Delving into Localization Errors for Monocular 3D Object Detection

Xinzhu Ma[1], Yinmin Zhang[3], Dan Xu[2], Dongzhan Zhou[1],
Shuai Yi[3], Haojie Li[4], Wanli Ouyang[1]
[1]The University of Sydney, [2]The Hong Kong University of Science and Technology,
[3]SenseTime Research, [4]Dalian University of Technology
{xinzhu.ma, d.zhou, wanli.ouyang}@sydney.edu.au,
{zhangyinmin, yishuai}@sensetime.com, danxu@cse.ust.hk, hjli@dlut.edu.cn

Abstract

Estimating 3D bounding boxes from monocular images is an essential component in autonomous driving, while accurate 3D object detection from this kind of data is very challenging. In this work, by intensive diagnosis experiments, we quantify the impact introduced by each sub-task and found the 'localization error' is the vital factor in restricting monocular 3D detection. Besides, we also investigate the underlying reasons behind localization errors, analyze the issues they might bring, and propose three strategies. First, we revisit the misalignment between the center of the 2D bounding box and the projected center of the 3D object, which is a vital factor leading to low localization accuracy. Second, we observe that accurately localizing distant objects with existing technologies is almost impossible, while those samples will mislead the learned network. To this end, we propose to remove such samples from the training set for improving the overall performance of the detector. Lastly, we also propose a novel 3D IoU oriented loss for the size estimation of the object, which is not affected by 'localization error'. We conduct extensive experiments on the KITTI dataset, where the proposed method achieves real-time detection and outperforms previous methods by a large margin. The code will be made available at: https://github.com/xinzhuma/monodle.

1. Introduction

Remarkable progress has been achieved in 3D detection, especially for LiDAR/stereo-based approaches [44, 19, 33, 9, 37], along with the advances in deep neural networks. In contrast, the accuracy of 3D detection from only monocular images [35, 2, 10, 25, 24, 12] is obviously lower than that from LiDAR or stereo. In this work, we aim to quantitatively identify the problem and propose our solutions.

To investigate and quantify the underlying factors that restrict the performance of monocular 3D object detection,

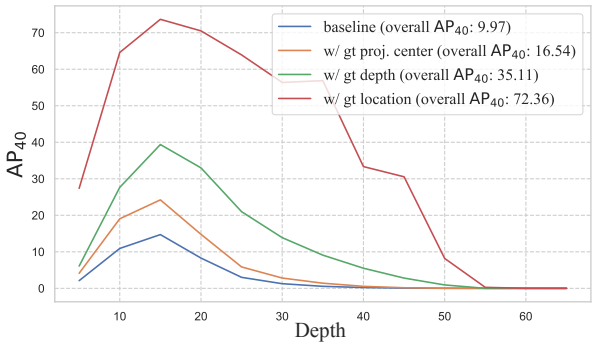

Figure 1: **Range-wise evaluation on the KITTI *validation* set.** Metric is AP_{40} of the Car category under moderate setting. The sampling interval is 10 m. For example, the corresponding value at horizontal axis 20 represents the overall performance of all samples between 15 m and 25 m.

we conduct intensive diagnostic experiments for this task, inspired by the error identifying methods [20, 43, 17, 1] commonly used in the 2D detection scope. Specifically, we build our baseline model (see Section 3.2 for details) based on CenterNet [43] and progressively replace predicted items with their ground-truth values. To better analyze the error patterns, we evaluate the results in a range-wise manner and show the summary of those experiments in Figure 1. Based on our investigation, we have the following three observations and corresponding designs.

Observation 1: The most striking feature in Figure 1 is the leap in performance when using ground-truth location, reaching a level similar to the state-of-the-art LiDAR-based methods, suggesting the localization error is the key factor in restricting monocular 3D detection. Furthermore, except for depth estimation, detecting the projected center of the 3D object also plays an important role in restoring the 3D position of the object. To this end, we revisit the misalignment between the center of the 2D bounding box and the projected center of the 3D object. Besides, we also confirm the necessity of keeping 2D detection related branches in

monocular 3D detector. In this way, 2D detection is used as the correlated auxiliary task to help learning the features shared with 3D detection, which is different from the existing work in [23] that discards 2D detection.

Observation 2: An apparent trend reflected in Figure 1 is that the detection accuracy significantly decreases with respect to the distance (the low performance of very close range objects will be discussed in supplementary materials). More importantly, all the models cannot output any true positive samples beyond a certain distance. We found that it is almost impossible to detect distant objects accurately with existing technologies due to the inevitable localization errors (see Section 4.4 for details). In this case, whether it is beneficial to add these samples into the training set becomes a question. In fact, there is a clear domain gap between 'bad' samples and 'easy-to-detect' samples and forcing the network to learn from those samples will reduce its representative ability for the others, which will thus impair the overall performance. Based on the observation above, we propose two schemes. The first scheme removes distant samples from the training set and the second scheme reduces the training loss weights of these samples.

Observation 3: We found that, except for localization error, there are also some other vital factors, such as dimension estimation, restricting monocular 3D detection (there is still 27.4% room for improvements even we use the ground-truth location). Existing methods in this scope tend to optimize each component of the 3D bounding box independently, and the studies in [35, 36] confirm the effectiveness of this strategy. However, the failure to consider the contribution of each loss item to the final metric (*i.e*. 3D IoU) may lead to sub-optimal optimization. To alleviate this problem, we propose an IoU oriented loss for 3D size estimation. The new IoU oriented loss dynamically adjust the loss weight for each side in sample level according its contribution rate to the 3D IoU.

In summary, the key contributions of this paper are as follows: First, we conduct intensive diagnostic experiments for monocular 3D detection. In addition to finding that the 'localization error' is the main problem restricting monocular 3D detection, we also quantify the overall impact of each sub-task. Second, we investigate the underlying reasons behind localization error, analyze the issues it might bring. Accordingly, we propose three novel strategies operating on annotations, training samples, and optimization losses to alleviate problems caused by localization error for boosting the detection.

Experimental results show the effectiveness of the proposed strategies. In particular, compared with existing best-performing monocular 3D object detection approaches, the proposed method achieves at least 1.6 points AP_{40} improvements on the bird's view detection and 3D object detection in the KITTI dataset.

2. Related Work

Standard monocular 3D detection. Here we briefly review the 'standard' monocular 3D detection approaches [27, 32, 29, 35, 2, 43] only use the RGB images, annotations and camera calibrations provided by KITTI dataset. [27, 29, 10] try to improve the representation ability of the models by introducing novel geometric constraints. OFTNet [32] presents an orthographic feature transform to map image-based features into an orthographic 3D space. MonoDIS [35] disentangles the loss for 2D/3D detection and jointly trains these two tasks in an end-to-end manner. M3D-RPN [2] extends the region proposal network (RPN) with 3D box parameters. These works are orthogonal to our analysis to localization error and the proposed strategies for handling it.

Monocular 3D detection using additional data. To better estimate the 3D bounding boxes, many methods are proposed for effectively using additional data [38, 26, 5, 39, 4, 12, 16, 13, 37, 25, 24, 3]. Specifically, [38, 26] use the CAD models as shape templates to get better object geometry. Deep MANTA [5], which takes 3D detection as a key-points detection task, uses more detailed annotated locations of keypoints, *e.g*. wheels, as training labels. Besides, [39, 4, 12] estimate the depth maps from off-the-shelf depth estimators [16, 13] trained from larger datasets, and use them to augment the input RGB images. In addition, [37, 25] propose to transform the estimated depth maps to pseudo-LiDAR representation, before applying existing LiDAR-based 3D detection designs, and achieve promising performance on KITTI benchmark. PatchNet [24] analyzes the underlying mechanism behind pseudo-LiDAR representation and proposes its corresponding image representation based implementation. Recently, Kinematic3D [3] propose to use 3D Kalman filter to capture the temporal cues from monocular videos. In contrast, our method does not use any extra data or annotation, and can still achieve better or competitive performance.

Misalignment between the definitions of object's center. To recover the 3D object position, there are two groups of methods. The first group [43, 10, 35] use 2D bounding box to obtain 3D position. In particular, CenterNet [43] regards the center of the 2D bounding box as the projected 3D position in the image plane and back-project it to 3D space with the help of estimated depth and camera parameters. However, generally speaking, the center of the 2D box and the center of 3D box are not the same. [10, 35] regress an offset to compensate for the difference between them. As the second group, SMOKE [23] removes the 2D detection and directly estimate 3D position using projected 3D center. This work considers the 2D related sub-tasks are redundant because 2D bounding boxes can be generated from 3D detection results. In this work, we revisit this problem and confirm that replacing the 2D center by the projected 3D

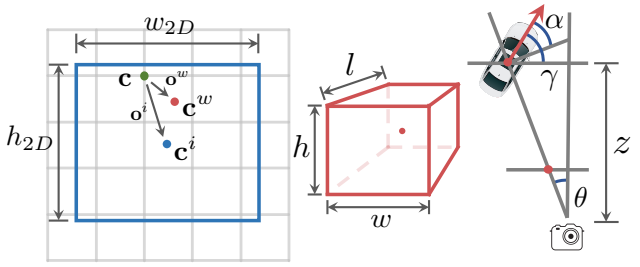

Figure 2: **Visualization of the notations** of 2D bounding box in the feature map scale (*left*), 3D bounding box in the 3D world space (*middle*), and orientation of the object from bird's view (*right*).

center can improve the localization accuracy. Besides, we also find that 2D detection is necessary, because it helps to learn shared features for 3D detection.

3. Approach

3.1. Problem Definition

Given are RGB images and the corresponding camera parameters, our goal is to classify and localize the objects of interest in 3D space. Each object is represented by its category, 2D bounding box $\mathbf{B_{2D}}$, and 3D bounding box $\mathbf{B_{3D}}$. Specifically, $\mathbf{B_{2D}}$ is represented by its center $\mathbf{c^i} = [x', y']_{2D}$ and size $[h', w']_{2D}$ in the image plane, while $\mathbf{B_{3D}}$ is defined by its center $[x, y, z]_{3D}$, size $[h, w, l]_{3D}$ and heading angle γ in the 3D world space.

3.2. Baseline Model

Architecture. We build our baseline model based on the anchor-free one-stage detector CenterNet [43]. Specifically, we use standard DLA-34 [41] as our backbone for a better speed-accuracy trade-off. On top of this, seven lightweight heads (implemented by one 3×3 conv layer and one 1×1 conv layer) are used for 2D detection and 3D detection. More design choices and implementation details can be found in the supplementary material.

2D detection. For 2D detection task, following [30, 43], the proposed model outputs a heatmap to indicate the classification score and the coarse center $\mathbf{c} = (u, v)$ of the object. In existing methods [43, 10, 35], $\mathbf{c}$ is supervised by the ground-truth 2D bounding box center. Another branch predict the offset $\mathbf{o^i} = (\Delta u^i, \Delta v^i)$ between the coarse center and the real center of 2D bounding box, and we can get the final 2D box center location $\mathbf{c^i} = \mathbf{c} + \mathbf{o^i}$. Finally, we use another branch to estimate the size $[w', h']_{2D}$ of 2D bounding box.

3D detection. As for 3D detection, a branch is used for predicting the offset $\mathbf{o^w} = (\Delta u^w, \Delta v^w)$ between the coarse center $\mathbf{c}$ and the center of projected 3D bounding box $\mathbf{c^w} = [x^w \ y^w]^T = \mathbf{c} + \mathbf{o^w}$. With the known camera intrinsic matrix $\mathbf{K} \in \mathbb{R}^{3\times3}$, we can recover the center of object in 3D world space by:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}_{3D} = \mathbf{K}^{-1} \begin{bmatrix} \mathbf{c^w} \cdot z \\ z \end{bmatrix} = \mathbf{K}^{-1} \cdot \begin{bmatrix} x^w \cdot z \\ y^w \cdot z \\ z \end{bmatrix}_{2D}, \quad (1)$$

where z is the output of depth branch. Finally, the last two branches are used to predict the 3D size $[h, w, l]_{3D}$ and orientation γ, respectively.

Losses. There are seven loss terms in total, one for foreground/background sample classification, two (center and size) for 2D detection, and four (center, depth, size, and heading angle) for 3D detection. We adopt the modified Focal Loss used in [20, 43] for classification sub-task. We use L1 Loss without any anchor for center and size regression in 2D detection task. For the 3D detection task, uncertainty modeling [18, 4] (see Section **??** in the supplementary for details) is used for depth estimation; L1 loss is used for 3D center refinement; and multi-bin loss [28] (we consider 12 non-overlap equal bins) is used for heading angle estimation. Lastly, for 3D size estimation, we use L1 loss in baseline (without anchor), and the proposed IoU loss in our model. The weights for all loss items are set to 1.

baseline	11.12	ground truth	99.97
w/ gt proj. center	23.90	w/o gt proj. center	46.33
w/ gt depth	38.01	w/o gt depth	25.25
w/ gt 3D location	78.84	w/o gt 3D location	12.13
w/ gt 3D size	11.96	w/o gt 3D size	80.50
w/ gt orientation	11.88	w/o gt orientation	70.89

Table 1: **Error analysis.** *Left:* We replace the outputs of 3D detection related branches with the ground truth values. *Right:* We replace the values of ground truth with the predicted results. Metric is AP_{40} for 3D detection under moderate setting on the KITTI val set. 'proj. center' denotes the projected 3D center $\mathbf{c^w}$ on the image plane.

3.3. Error Analysis

In this section, we explore what restricts the performance of monocular 3D detection. Inspired by CenterNet [43] and CornerNet [20] in the 2D detection field, we conduct an error analysis for different prediction items on KITTI *validation* set via replacing each predictions with ground truth value and evaluating the performance. Specifically, we replace each output head with its ground truth according to the practice of [20, 43]. As shown in Table 1, if we replace projected 3D center $\mathbf{c^w}$ predicted from baseline model with its ground-truth, the accuracy is improved from 11.12% to 18.97%. On the other hand, depth can improve the accuracy to 38.01%. If we consider both depth and projected center, *i.e.* replacing the predicted 3D locations $[x, y, z]_{3D}$ with ground-truth results, then the most obvious improvement is

Δu	Δv	5m	10m	20m	40m	60m
2	2	0.02	0.04	0.08	0.16	0.24
4	2	0.03	0.06	0.13	0.25	0.38
6	2	0.04	0.09	0.18	0.36	0.54
6	4	0.05	0.10	0.20	0.41	0.61
8	2	0.06	0.12	0.23	0.47	0.70
8	6	0.07	0.14	0.28	0.57	0.85

Table 2: **Localization error** (in meter) caused by center shifting in image plane (in pixel).

observed. Therefore, the low accuracy of monocular 3D detection is mainly caused by localization error. On the other hand, according to Equation 1, depth estimation and center localization jointly determine the position of the object in 3D world space. Compared with the ill-posed depth estimation from a monocular image, improving the accuracy of center detection is a more feasible way.

Table 2 shows localization errors introduced by inaccurate center detection. Furthermore, the mean shape of cars in KITTI dataset is $[1.53m, 1.63m, 3.53m]$ for $[h, w, l]_{3D}$. Suppose that all other quantities are correct and the localization error is aligned with the length l (resulting in the maximum tolerance), the IoU can be computed by:

$$IoU = \frac{3.53 - \Delta_{loc}}{3.53 + \Delta_{loc}}, \tag{2}$$

where Δ_{loc} represents the localization error. According to the official setting, the IoU threshold should be set to 0.7, thus the theoretically acceptable maximum error is $0.62m$. However, an error of only 4-8 pixels in the image (1-2 pixel in $4\times$ down sampling feature map) will cause the object at 60 meters cannot be detected correctly. Coupled with the errors accumulated by other tasks such as depth estimation (Figure 3 shows the errors of depth estimation), it becomes an almost impossible task to accurately estimate the 3D bounding box of distant objects from a single monocular image, unless the depth estimation is accurate enough (not achieved to date).

To better show the importance of center localization, we show the localization error in 3D space caused by shifting the center in image plane in Table 2.

3.4. Revisiting Center Detection

Our design for center detection. For estimating the coarse center $\mathbf{c}$, our design is simple. In particular, we 1) use the projected 3D center $\mathbf{c^w}$ as the ground-truth for the branch estimating coarse center $\mathbf{c}$ and 2) force our model to learn features from 2D detection simultaneously. This simple design is from our analysis below.

Analysis 1. As shown in Figure 4, there is a misalignment between the 2D bounding box center $\mathbf{c^i}$ and the projected

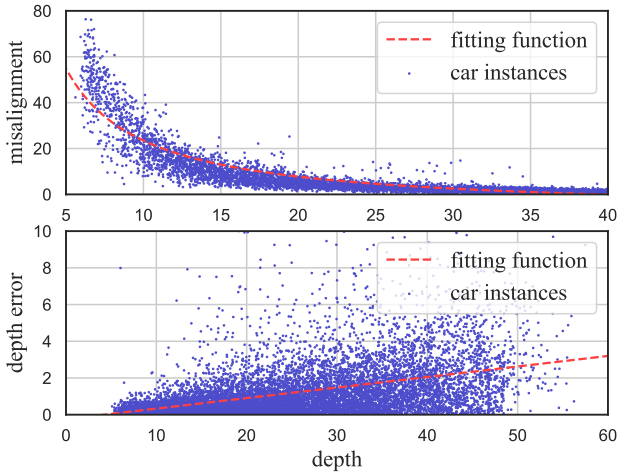

Figure 3: **Statistics.** *Top*: the misalignment (in pixel, collected on the KITTI *trainval* set under moderate setting) between the center of 2D bounding box and the projected 3D center in the image plane. *Bottom*: the depth errors (in meter, trained on the KITTI *training* set, tested on the *validation* set) These two statistics are presented as the function of the depth (x-axis).

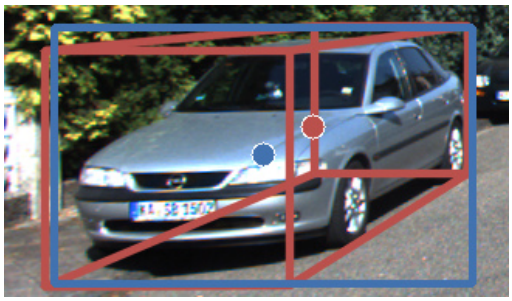

Figure 4: **Visualization of the misalignment** between the center of the 2D bounding box (blue) and the projected 3D center (red) in image plane.

center $\mathbf{c^w}$ of the 3D bounding box. According to the formulation in Equation 1, the projected 3D center $\mathbf{c^w}$ should be the key for recovering the 3D object center $[x, y, z]_{3D}$. The key problem here is what should be the supervision for the coarse center $\mathbf{c}$. Some works [10, 35] choose to use 2D box center $\mathbf{c^i}$ as its label, which is not related to the 3D object center, making the estimation of the coarse center not aware of the 3D geometry of the object. Here we choose to adopt the projected 3D center $\mathbf{c^w}$ as the ground-truth for the coarse center $\mathbf{c}$. This helps the branch for estimating the coarse center aware of 3D geometry and more related to the task of estimating 3D object center, which is the key of localization problem (see Section **??** in supplementary materials for visualizations).

Analysis 2. Note that SMOKE [23] also use the projected

3D center $\mathbf{c^w}$ as the label of the coarse center $\mathbf{c}$. However, they discard 2D detection related branches while we preserve them. In our design, the coarse center $\mathbf{c}$ supervised by the projected 3D center $\mathbf{c^w}$ is also used for estimating the 2D bounding box center $\mathbf{c^i}$. With our design, we force a 2D detection branch to estimate an offset $\mathbf{o^i} = \mathbf{c^i} - \mathbf{c}$ between the real 2D center and the coarse 2D center. This makes our model *aware of the geometric information of the object*. Besides, another branch is used to estimate the size of the 2D bounding box so that *the shared features can learn some cues that benefit to depth estimation due to the perspective projection.* In this way, the 2D detection serves as an auxiliary task that helps to learn better 3D aware features.

3.5. Training Samples

Different from [34, 22] which force network focus on the 'hard' samples, we argue that ignoring some extremely 'hard' cases can improve the overall performance for the monocular 3D detection task. Both the results shown in Figure 1 and the analysis conducted in Section 4.4 illustrate there is a strong relationship between the distance of the object and the difficulty of detecting it. According to this, two schemes are proposed on how to generate the object-level training weight w_i for sample i.

Scheme 1, hard coding. This scheme discard all samples over a certain distance:

$$w_i = \begin{cases} 1 & \text{if } d_i \leq s \\ 0 & \text{if } d_i > s \end{cases} \tag{3}$$

where d_i denotes the depth of sample i, and s is the threshold of depth which is set to 60 meters in our implementation. In this way, the samples with depth larger than s will not be used in the training phase.

Scheme 2, soft coding. The other one is soft encoding, and we generate it using a reverse sigmoid-like function:

$$w_i = \frac{1}{1 + e^{(d_i - c)/T}}, \tag{4}$$

where c and T are the hyper-parameters to adjust the center of symmetry and bending degree, respectively. When $c = s$ and $T \to 0$, it is equivalent to the hard encoding scheme. When $T \to \infty$, it is equivalent to using the same weight for all samples. By default, c and T are set to 60 and 1, and the empirical experiments in Section 4 find that scheme 1 and scheme 2 are both effective and have similar results.

3.6. IoU Oriented Optimization

Recently, some LiDAR based 3D detectors [40, 42] applied the IoU oriented optimization [31]. However, determining the 3D center of object is an very challenging task for monocular 3D detection, and the localization error often reaches several meters (see Section 4.4). In this case, localization related sub-tasks (such as depth estimation) will overwhelm others (such as 3D size estimation), if we apply IoU based loss function directly. Moreover, depth estimation from monocular image itself an ill-posed problem, and this kind of contradiction will make the training process collapse. Disentangling each loss item and optimize them independently is a another choice [35], but this ignores the correlation of each component to the final result. To alleviate this problem, we propose a IoU oriented optimization for 3D size estimation. Specifically, suppose all prediction items except the 3D size $\mathbf{s} = [h, w, l]_{3D}$ are completely correct, then we can get (details for deriving can be found in supplementary materials):

$$\frac{\partial IoU}{\partial h} : \frac{\partial IoU}{\partial w} : \frac{\partial IoU}{\partial l} \approx \frac{1}{h} : \frac{1}{w} : \frac{1}{l}. \tag{5}$$

Accordingly, we can adjust the weight of each side by its partial derivative w.r.t. IoU (in magnitude), and the loss function of the 3D size estimation can be modified to:

$$\mathcal{L}_{size} = ||\frac{(\mathbf{s} - \mathbf{s}^*)}{\mathbf{s}}||_1, \tag{6}$$

where $||\cdot||_1$ represent the L_1 norm. Note that, compared with the standard 3D size loss $\mathcal{L}'_{size} = ||\mathbf{s} - \mathbf{s}^*||_1$ used in the baseline model, our new loss's magnitude is changed. To compensate it, we compute $\mathcal{L}'_{size}$ once more, and dynamically generate the compensate weight $w_s = |\mathcal{L}'_{size}/\mathcal{L}_{size}|$, so that the mean value of the final loss function $w_s \cdot \mathcal{L}_{size}$ is equal to the standard one. By this way, the proposed loss can be regard as a re-distribution of the standard L1 loss.

3.7. Implementation

Training. We train our model on two GTX 1080Ti GPUs with a batch size of 16 in an end-to-end manner for 140 epochs. We use Adam optimizer with initial learning rate $1.25e^{-3}$, and decay it by ten times at 90 and 120 epochs. The weight decay is set to $1e^{-5}$ and the warmup strategy is also used for the first 5 epochs. To avoid over-fitting, we adopt the random cropping/scaling (for 2D detection only) and random horizontal flipping. Under this setting, it takes around 9 hours for whole training process.

Inference. During the inference phase, we obtain the prediction results from the parallel decoders. To decoding the results, similar to [43], we conduct the efficient non-maxima suppression (NMS) on center detection results using a 3×3 max pooling kernel. Then, we recover 2D/3D bounding boxes according to encoding strategy introduced in Section 3.2 and use the score of center detection as the confidence of predicted results. Finally, we discard predictions with confidence less than 0.2.

Method	Extra data	3D			BEV			AOS			Runtime
		Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	
Decoupled-3D [4]	Yes	11.08	7.02	5.63	23.16	14.82	11.25	87.34	67.23	53.84	-
AM3D [25]	Yes	16.50	10.74	9.52	25.03	17.32	14.91	-	-	-	∼400 ms
PatchNet [24]	Yes	15.68	11.12	10.17	22.97	16.86	14.97	-	-	-	∼400 ms
D4LCN [12]	Yes	16.65	11.72	9.51	22.51	16.02	12.55	90.01	82.08	63.98	-
Kinematic3D [3]	Yes	19.07	12.72	9.17	26.69	17.52	13.10	58.33	45.50	34.81	120ms
GS3D [21]	No	4.47	2.90	2.47	8.41	6.08	4.94	85.79	75.63	61.85	∼2000 ms
MonoGRNet [29]	No	9.61	5.74	4.25	18.19	11.17	8.73	-	-	-	60 ms
MonoDIS [35]	No	10.37	7.94	6.40	17.23	13.19	11.12	-	-	-	-
M3D-RPN [2]	No	14.76	9.71	7.42	21.02	13.67	10.23	88.38	82.81	67.08	161 ms
SMOKE [23]	No	14.03	9.76	7.84	20.83	14.49	12.75	92.94	87.02	77.12	**30 ms**
MonoPair [10]	No	13.04	9.99	8.65	19.28	14.83	12.89	91.65	86.11	76.45	57 ms
Ours	No	**17.23**	**12.26**	**10.29**	**24.79**	**18.89**	**16.00**	**93.46**	**90.23**	**80.11**	40 ms
Improvement	-	+2.47	+2.27	+1.64	+3.77	+4.06	+3.11	+0.52	+3.21	+2.99	-

Table 3: **Performance of the Car category on the KITTI *test* set.** Methods are ranked by moderate setting (same as KITTI leaderboard). We highlight the best results in **bold** and the second place in underlined.

Method	3D@IOU=0.7			BEV@IOU=0.7			3D@IOU=0.5			BEV@IOU=0.5		
	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
CenterNet [43]	0.60	0.66	0.77	3.46	3.31	3.21	20.00	17.50	15.57	34.36	27.91	24.65
MonoGRNet [29]	11.90	7.56	5.76	19.72	12.81	10.15	47.59	32.28	25.50	48.53	35.94	28.59
MonoDIS [35]	11.06	7.60	6.37	18.45	12.58	10.66	-	-		-	-	
M3D-RPN [2]	14.53	11.07	8.65	20.85	15.62	11.88	48.53	35.94	28.59	53.35	39.60	31.76
MonoPair [10]	16.28	12.30	10.42	24.12	18.17	15.76	55.38	42.39	**37.99**	**61.06**	**47.63**	**41.92**
Ours	**17.45**	**13.66**	**11.68**	**24.97**	**19.33**	**17.01**	**55.41**	**43.42**	37.81	60.73	46.87	41.89
Improvement	+1.17	+1.36	+1.26	+0.85	+1.16	+1.25	+0.03	+1.03	-0.18	-0.33	-0.80	-0.03

Table 4: **Performance of the Car category on the KITTI *validation* set.** Methods are ranked by moderate setting (same as KITTI leaderboard). We highlight the best results in **bold** and the second place in underlined.

4. Experimental Results

4.1. Setup

Dataset. We evaluate our method on the challenging KITTI dataset [14, 15], which provides 7,481 images for training and 7,518 images for testing. Since the ground truth for the test set is not available and the access to the test server is limited, we follow the protocol of prior works [6, 7, 8] to divide the training data into a training set (3,712 images) and a validation set (3,769 images). We conduct ablation studies based on this split and also report final results which trained on all 7,481 images and tested by KITTI official server.

Metrics. The KITTI dataset provides many widely used benchmarks for autonomous driving scenarios, including 3D detection, bird's eye view (BEV) detection, and average orientation similarity (AOS). We report the Average Precision with 40 recall positions (AP_{40}) [35] under three difficultly settings (easy, moderate, and hard) for those tasks. We mainly focus on the **Car** category, and also report the performances of the **Pedestrian** and **Cyclist** categories for reference. The default IoU threshold are 0.7, 0.5, 0.5 for these categories.

4.2. Main Results

Results on the KITTI *test* set. As shown in Table 3, we report our results of the **Car** category on KITTI *test* set. Overall, our method achieves superior results over previous methods across all settings under fair conditions. For instance, the proposed method obtains **2.47/2.27/1.64** improvements under easy/moderate/hard setting for 3D detection task. Besides, our method achieves **18.89/90.23** in BEV detection/AOS task under moderate setting, improving previous best results by **4.06/4.12** AP_{40}. Compared with the methods with extra data, the proposed method still get comparable performances, which further proves the effectiveness of our model.

Results on the KITTI *validation* set. We also present our model's performance on the KITTI *validation* set in Table 4. Note that some methods directly use the pre-trained model provided by DORN [13] as their depth estimator. However, the DORN's training set overlaps with the validation set of KITTI 3D, so we are not comparing these methods here. We can find that the proposed model performs better than all previous methods in 3D detection task. For BEV detec-

Method	Cat.	Easy	Mod.	Hard
M3D-RPN [2]	Ped.	5.65 / 4.92	4.05 / 3.48	3.29 / 2.94
MonoPair [10]	Ped.	10.99 / 10.02	7.04 / 6.68	6.29 / 5.53
Ours	Ped.	10.73 / 9.64	6.96 / 6.55	6.20 / 5.44
M3D-RPN [2]	Cyc.	1.25 / 0.94	0.81 / 0.65	0.78 / 0.47
MonoPair [10]	Cyc.	4.76 / 3.79	2.87 / 2.12	2.42 / 1.83
Ours	Cyc.	5.34 / 4.59	3.28 / 2.66	2.83 / 2.45

Table 5: **Benchmark for Pedestrian/Cyclist detection on the KITTI *test* set.** Metric is AP_{40} for BEV/3D detection task at 0.5 IoU threshold.

	Easy	Mod.	Hard
baseline	20.29 / 14.51	16.15 / 11.12	14.07 / 9.97
+ p.	23.10 / 15.78	18.15 / 12.65	16.11 / 10.62
+ p.+I.	23.89 / 16.12	18.34 / 12.97	16.69 / 10.99
+ p.+I.+s.	24.97 / 17.45	19.33 / 13.66	17.01 / 11.68

Table 6: **Results on accumulating the proposed approaches on the KITTI *validation* set.** Metric is AP_{40} of the Car category for BEV/3D detection. 'p.' denotes using projected 3D center for supervising the coarse center. 'I.' denotes using our IoU loss design. 's.' denotes the design for discarding distant samples.

tion task, our method outperforms all methods except for MonoPair. Compared with MonoPair, our method is better at detecting objects under strict conditions (0.7 IoU threshold), while MonoPair is slightly better at catching samples under loose conditions (0.5 IoU threshold). Also note that our method shows better performance consistency between the validation set and test set. This indicates that our method has better generalization ability, which is of great significance in autonomous/assisted driving.

Latency analysis. We test the proposed model on a single GTX 1080Ti GPU with a batchsize of 1 for runtime analysis. As shown in Table 3, the proposed method can run at 25 FPS, meeting the requirement of real-time detection. Specifically, our method runs $4\times$ faster than the two-stage detector M3D-RPN. Compared with MonoPair, which shares a similar framework as ours, our method can still save 16 ms for one image in the inference phase, mainly because: 1) we use standard DLA-34 as our backbone, instead of modified DL4-34 with DCN [11, 45]. 2) we apply fewer prediction heads in our model. 3) we don't need any post-processing. SMOKE [23] can run faster than our method. However, it only conducts 3D detection while the proposed method can perform 2D detection and 3D detection jointly.

Besides, although the detectors with pretrained depth estimator usually have promising performance, the additional depth estimator introduce lots of computational overheads (*e.g.* the most commonly used DORN [13] takes about 400 ms to process a standard KITTI image. See KITTI Depth Benchmark for more details).

	PC	RF	MT	Easy	Mod.	Hard
a	-	-	✓	98.08 / 1.32	92.31 / 1.04	84.75 / 1.16
b	-	✓	✓	98.08 / 13.98	92.31 / 10.81	84.75 / 9.59
c	✓	✓	-	94.55 / 12.31	88.79 / 10.30	79.29 / 8.82
d	✓	✓	✓	98.42 / 16.08	92.74 / 13.04	83.04 / 11.16

Table 7: **Analysis for center definition and multitask learning.** Metrics are AP_{40} of the Car category for 2D/3D detection tasks. 'PC', 'RF', and 'MT' represent 'projected 3D center', 'refinement', and 'multi-task learning'.

4.3. Pedestrian/Cyclist Detection

Here we present the Pedestrian/Cyclist detection results on the KITTI *test* set in Table 5. Compared with cars, pedestrians/cyclists are more difficult to detect, and only [2, 10] provide the performances of those categories on KITTI test set. Specifically, the proposed method performs better than [2] and gets comparable results with [10]. But it is important to note that, since the number of training samples for those two categories is quite small, the performance may fluctuate to some extent.

4.4. Analysis

Accumulation of the proposed designs. Table 6 shows experimental results evaluating how the proposed designs contribute to the overall performance for this task. Our design in Section 3.4, which uses projected 3D center for supervising center detection and influencing 2D detection ('+p.' in Table 6), improves 3D detection accuracy by 1.5. The IoU loss design in Section 3.6 further improves the accuracy by 0.3. And the design for discarding distant samples in Section 3.5 leads to 0.7 improvement.

Supervision for coarse center detection and multi-task learning. We show the performance changes caused by center definition and multi-task learning in Table 7. Specifically, from setting **a** (used in [43]) and setting **b** (used in [10, 35]) in the table, predicting an offset to compensate for the misalignment between 2D center and projected 3D center can improve the performance of 3D detection significantly. Then, using projected 3D center as the ground truth for coarse detection (setting **d**, our model) can further improve the performance. Besides, by comparing the setting **c** used in [23] and the setting **d** in our design, we can find the performance of 3D detection benefits from multi-task learning (performing 2D detection and 3D detection jointly). Note that the accuracy of 2D detection under setting **d** is also better than that under setting **c**, which suggests generating 2D bounding boxes from 3D detection may reduce the quality of the 2D detection results. The above conclusions are also reflected in Table 3 and 4.

Training samples. From Table 8, we can find that both removing some samples from training set appropriately and reducing the training weights of them can improve overall performance. Note that those samples are only a small part

Figure 5: **Qualitative results on the KITTI *test* set.** These results are based on proposed model trained on the KITTI *trainval* set, running at 25 FPS. We use blue, green, and red boxes to denote cars, pedestrians, and cyclists. LiDAR signals are only used for visualization. Best viewed in color with zoom in.

	Easy	Mod.	Hard
baseline	16.12	12.97	10.99
+ hard encoding, $s = 40$	14.25	11.25	9.63
+ hard encoding, $s = 60$	17.45	13.66	11.68
+ soft encoding, $c = 40$, T=1	14.50	11.74	9.95
+ soft encoding, $c = 60$, T=1	17.50	13.54	11.32
+ soft encoding, $c = 60$, T=5	17.25	13.03	11.01

Table 8: **Analysis for training samples.** Metrics is AP_{40} of the Car category for 3D detection.

of the whole training set and will not affect the representation learning of the network to the whole dataset. For example, in the 7,481 images in *trainval* set, only 1,301/767 samples beyond 60/65 meters, accounting for 4.5%/2.7% of the total 28,742 samples.

4.5. Qualitative Results

We visualize some representative outputs of the proposed method in Figure 5. To clearly show the object's position in the 3D world space, we also visualize the LiDAR signals. We can observe that our model outputs remarkably accurate 3D bounding boxes for the cases at a reasonable distance. We also find that our model outputs some false positive samples, *e.g.* the 3D box on the right in the sixth picture, and the foremost reason for that is the imprecise depth or center estimation. Note that the dimension and orientation estimation for those cases are still accurate.

5. Conclusion

In this paper, we systematically analyze the problems in monocular 3D detection and find the localization error is the bottleneck of this task. To alleviate this problem, we first revisit the misalignment between the center of the 2D bounding box and the projected center of 3D object. We argue that directly detecting projected 3D center can reduce the localization error and 2D detection is conducive to optimize 3D detection. Besides, we also find distant samples are almost impossible to detect accurately with the existing technologies, and discarding these samples from the training set will stop them from distracting the network. Finally, we also proposed an IoU oriented loss for 3D size estimation. Extensive experiments on the challenging KITTI dataset show the effectiveness of the proposed strategies.

6. Acknowledgement

This work was supported by SenseTime, the Australian Research Council Grant DP200103223, and Australian Medical Research Future Fund MRFAI000085.

References

[1] Daniel Bolya, Sean Foley, James Hays, and Judy Hoffman. Tide: A general toolbox for identifying object detection errors. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1

[2] 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. 1, 2, 6, 7

[3] Garrick Brazil, Gerard Pons-Moll, Xiaoming Liu, and Bernt Schiele. Kinematic 3d object detection in monocular video. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2, 6

[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. In *AAAI*, pages 10478–10485, 2020. 2, 3, 6

[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. 2

[6] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 6

[7] 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 *Advances in Neural Information Processing Systems*, pages 424–432, 2015. 6

[8] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 6

[9] Yilun Chen, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Dsgn: Deep stereo geometry network for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12536–12545, 2020. 1

[10] Yongjian Chen, Lei Tai, Kai Sun, and Mingyang Li. Monopair: Monocular 3d object detection using pairwise spatial relationships. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12093–12102, 2020. 1, 2, 3, 4, 6, 7

[11] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017. 7

[12] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 2, 6

[13] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2002–2011, 2018. 2, 6, 7

[14] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013. 6

[15] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. 6

[16] Clement Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2

[17] Derek Hoiem, Yodsawalai Chodpathumwan, and Qieyun Dai. Diagnosing error in object detectors. In *European conference on computer vision*, pages 340–353. Springer, 2012. 1

[18] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584, 2017. 3

[19] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019. 1

[20] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. 1, 3

[21] 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/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 6

[22] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 5

[23] Zechen Liu, Zizhang Wu, and Roland Toth. Smoke: Single-stage monocular 3d object detection via keypoint estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020. 2, 4, 6, 7

[24] Xinzhu Ma, Shinan Liu, Zhiyi Xia, Hongwen Zhang, Xingyu Zeng, and Wanli Ouyang. Rethinking pseudo-lidar representation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1, 2, 6

[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/CVF International Conference on Computer Vision (ICCV)*, October 2019. 1, 2, 6

[26] Fabian Manhardt, Wadim Kehl, and Adrien Gaidon. Roi-10d: Monocular lifting of 2d detection to 6d pose and metric

shape. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[27] 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. 2

[28] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 3

[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. 2, 6

[30] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 3

[31] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 658–666, 2019. 5

[32] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. *Proceedings of the British Machine Vision Conference (BMVC)*, 2018. 2

[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. 1

[34] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 5

[35] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel Lopez-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 1, 2, 3, 4, 5, 6, 7

[36] A. Simonelli, S. Rota Bulo, L. Porzi, M. Lopez Antequera, and P. Kontschieder. Disentangling monocular 3d object detection: From single to multi-class recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 2

[37] 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. 1, 2

[38] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio Savarese. Data-driven 3d voxel patterns for object category recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 2

[39] 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. 2

[40] Jun Xu, Yanxin Ma, Songhua He, and Jiahua Zhu. 3d-giou: 3d generalized intersection over union for object detection in point cloud. *Sensors*, 19(19):4093, 2019. 5

[41] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 3

[42] Dingfu Zhou, Jin Fang, Xibin Song, Chenye Guan, Junbo Yin, Yuchao Dai, and Ruigang Yang. Iou loss for 2d/3d object detection. In *2019 International Conference on 3D Vision (3DV)*, pages 85–94. IEEE, 2019. 5

[43] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. In *arXiv preprint arXiv:1904.07850*, 2019. 1, 2, 3, 5, 6, 7

[44] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018. 1

[45] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 7