Depth-conditioned Dynamic Message Propagation for Monocular 3D Object Detection

Li Wang[1*] Liang Du[1*] Xiaoqing Ye[2*] Yanwei Fu[1] Guodong Guo[2]
Xiangyang Xue[1] Jianfeng Feng[1] Li Zhang[1†]
[1]Fudan University [2]Baidu Inc.

Abstract

The objective of this paper is to learn context- and depth-aware feature representation to solve the problem of monocular 3D object detection. We make following contributions: (i) rather than appealing to the complicated pseudo-LiDAR based approach, we propose a depth-conditioned dynamic message propagation (DDMP) network to effectively integrate the multi-scale depth information with the image context; (ii) this is achieved by first adaptively sampling context-aware nodes in the image context and then dynamically predicting hybrid depth-dependent filter weights and affinity matrices for propagating information; (iii) by augmenting a center-aware depth encoding (CDE) task, our method successfully alleviates the inaccurate depth prior; (iv) we thoroughly demonstrate the effectiveness of our proposed approach and show state-of-the-art results among the monocular-based approaches on the KITTI benchmark dataset. Particularly, we rank 1^{st} *in the highly competitive KITTI monocular 3D object detection track on the submission day (November 16th, 2020). Code and models are released at* `https://github.com/fudan-zvg/DDMP`

1. Introduction

Object detection is a fundamental problem in computer vision. Although promising progress in 2D object detection has been made [16, 24, 33, 39, 42] with convolutional neural networks (CNNs) in recent years, 3D object detection that perceives 3D object location, physical dimension, and orientation, still remains challenging and critical in applications such as autonomous driving [10, 14], robotic grasp and navigation [9, 40], and Mixed Reality (MR) [38].

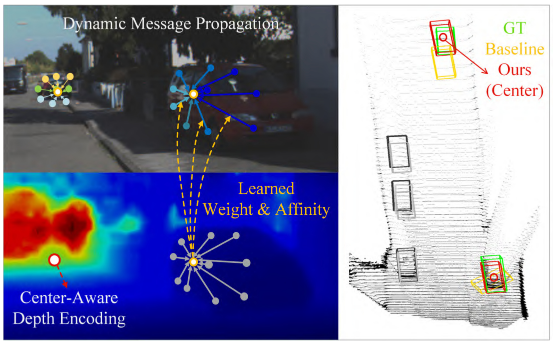
Dynamic Message Propagation
Learned Weight & Affinity
Center-Aware Depth Encoding
GT
Baseline
Ours (Center)

Figure 1. **Left**: DDMP adaptively samples context-aware nodes (top) in the image context and dynamically predicting hybrid depth-dependent filter weights and affinity matrices (bottom) for propagating information. **Right**: the improvement of DDMP-3D (red) over the baseline (yellow) via center-aware depth encoding.

LiDAR point cloud based methods [11, 20, 35, 45, 52] have excelled in 3D object detection and achieve superior performance, however, they still depend on the expensive LiDAR sensors and sparse data representation to make them scalable. Cheaper alternative such as perceiving RGB images that are captured by the monocular camera is drawing increasing attention. Some image-only based methods [1, 2, 6, 23, 27] aim to explore the 2D-3D geometric consistency for recovering reasonable 3D detection. Nevertheless, the performance is still far from satisfactory. This is because (i) scale variance caused by the perspective projection. The monocular views at far and near distance cause significant changes in object scales. It is difficult for conventional CNNs (*e.g.*, 2D conv) to process objects of different scales. (ii) lack of depth cues for the CNNs to capture the depth-aware feature for 3D reasoning.

Recent efforts have been made to pursue pseudo-LiDAR based approaches [25, 43, 44, 48]. The pseudo-LiDAR approaches first transform depth maps estimated from 2D images to point cloud data representations and then adopt existing LiDAR-based 3D detectors for prediction. Although

[*] The first three authors contributed equally to this work.

[†] Li Zhang (lizhangfd@fudan.edu.cn) is the corresponding author with School of Data Science, Fudan University. Li Wang and Xiangyang Xue are with School of Computer Science, Fudan University. Yanwei Fu is with the School of Data Science, MOE Frontiers Center for Brain Science, and Shanghai Key Lab of Intelligent Information Processing, Fudan University. Liang Du and Jianfeng Feng are with the Institute of Science and Technology for Brain-Inspired Intelligence, Fudan University.

improved performances on monocular 3D detection have been observed, they still suffer from the gap between inaccurate estimated depth and real-world depth. Additionally, LiDAR-based approaches only employ 3D spatial information to generate LiDAR-resembled point cloud but discard the semantic information from RGB images, which is a vital clue to perceive and distinguish objects.

Another line of research [8, 29, 36] focuses on the perspective of image and depth fusion learning. Arguably, with the depth assisted, the model can learn depth-aware feature representation, and both semantic and structure knowledge can be integrated for 3D reasoning. Specifically, D^{4}LCN [8] proposes to generate dynamic-depthwise-dilated kernels from depth features to integrate with image context. However, two issues still remain unsolved, (i) Its empirical design can not guarantee the discriminative power of the model and the local dilated convolution is not be able to fully capture the object context in the condition of perspective projection and occlusion. (ii) Its 3D detection performance heavily relies on the precision of estimated depth maps. The model has no clue to resolve the inferior 3D localization caused by the inaccurate depth prior.

To this end, for the first time we present a graph-based formulation, a novel depth-conditioned dynamic message propagation model (DDMP-3D), to effectively learn depth-aware feature representations for monocular 3D object detection. As shown in figure 1, specifically, considering each feature pixel as a node within a graph, we first dynamically sample the neighbourhoods of a node from the feature graph. This operation allows the network to gather the object contexts efficiently by adaptively selecting a subset of the most relevant nodes in the graph. For the sampled nodes, we further predict filter weights and affinity matrices dependent on the aligned depth feature to propagate information through the sampled nodes. Moreover, multi-scale depth features are explored over the propagation process, hybrid filter weights and affinity matrices are learned to adapt various scales of objects.

Additionally, to resolve the challenge of inaccurate depth prior, a center-aware depth encoding (CDE) is augmented as an auxiliary task append at the depth branch. It performs 3D object center regression task which explicitly guides the intermediate features of the depth branch to be *instance-aware* and further improves the localization of objects.

2. Related work

Image-only 3D detection Due to the absence of accurate depth information, the monocular 3D detection task remains challenging. Several works rely only on RGB information and geometry consistency to predict 3D boxes. For instance, Deep3DBox [27] utilizes bins classification to solve the orientation regression and enforces the 2D-3D box constraint to recover 3D locations and shape. M3D-RPN [1] leverages the geometric relationship between 2D and 3D perspectives by sharing the prior anchors and classification targets. For a better understanding of the spatial relationship between original 3D bounding boxes and the object, FQNet [23] infers the 3D IoU between 3D proposals and the object by drawing the projection results on the image plane to bring additional information. However, the orientation predictions can affect the following location prediction greatly, leading to a coupled 3D parameter regression. MonoPair [6] captures spatial relationships between paired objects to improve accuracy on occluded objects, inspired by the key point-based CenterNet [51].

Depth-assisted 3D detection An alternative solution to improve monocular detection performance is to utilize depth information. The pioneering work pseudo-LiDAR [25, 43] imitates the process of LiDAR-based (Point Cloud) 3D detection approaches by first estimating depth maps for input images from off-the-shelf methods such as DORN [12], then transforming it into 3D space and adopting LiDAR-based 3D detection approaches. Mono3D-PLiDAR [44] points out that the noise in pseudo-LiDAR data is a bottleneck to improve performance and adopts the instance mask instead of the bounding box for frustum lifting. With the help of the stereo network, more accurate depth estimation is achieved to aid the pseudo-LiDAR methods in [48]. Different to pseudo-LiDAR approaches that rely heavily on the accuracy of estimated depth, D^{4}LCN [8] carefully designs a local convolutional network, where the depth map was regarded as guidance to learn local dynamic depthwise-dilated kernels for images. However, we argue that the local dilated convolution is not able to fully capture the object context in the condition of perspective projection and occlusion, and the model has no mechanism to resolve the inferior 3D localization caused by the inaccurate depth prior.

In addition to the above-mentioned approaches, multi-task or leveraging auxiliary knowledge is exploited to improve the 3D detection performance. For instance, AM3D [25] designs two modules for background segmentation and RGB information aggregation respectively in order to enhance the 3D box estimation task.

Graph neural network A number of methods have been explored to model context for computer vision tasks, such as dilated convolution [49] and deformable convolution [7]. Graph neural networks [19, 41, 50], on the other hand, propagate information along graph-structured input data. These networks are superior to both fronts on capturing object context. In this paper, we present a graph-based formulation for depth-aware feature representation learning, with the goal of solving monocular 3D object detection.

3. Methodology

We first briefly introduce the graph message passing formulation. Then the overall framework and each crit-

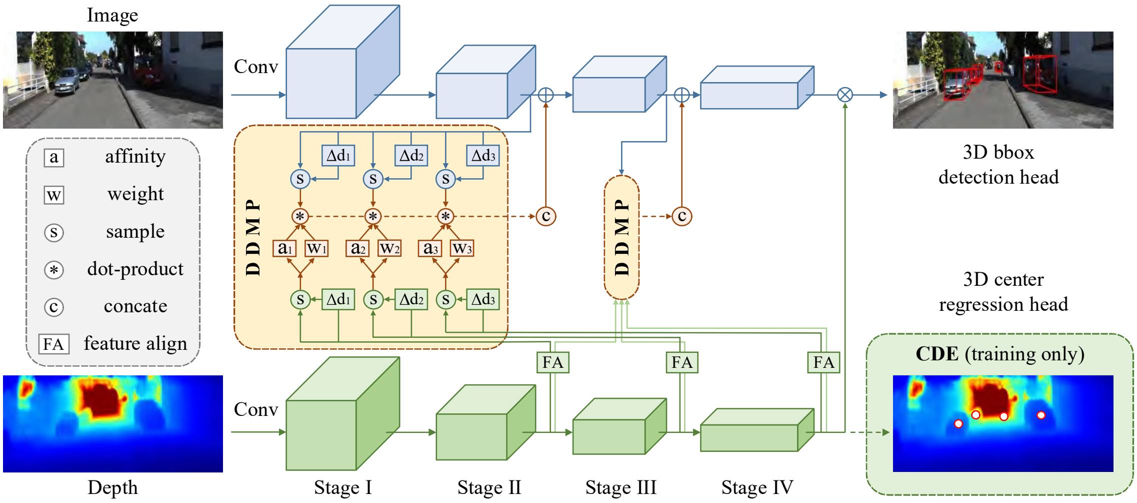

Figure 2. Schematic illustration of our proposed DDMP-3D. Two branches are involved including 3D detection branch (colored in blue) and depth feature extraction branch (colored in green). The DDMP modules in yellow color reveal the depth-conditioned dynamic message propagation. It dynamically samples context-aware nodes in the upper image branch and predicts the hybrid filter weights and affinities based on multi-scale depth features from the bottom branch for message propagation. CDE is the auxiliary task for joint-optimization training and is discarded during inference.

ical component are explained in detail, including depth-conditioned dynamic message propagation (DDMP) and center-aware depth feature encoding (CDE). Finally, we describe the instantiation of our model and loss functions.

3.1. Graph message passing

The message passing mechanism constructs a feature graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, A\}$, where $\mathcal{V}$ stands for nodes, $\mathcal{E}$ for edges and A for adjacency matrices. Each node in graph are represented as the latent feature vector $\mathbf{h}_i$, i.e., $\mathcal{V} = \{\mathbf{h_i}\}_1^N$, where N is total number of nodes. And $A \in R^{N \times N}$ is a binary or learnable matrix with self-loops describing the connections between nodes. Network aims to refine latent feature vectors $\mathbf{h}_i$ by extracting hidden structured information among the feature vectors at different node locations. The common message passing phase takes T iterations and is composed of a message calculation step M^t and a message updating step U^t. Given a latent feature vector $\mathbf{h}_i^{(t)}$ at iteration t, it samples K locally connected node field $v_j \subset \mathcal{V}, v_j \in R^{K \times C}$, where C is vector dimension and $K \ll N$. Thus the message calculation step for node i is:

$$\begin{aligned}\mathbf{m}_i^{t+1} &= M^t(A_{i,j}\{\mathbf{h}_1^{(t)}, ..., \mathbf{h}_K^{(t)}\}, \mathbf{w}_j) \\ &= \sum_{j \in \mathcal{N}(i)} A_{i,j}\mathbf{h}_j^{(t)}\mathbf{w}_j\end{aligned} \tag{1}$$

where $A_{i,j}$ is the connection relationship between latent nodes $\mathbf{h}_i^{(t)}$ and $\mathbf{h}_j^{(t)}$, $\mathcal{N}(i)$ contains the K number of sampled nodes for v_i, and $\mathbf{w}_j \in R^{C \times C}$ is a transformation matrix for message calculation on the hidden node $\mathbf{h}_j^{(t)}$. Then message updating step U^t updates the node $\mathbf{h}_i^{(t)}$ with a linear combination of the calculated message and the original node status:

$$\mathbf{h}_i^{t+1} = U^t(\mathbf{h}_i^t, \mathbf{m}_i^{t+1}) = \sigma(\mathbf{h}_i^t + \alpha_i^m \mathbf{m}_i^{t+1}) \tag{2}$$

where α_i^m is a learnable parameter for scaling the message, and the operation $\sigma(\cdot)$ is a non-linearity function *e.g.,* ReLU. By updating nodes T times, the network finally obtained refined features via message passing on each nodes.

3.2. Framework overview

We give an overview of the proposed DDMP-3D as in Figure 2. It consists of two branches, one for 3D regression (the upper branch colored in blue) while the other for depth feature extraction (the bottom branch colored in green). The RGB images are initially fed into the upper branch for feature extraction while corresponding depth maps estimated via off-the-shelf depth estimator are sent into the depth branch for extracting depth-aware features. Specifically, we first adaptively sample context-aware nodes in the image graph and then dynamically predict hybrid depth-dependent filter weights and affinity matrices. After obtaining the context- and depth- aware features from these two branches, we integrate them in a graph message propagation pattern via the DDMP module. Common 3D heads for 3D center, dimension, orientation regression are followed to achieve final 3D object boxes. Moreover, an auxiliary center regression task is performed during the training process to implicitly guide the depth sub-network to learn center-aware depth features for better object localization.

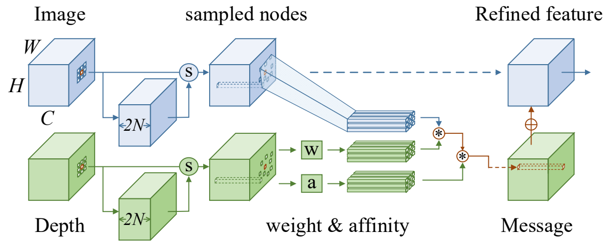

Figure 3. Illustration of DDMP module in a single scale pattern. Dynamic nodes are first sampled from the image and depth feature graph, for these sampled nodes, the filter weights and affinity matrices are learned from depth features to propagate the depth-conditioned message.

3.3. Depth-conditioned dynamic message propagation (DDMP)

We first introduce the process of our DDMP. The input feature maps are regarded as a graph, in which each pixel of the feature map is a node vector $v_i \in R^C$ and all the pixels make up the set $\mathcal{V} = \{v_i\}_1^N$, where C is the channel numbers of the input feature maps and N is the total number of pixels, and its status at time t is denoted as latent node $\mathbf{h}_i^{(t)}$. Hence, depth-conditioned message propagation is employed on feature nodes to deliver the context- and depth-aware information. In particular, it includes two steps: 1) Image feature node sampling, which selects a subset of relevant object nodes in the graph; 2) For these sampled nodes, the hybrid filter weights and affinity matrices are learned from multi-scale depth feature to enrich the propagated message.

Figure 3 shows the depth-conditioned dynamic message propagation process. We first sample the context-aware nodes in the image graph. Due to the diverse scales and occlusion of targets caused by perspective projection, we follow the strategies in [7,50] to explore dynamic sampling during the message propagation process. For each node v_i, the sampling number K determines the receptive field of v_i. To adaptively sample relevant nodes for v_i while taking different feature distribution into consideration, the network learns to walk around so as to choose the most effective nodes among the uniformly distributed neighboring nodes. We denote $\Delta\mathbf{d}_{i,j} \in R^D$ as predicted walk for a uniformly sampled relevant node $v_{i,j}$ with $j \in \mathcal{N}(i)$, where $\mathcal{N}(i)$ contains K number of sampled nodes for v_i and $D = 2$ is the space dimension along height and width. Then the node walk can be described as a matrix transformation:

$$\Delta\mathbf{d}_{i,j} = \mathbf{W}_{i,j}\mathbf{h}_i + \mathbf{b}_{i,j} \tag{3}$$

where $\mathbf{W}_{i,j}$ and $\mathbf{b}_{i,j}$ are matrix transformation parameters learned on image graph nodes, $\mathbf{h}_i$ is the latent vector for v_i.

After the above sampling, we obtain the dynamic image relevant nodes ${v'}_{i,j}$ based on v_i. According to the Equation 1, we need to learn the affinity matrix A and transformation matrix $\mathbf{w}$ for calculating the message. To effectively achieve depth-sensitive features, we generate hybrid depth-dependent filter weights and affinity matrices based on the multi-scale depth feature. Because of the ill depth prior, we do not sample the corresponding depth nodes, but generate another walk $\Delta\tilde{\mathbf{d}}_{i,j}$ for uniformly neighboring nodes on the depth feature graph. For stage l, we have obtained dynamic walked nodes ${v'}_{i,j}^l$ and $\tilde{v}_{i,j}^l$ from image and depth feature graph respectively. Note that we have aligned inconsistent scale depth features via down- or up-sampling operation. Then depth feature nodes $\tilde{v}_{i,j}^l$ learns to further generate affinity matrix $A_{i,j}^l$ and transformation matrix $\mathbf{w}_{i,j}^l$, as:

$$\mathbf{m}_i^{t+1} = \sum_{l\in L}\sum_{j\in\mathcal{N}(i)} \beta_l A_{i,j}^l \mathbf{h'}_j^{l,(t)} \mathbf{w}_{i,j}^l \tag{4}$$

$$\{A_{i,j}^l; \mathbf{w}_{i,j}^l\} = \tilde{\mathbf{W}}_{i,j}^l \tilde{v}_{i,j}^l + \tilde{\mathbf{b}}_{i,j}^l \tag{5}$$

In Equation 4, L stands for the layer from different level stages; $\mathbf{h'}_j^{l,(t)}$ is the latent vector for dynamic nodes ${v'}_{i,j}^l$ from stage l with node walk $\Delta\mathbf{d}_{i,j}$, and β_l is the balance weight for depth feature maps integration from different stages. $\tilde{\mathbf{W}}_{i,j}^l$ and $\tilde{\mathbf{b}}_{i,j}^l$ in Equation 5 are matrix transformation parameters generated by depth nodes. The calculated message is summarized as:

$$\mathbf{m}_i^{t+1} = \sum_{l\in L}\sum_{j\in\mathcal{N}(i)} \beta_l A_{i,j}^l \delta(\mathbf{h'}_j^{l,(t)}|\mathcal{V}; j; \Delta\mathbf{d}_{i,j})\mathbf{w}_{i,j}^l \tag{6}$$

where $\delta(\cdot)$ is a bilinear sampler which samples a new node $\mathbf{h'}_j^{l,(t)}$ calculated by $\Delta\mathbf{d}_{i,j}$ over the whole nodes $\mathcal{V}$ of graph.

3.4. Model instantiation

The introduced architecture is shown in Figure 2 and a more detailed scheme of the DDMP module is further depicted in Figure 3. DDMP module is embedded in the network within Stage II and III to propagate the context- and depth-aware message. To instantiate it, we first extract the hierarchical depth features from Stage II, III, and IV with different sizes, and adopt stride MaxPooling or interpolation operation to align these feature maps to the same size with corresponding stage image feature maps. Then we transform the channels of RGB and depth features from different stages to a fixed number C (*i.e.*, 256) with 1×1 convolutions before feeding them into DDMP. DDMP accepts RGB features as input feature nodes $\mathbf{F}$, $\mathbf{F} \in R^{C\times H\times W}$, where C, H and W are the channel, height, and width of the feature map, respectively. We denote $\mathbf{H}^{(0)}$ as an initial state of the latent feature map $\mathbf{H}$, which has the same dimension with $\mathbf{F}$, and $\mathbf{H}^{(0)} = \mathbf{F}$. The dynamic walk $\Delta\mathbf{d}$ for each image node is generated by applying 3×3 convolutional layers according to Equation 3. The same operation is done on depth features by generating $\Delta\tilde{\mathbf{d}}$ to obtain the sampled depth nodes. Hybrid dynamic affinity matrices and weights in Equation 5 are also calculated on hierarchical depth features $\mathbf{F}^l_{dep} \in R^{C\times H\times W}$ by applying 3×3 convolutional layers, where l stands for stage II / III / IV. Then we obtain affinity matrices $A^l \in R^{H\times W\times K}$ and another weights $W^l \in R^{H\times W\times K\times G}$, where K is sampled size (*i.e.*3×3) and G is group size. All message $\mathbf{M} \in R^{C\times H\times W}$ is calculated as Equation 6 using group convolutional layers and concated with $\mathbf{F}$, to produce a refined feature map $\mathbf{H}^{(1)}$ via a 1×1 convolutional, which is described as Equation 2. We perform $T=1$ to balance the performance and efficiency.

3.5. Center-aware depth feature encoding (CDE)

Generally, the depth map estimated from off-the-shelf algorithms sometimes inevitably lose appearance details or fail to discriminate the depth between the foreground instance and backgrounds, which leads to unreliable depth prior for the depth-assisted 3D object detection. It is already proved that multi-task strategy [15,46] can boost each single task to some degree, benefiting from the multi-fold regularization effect in the joint-optimization. Hence, we augment an auxiliary task to jointly optimize with the main 3D detection task as depicted in Figure 2. The augmented task with xyz (Equation 9) supervision in 3D space uniquely determines a point in 2D image plane, which imposes spatial constraints on the network to gain a better 3D instance-level understanding. With the better instance-awareness brought by CDE, our model is able to alleviate the inaccurate depth prior in situation like *occlusion* and *distant* objects. We adopt a similar network architecture for depth branch with the head only predicting 3D centers without predefined anchors. During training, we jointly optimize the losses of the two sub-branches in the light of guiding the intermediate depth features to be center-aware and thus enhancing the performance for 3D object detection. In this way, the proposed approach becomes more robust to the estimated depth accuracy, validation experiments can be found in Section 4. The auxiliary task is omitted at the inference phase without extra computational cost.

3.6. Objective functions

We follow single-stage approaches [8, 34] to pre-define anchors and then regress the positive samples. Firstly, 2D anchors $[\hat{x},\ \hat{y},\ \hat{w},\ \hat{h}]_{2d}$ are defined in 2D space for the 2D bounding box. $[\hat{x},\ \hat{y}]_p$ represents the 2D location projected from the 3D center. 3D anchor parameters $[\hat{z},\hat{w},\hat{h},\hat{l},\hat{\theta}]_{3d}$ denote 3D object center depth, physical dimension, and rotation, respectively. Then all ground truth 3D boxes are projected to 2D space to compute the intersection over union (IoU) with 2D anchors. Positive 2D anchors with IoU $\geq$ 0.5 combined with 3D anchor parameters are thus selected to participate in the regression for 3D bounding predictions. The predicted 2D-3D parameters include $([x,y,w,h]_{2d},[x,y]_p,[z,w,h,l,\theta]_{3d})$, which denote the classification scores, 2D bounding box locations, 3D center projections on 2D plane, 3D object center depths, physical dimensions and rotations, respectively. Similar to YOLOv3 [32], we adopt $([t_x,t_y,t_w,t_h]_{2d},[t_x,t_y]_p,[t_z,t_w,t_h,t_l,t_\theta]_{3d})$ to parameterize the corresponding predictions directly generated by the network. Hence the predicted 3D boxes is,

$$\begin{aligned}
[x,y]_{2d} &= [\hat{x},\hat{y}]_{2d} + [t_x,t_y]_{2d} * [\hat{w},\hat{h}]_{2d} \\
[w,h]_{2d} &= [\hat{w},\hat{h}]_{2d} * exp([t_w,t_h]_{2d}) \\
[x,y]_p &= [\hat{x},\hat{y}]_p + [t_x,t_y]_p * [\hat{w},\hat{h}]_{2d} \\
[w,h,l]_{3d} &= [\hat{w},\hat{h},\hat{l}]_{3d} * exp([t_w,t_h,t_l]_{3d}) \\
[z,\theta]_{3d} &= [\hat{z},\hat{\theta}]_{3d} + [t_z,t_\theta]_{3d},
\end{aligned} \tag{7}$$

where we use the same anchor for $[\hat{x},\hat{y}]_{2d}$ and $[\hat{x},\hat{y}]_p$. Therefore, the loss for the detection branch L_{det} contains classification loss, 2D bounding box regression loss and 3D box regression loss. We apply standard cross-entropy loss for classification and smooth L1 ($\tilde{L}1$) loss for box regression. The detection branch loss then is,

$$\begin{aligned}
L_{det} &= L_{cls} + L_{2d} + L_{3d} \\
L_{2d} &= \tilde{L1}([t_x,t_y,t_w,t_h]_{2d},\ [t_x^{gt},t_y^{gt},t_w^{gt},t_h^{gt}]_{2d}) \\
L_{3d} &= \tilde{L1}([t_x,t_y]_p,[t_x^{gt},t_y^{gt}]_p) \\
&\quad + \tilde{L1}([t_z,t_w,t_h,t_l,t_\theta]_{3d},\ [t_z^{gt},t_w^{gt},t_h^{gt},t_l^{gt},t_\theta^{gt}]_{3d}),
\end{aligned} \tag{8}$$

where $(*)^{gt}$ means (*)'s corresponding ground truth target.

For the auxiliary task of the depth branch, we regress the depth $[t'_z]_{3d}$ of the 3D object as well as the $[t'_x,t'_y]_p$ for projecting the 3D center onto the 2D image plane, which shares the same ground truth with the detection branch:

$$L_{dep} = \tilde{L1}([t'_x,t'_y]_p,[t_x^{gt},t_y^{gt}]_p) + \tilde{L1}([t'_z]_{3d},[t_z^{gt}]_{3d}). \tag{9}$$

The final training loss is a summation of L_{det} and L_{dep}. Inspired by focal loss, we utilize classification scores s_t to balance the samples:

$$L = (1-s_t)^\gamma (L_{det} + L_{dep}), \tag{10}$$

where γ is the focus parameter and set as 0.5.

4. Experiments

Dataset We conduct experiments on the KITTI dataset [13, 14] of 7,481 and 7,518 images for training for testing, respectively. As in [1, 8], we use two train-val splits of the KITTI dataset (“val1”/“val2”) to evaluate our DDMP-3D.

Evaluation metrics Precision-recall curves are adopted for evaluation, and we report the average precision (AP) results of 3D and Bird’s eye view (BEV) object detection on

Table 1. Comparison with SoTA methods on the KITTI test set at IoU = 0.7. Our DDMP-3D achieves new SoTA performance.

Method	Reference	Speed (FPS)	AP_{3D}			AP_{BEV}			GPU
			Mod.	Easy	Hard	Mod.	Easy	Hard	
FQNet [23]	CVPR 2019	2	1.51	2.77	1.01	3.23	5.40	2.46	1080Ti
ROI-10D [26]	CVPR 2019	5	2.02	4.32	1.46	4.91	9.78	3.74	-
MonoDIS [37]	ICCV 2019	-	7.94	10.37	6.40	13.19	17.23	11.12	Tesla V100
MonoPair [6]	CVPR 2020	17	9.99	13.04	8.65	14.83	19.28	12.89	-
UR3D [36]	ECCV 2020	8	8.61	15.58	6.00	12.51	21.85	9.2	GTX Titan X
RTM3D [30]	ECCV 2020	20	10.34	14.41	8.77	14.20	19.17	11.99	1080Ti
AM3D [25]	ICCV 2019	3	10.74	16.50	9.52	17.32	25.30	14.91	1080Ti
DA-3Ddet [47]	ECCV 2020	3	11.50	16.77	8.93	15.90	23.35	12.11	Titan RTX
D^{4}LCN [8]	CVPR 2020	5	11.72	16.65	9.51	16.02	22.51	12.55	1080Ti
Kinematic3D [2]	ECCV 2020	8	12.72	19.07	9.17	17.52	26.69	13.10	-
Ours	-	6	**12.78**	**19.71**	**9.80**	**17.89**	**28.08**	**13.44**	Tesla V100

Table 2. Comparison on the KITTI "val1", "val2" set. We report the average precision (in %) including $AP|_{R40}$ and $AP|_{R11}$ of "Car" on 3D object detection (AP_{3D}) at IoU = 0.7.

Method	Val1 [$AP|_{R40}$ / $AP|_{R11}$]			Val2 [$AP|_{R40}$ / $AP|_{R11}$]		
	Mod.	Easy	Hard	Mod.	Easy	Hard
OFT-Net [34]	- / 3.27	- / 4.07	- / 3.29	- / -	- / -	- / -
FQNet [23]	- / 5.50	- / 5.98	- / 4.75	- / 5.11	- / 5.45	- / 4.45
ROI-10D [26]	- / 6.93	- / 10.25	- / 6.18	- / -	- / -	- / -
MonoDIS [37]	- / 7.60	- / 11.06	- / 6.37	- / -	- / -	- / -
MonoGRNet [31]	7.56 / 10.19	11.90 / 13.88	5.76 / 7.62	- / -	- / -	- / -
GS3D [22]	- / 10.97	- / 13.46	- / 10.38	- / 10.51	- / 11.63	- / 10.51
shift R-CNN [28]	- / 11.29	- / 13.84	- / 11.08	- / -	- / -	- / -
MonoPSR [21]	- / 11.48	- / 12.75	- / 8.59	- / 12.24	- / 13.94	- / 10.77
SS3D [18]	- / 13.15	- / 14.52	- / 11.85	- / 8.42	- / 9.45	- / 7.34
RTM3D [30]	- / 16.86	- / 20.77	- / 16.63	- / 16.29	- / 19.47	- / 15.57
M3D-RPN [1]	11.07 / 17.06	14.53 / 20.27	8.65 / 15.21	10.07 / 16.48	14.51 / 20.40	7.51 / 13.34
Pseudo-LiDAR [43]	- /18.50	- / 28.20	- / 16.40	- / -	- / -	- / -
Decoupled-3D [3]	- / 18.68	- / 26.95	- / 15.82	- / -	- / -	- / -
UR3D [36]	13.35 / 18.76	23.24 / 28.05	10.15 / 16.55	11.10 / 16.75	22.15 / 26.30	9.15 / 13.60
AM3D [25]	- / 21.09	- / **32.23**	- / 17.26	- / -	- / -	- / -
D^{4}LCN [8]	16.20 / 21.71	22.32 / 26.97	12.30 / 18.22	- / 19.54	- / 24.29	- / 16.38
Ours	**20.39 / 23.12**	**28.12** / 31.14	**16.34 / 19.45**	**19.91 / 22.92**	**29.17 / 30.66**	**15.26 / 18.75**

KITTI validation and test set. The 40 recall positions-based metric $AP|_{R40}$ has been utilized by the KITTI test server instead of $AP|_{R11}$ since Aug. 2019. To compare with previous methods that only report $AP|_{R11}$ results, we also demonstrate $AP|_{R11}$ in all experiments except the comparison with state-of-the-art (SoTA) methods on KITTI test and validation set. Three levels of difficulty are defined in the benchmark according to the 2D bounding box height, occlusion, and truncation degree, namely, "Easy", "Mod.", and "Hard". The KITTI benchmark ranks all methods based on the AP_{3D} of "Mod.". As in [5], we adopt IoU = 0.7 as threshold for "Car" category. To validate the effectiveness on "Cyclist" and "Pedestrian" categories, we include the experiments with IoU = 0.5 for fair comparison. We denote AP for 3D and BEV as AP_{3D} and AP_{BEV}, respectively.

Training details Initial anchors for detection heads are constructed as in D^{4}LCN [8]. For 2D anchors, 12 scales ranging from 30 to 400 pixels in height following the power function of $30 * 1.265^n, n = 0, ..., 11$, combining with aspect ratios of [0.5, 1.0, 1.5] to generate a total of 36 anchors. Mean statistics across matched 3D ground truth are also calculated as the initialization of 3D anchor parameters. We employ ResNet-50 [17] as feature extraction backbone for RGB and depth branch. The input image size is scaled to 512×1760 and only horizontal flipping is applied to data augmentation. 2D space Non-Maximum Suppression (NMS) with an IoU threshold of 0.4 is finally used to drop the predicted bounding box redundancy. Note that the baseline method demonstrated in all experiments stands for simply dot-multiplication on RGB and depth features together, which is equivalent to treating each pixel on depth feature as a depth kernel. The depth maps used for all experiments except for Table 6 are obtained by off-the-shelf monocular depth estimator DORN [12].

Our model is trained by SGD with an initial learning rate 0.04, momentum 0.9, and weight decay 0.0005 respectively.

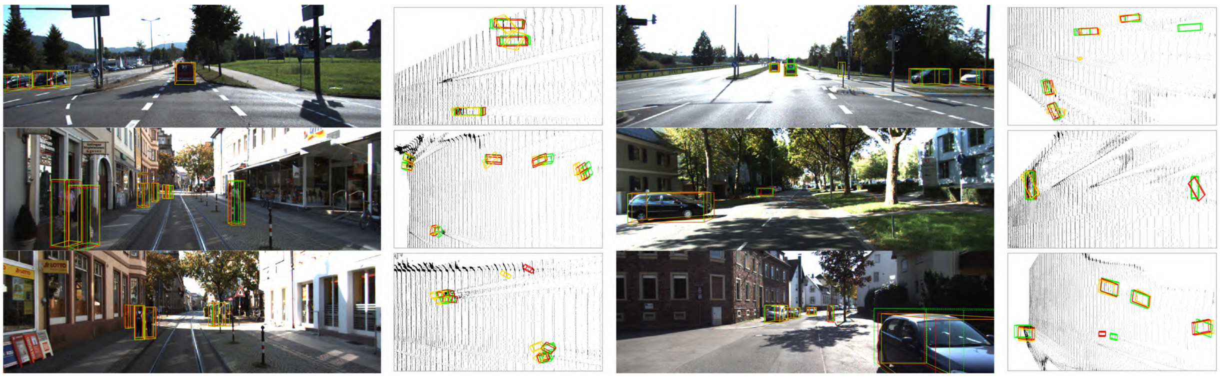

Figure 4. Qualitative comparison of ground truth (green), the baseline (yellow), and our method (red) on KITTI val set. For better visualization, the first and second columns show RGB and BEV images of point clouds converted from pre-estimated depth, respectively.

Table 3. 3D object detection ($\mathrm{AP}_{3\mathrm{D}}$) performance for "Cyclist" and "Pedestrian" on KITTI val split and test set ("val1"/test).

Method	Cyclist			Pedestrian		
	Mod.	Easy	Hard	Mod.	Easy	Hard
D^4LCN [8]	4.41 / 1.67	5.85 / 2.45	4.14 / 1.36	11.23 / 3.42	12.95 / 4.55	11.05 / 2.83
Baseline	4.07 / 0.17	4.50 / 0.32	4.08 / 0.17	6.93 / 2.32	8.11 / 3.05	6.78 / 1.81
Ours	**6.47 / 2.50**	**8.01 / 4.18**	**6.27 / 2.32**	**12.11 / 3.55**	**14.42 / 4.93**	**12.05 / 3.01**

We train the network with a batch size of 16 on 8 Nvidia Tesla v100 GPUs for 36k iterations.

4.1. Comparison with state-of-the-arts

Results on KITTI test and val set Table 1 and 2 show the 3D "Car" detection results on KITTI test and "val1", "val2" set at IoU = 0.7, respectively. We report both $AP|_{R40}$ and $AP|_{R11}$ on validation set while only $AP|_{R40}$ on test set. In the KITTI leaderboard, our proposed DDMP-3D consistently outperforms other alternatives and ranks 1st compared with the top-ranked monocular-based 3D object detection methods. Quantitatively, our method achieves the highest performance on "Moderate" set for both validation and test set, which is the main setting for ranking on the benchmark. Large performance margins are observed over the second top-performed method (D^4LCN [8]), 4.19% and 1.06% on validation and test set in view of "Mod.", respectively. This phenomenon indicates that our outstanding performance benefits from the effective context- and depth-aware features learning via depth-conditioned dynamic graph message propagation and center-aware depth feature encoding. Note that some cutting-edge monocular methods use extra information for 3D object detection, *e.g.*, AM3D [25] designs two extra modules for background points segmentation and RGB information aggregation, while we attain the appealing results and acceptable inference speed without bells and whistles.

Results on "Cyclist" and "Pedestrian" "Cyclist" and "Pedestrian" categories are much more challenging than the "Car" for monocular 3D object detection due to their non-rigid structures and small scale. We report these two categories respect to baseline and [8] in Table 3. Following [8], $\mathrm{AP}_{3\mathrm{D}}$ of "Cyclist" and "Pedestrian" on the "val1"($AP|_{R11}$) and test($AP|_{R40}$) set at IoU = 0.5 are reported. Thanks to our DDMP and CDE which provide better sensitivity to object location, we are able to localize these challenging categories to some degree and clearly outperform the alternatives.

4.2. Ablation study

Main ablative analysis In Table 4, we conduct ablation experiments to analyze the effectiveness of different components: I) Baseline: simply integrate depth features into RGB features by dot-product. II) DDMP (single-scale): depth-conditioned dynamic message propagation without multi-scale depth features. For the input of a certain DDMP, depth features from its corresponding single stage are extracted to generate affinity matrices and weights. For fair comparison, we keep the number of affinity matrices and weights the same with the following Group III. III) DDMP (multi-scale): dynamic graph message propagation via hybrid and hierarchical depth features. For fair comparison, the parameter numbers are exactly kept the same as Group II (single-scale). The critical difference with respect to Group II lies in that the depth features fed into the DDMP are from different stages rather than a single stage. IV) On the basis of Group III, CDE is added additionally to conduct a 3D center regression task with an auxiliary loss during the training phase, so as to guide the depth features to be center-aware.

As depicted in Table 4, we can observe that the performance continues to grow with the participation of components. Dynamic message propagation introduces an impres-

Table 4. Ablative analysis on KITTI "val1" split set for AP_{3D} and AP_{BEV} at IoU = 0.7. Experiment group (I) is our baseline method. Different experiment settings are explored: (II) using depth-conditioned dynamic message propagation with single scale, (III) performing multi-scale dynamic message propagation, (IV) our full approach with auxiliary center-regression task.

Group	DDMP	DDMP	CDE	AP_{3D}			AP_{BEV}		
	single-scale	multi-scale		Mod.	Easy	Hard	Mod.	Easy	Hard
I	-	-	-	18.82	26.03	16.27	24.18	33.06	19.63
II	✓	-	-	22.36	28.94	18.86	26.73	36.89	24.00
III	-	✓	-	22.84	28.12	19.09	27.05	37.11	24.20
IV	-	✓	✓	**23.12**	**31.14**	**19.45**	**27.46**	**37.71**	**24.53**

Table 5. Comparison of different auxiliary tasks on val split set.

Center	AP_{3D}			AP_{BEV}		
Regression	Mod.	Easy	Hard	Mod.	Easy	Hard
-	22.84	28.12	19.09	27.05	37.11	24.20
only z	23.07	28.11	19.19	26.95	36.70	24.06
xy	22.51	28.68	18.52	26.80	37.15	21.38
xyz	**23.13**	**31.14**	**19.45**	**27.46**	**37.71**	**24.53**

Table 6. Comparison of different depth estimators on val split set at IoU = 0.7. The first, second, and third rows show the results of D^{4}LCN [8], the baseline, and our method.

Depth	Method	AP_{3D} / AP_{BEV}		
		Mod.	Easy	Hard
DORN [12]	D^{4}LCN	21.71 / 19.54	26.97 / 24.29	18.22 / 16.38
	Baseline	18.82 / 24.18	26.03 / 33.06	16.27 / 19.63
	Ours	**23.12 / 27.46**	**31.14 / 37.71**	**19.45 / 24.53**
PSMNet [4]	D^{4}LCN	25.41 / -	30.03 / -	21.63 / -
	Baseline	25.41 / 33.34	35.26 / 46.75	20.69 / 27.27
	Ours	**30.83 / 36.20**	**41.76 / 52.87**	**24.78 / 29.34**

sive increase from 18.82% to 22.36% on the moderate setting which confirms the effectiveness of the graph message mechanism between image and depth. Group III shows that hybrid affinity matrices and weights learning leads to 0.48% performance gain on the moderate setting. It indicates that multi-scale depth features are useful in graph message propagation owing to the better depth perception on various scales of objects. Further, the auxiliary task brings a noticeable gain from 28.12% to 31.14% on the easy setting, which validates the effectiveness of the auxiliary task by learning center-aware depth features.

Qualitative comparisons of the baseline and our method are shown in Figure 4. 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 DDMP-3D can produce higher-quality 3D bounding boxes in different kinds of scenes. More quantitative and qualitative results are reported in our supplementary material.

Auxiliary task-guided depth encoding We explore the effect of different auxiliary tasks. We employ "z", "xy" and "xyz" center regressions on the depth branch (Equation 9). As shown in Table 5, depth task "z" improves the moderate 3D detection performance from 22.84% to 23.07% while auxiliary center estimation task brings substantial improvements on all the three subsets from (22.84% / 28.12% / 19.09%) to (23.13% / 31.14% / 19.45%). The improvement is especially significant on "Easy". This further suggests that center-awareness is useful for object localization.

Impact of different depth estimators For generalization ability validation on different depth estimation methods, we choose the monocular depth estimator DORN [12] as well as the more accurate stereo matching method PSMNet [4] to obtain depth maps for comparison. As shown in Table 6, the performance gain with respect to baseline and D^{4}LCN [8] are enhanced with the increasing accuracy of estimated depth. Besides, improvements can be noticed both in monocular and stereo depth predictors.

5. Conclusion

We have presented a depth-conditioned dynamic message propagation (DDMP-3D) network, a novel graph-based approach that learns context- and depth-aware feature representation for 3D object detection. It dynamically samples context-aware nodes in the image context and predicts the hybrid filter weights and affinities based on the aligned multi-scale depth features for message propagation. A center-aware depth encoding task is augmented and jointly trained with the whole network to resolve the challenge of inaccurate depth prior. This framework allows us to build a new state of the art among the monocular-based approaches, and this is demonstrated by the fact that we are rank 1^{st} in the highly competitive KITTI monocular 3D object detection track on the submission day (Nov 16th, 2020).

Acknowledgments

This work was supported by by National Key R&D Program of China (2019YFA0709502), Shanghai Municipal Science and Technology Major Project (No.2018SHZDZX01), ZJLab, Shanghai Center for Brain Science and Brain-Inspired Technology, and Shanghai Research and Innovation Functional Program (17DZ2260900).

References

[1] Garrick Brazil and Xiaoming Liu. M3d-rpn: Monocular 3d region proposal network for object detection. In *ICCV*, 2019. 1, 2, 5, 6

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

[3] Yingjie Cai, Buyu Li, Zeyu Jiao, Hongsheng Li, Xingyu Zeng, and Xiaogang Wang. Monocular 3d object detection with decoupled structured polygon estimation and height-guided depth estimation. *arXiv preprint*, 2020. 6

[4] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *CVPR*, 2018. 8

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

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

[7] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017. 2, 4

[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. *arXiv preprint*, 2019. 2, 5, 6, 7, 8

[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. 1

[10] Liang Du, Jingang Tan, Hongye Yang, Jianfeng Feng, Xiangyang Xue, Qibao Zheng, Xiaoqing Ye, and Xiaolin Zhang. Ssf-dan: Separated semantic feature based domain adaptation network for semantic segmentation. In *CVPR*, 2019. 1

[11] 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*, 2020. 1

[12] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, 2018. 2, 6, 8

[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] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. Structure aware single-stage 3d object detection from point cloud. In *CVPR*, 2020. 5

[16] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 1

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6

[18] 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*, 2019. 6

[19] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint*, 2016. 2

[20] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *IROS*, 2018. 1

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

[22] Buyu Li, Wanli Ouyang, Lu Sheng, Xingyu Zeng, and Xiaogang Wang. Gs3d: An efficient 3d object detection framework for autonomous driving. In *CVPR*, 2019. 6

[23] Lijie Liu, Jiwen Lu, Chunjing Xu, Qi Tian, and Jie Zhou. Deep fitting degree scoring network for monocular 3d object detection. In *CVPR*, 2019. 1, 2, 6

[24] 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. 1

[25] 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. 1, 2, 6, 7

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

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

[28] Andretti Naiden, Vlad Paunescu, Gyeongmo Kim, ByeongMoon Jeon, and Marius Leordeanu. Shift r-cnn: Deep monocular 3d object detection with closed-form geometric constraints. In *ICIP*, 2019. 6

[29] Erli Ouyang, Li Zhang, Mohan Chen, Anurag Arnab, and Yanwei Fu. Dynamic depth fusion and transformation for monocular 3d object detection. In *ACCV*, 2020. 2

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

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

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

[33] 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

[34] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. In *BMVC*, 2019. 5, 6

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

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

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

[38] Maximilian Speicher, Brian D. Hall, and Michael Nebeling. What is mixed reality? In *CHI Conference on Human Factors in Computing Systems*, 2019. 1

[39] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, 2019. 1

[40] Eugene Valassakis, Zihan Ding, and Edward Johns. Crossing the gap: A deep dive into zero-shot sim-to-real transfer for dynamics. In *IROS*, 2020. 1

[41] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018. 2

[42] Li Wang, Yao Lu, Hong Wang, Yingbin Zheng, Hao Ye, and Xiangyang Xue. Evolving boxes for fast vehicle detection. In *ICME*, 2017. 1

[43] 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. 1, 2, 6

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

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

[46] Bin Yang, Ming Liang, and Raquel Urtasun. Hdnet: Exploiting hd maps for 3d object detection. In *CoRL*, 2018. 5

[47] 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

[48] 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. 1, 2

[49] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016. 2

[50] Li Zhang, Dan Xu, Anurag Arnab, and Philip HS Torr. Dynamic graph message passing networks. In *CVPR*, 2020. 2, 4

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

[52] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, 2018. 1