This CVPR 2021 paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Objects are Different: Flexible Monocular 3D Object Detection

Yunpeng Zhang, Jiwen Lu*, Jie Zhou
Beijing National Research Center for Information Science and Technology, China
Department of Automation, Tsinghua University, China
zhang-yp19@mails.tsinghua.edu.cn; {lujiwen,jzhou}@tsinghua.edu.cn

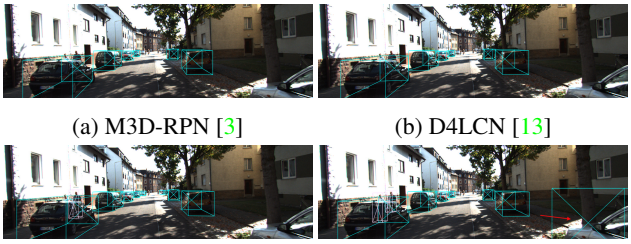
Abstract

The precise localization of 3D objects from a single image without depth information is a highly challenging problem. Most existing methods adopt the same approach for all objects regardless of their diverse distributions, leading to limited performance for truncated objects. In this paper, we propose a flexible framework for monocular 3D object detection which explicitly decouples the truncated objects and adaptively combines multiple approaches for object depth estimation. Specifically, we decouple the edge of the feature map for predicting long-tail truncated objects so that the optimization of normal objects is not influenced. Furthermore, we formulate the object depth estimation as an uncertainty-guided ensemble of directly regressed object depth and solved depths from different groups of keypoints. Experiments demonstrate that our method outperforms the state-of-the-art method by relatively 27% for the moderate level and 30% for the hard level in the test set of KITTI benchmark while maintaining real-time efficiency. Code will be available at https://github.com/zhangyp15/MonoFlex.

1. Introduction

3D object detection is an indispensable premise for machines to perceive the physical environment and has been widely used in autonomous driving and robot navigation. In this work, we focus on solving the problem with only information from monocular images. Most existing methods for 3D object detection require the LiDAR sensors [22, 33, 35, 40, 41, 49] for precise depth measurements or stereo cameras [8, 24, 37, 45] for stereo depth estimation, which greatly increases the implementation costs of practical systems. Therefore, monocular 3D object detection has been a promising solution and received much attention in the community [2, 3, 7, 10, 13, 20, 27, 31, 34].

For the challenging localization of 3D objects, most existing methods handle different objects with a unified approach. For example, [10, 25, 28, 52] utilize fully convolutional nets to predict objects of diverse distributions with shared kernels. However, we observe that the equal and joint processing of all objects can lead to unsatisfied performance: (1) As shown in Figure 1, the heavily truncated objects can be hardly detected by state-of-the-art methods [3, 13] but these objects are important to the safety of autonomous vehicles. (2) We empirically found that these hard samples can increase the learning burden and affect the prediction of general objects. Thus, unified approaches can fail in both finding every object and predicting precise 3D locations. To this end, we propose a flexible detector that considers the difference among objects and estimates their 3D locations in an adaptive way. Since the estimation of an object's 3D location is usually decomposed into finding the projected 3D center and the object depth [10, 28, 36, 52], we also consider the flexibility from these two aspects.



(a) M3D-RPN [3] (b) D4LCN [13]

(c) Baseline (d) Ours

Figure 1: Qualitative comparison among prior arts [3, 13], our baseline, and the proposed method. The cyan and pink bounding boxes represent detected cars and pedestrians. Our approach can effectively detect the heavily truncated object highlighted by the red arrow.

To localize the projected 3D center, we divide objects according to whether their projected centers are "inside" or "outside" the image. Then we represent inside objects exactly as the projected centers and outside objects as delicately chosen edge points so that two groups of objects are handled by the inner and edge regions of the feature map respectively. Considering it is still difficult for convolutional

*Corresponding author

filters to manage spatial-variant predictions, the edge fusion module is further proposed to decouple the feature learning and prediction of outside objects.

To estimate the object depth, we propose to combine different depth estimators with uncertainty estimation [18, 19]. The estimators include direct regression [10, 25, 36, 52] and geometric solutions from keypoints [2, 5]. We observe that computing depth from keypoints is usually an over-determined problem, where simply averaging results from different keypoints [5] can be sensitive to the truncation and occlusion of keypoints. As a result, we further split keypoints into M groups, each of which is exactly sufficient for solving the depth. To combine M keypoint-based estimators and the direct regression, we model their uncertainties and formulate the final estimation as an uncertainty-weighted average. The proposed combination allows the model to flexibly choose more suitable estimators for robust and accurate predictions.

Experimental results on KITTI [14] dataset demonstrate that our method significantly outperforms all existing methods, especially for moderate and hard samples. The main contributions of this paper can be summarized in two aspects: (1) We reveal the importance to consider the difference among objects for monocular 3D object detection and propose to decouple the prediction of truncated objects; (2) We propose a new formulation for object depth estimation, which utilizes uncertainties to flexibly combine independent estimators.

2. Related Work

Monocular 3D object Detection. Considering the difficulty in perceiving 3D environments from 2D images, most existing methods for monocular 3D object detection utilize extra information to simplify the task, which includes pre-trained depth estimation modules [30, 45, 46, 47], annotated keypoints [2] and CAD models [32]. Mono3D [7] first samples candidates based on the ground prior and scores them with semantic/instance segmentation, contextual information, object shape, and location prior. MonoPSR [21] estimates the instance point cloud and enforces the alignment between the object appearance and the projected point cloud for proposal refinement. Pseudo-LiDAR [45] lifts the monocular image into pseudo-LiDAR with estimated depth and then utilizes LiDAR-based detectors. AM3D [31] proposes a multi-modal fusion module to enhance the pseudo-LiDAR with color information. PatchNet [30] organizes pseudo-LiDAR into the image representation and utilizes powerful 2D CNN to boost the detection performance. Though these methods with extra information usually achieve better performance, they require more annotations for training and are usually less generalized.

Other purely monocular methods [3, 10, 27, 28, 34, 36] only utilize a single image for detection. Deep3DBox [34] presents the *MultiBin* method for orientation estimation and uses geometric constraints of 2D bounding boxes to derive 3D bounding boxes. FQNet [27] measures the fitting degree between projected 3D proposals and objects so that the best-fitted proposals are picked out. MonoGRNet [36] directly predicts the depth of objects with sparse supervision and combines early features to refine the location estimation. M3D-RPN [3] solves the problem with a 3D region proposal network and proposes the depth-aware convolutional layers to enhance extracted features. MonoPair [10] considers the pair-wise relationships between neighboring objects, which are utilized as spatial constraints to optimize the results of detection. RTM3D [25] predicts the projected vertexes of the 3D bounding box and solves 3D properties with nonlinear least squares optimization. Existing methods mostly neglect the difference among objects or only consider the general scale variance, which can suffer from predicting out-of-distribution objects and lead to downgraded performance. By contrast, our work explicitly decouples the heavily truncated objects with long-tail distribution for efficient learning and estimates the object depth by adaptively combining multiple depth estimators instead of utilizing one single method for all objects.

Uncertainty Estimation. Two major types of uncertainty are usually studied in Bayesian modeling [18]. The epistemic uncertainty describes the uncertainty of the model parameters, while the aleatoric uncertainty can capture the noise of observations, whose applications in object detection have been explored in [10, 11, 15]. Gaussian YOLO [11] models the uncertainty of predicted 2D boxes to rectify the detection scores. [15] predicts the bounding box as a Gaussian distribution and formulates the regression loss as the KL divergence. MonoPair [10] uses uncertainty to provide weights for the post-optimization between predicted 3D locations and pair-wise constraints. In this paper, we model the uncertainties of estimated depths from multiple estimators, which are used to quantify their contributions to the final combined prediction.

Ensemble Learning. Ensemble learning [1, 12, 17, 23, 39] generates multiple models strategically and combines their predictions for better performance. Traditional ensemble methods include bagging, boosting, stacking, gating network, and so on. [17] uses a gating network to combine the mixture of experts for classification. [1] proposes a tree-structured gate to hierarchically weight different experts for face alignment. Ensemble learning generally assumes the learners have identical structures but are trained with different samples or initializations, while our multiple depth estimators function in respectively different ways and are also supervised by substantially different loss functions. Therefore, we propose to formulate the combination as an uncertainty-guided average of all predictions.

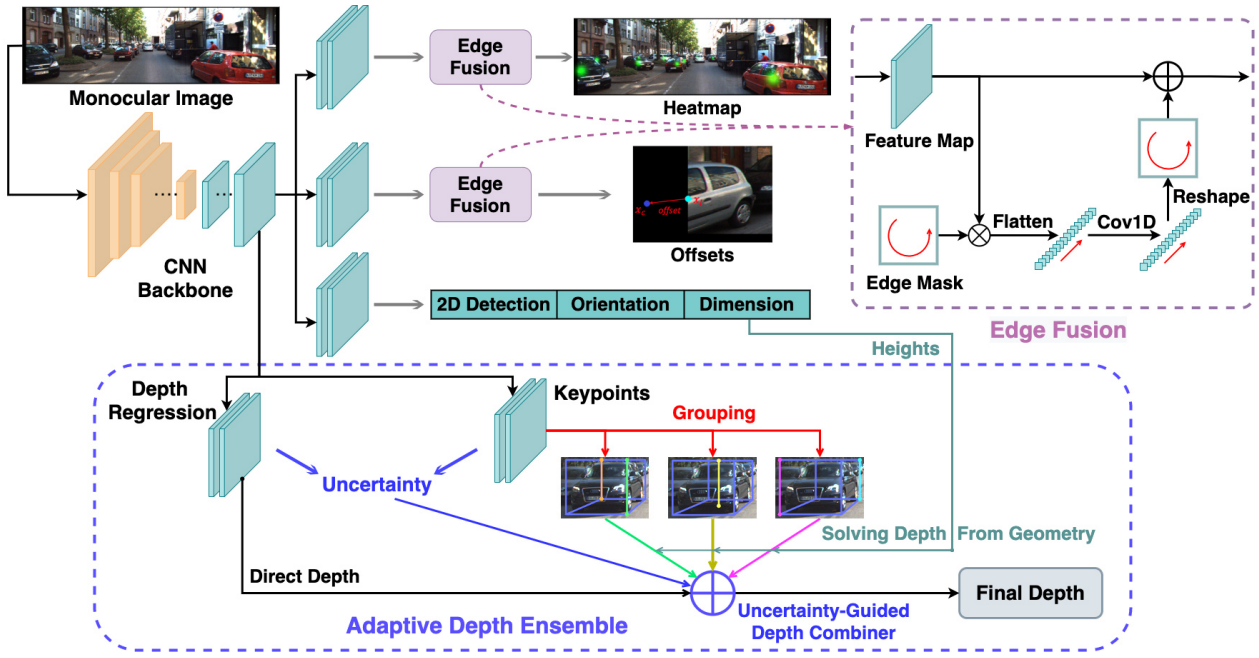


Figure 2: Overview of our framework. First, the CNN backbone extracts feature maps from the monocular image as the input for multiple prediction heads. The image-level localization involves the heatmap and offsets, where the edge fusion modules are used to decouple the feature learning and prediction of truncated objects. The adaptive depth ensemble adopts four methods for depth estimation and simultaneously predicts their uncertainties, which are utilized to form an uncertainty-weighted prediction.

3. Approach

3.1. Problem Statement

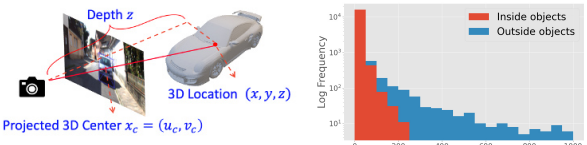
The 3D detection of an object involves estimating its 3D location (x, y, z), dimension (h, w, l), and orientation θ. The dimension and orientation can be directly inferred from appearance-based clues, while the 3D location is converted to the projected 3D center $\boldsymbol{x_c} = (u_c, v_c)$ and the object depth z as shown in Figure 3(a) and (1):

$$x = \frac{(u_c - c_u)z}{f}, \quad y = \frac{(v_c - c_v)z}{f} \tag{1}$$

where (c_u, c_v) is the principle point and f is the focal length. To this end, the whole problem is decomposed into four independent subtasks.

3.2. Framework Overview

As shown in Figure 2, our framework is extended from CenterNet [52], where objects are identified by their representative points and predicted by peaks of the heatmap. Multiple prediction branches are deployed on the shared backbone to regress objects' properties, including the 2D bounding box, dimension, orientation, keypoints, and depth. The final depth estimation is an uncertainty-guided combination of the regressed depth and the computed depths from estimated keypoints and dimensions. We present the design of decoupled representative points for normal and truncated objects in Section 3.3 and then introduce the regression of visual properties in Section 3.4. Finally, the adaptive depth ensemble is detailed in Section 3.5.



(a) Decomposing 3D locations (b) Offset distribution

Figure 3: (a) The 3D location is converted to the projected center and the object depth. (b) The distribution of the offsets $\boldsymbol{\delta_c}$ from 2D centers to projected 3D centers. Inside and outside objects exhibit entirely different distributions.

3.3. Decoupled Representations of Objects

Existing methods [10, 25, 52] utilize a unified representation $\boldsymbol{x_r}$, the center of 2D bounding box $\boldsymbol{x_b}$, for every object. In such cases, the offset $\boldsymbol{\delta_c} = \boldsymbol{x_c} - \boldsymbol{x_b}$ is regressed to derive the projected 3D center $\boldsymbol{x_c}$. We divide objects into two groups depending on whether their projected 3D centers are **inside** or **outside** the image and visualize the corresponding offsets $\boldsymbol{\delta_c}$ in Figure 3(b). Considering the sub-

stantially different offsets of two groups, the joint learning of $\boldsymbol{\delta_c}$ can suffer from long-tail offsets and we therefore propose to decouple the representations and the offset learning of inside and outside objects.

Inside Objects. For objects whose projected 3D centers are inside the image, they are directly identified by $\boldsymbol{x_c}$ to avoid regressing the irregular $\boldsymbol{\delta_c}$ like [10, 25]. Though we still need to regress the discretization error $\boldsymbol{\delta_{in}}$ due to the downsampling ratio S of the backbone CNN as in (2), it is much smaller than $\boldsymbol{\delta_c}$ and easier to regress.

$$\boldsymbol{\delta_{in}} = \frac{\boldsymbol{x_c}}{S} - \lfloor \frac{\boldsymbol{x_c}}{S} \rfloor \tag{2}$$

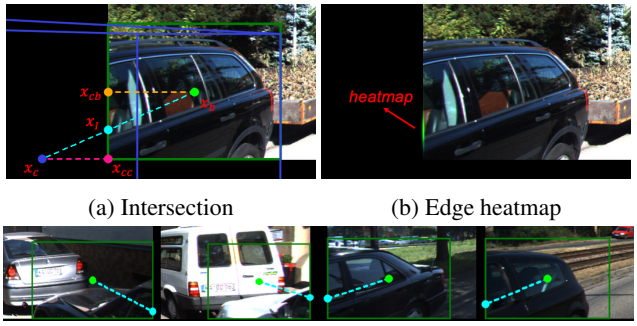
We follow [52] to generate the ground-truth heatmap for inside objects with circular Gaussian kernels centered at $\boldsymbol{x_c}$.

Outside Objects. To decouple the representation of outside objects, we propose to identify them by the intersection $\boldsymbol{x_I}$ between the image edge and the line from $\boldsymbol{x_b}$ to $\boldsymbol{x_c}$, as shown in Figure 4(a). It can be seen that the proposed intersection $\boldsymbol{x_I}$ is more physically meaningful than simply clamping $\boldsymbol{x_b}$ or $\boldsymbol{x_c}$ to the boundary. The prediction of $\boldsymbol{x_I}$ is achieved by the edge heatmap as shown in Figure 4(b), which is generated from a one-dimensional Gaussian kernel. We also compare our $\boldsymbol{x_I}$ and the commonly used $\boldsymbol{x_b}$ in Figure 4(c). Since 2D bounding boxes only capture the inside-image part of objects, the visual locations of $\boldsymbol{x_b}$ can be confusing and even on other objects. By contrast, the intersection $\boldsymbol{x_I}$ disentangles the edge area of the heatmap to focus on outside objects and offers a strong boundary prior to simplify the localization. Also, we regress the offsets from $\boldsymbol{x_I}$ to the target $\boldsymbol{x_c}$ as in (3):

$$\boldsymbol{\delta_{out}} = \frac{\boldsymbol{x_c}}{S} - \lfloor \frac{\boldsymbol{x_I}}{S} \rfloor \tag{3}$$

Edge Fusion. Though the representations of inside and outside objects are decoupled in the interior and marginal regions of the output feature, it is still difficult for shared convolutional kernels to handle spatial-variant predictions. Thus, we propose an edge fusion module to further decouple the feature learning and prediction of outside objects. As shown in the right part of Figure 2, the module first extracts four boundaries of the feature map and concatenates them into an edge feature vector in clockwise order, which is then processed by two 1D convolutional layers to learn unique features for truncated objects. Finally, the processed vector is remapped to the four boundaries and added to the input feature map. When applied to the heatmap prediction, the edge features can specialize in predicting the edge heatmap for outside objects so that the localization of inside objects is not confused. For regressing the offsets, the significant scale difference between $\boldsymbol{\delta_{in}}$ and $\boldsymbol{\delta_{out}}$ as shown in Figure 3(b) can be resolved with the edge fusion module.

Loss Functions. The penalty-reduced focal loss [26] is utilized for heatmap prediction as in [10, 25, 28]. We adopt



(a) Intersection (b) Edge heatmap

(c) Comparison between $\boldsymbol{x_I}$ and $\boldsymbol{x_b}$

Figure 4: Representations of outside objects. (a) The intersection $\boldsymbol{x_I}$ between the image edge and the line from $\boldsymbol{x_b}$ to $\boldsymbol{x_c}$ is used to represent the truncated object. (b) The edge heatmap is generated with 1D Gaussian distribution whose kernel size is proportional to the size of the 2D bounding box. (c) The always-on-edge intersection $\boldsymbol{x_I}$ (cyan) is a better representation than the 2D center $\boldsymbol{x_b}$ (green) for heavily truncated objects. Best viewed in color.

L1 loss for regressing $\boldsymbol{\delta_{in}}$ and log-scale L1 loss for $\boldsymbol{\delta_{out}}$ because it is more robust to extreme outliers. The offset loss is computed as (4):

$$L_{off} = \begin{cases} |\boldsymbol{\delta_{in}} - \boldsymbol{\delta_{in}^*}| & \text{if inside} \\ \log\left(1 + |\boldsymbol{\delta_{out}} - \boldsymbol{\delta_{out}^*}|\right) & \text{otherwise} \end{cases} \tag{4}$$

where $\boldsymbol{\delta_{in}}$ and $\boldsymbol{\delta_{out}}$ refer to predictions and $\boldsymbol{\delta_{in}^*}$ and $\boldsymbol{\delta_{out}^*}$ are ground-truth. Note that L_{off} is averaged separately for inside and outside objects due to the different formulations.

3.4. Visual Properties Regression

We elaborate on the regression of visual properties including the 2D bounding boxes, dimensions, orientations, and keypoints of objects in this section.

2D Detection. Since we do not represent objects as their 2D centers, we follow FCOS [44] to regress the distances from the representative point $\boldsymbol{x_r} = (u_r, v_r)$, which refers to $\boldsymbol{x_b}$ for inside and $\boldsymbol{x_I}$ for outside objects, to four sides of the 2D bounding box. If we denote the left-top corner as (u_1, v_1) and the right-bottom corner as (u_2, v_2), the regression target is then:

$$\begin{aligned} l^* &= u_r - u_1, \quad r^* = u_2 - u_r, \\ t^* &= v_r - v_1, \quad b^* = v_2 - v_r. \end{aligned} \tag{5}$$

GIoU loss [38] is adopted for 2D detection since it is robust to scale changes.

Dimension Estimation. Considering the small variance of object sizes within each category, we regress the relative changes with respect to the statistical average instead of absolute values. For each class c, the average dimension of the training set is denoted as $(\overline{h}_c, \overline{w}_c, \overline{l}_c)$. Assume the regressed log-scale offsets of dimensions are $(\delta_h, \delta_w, \delta_l)$ and

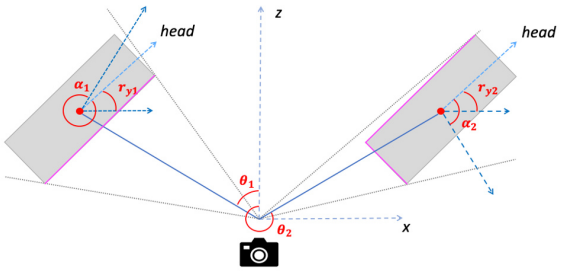


Figure 5: r_y, α, and θ are the global orientation, local orientation, and the viewing angle.

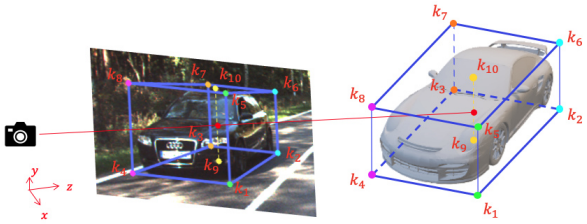


Figure 6: Keypoints include the projections of eight vertexes, top center and bottom center of the 3D bounding box.

the ground-truth dimensions are (h^*, w^*, l^*), the L1 loss for dimension regression is defined as:

$$L_{dim} = \sum_{k \in \{h,w,l\}} \left| \overline{k}_c e^{\delta_k} - k^* \right| \tag{6}$$

Orientation Estimation. The orientation can be represented as either the global orientation in the camera coordinate system or the local orientation relative to the viewing direction. For an object located at (x, y, z), its global orientation r_y and local orientation α satisfy (7):

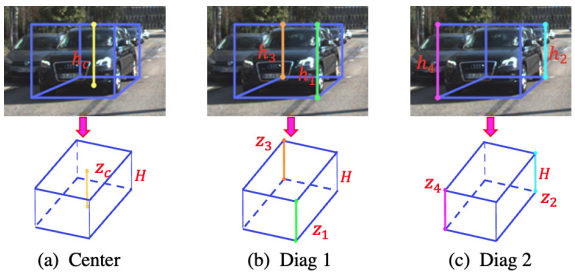
$$r_y = \alpha + \arctan(x/z) \tag{7}$$

As shown in Figure 5, objects with the same global orientations but different viewing angles will have different local orientations and visual appearances. Thus, we choose to estimate the local orientation with *MultiBin* loss [6], which divides the orientation range into N_o overlapping bins so that the network can determine which bin an object lies inside and estimate the residual rotation w.r.t the bin center.

Keypoint Estimation. As shown in Figure 6, we define $N_k = 10$ keypoints for each object which include the projections of eight vertexes $\{\boldsymbol{k}_i, i = 1...8\}$, bottom center $\boldsymbol{k}_9$ and top center $\boldsymbol{k}_{10}$ of the 3D bounding box. We regress the local offsets $\{\boldsymbol{\delta}_{ki} = \boldsymbol{k}_i - \boldsymbol{x_r}, i = 1...N_k\}$ from $\boldsymbol{x_r}$ to N_k keypoints with L1 loss:

$$L_{key} = \frac{\sum_{i=1}^{N_k} I_{in}(\boldsymbol{k}_i) \left| \boldsymbol{\delta}_{ki} - \boldsymbol{\delta}^*_{ki} \right|}{\sum_{i=1}^{N_k} I_{in}(\boldsymbol{k}_i)} \tag{8}$$

where $\boldsymbol{\delta}^*_{ki}$ is the ground-truth and $I_{in}(\boldsymbol{k}_i)$ indicates whether the keypoint $\boldsymbol{k}_i$ is inside the image.



(a) Center (b) Diag 1 (c) Diag 2

Figure 7: The depth of a supporting line of the 3D bounding box can be computed with the object height and the line's pixel height. We split ten keypoints into three groups, each of which can produce the center depth independently.

3.5. Adaptive Depth Ensemble.

We formulate the estimation of object depth as an adaptive ensemble of $M + 1$ independent estimators, including direct regression and M geometric solutions from keypoints. We first introduce these depth estimators and then present how we combine them with uncertainties.

Direct Regression. To directly regress the object depth, we follow [10, 52] to transform the unlimited network output z_o into the absolute depth z_r with the inverse sigmoid transformation:

$$z_r = \frac{1}{\sigma(z_o)} - 1, \quad \sigma(x) = \frac{1}{1 + e^{-x}} \tag{9}$$

To jointly model the uncertainty of the regressed depth, we follow [11, 18, 19] to utilize a modified L1 loss for depth regression:

$$L_{dep} = \frac{|z_r - z^*|}{\sigma_{dep}} + \log(\sigma_{dep}) \tag{10}$$

where σ_{dep} is the uncertainty of the regressed depth. When the model lacks confidence in its prediction, it will output a larger σ_{dep} so that L_{dep} can be reduced. The term $\log(\sigma_{dep})$ can avoid trivial solutions and encourage the model to be optimistic about accurate predictions.

Depth From Keypoints. With known camera matrices, we can utilize the relative proportion between pixel height and estimated object height to compute the object depth, which is similar to [5]. From our baseline model, the relative errors of predicted dimensions are 5.2%, 6.1%, and 11.8% for height, width, and length. Therefore, solving depth from height is not only independent of orientation estimation but suffers less from the error of dimension estimation. As shown in Figure 7, the estimated ten keypoints constitute five vertical supporting lines of the 3D bounding box. The depth z_l of each vertical line can be computed from its pixel height h_l and the object height H as (11):

$$z_l = \frac{f \times H}{h_l} \tag{11}$$

where f is the camera's focal length. The depth of the center vertical line z_c is exactly the object depth while averaging the depths of two diagonal vertical edges, namely z_1 and z_3 or z_2 and z_4, can also get the object depth. Therefore, the estimated ten keypoints are divided into three groups and generate respectively independent depths denoted as the center depth z_c, the diag_1 depth z_{d_1} and the diag_2 depth z_{d_2}.

To further supervise the computed depths from keypoints and also model their uncertainties, we adopt the L1 loss with uncertainty as follows:

$$L_{kd} = \sum_{k \in \{c, d_1, d_2\}} \left[\frac{|z_k - z^*|}{\sigma_k} + I_{in}(z_k) \log(\sigma_k) \right] \quad (12)$$

where z^* is ground-truth and $I_{in}(z_k)$ indicates whether all keypoints used for computing z_k are inside the image. Removing the log uncertainty for "invalid" depths computed from invisible keypoints allows the model to be fully pessimistic so that these depths are down-weighted in the ensemble. Note that we also restrict the gradients from these invalid depths to only update the uncertainty.

Uncertainty Guided Ensemble. Now that we have $M+1$ predicted depths $\{z_i, i = 1...M+1\}$ and their uncertainties $\{\sigma_i, i = 1...M+1\}$ from $M+1$ independent estimators, we propose to compute the uncertainty-weighted average, namely soft ensemble, as expressed in (13):

$$z_{soft} = \left(\sum_{i=1}^{M+1} \frac{z_i}{\sigma_i} \right) / \left(\sum_{i=1}^{M+1} \frac{1}{\sigma_i} \right) \quad (13)$$

The soft ensemble can assign more weights to those more confident estimators while being robust to potential inaccurate uncertainties. We also consider the hard ensemble where the estimator with minimal uncertainty is chosen as the final depth estimation. The performances of two ensemble ways are compared in Section 4.5.

Integral Corner Loss. As discussed in [36, 42], the separate optimization of multiple subtasks cannot ensure the optimal cooperation among different components. Therefore, we also supervise the coordinates of eight corners $\{\boldsymbol{v}_i = (x_i, y_i, z_i), i = 1, ..., 8\}$ from the predicted 3D bounding box, which is formed by the estimated dimension, orientation, offset, and soft depth z_{soft}, with L1 loss:

$$L_{corner} = \sum_{i=1}^{8} |\boldsymbol{v}_i - \boldsymbol{v}_i^*| \quad (14)$$

4. Experiments

The proposed method is evaluated on KITTI 3D Object Detection benchmark [14], which includes 7481 images for training and 7518 images for testing. We follow [9] to split the training images into *train* (3712) and *val* (3769) sets. Detection results are evaluated on three levels of difficulty: easy, moderate, and hard, which are defined by the bounding box height, occlusion, and truncation. All our reported results are produced by models that jointly detect multi-classes, including Car, Pedestrian, and Cyclist. Note that results for KITTI Bird's Eye View benchmark will be provided in the supplementary material for reference.

4.1. Implementation Details

We adopt the same modified DLA-34 [51] as our backbone network following [10, 28, 52]. All input images are padded to the same size of 384×1280. Every prediction head attached to the backbone consists of one $3 \times 3 \times 256$ conv layer, BatchNorm [16], ReLU, and another $1 \times 1 \times c_o$ conv layer, where c_o is the output size. The edge fusion module has similar settings except using 1D conv layer and empirically removing the ReLU activation. For *MultiBin* loss [34], we use four bins centered at $[0, \frac{\pi}{2}, \pi, -\frac{\pi}{2}]$. The model is trained using AdamW [29] optimizer with the initial learning rate as 3e-4 and weight decay as 1e-5. We train the model for 34k iterations with a batchsize of 7 on a single RTX 2080Ti GPU and the learning rate is divided by 10 at 22k and 30k iterations. The random horizontal flip is adopted as the only data augmentation.

4.2. Evaluation Metrics

The detection is evaluated by the average precision of 3D bounding boxes AP_{3D}. For the *val* set, we report both $\text{AP}_{3D}|_{R_{11}}$ and $\text{AP}_{3D}|_{R_{40}}$ for a comprehensive comparison with previous studies. For the *test* set, the $\text{AP}_{3D}|_{R_{40}}$ results from the test server are reported. The IoU thresholds for AP_{3D} are 0.7 for Car and 0.5 for Pedestrian and Cyclist.

4.3. Quantitative Results

In Table 1, we conduct a comprehensive comparison between our proposed method and existing arts on the *val* and *test* sets of KITTI benchmark for Car. Without bells and whistles, our method outperforms all prior methods that only take monocular images as input. For $\text{AP}_{3D}|_{R_{40}}$ on the *val* set, our method is 45%, 42% and 42% higher than the second-best method MonoPair [10] on three levels of difficulty. For the *test* set, our proposed method surpasses all existing methods, including those with extra information. The significant improvement on hard samples demonstrates that our method can effectively detect those heavily truncated objects, which are crucial for practical applications. We further show the results of Pedestrian and Cyclist on the *test* set in Table 2. Our method outperforms M3D-RPN [3] and Movi3D [43] while achieving comparable performance with MonoPair [10]. Finally, our method is also much faster than most existing methods, allowing for real-time inference. To sum up, our proposed framework achieves a state-of-the-art trade-off between performance and latency.

Methods	Extra	Time (ms)	Val, $AP_{3D}\|_{R_{11}}$			Val, $AP_{3D}\|_{R_{40}}$			Test, $AP_{3D}\|_{R_{40}}$		
			Easy	Mod	Hard	Easy	Mod	Hard	Easy	Mod	Hard
MonoPSR[20]	depth, LiDAR	120	12.75	11.48	8.59	-	-	-	10.76	7.25	5.85
UR3D[48]	depth	120	28.05	18.76	16.55	23.24	13.35	10.15	15.58	8.61	6.00
AM3D[31]	depth	-	32.23	21.09	17.26	28.31	15.76	12.24	16.50	10.74	9.52
PatchNet[30]	depth	-	35.10	22.00	19.60	31.60	16.80	13.80	15.68	11.12	10.17
DA-3Ddet[50]	depth, LiDAR	-	33.40	24.00	19.90	-	-	-	16.80	11.50	8.90
D4LCN[13]	depth	-	26.97	21.71	18.22	22.32	16.20	12.30	16.65	11.72	9.51
Kinem3D[4]	multi-frames	120	-	-	-	19.76	14.10	10.47	19.07	12.72	9.17
FQNet[27]	-	-	5.98	5.50	4.75	-	-	-	2.77	1.51	1.01
MonoGRNet[36]	-	60	13.88	10.19	7.62	-	-	-	9.61	5.74	4.25
MonoDIS[42]	-	100	18.05	14.98	13.42	-	-	-	10.37	7.94	6.40
M3D-RPN[3]	-	160	20.27	17.06	15.21	14.53	11.07	8.65	14.76	9.71	7.42
SMOKE[28]	-	**30**	14.76	12.85	11.50	-	-	-	14.03	9.76	7.84
MonoPair[10]	-	57	-	-	-	16.28	12.30	10.42	13.04	9.99	8.65
RTM3D[25]	-	55	20.77	16.86	16.63	-	-	-	14.41	10.34	8.77
Movi3D[43]	-	45	-	-	-	14.28	11.13	9.68	15.19	10.90	9.26
Ours	-	35	**28.17**	**21.92**	**19.07**	**23.64**	**17.51**	**14.83**	**19.94**	**13.89**	**12.07**

Table 1: Quantitative results for Car on KITTI *val/test* sets, evaluated by AP_{3D}. "Extra" lists the required extra information for each method. We divide existing methods into two groups considering whether they utilize extra information and sort them according to their performance on the moderate level of the *test* set within each group.

Methods	Test, $AP_{3D}\|_{R_{40}}$					
	Pedestrian			Cyclist		
	Easy	Mod	Hard	Easy	Mod	Hard
M3D-RPN[31]	4.92	3.48	2.94	0.94	0.65	0.47
Movi3D[43]	8.99	5.44	4.57	1.08	0.63	0.70
MonoPair[10]	**10.02**	**6.68**	**5.53**	3.79	2.12	1.83
Ours	9.43	6.31	5.26	**4.17**	**2.35**	**2.04**

Table 2: Quantitative results for Pedestrian and Cyclist on KITTI *test* set.

Representation		Decoupled Loss	Edge Fusion	Val, $AP_{3D}\|_{R_{40}}$		
Inside	Outside			E	M	H
$\boldsymbol{x_b}$	-			13.5	10.5	8.9
$\boldsymbol{x_b}$	$\boldsymbol{x_b}$			13.0	9.4	7.6
$\boldsymbol{x_c}$	-			15.3	11.0	9.6
$\boldsymbol{x_c}$	$\boldsymbol{x_I}$			13.9	10.2	8.9
$\boldsymbol{x_c}$	$\boldsymbol{x_I}$	√		14.2	11.7	9.8
$\boldsymbol{x_c}$	$\boldsymbol{x_I}$		√	14.6	11.7	9.7
$\boldsymbol{x_c}$	$\boldsymbol{x_I}$	√	√	15.9	**12.6**	**11.4**
$\boldsymbol{x_c}$	$\boldsymbol{x_{cc}}$	√	√	12.1	9.8	8.6
$\boldsymbol{x_c}$	$\boldsymbol{x_{cb}}$	√	√	**16.2**	12.0	10.2

Table 3: Ablation study on decoupled representations.

4.4. Ablation Study

4.4.1 Decoupled Representations

In Table 3, we compare various representations for inside and outside objects and validate the improvement from separate offset losses, namely the decoupled loss, and the edge fusion module. The second row which represents all objects with $\boldsymbol{x_b}$ is considered as the baseline. All models directly regress the object depth without ensemble.

We observe that: (1) Simply discarding outside objects can improve the performance compared to the baseline, demonstrating the necessity of decoupling outside objects. (2) Identifying inside objects as their projected 3D centers $\boldsymbol{x_c}$ is better than the 2D centers $\boldsymbol{x_b}$, possibly because the offsets from $\boldsymbol{x_b}$ to $\boldsymbol{x_c}$ are irregular and hard to learn. (3) The decoupled optimization of inside and outside offsets and the edge fusion module are crucial for the remarkable improvement on moderate and hard samples, where the heavily truncated objects belong. (4) Compared with $\boldsymbol{x_{cc}}$ and $\boldsymbol{x_{cb}}$ derived by clamping $\boldsymbol{x_c}$ and $\boldsymbol{x_b}$ to the image edge as shown in Figure 4(a), the proposed intersection $\boldsymbol{x_I}$ is a more effective representation for outside objects.

4.4.2 Object Depth Estimation

We compare different methods for object depth estimation in Table 4. "Direct Regression" refers to the best model in Table 3 with our decoupled representations but without estimating keypoints. "Keypoints" replaces the depth branch with keypoint prediction and solves the object depth from geometry as in Section 3.5. The regressed depth performs slightly better than the keypoints-based solutions. The import of uncertainties significantly improves both methods because it allows the model to neglect difficult outliers and focus on most moderate objects. By contrast, our adaptive depth ensemble method simultaneously performs both predictions and further combines them with the uncertainty-guided weights, outperforming all individual methods by an obvious margin.

4.5. Depth Combination

To further understand the effectiveness of the proposed depth ensemble, we compare the performance of each es-



Figure 8: **Qualitative Results.** We visualize the results of 3D object detection on KITTI *val* set, where predicted cars, pedestrians, and cyclists are represented in cyan, light pink, and red boxes. We use red ovals to emphasize those heavily truncated objects.

Depth Method	Val, $AP_{3D}\|_{R_{40}}$		
	Easy	Mod	Hard
Direct Regression	15.86	12.60	11.38
Direct Regression + σ	19.63	14.83	13.25
Keypoints	15.45	12.18	10.73
Keypoints + σ	18.42	14.76	12.49
Adaptive Ensemble	**23.64**	**17.51**	**14.83**

Table 4: Ablation study on object depth estimation.

timator and the combined depth from the ensemble model in Table 5. It can be observed that the joint learning consistently improves the performance of each depth estimator compared with results in Table 4, which can owe to enhanced feature learning. The combined depth from soft ensemble outperforms every individual estimator, especially for the moderate level of Car and all levels of Pedestrian. The hard ensemble is inferior, possibly due to its sensitivity to the mismatch between the actual depth error and the estimated uncertainty. We also provide the performance from the oracle depth which means the most accurate estimator is always chosen for each object by an oracle. It can be considered as the ideal upper bound of our depth ensemble. To match a predicted object with a ground-truth object, we require their 2D IoU to be larger than 0.5. We notice that our soft ensemble is very close to the oracle performance on Pedestrian, demonstrating the effectiveness of our proposed combination method. On the other hand, the oracle performance for Car reveals the enormous potential of combining different depth estimators, which can be left for future work.

4.6. Qualitative Results

From the qualitative results shown in Figure 8, our proposed framework can produce superior performance for ordinary objects in various street scenes. As highlighted by the red ovals, we can also successfully detect some extremely truncated objects which are crucial for the safety

Estimator	Val, $AP\|_{R_{40}}$					
	Car, IoU>0.7			Pedestrian, IoU>0.5		
	Easy	Mod	Hard	Easy	Mod	Hard
Regression	23.41	16.83	14.59	7.39	5.81	4.54
Key: Center	23.29	16.84	14.72	7.40	5.74	4.54
Key: $Diag_1$	23.13	16.70	14.50	7.30	5.64	4.52
Key: $Diag_2$	23.35	16.81	14.63	7.38	5.76	4.56
Hard Ensemble	22.58	16.80	14.58	7.51	6.38	4.64
Soft Ensemble	**23.64**	**17.51**	**14.83**	**8.16**	**6.45**	**5.16**
Oracle	26.28	19.98	17.07	8.54	6.72	5.55

Table 5: Quantitative analysis for the adaptive ensemble of depth estimators.

of autonomous driving, demonstrating the effectiveness of decoupling truncated objects.

5. Conclusion

In this paper, we have proposed a novel framework for monocular 3D object detection which flexibly handles different objects. We observe the long-tail distribution of truncated objects and explicitly decouple them with the proposed edge heatmap and edge fusion module. We also formulate the object depth estimation as an uncertainty-guided ensemble of multiple approaches, leading to more robust and accurate predictions. Experiments on KITTI benchmark show that our method significantly outperforms all existing competitors. Our work sheds light on the importance of flexibly processing different objects, especially for the challenging monocular 3D object detection.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China under Grant U1713214, Grant U1813218, Grant 61822603, in part by Beijing Academy of Artificial Intelligence (BAAI), and in part by a grant from the Institute for Guo Qiang, Tsinghua University.

References

[1] Estèphe Arnaud, Arnaud Dapogny, and Kévin Bailly. Tree-gated deep mixture-of-experts for pose-robust face alignment. *T-BIOM*, pages 122–132, 2019. 2

[2] Ivan Barabanau, Alexey Artemov, Evgeny Burnaev, and Vyacheslav Murashkin. Monocular 3d object detection via geometric reasoning on keypoints. *CoRR*, abs/1905.05618, 2019. 1, 2

[3] Garrick Brazil and Xiaoming Liu. M3d-rpn: Monocular 3d region proposal network for object detection. In *ICCV*, pages 9287–9296, 2019. 1, 2, 6, 7

[4] Garrick Brazil, Gerard Pons-Moll, Xiaoming Liu, and Bernt Schiele. Kinematic 3d object detection in monocular video. In *ECCV*, pages 135–152, 2020. 7

[5] 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. In *AAAI*, pages 10478–10485, 2020. 2, 5

[6] 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 *CVPR*, pages 2040–2049, 2017. 5

[7] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *CVPR*, pages 2147–2156, 2016. 1, 2

[8] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. In *NeurIPS*, pages 424–432, 2015. 1

[9] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. In *NeurIPS*, pages 424–432, 2015. 6

[10] Yongjian Chen, Lei Tai, Kai Sun, and Mingyang Li. Monopair: Monocular 3d object detection using pairwise spatial relationships. In *CVPR*, pages 12093–12102, 2020. 1, 2, 3, 4, 5, 6, 7

[11] Jiwoong Choi, Dayoung Chun, Hyun Kim, and Hyuk-Jae Lee. Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving. In *CVPR*, pages 502–511, 2019. 2, 5

[12] Thomas G Dietterich et al. Ensemble learning. *The handbook of brain theory and neural networks*, pages 110–125, 2002. 2

[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. In *CVPR*, pages 11672–11681, 2020. 1, 7

[14] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, pages 3354–3361, 2012. 2, 6

[15] Yihui He, Chenchen Zhu, Jianren Wang, Marios Savvides, and Xiangyu Zhang. Bounding box regression with uncertainty for accurate object detection. In *CVPR*, pages 2888–2897, 2019. 2

[16] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015. 6

[17] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, pages 79–87, 1991. 2

[18] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *NeurIPS*, pages 5580–5590, 2017. 2, 5

[19] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, pages 7482–7491, 2018. 2, 5

[20] Jason Ku, Alex D Pon, and Steven L Waslander. Monocular 3d object detection leveraging accurate proposals and shape reconstruction. In *CVPR*, pages 11867–11876, 2019. 1, 7

[21] Jason Ku, Alex D. Pon, and Steven L. Waslander. Monocular 3d object detection leveraging accurate proposals and shape reconstruction. In *CVPR*, pages 11867–11876, 2019. 2

[22] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *CVPR*, pages 12697–12705, 2019. 1

[23] Stefan Lee, Senthil Purushwalkam, Michael Cogswell, David Crandall, and Dhruv Batra. Why m heads are better than one: Training a diverse ensemble of deep networks. *CoRR*, abs/1511.06314, 2015. 2

[24] Peiliang Li, Xiaozhi Chen, and Shaojie Shen. Stereo r-cnn based 3d object detection for autonomous driving. In *CVPR*, pages 7644–7652, 2019. 1

[25] Peixuan Li, Huaici Zhao, Pengfei Liu, and Feidao Cao. Rtm3d: Real-time monocular 3d detection from object keypoints for autonomous driving. In *ECCV*, pages 644–660, 2020. 1, 2, 3, 4, 7

[26] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017. 4

[27] Lijie Liu, Jiwen Lu, Chunjing Xu, Qi Tian, and Jie Zhou. Deep fitting degree scoring network for monocular 3d object detection. In *CVPR*, pages 1057–1066, 2019. 1, 2, 7

[28] Zechen Liu, Zizhang Wu, and Roland Tóth. Smoke: Single-stage monocular 3d object detection via keypoint estimation. In *CVPRW*, pages 996–997, 2020. 1, 2, 4, 6, 7

[29] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 6

[30] Xinzhu Ma, Shinan Liu, Zhiyi Xia, Hongwen Zhang, Xingyu Zeng, and Wanli Ouyang. Rethinking pseudo-lidar representation. In *ECCV*, pages 311–327, 2020. 2, 7

[31] 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 *ICCV*, pages 6851–6860, 2019. 1, 2, 7

[32] Fabian Manhardt, Wadim Kehl, and Adrien Gaidon. Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape. In *CVPR*, pages 2069–2078, 2019. 2

[33] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IROS*, pages 922–928, 2015. 1

[34] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. In *CVPR*, pages 7074–7082, 2017. 1, 2, 6

[35] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *CVPR*, pages 918–927, 2018. 1

[36] Zengyi Qin, Jinglu Wang, and Yan Lu. Monogrnet: A geometric reasoning network for 3d object localization. In *AAAI*, pages 8851–8858, 2019. 1, 2, 6, 7

[37] Zengyi Qin, Jinglu Wang, and Yan Lu. Triangulation learning network: from monocular to stereo 3d object detection. In *CVPR*, pages 7615–7623, 2019. 1

[38] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *CVPR*, pages 658–666, 2019. 4

[39] Lior Rokach. Ensemble-based classifiers. *Artificial intelligence review*, pages 1–39, 2010. 2

[40] 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. In *CVPR*, pages 10529–10538, 2020. 1

[41] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *CVPR*, pages 770–779, 2019. 1

[42] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *CVPR*, pages 1991–1999, 2019. 6, 7

[43] Andrea Simonelli, Samuel Rota Bulò, Lorenzo Porzi, Elisa Ricci, and Peter Kontschieder. Towards generalization across depth for monocular 3d object detection. In *ECCV*, pages 767–782, 2020. 6, 7

[44] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *CVPR*, pages 9627–9636, 2019. 4

[45] 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 *CVPR*, pages 8445–8453, 2019. 1, 2

[46] Xinshuo Weng and Kris Kitani. Monocular 3d object detection with pseudo-lidar point cloud. In *ICCVW*, 2019. 2

[47] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3d object detection from monocular images. In *CVPR*, pages 2345–2353, 2018. 2

[48] Tae-Kyun Kim Xuepeng Shi, Zhixiang Chen. Distance-normalized unified representation for monocular 3d object detection. In *ECCV*, pages 91–107, 2020. 7

[49] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, page 3337, 2018. 1

[50] Xiaoqing Ye, Liang Du, Yifeng Shi, Yingying Li, Xiao Tan, Jianfeng Feng, Errui Ding, and Shilei Wen. Monocular 3d object detection via feature domain adaptation. In *ECCV*, pages 17–34, 2020. 7

[51] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *CVPR*, pages 2403–2412, 2018. 6

[52] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *CoRR*, abs/1904.07850, 2019. 1, 2, 3, 4, 5, 6