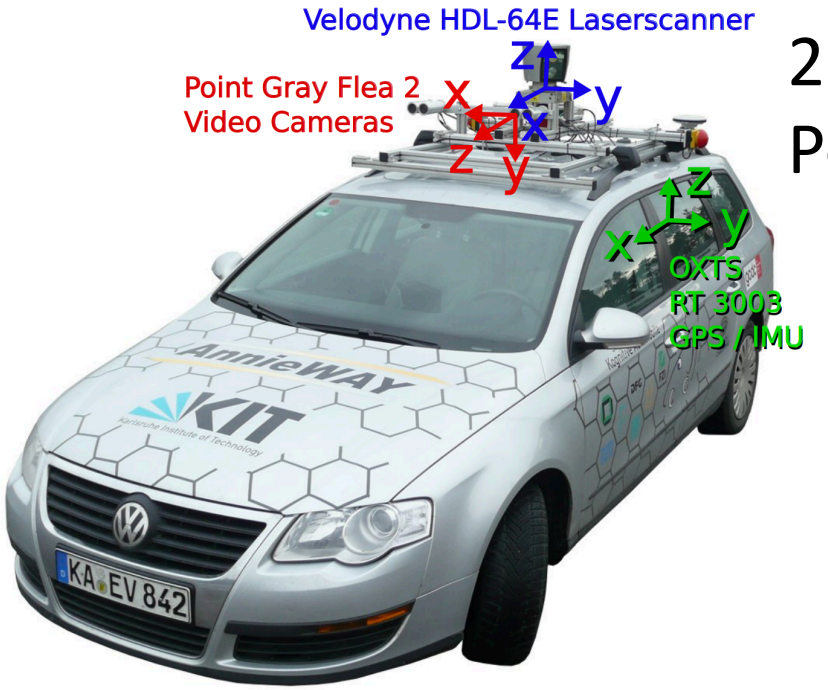
3D Object Detection from Point Cloud and RGB Image

Kang Peilin
Semester project presentation
Computer Vision Laboratory – CVLAB
Professor: Mathieu Salzmann

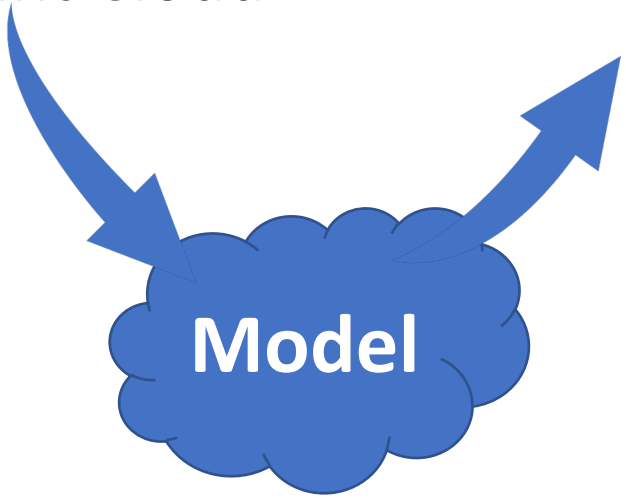
January 23, 2020

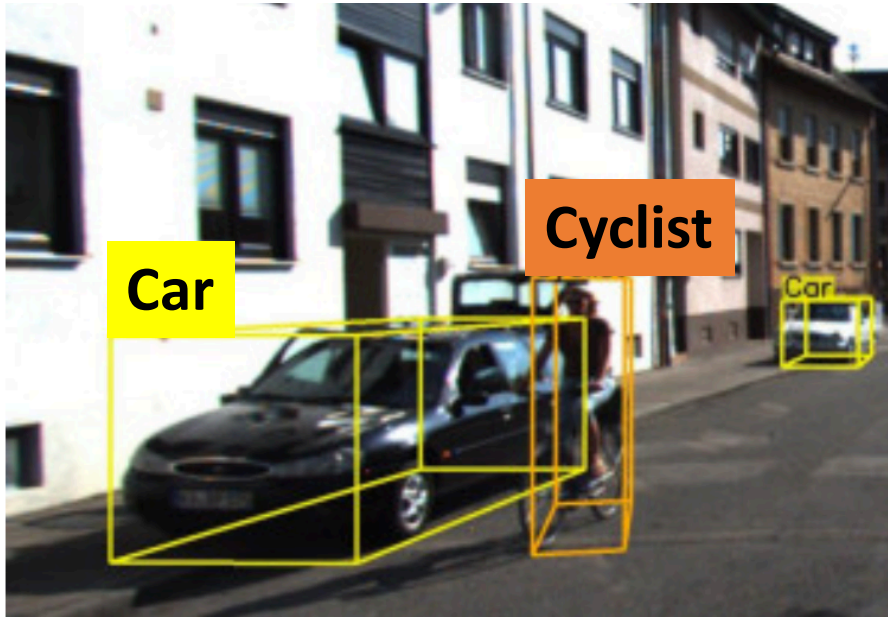
Autonomous Vehicles

A vehicle that is capable of sensing its environment and moving safely with little or no human input.



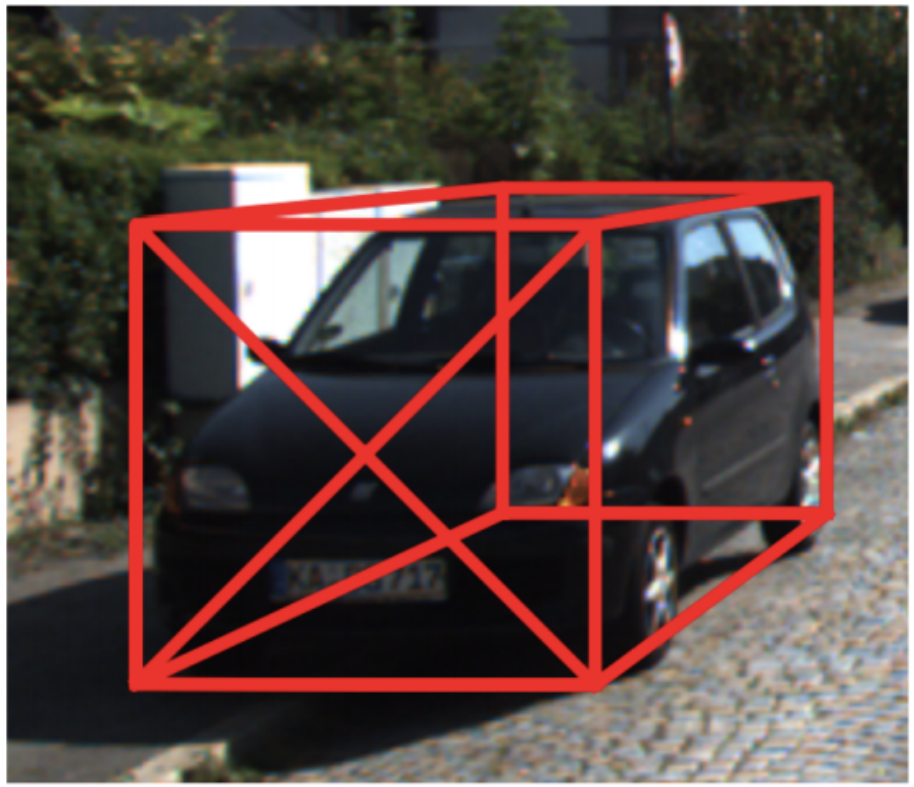
2D Images
Point Cloud





Estimating the localization and orientation of 3D bounding box of various objects

Why 3D Object Detection is Difficult?

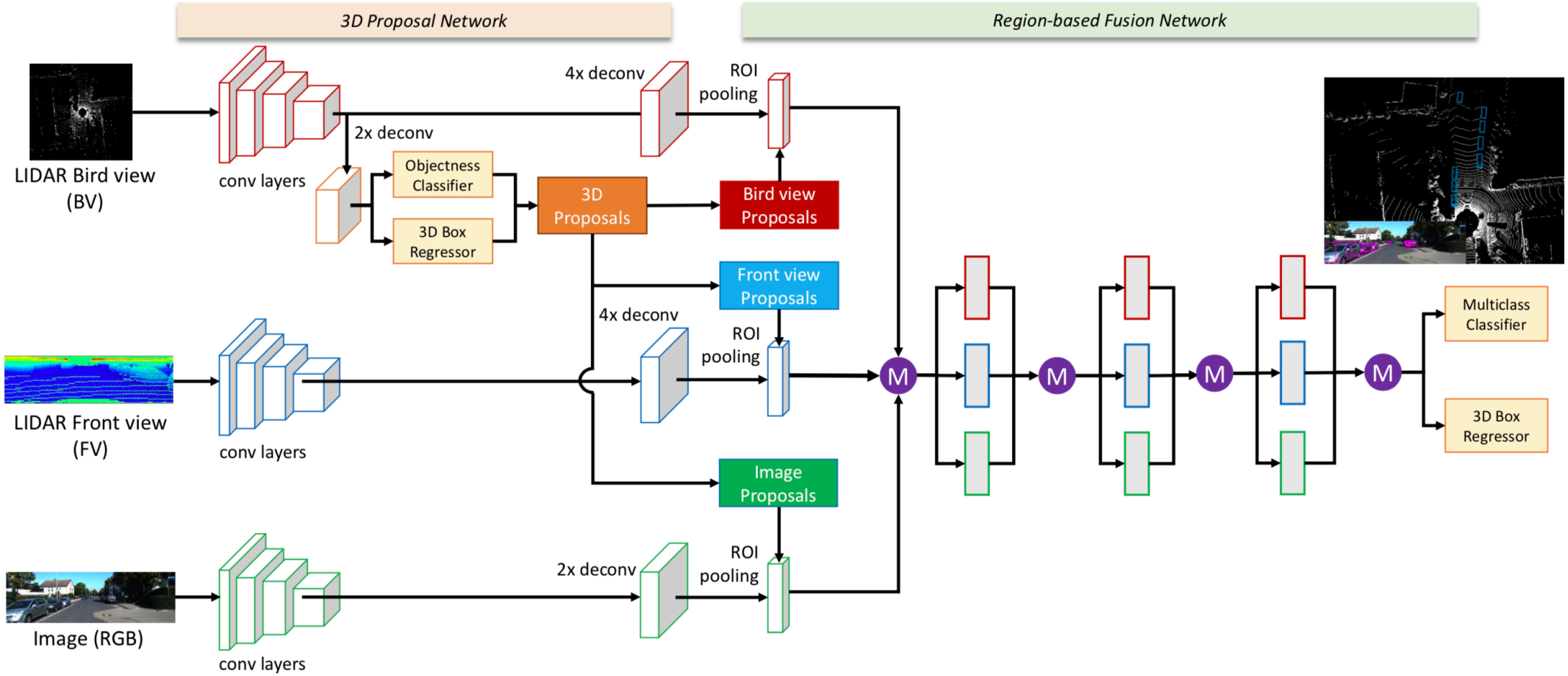


- **Point cloud has irregular data format**
- **Large search space of 7 Degrees-of-Freedom**

2D object detection 4 DoF	3D object detection 7 DoF
2D physical size (w, h)	3D physical size (w, h, l)
2D center location (x, y)	3D center location (x, y, z)
	yaw

- **Sparse representation of point cloud.**

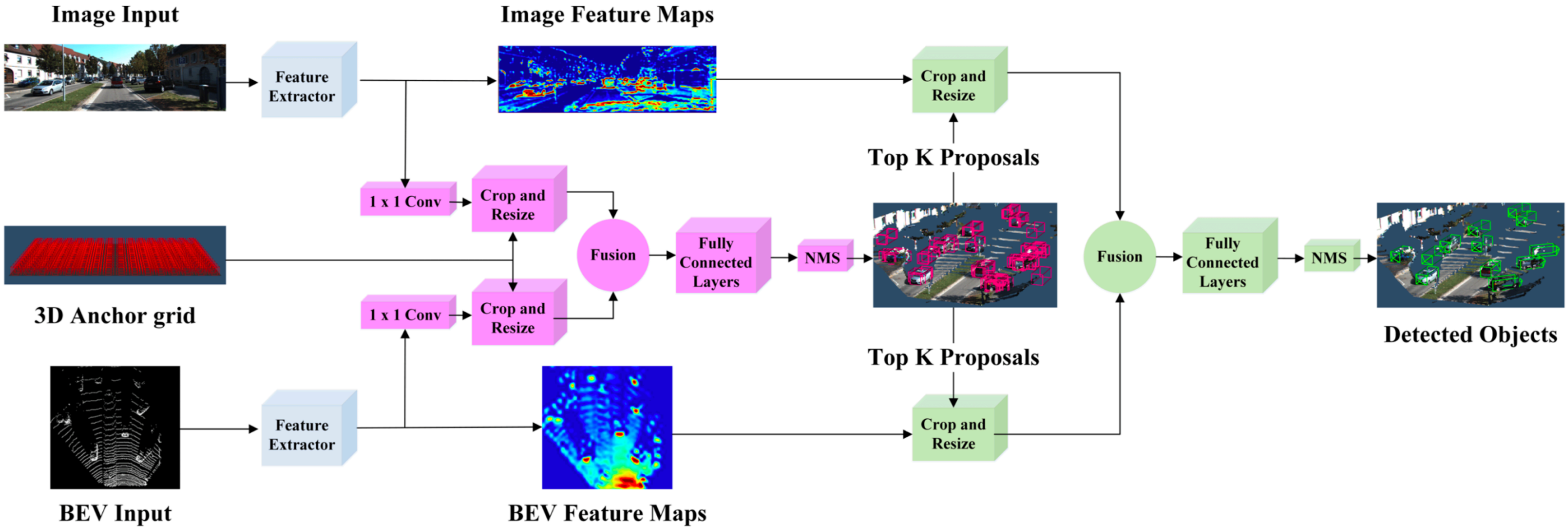
MV3D(2017) leverage the mature 2D detection frameworks



generates 3D object proposals from **bird's eye view** map and project them to three views.

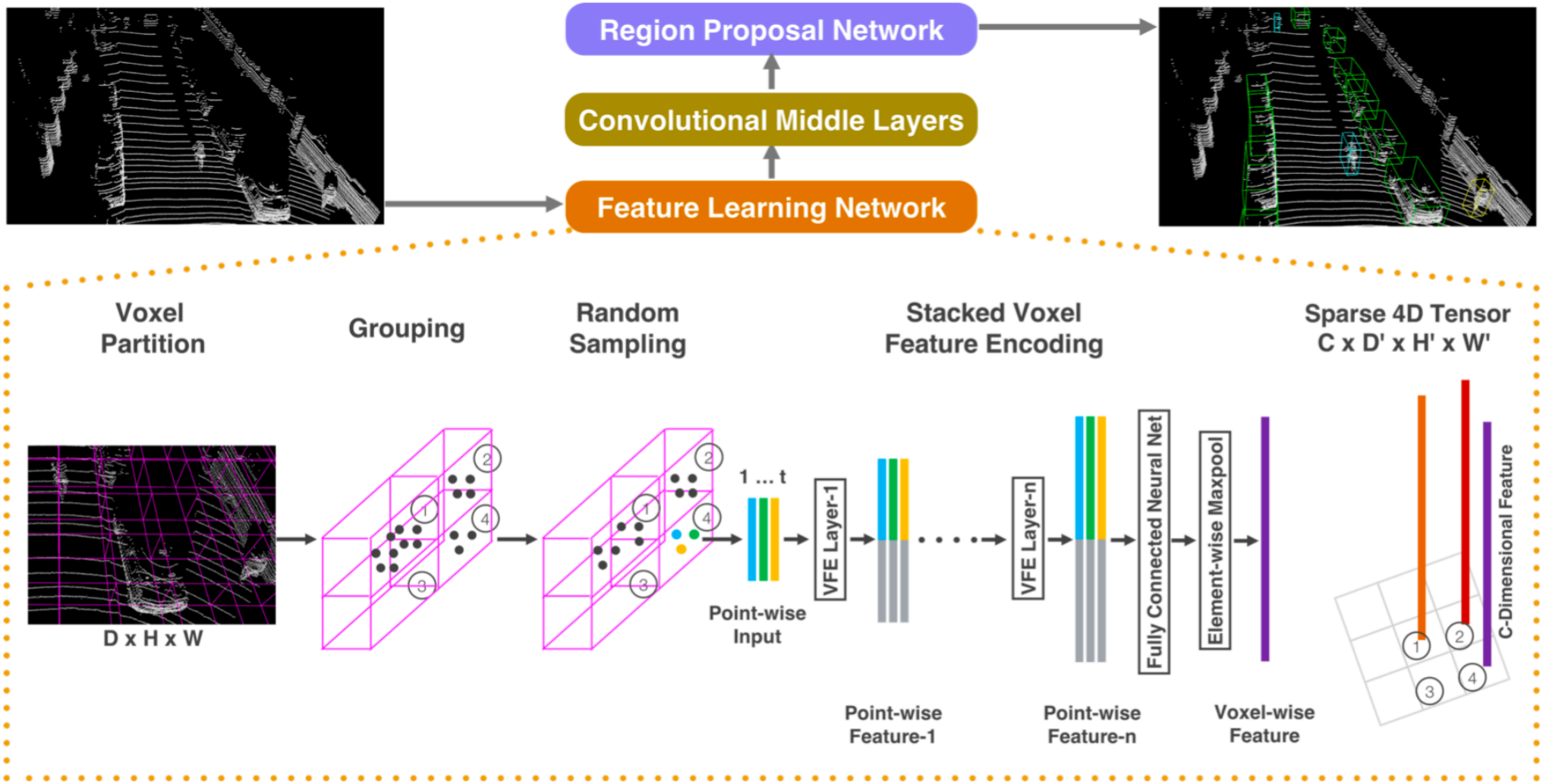
The **fused features** are used to jointly predict object class and do oriented 3D box regression.

AVOD (2018) leverage the mature 2D detection frameworks



The feature extractors are shown in **Blue**, the region proposal network in **Pink**, and the second stage detection network in **Green**

VoxelNet (2017) *Divides a point cloud into equally spaced 3D voxels*



PointRCNN (2019) (state-of-the-art)

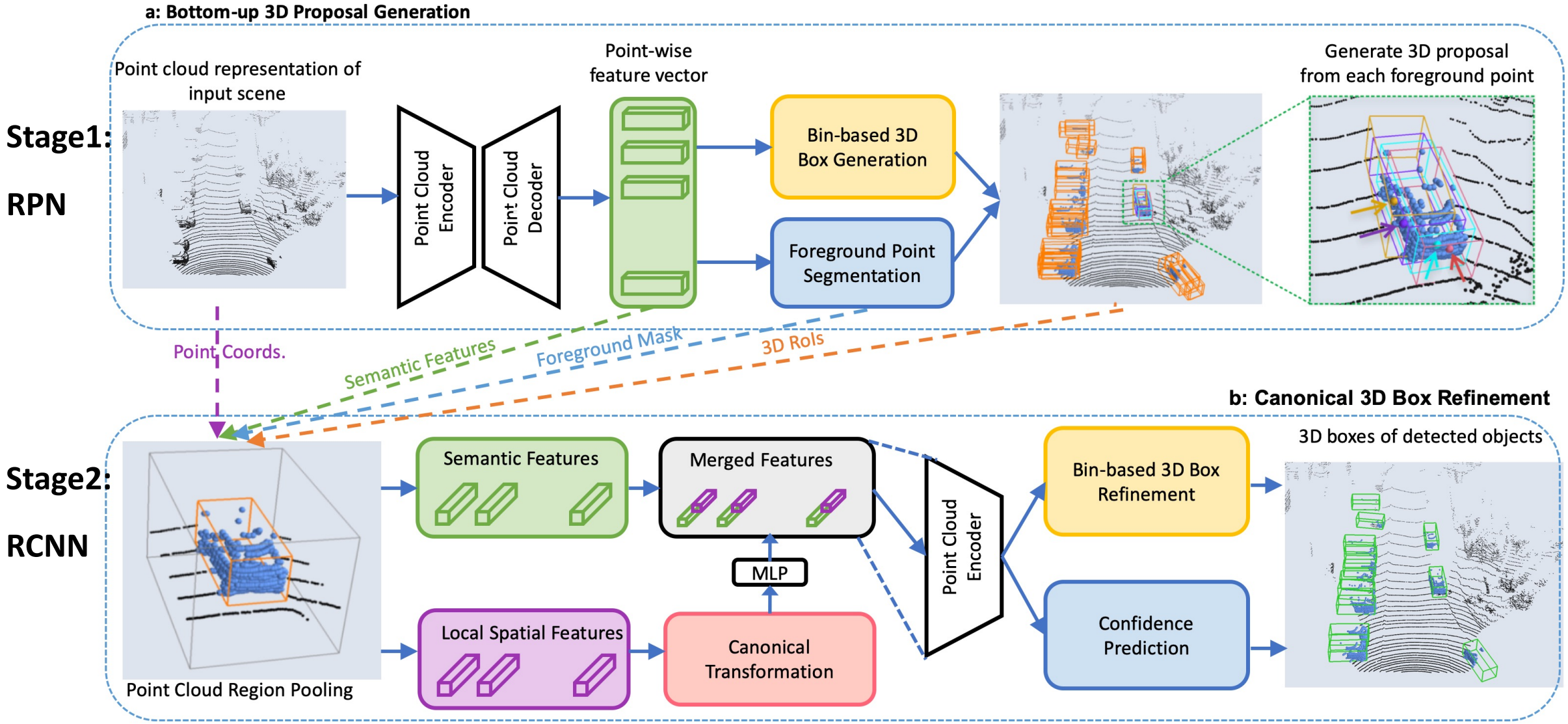
- Previous works focusing on transforming to BEV/FV (MV3D, AVOD) or voxels(VoxelNet) loses geometric information in raw point cloud

- **3D object detection from raw point cloud.**
- **Two stages framework** (similar with faster R-CNN)**:**

  1) bottom-up 3D proposal generation
  - Does not use RGB image, BEV, or voxels, use only point cloud.
  - Learns point wise feature vectors using **PointNet++**
  - Generates high quality 3D proposals from point cloud using point-wise feature vectors.

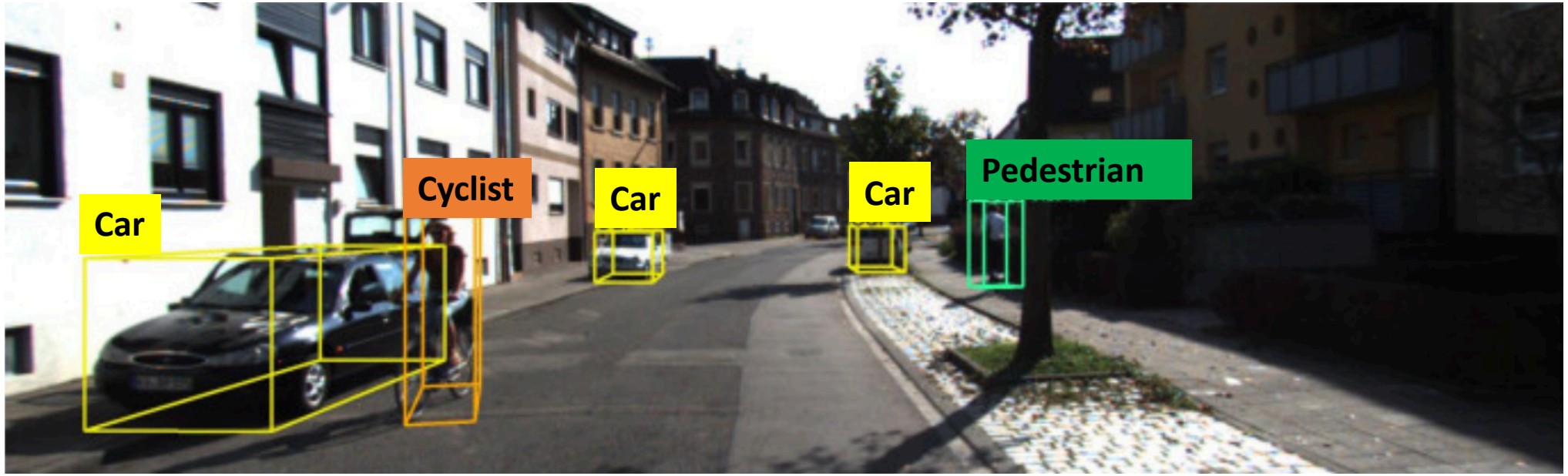
  2) refining proposals in the canonical coordinates
  - Transforming the pooled points of each proposal by canonical coordinates
  - Learning better local spatial features.

PointRCNN (2019) state-of-the-art



Goal: We want to improve the performance of PointRCNN by leveraging some image information properly

KITTI 3D object detection dataset



	Min. bounding box height	Max. occlusion level	Max. truncation
Easy	40 Px	Fully visible	15%
Moderate	25 Px	Partly occluded	30%
Hard	25 Px	Difficult to see	50%

7481 training samples

- *train* split (3712 samples)
- *val* split (3769 samples)

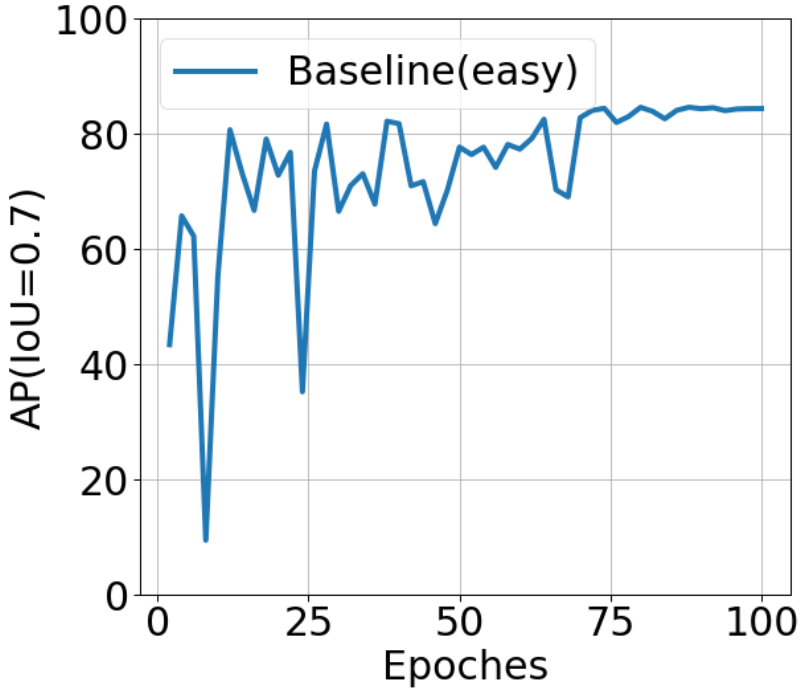
7518 testing samples

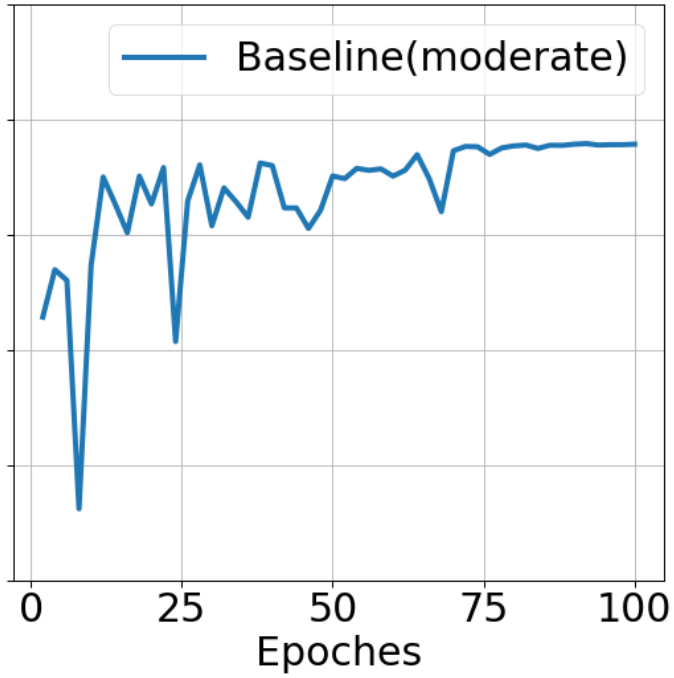
The PointRCNN detect one class at a time, here, we fucus on the detection of Cars.

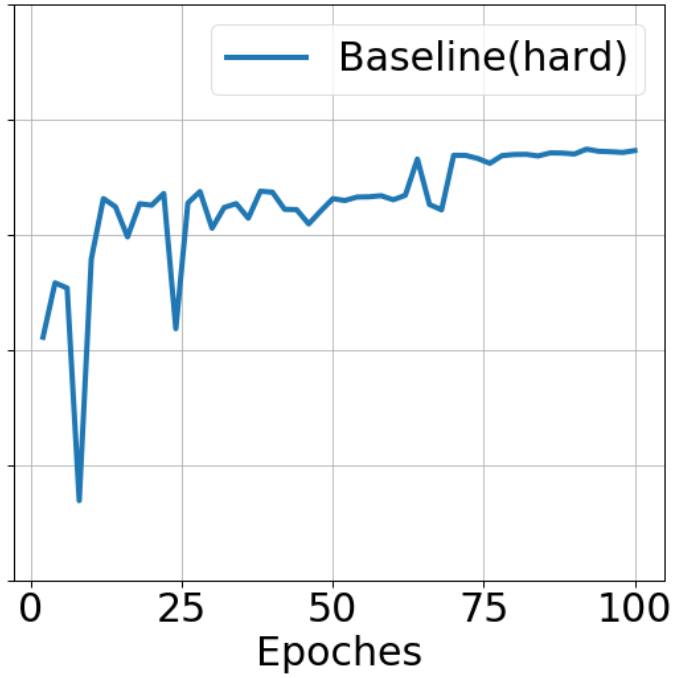
Evaluate PointRCNN (use as baseline)

Average Precision(AP) of 3D object detection with PointRCNN on the car class of KITTI val split set

Car (IoU=0.7)	Easy	Moderate	Hard
Baseline	84.37	75.76	74.65

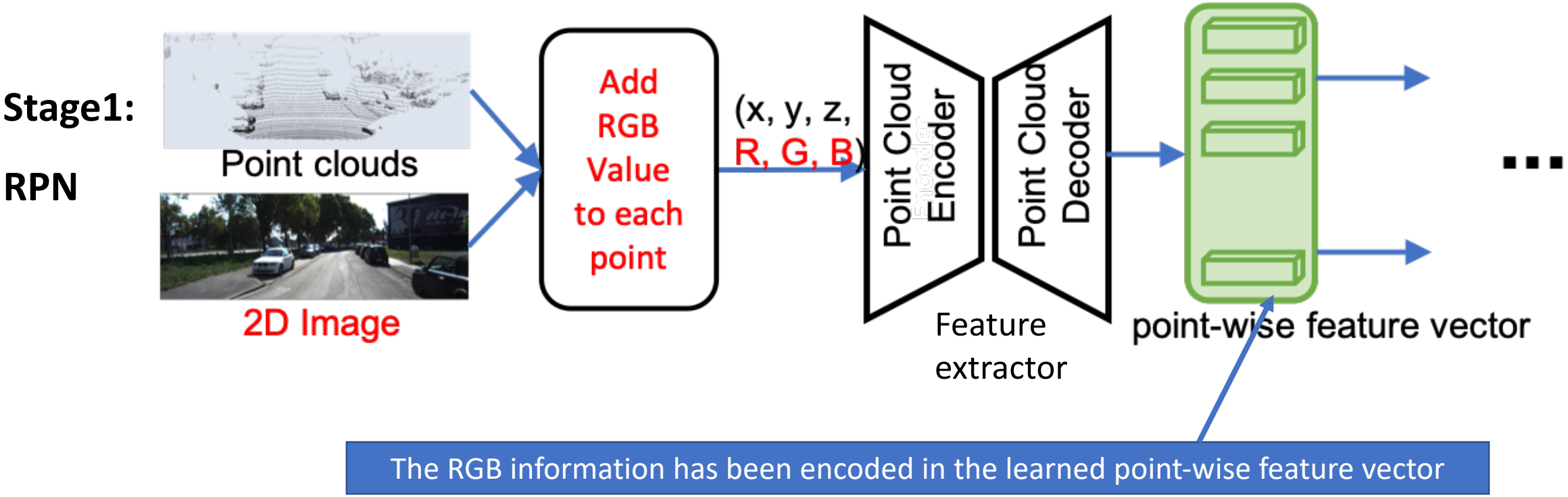






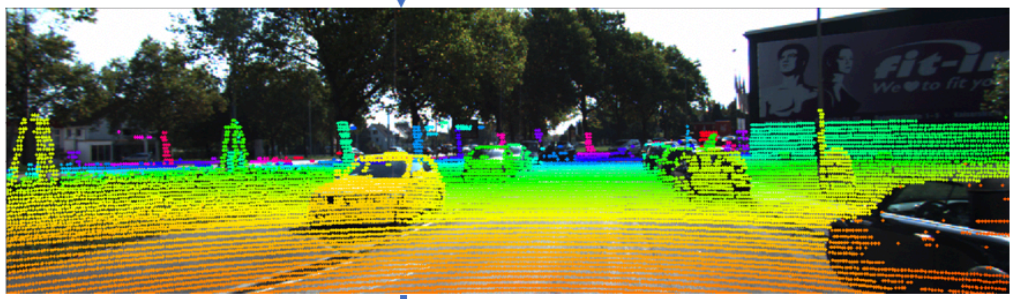
Method 1: Add RGB Data

add the RGB values before the first stage, leave the other parts of model unchanged
the newly added part is in **Red**

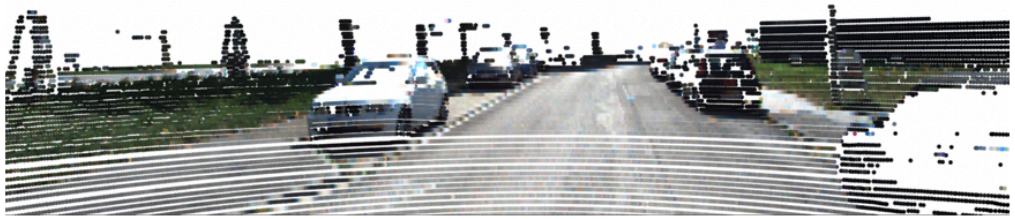


Method 1: Add RGB Data

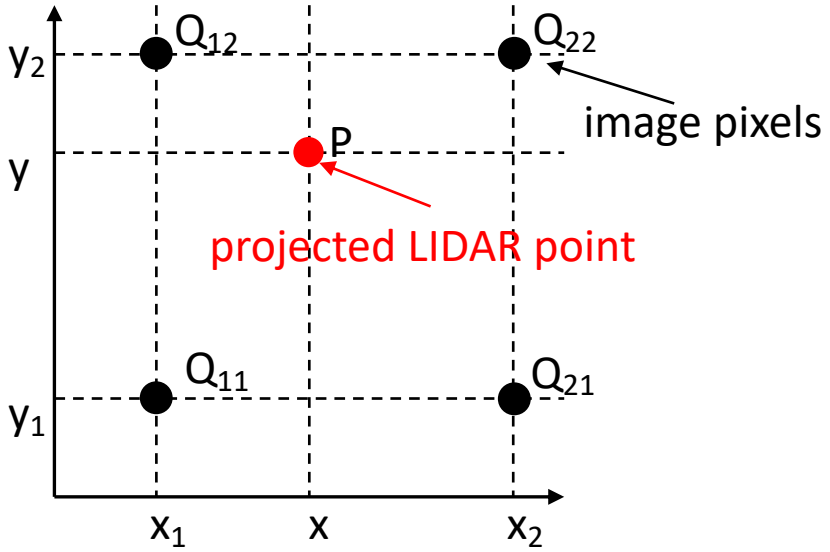
Project Point Clouds to 2D image



Use bilinear interpolation to assign RGB values to each point



Bilinear interpolation

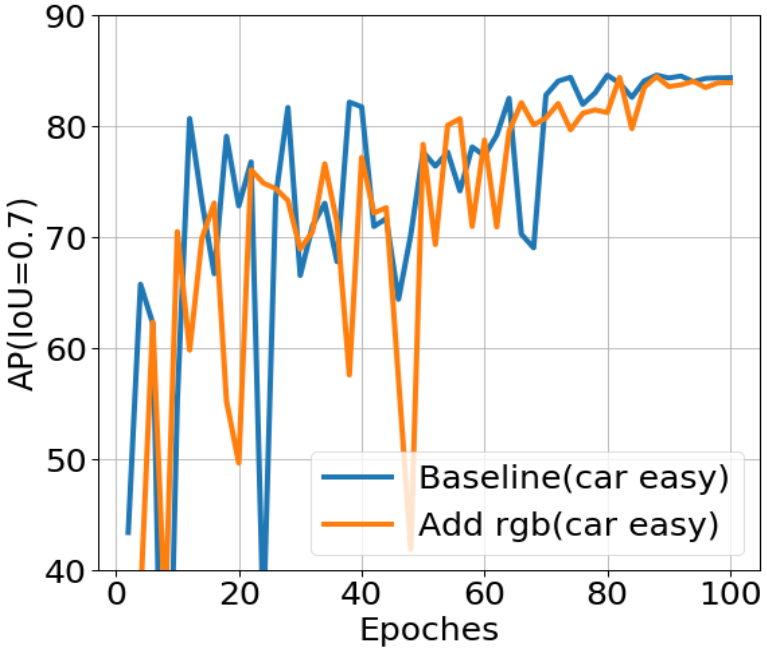


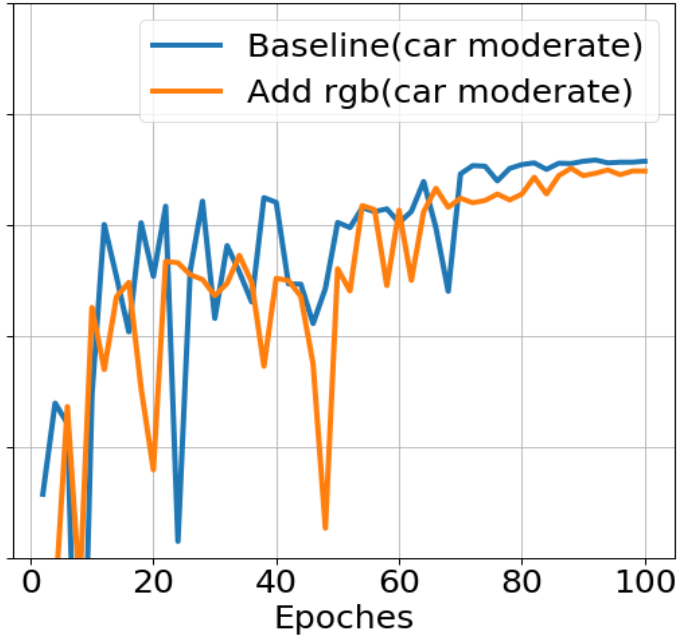
$$P = \frac{(x_2 - x)(y_2 - y)}{(x_2 - x_1)(y_2 - y_1)} \times [R, G, B]_{Q11} + \frac{(x_2 - x)(y - y_1)}{(x_2 - x_1)(y_2 - y_1)} \times [R, G, B]_{Q12}$$
$$+ \frac{(x - x_1)(y_2 - y)}{(x_2 - x_1)(y_2 - y_1)} \times [R, G, B]_{Q21} + \frac{(x - x_1)(y - y_1)}{(x_2 - x_1)(y_2 - y_1)} \times [R, G, B]_{Q22}$$

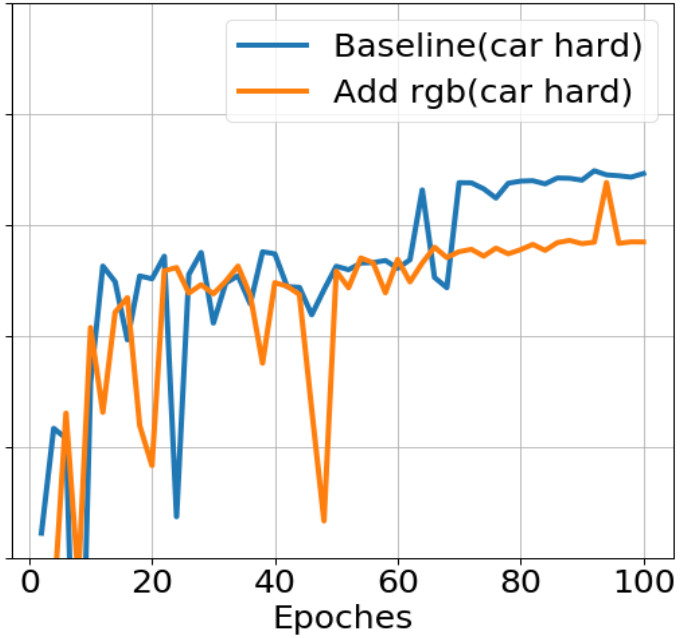
Evaluate PointRCNN after adding RGB data

Car (IoU=0.7)	Easy	Moderate	Hard
Baseline	84.37	75.76	74.65
Add RGB	83.93 ↓	74.88 ↓	68.49 ↓

AP for all the three difficult levels becomes worse, especially for the hard level







Why?

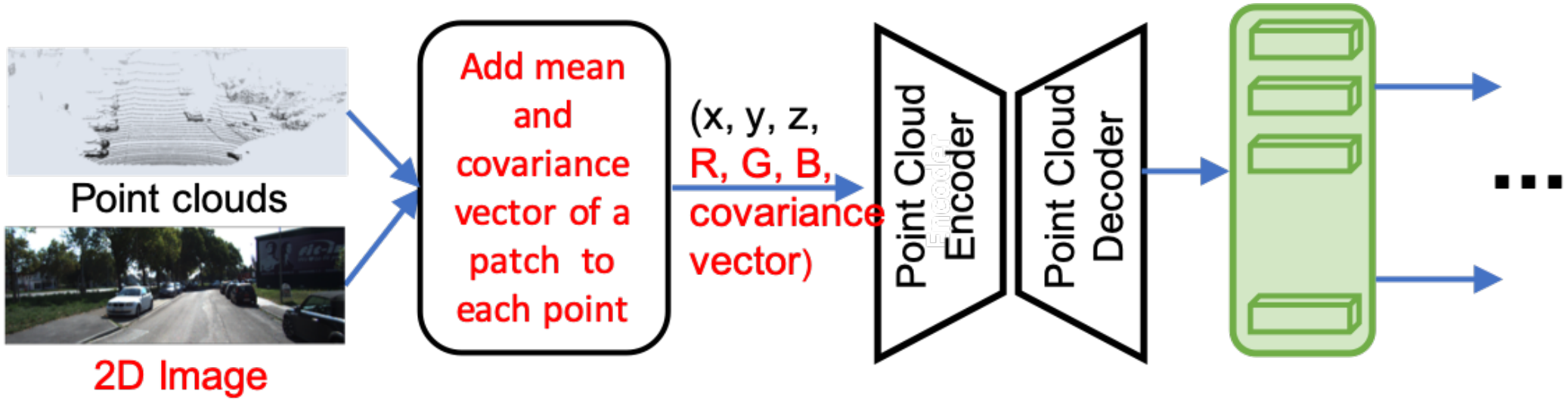
- Assumption1: the RGB value is not robust.

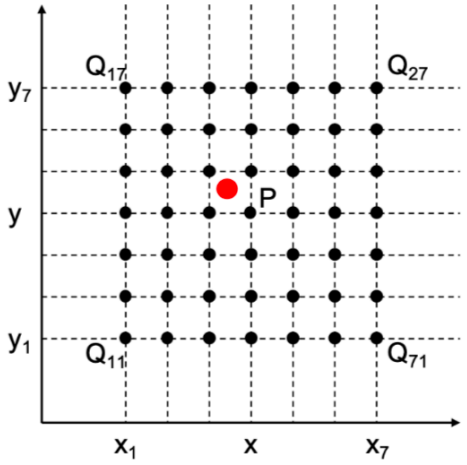
  For example, RGB value will be influenced by the illumination: when in the shadow, the color of the car may becomes similar to the color of the shadow of the tree.

- Assumption2: For hard level, the point cloud is few and sparse

  we only add sparse RGB values to the hard level cars and may induce some noise instead.

Method 2: Add Mean and Covariance(more robust)



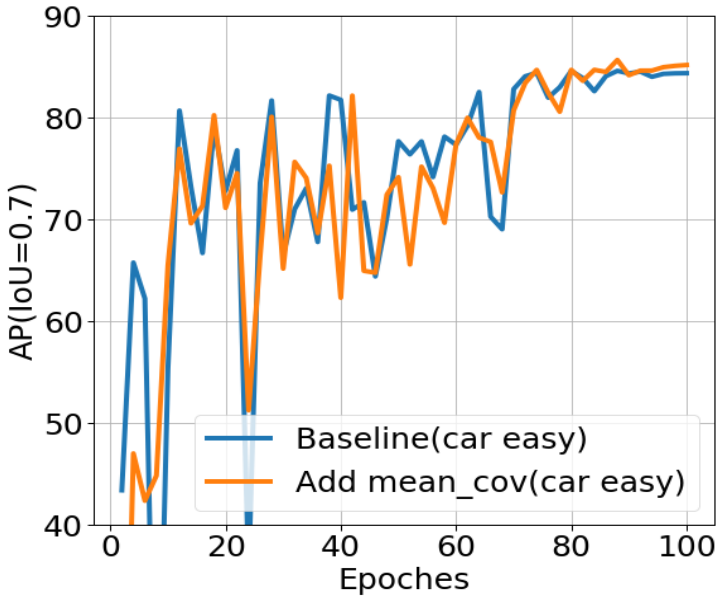


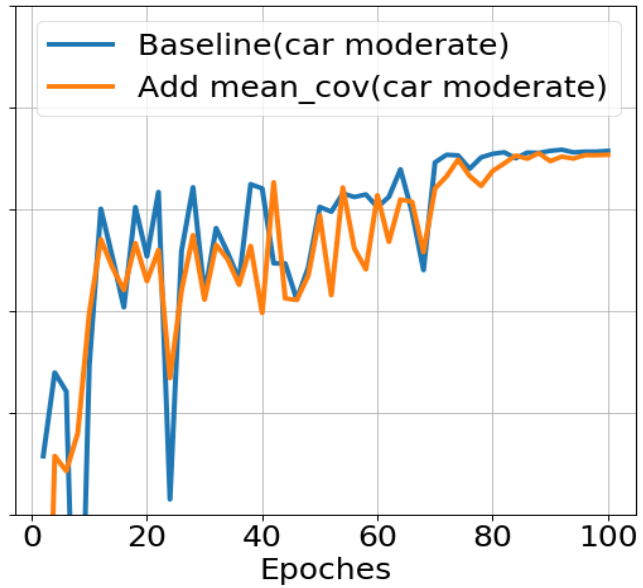
- patch size: 7x7
- Each patch has three channels R, G and B
- calculate the mean of each channel $\mathbb{E}[X] = \frac{1}{N}\sum_{i=0}^{N-1} X_i$
- calculate the covariance of three channels in pairs of two $Cov(X,Y) = \frac{1}{N}\sum_{i=0}^{N-1}(X_i - \mathbb{E}[X])(Y_i - \mathbb{E}[Y])$

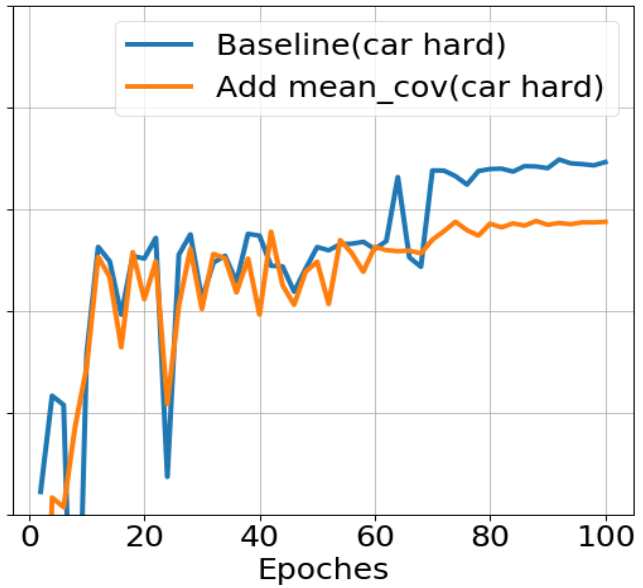
Evaluate PointRCNN after adding Mean and Covariance

Car (IoU=0.7)	Easy	Moderate	Hard
Baseline	84.37	75.76	74.65
Add RGB	83.93 ↓	74.88 ↓	68.49 ↓
Add mean and cov	85.17 ↑	75.37 ↓	68.78 ↓

- the performance of the easy level increases a little bit
- the performance of the hard level is almost as worse as adding RGB information

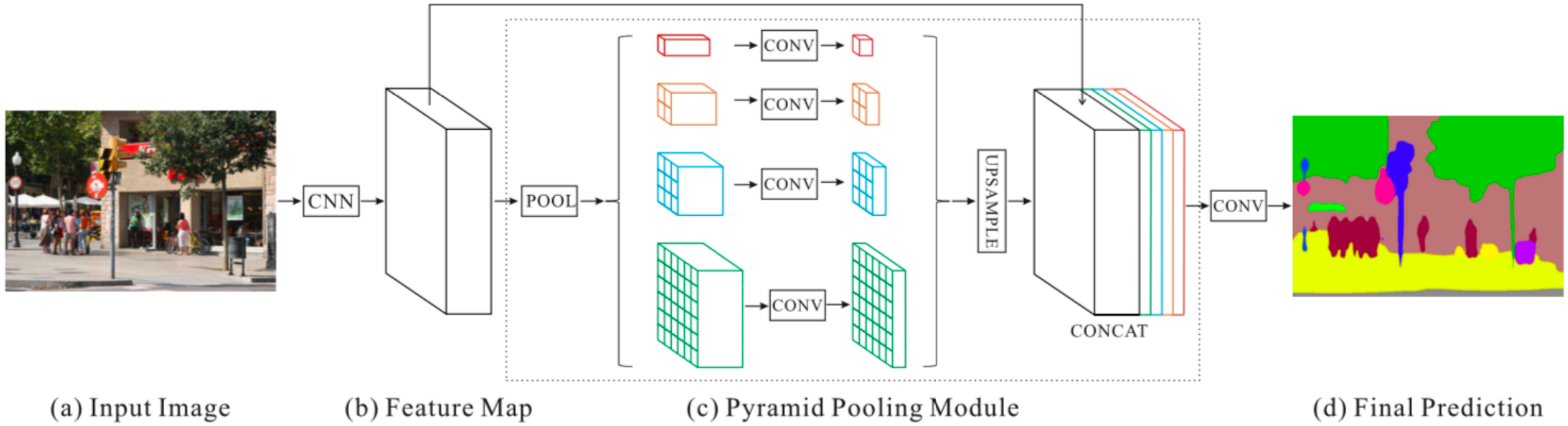






Method 3: Add Image Features

Use the PSPNet to extract the image features



Use CNN to get **global context information**

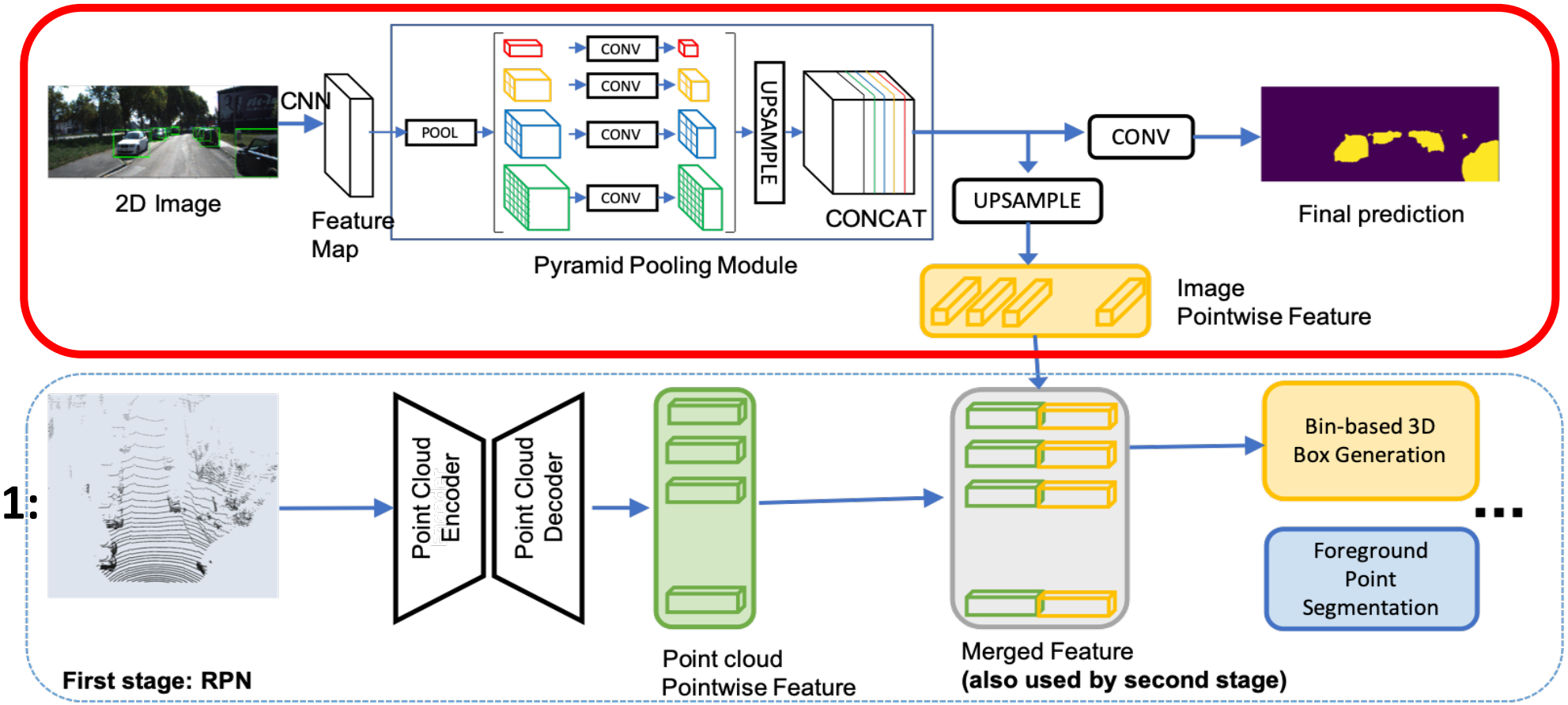
Use pyramid parsing module to get **local context information**

concatenated and fed into a conv layer to get the final pixel-wise prediction

Finetune: the last convolution layer of PSPNet (Pretrained on CityScapes) use the KITTI semantic segmentation dataset to segment cars from the background

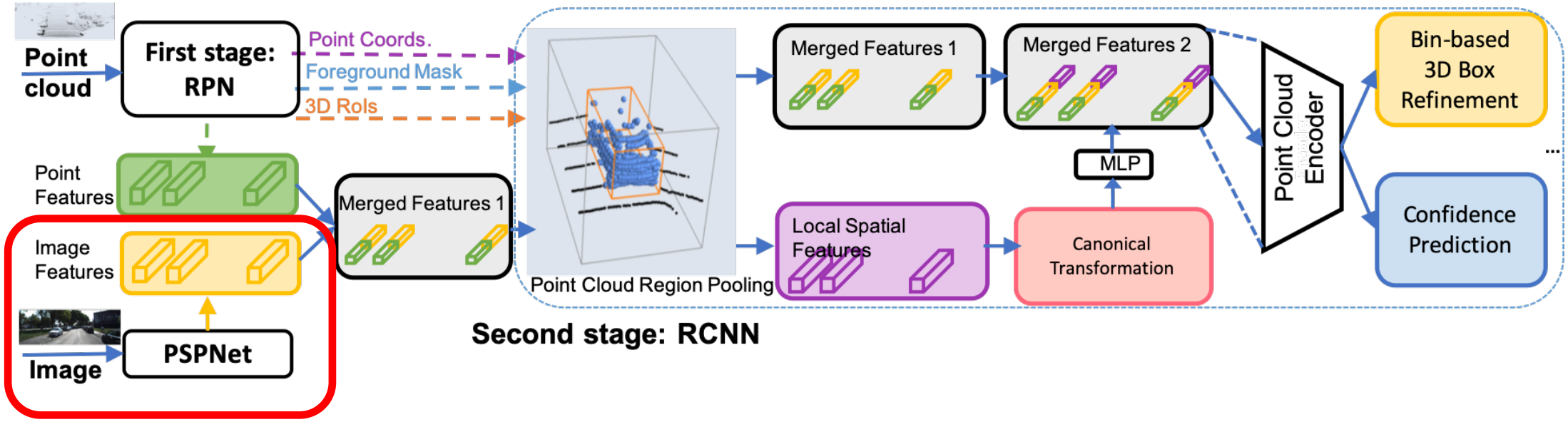
Method 3.1: Add Image Features to the first stage(RPN)

- add one more branch PSPNet to extract point-wise image features.
- concatenate the image features with point cloud features
- When training the PointRCNN model we will make the PSPNet fixed



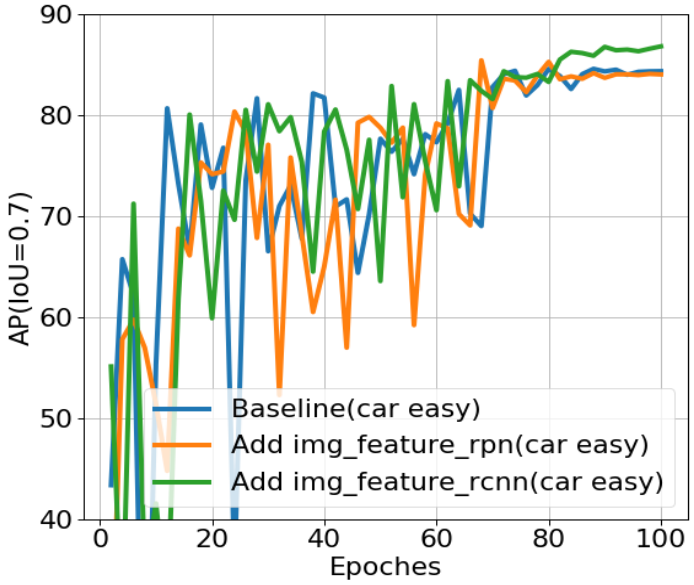
Method 3.2: Add Image Features to the Second stage(RCNN)

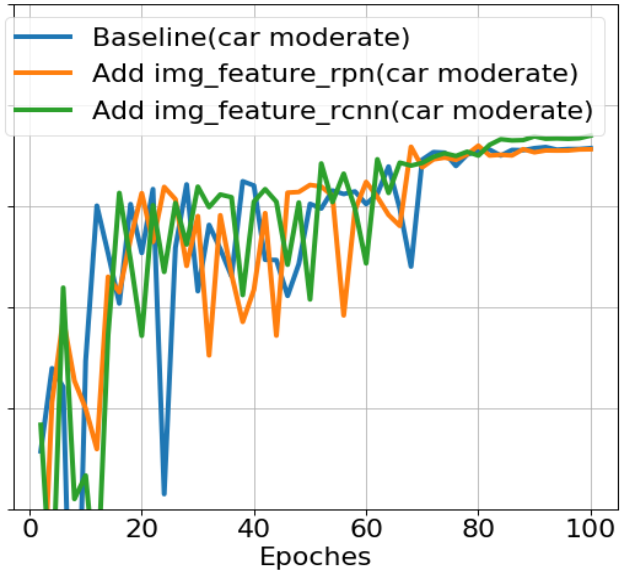
- leave the first stage RPN unchanged
- concatenate the image features with point cloud features before the first layer of the second RCNN stage
- train the PSPNet with the RCNN together

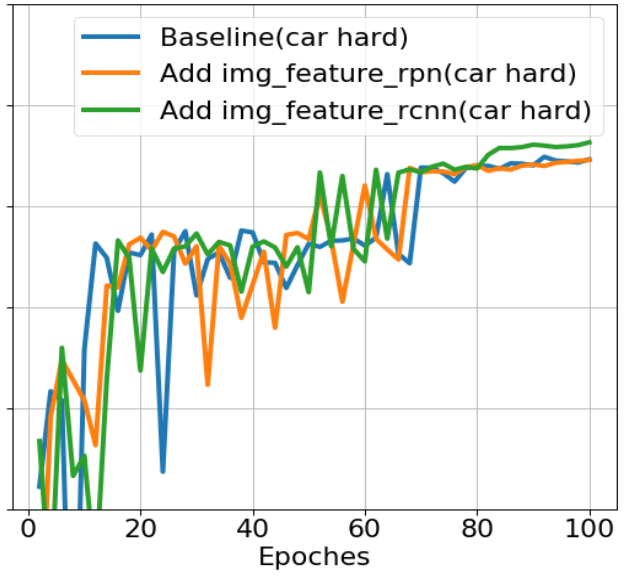


Evaluate PointRCNN after adding image features

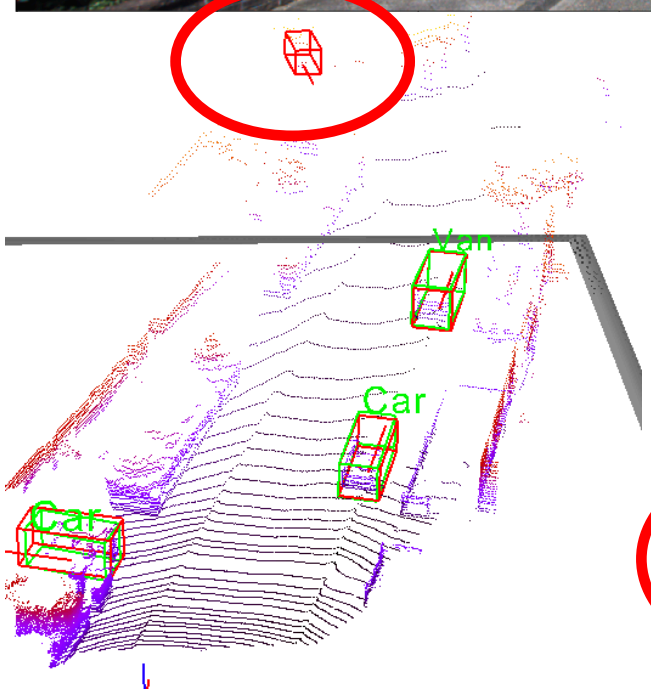
Car (IoU=0.7)	Easy	Moderate	Hard
Baseline	84.37	75.76	74.65
Add RGB	83.93 ↓	74.88 ↓	68.49 ↓
Add mean and cov	85.17 ↑	75.37 ↓	68.78 ↓
Add img feature rpn	84.05 —	75.64 —	74.58 —
Add img feature rcnn	**86.82** ↑	**76.99** ↑	**76.33** ↑



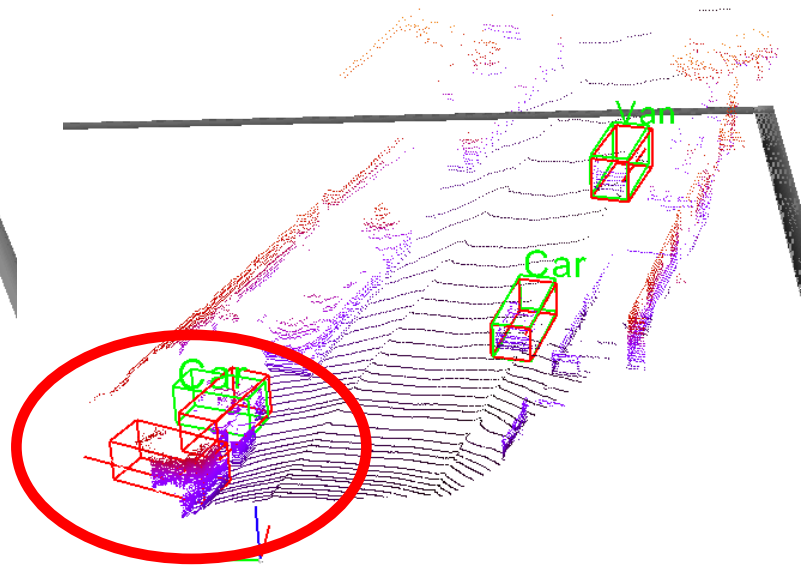




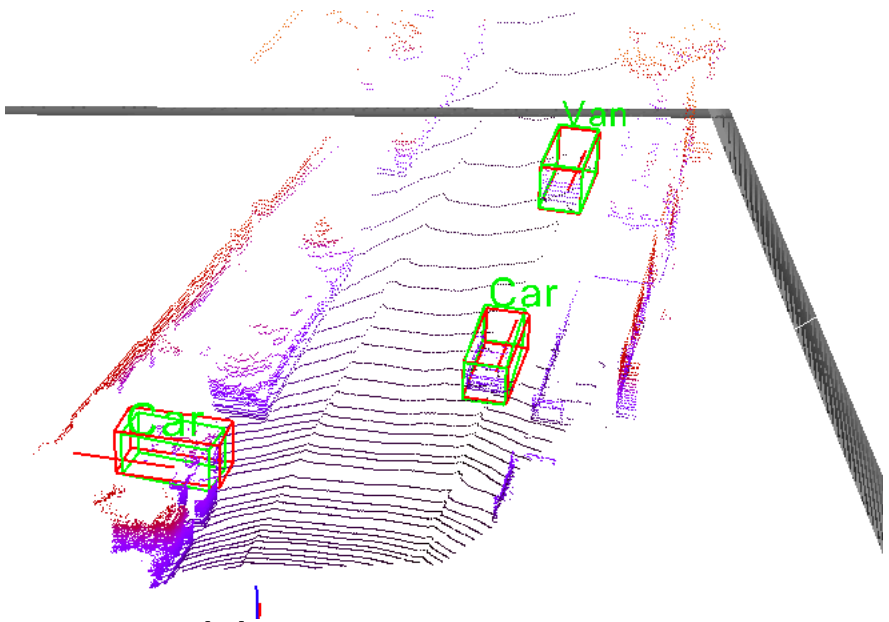
- Ground truth boxes are in green
- predicted boxes are in red.



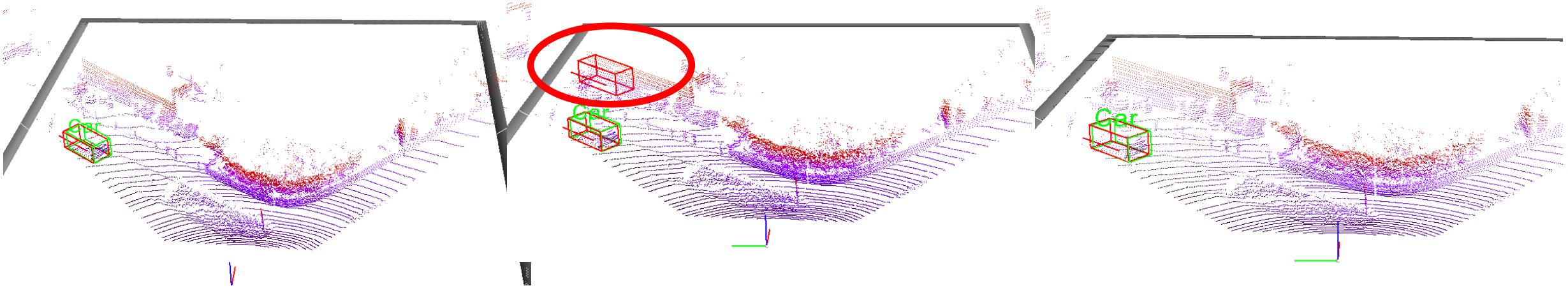
Baseline



Add RGB



Add img Feature RCNN



Baseline

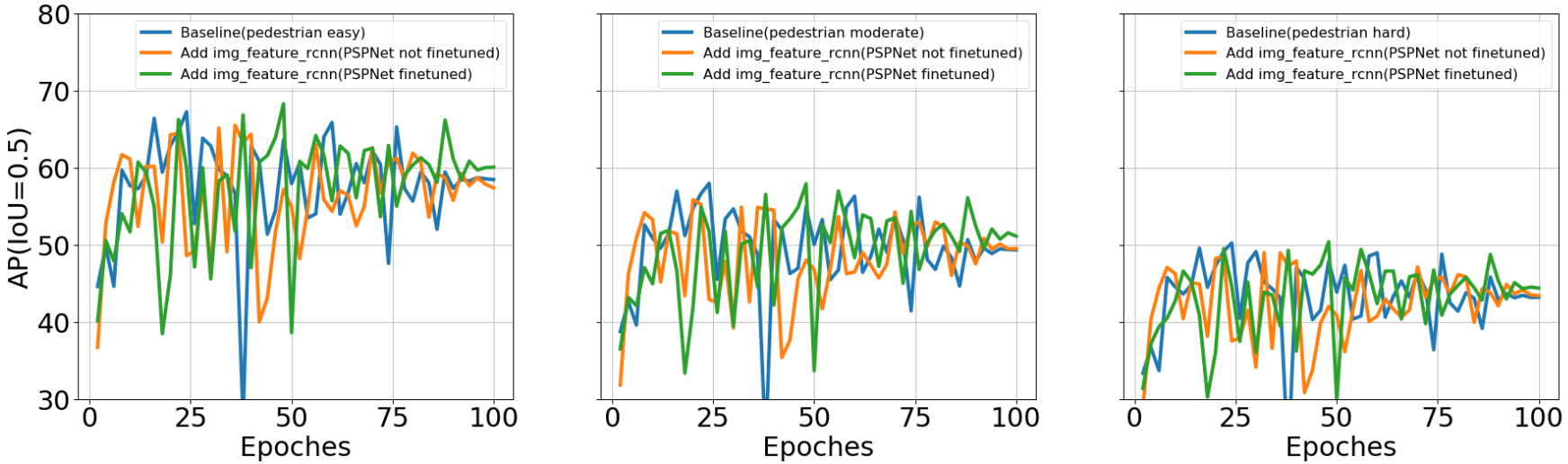
Add RGB

Add img Feature RCNN

Evaluate on Pedestrian (add image feature to RCNN)

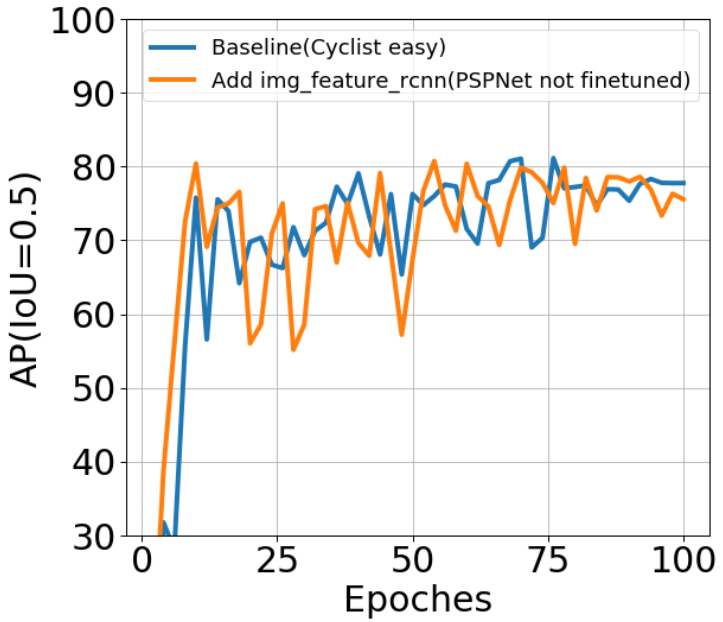
- **PSPNet not finetuned:** use the pretrained weights of PSPNet on CityScapes Dataset directly.
- **PSPNet finetuned:** finetune the last convolution layer of PSPNet use the KITTI semantic segmentation benchmark to segment pedestrian from the background.

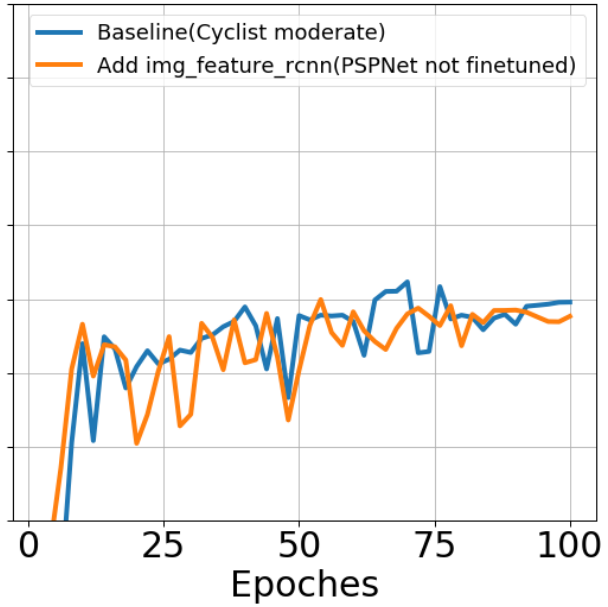
Pedestrian (IoU=0.5)	Easy	Moderate	Hard
PointRCNN(Baseline)	58.49	49.37	43.22
Add img feature rcnn (PSPNet not finetuned)	57.42 ↓	49.54 —	43.44 —
Add img feature rcnn (PSPNet finetuned)	**60.10** ↑	**51.14** ↑	**44.41** ↑

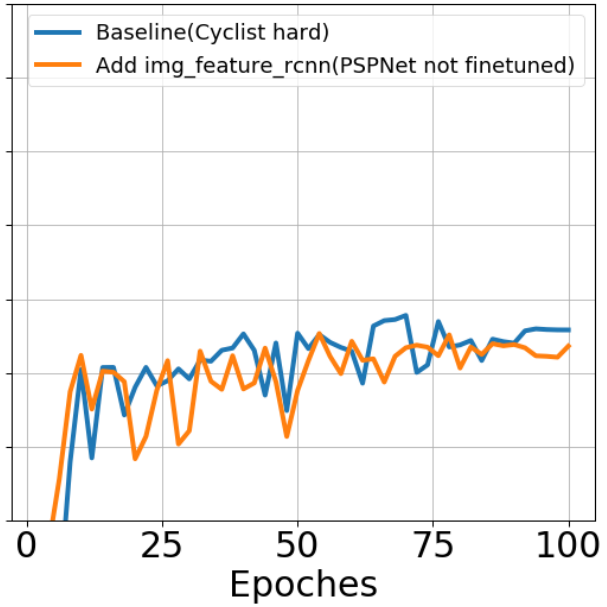


Evaluate on Cyclist (add image feature to RCNN)

Cyclist (IoU=0.5)	Easy	Moderate	Hard
PointRCNN(Baseline)	**77.77**	**59.60**	**55.83**
Add img feature rcnn (PSPNet not finetuned)	75.58 ↓	57.68 ↓	53.64 ↓







Conclusion

Adding simple image information can help very little or do harm to 3D object detection

- ***Add RGB***

The performance of all the three difficult levels becomes **worse**, especially for the hard level

- ***Add Mean and covariance***

The performance of **the easy level increases a little bit,** The performance of the hard level is almost as **worse** as adding RGB information

Adding well learned image features can improve the object detection performance

- ***Add finetuned Image Features (Fixed) to the first stage(RPN)***

The performance of all three levels is almost **the same** as the baseline

- ***Add finetuned Image Features (then train together with RCNN) to the Second stage(RCNN)***

The performance of **all the three difficult levels become better**.

- ***Add not finetuned Image Features (then train together with RCNN) to the Second stage(RCNN)***

The performance will become a littile bit **worse** or **similar** to baseline


THANKS
Questions?