This ICCV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Is Pseudo-Lidar needed for Monocular 3D Object detection?

Dennis Park* Rareș Ambruș* Vitor Guizilini Jie Li Adrien Gaidon
Toyota Research Institute
firstname.lastname@tri.global

Abstract

Recent progress in 3D object detection from single images leverages monocular depth estimation as a way to produce 3D pointclouds, turning cameras into pseudo-lidar sensors. These two-stage detectors improve with the accuracy of the intermediate depth estimation network, which can itself be improved without manual labels via large-scale self-supervised learning. However, they tend to suffer from overfitting more than end-to-end methods, are more complex, and the gap with similar lidar-based detectors remains significant. In this work, we propose an end-to-end, single stage, monocular 3D object detector, DD3D, that can benefit from depth pre-training like pseudo-lidar methods, but without their limitations. Our architecture is designed for effective information transfer between depth estimation and 3D detection, allowing us to scale with the amount of unlabeled pre-training data. Our method achieves state-of-the-art results on two challenging benchmarks, with 16.34% *and* 9.28% *AP for Cars and Pedestrians (respectively) on the KITTI-3D benchmark, and* 41.5% *mAP on NuScenes.*

1. Introduction

Detecting and accurately localizing objects in 3D space is crucial for many applications, including robotics, autonomous driving, and augmented reality. Hence, monocular 3D detection is an active research area [42, 57, 39, 69], owing to its potentially wide-ranging impact and the ubiquity of cameras. Leveraging exciting recent progress in depth estimation [11, 13, 15, 16, 31], pseudo-lidar detectors [69, 40, 57] first use a pre-trained depth network to compute an intermediate pointcloud representation, which is then fed to a 3D detection network. The strength of pseudo-lidar methods is that they monotonically improve with depth estimation quality, e.g., thanks to large scale training of the depth network on raw data.

*equal contribution
Code: https://github.com/TRI-ML/dd3d

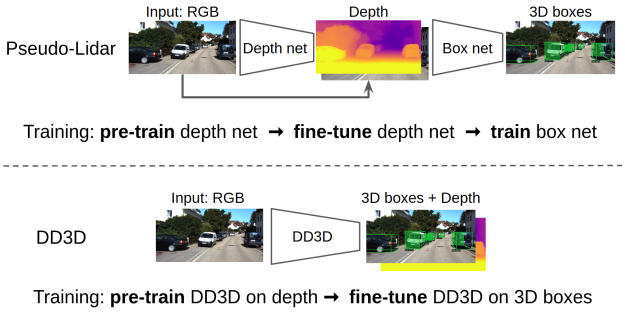

Figure 1: We introduce a single-stage 3D object detector, **DD3D**, that combines the best of both pseudo-lidar methods (scaling with depth pre-training) and end-to-end methods (simplicity and generalization performance). Our detector features a simple training protocol of depth pre-training and detection fine-tuning, compared to pseudo-lidar methods that require an additional depth fine-tuning step and tend to overfit to depth errors.

However, regressing depth from single images is inherently an ill-posed inverse problem. Consequently, errors in depth estimation account for the major part of the gap between pseudo-lidar and lidar-based detectors, a problem compounded by generalization issues that are not fully understood yet [56]. Simpler end-to-end monocular 3D detectors [3, 39] seem like a promising alternative, although they do not enjoy the same scalability benefits of unsupervised pre-training due to their single stage nature.

In this work, we aim to get the best of both worlds: the scalability of pseudo-lidar with raw data and the simplicity and generalization performance of end-to-end 3d detectors. To this end, our **main contribution** is a novel fully convolutional single-stage 3D detection architecture, **DD3D** (for Dense Depth-pre-trained 3D Detector), that can effectively leverage monocular depth estimation for pre-training (see Figure 1). Using a large corpus of unlabeled raw data, we show that DD3D scales similarly to pseudo-lidar methods, and that depth pre-training improves upon pre-training on large labeled 2D detection datasets like COCO [37], even with the same amount of data.

Our method sets a new state of the art on the task of monocular 3D detection on KITTI-3D [14] and nuScenes [5] with significant improvements compared to previous state-of-the-art methods. The simplicity of its training procedure and its end-to-end optimization allows for effective use of large-scale depth data, leading to impressive multi-class detection accuracy.

2. Related work

Monocular 3D detection. A large body of work in image-based 3D detection builds upon 2D detectors, and aims to *lift* them to 3D using various cues from object shapes and scene geometry. These priors are often injected by aligning 2D keypoints with their projective 3D counterparts [6, 1, 2, 21, 50], or by learning a low-dimensional shape representation [28, 42]. Other methods try to leverage geometric consistency between 2D and 3D structures, formulating the inference as a constrained optimization problem [44, 38, 45, 12, 43, 24, 27, 34]. These methods commonly use additional data (e.g. 3D CAD models, instance segmentation), or assume rigidity of objects. Inspired by Lidar-based methods [8, 68, 30], another line of work employs view transformation (i.e. birds-eye-view) to overcome the limitations of perspective *range* view, e.g. occlusion or variability in size [25, 54, 58]. These methods often require precise camera extrinsics, or are accurate only in close vicinity.

End-to-end 3D detectors. Alternatively, researchers have attempted to directly regress 3D bounding boxes from CNN features [57, 39, 73, 23, 3, 9]. Typically, these approaches extend standard 2D detectors (single-stage [52, 73] or two-stage [53]) by adding *heads* that predict various 3D cuboid parameterizations . The use of depth-aware convolution or dense 3D anchors [10, 51] enabled higher accuracy. Incorporating uncertainty in estimating the 3D box (or its depth) has also been shown to greatly improve detection accuracy [57, 9, 56]. In [57], the authors proposed a disentangled loss for 3D box regression that helps stabilize training. DD3D also falls in the end-to-end 3D detector category, however, our emphasis is on learning a good depth representation via large-scale self-supervised pre-training on raw data, which leads to robust 3D detection.

Pseudo-Lidar methods. Starting with the pioneering work of [65], these methods leverage advances in monocular depth estimation and train Lidar-based detectors on the resulting pseudo pointcloud, producing impressive results [67, 49, 66, 41]. Recent methods have improved upon [65] by correcting the monocular depth with sparse Lidar readings [69], decorating the pseudo pointcloud with colors [41], using 2D detection to segment foreground pointcloud regions [47, 67, 40], and structured sparsification of the monocular pointcloud [61]. Recently, [56] has shown a bias in the PL results on the representative KITTI-3D [14] benchmark, while this paradigm remains the state-of-the-art. Multiple researchers showed that *inaccurate depth estimation* is a major source of error in PL methods [56, 64]. In this work, we build our reference PL method based on [47] and [40] to investigate the benefits of using large-scale depth data and compare it with DD3D.

Monocular depth estimation. Estimating per-pixel depth is a key task for both DD3D (as a pre-training task) and PL (as the first of a two-stage pipeline). It is itself a thriving research area: the community has pushed toward accurate dense depth prediction via supervised [35, 29, 19, 48, 26, 18, 32, 13, 31] as well as self-supervised [46, 72, 60, 16, 15, 62] methods. We note that supervised monocular depth training used in this work require no annotations from human, allowing us to scale our methods to a large amount of raw data.

3. Dense depth pre-training for 3D detection

Given a single image and its camera intrinsics matrix as input, the goal of monocular 3D detection is to generate a set of multi-class 3D bounding boxes relative to camera coordinates. During inference, DD3D does not require any additional data, such as per-pixel depth estimates, 2D bounding boxes, segmentations, or 3D CAD models. DD3D is also *camera-aware*: it scales the depth of putative 3D boxes according to the camera intrinsics.

3.1. Architecture

DD3D is a fully convolutional single-stage network that extends FCOS [59] to perform 3D detection and dense depth prediction. The architecture (see Figure 2) is composed of a backbone network and three sub-networks (or *head*s) that are shared among all multi-scale features. The backbone takes an RGB image as input, and computes convolutional features at different scales. As in [59], we adopt a feature pyramid network (FPN) [36] as the backbone.

Three head networks are applied to each feature map produced by the backbone, and perform independent prediction tasks. The classification module predicts object category. It produces C real values, where C is the number of object categories. The 2D box module produces class-agnostic bounding boxes and center-ness by predicting 4 offsets from the feature location to the sides of each bounding box and a scalar associated with center-ness. We refer the readers to [59] for more details regarding the 2D detection architecture.

3D detection head. This head predicts 3D bounding boxes and per-pixel depth. It takes FPN features as input, and applies four 2D convolutions with 3×3 kernels that generate 12 real values for each feature location. These are decoded into 3D bounding box, per-pixel depth map, and 3D prediction confidence, as described below:

- $\mathbf{q} = (q_w, q_x, q_y, q_z)$ is the quaternion representation of

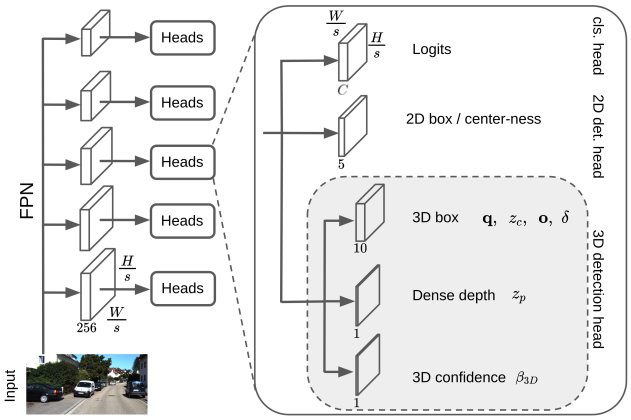

Figure 2: **DD3D** is a fully convolutional single-stage architecture that performs monocular 3D detection and dense depth prediction. To maximize reuse of pre-trained features, the inference of *dense depth* and *3D bounding box* share most of the parameters. The heads are shared among all FPN layers.

allocentric orientation [42, 57, 39] of the 3D bounding box. It is normalized and transformed to an egocentric orientation [42]. Note that we predict orientations with the full 3 degrees of freedom.

- $z_{\{c,p\}}$ represent the depth predictions. z_c is decoded to the z-component of 3D bounding box centers and therefore only associated with foreground features, while z_p is decoded to the monocular depth to the closest surface and is associated with every pixel. To decode them to metric depth, we *unnormalized* them using per-level parameters as follows:

$$d = \frac{c}{p} \cdot (\sigma_l \cdot z + \mu_l), \tag{1}$$

$$p = \sqrt{\frac{1}{f_x^2} + \frac{1}{f_y^2}}, \tag{2}$$

where $z \in \{z_c, z_p\}$ are network output, $d \in \{d_c, d_p\}$ are predicted depths, (σ_l, μ_l) are learnable scaling factors and offsets defined for each FPN level, p is the *pixel size* computed from the focal lengths, f_x and f_y, and c is a constant.

We note that this design of using camera focal lengths endows DD3D with *camera-awareness*, by allowing us to infer the depth not only from the input image, but also from the pixel size. We found that this is particularly useful for stable training. Specifically, when the input image is resized during training, we keep the ground-truth 3D bounding box unchanged, but modify the camera *resolution* as follows:

$$\mathbf{K} = \begin{bmatrix} r_x & r_y & 1 \end{bmatrix} \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{3}$$

where r_x and r_y are resize rates, and $\mathbf{K}$ is the new camera intrinsic matrix that is used in Eqs. 2 and 4. Finally, $\{z_p\}$ collectively represent low-resolution versions of the dense depth maps computed from each FPN features. To recover the full resolution of the dense depth maps, we apply bilinear interpolation to match the size of the input image.

- $\mathbf{o} = (\Delta_u, \Delta_v)$ represent offsets from the feature location to the 3D bounding box center projected onto the camera plane. This is decoded to the 3D center via *unprojection*:

$$\mathbf{C} = \mathbf{K}^{-1} \begin{bmatrix} u_b + \alpha_l \Delta_u \\ v_b + \alpha_l \Delta_v \\ 1 \end{bmatrix} d_c, \tag{4}$$

where (u_b, v_b) is the feature location in image space, α_l is a learnable scaling factor assigned to each FPN level.

- $\delta = (\delta_W, \delta_H, \delta_H)$ represents the deviation in the size of the 3D bounding box from the class-specific canonical size, i.e. $\mathbf{s} = (W_0 e^{\delta_W}, H_0 e^{\delta_H}, D_0 e^{\delta_D})$. As in [55], (W_0, H_0, D_0) is the canonical box size for each class, and is pre-computed from the training data as its average size.

- β_{3D} represents the confidence of the 3D bounding box prediction [57]. It is transformed into a probability: $p_{3D} = (1 + e^{-\beta_{3D}})^{-1}$, and multiplied by the class probability computed from the classification head [59] to account for the relative confidence to the 2D confidence. The adjusted probability is used as the final score for the candidate.

3.2. Losses

We adopt the classification loss and 2D bounding box regresssion loss from FCOS [59]:

$$\mathcal{L}_{2D} = \mathcal{L}_{reg} + \mathcal{L}_{cls} + \mathcal{L}_{ctr}, \tag{5}$$

where the 2D box regression loss L_{reg} is the IOU loss [70], the classification loss L_{cls} is the binary focal loss (i.e. one-vs-all), and the center-ness loss L_{ctr} is the binary cross entropy loss. For 3D bounding box regression, we use the *disentangled* L1 loss described in [57], i.e.

$$\mathcal{L}_{3D}(\mathbf{B}^*, \hat{\mathbf{B}}) = \frac{1}{8} ||\mathbf{B}^* - \hat{\mathbf{B}}||_1, \tag{6}$$

where $\mathbf{B}^*$ and $\hat{\mathbf{B}}$ are the 8 vertices of ground-truth and candidate 3D boxes. This loss is replicated 4 times by using only one of the predicted 3D box components (orientation, projected center, depth, and size), while replacing other three with their ground-truth values.

Also as in [57], we adopt the self-supervised loss for 3D confidence which uses the error in 3D box prediction to compute a surrogate target for 3D confidence (relative to the 2D confidence):

$$p^*_{\mathsf{3D}} = e^{-\frac{1}{T}\mathcal{L}_{\mathsf{3D}}(\mathbf{B}^*, \hat{\mathbf{B}})}, \tag{7}$$

where T is the temperature parameter. The confidence loss $\mathcal{L}_{conf}$ is the binary cross entropy between p_{3D} and p^*_{3D}. In summary, the total loss of DD3D is defined as follows:

$$\mathcal{L}_{\mathsf{DD}} = \mathcal{L}_{\mathsf{2D}} + \mathcal{L}_{\mathsf{3D}} + \mathcal{L}_{conf}. \tag{8}$$

3.3. Depth pre-training

During pre-training, we use per-pixel depth predictions from all FPN levels, i.e. $\{z_p\}$. We consider pixels that have valid ground-truth depth from the sparse Lidar pointclouds projected onto the camera plane, and compute L1 distance from the predicted values:

$$\mathcal{L}^l_{depth} = ||\mathbf{D}^* - \hat{\mathbf{D}}_l||_1 \odot \mathbf{M}, \tag{9}$$

$$\mathcal{L}_{depth} = \sum_l \mathcal{L}^l_{depth}, \tag{10}$$

where $\mathbf{D}^*$ is the ground-truth depth map, $\hat{\mathbf{D}}_l$ is the predicted depth map from the l-th level in FPN (i.e. interpolated z_p), and $\mathbf{M}$ is the binary indicator for valid pixels. We observed using all FPN levels in the objective, rather than e.g. using only the highest resolution features, enables stable training, especially when training *from scratch*. We also observed that the L1 loss yields stable training with large batch sizes and high-resolution input, compared to SILog loss [11, 31] that is popular in monocular depth estimation literature.

We note that the two paths in DD3D from the input image to the 3D bounding box and to the dense depth prediction differ only in the last 3×3 convolutional layer, and thus share nearly all parameters. This allows for effective transfer from the pre-trained representation to the target task.

While pre-training, the camera-awareness of DD3D allows us to use camera intrinsics that are substantially different from the ones of the target domain, while still enjoying effective transfer. Specifically, from Eqs. 1 and 2, the error in depth prediction, z, caused by the difference in resolution between the two domains is corrected by the difference in *pixel size*, p.

4. Pseudo-Lidar 3D detection

An alternative way to utilize a large set of image-Lidar frames is to adopt the *Pseudo-Lidar (PL)* paradigm and aim to improve the depth network component using the large scale data. PL is a two-stage method: first, given an image it applies a monocular depth network to predict per-pixel depth. The dense depth map is transformed into a 3D point cloud, and then a Lidar-based 3D detector is used to predict 3D bounding boxes. The modularity of PL methods enables us to quantify the role of improved depth predictors brought by a large-scale image-LiDAR dataset (see the supplementary material for additional details.)

Monocular depth estimation. The aim of monocular depth estimation is to compute the depth $\hat{D} = f_D(I(p))$ for each pixel $p \in I$. Similarly to Eq. 9, given the ground truth depth measurement $\mathbf{D}^*$ acquired from Lidar pointclouds, we define a loss by the error between the predicted and the ground-truth depth. Here we instead use the SILog loss [11, 31], which yields better performance than L1 for PackNet.

Network architecture. As the depth network, we use PackNet [17], a state-of-the-art monocular depth prediction architecture which uses packing and unpacking blocks with 3D convolutions. By avoiding feature sub-sampling, PackNet recovers fine structures in the depth map with high precision; moreover, PackNet has been shown to generalize better thanks to its increased capacity [17].

3D detection. To predict 3D bounding boxes from the input image and the estimated depth map, we follow the method proposed by [47, 40]. We first convert the estimated depth map into a 3D pointcloud similarly to Eq. 4, and concatenate each 3D point with the corresponding pixel values. This results in a 6D tensor encompassing colors along with 3D coordinates. We use an off-the-shelf 2D detector to identify proposal regions in input images, and apply a 3D detection network to each RoI region of the 6-channel image to produce 3D bounding boxes.

Backbone, detection head and 3D confidence. We follow [40] and process each RoI with a ResNet-18 [20] backbone that uses Squeeze-and-Excitation layers [22]. As the RoI contains both objects as well as background pixels, the resulting features are filtered via foreground masks computed based on the associated RoI depth map [41]. The detection head follows [40] and operates in 3 distance ranges, producing one bounding box for each range. The final output is then selected based on the mean depth of the input RoI. Following [57, 56], we modify the detection head to also output a 3D confidence value γ per detection, which is linked to the 3D detection loss.

Loss function. The 3D regression loss [47] is defined as:

$$\mathcal{L}^{\mathsf{PL}}_{\mathsf{3D}} = \mathcal{L}_{center} + \mathcal{L}_{size} + \mathcal{L}_{heading} + \mathcal{L}_{corners}. \tag{11}$$

In addition, we define a loss that links the predicted 3D confidence γ with the 3D bounding box coordinates loss [57] using a Binary Cross Entropy (BCE) formulation with target $\hat{\gamma} = e^{-\mathcal{L}_{corners}}$. The final PL 3D detection loss is: $\mathcal{L}_{\mathsf{PL}} = \mathcal{L}^{\mathsf{PL}}_{\mathsf{3D}} + \mathcal{L}^{\mathsf{PL}}_{conf}$.

5. Experimental Setup

5.1. Datasets

KITTI-3D. The KITTI-3D detection benchmark [14] consists of urban driving scenes with 8 object classes. The benchmark evaluates 3D detection accuracy on three classes (Car, Pedestrian, and Cyclist) using two average precision (AP) metrics computed with class-specific thresholds on intersection-over-union (IoU) of 3D bounding boxes or Bird-Eye-View (2D) bounding boxes. We refer to these metrics as *3D AP* and *BEV AP*. We use the revised $\text{AP}|_{R_{40}}$ metrics [55]. The training set consists of 7481 images, the test set of 7518 images. The objects in the test set are organized into three partitions according to their difficulty level (easy, moderate, hard), and are evaluated separately. For the analysis in Section 6.2, we follow the common practice of splitting the training set into 3712 and 3769 images [7], and report validation results on the latter. We refer to these splits as KITTI-3D *train* and KITTI-3D *val*.

nuScenes. The nuScenes 3D detection benchmark [5] consists of 1000 multi-modal videos with 6 cameras covering the full 360-degree field of view. The videos are split into 700 for training, 150 for validation, and 150 for testing. The benchmark requires to report 3D bounding boxes of 10 object classes over 2Hz-sampled video frames. The evaluation metric, *nuScenes detection score* (*NDS*), is computed by combining the detection accuracy (*mAP*) computed over four different thresholds on center distance with five *true-positive* metrics. We report NDS and mAP, along with the three true-positive metrics that concern 3D detection, i.e. *ATE*, *ASE*, and *AOE*.

KITTI-Depth. We use the KITTI-Depth dataset [14] to fine-tune the depth networks of our PL models. It contains over 93 thousands depth maps associated with the images in the KITTI *raw* dataset. The standard monocular depth protocol [71, 15, 16] is to use the *Eigen* splits [11]. However, as described in [56], up to a third of its training images overlap with KITTI-3D images, leading to biased results for PL models. To avoid this bias, we generate a new split by removing training images that are geographically close (i.e. within 50m) to any of the KITTI-3D images. We denote this split by *Eigen-clean* and use it to fine-tune the depth networks of our PL models.

DDAD15M. To pre-train DD3D and our PL models, we use an in-house dataset that consists of 25000 multi-camera videos of urban driving scenes. DDAD15M is a larger version of DDAD [16]: it contains high-resolution Lidar sensors to generate pointclouds and 6 cameras synchronized with 10 Hz scans. Most videos are 10-second long, which amounts to approximately 15M image frames in total. Unless noted differently, we use the entire dataset for pre-training.

5.2. Implementation detail

DD3D. We use V2-99 [33] extended to an FPN as the backbone network. When pre-training DD3D, we first initialize the backbone with parameters pre-trained on the 2D detection task using the COCO dataset [37], and perform the pre-training on dense depth prediction using the DDAD15M dataset.

We use a test-time augmentation by resizing and flipping the input images. We observed 2.3% gain in "Car" BEV AP on KITTI *val*, but no improvement on the nuScenes validation set. All metrics on DD3D in Section 6.2 are averages over 4 training runs. We observed the variance over the runs to be small, i.e. $0.5 \sim 1.2\%$ BEV AP.

PL. When training PackNet [16], we use only the front camera images of DDAD15M to pre-train PackNet, and train until convergence. We then fine-tune the network on KITTI *Eigen-clean* split for 5 epochs. For more details on training DD3D and PL, please refer to the supplementary material.

6. Results

6.1. Benchmark evaluation

In this section, we evaluate DD3D on the KITTI-3D and nuScenes benchmarks. DD3D is pre-trained on DDAD15M, and then fine-tuned on the training set of each dataset. We also evaluate PL on KITTI-3D. Its depth network is pre-trained on DDAD15M and fine-tuned on KITTI *Eigen-clean*, and its detection network is trained on the KITTI-3D *train*.

KITTI-3D. In Table 1 we compare the accuracy of DD3D to state-of-the-art methods on the KITTI-3D benchmark. DD3D achieves a significant improvement over all methods with $\mathbf{16.34}\%$ 3D AP on *Moderate Cars*, which amounts to a $\mathbf{23}\%$ improvement from the previous best method (13.25%). Qualitative visualization is presented in Figure 3. In Table 4 we evaluate DD3D also on the *Pedestrian* and *Cyclist* classes. DD3D outperforms all other approaches on the *Pedestrian* category, with an $\mathbf{80.5}\%$ improvement from the previous best method ($\mathbf{9.30}\%$ vs 5.14% 3D AP). On *Cyclist* DD3D achieves the second best result, reducing the gap to [27] that uses ground-truth pointclouds to train a per-instance pointcloud reconstruction module and a two-stage regression network to refine 3D object proposals.

We next report the accuracy of our PL detector in Table 1. The accuracy of our PL detector is on par with state-of-the-art methods, however it performs significantly worse than DD3D (13.05% vs. 16.34%). We will discuss this result in the context of generalizability of PL methods in Section 6.2.

nuScenes. In Table 2 we compare DD3D with other monocular methods reported on the nuScenes detection bench-

Methods	Car					
	BEV AP			3D AP		
	Easy	Med	Hard	Easy	Med	Hard
ROI-10D [42]	9.78	4.91	3.74	4.32	2.02	1.46
GS3D [34]	8.41	6.08	4.94	4.47	2.90	2.47
MonoGRNet [50]	18.19	11.17	8.73	9.61	5.74	4.25
MonoPSR [27]	18.33	12.58	9.91	10.76	7.25	5.85
MonoPL [67]	-	-	-	10.76	7.50	6.10
SS3D [23]	16.33	11.52	9.93	10.78	7.68	6.51
MonoDIS (single) [57]	17.23	13.19	11.12	10.37	7.94	6.40
M3D-RPN [3]	21.02	13.67	10.23	14.76	9.71	7.42
SMOKE [39]	20.83	14.49	12.75	14.03	9.76	7.84
MonoPair [9]	19.28	14.83	12.89	13.04	9.99	8.65
AM3D [41]	25.03	17.32	14.91	16.50	10.74	9.52
PatchNet [40]	22.97	16.86	14.97	15.68	11.12	10.17
RefinedMPL [61]	28.08	17.60	13.95	18.09	11.14	8.96
D4LCN [10]	22.51	16.02	12.55	16.65	11.72	9.51
Kinematic3D [4]	26.99	17.52	13.10	19.07	12.72	9.17
MonoDIS (multi) [55]	24.45	19.25	16.87	16.54	12.97	11.04
Demystifying [56]	-	-	-	**23.66**	13.25	11.23
PL	28.87	18.57	15.74	20.78	13.05	10.66
DD3D	**30.98**	**22.56**	**20.03**	23.22	**16.34**	**14.20**

Table 1: **KITTI-3D *test* set evaluation on *Car*.** We report $\text{AP}|_{R_{40}}$ metrics. **Bold** and underline denote the best and second best results.

Metric	AP[%]↑	ATE[m]↓	ASE[1-IoU]↓	AOE[rad]↓	NDS↑
CenterNet*	33.8	0.66	0.26	0.63	0.40
AIML-ADL*	35.2	0.70	0.26	0.39	0.43
DHNet*	36.3	0.67	0.26	0.40	0.44
PGDepth*	37.0	0.66	0.25	0.49	0.43
P.Pillars [30]	31.0	**0.52**	0.29	0.50	0.45
MonoDIS [55]	30.4	0.74	0.26	0.55	0.38
FCOS3D [63]	35.8	0.69	**0.25**	0.45	0.43
DD3D	**41.8**	0.57	**0.25**	**0.37**	**0.48**

Table 2: **nuScenes detection *test* set evaluation.** We present summary metrics of the benchmark. * denotes results reported on the benchmark that do not have associated publications at the time of writing. The underline denotes the second best published approach. Note that PointPillars [30] is a Lidar-based detector.

mark. The metrics are averages over all 10 categories of the dataset. DD3D outperforms all other methods with a $\mathbf{17}\%$ improvement in mAP compared to the previously best published method [63] ($\mathbf{41.8}\%$ vs. 35.8% mAP) as well as a 13% improvement compared to the best (unpublished) method. We note that DD3D even surpasses PointPillars [30], which is a Lidar based detector.

In Table 5 we compare per-class accuracy of DD3D with other methods on the three major categories, with various thresholds on distance. In general, DD3D offers significant improvements across all categories and thresholds. In particular, DD3D performs significantly better on the stricter criteria: comparing to the previous best method [63], the relative improvements averaged across the 3 object classes are $\mathbf{103.7}\%$ and $\mathbf{35.5}\%$ for 0.5m and 1.0m thresholds, respectively.

Methods	BEV AP		
	Easy	Mod	Hard
MonoDIS (single) [57]	18.5	12.6	10.7
M3D-RPN [3]	20.9	15.6	11.9
Kinematic3D [4]	27.8	19.7	15.1
DD3D with DLA-34 backbone			
DD3D (with DLA-34)	33.5	26.0	22.6
- DDAD15M	26.8	20.2	16.7
- COCO	31.7	24.0	20.3
DD3D with V2-99 backbone			
DD3D (with V2-99)	37.0	29.4	25.4
- DDAD15M	25.5	18.7	15.5
Pseudo-Lidar			
PL (DDAD15M → KITTI)	43.5	30.1	25.4
- KITTI	25.8	19.1	16.4
- DDAD15M	27.6	19.2	16.4

Table 3: **Ablative analysis of DD3D and PL on the KITTI-3D *validation* set.** As pre-training methods, KITTI denotes dense depth prediction using *Eigen-clean* split of KITTI-Depth dataset, DDAD15M denotes dense depth prediction using DDAD15M dataset, and COCO denotes initial pre-training phase on 2D detection. The right arrow (→) denotes sequential pre-training phases. We report BEV $\text{AP}|_{R_{40}}$ metrics on *Car*. The analysis is presented in Section 6.2.

6.2. Analysis

Here we provide a detailed analysis of DD3D, focusing on the role of depth pre-training and comparison with our PL approach. After pre-training the two models under various settings, we fine-tune them on KITTI-3D *train* on the 3D detection task and report AP metrics on KITTI-3D *val*.

6.2.1 Is depth-pretraining effective?

Ablating large-scale depth pre-training. We first ablate the effect of dense depth pre-training on the DDAD15M dataset, with results reported in Table 3. When we omit the depth pre-training and directly fine-tune DD3D with the DLA-34 backbone on the detection task, we observe a $\mathbf{5.3}\%$ loss in *Car Moderate BEV AP*. In contrast, when we instead remove the initial COCO-pretraining (i.e. pre-training on DDAD15M *from-scratch*), we observe a relatively small loss, i.e. 2.0%. For the larger backbone of V2-99, the effect of removing the depth pre-training is even more significant, i.e. $-\mathbf{10.7}\%$.

Dense depth prediction as pre-training task. To better quantify the effect of depth pre-training, we design a

Methods	Pedestrian						Cyclist					
	BEV AP			3D AP			BEV AP			3D AP		
	Easy	Med	Hard	Easy	Med	Hard	Easy	Med	Hard	Easy	Med	Hard
OFTNet [54]	1.28	0.81	0.51	0.63	0.36	0.35	0.36	0.16	0.15	0.36	0.16	0.15
SSD3D [23]	2.48	2.09	1.61	2.31	1.78	1.48	3.45	1.89	1.44			
M3D-RPN [3]	5.65	4.05	3.29	4.92	3.48	2.94	1.25	0.81	0.78	0.94	0.65	0.47
MonoPSR [27]	7.24	4.56	4.11	6.12	4.00	3.30	**9.87**	**5.78**	**4.57**	**8.37**	**4.74**	**3.68**
MonoDis (multi) [55]	9.07	5.81	5.09	7.79	5.14	4.42	1.47	0.85	0.61	1.17	0.54	0.48
DD3D	**15.90**	**10.85**	**8.05**	**13.91**	**9.30**	**8.05**	3.20	1.99	1.79	2.39	1.52	1.31

Table 4: **KITTI-3D *test* set *Pedestrian* and *Cyclist* results.**

Methods	Car [%] ↑				Pedestrian [%] ↑				Bicycle [%] ↑			
	0.5m	1.0m	2.0m	4.0m	0.5m	1.0m	2.0m	4.0m	0.5m	1.0m	2.0m	4.0m
CenterNet*	20.0	45.8	68.0	80.6	7.9	26.7	49.6	65.9	4.3	13.8	28.4	36.2
AIML-ADL*	14.2	36.8	58.5	71.0	9.6	30.8	54.6	69.4	5.2	21.6	37.3	46.2
DHNet*	15.2	37.9	59.4	71.5	10.5	31.7	55.7	69.9	5.6	24.2	38.9	48.0
PGDepth*	17.0	43.6	67.2	80.3	9.1	31.0	53.9	69.1	7.6	24.1	40.1	49.2
MonoDis (single) [57]	10.7	37.5	69.0	**85.7**	-	-	-	-	-	-	-	-
MonoDis (multi) [55]	10.6	36.1	65.0	80.5	6.7	30.0	48.5	64.7	4.4	17.5	32.8	43.9
FCOS3D [63]	15.3	43.8	68.9	81.7	8.7	30.3	52.9	67.1	7.9	25.0	39.2	47.1
DD3D	**30.2**	**59.7**	**77.4**	84.1	**18.7**	**42.4**	**61.9**	**70.2**	**15.7**	**32.6**	**45.2**	**50.0**

Table 5: **Detailed results on the nuScenes *test* set.** We report AP metrics on *Car*, *Pedestrian*, and *Bicycle* with varying thresholds on distance. The underline denotes the second best published approach.

Pre-train task	BEV AP			3D AP		
	Car	Ped.	Cyclist	Car	Ped.	Cyclist
COCO	20.2	7.8	**3.8**	13.9	6.0	3.1
+ nusc-det	20.5	7.9	**3.8**	14.0	6.0	3.0
+ nusc-depth	**21.8**	**8.5**	**3.8**	**15.2**	**6.6**	**3.3**

Table 6: **Depth vs. 2D detection as pre-training task.** Starting from a common initial model (COCO), we pre-train DD3D using two different tasks using the same set of images. *nusc-det* denotes 2D detection task, *nusc-depth* denotes dense prediction task, both using nuScenes images. We report the accuracy on the KITTI-3D *validation* set.

controlled experiment to further isolate its effect (Table 6). Starting from a single set of initial parameters (COCO), we consider two tasks for pre-training DD3D, 2D detection and dense depth prediction. The two pre-training phases use a *common* set of images that are annotated with both 2D bounding box labels and sparse depth maps obtained by projecting Lidar pointcloud. To further ensure that the comparison is fair, we applied the same number of training steps and batch size (15K and 512). The data used for this pre-training experiment is composed of 136571 images from the nuScenes [5] training set.

The experiment shows that, even with the smaller scale of pre-training data compared to DDAD15M (137K vs. 15M), the dense depth pre-training yields a notable difference in the downstream 3D detection accuracy (21.8% vs. 20.5% BEV AP on *Car Moderate*).

How does depth-pretraning scale? We next investigate how the unsupervised depth pre-training scales with respect to the size of pre-training data (Figure 4). For this experiment, we randomly subsample 1K, 5K, and 10K videos from DDAD15M to generate a total of 4 pre-training splits (including the complete dataset), that consist of 0.6M, 3M, 6M and 15M images respectively. Importantly, the downsampled splits contain fewer images as well as less diversity, as we downsample from the set of *videos*. We pre-train both DD3D and the PackNet of PL on each split, and subsequently train the detectors on KITTI-3D *train*. We note that DD3D and PL perform similarly at each checkpoint, and continue to improve as more depth data is used in pre-training, at least up to 15M images.

6.2.2 The limitations of PL methods.

In-domain depth fine-tuning of PL. Recall that training our PL 3D detector entails fine-tuning of the depth network in the target domain (i.e. *Eigen-clean*), after it is pre-trained on DDAD15M. We ablate the effect of the in-domain fine-tuning step in training the PL detector (Table 3). Note that in this experiment, the depth network (PackNet) is trained only on the pre-training domain (DDAD15M) and is directly applied on KITTI-3D without any adaptation. In this setting, we observe a significant loss in performance

Figure 3: **Qualitative visualization of DD3D detections.** The first two rows are from the KITTI-3D dataset and the last row is from nuScenes. None of the images were seen during training.

(30.1% $\rightarrow$ 19.1% BEV AP). This indicates that in-domain fine-tuning of the depth network is crucial for PL-style 3D detectors. This poses a practical hurdle in using PL methods, since one has to curate a separate in-domain dataset specifically for fine-tuning the depth network. We argue that this is the main reason that PL methods are exclusively reported on KITTI-3D, which is accompanied by KITTI-Depth as a convenient source for in-domain fine-tuning. This is unnecessary for end-to-end detectors, as shown in Tables 1 and 3.

Limited generalizability of PL. With the large-scale depth pre-training and in-domain fine-tuning, our PL detector offers excellent performance on KITTI-3D *val* (Table 3). However, the gain from the depth pre-training is not transferred to the benchmark results (Table 1). While the loss in accuracy between KITTI-3D *val* and the *test* sets is consistent with other methods [57, 4, 3], PL suffers particularly more (30.1% $\rightarrow$ 18.6% BEV AP), when compared to other methods including DD3D (29.4% $\rightarrow$ 22.56%). This reveals a subtle issue in the generalization of PL that is not well understood yet. We argue that the in-domain fine-tuning overfits to some image statistics that cause the performance gap between KITTI-3D *val* and KITTI-3D *test*, particularly more than other methods.

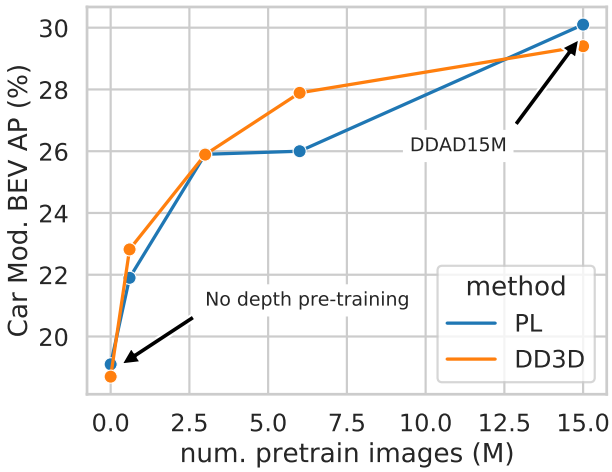

Figure 4: **DD3D and PL are pre-trained with varying sizes of depth data**, fine-tuned for 3D detection on the KITTI-3D *train*, and evaluated on KITTI-3D *val*. We pre-train the two methods on 4 subsets of DDAD15M with 0.6M, 3M, 6M, and 15M images.

7. Conclusion

We proposed *DD3D*, an end-to-end single-stage 3D object detector that enjoys the benefit of Pseudo Lidar methods (i.e. scaling accuracy using large-scale depth data), but without its limitation (i.e. impractical training, issues in generalization). This is enabled by pre-training DD3D using a large-scale depth dataset, and fine-tuning on the target task end-to-end. DD3D achieves excellent accuracy on two challenging 3D detection benchmarks.

References

[1] Junaid Ahmed Ansari, Sarthak Sharma, Anshuman Majumdar, J. Krishna Murthy, and K. Madhava Krishna. The earth ain't flat: Monocular reconstruction of vehicles on steep and graded roads from a moving camera. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2018, Madrid, Spain, October 1-5, 2018*, pages 8404–8410. IEEE, 2018. 2

[2] Ivan Barabanau, Alexey Artemov, Evgeny Burnaev, and Vyacheslav Murashkin. Monocular 3d object detection via geometric reasoning on keypoints. *arXiv preprint arXiv:1905.05618*, 2019. 2

[3] Garrick Brazil and Xiaoming Liu. M3d-rpn: Monocular 3d region proposal network for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9287–9296, 2019. 1, 2, 6, 7, 8

[4] Garrick Brazil, Gerard Pons-Moll, Xiaoming Liu, and Bernt Schiele. Kinematic 3d object detection in monocular video. In *European Conference on Computer Vision*, pages 135–152. Springer, 2020. 6, 8

[5] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar 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, 5, 7

[6] Florian Chabot, Mohamed Chaouch, Jaonary Rabarisoa, Celine Teuliere, and Thierry Chateau. Deep manta: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2

[7] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. 5

[8] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2

[9] Yongjian Chen, Lei Tai, Kai Sun, and Mingyang Li. Monopair: Monocular 3d object detection using pairwise spatial relationships. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12093–12102, 2020. 2, 6

[10] Mingyu Ding, Yuqi Huo, Hongwei Yi, Zhe Wang, Jianping Shi, Zhiwu Lu, and Ping Luo. Learning depth-guided convolutions for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 1000–1001, 2020. 2, 6

[11] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014. 1, 4, 5

[12] Jiaojiao Fang, Lingtao Zhou, and Guizhong Liu. 3d bounding box estimation for autonomous vehicles by cascaded geometric constraints and depurated 2d detections using 3d results, 2019. 2

[13] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2002–2011, 2018. 1, 2

[14] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012. 2, 5

[15] Clément Godard, Oisin Mac Aodha, and Gabriel Brostow. Digging into self-supervised monocular depth estimation. *arXiv preprint arXiv:1806.01260v3*, 2018. 1, 2, 5

[16] Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien Gaidon. 3d packing for self-supervised monocular depth estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2, 5

[17] Vitor Guizilini, Rares Ambrus, Sudeep Pillai, Allan Raventos, and Adrien 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. 4

[18] Vitor Guizilini, Rui Hou, Jie Li, Rares Ambrus, and Adrien Gaidon. Semantically-guided representation learning for self-supervised monocular depth. In *International Conference on Learning Representations (ICLR)*, April 2020. 2

[19] Vitor Guizilini, Jie Li, Rares Ambrus, Sudeep Pillai, and Adrien Gaidon. Robust semi-supervised monocular depth estimation with reprojected distances. In *Conference on Robot Learning (CoRL)*, October 2019. 2

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[21] Tong He and Stefano Soatto. Mono3d++: Monocular 3d vehicle detection with two-scale 3d hypotheses and task priors. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 8409–8416. AAAI Press, 2019. 2

[22] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 4

[23] Eskil Jörgensen, Christopher Zach, and Fredrik Kahl. Monocular 3d object detection and box fitting trained end-to-end using intersection-over-union loss. *arXiv preprint arXiv:1906.08070*, 2019. 2, 6, 7

[24] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *Proceedings of the IEEE international conference on computer vision*, pages 1521–1529, 2017. 2

[25] Youngseok Kim and Dongsuk Kum. Deep learning based vehicle position and orientation estimation via inverse perspective mapping image. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 317–323, 2019. 2

[26] Marvin Klingner, Jan-Aike Termöhlen, Jonas Mikolajczyk, and Tim Fingscheidt. Self-supervised monocular depth estimation: Solving the dynamic object problem by semantic guidance. *arXiv preprint arXiv:2007.06936*, 2020. 2

[27] Jason Ku, Alex D Pon, and Steven L Waslander. Monocular 3d object detection leveraging accurate proposals and shape reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11867–11876, 2019. 2, 5, 6, 7

[28] Abhijit Kundu, Yin Li, and James M. Rehg. 3d-rcnn: Instance-level 3d object reconstruction via render-and-compare. In *CVPR*, 2018. 2

[29] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *International Conference on 3D Vision (3DV)*, pages 239–248, 2016. 2

[30] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar 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. 2, 6

[31] Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. From big to small: Multi-scale local planar guidance for monocular depth estimation. 2019. 1, 2, 4

[32] Jae-Han Lee, Minhyeok Heo, Kyung-Rae Kim, and Chang-Su Kim. Single-image depth estimation based on fourier domain analysis. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 330–339, 2018. 2

[33] Youngwan Lee and Jongyoul Park. Centermask: Real-time anchor-free instance segmentation. 2020. 5

[34] Buyu Li, Wanli Ouyang, Lu Sheng, Xingyu Zeng, and Xiaogang Wang. Gs3d: An efficient 3d object detection framework for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1019–1028, 2019. 2, 6

[35] Bo Li, Chunhua Shen, Yuchao Dai, Anton Van, and Mingyi He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 201, pages 1119–1127, 2015. 2

[36] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 2

[37] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 1, 5

[38] Lijie Liu, Jiwen Lu, Chunjing Xu, Qi Tian, and Jie Zhou. Deep fitting degree scoring network for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[39] Zechen Liu, Zizhang Wu, and Roland Tóth. Smoke: single-stage monocular 3d object detection via keypoint estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 996–997, 2020. 1, 2, 3, 6

[40] Xinzhu Ma, Shinan Liu, Zhiyi Xia, Hongwen Zhang, Xingyu Zeng, and Wanli Ouyang. Rethinking pseudo-lidar representation. In *European Conference on Computer Vision*, pages 311–327. Springer, 2020. 1, 2, 4, 6

[41] Xinzhu Ma, Zhihui Wang, Haojie Li, Pengbo Zhang, Wanli Ouyang, and Xin Fan. Accurate monocular 3d object detection via color-embedded 3d reconstruction for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6851–6860, 2019. 2, 4, 6

[42] Fabian Manhardt, Wadim Kehl, and Adrien Gaidon. Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2069–2078, 2019. 1, 2, 3, 6

[43] Hee Min Choi, Hyoa Kang, and Yoonsuk Hyun. Multi-view reprojection architecture for orientation estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019. 2

[44] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7074–7082, 2017. 2

[45] Andretti Naiden, Vlad Paunescu, Gyeongmo Kim, Byeong-Moon Jeon, and Marius Leordeanu. Shift r-cnn: Deep monocular 3d object detection with closed-form geometric constraints. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 61–65, 2019. 2

[46] Sudeep Pillai, Rares Ambrus, and Adrien Gaidon. Superdepth: Self-supervised, super-resolved monocular depth estimation. In *arXiv preprint arXiv:1810.01849*, 2018. 2

[47] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 918–927, 2018. 2, 4

[48] Xiaojuan Qi, Renjie Liao, Zhengzhe Liu, Raquel Urtasun, and Jiaya Jia. Geonet: Geometric neural network for joint depth and surface normal estimation. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 283–291, 2018. 2

[49] Rui Qian, Divyansh Garg, Yan Wang, Yurong You, Serge Belongie, Bharath Hariharan, Mark Campbell, Kilian Q Weinberger, and Wei-Lun Chao. End-to-end pseudo-lidar for image-based 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5881–5890, 2020. 2

[50] Zengyi Qin, Jinglu Wang, and Yan Lu. Monogrnet: A geometric reasoning network for monocular 3d object localization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8851–8858, 2019. 2, 6

[51] Zengyi Qin, Jinglu Wang, and Yan Lu. Triangulation learning network: From monocular to stereo 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[52] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 2

[53] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. 2

[54] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. *arXiv preprint arXiv:1811.08188*, 2018. 2, 7

[55] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel Lopez Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection: From single to multi-class recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 3, 5, 6, 7

[56] Andrea Simonelli, Samuel Rota Bulò, Lorenzo Porzi, Peter Kontschieder, and Elisa Ricci. Demystifying pseudo-lidar for monocular 3d object detection. *arXiv preprint arXiv:2012.05796*, 2020. 1, 2, 4, 5, 6

[57] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1991–1999, 2019. 1, 2, 3, 4, 6, 7, 8

[58] Siddharth Srivastava, Frédéric Jurie, and Gaurav Sharma. Learning 2d to 3d lifting for object detection in 3d for autonomous vehicles. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2019, Macau, SAR, China, November 3-8, 2019*, pages 4504–4511. IEEE, 2019. 2

[59] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9627–9636, 2019. 2, 3

[60] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. Demon: Depth and motion network for learning monocular stereo. In *IEEE Conference on computer vision and pattern recognition (CVPR)*, volume 5, page 6, 2017. 2

[61] Jean Marie Uwabeza Vianney, Shubhra Aich, and Bingbing Liu. Refinedmpl: Refined monocular pseudolidar for 3d object detection in autonomous driving. *arXiv preprint arXiv:1911.09712*, 2019. 2, 6

[62] Sudheendra Vijayanarasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Katerina Fragkiadaki. Sfm-net: Learning of structure and motion from video. *arXiv preprint arXiv:1704.07804*, 2017. 2

[63] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Fcos3d: Fully convolutional one-stage monocular 3d object detection. *arXiv preprint arXiv:2104.10956*, 2021. 6, 7

[64] Xinlong Wang, Wei Yin, Tao Kong, Yuning Jiang, Lei Li, and Chunhua Shen. Task-aware monocular depth estimation for 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020. 2

[65] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q. Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[66] Yan Wang, Xiangyu Chen, Yurong You, Li Erran Li, Bharath Hariharan, Mark Campbell, Kilian Q Weinberger, and Wei-Lun Chao. Train in germany, test in the usa: Making 3d object detectors generalize. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11713–11723, 2020. 2

[67] Xinshuo Weng and Kris 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. 2, 6

[68] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[69] Yurong You, Yan Wang, Wei-Lun Chao, Divyansh Garg, Geoff Pleiss, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving. *arXiv preprint arXiv:1906.06310*, 2019. 1, 2

[70] Jiahui Yu, Yuning Jiang, Zhangyang Wang, Zhimin Cao, and Thomas Huang. Unitbox: An advanced object detection network. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 516–520, 2016. 3

[71] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1851–1858, 2017. 5

[72] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *arXiv preprint arXiv:1805.09817*, 2018. 2

[73] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 2