UC San Diego

Lecture 9:

Deep Learning on Point Cloud for Shape Analysis

Instructor: Hao Su

Feb 6, 2018

Agenda

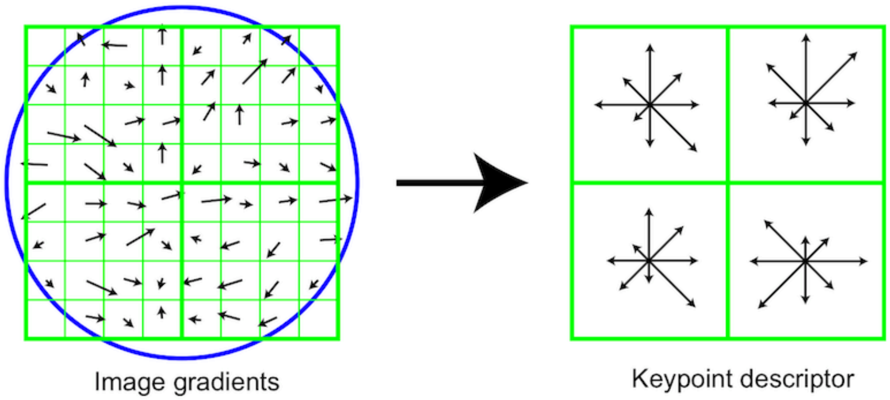
PointNet: A Basic Architecture for Point Cloud Processing

Using PointNet for 3D Object Detection

Image understanding: From feature engineering to learning

Feature engineering

SIFT
[Lowe, 1999]

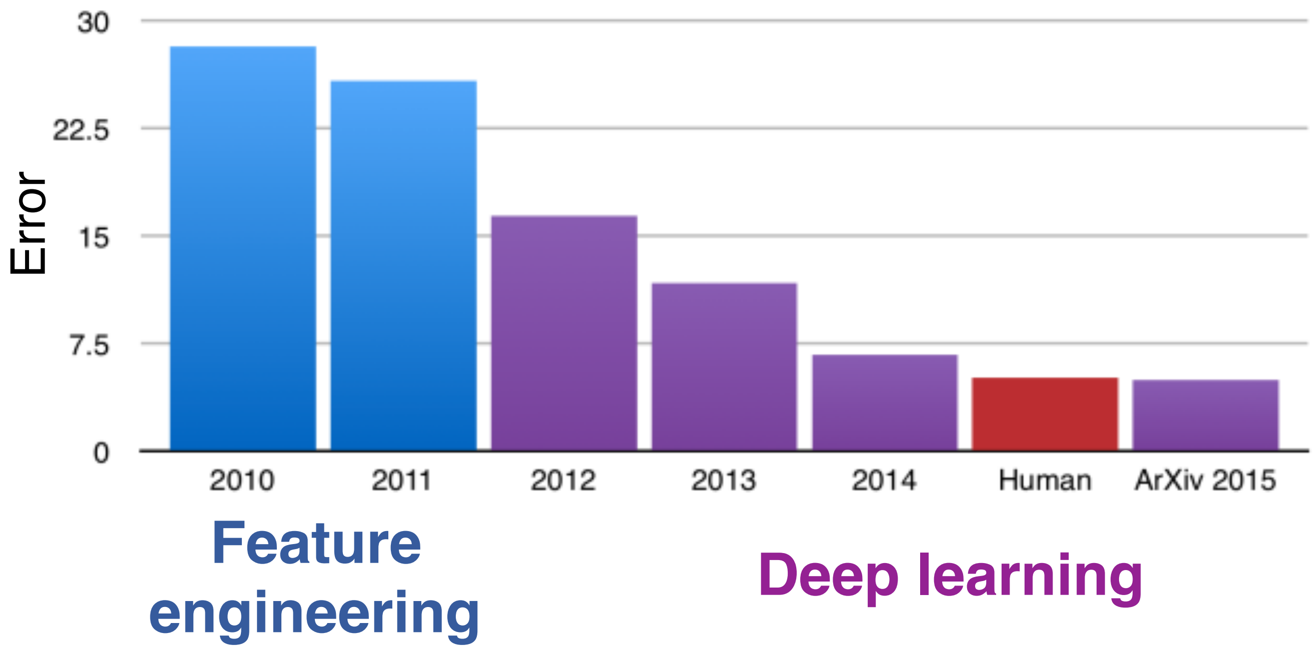


…

Image understanding: From feature engineering to learning

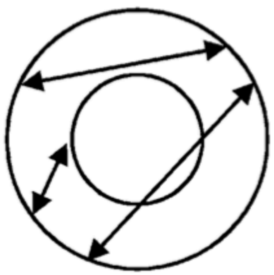
Feature learning

Object classification accuracy on ImageNet (ILSVRC)

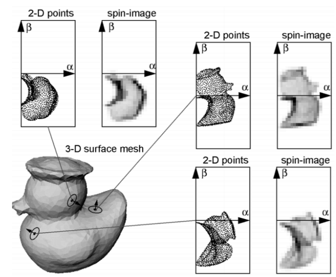


Prior art: Handcrafted 3D features

Representatives:



D2
[Osada, 2002]



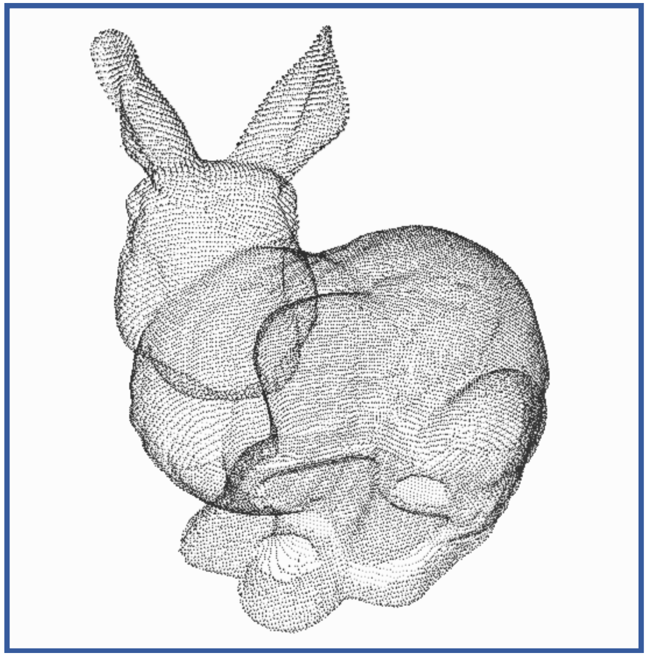
Spin Images
[Johnson, 1999]

Cons:

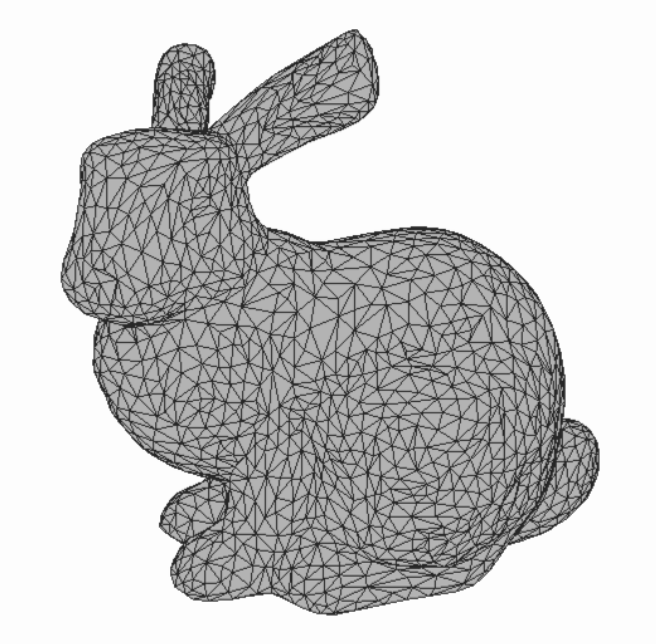
Hard **Representation-dependent** **Task-specific**

Fundamental challenge of 3D deep learning

Irregularity

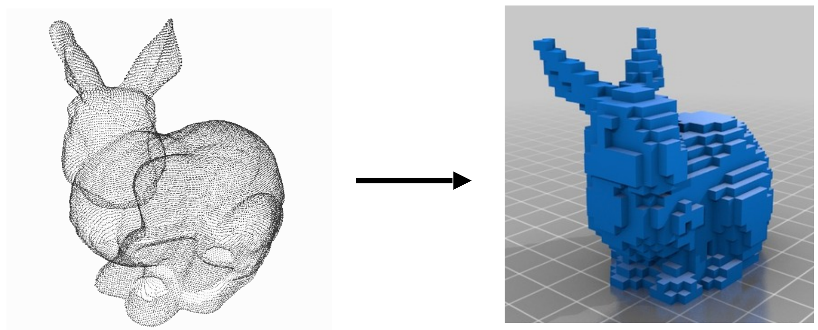


Point cloud
(The most common 3D sensor data)



Mesh
(The most common modeling data)

Solution 1: Convert irregular to regular



Point Volumetric

High space/time complexity $O(N^3)$

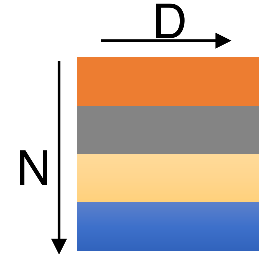
Information loss in voxelization

Solution 2: Directly process point cloud data

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

Properties of a desired neural network on point clouds

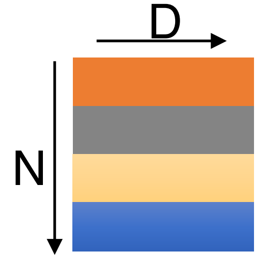
Point cloud: N **orderless** points, each represented by a D dim coordinate



2D array representation

Properties of a desired neural network on point clouds

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



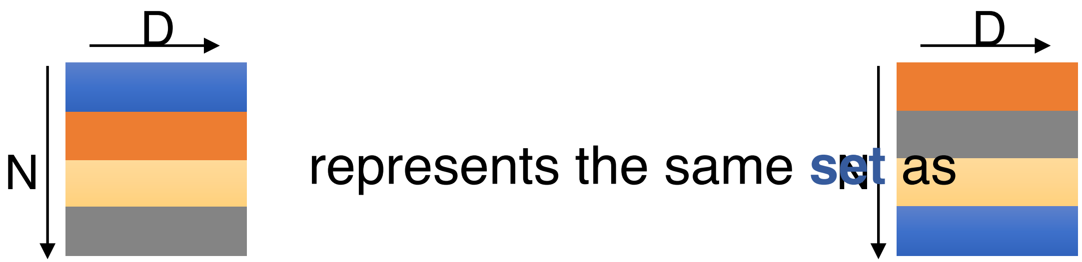
2D array representation

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



2D array representation

Permutation invariance

Permutation invariance:

$$f(x_1, x_2, \ldots, x_n) \equiv f(x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n}), \quad x_i \in \mathbb{R}^D$$

Examples:

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

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

…

Construct symmetric function family

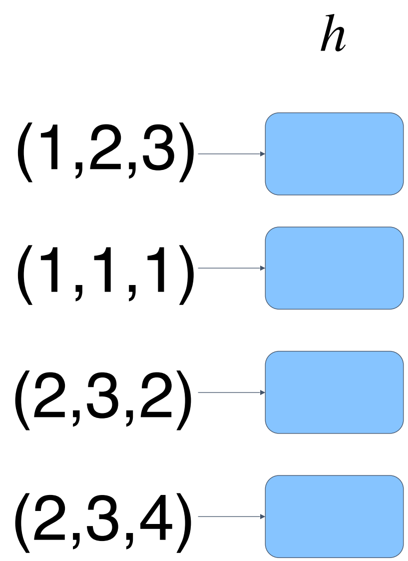
Observe:

$f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n))$ is symmetric if g is symmetric

Construct symmetric function family

Observe:

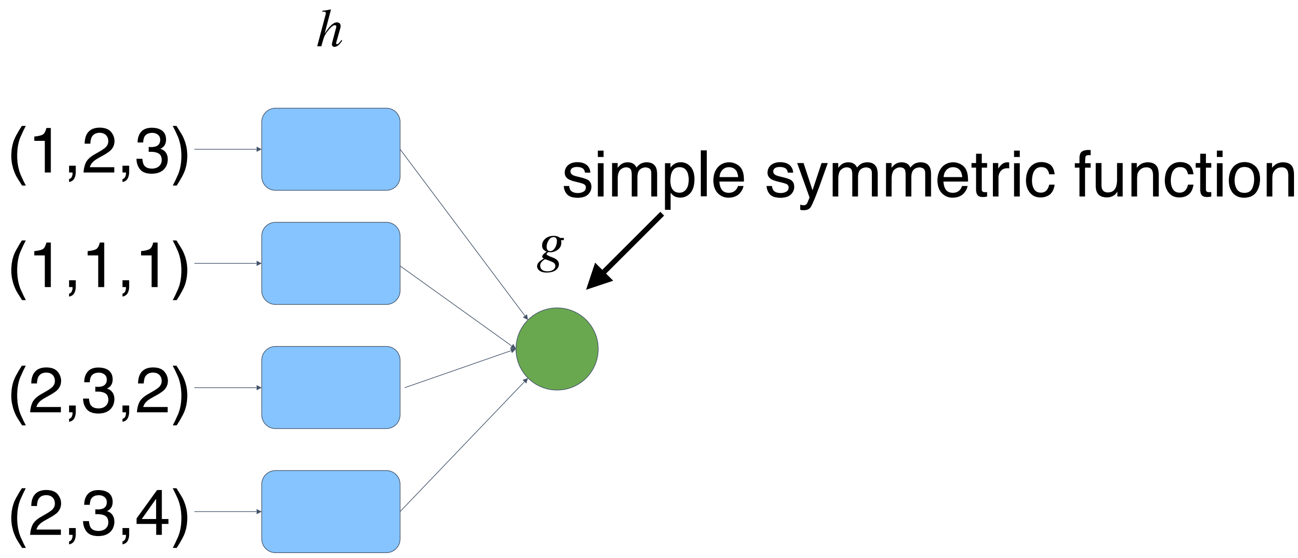
$f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n))$ is symmetric if g is symmetric



Construct symmetric function family

Observe:

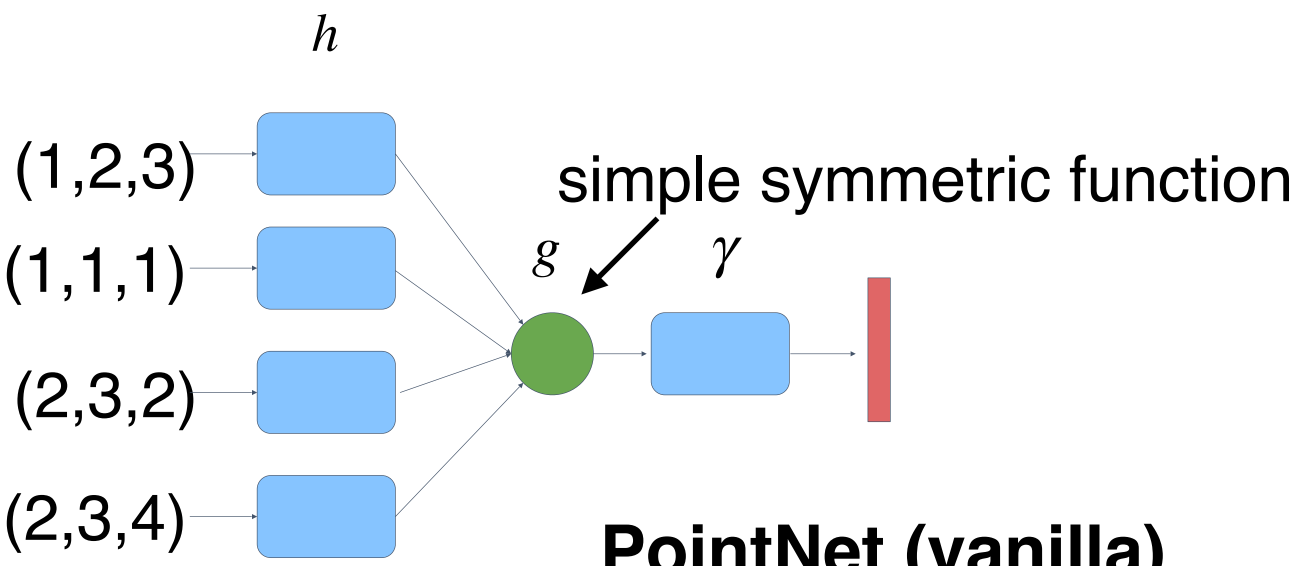
$f(x_1,x_2,\ldots,x_n)=\gamma\circ g(h(x_1),\ldots,h(x_n))$ is symmetric if g is symmetric



Construct symmetric function family

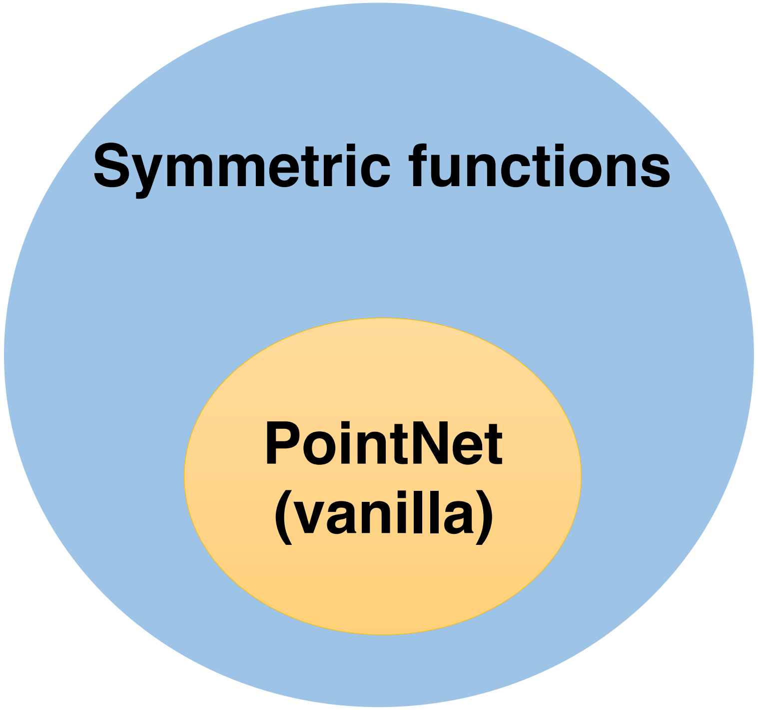
Observe:

$f(x_1, x_2, \ldots, x_n) = \gamma \circ g(h(x_1), \ldots, h(x_n))$ 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 $f : 2^{\mathcal{X}} \to \mathbb{R}$ can be arbitrarily approximated by PointNet.

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

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

PointNet (vanilla)

Properties of a desired neural network on point clouds

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

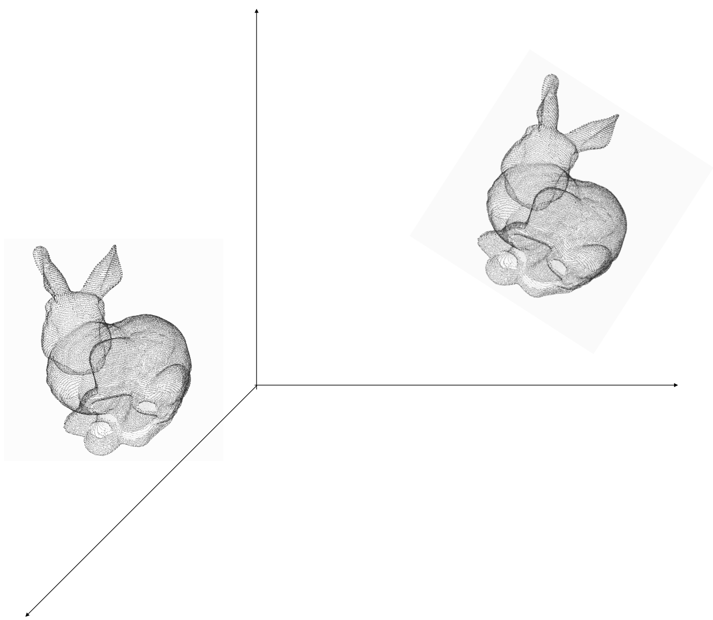
represents the same set as

2D array representation

Permutation invariance

Transformation invariance

Transformation invariance is desirable

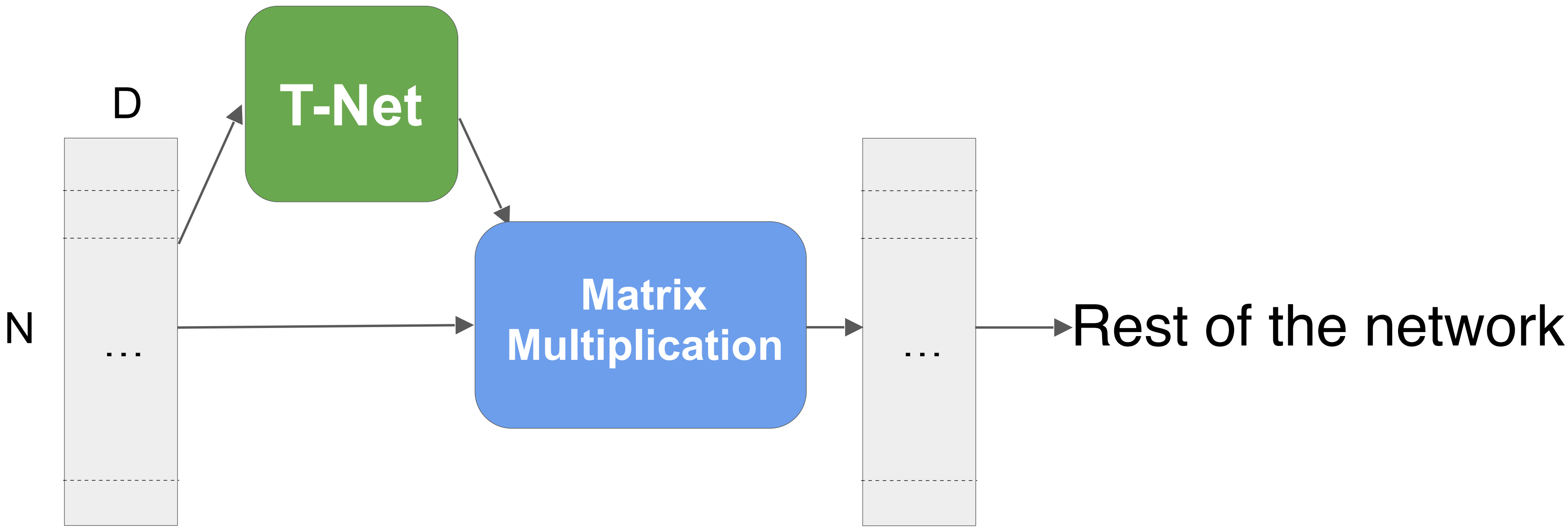


Let S be a shape. Then $f(T \cdot S) = f(S)$

f: classifier, T: transformation matrix

Incorporate transformer networks to input data

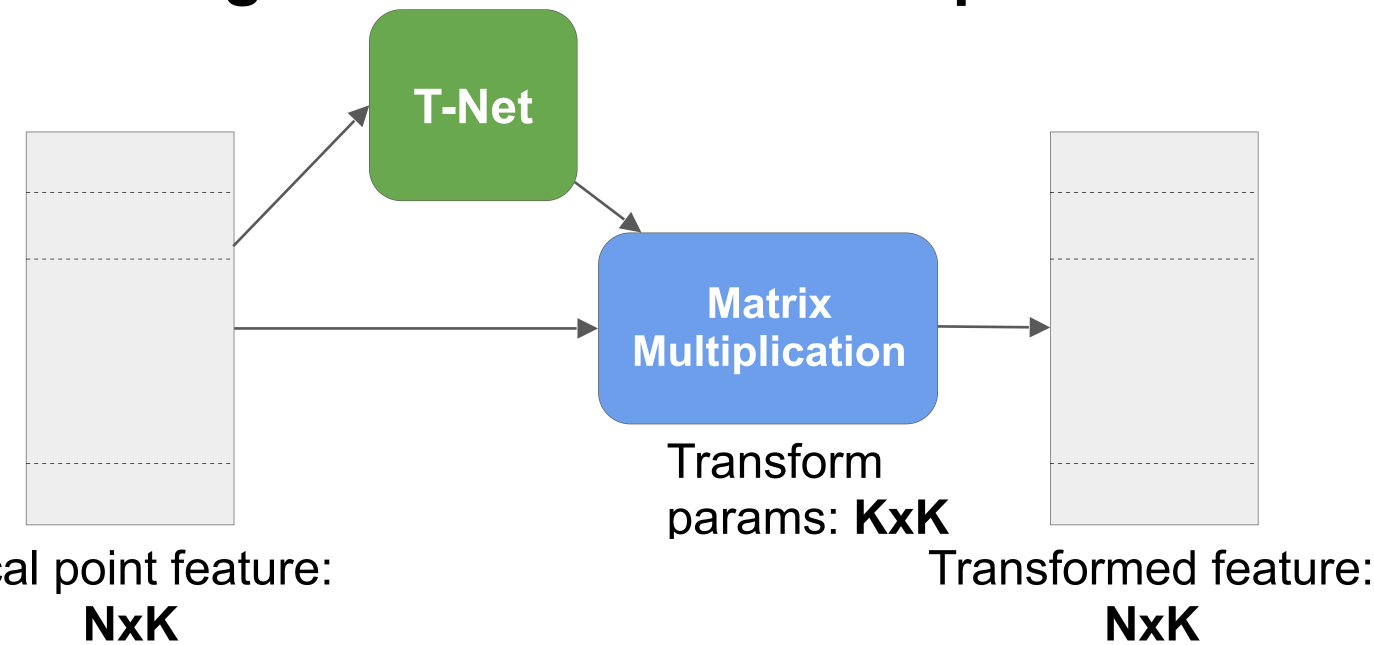
Input alignment to a canonical space



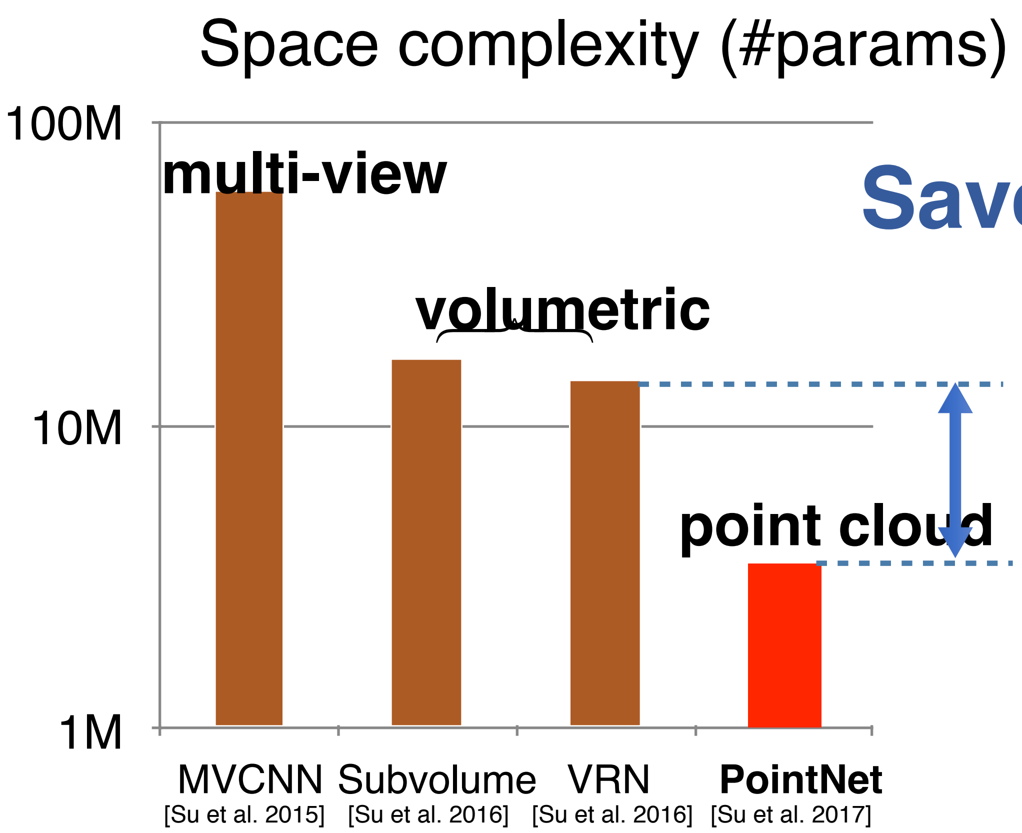
Incorporate transformer networks to feature space

Point Feature Transform: Feature alignment to a canonical space

Feature alignment to a canonical space



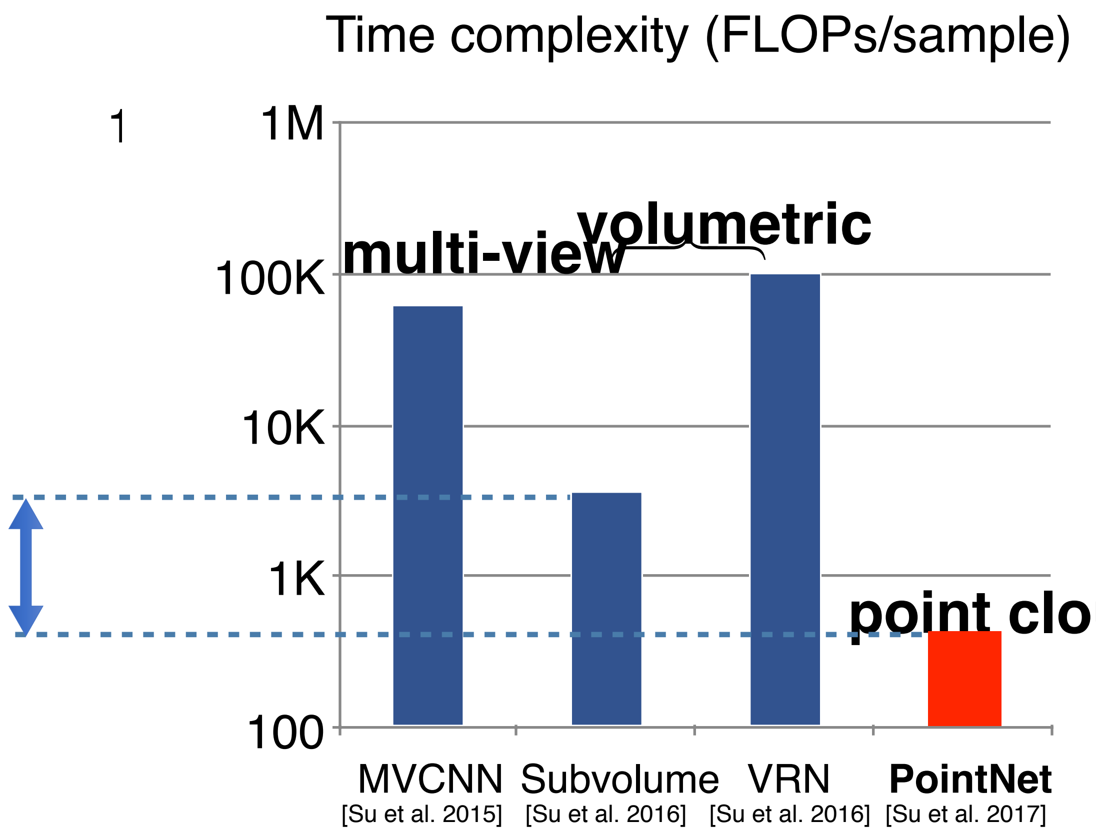
Efficiency of PointNet



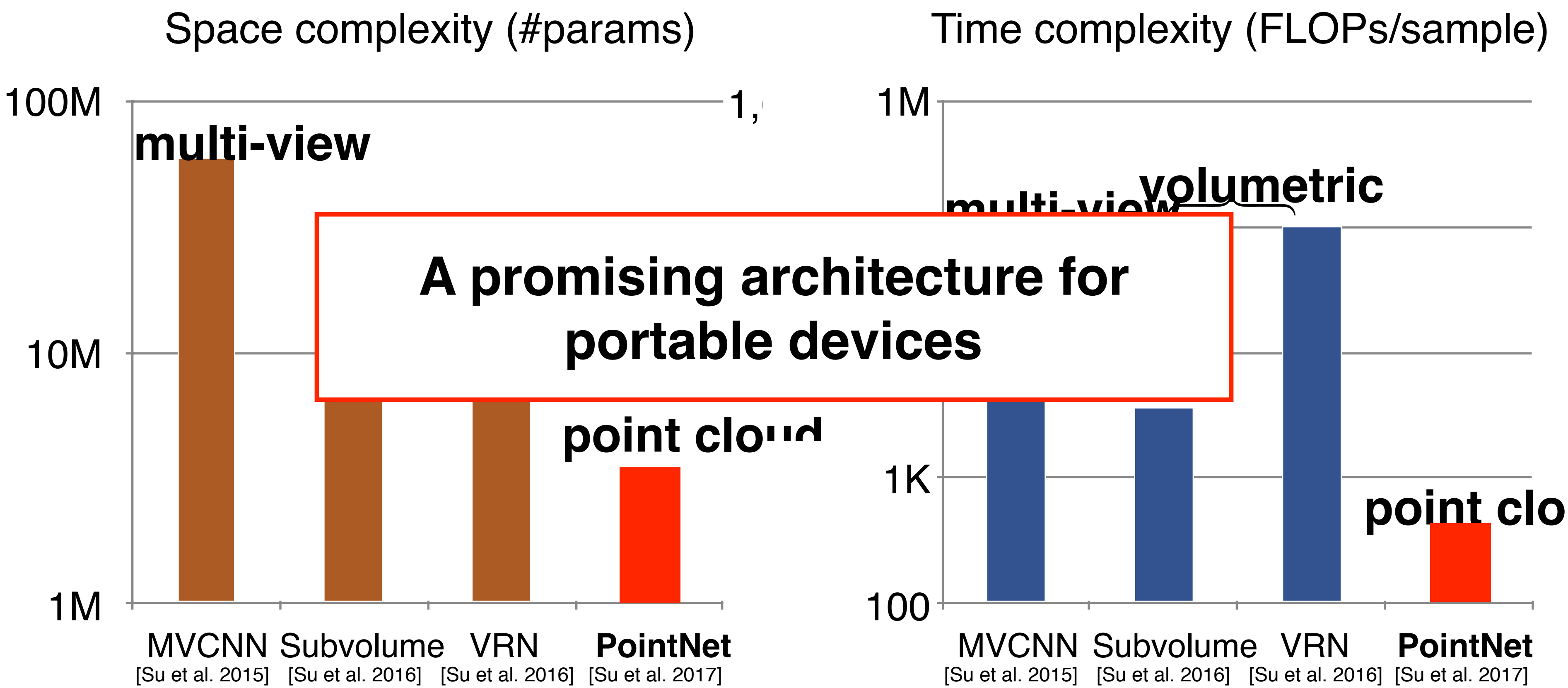
Saves 95% GPU memory

Efficiency of PointNet

Saves 95% time

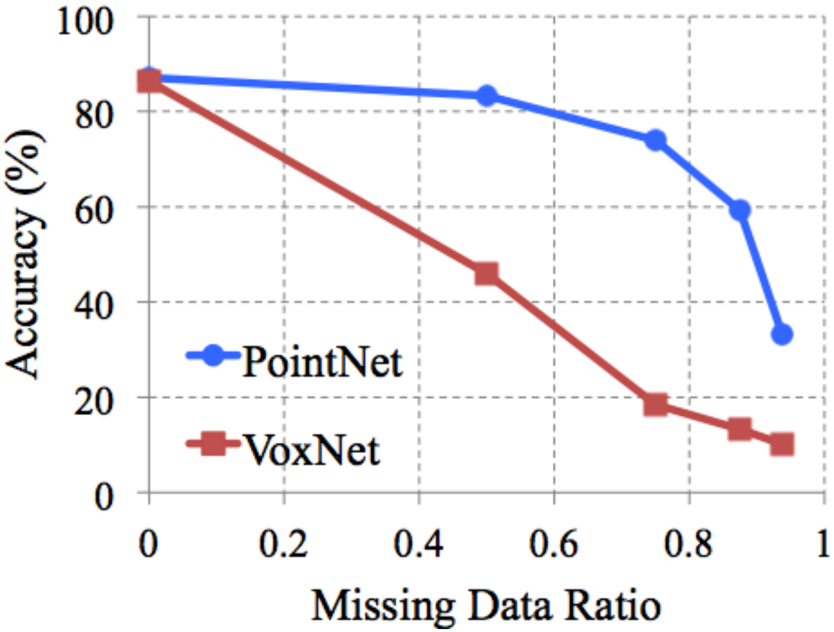


Efficiency of PointNet

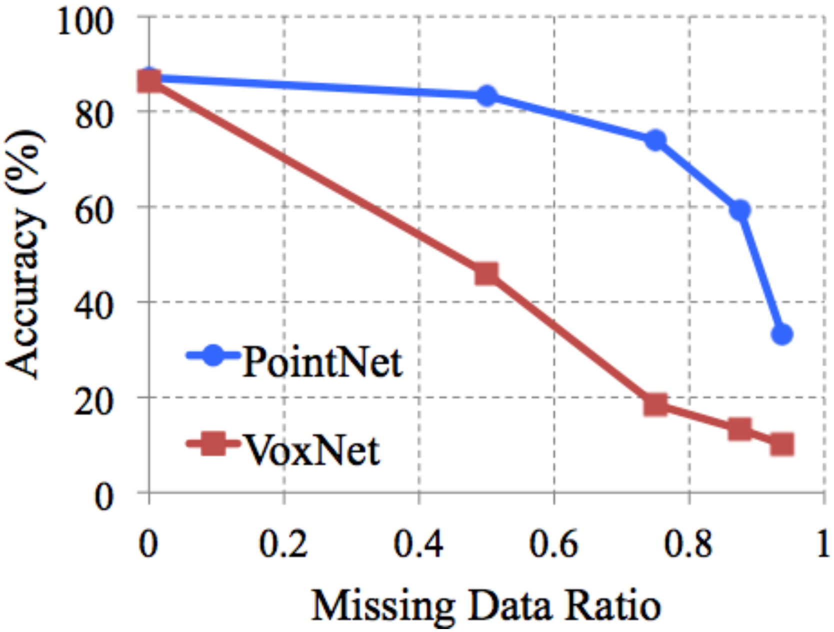


A promising architecture for portable devices

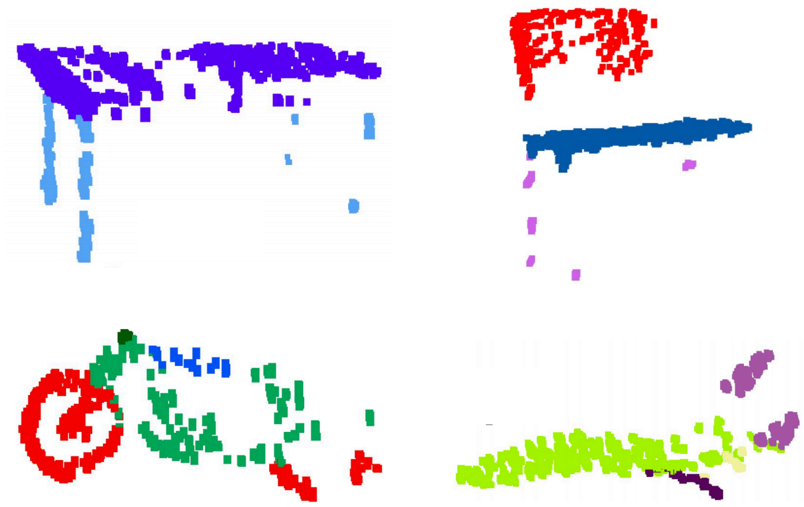
Robustness to data corruption



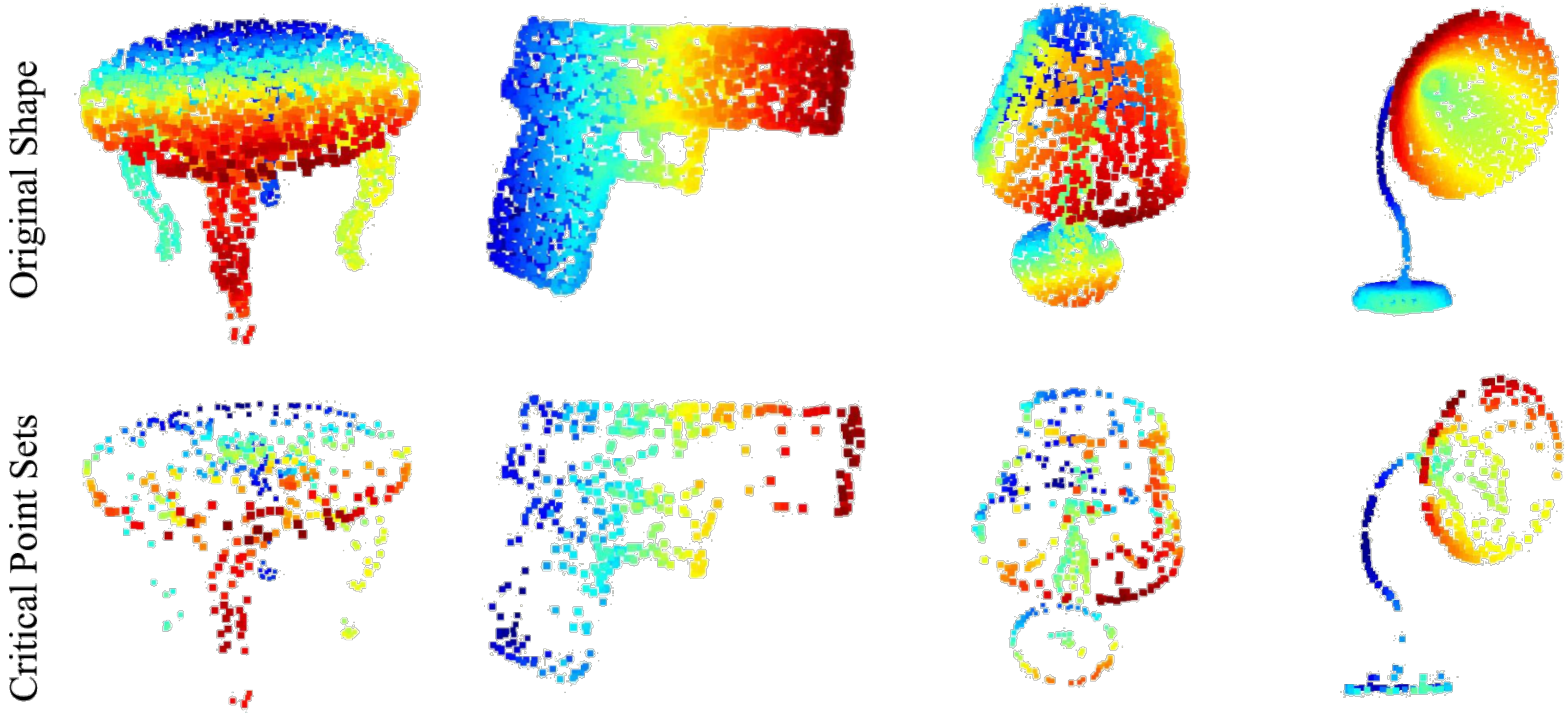
Robustness to data corruption



Segmentation from **partial scans**



Visualize what is learned by reconstruction



Salient points are discovered!

Agenda

PointNet: A Basic Architecture for Point Cloud Processing

Using PointNet for 3D Object Detection

Current State of Computer Vision

2D Deep Learning

Network Architectures:
AlexNet, Network in Network, VGG, GoogleNet, STN, ResNet, DenseNet, ...

Frameworks for Recognition:
R-CNN, Fast R-CNN, Faster-RCNN, SSD, YOLO, Feature Pyramid Network (FPN), Mask R-CNN etc.

3D Deep Learning

Network Architectures:
VoxNet, Multi-view CNN, FPNN, Octree CNN, Kd-network, PointNet, PointNet++ etc.

?

Current State of Computer Vision

2D Deep Learning

Network Architectures:
AlexNet, Network in Network, VGG, GoogleNet, STN, ResNet, DenseNet, ...

Frameworks for Recognition:
R-CNN, Fast R-CNN, Faster-RCNN, SSD, YOLO, Feature Pyramid Network (FPN), Mask R-CNN etc.

3D Deep Learning

Network Architectures:
VoxNet, Multi-view CNN, FPNN, Octree CNN, Kd-network, PointNet, PointNet++ etc.

This work: A novel framework for 3D object detection with PointNet architectures.

What is 3D Object Detection?

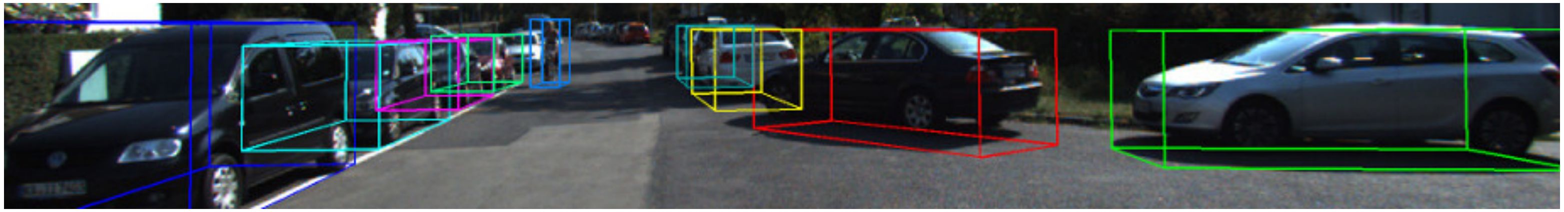
Input: RGB-D data

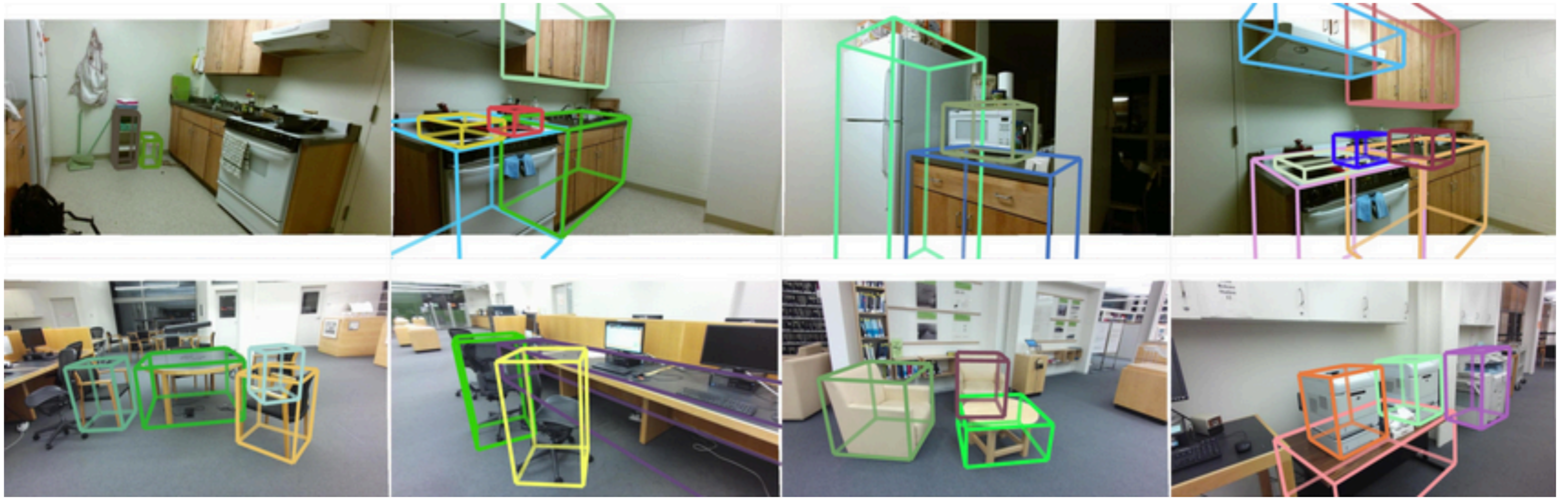
"D" can be sparse point cloud from LiDAR or dense depth map from indoor depth sensors

Output: Amodal 3D bounding boxes and semantic class labels for objects in the scene

"amodal" means the 3D box is for the "complete" object even if part of it is invisible.

What is 3D Object Detection?





What is 3D Object Detection?

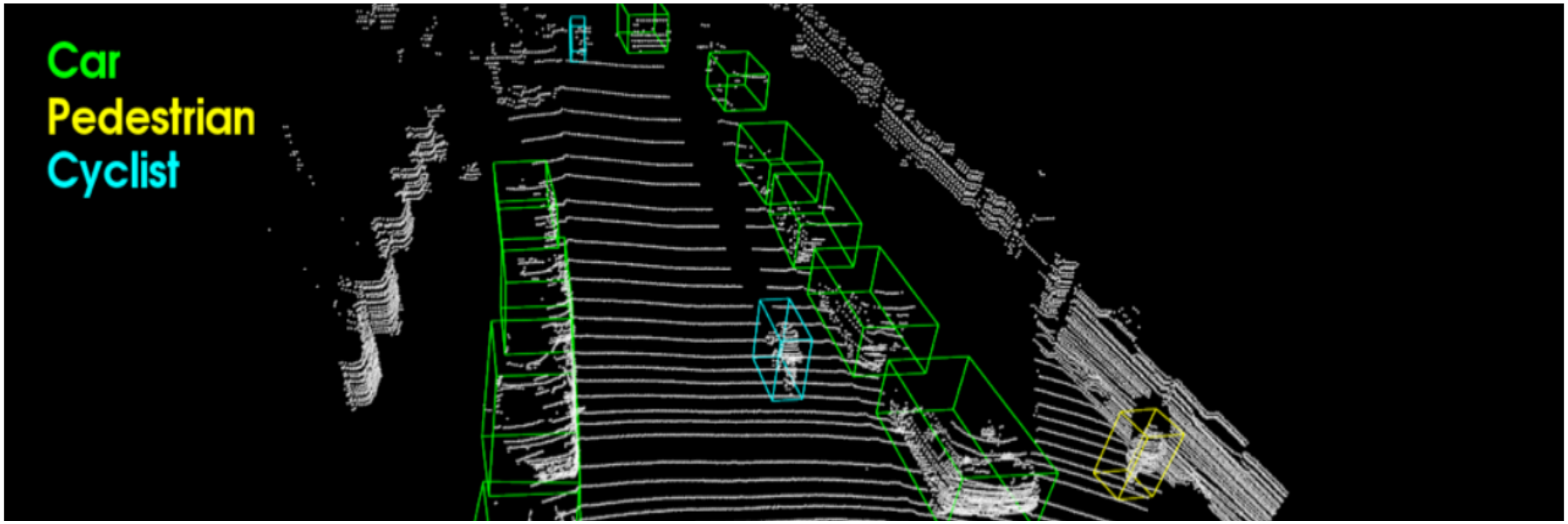
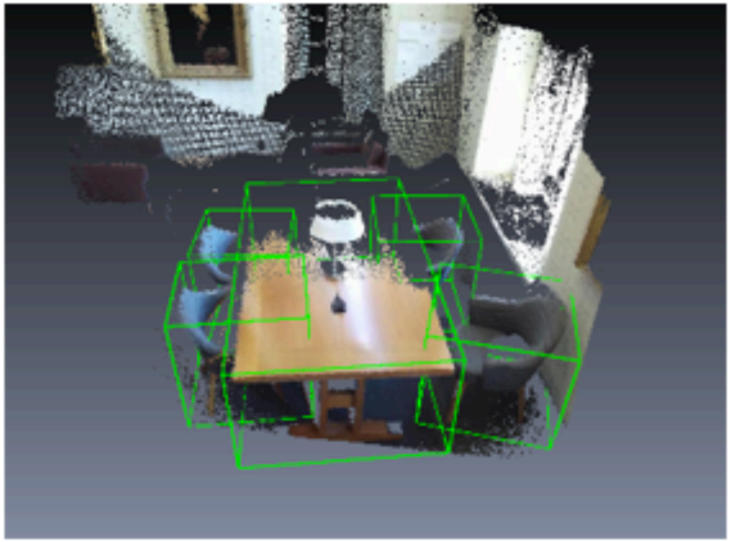


Figure from the recent VoxelNet paper from Apple.

What is 3D Object Detection?



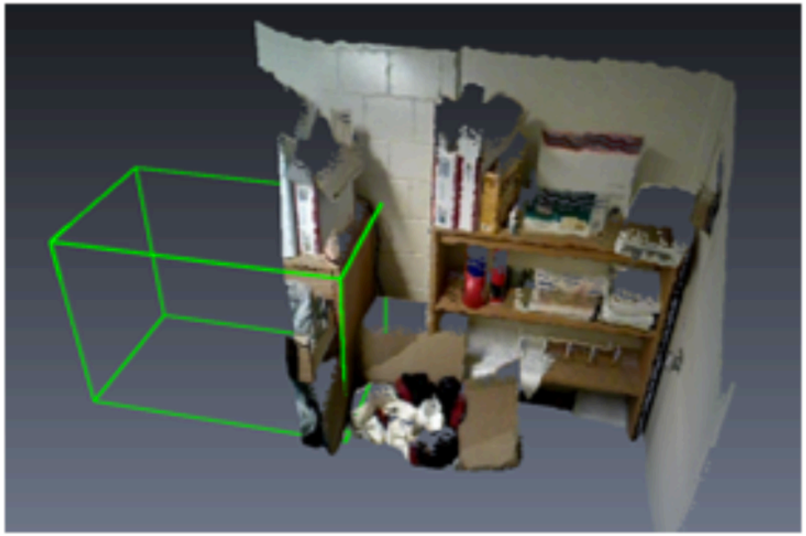
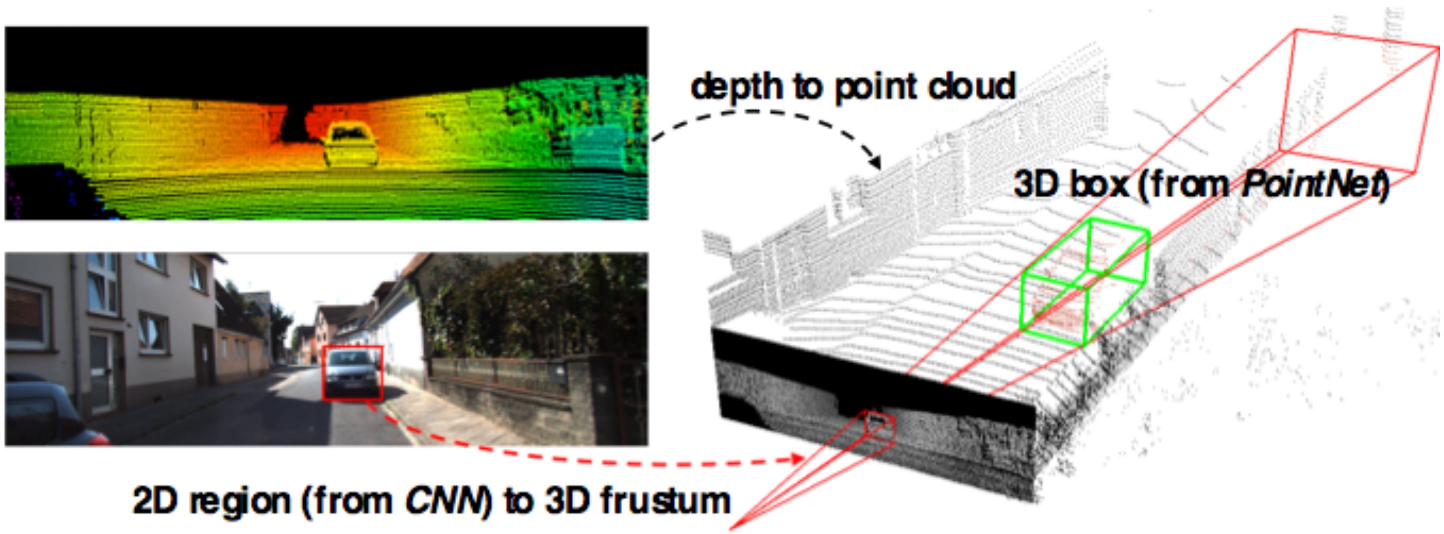


Figure from ICCV17 paper 2d-driven 3d object detection.

Frustum PointNets for 3D Object Detection



+ Leveraging mature 2D detectors for region proposal and 3D search space reduction
+ Solving 3D detection problem with 3D data and 3D deep learning architectures

Our method ranks No. 1 on KITTI 3D Object Detection Benchmark

We get 5% higher AP than Apple's recent CVPR submission and more than 10% higher AP than previous SOTA in easy category

Car

	Method	Setting	Code	Moderate	Easy	Hard	Runtime	Environment	Compare
1	F-PointNet			**70.39 %**	**81.20 %**	**62.19 %**	0.17 s	GPU @ 3.0 Ghz (Python)	
2	VxNet(LiDAR)			65.11 %	77.47 %	57.73 %	0.23 s	GPU @ 2.5 Ghz (Python + C/C++)	
3	AVOD			65.02 %	78.48 %	57.87 %	0.08 s	Titan X (pascal)	
4	MV3D			62.35 %	71.09 %	55.12 %	0.36 s	GPU @ 2.5 Ghz (Python + C/C++)	
X. Chen, H. Ma, J. Wan, B. Li and T. Xia: Multi-View 3D Object Detection Network for Autonomous Driving. CVPR 2017.									
5	MV3D (LIDAR)			52.73 %	66.77 %	51.31 %	0.24 s	GPU @ 2.5 Ghz (Python + C/C++)	
X. Chen, H. Ma, J. Wan, B. Li and T. Xia: Multi-View 3D Object Detection Network for Autonomous Driving. CVPR 2017.									
6	F-PC_CNN			42.67 %	50.46 %	40.15 %	0.5 s	GPU @ 3.0 Ghz (Matlab + C/C++)	
7	SDN			21.36 %	34.05 %	18.59 %	0.07 s	GPU @ 1.5 Ghz (Python)	
8	LMNetV2			15.24 %	14.75 %	12.85 %	0.02 s	GPU @ 2.5 Ghz (C/C++)	
9	3dSSD			14.97 %	14.71 %	19.43 %	0.03 s	GPU @ 2.5 Ghz (Python + C/C++)	
10	LMnet			9.19 %	11.32 %	9.19 %	0.1 s	GPU @ 1.1 Ghz (Python + C/C++)	

⋮

Our method ranks No. 1 on KITTI 3D Object Detection Benchmark

We are also 1st place for smaller objects (ped. and cyclist) winning with even bigger margins.

Pedestrian

	Method	Setting	Code	Moderate	Easy	Hard	Runtime	Environment	Compare
1	F-PointNet			**44.89 %**	**51.21 %**	**40.23 %**	0.17 s	GPU @ 3.0 Ghz (Python)	
2	VxNet(LiDAR)			33.69 %	39.48 %	31.51 %	0.23 s	GPU @ 2.5 Ghz (Python + C/C++)	
3	AVOD			25.87 %	32.67 %	25.01 %	0.08 s	Titan X (pascal)	
4	3dSSD			17.35 %	20.22 %	17.20 %	0.03 s	GPU @ 2.5 Ghz (Python + C/C++)	

⋮

Cyclist

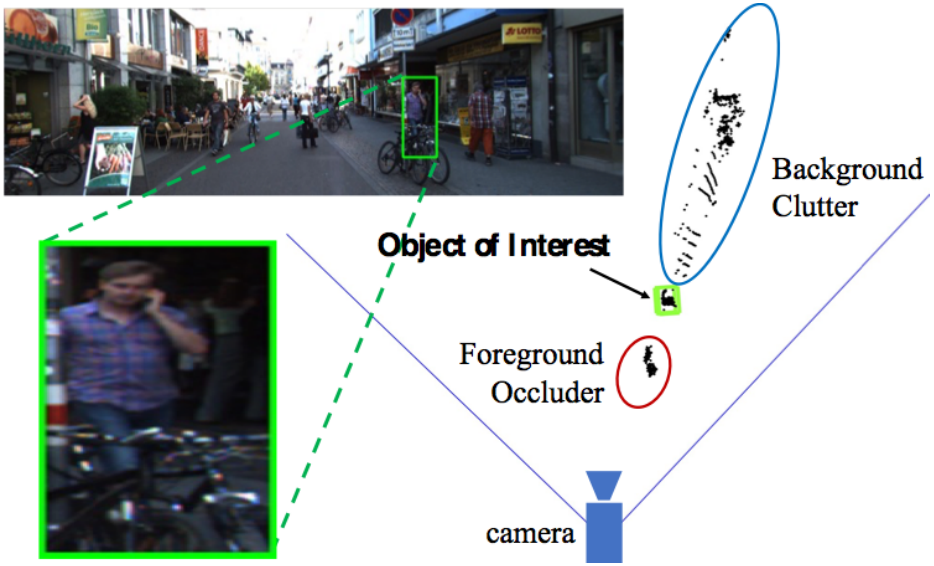
	Method	Setting	Code	Moderate	Easy	Hard	Runtime	Environment	Compare
1	F-PointNet			**56.77 %**	**71.96 %**	**50.39 %**	0.17 s	GPU @ 3.0 Ghz (Python)	
2	VxNet(LiDAR)			48.36 %	61.22 %	44.37 %	0.23 s	GPU @ 2.5 Ghz (Python + C/C++)	
3	AVOD			30.43 %	43.74 %	30.12 %	0.08 s	Titan X (pascal)	

⋮

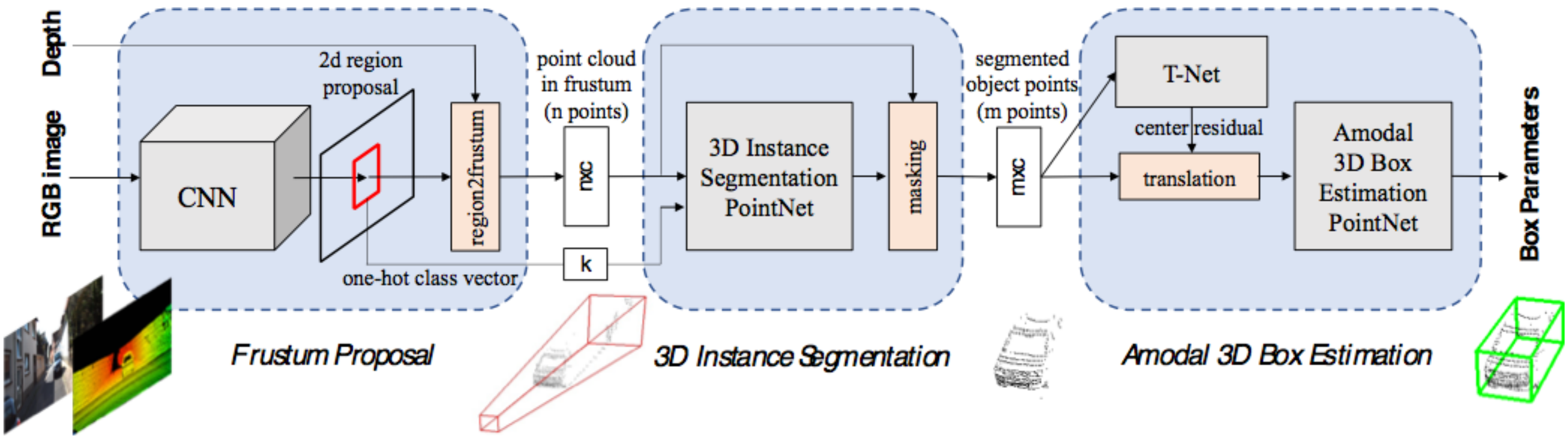
Frustum-based 3D Object Detection

Challenges:

- Occlusions and clutters are common in frustum point cloud.
- Largely varying ranges of points in frustums.

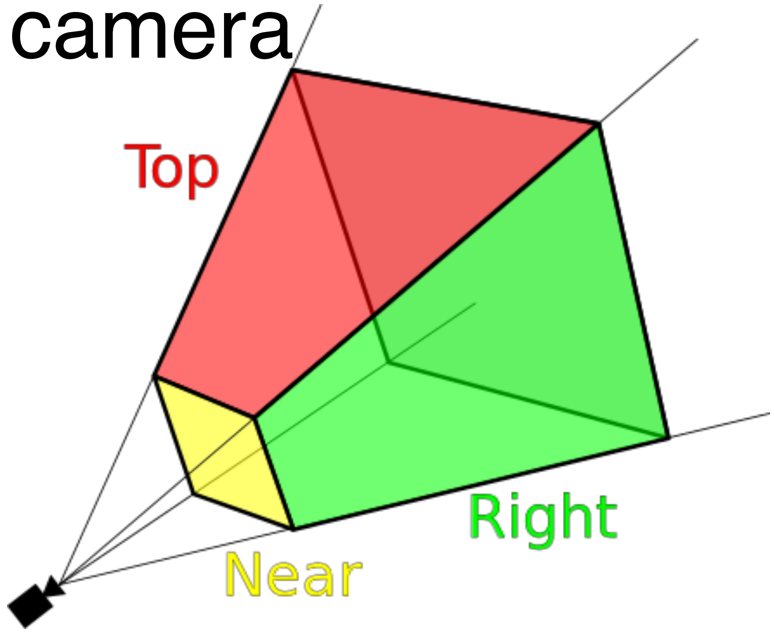


Frustum PointNets



Frustum Proposal

It is the **3D field of view** of the notional camera

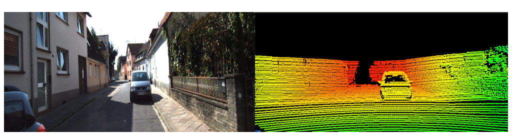


Frustum Proposal

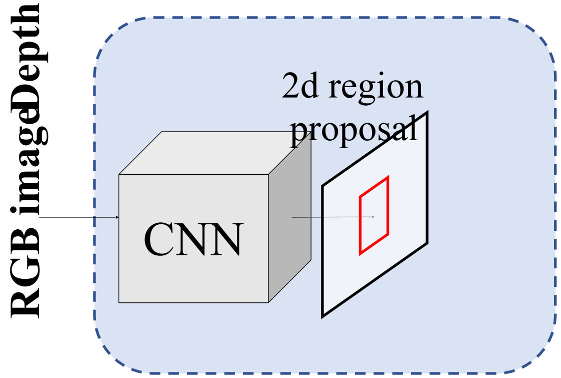
RGB image

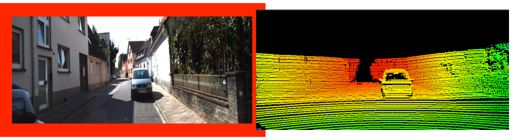
Depth

Input: RGB-D data



Frustum Proposal

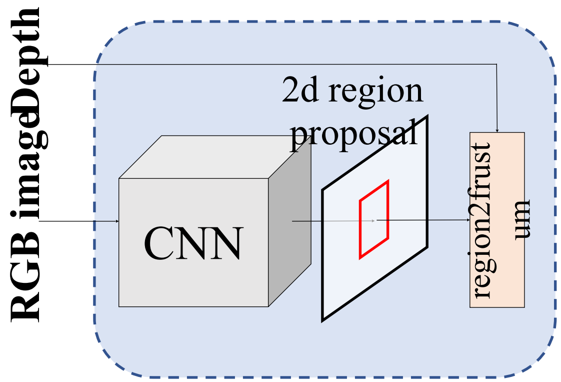




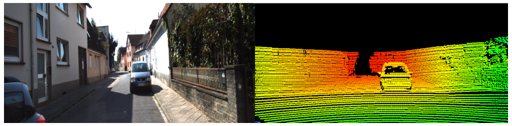
Input: RGB-D data

Image region proposal

Frustum Proposal



Frustum Proposal

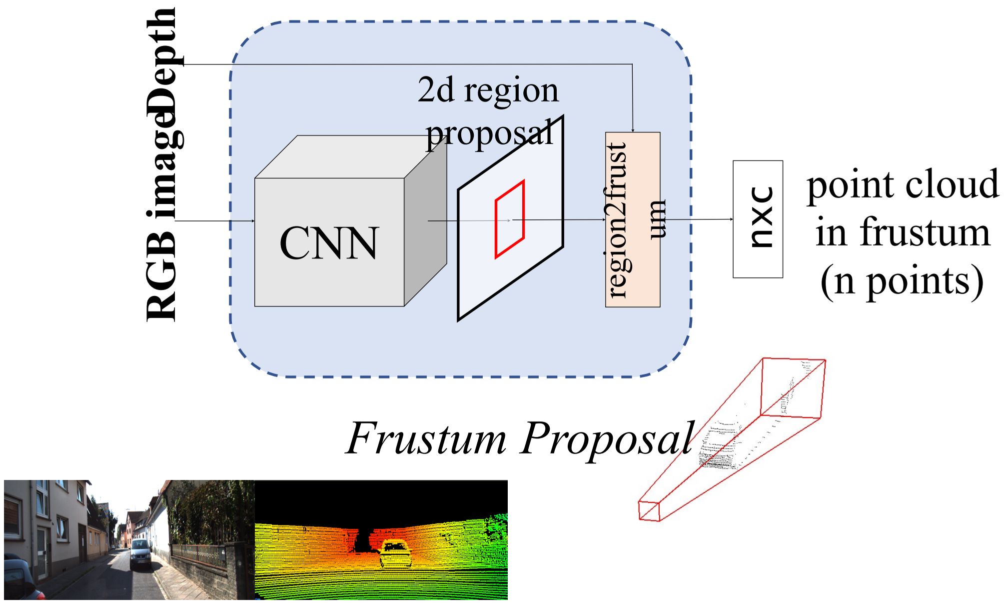


Input: RGB-D data

Image region proposal

2D-3D lifting from depth map

Frustum Proposal



Input: RGB-D data

Image region proposal

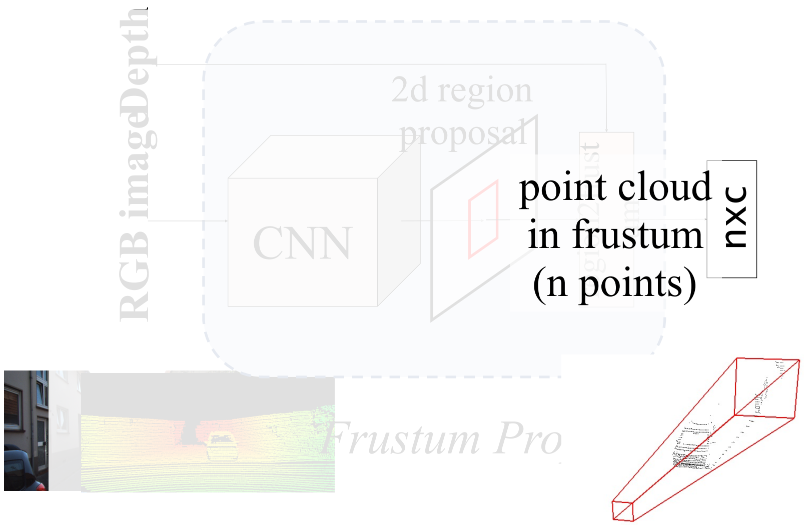
2D-3D lifting from depth map

Frustum point cloud extraction

3D Instance Segmentation in Frustums

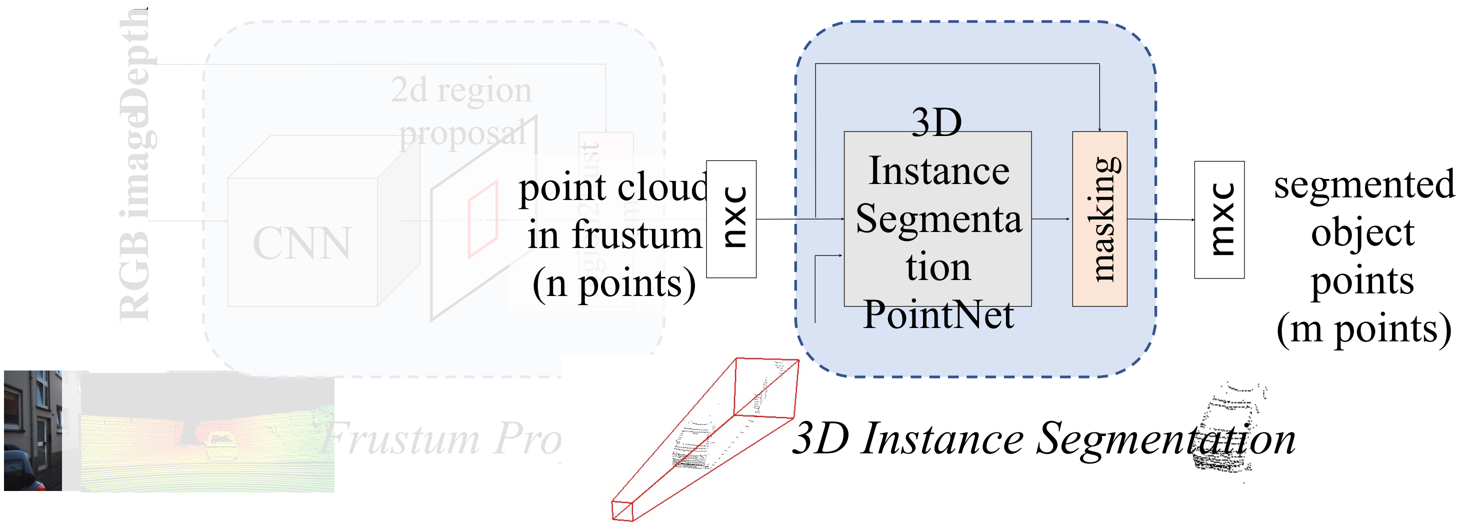
Localize object in frustum by point cloud segmentation.

3D Instance Segmentation in Frustums



Input: frustum point cloud

3D Instance Segmentation in Frustums



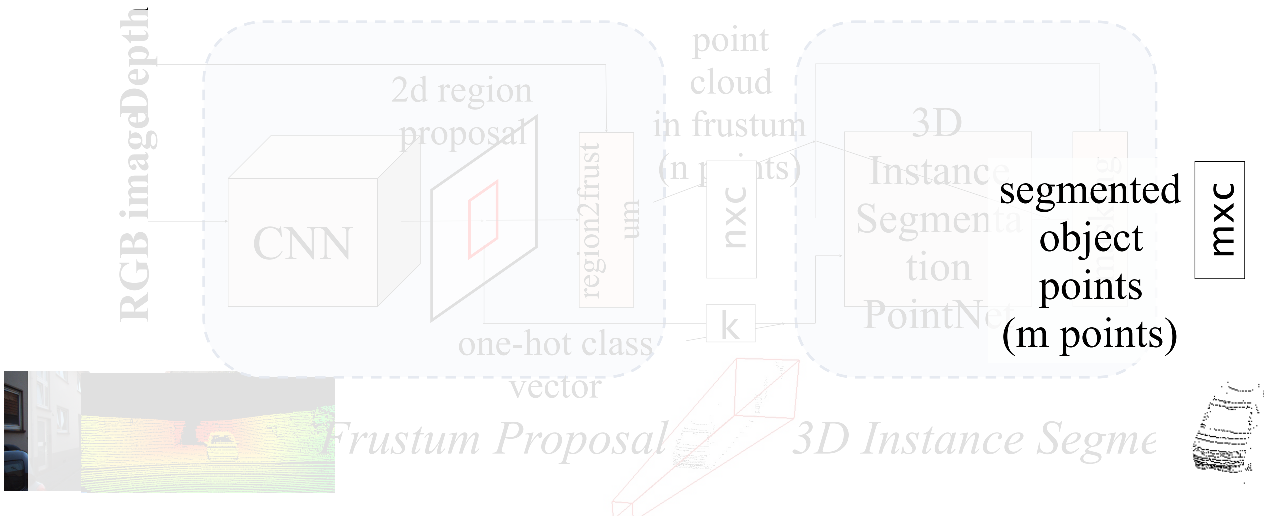
Input: frustum point cloud

Point cloud binary segmentation with PointNet: object of interest v.s. others

Amodal 3D Box Estimation

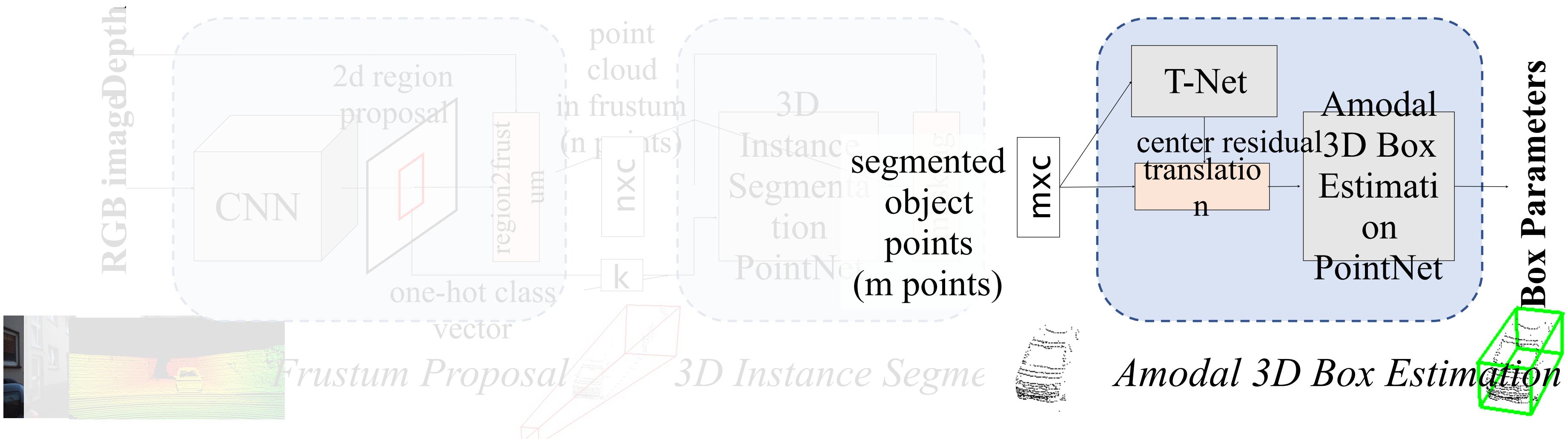
Estimate 3D bounding boxes from segmented object point clouds.

Amodal 3D Box Estimation



Input: object point cloud

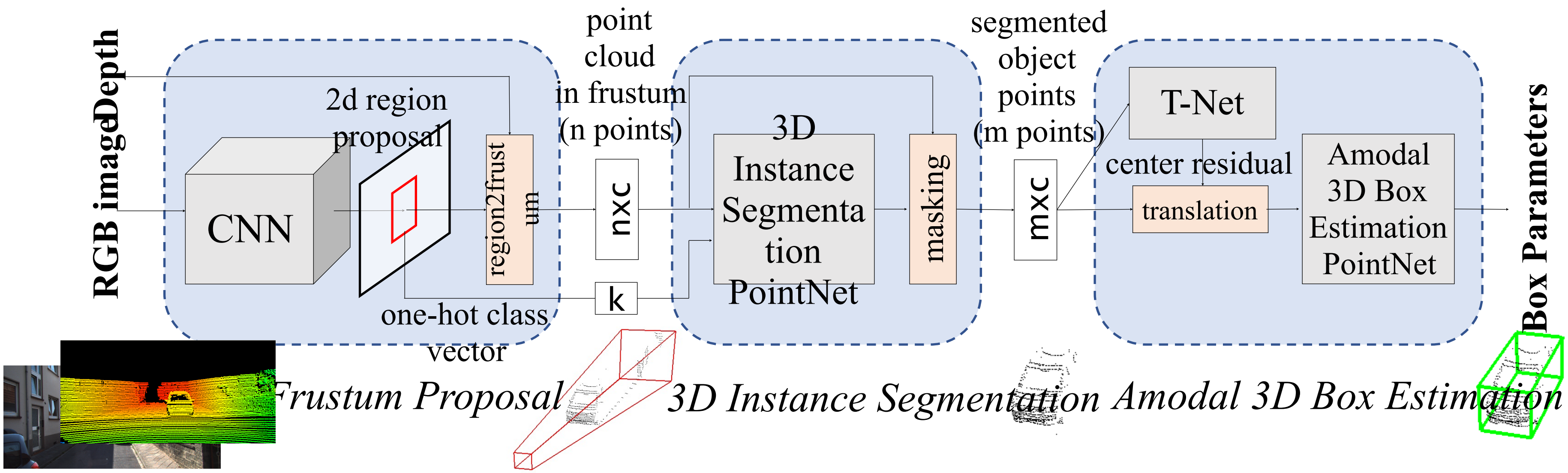
Amodal 3D Box Estimation



Input: object point cloud

A regression PointNet estimates amodal 3D bounding box for the object

Frustum PointNets



In comparison with Mask R-CNN
Mask R-CNN: 2D box -> 2D segmentation
Frustum PointNets: 2D box -> 3D frustum -> 3D segmentation -> 3D amodal box

Frustum PointNets: Key to our Success

- **Representation.** We use PointNets for 3D estimation in raw point clouds.

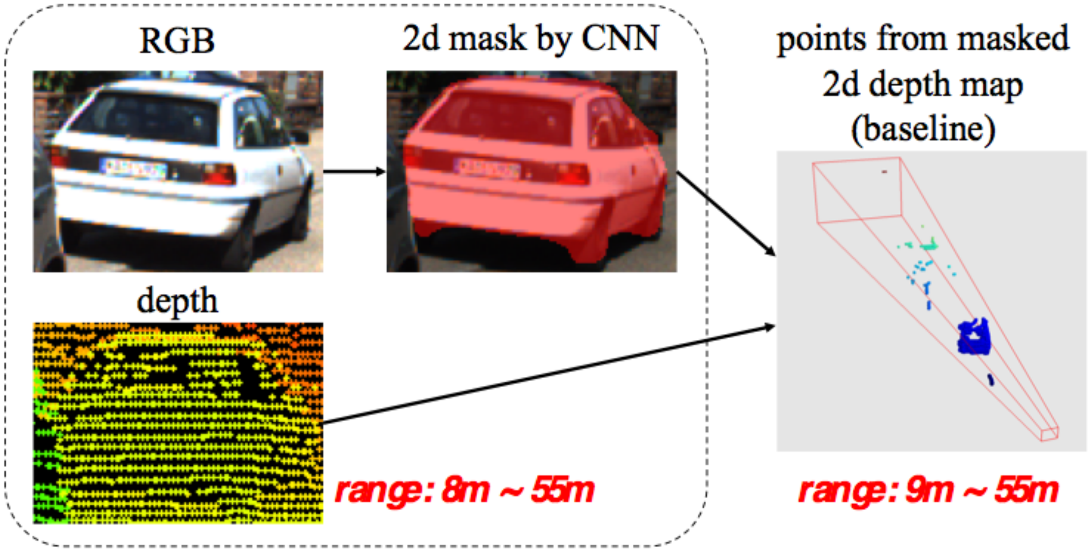
- **Coordinates Normalization.** A series of coordinate transformations canonicalize the learning problems.

- **Loss function.** We design specialized loss functions for 3D bounding box regression.

Frustum PointNets: Key to our Success

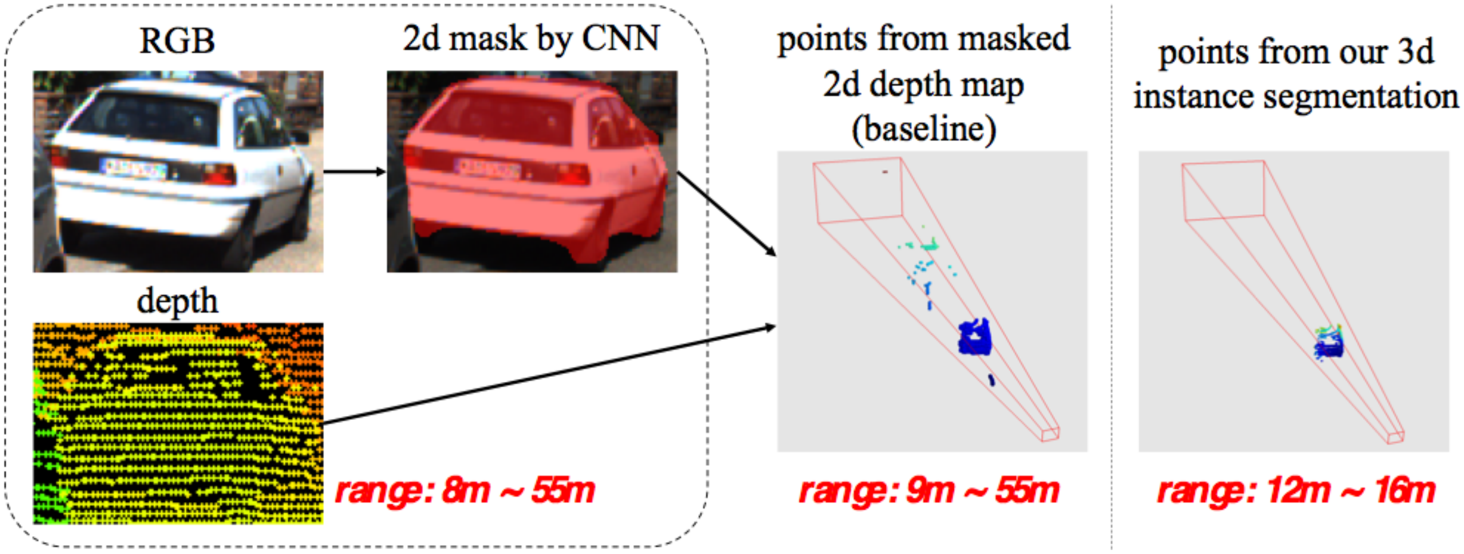
- **Representation.** We use PointNets for 3D estimation in raw point clouds.
- **Coordinates Normalization.** A series of coordinate transformations canonicalize the learning problems.
- **Loss function.** We design specialized loss functions for 3D bounding box regression.

Representation Matters



Baseline by 2D Mask RCNN

Representation Matters

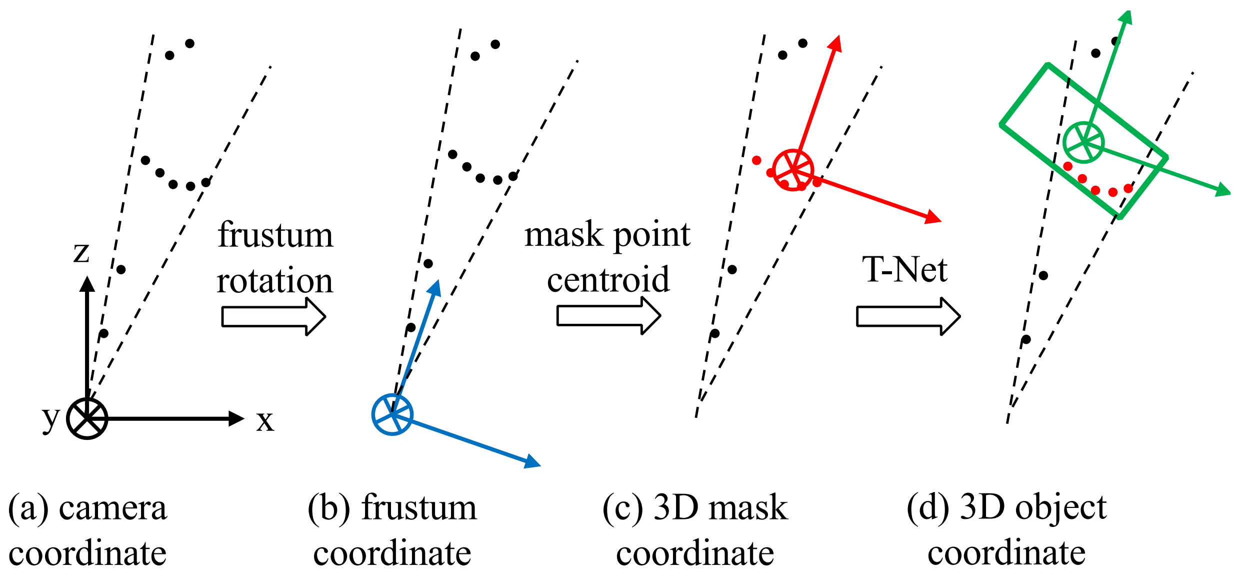


Baseline by 2D Mask RCNN Ours

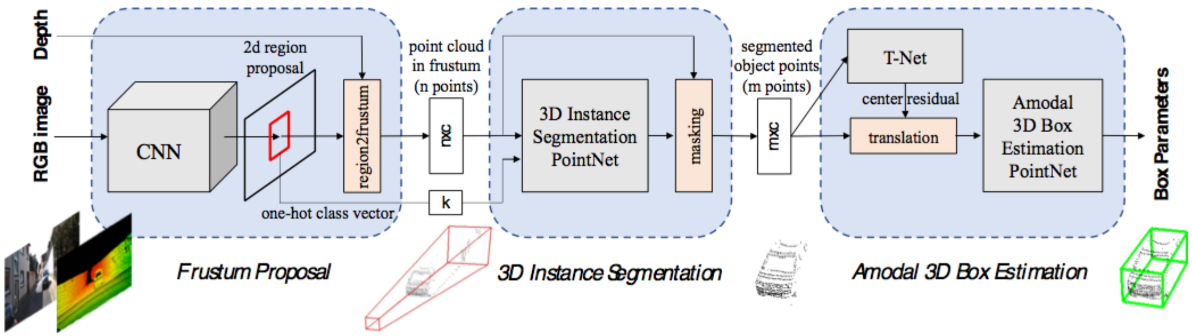
Frustum PointNets: Key to our Success

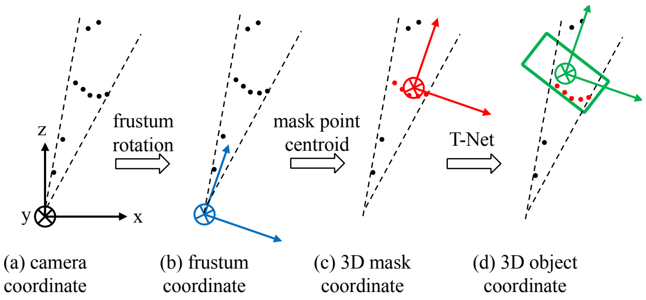
- **Representation.** We use PointNets for 3D estimation in raw point clouds.
- **Coordinates Normalization.** A series of coordinate transformations canonicalize the learning problems.
- **Loss function.** We design specialized loss functions for 3D bounding box regression.

Coordinates Normalization

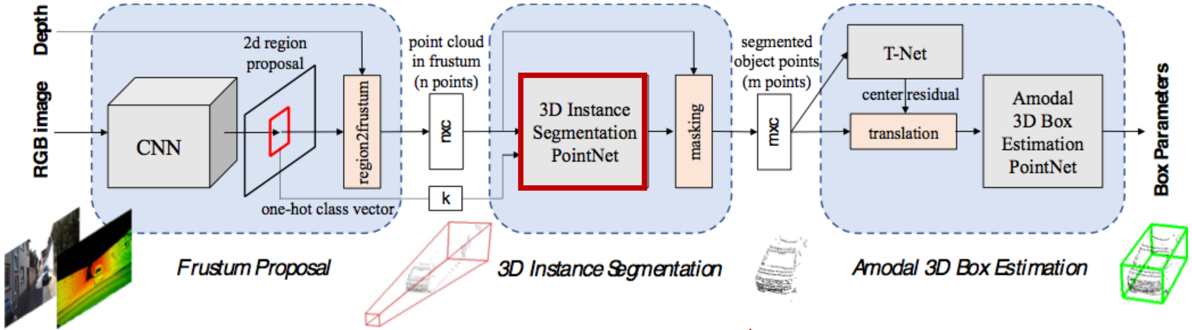


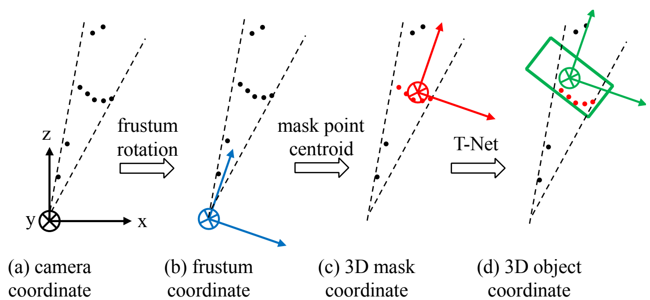
Coordinates Normalization

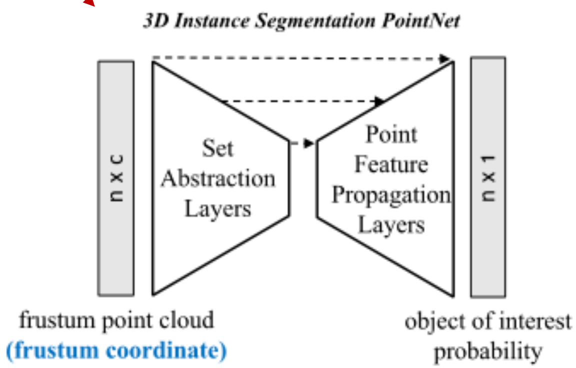




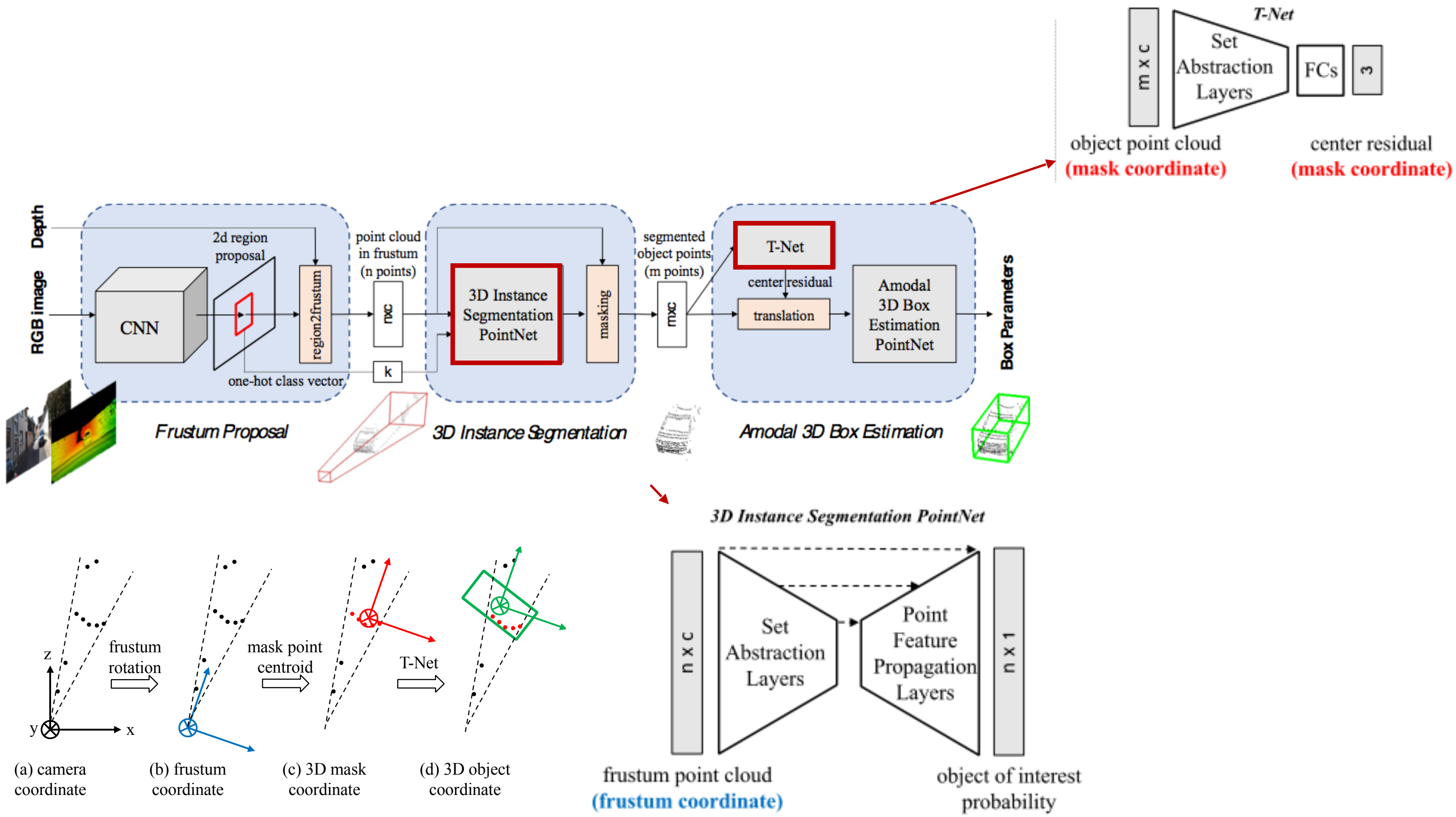
Coordinates Normalization



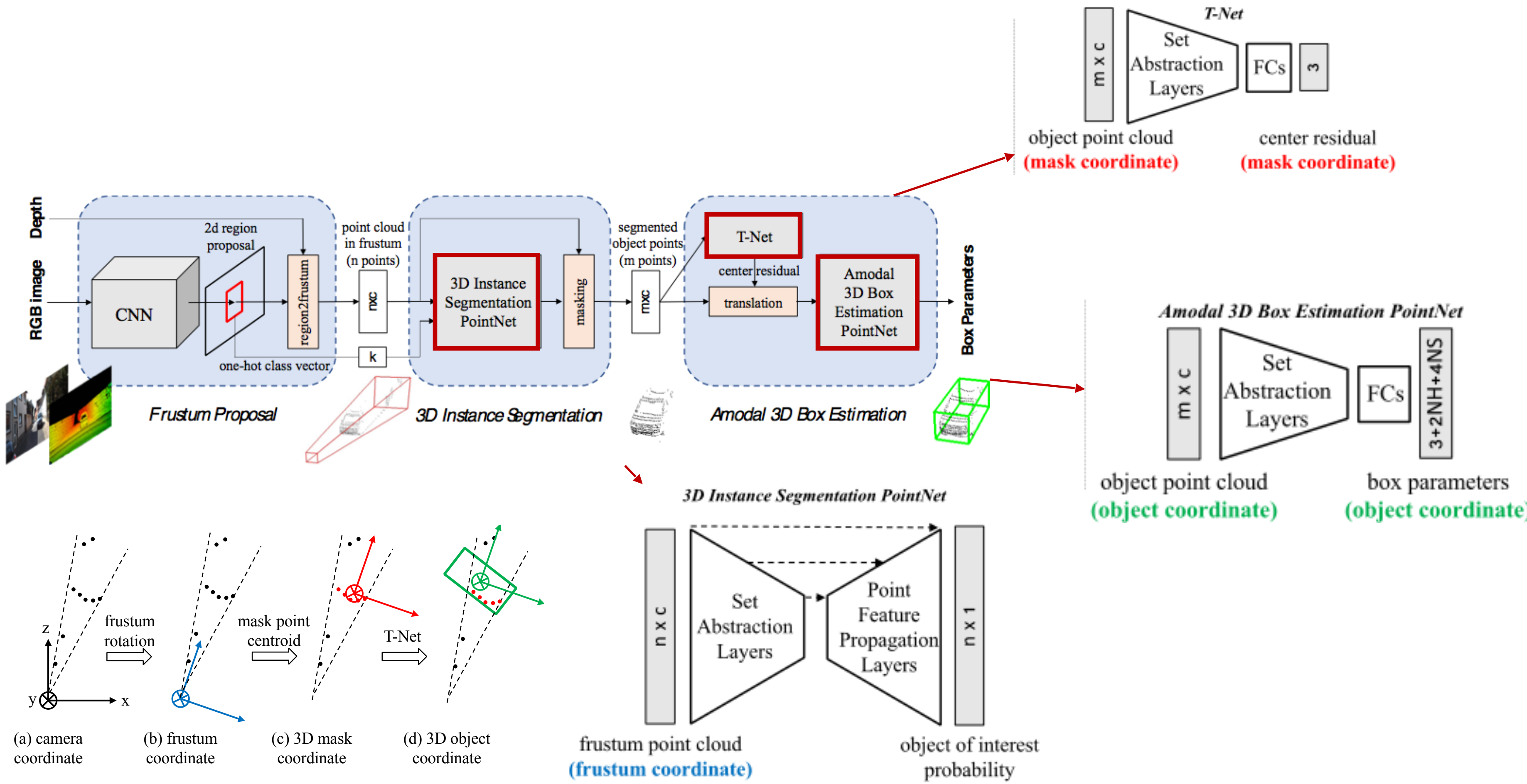




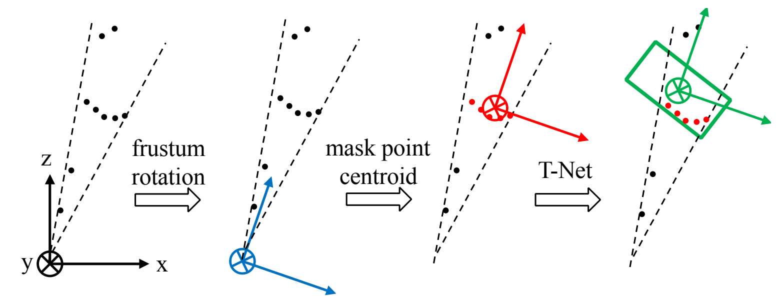
Coordinates Normalization



Coordinates Normalization



Coordinate Normalization

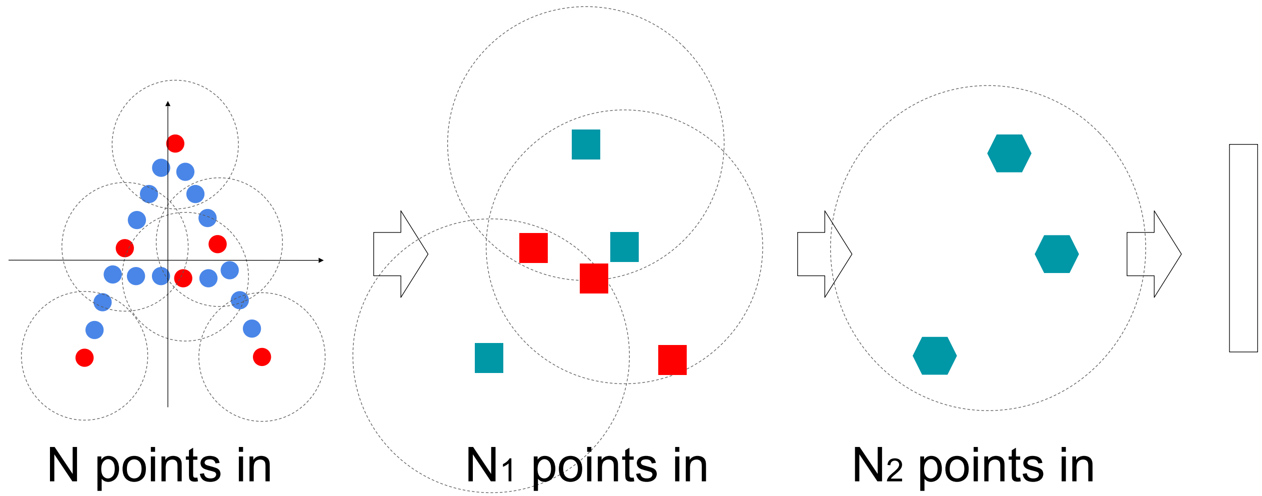


(a) camera coordinate (b) frustum coordinate (c) 3D mask coordinate (d) 3D object coordinate

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

Table 7. **Effects of point cloud normalization.** Metric is 3D box estimation accuracy with IoU=0.7.

PointNet v2.0: Multi-Scale PointNet



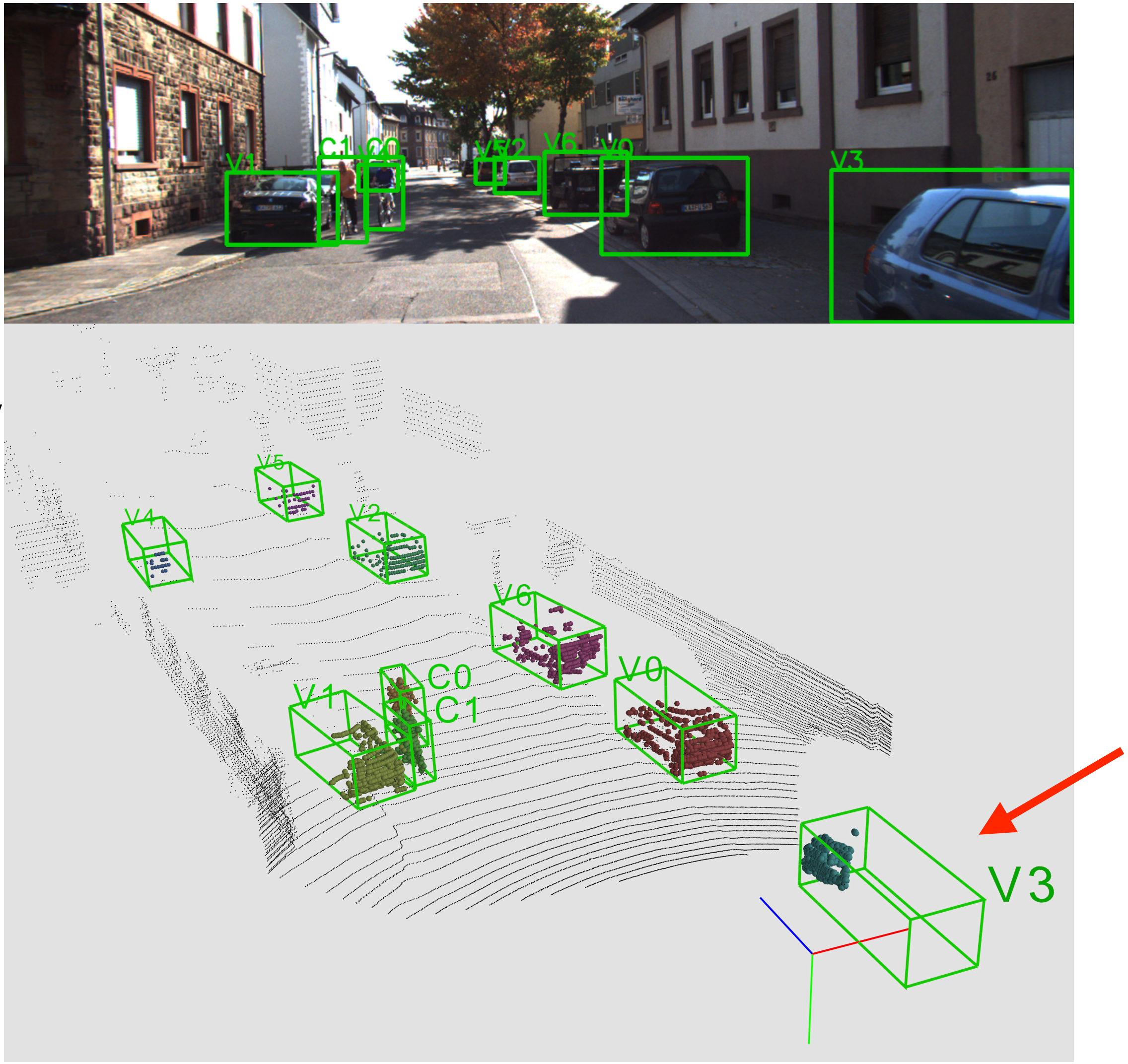
1. Larger receptive field in higher layers ✓
2. Less points in higher layers (more scalable) ✓
3. Weight sharing ✓
4. Translation invariance (local coordinates in local regions) ✓

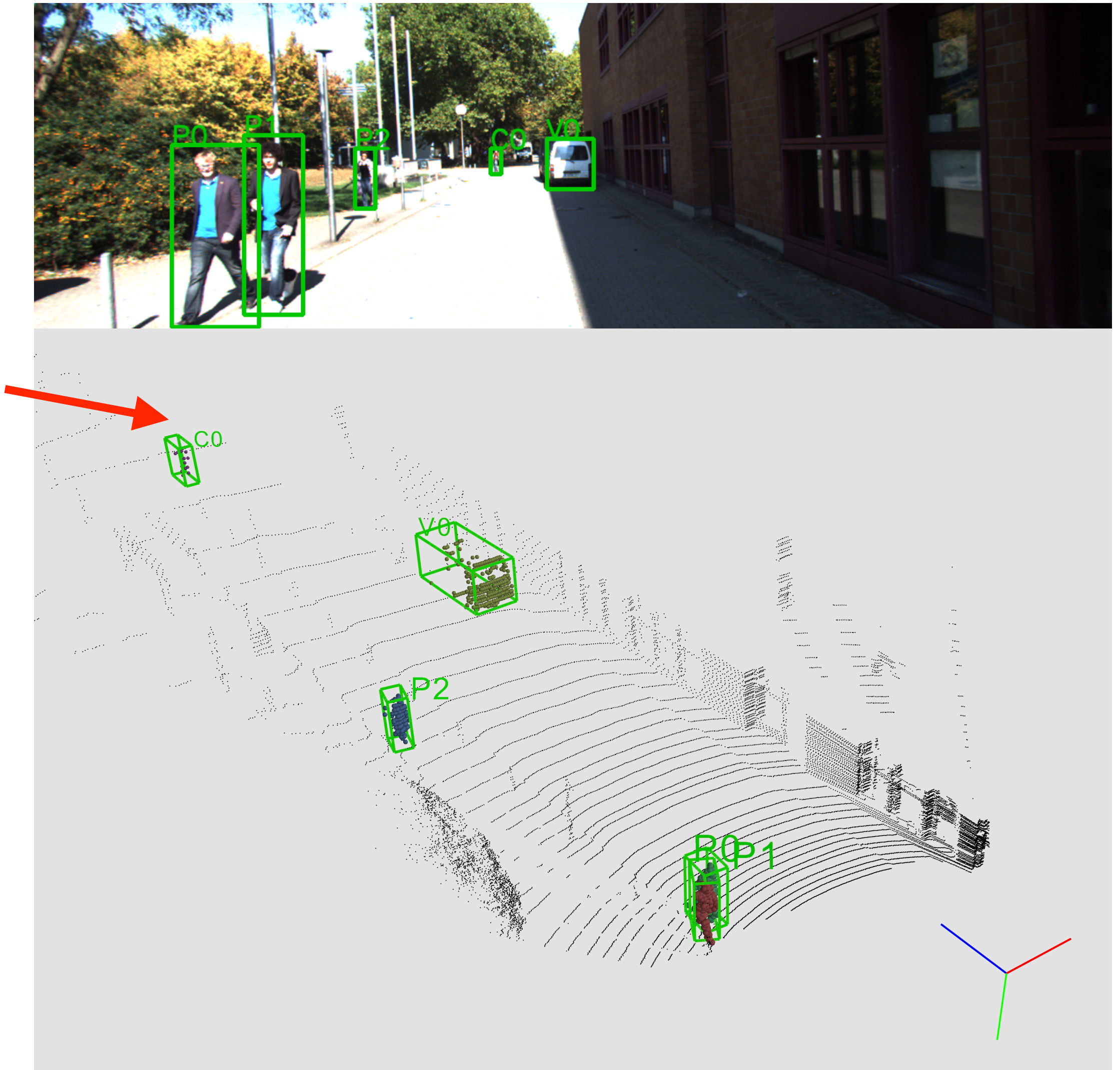
Frustum PointNets: Key to our Success

- **Representation.** We use PointNets for 3D estimation in raw point clouds.
- **Coordinates Normalization.** A series of coordinate transformations canonicalize the learning problems.
- **Loss function.** We design specialized loss functions for 3D bounding box regression.

Qualitative Results
(on KITTI and SUN-RGBD)

Remarkable box estimation accuracy even with a dozen of points or with very partial point cloud

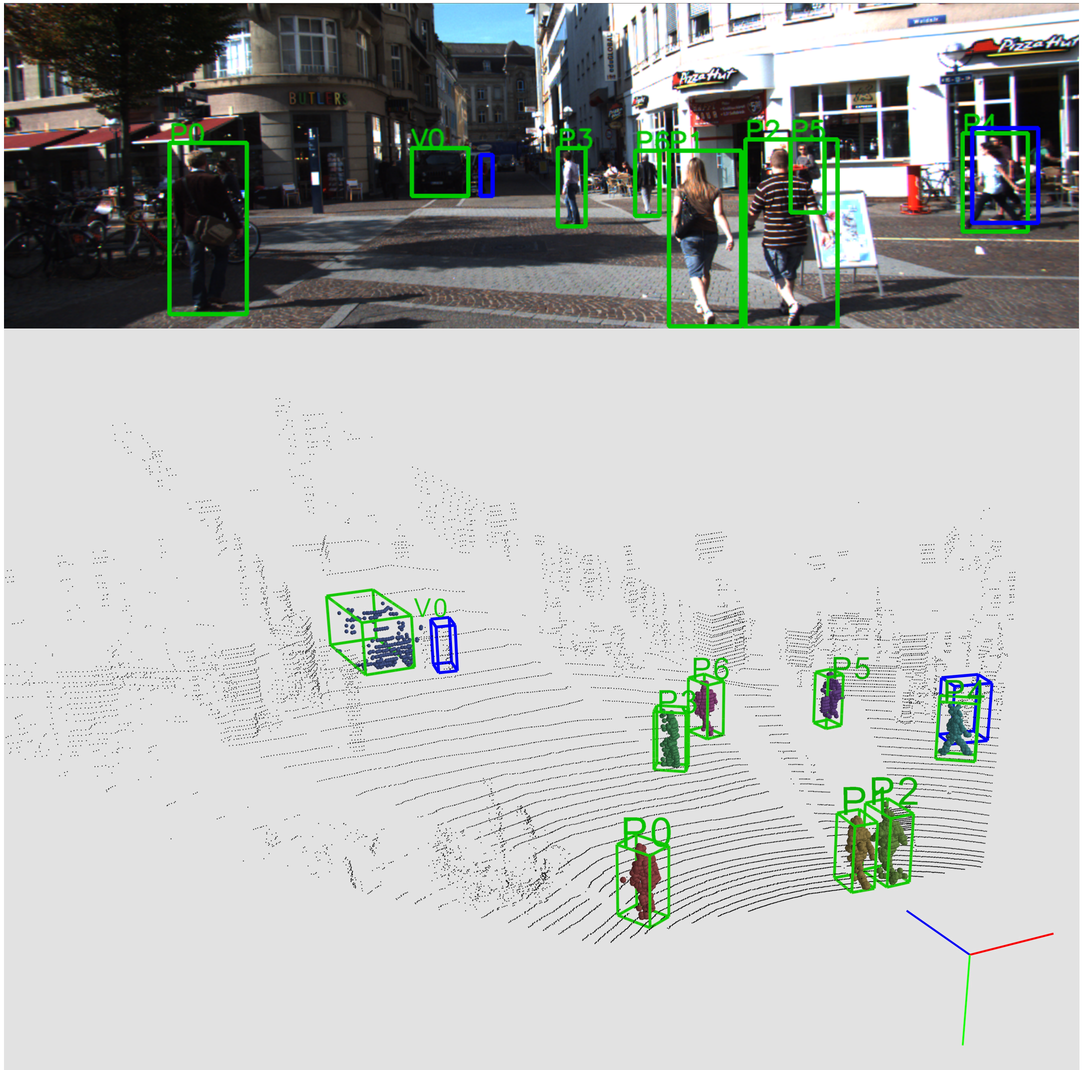



P0
P1
P2
C0
V0
C0
V0
P2
P0
P1


V3
V0
V2
V1
P0
V2
V1
occluding traffic sign..
V3
P0
V0

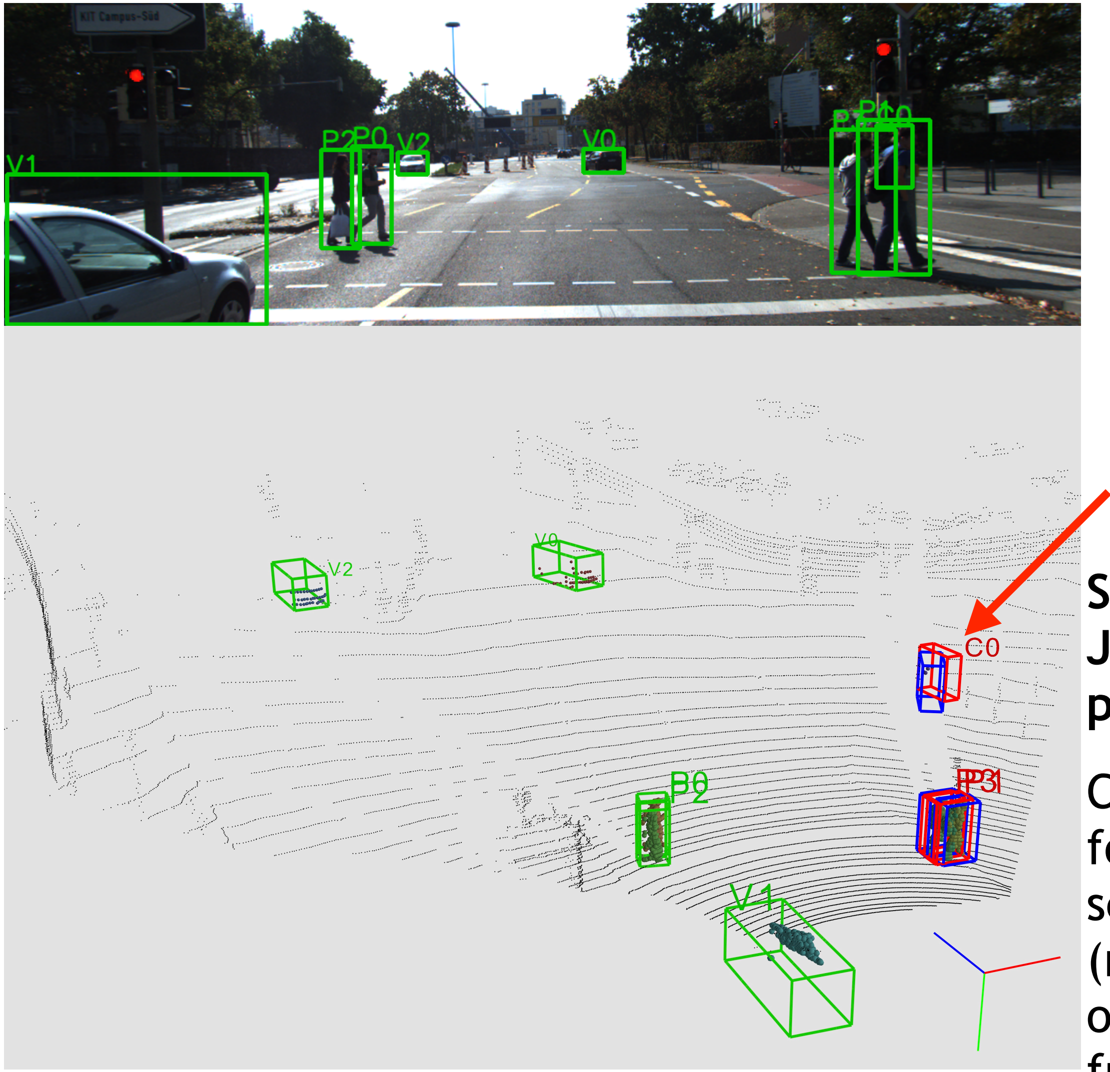
In-accurate box regression with too few LiDAR points

Image features could help.



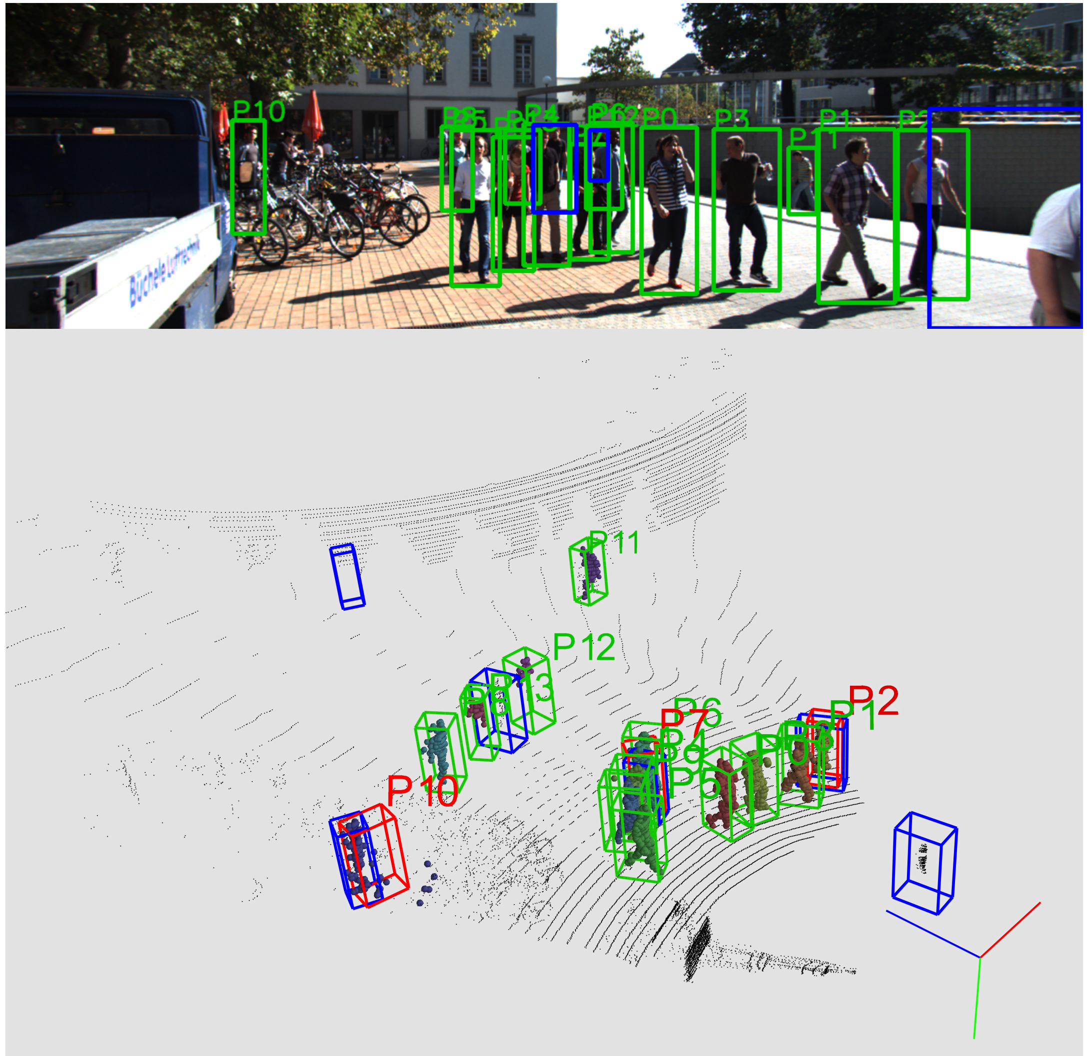
Missing 2D detection results in no 3D detection

Multiple ways for proposal
could help (e.g. bird's eye view, multiple 2D proposal networks)



Strong occlusion. Just 4 LiDAR points..

Challenging case for instance segmentation (multiple closeby objects in a single frustum)



Missed 2D detectio
in a complicated
scene with strong
occlusions

Challenging
segmentation case

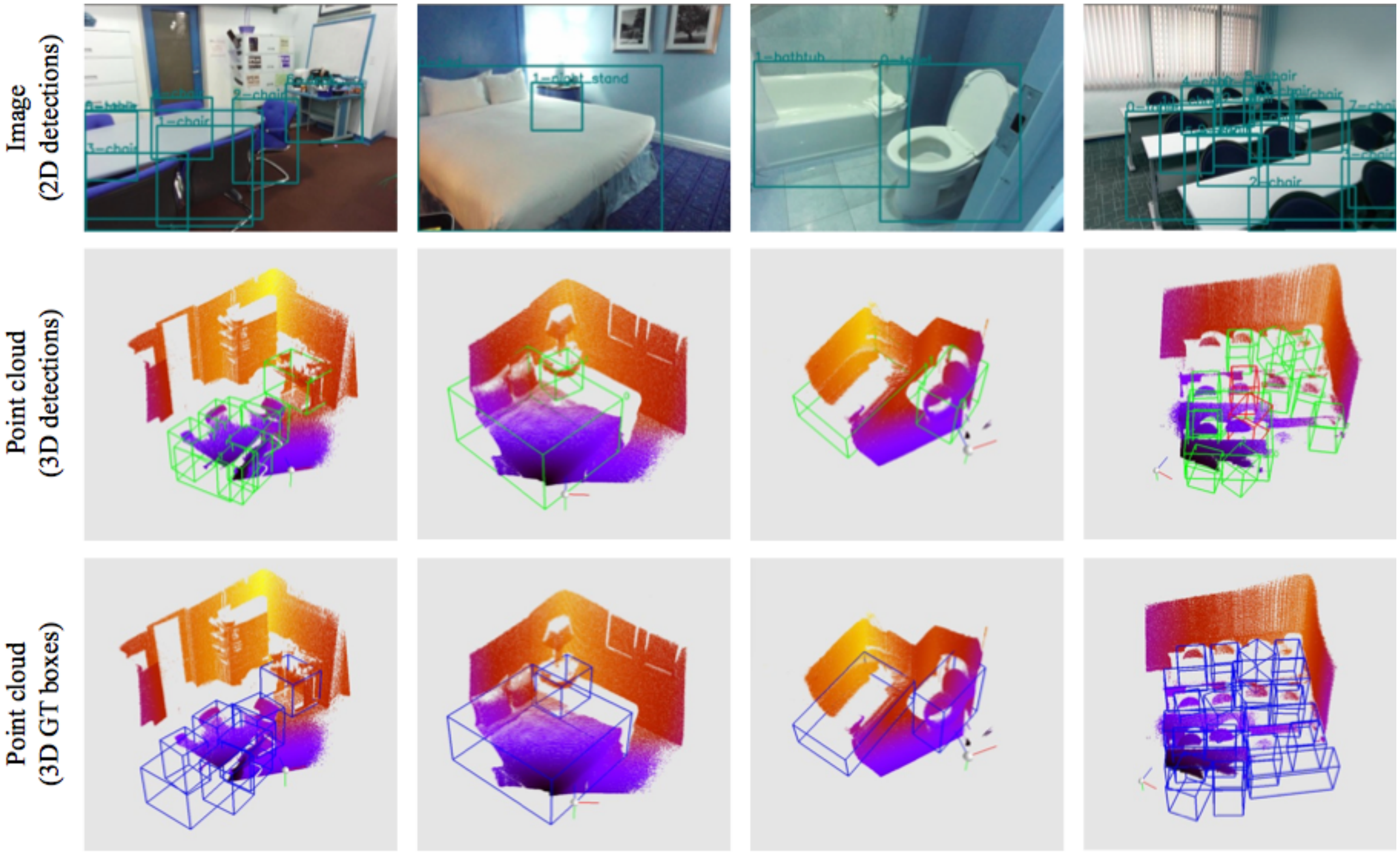


Image
(2D detections)
Point cloud
(3D detections)
Point cloud
(3D GT boxes)
1=night_stand
1=bathtub

	bathtub	bed	bookshelf	chair	desk	dresser	nightstand	sofa	table	toilet	Runtime	mAP
DSS [35]	44.2	78.8	11.9	61.2	20.5	6.4	15.4	53.5	50.3	78.9	19.55s	42.1
COG [30]	**58.3**	63.7	31.8	62.2	**45.2**	15.5	27.4	51.0	**51.3**	70.1	10-30min	47.6
2D-driven [16]	43.5	64.5	31.4	48.3	27.9	25.9	41.9	50.4	37.0	80.4	4.15s	45.1
Ours (v1)	43.3	**81.1**	**33.3**	**64.2**	24.7	**32.0**	**58.1**	**61.1**	51.1	**90.9**	0.12s	**54.0**

Table 5. **3D object detection AP on SUN-RGBD val set.** Evaluation metric is average precision with 3D IoU threshold 0.25 as proposed by [33]. Note that both COG [30] and 2D-driven [16] use room layout context to boost performance while ours and DSS [35] not. Compared with previous state-of-the-arts our method is 6.4% to 11.9% better in mAP as well as one to three orders of magnitude faster.

Opening in my Lab for Shape Processing

- Task: to make ShapeNet amiable for machine learning researchers (ShapeNet v2.0)
- You will gain a lot of experience for geometry processing
- Not much research into machine learning in the beginning, though, but
  - Can attend my group meetings
  - May have the opportunity to work on learning stuff in the future
  - Acknowledged as in the ShapeNet team
- Requirement:
  - Very strong programming ability
  - Past CG experience
  - Master thesis topic