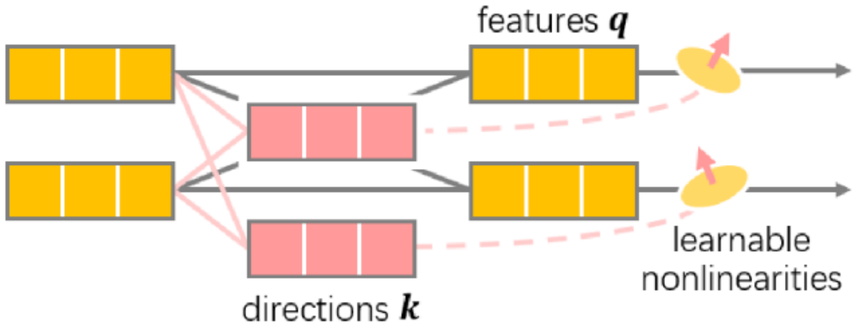


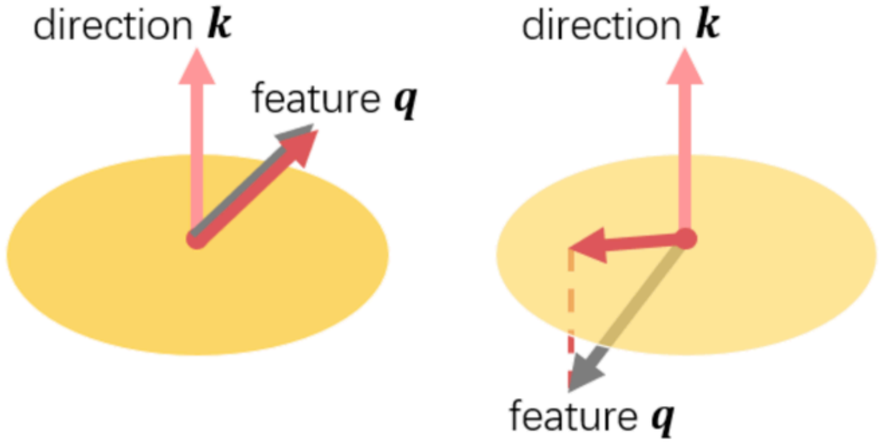
Deng C. et al. Vector Neurons: A general framework for so (3)-equivariant networks // ICCV 2021

Vector Neurons: 3D ReLU

- Let $v \in \mathbb{R}^{c\times 3}$ be the input
- ReLU is a half-space projection
- To generalize, consider $W, U \in \mathbb{R}^{1\times c}$
- Compute feature $q = Wv$ and direction $k = Uv$

$$v' = \begin{cases} q, & \text{if } \langle q, k\rangle \geq 0 \\ q - \langle q, \frac{k}{\|k\|}\rangle \frac{k}{\|k\|} & \text{otherwise} \end{cases}$$





Deng C. et al. Vector Neurons: A general framework for so (3)-equivariant networks // ICCV 2021

Equivariance in Practice

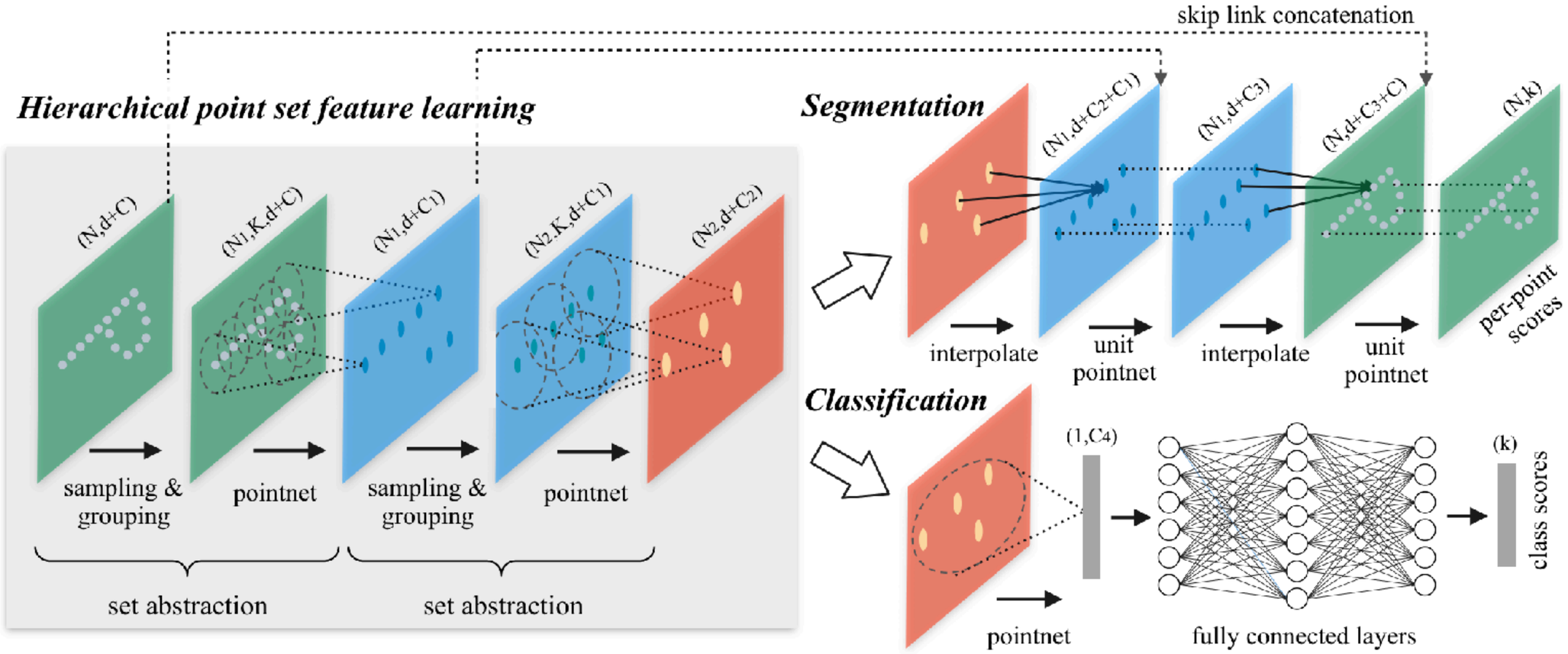
Methods	z/z	$z/\mathrm{SO}(3)$	$\mathrm{SO}(3)/\mathrm{SO}(3)$
Point / mesh inputs			
PointNet [25]	85.9	19.6	74.7
DGCNN [35]	90.3	33.8	88.6
VN-PointNet	77.5	77.5	77.2
VN-DGCNN	89.5	**89.5**	**90.2**

ShapeNet classification

Methods	$z/\mathrm{SO}(3)$	$\mathrm{SO}(3)/\mathrm{SO}(3)$
Point / mesh inputs		
PointNet [25]	38.0	62.3
DGCNN [35]	49.3	78.6
VN-PointNet	72.4	72.8
VN-DGCNN	**81.4**	**81.4**

ShapeNet segmentation

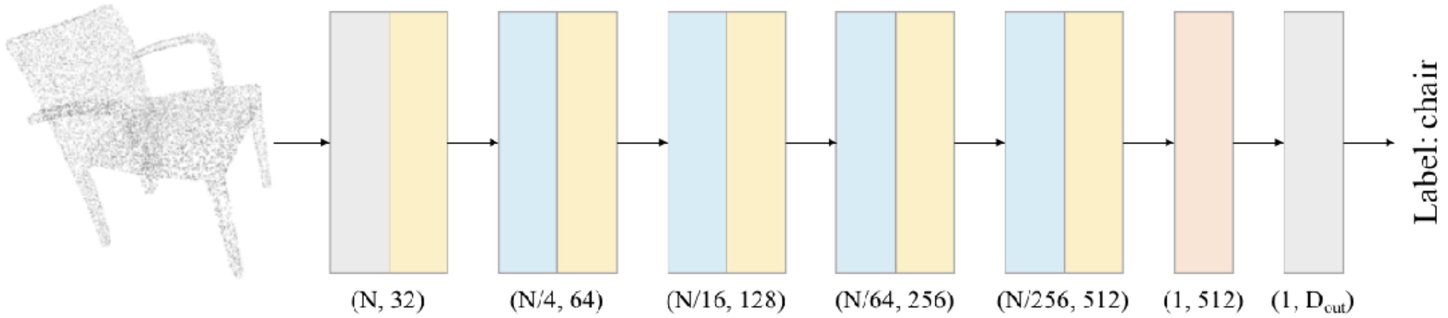
PointNet++



Qi C. R. et al. Pointnet++: Deep hierarchical feature learning on point sets in a metric space

Point Transformers

Classification



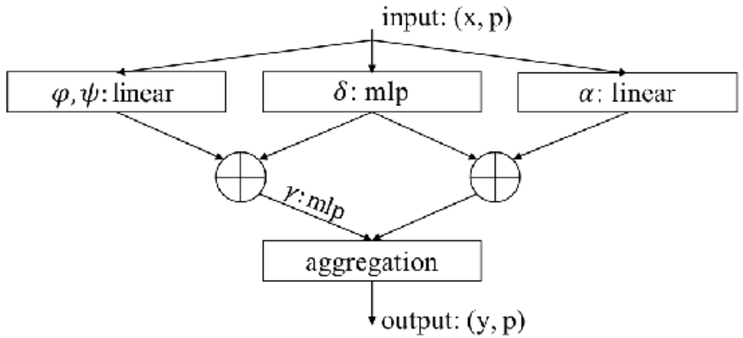
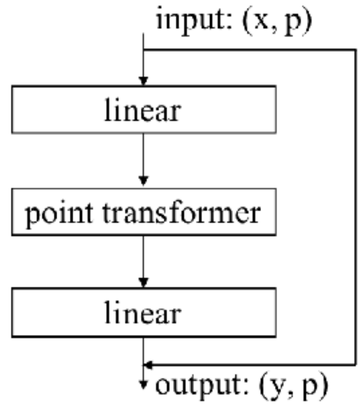
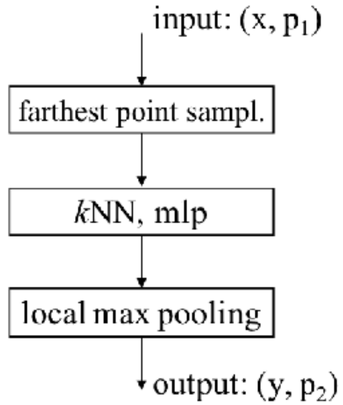


Figure 2. Point transformer layer.



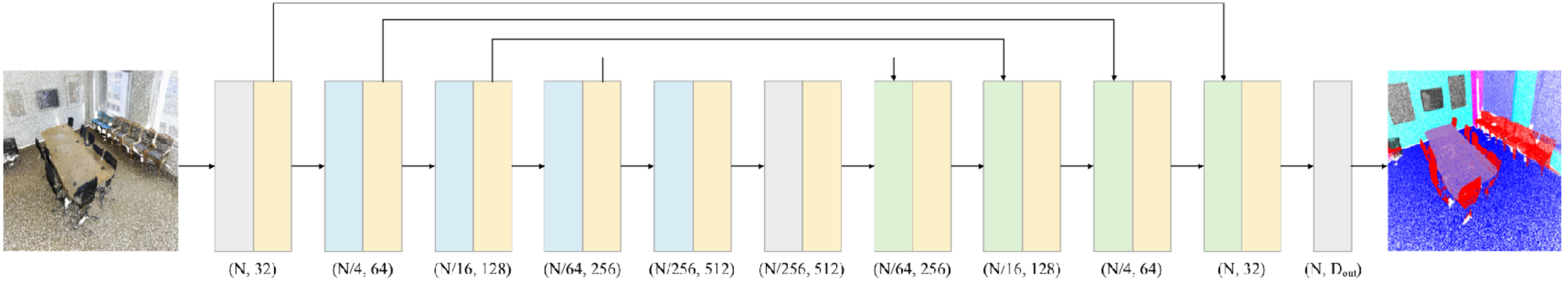
(a) point transformer block

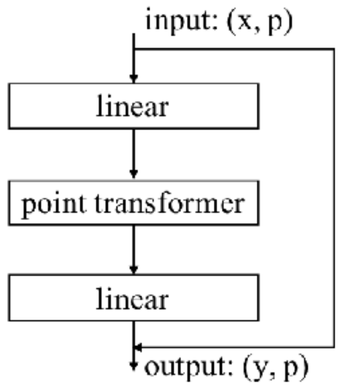


(b) transition down

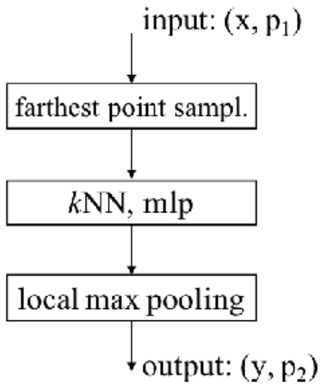
Point Transformers

Segmentation

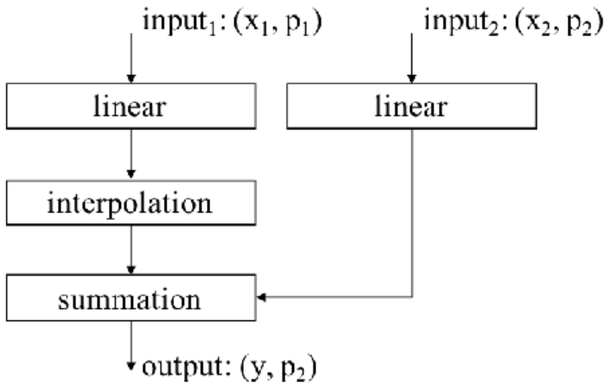




(a) point transformer block



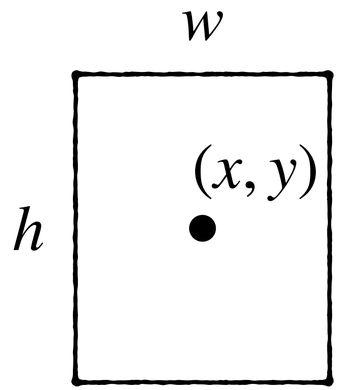
(b) transition down

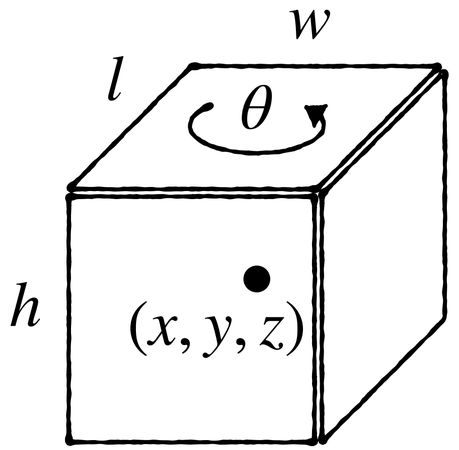


(c) transition up

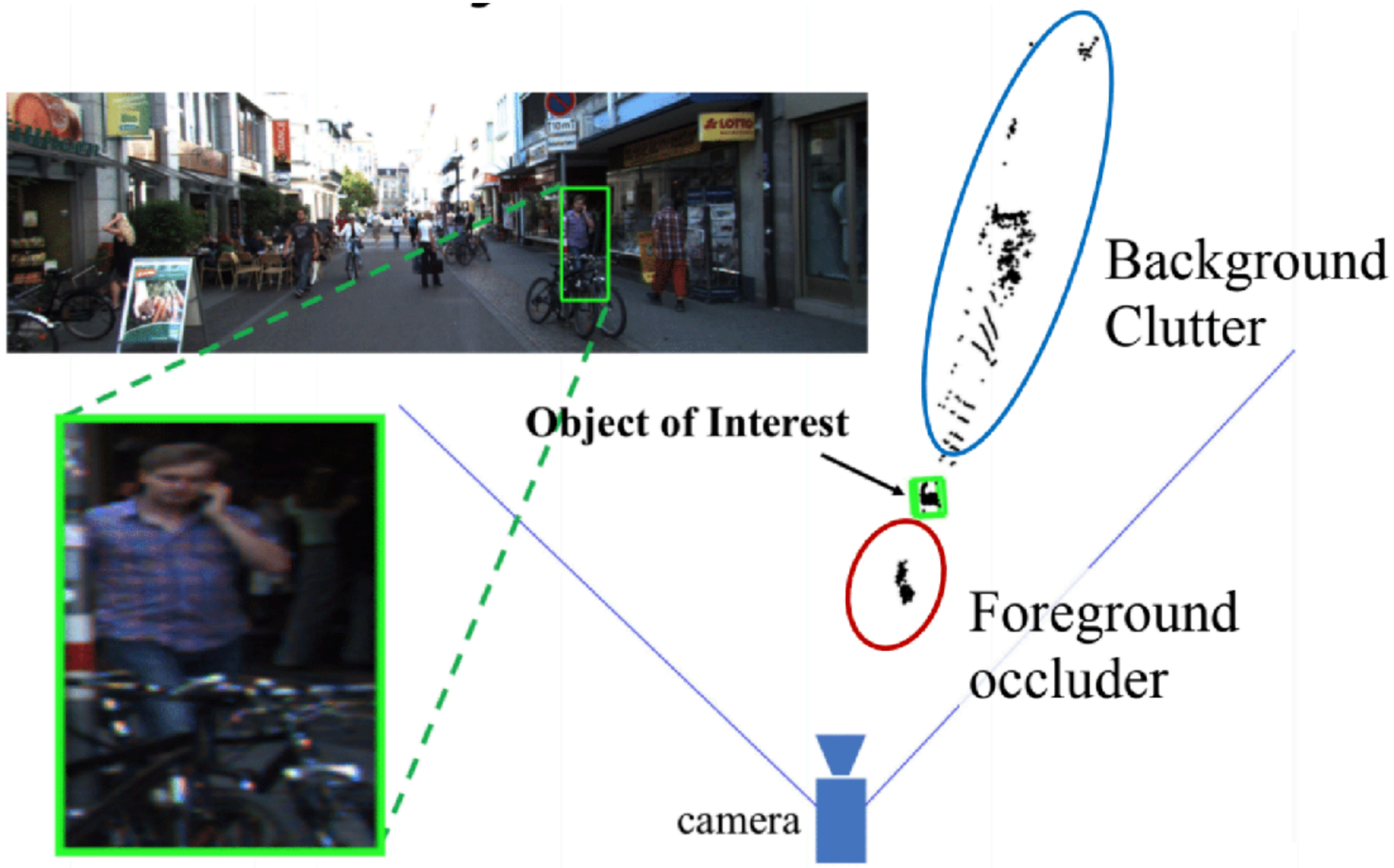
Object Detection is 3D

- Detection in 2D
  - Returns bounding box center, size, class confidence
- Detection in 3D
  - Center and size are 3D in this case, additionally returns box orientation

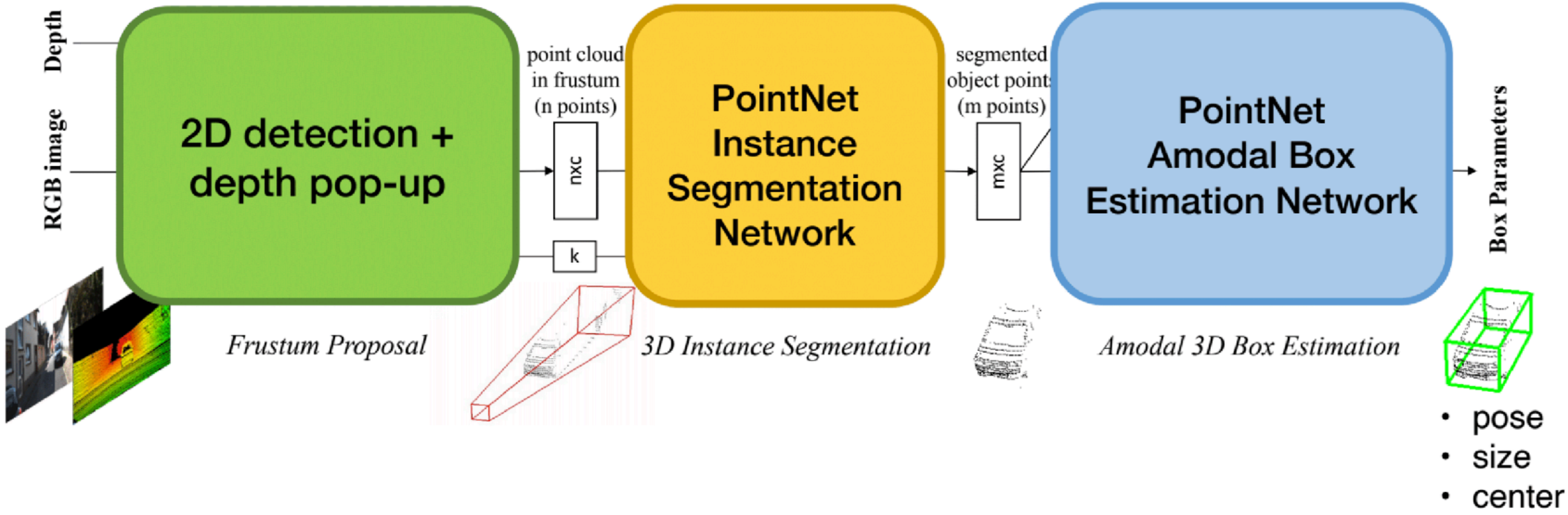




Challenges of RGBD Data

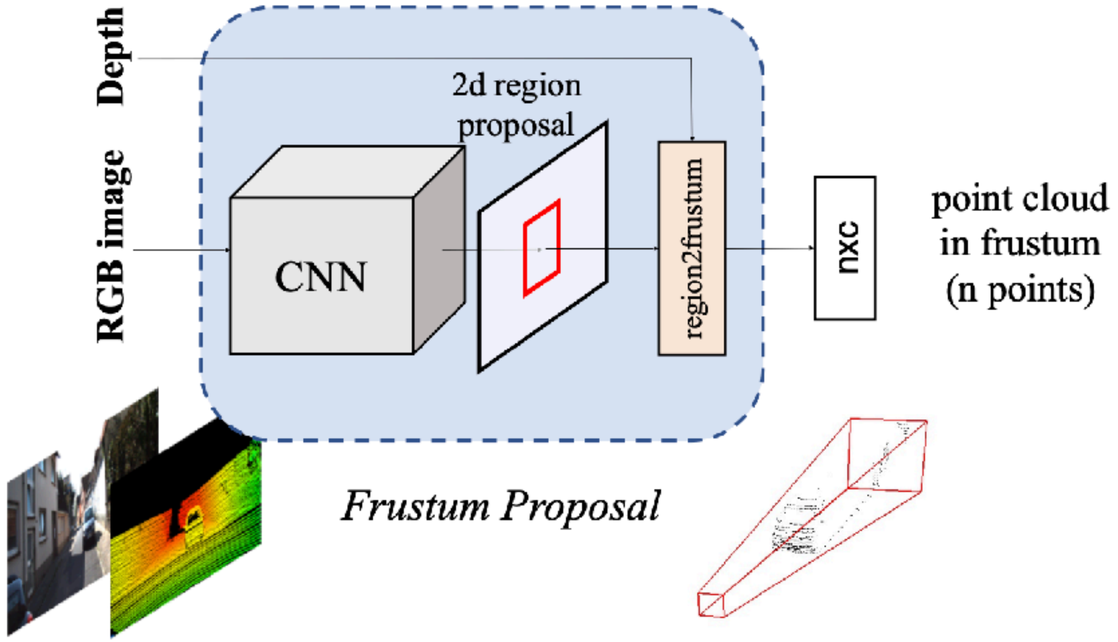


Frustrum PointNet



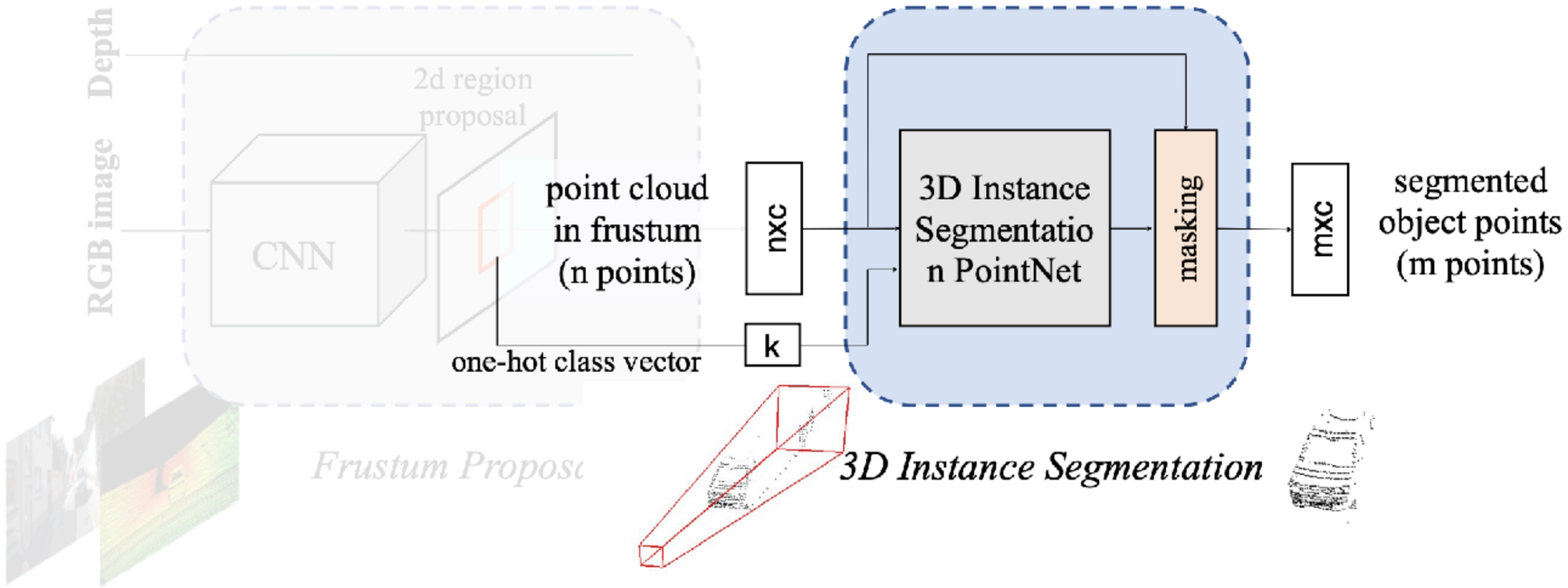
Qi C. R. et al. Frustum pointnets for 3d object detection from rgb-d data // CVPR 2018

I: Frustrum Proposal

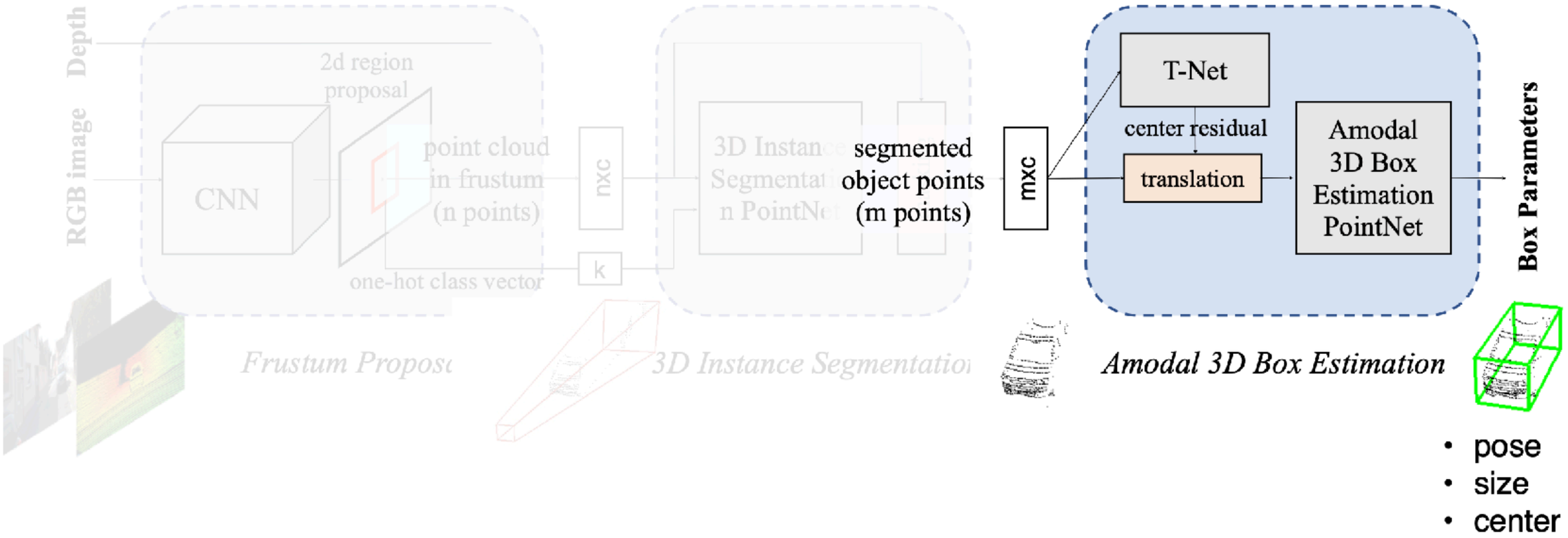


- RGB-D input
- Apply 2D detector
- Transform 2D bounding box into 3D frustum
- Remove points outside the frustum

II: Object Localisation



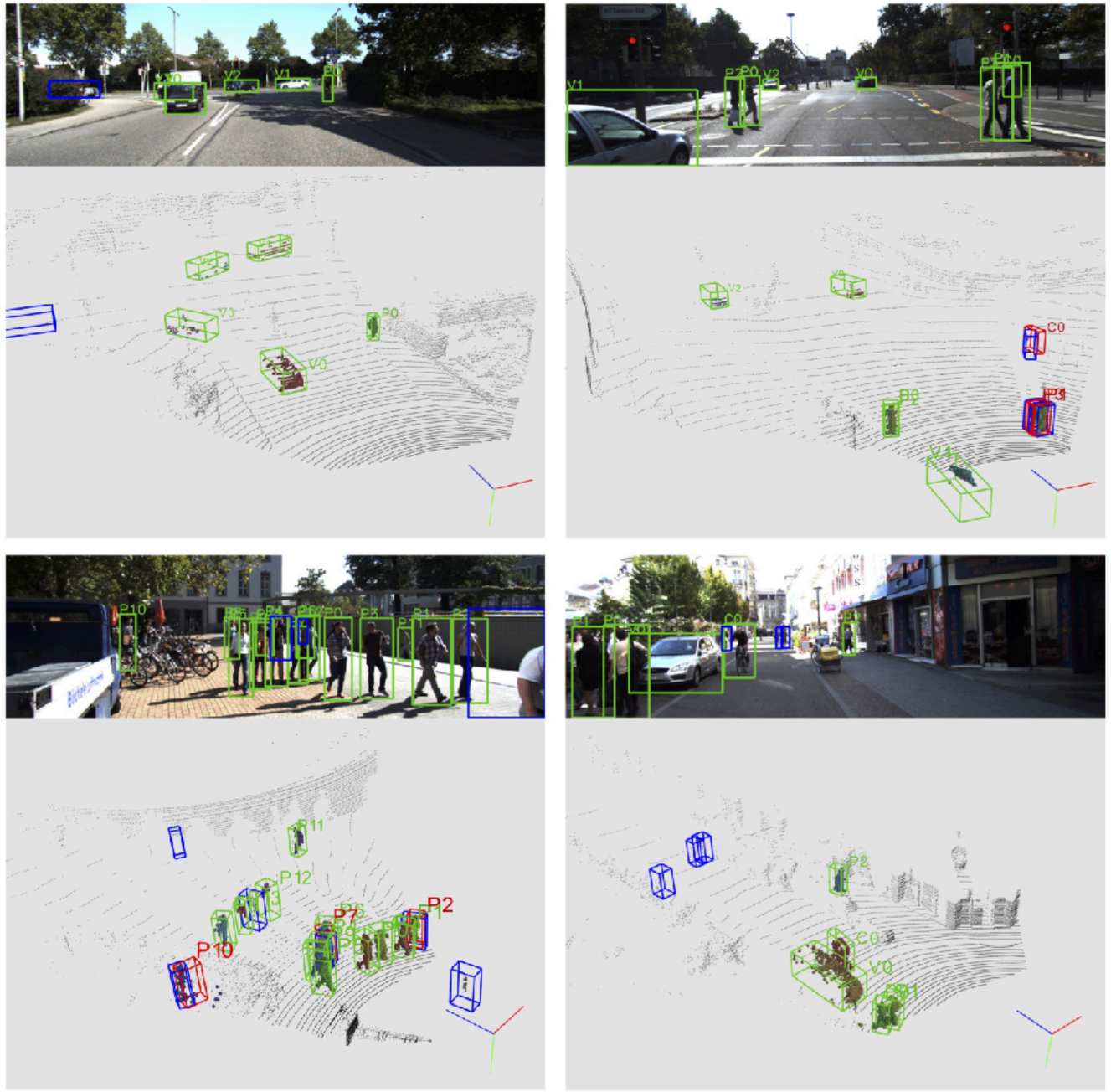
III: Bounding Box Estimation



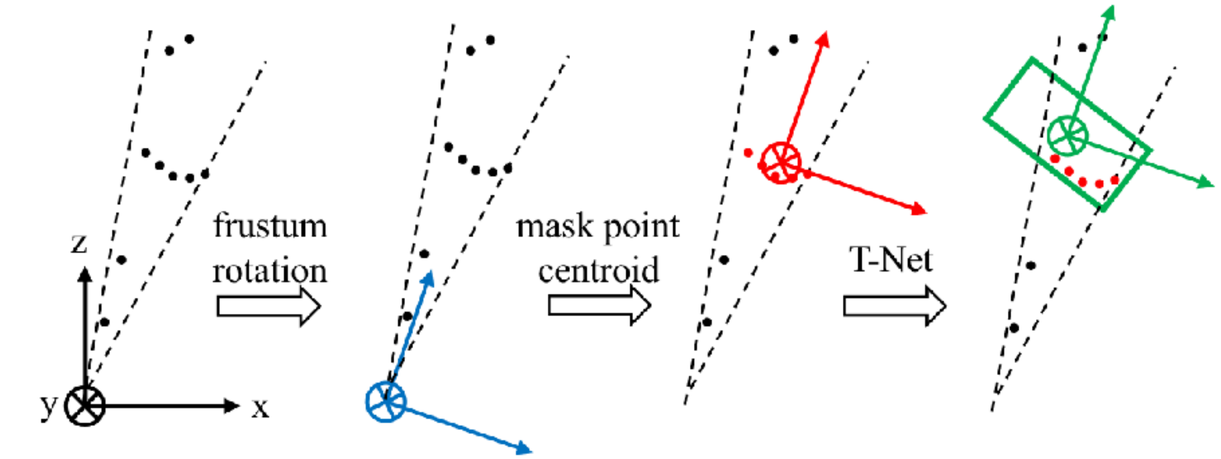
Some Restults

- Kitti 3D detection

Method	Cars		
	Easy	Moderate	Hard
DoBEM [42]	7.42	6.95	13.45
MV3D [6]	71.09	62.35	55.12
Ours (v1)	80.62	64.70	56.07
Ours (v2)	**81.20**	**70.39**	**62.19**



Coordinate Systems for Point Clouds



(a) camera coordinate (b) frustum coordinate (c) 3D mask coordinate (d) 3D object coordinate

frustum rot.	mask centralize	t-net	accuracy
-	-	-	12.5
√	-	-	48.1
-	√	-	64.6
√	√	-	71.5
√	√	√	**74.3**

Key Takeaways

- Invariance and equivariance in deep learning
- PointNet as an example of permutation-invariant architecture
- Vector Neurons for rotation equivariance
- Applications to classification, segmentation and 3D detection

- Next time: polygonal meshes & differentiable rendering