Learning Occupancy for Monocular 3D Object Detection

Liang Peng[3] Junkai Xu[1,3] Haoran Cheng[1,3] Zheng Yang[3] Xiaopei Wu[1]
Wei Qian[3] Wenxiao Wang[2] Boxi Wu[2] * Deng Cai[1]
[1]State Key Lab of CAD&CG, Zhejiang University
[2]School of Software Technology, Zhejiang University
[3]FABU Inc.
{pengliang, xujunkai, haorancheng}@zju.edu.cn

Abstract

Monocular 3D detection is a challenging task due to the lack of accurate 3D information. Existing approaches typically rely on geometry constraints and dense depth estimates to facilitate the learning, but often fail to fully exploit the benefits of three-dimensional feature extraction in frustum and 3D space. In this paper, we propose ***OccupancyM3D****, a method of learning occupancy for monocular 3D detection. It directly learns occupancy in frustum and 3D space, leading to more discriminative and informative 3D features and representations. Specifically, by using synchronized raw sparse LiDAR point clouds, we define the space status and generate voxel-based occupancy labels. We formulate occupancy prediction as a simple classification problem and design associated occupancy losses. Resulting occupancy estimates are employed to enhance original frustum/3D features. As a result, experiments on KITTI and Waymo open datasets demonstrate that the proposed method achieves a new state of the art and surpasses other methods by a significant margin.*

1. Introduction

Three dimensional object detection is a critical task in many real-world applications, such as self-driving and robot navigation. Early methods [52, 58, 78] typically rely on LiDAR sensors because they can produce sparse yet accurate 3D point measurements. In contrast, cameras provide dense texture features but lack 3D information. Recently, monocular-based methods [36, 39, 49, 53] for 3D detection, also known as monocular 3D detection, have gained significant attention from both industry and academia due to their cost-effectiveness and deployment-friendly nature.

Recovering accurate 3D information from a single RGB image poses a challenge. While previous researches have employed geometry constraints [7, 27, 36, 45] and dense depth estimates [38, 53, 67] to facilitate 3D reasoning, they often overlook the importance of discriminative and informative 3D features in 3D space, which are essential for effective 3D detection. They mainly focus on improving features in 2D space, with little attention paid to better feature encoding and representation in the frustum and 3D space.

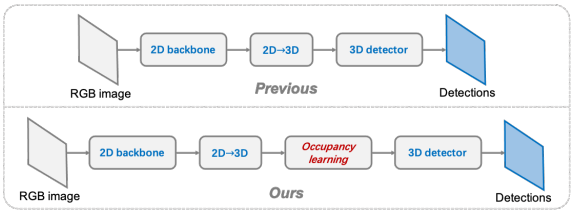

Figure 1. Overall design. We introduce occupancy learning for monocular 3D detection. Best viewed in color.

Towards this goal, in this paper we propose to learn occupancy in frustum and 3D space, to obtain more discriminative and informative 3D features/representations for monocular 3D detection. Specifically, we employ synchronized raw sparse LiDAR point clouds to generate voxel-based occupancy labels in frustum and 3D space during the training stage. Concerning the sparsity of LiDAR points, we define three occupancy statuses: *free, occupied, and unknown*. Based on this, we voxelize the 3D space and use ray tracing on each LiDAR point to obtain occupancy labels. With the occupancy labels, we can enforce explicit 3D supervision on intermediate 3D features. It allows the network to learn voxelized occupancy for current 3D space, which enhances the original 3D features. This process is also performed in the frustum space, enabling a more fine-grained manner in extracting three-dimensional features for near objects due to the perspective nature of camera imagery. Overall, we call the proposed occupancy learning method **OccupancyM3D**, and illustrate the framework overview in Figure 1.

To demonstrate the effectiveness of our method, we conduct experiments on the competitive KITTI and Waymo open datasets. As a result, the proposed method achieves

*Corresponding author.

state-of-the-art results with a significant margin over other methods. Our contributions can be summarized as follows:

- We emphasize the importance of feature encoding and representation in the frustum and 3D space for monocular 3D detection, and we propose to learn occupancy in both space.
- We propose a method to generate occupancy labels using synchronized raw sparse LiDAR points and introduce corresponding occupancy losses, enabling the network to learn voxelized occupancy in both frustum and 3D space. This occupancy learning process facilitates the extraction of discriminative 3D features in the network.
- Experiments demonstrate the superiority of the proposed method. Evaluated on challenging KITTI and Waymo open datasets, our method achieves new state-of-the-art (SOTA) results and outperforms other methods by a significant margin.

2. Related Work

2.1. LiDAR Based 3D Object Detection

LiDAR-based methods [19, 26, 56, 68, 69, 71, 77] currently dominate 3D object detection accuracy because of their precise depth measurements. Due to the unordered nature of point clouds, LiDAR-based methods are required to organize the input data. There are four main streams based on the input data representation: point-based, voxel-based, range-view-based, and hybrid-based. PointNet families [50, 51] are effective methods for feature extraction from raw point clouds, allowing point-based methods [52, 55, 59, 72] to directly perform 3D detection. Voxel-based methods [13, 70, 73, 77, 78] organize point clouds into voxel grids, making them compatible with regular convolutional neural networks. Range-view-based methods [1, 5, 15] convert point clouds into range-view to accommodate the LiDAR scanning mode. Hybrid-based methods [8, 46, 57, 71] use a combination of different representations to leverage their individual strengths. There is still a significant performance gap between monocular and LiDAR-based methods, which encourages researchers to advance monocular 3D detection.

2.2. Monocular 3D Object Detection

Significant progress has been made in advancing monocular 3D detection in recent years. The ill-posed problem of recovering instance level 3D information from a single image is challenging and important, attracting many researches. This is also the core sub-problem in monocular 3D detection. Early works [9, 45] resort to using scene priors and geometric projections to resolve objects' 3D locations. More recent monocular methods [2, 27, 29, 31, 35, 40, 76] employ more geometry constraints and extra priors like CAD models to achieve this goal. AutoShape [35] incorporates shape-aware 2D/3D constraints into the 3D detection framework by learning distinguished 2D and 3D keypoints. MonoJSG [31] reformulates the instance depth estimation as a progressive refinement problem and propose a joint semantic and geometric cost volume to model the depth error. As RGB images lack explicit depth information, many works rely on dense depth estimates. Some methods [37, 38, 67] directly convert depth map to pseudo LiDAR or 3D coordinate patches, and some works [53] use depth distributions to lift 2D image features to 3D space. Therefore, previous well-designed LiDAR 3D detectors can be easily employed on such explicit 3D features. Other researches [11, 12, 14, 47, 49, 64] also take advantage of depth maps or LiDAR point clouds as guidance for feature extraction and auxiliary information. While previous works have leveraged geometry constraints and dense depth estimates, they have not fully explored feature encoding and representation in the frustum and 3D space. To address this, our method focuses on learning occupancy for monocular 3D detection.

2.3. 3D Scene Representations

Recent researches [41, 43, 75] rapidly advance implicit representations. Implicit representations have the advantage of arbitrary-resolution on modeling the 3D scene. This nature is beneficial for fine-grained tasks such as 3D reconstruction and semantic segmentation. Different from them, monocular 3D detection is an instance level task, and we explore explicit occupancy learning using fixed-sized voxels. Implicit occupancy representations for this task can be explored in future works, which is an interesting and promising topic. Additionally, many bird's-eye-view (BEV) based works [20, 28, 30, 34, 53, 54] have been proposed recently. These works commonly employ BEV representations and obtain great success, especially for multi-camera BEV detection. The most related work to ours is CaDDN [53]. We follow its architecture design except for the proposed occupancy learning module, and we replace its 2D backbone with lightweight DLA34 [74].

Since occupancy can serve as a general representation for scene perception and understanding. There is a burst of related works recently. SimpleOccupancy [16] adopts occupancy for better depth estimation. OccDepth [42] explores the occupancy in a stereo setting for semantic occupancy prediction. TPVFormer [22] employs sparse 3D occupancy labels from LiDAR as the supervision to obtain 3D features. OccNet [63] uses a decoder with various voxel based attention to reconstruct height information, accumulating occupancy labels from point cloud sequences. Different from them, we concentrate on 3D detection and takes a step farther to explore the effectiveness of occupancy in monocular 3D detection. Please note that our work focuses on the monocular setting, and extending the method to the multi-camera setup is a potential avenue for future researches.

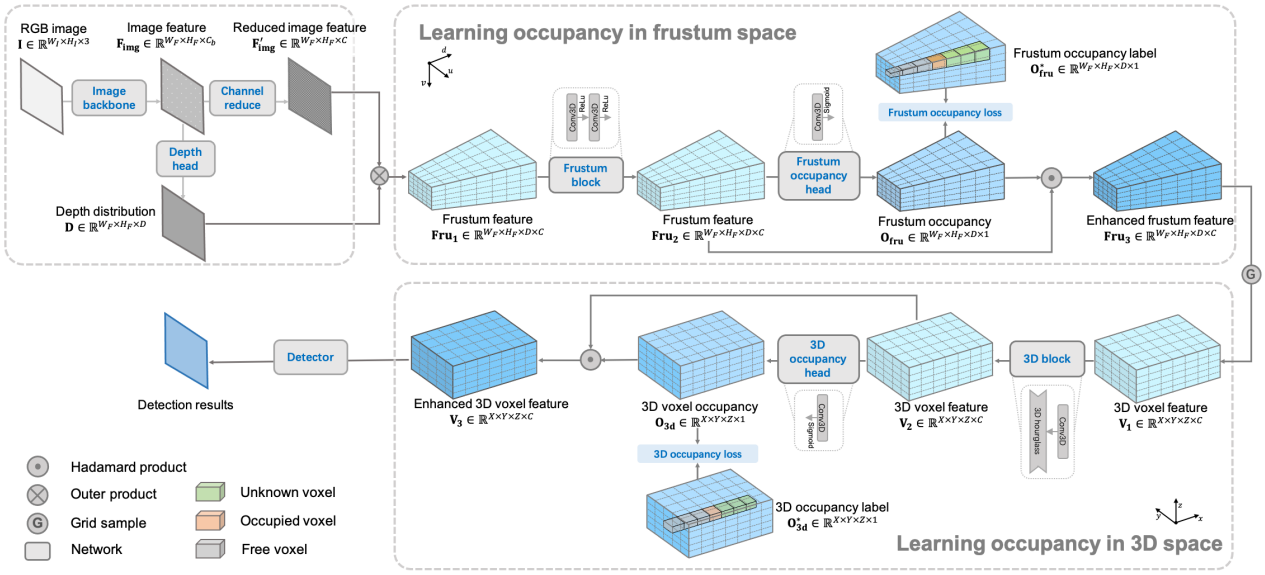

Figure 2. Network overview. Compared to previous works, our method employs two newly-proposed components for learning occupancy in frustum and 3D space. All network blocks in the proposed parts consist of vanilla 3D convolutions. Please refer to Section 3.1 for detailed feature passing description. For occupancy in frustum and 3D space and their network blocks, please see Section 3.2.1; For occupancy label generation, please see Section 3.2.2; For occupancy losses, please see Section 3.2.3; Best viewed in color with zoom in.

3. OccupancyM3D

3.1. Preliminary and Overview

Task Definition. We first describe the preliminary of this task and the method. At inference, monocular 3D detection takes only a single RGB image and outputs interested amodal 3D bounding boxes in the current scene. At the training stage, our method requires RGB images, 3D box labels annotated on LiDAR points and synchronized LiDAR data. It is worth noting that the system has been calibrated, and the camera intrinsics and extrinsics between the camera and LiDAR are available.

Network Overview. We present the network overview of our method in Figure 2. First, a single RGB image is fed into the DLA34 [74] backbone network to extract features. Then, we use these features to produce categorical depth distributions [53], which lifts 2D features to frustum space. After that, we employ the depth predictions and backbone features to generate frustum features. They are used for ***learning occupancy in frustum space***, and then are transformed to voxelized 3D features using grid-sampling. Such voxelized 3D features are employed to ***study occupancy in 3D space***. Occupancy learning in both frustum and 3D spaces can produce reasonable occupancy estimates that enhance the original features. The final enhanced voxelized 3D features are passed to the detection module to obtain final 3D detection results.

At the training stage, occupancy estimates are supervised by the generated occupancy labels in frustum and 3D space, respectively, using the proposed occupancy losses. We detail the occupancy learning in following sections.

3.2. Occupancy Learning

We consider a frustum voxel or regular 3D voxel to be occupied if it contains part of an object. We denote the resulting voxel states as **frustum occupancy** and **3D occupancy**, respectively. In this section, we introduce occupancy learning for monocular 3D detection. It is organized as four parts: ***occupancy in frustum/3D space***, ***occupancy labels***, ***occupancy losses***, and ***occupancy and depth***.

3.2.1 Occupancy in Frustum Space and 3D Space

After extracting backbone features, we employ a depth head to obtain dense category depth [53]. To save GPU memory, we use a convolution layer to reduce the number of feature channels, and the resulting feature is lifted to a frustum feature $\mathbf{Fru_1} \in \mathbb{R}^{W_F \times H_F \times D \times C}$ with the assistance of depth estimates. Then we extract frustum feature $\mathbf{Fru_2} \in \mathbb{R}^{W_F \times H_F \times D \times C}$ as follows:

$$\mathbf{Fru_2} = f_1(\mathbf{Fru_1}) \tag{1}$$

where f_1 denotes two 3D convolutions followed by ReLU activate functions. Then we use a 3D convolution layer f_2 and sigmoid function to obtain frustum occupancy $\mathbf{O_{fru}} \in \mathbb{R}^{W_F \times H_F \times D \times 1}$, which is supervised by corresponding labels $\mathbf{O^*_{fru}}$ as described in Section 3.2.2 and Section 3.2.3.

$$\mathbf{O_{fru}} = \mathrm{Sigmoid}(f_2(\mathbf{Fru_2})) \tag{2}$$

The frustum occupancy indicates the feature density in the frustum space, thus inherently can be employed to weight

original frustum features for achieving enhanced frustum feature $\mathbf{Fru_3} \in \mathbb{R}^{W_F \times H_F \times D \times C}$ as follows:

$$\mathbf{Fru_3} = \mathbf{O_{fru}} \odot \mathbf{Fru_2} \tag{3}$$

where $\odot$ denotes the Hadamard product (element-wise multiplication). The resulted frustum feature $\mathbf{Fru_3}$ is transformed to regular voxelized feature $\mathbf{V_1} \in \mathbb{R}^{X \times Y \times Z \times C}$ via grid-sampling [53]. The occupancy learning process is then repeated in the regular 3D space.

$$\mathbf{V_2} = f_3(\mathbf{V_1}); \mathbf{O_{3d}} = \text{Sigmoid}(f_4(\mathbf{V_2})); \mathbf{V_3} = \mathbf{O_{3d}} \odot \mathbf{V_2} \tag{4}$$

To better encode 3D features in the regular 3D space, we use a 3D hourglass-like design [6] in f_3, and f_4 is a 3D convolution. Finally, we have more informative 3D voxel feature $\mathbf{V_3} \in \mathbb{R}^{X \times Y \times Z \times C}$ for the detection module.

What is the rationale behind learning occupancy in both frustum and 3D space? Occupancy learning in both frustum and 3D space is beneficial because they have different nature. Frustum space has a resolution that depends on camera intrinsics and the downsample factor of the backbone network, while voxelized 3D space has a resolution that is decided by the pre-defined voxel size and detection range. Frustum voxels are irregular and vary in size based on the distance to the camera, which results in fine-grained voxels for objects that are closer and coarse-grained voxels for objects that are far away. In contrast, regular 3D voxels have the same size throughout the 3D space. On the other hand, frustum space is more fit to camera imagery, but objects in the frustum space cannot precisely represent the real 3D geometry. Thus the feature extraction and occupancy in frustum space have distortion for objects/scenes. Therefore, occupancy learning in both frustum and 3D space is complementary, and can result in more informative representations and features.

3.2.2 Occupancy Labels

Given a set of sparse LiDAR points $\mathbf{P} \in \mathbb{R}^{N \times 3}$, where N is the number of points and 3 is the coordinate dimension (X, Y, Z), we generate corresponding occupancy labels. The process is illustrated in Figure 3, and is operated on every LiDAR point. More formally, we first define three space status and represent them with numbers: ***free:0***, ***occupied:1***, ***unknown:-1***. We then describe the occupancy label generation process in the frustum and 3D space, respectively.

Occupancy label in frustum space: Let us denote the frustum occupancy label as $\mathbf{O^*_{fru}} \in \mathbb{R}^{W_F \times H_F \times D}$, where W_F and H_F are feature resolution, and D is the depth category. We first project LiDAR points onto the image plane to form a category depth index map. Each valid projected point has a category depth index, while invalid points (no

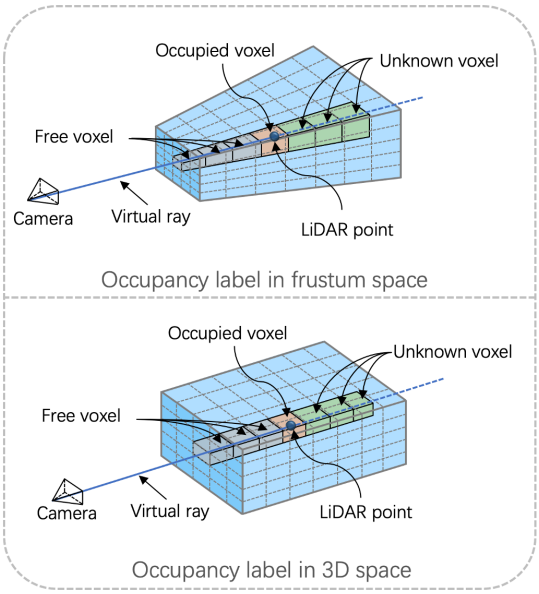

Figure 3. Occupancy label generation in frustum and 3D space. Best viewed in color with zoom in.

LiDAR points projections) are given negative indexes of -1. This index map is then downsampled to fit the feature resolution, resulting in $\mathbf{Ind} \in \mathbb{R}^{W_F \times H_F}$. Benefit from the camera projection nature, we can easily distinguish the space status as follows:

$$\mathbf{O^*_{fru}}_{i,j,d} = \begin{cases} 1 & if\ \mathbf{Ind}_{i,j} > -1\ and\ d = \mathbf{Ind}_{i,j}, \\ 0 & if\ \mathbf{Ind}_{i,j} > -1\ and\ d < \mathbf{Ind}_{i,j}, \\ -1 & otherwise. \end{cases} \tag{5}$$

where $i, j, d \in W_F, H_F, D$. Note that we do not consider unknown voxels in both the occupancy labels and occupancy losses. We use the known voxels, *i.e.*, the free and occupied voxels, to perform occupancy learning.

Occupancy label in 3D space: We denote $\mathbf{O^*_{3d}} \in \mathbb{R}^{X \times Y \times Z}$ as the 3D occupancy label, where X, Y, Z are determined by the pre-defined voxel size and detection range. We voxelize LiDAR points within the grid and set the voxels containing points to 1, and those without points to -1. In this way, occupied voxels can be easily achieved. To obtain the free voxels, we utilize ray tracing from each LiDAR point to the camera, where intersected voxels are set as free, filled by 0. We summarize the occupancy label in 3D space as follows:

$$\mathbf{O^*_{3d}}_{x,y,z} = \begin{cases} 1 & if\ \mathbf{Vol_{3d}}_{x,y,z} > 0, \\ 0 & if\ \mathrm{R}(\mathbf{O^*_{3d}}_{x,y,z}) \cap Ray_{point \to cam} \\ -1 & otherwise. \end{cases} \tag{6}$$

where $x, y, z \in X, Y, Z$. In this equation, $\mathbf{Vol_{3d}}$ denotes the voxelized grid. $\mathbf{Vol_{3d}} > 0$ when it is occu-

Approaches	Venue	Input	AP_{BEV} (IoU=0.7)$\|_{R_{40}}$			AP_{3D} (IoU=0.7)$\|_{R_{40}}$		
			Easy	Moderate	Hard	Easy	Moderate	Hard
Kinematic3D [3]	*ECCV20*	Video	26.69	17.52	13.10	19.07	12.72	9.17
DfM [66]	*ECCV22*	Video	31.71	22.89	19.97	22.94	16.82	14.65
Monodle [39]	*CVPR21*	Image	24.79	18.89	16.00	17.23	12.26	10.29
MonoFlex [76]	*CVPR21*	Image	28.23	19.75	16.89	19.94	13.89	12.07
CaDDN [53]	*CVPR21*	Image	27.94	18.91	17.19	19.17	13.41	11.46
GUP Net [36]	*ICCV21*	Image	30.29	21.19	18.20	22.26	15.02	13.12
AutoShape [35]	*ICCV21*	Image	30.66	20.08	15.95	22.47	14.17	11.36
PCT [65]	*NeurIPS21*	Image	29.65	19.03	15.92	21.00	13.37	11.31
MonoDTR [21]	*CVPR22*	Image	28.59	20.38	17.14	21.99	15.39	12.73
MonoJSG [31]	*CVPR22*	Image	32.59	21.26	18.18	24.69	16.14	13.64
DEVIANT [25]	*ECCV22*	Image	29.65	20.44	17.43	21.88	14.46	11.89
DID-M3D [49]	*ECCV22*	Image	32.95	22.76	19.83	24.40	16.29	13.75
MonoDDE [29]	*CVPR22*	Image	**33.58**	**23.46**	**20.37**	**24.93**	**17.14**	**15.10**
Cube R-CNN [4]	*CVPR23*	Image	31.70	21.20	18.43	23.59	15.01	12.56
NeuROCS † [44]	*CVPR23*	Image	*37.27*	*24.49*	*20.89*	*29.89*	*18.94*	*15.90*
MonoUNI [23]	*NeurIPS23*	Image	33.28	23.05	19.39	24.75	16.73	13.49
OccupancyM3D	-	Image	**35.38**	**24.18**	**21.37**	**25.55**	**17.02**	**14.79**

Table 1. Comparisons on KITTI *test* set for *Car* category. The red refers to the highest result and blue is the second-highest result. Our method outperforms other methods including monocular and video-based methods.

pied by LiDAR points. $\mathrm{R}(\cdot)$ refers to the voxel range and $\mathrm{R}(\mathbf{O_{3d}}_{i,j,d}) \cap Ray_{point \to cam}$ denotes that the voxel at index i, j, d intersects with a ray from LiDAR points to the camera. In this way, 3D occupancy labels are generated.

When generating voxel-based occupancy labels, there is a quantization error that arises due to the discretization process. A smaller voxel size results in lower quantization error, providing more fine-grained and accurate information. However, it requires more computation and GPU resources.

3.2.3 Occupancy Losses

We use generated occupancy labels $\mathbf{O^*_{fru}}$ and $\mathbf{O^*_{3d}}$ to supervised the predicted occupancy $\mathbf{O_{fru}}$ and $\mathbf{O_{3d}}$, respectively. We regard occupancy prediction as a simple classification problem and use focal loss [32] as the classification loss. Only valid voxels, *i.e.*, free and occupied voxels, contribute to the loss, and unknown voxels are ignored. We first obtain valid masks $\mathbf{M_{fru}} \in \mathbb{R}^{W_F \times H_F \times D}$ and $\mathbf{M_{3d}} \in \mathbb{R}^{X \times Y \times Z}$. $\mathbf{M_{fru}} = true$ if $\mathbf{O^*_{fru}} > -1$ otherwise $false$. $\mathbf{M_{3d}}$ is obtained using the similar way.

Therefore, the occupancy loss in frustum space is:

$$\mathcal{L}_{fru} = \mathrm{FL}(\mathbf{O^*_{fru}}[\mathbf{M_{fru}}], \mathbf{O_{fru}}[\mathbf{M_{fru}}]) \tag{7}$$

where $\mathrm{FL}(\cdot)$ refers to focal loss and $[\cdot]$ is the selection via mask. Similarly, we can obtain 3D occupancy loss as follows:

$$\mathcal{L}_{3d} = \mathrm{FL}(\mathbf{O^*_{3d}}[\mathbf{M_{3d}}], \mathbf{O_{3d}}[\mathbf{M_{3d}}]) \tag{8}$$

The final occupancy loss is their sum:

$$\mathcal{L}_{occupancy} = \mathcal{L}_{fru} + \mathcal{L}_{3d} \tag{9}$$

The occupancy loss allows the network to learn informative and discriminative features and representations, thus benefit downstream tasks. Therefore, the final loss of the network is:

$$\mathcal{L} = \mathcal{L}_{org} + \lambda \mathcal{L}_{occupancy} \tag{10}$$

where $\mathcal{L}_{org}$ denotes the original detection and depth losses in CaDDN [53] and λ is the occupancy loss weighting factor, which is set to 1 by default.

3.2.4 Occupancy and Depth

Occupancy has some similarities with 2D depth map, especially the frustum occupancy. They both can represent object geometry surface in the space. However, depth map is two-dimensional while occupancy is three-dimensional. Occupancy is beyond the depth and can base on it. It is able to express dense features of objects but not only the surface. For unknown space due to occlusion, the occupancy can infer reasonable results. Moreover, learning occupancy in frustum and 3D space allows the network to study more informative features under a higher dimension compared to 2D space.

Occupancy and depth are not mutually exclusive representations. In fact, they complement each other in the 3D object detection task. Without depth, the network has to deal with a large search space, making it challenging to learn reasonable occupancy features. Incorporating depth estimation provides the network with a good starting point and facilitates learning the occupancy features. Therefore, it is recommended to utilize both depth and occupancy in-

Methods	Venue	3D mAP / mAPH (IoU = 0.7)				3D mAP / mAPH (IoU = 0.5)			
		Overall	0 - 30m	30 - 50m	50m - ∞	Overall	0 - 30m	30 - 50m	50m - ∞
Comparison on LEVEL 1									
PatchNet[38]	*ECCV20*	0.39/0.37	1.67/1.63	0.13/0.12	0.03/0.03	2.92/2.74	10.03/9.75	1.09/0.96	0.23/0.18
CaDDN [53]	*CVPR21*	**5.03/4.99**	**14.54/14.43**	**1.47/1.45**	**0.10/0.10**	17.54/17.31	**45.00/44.46**	9.24/9.11	0.64/0.62
PCT [65]	*NeurIPS21*	0.89/0.88	3.18/3.15	0.27/0.27	0.07/0.07	4.20/4.15	14.70/14.54	1.78/1.75	0.39/0.39
MonoJSG [31]	*CVPR22*	0.97/0.95	4.65/4.59	0.55/0.53	0.10/0.09	5.65/5.47	20.86/20.26	3.91/3.79	0.97/0.92
DEVIANT [25]	*ECCV22*	2.69/2.67	6.95/6.90	0.99/0.98	0.02/0.02	10.98/10.89	26.85/26.64	5.13/5.08	0.18/0.18
DID-M3D [49]	*ECCV22*	-/-	-/-	-/-	-/-	**20.66/20.47**	40.92/40.60	**15.63/15.48**	**5.35/5.24**
NeuROCS † [44]	*CVPR23*	2.44/2.43	6.35/6.31	0.97/0.97	0.04/0.04	-/-	-/-	-/-	-/-
MonoUNI [23]	*NeurIPS23*	3.20/3.16	8.61/8.50	0.87/0.86	0.13/0.12	10.98/10.73	26.63/26.30	4.04/3.98	0.57/0.55
OccupancyM3D	-	**10.61/10.53**	**29.18/28.96**	**4.49/4.46**	**0.41/0.40**	**28.99/28.66**	**61.24/60.63**	**23.25/23.00**	**3.65/3.59**
Comparison on LEVEL 2									
PatchNet[38]	*ECCV20*	0.38/0.36	1.67/1.63	0.13/0.11	0.03/0.03	2.42/2.28	10.01/9.73	1.07/0.94	0.22/0.16
CaDDN [53]	*CVPR21*	**4.49/4.45**	**14.50/14.38**	**1.42/1.41**	**0.09/0.09**	16.51/16.28	**44.87/44.33**	8.99/8.86	0.58/0.55
PCT [65]	*NeurIPS21*	0.66/0.66	3.18/3.15	0.27/0.26	0.07/0.07	4.03/3.99	14.67/14.51	1.74/1.71	0.36/0.35
MonoJSG [31]	*CVPR22*	0.91/0.89	4.64/4.65	0.55/0.53	0.09/0.09	5.34/5.17	20.79/20.19	3.79/3.67	0.85/0.82
DEVIANT [25]	*ECCV22*	2.52/2.50	6.93/6.87	0.95/0.94	0.02/0.02	10.29/10.20	26.75/26.54	4.95/4.90	0.16/0.16
DID-M3D[49]	*ECCV22*	-/-	-/-	-/-	-/-	**19.37/19.19**	40.77/40.46	**15.18/15.04**	**4.69/4.59**
NeuROCS † [44]	*CVPR23*	2.29/2.28	6.32/6.29	0.94/0.93	0.03/0.03	-/-	-/-	-/-	-/-
MonoUNI [23]	*NeurIPS23*	3.04/3.00	8.59/8.48	0.85/0.84	0.12/0.12	10.38/10.24	26.57/26.24	3.95/3.89	0.53/0.51
OccupancyM3D	-	**10.02/9.94**	**28.38/28.17**	**4.38/4.34**	**0.36/0.36**	**27.21/26.90**	**61.09/60.49**	**22.59/22.34**	**3.18/3.13**

Table 2. Results on WaymoOD *val* set for *Vehicle* category. The red refers to the highest result and blue is the second-highest result. Our method outperforms other methods by significant margins on most metrics. Note that our method has the detection range limitation of $[2, 59.6](meters)$, while the perspective-view based method DID-M3D [49] does not have this shortcoming. Thus our method performs worse for objects within $[50m, \infty]$ under IoU=0.5 criterion.

formation to achieve better representations and features for monocular 3D detection.

4. Experiments

4.1. Implementation Details

We employ PyTorch [48] for implementation. The network is trained on 4 NVIDIA 3080Ti (12G) GPUs, with a total batch size of 8 for 80 epochs. We use Adam [24] optimizer with initial learning rate 0.001 and employ the one-cycle learning rate policy [61]. We use pre-trained DLA34 [74] backbone from [47]. We employ flip and crop data augmentation [36]. For KITTI [17], we fix the input image to 1280×384, detection range $[2, 46.8] \times [-30.08, 30.08] \times [-3.0, 1.0](meter)$ for x, y, z axes under the LiDAR coordinate system, respectively. We use voxel size $[0.16, 0.16, 0.16](meter)$. For Waymo [62], we downsample the input RGB image from 1920×1280 to 960×640 to meet GPU memory. We use detection range $[2, 59.6] \times [-25.6, 25.6] \times [-2.0, 2.0](meter)$ for x, y, z axes due to the larger depth domain on Waymo. We use voxel size $[0.16, 0.16, 0.16](meter)$.

4.2. Datasets and Metrics

Following the fashion in previous works, we conduct experiments on competitive KITTI and Waymo open datasets.

KITTI: KITTI [17] is a widely employed benchmark for autonomous driving. KITTI3D object dataset consists of 7,481 training samples and 7,518 testing samples, where labels on *test* set keep secret and the final performance is evaluated on KITTI official website. To conduct ablations, the training samples are further divided into a *train* set and a *val* set [10]. They individually contain 3,512 and 3,769 samples, respectively. KITTI has three categories: *Car*, *Pedestrian*, and *Cyclist*. According to difficulties (2D box height, occlusion and truncation levels), KITTI divides objects into *Easy*, *Moderate*, and *Hard*. Following common practice [35, 53, 60], we use $AP_{BEV}|_{R_{40}}$ and $AP_{3D}|_{R_{40}}$ under IoU threshold of 0.7 to evaluate the performance.

Waymo: Waymo open dataset (WaymoOD) [62] is a large modern dataset for autonomous driving. It has 798 sequences for training and 202 sequences for validation. Following previous works [49, 53], we use the front camera of the multi-camera rig and provide performance comparison on *val* set for the vehicle category. To make fair comparisons, we use one third samples of training sequences to train the network due to the large-scale and high frame rate

Approaches	Venue	Input	Pedestrian AP_{BEV}/AP_{3D}(IoU=0.5)$\|_{R_{40}}$			Cyclist AP_{BEV}/AP_{3D}(IoU=0.5)$\|_{R_{40}}$		
			Easy	Moderate	Hard	Easy	Moderate	Hard
DfM [66]	*ECCV22*	Video	-/13.70	-/8.71	-/7.32	-/**8.98**	-/**5.75**	-/**4.88**
Monodle [39]	*CVPR21*	Image	10.73/9.64	6.96/6.55	6.20/5.44	5.34/4.59	3.28/2.66	2.83/2.45
DDMP-3D [64]	*CVPR21*	Image	5.53/4.93	4.02/3.55	3.36/3.01	4.92/4.18	3.14/2.50	2.44/2.32
MonoRUn [7]	*CVPR21*	Image	11.70/10.88	7.59/6.78	6.34/5.83	1.14/1.01	0.73/0.61	0.66/0.48
MonoEF [79]	*CVPR21*	Image	4.61/4.27	3.05/2.79	2.85/2.21	2.36/1.80	1.18/0.92	1.15/0.71
MonoFlex [76]	*CVPR21*	Image	10.36/9.43	7.36/6.31	6.29/5.26	4.41/4.17	2.67/2.35	2.50/2.04
CaDDN [53]	*CVPR21*	Image	14.72/12.87	9.41/8.14	8.17/6.76	**9.67**/7.00	**5.38**/3.41	**4.75**/3.30
GUP Net [36]	*ICCV21*	Image	**15.62**/**14.95**	10.37/**9.76**	8.79/**8.41**	6.94/5.58	3.85/3.21	3.64/2.66
AutoShape [35]	*ICCV21*	Image	-/5.46	-/3.74	-/3.03	-/5.99	-/3.06	-/2.70
MonoCon [33]	*AAAI22*	Image	-/13.10	-/8.41	-/6.94	-/2.80	-/1.92	-/1.55
HomoLoss [18]	*CVPR22*	Image	13.26/11.87	8.81/7.66	7.41/6.82	6.81/5.48	4.09/3.50	3.78/2.99
MonoJSG [31]	*CVPR22*	Image	-/11.02	-/7.49	-/6.41	-/5.45	-/3.21	-/2.57
DEVIANT [25]	*ECCV22*	Image	14.49/13.43	9.77/8.65	8.28/7.69	6.42/5.05	3.97/3.13	3.51/2.59
Cube R-CNN [4]	*CVPR23*	Image	11.67/11.17	7.65/6.95	6.60/5.87	5.01/3.65	3.35/2.67	3.23/2.28
MonoUNI [23]	*NeurIPS23*	Image	**16.54**/**15.78**	**10.90**/**10.34**	**9.17**/**8.74**	8.25/7.34	**5.03**/**4.28**	**4.50**/**3.78**
OccupancyM3D	-	Image	**16.54**/14.68	**10.65**/9.15	**9.16**/7.80	**8.58**/**7.37**	4.35/3.56	3.55/2.84

Table 3. Comparisons on KITTI *test* set for *Pedestrian* and *Cyclist* categories. The red refers to the highest result and blue is the second-highest result. Our method achieves new state-of-the-art results.

of this dataset. Waymo divides objects to LEVEL 1 and LEVEL 2 according to the LiDAR point number within objects. For metrics, we employ the official mAP and mAPH under LEVEL 1 and LEVEL 2.

4.3. Results on KITTI and Waymo Datasets

We provide the performance comparisons on KITTI and WaymoOD. Table 1 shows the results of *Car* category on KITTI *test* set. Our method outperforms other methods including video-based methods. For example, the proposed method exceeds CaDDN [53] under all metrics, *e.g.*, 25.55/17.02/14.79 *vs.* 19.17/13.41/11.46 AP_{3D}. Please note that NeuROCS [44] requires instance masks for training and does not report results of Pedestrian and Cyclist on KITTI test set. Such irregularly shaped objects are challenging to NOCS prediction. Our method outperforms DID-M3D [49] by a margin of 2.43/1.42/1.54 AP_{BEV}, and performs better that Cube R-CNN [4] and MonoUNI [23]. When compared to the video-based method DfM [66], OccupancyM3D also shows better performance, *e.g.*, 24.18 *vs.* 22.89 AP_{BEV} under the moderate setting. In Table 3, we provide comparisons on other categories, namely, *Pedestrian* and *Cyclist*. The results demonstrate the superiority of our method on different categories. Concerning the overall performance on all categories, our method achieves a state-of-the-art on KITTI *test* set for monocular 3D detection.

We also evaluate our method on Waymo open dataset (WaymoOD) [62] and obtain promising results. As shown in Table 2, our method surpasses other methods with a significant margin. For example, under LEVEL 1 setting, OccupancyM3D outperforms CaDDN [53] by 5.58/5.54 mAP/mAPH (10.61/10.53 *vs.* 5.03/4.99) and 11.55/11.35 mAP/mAPH (28.99/28.66 *vs.* 17.54/17.31) with IoU 0.7 and 0.5 criterions, respectively. Compared to DID-M3D [49], under IoU criterion 0.5, our method outperforms it by 8.33/8.19 mAP/mAPH (28.99/28.66 *vs.* 20.66/20.47) and 7.84/7.71 mAP/mAPH (27.21/26.90 *vs.* 19.37/19.19) with LEVEL 1 and 2 settings, respectively. This success can be attributed to the fact that occupancy learning benefits from the diverse scenes present in large datasets. In other words, large datasets especially favor the proposed occupancy learning method. Interestingly, concerning objects within $[50m, \infty]$, our method performs worse than DID-M3D [49]. It is because our method is voxel-based, which has a detection range limitation ($[2, 59.6](meters)$ in our method). By contrast, DID-M3D is a perspective-based method, indicating that it does not have this limitation and can detect more faraway objects. We encourage future works to address this range limitation in our method.

4.4. Ablations

Following common practice in previous works, we perform ablations on KITTI *val* set to validate the effectiveness of each component. We compare the performance on *Car* category under IoU criterion 0.7.

We provide the main ablation in Table 4. It can be easily seen that occupancy learning significantly benefit the final detection performance. When enforcing occupancy learning in frustum space, the detection AP_{3D} increases from 21.04/17.05/15.01 to 24.69/17.79/15.16 (Exp. (a)→(b)). On the other hand, when enforcing occupancy learning in 3D space, the detection AP_{3D} is boosted to

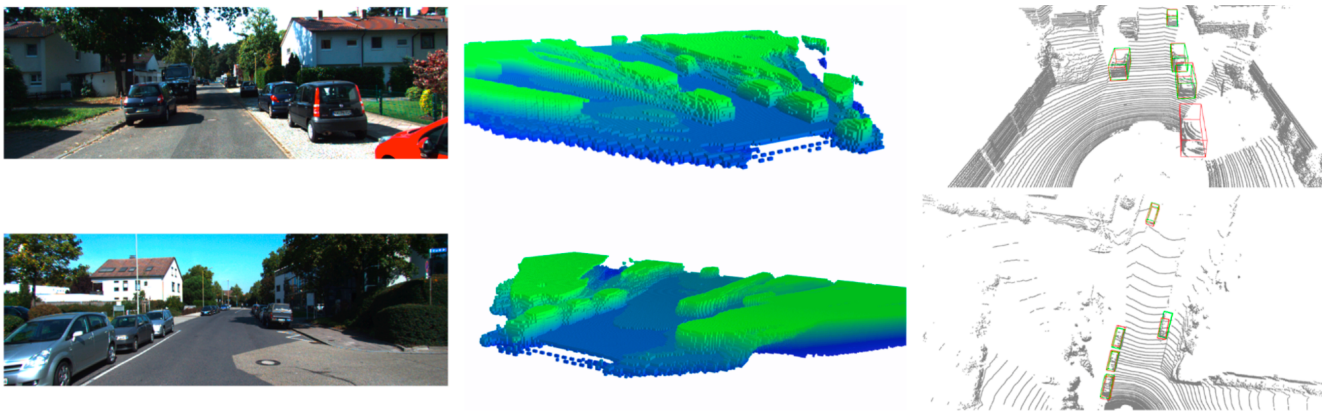

Figure 4. Qualitative results of occupancy predictions and 3D detections on KITTI *val* set. In 3D detections, Red boxes are our results and Green boxes denote ground-truths. The LiDAR point clouds in 3D detections are used only for visualization. We can see that the proposed method generates reasonable occupancy predictions for the current scene, which benefits downstream monocular 3D detection task. However, our method may fail to estimate heavily occluded objects (see right objects of the bottom picture). Most monocular 3D detection failure cases stem from the poor occupancy estimation. Best viewed in color with zoom in.

E.	OL-FS	OL-3DS	AP_{BEV}/AP_{3D} (IoU=0.7)$\|_{R_{40}}$ Easy	Moderate	Hard
(a)			30.32/21.04	24.58/17.05	22.02/15.01
(b)	✓		35.46/24.69	25.46/17.79	22.96/15.16
(c)		✓	33.15/24.64	25.45/18.88	22.68/16.38
(d)	✓	✓	**35.72/26.87**	**26.60/19.96**	**23.68/17.15**

Table 4. Main ablation. "E." in the table is the experiment ID; "OL-FS" refers to occupancy learning in frustum space; "OL-3DS" denotes occupancy learning in 3D space.

24.64/18.88/16.38 (Exp. (c)). Finally, the model obtains 5.83/2.91/2.14 AP_{3D} gains (Exp. (a)→(d)) by employing occupancy learning in both frustum and 3D space. This main ablation demonstrates the effectiveness of our method.

4.5. Qualitative Results

We present qualitative results of occupancy predictions and 3D detections in Figure 4. Our method can predict reasonable occupancy for the current scene, especially for foreground objects. This indicates the potential of occupancy learning in downstream tasks. Nevertheless, we can see that the occupancy estimates are not very accurate for heavily occluded objects (see right objects of the bottom picture), which leaves room for improvement in future works.

5. Limitation and Future Work

One significant drawback of this work is the voxel size limitation. Large voxels in explicit voxel-based representation can reduce computation overhead and GPU memory, but at the cost of failing to precisely describe the 3D geometry of the scene due to quantization errors. Conversely, smaller voxel sizes are able to express fine-grained 3D geometry but come at the significant expense of increased computation overhead and GPU memory usage. On the other hand, the voxel-based method has limited detection ranges and most detection failure cases stem from poor occupancy estimation. This work mainly focuses on occupancy learning in the monocular 3D detection task, and the exploration of its application in more downstream tasks such as multi-camera detection/segmentation and indoor 3D detection is less explored. We believe that it is an interesting and promising topic and encourage future works to alleviate the above limitations to advance the self-driving community.

6. Conclusion

In this paper, we propose to learn occupancy for monocular 3D detection, to obtain more discriminative and informative 3D features. To perform occupancy learning, we design occupancy labels by using synchronized raw sparse LiDAR point clouds and introduce corresponding occupancy losses. Ablations verify the effectiveness of each proposed component. To the best of our knowledge, this is the first work that introduces occupancy learning to monocular 3D detection. We conduct experiments on the challenging KITTI and Waymo open datasets. The results demonstrate that the proposed method achieves new state-of-the-art results and outperforms other methods by a large margin.

Acknowledgments

This work was supported in part by The National Nature Science Foundation of China (Grant Nos: 62273302, 62036009, 61936006, 62303406) and in part by Yongjiang Talent Introduction Programme (Grant No: 2023A-197-G, 2023A-194-G).

References

[1] Alex Bewley, Pei Sun, Thomas Mensink, Dragomir Anguelov, and Cristian Sminchisescu. Range conditioned dilated convolutions for scale invariant 3d object detection. *arXiv preprint arXiv:2005.09927*, 2020. 2

[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. 2

[3] Garrick Brazil, Gerard Pons-Moll, Xiaoming Liu, and Bernt Schiele. Kinematic 3d object detection in monocular video. In *European Conference on Computer Vision*, pages 135–152. Springer, 2020. 5

[4] Garrick Brazil, Abhinav Kumar, Julian Straub, Nikhila Ravi, Justin Johnson, and Georgia Gkioxari. Omni3d: A large benchmark and model for 3d object detection in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13154–13164, 2023. 5, 7

[5] Yuning Chai, Pei Sun, Jiquan Ngiam, Weiyue Wang, Benjamin Caine, Vijay Vasudevan, Xiao Zhang, and Dragomir Anguelov. To the point: Efficient 3d object detection in the range image with graph convolution kernels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2021. 2

[6] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5418, 2018. 4

[7] Hansheng Chen, Yuyao Huang, Wei Tian, Zhong Gao, and Lu Xiong. Monorun: Monocular 3d object detection by reconstruction and uncertainty propagation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10379–10388, 2021. 1, 7

[8] Qi Chen, Lin Sun, Ernest Cheung, and Alan L Yuille. Every view counts: Cross-view consistency in 3d object detection with hybrid-cylindrical-spherical voxelization. *Advances in Neural Information Processing Systems*, 33:21224–21235, 2020. 2

[9] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2156, 2016. 2

[10] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals using stereo imagery for accurate object class detection. *IEEE transactions on pattern analysis and machine intelligence*, 40(5):1259–1272, 2017. 6

[11] Yi-Nan Chen, Hang Dai, and Yong Ding. Pseudo-stereo for monocular 3d object detection in autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 887–897, 2022. 2

[12] Zhiyu Chong, Xinzhu Ma, Hong Zhang, Yuxin Yue, Haojie Li, Zhihui Wang, and Wanli Ouyang. Monodistill: Learning spatial features for monocular 3d object detection. *arXiv preprint arXiv:2201.10830*, 2022. 2

[13] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1201–1209, 2021. 2

[14] 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*, pages 11672–11681, 2020. 2

[15] Lue Fan, Xuan Xiong, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Rangedet: In defense of range view for lidar-based 3d object detection. *arXiv preprint arXiv:2103.10039*, 2021. 2

[16] Wanshui Gan, Ningkai Mo, Hongbin Xu, and Naoto Yokoya. A simple attempt for 3d occupancy estimation in autonomous driving. *arXiv preprint arXiv:2303.10076*, 2023. 2

[17] 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

[18] Jiaqi Gu, Bojian Wu, Lubin Fan, Jianqiang Huang, Shen Cao, Zhiyu Xiang, and Xian-Sheng Hua. Homography loss for monocular 3d object detection. *arXiv preprint arXiv:2204.00754*, 2022. 7

[19] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3d object detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11873–11882, 2020. 2

[20] Junjie Huang, Guan Huang, Zheng Zhu, and Dalong Du. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. *arXiv preprint arXiv:2112.11790*, 2021. 2

[21] Kuan-Chih Huang, Tsung-Han Wu, Hung-Ting Su, and Winston H Hsu. Monodtr: Monocular 3d object detection with depth-aware transformer. *arXiv preprint arXiv:2203.10981*, 2022. 5

[22] Yuanhui Huang, Wenzhao Zheng, Yunpeng Zhang, Jie Zhou, and Jiwen Lu. Tri-perspective view for vision-based 3d semantic occupancy prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9223–9232, 2023. 2

[23] Jinrang Jia, Zhenjia Li, and Yifeng Shi. Monouni: A unified vehicle and infrastructure-side monocular 3d object detection network with sufficient depth clues. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 5, 6, 7

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[25] Abhinav Kumar, Garrick Brazil, Enrique Corona, Armin Parchami, and Xiaoming Liu. Deviant: Depth equivariant network for monocular 3d object detection. In *European Conference on Computer Vision (ECCV)*, 2022. 5, 6, 7

[26] 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. 2

[27] Peixuan Li, Huaici Zhao, Pengfei Liu, and Feidao Cao. Rtm3d: Real-time monocular 3d detection from object keypoints for autonomous driving. *arXiv preprint arXiv:2001.03343*, 2020. 1, 2

[28] Yinhao Li, Zheng Ge, Guanyi Yu, Jinrong Yang, Zengran Wang, Yukang Shi, Jianjian Sun, and Zeming Li. Bevdepth: Acquisition of reliable depth for multi-view 3d object detection. *arXiv preprint arXiv:2206.10092*, 2022. 2

[29] Zhuoling Li, Zhan Qu, Yang Zhou, Jianzhuang Liu, Haoqian Wang, and Lihui Jiang. Diversity matters: Fully exploiting depth clues for reliable monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2791–2800, 2022. 2, 5

[30] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IX*, pages 1–18. Springer, 2022. 2

[31] Qing Lian, Peiliang Li, and Xiaozhi Chen. Monojsg: Joint semantic and geometric cost volume for monocular 3d object detection. *arXiv preprint arXiv:2203.08563*, 2022. 2, 5, 6, 7

[32] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 5

[33] Xianpeng Liu, Nan Xue, and Tianfu Wu. Learning auxiliary monocular contexts helps monocular 3d object detection. *arXiv preprint arXiv:2112.04628*, 2021. 7

[34] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petr: Position embedding transformation for multi-view 3d object detection. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVII*, pages 531–548. Springer, 2022. 2

[35] Zongdai Liu, Dingfu Zhou, Feixiang Lu, Jin Fang, and Liangjun Zhang. Autoshape: Real-time shape-aware monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15641–15650, 2021. 2, 5, 6, 7

[36] Yan Lu, Xinzhu Ma, Lei Yang, Tianzhu Zhang, Yating Liu, Qi Chu, Junjie Yan, and Wanli Ouyang. Geometry uncertainty projection network for monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3111–3121, 2021. 1, 5, 6, 7

[37] Xinzhu Ma, Zhihui Wang, Haojie Li, Pengbo Zhang, Wanli Ouyang, and Xin Fan. Accurate monocular 3d object detection via color-embedded 3d reconstruction for autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6851–6860, 2019. 2

[38] Xinzhu Ma, Shinan Liu, Zhiyi Xia, Hongwen Zhang, Xingyu Zeng, and Wanli Ouyang. Rethinking pseudo-lidar representation. *arXiv preprint arXiv:2008.04582*, 2020. 1, 2, 6

[39] Xinzhu Ma, Yinmin Zhang, Dan Xu, Dongzhan Zhou, Shuai Yi, Haojie Li, and Wanli Ouyang. Delving into localization errors for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4721–4730, 2021. 1, 5, 7

[40] 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 Conference on Computer Vision and Pattern Recognition*, pages 2069–2078, 2019. 2

[41] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019. 2

[42] Ruihang Miao, Weizhou Liu, Mingrui Chen, Zheng Gong, Weixin Xu, Chen Hu, and Shuchang Zhou. Occdepth: A depth-aware method for 3d semantic scene completion. *arXiv preprint arXiv:2302.13540*, 2023. 2

[43] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 2

[44] Zhixiang Min, Bingbing Zhuang, Samuel Schulter, Buyu Liu, Enrique Dunn, and Manmohan Chandraker. Neurocs: Neural nocs supervision for monocular 3d object localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21404–21414, 2023. 5, 6, 7

[45] 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. 1, 2

[46] Jongyoun Noh, Sanghoon Lee, and Bumsub Ham. Hvpr: Hybrid voxel-point representation for single-stage 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14605–14614, 2021. 2

[47] Dennis Park, Rares Ambrus, Vitor Guizilini, Jie Li, and Adrien Gaidon. Is pseudo-lidar needed for monocular 3d object detection? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3142–3152, 2021. 2, 6

[48] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037, 2019. 6

[49] Liang Peng, Xiaopei Wu, Zheng Yang, Haifeng Liu, and Deng Cai. Did-m3d: Decoupling instance depth for monocular 3d object detection. In *European Conference on Computer Vision*, 2022. 1, 2, 5, 6, 7

[50] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2

[51] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017. 2

[52] 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*, pages 918–927, 2018. 1, 2

[53] Cody Reading, Ali Harakeh, Julia Chae, and Steven L Waslander. Categorical depth distribution network for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8555–8564, 2021. 1, 2, 3, 4, 5, 6, 7

[54] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. *arXiv preprint arXiv:1811.08188*, 2018. 2

[55] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–779, 2019. 2

[56] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. *arXiv preprint arXiv:1907.03670*, 2019. 2

[57] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020. 2

[58] 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

[59] Weijing Shi and Raj Rajkumar. Point-gnn: Graph neural network for 3d object detection in a point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1711–1719, 2020. 2

[60] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1991–1999, 2019. 6

[61] Leslie N Smith. A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay. *arXiv preprint arXiv:1803.09820*, 2018. 6

[62] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2446–2454, 2020. 6, 7

[63] Wenwen Tong, Chonghao Sima, Tai Wang, Li Chen, Silei Wu, Hanming Deng, Yi Gu, Lewei Lu, Ping Luo, Dahua Lin, et al. Scene as occupancy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8406–8415, 2023. 2

[64] Li Wang, Liang Du, Xiaoqing Ye, Yanwei Fu, Guodong Guo, Xiangyang Xue, Jianfeng Feng, and Li Zhang. Depth-conditioned dynamic message propagation for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 454–463, 2021. 2, 7

[65] Li Wang, Li Zhang, Yi Zhu, Zhi Zhang, Tong He, Mu Li, and Xiangyang Xue. Progressive coordinate transforms for monocular 3d object detection. *Advances in Neural Information Processing Systems*, 34, 2021. 5, 6

[66] Tai Wang, Jiangmiao Pang, and Dahua Lin. Monocular 3d object detection with depth from motion. *arXiv preprint arXiv:2207.12988*, 2022. 5, 7

[67] 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

[68] Hai Wu, Chenglu Wen, Wei Li, Xin Li, Ruigang Yang, and Cheng Wang. Transformation-equivariant 3d object detection for autonomous driving. *arXiv preprint arXiv:2211.11962*, 2022. 2

[69] Xiaopei Wu, Liang Peng, Honghui Yang, Liang Xie, Chenxi Huang, Chengqi Deng, Haifeng Liu, and Deng Cai. Sparse fuse dense: Towards high quality 3d detection with depth completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5418–5427, 2022. 2

[70] Yan Yan, Yuxing Mao, and Bo Li. SECOND: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 2

[71] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. STD: Sparse-to-dense 3D object detector for point cloud. In *ICCV*, pages 1951–1960, 2019. 2

[72] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3DSSD: Point-based 3D single stage object detector. In *CVPR*, pages 11040–11048, 2020. 2

[73] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11784–11793, 2021. 2

[74] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2403–2412, 2018. 2, 3, 6

[75] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 2

[76] Yunpeng Zhang, Jiwen Lu, and Jie Zhou. Objects are different: Flexible monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3289–3298, 2021. 2, 5, 7

[77] Wu Zheng, Weiliang Tang, Li Jiang, and Chi-Wing Fu. Se-ssd: Self-ensembling single-stage object detector from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14494–14503, 2021. 2

[78] 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, 2

[79] Yunsong Zhou, Yuan He, Hongzi Zhu, Cheng Wang, Hongyang Li, and Qinhong Jiang. Monocular 3d object detection: An extrinsic parameter free approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7556–7566, 2021. 7