Rotation-Aware 3D Vehicle Detection From Point Cloud

HYUNJUN CHOI (Member, IEEE), JIYEOP JEONG (Member, IEEE), AND JIN YOUNG CHOI (Member, IEEE)
Department of Electrical and Computer Engineering, ASRI, Seoul National University, Seoul 08826, South Korea

Corresponding author: Jin Young Choi (jychoi@snu.ac.kr)

This work was supported by the Institute for Information & Communication Technology Promotion (IITP) grant through the Ministry of Science and ICT (MSIT), South Korea, under Grant 2017-0-00306 (Outdoor Surveillance Robots) and Grant 2014-3-00123 (Development of High Performance Visual BigData Discovery Platform for Large-Scale Realtime Data Analysis).

ABSTRACT Three-dimensional vehicle detection using LiDAR point clouds is important for the stability of autonomous driving. It can provide high-quality three-dimensional information for most obstacles. Although efficient algorithms based on a Bird’s Eye View (BEV) feature map have been developed, many research issues still remain; in particular, the existing methods show a limited accuracy in estimating the rotation angle of a 3D object. In this paper, to improve the accuracy of rotation angle estimation, we propose a rotation-aware 3D vehicle detector that extracts distinguishable features from the proposals with various angles of rotations. Experiments are conducted on KITTI dataset and Waymo Open dataset. Our approach improves detection accuracy as well as rotation angle estimation accuracy against the existing algorithms without much loss of computational efficiency.

INDEX TERMS Autonomous driving, convolutional neural network, 3D object detection, 3D vehicle detection, lidar, point cloud.

I. INTRODUCTION

Key technologies in self-driving vehicles include environmental awareness, accurate localization, and optimal path planning. Vehicle detection, specifically, not only raises environmental awareness, but also drastically improves the safety of unmanned vehicles. Recently, vehicle detection research has been expanded from 2D to 3D. Detection in 3D provides three-dimensional information about an object, including the exact distance from the vehicle to the object. This helps increase the stability of autonomous driving. Accurate three-dimensional object information is usually obtained from LiDAR sensors.

Much research has been done on 3D object detection from point clouds obtained by LiDAR sensors [1]–[14]. Conventional 3D object detection methods based on point clouds can be divided into two approaches. The first approach is to detect 3D objects through a network that works directly on the raw point cloud; such methods include F-PointNet [1], F-Convnet [13], STD [14], and PointRCNN [8]. The networks take raw point cloud data as an input and use PointNet++ [15] structure to localize objects in 3D. This method achieves a certain level of performance, but it comes with a large computational cost. The second approach projects the point cloud to 2D Bird’s Eye View (BEV) feature map, from which it then detects 3D objects [2], [3], [5]–[7], [9], [11], [16]. In this approach, there is a loss of information, which results in slightly lower performance than the raw point cloud-based approach. However, because of its benefits in terms of real-time application and efficiency, most methods adopt the 2D BEV feature map-based approach.

Existing methods based on the BEV approach do not consider the rotation of an object area on the BEV map in the pooling stage for feature extraction. The actual BEV representation of a 3D cuboid bounding box is arbitrarily rotated (i.e., not axis-aligned), unlike the 2D object bounding box [12]. Nevertheless, the existing BEV feature map-based methods still apply the conventional method of detecting 2D objects [17]–[19] in an image, even if they aim to detect 3D objects. They estimate the rotation angle of a 3D object by simply adding a rotation angle to the regression target. This causes an inaccurate estimation of the rotation of the 3D object on the BEV. As seen in Figure 1, proposals generated...
Figure 1. Result of 3D vehicle detection. 3D proposals made by the existing detector [9]. (a): scene image; (b): bird's eye view; (c): 3D view. Green boxes denote ground-truth boxes, and red boxes denote predicted boxes.

by the existing methods [9] show this inaccurate estimation. The proposals depicted as red-lined cuboids have similar but slightly different angles of rotation; this inaccurate estimation degrades the performance of 3D detection, which requires strict conditions to ensure safe driving.

In this paper, we propose a rotation-aware framework aiming to improve the accuracy of rotation angle estimation. This can be solved by obtaining distinctive feature representations from proposals with various angles of rotation. Existing methods use only the features extracted from the axis-aligned 2D box region. We use a different technique to obtain features for the rotated 2D box region. After dividing the rotated 2D box region into grid-based subregions, we obtain bilinear interpolated features for each sub-region. In this way, we can extract distinguishable features for proposals with different angles of rotation. As a result, the proposed rotation-aware framework able to choose the proposal with the most accurate rotation angle among the various box proposals with different rotation angles.

We also develop a rotation-aware end-to-end 3D vehicle detection network, referred to as a Rotation-aware 3D vehicle detector (RA3D). RA3D consists of a region proposal module and a rotation-aware prediction module. The region proposal module is the same as those in existing 2D BEV feature-based 3D detectors (e.g., SECOND [9] or PointPillars [11]), and the results of the module are the same as those of those detectors. For convenience, the RA3D of each region proposal module is named by prepending RA (Rotation-aware) to the name of an existing 3D detector (e.g., RA-SECOND or RA-PointPillars). In the rotation-aware prediction module, we select the more accurate proposals among those with various angles of rotation. We also refine the selected proposals to accurately fit the 3D-object region. The network consists of only fully-connected layers. It has two output nodes for classification and regression, respectively. RA3D is applicable to all methods of detecting 3D objects in an add-on manner; it is able to improve a model’s performance with only a small amount of extra computation.

Using KITTI dataset [20] and Waymo Open dataset [21], we validate our RA3D model in combination with the existing methods of BEV approach; SECOND [9], PointPillars [11], and PV-RCNN [16]. The results show that our rotation-aware model can improve detection accuracy as well as rotation angle estimation accuracy against the existing algorithms without much loss of computational efficiency.

II. RELATED WORKS

Point cloud based 3D object detection methods generally use one of two approaches: a raw point cloud based approach or a 2D BEV map based approach. The raw point cloud-based approach detects 3D objects with a network that works directly on the raw point cloud. Internally, this approach uses PointNet [22] or PointNet++ [15] to directly infer a 3D cuboid containing the object with sparse point data. The 2D BEV map-based approach projects the point cloud to a BEV feature map, then detects the 3D object in the 2D BEV map. Details of the two approaches are given in the following sections.

A. RAW POINT CLOUD-BASED APPROACH

The PointNet-based network used in the raw point cloud based approach receives one of two input types: the entire point cloud or only a part of the point cloud. Methods using the first input type include PointRCNN [8] and Sparse-To-Dense (STD) [14]; in these, 3D proposals are generated via point cloud segmentation. These methods perform well but because the PointNet-based network uses the entire point cloud, there is a heavy computational burden. Methods using the second type include F-PointNet [1] and F-ConvNet [13], in which the part of point cloud corresponding to the object is acquired from the result of a 2D object detection module. Hence, these methods depend highly on the performance of 2D object detection.

B. 2D BEV MAP-BASED APPROACH

Most works using this approach, focus on how to obtain a good BEV feature map from a point cloud. At the earliest stages, the BEV feature map was obtained by using hand-crafted features [2], [3], [6], [7]. For example, [2], [3] generate a 2D height-based feature map by slicing the point cloud along the z-axis. To mitigate the information loss caused by the hand-crafted features, there have been studies adopting learned features, such as VoxelNet [5], SECOND [9], and PointPillars [11]. VoxelNet [5] divides the point cloud into voxel units and applies PointNet to each unit. SECOND [9] improves its speed over VoxelNet’s by replacing the latter’s regular 3D convolution with its own improved convolution. PointPillars [11] divides the point cloud into pillar units and applies PointNet to each unit. Compared to SECOND and VoxelNet, which use 3D convolution to integrate voxel units, PointPillars use 2D
FIGURE 2. Proposed architecture of RA3D network for Rotation-aware 3D vehicle detection.

convolution to integrate pillar units. SECOND and PointPillars can operate in real-time: SECOND at 20Hz and PointPillars at 62Hz. Recently, PV-RCNN [16] extends SECOND that preserves more 3D structure information by adding a keypoint branch. A keypoint branch has a form of introducing VSA (Voxel Set Abstraction) in the middle of the voxelization process. The 3D region proposal from SECOND network is further refined using key point features extracted from RoI grid pooling. In our work, we attempt to alleviate existing methods’ inaccurate estimation of rotation angle by improving feature representation to consider the rotation angle of the object.

III. PROPOSED METHOD
As shown in Figure 2, the proposed RA3D network is composed of two modules: a 3D region proposal module (3DRPM) and a rotation-aware prediction module (RAPM). 3DRPM (Section III-A) conducts a process to obtain 3D proposals. RAPM (Section III-B) obtains the final detection result through the selection and refinement of the most accurate proposals among the various 3D proposals for an object.

A. 3D REGION PROPOSAL MODULE
The 3D region proposal module (3DRPM) in Figure 2 is the same as the existing BEV map-based 3D object detector for point clouds. For the module, we adopt SECOND [9], PointPillars [11], and PV-RCNN [16] which show the state-of-the-art performance among BEV map-based 3D detectors. In 3DRPM, a feature extraction network projects the point cloud to a 2D BEV feature map, on which the region proposal network (RPN) generates 3D proposals. RPN performs classification and regression based on 2D object detectors [17]–[19] and conducts additional regression on the position, height, and rotation angle of the z-axis. Based on the scores obtained from RPN, the top N 3D region proposals are selected, which are further refined by the rotation-aware prediction module (RAPM) described in the next section. In the case of PV-RCNN [16], 3D region proposals are obtained from the RPN of SECOND [9] network. However, these 3D region proposals are further refined by additional feature. These refined proposals are once more refined by the rotation-aware prediction module (RAPM).

B. ROTATION-AWARE PREDICTION MODULE
1) ROTATED 2D BOUNDING BOX REPRESENTATION
In general, 3D object detection is a task for estimating a 3D bounding box that has a shape of cuboid. A 3D bounding box can be represented as a 7-tuple $(x_{3D}, y_{3D}, z_{3D}, l_{3D}, w_{3D}, h_{3D}, \theta_{3D})$, where $x_{3D}$, $y_{3D}$, and $z_{3D}$ are the center coordinates; $l_{3D}$, $w_{3D}$, and $h_{3D}$ are the length, width, and height, respectively; and $\theta_{3D}$ is the rotation around the z-axis. 2D projection of a 3D bounding box to a BEV map results in a rotated 2D bounding box, which can be represented as a 5-tuple $(x_{2D}, y_{2D}, l_{2D}, w_{2D}, \theta_{2D})$, where $x_{2D}$, and $y_{2D}$ are the center coordinates; $l_{2D}$ and $w_{2D}$ are the length, and width, respectively; and $\theta_{2D}$ is the rotation around the z-axis. Therefore, 2D projection of a 3D bounding box to BEV map is a process of removing the height and position $(z_{3D}, h_{3D})$ of the 3D bounding box about the z-axis.
The formal expression is given by:

\[
\begin{align*}
    x_{2D} &= x_{3D}, \\
    y_{2D} &= y_{3D}, \\
    l_{2D} &= l_{3D}, \\
    w_{2D} &= w_{3D}, \\
    \theta_{2D} &= \theta_{3D}.
\end{align*}
\]  

(1)

2) ROTATED-ROI POOLING LAYER

Region of Interest (RoI) pooling [17] and RoI Align [19] techniques have been proposed for the axis-aligned 2D bounding box. To handle the arbitrary-oriented text detection task, Rotated-RoI pooling concept has been proposed in [23]. Rotated-RoI pooling can extract the feature with the shape of a 2D rotated bounding box by calculating the center location of each sub-region. Since the 2D rotated bounding box has rotation around the reference axis, Rotated-RoI pooling additionally has a backward alignment step that aligns the feature on the reference axis [23].

![Figure 3. RoI Align versus Rotated-RoI pooling process.](image)

(a)(b): RoI Align; (c)(d)(e): Rotated-RoI pooling. (a)(c): BEV feature map of point cloud; (b): RoI Align feature; (d) Axis-aligned feature map; (e): Rotated-RoI pooled feature.

In 3D vehicle detection, 3D region proposals are provided by a 3D detector. As 2D projection of 3D region proposals to a BEV feature map results in a rotated 2D bounding box, we can utilize the Rotated-RoI pooling for point cloud-based 3D vehicle detection. As shown in Figure 3, Rotated-RoI pooling can extract more accurate 2D projected feature of 3D region proposals than those of RoI Align from the 2D BEV feature map. Specifically, we split the 2D rotated bounding box into \( h_r \times w_r \) grid-based subregions, where \( h_r \) and \( w_r \) are height and width of pooling size, respectively as seen in Figure 3-(c). The center of each sub-region is obtained as a feature by bilinear interpolation of the features of its neighboring pixels. This feature is then aligned backward on the reference axis which is y-axis in Figure 3-(d). Practically, the center location (on the feature map) of each sub-region is computed using the rotated 2D bounding box representation. Then, the center location feature is obtained by bilinear interpolation. Finally, we obtain the Rotated-RoI pooled feature, as shown in Figure 3-(e). In detail, a BEV feature map has the dimension of \( H \times W \times C \), which are the height, width and channel dimensions, respectively. Considering the number of channels (\( C \)) of the BEV feature map, the final dimension of the pooled feature vector becomes \( h_r \times w_r \times C \) for each proposal. The Rotated-RoI pooled feature vector reshapes into a 1D vector with size \( h_r \cdot w_r \cdot C \) and enters the fully-connected layer for classification and regression.

3) LEARNING OF CLASSIFICATION

In this step, the goal is to learn how to choose the most accurate proposal among 2D boxes rotated with various rotation angles based on the confidence score that is output of the classification node. SECOND and PointPillars select rotated boxes with intersection-over-union (IoU) below 0.45 for negative samples, and above 0.6 for positive samples. We further reinforce this procedure to get negative samples with IoU below 0.65 and positive samples with IoU above 0.7. For the loss function for classification, the focal cross entropy loss is:

\[
L_{\text{cls}} = FL(p_t) = -\alpha(1 - p_t)^{\gamma} \log(p_t),
\]

(2)

where \( p_t \) is the measured probability of the model, and \( \alpha \) and \( \gamma \) are parameters for focal loss [24]. Following [9, 11], we set \( \alpha = 0.25 \) and \( \gamma = 2 \). To confirm the awareness of rotation in our method, it is illustrated in Figure 4; (b) is the proposal, (c) is the result of the existing detector SECOND [9], and (d) is the classification result of our method. Comparing Figures 4-(c) and 4-(d), our method estimates rotation angle more accurately than the existing method.
4) LEARNING OF REGRESSION

In the proposal refinement step, regression is performed on the position, size, and rotation angle of the object’s BEV. The regression is done for five elements ($\delta x$, $\delta y$, $\delta w$, $\delta l$, $\delta \theta$) of the regression residual, where $\delta x$ and $\delta y$ are for the center coordinates; $\delta l$, $\delta w$ are for the length, and width, respectively; and $\delta \theta$ is for the rotation around the z-axis. The regression residuals between ground truth and anchors are defined by:

\[
\begin{pmatrix}
\delta x \\
\delta y
\end{pmatrix}
= \begin{pmatrix}
\cos \theta_a & \sin \theta_a \\
-\sin \theta_a & \cos \theta_a
\end{pmatrix}
\begin{pmatrix}
x_a-x_g \\
y_a-y_g
\end{pmatrix}, \quad \begin{pmatrix}
\delta w \\
\delta l
\end{pmatrix} = \begin{pmatrix}
\log\left(\frac{w_g}{w_a}\right) \\
\log\left(\frac{l_g}{l_a}\right)
\end{pmatrix}, \quad \delta \theta = \theta_g - \theta_a, \quad (3)
\]

where, $d_a = \sqrt{l_a^2 + w_a^2}$; $(x_a, y_a, w_a, l_a, \theta_a)$ is the regression element for the anchor box; and $(x_g, y_g, w_g, l_g, \theta_g)$ is the regression element for the ground-truth box.

Rotated-RoI pooling extracts feature vectors that are aligned with the orientation of the rotated box. Therefore, we use a coordinate system based on the orientation axes. $\delta x$ and $\delta y$ are the rotated offsets to be aligned with the orientation axes. For the regression loss, a smooth $L_1$ loss is used for each component, $x$, of ($\delta x$, $\delta y$, $\delta w$, $\delta l$, $\delta \theta$) as:

\[
L_{\text{smooth}L_1}(x) = \begin{cases} 
0.5|x|^2 & \text{if } |x| < 1 \\
|x| - 0.5 & \text{otherwise}
\end{cases} \quad (5)
\]

The total regression loss is sum of the smooth $L_1$ loss for all components, i.e.,:

\[
L_{\text{reg}} = \sum_{x \in (x, y, w, l, \theta)} L_{\text{smooth}L_1}(x). \quad (6)
\]

Finally, the training is done, with total loss that combines classification loss and regression loss as:

\[
L_{\text{total}} = \beta_1 L_{\text{cls}} + \beta_2 L_{\text{reg}}, \quad (7)
\]

where we use the settings of $\beta_1 = 1.0$, $\beta_2 = 2.0$ in [9] and [11].

The effectiveness of the proposal refinement process is illustrated in Figure 5: (a) and (b) are the result of SECONDS [9], and (c) and (d) are the regression result of our method.

**IV. EXPERIMENTAL RESULT**

**A. DATASET**

1) KITTI DATASET

The KITTI object detection benchmark [20] has 7,481 training images with point clouds and 7,518 test images with point clouds. We trained the network using only point clouds. For evaluation on the validation set, we split the KITTI training samples into a training set containing 3,712 samples and a validation set containing 3,769 samples following [2], which is referred to as ‘5:5 split’. For evaluation on the test set, we use the best model on the ‘5:5 split’ validation set.

2) WAYMO OPEN DATASET

Waymo Open dataset [21] has total of 1,000 sequences including 798 training sequences (158k point cloud samples) and 202 validation sequences (40k point clouds samples). Waymo Open dataset provides annotations for objects in the full 360 angle. Based on the list of Waymo training sequences in the public code of PV-RCNN [16], we sampled the first one among every 5 sequences from 798 sequences for our training set. Hence the training set includes 160 sequences. We evaluate our model on the 202 validation sequences.

**B. IMPLEMENTATION DETAILS**

We used the pre-trained networks of SECONDS [9], PointPillars [11], or PV-RCNN [16] for our 3D region proposal module (3DRPM); the network configurations are the same as those of the respective original paper. Re-implementation was done based on the public code,1 which showed some differences from the performance stated in the original papers. After we trained 3DRPM, we froze its weights and trained the rotation-aware prediction module (RAPM). In particular, we also used version 1.5 of SECONDS, which has a slightly different structure from that of the original. We selected the top $N$ proposals in 3DRPM. PV-RCNN has the same BEV feature map structure with SECONDS V1.5. All the configuration of RAPM is the same as SECONDS V1.5. Thus, the implementation details for PV-RCNN follow those of SECONDS V1.5, as discussed in section IV-B, IV-C, and IV-D. $N$ was set to 1000 for training and 300 for evaluating.

Our RAPM consists of a pooling layer and a fully-connected layer. Table 1 summarizes the architecture and the size (number of parameters) of RAPM. The pooling layer extracts features of proposals from the BEV feature map.

---

1SECONDS: https://github.com/traveller59/second.pytorch
PointPillars: https://github.com/nutonomy/second.pytorch
PV-RCNN: https://github.com/open-mmlab/OpenPCDet
Following the original works of SECOND V1.5 and PointPillars, the dimensions of the BEV feature maps (on KITTI dataset) are $200 \times 176 \times 128$ and $248 \times 216 \times 384$, respectively. The fully-connected layer has been designed in the form of a series of nodes as $[h_r \cdot w_r \cdot C, 1024, 1024, 6]$; $h_r$ and $w_r$ are the height and width of the pooling size, respectively, and $C$ is the number of channels of the BEV feature map. The input layer has the same dimension as the 1D reshaped form of the Rotated-RoI pooled feature vector, which was specified in Section III-B2. The output layer consists of a classification and a regression node, which have the numbers of 1 and 5, respectively. For RA-SECOND V1.5 and RA-PointPillars, $h_r \times w_r$ is set to $7 \times 7$ and $9 \times 9$, respectively. Hence, the size of RAPM can be calculated for both RA-SECOND V1.5 and RA-PointPillars as laid out in Table 1.

| Table 1. RAPM architecture with the number of parameters for both RA-SECOND V1.5(RA-SE) and RA-PointPillars(RA-PP). |
| --- |
| **Layer** | **Filter** | **Output** | **Number of parameters (ea)** |
| --- | --- | --- | --- |
| input: BEV map | $(1, H, W, C)$ | | **RA-SE** | **RA-PP** |
| RROI pooling ($N$ proposals) | $(h_r, w_r)$ | $(N, h_r, w_r, C)$ | 0 | 0 |
| reshape | $(N, h_r \cdot w_r \cdot C)$ | | 0 | 0 |
| fc1 | $(1024)$ | $(h_r \cdot w_r \cdot C, 1024)$ | $6,423,352$ | $31,851,520$ |
| fc2 | $(1024)$ | $(1024, 1024)$ | $1,049,600$ | $1,049,600$ |
| fc3 | $(6)$ | $(1024, 6)$ | $6,130$ | $6,130$ |

Moreover, by adding the size of RAPM to the size of the existing network (SECOND V1.5 or PointPillars), we can calculate the total size of RA-SECOND V1.5 and RA-PointPillars. Finally, Table 2 summarizes the network size comparison of our RA3D network and the existing networks. The number of parameters used in RAPM is greater than the number of parameters used in the existing network. As seen in Table 1, the main cause for this is that the number of nodes in fc1 and fc2 are set to 1024. In the ablation study, we considered the effect of downsizing our network by reducing the number of nodes in fc1 and fc2.

| Table 2. Comparison of network size. |
| --- |
| **Method** | **Number of parameters (ea)** |
| --- | --- |
| SECOND V1.5 | $1,776,308$ |
| RA-SECOND V1.5 | $9,255,610(+47,479,302)$ |
| PointPillars | $2,236,928$ |
| RA-PointPillars | $35,144,198(+432,907,270)$ |

We trained the network with Adam optimizer. The learning rate was initially set to 0.0002 and exponentially decayed by a ratio of 0.8 every $\frac{1}{16}$ of the total of 16 epochs. In our experimental environment (using a single GeForce GTX 1080 Ti GPU), the mini-batch size for our network was set to one (maximum) due to GPU memory limitations. We selected the best validation model that was evaluated per epoch. Data augmentation methods, non-maximum suppression (NMS), and related parameters followed the original works in the SECOND [9], PointPillars [11], and PV-RCNN [16].

### C. SELF ANALYSIS

#### 1) TRAINING PROCESS
As described in Section IV-B, we trained our RA3D network for total of 16 epochs with Adam optimizer and initial learning rate set to 0.0002. Figure 6 presents the training loss of both RA-SECOND V1.5 and RA-PointPillars during training on the validation set (5:5 split). As seen in Eq. (7), the training loss is a weighted sum of classification loss and regression loss, which are shown separately in Figure 6. Both classification loss and regression loss have a form that converges while decreasing during training. RA-SECOND V1.5 shows less regression loss and less classification loss than RA-PointPillars. We can infer the reason why RA-SECOND V1.5 has better performance than RA-PointPillars. During training, the learning rate decays exponentially by a ratio of 0.8 every $\frac{1}{16}$ of the total of 16 epochs. Hence, a spike in the loss occurs commonly in every step of learning rate decay. To reduce the spike and make the training process more stable, we can think of a way to control the rate of decay.

#### 2) INFERENCE TIME ANALYSIS
The total computation time required for our method is obtained by adding the additional computation time to the time required for the existing methods [9], and [11]. As described in Section III-B, additional computation is required in our Rotation-aware prediction module (RAPM), which consists of a pooling layer and a fully-connected layer. We measured the average computation time taken by RAPM during the inference of the 5:5 split validation set.

Figure 7 shows the computation time of RA-SECOND V1.5 and Figure 8 that of RA-PointPillars, versus the number of proposals. We selected the top $N$ proposals for RAPM as described in Section III-A. In the figures, the trends of
inference time and mean average precision (mAP) versus variation of $N$ are depicted. In addition, the trends of BEV and 3D performance versus $N$ are also depicted. When evaluating RA-SECOND V1.5 and RA-PointPillars, we selected $N = 300$ (the red point in the figure), which showed stable performance and relatively reasonable computation time. The additional computation time was $1.96 \text{ ms}$ for RA-SECOND V1.5 and $6.63 \text{ ms}$ for RA-PointPillars, which reduces the final speed of RA-SECOND V1.5 from $25Hz$ to $23.8Hz$ and that of RA-PointPillars from $62Hz$ to $43.9Hz$. Furthermore, additional efficiency can be achieved by adjusting the number of proposals. For instance, as seen in Figures 7 and 8, $N = 100$, requires less computational time than $N = 300$ without sacrificing much accuracy.

### TABLE 3. 3D detection performance (KITTI validation set, Moderate samples) of RA-SECOND V1.5 depending on classification and regression loss weight.

| Classification loss weight ($\beta_1$) | Regression loss weight ($\beta_2$) | Car AP 3D for Mod (%) |
|----------------------------------------|-----------------------------------|-----------------------|
| 0.5                                    | 1.5                               | 79.05                 |
|                                        | 2.0                               | 78.80                 |
|                                        | 2.5                               | 79.00                 |
| 1.0                                    | 1.5                               | 78.96                 |
|                                        | 2.0                               | 79.64                 |
|                                        | 2.5                               | 78.97                 |

3) WEIGHTING PARAMETER IN LOSS

In Section III-B4, we defined our total loss as a weighted sum of classification loss, and regression loss which is specified in Eq. (7). We set our weight parameters as $\beta_1 = 1.0, \beta_2 = 2.0$ following the settings of [9], [11]. We conducted our experiment on the KITTI validation set (5:5 split) that evaluates 3D detection performance depending on the weight of loss variation. Table 3 summarizes the 3D detection performance of RA-SECOND V1.5 for moderate samples in a variation of $\beta_1$ and $\beta_2$. For the variation of the weight parameter, the variation of 3D detection performance is up to the mean average precision of 0.13%. The best performance can be achieved in the original settings of $\beta_1 = 1.0, \beta_2 = 2.0$.

### TABLE 4. Ablation results (5:5 split) of RA3D network. Note that $\dagger$: SECOND V1.5, $\ddagger$: PointPillars.

| Method                          | Car AP BEV (%) | Car AP 3D (%) |
|--------------------------------|----------------|---------------|
|                                | Easy | Mod | Hard  | Easy | Mod | Hard  |
| B1: baseline$\dagger$          | 90.06| 87.51| 86.07 | 87.97| 77.58| 76.50 |
| S1: B1+cls                     | 90.44| 88.12| 86.72 | 89.29| 78.78| 77.55 |
| S2: B1+reg                     | 90.03| 87.63| 86.14 | 87.99| 77.71| 76.57 |
| S3: B1+cls+reg                | 90.52| 88.22| 86.78 | 89.61| 79.04| 77.81 |

### TABLE 5. RA-SECOND V1.5: Performance variation of RA3D network depending on different pooling resolutions.

| Pooling resolution | Car AP BEV (%) | Car AP 3D (%) |
|--------------------|----------------|---------------|
|                    | Easy | Mod | Hard  | Easy | Mod | Hard  |
| $1 \times 1$       | 89.37| 81.28| 81.08 | 83.98| 73.85| 69.34 |
| $3 \times 3$       | 90.24| 87.72| 86.03 | 88.11| 77.89| 76.58 |
| $5 \times 5$       | 90.24| 87.85| 86.22 | 88.77| 78.46| 77.11 |
| $7 \times 7$       | 90.52| 88.22| 86.78 | 89.61| 79.04| 77.81 |
| $9 \times 9$       | 90.37| 88.13| 86.89 | 88.94| 78.56| 77.23 |
| $11 \times 11$     | 90.04| 87.82| 86.12 | 88.49| 78.13| 76.79 |

### TABLE 6. RA-PointPillars: Performance variation of RA3D network depending on different pooling resolutions.

| Pooling resolution | Car AP BEV (%) | Car AP 3D (%) |
|--------------------|----------------|---------------|
|                    | Easy | Mod | Hard  | Easy | Mod | Hard  |
| $7 \times 7$       | 90.49| 87.75| 82.39 | 85.87| 76.54| 70.98 |
| $9 \times 9$       | 90.50| 87.91| 85.17 | 84.24| 76.10| 71.08 |
| $11 \times 11$     | 90.43| 87.89| 86.58 | 84.37| 76.17| 71.09 |

### TABLE 7. 3D detection (KITTI validation set) performance of RA-SECOND V1.5 depending on the number of nodes in fc1 and fc2 (in RAPM).

| Number of nodes in fc1, fc2 | Size ratio (%) | Car AP 3D (%) |
|-----------------------------|----------------|---------------|
| 1024, 1024                  | 100.00         | 89.61| 79.04| 77.81 |
| 512, 512                    | 46.50          | 89.59| 79.99| 77.72 |
| 236, 236                    | 22.37          | 89.42| 78.85| 77.59 |
| 128, 128                    | 10.97          | 89.40| 78.88| 77.71 |
| 64, 64                      | 5.43           | 89.55| 78.83| 77.66 |
| 32, 32                      | 2.70           | 89.33| 78.89| 77.70 |

D. ABLATION STUDIES

1) ROTATION-AWARE MODEL ANALYSIS

As seen in Section III-B, our RA3D network has six output nodes which are one for classification and five for regression.
In Table 4, the effect of each node is shown for the 5:5 split KITTI validation set. We selected the model with the best performance for the test. As seen in the table, in the 5:5 split set, the use of both classification and regression works best for both RA-SECOND V1.5 and RA-PointPillars.

2) POOLING RESOLUTION

Tables 5 and 6 show the performance of RA-SECOND V1.5 and RA-PointPillars, respectively, on the KITTI validation set (5:5 split) according to the resolution of Rotated-RoI pooling. The best performance is achieved on 7 × 7 for RA-SECOND V1.5 and 9 × 9 for RA-PointPillars. The 7 × 7 patch covers a vehicle in the feature map with size of 200 × 176 for SECOND, and the 9 × 9 patch also covers a vehicle in the feature map with size of 248 × 216 for PointPillars. Hence, the used pooling resolution becomes almost the maximum resolution. On the other hand, 1 × 1 pooling in RA-SECOND V1.5 has the same effect as non-rotated RoI pooling, which illustrates the effect of Rotated-RoI pooling.

3) NETWORK DOWNSIZING

As mentioned in Section IV-B, we can downsize our network by reducing the number of nodes in fc1 and fc2. We conducted an experiment on the KITTI validation set (5:5 split) to evaluate 3D detection performance depending on the downsizing ratio of our network (precisely, RAPM). Table 7 presents the 3D detection performance of RA-SECOND V1.5 depending on the number of nodes in fc1 and fc2. When
FIGURE 9. Qualitative results (RA-SECOND) of detection on KITTI validation set. (a): Scene image; (b): original 3D detection result (SECOND [9]); (c): rotation-aware 3D detection result. Green boxes denote ground-truth boxes and red boxes denote predicted boxes.

FIGURE 10. Qualitative results (RA-PointPillars) of detection on KITTI validation set. (a): Scene image; (b): original 3D detection result (PointPillars [11]); (c): rotation-aware 3D detection result. Green boxes denote ground-truth boxes and red boxes denote predicted boxes.

reducing the number of nodes from 1024 to 32, the size of the network is reduced to 2.70%. On the contrary, the 3D detection performance for moderate samples is declined by the mean average precision of 0.15%. Network downsizing can be applied for an environment which is sensitive to the number of parameters without much loss of 3D detection performance.

E. PERFORMANCE EVALUATION

1) EVALUATION USING THE KITTI TEST SET
Since the KITTI test set labels are not publicly available, we can only quantitatively evaluate our method by submission to the KITTI Vision Benchmark Suite [20]. Table 8 presents the performance of our 3D detector on the KITTI test set. Similar to the validation results in Section IV-E2, RA-SECOND V1.5, RA-PointPillars, and RA-PV-RCNN achieve better performance than the baseline in both BEV and 3D detection. Although our approach requires additional computation for the Rotation-aware 3D detector, it still achieves good efficiency as summarized in Section IV-C2 with negligible (up to 7 ms) additional computation.

2) EVALUATION USING THE KITTI VALIDATION SET
Table 9 presents the performance of our 3D detector on the KITTI validation set (5:5 split) in [2]. It can be seen, that our approach achieves better performance than the baseline in both BEV and 3D detection. Our Rotation-aware 3D detectors improve BEV localization of the BEV based detectors SECOND [9], PointPillars [11], and PV-RCNN [16]. RA-SECOND V1.5 shows fast speed and performance comparable to non-real-time state-of-the-art algorithms such as STD [14] and F-convNet [13].

Figures 9 and 10 show the qualitative detection results on the KITTI validation set for RA-SECOND V1.5 and RA-PointPillars, respectively. The images in (b) and (c) of these
<TABLE>
| Method                   | Modality | Easy | Mod | Hard |
|--------------------------|----------|------|-----|------|
| SECOND V1.5-val         | Lidar    | 92.70 | 92.51 | 3.19 |
| Ours (RA-SECOND V1.5-val) | Lidar    | 92.87 | 92.73 | 4.45 |
| PointPillars-val        | Lidar    | 90.72 | 90.55 | 4.96 |
| Ours (RA-PointPillars-val) | Lidar    | 90.76 | 90.68 | 3.40 |
| PV-RCNN-val             | Lidar    | 95.9606 | 95.9343 | 1.8971 |
| Ours (RA-PV-RCNN-val)   | Lidar    | 95.9514 | 95.9254 | 1.8864 |
| SECOND V1.5-test        | Lidar    | 94.16 | 94.04 | 5.01 |
| Ours (RA-SECOND V1.5-test) | Lidar    | 95.09 | 94.92 | 4.85 |
| PointPillars-test       | Lidar    | 93.54 | 93.19 | 7.01 |
| Ours (RA-PointPillars-test) | Lidar    | 93.96 | 93.90 | 2.90 |
| PV-RCNN-test            | Lidar    | 95.56 | 95.55 | 1.17 |
| Ours (RA-PV-RCNN-test)  | Lidar    | 95.63 | 95.62 | 1.17 |
| SubCNN [26]-test        | Image    | 90.81 | 90.67 | 4.50 |
| Deep3DBox [27]-test     | Image    | 92.98 | 92.90 | 3.36 |
| DeepMANTA [28]-test     | Image    | 97.25 | 97.19 | 2.85 |
| 3D-RCNN [29]-test       | Image    | 90.02 | 89.98 | 2.42 |

TABLE 11. Evaluation on the Waymo Open Dataset (version 1.2 released in March 2020) with 202 validation sequences: Vehicle class AP(Average Precision), APH(Average Precision Weighted by Heading), and AAE(Average Angular Error).

| Method                     | LEVEL1 Vehicle | LEVEL2 Vehicle |
|---------------------------|----------------|----------------|
|                           | AP(%)           | APH(%)         | AAE(°)          | AP(%)           | APH(%)         | AAE(°)          |
| SECOND V1.5               | 69.59           | 61.99          | (−0.37)         | 10.40           | 9.07           | (−0.02)         |
| Ours (RA-SE)              | 69.92           | 62.36          | (−0.38)         | 10.27           | 9.05           | (−0.02)         |
| PV-RCNN                   | 73.74           | 75.35          | 10.26           | 64.87           | 64.48          | 10.37           |
| Ours (RA-PV)              | 74.05           | 75.48          | 10.07           | 64.99           | 64.48          | 10.16           |

V. CONCLUSION
The existing algorithms, which project point cloud to BEV map and perform 3D detection on the map, have shown its limitation while estimating the angle of rotation. We designed a network that improves the accuracy of rotation angle estimation. Experiments using KITTI dataset and Waymo Open dataset have shown that our approach improves detection performance against the existing algorithms [9], [11], [16] without loss of computational efficiency. Although our study is able to increase the detection accuracy by estimating more accurate angle of rotation, the detection accuracy highly depends on the performance of the 3D region proposal module. The improvement of the 3D region proposal module
performance will contribute to development of a reliable 3D vehicle detection system.

CODE AVAILABILITY

The source-code for Rotation-aware 3D vehicle detector is available from https://github.com/hyunjunChhoi/Rotation-aware-3D-vehicle-detector.

REFERENCES

[1] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, “Frustum PointNets for 3D object detection from RGB-D data,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 918–927.

[2] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, “Multi-view 3D object detection network for autonomous driving,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 6526–6534.

[3] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, “Joint 3D proposal generation and object detection from view aggregation,” in Proc. IEEE/RSJ Int. Conf. Intel. Robots Syst. (IROS), Oct. 2018, pp. 1–8.

[4] M. Liang, B. Yang, S. Wang, and R. Urtasun, “Deep continuous fusion for multi-sensor 3D object detection,” in Proc. Eur. Conf. Comput. Vis. (ECCV), 2018, pp. 641–656.

[5] Y. Zhou and O. Tuzel, “VoxelNet: End-to-end learning for point cloud based 3D object detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 4490–4499.

[6] B. Yang, W. Luo, and R. Urtasun, “PIXOR: Real-time 3D object detection from point clouds,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 7652–7660.

[7] M. Simony, S. Milzy, K. Amendeg, and H.-M. Gross, “Complex-YOLO: An Euler-region-proposal for real-time 3D object detection on point clouds,” in Proc. Eur. Conf. Comput. Vis. (ECCV) Workshops, Sep. 2018.

[8] S. Shi, X. Wang, and H. Li, “PointRCNN: 3D object proposal generation and detection from point cloud,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 770–779.

[9] Y. Yan, Y. Mao, and B. Li, “SECOND: Sparsely embedded convolutional detection,” Sensors, vol. 18, no. 10, p. 3337, 2018.

[10] K. Shin, Y. P. Kwon, and M. Tomizuka, “RoarNet: A robust 3D object detection based on RegiOn approximation refinement,” in Proc. IEEE/RSJ Int. Conf. Intel. Vehicles Syst. (IV), Jun. 2019, pp. 2510–2515.

[11] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, “PointPillars: Fast encoders for point clouds from point clouds,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2019, pp. 12607–12705.

[12] M. Liang, B. Yang, Y. Chen, R. Hu, and R. Urtasun, “Multi-task multi-sensor fusion for 3D object detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 7345–7353.

[13] Z. Wang and K. Jia, “Frustum ConvNet: Sliding frustums to agglomerate local point-wise features for amodal 3D object detection,” in Proc. IEEE/RSJ Int. Conf. Intel. Robots Syst. (IROS), Nov. 2019, pp. 1742–1749.

[14] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia, “STD: Sparse-to-dense 3D object detector for point cloud,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 1951–1960.

[15] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, “PointNet: Deep learning on point sets for 3D classification and segmentation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 652–660.

[16] J. Ma, W. Shao, H. Ye, L. Wang, H. Wang, Y. Zheng, and X. Xue, “Arbitrary-oriented scene text detection via rotation proposals,” IEEE Trans. Multimedia, vol. 20, no. 11, pp. 3111–3122, Nov. 2018.

[17] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, “Focal loss for dense object detection,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 2980–2988.

[18] B. Yang, M. Liang, and R. Urtasun, “HDNET: Exploiting HD maps for 3D object detection,” in Proc. Conf. Robot Learn., 2018, pp. 146–155.

[19] Y. Xiang, W. Choi, Y. Lin, and S. Savarese, “Subcategory-agnostic convolutional neural networks for object proposals and detection,” in Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV), Mar. 2017, pp. 924–933.

[20] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecká, “3D bounding box estimation using deep learning and geometry,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 5632–5640.

[21] F. Chabot, M. Chaouch, J. Rabarisoa, C. Teuliere, and T. Chateau, “Deep MANTA: A coarse-to-fine many-task network for joint 2D and 3D vehicle analysis from monocular image,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 2040–2049.

[22] A. Kundu, Y. Li, and J. M. Rehg, “3D-RCNN: Instance-level 3D object reconstruction via render-and-compare,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 3559–3568.

HYUNJIN CHOI (Member, IEEE) received the B.S. degree in electrical engineering from Seoul National University, Seoul, South Korea, in 2017, where he is currently pursuing the Ph.D. degree in electrical and computer engineering. His current research interests include 3D object detection, autonomous driving, and pattern recognition.

JIYEOP JEONG received the B.S. degree in electrical engineering from Korea University, South Korea, in 2014. He is currently pursuing the Ph.D. degree in electrical and computer engineering with Seoul National University, Seoul, South Korea. His current research interests include crowd counting, visual tracking, and deep learning.

JIN YOUNG CHOI (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in control and instrumentation engineering from Seoul National University, Seoul, South Korea, in 1982, 1984, and 1993, respectively. From 1984 to 1989, he joined the project of TDX switching system with the Electronics and Telecommunications Research Institute (ETRI), Daejeon, South Korea. From 1992 to 1994, he was with the Basic Research Department, ETRI, where he was a Senior Member of Technical Staff involved in the neural information processing system. Since 1994, he has been with Seoul National University, where he is currently a Professor with the School of Electrical Engineering. He is also with the Automation and Systems Research Institute, the Engineering Research Center for Advanced Control and Instrumentation, and the Automatic Control Research Center, Seoul National University. From 1998 to 1999, he was a Visiting Professor with the University of California at Riverside, Riverside, CA, USA. His current research interests include adaptive and learning systems, visual surveillance, motion pattern analysis, object detection and tracking, and pattern learning and recognition.