Fast-CLOCs: Fast Camera-LiDAR Object Candidates Fusion for 3D Object Detection

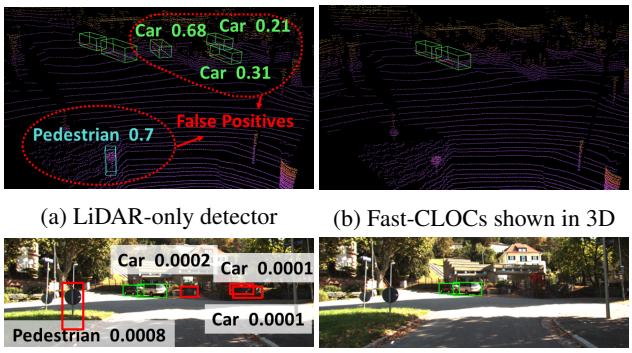
Su Pang, Daniel Morris, and Hayder Radha
Michigan State University
{pangsu,dmorris,radha}@msu.edu

Abstract

When compared to single modality approaches, fusion-based object detection methods often require more complex models to integrate heterogeneous sensor data, and use more GPU memory and computational resources. This is particularly true for camera-LiDAR based multimodal fusion, which may require three separate deep-learning networks and/or processing pipelines that are designated for the visual data, LiDAR data, and for some form of a fusion framework. In this paper, we propose Fast Camera-LiDAR Object Candidates (Fast-CLOCs) fusion network that can run high-accuracy fusion-based 3D object detection in near real-time. Fast-CLOCs operates on the output candidates before Non-Maximum Suppression (NMS) of any 3D detector, and adds a lightweight 3D detector-cued 2D image detector (3D-Q-2D) to extract visual features from the image domain to improve 3D detections significantly. The 3D detection candidates are shared with the proposed 3D-Q-2D image detector as proposals to reduce the network complexity drastically. The superior experimental results of our Fast-CLOCs on the challenging KITTI and nuScenes datasets illustrate that our Fast-CLOCs outperforms state-of-the-art fusion-based 3D object detection approaches. We will release the code upon publication.

1. Introduction

The ability of an autonomous vehicle to perform 3D object detection is essential for safe planing and navigation. Significant progress has been made in both 3D object detection using LiDAR [42, 37, 28, 32, 19, 31], and 2D object detection[1] using video [30, 5, 22, 23]. LiDAR is able to generate accurate 3D measurements but has low vertical resolution. Fig 1 illustrates the difficulty for LiDAR-only detector in detecting objects from just a few points and no texture in point cloud. Cameras can provide abundant texture information at very high resolution but no direct depth information. The complementary nature of these two types of sensors has inspired the development of mutimodal fusion methods to achieve better perception. However, it has been challenging to design fusion-based neural networks for effectively exploiting both modalities in a way that demonstrates improvements over single-modality-based methods. Most of the leading entries in the challenging open 3D detection benchmarks, such as KITTI [11] and nuScenes [2], are LiDAR-only based methods. In addition, fusion methods typically add computational complexity, and consequently, do not achieve real time operation.



(a) LiDAR-only detector (b) Fast-CLOCs shown in 3D

(c) 3D-Q-2D detector (d) Fast-CLOCs shown in img

Figure 1: Fast-CLOCs can leverage the proposed 3D-Q-2D detector to remove false positive detections. (a) LiDAR-only detector has four false positives. The numbers shown in the figure are LiDAR confidence scores. (c) The 3D-Q-2D detector suppresses these false positives by allocating them very small visual confidence scores. These LiDAR false positives are more easily removed with image appearance information. (b) and (d) show the proposed Fast-CLOCs fusion result with false positives removed.

LiDAR-camera fusion methods can be divided into three categories. First, cascaded fusion, also referred to as 2D driven 3D detection, exploits 2D detectors to narrow down the 3D processing field and guide the 3D detector. Second, integrated fusion, which combines 3D point cloud and 2D images either at the input or at an intermediate stage.

[1]Note that in this paper, we use "2D detections" to denote detections in the camera RGB image plane, not a top-down bird-eye view.

This architecture has the best possible opportunity for cross-modal interaction, but at the same time inherent challenges regarding data differences between modalities including alignment, representation, and sparsity [9, 18, 36, 21]. Third, separated fusion, also referred to as late fusion or detection-level fusion, which processes each modality on a separate path and fuses the outputs at the decision level. This architecture is easy to train, as single modality algorithms can be trained using their own sensor data, but involves less interaction between different modalities.

Recently, a Camera-LiDAR Object Candidates (CLOCs) based fusion has been proposed [26]. This performs detection-level fusion using any pair of pre-trained 2D and 3D detectors to generate better 2D/3D detections. The CLOCs fusion step is fast and adds negligible delay to the perception system. More importantly, CLOCs provides significant improvements to the performance of the underlying 3D and 2D detection methods used. However, by requiring a 2D detector and a 3D detector to run simultaneously, CLOCS uses high GPU memory and compute power. We propose Fast-CLOCs: Fast Camera-LiDAR Object Candidates fusion framework to achieve improved accuracy for 3D object detection with near real-time performance (Fig. 2). Unlike the original CLOCs, Fast-CLOCs eliminates a separate heavy 2D detector; and instead, uses a 3D detector-cued 2D image detector (3D-Q-2D) to reduce memory and computation. This proposed architecture delivers the following contributions:

- Fast-CLOCs uses any 3D detector without re-training.
- The proposed 3D-Q-2D image detector within Fast-CLOCs outperforms state-of-the-art (SOTA) image-based detectors in 2D object detection.
- Fast-CLOCs is significantly more memory and computationally efficient than SOTA fusion methods, and can run in near real-time on a single desktop-level GPU.
- Fast-CLOCs improves the SOTA LiDAR-camera fusion performance on the KITTI and nuScenes datasets.

The rest of the paper is organized as follows. Related works are reviewed in Section 2. Section 3 discusses challenges and opportunities for LiDAR-camera fusion, and why we choose a separate fusion architecture. Section 4 details our Fast-CLOCs fusion network. Experimental results on KITTI and nuScenes are provided in Section 5 followed by the conclusion in Section 6.

2. Related Work

Related LiDAR-camera fusion methods are organized in three categories: cascaded fusion, integrated fusion and separate fusion. Each is reviewed in turn.

Cascaded Fusion: This type of fusion exploits mature image-based 2D detectors to narrow down the 3D processing domain to the corresponding cropped region in the image. The image processing and LiDAR point cloud processing are done in a cascaded style. The 3D detection are only based on cropped LiDAR point clouds. The problem for this type of fusion is that the 2D image-based proposal generation might fail in some cases that could only be observed from 3D space. This is problematic, since based on our statistics in Fig. 7, the recall of image-based detectors is usually lower than LiDAR-based detectors. Frustum PointNet [27], Pointfusion [36] and Frustum ConvNet [34] are the representatives of this category.

Integrated Fusion: In an integrated fusion architecture, the 3D point cloud and 2D image are either combined at the input or at the intermediate stage. MV3D [9] and AVOD [18] project the raw point cloud into bird's eye view (BEV) to form a multi-channel BEV image. A deep fusion based 2D CNN is used to extract features from this BEV image as well as the front camera image for 3D bounding box regression. But the crop and resize operations used in these methods result in inaccurate correspondence between the feature vectors. To improve feature correspondence, MMF [20] adopts continuous convolution [21] to build dense LiDAR BEV feature maps and does point-wise feature fusion with dense image feature maps. 3D-CVF [40] employs auto-calibrated projection to transform the 2D camera features to a smooth spatial feature map with the highest correspondence to the LiDAR features in the BEV domain, then a gated feature fusion network is applied to fuse the features. PointPainting [33] works by projecting lidar points into the semantic segmented image and appending the class scores to each point, this "painted" point cloud can then be fed to any LiDAR-based detectors. EPNet [16] presents a fusion module to augment the point features with semantic image features in a point-wise manner. The overall performance of these fusion based methods is worse than LiDAR-only based methods. In KITTI 3D detection leaderboard, 3D-CVF has the best performance among all these integrated fusion method, but it is still worse than other LiDAR-only detectors such as PV-RCNN [31].

Separate Fusion: Separate fusion processes each modality independently and fuses the detections at the decision level. Not many works are published in this category. CLOCs [26] exploits the geometric and semantic consistencies between 2D and 3D detections and automatically learns probabilistic dependencies between detections. The idea is not complicated and yet CLOCs has the highest ranking among fusion-based methods on the KITTI 2D, 3D and BEV object detection leaderboards.

3. Camera-LiDAR Fusion

In this section, we discuss existing issues for Camera-LiDAR fusion, including what information LiDAR and camera can each contribute towards 3D object detection, and why we choose separate fusion.

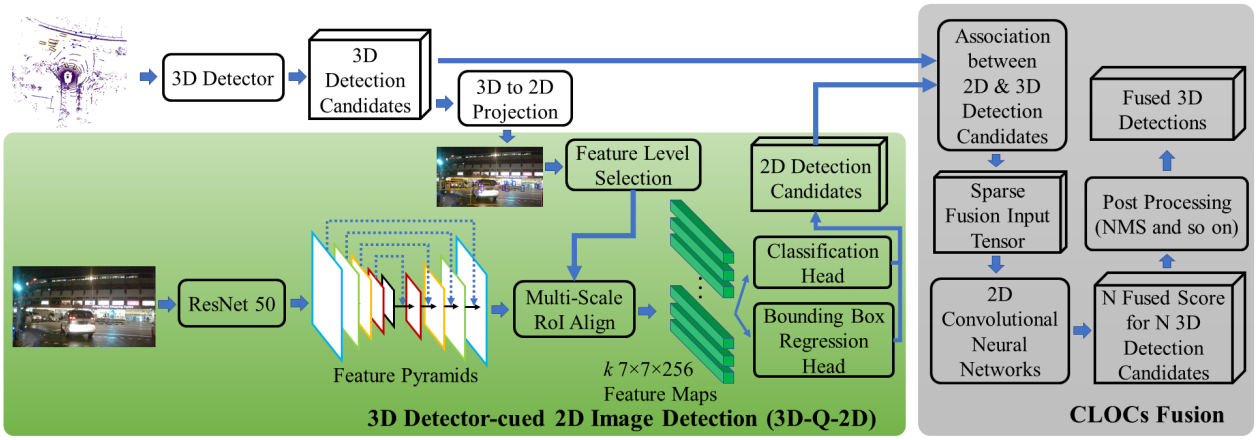


Figure 2: Fast-CLOCs system architecture. There are three primary components of the system: (1) 3D object detector; (2) 3D detector-cued 2D image detector (3D-Q-2D); (3) the original CLOCs fusion [26]. Compared to original CLOCs, we propose to leverage a lightweight 3D-Q-2D detector (green block), instead of a separate complete 2D image detector, to significantly reduce the GPU memory and computational cost.

3.1. Existing Issues for Camera-LiDAR Fusion

Data association between different sensor modalities is arguably the most challenging issue for multimodal sensor fusion. While 3D LiDAR points can be projected into a corresponding image to obtain a 2D–3D association, the LiDAR points are typically sparse resulting in incomplete pixel association, and moreover, association typically contains errors. There are multiple reasons for erroneous associations including differing visibility, occlusions due to displaced sensor viewpoints, as well as scan-time differences from a moving platform, as illustrated in Fig. 3.

These mismatches are particularly problematic for integrated fusion. These methods rely on fused data structures including image-augmented LiDAR point cloud [33, 16], LiDAR-augmented depth image [6] and BEV grid-map that stores visual information and LiDAR points from a pillar space into a grid [9, 18]. The mismatch error in the original pixel association can harm the downstream learning due to difficulty in distinguishing backgrounds and foregrounds.

The mismatch exists in feature domains as well. Aligning features using interpolation and cropping can result in inaccurate correspondence between these feature vectors and therefore likely degrading the final fusion performance.

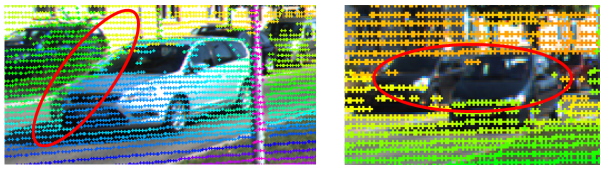
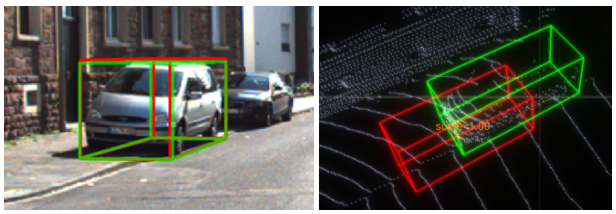


Figure 3: Examples of mismatch in LiDAR and camera. Left ellipse shows many points from the background projected on the vehicle. The right ellipse shows missing LiDAR points and so no association for camera pixels.



(a) (b)

Figure 4: Example of M3D-RPN [1] SOTA monocular image-only 3D detections. This example highlights the error in 3D properties learned from monocular images. The red bounding boxes represent the predictions from M3D-RPN, the green bounding boxes stand for the ground-truth. As shown in (a), the prediction looks perfect in image, but in 3D space (b), there is significant location error.

3.2. LiDAR and Camera Contributions to Fusion

LiDAR and cameras bring different types of information to the detection task. LiDAR provides the best source of 3D properties and shape information including 3D location and size. However, due to the sparsity and lack of texture information, semantic classification is a relatively weak spot for LiDAR compared to camera, especially for smaller objects.

For camera images, besides object classification, the 3D properties of the object can also be learned from appearance and relative 2D size [25, 1, 10, 29]. These 3D properties can be used for fusion-based 3D object detection. However, we observed that the 3D properties obtained from a monocular images are often orders of magnitude worse than from a LiDAR point cloud, as illustrated in Fig. 4. Therefore, we believe it is better to rely on images for classification than for fine-tuning the 3D properties.

3.3. Why Separate Fusion

Separate fusion can avoid some of the issues discussed above. There are fewer ambiguities and nearly no mismatch in the final detection candidates compared to raw input data and intermediate features. Detection candidates are also compact and simple to encode, enabling practical deployment for real-time applications. As we show next, Fast-CLOCs is not strictly separable, as 2D detections are cued by 3D, but it maintains the advantages of separate fusion.

4. Fast-CLOCs

A high-level diagram of the proposed Fast-CLOCs architecture is shown in Fig. 2. This illustrates the three primary components of the system: (1) 3D object detector, (2) 3D-Q-2D image detector, and (3) CLOCs fusion. The details of the system will be described and illustrated in this section.

The 3D detector processes the 3D point cloud to generate 3D *candidates*. Here detection candidates are used *before* Non-Maximum Suppression (NMS) because many correct detections are mistakenly suppressed during NMS [26]. A lightweight 3D-Q-2D image detector is proposed to generate high-accuracy 2D detections for fusion. This proposed detector uses the projected 3D candidates as its region proposals and refines them to predict its own 2D detections. Then, CLOCs [26] fusion is applied to fuse the 3D and 2D candidates and produce more accurate 3D detection results.

4.1. 3D-Q-2D Image Detector

4.1.1 The Input Data

We propose incorporating a 3D-Q-2D image detector instead of a separate complete 2D image detector used by the original CLOCs to significantly reduce the GPU memory and computational cost. As shown in Fig. 5, for traditional RCNN-style multi-stage 2D image detectors [30, 12, 4], the first stage is designed to generate foreground region proposals with high recall rate, then the following stages are applied for further classification and bounding box refinement. The quality of the proposals is of a paramount importance for the final detection output. The 3D detection candidates from the 3D detectors are of high quality and high recall rate; therefore, the projected 3D detection candidates can be leveraged as the 2D proposals for the 2D image detector with no extra costs. We project all the 3D detection candidates into the image plane using the calibration parameters between LiDAR and camera, and these constitute *cues* for our 3D-Q-2D image detector.

Note that there are 8 corner points for each 3D bounding box. After projecting them into the image plane, we use the maximum and minimum xy pixel coordinates to build an axis-aligned 2D bounding box. In this paper, "projected 3D detection candidates" refers to the corresponding axis-aligned 2D bounding box in the image plane.

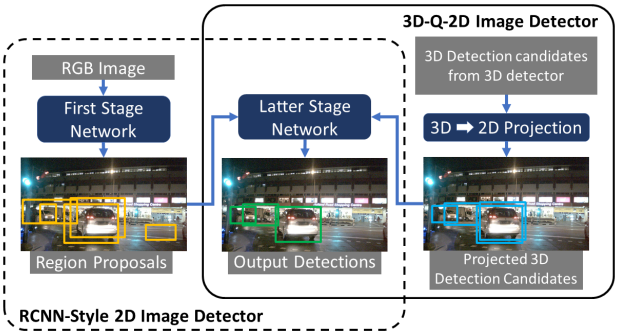


Figure 5: Architecture comparison of our 3D-Q-2D image detector and RCNN-style 2D image detector. The 3D detection candidates from the 3D detectors can be leveraged as the 2D proposals for the 2D detector at no extra cost. Therefore, there is no need to have a computationally expensive first-stage network in our proposed 2D detector.

4.1.2 The Backbone Feature Extraction Network

We use ResNet-50 with Feature Pyramid Network (FPN) [14, 22] pretrained in COCO [24] as the backbone. The FPN produces a multi-scale feature pyramid in which all levels are semantically strong, improving performance compared to producing the single-scale output feature map from the backbone. The feature level selection module assigns the level of a proposal based on its size in the original image. Small size proposals will be assigned to low-level high-resolution feature maps. The corresponding region of interest (RoI) in the feature map are calculated based on the downsample rate of that feature map (*e.g.,* 4, 8, 16). Multi-scale RoI align [13] is applied to extract features within the RoI in the feature maps. Then, the extracted feature maps are converted into fixed spatial extent of $H \times W$ (*e.g.,* 7×7). Bilinear interpolation is applied to avoid quantization of the RoI boundaries or bins.

4.1.3 Detection Head

A bounding box regression head and a classification head are attached to the k output RoI feature maps, where k is the number of projected 3D detection candidates. The 2D bounding box is parameterized using top left and bottom right pixel coordinates. The bounding box prediction and training parameterizations are defined as follows:

$$w^{pro} = x_2^{pro} - x_1^{pro}, \quad h^{pro} = y_2^{pro} - y_1^{pro} \tag{1}$$

$$t_{x1} = (x_1 - x_1^{pro})/w^{pro}, \quad t_{x2} = (x_2 - x_2^{pro})/w^{pro} \tag{2}$$

$$t_{y1} = (y_1 - y_1^{pro})/h^{pro}, \quad t_{y2} = (y_2 - y_2^{pro})/h^{pro} \tag{3}$$

$$t^*_{x1} = (x^*_1 - x_1^{pro})/w^{pro}, \quad t^*_{x2} = (x^*_2 - x_2^{pro})/w^{pro} \tag{4}$$

$$t^*_{y1} = (y^*_1 - y_1^{pro})/h^{pro}, \quad t^*_{y2} = (y^*_2 - y_2^{pro})/h^{pro} \tag{5}$$

where variables (x_1, y_1) and (x_2, y_2) are the top left and bottom right pixel coordinates of the 2D bounding box, with height, h, and wdith, w. Variables x, x^{pro} and x^* are for the predicted 2D box, projected 2D box (from 3D detector) and ground-truth 2D box, respectively.

The classification head maps the RoI features into softmax probabilities using $(n + 1)$ classes, with n for objects and 1 for background. Here note that although we have softmax probabilities for each class, the output class of the 2D image detector is still determined by the 3D detector. This is because for fusion of a 3D detection with class #A, we ask from the image detecor "how likely is it that there is an object with class #A in the projected region?", rather than "what is the class of the object within the projected region?". The latter is more challenging, requires a more complicated network, and not needed for our fusion task.

4.1.4 Multi-Task Loss and Training

We have two loss functions for the proposed 2D detector, namely smooth L1 loss for bounding box regression, and focal loss [23] for classification and to address class imbalances between targets and backgrounds. During training, the intersection of union (IoU) between projected 3D detection candidates (2D proposals) and the 2D ground-truths are calculated. Proposals with IoU larger than θ_{up} are labeled positive and those with IoU smaller than θ_{low} are labeled negative examples. Different classes have different IoU thresholds. Only positive examples are included in the bounding box regression loss, while both positive and negative examples contribute to the classification loss. The final loss is the weighted sum of the two losses, following [17] to use uncertainty to weigh the two losses during training.

The proposed network is trained using stochastic gradient descent (SGD). We use the Adam optimizer with max learning rate learning rate of $1 * 10^{-4}$, weight decay 0.01, and momentum 0.85 to 0.95 for 20 epochs.

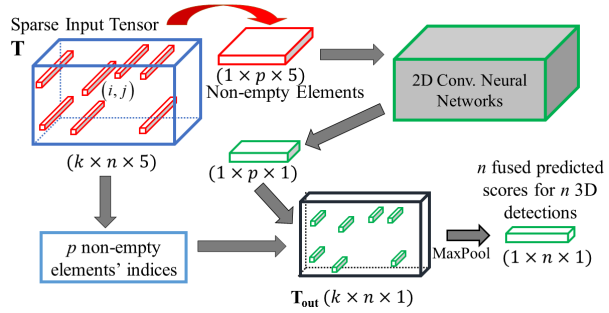


Figure 6: CLOCs [26] fusion architecture.

4.2. CLOCs Fusion Network

CLOCs fusion is taken from [26] and illustrated in Fig. 6. For completeness we summarize it here including several small improvements.

4.2.1 Geometric and Semantic Consistencies

CLOCs leverages an image-based IoU between 2D detections and projected 3D detections to quantify geometric consistencies between 2D and 3D detections. Semantic consistency ensures that only detection candidates with the same category are associated during fusion.

4.2.2 2D–3D Association via a Sparse Tensor

The goal of the encoding step is to convert all 2D and 3D detection candidates into a consistent joint representation which can be fed into the CLOCs fusion network. For k 2D detections and n 3D detections, a $k \times n \times 5$ tensor $\mathbf{T}$ is built, as shown in Fig. 6. For each element $\mathbf{T}_{i,j}$, there are 5 channels denoted as follows:

$$\mathbf{T}_{i,j} = \{IoU_{i,j}, s_i^{2D}, s_j^{3D}, d_j, flag\} \tag{6}$$

where $IoU_{i,j}$ is the IoU between i_{th} 2D detection and j_{th} projected 3D detection, s_i^{2D} and s_j^{3D} are the confident scores for i_{th} 2D detection and j_{th} 3D detection respectively. d_j represents the normalized distance between the j_{th} 3D bounding box and the LiDAR. Unlike standard CLOCs, we add another $flag$ channel to highlight whether a 3D detection overlaps with at least one 2D detection. There are two reasons for this. First, we want to keep the 3D detection candidate that has no 2D detections overlap with it. Because, as shown in Section 4.3, the recall rate for the 3D LiDAR-based detector is higher than the 2D image-based detector. Second, adding this new channel helps the network distinguish this case from other examples with very small IoU and s^{2D}.

For each projected 3D detection, only few 2D detections intersect with it, and elements $\mathbf{T}_{i,j}$ with zero IoU are set as empty as they are geometrically inconsistent. Therefore the $k \times n \times 5$ input tensor is a sparse tensor. The fusion network only needs to learn from these overlapping examples which are non-empty elements in the sparse input tensor.

4.2.3 Generating Fused Confidence Scores

The non-empty elements are fed to the fusion layers to generate fused confidence scores for all potential associations. Unlike the original CLOCs, we use Residual blocks [14] instead of standard 1×1 convolutional layers to improve the robustness and performance. As shown in Fig. 6, assuming there are p non-empty elements from the sparse input tensor, the output has a shape of $1 \times p \times 1$ which represents the fused confident scores for p potential 2D and 3D associations. From this the output tensor $\mathbf{T}_{out}$ of shape $k \times n \times 1$ is built by filling p outputs based on their indices (i, j) in the input tensor and set negative infinity elsewhere. Finally

the output tensor is mapped to the fused probability map for n 3D detections through maxpooling in the first dimension.

4.3. The Scalability of Fast-CLOCs

Fast-CLOCs is a detection level fusion approach with a 3D-Q-2D block that leverages 3D detections to cue the 2D detector. Thus it is crucial that the 3D detector have a high recall rate, as detections it misses will also be missed by the 2D detector. Based on our statistics, we believe this is not a problem because SOTA 3D-based detectors have far fewer misses than SOTA 2D detectors. Fig. 7 shows statistics on the KITTI dataset where we compare the number of true positives and misses by SOTA 2D and 3D detectors. The 3D detector misses only 219 objects that are detected by the 2D image detector whereas the 2D detector misses 2491 objects detected by 3D LiDAR detector. This high effective recall rate of the 3D detector validates using the 3D-Q-2D detector to replace an independent 2D image detector in fusion.

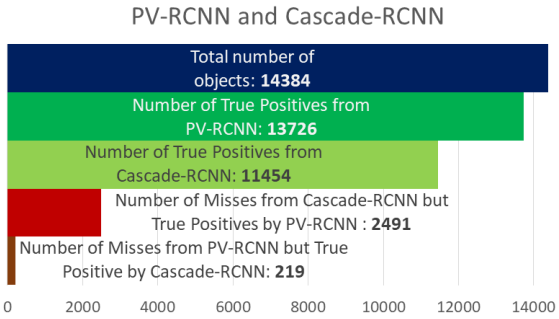


Figure 7: KITTI validation car detection comparison of true positives and misses from a 2D detector (Cascade-RCNN [4]) and 3D detector (PV-RCNN [31]). Only 219 objects are missed by 3D detector but detected by 2D detector, while 2491 are missed by 2D detector but detected by 3D detector. This shows that 3D detector has higher recall rate than 2D detector, and validates using 3D-Q-2D replacing an independent 2D image detector in fusion.

Detector	Input Data	2D AP (%)		
		easy	moderate	hard
MS-CNN [3]	Img	90.83	89.88	79.16
Cascade-RCNN [4]	Img	91.35	90.59	80.64
SECOND [37]	LiDAR	97.87	92.37	89.87
SECOND+3D-Q-2D	LiDAR+Img	98.75	95.56	90.14
PV-RCNN [31]	LiDAR	98.26	94.42	89.24
PV-RCNN+3D-Q-2D	LiDAR+Img	98.52	95.08	89.48

Table 1: Comparison of 2D detection performance with SOTA 2D image-based detectors and corresponding 3D detectors on car class in KITTI validation set.

5. Experimental Results

We evaluate our Fast-CLOCs on the challenging KITTI [11] and nuScenes [2] object detection benchmarks. We implement Fast-CLOCs using three 3D detectors: SECOND [37], PV-RCNN [31] and CenterPoint [39], termed Fast-CLOCs-SEC, Fast-CLOCs-PV and Fast-CLOCs-CP respectively.

5.1. Datasets

KITTI: There are 7481 training samples and 7518 testing samples in KITTI object detection dataset. The true positive metric is based on 2D/3D IoU. Methods are evaluated using average precision. All the instances are classified into three difficulty levels: easy, moderate and hard, based on their 2D bounding boxes' height, occlusion level and truncation level. For experimental studies, we follow the convention in [8] to split the original training samples into 3712 training samples and 3769 validation samples.

nuScenes: nuScenes 3D detection dataset contains 1000 driving segments of 20 seconds each, with 750, 150 and 150 segments for training, validation and testing respectively. The annotation rate is 2Hz, so there are 28k, 6k and 6k annotated frames for training, validation and testing respectively. There are 10 classes of objects. The main evaluation metrics are mean average precision over all classes and nuScenes detection score (NDS) [2]. The true positive metric is based on BEV center distance instead of IoU.

5.2. Evaluation Results

3D-Q-2D Image Detector: Before evaluating our Fast-CLOCs in 3D object detection, we first evaluate the proposed 3D-Q-2D image detector. For Fast-CLOCs, the quality of the input 2D detection candidates is of a paramount importance. We use SECOND [37] and PV-RCNN [31] as cueing 3D detectors for KITTI dataset, termed as SECOND+3D-Q-2D and PV-RCNN+3D-Q-2D respectively. CenterPoint is implemented for nuScenes dataset and termed as CenterPoint+3D-Q-2D.

Table 2 shows the 2D detection performance of our CenterPoint+3D-Q-2D in nuScenes validation set. Since nuScenes does not provide 2D evaluation metrics, we apply the KITTI 2D evaluation metrics here. 3D detections from CenterPoint [39] are projected to the image plane and are evaluated in 2D detection metrics for comparison. All 2D detections are evaluated by Average Precision (AP) with different IoU thresholds for different classes. As shown in Table 2, compared to CenterPoint, our CenterPoint+3D-Q-2D has much better performance for all classes except for pedestrian. The main reason for this we think is for nuScenes dataset, there are some driving sequences from rainy weathers and night time. Detecting pedestrian in these

Method	Barrier IoU=0.5	Traffic Cone IoU=0.5	Bicycle IoU=0.5	Motorcycle IoU=0.5	Pedestrian IoU=0.5	Car IoU=0.7	Bus IoU=0.7	Construction Vehicle IoU=0.7	Trailer IoU=0.7	Truck IoU=0.7
CenterPoint [39]	71.06	56.05	35.78	55.89	**74.76**	63.64	61.58	4.49	26.67	55.43
CenterPoint+3D-Q-2D	**72.60**	**59.63**	**56.22**	**65.25**	72.99	**76.25**	**73.08**	**15.94**	**34.26**	**56.84**

Table 2: 2D detection performance on nuScenes validation set. Since nuScenes does not provide 2D evaluation metrics, we apply the KITTI 2D evaluation metrics here. All 2D detections are evaluated by Average Precision (AP) with different 2D IoU thresholds for different classes. CenterPoint is a 3D detector that here we evaluate on its 2D performance. By using CeterPoint to cue a 2D detector, we can obtain much improved 2D performance which is better for fusion.

Detector	S*	3D AP (%) *Car / Pedestrian / Cyclist*			Bird's Eye View AP (%) *Car / Pedestrian / Cyclist*		
		easy	moderate	hard	easy	moderate	hard
PV-RCNN (baseline) [31]	L	87.45 / 47.30 / **77.33**	80.28 / 39.42 / **62.02**	76.21 / 36.97 / **55.52**	91.91 / 52.42 / **79.62**	88.10 / 44.85 / **64.53**	85.41 / 42.56 / **57.91**
Fast-CLOCs-PV (Ours)	F	89.11 / **52.10** / **82.83**	**80.34** / **42.72** / **65.31**	**76.98** / **39.08** / **57.43**	93.02 / **57.19** / **83.34**	89.49 / **48.27** / **67.55**	**86.39** / **44.55** / **59.61**
Improvement (Fast-CLOCs-PV over PV-RCNN)	-	*+1.66 / +4.8 / +5.5*	*+0.06 / +3.3 / +3.29*	*+0.77 / +2.11 / +1.91*	*+1.11 / +4.77 / +3.72*	*+1.36 / +3.42 / +3.02*	*+0.98 / +1.99 / +1.7*
CLOCs_PVCas [26]	F	88.94 / —- / —-	**80.67** / —- / —-	**77.15** / —- / —-	93.05 / —- / —-	**89.80** / —- / —-	**86.57** / —- / —-
CLOCs_SecCas [26]	F	86.38 / —- / —-	78.45 / —- / —-	72.45 / —- / —-	91.16 / —- / —-	88.23 / —- / —-	82.63 / —- / —-
F-PointNet [27]	F	82.19 / **50.53** / 72.27	69.79 / **42.15** / 56.12	60.59 / **38.08** / 49.01	91.17 / **57.13** / 77.26	84.67 / **49.57** / 61.37	74.77 / **45.48** / 53.78
AVOD [18]	F	76.39 / 36.10 / 57.19	66.47 / 27.86 / 42.08	60.23 / 25.76 / 38.29	89.75 / 42.58 / 64.11	84.95 / 33.57 / 48.15	78.32 / 30.14 / 42.37
EPNet [16]	F	**89.81** / —- / —-	79.28 / —- / —-	74.59 / —- / —-	**94.22** / —- / —-	88.47 / —- / —-	83.69 / —- / —-
UberATG-MMF [20]	F	88.40 / —- / —-	77.43 / —- / —-	70.22 / —- / —-	93.67 / —- / —-	88.21 / —- / —-	81.99 / —- / —-
UberATG-ContFuse [21]	F	83.68 / —- / —-	68.78 / —- / —-	61.67 / —- / —-	**94.07** / —- / —-	85.35 / —- / —-	75.88 / —- / —-
PI-RCNN [35]	F	84.37 / —- / —-	74.82 / —- / —-	70.03 / —- / —-	91.44 / —- / —-	85.81 / —- / —-	81.00 / —- / —-
3D-CVF [40]	F	**89.20** / —- / —-	80.05 / —- / —-	73.11 / —- / —-	93.52 / —- / —-	**89.56** / —- / —-	82.45 / —- / —-

*In S column, L represents LiDAR-only. F stands for camera-LiDAR fusion-based approach.

Table 3: Performance comparison of object detection with SOTA camera-LiDAR fusion methods on all classes of KITTI test set. Evaluation is by Average Precision (AP) with best in green and second-best in blue. Fast-CLOCs fusion outperforms other SOTA fusion-based detectors in most measures, and has same level of performance compared to CLOCs_PVCas [26]. While CLOCs_PVCas requires significantly more computation (shown in Table. 5).

scenarios from camera images are more challenging than using LiDAR. Table 1 compares 2D detection performance with SOTA 2D image detectors and corresponding 3D detectors on car class in KITTI validation set. This shows our 3D-Q-2D image detector outperforming SOTA 2D image detectors by a large margin.

5.3. Main Results

We present our 3D detection results on the test set of KITTI in Table 3 and the test set of nuScenes in Table 4. In the KITTI test set, our Fast-CLOCs-PV outperforms most of the SOTA fusion-based methods in all three classes. Note that the official open-source code of PV-RCNN performs slightly worse than the private one owned by the PV-RCNN authors shown on the KITTI leaderboard, and our Fast-CLOCs-PV result is based on the open-source PV-RCNN. The baseline PV-RCNN in Table 3 refers to the open-source PV-RCNN. As shown in Table 3, compared to baseline PV-RCNN, our Fast-CLOCs-PV increases the performance in 3D and BEV object detection by a large margin. Only CLOCs_PVCas has slightly better performance than ours (less than 0.4% AP) in car class. But CLOCs_PVCas requires running PV-RCNN and Cascade-RCNN simultaneously which cannot be deployed on a single desktop-level

Method	Setting*	mAP ↑	NDS ↑
CenterPoint [39]	L	58.0	65.5
Fast-CLOCs-CP (Ours)	F	**63.1**	**68.7**
Improvement (Fast-CLOCs-CP over CenterPoint)	-	*+5.1*	*+3.2*
PointPainting [33]	F	46.4	58.1
3D-CVF [40]	F	52.7	62.3
WYSIWYG [15]	L	35.0	41.9
PointPillars [19]	L	40.1	55.0
CVCNet [7]	L	55.3	64.4
PMPNet [38]	L	45.4	53.1
SSN [44]	L	46.3	56.9
CBGS [43]	L	52.8	63.3

*In Setting column, L represents LiDAR-only. F stands for camera-LiDAR fusion-based approach.

Table 4: Performance comparison of 3D object detection with SOTA methods on nuScenes test set. We show the primary evaluation metrics nuScenes Detection Score (NDS) [2] and mean Average Precision (mAP).

Method	3D Detector			2D Detector			Fusion GPU Mem	Total	
	Name	GPU Mem	Speed	Name	GPU Mem	Speed		GPU Mem	Speed
CLOCs_PVCas [26]	PV-RCNN	3-4GB	9.5Hz	Cascade-RCNN	5+GB	4Hz	∼1GB	10+GB	<4Hz
CLOCs_SecCas [26]	SECOND	2.5-3GB	28Hz				∼1GB	10+GB	<4Hz
Fast-CLOCs-PV	PV-RCNN	3-4GB	9.5Hz	3D-Q-2D	1-1.5GB	18Hz	∼1GB	6.5GB	∼8Hz
Fast-CLOCs-CP	CenterPoint	3.5-4GB	10Hz				∼1GB	6.5GB	∼8Hz
Fast-CLOCs-SEC	SECOND	2.5-3GB	28Hz				∼1GB	5.5GB	∼13Hz

Table 5: Comparison of running speed and GPU memory usage between CLOCs [26] and Fast-CLOCs on RTX 3080 GPU. The original CLOCs [26] requires running Cascade-RCNN [4] in addition to a 3D detector simultaneously, so it cannot run all together on a single desktop-level GPU system, while Fast-CLOCs can and has better or same level of detection performance. We cannot test running CLOCs_PVCas and CLOCs_SecCas on our RTX 3080 due to out of GPU memory issue, but since Cascade-RCNN can only run with around 4Hz, their final inference speeds would be definitely slower than 4Hz. While Fast-CLOCs is much faster than that.

GPU platform, while Fast-CLOCs-PV can. 3D-CVF's performance is close to ours in easy and moderate level for car class, but our Fast-CLOCs-PV outperforms 3D-CVF in hard level by a large margin (nearly 4% AP better). In nuScenes test set, our Fast-CLOCs-CP outperforms other published SOTA methods in both mean Average Precision (mAP) and nuScenes Detection Score (NDS). Our Fast-CLOCs-CP outperforms baseline method CenterPoint by 5.1% and 3.2% in mAP and NDS.

Speed and Memory: For self-driving vehicles, the computing resources are limited. It is challenging to install server-level multi-GPU system on the vehicle. Compared to other heavy fusion-based approaches, Fast-CLOCs can run in near real-time (8-13Hz) on single standard desktop-level GPU. All our experiments are done using NVIDIA RTX 3080 with 10 GB GPU memory. The running speed and GPU memory comparison results are shown in Table. 5. The original CLOCs [26] and some other fusion-based approaches [33, 41] require running Cascade-RCNN [4] or Mask-RCNN [13] in addition to a 3D detector simultaneously. These methods above are slow and cannot be deployed on a single desktop-level GPU system, while Fast-CLOCs can and has better or same level of detection performance with faster speed.

Qualitative Results: Due space constraints qualitative results and analyses are in the supplementary materials.

Ablation Study: We evaluate the contributions of each component within the proposed 3D-Q-2D image detector, the results are shown in Table 6. RoI Align [13] applies bilinear interpolation to avoid quantization of the RoI boundaries and further improve the detection performance. FPN provides a multi-scale feature pyramids which are helpful for detecting small objects (more than 2% AP improvement in hard level). The box regression module refines the projected 3D detection candidates and generates better 2D bounding boxes. Ablation study on the modifications of the CLOCs fusion network is provided in the supplementary materials.

focal loss	RoI Align	FPN	box regression	2D AP (%)		
				easy	moderate	hard
				95.97	90.04	86.52
✓				97.03	90.67	87.20
✓	✓			97.23	91.17	87.26
✓	✓	✓		97.83	92.66	89.54
✓	✓	✓	✓	**98.75**	**95.56**	**90.14**

Table 6: Ablation studies of different components in the proposed 3D-Q-2D image detector on KITTI validation set. SECOND [37] is applied to cue the 2D detector. RoI Align [13] applies bilinear interpolation to avoid quantization of the RoI boundaries or bins. Box regression represents the 2D bounding box regression head, without it the proposed 2D detector would output projected 3D boxes directly with visual confidence score.

6. Conclusion

In this paper, we propose Fast Camera-LiDAR Object Candidates (Fast-CLOCs) fusion that can run high-accuracy 3D object detection in near real-time. Fast-CLOCs introduces a lightweight 3D-Q-2D image detector to extract visual features from the image domain to improve 3D detections significantly. Compared to other separate fusion-based approaches that run independent 2D and 3D detectors simultaneously, Fast-CLOCs requires much less GPU memory and operates in real time on a single desktop-level GPU. At the same time, Fast-CLOCs achieves top or second-to-top performance in most categories compared to other fusion methods on KITTI and nuScenes datasets.

References

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

[2] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[3] Zhaowei Cai, Quanfu Fan, Rogerio S Feris, and Nuno Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. In *european conference on computer vision*, pages 354–370. Springer, 2016.

[4] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: High quality object detection and instance segmentation. *arXiv preprint arXiv:1906.09756*, 2019.

[5] Roberto Calandra, André Seyfarth, Jan Peters, and Marc P. Deisenroth. Bayesian optimization for learning gaits under uncertainty. *Annals of Mathematics and Artificial Intelligence (AMAI)*, 76(1):5–23, 2016.

[6] Luca Caltagirone, Mauro Bellone, Lennart Svensson, and Mattias Wahde. Lidar–camera fusion for road detection using fully convolutional neural networks. *Robotics and Autonomous Systems*, 111:125–131, 2019.

[7] Qi Chen, Lin Sun, Ernest Cheung, and Alan L Yuille. Every view counts: Cross-view consistency in 3d object detection with hybrid-cylindrical-spherical voxelization. *Advances in Neural Information Processing Systems*, 2020.

[8] 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 *Advances in Neural Information Processing Systems*, pages 424–432, 2015.

[9] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017.

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

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

[12] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

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

[15] Peiyun Hu, Jason Ziglar, David Held, and Deva Ramanan. What you see is what you get: Exploiting visibility for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11001–11009, 2020.

[16] Tengteng Huang, Zhe Liu, Xiwu Chen, and Xiang Bai. Epnet: Enhancing point features with image semantics for 3d object detection. In *European Conference on Computer Vision*, pages 35–52. Springer, 2020.

[17] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491, 2018.

[18] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8. IEEE, 2018.

[19] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[20] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7345–7353, 2019.

[21] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 641–656, 2018.

[22] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017.

[23] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

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

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

[26] Su Pang, Daniel Morris, and Hayder Radha. Clocs: Camera-lidar object candidates fusion for 3d object detection. *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[27] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 918–927, 2018.

[28] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

[29] Cody Reading, Ali Harakeh, Julia Chae, and Steven L. Waslander. Categorical depth distributionnetwork for monocular 3d object detection. *CVPR*, 2021.

[30] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

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

[32] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–779, 2019.

[33] Sourabh Vora, Alex H. Lang, Bassam Helou, and Oscar Beijbom. Pointpainting: Sequential fusion for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

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

[35] Liang Xie, Chao Xiang, Zhengxu Yu, Guodong Xu, Zheng Yang, Deng Cai, and Xiaofei He. Pi-rcnn: An efficient multi-sensor 3d object detector with point-based attentive cont-conv fusion module. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12460–12467, 2020.

[36] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. Pointfusion: Deep sensor fusion for 3d bounding box estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 244–253, 2018.

[37] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018.

[38] Junbo Yin, Jianbing Shen, Chenye Guan, Dingfu Zhou, and Ruigang Yang. Lidar-based online 3d video object detection with graph-based message passing and spatiotemporal transformer attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11495–11504, 2020.

[39] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3d object detection and tracking. *CVPR*, 2021.

[40] Jin Hyeok Yoo, Yecheol Kim, and Ji Song Kim. 3d-cvf: Generating joint camera and lidar features using cross-view spatial feature fusion for 3d object detection.

[41] Wenwei Zhang, Zhe Wang, and Chen Change Loy. Multi-modality cut and paste for 3d object detection. *arXiv preprint arXiv:2012.12741*, 2020.

[42] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018.

[43] Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. Class-balanced grouping and sampling for point cloud 3d object detection. *arXiv preprint arXiv:1908.09492*, 2019.

[44] Xinge Zhu, Yuexin Ma, Tai Wang, Yan Xu, Jianping Shi, and Dahua Lin. Ssn: Shape signature networks for multi-class object detection from point clouds. *ECCV*, 2020.