OPA-3D: Occlusion-Aware Pixel-Wise Aggregation for Monocular 3D Object Detection

Yongzhi Su, Yan Di, Guangyao Zhai, *Graduate Student Member, IEEE*, Fabian Manhardt, Jason Rambach, Benjamin Busam, Didier Stricker, and Federico Tombari

***Abstract*—Monocular 3D object detection has recently made a significant leap forward thanks to the use of pre-trained depth estimators for pseudo-LiDAR recovery. Yet, such two-stage methods typically suffer from overfitting and are incapable of explicitly encapsulating the geometric relation between depth and object bounding box. To overcome this limitation, we instead propose to jointly estimate dense scene depth with depth-bounding box residuals and object bounding boxes, allowing a two-stream detection of 3D objects that harnesses both geometry and context information. Thereby, the geometry stream combines visible depth and depth-bounding box residuals to recover the object bounding box via explicit occlusion-aware optimization. In addition, a bounding box based geometry projection scheme is employed in an effort to enhance distance perception. The second stream, named as the Context Stream, directly regresses 3D object location and size. This novel two-stream representation enables us to enforce cross-stream consistency terms, which aligns the outputs of both streams, and further improves the overall performance. Extensive experiments on the public benchmark demonstrate that OPA-3D outperforms state-of-the-art methods on the main Car category, whilst keeping a real-time inference speed.**

***Index Terms*—Computer vision for transportation, deep learning for visual perception, object detection.**

I. Introduction

MONOCULAR 3D object detection empowers a wide spectrum of applications, e.g., autonomous driving [1], robotics manipulation [2], [3], [4], and scene understanding [5]. The bottleneck in this task is scale-ambiguous distance perception [6], [7], [8] induced by the perspective projection onto the image plane. As consequence, a few pixels in image space can make large differences in 3D, which in turn makes it very hard to directly predict the pose of objects far away [8].

To overcome this limitation, pseudo-LiDAR detectors commonly employ pre-trained depth estimation networks to generate intermediate point cloud representations of the scene, in an effort to better leverage 3D scene priors. Subsequently, a 3D object detector is utilized to jointly regress object location, size and heading direction. The main advantage of pseudo-LiDAR methods resides in the fact that their performance on object detection can be automatically improved when updating the depth generator. However, efficiency and generalization ability are strong bottlenecks, restricting the performance of such methods [9]. To address this issue and enhance distance perception at the same time, single-stage monocular 3D object detectors [10], [11], [12] design geometric depth priors, e.g., localization errors [8], geometry projection [7], [10], or spatial relationship [13], for the object center estimation. These single-stage 3D detectors demonstrate promising results on public datasets, however, their performance is still inferior to methods pre-trained with a depth dataset, with 16.46% of Monocon [11] compared to 16.87% of DD3D [6] on the main category in KITTI-3D [1].

Another challenge is (self-)occlusion. In a cluttered traffic scene, objects are often occluded or truncated, which complicates inference due to insufficient context cues. Thereby, self-occlusion remains challenging since benchmarks, such as KITTI-3D, only provide occlusion and truncation level and fail to offer detailed ground truth such as object visibility masks. Thus, the majority of methods [6], [7], [10] simply regress all target information from the image regardless of handling the occlusion. As exception, MonoRun [14] utilizes a latent vector to encapsulate occlusion, truncation and shape cues of the object. Moreover, [15] leverages view transformations to generate novel observations of the scene, mitigating the occlusion caused by perspective projection.

On the contrary, in this paper we propose an **O**cclusion-Aware **P**ixel-Wise **A**ggregation network (OPA-3D) that takes advantage of geometry and context cues to regress 3D object bounding boxes in traffic scenes, combining the advantages of both pseudo-LiDAR and single-stage detectors. OPA-3D computes a shared feature embedding and subsequently stacks two streams, i.e., *Geometry Stream* and *Context Stream*, on top of it.

Manuscript received 1 August 2022; accepted 11 January 2023. Date of publication 19 January 2023; date of current version 31 January 2023. This letter was recommended for publication by Associate Editor N. J. Sanket and Editor M. Vincze upon evaluation of the reviewers' comments. This work was supported in part by EU Horizon Europe Framework Program under Grant 101058236 (HumanTech). *(Yongzhi Su and Yan Di contributed equally to this work.) (Corresponding author: Yan Di.)*

Yongzhi Su and Didier Stricker are with the German Research Center for Artificial Intelligence (DFKI GmbH), 67663 Kaiserslautern, Germany, and also with the Faculty of Computer Science, TU Kaiserslautern, 67663 Kaiserslautern, Germany (e-mail: yongzhi.su@dfki.de; didier.stricker@dfki.de).

Yan Di, Guangyao Zhai, and Benjamin Busam are with the Faculty of Computer Science, Technische Universität München, 85748 Garching bei München, Germany (e-mail: diy17@mails.tsinghua.edu.cn; guangyao.zhai@tum.de; b.busam@tum.de).

Fabian Manhardt is with Google, 8002 Zurich, Switzerland (e-mail: fabian.manhardt@tum.de).

Jason Rambach is with the German Research Center for Artificial Intelligence (DFKI GmbH), 67663 Kaiserslautern, Germany (e-mail: jason.rambach@dfki.de).

Federico Tombari is with the Faculty of Computer Science, Technische Universität München, 85748 Garching bei München, Germany, and also with the Google, 8002 Zurich, Switzerland (e-mail: tombari@in.tum.de).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2023.3238137, provided by the authors.

Digital Object Identifier 10.1109/LRA.2023.3238137

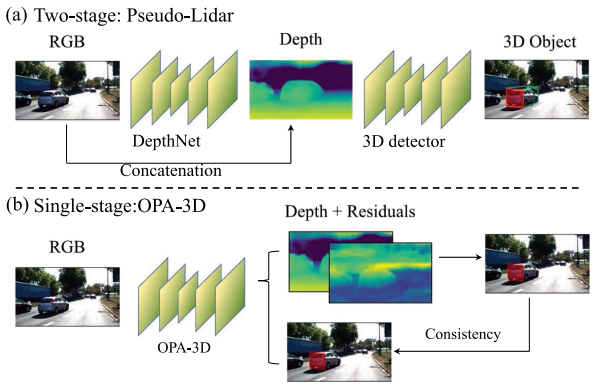

Fig. 1. Compared to conventional two-stage pseudo-Lidar methods that simply append the utilized object detector to a pre-trained depth estimator, OPA-3D takes advantage of joint dense depth estimation (which can be pre-trained) and direct object bounding box regression, enabling geometry-guided cross-task consistency terms. Results on public datasets demonstrate that our new network architecture outperforms all pseudo-Lidar detectors.

As for Context Stream, we adopt GUPNet [10] to directly regress object bounding boxes from an RGB image. Geometry Stream instead recovers the 3D information from geometrical cues. To recover the 3D bounding box, Geometry Stream predicts 1) the dense depth map, and 2) the residual vectors from each point on the object surface to the corresponding bounding box faces, along with 3) the uncertainty of each residual vector. We show in Section III-C that the 3D bounding box can be estimated via solving an occlusion-aware quadratic function. In addition, we also introduce a novel Bird-Eye-View (BEV) bounding box projection scheme which offers geometric constraints for the bounding box center estimation. Since the actual 3D bounding box can be derived from both streams, we are able to enforce consistency terms to align the individual outputs, which in turn enhances the overall performance. In Fig. 1, we compare OPA-3D with conventional pseudo-LiDAR methods [16], [17]. In particular, OPA-3D is end-to-end trainable and also allows for depth pre-training.

Our main contributions can be summarized as follows:

1) We propose OPA-3D, which leverages both geometry and context information with two parallel branches. Our novel uncertainty-based occlusion-aware bounding box recovery from predicted dense depth map and depth-bounding box residuals facilitates cross-branch consistencies.
2) To enhance the distance perception, we further propose novel bounding box projection based 2D-3D geometric consistencies in BEV.
3) OPA-3D performs effectively on real-world KITTI-3D and nuScenes, whilst enabling real-time applications with an inference speed of 25 Hz.

II. Related Work

In this section we highlight the three main existing branches of related work in monocular 3D object detection.

A. Monocular 3D Detection With Depth Prediction

Taking advantage of the development of deep learning methods, especially in monocular depth estimation, approaches such as [16], [17] proposed to initially estimate the depth map from a single RGB image and subsequently transfer it into a PL (pseudo-LiDAR) point cloud. LiDAR-based object detectors can be directly applied to estimate the 3D object bounding box from the PL point cloud representation. Subsequent work attempted to strengthen this pipeline by fully leveraging the information from the RGB image [17], [18]. [19], [20] also showed improvement by training the PL methods in an end-to-end manner. Although the monocular depth estimation can be trained with rich raw video or stereo data, the task itself is still ill-conditioned and ambiguous, making it the main error source for the PL 3D object detection [21]. To overcome the limitation of imprecise distance perception, learning the depth map as an auxiliary task has been considered a better choice [6] which can exploit the additional large-scale training data while not restricting the 3D object detection to the depth error. Our proposed approach also benefits from the additional available data for depth training.

B. Monocular 3D Detection With Shape Priors

Following the work of Murthy et al. [22], a branch of approaches start with integrating shape prior in the 3D detection pipeline. Early works simplified the shape as keypoints [23], [24]. Combined with CAD models, the keypoints can be utilized to determine object size or solve the object pose with a PnP (Perspective-n-Point) solver. However, the labeling of the keypoints is time-consuming and often inaccurate. To mitigate this, [12] automated the labeling process by proposing a deformable model-fitting pipeline. Taken into account that the understanding of object shape contributes to the object 3D detection, [25], [26] proposed the dense object shape matching from LiDAR point cloud or using a render-based loss. Unlike explicitly applying the object shape, we encode the object shape implicitly in the depth residual prediction. We define the 3D bounding box as the optimal one given by the dense residual depth prediction, which also minimizes the influence of the inaccurate object shape annotation.

C. Monocular 3D Detection With Geometry Consistency

Deep3DBox [27] firstly defined the geometry consistency to lift the 2D detection to 3D detection by assuming that the 3D bounding box should fit the 2D bounding box tightly. [28] extended the prior work by formulating this geometry constraint in both image view and BEV (Bird's Eye View). However, this formulation assumes that the 2D predictions are accurate. To tackle this issue, [29], [30], [31] used 3D detection shifted from 2D detection only as an initial prediction. [25], [29] generated more 3D prediction proposals based on the initial prediction, while [30], [31] refined the initial prediction along with other geometrically constraints, such as object orientation. Very recently, MonoPair [13] adopted the geometric relationship between objects, enabling the optimization of 3D predictions across all object instances in the image. We propose a similar 2D-3D bounding box geometry constraint in the BEV and reduce the side effect from inaccurate 2D bounding box by adding a bounding box confidence rate.

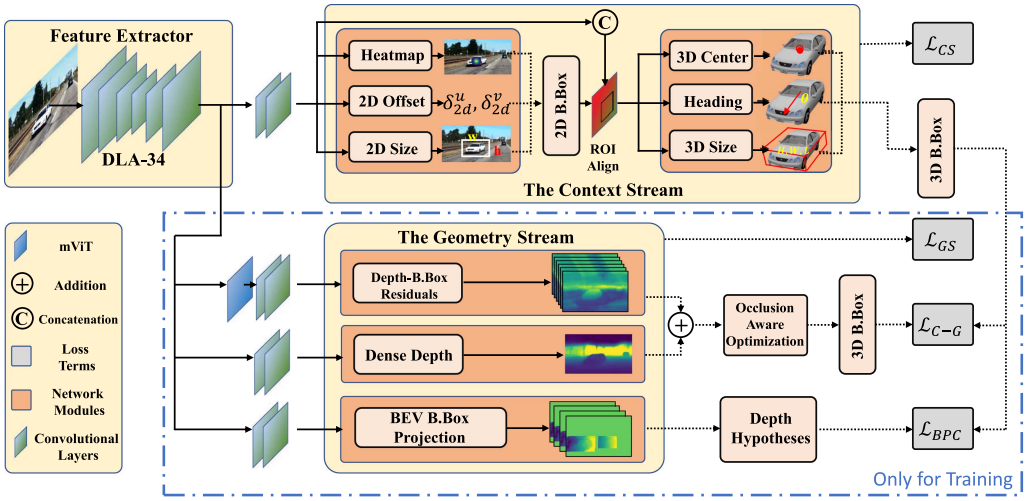

Fig. 2. Schematic overview of our OPA-3D architecture. We first extract features from the given image, then the features are further processed in two streams. The first two heads in the Geometry Stream are used for dense depth and depth-bounding box residuals (DBR) estimation. By pixel-wise aggregation of the two predictions, we can recover the 3D bounding box parameters with an occlusion-aware optimization function. The last bounding box projection head is applied to provide 2D-3D constraints to enhance distance perception. We follow GUP-Net [10] to design the 2D and 3D detection heads as the Context Stream.

III. Methodology

Given a single RGB image, our goal is to estimate the 3 Degrees-of-Freedom (DoF) object center together with the 3 DoF object size and the 1 DoF heading direction and an optional detection confidence for each object of interest. Moreover, in line with other works [10], [11], we require a lot of appropriate training data for depth pre-training. Nevertheless, as the depth prediction head of OPA-3D can be trained with LiDAR points as well as depth maps, we can easily make use of many large-scale depth datasets, e.g. KITTI-Depth [1]. Note that only RGB images are required during inference.

Overview: In contrast to pseudo-Lidar methods that detect 3D objects in a two-stage fashion, our method OPA-3D, as illustrated in Fig. 2, is a single-stage method which can be trained fully end-to-end. Given a single RGB image as input, we use a backbone to extract features, which are then fed into the Geometry Stream and Context Stream to perform respective prediction tasks.

Thereby, the first two heads in the Geometry Stream (GS) are designed for dense depth and depth-bounding box residuals (DBR) estimation. We show that the 3D bounding box parameters can be occlusion-aware recovered in closed-form with the pixel-wise aggregation of the output from the two heads. In the last bounding box projection head, we additionally predict the 2D projections of the BEV bounding box corners to enforce geometry-guided 2D-3D constraints for distance perception. Our Context Stream (CS) is identical to GUP-Net [10], which first predicts 2D bounding boxes and then uses ROI-Align [32] to extract object-centered features for 3D bounding box prediction. Compared to the CS, our GS gains more specific geometric information such as object shape and location. We use the consistency terms ($\mathscr{L}_{C-G}$ and $\mathscr{L}_{BPC}$) to ensure the promotion of both streams and that the feature encoder can learn rich features from both streams. On the otherside, CS is faster and more stable since all cues are leveraged and aggregated, while GS only harnesses depth-related geometric information. Thus, we only use CS during inference.

A. Context Stream

To localize the object in the image, we let our network predict a coarse 2D object center $\boldsymbol{C}_o$ in form of a heatmap, together with a 2D offset $\boldsymbol{\delta}_{2d} = \{\delta^u_{2d}, \delta^v_{2d}\}$ and 2D size h, w, so to obtain the final refined 2D bounding box using $\boldsymbol{C}_{2d} = \boldsymbol{C}_o + \boldsymbol{\delta}_{2d}$. To guide the model to extract object-centered features, the 2D bounding box indicated region-of-interest (ROI) features are cropped and resized with ROI-Align [32] and then concatenated with normalized coordinate map [33] to form the input for the 3D detection head. Finally, the 3D object center $\boldsymbol{C}^{CS}_{3d}$, heading θ and size H_{CS}, W_{CS}, L_{CS} are predicted from the ROI features and employed to calculate the respective 3D bounding box. Notice that in the following, we use the subscript 'CS' to denote predictions from the Context Stream. For supervision we follow the loss terms from GUP-Net [10], denoted as $\mathscr{L}_{CS}$.

B. Geometry Stream

In the section, we introduce the three predition heads in the Geometry Stream and their supervision.

Dense Depth Estimation: We follow Adabins [34] to adopt a transformer-based network to divide the depth range into adaptive context-guided bins, which turns dense depth estimation into a linear combination of Softmax scores. The estimated depth map $\mathbf{D}$ is $1 \times H/4 \times W/4$.

DBR: DBR was first introduced in GPV-Pose [2] for RGB-D based category-level pose estimation. It describes the displacement vectors from the observed object surface towards its projections on all of the 6 bounding box faces, as shown in Fig 3(a). We follow and develop DBR by leveraging a very small convolutional branch for regression of DBR $\mathbf{R}$ together with corresponding uncertainties $\mathbf{U}_P$, where $\mathbf{R}, \mathbf{U}_P \in \mathbb{R}^{6 \times \frac{H}{4} \times \frac{W}{4}}$.

BEV Bounding Box Projection: The output of this head is designed for enforcing several novel 2D-3D geometric constraints for improved distance perception (details see Section III-D). As shown in Fig. 3, for each BEV bounding box corner $\boldsymbol{P}_i$ with $i \in \{1, \ldots, 4\}$, $\boldsymbol{\rho}_i = \{\rho^x_i, \rho^y_i\}$ denotes the perspective projection

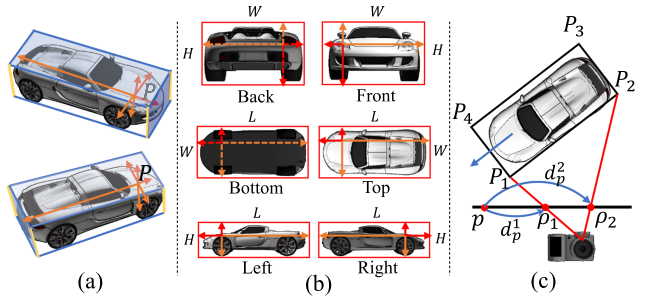

Fig. 3. Illustration of DBR and Bounding Box Projection. (a) illustrates Depth-Bounding Box Residuals (DBR) in two views. Given a visible point $\boldsymbol{P}$ on the object surface, we calculate its displacement vectors $\boldsymbol{R}$ to each of the bounding box faces along the direction of the corresponding plane normal (red arrow for positive direction while orange for negative). In (b), we demonstrate DBR from the view of the six bounding box faces. If $\boldsymbol{P}$ is invisible, we use dashed lines. H, W, L refer to 3D object height, width and length. (c) shows the bounding box projection from the bird's-eye view. Since we assume the y axis points straight downwards, from the bird-eye view the surrounding lines (yellow lines in (a)) of the object box project to 4 points $\boldsymbol{P_i}, i = 1, \ldots, 4$. We project $\boldsymbol{P_i}$ onto the image and yield $\boldsymbol{\rho_i}, i = 1, \ldots, 4$, We utilize the bounding box projection head to predict the 1D displacement vector d_p^i from each pixel $\boldsymbol{p}$ to the projections $\boldsymbol{\rho_i}$ along x-axis.

of $\boldsymbol{P}_i$ onto the image plane given the known camera intrinsics. Since the y axis is assumed to be vertical to the ground, we estimate the displacement d_p^i and uncertainty u_p^i from a pixel $\boldsymbol{p}$ to the location $\boldsymbol{\rho}_i$ merely along the x-axis. The final projected locations of the corners along the x-axis can be recovered as $\rho_i^x = \sum_{\boldsymbol{p}\in\Omega}(p^x + d_p^i) * exp(u_p^i) / \sum_{\boldsymbol{p}\in\Omega} exp(u_p^i)$, where Ω includes all pixels in the 2D object bounding box and $exp(*)$ is the exponential function that transforms uncertainty scores $\boldsymbol{u_p}$ into weights.

GS Supervision: Assuming the prediction follows the Laplacian distribution, one can formulate the supervision within an uncertainty framework using the Laplacian Aleatoric Uncertainty (LAU) loss [13], which is defined as,

$$\mathscr{L}_{LAU}(u, pr) = \frac{\sqrt{2}}{u} \left| pr - pr^{gt} \right| + log(u), \tag{1}$$

where pr is the prediction, pr^{gt} is the corresponded ground truth and u refers to the estimated uncertainty for this prediction.

Since we need to predict DBR and BEV bounding box along with their uncertainties, we supervise both prediction w.r.t the LAU loss. We use a standard $\mathscr{L}_1$ loss to train our dense depth prediction. The final loss for the geometry stream $\mathscr{L}_{GS}$ can be computed by adding up the individual contributions of the presented three heads.

C. 3D Bounding Box From Geometry Stream

GPV-Pose [2] leverages DBR to directly establish geometry-aware consistency with the predicted pose from RGB-D images. We extend this idea for RGB-only 3D object detection by introducing two novel points, 1) explicitly recovering the 3D bounding box from RGB input and 2) uncertainty-based occlusion handling strategy. 1) We explicitly recover the 3D object bounding box in closed-form via differentiable post-optimization, enabling direct supervision with the ground truth 3D object bounding box parameters for more effective feature extraction. As shown in Fig. 3, for each visible pixel $\boldsymbol{p}$, the DBR head outputs the 1D displacement vector R^j that maps the back-projected 3D point $\boldsymbol{P}$ to the corresponding projection $\boldsymbol{B}_P^j$ on each bounding box face j with $\boldsymbol{B}_P^j = \boldsymbol{P} + \boldsymbol{n_j^T} R^j$ and $j \in \{front, back, left, right, top, bottom\}$. $\boldsymbol{n_j}$ is the face normal. We additionally estimate an uncertainty U_P^j for each point P to measure the reliability of DBR. Since $\boldsymbol{B}_P^j$ is supposed to be on the bounding box face, we define the uncertainty-weighted plane fitting loss f_s for all $\boldsymbol{B}_P^j$ as follows,

$$f_s = \sum_{j\in Bb_s} \sum_{p\in\Omega} (1 - U_P^j) \left(\boldsymbol{n}_j^T \boldsymbol{B}_P^j - \boldsymbol{n}_j^T \boldsymbol{C}_{GS}^{3d} + \frac{S_{GS}^j}{2} \right)^2 + \sum_{j\in Bb_t} \sum_{p\in\Omega} (1 - U_P^j)(\boldsymbol{B}_P^j - \boldsymbol{C}_{GS}^{3d} + \frac{S_{GS}^j}{2})^2, \tag{2}$$

Thereby, Bb_s contains the $\{front, back, left, right\}$ faces of the 3D bounding box, and Bb_t contains the $\{top, bottom\}$ faces. $H_{GS}, W_{GS}, L_{GS}, \boldsymbol{C}_{GS}^{3d}$ are the target bounding box height, width, length and the 3D center needed to be recovered from the GS. $S_{GS}^j \in \{-L_{GS}, -W_{GS}, L_{GS}, W_{GS}, H_{GS}, -H_{GS}\}$ for corresponding $j \in \{Bb_s \cup Bb_t\}$, e.g. $S_{GS}^{back} = -W_{GS}$ in case of $j = back$. Since the target parameters $H_{GS}, W_{GS}, L_{GS}, \boldsymbol{C}_{GS}^{3d}$ are decoupled in (2), we can calculate the partial derivatives of f_s w.r.t. these variables $\{\frac{\partial f}{\partial W_{GS}}, \frac{\partial f}{\partial L_{GS}}, \frac{\partial f}{\partial H_{GS}}, \frac{\partial f}{\partial \boldsymbol{C}_{GS}^{3d}}\}$ and directly recover $H_{GS}, W_{GS}, L_{GS}, \boldsymbol{C}_{GS}^{3d}$ by setting the partial derivatives to be zero and solving the equations.

2) Trying to minimize (2) alone could lead U_P^j converge to 1.0. However, as a confidence term, U_P^j should have a geometric meaning. Therefore, we introduce the regularization term,

$$f_t = \alpha \sum_{p,j} (U_P^j)(W_{GS} - W_\xi)^2 + \beta \sum_{p,j} (U_P^j)(L_{GS} - L_\xi)^2 + \gamma \sum_{p,j} (U_P^j)(H_{GS} - H_\xi)^2, \tag{3}$$

where $\{W_\xi, L_\xi, H_\xi\}$ are prior object width, length and height. $\{\alpha, \beta, \gamma\}$ are small constant weights to balance each term. Thereby, we leverage $\{U_P^j, \alpha, \beta, \gamma\}$ to apply our occlusion handling strategy. In general, if the observations are reliable to recover the bounding box faces, we adopt the plane fitting loss in (2) to calculate face normal and center. However, when the observations are limited due to (self-)occlusions, we instead use prior knowledge to reconstruct the bounding box. Specifically, for the occluded bounding box faces in j, the corresponding uncertainty U_P^j is large, signalizing that it is hard to predict their bounding boxes directly. As shown in Fig. 3(b), when considering the *front* view, the *back* side is completely self-occluded by the object. Thus, without any prior knowledge, it is infeasible to infer the object's length from this view. It is worth noting that this also applies to occlusion caused by other objects. Therefore, forcing the network to estimate bounding boxes with these ill-posed observations leads to unstable training and, hence, converges in performance deterioration. We solve the optimization problem objective to $f = f_s + f_t$ to mitigate this problem. If U_P^j for a certain bounding box face j is very large (or in other words the prediction from the image is not reliable), the corresponding

term in f_s will be small. In this case, the 3D bounding box will be predicted close to the prior size $\{W_\xi, L_\xi, H_\xi\}$ to ensure the minimum of f_t.

To summarize, our optimization-based bounding box recovery explicitly describes the pipeline from image to depth and finally to the object bounding box. Since H_{GS}, W_{GS}, L_{GS}, $\boldsymbol{C}^{3d}_{GS}$ are solved in a differentiable manner, we can directly supervise the Geometry Stream with ground truth bounding box parameters besides the respective supervision on dense depth and DBR, enforcing the network to learn more effective features for 3D object detection.

D. Object Distance From BEV Bounding Box Projection

To enhance distance perception, geometric projection of object height is widely harnessed in literature [8], [10]. In this paper, we further extend this idea and propose object width and length related geometric priors to establish several novel 2D-3D constraints. As shown in Fig. 3, we have

$$\begin{cases} z_1\rho^x_1 = \mathbf{K}\boldsymbol{P}_1 = \mathbf{K}(\boldsymbol{C} + \frac{W}{2}\boldsymbol{n}_2 + \frac{L}{2}\boldsymbol{n}_1), \\ z_2\rho^x_2 = \mathbf{K}\boldsymbol{P}_2 = \mathbf{K}(\boldsymbol{C} + \frac{W}{2}\boldsymbol{n}_2 - \frac{L}{2}\boldsymbol{n}_1), \end{cases} \tag{4}$$

where z_1, z_2 are the z coordinates of P_1 and P_2, K denotes the camera intrinsic matrix and W, L, C are the target bounding box parameters. When solving the above equation, we can obtain the depth z_C of the object center as follows

$$z_C = \frac{f_x n_x + c_x n_z}{\rho^x_1 - \rho^x_2} L - \frac{n_z(\rho^x_1 + \rho^x_2)}{2(\rho^x_1 - \rho^x_2)} L - \frac{n_x}{2} W, \tag{5}$$

where f_x, c_x are camera focal length and its optical center along the x axis. Further, $\boldsymbol{n}_1 = [n_x, 0, n_z]^T$ is the normalized heading direction of the object (*c.f.* blue arrow in Fig. 3(c)). Note that we again only utilize the x coordinate of the BEV bounding box projections. Since the y axis points straight downwards, the projections of the surrounding lines (yellow lines in Fig. 3(a)) are parallel to the y axis on the image plane, thus regressing their x coordinate via pixel-wise aggregation is more reliable than exactly locating their 2D positions. We can build a 2D-3D constraint based on each BEV bounding box edge, resulting in up to 4 constraints.

E. Geometry Consistency Loss

In this section, we introduce how to build cross-stream consistency losses between the predictions from the geometry stream and the context stream.

Two-Stream Consistency: Notice that our proposed model is able to predict the 3D object bounding box from both presented streams. Whereas the geometry stream recovers the 3D bounding box via optimization of (2) and (3), which is fully differentiable, the context stream instead directly regresses all target parameters $\{H_{CS}, W_{CS}, L_{CS}, \boldsymbol{C}^{3d}_{CS}\}$ from the RGB image. As consequence, we can naturally enforce consistency between both streams according to

$$\begin{aligned} \mathscr{L}_{C-G} = \; & \|H_{GS} - H_{CS}\| + \|W_{GS} - W_{CS}\| + \|L_{GS} - L_{CS}\| \\ & + \left\|\boldsymbol{C}^{GS}_{3d} - \boldsymbol{C}^{CS}_{3d}\right\|. \end{aligned} \tag{6}$$

Bounding Box Projection Consistency: For each object we can generate up to 4 hypotheses for depth z_C following (5). Thus, given the predicted 3D length and width from the Context Stream, we define our consistency loss for object depth as

$$\mathscr{L}_{BPC} = \sum_{j \in BL} \omega_j \left\| z^j_c - C^z_{CS} \right\|, \tag{7}$$

where C^z_{CS} is the z coordinate of $\boldsymbol{C}^{3d}_{CS}$, BL contains all four corner pairs of BEV bounding box edges and ω_j measures the weight of each term. When considering $\boldsymbol{\rho}_1$ and $\boldsymbol{\rho}_2$, ω_{12} is defined as $\omega_{12} = v * [1 - exp(-k|\rho^x_1 - \rho^x_2|)]$, where k is a constant defined according to the heading direction, $v = 1$ if $\{\boldsymbol{P}_1, \boldsymbol{P}_2\}$ are visible and otherwise $v = 0$. This weight definition enforces that a visible projection with larger observation angle is preferred.

F. Overall Loss Function

The overall loss function is a combination of all four previously introduced loss terms,

$$\mathscr{L}_{Overall} = \mathscr{L}_{CS} + \mathscr{L}_{GS} + \mathscr{L}_{C-G} + \mathscr{L}_{BPC}. \tag{8}$$

Thereby, the loss terms will be automatically weighted from 0 to 1 with Hierarchical Task Learning (HTL) scheme as proposed in GUP-Net [10], which allows defining preliminary tasks for a specific loss. This specific loss will then start to be used when its preliminary task has been properly trained. The ranking of tasks in the Context Stream are identical to GUP-Net [10]. The Geometry Stream starts with dense depth prediction and BEV bounding box prediction, which in turn have no preliminary tasks. In contrast, DBR prediction requires dense depth estimation as preliminary task. The two consistency terms kick in once the predictions become stable. In essence, both consistency losses depend on the entire Context Stream. Moreover, the $\mathscr{L}_{C-G}$ also requires stable dense depth computation and DBR prediction, while the $\mathscr{L}_{BPC}$ relies on a steady BEV bounding box prediction.

IV. Experiments

A. Experimental Setup

Datasets: We provide detailed evaluation on the commonly-used **KITTI-3D** [1] and **nuScenes** [43] benchmarks. For KITTI-3D, we use the official KITTI-3D split of 7481 training and 7581 testing images. We further follow [6], [7], [10] and split the training data into a training and validation subset, each containing around 3.7 k samples, for ablation purposes. For nuScenes, since the dataset is quite large, we only use the images from front camera (∼1/6 of all training data) for training, and evaluate on the public validation split. On both datasets, we also leverage the provided LiDAR points to supervise depth prediction. In this paper, we mainly focus on three common categories, i.e. *car, pedestrian, cyclist*. Following previous works [10], [11], we conduct ablation studies mainly on the *car* category.

Implementation Details: The overall network is trained in an end-to-end manner. For a fair comparison with the state-of-the-art, we use DLA-34 [41] with a downsampling ratio of 4 as our backbone network for feature extraction. We train the network

TABLE I
COMPARISON WITH STATE-OF-THE-ART METHODS OF THE *CAR* CATEGORY ON KITTI-3D

Method	Extra Data	Test Split						Validation Split						Time (sec)
		$AP_{3D\|R40\|IOU\geq0.7}$			$AP_{BEV\|R40\|IOU\geq0.7}$			$AP_{3D\|R40\|IOU\geq0.7}$			$AP_{BEV\|R40\|IOU\geq0.7}$			
		Mod.	Easy	Hard	Mod.	Easy	Hard	Mod.	Easy	Hard	Mod.	Easy	Hard	
ROI-10D [33]	None	2.02	4.32	1.46	4.91	9.78	3.74	6.63	9.61	6.29	9.91	14.50	8.73	0.2
MonoPair [13]	None	9.99	13.04	8.65	14.83	19.28	12.89	12.30	16.28	10.42	18.17	24.12	15.76	0.057
DLE [8]	None	12.26	17.23	10.29	18.89	24.79	16.00	13.66	17.45	11.68	19.33	24.97	17.01	0.040
GUP-Net [10]	None	15.02	22.26	13.12	21.19	30.29	18.20	16.46	22.76	13.72	22.94	31.07	19.75	0.034
Monocon [11]	None	16.46	22.50	13.95	22.10	31.12	19.00	19.01	**26.33**	15.98	-	-	-	0.026
Monoflex [35]	None	13.89	19.94	12.07	19.75	28.23	16.89	17.51	23.64	14.83	-	-	-	0.030
DD3D [6]	LiDAR*	16.87	23.19	**14.36**	**23.41**	32.35	**20.42**	(16.92)	-	-	(24.77)	-	-	0.033
D4LCN [36]	Depth	11.72	16.65	9.51	16.02	22.51	12.55	-	-	-	-	-	-	0.20
MonoDTR [37]	LiDAR	15.39	21.99	12.73	20.38	28.59	17.14	18.57	24.52	15.51	25.35	33.33	21.68	0.037
CaDNN [38]	LiDAR	13.41	19.17	11.46	18.91	27.94	17.19	16.31	23.57	13.84	-	-	-	0.63
AutoShape [12]	CAD Model	14.17	22.47	11.36	20.08	30.66	15.59	14.65	20.09	12.07	-	-	-	0.050
MonoDistill [39]	LiDAR	16.03	22.97	13.60	22.59	31.87	19.72	18.47	24.31	15.76	25.40	33.09	**22.16**	0.040
Ours	LiDAR	**17.05**	**24.60**	14.25	22.53	**33.54**	19.22	**19.40**	24.97	**16.59**	**25.51**	**33.80**	22.13	0.040

Overall best results are in bold and the second best results are underlined. Methods are ranked according to the moderate AP for the IoU metric at a 0.7 threshold as given in the KITTI-3D benchmark. Note that on the test split DD3D utilizes deeper V2-99 [40] instead of DLA-34 as the feature extractor, and also pre-trains the network on the unreleased large-scale DD3D15M dataset, while OPA-3D only uses DLA-34 and pretrains on the clean split of KITTI-Depth dataset [6] for fair comparison with the SOTA. On the validation split, we report the result of DD3D with DLA-34 [41] as the backbone. Extra data of depth refers to the depth provided by a pretrained depth estimation network DORN [42], while LiDAR* means extra training data containing LiDAR point clouds beside the KITTI dataset.

TABLE II
COMPARISON WITH STATE-OF-THE-ART METHODS OF THE *PEDESTRIAN* AND *CYCLIST* ON KITTI-3D *TEST* SPLIT. $AP_{3D|R40|IOU\geq0.7}$ IS REPORTED

Method	Pedestrian			Cyclist		
	Mod.	Easy	Hard	Mod.	Easy	Hard
MonoPair [13]	6.68	10.02	5.53	2.12	3.79	1.83
DLE [8]	6.55	9.64	5.44	2.66	4.59	2.45
GUP-Net [10]	9.76	14.95	8.41	3.21	5.58	2.66
Monocon [11]	8.41	13.10	6.94	1.92	2.80	1.55
DD3D* [6]	**11.04**	**16.64**	**9.38**	**4.79**	**7.52**	**4.22**
D4LCN [36]	3.42	4.55	2.83	1.67	2.45	1.36
CaDNN [38]	8.14	12.87	6.76	3.41	7.00	3.30
Ours	10.49	15.65	8.80	3.45	5.16	2.86

Overall best results are in bold and the second best results are underlined.

TABLE III
ABLATION STUDY OF THE *CAR* CATEGORY ON KITTI-3D *VAL* SPLIT

Stage	Geometry Stream	3D $AP_{IoU\geq0.7}$		
		Mod.	Easy	Hard
A1	Baseline (GUP-Net)	15.76	22.01	13.11
A2	A1 + Dense Depth Prediction	17.31	22.76	14.58
B1	A2 + $\mathscr{L}_{C-G}$	18.10	24.81	15.27
B2	B1 + pretrained depth prediction	18.77	25.81	15.65
C	A1 + $\mathscr{L}_{BPC}$	18.00	24.51	15.15
D	B2 + C	19.40	24.97	16.59

with a batchsize of 8 using the AdamW [44] optimizer. The learning rate increased from $1e-5$ to $1.25e-3$ in the first 5 epochs using a cosine warmup. We train our networks for 200 epochs and decay the learning rate by 0.1 after the 110-th and 150-th epoch. We empirically set α, β, γ in Section III-C to $1e-3$ for occlusion handling. We adopt the HTL training scheme proposed in GUP-Net to balance each loss term, as explained under (8). The input images have been resized to 1280×380 on the KITTI-3D dataset, while 1600×896 on the Nuscenes dataset. We augment the input data by randomly cropping and flipping of the input images, as well as random brightness changes. The network has been pretrained with the KITTI-Depth dataset [1] by enabling only the dense depth prediction head for 25 epochs using the clean training split (removed the overlapping of the test images in KITTI-3D) [6].

B. Comparison With State-of-The-Art

Results for Car on KITTI-3D: We first compare with other state-of-the-art (SOTA) monocular 3D detectors on the KITTI-3D *test* set for the *car* category in Table I. It can be seen that OPA-3D exceeds all other SOTA methods and shows an obvious improvement (2.03%) over the GUP-Net [10] baseline on the moderate criteria. Similarly, we are also on par or better than Monocon [11], which is similar to GUP-Net [10], yet leverages several auxiliary tasks to improve 3D detection. In contrast to the auxiliary tasks in Monocon [11], our proposed geometry stream is able to utilize the depth information for training, thus achieving larger performance improvements.

Note that DD3D [6] leverages an in-house dataset with 15 M images for depth pre-training, together with a feature pyramid network based on the much larger V2-99 backbone [40]. Nonetheless, despite the use of this large amount of training data for depth, we can still slightly exceed DD3D on *test* and, especially, on *validation* with an 3D IoU of 19.40% *vs.* 16.92% with the same backbone.

Results for Pedestrian and Cyclist on KITTI-3D: Interestingly, we notice larger improvements over Monocon [11] for the *Pedestrian* and the *Cyclist* categories, hinting that its auxiliary tasks cannot be learned well with small objects. However, we also observed that OPA-3D is less effective on the *Cyclist* category. We attribute this drop in performance to the thin structure of cyclists. As our model is trained with depth from LiDAR point clouds, missing depth information for cyclists induces a weakened perception of depth.

Results on nuScenes: In Table IV, we compare OPA-3D with two baselines: CenterNet [45] and GUP-Net [10]. It can be seen clearly that our method outperforms competitors under all metrics. Note that we trained all approaches and used only partial training images in this experiment.

TABLE IV
COMPARISON WITH BASELINES ON NUSCENES BENCHMARK

Method	Car	Pedestrian	All		
	AP↑	AP↑	mAP↑	mATE↓	NDS↑
CenterNet [45]	0.092	0.066	0.037	0.912	0.097
GUP-Net [10]	0.161	0.135	0.054	0.865	0.141
Ours	**0.218**	**0.194**	**0.077**	**0.776**	**0.189**

Overall best results are in bold. We trained all three approaches on a partial nuScenes training set (only images from the front camera, about 1/6 of all training data), and evaluated on the full nuScenes validation set. We report the metrics used on nuScenes benchmark. ↑ means higher is better, while ↓ indicates lower is better.

C. Ablation Study

In Table III we present several ablation studies on the KITTI-3D *validation* split w.r.t. the *Car* category. Row 'A1' refers to the results of the baseline model [10]. In row 'A2,' we research the improvement introduced by learning dense depth prediction as an auxiliary task. Although we observed a large improvement, we show in the following that the learning tasks in our proposed Geometry Stream contribute more.

Effect of Two-Stream Consistency Loss: In this experiment, we disable the BEV bounding box head and $\mathscr{L}_{BPC}$. The GS recovers 3D bounding box from dense depth and DBR prediction, and the $\mathscr{L}_{C-G}$ ensures the consistency preditions from GS and CS. When removing this branch, we notice a drop on *Car* under moderate criteria (Table III B1 vs A0, also B1 vs A1), proving the effectiveness of our proposed $\mathscr{L}_{C-G}$. Since the $\mathscr{L}_{C-G}$ requires dense depth map prediction, we are able to easily pre-train the network on a depth dataset. After pre-training with the KITTI-Depth dataset [1], the results further increased by 0.61%. This improvement indicates that the training of $\mathscr{L}_{C-G}$ benefits from an additional depth dataset, which can be transformed from Lidar raw data without annotation effort.

Effect of BEV bounding box projection: We also want to ablate the usefulness of $\mathscr{L}_{BPC}$, by removing the dense depth prediction, DBR prediction and $\mathscr{L}_{C-G}$. The model again outperforms the baseline model clearly by 2.24% (Table III C vs A1), which supports the effectiveness of $\mathscr{L}_{BPC}$. As no depth information is used, the improvements by $\mathscr{L}_{BPC}$ are overall slightly less high than by $\mathscr{L}_{C-G}$.

Effect of combination of both consistency loss: Finally, by integrating both consistency losses into the baseline model, we can obtain the results for our proposed OPA-3D network. OPA-3D exceeds the models having only one consistency term. In summary, taking advantage of the additional depth training data and our geometric consistencies, OPA-3D is capable of yielding SOTA results.

D. Qualitative Results

We show some qualitative results in Fig. 4. OPA-3D predictions are visualized in green, while the prediction from the baseline model [10] are shown in blue. We also render the ground truth BEV bounding box in red. Further, in the first three columns, the car's front, back, or right side is self-occluded, while in the last two column, the car is occluded by other objects. As can be observed, OPA-3D is able to predict more accurate bounding boxes under (self-)occlusion.

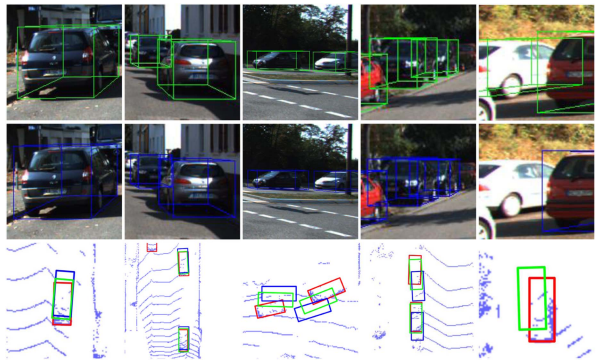

Fig. 4. We visualize example predictions on the KITTI-3D validation set. Our OPA-3D predictions are visualized in green, while the prediction from the baseline model [10] are shown in blue. We also render the ground truth BEV bounding box in red. We notice that OPA-3D produces more accurate predictions for the bounding box faces under self-occlusion as well as external occlusions.

V. CONCLUSION

In this paper we proposed a novel Occlusion-Aware Pixel-Wise Aggregation network OPA-3D, which jointly predicts the object bounding box from the geometry stream and the context stream, allowing for cross-branch consistency. We further propose to leverage the bounding box projection to establish several 2D-3D constraints to promote distance prediction. Extensive experiments on public benchmarks demonstrate that OPA-3D achieves state-of-the-art performance, whilst being able to achieve real time performance.

REFERENCES

[1] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.

[2] Y. Di et al., "GPV-pose: Category-level object pose estimation via geometry-guided point-wise voting," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 6781–6791.

[3] G. Zhai et al., "MonoGraspNet: 6-DoF grasping with a single RGB image," 2022, *arXiv:2209.13036*.

[4] R. Zhang, Y. Di, Z. Lou, F. Manhardt, F. Tombari, and X. Ji, "RBP-pose: Residual bounding box projection for category-level pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 655–672.

[5] Y. Nie, X. Han, S. Guo, Y. Zheng, J. Chang, and J. J. Zhang, "Total3Dunderstanding: Joint layout, object pose and mesh reconstruction for indoor scenes from a single image," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 55–64.

[6] D. Park, R. Ambrus, V. Guizilini, J. Li, and A. Gaidon, "Is pseudo-LiDAR needed for monocular 3D object detection?," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 3142–3152.

[7] X. Shi, Q. Ye, X. Chen, C. Chen, Z. Chen, and T.-K. Kim, "Geometry-based distance decomposition for monocular 3D object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 15172–15181.

[8] X. Ma et al., "Delving into localization errors for monocular 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 4721–4730.

[9] A. Simonelli, S. R. Bulo, L. Porzi, M. López-Antequera, and P. Kontschieder, "Disentangling monocular 3D object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1991–1999.

[10] Y. Lu et al., "Geometry uncertainty projection network for monocular 3D object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 3111–3121.

[11] X. Liu, N. Xue, and T. Wu, "Learning auxiliary monocular contexts helps monocular 3D object detection," in *Proc. AAAI Conf. Artif. Intell.*, vol. 36, no. 2, 2022.

[12] Z. Liu, D. Zhou, F. Lu, J. Fang, and L. Zhang, "Autoshape: Real-time shape-aware monocular 3D object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 15641–15650.

[13] Y. Chen, L. Tai, K. Sun, and M. Li, "Monopair: Monocular 3D object detection using pairwise spatial relationships," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 12093–12102.

[14] H. Chen, Y. Huang, W. Tian, Z. Gao, and L. Xiong, "Monorun: Monocular 3D object detection by reconstruction and uncertainty propagation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 10379–10388.

[15] T. Roddick, A. Kendall, and R. Cipolla, "Orthographic feature transform for monocular 3D object detection," 2018, *arXiv:1811.08188*.

[16] Y. Wang, W.-L. Chao, D. Garg, B. Hariharan, M. Campbell, and K. Q. Weinberger, "Pseudo-LiDAR from visual depth estimation: Bridging the gap in 3D object detection for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8445–8453.

[17] X. Weng and K. Kitani, "Monocular 3D object detection with pseudo-LiDAR point cloud," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops*, 2019.

[18] X. Ma, Z. Wang, H. Li, P. Zhang, W. Ouyang, and X. Fan, "Accurate monocular 3D object detection via color-embedded 3D reconstruction for autonomous driving," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 6851–6860.

[19] R. Qian et al., "End-to-end pseudo-LiDAR for image-based 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 5881–5890.

[20] X. Ma, S. Liu, Z. Xia, H. Zhang, X. Zeng, and W. Ouyang, "Rethinking pseudo-LiDAR representation," in *Proc. Eur. Conf. Comput. Vis.*, Cham, Switzerland, 2020, pp. 311–327.

[21] X. Wang, W. Yin, T. Kong, Y. Jiang, L. Li, and C. Shen, "Task-aware monocular depth estimation for 3D object detection," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 12257–12264.

[22] J. K. Murthy, G. S. Krishna, F. Chhaya, and K. M. Krishna, "Reconstructing vehicles from a single image: Shape priors for road scene understanding," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 724–731.

[23] F. Chabot, M. Chaouch, J. Rabarisoa, C. Teuliere, and T. Chateau, "Deep manta: A coarse-to-fine many-task network for joint 2D and 3D vehicle analysis from monocular image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2040–2049.

[24] J. A. Ansari, S. Sharma, A. Majumdar, J. K. Murthy, and K. M. Krishna, "The earth ain't flat: Monocular reconstruction of vehicles on steep and graded roads from a moving camera," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 8404–8410.

[25] J. Ku, A. D. Pon, and S. L. Waslander, "Monocular 3D object detection leveraging accurate proposals and shape reconstruction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 11867–11876.

[26] A. Kundu, Y. Li, and J. M. Rehg, "3D-RCNN: Instance-level 3D object reconstruction via render-and-compare," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3559–3568.

[27] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, "3D bounding box estimation using deep learning and geometry," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 7074–7082.

[28] H. M. Choi, H. Kang, and Y. Hyun, "Multi-view reprojection architecture for orientation estimation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop*, 2019, pp. 2357–2366.

[29] L. Liu, J. Lu, C. Xu, Q. Tian, and J. Zhou, "Deep fitting degree scoring network for monocular 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1057–1066.

[30] A. Naiden, V. Paunescu, G. Kim, B. Jeon, and M. Leordeanu, "Shift R-CNN: Deep monocular 3D object detection with closed-form geometric constraints," in *Proc. IEEE Int. Conf. Image Process.*, 2019, pp. 61–65.

[31] B. Li, W. Ouyang, L. Sheng, X. Zeng, and X. Wang, "GS3D: An efficient 3D object detection framework for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1019–1028.

[32] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2961–2969.

[33] F. Manhardt, W. Kehl, and A. Gaidon, "Roi-10D: Monocular lifting of 2D detection to 6D pose and metric shape," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2069–2078.

[34] S. F. Bhat, I. Alhashim, and P. Wonka, "Adabins: Depth estimation using adaptive bins," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 4009–4018.

[35] Y. Zhang, J. Lu, and J. Zhou, "Objects are different: Flexible monocular 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 3289–3298.

[36] M. Ding et al., "Learning depth-guided convolutions for monocular 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2020, pp. 1000–1001.

[37] K.-C. Huang, T.-H. Wu, H.-T. Su, and W. H. Hsu, "MonoDTR: Monocular 3D object detection with depth-aware transformer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 4012–4021.

[38] C. Reading, A. Harakeh, J. Chae, and S. L. Waslander, "Categorical depth distribution network for monocular 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 8555–8564.

[39] Z. Chong et al., "Monodistill: Learning spatial features for monocular 3D object detection," 2022, *arXiv:2201.10830*.

[40] Y. Lee and J. Park, "Centermask: Real-time anchor-free instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 13906–13915.

[41] F. Yu, D. Wang, E. Shelhamer, and T. Darrell, "Deep layer aggregation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2403–2412.

[42] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2002–2011.

[43] H. Caesar et al., "nuscenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11621–11631.

[44] I. Loshchilov and F. Hutter, "Fixing weight decay regularization in adam," 2017, *arXiv:1711.05101*.

[45] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," in 2019, *arXiv:1904.07850*.