MonoDTR: Monocular 3D Object Detection with Depth-Aware Transformer

Kuan-Chih Huang[1]   Tsung-Han Wu[1]   Hung-Ting Su[1]   Winston H. Hsu[1,2]

[1] National Taiwan University   [2] Mobile Drive Technology

Abstract

Monocular 3D object detection is an important yet challenging task in autonomous driving. Some existing methods leverage depth information from an off-the-shelf depth estimator to assist 3D detection, but suffer from the additional computational burden and achieve limited performance caused by inaccurate depth priors. To alleviate this, we propose MonoDTR, a novel end-to-end depth-aware transformer network for monocular 3D object detection. It mainly consists of two components: (1) the Depth-Aware Feature Enhancement (DFE) module that implicitly learns depth-aware features with auxiliary supervision without requiring extra computation, and (2) the Depth-Aware Transformer (DTR) module that globally integrates context- and depth-aware features. Moreover, different from conventional pixel-wise positional encodings, we introduce a novel depth positional encoding (DPE) to inject depth positional hints into transformers. Our proposed depth-aware modules can be easily plugged into existing image-only monocular 3D object detectors to improve the performance. Extensive experiments on the KITTI dataset demonstrate that our approach outperforms previous state-of-the-art monocular-based methods and achieves real-time detection. Code is available at https://github.com/kuanchihhuang/MonoDTR.

1. Introduction

Three-dimensional (3D) object detection is a fundamental problem and enables various applications such as autonomous driving. Previous methods have achieved superior performance based on the accurate depth information from multiple sensors, such as LiDAR signal [16,22,39,40] or stereo matching [6, 23, 44, 52]. In order to lower the sensor costs, some image-only monocular 3D object detection methods [2, 7, 20, 31, 33, 50] have been proposed and made impressive progress relying on geometry constraints between 2D and 3D. However, the performance is still far from satisfactory without the aid of depth cues.

Recently, several works have tried to produce estimated depth from the pre-trained depth estimation models to assist monocular 3D object detection. Pseudo-LiDAR-based
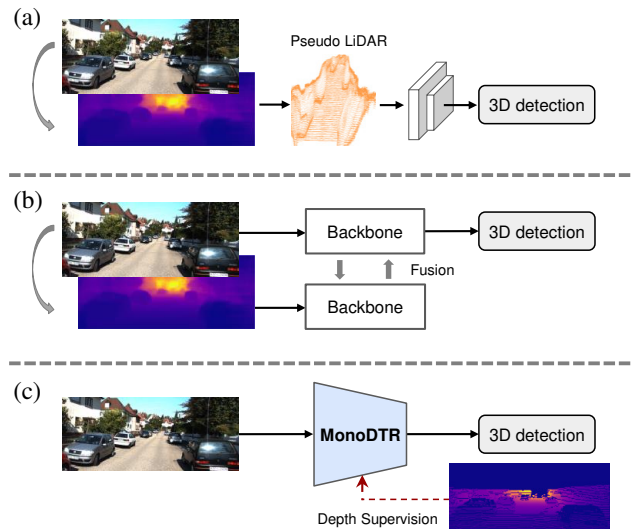

Figure 1. **Comparison of different depth-assisted monocular 3D object detection frameworks.** (a) Pseudo-LiDAR-based methods [31, 52, 53] lift images to 3D coordinate via monocular depth estimation, followed by a 3D LiDAR-based detector to recover object locations. (b) Fusion-based methods [10, 34, 48] extract features from images and estimated depth maps, then fuse them to predict objects. (c) Our MonoDTR learns depth-aware features via additional depth supervision and performs 3D object detection in an end-to-end manner. Note that our depth supervision is only leveraged in the training stage.

approaches [31, 52, 53] convert estimated depth maps into 3D point clouds to imitate LiDAR signals, followed by the existing LiDAR-based detector for 3D object detection (see Figure 1(a)). Some fusion-based approaches [10,34,48] apply several fusion strategies to combine features extracted from depths and images to detect objects (see Figure 1(b)). These methods, though better localize objects with the help of estimated depth, may suffer from the risk of learning 3D detection on inaccurate depth maps. Also, the additional computational cost of the depth estimator makes it impractical for real-world applications [32].

To address the above issues, we propose MonoDTR, a novel end-to-end depth-aware transformer network for monocular 3D object detection (see Figure 1(c)). A depth-aware feature enhancement (DFE) module is introduced to

learn depth-aware features with the auxiliary depth supervision, which avoids obtaining inaccurate depth priors from the pre-trained depth estimator. Furthermore, the DFE module is lightweight yet effective in assisting 3D object detection without constructing the complicated architecture to extract features from off-the-shelf depth maps. It significantly reduces computational time compared with previous depth-assisted methods [10, 31, 48] (see Table 1).

In addition, unlike previous fusion-based methods (*e.g.*, D⁴LCN [10] and DDMP-3D [48]) that apply carefully designed convolution kernels for context- and depth-aware features, we develop the first transformer-based fusion module to globally integrate the image and depth information. The transformer encoder-decoder structure [47] has been proven to capture long-range dependency effectively; thus, we apply it to model the relationship between context- and depth-aware features. To better represent the property of the 3D object, we utilize depth-aware features to replace the commonly used object queries [3, 18, 60] as input of the transformer decoder, which can provide more meaningful cues for 3D reasoning. Furthermore, we introduce a novel depth positional encoding (DPE) to involve depth-aware hints to the transformer, achieving better performance than conventional pixel-wise positional encodings.

We summarize our contributions as follows:

1. We propose a novel framework, MonoDTR, learning depth-aware features via auxiliary supervision to assist monocular 3D object detection, which avoids introducing high computational cost and inaccurate depth priors from using the off-the-shelf depth estimator.

2. We present the first depth-aware transformer module to integrate context- and depth-aware features efficiently. A novel depth positional encoding (DPE) is proposed to inject depth positional hints into the transformer.

3. Experimental results on the KITTI dataset show that our approach outperforms state-of-the-art monocular-based methods and achieves real-time detection. Furthermore, the proposed depth-aware modules can be easily plug-and-play in existing image-only frameworks to improve performance.

2. Related Work

Image-only monocular 3D object detection. Recently, several works only adopt a single image for 3D object detection [1, 27, 33, 37, 42, 43, 51, 56]. Due to the lack of depth information from images, these methods mainly rely on geometric consistency to predict objects. Deep3Dbox [33] solves orientation prediction by proposed novel *Multi-Bin* loss and enforces constraint between 2D and 3D boxes with geometric prior. M3D-RPN [1] generates 3D object proposals with 2D bounding box constraints and proposes

a depth-aware convolution to predict 3D objects. OFT-Net [38] introduces an orthographic feature transform to map image-based features into a 3D voxel space. Besides, MonoPair [7] explores spatial pair-wise relationship between objects to improve detection performance. M3DSSD [29] presents a two-step feature alignment approach to solve the feature mismatching problem. Furthermore, some works [24, 27, 32, 58] predict keypoints of the 3D bounding box as an intermediate task to recover the location of objects. However, the above purely monocular methods fail to accurately localize objects due to the lack of depth cues.

Depth-assisted monocular 3D object detection. To further improve the performance, many approaches propose using depth information to aid 3D object detection [10, 30, 31, 48, 53, 54]. Some prior works [31, 52, 53] transform image into pseudo-LiDAR representation by leveraging off-the-shelf depth estimator and calibration parameters, followed by the existing LiDAR-based 3D detector to predict objects, leading to progressive improvement. Patch-Net [30] reveals that the success of pseudo-LiDAR comes from the coordinate transformation and organizes it into the image representation, which can benefit from the powerful CNNs networks. D⁴LCN [10] and DDMP-3D [48] focus on developing the fusion-based approach between image and estimated depth with carefully designed convolutional networks. Besides, CaDDN [35] learns categorical depth distributions for each pixel to construct bird's-eye-view (BEV) representations and recovers bounding boxes from the BEV projection. However, most of the abovementioned methods directly using pre-trained depth estimators suffer from additional computational cost and only achieve limited performance caused by inaccurate depth priors.

Transformer. Transformer [47] was firstly introduced in sequential modeling and has considerable improvement in natural language processing (NLP) tasks. The self-attention mechanism is the core component in the transformer with its capability of capturing the long-range dependencies. Recently, transformer architecture has been successfully leveraged in the computer vision field, such as image classification [12] and human-object interaction [18]. In addition, DETR [3] proposes developing object detection with the transformer without relying on many hand-designed components used in traditional pipelines.

Though the transformer can perform well in most visual tasks, its usage in monocular 3D object detection has not been explored. In the image-based 3D detection task, the object size at far and near distance in the image varies significantly due to the perspective projection [10, 48], which makes it challenging to utilize the learned object query mentioned in DETR [3] to fully represent the object property. Thus, in this paper, we propose to globally integrate context- and depth-aware features with transformers and inject depth hints into the transformer for better 3D reasoning.
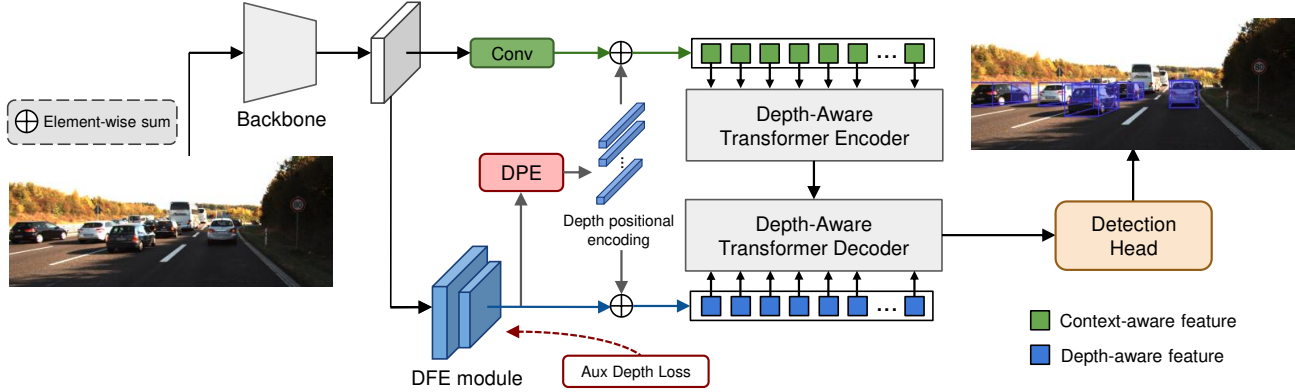
Figure 2. **The overall framework of our proposed MonoDTR.** The input image is first sent to the backbone to extract the features. The Depth-Aware Feature Enhancement (DFE) module learns depth-aware features via auxiliary supervision (Section 3.2), and context-aware features are extracted by convolution layers in parallel. The Depth-Aware Transformer (DTR) module then integrates two kinds of features, while the Depth Positional Encoding (DPE) module injects depth positional hints into the transformer (Section 3.3). Finally, the detection head is applied to predict the 3D bounding boxes (Section 3.4). Note that the auxiliary depth supervision is only used in the training phase.

3. Proposed Approach

3.1. Framework Overview

Figure 2 presents the framework of our MonoDTR, which mainly consists of four components: the backbone, the depth-aware feature enhancement (DFE) module, the depth-aware transformer (DTR) module, and the 2D-3D detection head. We adopt DLA-102 [55] as our backbone network following [29]. Given an input RGB image with resolution $H_{\text{inp}} \times W_{\text{inp}}$, the backbone outputs a feature map $\mathbf{F} \in \mathbb{R}^{C \times H \times W}$, where $H = \frac{H_{\text{inp}}}{8}$, $W = \frac{W_{\text{inp}}}{8}$, and $C = 256$. The DFE module is presented to implicitly learn depth-aware features (Section 3.2), while several convolution layers are applied to extract context-aware features in parallel. Then, we globally integrate two kinds of features by the DTR module and first attempt to inject depth positional hints into the transformer through the depth positional encoding (DPE) module (Section 3.3). Consequently, the anchor-based detection heads and loss functions are adopted for 2D and 3D object detection (Section 3.4).

3.2. Depth-Aware Feature Enhancement Module

Existing depth-assisted methods [10,48,52], using off-the-shelf depth estimators, suffer from the risk of introducing inaccurate depth priors and extra computation burden. To alleviate this, we propose a depth-aware feature enhancement (DFE) module for depth reasoning as in Figure 3. The precise depth map is utilized for auxiliary supervision in the training stage, making the DFE module implicitly learn the depth-aware features. Compared with previous works that apply an additional backbone [10, 48] or complicated architectures [35] to encode depths, we generate depth-aware features to assist 3D object detection with a lightweight module, significantly reducing the computation budget.
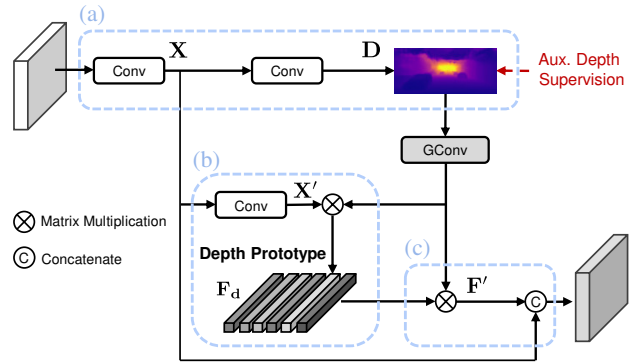

Figure 3. **The architecture of depth-aware feature enhancement (DFE) module.** The DFE module aims to implicitly learn depth-aware features via auxiliary supervision. (a) Generate initial depth-aware feature $\mathbf{X}$ and predict depth distribution $\mathbf{D}$. (b) Estimate feature representation of depth prototype $\mathbf{F_d}$. (c) Produce depth prototype enhanced feature $\mathbf{F'}$, and fuse with initial depth-aware feature $\mathbf{X}$. See Section 3.2 for details.

Learning initial depth-aware feature. To generate depth-aware features, we leverage an auxiliary depth estimation task and consider it as a sequential classification problem [13,35]. As illustrated in Figure 3(a), given the input feature $\mathbf{F} \in \mathbb{R}^{C \times H \times W}$ from the backbone, we adopt two convolution layers to predict the probability of discretized depth bins $\mathbf{D} \in \mathbb{R}^{D \times H \times W}$, where D is the number of depth categories (bins). The probability represents the confidence that the depth value of each pixel belongs to a certain depth bin. To discretize the depth ground truth from continuous space to discretization intervals, we utilize linear-increasing discretization (LID) [35,46] to formulate the depth bins (more details can be found in supplementary materials). To this end, the intermediate feature map $\mathbf{X} \in \mathbb{R}^{C \times H \times W}$ can be regarded as initial depth-aware features.

Depth prototype representation learning. To further enhance the capability of depth representation, we augment the feature of each pixel by introducing the central representation of the corresponding depth category (bin), inspired from the class center in [57]. The feature center of each depth category (regarded as the depth prototype) can be computed by aggregating the depth-aware features of each pixel belonging to a specified category. In practice, we first apply a group convolution [19] to the predicted depth map $\mathbf{D}$ to merge the adjacent depth categories (bins), reducing the class number from D to $D' = D/r$ with scale r. It helps to share similar depth cues and reduce computation. The representation of depth prototype $\mathbf{F}_d$ can be generated by gathering the feature of all pixels $\mathbf{X}'$ weighted by their probability to the depth category d:

$$\mathbf{F}_d = \sum_{i \in \mathcal{I}} \tilde{P}_{di} \mathbf{X}'_i, d = \{1, ..., D'\}, \quad (1)$$

where $\mathbf{X}'_i$ denotes the feature of the ith pixel in $\mathbf{X}'$, $\mathcal{I} \in \mathbb{R}^{H \times W}$ is the set of pixels in the feature map, and $\tilde{P}_{di}$ is the normalized probability to dth depth prototype. In this manner, $\mathbf{F}_d$ can express the global context information of each depth category as shown in Figure 3(b).

Feature enhancement with depth prototype. Now we can reconstruct new depth-aware features based on the depth prototype representation, which allows each pixel to understand the presentation of the depth category from the global view. The reconstructed feature $\mathbf{F}'$ is calculated as:

$$\mathbf{F}' = \sum_{d=1}^{D'} \tilde{P}_{di} \mathbf{F}_d. \quad (2)$$

Consequently, we obtain the enhanced depth feature by concatenating the initial depth-aware feature $\mathbf{X}$ and the reconstructed features $\mathbf{F}'$, followed by a simple 1×1 convolution layer, as shown in Figure 3(c).

3.3. Depth-Aware Transformer

Inspired by the tremendous success of the transformer [47] on modeling the long-range relationships, we exploit the transformer encoder-decoder architecture to construct the depth-aware transformer (DTR) module to globally integrate the context- and depth-aware features.

Transformer encoder. Our transformer encoder aims to improve context-aware features, which is built similar to previous works [3, 61]. The main component in the transformer is the self-attention mechanism [47]. Given the inputs: query $\mathbf{Q} \in \mathbb{R}^{N \times C}$, key $\mathbf{K} \in \mathbb{R}^{N \times C}$, and value $\mathbf{V} \in \mathbb{R}^{N \times C}$ with sequence length N, a single head self-attention layer can be briefly formulated as:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{C}})\mathbf{V}. \quad (3)$$
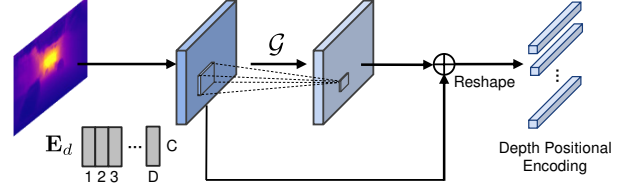

Figure 4. **The proposed depth positional encoding (DPE) module.** The DPE module generates depth positional encoding based on the depth category predicted by the DFE module. See Section 3.3 for details.

We take the flatten context-aware feature $\mathbf{X}_c \in \mathbb{R}^{N \times C}$, where $N = H \times W$, as the input to feed into the transformer encoder. The encoded context-aware feature can be obtained through multi-head self-attention operation and the feed-forward network (FFN).

Transformer decoder. The decoder is also built upon the standard transformer architecture. We propose utilizing the depth-aware features as the input of the decoder instead of learnable embeddings (object query) [3], which is different from the common usage in previous encoder-decoder vision transformer works [3, 18, 45, 60]. The main reason is that, in the monocular 3D object detection task, the camera views at near and far distances often cause significant changes in object scale due to the perspective projection [10, 48]. It makes the simple learnable embedding hard to fully represent the object's property and handle complex scale variant situations. In contrast, plentiful distance-aware cues are hidden in the depth-aware features. Thus, we propose adopting depth-aware features as the input of the transformer decoder. To this end, the decoder can take the power of cross-attention modules in the transformer to efficiently model the relationship between context- and depth-aware features, leading to better performance.

Depth positional encoding (DPE). Positional encoding [47] plays an important role for the transformer to introduce the location information. It is often generated with sinusoidal functions or in a learnable manner according to the pixel location of the image in vision tasks. Observing that the depth information is much better for the machine to understand the 3D world than the pixel-level relation, we first propose a general depth positional encoding (DPE) module to embed the depth positional hints for each pixel to the transformer. Specifically, as shown in Figure 4, the depth bin encodings $\mathbf{E}_d = [e_1, \ldots, e_D] \in \mathbb{R}^{D \times C}$ are constructed with learnable embeddings for each depth interval introduced in Section 3.2. The initial depth positional encoding $\mathbf{P} \in \mathbb{R}^{H \times W \times C}$ can be looked up from $\mathbf{E}_d$ according to the argmax of each pixel predicted depth category $\mathbf{D}$. To further represent the positional cues from local neighborhoods, a convolution layer $\mathcal{G}$ with the kernel size of 3×3 is applied and added to $\mathbf{P}$ to obtain final encoding, referred to as depth positional encoding (DPE).

Computation reduction. The standard self-attention layer in Equation 3 leads to $\mathcal{O}(N^2)$ time and memory, which damages the computational budget. To mitigate this issue, more recent works [8, 17, 49] make efforts on accelerating the attention operation. Among these methods, Linear transformer [17] proposes to approximate softmax operation with the linear dot product of features. Specifically, the similarity function in original transformer [47] can be formulated as: $\text{sim}(q, k) = \exp(\frac{q^\top k}{\sqrt{C}})$. It is replaced with $\text{sim}(q, k) = \phi(q)\phi(k)$ in [17], where $\phi(x) = \text{elu}(x) + 1$, and $\text{elu}(\cdot)$ is the exponential linear unit [11] activation function. To this end, $\phi(K)^\top$ and V can be combined first to reduce computation to $\mathcal{O}(N)$. We refer the readers to [17] for more details. In our transformer, we consider applying the linear attention described in [17] to replace the vanilla self-attention for higher inference speed.

3.4. 2D-3D Detection and Loss

Anchor definition. We adopt the single-stage detector [26, 36] with the pre-defined 2D-3D anchors to regress the bounding box. Each pre-defined anchor consists parameters with 2D bounding box $[x_{2d}, y_{2d}, w_{2d}, h_{2d}]$ and 3D bounding box $[x_p, y_p, z, w_{3d}, h_{3d}, l_{3d}, \theta]$. $[x_{2d}, y_{2d}]$ and $[x_p, y_p]$ represent the 2D box center and 3D object center projected to image plane. $[w_{2d}, h_{2d}]$ and $[w_{3d}, h_{3d}, l_{3d}]$ represent the physical dimension of 2D and 3D bounding box, respectively. z denotes the depth of 3D object center. θ is the observation angle. During training, we project all ground truth into the 2D space to calculate the intersection over union (IoU) with all 2D anchors. The anchor with IoU greater than 0.5 is chosen to assign with the corresponding 3D box for optimization.

Output transformation. Similar to prior works [10, 29, 48, 56], we follow Yolov3 [36] to predict $[t_x, t_y, t_w, t_h]_{2d}$ and $[t_x, t_y, t_w, t_h, t_l, t_z, t_\theta]_{3d}$ for each anchor, which aims at parameterizing the residual value for 2D and 3D bounding box, and also predict the classification scores cls. The output bounding box can be restored based on the anchor and the network prediction as follows:

$$
\begin{aligned}
[\hat{x}_{2d}, \hat{y}_{2d}] &= [t_x, t_y]_{2d} * [w_{2d}, h_{2d}] + [x_{2d}, y_{2d}] \\
[\hat{x}_p, \hat{x}_p] &= [t_x, t_y]_{3d} * [w_{2d}, h_{2d}] + [x_p, y_p] \\
[\hat{w}_{3d}, \hat{h}_{3d}, \hat{l}_{3d}] &= \exp([t_w, t_h, t_l]_{3d}) * [w_{3d}, h_{3d}, l_{3d}] \\
[\hat{w}_{2d}, \hat{h}_{2d}] &= \exp([t_w, t_h]_{2d}) * [w_{2d}, h_{2d}] \\
[\hat{z}, \hat{\theta}] &= [t_z, t_\theta]_{3d} + [z, \theta],
\end{aligned}
\tag{4}
$$

where $\hat{(\cdot)}$ denotes the recovered parameters of the 3D object. Note that we apply the same anchor center for 2D box center $[x_{2d}, y_{2d}]$ and 3D projection center $[x_p, y_p]$.

Loss function. The overall loss $\mathcal{L}$ contains a classification loss $\mathcal{L}_{cls}$ for objectness and class, a bounding box regression loss $\mathcal{L}_{reg}$ to optimize Equation 4, and a depth loss $\mathcal{L}_{dep}$

with auxiliary depth supervision described in Section 3.2:

$$
\mathcal{L} = \mathcal{L}_{cls} + \mathcal{L}_{reg} + \mathcal{L}_{dep}. \tag{5}
$$

We adopt the focal loss [25] to balance the samples for the classification task, and the smoothed-L1 loss [15] for the regression task. For the depth categorical prediction described in Section 3.2, we utilize the focal loss [25]:

$$
\mathcal{L}_{dep} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \text{FL}(\mathbf{D}(p), \hat{\mathbf{D}}(p)), \tag{6}
$$

where $\mathcal{P}$ is the pixel region on the image with the valid depth labels, and $\hat{\mathbf{D}}$ is the depth bins ground truth generated from LiDAR (more details are provided in the supplementary material).

4. Experiments

4.1. Setup

Dataset. We evaluate the proposed approach on the challenging KITTI 3D object detection dataset [14], which is the most commonly used benchmark for the 3D object detection task. It contains 7481 images for training and 7518 images for testing. We follow [5] to divide training samples into the training set (3712) and the validation set (3769). The ablation studies are conducted based on this split.

Evaluation metric. The average precision (AP) is used as the metric for evaluation in both 3D object detection and bird's eye view (BEV) detection tasks. We utilize 40 recall positions metric AP_{40} instead of original AP_{11} to avoid the bias [43]. The difficulty of the detection in the benchmark is divided into three levels ("Easy", "Mod.", "Hard") according to size, occlusion, and truncation. All methods are ranked based on AP_{3D} of moderate setting (Mod.) same as the KITTI benchmark. The thresholds of Intersection over Union (IoU) are 0.7, 0.5, 0.5 for cars, cyclists, and pedestrians categories following the official setting.

Implementation details. We use Adam optimizer to train our network for 120 epochs with batch size 4. The learning rate starts at 0.0001 and decays with a cosine annealing schedule. We apply 48 anchors on each pixel of the feature map with 3 aspect ratios of $\{0.5, 1.0, 1.5\}$, and 12 scales in height following the exponential function $24 \times 2^{i/4}, i = \{0, ..., 15\}$. For 3D anchor parameters, we calculate the mean and variance statistics of 3D ground truth in the training dataset as prior statistical knowledge of each anchor. Following [56], we crop the top 100 pixels of each image to reduce inference time, and all images are resized to 288 $\times$ 1280. In the training stage, we apply random horizontal mirroring as data augmentation. In the inference stage, we drop the predictions with a confidence score lower than 0.75 and adopt Non-Maximum Suppression (NMS) with IoU 0.4 to reduce redundancy.

Method	Reference	Time(ms)	AP$_{3D}$@IoU=0.7			AP$_{BEV}$@IoU=0.7		
			Easy	Mod.	Hard	Easy	Mod.	Hard
MonoPSR [20]	CVPR 2019	200	10.76	7.25	5.85	18.33	12.58	9.91
M3D-RPN [1]	ICCV 2019	160	14.76	9.71	7.42	21.02	13.67	10.23
MonoPair [7]	CVPR 2020	60	13.04	9.99	8.65	19.28	14.83	12.89
AM3D [31]	ICCV 2019	400	16.50	10.74	9.52	25.03	17.32	14.91
MoVi-3D [42]	ECCV 2020	45	15.19	10.90	9.26	22.76	17.03	14.85
PatchNet [30]	ECCV 2020	400	15.68	11.12	10.17	22.97	16.86	14.97
M3DSSD [29]†	CVPR 2021	-	17.51	11.46	8.98	24.15	15.93	12.11
D4LCN [10]	CVPR 2020	200	16.65	11.72	9.51	22.51	16.02	12.55
MonoDLE [32]	CVPR 2021	40	17.23	12.26	10.29	24.79	18.89	16.00
MonoRUn [4]	CVPR 2021	70	19.65	12.30	10.58	27.94	17.34	15.24
GrooMeD-NMS [21]	CVPR 2021	120	18.10	12.32	9.65	26.19	18.27	14.05
MonoRCNN [41]	ICCV 2021	70	18.36	12.65	10.03	25.48	18.11	14.10
Kinematic3D [2]	ECCV 2020	120	19.07	12.72	9.17	26.69	17.52	13.10
DDMP-3D [48]	CVPR 2021	180	19.71	12.78	9.80	28.08	17.89	13.44
CaDDN [35]	CVPR 2021	630	19.17	13.41	11.46	27.94	18.91	**17.19**
DFRNet [62]	ICCV 2021	180	19.40	13.63	10.35	28.17	19.17	14.84
MonoEF [59]	CVPR 2021	30	**21.29**	13.87	11.71	**29.03**	19.70	**17.26**
MonoFlex [58]	CVPR 2021	30	19.94	13.89	**12.07**	28.23	**19.75**	16.89
GUPNet [28]†	ICCV 2021	-	20.11	**14.20**	11.77	-	-	-
MonoDTR (Ours)	-	37	**21.99**	**15.39**	**12.73**	**28.59**	**20.38**	17.14

Table 1. **Detection performance of Car category on the KITTI test set.** The best and second best results are highlighted in **red** and **blue**, respectively. † indicates the results are reported in their papers.

Method	AP$_{3D}$@IoU=0.7			AP$_{BEV}$@IoU=0.7			AP$_{3D}$@IoU=0.5			AP$_{BEV}$@IoU=0.5		
	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
M3D-RPN [1]	14.53	11.07	8.65	20.85	15.62	11.88	48.53	35.94	28.59	53.35	39.60	31.76
MonoPair [7]	16.28	12.30	10.42	24.12	18.17	15.76	55.38	42.39	37.99	61.06	47.63	41.92
MonoDLE [32]	17.45	13.66	11.68	24.97	19.33	17.01	55.41	43.42	37.81	60.73	46.87	41.89
Kinematic3D [2]	19.76	14.10	10.47	27.83	19.72	15.10	55.44	39.47	31.26	61.79	44.68	34.56
GrooMeD-NMS [21]	19.67	14.32	11.27	27.38	19.75	15.92	55.62	41.07	32.89	61.83	44.98	36.29
MonoRUn [4]	20.02	14.65	12.61	-	-	-	59.71	43.39	38.44	-	-	-
CaDDN [35]	23.57	16.31	13.84	-	-	-	-	-	-	-	-	-
GUPNet [29]	22.76	16.46	13.72	31.07	22.94	19.75	57.62	42.33	37.59	61.78	47.06	40.88
MonoFlex [58]	23.64	17.51	14.83	-	-	-	-	-	-	-	-	-
MonoDTR (Ours)	**24.52**	**18.57**	**15.51**	**33.33**	**25.35**	**21.68**	**64.03**	**47.32**	**42.20**	**69.04**	**52.47**	**45.90**

Table 2. **Detection performance of Car category on the KITTI validation set.** We utilize **bold** to highlight the best results.

4.2. Main Results

Results of the Car category on the KITTI test set. As shown in Table 1, we compare our MonoDTR with several state-of-the-art monocular 3D object detection methods on the KITTI test set. It can be observed that our approach achieves better performance than other methods in terms of the moderate level of the two tasks, which is the most important metric in the benchmark. Furthermore, it is worth noting that our approach outperforms other depth-assisted methods by large margins. For instance, compared to top three depth-assist methods, DFRNet [62], CaDDN [35] and DDMP-3D [48], our MonoDTR obtains **2.59/1.76/2.38**, **2.82/1.98/1.27** and **2.28/2.61/2.93** improvements in AP$_{3D}$ at IoU threshold 0.7 on three settings, which indicates the effectiveness of the proposed depth-aware modules.

Results of the Car category on the KITTI validation set. We also conduct experiments on the KITTI validation dataset under different IoU thresholds and tasks as listed in Table 2. Our approach obtains superior performance over several image-only methods, benefiting from the auxiliary depth supervision. Specifically, compared with GUP-Net [28], our method achieves significant improvements of **6.41/4.99/4.61** in AP$_{3D}$ and **7.26/5.41/5.02** in AP$_{BEV}$ at IoU threshold 0.5 on the easy, moderate, and hard settings.

Results of Pedestrians and Cyclists categories on the KITTI test set. We further present the performance of pedestrians and cyclists categories in Table 3. Detecting these two categories is more challenging than cars due to their smaller size and non-rigid body, making it difficult to precisely locate the position. Overall, our model significantly outperforms all methods on pedestrian category with

Method	AP$_{3D}$(Ped.)			AP$_{3D}$(Cyc.)		
	Easy	Mod.	Hard	Easy	Mod.	Hard
MonoDLE [32]	9.64	6.55	5.44	4.59	2.66	2.45
MonoPair [7]	10.02	6.68	5.53	3.79	2.12	1.83
MonoFlex [58]	9.43	6.31	5.26	4.17	2.35	2.04
D4LCN [10]	4.55	3.42	2.83	2.45	1.67	1.36
DDMP3D [48]	4.93	3.55	3.01	4.18	2.50	2.32
CADDN [35]	12.87	8.14	6.76	**7.00**	**3.41**	**3.30**
MonoDTR (Ours)	**15.33**	**10.18**	**8.61**	5.05	3.27	3.19

Table 3. **Detection performance of Pedestrian and Cyclist categories on the KITTI test set** at 0.5 IoU threshold. We utilize **bold** to highlight the best results.

a considerable margin. For the cyclist 3D detection, we also achieve competitive results to CaDDN [35] and obtain better performance than other methods.

Running time analysis. We measure the average running time for processing the whole validation set with batch size 1 on a single Nvidia Tesla v100 GPU. As shown in Table 1, our model can achieve real-time performance at 27 FPS, which confirms the efficiency of our approach. Compared with state-of-the-art depth-assisted methods, our MonoDTR runs $17\times$ and $4.8\times$ faster than CaDDN [35] and DDMP-3D [48], respectively. The main reasons can be summarized as follows: (1) CaDDN [35] builds the bird's eye view representation from predicted depth maps to perform 3D detection, which applies more complicated architecture to generate precise depth predictions, leading to slow speed. (2) Fusion-based methods [10, 48] often utilize two separate backbones for extracting features of image and depth, which is time-consuming. Note that the depth estimator also takes additional inference time, which is not included in Table 1. On the contrary, our model learns depth-aware features through the lightweight DFE module with auxiliary supervision, which reduces running time significantly.

4.3. Ablation Study

Effectiveness of each proposed components. In Table 4, we conduct an ablation study to analyze the effectiveness of the proposed components: (a) Baseline: only using context-aware features for 3D object detection, *i.e.*, without all proposed depth-aware modules. (b) Replacing depth-aware features with object query [3] in the transformer, *i.e.* baseline + DETR-like transformer. (c) Replacing depth-aware features with features extracted from depth images generated by DORN [13]. (d) Integrating context- and depth-aware features with the convolutional concatenate operation. (e) Full model without depth prototype enhanced feature $\mathbf{F'}$. (f) MonoDTR (full model).

Firstly, we can observe from (b→f) that utilizing depth-aware features to replace the object query in the transformer can provide meaningful depth hints and improve the performance. Besides, compared to our end-to-end training

	Ablation	AP$_{3D}$@IoU=0.7		
		Easy	Mod.	Hard
(a)	Baseline	19.35	15.47	12.83
(b)	depth-aware feature → object query	20.09	16.10	14.07
(c)	depth-aware feature → DORN [13]	24.08	17.10	14.02
(d)	DTR → concat. operation	23.39	17.65	14.82
(e)	w/o depth prototype enhancement	23.72	18.22	15.36
(f)	MonoDTR (full model)	**24.52**	**18.57**	**15.51**

Table 4. **Analysis of different components of our approach** on the KITTI validation set for Car category.
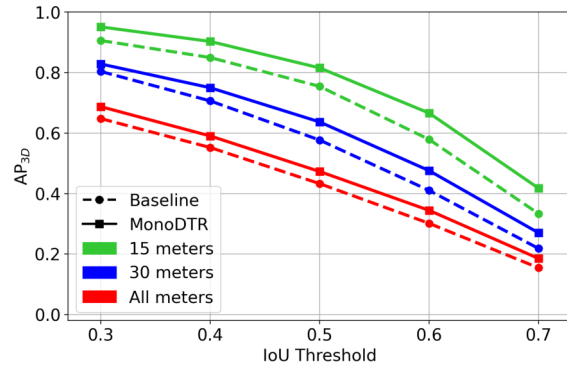

Figure 5. **Comparison of AP with different object depth ranges and IoU thresholds between baseline and MonoDTR** on the KITTI validation set for Car category. Best viewed in color.

framework (f), simply utilizing depth priors from the pre-trained depth estimator (c) leads to worse results. Next, we demonstrate that applying our depth-aware transformer (DTR) module (f) can more effectively integrate context- and depth-aware features than simple convolutional concatenation (d). Furthermore, utilizing our proposed depth prototype enhancement module can boost the performance (e→f). Finally, by applying all the designed modules, our full model (f) achieves significant improvement compared to the baseline (a). Also, an in-depth analysis in Figure 5 suggests that our method surpasses the baseline under different IoU thresholds and object depths. These results prove the effectiveness of our depth-aware modules.

Comparison with different positional encodings. We investigate the effectiveness of the proposed depth positional encoding (DPE) in Table 5. Compared with several commonly used positional encodings, including absolute positional encoding (APE) [12], conditional positional encoding (CPE) [9], sinusoidal positional encoding [47], and without using positional encoding (No PE), our proposed DPE achieves better performance on KITTI validation set. We believe that encoding the depth-aware cues is more effective for learning the position representation of 3D tasks than pixel-level encodings.

Plugging into the existing image-only methods. Our proposed approach is flexible to extend to existing image-only 3D object detectors to improve the depth reasoning capa-
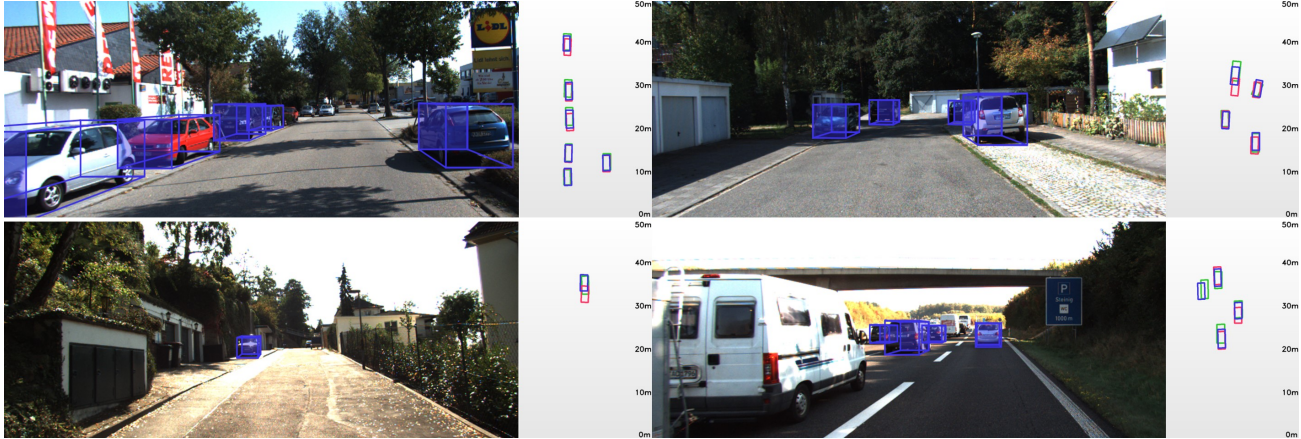
Figure 6. **Qualitative examples on the KITTI validation set.** We provide the predictions on the image view (left) and bird eye view (right). The purple boxes in the image and BEV plane represent the predictions from MonoDTR. The green and pink boxes on BEV are the ground truth and the predictions from baseline (without depth-aware modules), respectively. Best viewed in color and zoomed in.

| Positional Enc. | AP$_{3D}$@IoU=0.7 | | | AP$_{BEV}$@IoU=0.7 | | |
	Easy	Mod.	Hard	Easy	Mod.	Hard
No PE	23.65	17.76	15.05	31.33	24.02	20.83
Sinusoidal [47]	22.73	17.63	14.74	31.78	24.40	20.97
APE [12]	23.85	17.55	14.59	32.52	23.47	19.92
CPE [9]	24.34	18.04	15.14	33.01	24.69	20.48
DPE (Ours)	**24.52**	**18.57**	**15.51**	**33.33**	**25.35**	**21.68**

Table 5. **Comparison of different positional encoding mechanisms** on the KITTI validation set for Car category.

bility. We respectively plug our depth-aware modules into three popular monocular 3D object detectors: M3D-RPN [1], GAC [56], and MonoDLE [32], based on their official codes[123]. In practice, we take the features from the above models (before the detection head) as the initial features, and utilize our proposed modules (DFE, DTR, and DPE modules) to generate final integrated features, followed by their original detection head to detect 3D objects. As shown in Table 6, with the aid of our proposed depth-aware modules, these detectors achieve further improvements on the KITTI validation set, which demonstrates the flexibility and efficiency of our approach.

4.4. Qualitative Results

We provide the qualitative examples on the KITTI validation set in Figure 6. Compared with the baseline model without the aid of depth-aware modules, the predictions from MonoDTR are much closer to the ground truth. It shows that the proposed depth-aware modules can help to locate the object precisely. More qualitative results are included in the supplementary material.

---

[1] https://github.com/garrickbrazil/M3D-RPN
[2] https://github.com/Owen-Liuyuxuan/visualDet3D
[3] https://github.com/xinzhuma/monodle

| Method | AP$_{3D}$@IoU=0.7 | | | AP$_{BEV}$@IoU=0.7 | | |
	Easy	Mod.	Hard	Easy	Mod.	Hard
M3D-RPN [1]	14.53	11.07	8.65	20.85	15.62	11.88
M3D-RPN + **Ours**	**20.96**	**16.44**	**14.63**	**25.24**	**20.52**	**17.43**
Improvement	+6.43	+5.37	+5.98	+4.39	+4.90	+5.55
GAC [56]*	21.58	15.17	11.35	28.62	19.99	15.42
GAC + **Ours**	**24.30**	**17.28**	**13.35**	**33.02**	**23.06**	**18.22**
Improvement	+2.72	+2.11	+2.00	+4.40	+3.07	+2.80
MonoDLE [32]	17.45	13.66	11.68	24.97	19.33	17.01
MonoDLE + **Ours**	**18.68**	**15.69**	**13.41**	**26.67**	**21.40**	**18.67**
Improvement	+1.23	+2.03	+1.73	+1.70	+2.07	+1.66

Table 6. **Extension on existing image-only monocular 3D object detectors.** * indicates that we retrained without using extra right images. See details in Section 4.3.

5. Conclusion

In this paper, we propose a depth-aware transformer network for monocular 3D object detection. The proposed lightweight DFE module implicitly learns depth-aware features in an end-to-end fashion to avoid obtaining inaccurate depth priors and high computational cost from an off-the-shelf depth estimator. We also introduce the depth-aware transformer to globally integrate context- and depth-aware features, while the novel depth positional encoding (DPE) is designed to inject depth hints into the transformer. Comprehensive experiments on the KITTI dataset validate that our model achieves real-time detection and outperforms previous state-of-the-art monocular-based methods.

References

[1] Garrick Brazil and Xiaoming Liu. M3d-rpn: Monocular 3d region proposal network for object detection. In *ICCV*, 2019. 2, 6, 8

[2] Garrick Brazil, Gerard Pons-Moll, Xiaoming Liu, and Bernt Schiele. Kinematic 3d object detection in monocular video. In *ECCV*, 2020. 1, 6

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 2, 4, 7

[4] Hansheng Chen, Yuyao Huang, Wei Tian, Zhong Gao, and Lu Xiong. Monorun: Monocular 3d object detection by reconstruction and uncertainty propagation. In *CVPR*, 2021. 6

[5] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. In *NeurIPS*, 2015. 5

[6] Yilun Chen, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Dsgn: Deep stereo geometry network for 3d object detection. In *CVPR*, 2020. 1

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

[8] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers. In *ICLR*, 2021. 5

[9] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 2021. 7, 8

[10] Mingyu Ding, Yuqi Huo, Hongwei Yi, Zhe Wang, Jianping Shi, Zhiwu Lu, and Ping Luo. Learning depth-guided convolutions for monocular 3d object detection. In *CVPR*, 2020. 1, 2, 3, 4, 5, 6, 7

[11] Sepp Hochreiter Djork-Arné Clevert, Thomas Unterthiner. Fast and accurate deep network learning by exponential linear units (elus). In *ICLR*, 2016. 5

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 2, 7, 8

[13] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep Ordinal Regression Network for Monocular Depth Estimation. In *CVPR*, 2018. 3, 7

[14] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 5

[15] Ross Girshick. Fast r-cnn. In *ICCV*, 2015. 5

[16] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3d object detection from point cloud. In *CVPR*, 2020. 1

[17] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *ICML*, 2020. 5

[18] Bumsoo Kim, Junhyun Lee, Jaewoo Kang, Eun-Sol Kim, and Hyunwoo J. Kim. Hotr: End-to-end human-object interaction detection with transformers. In *CVPR*, 2021. 2, 4

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 4

[20] Jason Ku, Alex D Pon, and Steven L Waslander. Monocular 3d object detection leveraging accurate proposals and shape reconstruction. In *CVPR*, 2019. 1, 6

[21] Abhinav Kumar, Garrick Brazil, and Xiaoming Liu. Groomed-nms: Grouped mathematically differentiable nms for monocular 3d object detection. In *CVPR*, 2021. 6

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

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

[24] Peixuan Li, Huaici Zhao, Pengfei Liu, and Feidao Cao. Rtm3d: Real-time monocular 3d detection from object keypoints for autonomous driving. In *ECCV*, 2020. 2

[25] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 5

[26] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Cheng-Yang Fu Scott Reed, and Alexander C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 5

[27] Zechen Liu, Zizhang Wu, and Roland Tóth. SMOKE: Single-stage monocular 3d object detection via keypoint estimation. In *CVPR Workshops*, 2020. 2

[28] 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 *ICCV*, 2021. 6

[29] Shujie Luo, Hang Dai, Ling Shao, and Yong Ding. M3dssd: Monocular 3d single stage object detector. In *CVPR*, 2021. 2, 3, 5, 6

[30] Xinzhu Ma, Shinan Liu, Zhiyi Xia, Hongwen Zhang, Xingyu Zeng, and Wanli Ouyang. Rethinking pseudo-lidar representation. In *ECCV*, 2020. 2, 6

[31] Xinzhu Ma, Zhihui Wang, Haojie Li, Pengbo Zhang, Xin Fan, and Wanli Ouyang. Accurate monocular object detection via color-embedded 3d reconstruction for autonomous driving. In *ICCV*, 2019. 1, 2, 6

[32] 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 *CVPR*, 2021. 1, 2, 6, 7, 8

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

[34] Erli Ouyang, Li Zhang, Mohan Chen, Anurag Arnab, and Yanwei Fu. Dynamic depth fusion and transformation for monocular 3d object detection. In *ACCV*, 2020. 1

[35] Cody Reading, Ali Harakeh, Julia Chae, and Steven L. Waslander. Categorical depth distributionnetwork for monocular 3d object detection. In *CVPR*, 2021. 2, 3, 6, 7

[36] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 5

[37] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. In *BMVC*, 2019. 2

[38] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. In *BMVC*, 2019. 2

[39] 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*, 2020. 1

[40] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointr-cnn: 3d object proposal generation and detection from point cloud. In *CVPR*, 2019. 1

[41] Xuepeng Shi, Qi Ye, Xiaozhi Chen, Chuangrong Chen, Zhixiang Chen, and TaeKyun Kim. Geometry-based distance decomposition for monocular 3d object detection. In *ICCV*, 2021. 6

[42] Andrea Simonelli, Samuel Rota Bulò, Lorenzo Porzi, Elisa Ricci, and Peter Kontschieder. Towards generalization across depth for monocular 3d object detection. In *ECCV*, 2020. 2, 6

[43] Andrea Simonelli, Samuel Rota Rota Bulò, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *ICCV*, 2019. 2, 5

[44] Jiaming Sun, Linghao Chen, Yiming Xie, Siyu Zhang, Qinhong Jiang, Xiaowei Zhou, and Hujun Bao. Disp r-cnn: Stereo 3d object detection via shape prior guided instance disparity estimation. In *CVPR*, 2020. 1

[45] Bin Tan, Nan Xue, Song Bai, Tianfu Wu, and Gui-Song Xia. Planetr: Structure-guided transformers for 3d plane recovery. In *ICCV*, 2021. 4

[46] Yunlei Tang, Sebastian Dorn, and Chiragkumar Savani. Center3d: Center-based monocular 3d object detection with joint depth understanding. *arXiv preprint arXiv:2005.13423*, 2020. 3

[47] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 2, 4, 5, 7, 8

[48] 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 *CVPR*, 2021. 1, 2, 3, 4, 5, 6, 7

[49] Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020. 5

[50] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. FCOS3D: Fully convolutional one-stage monocular 3d object detection. In *ICCV Workshops*, 2021. 1

[51] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Probabilistic and Geometric Depth: Detecting objects in perspective. In *CoRL*, 2021. 2

[52] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *CVPR*, 2019. 1, 2, 3

[53] Xinshuo Weng and Kris Kitani. Monocular 3d object detection with pseudo-lidar point cloud. In *ICCV Workshops*, 2019. 1, 2

[54] Yurong You, Yan Wang, Wei-Lun Chao, Divyansh Garg, Geoff Pleiss, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving. In *ICLR*, 2020. 2

[55] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *CVPR*, 2018. 3

[56] Ming Liu Yuxuan Liu, Yuan Yixuan. Ground-aware monocular 3d object detection for autonomous driving. In *ICRA*, 2021. 2, 5, 8

[57] Fan Zhang, Yanqin Chen, Zhihang Li, Zhibin Hong, Jingtuo Liu, Feifei Ma, Junyu Han, and Errui Ding. Acfnet: Attentional class feature network for semantic segmentation. In *ICCV*, 2019. 4

[58] Yunpeng Zhang, Jiwen Lu, and Jie Zhou. Objects are different: Flexible monocular 3d object detection. In *CVPR*, 2021. 2, 6, 7

[59] Yunsong Zhou, Yuan He, Hongzi Zhu, Cheng Wang, Hongyang Li, and Qinhong Jiang. Monocular 3d object detection: An extrinsic parameter free approach. In *CVPR*, 2021. 6

[60] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 2, 4

[61] Cheng Zou, Bohan Wang, Yue Hu, Junqi Liu, Qian Wu, Yu Zhao, Boxun Li, Chenguang Zhang, Chi Zhang, Yichen Wei, and Jian Sun. End-to-end human object interaction detection with hoi transformer. In *CVPR*, 2021. 4

[62] Zhikang Zou, Xiaoqing Ye, Liang Du, Xianhui Cheng, Xiao Tan, Li Zhang, Jianfeng Feng, Xiangyang Xue, and Errui Ding. The devil is in the task: Exploiting reciprocal appearance-localization features for monocular 3d object detection. In *ICCV*, 2021. 6