# M3DSSD: Monocular 3D Single Stage Object Detector

Shujie Luo[1] Hang Dai[3*] Ling Shao[3,4] Yong Ding[2*]
[1]College of Information Science and Electronic Engineering, Zhejiang University
[2]School of Micro-Nano Electronics, Zhejiang University
[3]Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE
[4]Inception Institute of Artificial Intelligence, Abu Dhabi, UAE
*Corresponding authors{hang.dai@mbzuai.ac.ae, dingy@vlsi.zju.edu.cn}.

## Abstract

*In this paper, we propose a Monocular 3D Single Stage object Detector (M3DSSD) with feature alignment and asymmetric non-local attention. Current anchor-based monocular 3D object detection methods suffer from feature mismatching. To overcome this, we propose a two-step feature alignment approach. In the first step, the shape alignment is performed to enable the receptive field of the feature map to focus on the pre-defined anchors with high confidence scores. In the second step, the center alignment is used to align the features at 2D/3D centers. Further, it is often difficult to learn global information and capture long-range relationships, which are important for the depth prediction of objects. Therefore, we propose a novel asymmetric non-local attention block with multi-scale sampling to extract depth-wise features. The proposed M3DSSD achieves significantly better performance than the monocular 3D object detection methods on the KITTI dataset, in both 3D object detection and bird's eye view tasks. The code is released at https://github.com/mumianyuxin/M3DSSD.*

## 1. Introduction

Three-dimensional (3D) object detection enables a machine to sense its surrounding environment by detecting the location and category of objects around it. Therefore, 3D object detection plays a crucial role in systems that interact with the real world, such as autonomous vehicles and robots. The goal of 3D object detection is to generate 3D Bounding Boxes (BBoxes) parameterized by size, location, and orientation to locate the detected objects. Most existing methods rely heavily on LiDAR [28, 32, 35, 34, 33], because LiDAR can generate point cloud data with high-precision depth information, which enhances the accuracy of 3D object detection. However, the high cost and short service life make it difficult for LiDAR to be widely used in practice. Although binocular camera-based methods [21, 30, 17, 11, 7] achieve good detection results, this is still not a cheap option, and there are often difficulties in calibrating binocular cameras. In contrast, the monocular camera is cost-effective, very easy to assemble, and can provide a wealth of visual information for 3D object detection. Monocular 3D object detection has vast potential for applications, such as self-driving vehicles and delivery robots.

Monocular 3D object detection is an extremely challenging task without the depth provided during the imaging process. To address this, researchers have made various attempts on the depth estimation from monocular images. For instance, [5, 2] utilize CAD models to assist in estimating the depth of the vehicle. Similarly, a pre-trained depth estimation model is adopted to estimate the depth information of the scene in [37, 1, 40]. However, such methods directly or indirectly used 3D depth ground-truth data in monocular 3D object detection. Meanwhile, the methods [3, 12] without depth estimation can also achieve high accuracy in the 3D object detection task. In this paper, we propose a 3D object detector for monocular images that achieves state-of-the-art performance on KITTI benchmark [15].

Humans can perceive how close the objects in a monocular image are from the camera. Why is that? When the human brain interprets the depth of an object, it compares the object with all other objects and the surrounding environment to obtain the difference in visual effect caused by the relative position relationship. For objects of the same size, the bigger, the closer from a fixed perspective. Inspired by this, we propose a novel Asymmetric Non-local Attention Block (ANAB) to compute the response at a position as a weighted sum of the features at all positions. Inspired by [10, 46], we use both the local features in multiple scales and the features that can represent the global information to learn the depth-wise features. The multi-scale features can reduce computational costs. The attentive maps in multiple scales shows an explicit correlation between the sampling spatial resolution and the depth of the objects.

In one-stage monocular 3D object detection methods, 2D and 3D BBoxes are detected simultaneously. However, for anchor-based methods, there exists feature mismatching in the prediction of 2D and 3D BBoxes. This occurs for two reasons: (1) the receptive field of the feature does not match the shape of the anchor in terms of aspect ratio and size; (2) the center of the anchor, generally considered as the center of the receptive field for the feature map, does not overlap with the center of the object. The misalignment affects the performance of 3D object detection. Thus, we propose a two-step feature alignment method, aiming at aligning the features in 2D and 3D BBox regression. In the first step, we obtain the target region according to the classification confidence scores for the pre-defined anchors. This allows the receptive field of the feature map to focus on the pre-defined anchor regions with high confidence scores. In the second step, we use the prediction results of the 2D/3D center to compute the feature offset that can mitigate the gap between the predictions and its corresponding feature map.

We summarize our contributions as follows:

- We propose a simple but very efficient monocular 3D single-stage object detection (M3DSSD) method. The M3DSSD achieves significantly better performance than the monocular 3D object detection methods on the KITTI dataset for car, pedestrian, and cyclist object class using one single model, in both 3D object detection and bird's eye view tasks.
- We propose a novel asymmetric non-local attention block with multi-scale sampling for the depth-wise feature extraction, thereby improving the accuracy of the object depth estimation.
- We propose a two-step feature alignment module to overcome the mismatching in the size of the receptive field and the size of the anchor, and the misalignment in the object center and the anchor center.

## 2. Related Work

In order to estimate depth information in monocular images, researchers have proposed many different approaches. For instance, [42, 8, 23] utilize point cloud data to obtain accurate 3D spatial information. Pointfusion [42] uses two networks to process images and raw point cloud data respectively, and then fuses them at the feature level. MV3D [8] encodes the sparse point cloud with a multi-view representation and performs region-based feature fusion. Liang et al. [23] exploit the point-wise feature fusion mechanism between the feature maps of LiDAR and images. LiDAR point cloud and image fusion methods have achieved promising performance. However, LiDAR cannot be widely used in practice at present due to its expensive price.

CAD models of vehicles are also used in monocular 3D object detection. Barabanau et al. [2] detects 3D objects via geometric reasoning on key points. Specifically, the dimensions, rotation, and key points of a car are predicted by a convolutional neural network. Then, according to the key points' coordinates on the image plane and the corresponding 3D coordinates on the CAD model, simple geometric reasoning is performed to obtain the depth and 3D locations of the car. Deep MANTA [5] predicts the similarity between a vehicle and a predefined 3D template, as well as the coordinates and visibility of key points, using a convolutional neural network. Finally, given the 2D coordinates of an object's key points and the corresponding 3D coordinates on the 3D template, the vehicle's location and rotation can be solved by a standard 2D/3D matching [19]. However, it is difficult to collect CAD models in all kinds of vehicles.

Monocular depth estimation networks are adopted in [37, 13, 25, 1, 41, 4] to estimate depth or disparity maps. Most of the methods transform the estimated depth map into a point cloud representation and then utilize the approaches based on LiDAR to regress the 3D BBoxes. The performance of these methods relies heavily on the accuracy of the depth map. D4LCN [13] proposed a new type of convolution, termed depth-guided convolution, in which the weights and receptive fields of convolution can be automatically learned from the estimated depth. The projection of the predicted 3D BBox should be consistent with the predicted 2D BBox. This is utilized to build geometric constraints in [27, 14, 26] to determine the depth. Thanks to the promising performance of convolutional neural networks in 2D object detection, more and more approaches [3, 20, 31, 30, 29, 24, 12, 16] have been proposed to directly predict 3D BBoxes using well-designed convolutional neural network for monocular 3D object detection. GS3D [20] proposed a two-stage 3D object detection framework, in which the surface feature extraction is utilized to eliminate the problem of representation ambiguity brought by using a 2D bounding box. M3D-RPN [3] proposed an anchor-based single-stage 3D object detector that generates both 2D and 3D BBoxes simultaneously. M3D-RPN achieves good performance, but it does not solve the problem of feature misalignment.

## 3. Method

In this section, we describe the proposed M3DSSD, which consists of four main components: the backbone, the feature alignment, the asymmetric non-local attention block, and the 2D-3D prediction heads, as shown in Fig. 1. The details of each component are described below.

### 3.1. Backbone

Following [43], we adopt the Deep Layer Aggregation network DLA-102 as the backbone. To adaptively change

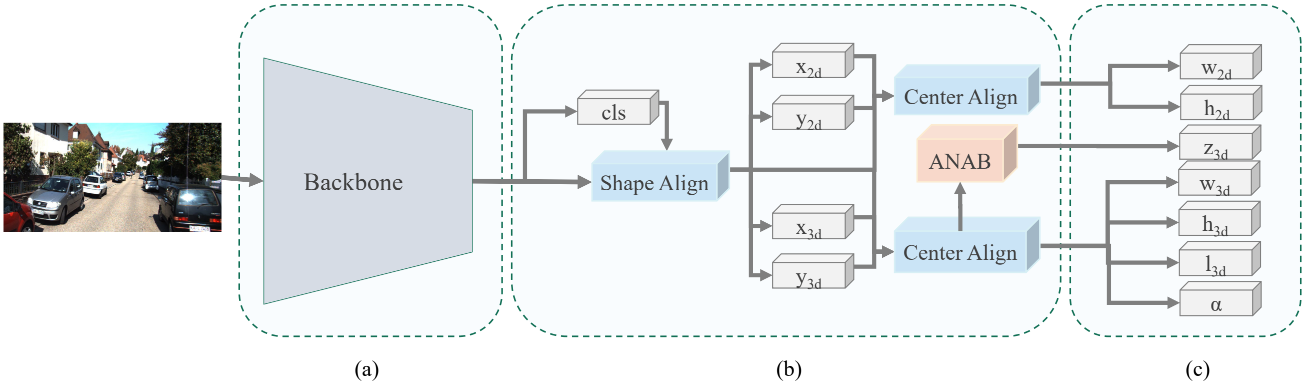

Figure 1: The architecture of M3DSSD. (a) The backbone of the framework, which is modified from DLA-102 [43]. (b) The two-step feature alignment, classification head, 2D/3D center regression heads, and ANAB especially designed for predicting the depth $z_{3d}$. (c) Other regression heads.

the receptive field and enhance the feature learning [44, 24], all the convolution in hierarchical aggregation connections are replaced with Deformable Convolution (DCN) [45]. The down-sampling ratio is set to 8, and the size of the output feature map is $256 \times H/8 \times W/8$, where $H$ and $W$ are the height and width of the input image.

## 3.2. Feature Alignment

Anchor-based methods often suffer from feature mismatching. On one hand, this occurs if the receptive field of the feature does not match the shape of the anchor in terms of aspect ratio and size. On the other hand, the center of the anchor, generally considered as the center of the receptive field of the feature, might not overlap with the center of the object. The proposed feature alignment consists of shape alignment and center alignment: (1) shape alignment aims at forcing the receptive field of the feature map to focus on the anchor with the highest classification confidence score; (2) center alignment is performed to reduce the gap between the feature on the center of the object and the feature that represents the center of the anchor. Different from previous feature alignment methods [10, 38] that are applied to one-stage object detection via a two-shot regression, the proposed feature alignment can be applied in one shot, which is more efficient and self-adaptive.

**Shape alignment** We can first obtain the foreground region according to the classification results. Then, the receptive fields of the features in the foreground regions can focus on the anchor with the highest confidence scores, as shown in Fig. 2. This makes sense because among all the anchors located at the same position, the one with the highest confidence is more likely to remain after the NMS algorithm. We use a convolution termed AlignConv in the implementation of shape alignment and center alignment. AlignConv is similar to the deformable convolution [45]. The difference is that the offset of the former is computed from the prediction results. The normal convolution can be considered as a special case of AlginConv where the offset equals zero. Unlike the RoI convolution proposed in [9], we align the shape of the receptive field or the location of the center in one shot. When performing shape alignment on the feature map with stride $S$, the offset $(O_i^{sa}, O_j^{sa})$ of the convolution with kernel size $k_h \times k_w$ is defined as:

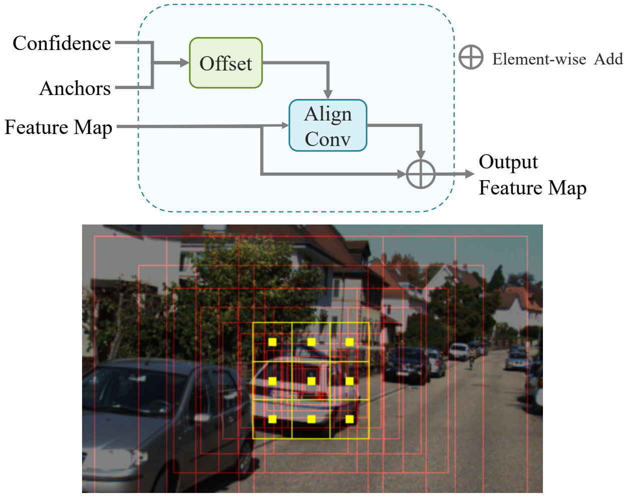

Figure 2: The architecture of shape alignment and the outcome of shape alignment on objects. The yellow squares indicate the sampling location of the AlignConv, and the anchors are in red.

$$O_i^{sa} = (\tfrac{h_a}{S \times k_h} - 1) \times (i - \tfrac{k_h}{2} + 0.5), \tag{1}$$

$$O_j^{sa} = (\tfrac{w_a}{S \times k_w} - 1) \times (j - \tfrac{k_w}{2} + 0.5), \tag{2}$$

where $h_a, w_a$ are the height and the width of the anchor with the highest confidence.

**Center alignment** The purpose of center feature alignment is to align the feature at the center of the object to the feature that represents the center of the anchor. As shown

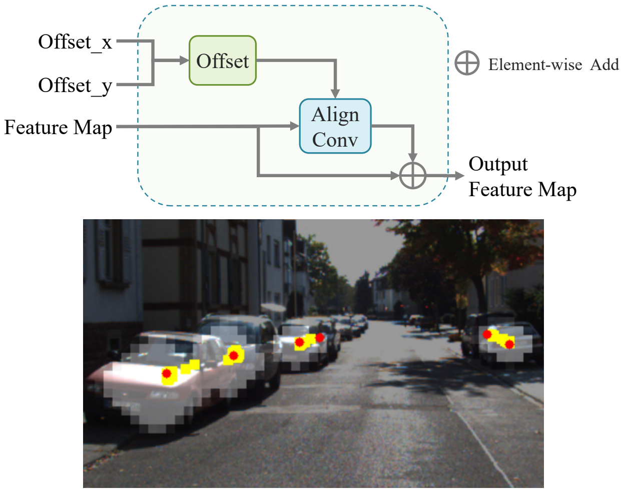

Figure 3: The architectures of center alignment and the outcome of the center alignment. When applying center alignment to objects, the sampling locations on the foreground regions (in white) all concentrate on the centers of objects (in yellow) after center alignment, which are near to the true centers of objects (in red).

in Fig. 3, the prediction results from the 2D/3D center regression are used to compute the offset of the convolution on the feature map with stride $S$:

$$O_i^{ca} = \frac{y_r}{S}, \quad O_j^{ca} = \frac{x_r}{S}, \tag{3}$$

where $x_r$ and $y_r$ are the prediction results of the 2D/3D centers in objects, respectively. As shown in Fig. 3, when center alignment with a $1 \times 1$ convolutional kernel is applied to the feature map, the sampling position is adaptively concentrated on the center of objects.

### 3.3. Asymmetric Non-local Attention Block

We propose a novel asymmetric non-local attention block to improve the accuracy of the depth $z_{3d}$ prediction by extracting the depth-wise features that can represent the global information and the long-range dependencies. The standard non-local block [39] is promising in establishing long-range dependencies, but its computational complexity is $O(N^2C)$, where $N = h \times w$, $h$, $w$ and $C$ indicate the spatial height, width, and channel number of the feature map, respectively. This is very computationally expensive and inefficient compared to normal convolutions. Thus, the applications are limited. The Asymmetric Pyramid Non-local Block [46] reduces the computational cost by decreasing the number of feature descriptors using pyramid pooling. However, pyramid pooling on the same feature map may lead to features with low resolution being replaced with high-resolution features. In other words, there exists redundancy in the computational cost regarding the image resolution. As such, we propose an Asymmetric Non-local Attention Block (ANAB), which can extract multi-scale features to enhance the feature learning with a low computational cost.

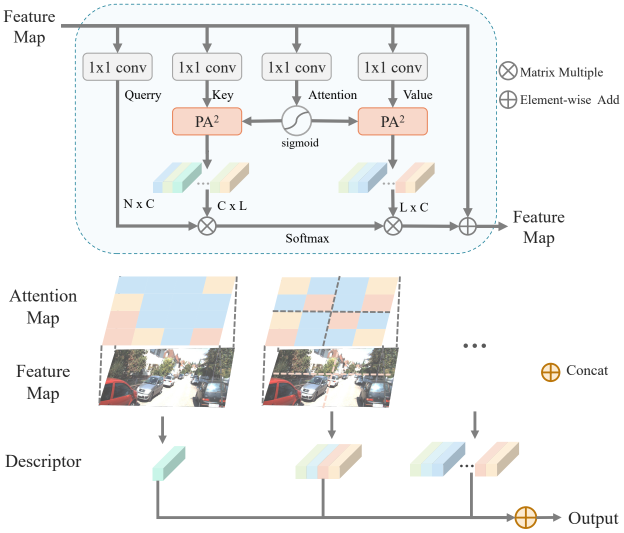

Figure 4: Top: Asymmetric Non-local Attention Block. The key and query branches share the same attention maps, which forces the key and value to focus on the same place. Bottom: Pyramid Average Pooling with Attention ($PA^2$) that generates different level descriptors in various resolutions.

As shown at the top of Fig. 4, we use the pyramidal features of the $key$ and $value$ branches to reduce the computational cost. The bottom of Fig. 4 illustrates the Pyramid Average Pooling with Attention ($PA^2$) module. The different levels of the feature pyramid have different receptive fields, thereby modeling regions with different scales. Two matrix multiplications are performed in ANAB. First, the similarity matrix between the reshaped feature matrices $\mathbf{M}_Q$ and $\mathbf{M}_K$ obtained from $querry$ and $key$ is defined as:

$$\mathbf{M}_S = \mathbf{M}_Q \times \mathbf{M}_K^T, \quad \mathbf{M}_Q \in \mathbb{R}^{N \times C}, \mathbf{M}_K \in \mathbb{R}^{L \times C}. \tag{4}$$

Then, the softmax function is used to normalize the last dimension of the similarity matrix and multiply it by the reshaped feature matrix $\mathbf{M}_V$ obtained from $value$ to get the output:

$$\mathbf{M}_{out} = Softmax(\mathbf{M}_S) \times \mathbf{M}_V, \quad \mathbf{M}_V \in \mathbb{R}^{L \times C}. \tag{5}$$

where $L$ is the number of features after sampling. The standard non-local block [39] has computational complexity $O(N^2C)$, while the complexity of ANAB is $O(NLC)$. In practice, $L$ is usually significantly smaller than $N$. In our case, we use a four-level downsampling strategy on the feature map $48 \times 160$. The resolution of the four-level feature pyramid is set to $i \in \{1\times1, 4\times4, 8\times8, 16\times16\}$, the sum of which is the total number $L$ of features after downsampling. So $L = 377$ is much smaller than $N = 7680$.

Another effective component of ANAB is the application of the multi-scale attention maps to the $key$ and $value$

branches in $PA^2$ module, as shown at the bottom of Fig. 4. The motivation is to keep the key information of the origin feature map when greatly reducing the dimensions of matrices $\mathbf{M}_K$ and $\mathbf{M}_V$ from $N \times C$ to $L \times C$. The spatial attention maps generated by a $1 \times 1$ convolutional layer are used as weights. This module adaptively adjusts the weights to pay more attention to the useful information and suppress the less useful information. The attentive map can be treated as a mask performed on multi-scale features. We use the average pooling with attention to downsample the feature maps. Such a weighted average pooling operation offers an efficient way to gather the key features.

### 3.4. 2D-3D Prediction and Loss

**Anchor definition.** We adopt a one-stage 2D-3D anchor-based network as our detector. To detect the 2D and the 3D BBoxes simultaneously, our predefined anchor contains the parameters of both the 2D BBoxes $[w, h]_{2d}$ and the 3D BBoxes $[z, w, h, l, \alpha]_{3d}$. $\alpha$ is the observation angle of the object that measures the angle at which the camera views the object. Compared with the rotation angle of the object, the observation angle is more meaningful for monocular 3D object detection [26]. The dimension of the object is given by $[w, h, l]_{3d}$. We project the center of the object onto the image plane to encode the 3D location of the object into the anchor:

$$\begin{bmatrix} X_p & Y_p & 1 \end{bmatrix}^{\mathrm{T}} \cdot Z_p = \mathbf{K} \cdot \begin{bmatrix} X & Y & Z & 1 \end{bmatrix}^{\mathrm{T}}, \tag{6}$$

where $(X_p, X_p)$ are the coordinates of the 3D point projected onto the image plane, and $(X, Y, Z)$ are the 3D space coordinates in the camera coordinate system. $K \in \mathbb{R}^{3\times 4}$ is the intrinsic camera matrix, which is known at both the training and testing phase. We obtain the 3D parameters of each anchor by computing the mean of the corresponding 3D parameters of the objects whose intersection over union (IoU) is greater than a given threshold (0.5) with the predefined 2D anchors $[w, h]_{2d}$.

**Output transformation.** Given the detection outputs $cls$, $[t_x, t_y, t_w, t_h]_{2d}$ and $[t_x, t_y, t_z, t_w, t_h, t_l, t_\alpha]_{3d}$ for each anchor, the 2D BBox $[X, Y, W, H]_{2d}$ and 3D BBox $[X, Y, Z, W, H, L, A]_{3d}$ can be restored from the output of the detector by:

$$\begin{aligned}
&[X, Y]_{2d} = [t_x, t_y]_{2d} \otimes [w, h]_{2d} + [x, y]_{2d} \\
&[W, H]_{2d} = \exp([t_w, t_h]_{2d}) \otimes [w, h]_{2d} \\
&[X_p, Y_p]_{3d} = [t_x, t_y]_{3d} \otimes [w, h]_{2d} + [x, y]_{2d} \\
&[W, H, L]_{3d} = \exp([t_w, t_h, t_l]_{3d}) \otimes [w, h, l]_{3d} \\
&[Z_p, A]_{3d} = [t_z, t_\alpha] + [z, \alpha]_{3d},
\end{aligned} \tag{7}$$

where $\otimes$ denotes the element-wise product and $A$ is the rotation angle. During the inference phase, $[X, Y, Z]_{3d}$ can be obtained by projecting $[X_p, Y_p, Z_p]$ back to the camera coordinate system using the inverse operation of Eqn. 6.

**Loss function.** We employ a multi-task loss function to supervise the learning of the network, which is composed of three parts: a classification loss, 2D BBox regression loss, and 3D BBox regression loss. The 2D regression and 3D regression loss are regularized with weights $\lambda_1$ and $\lambda_2$:

$$L = L_{cls} + \lambda_1 L_{2d} + \lambda_2 L_{3d}, \tag{8}$$

For the classification task, we employ the standard cross entropy loss function:

$$L_{cls} = -\log\left(\frac{\exp(c')}{\sum \exp(c_i)}\right). \tag{9}$$

For the 2D BBox regression task, we use $-\log(IoU)$ as the loss function for the ground-truth 2D BBox $\hat{b}_{2d}$ and the predicted 2D BBox $b'_{2d}$, similar to [3]:

$$L_{2d} = -\log(IoU(b'_{2d}, \hat{b}_{2d})). \tag{10}$$

A smooth L1 loss function is employed to supervise the regression of 3D BBoxes:

$$\begin{aligned}
L_{3d} &= \sum_{v_{3d} \in P_{3d}} SmoothL_1(v'_{3d}, \hat{v}_{3d}), \\
P_{3d} &= \{t_x, t_y, t_z, t_w, t_h, t_l, t_\alpha\}_{3d}.
\end{aligned} \tag{11}$$

## 4. Experiments

### 4.1. Evaluation Dataset

We evaluate our framework on the challenging KITTI benchmark for 3D object detection and bird's eye view tasks. The KITTI dataset contains 7481 images with labels and 7518 images for testing, covering three main categories of objects: cars, pedestrians, and cyclists. We use common split methods [7] to divide the images with labels into the training set and the validation set. We pad the images to the size of $384 \times 1280$ in both the training and inference phase. In the training phase, in addition to the conventional data augmentation methods of random translation and horizontal mirror flipping, the random scaling operation is applied for monocular images.

### 4.2. Implementation Details

We implement our model with PyTorch. We adopt the SGD optimizer with momentum to train the network with a CPU E52698 and GPU TITAN V100, in an end-to-end manner, for 70 epochs. The momentum of the SGD optimizer is set to 0.9, and weight decay is set to 0.0005. The mini-batch size is set to 4. The learning rate increases linearly from 0 to the target learning rate of 0.004 in the first epoch and then decreases to $4\times10^{-8}$ with cosine annealing.

| Methods | Extra | $AP_{3d}(val/test)$ $IoU \geq 0.7$ Easy | Moderate | Hard | $AP_{BEV}(val/test)$ $IoU \geq 0.7$ Easy | Moderate | Hard |
|---|---|---|---|---|---|---|---|
| MonoFENet[1] | Depth | 17.54 / 8.35 | 11.16 / 5.14 | 9.74 / 4.10 | 30.21 / 17.03 | 20.47 / 11.03 | 17.58 / 9.05 |
| AM3D[25] | Depth | 32.23 / 16.50 | 21.09 / 10.74 | 17.26 / 9.52 | 43.75 / 25.03 | 28.39 / 17.32 | 23.87 / 14.91 |
| D4LCN[13] | Depth | 26.97 / 16.65 | 21.71 / 11.72 | 18.22 / 9.51 | 34.82 / 22.51 | 25.83 / 16.02 | 23.53 / 12.55 |
| GS3D[20] | None | 13.46 / 4.47 | 10.97 / 2.90 | 10.38 / 2.47 | - / 8.41 | - / 6.08 | - / 4.94 |
| MonoPSR[18] | None | 12.75 / 10.76 | 11.48 / 7.25 | 8.59 / 5.85 | 20.63 / 18.33 | 18.67 / 12.58 | 14.45 / 9.91 |
| MonoGRNet[29] | None | 13.88 / 9.61 | 10.19 / 5.74 | 7.62 / 4.25 | - / 18.19 | - / 11.17 | - / 8.73 |
| SS3D[16] | None | 14.52 / 10.78 | 13.15 / 7.68 | 11.85 / 6.51 | - / 16.33 | - / 11.52 | - / 9.93 |
| MonoDIS[36] | None | 18.05 / 10.37 | 14.98 / 7.94 | 13.42 / 6.40 | 24.26 / 17.23 | 18.43 / 13.19 | 16.95 / 11.12 |
| MonoPair[12] | None | - / 13.04 | - / 9.99 | - / 8.65 | - / 19.28 | - / 14.83 | - / **12.89** |
| SMOKE[24] | None | 14.76 / 14.03 | 12.85 / 9.76 | 11.50 / 7.84 | 19.99 / 20.83 | 15.61 / 14.49 | 15.28 / 12.75 |
| M3D-RPN[3] | None | 20.27 / 14.76 | 17.06 / 9.71 | 15.21 / 7.42 | 25.94 / 21.02 | 21.18 / 13.67 | 17.90 / 10.23 |
| RTM3D[22] | None | 20.77 / 14.41 | 16.86 / 10.34 | 16.63 / 8.77 | 25.56 / 19.17 | 22.12 / 14.20 | 20.91 / 11.99 |
| M3DSSD(ours) | None | **27.77 / 17.51** | **21.67 / 11.46** | **18.28 / 8.98** | **34.51 / 24.15** | **26.20 / 15.93** | **23.40** / 12.11 |

Table 1: AP scores on $val$ and $test$ set of 3D object detection and bird's eye view for cars.

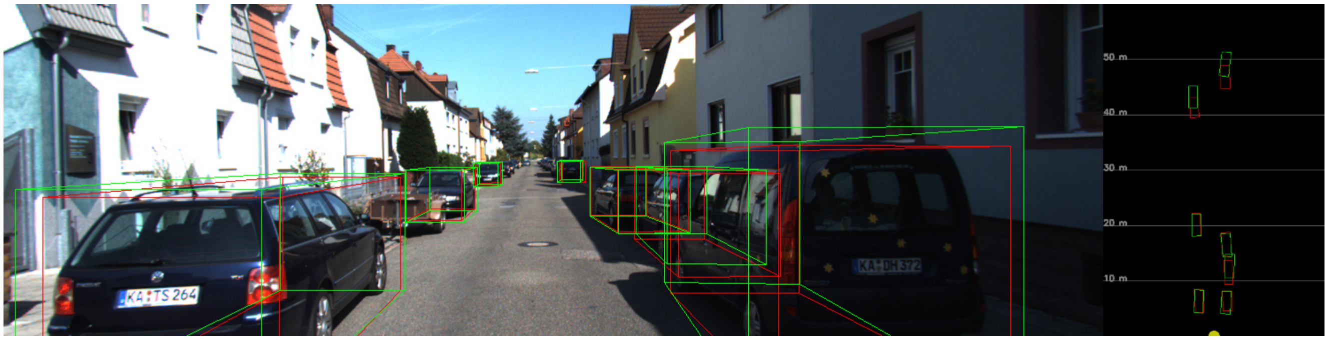

Figure 5: Qualitative results of 3D detection (left) and bird's eye view (right), prediction in green and ground-truth in red.

Terms $\lambda_1$ and $\lambda_2$ in Eqn. 8 are both set to 1.0. We lay 36 anchors on each pixel of the feature map, the size of which increases from 24 to 288 following the exponential function of $24 \times 12^{i/11}, i \in \{0, 1, 2, \ldots, 11\}$, and the aspect ratio is set to $\{0.5, 1.0, 1.5\}$. We apply online hard-negative mining by sampling the top 20% high loss boxes in each minibatch in the training phase. In the inference phase, we apply NMS with 0.4 IoU criteria on the 2D BBox and filter out the objects with a confidence lower than 0.75. The post-optimization algorithm proposed in [3] is used to make the rotation angle more reasonable. The algorithm uses projection consistency to optimize the rotation angle. The rotation angle is optimized iteratively to minimize the L1 loss of the projection of the predicted 3D BBox and the predicted 2D BBox.

### 4.3. Performance Evaluation

We set the network after removing the feature alignment module and ANAB from M3DSSD as the baseline. More specifically, for the baseline, the feature map output from the backbone is directly used for classification and 2D BBox regression and 3D BBox regression.

We evaluate our framework on the KITTI benchmark for both bird's eye view and 3D object detection tasks. The average precision (AP) of Intersection over Union (IoU) is used as the metric for evaluation in both tasks and it is divided into easy, moderate, and hard according to the height, occlusion, and truncation level of objects. Note that the official KITTI evaluation has been using $AP|_{R40}$ with 40 recall points instead of $AP|_{R11}$ with 11 recall points since October 8, 2019. However, most previous methods evaluated on the validation used $AP|_{R11}$. Thus, we report the $AP|_{R40}$ for the test dataset and $AP|_{R11}$ for the validation dataset for a fair comparison. We set the threshold of IoU to 0.7 for cars and 0.5 for pedestrians and cyclists as the same as the official settings. Fig. 5 shows qualitative results for 3D object detection and bird's eye view. The detection results and depth predictions are less accurate with further distance. The videos of 3D object detection results and the additional results can be found in the supplemental material.

**Bird's eye view.** The bird's eye view task is to detect objects projected on the ground, which is closely related to

| Methods | *Pedestrian* $AP_{3D}/AP_{bev}$ Easy | Moderate | Hard | *Cyclist* $AP_{3D}/AP_{bev}$ Easy | Moderate | Hard |
|---|---|---|---|---|---|---|
| M3D-RPN[3] | 4.92 / 5.65 | 3.48 / 4.05 | 2.94 / 3.29 | 0.94 / 1.25 | 0.65 / 0.81 | 0.47 / 0.78 |
| D4LCN[13] | 4.55 / 5.06 | 3.42 / 3.86 | 2.83 / 3.59 | 2.45 / 2.72 | **1.67** / 1.82 | 1.36 / **1.79** |
| SS3D[16] | 2.31 / 2.48 | 1.78 / 2.09 | 1.48 / 1.61 | **2.80 / 3.45** | 1.45 / 1.89 | 1.35 / 1.44 |
| M3DSSD(ours) | **5.16 / 6.20** | **3.87 / 4.66** | **3.08 / 3.99** | 2.10 / 2.70 | 1.51 / **2.01** | **1.58** / 1.75 |

Table 2: Detection performance for pedestrians and cyclists on *test* set, at 0.5 $IoU$ threshold.

the 3D location of objects. The detection results for cars on both the *val* and *test* set are reported in Tab. 1. M3DSSD achieves state-of-the-art performance on the bird's eye view task compared to approaches with and without depth estimation. Our method has significant improvement compared to the methods without depth estimation.

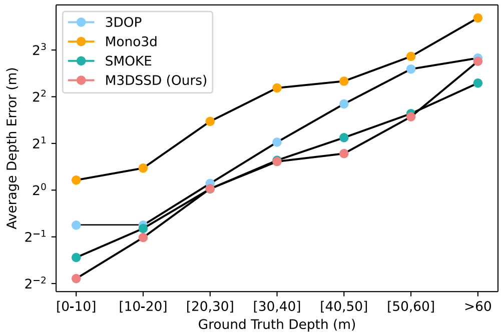

Figure 6: The average depth estimation error varies along with the ground truth depth. Best viewed in color.

| Methods | $AP_{3d}/AP_{BEV}$ $IoU \geq 0.7$ Easy | Mod. | Hard |
|---|---|---|---|
| Baseline w/ ANAB † | 25.70 / 33.48 | 19.02 / 24.79 | 17.31 / 20.15 |
| †w/ Shape Alignment | 27.26 / 33.64 | 21.56 / 25.24 | 18.07 / 22.81 |
| †w/ Center Alignment | 27.33 / **34.85** | 21.51 / 25.96 | 18.03 / 23.26 |
| †w/ Full Alignment | **27.77** / 34.51 | **21.67 / 26.20** | **18.28 / 23.40** |

Table 3: Ablation study on feature alignment.

**3D object detection for cars.** The 3D object detection task aims to detect 3D objects in the camera coordinate system, which is more challenging than the bird's eye view task due to the additional y-axis. Compared with the approaches without depth estimation, Tab. 1 shows that M3DSSD achieves better performance in both the *val* and *test* set. Note that M3DSSD is better than most of the approaches with depth estimation. Further, our method achieves competitive performance against D4LCN that adopts a pre-trained model for depth estimation [13].

Fig. 6 shows the average depth estimation error with the different ground truth depth ranges [24]. We compared our proposed method with SMOKE [24], Mono3D [6] and 3DOP [7] on the same validation set. Fig. 6 demonstrates that the proposed M3DSSD achieves better performance at all distance ranges, except for the distance greater than 60m, where the number of samples is usually small.

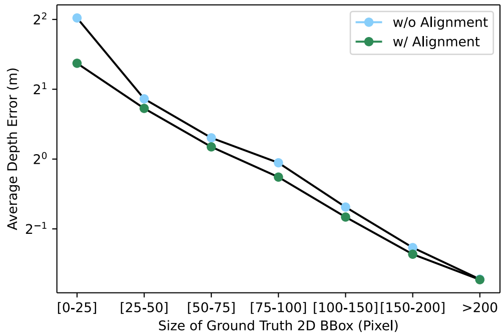

Figure 7: The average depth estimation error varies along with the size of objects that is the average of the length and the width of the 2D BBox. Best viewed in color.

**3D object detection for pedestrians and cyclists.** Compared with cars, 3D object detection for pedestrians and cyclists is more challenging. This is because the size of pedestrians and bicycles is relatively small. In addition, people are non-rigid bodies, and their shapes vary a lot, thereby making it difficult to locate pedestrians and cyclists. We report the detection results for pedestrians and cyclists on the test set of the KITTI benchmark in Tab. 2. Since some methods did not report the pedestrian and the cyclist results, we compare our model with M3D-RPN [3], D4LCN [13], and SS3D [16]. Our model achieves competitive performance in both 3D detection and bird's eye view tasks for pedestrians and bicycles, especially for the pedestrian category. Note that we train only one single model to detect the three object classes simultaneously.

## 4.4. Ablation Study

**Feature alignment.** We evaluate the feature alignment strategies, including shape alignment, center alignment, and full alignment (both center alignment and shape alignment). As shown in Tab. 3, that the proposed shape alignment, center alignment, and full alignment achieve better results compared to the case without alignment.

Fig. 7 illustrates the average depth estimation error varies with the size of objects for the model with and without fea-

| Methods | $AP_{3d}/AP_{BEV}$ $IoU \geq 0.7$ | | | Methods | GPU time | GPU memory |
|---|---|---|---|---|---|---|
| | Easy | Moderate | Hard | | (ms) | (Gbyte) |
| baseline | 23.40 / 28.66 | 18.32 / 23.53 | 16.62 / 19.54 | Non-local [39] | 5.89 / 104.12 | 1.97 / 15.67 |
| ANB | 23.65 / 29.19 | 18.47 / 23.65 | 16.54 / 19.50 | ANB | **1.68 / 5.92** | **1.09 / 1.43** |
| ANAB | **25.70 / 33.48** | **19.02 / 24.79** | **17.31 / 20.15** | ANAB | 1.86 / 6.76 | 1.22 / 1.91 |

Table 4: Ablation study on non-local blocks with detection accuracy, GPU time, and memory for different input sizes.

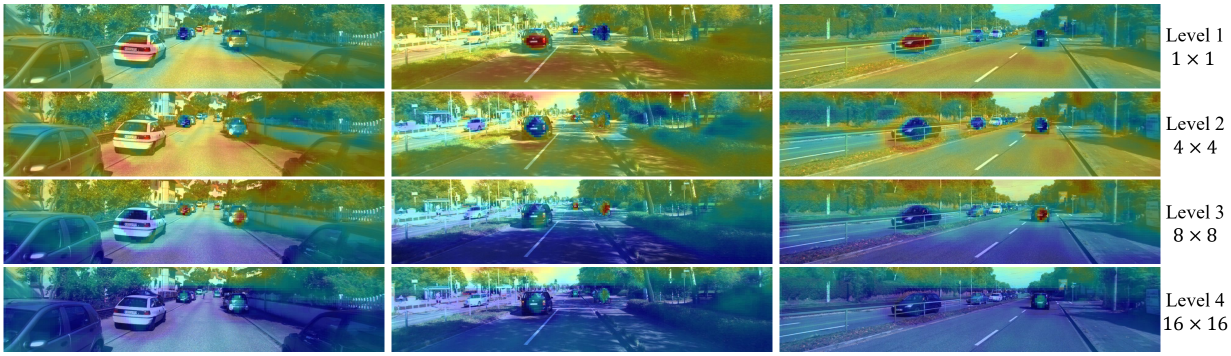

Figure 8: Visualization for attention maps in $PA^2$ with a four-level feature pyramid $\{1 \times 1, 4 \times 4, 8 \times 8, 16 \times 16\}$.

ture alignment. The x-axis is set as the size of the 2D BBox $(w_{2d}+h_{2d})/2$. It shows that the proposed feature alignment module is effective on objects of different sizes, especially for the small objects in $[0 - 25]$. This also explains why M3DSSD outperforms other methods in small object detection such as pedestrians and cyclists.

**Asymmetric non-local attention block.** We compare the Asymmetric Non-local Block (ANB), and our proposed Asymmetric Non-local Attention Block (ANAB), which applies pyramid average pooling on the feature map with attentions. We use the same sampling size for both methods. Tab. 4 shows that the network with ANAB achieves the best performance. With a similar computational time, the proposed ANAB has better detection accuracy than ANB. Meanwhile, both methods cost much less GPU time and memory than the standard non-local block [39]. The attention module costs a little more consuming time with significant improvement, especially in easy tasks. Tab. 4 on the right shows the GPU time and memory regarding the input size $[1, 256, 48, 160]$ and $[1, 256, 96, 320]$. This shows that the computational cost is closer to the theoretical analysis in Sect. 3.3 with a larger input size. ANAB has extra pooling layers, convolutional layers, and an element-wise multiplication, which are not considered in the theoretical analysis.

In ANAB, the attention maps are assigned to the multi-scale pooling operations for the depth-wise feature extraction. Fig. 8 shows that the attention map for $1 \times 1$ feature pyramid has larger weights on the objects which are close to the camera, while the attention map for the higher-level feature pyramid assigns larger weights on the objects that are away from the camera. The attention maps in different levels show a correlation between the resolution of the feature pyramid and the object depth. This lies in the fact that the feature pyramid with low resolution has a large receptive field that is sensitive to the object in large size, while the feature pyramid with high resolution has a small receptive field that is sensitive to the object in small size. For the size of the same-class object from a fixed perspective, the smaller, the farther. The depth-wise attention maps enhance the capability of perceiving the depth of objects, thereby improving the performance of object depth estimation.

## 5. Conclusion

In this work, we propose a simple and very effective monocular single-stage 3D object detector. We present a two-step feature alignment approach to address the feature mismatching, which enhances the feature learning for object detection. The asymmetric non-local attention block enables the network to extract depth-wise features, which improves the performance of the depth prediction in the regression head. Compared to the methods with or without the estimated depth as an extra input, M3DSSD achieves better performance on the challenging KITTI dataset for car, pedestrian, and cyclist object class using one single model, for both bird's eye view and 3D object detection.

**Acknowledgement:** This work was supported in part by the National Key Research and Development Program of China (2018YFE0183900). Hang Dai would like to thank the support from MBZUAI startup fund (GR006).

# References

[1] Wentao Bao, Bin Xu, and Zhenzhong Chen. Monofenet: Monocular 3d object detection with feature enhancement networks. *IEEE Transactions on Image Processing*, 2019.

[2] Ivan Barabanau, Alexey Artemov, Evgeny Burnaev, and Vyacheslav Murashkin. Monocular 3d object detection via geometric reasoning on keypoints. *arXiv preprint arXiv:1905.05618*, 2019.

[3] Garrick Brazil and Xiaoming Liu. M3d-rpn: Monocular 3d region proposal network for object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9287–9296, 2019.

[4] Yingjie Cai, Buyu Li, Zeyu Jiao, Hongsheng Li, Xingyu Zeng, and Xiaogang Wang. Monocular 3d object detection with decoupled structured polygon estimation and height-guided depth estimation. *arXiv preprint arXiv:2002.01619*, 2020.

[5] Florian Chabot, Mohamed Chaouch, Jaonary Rabarisoa, Céline Teuliere, and Thierry 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*, pages 2040–2049, 2017.

[6] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2156, 2016.

[7] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals using stereo imagery for accurate object class detection. *IEEE transactions on pattern analysis and machine intelligence*, 40(5):1259–1272, 2017.

[8] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017.

[9] Yuntao Chen, Chenxia Han, Naiyan Wang, and Zhaoxiang Zhang. Revisiting feature alignment for one-stage object detection. *arXiv preprint arXiv:1908.01570*, 2019.

[10] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. Aˆ 2-nets: Double attention networks. In *Advances in Neural Information Processing Systems*, pages 352–361, 2018.

[11] Yilun Chen, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Dsgn: Deep stereo geometry network for 3d object detection. *arXiv preprint arXiv:2001.03398*, 2020.

[12] Yongjian Chen, Lei Tai, Kai Sun, and Mingyang Li. Monopair: Monocular 3d object detection using pairwise spatial relationships. *arXiv preprint arXiv:2003.00504*, 2020.

[13] Mingyu Ding, Yuqi Huo, Hongwei Yi, Zhe Wang, Jianping Shi, Zhiwu Lu, and Ping Luo. Learning depth-guided convolutions for monocular 3d object detection. *arXiv preprint arXiv:1912.04799*, 2019.

[14] Nils Gählert, Marina Mayer, Lukas Schneider, Uwe Franke, and Joachim Denzler. Mb-net: Mergeboxes for real-time 3d vehicles detection. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 2117–2124. IEEE, 2018.

[15] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[16] Eskil Jörgensen, Christopher Zach, and Fredrik Kahl. Monocular 3d object detection and box fitting trained end-to-end using intersection-over-union loss. *arXiv preprint arXiv:1906.08070*, 2019.

[17] Hendrik Königshof, Niels Ole Salscheider, and Christoph Stiller. Realtime 3d object detection for automated driving using stereo vision and semantic information. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 1405–1410. IEEE, 2019.

[18] Jason Ku, Alex D Pon, and Steven L Waslander. Monocular 3d object detection leveraging accurate proposals and shape reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11867–11876, 2019.

[19] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81(2):155, 2009.

[20] Buyu Li, Wanli Ouyang, Lu Sheng, Xingyu Zeng, and Xiaogang Wang. Gs3d: An efficient 3d object detection framework for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1019–1028, 2019.

[21] Peiliang Li, Xiaozhi Chen, and Shaojie Shen. Stereo r-cnn based 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7644–7652, 2019.

[22] Peixuan Li, Huaici Zhao, Pengfei Liu, and Feidao Cao. Rtm3d: Real-time monocular 3d detection from object keypoints for autonomous driving. *arXiv preprint arXiv:2001.03343*, 2020.

[23] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7345–7353, 2019.

[24] Zechen Liu, Zizhang Wu, and Roland Tóth. Smoke: Single-stage monocular 3d object detection via keypoint estimation. *arXiv preprint arXiv:2002.10111*, 2020.

[25] Xinzhu Ma, Zhihui Wang, Haojie Li, Pengbo Zhang, Wanli Ouyang, and Xin Fan. Accurate monocular 3d object detection via color-embedded 3d reconstruction for autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6851–6860, 2019.

[26] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7074–7082, 2017.

[27] Andretti Naiden, Vlad Paunescu, Gyeongmo Kim, Byeong-Moon Jeon, and Marius Leordeanu. Shift r-cnn: Deep

monocular 3d object detection with closed-form geometric constraints. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 61–65. IEEE, 2019.

[28] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017.

[29] Zengyi Qin, Jinglu Wang, and Yan Lu. Monogrnet: A geometric reasoning network for monocular 3d object localization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8851–8858, 2019.

[30] Zengyi Qin, Jinglu Wang, and Yan Lu. Triangulation learning network: from monocular to stereo 3d object detection. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7607–7615. IEEE, 2019.

[31] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. *arXiv preprint arXiv:1811.08188*, 2018.

[32] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. *arXiv preprint arXiv:1912.13192*, 2019.

[33] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–779, 2019.

[34] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[35] Weijing Shi et al. Point-gnn: Graph neural network for 3d object detection in a point cloud. *arXiv preprint arXiv:2003.01251*, 2020.

[36] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1991–1999, 2019.

[37] Jean Marie Uwabeza Vianney, Shubhra Aich, and Bingbing Liu. Refinedmpl: Refined monocular pseudolidar for 3d object detection in autonomous driving. *arXiv preprint arXiv:1911.09712*, 2019.

[38] Jiaqi Wang, Kai Chen, Shuo Yang, Chen Change Loy, and Dahua Lin. Region proposal by guided anchoring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2965–2974, 2019.

[39] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.

[40] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8445–8453, 2019.

[41] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3d object detection from monocular images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2345–2353, 2018.

[42] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. Pointfusion: Deep sensor fusion for 3d bounding box estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 244–253, 2018.

[43] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2403–2412, 2018.

[44] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.

[45] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9308–9316, 2019.

[46] Zhen Zhu, Mengde Xu, Song Bai, Tengteng Huang, and Xiang Bai. Asymmetric non-local neural networks for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 593–602, 2019.