Dense Voxel Fusion for 3D Object Detection

Anas Mahmoud Jordan S. K. Hu Steven L. Waslander
University of Toronto Robotics Institute
{nas.mahmoud, jordan.hu}@mail.utoronto.ca, steven.waslander@robotics.utias.utoronto.ca

Abstract

Camera and LiDAR sensor modalities provide complementary appearance and geometric information useful for detecting 3D objects for autonomous vehicle applications. However, current end-to-end fusion methods are challenging to train and underperform state-of-the-art LiDAR-only detectors. Sequential fusion methods suffer from a limited number of pixel and point correspondences due to point cloud sparsity, or their performance is strictly capped by the detections of one of the modalities. Our proposed solution, Dense Voxel Fusion (DVF) is a sequential fusion method that generates multi-scale dense voxel feature representations, improving expressiveness in low point density regions. To enhance multi-modal learning, we train directly with projected ground truth 3D bounding box labels, avoiding noisy, detector-specific 2D predictions. Both DVF and the multi-modal training approach can be applied to any voxel-based LiDAR backbone. DVF ranks 3^{rd} among published fusion methods on KITTI's 3D car detection benchmark without introducing additional trainable parameters, nor requiring stereo images or dense depth labels. In addition, DVF significantly improves 3D vehicle detection performance of voxel-based methods on the Waymo Open Dataset.

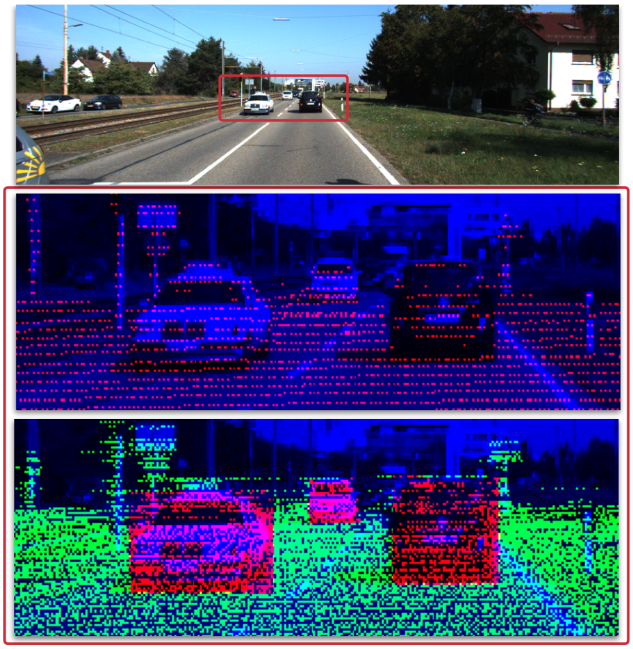

Figure 1. **Top:** an input image, **middle:** a cropped, zoomed-in image, where **red** points represent LiDAR returns projected onto the image plane, **bottom:** multi-scale dense voxel centers projected onto the image plane, where **green** and **red** points are associated with background and foreground voxel features, respectively. DVF increases the number of correspondences between image and LiDAR features.

1. Introduction

Both camera and LiDAR sensors are widely used to enable 3D perception tasks for autonomous vehicles. These sensors have differing strengths and weaknesses in terms of range, resolution, and robustness to lighting and weather conditions [11]. For instance, LiDAR point clouds provide high quality range information to 3D objects. However, the sparsity of LiDAR returns increases as a function of distance [10], leading to poorly resolved objects at long range. On the other hand, images from cameras provide a dense representation with fine-grained texture and color information even at significant range, but provide no direct depth measurement of 3D objects. Given this complementary information, 3D object detection models using camera and LiDAR should outperform their LiDAR-only counterparts. However, sequential fusion [16, 25, 14, 22] either suffers from a limited number of pixel and point correspondences due to point cloud sparsity, or their performance is strictly capped by the detections of one of the modalities. Furthermore, end-to-end fusion [26, 33, 7, 10, 9] currently underperforms relative to state-of-the-art LiDAR-only models on the KITTI and Waymo Open Dataset 3D object detection benchmarks [5, 21]. One potential reason for this performance gap is that training multi-modal networks is more challenging than uni-modal networks [23]. First, multi-modal networks tend to have more trainable parameters and are thus prone to overfitting. In addition, different modal-

ity backbones overfit and generalize at different rates, but end-to-end fusion models are usually trained jointly with a single optimization strategy which can lead to sub-optimal solutions [23]. Further, the misalignment between the viewpoint of features concatenated from LiDAR bird's-eye-view (BEV) and image feature maps in current fusion architectures [9, 3] is not ideal for 3D object detection tasks, and leads to reduced performance [22]. Sequential fusion methods fuse image predictions either at the input of the LiDAR-only detector [22, 16, 25], or by fusing the output of 3D LiDAR detectors with image predictions [14]. However, current sequential methods have limited fusion capabilities as the recall of 3D objects is strictly capped by the 2D predictions [16] or image predictions are fused only at the sparse projected point cloud [22]. Finally, effective LiDAR data-augmentation techniques like ground truth sampling [32], which accelerate the convergence of LiDAR-only methods, cannot be extended to fusion methods directly because of the missing correspondence of the added LiDAR ground truth objects in the image data.

To resolve the identified issues, we propose Dense Voxel Fusion (DVF), a sequential fusion method, that first assigns voxel centers to the 3D location of the occupied LiDAR voxel features. Voxel centers are then projected to the image plane to sample the foreground weight from image-based 2D bounding box predictions. The corresponding voxel features are then fused with image predictions using a parameter-free weighting approach. To enhance multi-modal learning, we train directly with ground truth projected 3D bounding boxes, avoiding noisy, detector-specific 2D predictions. We also use LiDAR ground truth sampling to both simulate missed 2D detections and to accelerate training convergence. We summarize our approach with two contributions:

Dense Voxel Fusion. We propose a novel dense voxel fusion method that fuses LiDAR data at the voxel feature representation level with image-based 2D bounding box predictions. Weighting voxel features instead of the sparse LiDAR point features results in dense sampling of image information (see Figure 1), improving the detection of occluded (see Table 4) and distant objects (see in the Appendix). To improve robustness against image false detections, we propose a weighting approach that minimizes the coupling of voxel and image features by weighting voxel features using image predictions, and by adding a skip connection to propagate voxel features associated with image missed detections through the LiDAR backbone. This leads to detections that are not strictly capped by image predictions. Finally, DVF does not introduce new learnable parameters and can be used with any voxel-based 3D object detector.

Multi-Modal Training. We reason that training using accurate ground truth projected 3D labels while simulating image false detections eliminates the reliance on a specific 2D object detector and generalizes better than training using erroneous 2D object detector predictions. To this end, we propose training DVF using ground truth foreground masks generated by 3D bounding box projections. To train robust fusion models, we propose using ground truth sampling [32] to simulate missed image detections. This encourages DVF to learn to detect objects in the LiDAR point cloud that are missed by the image stream and enables using ground truth sampling to accelerate learning convergence. Moreover, since amodal 2D bounding boxes are used, occluded foreground objects are implicitly used to simulate image false positives. During inference time, we propose using pixel-wise foreground aggregation of predicted 2D bounding boxes from any 2D object detector.

DVF outperforms baseline detectors and existing fusion methods when applied to voxel-based detectors. Our proposed training strategy also results in better generalization compared to training using erroneous predictions from 2D object detectors. Our test results on KITTI 3D object detection benchmark [5] rank 3^{rd} among published fusion methods without introducing additional trainable parameters, nor requiring stereo images or dense depth labels. DVF also generalizes to the substantially larger Waymo Open Dataset [21] dataset, improving performance of voxel-based methods on the *val* set.

2. Related Work

Camera and LiDAR fusion methods can either be trained jointly, in an end-to-end fashion, or independently. We categorize fusion methods based on their training scheme. Here, we present methods that only rely on ground truth 3D or 2D bounding box labels.

End-to-End Fusion. MV3D [3] and AVOD [9] utilize an independent backbone for each sensor modality. Features are extracted by projecting a shared set of 3D anchors on BEV and image features. Clearly, fusing misaligned features is not ideal for the task of 3D object detection [22]. ContFuse [26] and MMF [10] fuse image and LiDAR BEV features using a continuous fusion layer. ContFuse [26] attempts to learn a dense BEV representation by querying the nearest K 3D points corresponding to each BEV pixel, and then projecting K points onto the image plane to extract image features. Image features are then used to densify the BEV representation. A drawback of this approach is feature smearing, where multiple image feature vectors are associated with each BEV feature vector. To address the issue of limited correspondences between BEV and image features, MMF [10] learns a pixelwise dense depth map to construct Pseudo-LiDAR points [24], where continuous fusion is applied on the dense point cloud. However, improvement from dense fusion is relatively small and no improvement is

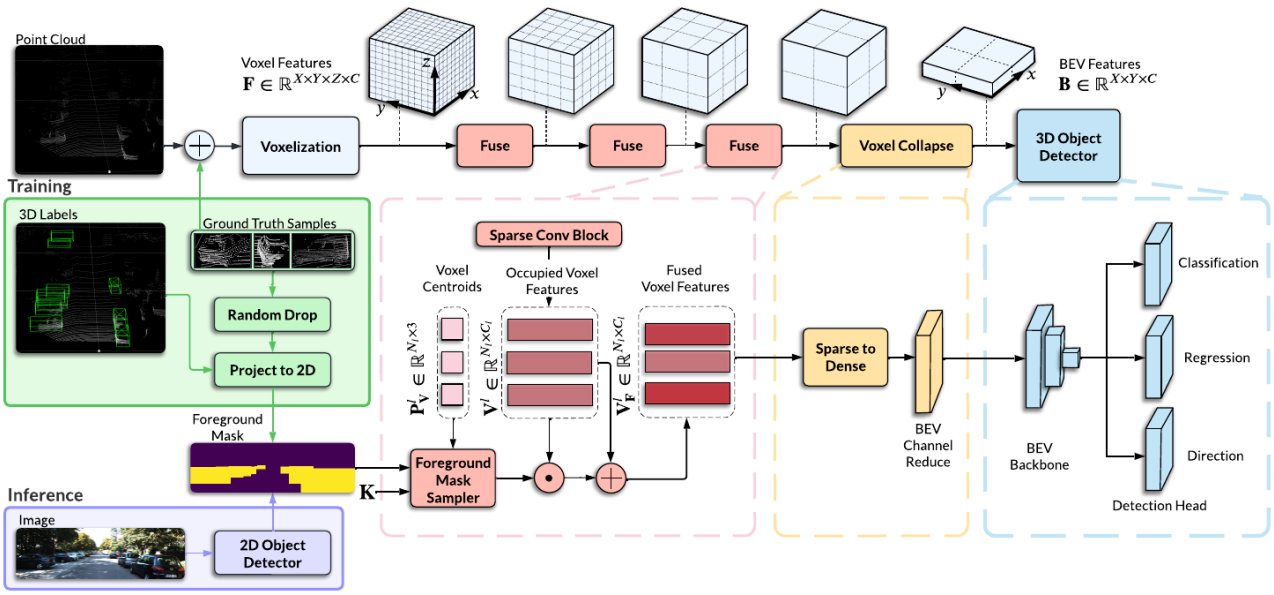

Figure 2. Dense Voxel Fusion (Section 3.1) can be applied to any voxel-based LiDAR detector without introducing new learnable parameters. During training (green), ground truth 3D bounding boxes are projected onto the image plane to construct the 2D foreground mask. In addition, ground truth samples from other scenes are used to simulate image missed detections by randomly dropping corresponding projected 3D bounding boxes from the 2D foreground mask. During inference (purple), predicted 2D bounding boxes from any 2D object detector and their associated confidences are used to construct a predicted foreground heatmap. The fusion module (Section 3.2) utilizes centers of multi-scale occupied voxels to localize voxel features and densely sample the foreground heatmap encouraging the propagation of voxel features towards foreground objects. The proposed multi-modal training strategy (Section 3.3) retains the performance of the underlying single and two-stage LiDAR detector while utilizing the predicted 2D foreground heatmap to boost the performance of 3D detectors.

observed in 3D AP on the KITTI dataset [5]. To address the challenge of view misalignment between the LiDAR's BEV and the camera's front view, 3D-CVF [33] proposes combining camera and LiDAR voxel features using the cross-view spatial feature fusion strategy, while EPNet [7] uses a point-based geometric stream [19]. One of the main challenges of jointly training the LiDAR and image streams is the different overfitting rate of each sensor modality backbone [23]. In addition, ground truth sampling [32], an effective LiDAR data-augmentation method, is not directly extendable to end-to-end fusion methods due to the difficulty of inserting additional objects into the image stream.

Sequential Fusion. Early sequential fusion methods rely on a two-stage seeding approach. F-PointNet [16] and F-ConvNet [25] reduce the search space for 3D objects, by constraining box predictions to lie within frustums generated from 2D bounding box detections. The main drawback of seeding methods is that the recall of 3D objects is strictly capped by the recall of the 2D object detector. To address this issue, PointPainting [22] generates semantic information for each pixel, and projects 3D points into the image to associate semantic information with each point to improve 3D object detection. In addition, FusionPainting [31] fuses 2D and 3D semantic segmentation predictions and then paints the input point cloud using the fused semantic labels. A drawback of point painting methods is the limited number of correspondences between 3D points and 2D image pixels as seen in the second row of Figure 1. This is especially problematic for occluded and mid-to-long range objects, where the LiDAR returns are quite sparse. In addition, the painted version of PointRCNN [19] submitted to KITTI [5] severely degrades the performance of the LiDAR-only baseline [22]. We reason using predicted segmentation scores for training is sub-optimal as performance is limited by noisy segmentation predictions. Finally, CLOCs [14] fuses the output predictions of 2D and 3D object detectors by learning to leverage multi-modal geometric and semantic consistencies, in order to predict accurate confidence scores for 3D detections. Since CLOCs is a late fusion method, detections missed by the LiDAR-only detector due to sparsity of LiDAR returns cannot be recovered from image predictions.

DVF addresses the aforementioned issues by densely fusing image predictions with LiDAR voxel features while avoiding noisy, detector-specific 2D predictions during training. In addition, DVF is capable of using LiDAR

ground truth sampling to simulate missed 2D detections and to accelerate training convergence.

3. Methodology

3.1. Overall Framework

The framework of DVF is shown in Figure 2. The network takes a point cloud as input, in addition to 3D ground truth boxes during training or 2D detections during inference. The ground truth 3D bounding boxes are projected onto the image plane to simulate 2D detections, and pixels within the 2D bounding boxes are assigned a foreground confidence sampled from a uniform distribution $\mathcal{U}_{[a,b]}$. Further, we leverage ground truth sampling to insert additional objects into the LiDAR point cloud, while randomly dropping their corresponding 2D masks in the image plane in order to simulate missed image detections. This training approach only uses LiDAR data and 3D object annotations, making the network training independent of the image-based 2D detector used at inference time (see **??**).

The point cloud is then voxelized, and a 3D sparse convolution is applied to the voxel grid. Each occupied voxel feature vector is then assigned the voxel center location and subsequently projected onto the image plane to sample from the image foreground mask. The voxel features are then weighted using the sampled 2D foreground confidence. To maintain detection performance despite image false negatives and enable extraction of contextual features from background voxels, a skip connection is added to the weighted voxel features which are passed to the next convolutional block. This process is repeated at the output of each convolutional layer to densely sample the foreground mask. The last convolutional layer features are then projected into a BEV grid and a detection head is used to predict the bounding box proposals. During the testing phase, predicted bounding boxes from any 2D object detector and their corresponding confidence values are used to construct a foreground confidence heatmap. The proposed dense fusion backbone can be used to boost the performance of any voxel-based LiDAR detector while not requiring matched 2D image labels during the training phase.

3.2. Dense Voxel Fusion

Due to the occlusion and sparsity inherent in LiDAR point clouds, we propose a dense fusion approach at the voxel level that augments LiDAR point cloud information with dense details from image data. We fuse a foreground mask constructed from the predicted 2D bounding box of any 2D object detector with any voxel-based LiDAR stream. The fusion occurs between the sparse convolution blocks of the Region Proposal Network (RPN). Each block computes a set of voxel features $\mathbf{V}^l \in \mathbb{R}^{N_l \times C_l}$, where l is the index of the convolutional block, N_l is the number of occupied voxels and C_l is the number of channels computed by the l^{th} block. Each voxel feature $\mathbf{v}_i^l \in \mathbb{R}^{C_l}$ is assigned a 3D point $\mathbf{p}_{\mathbf{v}_i}^l$. Here, $\mathbf{p}_{\mathbf{v}_i}^l$ is an instance of the set $\mathbf{P}_\mathbf{V}^l \in \mathbb{R}^{N_l \times 3}$ and corresponds to the center of the occupied voxel $\mathbf{v}_i^l$. Now that each voxel feature is localized using known camera calibration parameters $\mathbf{K}$, $\mathbf{P}_\mathbf{V}^l$ are projected onto the image plane to generate a set of 2D pixel locations $\mathbf{P}_\mathbf{C}^l \in \mathbb{R}^{N_l \times 2}$. Each pixel location $\mathbf{p}_{\mathbf{c}_i}^l \in \mathbb{R}^2$ is used to sample the foreground confidence $\rho_i^l \in [0.0, 1.0]$ via 2D interpolation from the foreground mask.

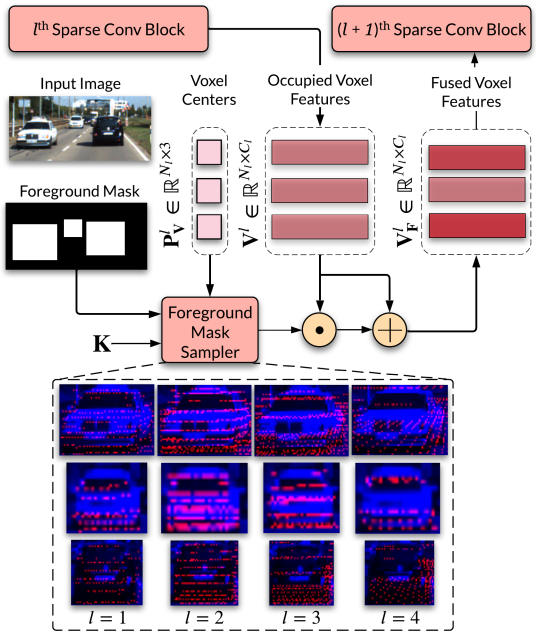

Figure 3. Illustration of Dense Voxel Fusion. The foreground mask is sampled using a set of voxel centers $\mathbf{p}_\mathbf{v}^l$ associated with occupied voxel features $\mathbf{V}^l$ learned by l^{th} sparse convolution block. Fused features from block l are processed by block $l+1$. The projection of the foreground voxels (shown as red points) in sets $\mathbf{p}_\mathbf{v}^1$, $\mathbf{p}_\mathbf{v}^2$, $\mathbf{p}_\mathbf{v}^3$, $\mathbf{p}_\mathbf{v}^4$ is depicted for the 3 cars in the scene. At each l, a new set of centers is used to sample the mask at new pixel locations, resulting in a dense correspondence between voxel features and image pixels. Figure 1 shows voxel and pixel correspondences aggregated from all 4 blocks.

To fuse sampled pixel foreground confidence and voxel features, we propose a parameter-free weighting function that minimizes the coupling of learned voxel features and image-based 2D predictions. This is contrary to sparse fusion methods [22, 11] that concatenate semantic and point features at the input point cloud resulting in the coupling of multi-modal feature extraction, and thus might not be robust to image false positives and false negatives. The proposed method weights each voxel feature $\mathbf{v}_i^l$ with the sampled pixel foreground confidence ρ_i^l using:

$$\mathbf{v}_{\mathbf{f}_i}^l = \rho_i^l \mathbf{v}_i^l + \mathbf{v}_i^l \quad \forall i \in [0, N_l - 1] \tag{1}$$

Figure 4. 2D foreground heatmap generation during testing time. Darker pixels indicate lower probability of a pixel belonging to a foreground object.

where $\mathbf{v}^l_{\mathbf{f}_i}$ is the weighted feature vector. By weighting voxel features based on the foreground confidence, fused voxel feature vectors stay in the same feature space prior to the fusion step (i.e., $\mathbf{v}^l_i$, $\mathbf{v}_{\mathbf{f}}{}^l_i \in \mathbb{R}^{C_l}$) effectively decoupling geometric feature extraction from erroneous 2D semantic labels. To address the challenge of image false negatives, a skip connection is added to preserve the voxel features where objects are not detected by the camera stream. In addition, by preserving occupied background voxel features, contextual information can be aggregated which is useful for detecting and accurately localizing foreground objects.

Figure 3 depicts DVF applied to the output of 4 sequential sparse convolutional blocks. The sparse convolutional block consists of a downsampling step which reduces the resolution of the 3D voxel grid by a factor of 2. Due to the downsampling step and the propagation of voxel features to neighbouring voxels after the convolution operation, a new set of voxel features are computed resulting in a new set of voxel centers at the output of each block l. To enable dense fusion, multi-scale voxel centers are used to sample the foreground mask. Figure 3 shows the foreground voxel centers projected onto the 3 foreground objects in the scene (shown as red points). Here, we show the projection of foreground centers at the output of blocks $l = 1, \ldots, 4$. Repeatedly sampling the foreground mask at multi-scale voxel centers leads to an increase in the number of correspondences between the voxel features and image-pixels far beyond the sparse correspondence between 3D points and image-pixels.

3.3. Multi-Modal Training Strategy

Training multi-modal networks is challenging mainly due to the different rates at which the backbone of each modality overfits [23]. To overcome this issue, the image-based 2D object detector is not trained jointly with the 3D detector. Therefore, DVF is robust to image false positives and false negatives that are otherwise introduced when jointly training a 2D object detector. Our training strategy consists of training using ground truth projected 3D bounding boxes while simulating image-based false positives and negatives throughout the training phase. Our key finding (see Table 7) is that training the 3D detector with noise-free ground truth foreground masks while simulating 2D object detection failures generalizes much better on single and two-stage detectors when compared to training with predicted 2D bounding boxes.

Foreground Heatmap Generation As depicted in Figure 2, during the training phase, 3D ground truth bounding boxes are used to construct the foreground mask. The corners of each 3D bounding box are projected onto the image plane using known camera calibration parameters. Then, an axis-aligned 2D bounding box is computed for each set of corners associated with a foreground object in the scene. Since 3D ground truth bounding boxes are used, the proposed fusion method does not require 2D image labels, nor does it require linking of 2D and 3D ground truth boxes. Training directly with ground truth 2D bounding box labels ensures all foreground objects in the scene are accurately recalled by the image stream, providing the fusion model with consistent correspondence between LiDAR and image information. To simulate varying class confidence of predicted bounding boxes, the class confidence of projected 3D bounding boxes is drawn from a uniform distribution $\mathcal{U}_{[a,b]}$. In our experiments, we set $a = 0.8$ and $b = 1.0$ to simulate detections with high confidence.

Figure 4 illustrates the foreground mask generated during testing time. For input image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$, a set of M 2D bounding boxes $\mathbf{B} \in \mathbb{R}^{N \times 4}$ are predicted. $\mathbf{b}^m \in \mathbf{B}$ corresponds to the m^{th} bounding box boundaries denoted by u^m_1, u^m_2, v^m_1 and v^m_2. Each $\mathbf{b}^m$ is associated with a predicted confidence $c^m \in [0.0, 1.0]$. Figure 4 shows $M = 6$ predictions. For each $\mathbf{b}^m$, a foreground mask $\mathbf{f_r} \in \mathbb{R}^{H \times W}$ is populated using:

$$\mathbf{fr}^m(i,j) = \begin{cases} c^m; & \text{if } v^m_1 \leq i \leq v^m_2 \\ & u^m_1 \leq j \leq u^m_2 \qquad \forall i,j \\ 0.0; & \text{otherwise} \end{cases} \tag{2}$$

The foreground mask used for inference aggregates M individual foreground masks using the pixel-wise max operator in (3) to preserve high confidence values where foreground objects are partially occluded by each other (e.g., second and third prediction from top in Figure 4).

$$\mathbf{fr}_{agg}(i,j) = \max\{\mathbf{fr}^0(i,j), ..., \mathbf{fr}^{M-1}(i,j)\} \quad \forall i,j \tag{3}$$

Simulating Missed Detections To simulate image false negatives during training, K random samples corresponding to ground truth objects from point cloud scenes in the training set are appended to the point cloud. A random subset of the K samples are not projected onto the foreground mask and thus, the 3D object detector is trained to detect foreground objects that are completely missed by the image stream. DVF adds a skip connection to the learned voxel

features, which preserves the voxel features of foreground objects incorrectly labelled as background voxels due to image missed detections, enabling the use of ground truth sampling. Training LiDAR detectors with ground truth sampling [32] accelerates convergence, however, many fusion methods, do not employ ground truth sampling due to the missing image correspondence, which can result in a large gap between fusion and LiDAR-only detectors [11]. Our multi-modal training strategy enables utilizing ground truth sampling for accelerating convergence of fusion models while improving robustness against image missed detections.

4. Experimental Results

We evaluate the effectiveness of DVF on KITTI 3D object detection benchmark [5] and Waymo Open Dataset [21] applied to: (1) Voxel-based single-stage detector SECOND [32], (2) Voxel-based two-stage detector Voxel-RCNN [4] and (3) Point-Voxel-based two-stage detector PV-RCNN [18]. The KITTI 3D object detection benchmark [5] is split into 7,481 and 7,518 training and testing samples respectively. The training samples are further split into 3,712 samples for *train* set and 3,769 samples for *val* set [2]. Our evaluation is done using average precision with 40 recall points $AP|_{R40}$ and an IoU threshold of 0.7, 0.5 and 0.5 for car, pedestrian and cyclist classes respectively. Waymo Open Dataset [21] consists of 798 training and 202 validation sequences. We train on 10% from the training set (15,467 samples) and evaluate on the truncated point cloud within the front camera field-of-view (50.4°). Waymo Open Dataset uses both standard average precision (AP) and heading (APH) for evaluation metrics with an IoU threshold of 0.7 for vehicles. Waymo Open Dataset is split into two difficulties: LEVEL_1 only includes 3D labels with more than five LiDAR points, while LEVEL_2 considers all 3D labels with at least one LiDAR point.

Input Parameters The initial 3D grid is constructed based on the range of the point cloud input and the grid resolution set for each dimension in 3D space. KITTI uses a point cloud range of $[0.0, 70.0]$, $[-40.0, 40.0]$ and $[-1.0, 3.0]$ meters that are voxelized into a grid resolution of $(0.05, 0.05, 0.1)$ along X, Y and Z, respectively. Waymo Open Dataset uses a point cloud range of $[0.0, 75.2]$, $[-75.2, 75.2]$ and $[0, 4]$ meters and a grid resolution of $(0.1, 0.1, 0.15)$ along X, Y and Z, respectively. Only points in the field-of-view of the front camera are considered in both datasets.

Training and Inference Details The sparse convolution backbone with DVF is implemented in PyTorch [15]. All baselines and DVF models are trained using 2 NVIDIA Tesla P100 GPUs with a batch size of 2 for 80 epochs on KITTI and 4 NVIDIA Tesla V100 GPUs with a batch size of 4 for 50 epochs on Waymo Open Dataaset. Adam [8] optimizer and one-cycle learning rate policy is used [20].

Method	Modality	Reference	Car - 3D AP		
			Easy	Mod.	Hard
SA-SSD [6]	L	CVPR-2020	88.75	79.79	74.16
PV-RCNN [18]	L	CVPR-2020	90.25	81.43	76.82
Voxel R-CNN [4]	L	AAAI-2021	90.90	81.62	77.06
CT3D [17]	L	ICCV-2021	87.83	81.77	77.16
Pyramid-PV [12]	L	ICCV-2021	88.39	82.08	77.49
VoTr-PV [13]	L	ICCV-2021	89.90	82.09	**79.14**
SPG-PV [30]	L	ICCV-2021	90.50	82.13	78.90
SE-SSD [34]	L	CVPR-2021	**91.49**	82.54	77.15
BtcDet [29]	L	AAAI-2022	90.64	**82.86**	78.09
MV3D [3]	L+I	CVPR-2017	74.97	63.63	54.00
ContFuse [26]	L+I	ECCV-2018	83.68	68.78	61.67
F-PointNet [16]	L+I	CVPR-2018	82.19	69.79	60.59
AVOD-FPN [9]	L+I	IROS-2018	83.07	71.76	65.73
MMF [10]	L+I	CVPR-2019	88.40	77.43	70.22
EPNet [7]	L+I	ECCV-2020	89.81	79.28	74.59
3D-CVF [33]	L+I	ECCV-2020	89.20	80.05	73.11
CLOCs-PV [14]	L+I	IROS-2020	88.94	80.67	77.15
DVF-PV (ours)	L+I	-	**90.99**	**82.40**	**77.37**

Table 1. 3D detection results on the KITTI *test* set using $AP|_{R_{40}}$ metric. Models ending with PV use PV-RCNN as their base LiDAR architecture. L and I represent the LiDAR point cloud and the camera image, respectively.

During training, LiDAR-based global data-augmentation strategies are applied, including global scaling, global rotation around Z axis, and random flipping along the X axis. For DVF models, all global linear transformations of the point cloud are constructed and then reversed before using known camera parameters to sample either the ground truth foreground mask or the predicted foreground heatmap. For training DVF models, we set the number of ground truth samples K to 5 for each class and the dropout percentage to 50%. During training phase, the foreground mask is constructed using 3D ground truth bounding box projections. For inference, pixel-wise aggregation, as outlined in (3), is used to compute the input predicted foreground heatmap using of Cascade-RCNN [1] 2D detections. For experiments on KITTI *val* set, we use 2D detections from Cascade-RCNN [1] published by CLOCs [14]. For KITTI *test* set results, we train Cascade-RCNN implemented in Detectron2 [28] on *train* and *val* set. For inference on Waymo Open Dataset *val* set, 2D predictions from a Cascade-RCNN [1] detector pre-trained on COCO dataset is used.

4.1. KITTI Dataset Results

Table 1 shows the results of DVF applied to PV-RCNN [18] (DVF-PV) on the KITTI [5] *test* set compared to state-of-the-art LiDAR-only and Camera and LiDAR fusion methods. Each group is listed in order of performance of the moderate difficulty level car class. DVF-PV outperforms state-of-the-art fusion method CLOCs [14] on $AP|_{R_{40}}$ by +2.05%, +1.73% and +0.22% on the car class

on easy, moderate, and hard difficulties, respectively. It is important to note that DVF-PV and CLOCs-PV both use PV-RCNN [18] and Cascade-RCNN [1] for 3D and 2D detections respectively. Moreover, DVF-PV outperforms PV-RCNN [18] on $AP|_{R_{40}}$ by +0.74%, +0.97% and +0.55% on the car class on easy, moderate, and hard difficulties, respectively. PV-RCNN [18] learns foreground segmentation of voxel features using LiDAR point supervision yet underperforms compared to weighting voxel features using dense image predictions via DVF. SE-SSD [34] is a LiDAR single stage detector that employs a consistency loss between teacher and student predictions. While BtcDet [29] learns object shape priors for partially occluded objects. These techniques are orthogonal to our proposed fusion framework and incorporating them will be a goal for future work. Finally, current state-of-the-art fusion method SFD [27] requires dense depth labels to train the depth completion network, while VPFNet [35] requires stereo images. Both methods have only been tested on KITTI dataset and can only be applied to two-stage detectors.

We also show in Table 2 that DVF can be used to improve the performance of Voxel-based single-stage detectors SECOND [32], Voxel-based two-stage detector Voxel-RCNN [4] and Point-Voxel-based two-stage detectors PV-RCNN [18]. The relative gains achieved on the *val* set for DVF + PV-RCNN generalize to the *test* set. On the contrary, using the same 3D and 2D detectors, CLOCs-PV [14] achieves comparable performance to DVF on *val* set, though these gains do not persist on the *test* set.

Method	Car - 3D AP			Car - BEV AP		
	Easy	Mod.	Hard	Easy	Mod.	Hard
SECOND	90.29	81.83	79.01	92.60	90.02	88.05
DVF + SECOND	**92.03**	**82.84**	**79.72**	**94.43**	**90.85**	**88.35**
Improvement	*+1.74*	*+1.01*	*+0.71*	*+1.83*	*+0.83*	*+0.30*
Voxel-RCNN	92.44	85.20	82.89	95.58	91.30	89.01
DVF + Voxel-RCNN	**92.96**	**85.81**	**83.20**	**96.24**	**91.93**	**89.55**
Improvement	*+0.52*	*+0.61*	*+0.31*	*+0.66*	*+0.63*	*+0.54*
PV-RCNN	91.88	84.83	82.55	93.73	90.65	88.55
DVF + PV-RCNN	**93.07**	**85.84**	**83.13**	**96.21**	**91.66**	**89.17**
Improvement	*+1.19*	*+1.01*	*+0.58*	*+2.48*	*+1.01*	*+0.62*

Table 2. 3D detection results on KITTI *val* set for SECOND [32], Voxel-RCNN [4] and PV-RCNN [18] on car class. Results are shown using 3D and BEV AP $|_{R_{40}}$.

Method	Vehicle LEVEL_1		Vehicle LEVEL_2	
	AP	APH	AP	APH
SECOND	60.93	60.35	56.43	55.89
SECOND + DVF	**61.99**	**61.41**	**57.45**	**56.91**
Improvement	*+1.06*	*+1.06*	*+1.02*	*+1.02*
Voxel-RCNN	64.96	64.32	60.15	59.55
Voxel-RCNN + DVF	**67.17**	**66.53**	**62.18**	**61.58**
Improvement	*+2.21*	*+2.21*	*+2.03*	*+2.03*
PV-RCNN	62.32	61.30	57.65	56.70
PV-RCNN + DVF	**67.62**	**67.09**	**62.66**	**62.17**
Improvement	*+5.30*	*+5.79*	*+5.01*	*+5.47*

Table 3. 3D detection results on Waymo Open Dataset *val* set for SECOND [32], Voxel-RCNN [4], and PV-RCNN [18]. Results are shown using AP and APH.

4.2. Waymo Open Dataset Results

Section 4.2 shows the results of DVF on the larger and more complex Waymo Open Dataset *val* set. Even with the lower voxel resolution, DVF provides meaningful dense voxel features with an increase of +1% AP/APH on SECOND, a +2% AP/APH increase on Voxel-RCNN and +5% AP/APH increase on PV-RCNN. We attribute the large gain on Waymo Open Dataset to the intra-class variance within the Waymo Open Dataset vehicle class. Unlike KITTI, Waymo Open Dataset vehicles vary in size and local geometry significantly, encapsulating both small objects, such as motorcycles, and large objects, such as trucks. The 2D foreground image information provides complementary information to gauge different vehicle types for accurate 3D object detection, especially when leveraging a second stage for bounding box refinement.

4.3. Ablation Studies

We provide ablation studies on our proposed fusion method and training strategy.

Point versus Voxel Fusion In Table 4, we compare the performance of DVF to PointPainting [22], a sparse fusion method. Both fusion methods use the proposed training strategy of using ground truth 2D labels while simulating image false detections. We conduct inference using 2D predictions from Cascade-RCNN [1]. In addition, during inference, we utilize ground truth 2D bounding boxes for all models to determine an upper bound on the potential gains for PointPainting and DVF. Using Cascade-RCNN predictions, DVF achieves a relative 3D $AP|_{R_{40}}$ gain over PointPainting of +0.57%, +1.73%, +0.62% for SECOND [32], Voxel-RCNN [4] and PV-RCNN [18] for car moderate settings, respectively. This can be attributed to DVF substantially increasing the number of correspondences between

Method	**C-RCNN** Car - 3D AP			**GT** Car - 3D AP		
	Easy	Mod.	Hard	Easy	Mod.	Hard
Painted SECOND	91.26	82.27	79.43	91.24	82.31	79.57
DVF + SECOND	**92.03**	**82.84**	**79.72**	**92.11**	**85.36**	**82.64**
Improvement	*+0.77*	*+0.57*	*+0.29*	*+0.87*	*+3.05*	*+3.07*
Painted Voxel-RCNN	92.74	84.08	82.54	92.73	84.99	83.11
DVF + Voxel-RCNN	**92.96**	**85.81**	**83.20**	**95.50**	**86.72**	**85.70**
Improvement	*+0.22*	*+1.73*	*+0.66*	*+2.77*	*+1.73*	*+2.59*
Painted PV-RCNN	92.47	85.22	82.68	92.85	86.20	83.66
DVF + PV-RCNN	**93.07**	**85.84**	**83.13**	**94.44**	**86.71**	**86.22**
Improvement	*+0.60*	*+0.62*	*+0.45*	*+1.59*	*+0.51*	*+2.56*

Table 4. Comparison between Pointpainting [22] and DVF. The same training strategy is used for both fusion models. Inference on the KITTI *val* set is performed using both using Cascade-RCNN [1] 2D predictions **C-RCNN** and projected ground truth 3D bounding boxes **GT**. Improvements in 3D$AP|_{R_{40}}$ are shown relative to Painted models.

Method	Car - 3D AP			Pedestrian - 3D AP			Cyclist - 3D AP		
	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
Painted SECOND	**90.87**	81.16	78.79	**57.65**	**53.38**	**49.16**	82.34	65.49	61.77
DVF +SECOND	90.83	**82.12**	**79.17**	57.00	52.82	48.70	**86.72**	**66.70**	**62.68**
Improvement	*-0.04*	*+0.96*	*+0.38*	*-0.65*	*-0.56*	*-0.47*	*+4.37*	*+1.21*	*+0.91*
Painted VoxelRCNN	92.38	84.42	82.71	68.56	62.09	57.34	90.08	71.74	**68.27**
DVF +VoxelRCNN	**92.45**	**85.28**	**82.84**	**70.13**	**62.76**	**57.65**	**90.93**	**72.60**	68.24
Improvement	*+0.07*	*+0.85*	*+0.13*	*+1.57*	*+0.67*	*+0.31*	*+0.85*	*+0.86*	*-0.03*
Painted PV-RCNN	92.30	84.85	82.75	65.32	58.20	53.82	88.95	70.87	67.33
DVF +PV-RCNN	**92.34**	**85.25**	**82.97**	**66.08**	**59.18**	**54.68**	**90.93**	**72.46**	**68.05**
Improvement	*+0.04*	*+0.40*	*+0.22*	*+0.75*	*+0.98*	*+0.86*	*+1.98*	*+1.59*	*+0.72*

Table 5. Multi-class 3D detection results on KITTI *val* set. Improvements in 3D$AP|_{R_{40}}$ are shown relative to Painted models.

Method	Points Dropped	Car - 3D AP			Pedestrian - 3D AP			Cyclist - 3D AP		
		Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
Painted SECOND	10%	**90.06**	80.03	77.22	56.38	51.09	46.64	80.28	62.51	58.71
DVF + SECOND		89.58	**81.74**	**78.93**	**57.33**	**52.71**	**48.32**	**84.11**	**63.97**	**60.30**
Improvement		*-0.48*	*+1.71*	*+1.70*	*+0.95*	*+1.62*	*+1.68*	*+3.83*	*+1.46*	*+1.59*
Painted SECOND	20%	90.11	79.36	76.72	55.65	50.69	45.99	80.66	62.02	58.16
DVF + SECOND		**90.51**	**81.54**	**78.18**	**57.32**	**52.61**	**47.89**	**82.51**	**62.31**	**58.76**
Improvement		*+0.40*	*+2.18*	*+1.46*	*+1.67*	*+1.92*	*+1.90*	*+1.86*	*+0.28*	*+0.61*

Table 6. Effect of point cloud sparsity on DVF and pointpainting. The points dropped column corresponds to the percentage of point cloud dropped before training both models.

image pixels and multi-scale occupied voxel centers compared to the limited number of correspondences between image pixels and 3D points. In addition, DVF achieves a consistent relative gain over PointPainting of more than +2.50% on hard setting for all models when ground truth 2D labels $\mathbf{GT}$ are used, indicating that densely fusing features is especially beneficial for far-away and occluded objects with fewer LiDAR points (see in the Appendix). Experiments with $\mathbf{GT}$ 2D labels demonstrate the significant gain of DVF models over sparse methods with improvements in 2D object detectors. Finally, we also show in Table 5 that DVF can generalize to a multi-class setting.

Effect of Point Cloud Sparsity DVF increases the number of correspondences between LiDAR and image data as seen in Figure 1. This is contrary to Pointpainting [22] where fusion is at the 3D point level, and is therefore sparse, especially at mid-to-long range. We conduct an experiment on the effect of point cloud sparsity on the relative performance between Pointpainting and DVF using SECOND [32]. In Table 6 we randomly drop 10% and 20% of the points and train SECOND models. DVF outperforms Pointpainting over all classes. We attribute this to densely fusing image detections at the voxel rather than the point level. This also shows that DVF can be useful when LiDAR returns become more sparse in adverse weather conditions (i.e., rain or snow). Furthermore, we show using range-based evaluation that dense fusion is especially useful at far range, where more objects are occluded and LiDAR returns are sparse (see in the Appendix).

Effect of Training Using Ground Truth Labels Table 7 shows the results of training DVF models using ground truth 2D labels versus training using 2D predictions from Cascade-RCNN [1]. Inference is done using predictions from Cascade-RCNN. Training with ground truth labels generalizes better on the *val* set and outperforms training on Cascade-RCNN [1] predictions by $+1.45\%$, $+0.67\%$, $+1.85\%$ for SECOND [32], Voxel-RCNN [4] and PV-RCNN [18] respectively for the car moderate setting. Training using erroneous bounding box predictions provided by Cascade-RCNN, result in an incorrect association between image predictions and LiDAR voxel features. In addition, adding ground truth-sampled LiDAR objects without projection to the image plane increases the number of 2D missed detections. This leads to a fusion model that learns not to rely on the inconsistent 2D predictions. On the other hand, not using ground truth sampling severely degrades the performance of the fusion model, making it less robust to 2D missed detections. Training directly with ground truth 2D bounding box labels ensures all foreground objects in the scene are accurately recalled by the image stream, providing the fusion model with a consistent correspondence between LiDAR and image information. Since perfect 2D recall is achieved during training, ground truth sampling can be introduced to simulate image missed detections and to accelerate convergence. This multi-modality training approach leads to better generalization compared to erroneous 2D predictions and decouples the training of the image-based 2D detector from the 3D LiDAR detector.

Method	2D Source		Car - 3D AP		
	C-RCNN	GT	Easy	Mod.	Hard
DVF + SECOND	✓		90.37	81.39	78.69
		✓	**92.03**	**82.84**	**79.72**
Improvement			*+1.66*	*+1.45*	*+1.03*
DVF + Voxel-RCNN	✓		92.42	85.14	82.86
		✓	**92.96**	**85.81**	**83.20**
Improvement			*+0.54*	*+0.67*	*+0.34*
DVF + PV-RCNN	✓		92.10	83.99	82.73
		✓	**93.07**	**85.84**	**83.13**
Improvement			*+0.97*	*+1.85*	*+0.40*

Table 7. Comparison between training DVF using ground truth 2D bounding boxes and Cascade-RCNN [1] 2D predictions. Inference on KITTI *val* set is performed using Cascade-RCNN 2D predictions.

5. Conclusion

We present DVF, a novel fusion method that generates multi-scale multi-modal dense voxel feature representations, improving expressiveness in low point density regions. DVF can be applied to any voxel-based LiDAR backbone without introducing additional learnable parameters. Our test results on KITTI 3D object detection benchmark [5] rank 3^{rd} among published fusion methods without introducing additional trainable parameters, nor requiring stereo images or dense depth labels. DVF also generalizes to the substantially larger Waymo Open Dataset [21] dataset, improving performance of voxel-based methods on the *val* set. Our multi-modal training strategy relies solely on LiDAR data and object annotations, making the network training independent of the 2D detector that is used at inference time.

References

[1] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: High quality object detection and instance segmentation, 2019.

[2] 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*, volume 28, 2015.

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

[4] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. *AAAI*, pages 1201–1209, 2021.

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

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

[7] Tengteng Huang, Zhe Liu, Xiwu Chen, and X. Bai. Epnet: Enhancing point features with image semantics for 3d object detection. In *ECCV*, 2020.

[8] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[9] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander. Joint 3d proposal generation and object detection from view aggregation. In *IROS*, 2018.

[10] M. Liang, B. Yang, Y. Chen, R. Hu, and R. Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *CVPR*, 2019.

[11] Anas Mahmoud and Steven L. Waslander. Sequential fusion via bounding box and motion pointpainting for 3d objection detection. In *CRV*, 2021.

[12] Jiageng Mao, Minzhe Niu, Haoyue Bai, Xiaodan Liang, Hang Xu, and Chunjing Xu. Pyramid r-cnn: Towards better performance and adaptability for 3d object detection. In *ICCV*, October 2021.

[13] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. Voxel transformer for 3d object detection. In *ICCV*, October 2021.

[14] Su Pang, Daniel Morris, and Hayder Radha. Clocs: Camera-lidar object candidates fusion for 3d object detection. In *IROS*, 2020.

[15] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[16] 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*, June 2018.

[17] Hualian Sheng, Sijia Cai, Yuan Liu, Bing Deng, Jianqiang Huang, Xian-Sheng Hua, and Min-Jian Zhao. Improving 3d object detection with channel-wise transformer. In *ICCV*, October 2021.

[18] Shaoshuai Shi, Chaoxu Guo, L. Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. *CVPR*, pages 10526–10535, 2020.

[19] S. Shi, X. Wang, and H. Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *CVPR*, 2019.

[20] Leslie N. Smith. A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay, 2018.

[21] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, Benjamin Caine, V. Vasudevan, Wei Han, J. Ngiam, Hang Zhao, A. Timofeev, S. Ettinger, Maxim Krivokon, A. Gao, Aditya Joshi, Y. Zhang, Jon Shlens, Zhi-Feng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, 2020.

[22] Sourabh Vora, Alex H. Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential fusion for 3d object detection. In *CVPR*, June 2020.

[23] Weiyao Wang, Du Tran, and Matt Feiszli. What makes training multi-modal classification networks hard? In *CVPR*, June 2020.

[24] 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*, June 2019.

[25] Z. Wang and K. Jia. Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3d object detection. In *IROS*, 2019.

[26] Zining Wang, Wei Zhan, and Masayoshi Tomizuka. Fusing bird view LIDAR point cloud and front view camera image for deep object detection. *CoRR*, abs/1711.06703, 2017.

[27] Xiaopei Wu, Liang Peng, Honghui Yang, Liang Xie, Chenxi Huang, Chengqi Deng, Haifeng Liu, and Deng Cai. Sparse fuse dense: Towards high quality 3d detection with depth completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5418–5427, 2022.

[28] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[29] Qiangeng Xu, Yiqi Zhong, and Ulrich Neumann. Behind the curtain: Learning occluded shapes for 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2893–2901, 2022.

[30] Qiangeng Xu, Yin Zhou, Weiyue Wang, Charles R. Qi, and Dragomir Anguelov. Spg: Unsupervised domain adaptation for 3d object detection via semantic point generation. In *ICCV*, October 2021.

[31] Shaoqing Xu, Dingfu Zhou, Jin Fang, Junbo Yin, Zhou Bin, and Liangjun Zhang. Fusionpainting: Multimodal fusion with adaptive attention for 3d object detection. *ITSC*, 2021.

[32] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10), 2018.

[33] Jin Hyeok Yoo, Yecheol Kim, Jisong Kim, and Jun Won Choi. 3d-cvf: Generating joint camera and lidar features using cross-view spatial feature fusion for 3d object detection. In *ECCV*, 2020.

[34] Wu Zheng, Weiliang Tang, Li Jiang, and Chi-Wing Fu. Se-ssd: Self-ensembling single-stage object detector from point cloud. In *CVPR*, June 2021.

[35] Hanqi Zhu, Jiajun Deng, Yu Zhang, Jianmin Ji, Qiuyu Mao, Houqiang Li, and Yanyong Zhang. Vpfnet: Improving 3d object detection with virtual point based lidar and stereo data fusion. *arXiv preprint arXiv:2111.14382*, 2021.