RoarNet: A Robust 3D Object Detection based on RegiOn Approximation Refinement

Kiwoo Shin*†, Youngwook Paul Kwon*‡ and Masayoshi Tomizuka†

***Abstract*— We present RoarNet, a new approach for 3D object detection from 2D image and 3D Lidar point clouds. Based on two stage object detection framework ([1], [2]) with PointNet [3] as our backbone network, we suggest several novel ideas to improve 3D object detection performance.**

The first part of our method, RoarNet_2D, estimates the 3D poses of objects from a monocular image, which approximates where to examine further, and derives multiple candidates that are geometrically feasible. This step significantly narrows down feasible 3D regions, which otherwise requires demanding processing of 3D point clouds in a huge search space.

Then the second part, RoarNet_3D, takes the candidate regions and conducts in-depth inferences to conclude final poses in a recursive manner. Inspired by PointNet, RoarNet_3D processes 3D point clouds directly without any loss of data, leading to precise detection.

We evaluate our method in KITTI, a 3D object detection benchmark. Our result shows that RoarNet has superior performance to state-of-the-art methods that are publicly available. Remarkably, RoarNet also outperforms state-of-the-art methods even in settings where Lidar and camera are not time synchronized, which is practically important for actual driving environment.

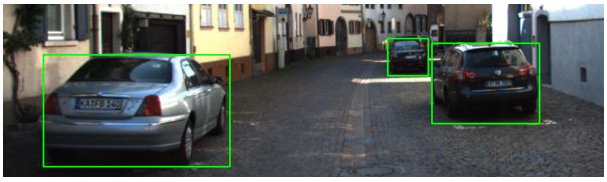
RoarNet is implemented in Tensorflow [4] and publicly available with pretrained models.

I. Introduction

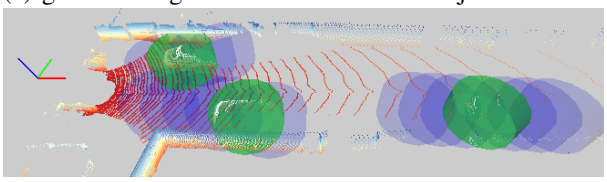
Recently, 3D object detection has become a crucial component in various fields such as mobile robots and autonomous vehicles. 3D object detection helps to understand the geometry of physical objects in 3D space that are important to predict future motion of objects. While there has been remarkable progress in the fields of image based 2D object detection and instance segmentation, 3D object detection is less explored in the literature. In this work, we study 3D object detection, which predicts 3D bounding boxes of objects from 2D image and 3D point clouds.

Most current 3D object detection systems transform 3D point clouds into 2D images by projecting point clouds onto ground plane (Bird's Eye View) and/or depth map (Perspective View). These systems apply convolutional neural networks on those transformed images to detect objects. Those approaches often rely on sensor-fusion methods to compensate the loss of data that occurs during projecting 3D point clouds onto lower dimensional 2D planes. However, these sensor-fusion based approaches require high quality synchronization between 2D camera sensor and 3D Lidar sensor, which itself is very challenging due to different sensor operating frequencies. When the synchronization condition breaks down, the performance of 3D object detection degrades significantly (Section IV-A).

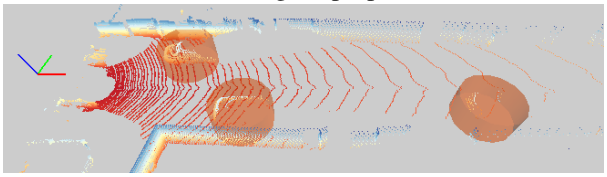
Recently, [5] predicts objects as 2D rectangular bounding boxes on the image plane and extend those boxes into 3D space along projection lines in the form of frustum. This makes it possible to filter the most of 3D point clouds out that are irrelevant to objects, and to process only those 3D point clouds that belong to objects directly without transforming the points to the 2D image plane. However, this approach is also sensitive to synchronization quality between sensors.



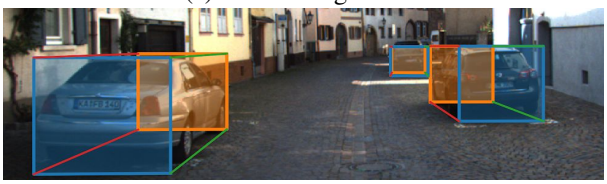
(a) geometric agreement search on 2D object detection



(b) 3D region proposals



(c) 3D box regression



(d) resulting 3D bounding boxes

Fig. 1: **Detection pipeline of RoarNet**. Our model (a) predicts region proposals in 3D space using geometric agreement search, (b) predicts objectness in each region proposal, (c) predicts 3D bounding boxes, (d) calculates IoU (Intersection over Union) between 2D detection and 3D detection.

The authors are with the department of Mechanical Engineering, University of California, Berkeley, CA 94720, US. {kiwoo.shin, young, tomizuka}@berkeley.edu

*The authors contributed equally.

†Mechanical Systems Control Lab, University of California, Berkeley, CA, USA.

‡Phantom AI Inc., CA, USA.

In this work, we propose a robust 3D detector, named *RoarNet* (RegiOn Approximation Refinement Network), which helps to improve 3D object detection performance and reduce problems caused by sensor synchronization issue. RoarNet consists of two parts: RoarNet_2D and RoarNet_3D.

Inspired by geometric interpretation for monocular images in [6], RoarNet_2D estimates the 3D poses of objects from a monocular image and derives multiple candidate locations that are geometrically feasible, where the candidates are the input for RoarNet_3D. This scheme significantly narrows down feasible 3D regions, which otherwise requires demanding processing of 3D point clouds in a huge search space (Section III-A).

Obtaining 3D region proposals predicted from 2D image, RoarNet_3D, a two-stage 3D object detector, gradually refines a search space making its training process efficient. The architecture of our model is analogous to standard two stage object detectors for 2D image such as Fast-RCNN and Faster-RCNN [1], [2], and we adopt several modifications in order to make training of each stage easier (Section III-B).

The key difference compared to [5] is that our model does not filter out point clouds by using 2D bounding box. Instead, our model takes the whole point clouds that are located inside region proposals which have the shape of standing cylinders. This leads to our model being more robust to sensor synchronization than state-of-the-art methods. We compare our method to other state-of-the-art 3D detection models in both synchronized and asynchronized conditions in Section IV-A.

The detection pipeline of our model consists of three components as in Figure 1: (a) From a 2D image, our model predicts region proposals in 3D space. There can be multiple region proposals for a single detected object. (b) Using 3D point clouds sampled from the region proposals, we predict objectness in order to remove region proposals without foreground objects. At this step, we also predict the location of an object relative to given region proposals. We recursively use the relative location prediction as the center of region proposals for the next detection step. (c) Finally, our model predicts all coordinates for 3D bounding box regression including location, rotation and size. Practically, we repeat this step twice for better performance. (d) To evaluate confidence of each detection, we calculate IoU (Intersection over Union) between the 2D detection and the 3D detection projected onto 2D image. The higher the correspondence between 2D detection and 3D detection is, the higher the confidence of detection is.

We evaluate our model on the 3D object detection task, provided by the KITTI benchmark. Our experiments show that RoarNet outperforms the state-of-the-art 3D object detection methods that are publicly available. We also evaluate our model in settings where camera and the Lidar are not time synchronized and the result shows that our model consistently performs better in these challenging settings.

All codes are implemented in Tensorflow and Cython and publicly available with several pretrained models. Additional materials are also available in https://sites.google.com/berkeley.edu/roarnet.

II. Related Work

Monocular pose estimation Due to the projection characteristics of camera sensors, monocular 3D pose estimation is very challenging. To overcome such difficulty, previous works often rely on domain knowledge or external data/information. For example, human pose estimation applications were approached using a tracker ([7]), through transferred learning from 2D and 3D datasets ([8]) combined with the known skeleton topology of a human body. In autonomous driving applications, [9] trains a network to predict 36 control points per each vehicle that conveys 3D shape information. However, this method requires additionally annotating the auxiliary control point, which are very expensive to obtain. [6] proposes a novel method to predict physical dimensions (i.e, height, width, length in meters) and an orientation of vehicle without any additional data. Then, it can predict the location of object (i.e., X, Y, Z in the world coordinate) by solving an over-constrained system of linear equations system. Since we find this method useful, we explore the method in more detail in Section III-A where we modify the method to be more computationally efficient and use it as our first building block for predicting region proposals in 3D space from a 2D image.

3D point clouds processing Since autonomous driving applications require very high level of accuracy in 3D pose estimation that monocular algorithms cannot provide, many algorithms using Lidar sensors are proposed. There are three popular representations to handle unstructured point clouds: (1) The first representation is using a 3D voxel grid [10], [11], [12]. In autonomous driving applications, however, sparse points clouds generally make voxel representation computationally redundant. (2) The second is to project an point cloud onto one or more 2D planes [13], [14], [15]. These representations are usually compact and efficient, and can be treated as images. However, information loss by projection is inevitable. (3) The third one is to use the point clouds directly without any structured form. PointNet [3], [16] showed how to digest point clouds directly for object classification and segmentation, and Frustum PointNet (F-PointNet) [5] selects only necessary 3D points utilizing 2D detection results (i.e., 3D points within a frustum region that a camera position and a 2D bounding box make), and conducts detection using a PointNet scheme.

Similar to F-PointNet, advanced algorithms use both images and point clouds in a sensor fusion manner to enhance performance [17], [18]. Among these, F-PointNet and Aggregate View Object Detection (AVOD) [17] show the state-of-the-art performance on the public KITTI dataset leader board. RoarNet outperforms these methods in the standard 3D object detection, and our analysis shows that RoarNet shows better robustness in an even more general setting.

III. Designing a RoarNet Detector

The main idea behind RoarNet is to construct sequential networks that gradually refines a search space at each step

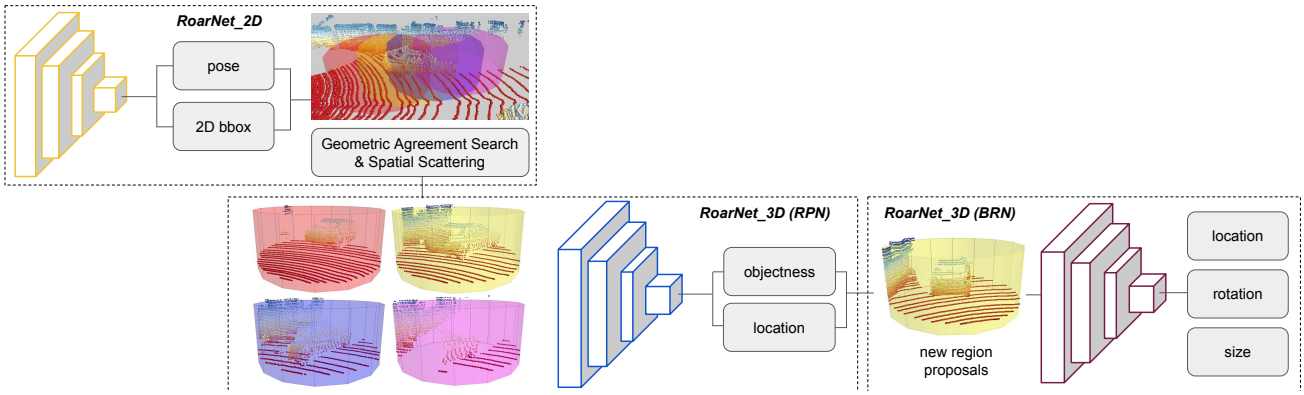


Fig. 2: Architecture of RoarNet

in order to assign each network a simple task, and thus leads to efficient training and prediction.

Figure 2 shows the architecture of RoarNet. The model first predicts the 2D bounding boxes and a 3D poses of objects from a 2D image. For each 2D object detection, geometric agreement search is applied to predict the location of object in 3D space. Centered on each location prediction, we set region proposal which has a shape of standing cylinder. Taking the prediction error in bounding box and pose into account, there can be multiple region proposals for a single object.

Each region proposal is responsible for detecting a single object. Taking the point clouds sampled from each region proposal as input, our model predicts the location of an object relative to the center of region proposal, which recursively serves for setting new region proposals for the next step. Our model also predicts objectness score which reflects the probability of an object being inside the region proposal. Only those proposals with high objectness scores are considered at the next step.

At a final step, the model sets new region proposals at previously predicted locations. Our model predicts all coordinates required for 3D bounding box regression including location, rotation, and size of the objects. For practical reason, we observe that repeating this step more than once gives better detection performance.

In Section III-A, we explain RoarNet_2D that bridges image based 2D object detection to point clouds based 3D object detection. In Section III-B, we describe RoarNet_3D, which predicts 3D bounding box using point clouds.

A. *RoarNet_2D*

Geometric agreement search For our initial seeds of 3D region proposals, we utilize a method suggested by [6] for monocular pose estimation, which we call *geometric agreement search*: Given that the 3D pose of an object can be represented by seven degrees of freedom (localization in the camera coordinate X, Y, Z, physical dimensions of width, height and length W, H, L, and heading angle Θ), a 2D bounding box window and the projection of its 3D pose (i.e., 3D box formed by X, Y, Z, W, H, L, Θ and camera projection matrix P) should agree. [6] showed that (1) a network can regress $\{W, H, L, \Theta\}$ per object, (2) there are only finite number of possible combinatorial configurations that a 3D box can locate to tightly fit a given 2D box, and (3) at each configuration, translation X, Y, Z can be solved from known (regressed) W, H, L, Θ using an over-constrained system of linear equations. Then, the best configuration that minimizes projection error is selected.

More formally, for an object, let b_{2D} be its 2D bounding box (from a 2D detector). At each configuration c, one can calculate a 3D bounding box candidate b^c_{3D} as

$$b^c_{3D} = B(W, H, L, \Theta; c, b_{2D}) \tag{1}$$

where B is the over-constrained linear equation system aforementioned. The best configuration c^* can be obtained by checking the agreement between b_{2D} and the projection of 3D box b^c_{3D}.

$$b^c_{PROJ} = T(b^c_{3D}; P) \tag{2}$$

$$c^* = \arg\max_{c \in C} \text{IoU}(b_{2D}, b^c_{PROJ}) \tag{3}$$

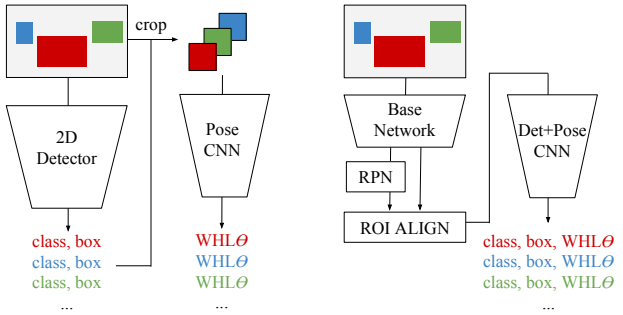
where T is projective transformation onto the image coordinate, IoU is a widely-used intersection-over-union measure, and C is the finate configuration set.[1]

One drawback of [6] is that the $\{W, H, L, \Theta\}$ inference and inverse projection process should be done after running a separate 2D object detection and should be conducted for each detected vehicle. In other words, when an image includes k objects, there should k-time computation of the network.

Aiming better computation efficiency, we build an unified network that combines the 2D object detection and $\{W, H, L, \Theta\}$ inference as illustrated in Figure 3b. In other words, the 2D bounding boxes and $\{W, H, L, \Theta\}$s of k objects can be inferred with only one forward calculation of the unified network.

Spatial scattering Note that the role of RoarNet_2D, as a 3D region proposer, is to provide proposals of higher recall. Since the monocular pose estimation suffers from limited

[1]We refer [6] for the details about the configuration set C, and the over-constrained system of linear equations B.



(a) Architecture in [6] (b) RoarNet_2D architecture

Fig. 3: Architecture of RoarNet_2D

accuracy, it is necessary to scatter our initial monocular pose estimation in order to increase the number of feasible pose candidates, and therefore, increase recall: For each object (i.e., its bounding box b_{2D}, regressed pose $XYZWHL\Theta$, and the best configuration c^*), we first set a *scattering range* by considering two extreme cases where the true physical size could actually be $1-s$ times smaller and $1+s$ times larger than the regressed size WHL $(0 < s < 1)$, which results in differently located 3D boxes by Equation (1):

$$b_{3D}^{c^*\ small} = B((1-s)W, (1-s)H, (1-s)L, \Theta; c^*, b_{2D})$$
$$b_{3D}^{c^*\ large} = B((1+s)W, (1+s)H, (1+s)L, \Theta; c^*, b_{2D}).$$

Recall that Equation (1) means the geometric constraint that the projection of the 3D box of an object should match with its 2D box, i.e., for the same 2D bounding box, smaller 3D boxes result in closer locations to the camera origin. Given these two extreme boxes, we divide the line of their two center points, $p1$ and $p2$, into an equal stride distance m. RoarNet_2D detector finally provides $\lceil \|p_1 - p_2\|/m \rceil$ 3D points per object for RoarNet_3D to start.[2]

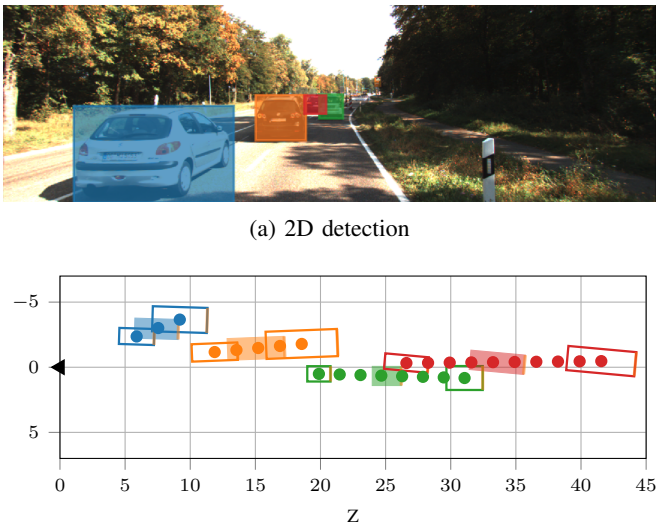
We visualize the process of RoarNet_2D detector in Figure 4. RoarNet_2D detector predicts 2D bounding boxes (Figure 4a) as well as their physical sizes WHL and heading angles Θ, which lead to calculate their positions XYZ (color-filled boxes in Figure 4b). For each object, we consider two extreme deviations (non-filled boxes in Figure 4b), and collect the uniform linear subdivision between the center points of the extreme poses (colored dots in Figure 4b).

Note that the geometric agreement search and spatial scattering scheme significantly narrows down feasible 3D regions into a few linear regions, which otherwise requires a huge search space. Moreover, by virtue of geometric agreement constraints, our resulting proposals natively distribute (1) along the projection rays of the camera, and (2) in larger areas for more challenging further objects without bells and whistles.

B. RoarNet_3D

Network architecture The RoarNet_3D is designed to predict a 3D bounding box that optimally fits for a given

[2] $s = .5, m = 1.6$ for experiments; $s = .2, m = 1.25$ for Figure 3.



(a) 2D detection

(b) Geometric agreement search and spatial scattering

Fig. 4: **RoarNet_2D**. An unified architecture detects 2D bounding boxes and 3D poses illustrated as color-filled boxes in (a) and (b), respectively. For each object, two extreme cases are shown as non-filled boxes, and final equally-spaced candidate locations as colored dots in (b). All calculations are derived in 3D space despite bird's eye view (i.e., XZ plane) visualization.

object by using point clouds. While building RoarNet_3D as a two-stage object detector, the backbone network is inspired by the PointNet[3], which uses max-pooling layers in the middle to get a global feature directly from unstructured point clouds. For more details, we refer readers to [3], [5], [16]. In this work, we use a simplified version of PointNet shown in Figure 5.

RoarNet_3D consists of two networks, called RPN (region proposal network) and BRN (box regression network), those have same structure except for the number of output as shown in Figure 5 and Table I.

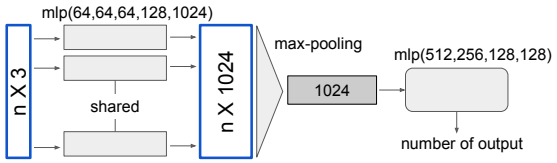


Fig. 5: **Our backbone network** is a simplified version of PointNet without T-Net in the original paper [3].

Number of Outputs	RPN	BRN
location	3	3
rotation	0	2*N_R
size	0	4*N_C
objectness	1	0

TABLE I: Number of output at each network

The location is predicted by 3 coordinates (t_x, t_y, t_z) for

(x, y, z) directions which is relative to center of region proposals. If a center of region proposal is offset from the origin by (c_x, c_y, c_z), then the location prediction corresponds to:

$$\begin{aligned} x &= c_x + 2 * (\sigma(t_x) \text{ - } 0.5) * m_x, \\ y &= c_y + 2 * (\sigma(t_y) \text{ - } 0.5) * m_y, \\ z &= c_z + 2 * (\sigma(t_z) \text{ - } 0.5) * m_z \end{aligned} \tag{4}$$

We constrain the location prediction be bounded by (m_x, m_y, m_z) from center of region proposal.

The rotation angle is predicted by $2{*}N_R$ coordinates $(t_{r\_\text{cls}(i)}, t_{r\_\text{reg}(i)})_{i=1}^{N_R}$ which is a hybrid formulation of <cls+reg> structure. We equally divide [0, pi) to N_R bins.

The size is predicted by $4{*}N_C$ coordinates, $(t_{size\_cls(i)}, t_{h(i)}, t_{w(i)}, t_{l(i)})_{i=1}^{N_C}$ which is also a hybrid formulation of <cls+reg> structure. We use K-Means method to get N_C clusters.

The objectness is predicted by the output t_o which reflects the probability of object or not object for each region proposal. We use sigmoid function to bound its value in a range of [0.0, 1.0).

C. Training and prediction

During training each network, we optimize the following multi-task loss for RPN and BRN:

$$\begin{aligned} L_{\text{RPN}} &= \lambda_{\text{obj}} * L_{\text{obj}} + \mathbb{1}^{\text{obj}}[L_{\text{loc}}], \\ L_{\text{BRN}} &= \mathbb{1}^{\text{3D IoU}<0.8}\Big[L_{\text{loc}} + L_{\text{rot-cls}} + \mathbb{1}^{\text{rot-cls}}[L_{\text{rot-reg}}] \\ &\qquad + L_{\text{size-cls}} + \mathbb{1}^{\text{size-cls}}[L_{\text{size-reg}}]\Big] \end{aligned} \tag{5}$$

$L_{\text{loc}}, L_{\text{rot-reg}}$, and $L_{\text{size-reg}}$ are regression loss for location, rotation and size, which are represented as huber loss. $L_{\text{obj}}, L_{\text{rot-cls}}$, and $L_{\text{size-cls}}$ are classification loss for objectness, rotation and size, which are represented as cross-entropy loss. $\mathbb{1}^{\text{obj}}$ denotes if objectness is true for a given region proposal. $\mathbb{1}^{\text{3D IoU}<0.8}$ is used for improving prediction performance for more general cases.

We down-sample point clouds with resolution of 0.1m for each axis. At each region proposal, we randomly sample 256 point clouds for training and 512 point clouds for prediction.

We train each network with batch of 512 for 500k iterations. Learning rate is 5e-3 for initial 100k and 5e-4 for rest of steps. It takes about two days for training each network with Titan X (not pascal).

Non-maximal suppression (NMS) is used to reduce redundant prediction at testing. We apply NMS on bird's eye view boxes with threshold of 0.05 to remove overlapping objects.

IV. Experiments

Dataset We conduct our experiments in KITTI dataset, the 3D object detection benchmark. It provides synchronized 2D images and 3D LiDAR point clouds with annotations for car, pedestrian, and cyclist class. In this work, we focus on car class which has most training examples. The detection results are evaluated based on three difficulty levels: easy, moderate, and hard and we evaluate on moderate level, a standard metric for performance evaluation. 3D object detection performance is evaluated at 0.7 IoU threshold.

Method	Easy	Moderate	Hard
MV3D [20]	71.09	62.35	55.12
VoxelNet [21]	77.47	65.11	57.73
UberATG-ContFuse [18]	82.54	66.22	64.04
F-PointNet (v2) [5]	81.20	70.39	62.19
AVOD (FPN) [17]	81.94	71.88	**66.38**
Ours	**83.71**	**73.04**	59.16

TABLE II: 3D object detection performance publicly available on the KITTI *test* set, with 3D IoU threshold of 0.7

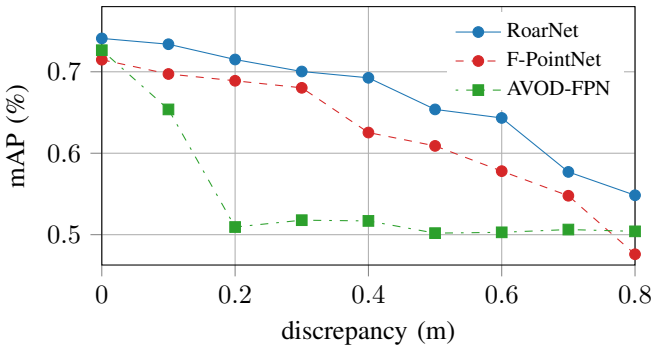


Fig. 6: A comparison of the 3D object detection performance in where Lidar and camera are not time synchronized.

Following [5], [17], [19], we split training set into `train` set of 3,717 frames and `val` set of 3,769 frames such that frames in each split belong to different video clips.

A. Comparison of the 3D object detection performance

Experiment settings We evaluate our method in two settings. First, we evaluate our method in the original KITTI evaluation setting where the Lidar and the camera are well-synchronized each frame. This is a standard metric for ranking in KITTI benchmark leaderboard. Second, we evaluate our method in a more general case where the two sensors are not synchronized. To simulate such case, we randomly translate the whole point clouds and re-generate ground truth labels according to the amount of translation of point clouds. This means that we regard the Lidar as the primary sensor. We constrain the translation of point clouds within 0.8m for x, y axis (i.e., parallel to the ground plane) and 0.2m for z axis (i.e., orthogonal to the ground plane).

Experiment results First, we evaluate RoarNet in a setting where the Lidar and the camera are synchronized, and compare it to publicly available 3D object detection methods on the KITTI benchmark. Table II shows that RoarNet shows state-of-the-art performance for 3D object detection in both easy and moderate level metric.

Second, we compare RoarNet to the two state-of-the-art methods, AVOD (FPN) and F-PointNet (v1) in a setting where sensors are *not* synchronized. Those methods are selected since the AVOD (FPN) is the best among sensor-fusion based methods [17] and the F-PointNet (v1) is the best among methods that directly process 3D point clouds [5],

[21]. [3]

Figure 6 shows that RoarNet performs better than two state-of-the-art methods when two sensors are not synchronized. When sensors are synchronized, all three methods show the recall of 82.5%. When two sensors are asynchronized by 0.8m, the recall of our model degrades to 72.5%, while the recall of F-PointNet degrades to 67.5% and the recall of AVOD (FPN) degrades to 65%.

B. Region proposals analysis

In this section, we analyze the effect of spatial scattering parameter s and objectness threshold in RoarNet_3D (RPN) for refining a search space, as shown in Figure 7.

The smaller the value s, the higher confidence we have on monocular pose estimation. However, only 26.3% of objects are captured in region proposals when we predict the location of object directly from monocular pose estimation ($s = 0$). As we increase s, more objects are captured in region proposals, but number of region proposals are also linearly increased, which becomes the bottleneck of our detection pipeline. Aiming high recall, we use $s = 0.5$ in our implementation.

The search space is further refined by RoarNet_3D (RPN). In our implementation, we use objectness threshold of 0.25, that gives 83.2% of recall with less than two region proposals per ground truth object.

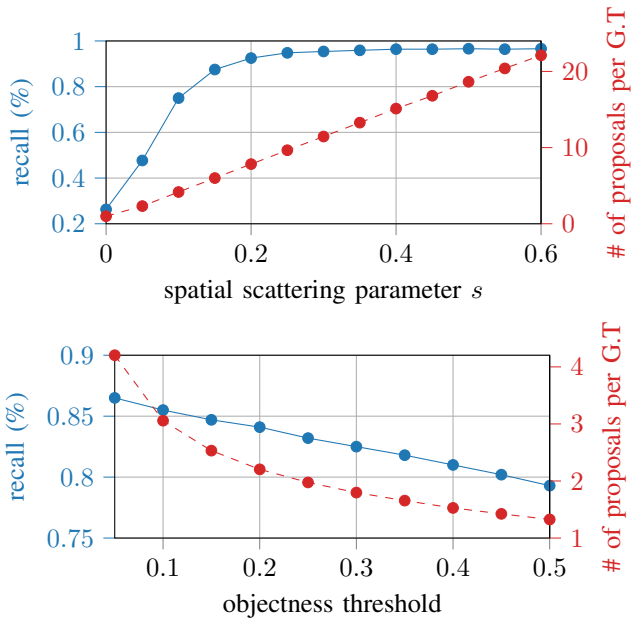
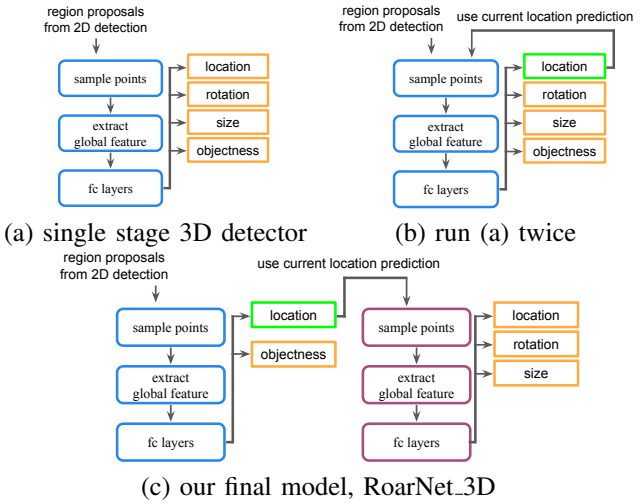


Fig. 7: The effect of spatial scattering parameter s and objectness threshold

C. Network design analysis

In this section, we compare three network architectural designs shown in Figure 8.

[3]We train all methods for car class only. All methods are trained and evaluated in same train/val split.



(a) single stage 3D detector (b) run (a) twice

(c) our final model, RoarNet_3D

Fig. 8: A detection pipeline of several network architectures

Figure 8(a) represents a single stage 3D object detector, which predicts 3D bounding box along with objectness in a single step. This approach is inspired by YOLO detector [22], [23], which shows promising results in a 2D object detection. However, (a) shows the recall of 67.5% and mAP of 54.3%.

Without any further training step, we only modify the detection pipeline of (a) to use location predicted at current step as region proposals for the next step. This simple modification immediately improves the performance to 59.9% with an increase of 5.6% from (a).

This result inspires us to build our final model, RoarNet_3D in Figure 8(c) that specializes each detection step to a specific task and remove redundant predictions. This modification leads significant performance improvement such that recall is 82.5% and mAP is 74.02%.

V. Conclusion

We have proposed RoarNet, a new approach for 3D object detection from an 2D image and 3D Lidar point clouds. RoarNet refines search space recursively at each step in order to make training and prediction efficient. We first estimate 3D poses from a monocular input image, and derives multiple geometrically feasible candidates nearby the initial estimates. We adopt a two-stage object detection framework to further refine search space effectively from 3D point clouds. Our model shows superior performance to state-of-the-art methods in KITTI, a 3D object detection benchmark. RoarNet outperforms even in the setting where Lidar and camera are not time synchronized, which is practically important results in order to extend current single frame based detection into video frame based detection in the future research.

ACKNOWLEDGMENT

The work was in part supported by Berkeley Deep Drive. Kiwoo Shin is supported by Samsung Scholarship.

REFERENCES

[1] R. Girshick, "Fast R-CNN," in *IEEE International Conference on Computer Vision*, 2015.

[2] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *Neural Information Processing Systems*, 2015, pp. 1–10.

[3] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017.

[4] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," in *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, ser. OSDI'16. Berkeley, CA, USA: USENIX Association, 2016, pp. 265–283. [Online]. Available: http://dl.acm.org/citation.cfm?id=3026877.3026899

[5] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum PointNets for 3D Object Detection from RGB-D Data," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018.

[6] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, "3D Bounding Box Estimation Using Deep Learning and Geometry," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017.

[7] M. Andriluka, S. Roth, and B. Schiele, "Monocular 3d pose estimation and tracking by detection," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2010.

[8] D. Mehta, H. Rhodin, D. Casass, O. Sotnychenko, W. Xu, and C. Theobalt, "Monocular 3d human pose estimation in the wild using improved cnn supervision," in *International Conference on 3D Vision*, 2017.

[9] F. Chabot, M. Chaouch, J. Rabarisoa, and T. Chateau, "Deep MANTA: A Coarse-to-fine Many-Task Network for joint 2D and 3D vehicle analysis from monocular image," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017.

[10] D. Zeng Wang and I. Posner, "Voting for Voting in Online Point Cloud Object Detection," in *Robotics: Science and Systems XI*. Robotics: Science and Systems Foundation, Jul. 2015.

[11] B. Li, T. Zhang, and T. Xia, "Vehicle Detection from 3D Lidar Using Fully Convolutional Network," in *Robotics: Science and Systems*, 2016.

[12] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner, "Vote3Deep: Fast Object Detection in 3D Point Clouds Using Efficient Convolutional Neural Networks," in *IEEE International Conference on Robotics and Automation*, 2017.

[13] W. Luo, B. Yang, and R. Urtasun, "Fast and Furious: Real Time End-to-End 3D Detection, Tracking and Motion Forecasting With a Single Convolutional Net," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018.

[14] M. Simon, S. Milz, K. Amende, and H.-M. Gross, "Complex-YOLO: Real-time 3D Object Detection on Point Clouds," in *European Conference on Computer Vision*, Mar. 2018.

[15] B. Yang, W. Luo, and R. Urtasun, "PIXOR: Real-Time 3D Object Detection From Point Clouds," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, p. 9.

[16] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," in *Neural Information Processing Systems*, 2017.

[17] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. Waslander, "Joint 3D Proposal Generation and Object Detection from View Aggregation," *arXiv:1712.02294 [cs]*, Dec. 2017.

[18] M. Liang, S. Wang, B. Yang, and R. Urtasun, "Deep Continuous Fusion for Multi-Sensor 3D Object Detection," in *European Conference on Computer Vision*, 2018, p. 16.

[19] Z. Cai, Q. Fan, R. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection," in *European Conference on Computer Vision*, 2016.

[20] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017.

[21] Y. Zhou and O. Tuzel, "VoxelNet: End-to-End Learning for Point Cloud Based 3d Object Detection," Nov. 2017, arXiv: 1711.06396.

[22] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[23] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017.