國立臺灣大學電機資訊學院資訊工程學系

碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

深度引導跨視角多目相機三維物體檢測

CrossDTR: Cross-view and Depth-guided Transformers for 3D Object Detection

曾靖渝

Ching-Yu Tseng

指導教授: 陳文進 博士、徐宏民 博士

Advisor: Wen-Chin Chen, Ph.D., Winston H.Hsu, Ph.D.

中華民國 112 年 6 月

June, 2023

doi:10.6342/NTU202301047

國立臺灣大學碩士學位論文
口試委員會審定書
MASTER'S THESIS ACCEPTANCE CERTIFICATE
NATIONAL TAIWAN UNIVERSITY

深度引導跨視角多目相機三維物體檢測

CrossDTR: Cross-view and Depth-guided Transformers for 3D Object Detection

本論文係曾靖渝君（學號 R10922045）在國立臺灣大學資訊工程學系完成之碩士學位論文，於民國 112 年 6 月 30 日承下列考試委員審查通過及口試及格，特此證明。

The undersigned, appointed by the Department of Computer Science and Information Engineering on 30 June 2023 have examined a Master's thesis entitled above presented by Ching-Yu Tseng (student ID: R10922045) candidate and hereby certify that it is worthy of acceptance.

口試委員 Oral examination committee:

	陳文進	葉梅珍
（指導教授 Advisor）		
	陳奕廷	陳駿丞

系主任/所長 Director: 洪士灝

doi:10.6342/NTU202301047

摘要

為了在自駕車中以低成本實現準確的三維物體檢測，許多多目相機方法被提出來解決單相機方法中的遮擋問題。然而，由於缺乏準確的深度估計，現有的多目相機方法通常會在深度方向的射線上因難以檢測的小型物體（如行人）而預測多個邊界框，導致召回率極低。此外，直接將通常由大型網絡結構組成的深度預測模塊應用於現有的多目相機方法，無法滿足自駕車應用的即時預測要求。為了解決這些問題，我們提出了用於深度引導跨視角多目相機三維物體檢測（CrossDTR）。首先，我們設計了輕量級的「深度預測器」，以在監督過程中生成精確的物體稀疏深度圖和低維深度嵌入向量，而無需額外的深度數據集來監督。其次，我們開發了一個「深度引導跨視角多目變換器」，用於融合來自不同相機視角的深度嵌入和影像特徵，並生成三維邊界框。廣泛的實驗表明，我們的方法在行人檢測方面總共超過現有的多目相機方法 10%，在整體平均精度（mAP）和標准化檢測得分（NDS）指標方面超過約 3%。此外，計算分析顯示，我們的方法比先前的方法快 5 倍。我們的代碼將在 https://github.com/sty61010/CrossDTR 公開提供。

關鍵字：電腦視覺、自駕車、物件偵測

doi:10.6342/NTU202301047

doi:10.6342/NTU202301047

Abstract

To achieve accurate 3D object detection at a low cost for autonomous driving, many multi-camera methods have been proposed and solved the occlusion problem of monocular approaches. However, due to the lack of accurate estimated depth, existing multi-camera methods often generate multiple bounding boxes along a ray of depth direction for difficult small objects such as pedestrians, resulting in an extremely low recall. Furthermore, directly applying depth prediction modules to existing multi-camera methods, generally composed of large network architectures, cannot meet the real-time requirements of self-driving applications. To address these issues, we propose Cross-view and Depth-guided Transformers for 3D Object Detection, CrossDTR. First, our lightweight *depth predictor* is designed to produce precise object-wise sparse depth maps and low-dimensional depth embeddings without extra depth datasets during supervision. Second, a *cross-view depth-guided transformer* is developed to fuse the depth embeddings as well as image features from cameras of different views and generate 3D bounding boxes. Extensive experiments

doi:10.6342/NTU202301047

demonstrated that our method hugely surpassed existing multi-camera methods by 10 percent in pedestrian detection and about 3 percent in overall mAP and NDS metrics. Also, computational analyses showed that our method is 5 times faster than prior approaches. Our codes will be made publicly available at https://github.com/sty61010/CrossDTR.

Keywords: Computer Vision, Autonomous Driving, Object Detection

doi:10.6342/NTU202301047

Contents

Page

Verification Letter from the Oral Examination Committee **i**

摘要 **iii**

Abstract **v**

Contents **vii**

List of Figures **ix**

List of Tables **xi**

Chapter 1 Introduction **1**

Chapter 2 Related Work **5**

2.0.1 Monocular 3D Object Detection 5

2.0.2 Multi-Camera 3D Object Detection 6

2.0.3 Depth-guided Monocular Methods 6

Chapter 3 Method **9**

3.0.1 Problem Definition 9

3.0.2 Overall Architecture 9

3.0.3 Object-wise Sparse Depth Map 10

3.0.4 Depth Predictor 11

3.0.5 Cross-view and Depth-guided Transformer 12

doi:10.6342/NTU202301047

3.0.6 3D Detection Head and Loss 13

Chapter 4 Experiments **15**

4.0.1 Setup 15

4.0.2 Quantitative Results 17

4.0.3 Ablation Study 18

4.0.4 False Positive Predictions Results 18

4.0.5 Qualitative Results 19

Chapter 5 Conclusion **21**

References **23**

doi:10.6342/NTU202301047

List of Figures

1.1 **Multi-camera methods suffer from inaccurate depth estimation.** Red and green bounding boxes represent inaccurate and accurate predictions respectively. The above 2D-to-3D projection diagram mainly shows that (a) previous multi-view methods usually produce a row of false positives predictions alone a ray of depth, but (b) our method, guided with depth hints, can precisely predict only one bounding box. Plot (c) and (d) demonstrate the corresponding bird-eye view predictions of (a) and (b). . 2

1.2 **The overall framework of our proposed CrossDTR.** First, Multi-view images are fed into a feature extractor backbone to generate image features. Then, image features are fed into our Depth Predictor (in Sec. 3.0.4) to produce depth embeddings and predicted depth maps. Lastly, given image features, depth embeddings, and 3D object queries, our Cross-view and Depth-guided Transformer (in Sec. 3.0.5) conducts cross-view attention and cross-depth attention to generate 3D bounding boxes. Note that we minimize the difference between predicted depths and our generated object-wise sparse depth maps (in Sec. 3.0.3) under supervision during training. Best viewed in color. 4

doi:10.6342/NTU202301047

3.1 **Visualization of false positive predictions on the nuScenes validation set.** We provide two qualitative examples, (1) and (2), with bird-eye-view (BEV) and camera-view representations. In the first row, the left and right images illustrate the focused areas of the global and zoomed-in BEVs, where blue and orange bounding boxes represent predictions and ground truth respectively. In the second row, the images in the camera view show predictions from models. Under a fair comparison with the same network backbone (PETR [20] with ResNet50 backbone [8]), our cross-view and depth-guided method (c) effectively mitigates the false positive issue in prior multi-view baselines (a), i.e. **does not produce repeated bounding boxes along the ray of depth**. Also, we even surpass methods equipped with a heavy-weight pretrained depth prediction module (b). Best viewed in color and zoom-in. 14

4.1 **The precision-recall curve of pedestrian class.** Fig. 4.1 shows the comparison of AP between baseline (dotted lines) and our method (solid lines). The red, blue, and green colors represent the distance threshold at 0.5, 1.0, and 4.0 respectively. Regardless of distance, our method hugely outperforms the baseline on small object (e.g. pedestrian) detection. 19

doi:10.6342/NTU202301047

List of Tables

3.1 **Comparison with SOTA methods on the nuScene validation set.** PETR[20] are trained with CBGS[48]. The best results are shown in **bold**. 14

4.1 **Comparison with lightweight multi-view methods.** We utilize **bold** to highlight the best results. 16

4.2 **Ablation study of depth-guided module**. DE denotes Depth Embedding and DDN Loss denotes Depth Distribution Network Loss. Our method was built on PETR[20] with DDN Loss and DE. 16

4.3 **Study of false positive predictions for the pedestrian class.** We choose PETR[20] with ResNet50 [8] and PETR[20] with depth-pretrained VoVNetV2 [16] as baselines and compare them with our method, CrossDTR, with different distance thresholds. We utilize **bold** to highlight the best results. . . 17

doi:10.6342/NTU202301047

doi:10.6342/NTU202301047

Chapter 1 Introduction

Detecting instances of objects in the 3D space from sensor information, i.e. 3D object detection, is crucial for various intelligent systems, such as autonomous driving and indoor robotics. Previous work tends to rely on accurate depth information from different sensors, such as LiDAR signals [14, 40, 41] and binocular information [4, 17], to accomplish superior performance. In recent years, in order to achieve high-quality detection at a low cost, several methods based on commodity cameras have been proposed. Among them, since naive monocular detection [29, 30, 35, 36, 39, 42, 47] suffers from the problems of occlusion and deficiency of cross-view information, methods transforming camera information from multiple views into Bird-Eye-View [5, 10, 11, 18, 20, 21, 31, 37, 45], called multi-view methods, has received increasing attention.

Though these multi-view methods have made some progress with cross-view information and Bird-Eye-View representation [9–11, 18, 28, 46], we observe existing practices suffered from either extremely low recall in small objects due to inaccurate depth or an unaffordable computational burden because of complex depth prediction modules. Specifically, while some methods fusing information from multiple views [5, 10, 11, 18, 20, 21, 31, 37, 45] easily locate the pixel coordinate of small objects in images, they can hardly estimate precise distances of objects from the image plane. Consequently, these methods tend to predict a row of false positive bounding boxes along a ray of depth direction in

doi:10.6342/NTU202301047

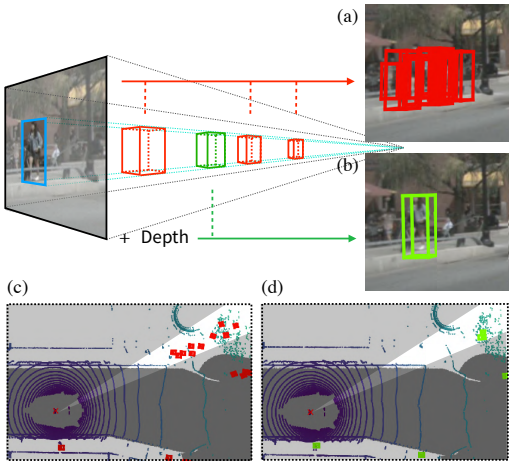

Figure 1.1: **Multi-camera methods suffer from inaccurate depth estimation.** Red and green bounding boxes represent inaccurate and accurate predictions respectively. The above 2D-to-3D projection diagram mainly shows that (a) previous multi-view methods usually produce a row of false positives predictions alone a ray of depth, but (b) our method, guided with depth hints, can precisely predict only one bounding box. Plot (c) and (d) demonstrate the corresponding bird-eye view predictions of (a) and (b).

candidate regions when detecting small objects (shown in Fig. 1.1), leading to low recall in perception and poor follow-up prediction and planning. In addition, some previous monocular approaches utilized complex depth prediction modules [6, 44] or large-scale depth-pretrained backbone [7, 15, 16, 26] to provide depth cues. Nevertheless, directly applying them to existing multi-camera methods, generally composed of large network architectures, cannot meet the real-time requirements of the self-driving application (Tab. 4.1). From the two observations above, we conclude that a module is needed to obtain depth hints from multiple cameras and fuse both depth and image information from different views in real time.

To achieve the goal, we proposed CrossDTR, a novel end-to-end Cross-view and Depth-guided Transformer network for multi-camera 3D object detection as shown in Fig. 1.2. To efficiently obtain depth hints for downstream 3D object detection, we leverage a lightweight *depth predictor* to produce precise depth maps for each view (Sec. 3.0.4). Specifically, inspired by previous depth-aware methods [6, 12, 13, 43, 44], the depth pre-

doi:10.6342/NTU202301047

dictor is supervised with our generated object-wise sparse depth maps without extra depth dataset (Sec. 3.0.3). Then, to fused the depth and image information from multi-view cameras effectively, we propose a novel *cross-view and depth-guided transformer* (Sec. 3.0.5). In short, the Transformer Encoder is used to compress high-resolution depth maps into low-dimensional depth embeddings, and the Transformer Decoder performs the cross-attention mechanism among depth as well as image information from multi-views.

Experimental results demonstrated that our depth-guided method resolves the problem of false positive predictions on small objects (Fig. 1.1) and achieves overall improvement with the limited computational burden. Compared with existing multi-camera methods on the nuScenes dataset [1], we increased by 10 percent Average precision (AP) in pedestrian detection and about 3 percent in mAP and NDS metrics. Also, computational analyses demonstrated that our lightweight method is 5 times faster than prior methods under similar network backbones. To sum up, the overall contributions of this work can be summarized as follows:

- We build up a novel cross-view and depth-guided perception framework, Cross-DTR, to insert accurate depth cues into multi-view detection methods.
- Our proposed depth-guided module can alleviate the problem of false positive predictions along the direction of depth for small objects.
- Our framework achieves state-of-the-art 3D detection performance on the nuScenes dataset [1] with fewer computational budgets compared with existing multi-view or depth-guided methods.

doi:10.6342/NTU202301047

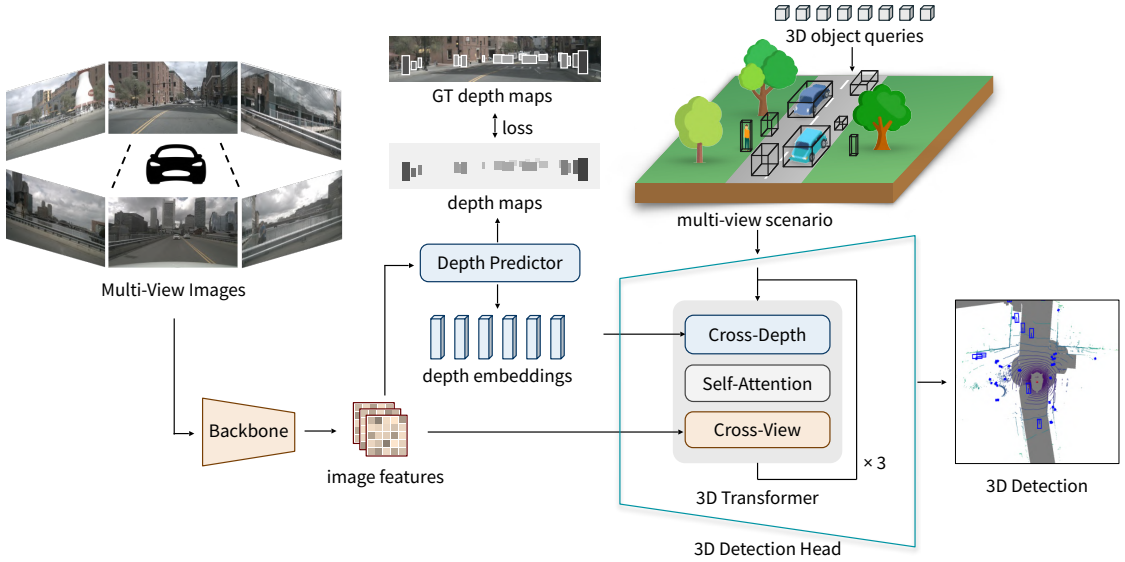

Figure 1.2: **The overall framework of our proposed CrossDTR.** First, Multi-view images are fed into a feature extractor backbone to generate image features. Then, image features are fed into our Depth Predictor (in Sec. 3.0.4) to produce depth embeddings and predicted depth maps. Lastly, given image features, depth embeddings, and 3D object queries, our Cross-view and Depth-guided Transformer (in Sec. 3.0.5) conducts cross-view attention and cross-depth attention to generate 3D bounding boxes. Note that we minimize the difference between predicted depths and our generated object-wise sparse depth maps (in Sec. 3.0.3) under supervision during training. Best viewed in color.

doi:10.6342/NTU202301047

Chapter 2 Related Work

2.0.1 Monocular 3D Object Detection

Monocular 3D object detection [29, 30, 35, 36, 39, 42] has received lots of attention due to the low cost of commodity cameras. It originates from Orthographic Feature Transform (OFT) [30], which projects camera features into ego pose coordinate uniformly, voxelized the features, and uses the detector from PointPillar [14]. OFT [30] is the first method that deals with camera features in a LiDAR-based [14, 40, 41] technique, but OFT [30] predicts the depth value of each pixel uniformly and thus results in inaccurate depth estimation. Extend from OFT, Pseudo-lidar [39, 42] and CaDDN [29] methods use a convolutional neural network to predict depth distribution and auxiliary loss to enhance performance. Apart from the above methods, the other monocular methods [35, 36] directly regression 3D representation in camera coordinate. FCOS3D [36] is built on FCOS [34], which is a single-stage 2D object detection framework. Furthermore, PGD [35] is an improved version of FCOS3D [36] with extra depth information. In conclusion, depth information is critical for monocular methods to enhance performance.

doi:10.6342/NTU202301047

2.0.2 Multi-Camera 3D Object Detection

Multi-camera methods [5, 10, 11, 18, 20, 21, 31, 37, 45] have been proved to solve the problem of occlusion through temporal and spatial information. DETR-based methods [18, 20, 21, 37] were first proposed. DETR3D [37] was built based on DETR [2] and utilized the attention mechanism to select features from different camera. Unlike DETR [2] in the 2D space, we can not assure the candidate area of objects in the 3D space, so DETR3D [37] initialize object query randomly with uniform distribution and pick features by projecting object position into camera coordinate. As an improved version of DETR3D [37], PETR [20] enhance performance by initializing object queries with the method from Anchor DETR [38] and applying 3D positional embedding to embed 3D coordinate frustum into multi-head attention. PETRv2 [21] optimized PETR by adding temporal information. Additionally, some researchers regard that Bird-Eye-View (BEV) [9, 23, 25, 27, 28, 32, 46] representations can provide better space concepts in 3D coordinates. BEVDet methods [10, 11] transform image features into BEV according to Lift-Splat-Shoot (LSS) Method [28] and propose BEV data augmentation [11] to avoid over-fitting. BEVFormer [18] proposed BEV query and use the deformable attention [49] to suppress computation. BEVFormer[18] also applies temporal information by cross-attention between BEV queries from different time stamps. However, those methods still lack accurate depth estimation.

2.0.3 Depth-guided Monocular Methods

Monocular methods can not achieve competitive performance in comparison with LiDAR-based methods [14, 40, 41] and binocular methods [4, 17]. The main reason is

doi:10.6342/NTU202301047

inaccurate depth estimation, and thus the depth-aware methods [3, 6, 12, 13, 43, 44] are proposed recently. ASTransformer [3] first proposed to use depth maps supervision to enhance the performance of depth estimation. Besides, both MonoDTR [12] and MonoDETR [43] compose depth-aware embedding from depth maps and are supervised by ground truth depth maps to enhance the performance of monocular object detection. However, all the above methods still build on monocular methods and will suffer from complex post-processing between cameras to aggregate all information and remove repeated predicted bounding boxes. Multi-camera method with a depth-guided module is still missing so far.

doi:10.6342/NTU202301047

doi:10.6342/NTU202301047

Chapter 3 Method

3.0.1 Problem Definition

In this work, we aim to precisely detect instances of objects in 3D space given multiple scanned RGB images [24]. Specifically, let $\mathbf{I}_{sensors} = \{\mathcal{I}_1, ..., \mathcal{I}_{N_{cameras}}\}$ represent a set of scanned multi-view images, and $\mathbf{B}_{LiDAR} = \{\mathcal{B}_1, ..., \mathcal{B}_{N_{box}}\}$ denotes a set of ground truth bounding boxes, a 3D bounding box $\hat{\mathcal{B}}_i \in \mathbf{B}_{LiDAR}$ is formulated as a vector with 7 degree of freedom:

$$\hat{\mathcal{B}}_i = (x_c, y_c, z_c, l, w, h, \theta), \tag{3.1}$$

where (x_c, y_c, z_c) denotes the center of each bounding box. (l, w, h) represents the length, width, and height of the cuboid respectively. θ means orientation (yaw) of each object. Formally, given a set of predicted bounding boxes $\hat{\mathbf{B}}_{LiDAR}$, a multi-view 3D object detector $\underline{\mathbf{f}}_{Det}$ is defined as follows:

$$\hat{\mathbf{B}}_{LiDAR} = \underline{\mathbf{f}}_{Det}(\mathbf{I}_{sensors}). \tag{3.2}$$

3.0.2 Overall Architecture

Fig. 1.2 illustrates our architecture. First, we feed our multi-view images $\{\mathcal{I}_1, ..., \mathcal{I}_{N_{cameras}}\}$ into our model. Without external data, [8] is applied as our backbone to extract fea-

doi:10.6342/NTU202301047

tures $\mathcal{F}_{view}$ for each view. Then, we feed image features $\mathcal{F}_{view}$ into Depth Predictor (in Sec. 3.0.4). Given single-view image features $\mathcal{F}_{view}$, the Depth Predictor produces low-dimensional depth embeddings and depth maps by a Transformer encoder. During training, we minimize the difference between the predicted depth maps and our generated sparse depth maps (in Sec. 3.0.3) in a supervised manner. Lastly, given image features, depth embeddings, and 3D object queries, our Cross-view and Depth-guided Transformer (in Sec. 3.0.5) conducts cross-view attention and cross-depth attention to generate 3D bounding boxes.

3.0.3 Object-wise Sparse Depth Map

Unlike some prior depth-guided monocular methods requiring costly dense depth maps during training, we extend [43] to multi-view scenarios and leverage only the sparse depth hints provided by the raw LiDAR data, which is more cost-effective. We first detail our depth generation process below.

Given a camera matrix $T \in \mathbb{R}^{3\times4}$ and a point $p \in \mathbb{R}^3$ in the LiDAR coordinate, we define the transformation function $\mathcal{T}$ from the LiDAR coordinate to the camera coordinate as follows:

$$\begin{aligned} \mathcal{T}(T,p) &= \begin{bmatrix} u & v & d \end{bmatrix}^T, \\ \text{where } d \cdot \begin{bmatrix} u & v & 1 \end{bmatrix}^T &= T \cdot (p \oplus 1), \end{aligned} \tag{3.3}$$

and $\oplus$ denotes tensor concatenation. We transform the center point and corners of each bounding box into each camera view by (3.3). Let $p^{m,i}_{centers}, d^{m,i}_{centers}, \mathbf{P}^{m,i}_{corners}$ be the center point, the depth value of the center, and the set of corner points of the bounding box $\mathcal{B}_i$ in

doi:10.6342/NTU202301047

the m-th camera coordinate, then

$$p^{m,i}_{centers} = \begin{bmatrix} u_c & v_c & d_c \end{bmatrix}^T = \mathcal{T}(T_m, p_i), \tag{3.4}$$

$$\mathbf{P}^{m,i}_{corners} = \{\mathcal{T}(T_m, p) \mid p \in \mathcal{C}(\mathcal{B}_i)\}, \tag{3.5}$$

where $T_m \in \mathbf{T}$, $p_i = \begin{bmatrix} x_c & y_c & z_c \end{bmatrix}^T$ and $x_c, y_c, z_c \in \mathcal{B}_i$. $\mathcal{C}(\mathcal{B})$ is the function returning 8 corners of the given 3D bounding box $\mathcal{B}$. We further extract the 2D bounding box $\mathcal{B}^{2d}_{m,i} = (u^i_{min}, u^i_{max}, v^i_{min}, v^i_{max})$ with respect to $\mathcal{B}_i$ in the m-th camera from $\mathbf{P}^{m,i}_{corners}$ by getting the minimum and maximum (u, v) value in $\mathbf{P}^{m,i}_{corners}$.

Next, we collect all valid 2D bounding boxes $\mathcal{B}^{2d}_{m,i}$ for each camera and their depth values $d^{m,i}_{centers}$ to a new set $\boldsymbol{\mathcal{V}}_m$. We set the pixel value to the depth of the object center point if the pixel lies in an object's bounding box. If the pixel lies in multiple bounding boxes, we set it to the nearest one. Lastly, the object-wise sparse depth map is obtained by adopting linear-increasing discretization (LID) [33].

3.0.4 Depth Predictor

Inspired by previous depth-guided methods [12, 43] and other methods [9–11, 28, 29], we utilize the Depth Predictor from [43] to learn depth information from object-wise sparse depth maps. To save the memory of our model, we use lightweight architecture built by convolution layers to predict depth distribution and match the number of depth bins as 3D positional embedding [20, 21]. Given image features $\mathcal{F}_{view}$, we use light-weight network $\underline{\mathbf{f}}_{ddn}$ to predict depth logits $\hat{\mathcal{D}}$ and depth probability $\hat{\mathcal{D}}_{prob}$ among each depth map. Besides, we utilize Transformer encoder ψ_i to encode image features $\mathcal{F}_{view}$

doi:10.6342/NTU202301047

into depth embeddings $\mathcal{F}_{depth}$ with multi-head attention ψ_i:

$$\begin{aligned} \mathcal{E}_0 &= \mathcal{F}_{view}, \\ \mathcal{E}_i &= \psi_i(\mathcal{E}_{i-1}, \psi_{i-1}), i = 1, ..., I, \\ \mathcal{F}_{depth} &= \mathcal{E}_I, \end{aligned} \tag{3.6}$$

where $\mathcal{E}_i \in \mathbb{R}^{L \times C}$ represents the depth embeddings from the Transformer encoder in the depth predictor. C is the size of the embeddings. $L = H_d \times W_d$ is the length of depth embeddings. The number i denotes the i-th layer in the encoder, and the encoder contains total I layers. The final depth embedding is utilized in Sec. 3.0.5.

3.0.5 Cross-view and Depth-guided Transformer

As the Transformer has successfully been used to fuse different modalities, we adopt it to combine both image features and depth embeddings. We adopt PETR [20] methods to conduct attention between different views. Besides, inspired by MonoDETR [43], we insert depth embedding into multi-view attention from PETR [20].

Cross-view Attention. We follow cross-view attention as methods from PETR [20], and we feed image features $\mathcal{F}_{view}$ as keys and values. We utilize 3D positional embedding [20] as query positional embedding.

Cross-depth Attention. Previous methods [20, 21] only use visual information and thus lack depth cues for the detector. Inspired by [43], [12] and methods [3, 6, 12, 13, 43, 44], we suggest inserting depth hints as depth embedding into the detector to provide more detailed information for small objects. After we obtain depth embeddings $\mathcal{F}_{depth} \in \mathbb{R}^{B \times N \times C \times H_d \times W_d}$ from Sec. 3.0.4, we flatten depth embeddings into $\mathcal{F}_{depth} \in \mathbb{R}^{\acute{L} \times B \times C}$

doi:10.6342/NTU202301047

where $\acute{L} = N \times H_d \times W_d$ indicates the flatten depth embeddings $\mathcal{F}_{depth}$ among all camera views. Then, we follow the previous Sec. 3.0.5 and select depth embeddings as keys and values for multi-head attention. Those depth embeddings can not only learn pixel-level depth hints in a single view in Depth Predictor (in Sec. 3.0.4) but also consider depth messages from the other views during cross attention mechanism.

3.0.6 3D Detection Head and Loss

3D Detection Head. To learn information between the camera view features and 3D position, we adopt the 3D Detection Head from PETR [20]. The 3D Detection Head from PETR [20] initialize 3D reference points among 3D space and adopt multi-layer perception to learn the candidate area in the ego-pose coordinate. Then, the 3D Detection Head generates 7 degrees of freedom to represent bounding boxes in the ego-pose coordinate.

Depth Distribution Network Loss. To conduct the depth-guided method on a predefined depth map in Sec. 3.0.3, we borrow the depth-guided method from [43] and refer to CaDDN [29] and adopt Depth Distribution Network Loss (DDN Loss) to regularize the predicted depth map values and predicted depth map logits. Following CaDDN [29], we build our loss as the following equation:

$$\underline{\mathbf{L}}_{ddn} = \frac{1}{W_d \cdot H_d} \sum_{u=1}^{W_d} \sum_{v=1}^{H_d} \underline{\mathbf{FL}}(\mathcal{D}(u,v), \hat{\mathcal{D}}(u,v)), \tag{3.7}$$

where W_d and H_d represent the size of depth map logits, and (u, v) denotes the position of pixels. $\underline{\mathbf{FL}}$ means adopted Focal Loss [19].

doi:10.6342/NTU202301047

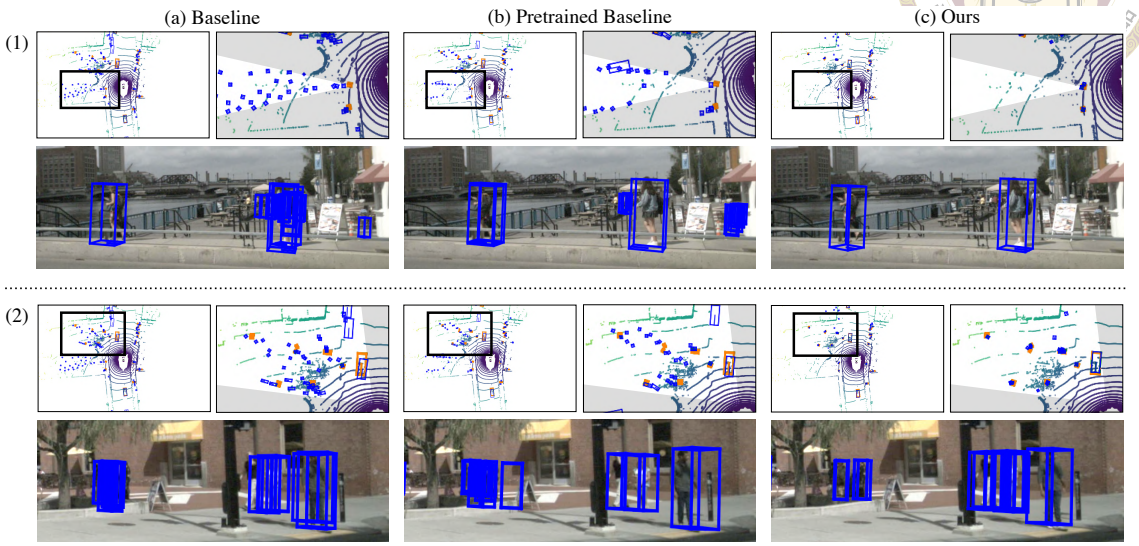

Figure 3.1: **Visualization of false positive predictions on the nuScenes validation set.** We provide two qualitative examples, (1) and (2), with bird-eye-view (BEV) and camera-view representations. In the first row, the left and right images illustrate the focused areas of the global and zoomed-in BEVs, where blue and orange bounding boxes represent predictions and ground truth respectively. In the second row, the images in the camera view show predictions from models. Under a fair comparison with the same network backbone (PETR [20] with ResNet50 backbone [8]), our cross-view and depth-guided method (c) effectively mitigates the false positive issue in prior multi-view baselines (a), i.e. **does not produce repeated bounding boxes along the ray of depth**. Also, we even surpass methods equipped with a heavy-weight pretrained depth prediction module (b). Best viewed in color and zoom-in.

Table 3.1: **Comparison with SOTA methods on the nuScene validation set.** PETR[20] are trained with CBGS[48]. The best results are shown in **bold**.

Methods	Backbone	Img Size	#param.	FPS↑	GFLOPs↓	mAP↑	NDS↑	mATE↓	mASE↓	mAOE↓	mAVE↓	mAAE↓
CenterNet[47]	DLA	1600*900	-	-	-	0.306	0.328	0.716	0.264	0.609	1.426	0.658
FCOS3D[36]	ResNet101	1600*900	52.5M	1.7	2008.2	0.295	0.372	0.806	0.268	0.511	1.315	**0.170**
PGD[35]	ResNet101	1600*900	53.6M	1.4	2223.0	0.335	0.409	0.732	**0.263**	**0.423**	1.285	0.172
Detr3D[37]	ResNet101	1600*900	**51.3M**	2.0	1016.8	0.303	0.374	0.860	0.278	0.437	0.967	0.235
BEVDet[11]	Swin-T	1408*512	126.6M	1.9	2962.6	0.349	0.417	**0.637**	0.269	0.490	0.914	0.268
PETR[20]	ResNet101	1408*512	59.2M	5.3	504.6	0.357	0.421	0.710	0.270	0.490	0.885	0.224
CrossDTR	ResNet101	1408*512	53.3M	**5.8**	**483.9**	**0.370**	**0.426**	0.773	0.269	0.482	**0.866**	0.203

doi:10.6342/NTU202301047

Chapter 4 Experiments

4.0.1 Setup

Dataset. In this paper, we use the nuScenes dataset [1] as our benchmark, which provides camera, radar, and LiDAR sensor data with 3D bounding box annotations. Its data is mainly composed of camera data and provides only sparse LiDAR data as an auxiliary. As it lacks data to provide depth information like dense LiDAR or depth maps, we generate object-wise sparse depth maps (in Sec. 3.0.3 for the cross-view and depth-guided method. The nuScenes dataset [1] contains 1000 scenes and each scene is 20 seconds in length and annotated at 2HZ.

Implementation Details. Following training policies from [20, 21, 37], we use features from backbones [8, 15, 16, 22] with downsample scale of $\frac{1}{16}$ and $\frac{1}{32}$. And we feed the features into both depth predictor and cross-view attention. Besides, we follow [20, 21] to sample 64 points for depth in 3D positional embeddings and also for depth predictor to estimate depth distribution. Specifically, we supervise generated depth maps only during training. We use the AdamW optimizer with a learning rate of 2e-4 to train our model. We train our model for 24 epochs on 4 Nvidia 3090 GPUS with a total batch size of 8 for 48 hours. We use input images at resolution 512×1408 for our baseline model.

doi:10.6342/NTU202301047

Table 4.1: **Comparison with lightweight multi-view methods.** We utilize **bold** to highlight the best results.

Methods	Backbone	#param.	FPS↑	GFLOPs↓	mAP↑
DETR3D[37]	ResNet101	51.3M	2.0	1016.8	0.303
BEVDet[11]	ResNet50	54.1M	9.3	452.0	0.299
BEVFormer[18]	ResNet50	68.7M	2.3	1303.5	0.252
PETR[20]	ResNet50	36.6M	10.4	297.2	0.317
CrossDTR	ResNet50	**31.8M**	**10.6**	**268.1**	**0.326**

Table 4.2: **Ablation study of depth-guided module**. DE denotes Depth Embedding and DDN Loss denotes Depth Distribution Network Loss. Our method was built on PETR[20] with DDN Loss and DE.

Methods	DE	DDN Loss	mAP↑	NDS↑
PETR[20]			0.357	0.421
CrossDTR	✓		0.366	0.423
CrossDTR	✓	✓	**0.370**	**0.426**

Baselines. We compare CrossDTR with both monocular and multi-camera approaches. CenterNet[47], FCOS3D [36], and PGD [35] represent monocular approaches, while DETR3D [37], PETR [20], and BEVDet [11] serve as the baselines of multi-camera ones. For fair comparison, we only adopt the performance of these approaches without tricks such as test-time augmentation [35, 36], CBGS [48], and oversampling [48]. Besides, the comparison with the lightweight BEVFormer [18] (from their official repository) without encoding extra temporal information is also included.

Evaluation Metrics. We report mean Average Translation Error (mATE), mean Average Scale Error (mASE), mean Average Orientation Error (mAOE), mean Average Velocity Error (mAVE), mean Average Attribute Error (mAAE), mean Average Precision (mAP), and nuScenes detection score (NDS). mAP estimates the distance between the centers of a predicted and a ground-truth 3D bounding box. To evaluate the efficacy of our method in solving false positive predictions, we report the Average Precision of pedestrian as our

doi:10.6342/NTU202301047

Table 4.3: **Study of false positive predictions for the pedestrian class.** We choose PETR[20] with ResNet50 [8] and PETR[20] with depth-pretrained VoVNetV2 [16] as baselines and compare them with our method, CrossDTR, with different distance thresholds. We utilize **bold** to highlight the best results.

Methods	Depth-pretrained	Backbone	AP (Pedestrain) @ Dist.↑		
			[0.5]	[1.0]	[4.0]
PETR[20]		ResNet50	0.09	0.401	0.809
	✓	VoVNetV2	0.102	0.426	0.870
CrossDTR		ResNet50	**0.320**	**0.689**	**0.875**

metrics. We also take Frame Per Second (FPS) and Giga Flops (GFLOPs) into consideration to evaluate the real-time ability of multi-camera models.

4.0.2 Quantitative Results

Comparison with State-of-the-art. As shown in Tab. 3.1, our method surpasses other previous methods and achieves the state-of-the-art performance of mAP and NDS on the validation dataset [1]. To begin with, CenterNet [47], FCOS3D [36], and PGD[35] are classic monocular baseline. Our method exceeds by more than **3 percent on mAP and 2 percent on NDS**. Additionally, compared with the SOTA multi-camera methods (starting from the fifth row), our method still surpasses all of them by at least **1.3 percent on mAP and 0.5 percent on NDS**. Swin-T represents Swin-Transformer [22], which is the strongest backbone among the Tab. 3.1. Our method with ResNet101 also beats BEVDet [11] with Swin-T [22]. Then, our method needs the least computational resource **(483.9 GFLOPs and 5.8 FPS)**. Our method is lightweight and can conduct real-time 3D detection on the nuScenes [1] dataset.

Comparison with lightweight multi-view methods. Tab. 4.1 shows the comparison between our method and previous multi-camera methods. Note that all the scores are

doi:10.6342/NTU202301047

from their official repositories. Since we conduct our experiments on the validation set with limited computation resources, we choose a smaller backbone ResNet50 [8] to extract features from input images at resolution 512×1408. Our proposed method overtakes all previous multi-camera methods, even against DETR3D [37] with stronger ResNet101 backbone [8] and BEVFormer [18] with temporal information. Our method surpasses the second best method by **0.9 percent on mAP and 1.1 percent on NDS**. Besides, our model contains the least parameters as shown in Tab. 4.1 and attains the highest score **(10.6) on FPS**. The result shows that our model can conduct real-time detection.

4.0.3 Ablation Study

Tab. 4.2 shows the effectiveness of our cross-view and depth-guided module. We conduct an ablation study to verify the effectiveness of depth embedding (DE) and Depth Distribution Network Loss (DDN Loss). We take PETR [20] as baseline model . We find the performance is improved by 0.9 percent on mAP and 0.2 percent on NDS when we plug depth embeddings into the cross-attention, and the full model achieves the best performance increasing by **1.3 percent on mAP and 0.5 percent on NDS**.

4.0.4 False Positive Predictions Results

To verify whether our method can resolve the false positive problem, we consider Average Precision (AP). Tab. 4.3 shows the AP of the pedestrian class with different distance thresholds on the validation set. Our method surpasses our baseline with a depth-pretrained backbone by **over 10 percent on average** among each threshold. Moreover, Fig. 4.1 illustrates our overall performance predominantly exceeds the baseline on all

doi:10.6342/NTU202301047

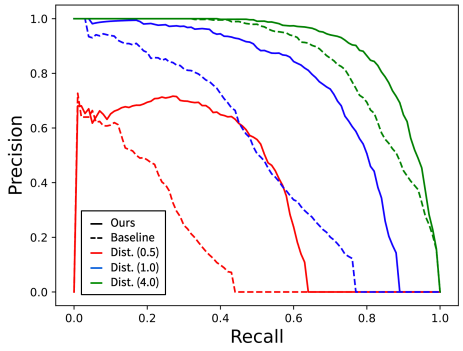

Figure 4.1: **The precision-recall curve of pedestrian class.** Fig. 4.1 shows the comparison of AP between baseline (dotted lines) and our method (solid lines). The red, blue, and green colors represent the distance threshold at 0.5, 1.0, and 4.0 respectively. Regardless of distance, our method hugely outperforms the baseline on small object (e.g. pedestrian) detection.

thresholds and thus resolves the false positive issue. Red, blue, and green represent the distance threshold at 0.5, 1.0, and 4.0 respectively.

4.0.5 Qualitative Results

Fig. 3.1 shows the qualitative result. Orange and blue bounding boxes represent ground truth and predictions respectively. As shown in Fig. 3.1, both PETR [20] with ResNet50 backbone [8] and PETR [20] with depth-pretrained VoVNetV2 [7, 15, 16, 26] still predict a row of false positive predictions along the direction of depth for small objects. Since depth-pretrained backbones are generally pretrained on the external dataset and contain different settings on camera matrices, we suggest that those backbones can narrowly deal with the false positive problem due to weak depth estimation. Nevertheless, our method can predominately alleviate this problem due to referred depth information from the internal dataset [1].

doi:10.6342/NTU202301047

doi:10.6342/NTU202301047

Chapter 5 Conclusion

In this paper, we design an end-to-end Cross-view and Depth-guided Transformer, called CrossDTR, for 3D object detection. To address the false positive bounding boxes commonly existing in prior multi-view approaches, a lightweight Depth Predictor, supervised by our produced object-wise sparse depth maps, is proposed to generate low-dimensional depth embeddings. Furthermore, to combine image and depth hints from different views, a Cross-view and Depth-guided Transformer is developed to fuse this information efficiently. We are optimistic that our proposed method would pave a new way for developing a cost-effective and real-time 3D object detector.

doi:10.6342/NTU202301047

doi:10.6342/NTU202301047

References

[1] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuscenes: A multimodal dataset for autonomous driving. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 11621–11631, 2020.

[2] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers. In European conference on computer vision, pages 213–229. Springer, 2020.

[3] W. Chang, Y. Zhang, and Z. Xiong. Transformer-based monocular depth estimation with attention supervision. 2021.

[4] Y. Chen, S. Liu, X. Shen, and J. Jia. Dsgn: Deep stereo geometry network for 3d object detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 12536–12545, 2020.

[5] Z. Chen, Z. Li, S. Zhang, L. Fang, Q. Jiang, and F. Zhao. Graph-detr3d: Rethinking overlapping regions for multi-view 3d object detection. arXiv preprint arXiv:2204.11582, 2022.

[6] M. Ding, Y. Huo, H. Yi, Z. Wang, J. Shi, Z. Lu, and P. Luo. Learning depth-guided convolutions for monocular 3d object detection. In Proceedings of the IEEE/CVF

doi:10.6342/NTU202301047

Conference on Computer Vision and Pattern Recognition Workshops, pages 1000–1001, 2020.

[7] V. Guizilini, R. Ambrus, S. Pillai, A. Raventos, and A. Gaidon. 3d packing for self-supervised monocular depth estimation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 2485–2494, 2020.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.

[9] A. Hu, Z. Murez, N. Mohan, S. Dudas, J. Hawke, V. Badrinarayanan, R. Cipolla, and A. Kendall. Fiery: Future instance prediction in bird's-eye view from surround monocular cameras. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 15273–15282, 2021.

[10] J. Huang and G. Huang. Bevdet4d: Exploit temporal cues in multi-camera 3d object detection. arXiv preprint arXiv:2203.17054, 2022.

[11] J. Huang, G. Huang, Z. Zhu, and D. Du. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. arXiv preprint arXiv:2112.11790, 2021.

[12] K.-C. Huang, T.-H. Wu, H.-T. Su, and W. H. Hsu. Monodtr: Monocular 3d object detection with depth-aware transformer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4012–4021, 2022.

[13] A. Kumar, G. Brazil, E. Corona, A. Parchami, and X. Liu. Deviant: Depth equivariant network for monocular 3d object detection. arXiv preprint arXiv:2207.10758, 2022.

doi:10.6342/NTU202301047

[14] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 12697–12705, 2019.

[15] Y. Lee, J.-w. Hwang, S. Lee, Y. Bae, and J. Park. An energy and gpu-computation efficient backbone network for real-time object detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, pages 0–0, 2019.

[16] Y. Lee and J. Park. Centermask: Real-time anchor-free instance segmentation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 13906–13915, 2020.

[17] P. Li, X. Chen, and S. Shen. Stereo r-cnn based 3d object detection for autonomous driving. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7644–7652, 2019.

[18] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Q. Yu, and J. Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. arXiv preprint arXiv:2203.17270, 2022.

[19] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In Proceedings of the IEEE international conference on computer vision, pages 2980–2988, 2017.

[20] Y. Liu, T. Wang, X. Zhang, and J. Sun. Petr: Position embedding transformation for multi-view 3d object detection. arXiv preprint arXiv:2203.05625, 2022.

[21] Y. Liu, J. Yan, F. Jia, S. Li, Q. Gao, T. Wang, X. Zhang, and J. Sun. Petrv2: A

doi:10.6342/NTU202301047

unified framework for 3d perception from multi-camera images. arXiv preprint arXiv:2206.01256, 2022.

[22] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 10012–10022, 2021.

[23] C. Lu, M. J. G. van de Molengraft, and G. Dubbelman. Monocular semantic occupancy grid mapping with convolutional variational encoder–decoder networks. IEEE Robotics and Automation Letters, 4(2):445–452, 2019.

[24] J. Mao, S. Shi, X. Wang, and H. Li. 3d object detection for autonomous driving: A review and new outlooks. arXiv preprint arXiv:2206.09474, 2022.

[25] B. Pan, J. Sun, H. Y. T. Leung, A. Andonian, and B. Zhou. Cross-view semantic segmentation for sensing surroundings. IEEE Robotics and Automation Letters, 5(3):4867–4873, 2020.

[26] D. Park, R. Ambrus, V. Guizilini, J. Li, and A. 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.

[27] L. Peng, Z. Chen, Z. Fu, P. Liang, and E. Cheng. Bevsegformer: Bird's eye view semantic segmentation from arbitrary camera rigs. arXiv preprint arXiv:2203.04050, 2022.

[28] J. Philion and S. Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In European Conference on Computer Vision, pages 194–210. Springer, 2020.

doi:10.6342/NTU202301047

[29] C. Reading, A. Harakeh, J. Chae, and S. 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.

[30] T. Roddick, A. Kendall, and R. Cipolla. Orthographic feature transform for monocular 3d object detection. arXiv preprint arXiv:1811.08188, 2018.

[31] D. Rukhovich, A. Vorontsova, and A. Konushin. Imvoxelnet: Image to voxels projection for monocular and multi-view general-purpose 3d object detection. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pages 2397–2406, 2022.

[32] A. Saha, O. Mendez, C. Russell, and R. Bowden. Translating images into maps. In 2022 International Conference on Robotics and Automation (ICRA), pages 9200–9206. IEEE, 2022.

[33] Y. Tang, S. Dorn, and C. Savani. Center3d: Center-based monocular 3d object detection with joint depth understanding. In DAGM German Conference on Pattern Recognition, pages 289–302. Springer, 2020.

[34] Z. Tian, C. Shen, H. Chen, and T. He. Fcos: Fully convolutional one-stage object detection. In Proceedings of the IEEE/CVF international conference on computer vision, pages 9627–9636, 2019.

[35] T. Wang, Z. Xinge, J. Pang, and D. Lin. Probabilistic and geometric depth: Detecting objects in perspective. In Conference on Robot Learning, pages 1475–1485. PMLR, 2022.

[36] T. Wang, X. Zhu, J. Pang, and D. Lin. Fcos3d: Fully convolutional one-stage monoc-

doi:10.6342/NTU202301047

ular 3d object detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 913–922, 2021.

[37] Y. Wang, V. C. Guizilini, T. Zhang, Y. Wang, H. Zhao, and J. Solomon. Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In Conference on Robot Learning, pages 180–191. PMLR, 2022.

[38] Y. Wang, X. Zhang, T. Yang, and J. Sun. Anchor detr: Query design for transformer-based detector. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, pages 2567–2575, 2022.

[39] X. Weng and K. Kitani. Monocular 3d object detection with pseudo-lidar point cloud. In Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, pages 0–0, 2019.

[40] Y. Yan, Y. Mao, and B. Li. Second: Sparsely embedded convolutional detection. Sensors, 18(10):3337, 2018.

[41] T. Yin, X. Zhou, and P. Krahenbuhl. Center-based 3d object detection and tracking. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 11784–11793, 2021.

[42] Y. You, Y. Wang, W.-L. Chao, D. Garg, G. Pleiss, B. Hariharan, M. Campbell, and K. Q. Weinberger. Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving. arXiv preprint arXiv:1906.06310, 2019.

[43] R. Zhang, H. Qiu, T. Wang, X. Xu, Z. Guo, Y. Qiao, P. Gao, and H. Li. Monodetr: Depth-aware transformer for monocular 3d object detection. arXiv preprint arXiv:2203.13310, 2022.

doi:10.6342/NTU202301047

[44] Y. Zhang, X. Ma, S. Yi, J. Hou, Z. Wang, W. Ouyang, and D. Xu. Learning geometry-guided depth via projective modeling for monocular 3d object detection. arXiv preprint arXiv:2107.13931, 2021.

[45] Y. Zhang, Z. Zhu, W. Zheng, J. Huang, G. Huang, J. Zhou, and J. Lu. Beverse: Unified perception and prediction in birds-eye-view for vision-centric autonomous driving. arXiv preprint arXiv:2205.09743, 2022.

[46] B. Zhou and P. Krähenbühl. Cross-view transformers for real-time map-view semantic segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 13760–13769, 2022.

[47] X. Zhou, D. Wang, and P. Krähenbühl. Objects as points. arXiv preprint arXiv:1904.07850, 2019.

[48] B. Zhu, Z. Jiang, X. Zhou, Z. Li, and G. Yu. Class-balanced grouping and sampling for point cloud 3d object detection. arXiv preprint arXiv:1908.09492, 2019.

[49] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai. Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159, 2020.

doi:10.6342/NTU202301047