Machine Perception

Lecture 4: Range and depth sensing (Part II)

Piotr Skrzypczyński
Institute of Robotics and Machine Intelligence
Poznań University of Technology

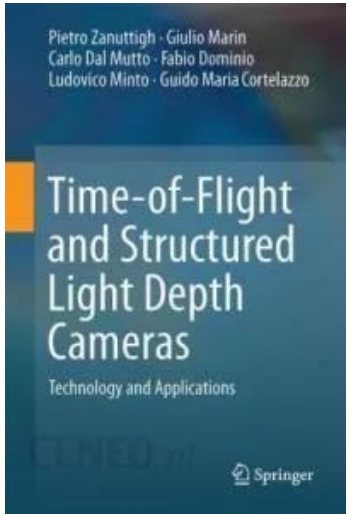
Lecture outline

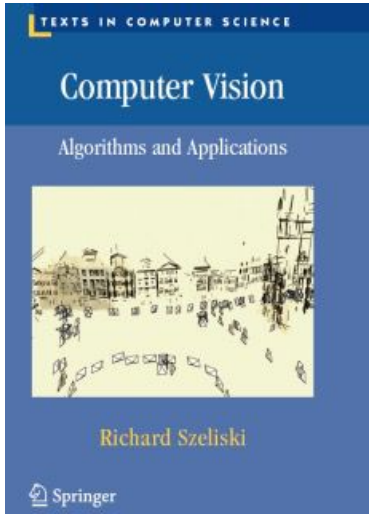
- Scene representations: depth images, point clouds and other.
- Processing of point clouds: Point Cloud Library, Iterative Closest Points as registration algorithm.
- Point clouds in machine learning, PointNet/PointNet++, voxel representations.

Literature

1. P. Zanuttigh , G. Marin , C. Dal Mutto , F. Dominio , L. Minto , G. Cortelazzo, Time-of-Flight and Structured Light Depth Cameras, Springer, 2016.

2. R. Szeliski, Computer Vision, Algorithms and Applications, 2nd edition, Spronger, 2022.

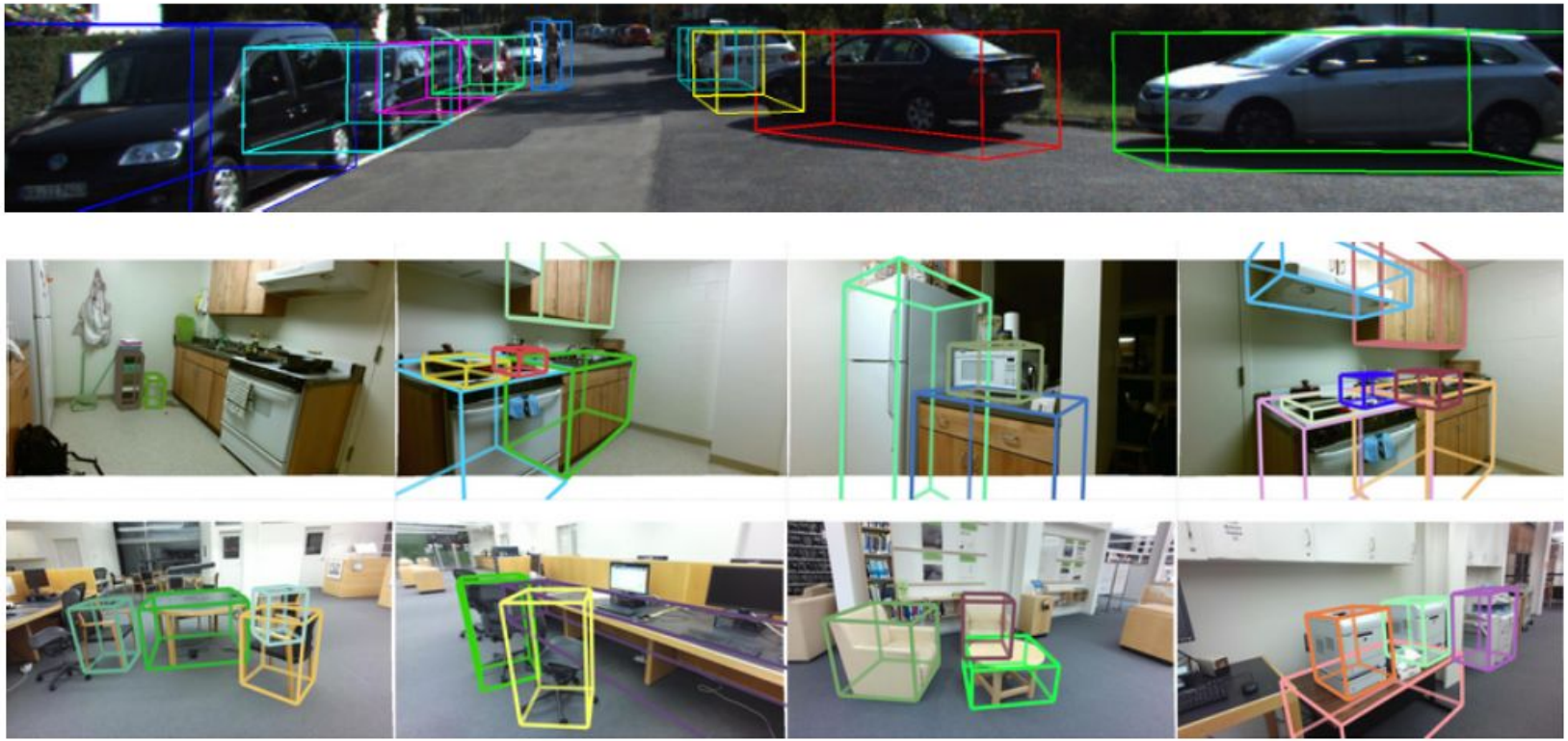




Introduction

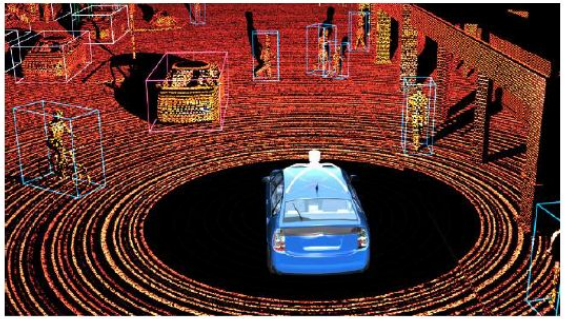
Laser scanners and depth cameras are commonly used in robot perception and navigation systems. These are active sensors, radiating light energy in the direction of observed objects and receiving the energy reflected from them. They have a number of advantages over passive vision systems, among which the most significant seem to be their independence from the natural illumination of the scene and their ability to measure distances directly, without complex and time-consuming image processing.

Applications of 3D object detection

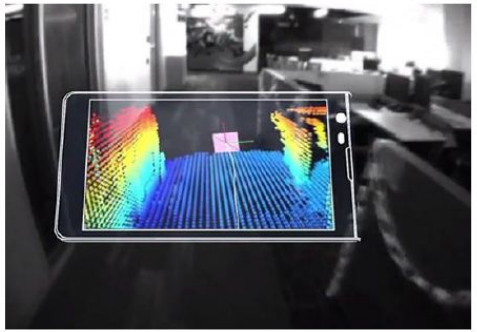


Applications of 3D object detection

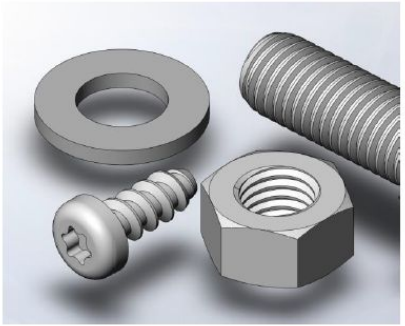
Robot Perception

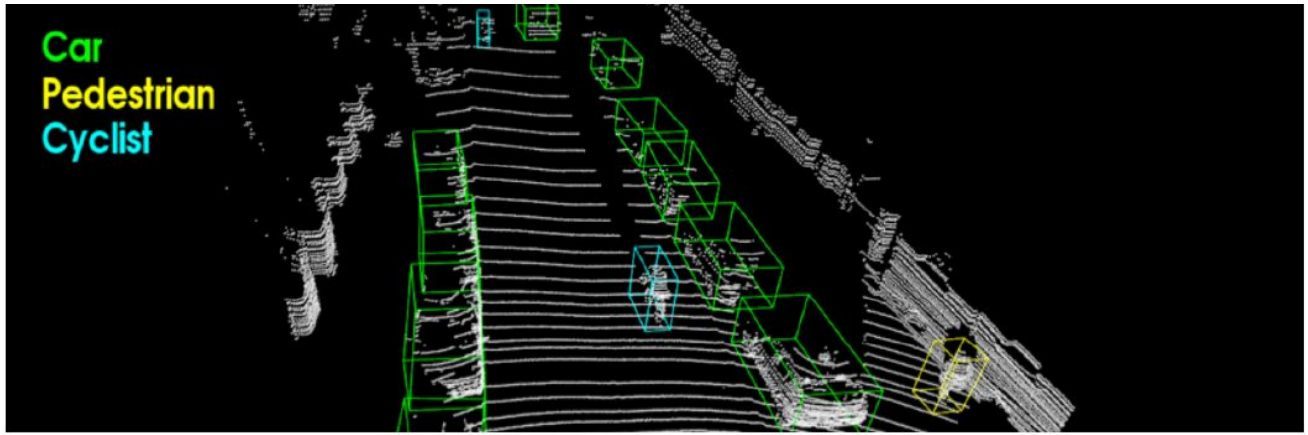


Augmented Reality



Shape Design





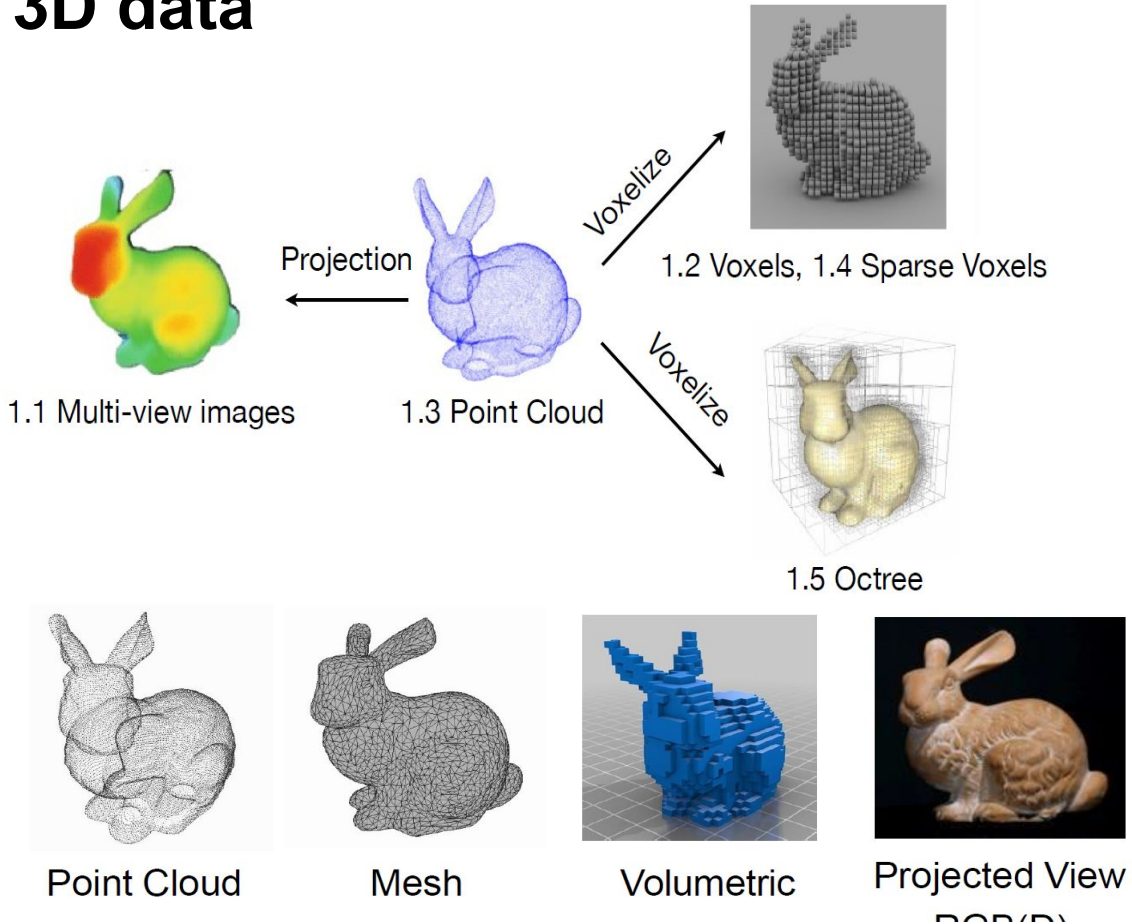
Representations of 3D data

Images:

- Dense and regular
- Can be processed by conventional CNNs

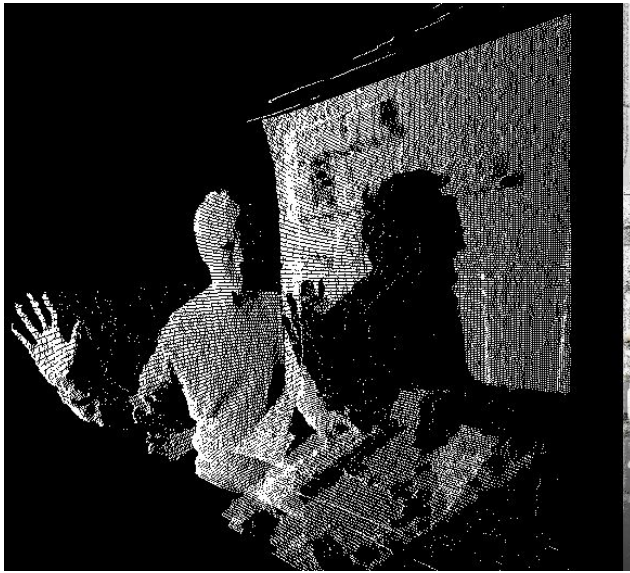
3D range data:

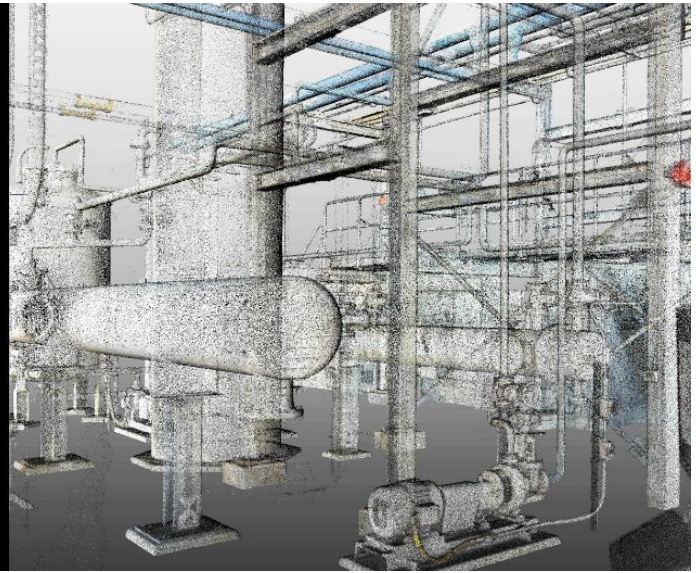
- Extremely sparse
- Irregularly stored in memory
- Usually processed by specialized operators



Representations of 3D data

A **point cloud** is a collection of points that store 3D coordinate information and (optionally) colour and any other local features (e.g. calculated surface normal vectors). These points can be elements of a matrix and correspond to individual pixels of the depth camera image (organised cloud, which is a simple generalisation of the flat image), or as a list with an arbitrary order of points (unorganised cloud).

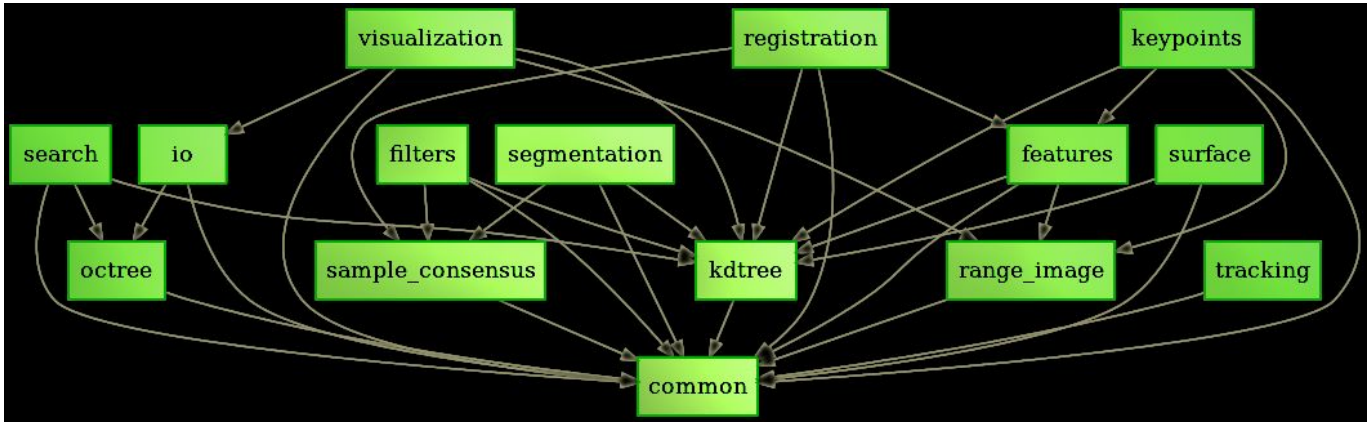




The unorganised cloud is a more general form, not restricted by one particular viewpoint. Thanks to this property, it is not burdened by the problem of point obscuration and local resolution.

Point Cloud Library

- PCL is a large scale, open project for 2D/3D image and point cloud processing (in C++, with python bindings).
- The PCL framework contains numerous state-of-the art algorithms including ltering, feature estimation, surface reconstruction, registration, model tting and segmentation.
- PCL is cross-platform, and has been successfully compiled and deployed on Linux, Windows, Android.
- Website: pointclouds.org



Point Cloud Library

PCD File Format

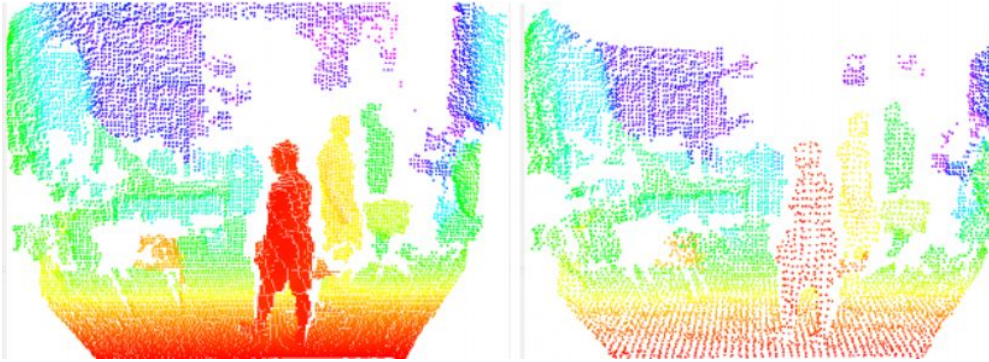
A simple le format for storing multi-dimensional point data. It consists of a text header (with the elds below), followed by the data in ASCII (w/ points on separate lines) or binary (a memory copy of the points vector of the PC).

- VERSION - the PCD le version (usually .7)
- FIELDS - the name of each dimension/eld that a point can have (e.g. FIELDS x y z)
- SIZE - the size of each dimension in bytes (e.g. a float is 4)
- TYPE - the type of each dimension as a char (I = signed, U = unsigned, F = float)
- COUNT - the number of elements in each dimension (e.g. x, y, or z would only have 1, but a histogram would have N)
- WIDTH - the width of the point cloud
- HEIGHT - the height of the point cloud
- VIEWPOINT - an acquisition viewpoint for the points: translation (tx ty tz) + quaternion (qw qx qy qz)
- POINTS - the total number of points in the cloud
- DATA - the data type that the point cloud data is stored in (ascii or binary)

Point Cloud Library

When working with 3D data, there are many reasons for filtering your data:

- Restricting range (PassThrough)
- Downsampling (VoxelGrid)
- Outlier removal (StatisticalOutlierRemoval / RadiusOutlierRemoval)
- Selecting indices



Downsampling to a Voxel Grid

Voxelize the cloud to a 3D grid. Each occupied voxel is approximated by the centroid of the points inside of it.

filtering.cpp

```
// Downsample to voxel grid
pcl::VoxelGrid<pcl::PointXYZ> vg;
vg.setInputCloud (cloud);
vg.setLeafSize (0.01f, 0.01f, 0.01f);
vg.filter (*cloud_filtered);
```

Iterative Closest Point

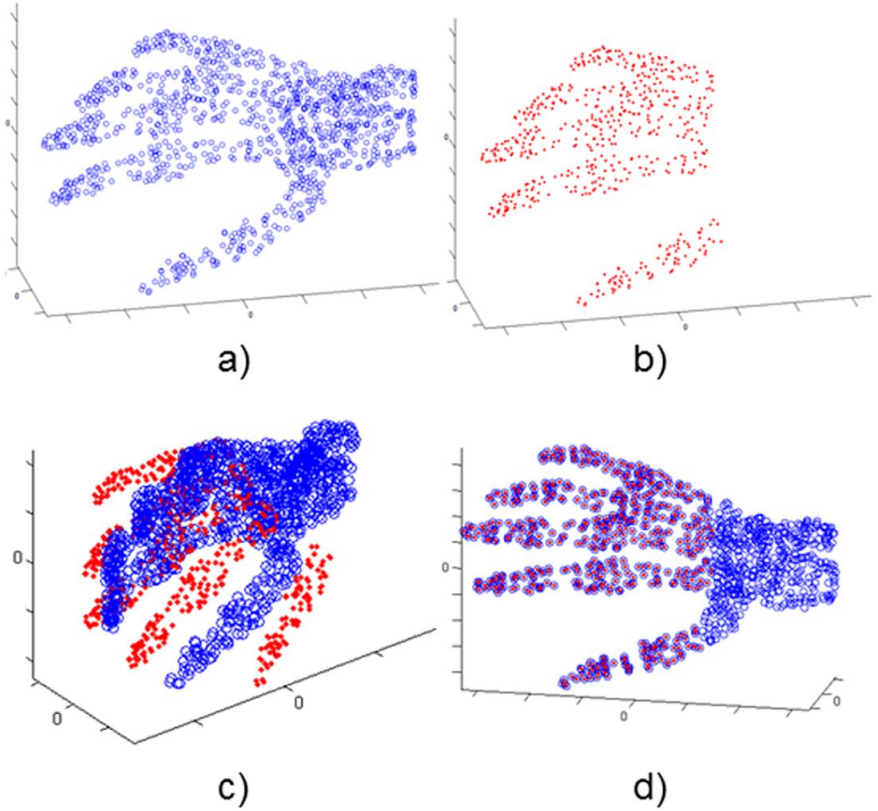
ICP iteratively revises the transformation (translation, rotation) needed to minimize the distance between the points of two raw scans.

Inputs: points from two raw scans, initial estimation of the transformation, criteria for stopping the iteration.

Output: refined transformation.

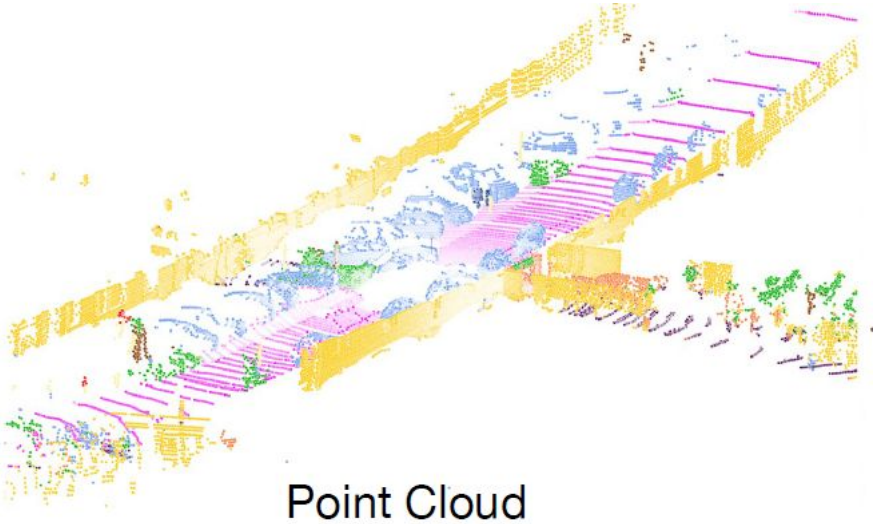
The algorithm steps are :

1. Associate points by the nearest neighbor criteria.
2. Estimate transformation parameters using a mean square cost function.
3. Transform the points using the estimated parameters.
4. Iterate (re-associate the points and so on).



Range images

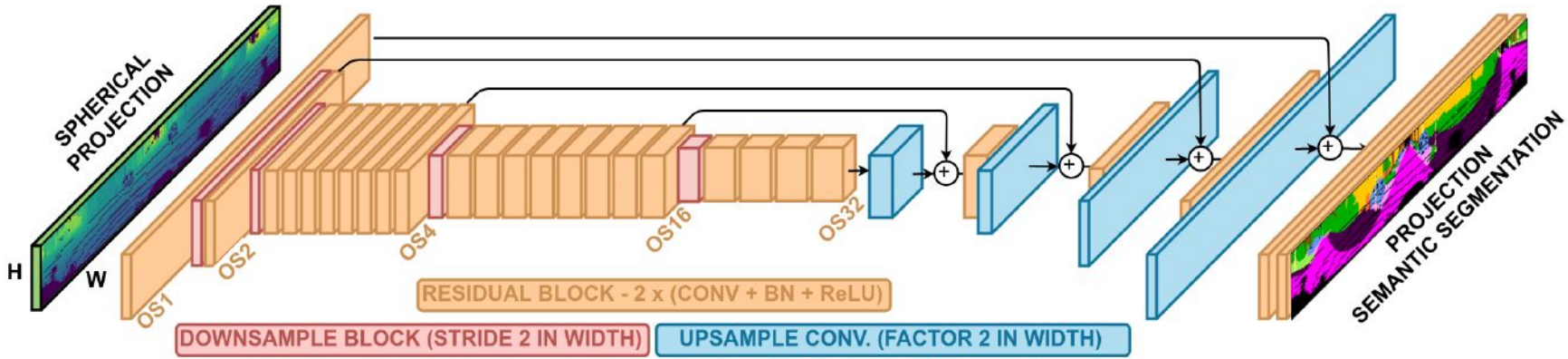
The spherical projected view is also known as the range image, and is widely used in the field of LiDAR point cloud processing. Successful applications include object proposal, object detection, object segmentation, motion prediction, semantic segmentation and LiDAR odometry



Range images

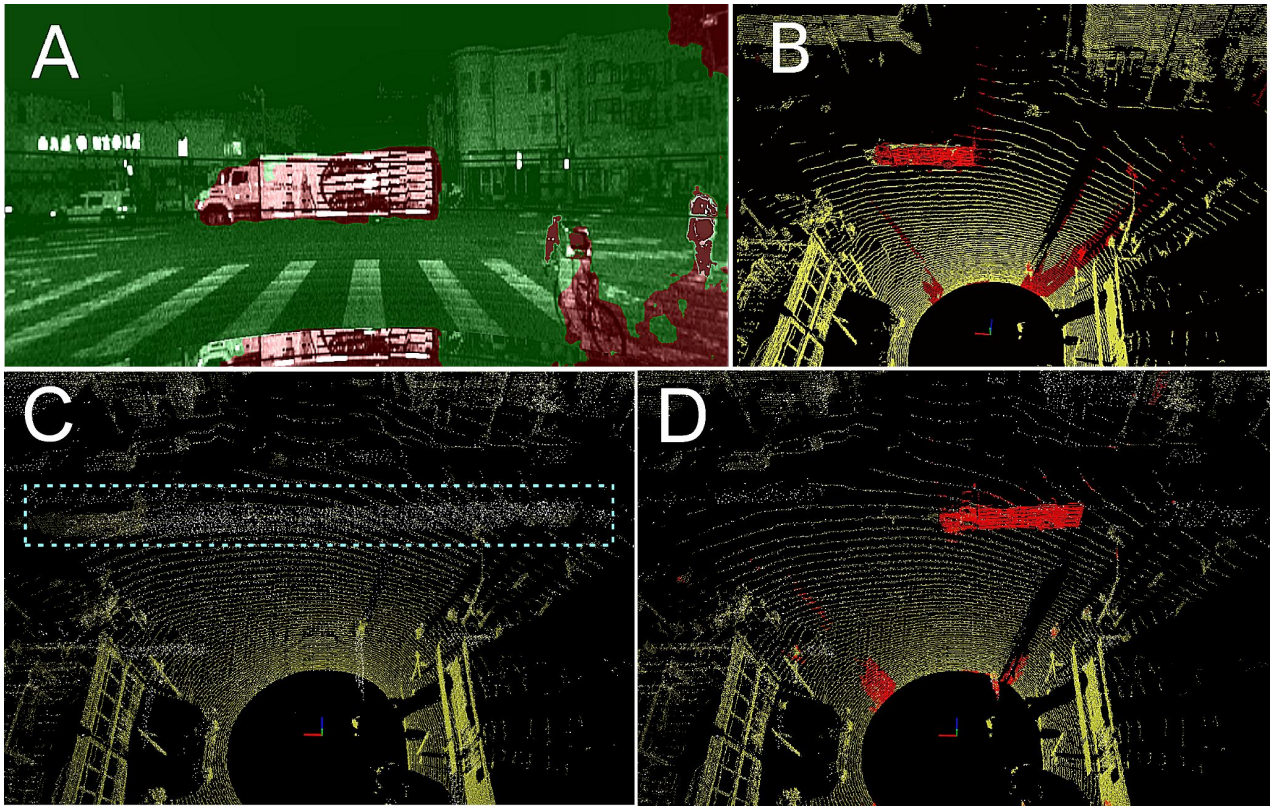
RangeNet: 3D semantic segmentation on range images

- Similar idea to 2D image segmentation: using a U-net architecture to perform semantic segmentation on spherical projections (range images).
- Pros: Can directly utilize 2D CNNs, easy to deploy on hardware;
- Cons: Can introduce geometric distortion, works better for semantic segmentation and requires



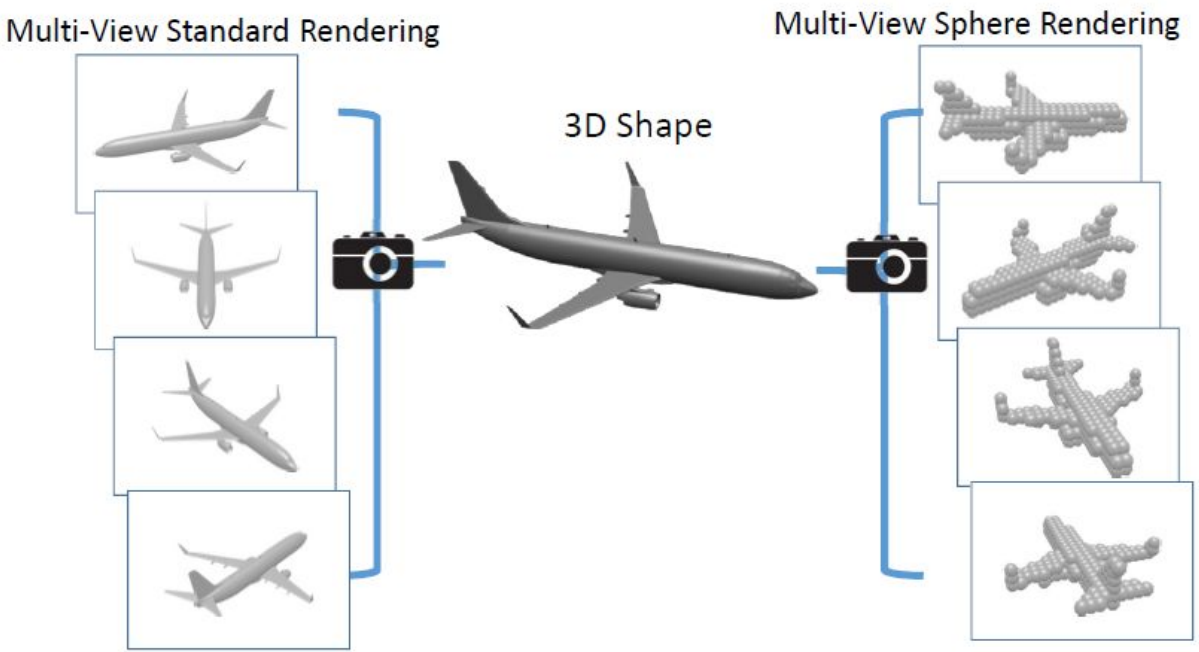
Range images

- Example of non-stationary objects segmentation and removal on a short sequence from OS1-128: segmented intensity image (A), rejected points (red) in a single scan (B), and the resulting map without (C) and with (D) the proposed method of rejecting non-stationary objects.
- Non-stationary objects can be detected using neural network models trained with 2-D grayscale images in the supervised or training process. The point clouds are filtered using the corresponding intensity images with labeled pixels.



Range images

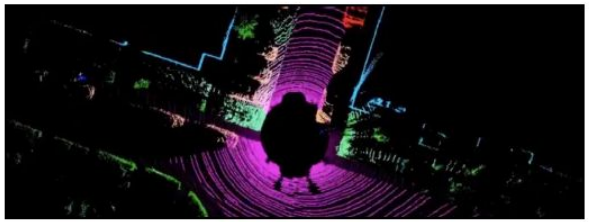
- **Project 3D point cloud into image like representation**
- Multi-view images can be obtained by rendering the point cloud from different viewing angles.
- One can apply standard 2D CNN on each rendered image and aggregate the results with average / max pooling to obtain the final predictions.



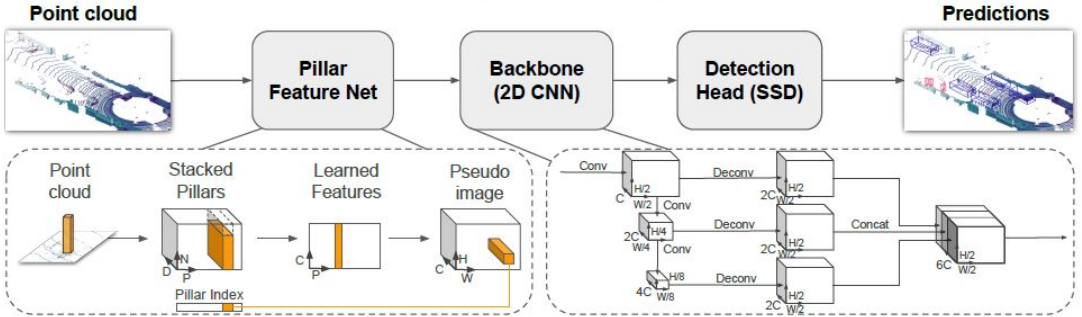
Bird’s Eye View projection

PointPillars: Cartesian projection

- PointPillars flattens the point cloud in the BEV space and applies PointNet within each grid on the BEV plane.
- After that, the entire feature map is fed through a 2D CNN to produce object detection predictions.
- P: number of pillars (~10000), N: maximum points of points within each pillar (20), D/C: input/output channels (10~64).



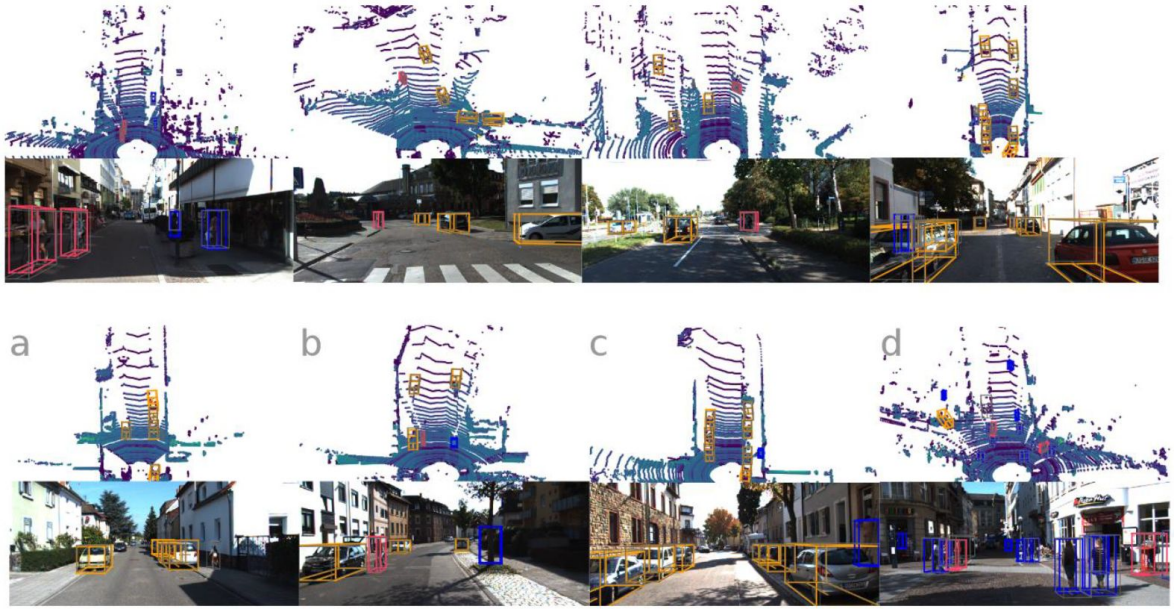
3D scene in the BEV



Bird’s Eye View projection

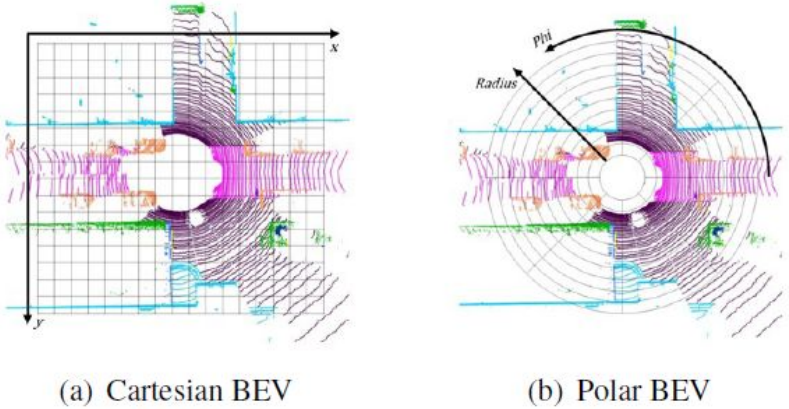
PointPillars: Cartesian projection

- PointPillars flattens the point cloud in the BEV space and applies PointNet within each grid on the BEV plane.
- After that, the entire feature map is fed through a 2D CNN to produce object detection predictions.
- P: number of pillars (~10000), N: maximum points of points within each pillar (20), D/C: input/output channels (10~64).

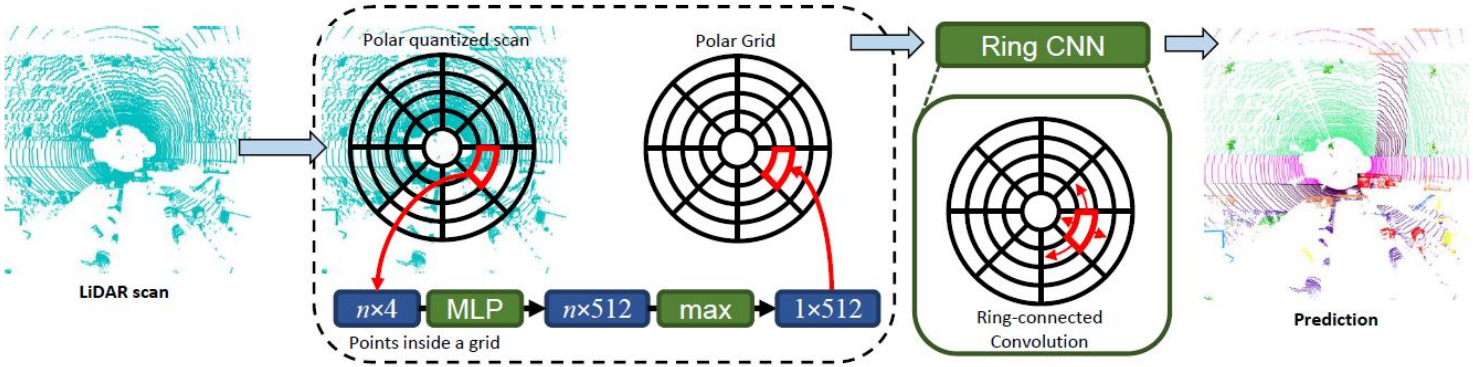


Bird's Eye View projection

- From Cartesian coordinates to polar coordinates on the xy (BEV) plane:
- $\rho = \sqrt{x^2 + y^2}$,
- $\theta = \arctan \frac{y}{x}$.

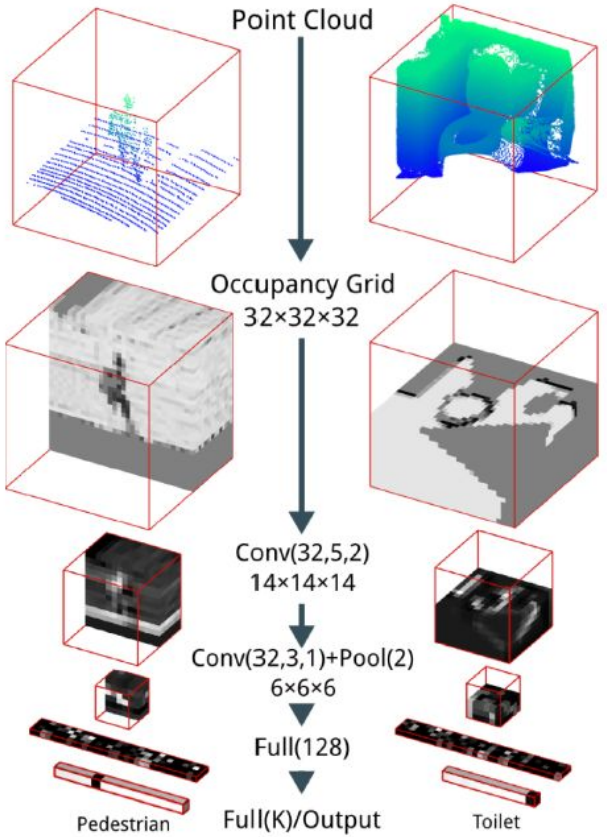


(a) Cartesian BEV (b) Polar BEV



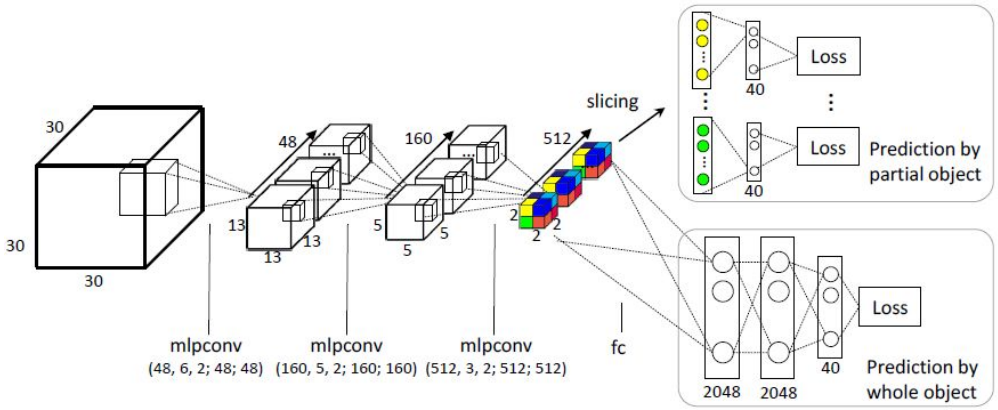
Voxelization

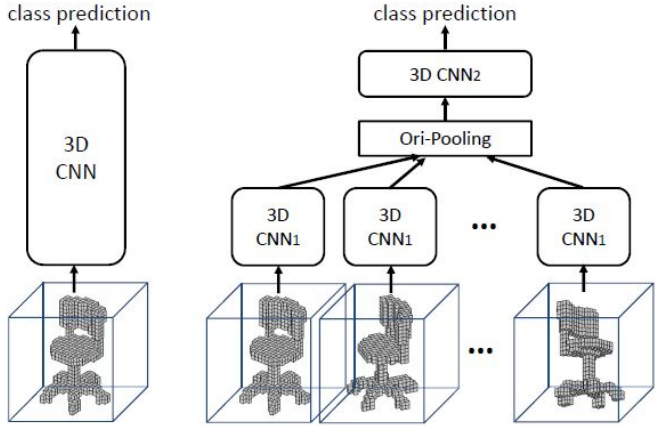
- For a 3D point $p = [x, y, z]$ with features $f \in \mathbb{R}^C$, voxelization under voxel size r means that the coordinates of point p will be quantized to $\hat{p} = [\text{round}(\frac{x}{r}), \text{round}(\frac{y}{r}), \text{round}(\frac{z}{r})]$.
- We create a 4D tensor V of size $H \times W \times D \times C$, where $[H, W, D]$ is the upper bound of $\hat{p}$ for all $p \in P = \{(p_i, f_i)\}$. Then, we insert the feature f_i to location $\hat{p}_i$ of the 4D tensor V.



Deep learning on voxels

3D CNN on Voxelized Point Cloud

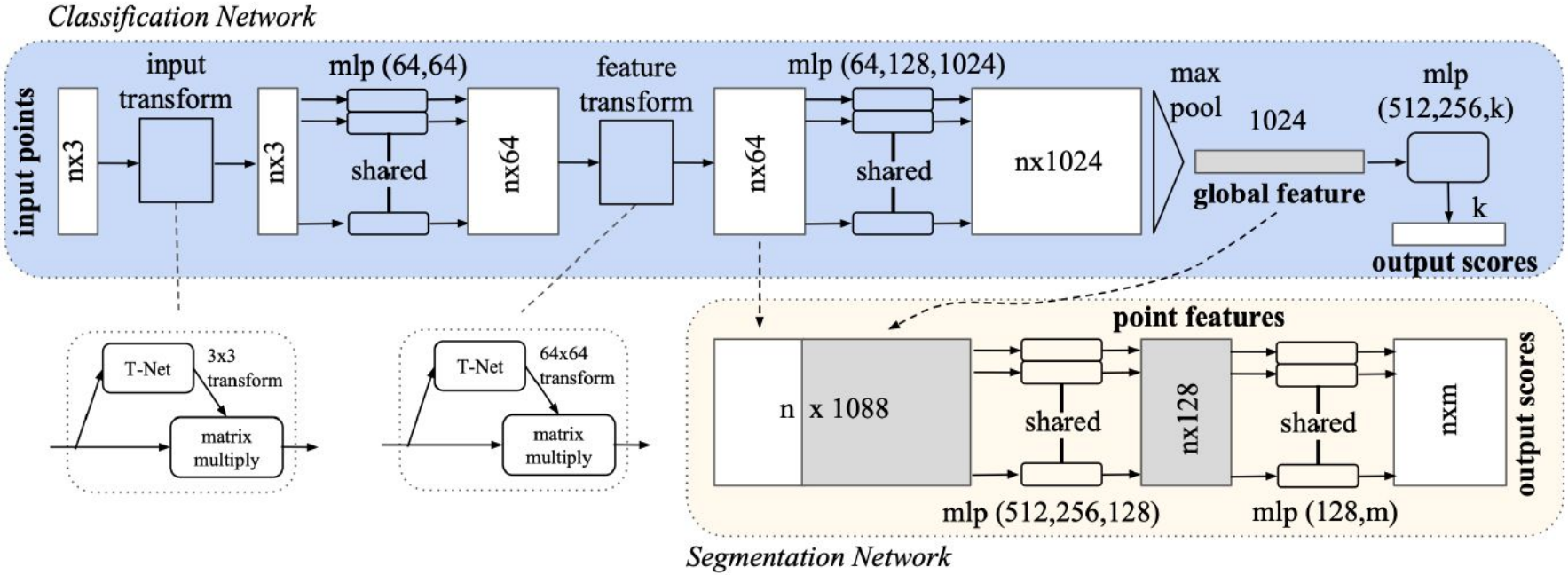




- Subvolume supervision (left): predicting the classification labels from partial voxel feature maps. This can help reduce overfitting.
- Orientation pooling (right): CNNs are not rotationally-invariant. Feeding voxels from different angles to the network and aggregate the results like in MVCNNs.

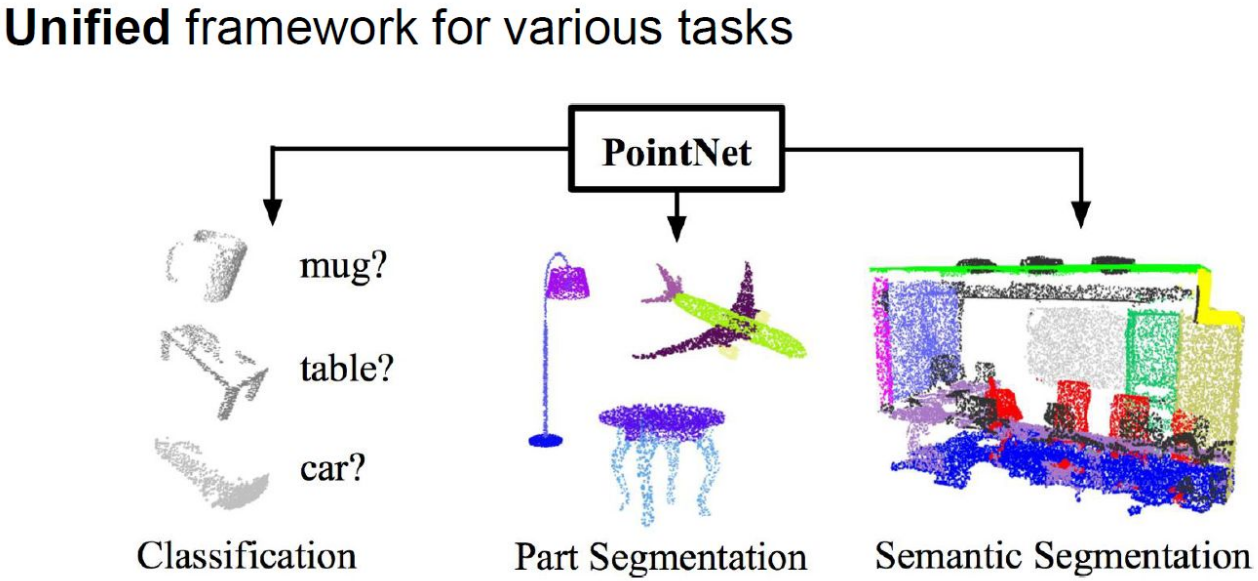
Direct processing of point clouds / PointNet

PointNet models point clouds with MLP layers and a symmetric (i.e. permutation invariant) function (such as max pooling).



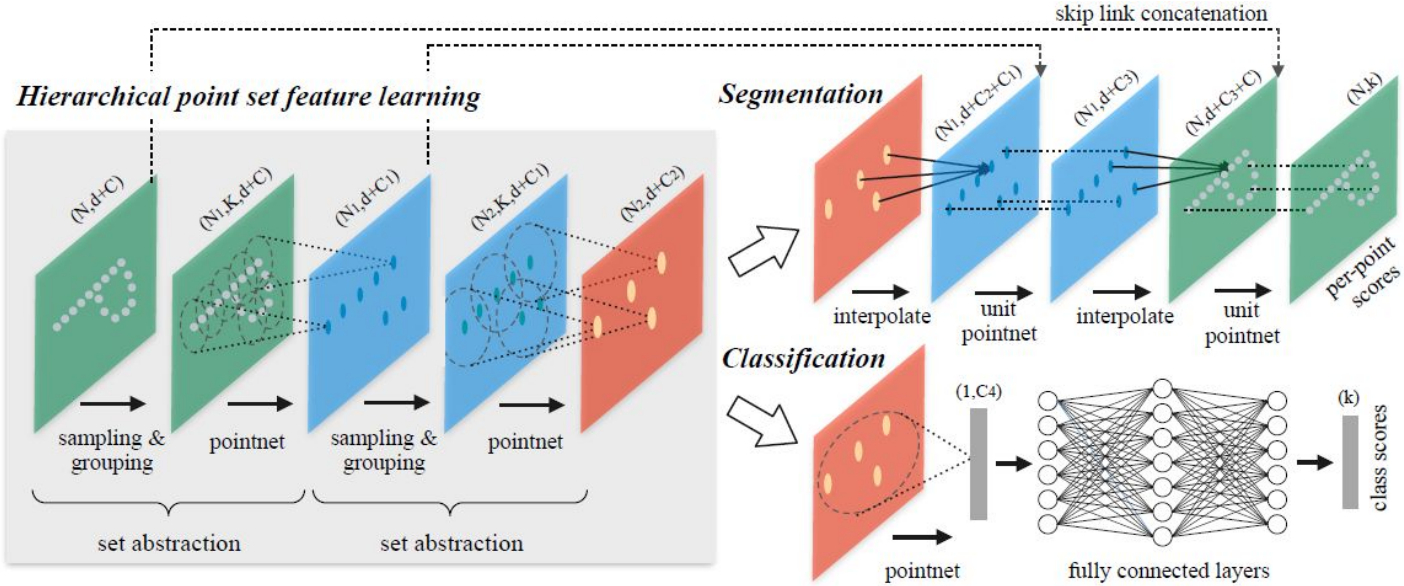
Direct processing of point clouds / PointNet

- Unordered point set as input
- Model needs to be invariant to permutations.
- Invariance under geometric transformations
- Point cloud rotations should not alter classification results.



Direct processing of point clouds / PointNet

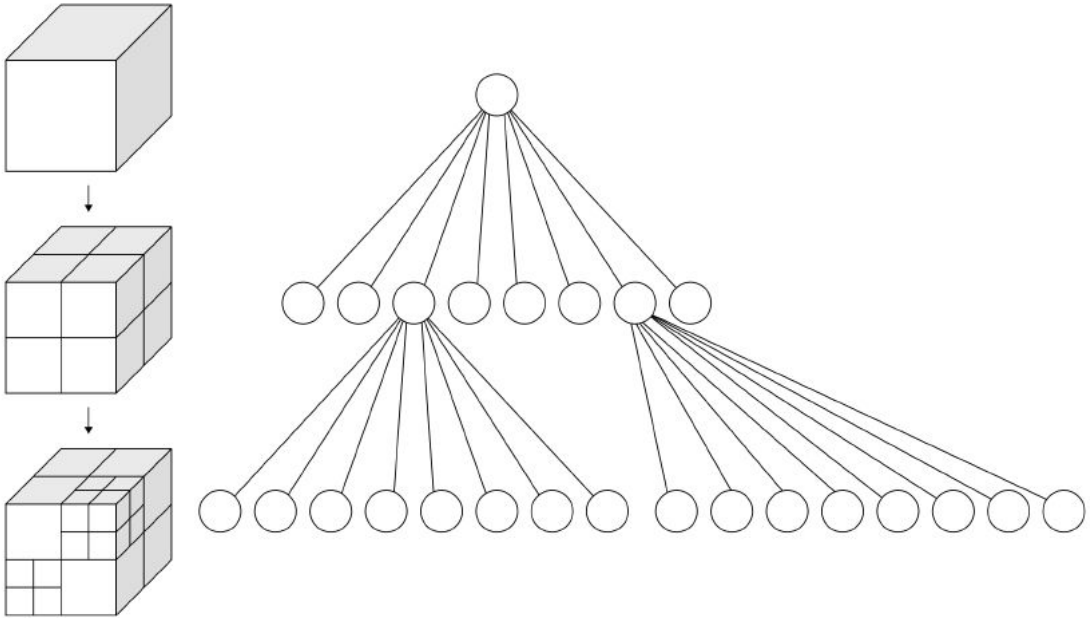
Direct processing of point clouds / PointNet



- PointNet++ is a hierarchical version of PointNet.
- It uses furthest point sampling to downsample the point clouds and kNN (fixed number of neighbors) / ball query (fixed distance) to define the neighborhood of each point. PointNet is applied within the neighborhood of all the points.

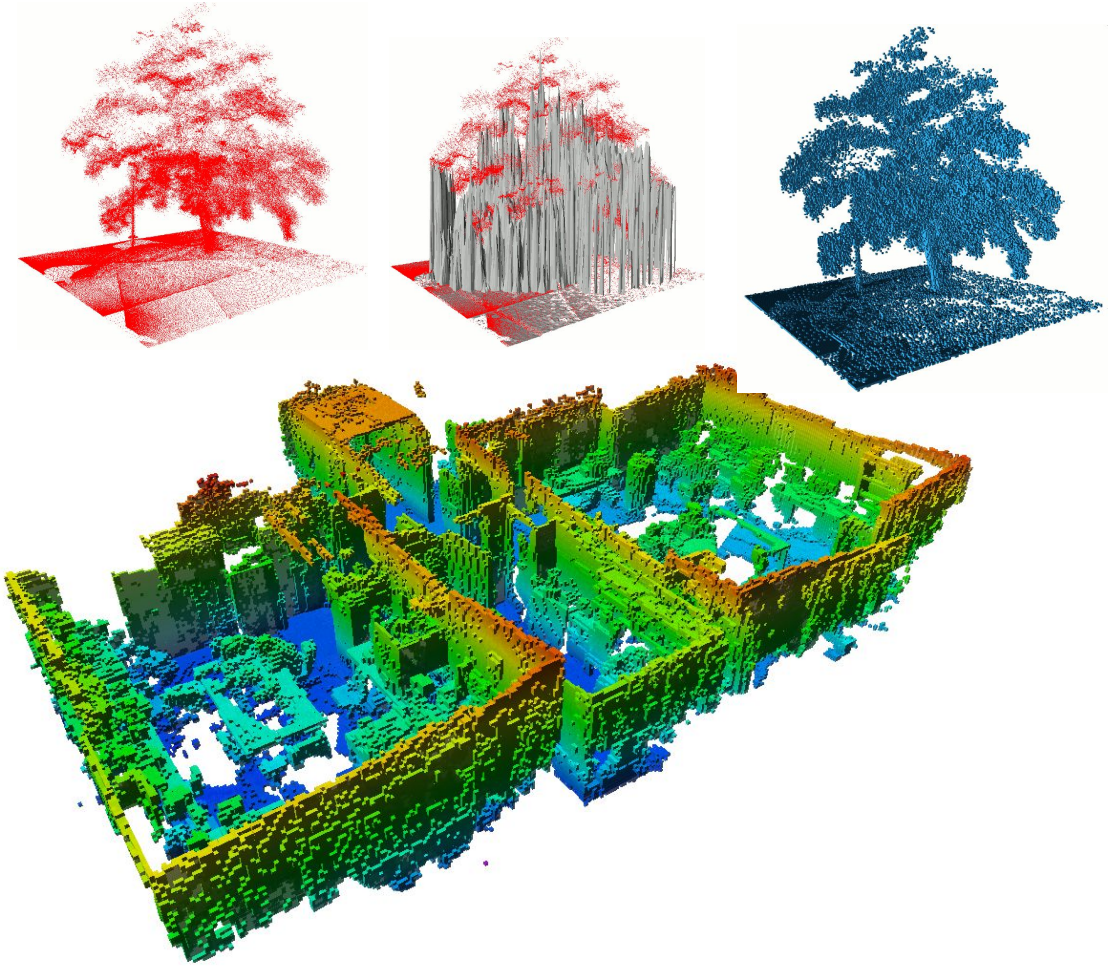
Octree

- A compact and recursive representation for 3D data.
- Octree is a data structure that recursively divides the 3D space into finer granularity voxel grids.
- It has different voxel resolution at different spatial locations. Lower voxel resolution is used for less informative areas.



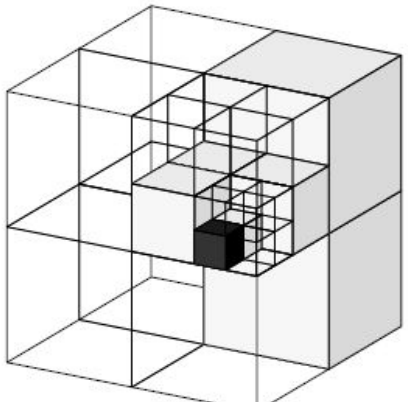
OctoMap

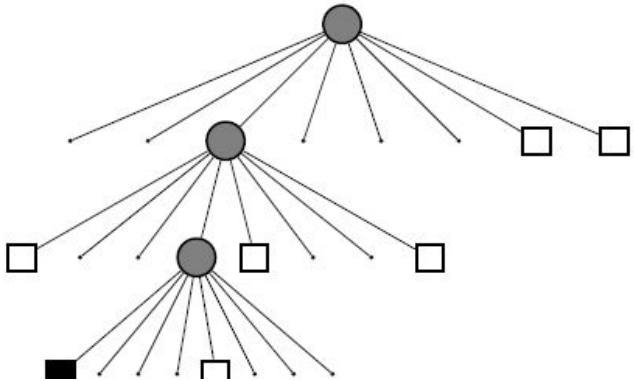
- A volumetric 3D environment model based on octrees that uses probabilistic occupancy estimation.
- It explicitly represents not only occupied space, but also free and unknown areas.
- A compression method keeps the 3D models compact.
- The framework is available as an open-source C++ library and has already been successfully applied in several robotics projects.

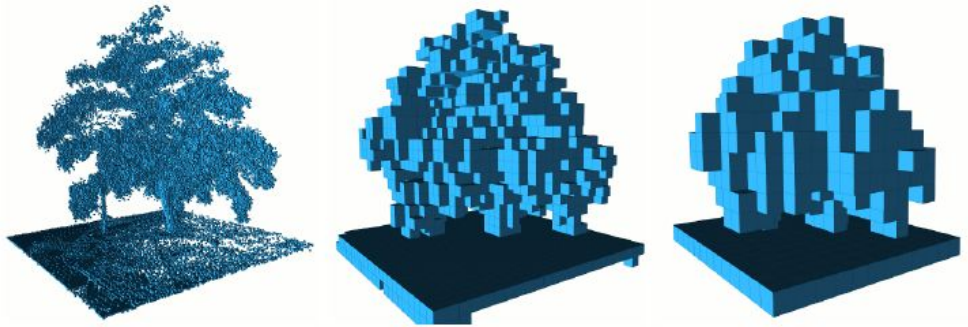


OctoMap

- An octree storing free (shaded white) and occupied (black) cells. The volumetric model is shown on the left and the corresponding tree representation on the right.
- By limiting the depth of a query, multiple resolutions of the same map can be obtained at any time.
- Latest released version: http://octomap.github.io/octomap/







Outcome of the lecture

- A brief review of the active range sensing technology used in robotics and similar areas (self-driving cars, drones).
- Physical principles used in active range sensors shape the practical properties of these sensors (range, accuracy, errors, speed of data acquisition)
- Classic 2D range sensors (laser scanner) used in simple mobile robots (indoor, industrial AGV)
- Modern 3D laser scanners - LiDARs used in self-driving cars, some drones
- Depth cameras used in AR, consumer applications.

IRIM

POZNAN UNIVERSITY OF TECHNOLOGY