The Devil is in the Task: Exploiting Reciprocal Appearance-Localization Features for Monocular 3D Object Detection

Zhikang Zou[1]* Xiaoqing Ye[1]* Liang Du[2]* Xianhui Cheng[4]* Xiao Tan[1]
Li Zhang[3] Jianfeng Feng[2] Xiangyang Xue[4] Errui Ding[1]

[1] Baidu Inc., China
[2] Institute of Science and Technology for Brain-Inspired Intelligence, Fudan University,
MOE Key Laboratory of Computational Neuroscience and Brain-Inspired Intelligence, Fudan University
[3] School of Data Science, Fudan University
[4] School of Computer Science, Fudan University

Abstract

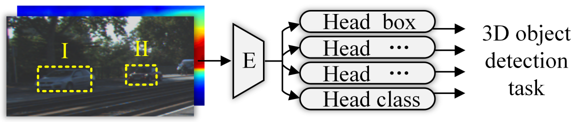
Low-cost monocular 3D object detection plays a fundamental role in autonomous driving, whereas its accuracy is still far from satisfactory. In this paper, we dig into the 3D object detection task and reformulate it as the sub-tasks of object localization and appearance perception, which benefits to a deep excavation of reciprocal information underlying the entire task. We introduce a Dynamic Feature Reflecting Network, named DFR-Net, which contains two novel standalone modules: (i) the Appearance-Localization Feature Reflecting module (ALFR) that first separates task-specific features and then self-mutually reflects the reciprocal features; (ii) the Dynamic Intra-Trading module (DIT) that adaptively realigns the training processes of various sub-tasks via a self-learning manner. Extensive experiments on the challenging KITTI dataset demonstrate the effectiveness and generalization of DFR-Net. We rank 1st among all the monocular 3D object detectors in the KITTI test set (till March 16th, 2021). The proposed method is also easy to be plug-and-play in many cutting-edge 3D detection frameworks at negligible cost to boost performance. The code will be made publicly available.

1. Introduction

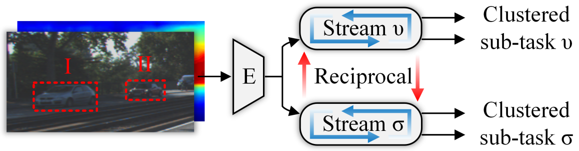
Building on the promising progress achieved by 2D object detection in recent years [27, 26], vision- and LiDAR-based 3D object detection have received increasing attention from both industry and academia due to their critical roles in outdoor autonomous driving [14] and indoor robotic navigation. 3D object detectors based on expensive LiDAR sensors [35, 29, 10, 11] have been widely developed and excelled in 3D object detection, whereas a much cheaper alternative, i.e., monocular 3D object detection, remains an open and challenging research field.

Monocular 3D object detectors can be roughly divided

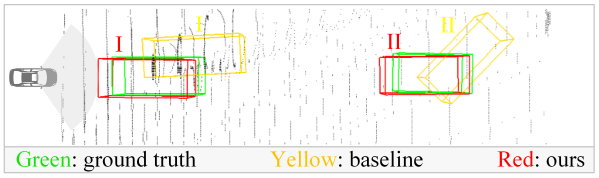
*indicates equal contribution



(a) Baseline: tasks with independent heads



(b) Ours: clustered tasks with self-mutual feature reflecting



(c) Predictions of the baseline method and ours (BEV)

Figure 1. Comparison of the baseline method (D^{4}LCN [8]) and our proposed DFR-Net. (a) The baseline method first uses the encoder ("E") to extract RGB and predicted depth features and then adopts independent heads to decode the shared features for 3D detection tasks. (b) In our method, we first cluster the sub-tasks with common characteristics to build up separated task streams and then exploit the reciprocal information between the streams via self-mutual feature reflecting. (c) The ground truth, the prediction of the baseline method, and our prediction are shown in the bird's-eye view (BEV) pseudo LiDAR for better visualization.

into three categories according to different input data representations: RGB image-based, pseudo LiDAR-based, and depth-assisted image-based methods: (i) RGB image-based methods aim to leverage geometry constraints [21] or semantic knowledge [4] to explore 2D-3D geometric consistency for recovering 3D location and dimension. However, the performance is still far from satisfactory due to the lack of reliable depth prior and the variance of the object scale caused by perspective projection. (ii) Pseudo LiDAR-based methods [32, 33] utilize depth estimation to reconstruct point clouds from image pixels. Afterwards cutting-edge LiDAR-based approaches such as [23, 29] can be directly borrowed. Recent works [32, 33, 37] have demonstrated the effectiveness of pseudo LiDAR-based methods. Nevertheless, due to the inaccurate depth prediction, a lack of RGB context as well as the inherent difference between real- and pseudo-LiDAR, the performance is limited. (iii) Depth-assisted methods such as [8, 30] focus on the integration strategy of RGB and depth features, whereas the network is unable to resolve the inferior 3D localization due to the mis-estimated depth map. In other words, the performance relies heavily on the quality of depth maps.

Humans can get some hints about 3D information even from monocular cues because the brain has the capability to utilize reciprocal visual information from different perception tasks [9], e.g., object localization and appearance perception (classification). For example, if we know the category and size of an object, we will know how far away the object is. On the other hand, if we know the localization and fuzzy scale of a distant or occluded unknown object, we may accordingly guess its category.

Inspired by humans' object perception system, we introduce a novel dynamic feature reflecting network for monocular 3D object detection, named DFR-Net. A novel appearance-localization feature reflecting module (ALFR) is designed, where 3D detection tasks are divided into two categories, the appearance perception tasks and the object localization tasks. Distinct tasks are sent to one of the two streams accordingly to delve into the task-specific features within each task, where reciprocal features between two categories self-mutually reflect. Here the terminology "reflect" denotes task-wise implicit feature awareness and correlation. To further optimize the multi-task learning, we propose a dynamic intra-trading module, named DIT, which realigns the training process of two sub-tasks in a self-learning manner. Figure 1 shows the comparison of independent heads (baseline D^{4}LCN [8]) and our proposed DFR-Net. DFR-Net exploits and leverages reciprocal appearance-localization features for 3D reasoning and achieves superior performance. The proposed module is demonstrated to be effective on various image-based and depth-assisted image-based backbone networks (e.g., M3D-RPN [2] and D^{4}LCN [8]).

Our main contributions are summarized as follows:

- We introduce a simple yet effective dynamic feature reflecting network (DFR-Net) for monocular 3D object detection, which exploits the reciprocal information underlying the task, allowing the sub-tasks to benefit from each other to alleviate the ill-posed problem of monocular 3D perception.
- We present an appearance-localization feature reflecting module (ALFR) that first separates two task streams and then self-mutually reflects the subtask-aware features.
- We investigate a dynamic intra-trading module (DIT) that reweights different task losses to realign the multi-task training process in a self-learning manner.
- We achieve new state-of-the-art monocular 3D object detection performance on the KITTI benchmark. The method can be plug-and-play in many other frameworks to boost the performance at negligible cost.

2. Related work

Image-based detection Single image-based 3D object detection is much more challenging compared with stereo- and LiDAR-based detection because monocular 3D perception is an ill-posed problem, thus spatial information is not sufficient and precise enough for reliable object localization. Some pioneering works [5, 4, 34, 24, 1, 21, 15, 28] attempted to use RGB and auxiliary information, e.g., semantic knowledge and geometry consistency, to alleviate this issue. Brazil et al. introduced M3D-RPN [2] that exploits the geometric relationship between 2D and 3D perspectives through sharing anchors and classification targets. Inspired by the key point-based 2D object detector CenterNet [38], RTM3D [22] estimates the nine projected key points of a 3D bounding box in image space to construct the geometric relationship of 3D and 2D to recover the 3D object information, whereas MonoPair [7] leverages spatial relationships between paired objects to effectively detect occluded objects. Simonelli et al. proposed an alternative method MonoDIS [31] that leverages a novel disentangling transformation for 2D and 3D detection losses, which conducts disentangled training of the losses from heterogeneous sets of parameters to make the training process of monocular 3D detection more stable. However, this approach ignores to exploit and utilize the reciprocal information underlying the entire task. Recently, researchers endeavor to adopt extra information to introduce much richer representations for the task. Monocular sequence-based Kinematic3D [3] efficiently utilizes 3D kinematic motion from videos to boost 3D detection performance. Reading et al. introduced CaDDN [25] that learns categorical depth distribution for each pixel in 2D space to project contextual information to the appropriate depth interval in 3D space.

Pseudo depth-assisted detection Monocular depth estima-

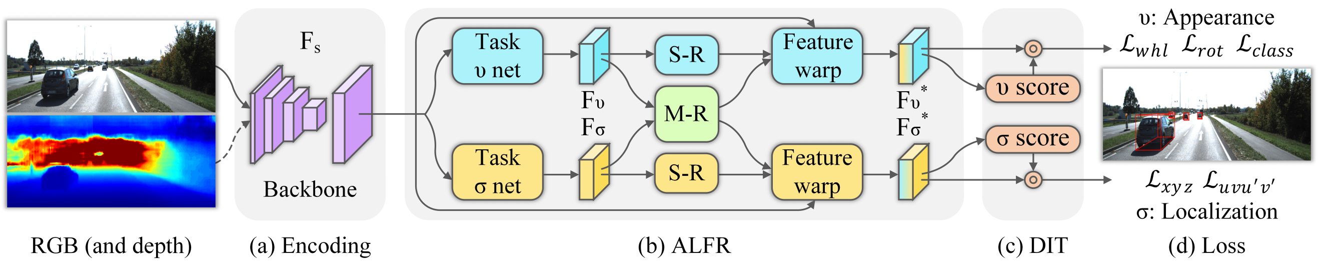


Figure 2. Schematic illustration of the proposed DFR-Net. (a) The RGB image and pseudo depth map (optional) are encoded as the shared feature F_s. (b) The proposed ALFR module first separates the feature into F_υ and F_σ and then self-reflects (S-R) and mutual-reflects (M-R) the features and finally warps the features to get $F_\upsilon{}^*$ and $F_\sigma{}^*$. (c) the DIT module dynamically predicts scores from two warped features and realigns the training process according to the scores in a self-learning manner. (d) the losses for the task υ (appearance perception) and the task σ (object localization).

tion [12] opens up an alternative and effective way for accurate monocular 3D object detection [20]. D^{4}LCN [8] introduced a local convolutional 3D object detection network, where depth maps were regarded as the guidance to learn local dynamic depthwise-dilated kernels for images, while the local dilated convolution can not fully capture the object context in the condition of perspective projection and occlusion. Different from these methods, pseudo LiDAR-based methods such as [32, 37] converting depth maps into artificial point clouds and adopting off-the-shelf LiDAR-based algorithms. Ma et al. observed that the efficacy of pseudo-LiDAR representation comes from the coordinate transformation and proposed PatchNet [18] that organizes the pseudo-LiDAR data as the image representation. However, most of the abovementioned methods are not fast enough for real-time applications.

3. Methodology

3.1. Pipeline overview

The goal of this work is to dig deep into the 3D detection problem and reformulate it as object localization and appearance perception sub-tasks that can benefit from each other via reciprocal feature reflecting. Given a single image, 3D object detection aims to predict categories, 3D locations, dimensions as well as direction. Inspired by 2D object detection, categories of an instance can be inferred from the appearance features. Besides, since the dimensions of a certain object type usually have similar sizes, the rough dimension information can also be deduced from the appearance features. On the contrary, 3D locations vary along with the positions within the image. Motivated by this observation, we reformulate the task and propose the DFR-Net to divide the shared features into two siamese task-specific streams of localization and appearance, and exploit the intrinsic reciprocal information underlying the task to boost the performance at negligible cost.

As shown in Figure 2, the entire network is built upon an encoder-decoder architecture. For backbone network, our DFR-Net can employ various monocular 3D object detection approaches, such as M3D-RPN [2] and D^{4}LCN [8], etc. We adopt a depth-assisted monocular method [8] to instantiate our model. Given the RGB image and estimated depth map, we fetch the shared features of the last convolutional layer of the encoder and feed them into two sub-task streams to explore the task-specific information. We design the self-mutual appearance-localization feature reflecting (ALFR) module to take a deep look into the implicit interaction across sub-tasks. In specific, two modules termed self-reflect module (S-R) and mutual-reflect module (M-R), are proposed. S-R is to delve into the task-specific features within each task whereas M-R combines the corresponding features between different tasks to diffuse and aggregate the mutual characteristics. Previous works [2, 8, 22] usually leverage distinct head over the shared feature extracted by the encoder to regress disentangled targets. Differently, we learn appearance-dependent information (rotation, 3D dimension, category) from the appearance-aware stream and location-dependent knowledge (2D, 3D location) from the localization-aware stream. In order to realign the multi-task training process, we further design a DIT module to reweight different task losses for the joint optimization of each sub-task and thus contribute to the overall precision of 3D object perception.

3.2. Appearance-localization feature reflecting

To take a deep excavation of reciprocal information underlying monocular 3D object detection task, we propose an appearance-localization feature reflecting module (ALFR) to separate the shared feature into task-specific features and self-mutually reflect the reciprocal relations. As is shown in Figure 3, given a shared feature map F_s, the module first applies two convolutional layers to generate two task-specific feature maps: appearance-specific feature F_υ and localization-specific feature F_σ. Then we feed F_υ into two convolution layers to generate two new feature maps $F_{\upsilon 1}$

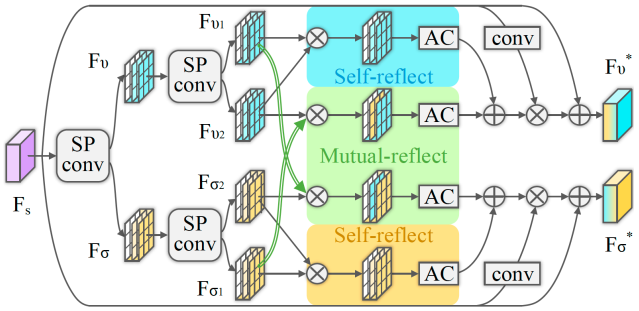


Figure 3. Illustration of the proposed ALFR. "SP conv" and "AC" denote the convolutions that separate the feature into different streams and the activation function, respectively. Self-reflect (S-R) and mutual-reflect (M-R) sub-modules are highlighted with different colors.

and $F_{\upsilon 2}$, and meanwhile utilize F_σ to generate another two new feature maps $F_{\sigma 1}$ and $F_{\sigma 2}$ in the same way. We design a self-reflect module (S-R) to capture the pair-wise context information within each task. Take the appearance stream in the upper part as an example (colored in blue), S-R takes context-aware $F_{\upsilon 1}$ and $F_{\upsilon 2}$ as inputs to calculate the self-reflect attention map of appearance $W_{\upsilon s}$. Besides, we feed $F_{\upsilon 2}$ and $F_{\sigma 1}$ into mutual-reflect (M-R) module to build mutual correlations across tasks and obtain the the mutual-reflect attention map of appearance $W_{\upsilon m}$. The self-reflect and mutual-reflect attention maps are combined via a learnable scale parameter to get the appearance-aware attention map W_υ. To avoid the negative impact of noisy attention at the initial stage of the network, we design an adaptive residual connection between $W_{\upsilon m}$ and the shared input F_s to get the final appearance-specific features F_υ^*.

In detail, the shared feature can be defined as $F_s \in \mathbb{R}^{C \times H \times W}$. Then the output feature maps are $\{F_{\upsilon 1}, F_{\upsilon 2}, F_{\sigma 1}, F_{\sigma 2}\} \in \mathbb{R}^{C/r \times H \times W}$, where the reduction ratio r is to reduce parameter overhead. We reshape them to $\mathbb{R}^{C/r \times N}$, where $N = H \times W$ represents the pixel number of each channel in features. In S-R, we perform a matrix multiplication between the transpose of $F_{\upsilon 1}$ and $F_{\upsilon 2}$ and apply a softmax layer to calculate the self-reflect attention map of appearance $W_{\upsilon s} \in \mathbb{R}^{N \times N}$:

$$W_{\upsilon s} = \frac{exp((F_{\upsilon 1})^T \cdot F_{\upsilon 2})}{\sum_{j=1}^{N} exp((F_{\upsilon 1})^T \cdot F_{\upsilon 2})} \tag{1}$$

In M-R, we perform a matrix multiplication between the transpose of $F_{\upsilon 2}$ and $F_{\sigma 1}$ and then apply a softmax layer to obtain the mutual-reflect attention map of appearance $W_{\upsilon m} \in \mathbb{R}^{N \times N}$, which can be formulated as:

$$W_{\upsilon m} = \frac{exp((F_{\sigma 1})^T \cdot F_{\upsilon 2})}{\sum_{j=1}^{N} exp((F_{\sigma 1})^T \cdot F_{\upsilon 2})} \tag{2}$$

Then we perform an element-wise sum operation to combine the self-reflect and mutual reflect attention maps and output the appearance-aware attention maps W_υ as:

$$W_\upsilon = \lambda_\upsilon * W_{\upsilon s} + (1 - \lambda_\upsilon) * W_{\upsilon m} \tag{3}$$

We send the share feature into a convolution layer to generate a new feature map $F_{\upsilon s}$ and reshape it from $\mathbb{R}^{C \times H \times W}$ to $\mathbb{R}^{C \times N}$ and perform matrix multiplication between $F_{\upsilon s}$ and the transpose of W_υ:

$$F'_{\upsilon s} = F_{\upsilon s} \cdot (W_\upsilon)^T \tag{4}$$

Finally, we reshape $F'_{\upsilon s} \in \mathbb{R}^{C \times N}$ to $\mathbb{R}^{C \times H \times W}$ and combine it with the shared feature F_s via a learnable parameter β_υ to get the final appearance-specific features F_υ^*:

$$F_\upsilon^* = F_s + \beta_\upsilon * F'_{\upsilon s} \tag{5}$$

The final localization-specific F_σ^* is generated similarly.

3.3. Dynamic intra-trading

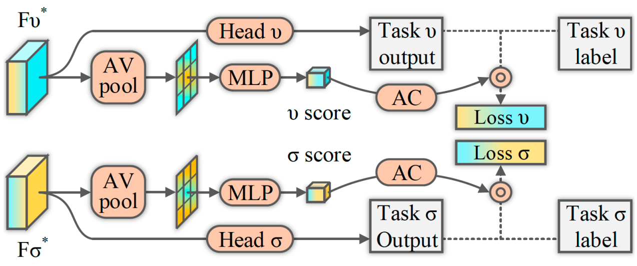


Figure 4. Illustration of the proposed DIT. "AV pool" denotes the average pooling operation. "AC" represents the activation function. The scores that realigns the training processes of tasks υ and σ are generated from F_υ^* and F_σ^*, respectively. The losses of subtasks are reweighted via the self-learned scores.

Thanks to the ALFR module, we can generate appearance-related output ($\{class, w, h, l, rot\}$) from the appearance-aware feature F_υ^* and localization-related output ($\{u, v, u', v', x, y, z\}$) from the localization-aware feature F_σ^*. To measure the differences between the outputs and the ground truth, we define the the appearance-aware loss L_υ for the task υ and the localization-aware loss L_σ for the task σ. The naive mode to combine these two losses could be to conduct a linear sum directly, but model performance will be badly affected by the choice of the hyperparameters weights for each task loss. To enhance the joint optimization of each task, we propose a novel dynamic intra-trading module (DIT) to adaptively learn the confidence scores depending on the actual contribution of the task-related losses for the joint learning. In specific, we first sent the appearance-aware features F_υ to the average pooling layer to aggregate the context information at the spatial level. Then the outputs go through the multilayer perceptron (MLP) and a Sigmoid layer to get the appearance-aware trading score S_υ, which indicates the semantic confidence contained in the input sample. In the same way, we utilize the localization-aware features F_{σ} to obtain the localization-aware trading score S_{σ}. These two scores are used to guide the network learning by weighting the task-related losses. The total trading loss can be formulated as:

$$L = S_v * L_v + S_{\sigma} * L_{\sigma} - log(S_v * S_{\sigma}) \tag{6}$$

where $log(S_v * L_v)$ is the regularization term for the training weights. When S_v or S_{σ} increases, $log(S_v * L_v)$ increases to maintain the overall balance of the total losses and vice versa. Moreover, the appearance-aware trading score S_v will turn small when the input samples are not confident in semantic information. The same applies to the localization-aware trading score S_{σ}. This strategy avoids the negative effect of much noise on the network optimization. In this way, the networks can adjust the proportion of loss back-propagation during the training process and thus promote the learning accuracy of two tasks.

3.4. Objective functions

Following the baseline method [2], we define the losses of each group as:

$$L_{class} = -log(\frac{exp(class_t)}{\sum_j^{n_c} exp(class_j)}) \tag{7}$$

$$L_{rot} = SmoothL1([rot], [rot_g]) \tag{8}$$

$$L_{whl} = SmoothL1([w, h, l], [w_g, h_g, l_g]) \tag{9}$$

$$L_{uvu'v'} = -log(IOU([uvu'v'], [u_g v_g u'_g v'_g])) \tag{10}$$

$$L_{xyz} = SmoothL1([x, y, z], [x_g, y_g, z_g]) \tag{11}$$

where n_c indicates the number of the categories in the training set. We use the standard cross-entropy (CE) for the classification loss and the Smooth L1 for other regression losses. The appearance-aware and the localization-aware losses are formulated as follows:

$$L_v = L_{class} + L_{rot} + L_{whl} \tag{12}$$

$$L_{\sigma} = L_{uvu'v'} + L_{xyz} \tag{13}$$

The total loss is defined in Equation 6.

4. Experiments

Dataset Experiments are conducted on the challenging KITTI dataset [13, 14], which contains 7,481 and 7,518 images for training and testing, respectively. Following previous works [2, 8], we utilize two train-val splits: "val1" split contains 3,712 training and 3,769 validation images while "val2" split employs 3,682 images for training and 3,799 images for validation. We comprehensively analyze the performance of the proposed DFR-Net with other methods on the test and two validation sets.

Evaluation metrics For evaluation, we use precision-recall curves and report the average precision (AP) performance of bird's eye view (BEV) detection and 3D object detection on the KITTI validation and test set. The KITTI test server uses the 40 recall positions-based metric (R40) instead of the 11 recall positions-based metric (R11) after Aug. 2019. We denote AP for 3D and BEV detection as $\mathrm{AP}_{\mathrm{3D}}$ and $\mathrm{AP}_{\mathrm{BEV}}$, respectively. In the benchmark, three levels of difficulty are defined according to the 2D bounding box height, occlusion, and truncation degree, namely, "Easy", "Mod.", and "Hard", and the KITTI benchmark ranks all approaches based on the $\mathrm{AP}_{\mathrm{3D}}$ of "Mod.". Following previous methods [6, 8], IoU = 0.7 is adopted as the threshold for the "Car" category, and IoU = 0.5 is adopted as the threshold for the "Cyclist" and "Pedestrian" categories.

Training details Our experimental settings are strictly consistent with our image-based and depth-assisted baseline methods [2, 8] for fair comparison. For M3D-RPN [2], we use a single Nvidia Tesla v100 GPU to train the model for 50k iterations. The learning rate is set to 0.004 with a poly rate using power as 0.9. We use a batch size of 2 and a weight decay of 0.9. For $\mathrm{D^4LCN}$ [8], the network is optimized by SGD with a momentum of 0.9 and a weight decay of 0.0005. We use 4 Nvidia Tesla v100 GPUs to train the model for 40k iterations. The base learning rate is set to 0.01 and power to 0.9. For both methods, the input images are scaled to 512×1760, and horizontal flipping is the only data augmentation. The reduction ratio r is set to 8. During inference, we apply non-maximum suppression (NMS) on the box outputs in the 2D space using IoU criteria of 0.4 and filter boxes with scores below 0.75.

4.1. Comparison with state-of-the-arts

Results on the KITTI test set We first report the 3D "Car" detection results on KITTI test set at IoU = 0.7 in Table 1. Our plug-and-play DFR-Net has two versions: (a) DFR-Net (I): the image-only-based model based on the backbone of M3D-RPN [2] (the pink row); (b) DFR-Net (I+D): the depth-assisted model based on the backbone of $\mathrm{D^4LCN}$ [8] (the cyan row). In the KITTI leaderboard, our DFR-Net (I+D) ranked 1^{st} among all the monocular-based 3D object detection methods. Note that among all the image-only-based detectors, our DFR-Net (I) still ranks 1^{st} and outperforms them with a considerable margin.

Compared with Kinematic3D [3] that utilizes multiple frames to leverage the temporal motion information to boost the performance, our method achieves superior performance with an improvement of (0.33% / 0.91% / 1.18%

Method	Reference	Speed (FPS)	Extra Info.	AP$_{3D}$ Mod.	AP$_{3D}$ Easy	AP$_{3D}$ Hard	AP$_{BEV}$ Mod.	AP$_{BEV}$ Easy	AP$_{BEV}$ Hard	GPU
FQNet[16]	CVPR 2019	2	-	1.51	2.77	1.01	3.23	5.40	2.46	1080Ti
MonoGRNet[24]	AAAI 2019	16	-	5.74	9.61	4.25	11.17	18.19	8.73	Tesla P40
MonoDIS[31]	ICCV 2019	-	-	7.94	10.37	6.40	13.19	17.23	11.12	Tesla V100
MonoPair[7]	CVPR 2020	17	-	9.99	13.04	8.65	14.83	19.28	12.89	-
UR3D [30]	ECCV 2020	8	-	8.61	15.58	6.00	12.51	21.85	9.2	GTX Titan X
M3D-RPN [2]	ICCV 2019	6.2	-	9.71	14.76	7.42	13.67	21.02	10.23	1080Ti
RTM3D[22]	ECCV 2020	20	-	10.34	14.41	8.77	14.20	19.17	11.99	1080Ti
DFR-Net (I)	-	6.1	-	**11.89**	**17.30**	**9.32**	**16.47**	**24.38**	**13.33**	1080Ti
AM3D[19]	ICCV 2019	3	Depth	10.74	16.50	9.52	17.32	25.03	14.91	1080Ti
PatchNet[18]	ECCV 2020	3	Depth	11.12	15.68	10.17	16.86	22.97	14.97	1080
DA-3Ddet[36]	ECCV 2020	3	D + L	11.50	16.77	8.93	15.90	23.35	12.11	Titan RTX
D^{4}LCN[8]	CVPR 2020	5.6	Depth	11.72	16.65	9.51	16.02	22.51	12.55	1080Ti
Kinematic3D[3]	ECCV 2020	8	Video	12.72	19.07	9.17	17.52	26.69	13.10	-
CaDDN [25]	CVPR 2021	2	LiDAR	13.41	19.17	**11.46**	18.91	27.94	**17.19**	Tesla V100
DFR-Net (I+D)	-	5.5	Depth	**13.63**	**19.40**	10.35	**19.17**	**28.17**	14.84	1080Ti

Table 1. Comparison with state-of-the-art (SoTA) methods on the KITTI test set at IoU = 0.7 (R40). "Depth" and "Video" denote using prior depth estimation and video sequence as an extra input, respectively. "LiDAR" denotes using LiDAR point clouds as extra supervision. "D + L" denotes using both "Depth" and "LiDAR". Based on the encoding backbone of M3D-RPN [2] (the pink row), we rank 1st among all the image-only-based methods. Based on the backbone of D^{4}LCN [8] (the cyan row), we rank 1st among all the competitors in the KITTI monocular 3D object detection track with a high inference speed (2× faster than the second even with a much lighter GPU).

Method	val1 Easy	val1 Mod.	val1 Hard	val2 Easy	val2 Mod.	val2 Hard
M3D-RPN [2]	14.53	11.07	8.65	14.57	10.07	7.51
Ours	**19.55**	**14.79**	**11.04**	**19.38**	**14.33**	**10.63**
Improvement	*+5.02*	*+3.72*	*+2.39*	*+4.81*	*+4.26*	*+3.12*
D^{4}LCN [8]	22.32	16.20	12.30	22.07	14.41	10.39
Ours	**24.81**	**17.78**	**14.41**	**24.30**	**17.23**	**12.52**
Improvement	*+2.49*	*+1.58*	*+2.11*	*+2.23*	*+2.82*	*+2.13*

Table 2. AP$_{3D}$ performance for the "Car" category on KITTI "val1" and "val2" split set at IoU = 0.7 (R40).

Method	Pedestrian Easy	Pedestrian Mod.	Pedestrian Hard	Cyclist Easy	Cyclist Mod.	Cyclist Hard
M3D-RPN [2]	4.92	3.48	2.94	0.94	0.65	0.47
Ours	**6.62**	**4.58**	**4.17**	**1.63**	**1.01**	**1.02**
Improvement	*+1.70*	*+1.10*	*+1.23*	*+0.69*	*+0.36*	*+0.55*
D^{4}LCN[8]	4.55	3.42	2.83	2.45	1.67	1.36
Ours	**6.09**	**3.62**	**3.39**	**5.69**	**3.58**	**3.10**
Improvement	*+1.54*	*+0.20*	*+0.56*	*+3.24*	*+1.91*	*+1.74*

Table 3. AP$_{3D}$ performance for "Cyclist" and "Pedestrian" on KITTI test set at IoU = 0.5 (R40).

) on "Easy", "Mod.", and "Hard", respectively. Compared with the previous top-ranked CaDNN [25], our method still carries out superior results on "Easy" and "Mod." and a comparable result on "Hard". Note that the proposed DFR-Net (I+D) can get a real-time speed of 40 FPS on Tesla V100, which is 20 times faster than CaDNN [25]. The proposed ALFR module only occupies a small computational cost. Therefore, the inference speed and model size are comparable with the baseline method D^{4}LCN (5.5 vs. 5.6 FPS; 355 vs. 352 Mb).

Results on the KITTI validation set We evaluate the proposed framework compared with the cutting-edge image-based [2] and depth-assisted [8] baseline methods on the "val1" and "val2" split sets using AP$_{40}$ as the evaluation metric, as shown in Table 2. Since D^{4}LCN [8] report only the result of "val1" split, we reproduce the results by using the official public code. The proposed method improves the overall accuracy by a large margin compared to the baseline. For instance, the AP$_{3D}$ performance of M3D-RPN [2] on "val1" set gains (5.02% / 3.72% / 2.39%) improvement with the contribution of our DFR-Net on "Easy", "Mod.", and "Hard", respectively. Qualitative comparisons of the baseline and our method are shown in Figure 5. The ground truth, baseline, and our method are colored in green, yellow, and red, respectively. For better visualization, the first and second columns show RGB images and BEV images of pseudo point clouds, respectively. Compared with the baseline, our DFR-Net can produce higher-quality 3D bounding boxes in different scenes. More quantitative and qualitative results are reported in our supplementary material.

Results on "Cyclist" and "Pedestrian" Due to the non-rigid structures and small scale of "Cyclist" and "Pedestrian" categories, it is much more challenging to detect these two categories. Pseudo-LiDAR based methods such as PatchNet [18] and DA-3Ddet [36] fail to detect "Cyclist" and "Pedestrian". We report these two categories with re-

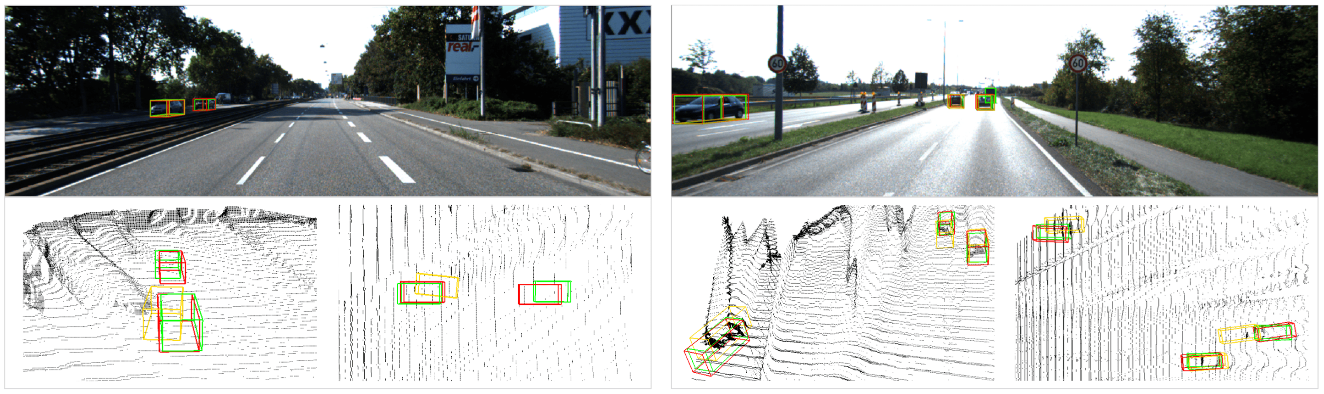


Figure 5. The qualitative comparison of the ground truth (green), the baseline (yellow), and our method (red) on the KITTI validation set. For better visualization, the first row shows RGB images, and the second row shows the front view (left) and BEV (right) pseudo LiDAR, respectively. Due to the reciprocal feature reflecting, the proposed approach can predict accurate 3D bounding boxes of distant objects even with inaccurate depth estimation.

spect to two baseline methods [2, 8] in Table 3. Following [2, 8], $\mathrm{AP}_{\mathrm{3D}}$ of "Cyclist" and "Pedestrian" on the test set at IoU = 0.5 (R40) are reported. Thanks to the reciprocal information underlying the task for 3D reasoning, we are able to localize these challenging categories to some degree and consistently outperform the baselines.

4.2. Ablation study

In this section, we choose M3D-RPN [2] as the baseline and all experiments are conducted on KITTI "val1" split set.

Main ablative analysis The DFR-Net consists of two modules: the ALFR and DIT modules. The ALFR module includes two sub-modules: the self-reflect (S-R) and mutual-reflect (M-R). To demonstrate the effectiveness of each module, we experiment with different combinations of sub-modules and the results are shown in Table 4. We can observe that the performance continues to grow with the participation of components. Specifically, from the comparison of group I and group II or group III and group IV, we find that adding the S-R module contributes to the model, improving the $\mathrm{AP}_{\mathrm{3D}}$ (R40) performance on "Mod." from 11.07% to 13.08% and 13.01% to 13.39%, respectively. The same goes for the M-R module. Meanwhile, combining two modules (S-R and M-R) together works better than merely using one of them separately. This can be concluded from the results of groups II, III, and IV, where the $\mathrm{AP}_{\mathrm{3D}}$ (R40) performance on "Mod." attains better performance 13.39% compared to 13.08% or 13.01%. The above conclusions prove the effectiveness of our ALFR module. When simultaneously embedding the DIT module into the networks, the proposed model attains the best performance regardless of $\mathrm{AP}_{\mathrm{3D}}$ or $\mathrm{AP}_{\mathrm{BEV}}$ metric, which validates the effectiveness of the DIT module.

Different strategies of task clustering We conduct an incisive analysis on the effect of different task clustering strategies. The results are shown in Table 5. Since the task partition of some element variables is relatively certain, such as $\{x, y, z, u, v, u', v'\}$ belongs to object localization task ("Loc") while $class$ remains with appearance perception task ("App"), we focus on the attribution of rotation and dimension $\{w, h, l\}$. The first and second row results demonstrate that $\{w, h, l\}$ clustered by the appearance task achieves a better performance of (18.55% / 14.14% / 11.29%) than clustered by the localization task (17.21% / 13.35% / 10.73%). The first and third row results reveal that assigning the "rotation" to the appearance task attains a 0.96% gain on "Mod." compared to assigning it to localization task. When simultaneously allocating $\{w, h, l\}$ and rotation to the appearance-aware task achieves the best performance, which proves the effectiveness of our choice from the perspective of experiments.

Information flow in the ALFR We further dig into the information flow in the M-R module of our ALFR and experiment on the "val1" split set. Table 6 reports the final performance of different forms of information flow. The M-R module is composed of two information flow: appearance to localization ("App$\rightarrow$ Loc") and localization to appearance ("Loc$\rightarrow$ App"). Note that "None" denotes the DFR-Net without the M-R module (group V in Table 4). From the table, we can find that adding one of these types of information can be beneficial. In specific, adding appearance to localization flow improves the "Mod." performance from 13.46% to 13.64%, while adding localization to appearance flow promotes to 13.73%. The combination of two flows achieves the best accuracy.

Different settings of the DIT In order to further dig into the impact of the DIT module, we define some variants of the DIT module based on different settings: (a) "DIT-init": initialize the trading scores for each task instead of network generation; (b) "DIT-cross": appearance-aware task and localization-aware task generate each other's trading scores; (c) "DIT-shared": the trading scores of each task

Group	S-R	M-R	DIT	AP_{3D} (R11 / R40) Easy	AP_{3D} Mod.	AP_{3D} Hard	AP_{BEV} (R11 / R40) Easy	AP_{BEV} Mod.	AP_{BEV} Hard
I	-	-	-	20.27 / 14.53	17.06 / 11.07	15.21 / 8.65	25.94 / 20.85	21.18 / 15.62	17.90 / 11.88
II	✓	-	-	20.75 / 17.30	16.57 / 13.08	14.98 / 10.41	27.62 / 24.61	22.65 / 18.03	18.50 / 14.61
III	-	✓	-	20.27 / 17.26	17.11 / 13.01	14.36 / 10.50	25.09 / 23.26	21.65 / 17.79	17.47 / 13.82
IV	✓	✓	-	21.08 / 18.00	17.10 / 13.39	15.19 / 10.78	26.53 / 24.87	22.01 / 18.35	17.66 / 15.03
V	✓	-	✓	21.23 / 17.56	17.13 / 13.46	15.23 / 10.79	27.57 / 24.56	22.70 / 18.46	18.45 / 15.15
VI	-	✓	✓	**22.81** / 18.06	18.15 / 13.88	16.10 / 10.23	27.64 / 24.85	23.07 / 18.60	19.01 / 14.35
VII	✓	✓	✓	22.04 / **19.55**	**18.43 / 14.79**	**16.96 / 11.04**	**28.63 / 26.60**	**23.15 / 19.80**	**19.31 / 15.34**

Table 4. Ablative analysis on the "Car" category on KITTI "val1" split set for AP_{3D} and AP_{BEV} at IoU = 0.7

Task υ Loc: xyz, uvu'v'		Task σ App: class		AP_{3D} Easy	Mod.	Hard
rotation	whl	-	-	17.21	13.35	10.73
rotation	-	-	whl	18.55	14.14	**11.29**
-	whl	rotation	-	17.62	14.31	10.88
-	-	rotation	whl	**19.55**	**14.79**	11.04

Table 5. AP_{3D} and AP_{BEV} comparison of different task clustering strategies on "val1" split set at IoU = 0.7 (R40).

Method	AP_{3D} Easy	AP_{3D} Mod.	AP_{3D} Hard	AP_{BEV} Easy	AP_{BEV} Mod.	AP_{BEV} Hard
None	17.56	13.46	10.79	24.56	18.46	15.15
App $\rightarrow$ Loc	18.59	13.64	10.98	24.89	18.92	15.11
Loc $\rightarrow$ App	18.67	13.73	**11.14**	25.77	19.23	**15.82**
Ours	**19.55**	**14.79**	11.04	**26.60**	**19.80**	15.34

Table 6. AP_{3D} and AP_{BEV} comparison of different feature reflecting strategies on "val1" split set at IoU = 0.7 (R40).

are generated from the shared feature. As shown in Table 7, DIT-init improves the AP_{3D} performance from 13.39% to 14.07% in Mod. setting. However, it is obvious that DIT-cross distinctly drops the overall accuracy. This is because the final outputs of each task stream are specific to the corresponding task after the encoding of the ALFR module. Thus, there will be a lot of noise in predicting the confidence for another task, which affects the network learning. DIT-shared achieves better results than the above designs, reaching to AP_{3D} 19.56% on "Easy", which explains the shared-features contain the rich contextual information required by the two tasks. When equipped with the proposed DIT, the model can get the best performance, which demonstrates the effectiveness of our module.

Generalization ability For generalization ability validation, we extend our method to 2D detection task. We choose the well known SSD [17] as the baseline and apply the proposed ALFR and DIT module for comparison. As illustrated in Table 8, we perform experiments on the VOC dataset in 07++12 setting: training on the union of VOC2007 and VOC2012 trainval set and testing on the VOC2007 test set. The experiment results show that the

Method	AP_{3D} Easy	AP_{3D} Mod.	AP_{3D} Hard	AP_{BEV} Easy	AP_{BEV} Mod.	AP_{BEV} Hard
None	18.00	13.39	10.78	24.87	18.35	15.03
DIT-init	18.67	14.07	10.50	25.43	19.17	14.07
DIT-cross	15.59	12.44	9.45	22.95	17.74	14.57
DIT-shared	**19.56**	13.87	10.91	26.20	19.66	15.31
Ours	19.55	**14.79**	**11.04**	**26.60**	**19.80**	**15.34**

Table 7. AP_{3D} and AP_{BEV} comparison of different dynamic intra-trading methods on "val1" split set at IoU = 0.7 (R40).

2D detector achieves consistent performance gain via the combination of the proposed modules, which demonstrate the versatility ability of our model.

Method	data	mAP
SSD300	07++12	77.2
+ALFR+DIT	07++12	**78.0**

Table 8. The detection results on PASCAL VOC2007 test set via the combination of our model and SSD [17].

5. Conclusion

We have proposed a dynamic feature reflecting network (DFR-Net). The proposed ALFR module separates the appearance perception and object localization decoding streams to exploit and reflect reciprocal information between sub-tasks in a self-mutual manner. Our DIT module further scores the features of sub-tasks in a self-learning manner and accordingly realigns the multi-task training process. Extensive experiments on the KITTI dataset demonstrate the effectiveness and the efficiency of our DFR-Net. It is worth mentioning that DFR-Net ranks 1^{st} in the highly competitive KITTI monocular 3D object detection track. Besides, ablations on 2D detector SSD verify the generalization ability of the proposed modules. Our method can also be amazingly plug-and-play on several cutting-edge frameworks at negligible cost. In future work, we will apply the proposed modules to more cutting-edge 3D detection approaches and other area to further verify the general ability of our model.

References

[1] Yousef Atoum, Joseph Roth, Michael Bliss, Wende Zhang, and Xiaoming Liu. Monocular video-based trailer coupler detection using multiplexer convolutional neural network. In *ICCV*, 2017. 2

[2] Garrick Brazil and Xiaoming Liu. M3d-rpn: Monocular 3d region proposal network for object detection. In *ICCV*, 2019. 2, 3, 5, 6, 7

[3] Garrick Brazil, Gerard Pons-Moll, Xiaoming Liu, and Bernt Schiele. Kinematic 3d object detection in monocular video. 2020. 2, 5, 6

[4] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *CVPR*, 2016. 2

[5] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. In *NeurIPS*, 2015. 2

[6] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *CVPR*, 2017. 5

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

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

[9] Liang Du, Jingang Tan, Xiangyang Xue, Lili Chen, Hongkai Wen, Jianfeng Feng, Jiamao Li, and Xiaolin Zhang. 3dcfs: Fast and robust joint 3d semantic-instance segmentation via coupled feature selection. In *ICRA*, 2020. 2

[10] Liang Du, Xiaoqing Ye, Xiao Tan, Jianfeng Feng, Zhenbo Xu, Errui Ding, and Shilei Wen. Associate-3ddet: Perceptual-to-conceptual association for 3d point cloud object detection. In *CVPR*, pages 13326–13335, 2020. 1

[11] Liang Du, Xiaoqing Ye, Xiao Tan, Edward Johns, Bo Chen, Errui Ding, Xiangyang Xue, and Jianfeng Feng. Ago-net: Association-guided 3d point cloud object detection network. *TPAMI*, 2021. 1

[12] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, 2018. 3

[13] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *IJRR*, 2013. 5

[14] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 1, 5

[15] Jason Ku, Alex D Pon, and Steven L Waslander. Monocular 3d object detection leveraging accurate proposals and shape reconstruction. In *CVPR*, 2019. 2

[16] Lijie Liu, Jiwen Lu, Chunjing Xu, Qi Tian, and Jie Zhou. Deep fitting degree scoring network for monocular 3d object detection. In *CVPR*, 2019. 6

[17] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 8

[18] Xinzhu Ma, Shinan Liu, Zhiyi Xia, Hongwen Zhang, Xingyu Zeng, and Wanli Ouyang. Rethinking pseudo-lidar representation. In *ECCV*, 2020. 3, 6

[19] 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 *ICCV*, 2019. 6

[20] Fabian Manhardt, Wadim Kehl, and Adrien Gaidon. Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape. In *CVPR*, 2019. 3

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

[22] Pengfei Liu Feidao Cao Peixuan Li, Huaici Zhao. Rtm3d: Real-time monocular 3d detection from object keypoints for autonomous driving. In *ECCV*, 2020. 2, 3, 6

[23] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *CVPR*, 2018. 2

[24] Zengyi Qin, Jinglu Wang, and Yan Lu. Monogrnet: A geometric reasoning network for monocular 3d object localization. In *AAAI*, 2019. 2, 6

[25] Cody Reading, Ali Harakeh, Julia Chae, and Steven L. Waslander. Categorical depth distribution network for monocular 3d object detection. 2021. 2, 6

[26] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint*, 2018. 1

[27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1

[28] Cosimo Rubino, Marco Crocco, and Alessio Del Bue. 3d object localisation from multi-view image detections. *TPAMI*, 40(6):1281–1294, 2017. 2

[29] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *CVPR*, 2019. 1, 2

[30] Xuepeng Shi, Zhixiang Chen, and Tae-Kyun Kim. Distance-normalized unified representation for monocular 3d object detection. In *ECCV*, 2020. 2, 6

[31] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3d object detection. In *ICCV*, 2019. 2, 6

[32] 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 *CVPR*, 2019. 2, 3

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

[34] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3d object detection from monocular images. In *CVPR*, 2018. 2

[35] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 2018. 1

[36] Xiaoqing Ye, Liang Du, Yifeng Shi, Yingying Li, Xiao Tan, Jianfeng Feng, Errui Ding, and Shilei Wen. Monocular 3d object detection via feature domain adaptation. In *ECCV*, 2020. 6

[37] 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*, 2019. 2, 3

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