RQR3D: Reparametrizing the regression targets for BEV-based 3D object detection

Supplementary Material

6. Obtaining the 3D bounding box parameters

Evaluation tool from nuScenes [44] requires 3D bounding boxes to be defined as $(x_{\text{ctr}}, y_{\text{ctr}}, z_{\text{ctr}}, w, l, h, \theta)$. In this section, we provide the details about how we obtain this representation using RQR3D outputs, $(x_{\text{min}}, y_{\text{min}}, x_{\text{max}}, y_{\text{max}})$ and $(u, v, \arg\min_u, \arg\min_v\ d_x, d_y, z_{\text{ctr}}, h)$.

Recalling that u represents the minimum of the offsets on x-axis, v represents the minimum of the offsets on y-axis, and $(\arg\min_u, \arg\min_v)$ represent their indices, respectively. We first reassign (u, v) with respect to $(x_{\min}, y_{\min})$ as follows:

$$u = \begin{cases} u, & \text{if } \arg\min_u < 0.5 \\ (x_{\text{max}} - x_{\text{min}}) - u, & \text{otherwise} \end{cases} \tag{5}$$

$$v = \begin{cases} v, & \text{if } \arg\min_v < 0.5 \\ (y_{\text{max}} - y_{\text{min}}) - v, & \text{otherwise} \end{cases} \tag{6}$$

Then, we calculate (w, l) as

$$A = \sqrt{(x_{\max} - u)^2 + v^2}, \quad B = \sqrt{u^2 + (y_{\max} - v)^2}. \tag{7}$$

$$w = \begin{cases} A, & d_x d_y \geq 0 \\ B, & \text{otherwise} \end{cases}, \qquad l = \begin{cases} B, & d_x d_y \geq 0 \\ A, & \text{otherwise} \end{cases}. \tag{8}$$

and $(x_{\text{ctr}}, y_{\text{ctr}})$ as

$$x_{\text{ctr}} = \frac{x_{\text{min}} + x_{\text{max}}}{2}, \quad y_{\text{ctr}} = \frac{y_{\text{min}} + y_{\text{max}}}{2}. \tag{9}$$

z_{ctr}, h can be used directly as regressed. Finally, we obtain θ using (d_x, d_y) as follows.

$$\theta = \arctan2(d_y, d_x). \tag{10}$$

Similarly, one can obtain the orientation using $(u, v, x_{\text{max}} - u, y_{\text{max}} - v)$ in a similar fashion to (w, l) calculation. However, in this case the angular range would be 180^o. Although the sign of (d_x, d_y) can be used to find the correct quadrant and extend the angular range to 360^o, we observe that calculating the orientation directly from (d_x, d_y) yields lower angular error.

In nuScenes convention, the corners of $\mathcal{B}$ are in the order of *front-left, front-right, rear-right, rear-left*. Let V_1, V_2, V_3, V_4 be the the bottom corners of $\mathcal{B}$ in this given order.

$$V_1 = \begin{cases} (x_{\text{max}},\ y_{\text{min}} + v,\ z_{\text{bottom}}), & \text{if } d_x \geq 0 \wedge d_y \geq 0 \\ (x_{\text{min}} + u,\ y_{\text{min}},\ z_{\text{bottom}}), & \text{if } d_x \geq 0 \wedge d_y < 0 \\ (x_{\text{max}} - u,\ y_{\text{max}},\ z_{\text{bottom}}), & \text{if } d_x < 0 \wedge d_y \geq 0 \\ (x_{\text{min}},\ y_{\text{max}} - v,\ z_{\text{bottom}}), & \text{if } d_x < 0 \wedge d_y < 0 \end{cases} \tag{11}$$

$$V_2 = \begin{cases} (x_{\text{max}} - u,\ y_{\text{max}},\ z_{\text{bottom}}), & \text{if } d_x \geq 0 \wedge d_y \geq 0 \\ (x_{\text{max}},\ y_{\text{min}} + v,\ z_{\text{bottom}}), & \text{if } d_x \geq 0 \wedge d_y < 0 \\ (x_{\text{min}},\ y_{\text{max}} - v,\ z_{\text{bottom}}), & \text{if } d_x < 0 \wedge d_y \geq 0 \\ (x_{\text{min}} + u,\ y_{\text{min}},\ z_{\text{bottom}}), & \text{if } d_x < 0 \wedge d_y < 0 \end{cases} \tag{12}$$

$$V_3 = \begin{cases} (x_{\text{min}},\ y_{\text{max}} - v,\ z_{\text{bottom}}), & \text{if } d_x \geq 0 \wedge d_y \geq 0 \\ (x_{\text{max}} - u,\ y_{\text{max}},\ z_{\text{bottom}}), & \text{if } d_x \geq 0 \wedge d_y < 0 \\ (x_{\text{min}} + u,\ y_{\text{min}},\ z_{\text{bottom}}), & \text{if } d_x < 0 \wedge d_y \geq 0 \\ (x_{\text{max}},\ y_{\text{min}} + v,\ z_{\text{bottom}}), & \text{if } d_x < 0 \wedge d_y < 0 \end{cases} \tag{13}$$

$$V_4 = \begin{cases} (x_{\text{min}} + u,\ y_{\text{min}},\ z_{\text{bottom}}), & \text{if } d_x \geq 0 \wedge d_y \geq 0 \\ (x_{\text{min}},\ y_{\text{max}} - v,\ z_{\text{bottom}}), & \text{if } d_x \geq 0 \wedge d_y < 0 \\ (x_{\text{max}},\ y_{\text{min}} + v,\ z_{\text{bottom}}), & \text{if } d_x < 0 \wedge d_y \geq 0 \\ (x_{\text{max}} - u,\ y_{\text{max}},\ z_{\text{bottom}}), & \text{if } d_x < 0 \wedge d_y < 0 \end{cases} \tag{14}$$

where $z_{bottom} = z_{\text{ctr}} - {}^{h}/_{2}$. Then, let V_5, V_6, V_7, V_8 be the the top corners of $\mathcal{B}$ in the same order. Similarly, their coordinates can be obtained using Eq. 11-14 replacing z_{bottom} with $z_{top} = z_{\text{ctr}} + {}^{h}/_{2}$.

7. Implementation Details

All models are trained using a batch size of 8 and an initial learning rate of 7.5×10^{-5}, optimized using the Adam optimizer. Training is conducted over 20 epochs with a multi-step learning rate schedule: the learning rate is reduced by a factor of 10 at epochs 15 and 18. The bird's eye view (BEV) representation covers a spatial extent of 100 meters by 100 meters, corresponding to a 50-meter range in all directions from the ego vehicle. The depth prediction network estimates depth within the range of 2 to 50 meters,

discretized into 48 bins. All experiments are carried out on four NVIDIA A5000 GPUs.

We apply standard BEV augmentations as commonly used in the literature [14]. Horizontal and vertical flipping of the BEV, along with corresponding image adjustments, are applied independently with a probability of 0.5. As a result, each of the four flipping configurations (none, horizontal only, vertical only, and both) occurs with equal probability of 0.25. We do not apply random scaling or image-space rotations. Instead, to augment the BEV feature representation, we apply discrete BEV rotations. With a probability of 0.5, the BEV is rotated by an angle randomly selected from the set [45, 90, 135, 180, 225, 270, 315] otherwise, no rotation is applied. These rotations affect only the projection of sensor information (camera and radar) into BEV space and do not modify the raw images or radar PC data. Additionally, we apply BEV scaling with a probability of 0.3. This augmentation reduces the spatial coverage from 50 meters to 25 meters, thereby increasing the resolution of nearby regions and enabling more precise detection of objects in close proximity.

8. Additional Experiments

8.1. Projection Methods

In Table 8, we evaluate the contribution of various projection methods to the overall performance. Our baseline model employs Lift-Splat projection with BEVDepth's depth distribution module, denoted as DN. We compare this baseline with three different versions: i) Lift-Splat projection with a simpler depth distribution, denoted as LSS, ii) IPM-based projection, and iii) MLP-based projection. The IPM, LSS, and MLP versions achieve comparable performance with each other, while the DN version yields the highest performance by leveraging the enhanced depth distribution module.

Table 8. Projection methods.

Projection	mAP	mATE	mASE
LSS	54.5	0.444	0.278
IPM	54.5	0.439	0.279
MLP	54.6	0.439	0.285
DN (Baseline)	55.3	0.432	0.277

8.2. Qualitative Comparison between CenterPoint and RQR3D

RQR3D head have a superior orientation regression compared to CenterPoint head with angle-based targets. This has been established quantitatively in the main paper. Figure 3 shows an example where CenterPoint head estimates the orientation drastically different between two consecutive frames, whereas RQR3D head provides a more stable estimation.

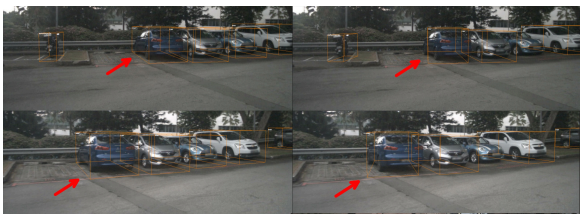

Figure 3. Orientation stability of RQR3D head (left) compared to CenterPoint head [20] (right)

8.3. Inference Time Analysis

The RQR3D model (with single frame, 704x256 image resolution and R50 backbone), when executed with FP16 precision, utilizes approximately 2.6 GB of memory—equivalent to 10% of the available capacity—on the NVIDIA A5000 GPU. Under these conditions, the model achieves an average inference speed of 14.8 frames per second (fps). For comparative purposes, performance estimates are referenced against the NVIDIA RTX 3090 GPU. The A5000 and RTX 3090 offer peak FP16 computational throughputs of 222.2 and 285.5 TFLOPs, respectively. Given that the inference process utilizes only 1.5% of the system memory and 30% CPU resources, I/O bottlenecks are negligible, and compute performance becomes the primary scaling factor. Accordingly, inference speed is expected to scale by a factor of approximately 1.28 when transitioning from the A5000 to the RTX 3090, yielding an estimated equivalent performance of 19 fps for RQR3D on the latter. During this inference, the auxiliary segmentation head and 2D object detection heads are removed. Similarly the objectness head outputs are not used during inference, therefore objectness head is also removed.