Accurate Monocular 3D Object Detection via Color-Embedded 3D Reconstruction for Autonomous Driving

Xinzhu Ma[1], Zhihui Wang[1,2], Haojie Li[1,2,*], Pengbo Zhang[1], Wanli Ouyang[3], Xin Fan[1,2]
[1]Dalian University of Technology, China
[2]Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, China
[3]The University of Sydney, SenseTime Computer Vision Research Group, Australia
{maxinzhu@mail., zhwang@, hjli@, bobo96@mail., xin.fan}dlut.edu.cn
wanli.ouyang@sydney.edu.au

Abstract

In this paper, we propose a monocular 3D object detection framework in the domain of autonomous driving. Unlike previous image-based methods which focus on RGB feature extracted from 2D images, our method solves this problem in the reconstructed 3D space in order to exploit 3D contexts explicitly. To this end, we first leverage a standalone module to transform the input data from 2D image plane to 3D point clouds space for a better input representation, then we perform the 3D detection using PointNet backbone net to obtain objects' 3D locations, dimensions and orientations. To enhance the discriminative capability of point clouds, we propose a multi-modal features fusion module to embed the complementary RGB cue into the generated point clouds representation. We argue that it is more effective to infer the 3D bounding boxes from the generated 3D scene space (i.e., X,Y, Z space) compared to the image plane (i.e., R,G,B image plane). Evaluation on the challenging KITTI dataset shows that our approach boosts the performance of state-of-the-art monocular approach by a large margin.

1. Introduction

In recent years, with the development of technologies in computer vision and deep learning [12, 34, 36], numerous impressive methods are proposed for accurate 2D object detection [9, 8, 11, 32, 17, 24, 41, 18]. However, beyond getting 2D bounding boxes or pixel masks, 3D object detection is eagerly in demand in many applications such as autonomous driving and robotic applications because it can describe objects in a more realistic way. Now, this problem received more and more the concern of scholars. Because

* Corresponding author: hjli@dlut.edu.cn

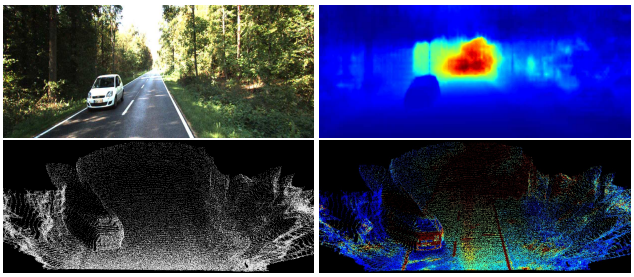

Figure 1. **Different representations of input data.** *Top left:* RGB image. *Top right:* Depth map. *Bottom left:* Point cloud. *Bottom right:* RGB augmenting point cloud (only R-channel is mapped for this visualization). Note that all the representations we mentioned can be generated by a single RGB image.

LiDAR provides reliable depth information that can be used to accurately localize objects and characterize their shapes, many approaches [14, 19, 21, 27, 5, 33, 40] use LiDAR point clouds as their input, and get impressive detection results in autonomous driving scenarios. In contrast, some other studies [1, 4, 3, 37, 23, 35, 15] are devoted to replace the LiDAR with cheaper monocular cameras, which are readily available in daily life. As LiDAR is much more expensive and inspired by the remarkable progress in image-based depth prediction techniques, this paper focuses on the high performance detection of 3D object utilizing only monocular images. However, image-based 3D detection is very challenging, and there is a huge gap between the performance of image-based methods and LiDAR-based methods. We show in this work that we can largely boost the performance of image-based 3D detection by transforming the input data representation.

Typical image-based 3D object detection approaches [1, 3, 4, 35] adopted the pipeline similar to 2D detectors and mainly focused on RGB features extracted from 2D images. However, these features are not suitable for 3D related tasks

because of the lack of spatial information. This is one of the main reasons why early studies failed to get better performance. An intuitive solution is that we can use a CNN to predict the depth maps [38, 39, 6] and then use them as input if we do not have the depth data available. Although depth information is helpful to 3D scene understanding, simply using it as an additional channel of RGB images such as [37] does not compensate for the performance difference between image-based methods and LiDAR-based method. There is no doubt that LiDAR data is much more accurate than estimated depth, here we argue that the performance gap not only due to the accuracy of the data, but also its representation (see Fig. 1 for different input representations on monocular 3D detection task). In order to narrow the gap and and make the estimated depth a bigger role, we need a more explicit representation form such as point cloud which describes a real world 3D coordinates rather than depth with a relative position in images. For example, objects with different positions in 3D world may have the same coordinates in image plane, which brings difficulties for the network to estimate the final results. The benefits for transform depth map into point cloud can be enumerated as follow: (1) Point cloud data shows the spatial information explicitly, which make it easier for network to learn the non-linear mapping from input to output. (2) Richer features can be learnt by the network because some specific spatial structures exist only in 3D space. (3) The recent significant progress of deep learning on point clouds provides a solid building brick, which we can estimate 3D detection results in a more effective and efficient way.

Based on the observations above, a monocular 3D object detection framework is proposed. The main idea for the design of our method is to find a better input representation. Specifically, we first learn to use front-end deep CNNs and the input RGB data to produce two intermediate tasks involving 2D detection [25, 26, 8] and depth estimation [6, 39] (see Fig. 2). Then, we transform depth maps into point clouds with the help of camera calibration files in order to give the 3D information explicitly and used them as input data for subsequent steps. Besides, another crucial component that ensures the performance of proposed method is multi-modal features fusion module. After aggregating RGB information which is complementary to 3D point clouds, the discriminative capability of features used to describe 3D object are further enhanced. Note that, when the optimization of the all networks are finished, the inference phase is only based on the RGB input.

The contributions of this paper can be summarized as:

- We propose a new framework for monocular 3D object detection which transforms t 2D image to 3D point cloud and performs the 3D detection effectively and efficiently.
- We design an features fusion strategy to fully exploit the advantages of RGB cue and point cloud to boost the detection performance, which can be also applied in other scenarios such as LiDAR-based 3D detection.
- Evaluation on the challenging KITTI dataset [7] shows our method outperform all state-of-the-art monocular methods by around 15% and 11% higher AP on 3D localization and detection tasks, respectively.

2. Related Work

We briefly review existing works on 3D object detection task based on LiDAR and images in autonomous driving scenario.

Image-based 3D Object Detection: In the early works, monocular-based methods share similar framework with 2D detection [8], but it is much more complicated for estimating the 3D coordinates (x, y, z) of object center, since only image appearance cannot decide the absolute physical location. Mono3D [3] and 3DOP [4] focus on 3D object proposals generation using prior knowledge (e.g., object size, ground plane) from monocular and stereo images, respectively. Deep3DBox [23] introduces geometric constraints based on the fact that the 3D bounding box should fit tightly into 2D detection bounding box. Deep MANTA [1] encodes 3D vehicle information using key points, since they are rigid objects with well known geometry. Then the vehicle recognition in Deep MANTA can be considered as extra key points detection.

Although these methods propose some effective prior knowledge or reasonable constraints, they fail to get promising performance because of the lack of spatial information. Another recently proposed method [37] for monocular 3D object detection introduces a multi-level fusion based scheme utilizes a stand-alone module to estimate the disparity information and fuse it with RGB information in the input data encoding, 2D box estimation and 3D box estimation phase, respectively. Although it used depth (or disparity) many times, they only regard it as auxiliary information of RGB features, and do not make full use of its potential value. In comparison, our method takes the generated depth as the core feature and transform it into 3D space to explicitly make use of its spatial information. Pseudo-LiDAR [31] also find that data presentation plays an important role in 3D detection task. It pays more attention to verify the universality of point cloud representation and applies the generated points to some different existing 3D detection methods *without any modifications*. In contrast, in addition to transforming data representations, we further design a dedicated 3D detection framework for monocular images.

LiDAR-based 3D Object Detection: Although our approach is for monocular image data, we transform the data

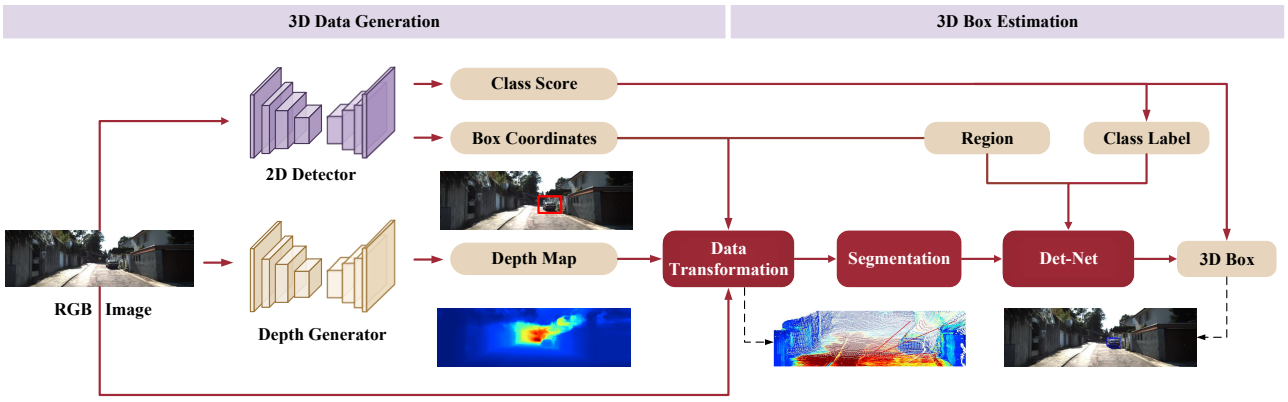

Figure 2. The proposed framework for monocular 3D object detection.

representation into point cloud which is same to LiDAR-based methods. So, we also introduce some typical approach based on LiDAR. MV3D [5] encode 3D point clouds with multi-view feature maps, enabling region-based representation for multimodal fusion. With the development of deep learning on raw point clouds [28, 29, 13], several detection approaches only based on raw LiDAR data are also proposed. Qi *et al*. [27] extend PointNet to 3D detection task by extracting the frustum point clouds corresponding to their 2D detections. VoxelNet [42] divides point clouds into equally spaced 3D voxels and transforms a group of points within each voxel into a unified feature representation. Finally, 2D convolution layers are used on these high-level voxel-wise features to get spatial features and give prediction results. Despite these two methods get a promising detection results, they do not make a good use of RGB information. In comparison, we also introduce a RGB features fusion module to enhance the discriminative capability of point clouds.

3. Proposed Method

In this section, we describe the proposed framework for monocular-based 3D object detection. We first present an overview of the proposed method, and then introduce the details of it. Finally, we show the optimization and implementation details for the overall network.

3.1. Approach Overview

As shown in Fig. 2, the proposed 3D detection framework consists of two main stages. In 3D data generation phase, we trained two deep CNNs to do intermediate tasks (2D detection and depth estimation) to get position and depth information. In particular, we transfer the generated depth into point cloud which is a better representation for 3D detection, and then we use 2D bounding box to get the prior information about the location of the RoI (region of interest). Finally, we extract the points in each RoI as our input data for subsequent steps. In 3D box estimation phase, in order to improve the final task, we design two modules for background points segmentation and RGB information aggregation, respectively. After that, we use PointNet as our backbone net to predict the 3D location, dimension and orientation for each RoI. Note that the confidence scores of 2D boxes are assigned to their corresponding 3D boxes.

3.2. 3D Data Generation

Intermediate tasks. As we all know that 3D detection using only monocular images is a very challenging task because image appearance can not determine the 3D coordinates of the object. Therefore, we train two deep CNN to generate depth map and 2D bounding box to provide spatial information and position prior. We adopt some existing algorithms to do these intermediate tasks, and give a detailed analysis of the impact of these algorithms on overall performance in experiment part.

Input representation. This work focuses more on how to use depth information than on how to get them. We believe that one of the main reasons why previous images-based 3D detectors fails to get better results is they don't make good use of depth maps. Simply using depth map as an additional channel of RGB image such as [39, 20], and then expecting neural network to extract effective features automatically is not the best solution. In contrast, we transform the estimated depth into point cloud with the help of camera calibration file provided by KITTI (see Fig. 1 for different input representations) and then use it as our data input form. Specifically, given a pixel coordinate (u, v) with depth d in the 2D image space, the 3D coordinates (x, y, z) in camera coordinate system can be computed as:

$$\begin{cases} z = d, \\ x = (u - C_x) * z/f, \\ y = (v - C_y) * z/f, \end{cases} \tag{1}$$

where f is the focal length of the camera, (C_x, C_y) is the principal point. The input point cloud S can be generated using depth map and 2D bounding box $\mathbf{B}$ as follow:

$$S = \{\, p \mid p \leftarrow \mathbf{F}(v),\ v \in \mathbf{B}\}, \tag{2}$$

where v is the pixel in depth map and $\mathbf{F}$ is the transforming function introduced by Eq. 1. It should be noted that, like most of monocular-based methods, we use camera calibration file in our approach. Actually, we can also use a point cloud encoder-decoder net to learn a mapping from (u, v, d) to (x, y, z), thus we don't need camera during the testing phase any more. In our measurements, we observe that there is no visible performance difference between these two methods. This is because the error introduced in the point cloud generation phase is much less than the noise contained in the depth map itself.

3.3. 3D Box Estimation

Point segmentation. After the 3D data generation phase, the input data is encoded as points cloud. However, there are many background points in these data and these background points should be discarded in order to estimate the position of target accurately. Qi *et al*. [27] propose a 3D instance segmentation PointNet to solve this problem in LiDAR data. But that strategy requires additional preprocessing to generate segmentation labels from 3D object ground truth. More importantly, there will be severe noise even if we use the same labelling method because the points we reconstruct are relatively unstable. For these reasons, we propose a simple but effective segmentation method based on depth prior to segment the points. Specifically, we first compute the depth mean in each 2D bounding box in order to get the approximate position of RoI, and use it as the threshold. All points with Z-channel value greater than this threshold are considered as background points. The processed point set S' can be expressed as:

$$S' = \{\, p \mid p_v \leq \frac{\sum_{p \in S} p_v}{|S|} + r,\ p \in S\}, \tag{3}$$

where p_v denotes the Z-channel value (which is equal to depth) of the point and r is a bias used to correct the threshold. Finally, we randomly select a fixed number of points in point set S' as the output of this module in order to ensuring consistency of number of subsequent network's input points.

3D box estimation. Before we estimate final 3D results, we follow [27] to predict the center δ of RoI using a lightweight network and use it to update the point cloud as follow:

$$S'' = \{\, p \mid p - \delta,\ p \in S'\}, \tag{4}$$

where S'' is the set of points we used to do final task. Then, we choose PointNet [28] as our 3D detection backbone network to estimate the 3D object which is encoded by its center (x, y, z), size (h, w, l) and heading angle θ. Same as other works, we only consider one orientation because of the assumption that the road surface is flat and the other two angles do not have possible variation. One other thing to note is that the center C we estimate here is a 'residual' center, which means the real center is $C + \delta$. Finally, we assign the confidence scores of the 2D bounding boxes to their corresponding 3D detection results.

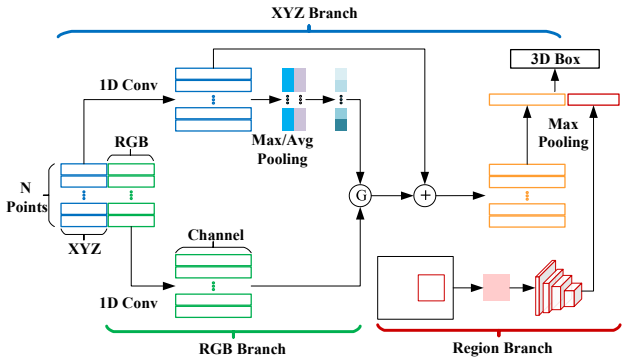

Figure 3. 3D box estimation (Det-Net) with RGB features fusion module. $\mathbf{G}$ is an attention map generated using Eq. 6.

3.4. RGB Information Aggregation

In order to further improve the performance and robustness of our method, we propose to aggregate complementary RGB information to point cloud. Specifically, we add RGB information to the generated point cloud by replacing Eq. 2 with:

$$S = \{\, p \mid p \leftarrow [\mathbf{F}(v), \mathbf{D}(v)],\ v \in \mathbf{B}\}, \tag{5}$$

where $\mathbf{D}$ is a function which output the corresponding RGB values of input point. In this way, the points are encoded as 6D vectors: $[x, y, z, r, g, b]$. However, simply relying on this simple method (we call it 'plain concat' in experiment part) to add RGB information is not feasible. So, as shown in Fig. 3, we introduce an attention mechanism for the fusion task. The attention mechanism has been successfully applied in various tasks such as image caption generation and machine translation for selecting useful information. Specifically, we utilize the attention mechanism for guiding the message passing between the spatial features and RGB features. Since the passed information flow is not always useful, the attention can act as a gate function to control the flow, in other words to make the network automatically learn to focus or to ignore information from other features. When we pass RGB message to its corresponding point, an attention map $\mathbf{G}$ is first produced from the feature maps $\mathbf{F}$ generated from XYZ branch as follow:

$$\mathbf{G} \leftarrow \sigma(f([\mathbf{F}^{xyz}_{max}, \mathbf{F}^{xyz}_{avg}])), \tag{6}$$

Method	Data	IoU=0.5			IoU=0.7		
		Easy	Moderate	Hard	Easy	Moderate	Hard
Mono3D [3]	Mono	30.50	22.39	19.16	5.22	5.19	4.13
Deep3DBox [23]	Mono	30.02	23.77	18.83	9.99	7.71	5.30
Multi-Fusion [37]	Mono	55.02	36.73	31.27	22.03	13.63	11.60
ROI-10D [20]	Mono	-	-	-	14.76	9.55	7.57
Psudeo-LiDAR [31]	Mono	70.8	49.4	42.7	40.6	26.3	22.9
Ours	Mono	**72.64**	**51.82**	**44.21**	**43.75**	**28.39**	**23.87**

Table 1. 3D localization performance: Average Precision (AP_{loc}) (in %) of bird's eye view boxes on KITTI *validation* set.

Method	Data	IoU=0.5			IoU=0.7		
		Easy	Moderate	Hard	Easy	Moderate	Hard
Mono3D [3]	Mono	25.19	18.20	15.52	2.53	2.31	2.31
Deep3DBox [23]	Mono	27.04	20.55	15.88	5.85	4.10	3.84
Multi-Fusion [37]	Mono	47.88	29.48	26.44	10.53	5.69	5.39
ROI-10D [20]	Mono	-	-	-	10.25	6.39	6.18
MonoGRNet [30]	Mono	50.51	36.97	30.82	13.88	10.19	7.62
Psudeo-LiDAR [31]	Mono	66.3	42.3	38.5	28.2	18.5	16.4
Ours	Mono	**68.86**	**49.19**	**42.24**	**32.23**	**21.09**	**17.26**

Table 2. 3D detection performance: Average Precision (AP_{3D}) (in %) of 3D boxes on KITTI *validation* set.

where f is the nonlinear function learned from a convolution layer and σ is a sigmoid function for normalizing the attention map. Then the message is passed with the attention map controlled as follow:

$$\mathbf{F}^{xyz} \leftarrow \mathbf{F}^{xyz} + \mathbf{G} \odot \mathbf{F}^{rgb}, \tag{7}$$

where $\odot$ denotes element-wise multiplication. In addition to point-level features fusion, we also introduce another branch to provide object-level RGB information. In particular, we first crop the RoI from RGB image and resize it to 128×128. Then we use a CNN to extract the object-level feature maps $\mathbf{F}^{obj}$ and the final feature maps set $\mathbf{F}$ obtained from the fusion module is: $\mathbf{F} \leftarrow CONCAT(\mathbf{F}^{xyz}, \mathbf{F}^{obj})$, where $CONCAT$ denotes the concatenation operation.

3.5. Implementation Details.

Optimization. The whole training process is performed with two phases. In the first phase, we only optimize the intermediate nets according to the training strategies of original papers. After that, we simultaneously optimize the two networks for 3D detection jointly with a multi-task loss function:

$$L = L_{loc} + L_{det} + \lambda L_{corner}, \tag{8}$$

where L_{loc} is the loss function for the lightweight location net (center only) and L_{loc} is for 3D detection net (center, size and heading angle). We also use the corner loss [27] where the output targets are first decoded into oriented 3D boxes and then smooth L1 loss is computed on the (x, y, z) coordinates of eight box corners directly with regard to ground truth. We train the nets for 200 epochs using Adam optimizer with batch size of 32. The learning rate is initially set to 0.001 and reduced by half for every 20 epochs. The whole training process can be completed in one day.

Implementation details. The proposed method is implemented base on PyTorch and on Nvidia 1080Ti GPUs. The two intermediate networks of proposed method naturally supports any network structure. We implement some different methods as described in their papers exactly, and the relevant analysis can be found in experimental part. For the 3D detection nets, we use PointNet as our backbone nets and train them from scratch with random initialization. Moreover, the dropout strategy with keep rate 0.7 is applied into every fully connected layers except the last one. For the RGB values, we first normalize the range of them to (0, 1) by dividing 255, and then the data distribution of each color channel is regularized into standard normal distribution. For the region branch in RGB features fusion module, we use ResNet-34 with half channels and global pooling to get the 1×1×256 features.

4. Experimental Results

We evaluate our approach on the challenging KITTI dataset [7] which provides 7,481 images for training and 7,518 images for testing. Detection and localization tasks are evaluated in three regimes: *easy*, *moderate* and *hard*, according to the occlusion and truncation levels of objects. Since the ground truth for the test set is not available and the access to the test server is limited, we conduct comprehensive evaluation using the protocol described in [3, 4, 5], and

subdivide the training data into a *training* set and a *validation* set, which results in 3,712 data samples for training and 3,769 data samples for validation. The split avoids samples from the same sequence being included in both *training* and *validation* set[3].

4.1. Comparing with other methods

Baselines. As this work aims at monocular 3D object detection, our approach is mainly compared to other methods with only monocular images as input. Here five methods are chosen for comparisons: Mono3D [3], Deep3DBox [23] and Multi-Fusion [37], ROI-10D [20], MonoGRNet [30] and Pseudo-LiDAR [31].

Car. The evaluation results of 3D localization and detection tasks on KITTI *validation* set are presented in Table 1 and 2, respectively. The proposed method consistently outperforms all the competing approaches across all three difficulty levels. For localization task, the proposed method outperforms Multi-Fusion [37] by ∼**15** AP_{loc} in *moderate* setting. For 3D detection task, our method achieves ∼**12.2** and ∼**10.9** AP_{3D} improvement (moderate) over the recently proposed MonoGRNet [30] under IoU thresholds of 0.5 and 0.7. In the *easy* setting, our improvement is more prominent. Specifically, our method achieves ∼**21.7** and ∼**18.4** improvement over previous state-of-the-art on localization and detection tasks (IoU=0.7). Note that there is no complicated prior knowledge or constraints such as [3, 4, 20], which strongly confirms the importance of data representation.

Compared with Pseudo-LiDAR [31], which is concurrent to this work, the proposed method has about ∼**1.5** AP improvement on each metric. This is because of the modification of the background points segmentation algorithm and the introduction of RGB information. We will discuss this in detail in Sec. 4.2.

Table 3 shows the results on *testing* set, and more details, such as Precision-Recall curve, can be found on KITTI official server. The *testing* set results also show the superiority of our method in performance compared with others.

Method	Task	Easy	Moderate	Hard
Multi-Fusion [37]	Loc.	13.73	9.62	8.22
RoI-10D [20]	Loc.	16.77	12.40	11.39
Ours	Loc.	**27.91**	**22.24**	**18.62**
Multi-Fusion [37]	Det.	7.08	5.18	4.68
RoI-10D [20]	Det.	12.30	10.30	9.39
Ours	Det.	**21.48**	**16.08**	**15.26**

Table 3. AP(%) for 3D localization (Loc.) and 3D detection (Det.) tasks on the KITTI *testing* set.

4.2. Detailed analysis of proposed method

In this section we provide analysis and ablation experiments to validate our design choices.

RGB information. We further evaluate the effect of the proposed RGB fusion module, and the baselines are the proposed method without RGB values and using them as additional channels of generated points. Table 4 shows the relevant results for *Car* category on KITTI. It can be seen that the proposed module obtains around **2.1** and **1.6** mAP improvement (moderate) on localization and detection task, respectively. The qualitative comparisons can be found in Fig 6. Quantitative and qualitative results both show the effectiveness of proposed RGB fusion module. Besides, one thing to note is that incorrect use of RGB information such as plain concat will lead to performance degradation.

	Task	Easy	Moderate	Hard
w/o RGB	Loc.	41.29	26.28	22.75
plain concat	Loc.	36.17	25.34	21.94
ours	Loc.	**43.75**	**28.39**	**23.87**
w/o RGB	Det.	30.73	19.46	16.72
plain concat	Det.	27.20	18.25	16.15
ours	Det.	**32.23**	**21.09**	**17.26**

Table 4. Ablation study of RGB information. The metric is $AP_{3D}^{0.7}$ on KITTI *validation* set.

Points segmentation. We compare the proposed points segmentation method and the 3D segmentation PointNet which is used in [27]. The baseline is to estimate 3D boxes directly using point clouds with noise which can be regarded as all points are classified into positive samples. As shown in Table 5, our prior-based method outperforms baseline and segmentation PointNet obviously which proves the effectiveness of the proposed method and Table 6 shows that the proposed method is robust for varying thresholds. Meanwhile, the experimental results also show that the learning-based method is not applicable to approximate point clouds segmentation task because it's difficult to obtain reliable labels. Besides, the proposed method is also much faster than segmentation PointNet (about 5ms on CPU *v.s.* 20ms on GPU for each proposal).

	IoU	Easy	Moderate	Hard
w/o segmentation	0.5	66.42	44.53	40.60
seg-net used in [27]	0.5	67.01	45.51	40.65
ours	0.5	**68.86**	**49.19**	**42.24**
w/o segmentation	0.7	27.04	18.22	16.13
seg-net used in [27]	0.7	29.49	18.70	16.57
ours	0.7	**32.23**	**21.09**	**17.26**

Table 5. Ablation study of points segmentation. The metric is $AP_{3D}^{0.7}$ on KITTI *validation* set.

r	Easy	Moderate	Hard
-0.5	31.13	20.01	16.81
0.0	31.87	20.55	17.03
0.5	**32.23**	**21.09**	**17.26**
1.0	31.93	20.93	17.18

Table 6. $AP_{3D}^{0.7}(\%)$ of different points segmentation threshold r (in meters) for 3D detection on the KITTI *validation* set.

Depth maps. As described in Sec. 3, our approach depends on the point clouds generated from the output of depth generator. In order to study the impact of quality of depth maps on the overall performance of proposed method, we implemented four different depth generators [10, 16, 22, 2]. From the results shown in Table 7, we find that 3D detection accuracy increases significantly when using more accurate depth (more details about the accuracy of depth maps can be found in the supplement material). It's worth noting that even if we use the unsupervised monocular depth generator [10], the proposed method still outperforms [20] by a large margin.

Depth	Task	Easy	Mod.	Hard
MonoDepth[10]	Loc.	32.42	20.26	17.21
DORN[16]	Loc.	43.75	28.39	23.87
DispNet[22]	Loc.	47.41	30.72	25.66
PSMNet [2]	Loc.	60.18	34.01	30.32
MonoDepth[10]	Det.	23.12	15.45	14.19
DORN[16]	Det.	32.23	21.09	17.26
DispNet[22]	Det.	36.97	23.69	19.25
PSMNet [2]	Det.	45.85	26.03	23.16

Table 7. Comparisons of different depth generators. Metrics are $AP_{loc}^{0.7}$ and $AP_{3D}^{0.7}$ on KITTI *validation* set.

Sampling quantity. Some studies such as [28, 29] observe that classification/segmentation accuracy will decrease dramatically as the number of points decreases, and we will show that our approach is not so sensitive to the number of points. In our approach, we randomly select a fixed number (512 points for default configuration) of point clouds to do 3D detection task. Table. 8 shows the performance of proposed method under different sampling quantity. According to the results, AP_{3D} will increase as the number of points increases at the beginning. Then, after reaching a certain level ($\sim$512 points), the performance tends to be stable. It is worth noting that we still get a relatively good detection performance even if there are few sampling points.

Robustness. We show that the proposed method is robust to various kinds of input corruptions. We first set the sampling quantity to 512 in training phase, but use different values in the testing phase. Fig. 4.2 shows that the proposed method has more than 70% AP_{3D} even when 80% of the points are missed. Then, we test the robustness of model to point perturbations, and the results are shown in Fig 4.2.

Sampling Quantity	Easy	Mod.	Hard
64	27.91	19.41	16.31
128	29.72	19.62	16.64
256	30.99	20.71	17.18
512	**32.23**	**21.09**	**17.26**
1024	31.44	21.01	17.23

Table 8. Comparisons of different sampling quantity. The metric is $AP_{3D}^{0.7}(\%)$ on KITTI *validation* set. Note that the number of sample points is consistent at the training and testing phase.

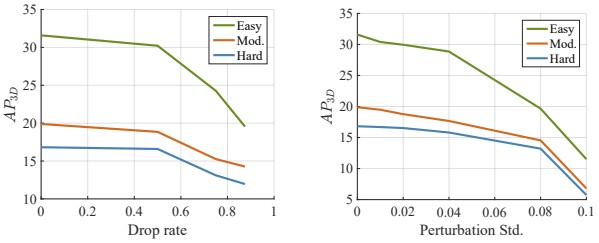

Figure 4. *Left*: robustness test of random point dropout. *Right*: robustness test of random perturbations (Gaussian noise is added into each point independently). The metric is $AP_{3D}^{0.7}(\%)$ for *Car* on KITTI *validation* set.

Network architecture. We also investigate the impact of different 3D detection network architectures on overall performance (the previously reported results are all based on PointNet), and the experimental result are shown in Table. 9.

	Data	Easy	Mod.	Hard
PointNet [28]	Mono	32.23	21.09	17.26
PointNet++ [29]	Mono	33.17	21.71	17.61
RSNet [13]	Mono	33.93	22.34	17.79

Table 9. Comparisons of different 3D detection network architectures. The metric is $AP_{3D}^{0.7}(\%)$ on KITTI *validation* set.

4.3. Qualitative Results and Failure Mode

We visualize some detection results of our approach in Fig. 5 and a typical localization result in Fig. 7. In general, our algorithm can get a good detection result. However, because it's a 2D-driven framework, the proposed method will fail if the 2D box is a false positive sample or missing. Besides, for distant objects, our algorithm is difficult to give accurate results because the depth is not reliable (the leftmost car in Fig. 7 is 70.35 meters away from the camera).

5. Conclusions

We proposed a framework for accurate 3D object detection with monocular images in this paper. Unlike other

Figure 5. **Qualitative comparisons of 3D detection results:** 3D Boxes are projected to the image plane. White boxes represent our predictions, and blue boxes come from ground truth.

Figure 6. **Qualitative comparisons of RGB information:** 3D Boxes are projected to the image plane. The detection results using XYZ information only are represented by write boxes, and blue boxes come from the model trained with RGB features fusion module. The proposed RGB fusion method can improve the 3D detection accuracy, especially for occlusion/truncation cases.

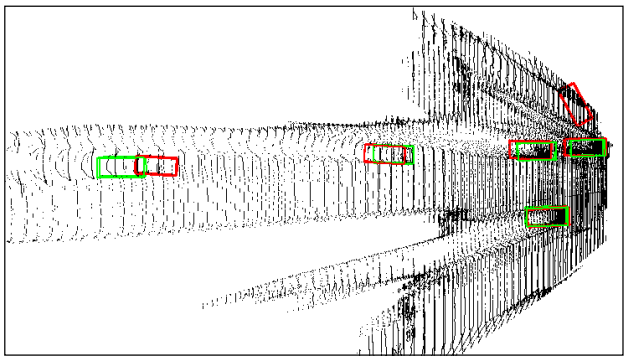

Figure 7. **A qualitative result of 3D localization :** 3D Boxes are projected to the ground plane. Red boxes represent our predictions, and green boxes come from ground truth.

image-based methods, our method solves this problem in the reconstructed 3D space in order to exploit 3D contexts explicitly. We argue that the point cloud representation is more suitable for 3D related tasks than depth maps. Besides, we propose a multi-modal feature fusion module to embed the complementary RGB cue into the generated point clouds representation to enhance the discriminative capability of generated point clouds. Our approach significantly outperforms existing monocular-based method for 3D localization and detection tasks on KITTI benchmark. In addition, the extended versions verifies the design strategy can also be applied to stereo-based and LiDAR-based methods.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grants No. 61976038, 61932020 and No. 61772108.

References

[1] Florian Chabot, Mohamed Chaouch, Jaonary Rabarisoa, Céline Teulière, and Thierry Chateau. Deep manta: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2040–2049, 2017. 1, 2

[2] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5410–5418, 2018. 7

[3] 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)*, pages 2147–2156, 2016. 1, 2, 5, 6

[4] 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. 1, 2, 5, 6

[5] 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)*, volume 1, page 3, 2017. 1, 3, 5

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

[7] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 2, 5

[8] Ross Girshick. Fast r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015. 1, 2

[9] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 580–587, 2014. 1

[10] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, page 7, 2017. 7

[11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988. IEEE, 2017. 1

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 1

[13] Qiangui Huang, Weiyue Wang, and Ulrich Neumann. Recurrent slice networks for 3d segmentation of point clouds. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2626–2635, 2018. 3, 7

[14] Bo Li. 3d fully convolutional network for vehicle detection in point cloud. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 1513–1518. IEEE, 2017. 1

[15] Buyu Li, Wanli Ouyang, Lu Sheng, Xingyu Zeng, and Xiaogang Wang. Gs3d: An efficient 3d object detection framework for autonomous driving. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1

[16] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 641–656, 2018. 7

[17] Tsung-Yi Lin, Priyal Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE transactions on pattern analysis and machine intelligence*, 2018. 1

[18] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *arXiv preprint arXiv:1809.02165*, 2018. 1

[19] Wenjie Luo, Bin Yang, and Raquel Urtasun. Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3569–3577, 2018. 1

[20] Fabian Manhardt, Wadim Kehl, and Adrien Gaidon. Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 3, 5, 6, 7

[21] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015. 1

[22] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4040–4048, 2016. 7

[23] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Košecká. 3d bounding box estimation using deep learning and geometry. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5632–5640. IEEE, 2017. 1, 2, 5, 6

[24] Wanli Ouyang, Kun Wang, Xin Zhu, and Xiaogang Wang. Chained cascade network for object detection. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1

[25] Wanli Ouyang and Xiaogang Wang. Joint deep learning for pedestrian detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2056–2063, 2013. 2

[26] Wanli Ouyang, Hui Zhou, Hongsheng Li, Quanquan Li, Junjie Yan, and Xiaogang Wang. Jointly learning deep features, deformable parts, occlusion and classification for pedestrian detection. *IEEE transactions on pattern analysis and machine intelligence*, 40(8):1874–1887, 2017. 2

[27] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1, 3, 4, 5, 6

[28] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 3, 4, 7

[29] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5099–5108, 2017. 3, 7

[30] 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. 5, 6

[31] Jiaxiong Qiu, Zhaopeng Cui, Yinda Zhang, Xingdi Zhang, Shuaicheng Liu, Bing Zeng, and Marc Pollefeys. Deeplidar: Deep surface normal guided depth prediction for outdoor scene from sparse lidar data and single color image. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2, 5, 6

[32] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 1

[33] Zhile Ren and Erik B Sudderth. Three-dimensional object detection and layout prediction using clouds of oriented gradients. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1525–1533, 2016. 1

[34] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1

[35] Shuran Song and Jianxiong Xiao. Sliding shapes for 3d object detection in depth images. In *European conference on computer vision*, pages 634–651. Springer, 2014. 1

[36] Shuyang Sun, Jiangmiao Pang, Jianping Shi, Shuai Yi, and Wanli Ouyang. Fishnet: A versatile backbone for image, region, and pixel level prediction. In *Advances in Neural Information Processing Systems*, pages 754–764, 2018. 1

[37] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3d object detection from monocular images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1, 2, 5, 6

[38] Dan Xu, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing. 2018. 2

[39] Dan Xu, Elisa Ricci, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Monocular depth estimation using multi-scale continuous crfs as sequential deep networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. 2, 3

[40] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1

[41] Xingyu Zeng, Wanli Ouyang, Junjie Yan, Hongsheng Li, Tong Xiao, Kun Wang, Yu Liu, Yucong Zhou, Bin Yang, Zhe Wang, et al. Crafting gbd-net for object detection. *IEEE transactions on pattern analysis and machine intelligence*, 40(9):2109–2123, 2017. 1

[42] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 3