APPLIED SPATIAL SENSING AND REASONING

CASE STUDIES

Dr TIAN Jing

tianjing@nus.edu.sg

Agenda

- Point cloud data classifications
- Other applications

Point cloud

- The point cloud data can be represented as a numpy array with N rows and 3 columns. Each row corresponds to a single point, which is represented using at least 3 values for its position in space (x, y, z).

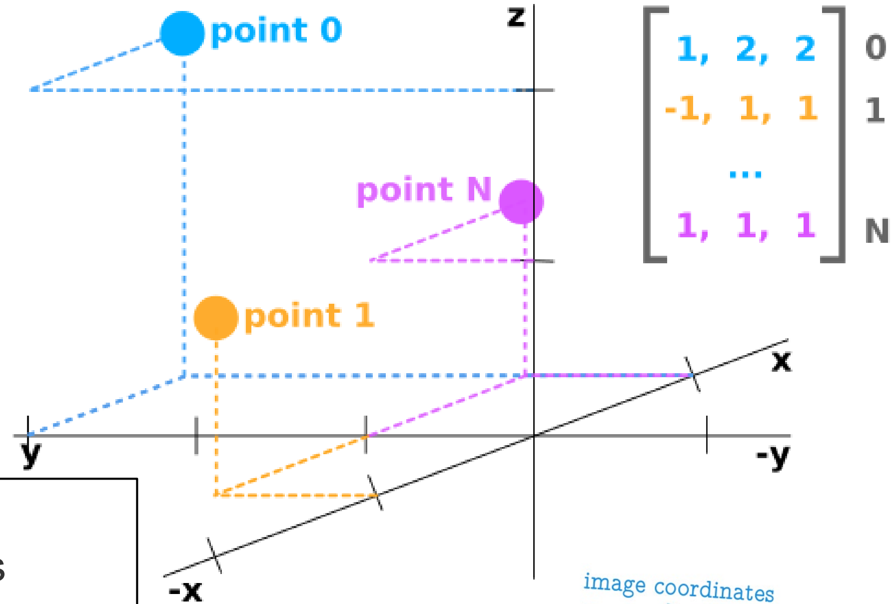
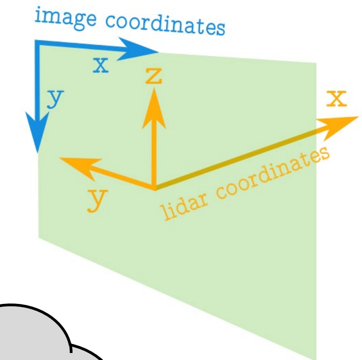


Image
- The coordinate values in an image are always positive.
- The origin is located on the upper left hand corner.
- The coordinates are integer values.

Point cloud
- The coordinate values in point cloud can be positive or negative.
- The coordinates can take on real numbered values.
- The positive x axis represents forward.
- The positive y axis represents left.
- The positive z axis represents up.



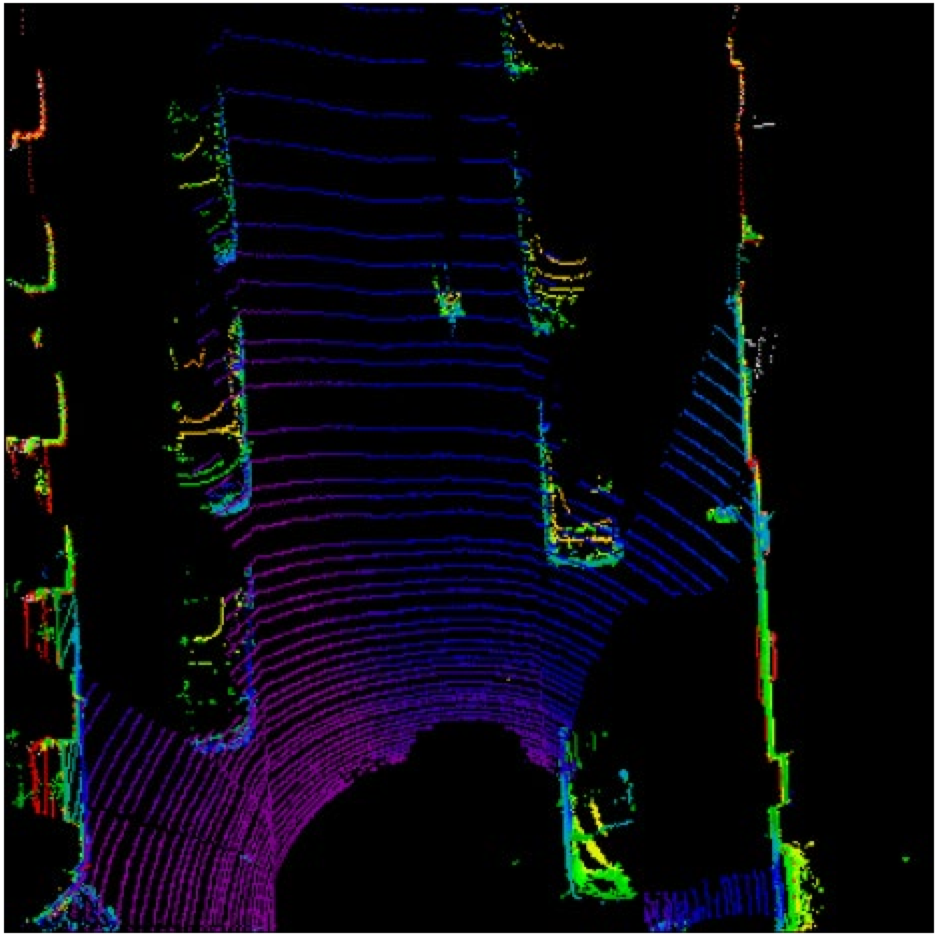
Careful: refer specification of your LiDAR sensor

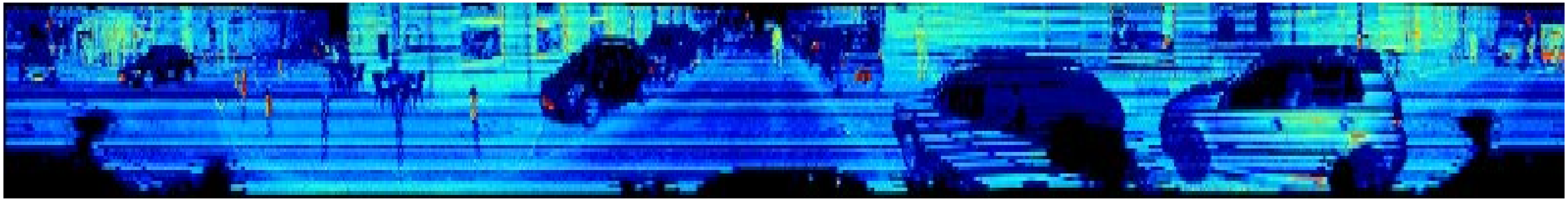
Reference: http://ronny.rest/tutorials/module/pointclouds_01/point_cloud_data/

Point cloud

- Bird overview view
- Panoramic view

RPLIDAR A1
http://www.slamtec.com/en/lidar/a1

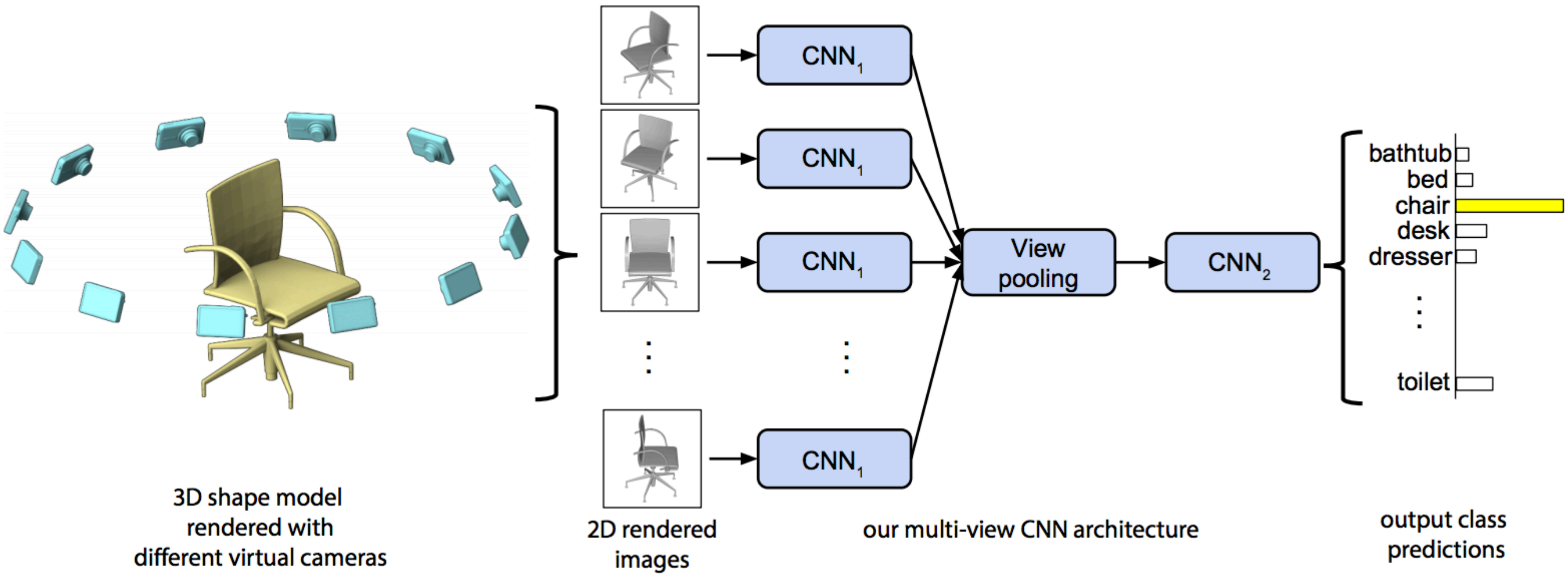




Reference: http://ronny.rest/tutorials/module/pointclouds_01/point_cloud_panoramic360/

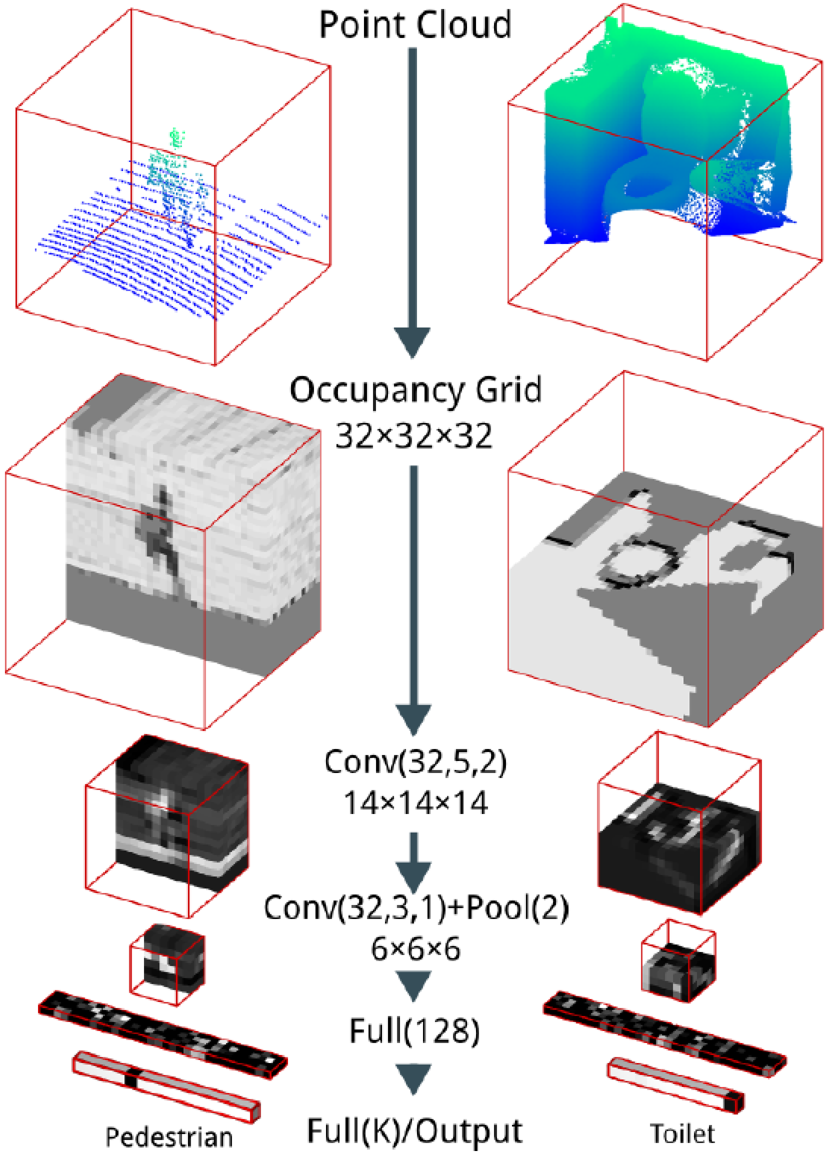
MVCNN

- First CNN: Extract image features from individual view
- View pooling: Element-wise max-pooling across all views.
- Second CNN: Extract shape features from pooled image



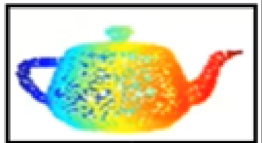
Reference: H. Su, S. Maji, E. Kalogerakis, E. Learned-Miller, "Multi-view Convolutional Neural Networks for 3D Shape Recognition," *IEEE Inter. Conf. on Computer Vision* (ICCV), Santiago, Chile, Dec. 2015, pp. 945-953, http://vis-www.cs.umass.edu/mvcnn/

VoxNet



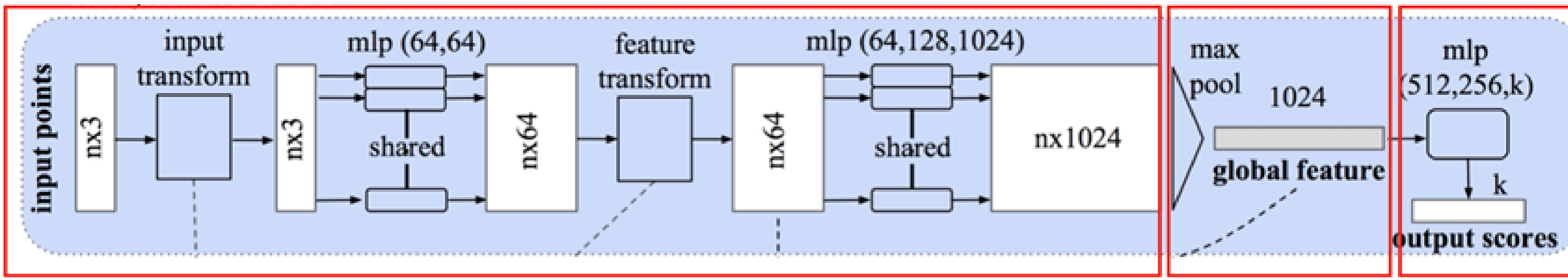
Reference: D. Maturana and S. Scherer, "VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition," IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), Hamburg, Germany, Oct. 2015, pp. 922-928.

PointNet



It uses a shared *multi-layer perceptron* (MLP) to map each of the n points from 3D (x, y, z) to 64 dimensions (i.e., mapping is identical on the n points). It is repeated to map the n points from 64 dimensions to 1024 dimensions. Max pooling is used to create a global feature vector. Finally, a three-layer fully-connected network is used to map the global feature vector to k output classification scores.

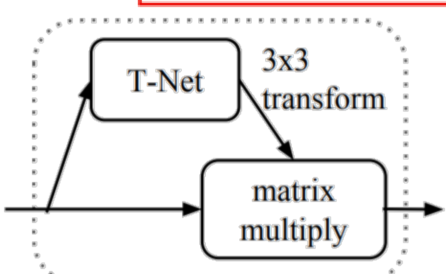
Teapot

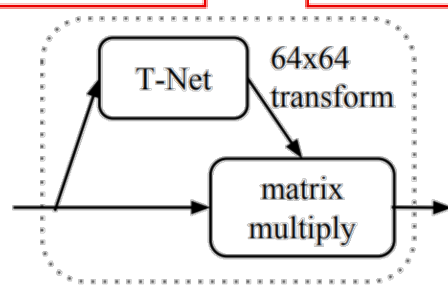


Non-linear transformations

Aggregates point features

Classification





Reference: R. Qi, H. Su, M. Kaichun, L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," CVPR 2017, pp. 77-85.

PointNet (1)

Input point cloud data

$n \times 3$

0.224, 0.325, 0.682

0.129, 0.262, 0.126

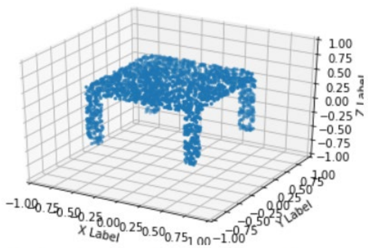
0.287, -0.024, 0.135

-0.331, 0.206, 0.554

…, …, …

Problem statement:
To perform classification based on a set of points.

Classification result: Table



Reference

- [The author's presentation in CVPR 2017], https://www.youtube.com/watch?v=Cge-hot0Oc0
- [A Chinese tutorial video with English slides by the PointNet author in Bilibili], https://www.bilibili.com/video/BV1As411377S?from=search&seid=18233926180086373014
- Point cloud classification with PointNet, https://keras.io/examples/vision/pointnet/, created on May 2020.

PointNet (2)

Input point cloud data

$n \times 3$

0.224, 0.325, 0.682

0.129, 0.262, 0.126

0.287, -0.024, 0.135

-0.331, 0.206, 0.554

…, …, …

Feature extraction (to be designed in following slides)

MLP

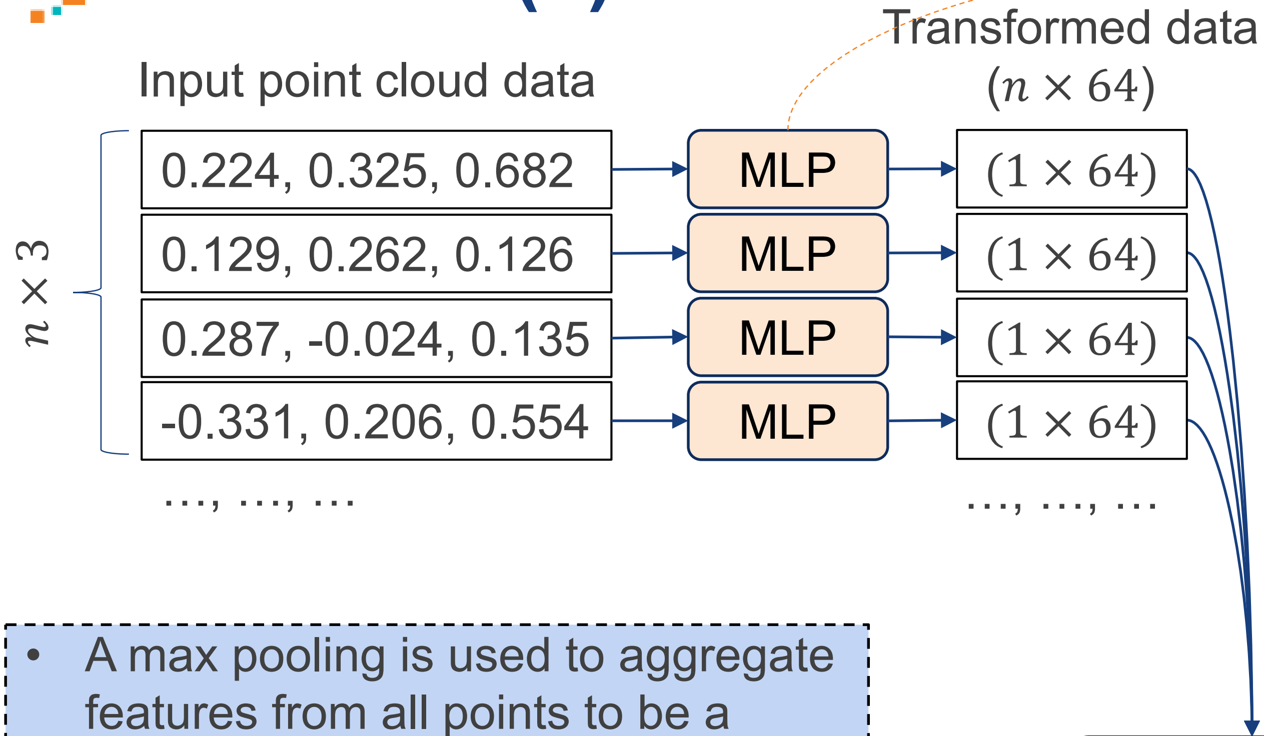
- A MLP is used for classification (as what we do in fully-connected layers in the CNN model).

Classification Loss (cross entropy loss)

One-hot vector

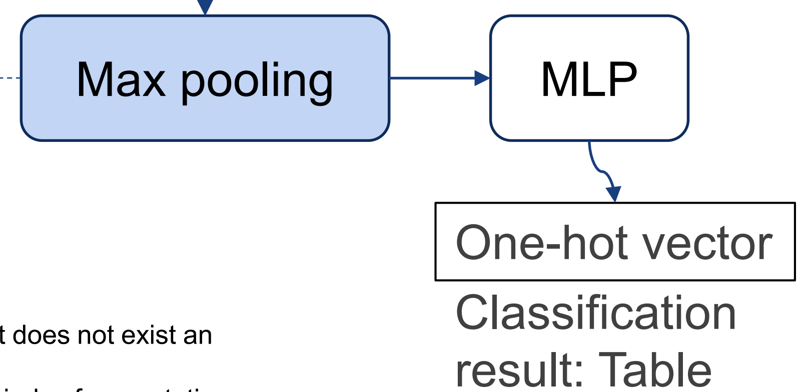
Classification result: Table

PointNet (3)



- A MLP $(3, 64)$ is applied on each point to transform point to higher dimension space (recall what we have learned in ISSM).
- The MLP is sharable so it doesn't depend on the order of points.
- The # of nodes $(3, 64)$ in MLP is selected in experiments.

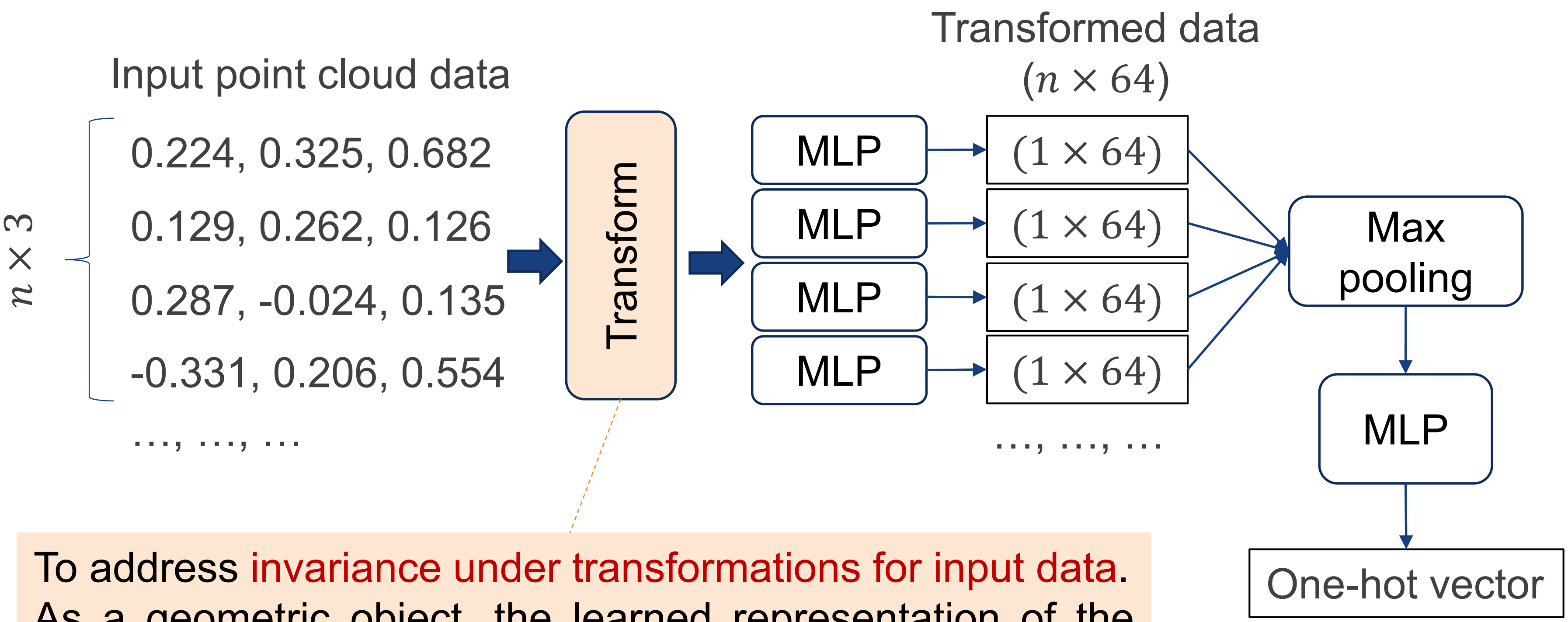
- A max pooling is used to aggregate features from all points to be a global feature for further classification.
- The max pooling function doesn't depend on the order of points.



Discussions: Why not other ideas? (quoted from the original paper)

- Sort inputs
- →While sorting sounds like a simple solution, in high dimensional space there in fact does not exist an ordering that is stable w.r.t. point perturbations in the general sense.
- Treat the input as a sequence to train an RNN, but augment the training data by all kinds of permutations.
- → While RNN has relatively good robustness to input ordering for sequences with small length (dozens), it's hard to scale to thousands of input elements, which is the common size for point sets.

PointNet (4)

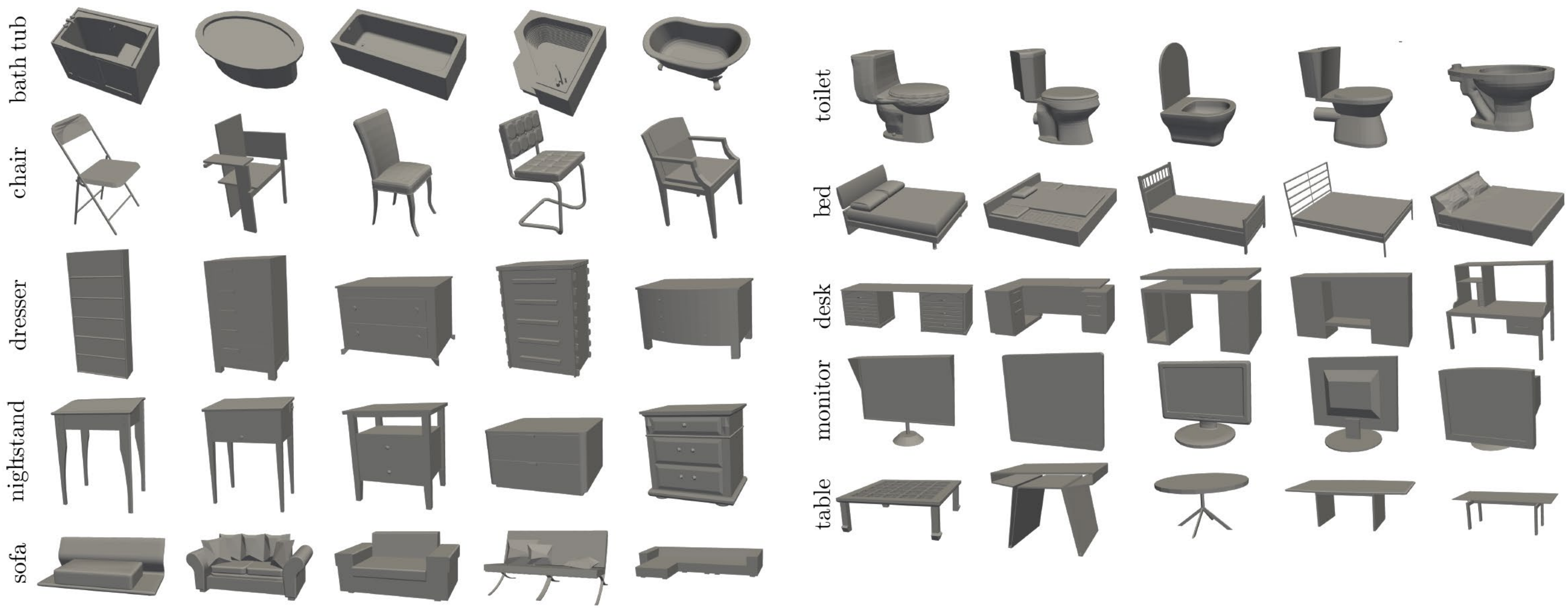


To address invariance under transformations for input data. As a geometric object, the learned representation of the point set should be invariant to certain transformations. For example, rotating and translating points all together should not modify the point cloud category of the points.

Classification result: Table

Example

- Objective: Point cloud classification
- Dataset: ModelNet10, http://modelnet.cs.princeton.edu/

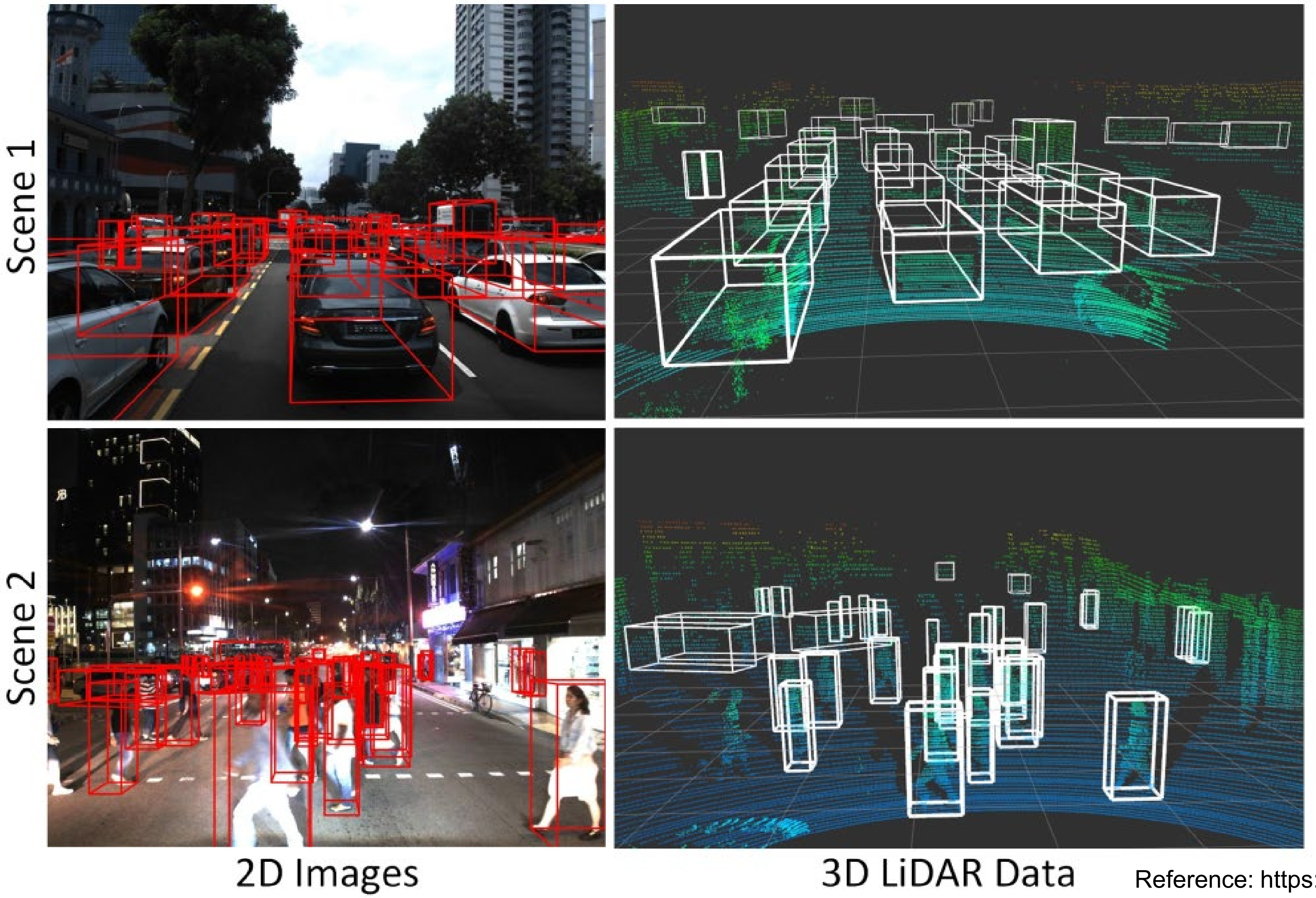


Agenda

- Point cloud data classifications
- Other applications

RGB-LiDAR dataset

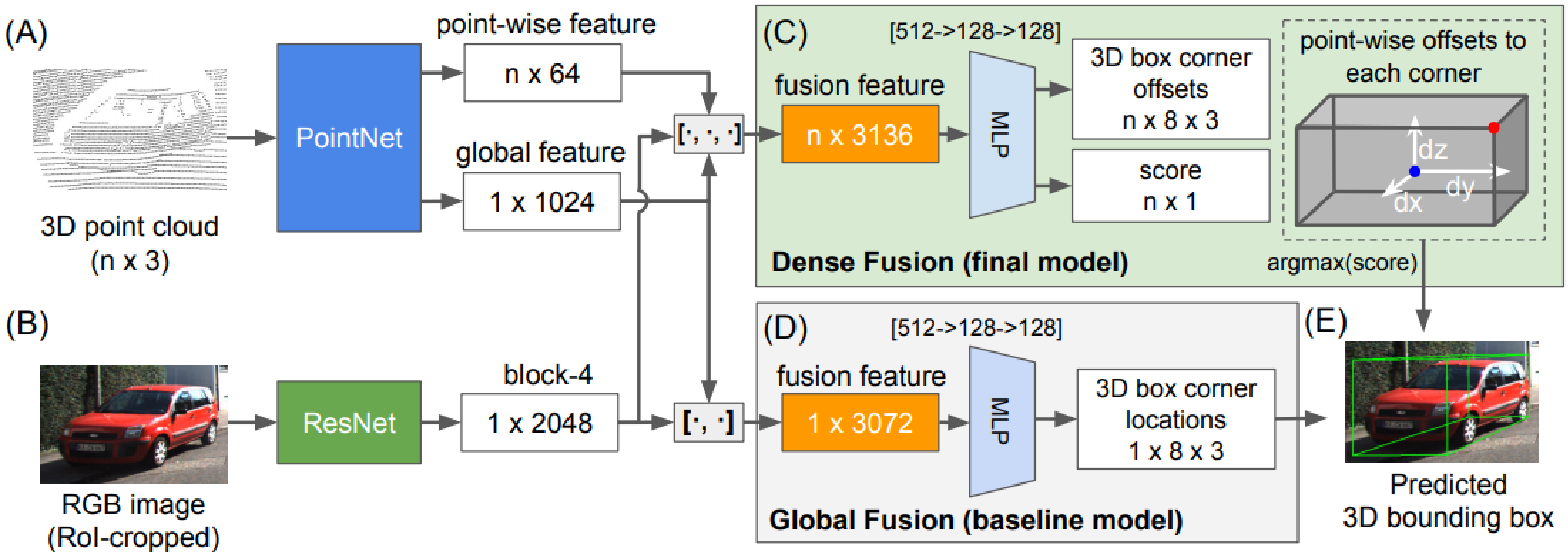
- 230K human-labeled 3D object annotations in 39,179 LiDAR point cloud frames and corresponding frontal-facing RGB images.
- Captured at different times (day, night) and weathers (sun, cloud, rain).



Reference: https://github.com/I2RDL2/ASTAR-3D

Point Fusion

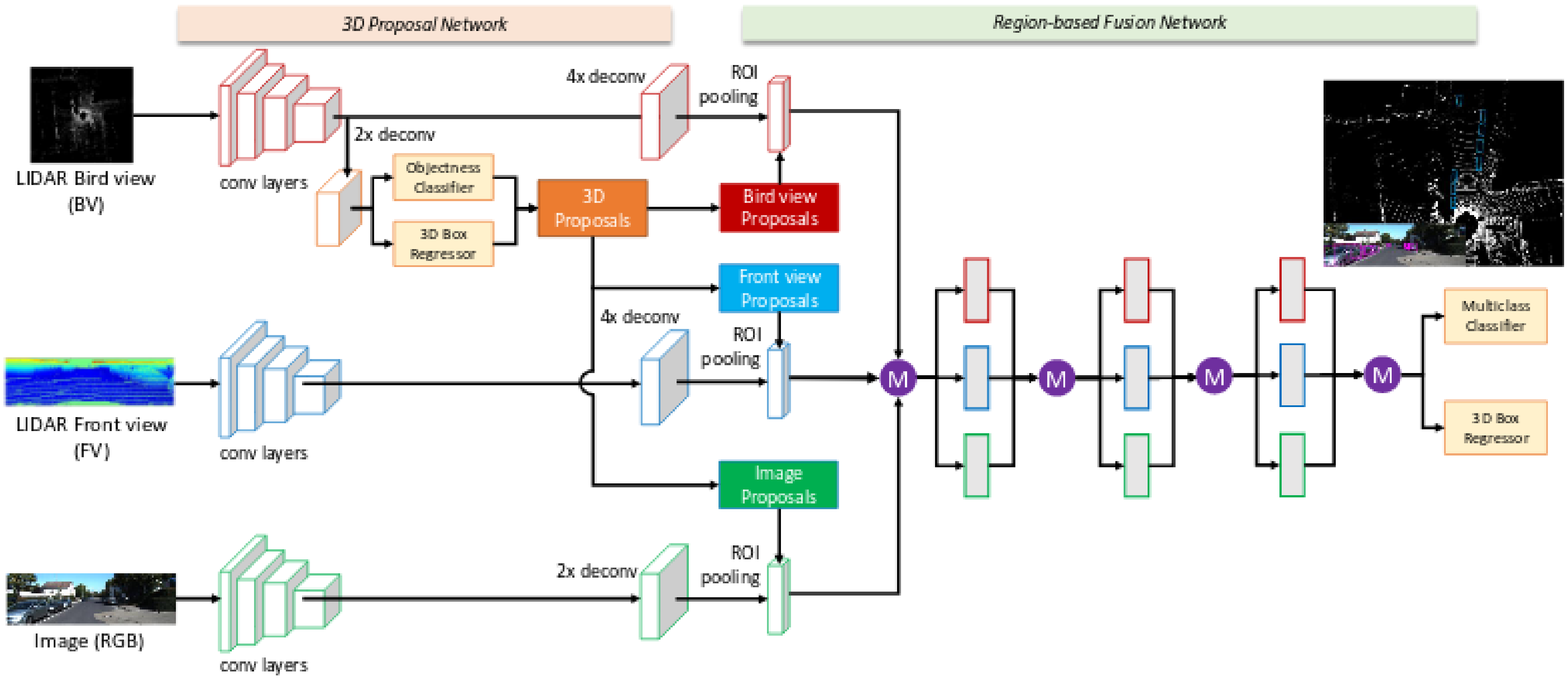
PointFusion has two feature extractors: a PointNet variant that processes raw point cloud data, and a CNN that extracts visual features from an input image. We present two fusion network formulations: a vanilla global architecture that directly regresses the box corner locations (D), and a novel dense architecture that predicts the spatial offset of each of the 8 corners relative to each input point, as illustrated in (C): for each input point, the network predicts the spatial offset (white arrows) from a corner (red dot) to the input point (blue), and selects the prediction with the highest score as the final prediction (E).



Reference: D. Xu, et al., PointFusion: Deep Sensor Fusion for 3D Bounding Box Estimation, https://arxiv.org/abs/1711.10871

MV3D

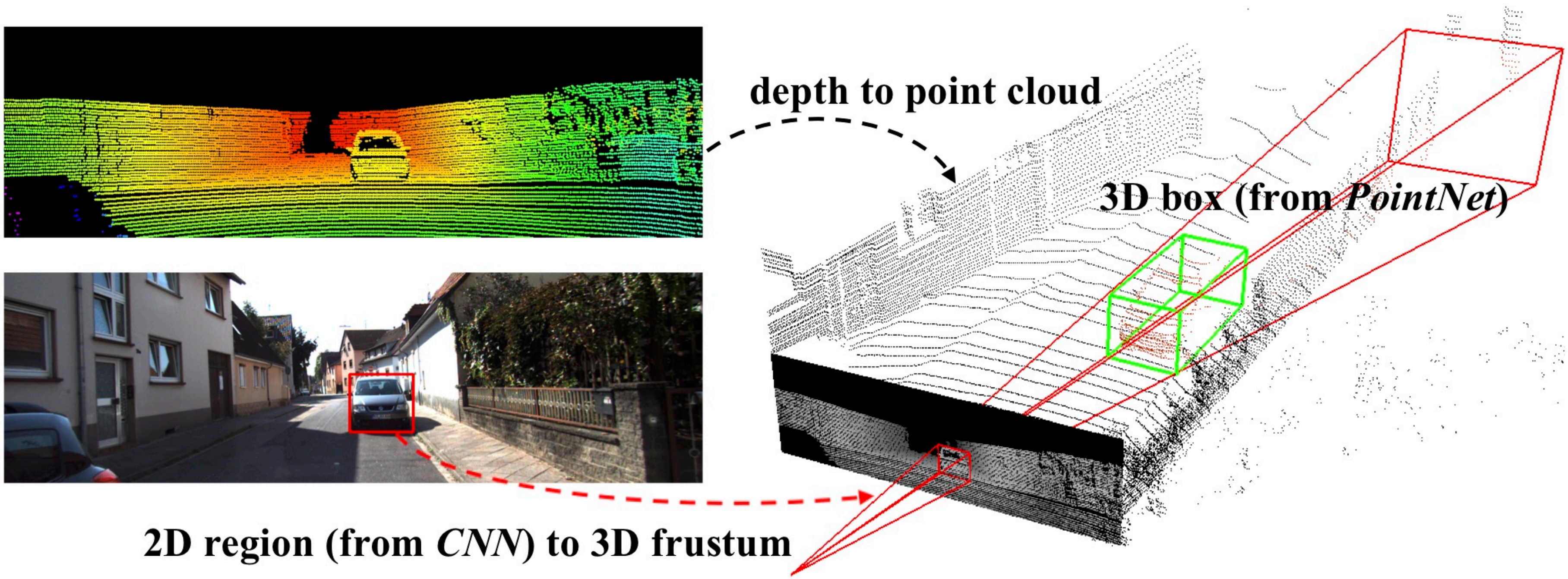
Multi-view V3D (MV3D) simply takes two separate 2D views of the point cloud: one from the front and one from the top (birds' eye). MV3D also uses the 2D camera image associated with each LiDAR scan. That provides three separate 2D images (LiDAR front view, LiDAR top view, camera front view).



Reference: X. Chen, H. Ma, J. Wan, B. Li, T. Xia, "Multi-View 3D Object Detection Network for Autonomous Driving," https://arxiv.org/abs/1611.07759

Frustum PointNet

In our pipeline, we firstly build object proposals with a 2D detector running on RGB images, where each 2D bounding box defines a 3D frustum region. Then based on 3D point clouds in those frustum regions, we achieve 3D instance segmentation and amodal 3D bounding box estimation, using PointNet network.



Reference: C. Qi, W. Liu, C. Wu, H. Su, L. Guibas, "Frustum PointNets for 3D Object Detection from RGB-D Data,“ https://github.com/charlesq34/frustum-pointnets

Thank you!

Dr TIAN Jing
Email: tianjing@nus.edu.sg