Sniffer Faster R-CNN ++: An Efficient Camera-LiDAR Object Detector with Proposal Refinement on Fused Candidates

SUDIP DHAKAL, University of North Texas, USA
DOMINIC CARRILLO, University of North Texas, USA
DEYUAN QU, University of North Texas, USA
QING YANG, University of North Texas, USA
SONG FU, University of North Texas, USA

In this paper we present Sniffer Faster R-CNN++, an efficient Camera-LiDAR late fusion network for low complexity and accurate object detection in autonomous driving scenarios. The proposed detection network architecture operates on output candidates of any 3D detector and proposals from regional proposal network of any 2D detector to generate final prediction results. In comparison to the single modality object detection approaches, fusion based methods in many instances suffer from dissimilar data integration difficulties. On one hand, fusion based network models are complicated in nature and on the other hand they require large computational overhead and resources, processing pipelines for training and inference specially, the early fusion and deep fusion approaches. As such, we devise a late fusion network that in-cooperates pre-trained, single-modality detectors without change, performing association only at the detection level. In addition to this, lidar based method fail to detect distant object due to its sparse nature so we devise proposal refinement algorithm to jointly optimize detection candidates and assist detection for distant objects. Extensive experiments on both the 3D and 2D detection benchmark of challenging KITTI dataset illustrate that our proposed network architecture significantly improves the detection accuracy, accelerating the detection speed.

CCS Concepts: • **Senfor Fusion** → **Effeciency**; • **Late Fusion** → *Flexibility*; Resuability; • **Proposal Refinement** → Accuracy.

Additional Key Words and Phrases: object detection, late fusion, proposal refinement, candidates fusion, regional proposal network

1 INTRODUCTION

Driven significantly by the interest in self-driving vehicles, compelling research effort has been devoted to both 2D and 3D object detection. For 2D object detection, while camera provide high resolution shape and texture information they suffer from inability to detect occluded object in complex scenes as camera data is mainly captured in the lower position of the front view [41]. This brings severe challenges to object detection and semantic segmentation. LiDAR sensors on the other hand, which facilitates 3D object detection, provide accurate 360°field of view 3D measurements and depth information but is vulnerable to extreme weather condition and also suffers from its sparse nature which is more prominent for distant object as seen in Fig. 1. Both Camera

Authors' addresses: Sudip Dhakal, sudipdhakal@my.unt.edu, University of North Texas, 1155 Union Cir, Denton, Texas, USA, 76203; Dominic Carrillo, dominiccarrillo@my.unt.edu, University of North Texas, 1155 Union Cir, Denton, Texas, USA, 76203; Deyuan Qu, deyuanqul@my.unt.edu, University of North Texas, 1155 Union Cir, Denton, Texas, USA, 76203; Qing Yang, University of North Texas, 1155 Union Cir, Denton, Texas, USA, qing.yang@unt.edu; Song Fu, University of North Texas, 1155 Union Cir, Denton, Texas, USA, song.fu@unt.edu.

2833-0528/2023/10-ART $15.00
https://doi.org/10.1145/3631138

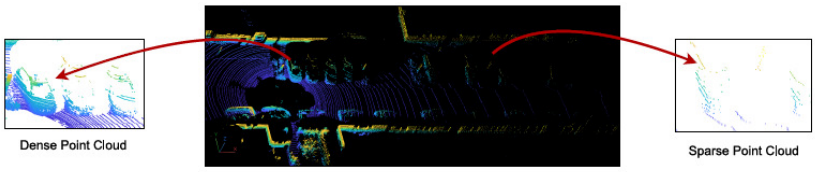


Fig. 1. Illustration of dense LiDAR point cloud for nearby objects and its sparse nature as the distance increases.

based methods [24] [2] [14] [23] and LiDAR based methods [42] [39] [22] [27] [12] [26] has been well studied in autonomous driving domain. Although the data of the two modalities excel in various areas when used separately, the complementary of LiDAR and Camera or simply fusion based methods is expected to make the combination result in a better performance on perception [7]. Instead, it has been strenuous to design fusion-based networks that exploits both modalities adequately and get improvements over single-modality based methods. Careful examination of challenging open 3D detection benchmarks, such as KITTI [8] and nuScenes [1], show that single modality based methods are the leading entries in the leader-board and hence there is still plenty of room for improvement for the fusion based methods. Additionally, fusion methods also do not achieve real time operation as they are prone to adding computational complexity. Existing approaches in the literature for fusing LiDAR and camera broadly follow three approaches: early fusion, deep fusion and late fusion as shown in Fig. 2. In early fusion, raw data from both lidar and camera sensor are fed into the early fusion deep learning based pipeline. This allows the powerful deep learning methods to find their own features carrying the highest information gain and therefore get a better performance in terms of precision of classification. However one major disadvantage of early fusion is that given heterogeneous data from different modalities it needs to use very complicated architectures of neural networks to find the common ground. As such, the computational power required to run such system will be enormous as the amount of data needed for training will also be huge. Similarly, in deep fusion where the features are combined after feature extraction, they also require support of deep learning based pipelines and networks. In contrast to this, late fusion would regard every sensor as a separate unit, where each sensor has its own processing pipeline and can incorporate pre-trained, single modality detector without any change. This allows for the pipelines to be fused into a perception output only when the detection and classification takes place which is at the decision level. The major advantages of late fusion methods can be summarized into following points:-

Flexibility and Simplicity As mentioned earlier, late fusion methods can incorporate any types of 2D and 3D models as they operate on the output results. Additionally, for a given network, it is also easy and simple to encode the detection candidate data that late fusion operates on.

Re-usability Pre-trained models for both 2D and 3D detector can be reused without going through the complications of training such networks. Single modality algorithms can be trained using their own sensor data.

Heterogeneous Data For each modality a separate algorithm can be trained based on requirement and later the output can be combined. One major advantage of such approach is that, especially in autonomous driving system that deals with multiple sensor, issue of data format, data alignment, representation and sparsity can be neglected or ignored. Data from each modality pass through separate pipeline thus providing a independent processing methodology such that there is no complexity in combining the diverse input data.

Based on this observation, we adopt a late fusion approach for low complexity and accurate object detection in autonomous driving scenarios. Our approach similar to the candidate fusion approach presented in CLOCs [17] operates on the output candidates of any 3D detector such as PV-RCNN, PointPillars before NMS (non-maximum suppression) and proposals from Regional Proposal Network (RPN) of any 2D detector such as Faster R-CNN,

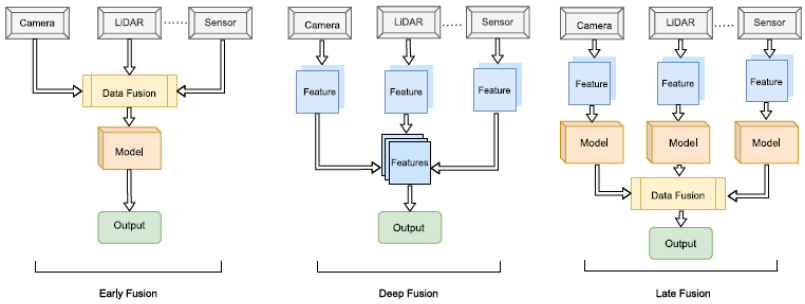


Fig. 2. Illustration of the architecture of various fusion based approaches in the existing literature.

Cascade RC-NN and so on to generate final prediction results. Similarly, since lidar point clouds are sparse for distant objects, their performance drops significantly on them. Unlike LiDAR scans, camera images provide high-resolution sampling and rich texture information even for the distant objects and thus can complement LiDAR in such scenario. In light of this evidence, our approach takes cues from camera data and combines them with LiDAR data to boost the detection performance for distant object and also in overall detection model. Apart from this, we also study the possibility of using different clustering algorithm for point cloud data and examine their application in different scenarios. Although our experiment show that there are possible alternatives to generating object candidates via other computationally small point cloud data clustering algorithm, our statistics also suggest that the accuracy suffers a major blow while adopting such methods. Therefore, finding the optimal trade off between accuracy and inference speed is a very complicated task in object detection domain. The proposed architecture delivers the following contributions:

• a versatile and flexible approach for optimizing object detection in autonomous driving systems.
• a novel proposal refinement algorithm to jointly optimize detection candidates form both LiDAR and Camera sensors and also assist detection for distant objects.
• evaluation on the KITTI dataset shows we significantly improve any 2D detectors state- of-the-art image-based methods and LiDAR-based method.
• a comprehensive study of clustering algorithm for point cloud data and their feasibility as an alternative to 3D detectors for creating 3D candidate objects.

2 RELATED WORKS

In the present context, there are stacks of methods that have been implemented for the object detection task in autonomous vehicles. Related object detection approaches in ADS can be summarized into three-fold discussed below.

2.1 3D object detection from point clouds

In order to learn discriminative features from sparse 3D point cloud data in different ways, a number of state-of-the-art 3D object detection methods has been proposed. Some approaches such as [11] [13] [40] generate 3D candidate boxes by projecting point cloud to bird's eye view (BEV) and utilizing 2D CNN to learn the point cloud features. Others,[42] [5] grouped the points into voxels and utilized 3D CNN to learn the features from each voxel and generate 3D boxes. There are two major drawback of such approaches. First, there is major information loss while projecting point cloud to bird eye view due to data quantization. Second, such approaches follow the early and deep fusion methodology using large number of 3D CNNs, thereby increasing both the computational

overhead and memory requirement for training and inference. Another set of 3D detection methods convert lidar points to pillars commonly called as pillar based method. Pointpillar[12] and PIXOR [40] further simplify 3D voxels to 2D pillars, where all the voxels with the same z-axis are combined into a single pillar. Existing 2D CNN can then be utilized to process these 2D pillars into bounding boxes. Although pillar based method solves the expensive processing nature of voxel based method, they also add another layer to already stacked convolutional neural network and hence requires large processing time and memory. In contrast, our late fusion approach benefit from candidate object obtained form pre-trained models of any 2D and 3D detector and the combined application along with the proposal refinement algorithm adds negligible delay to the perception system.

2.2 2D object detection from images

Based on the rich texture information provided by camera image, a number of state of art 2D object detection methods has been proposed. One-stage methods such as YOLO [23], SSD [15],and two-stage methods such as Faster R-CNN, Cascade R-CNN, Mask R-CNN [9] are the most popular object detection methods in two dimensional domain. One stage detectors are generally faster in terms of execution speed as they use a single feed-forward neural network that creates bounding boxes and classifies objects in the same stage but are less accurate. In contrast, two-stage methods, since they are based on established convolutions neural network as the state-of-the-art for learning image features and detection, are more accurate but take long execution time. Two-stage deep learning based object detectors consists of two stages 1) regional proposal network (RPN) and 2) regression and object classification. In the first stage, several region of interest (RIOs) are proposed and processed in an input image with each proposal having the certain probability of containing object of interest. Similarly, in the second stage, based on these RIOs, the most promising ones are selected and object present on those ROIs are classified based on the features learned. Our approach on the other hand leverages from these regions proposals obtained from the RPN, compare them with object candidates obtained from any 3D detector through the application of our novel proposal refinement algorithm and chooses only the best among the best candidates for generating final detection boxes.

2.3 Object Detection Based on Integrated Fusion

As discussed earlier, in an integrated fusion approach, data from lidar and camera are linked together either at the input stage where raw and pre-processed data obtained from sensors and fused together, or at the intermediate stage where features obtained from sensors are combined, or at the decision stage where output of different models are combined together. MVX-Net [29] proposed a multi-model voxelnet to augment LiDAR points with semantic image features and learn to fuse image and LiDAR features at early stages for accurate object detection. Similar to this approach, PointPainting [32] projects the lidar points into the semantically segmented image, creating a painted point cloud, which is fed to any LiDAR based detectors to generate output 3D bounding boxes. Similarly, EPNet [10] uses a point-wise manner to to augment the point features from lidar point cloud data with semantic image features in a fusion module. Other methods such as Frustum PointNets [21], Frustum convnet [34] first detect objects in 2D images then use the information to further process the point cloud data. VOTENET [20] and ImVoteNEt [19] also use the integrated fusion approach to complement lidar point cloud with both geometric and semantic/texture cues from images using the concept of voting schemes for promising 2D boxes which are then appended to seed point in 3D for object proposals. Besides geometric cues from the 2D votes, each pixel also passes semantic and texture cues to the 3D points, as either features extracted per-region, or ones extracted per-pixel. This helps to significantly improve 3D detection. Finally, methods like MV3D [4], AVOD [11], form a multi-channel bird eye view image by projecting raw point cloud into BEV. And using 2D CNN, features are extracted from this transformed image along with another image from front camera for 3D bounding box regression. Most of these approaches are either early fusion or middle fusion methods, that requires

transformation of either point cloud data or camera images because these are heterogeneous data and combining them either causes loss in information or create a complex architecture. In contrast to such approaches, our methods uses late fusion approach and hence doesn't have to deal with the heterogeneous data.

2.4 Limitation of Prior Works

Currently, there exists a multitude of fusion-based approaches in contemporary research. Among these methods that combine lidar and camera data, most of them follow either early fusion that combines raw data from lidar and camera sensor at the early stage or deep fusion approach that combines features extracted from these data obtained from different modalities at the feature level. Methods such as PointPainting [32], PointAugmenting [33], follow an early fusion approach, where they combine raw image and point cloud data from camera and lidar sensor respectively. One limitation with such an approach is that there is a problem of complexity and interpretability. For instance, [32] requires complex model architecture with a high number of input channels to get semantic labels from camera data and further combine it with lidar point cloud data to create painted point cloud. This complexity can make the model harder to train, optimize, and interpret. Also from architecture point of view, combining different modalities at an early stage may capture different levels of semantics or object characteristics. Early fusion may struggle to bridge the semantic gap between modalities, making it challenging to capture high-level semantic information. Similarly, the dimensional input data in such methods is also huge which can lead to an increase in computational demands. This can slow down training and inference times. These methods also require change in data input as the data is either augmented in case of [33] or transformed in case of [32]. Such alteration also requires additional computation and can result in information loss.

On the other hand, deep fusion based methods such as MVXNet [30], AVOD [11], and MV3D [4]which combine features extracted from different models, also have their own limitations. For instance, [30], uses features extracted from camera images to append them to the 3D points at feature level. This can create feature alignment problems, particularly when modalities have varying resolutions, scales, or data characteristics. Failure to align features properly can lead to suboptimal fusion results. Similarly, [11] also has a complex architecture as it requires multimodal feature fusion for features extracted from both image and BEV lidar point cloud data. This can lead to limited adaptability to new modalities as incorporating new sensor modalities or making changes to the sensor configuration might require significant adjustments to AVOD's architecture and fusion mechanisms, potentially hindering adaptability. Another deep fusion based method PI-RCNN [36] also follows a similar architecture. First, an image segmentation sub-network extracts semantic features from RGB-image. Meanwhile, the stage-1 of detection sub-network generate 3D proposals from raw LIDAR points. Then, the 3D points and semantic feature maps are fed into the separate module to conduct point-wise fusion and supplement the features of points. Finally, the stage-2 of detection sub-network takes the point-wise features augmented from image semantics as input to obtain the final prediction of the 3D bounding box. Such additional sub networks can again make the architecture complex and increase the computation overhead. Deep Continuous Fusion [13], EPNet [10] and 4D-Net [18] attempt to fuse the two modalities by sharing the information between 2D and 3D backbones. However, an important limitation in those works is a lack effective alignment mechanism between camera and lidar features leading to a suboptimal performance.

2.5 Distinctive Features of Our Approach

Diverging from existing approaches, our work stands apart both in terms of architecture and implementation. Notably, we embrace a late fusion strategy, driven by a clear recognition of the manifold benefits it offers. Early fusion based approaches like [33], [32], despite their advantages, present intricacies in model architecture, interpretation, computational demands, and potential information loss. Our late fusion method on the other hand, make use of pretrained 2D and 3D detector to generate the bounding boxes and process these boxes

together to get a better result. Our approach is different in a sense, it is simple, easy to implement and pretty straightforward. Our approach is a probabilistic driven learning based fusion technique that is designed to exploit the geometric and semantic information from 2D and 3D detectors. UnLike the existing early fusion, which requires data augmentation or transformation at an early stage, our late fusion approach only combines bounding boxes obtained from respective detectors at the final stage. This makes it easier for training and inference as there is no complexity introduced due to difference in data from heterogenous modalities. Also, we reuse the already provided pretrained model, thus saving time and computation overhead.

Similarly, different from deep fusion based approaches, that requires feature level data encoding and combination, thus suffering from feature misalignment, and potential information loss, our method follows a simple approach thus making it easy to use and incorporate any types of 2D and 3D detectors as our model only operate on the output result. Additionally, features extracted from different modalities have different formats and thus require careful computation. Also, combining high-dimensional features from different modalities, can increase the dimensionality of the input space. Our method on the other hand is only designed to exploit the geometric and semantic information from 2D and 3D detectors. Overall, our implementation makes the computation faster, assuming we already have the pretrained 2D and 3D detectors.

Furthermore, our approach distinguishes itself from its late fusion-based counterparts like CLOC in two fundamental ways. Firstly, CLOC employs intersection over union to evaluate and compare bounding boxes. In contrast, our method takes a different route, utilizing our proposal refinement algorithm to meticulously assess and refine proposals, as well as candidate objects stemming from 2D and 3D detectors. This strategic choice is underpinned by the goal of ensuring the accurate detection of distant objects — a common limitation in conventional methods. Moreover, another departure lies in the fact that while CLOC relies on the final detection boxes generated by 2D detectors, our approach leans on the proposals extracted from the region proposal networks of 2D detectors. This decision is motivated by the intention to maximize the incorporation of proposals originating from camera sensors. The primary rationale behind this choice is to enhance the efficacy of distant object detection by encompassing as many relevant proposals as possible.

3 MOTIVATION

LIDAR sensors play a crucial role by providing precise 3D point cloud data that accurately captures objects' shapes, sizes, and positions. When this information is combined with camera data, the vehicle's comprehension of its surroundings becomes more comprehensive. Moreover, the integration of candidate objects derived from accurate 3D detectors which fuses the most refined information from LIDAR sensors, results in an enhanced overall understanding of objects within the environment. This is particularly crucial for detecting and recognizing objects such as pedestrians, cyclists, and small vehicles. Furthermore, the integration of 3D detectors introduces an added layer of redundancy and robustness to the perception system. In situations where camera sensors face challenges due to environmental conditions or technical issues, the 3D detectors can step in to ensure the system's reliability, thereby upholding the safety of the vehicle and its occupants. The fusion of geometric features provided by 3D detectors with the visual cues of color and texture from camera data also contributes to improved object classification. An additional motivation behind the incorporation of 3D detectors lies in their capability to identify objects even when they are partially obscured by obstacles. This addresses a significant limitation in object detection, ensuring that the perception system maintains accurate awareness of objects, irrespective of occlusions. As a result, the integration of 3D detectors not only enhances accuracy but also bolsters the system's adaptability and performance in complex and dynamic environments.

3.1 Drawbacks of Integrating 3D detectors

Incorporating 3D detectors also brings about drawbacks. Among these, a significant concern is the potential for false positives to be introduced into the system. While the majority of candidate objects derived from 3D detectors are reliable, a subset of them may lead to the inclusion of incorrect bounding boxes or erroneous candidate objects within the model. This phenomenon is clearly illustrated in Fig. 3, where the red-colored candidate objects originating from PV-RCNN are instances of false detections. These erroneous candidate objects can lead to both misclassification and incorrect detection outcomes.

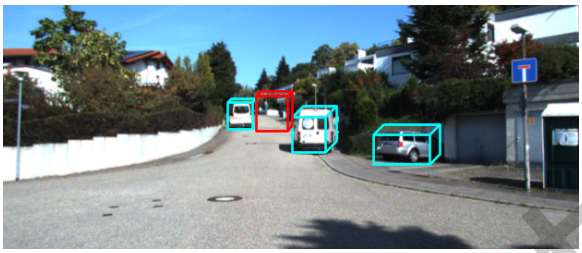


Fig. 3. Illustration of false positive (denoted by red bounding box) introduced into the model due to 3D detectors.

4 SNIFFER FASTER R-CNN++ ARCHITECTURE

In this section, we present our proposed two-stage late fusion framework, Sniffer Faster R-CNN++ for detection and classifying object from combined application of lidar point cloud and camera image. The proposed method, illustrated in Fig. 4, uses four primary components: (1) 3D Candidate Network, (2) 2D Regional Proposal Network, (3) a proposal refinement algorithm and finally (4) Cloc's Fusion. For generating 3D detection candidates we use PVR-CNN as our primary 3D detector, we can also use any 3D detectors. We also study the practicality of using existing data clustering algorithm such as DBSCAN, CCL and RANSAC for rapid 3D detection candidate generation which is discussed in subsection below. Similarly, we use 2D proposals from RPN of Faster R-CNN, which passes through Proposal Refinement algorithm along with the 3D detection candidate as a sparse tensor together. Best candidate proposals are then selected from this group of input tensor and CLOC's fusion approach with small modification is implemented to get the final detection result as shown in the figure.

4.1 2D-3D Association via Sparse Tensor

We associate 2D proposals and 3D detection candidates into a consistent joint representation which through the application of proposal refinement algorithm is fed into the CLOC's fusion network. The output from the RPN of any 2D detector (Faster R-CNN in our case) are a set of 2D bounding boxes in the image plane with four coordinates of top left and bottom right corner namely xmin, ymin, xmax, ymax along with a confidence score denoted as c_i^{2D} and can be represented as:

$$R^{2D} = \{ R_1^{2D} , R_2^{2D} , R_3^{2D} , \dots R_k^{2D} \},$$

$$R_i^{2D} = \{ [xmin , ymin , xmax , ymax , c_i^{2D}] \}$$

Most 2D detector have 1000 proposal per image as a default setting but we limit the number of proposal for each image to 500 by eliminating proposals that have the lowest confidence score. However for some proposals, even though they have low confidence, we still keep them to provide cues to the LiDAR for detecting distant object. This is discussed thoroughly in the section below. A 3D bounding box is represented as $(x, y, z, h, w, l, \theta)$

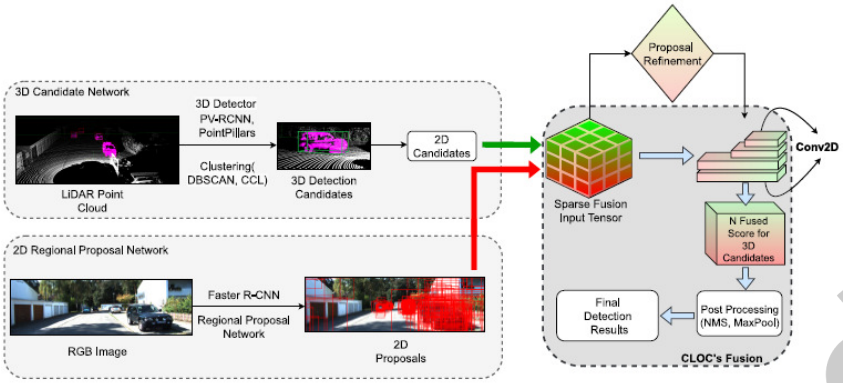


Fig. 4. The overall architecture of our proposed Sniffer Faster R-CNN++ netowork, which comprises of a) 3D candidate network for generating 3D candidate objecs, b) 2D Regional Proposal Network for generating regional proposals, c) Proposal Refinement for refining the candidate objects and proposals and d) CLOC's fusion netowrk for final predication results.

in the LiDAR coordinate, where (x, y, z) is the object center location, (h, w, l) is the object size, and angle θ is the object orientation from the bird's view with confidence score c_i^{3D} and can be represented as:

$$L^{3D} = \{ L_1^{3D}, L_2^{3D}, L_3^{3D}, \ldots L_k^{3D} \},$$
$$L_i^{3D} = \{ [x, y, z, h, w, l, \theta, c_i^{3D}] \}$$

Here L^{3D} is the set of all the detection candidates in one LiDAR scan. For 3D detectors, we take the object candidates prior to applying NMS to encompass as many candidate proposals as possible. These candidate object are then converted to 2D format that are aligned accurately along the camera coordinate. This is achieved via KITTI transformation and projection matrix provided by the KITTI dataset. The result from this transformation are set of 2D candidate object which can then be associated with the 2D proposals accordingly. The ultimate goal of this association is to create a consistent joint representation through the combined application of all 2D and 3D detection candidates so that the result can be fed to the proposal refinement algorithm and finally to the CLOC's fusion network.

4.2 Proposal Refinement

The proposal refinement algorithm [6] as shown in Algorithm 1 inputs two types of proposals, R2D and L3D from camera and lidar sensor respectively as discussed earlier. It prudently chooses only the best proposals from both sets of proposals. For a given image, we compare each candidate object or proposal from LiDAR proposal set (3D detector) (L) with each proposal from RPN of Faster R-CNN (R) i.e for each bbox (proposal) with coordinates (r_1, r_2, r_3, r_4) in R, we compare each bbox (proposal) with coordinates (l_1, l_2, l_3, l_4)in L for a given scene [12]. A certain threshold λ is assigned to this algorithm to compensate as many object candidates as possible and for that given threshold, if two adjacent bbox coordinates meets the criteria then we keep those candidates. In case, if the threshold is not met, then such candidate boxes are filtered out or simply removed from the tensor as shown in Fig. 5. A less then equal to value for this threshold accommodates the candidate boxes that might be too small in comparison whereas a greater then equal to value accommodates massive candidate boxes. The application of absolute subtraction of the candidate boxes coordinates can achieve both these cases.

While CLOC completely ignores the 2D detection candidate that doesn't have any overlapping 3D candidate object, we assume that some 2D proposals can provide useful cues to the final detection because LiDAR proposal

normally fail to detect object at far distance where the point cloud is sparse. In such scenario 2D RGB proposals can complement or help the missed detection from LiDAR candidate object and eventually obtain better detection result when used together. Based on our observation, most 2D proposals which represent object at large distance are very small in size which is obvious form practical point of view. As such, we keep 2D proposals that have a total area below a given threshold 'beta' β. Finally, the output of this algorithm with the application of λ and β as predefined threshold gives certain number of candidate boxes per given point cloud or image but due to the fact that same candidate object meets the threshold for multiple other candidate objects, it will be the resulting product multiple times. We manually check the repeated presence of candidates and remove the repeated ones. Finally, the sniffer candidates object are obtained as the output as illustrated in Algorithm 1 and these are handed over to the multiple conv2D for further processing.

Algorithm 1: Proposal Refinement Algorithm

Input: set of RGB proposal $r^i \in R$, $0 < i \leq 1000$.
set of LiDAR proposal $\ell^j \in L$, $0 < j \leq 100$.
Output: Final candidate objects S^*.

$r_1, ..., r_4$ are vertex coordinates in r, RGB proposal.
$\ell_1, ..., \ell_4$ are vertex coordinates in ℓ, LiDAR candidate object.
for *each i and j* **do**
  if $|r_1^i - \ell_1^j|$ *and* $|r_2^i - \ell_2^j|$ *and* $|r_3^i - \ell_3^j|$ *and* $|r_4^i - \ell_4^j| < \lambda$ **then**
    Keep r and ℓ as s_k;
  else
    Filter out r and ℓ ;
  end
end
for *each i* **do**
  if *area of* $r < \delta$ **then**
    Keep r as s_k;
  end
end
return final candidate objects $S^* = \{s_1, s_2, ..., s_n\}$, $n \leq k$.

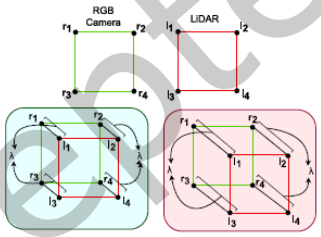


Fig. 5. Illustration of the working mechanism of Proposal Refinement Algorithm.

We have made a minor modification to our proposal refinement algorithm obtained from [6]. Due to the sparse nature of the point cloud data for distant objects provided by the LiDAR sensor, the candidate objects produced by the 3D detector could potentially miss these objects. Camera sensors on the other hand have the capability to encompass these objects as it provides color, and texture cues even for distant objects. Based on our observation, most 2D proposals which represent objects at large distances are very small in size which is obvious from a practical point of view. Furthermore, it is evident that proposals with a considerably small size also exhibit a diminutive cross-sectional area. As a result, in our algorithm, we take measures to retain a substantial number of these minuscule proposals with the aim of capturing distant objects. This involves applying a specific threshold value to delineate the inclusion of such proposals from the camera sensor. We ensure that proposals with a cumulative area falling below the defined threshold, denoted as β , are preserved.

4.3 Point Cloud Data Clustering

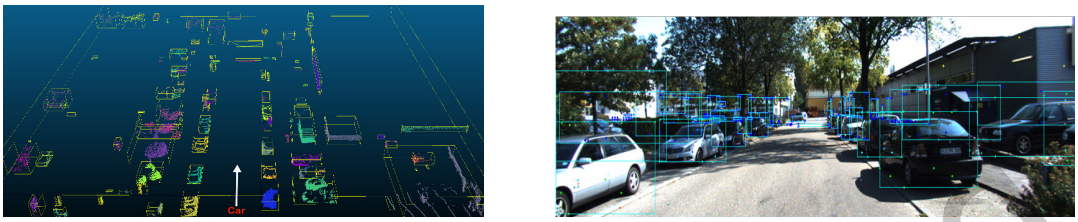
a. DBSCAN: DBSCAN [25] stands for Density-Based Spatial Clustering of Applications with Noise. It is a popular unsupervised clustering algorithm used for identifying clusters of points in a given set of sparse or scattered points. In a given cluster, if a point has several other point close to each other then DBSCAN works by grouping those points into a single cluster thus identifying regions of high density of clusters. Any regions with low density are discarded as outlier or noise. Mainly, there are two parameters taken by DBSCAN algorithm. The first one is epsilon, which is the maximum distance that two points can be from each other and still be considered as a part of the same cluster. The second one is minPts, which is the minimum number of points required to form a cluster or a dense region. In case of point cloud data from LiDAR, which is basically a collection of scanned points from lidar scan we can use DBSCAN for clustering dense points in the given point cloud. In order to do that, first we have to represent the point cloud data as s set of points in a feature space. We can use different properties of the points such as location, intensity or color to construct feature space in a given point cloud data. Once the feature space is constructed, DBSCAN can be applied to the point cloud data to identify clusters of points. Once the clusters are identified we use axis-aligned bounding box method to draw 3D bounding box around that cluster. These boxes are further projected from lidar 3D coordinate to camera 2D coordinate system and used a possible alternative to candidate object from 3D detector.

b. RANSAC: RANSAC [16] stands for "Random Sample Consensus." We can also use this algorithm for fitting models to data in the presence of noise or outliers. It randomly selects a subset of data points and uses these points to fit a model. The next step in this algorithm is to test the model on the remaining points, identifying the inliers that fit the model well and pruning the outliers that do not. The same process is repeated several times, and the best fit to the data is the model with the highest number of inliers. In case of point cloud data, we can use RANSAC, for clustering or segmentation of point cloud, by fitting models to subset of points that are most likely to belong to same cluster. For instance, for data that has 3D points in space, it can be used to fit planes or spheres to subset of points that are in close proximity to each other which are stored as inliers and assigned to a cluster and the points in far proximity to a given threshold are discarded altogether or assigned to other clusters.

c. Connected Component Labeling: Connected component labeling [31] is a technique used in image processing and computer vision to identify and isolate distinct regions or objects in an image. To obtain connected components from LiDAR data, point clouds are segmented into smaller parts separated by a minimum distance. Each part is a set of connected points. We derive this approach from the classic image processing algorithm called Connected Component Labeling (CCL). From a given set of point clouds, CCL is used to detect connected regions. The next step is to generate 3D proposals from these regions and a 3D grid is set for this propose. This grid is deduced from the octree structure which divides a 3D space into at most eight parts to store points. We can control the gap between two adjacent component by selecting different octree levels. Connected regions or components with points less then a certain number or threshold will be ignored . As shown in given Fig. 6 CCL segments LiDAR point clouds into disconnected components, with each box indication a potential object in the sensing data. Each box is then projected from LiDAR cordinate to image cordinate system following the same approach described above to get the final 2D object candidates.

Based on LiDAR point cloud clustering and segmentation we generate axis aligned candidate object by following the same approach as above. One main advantage of using RANSAC is that it is computationally inexpensive and very simple as compared to any 3D detector. It can also handle datasets with irregularly shaped clusters and outliers which is a prominent feature of lidar point cloud data. Additionally, it doesn't require the number of clusters to be specified beforehand, which is particularly useful when working with large dataset such as point cloud data. The only downside of using such lightweight algorithm is that the candidate object obtained

from these algorithm are significantly less accurate in comparison to 3D detector. We prove this hypothesis, through various experiments in our experiment section. Our main aim to include these algorithm is to study the possibility and practicality of used such lightweight methods in fusion architecture.



(a) 3D proposals generated from point clouds

(b) 2D proposals projected on an image

Fig. 6. CCL generates 3D and 2D proposals from point clouds and an image.

4.4 CLOC's Network and Training Details

We adopt the fusion network provided by CLOC's fusion architecture. The candidate results obtained from Proposal Refinement Algorithm denoted by S, are fed to the fusion layers which is a set of 1 x 1 2D convolution layers. Similar to CLOC's network we employ four convolution layers successively to general fused confidence scores for all the potentials association. The layers are: Conv2D(4, 18, (1,1), 1), Conv2D(18,36, (1,1), 1), Conv2D(36, 36, (1,1), 1) and Conv2D(36, 1,(1,1), 1), which yields a tensor of size $1 \times S \times 1$. Each 2D layer has four parameters namely, cin, cout, k,d representing the number of input channels, number of output channels, kernel size and stride respectively. Additionally, we also employ ReLU [22] after each convolution layer is applied for the first three layers. Finally an output tensor Tout is obtained of shape kxnx1 by filling p outputs based on indices (i,j) and this tensor is mapped to desired learning targets, a probability score map of size $1 \times n$, through maxpooling in the first dimension, as mentioned in [17]. Similar to CLOC's architecture we use a cross entropy loss for target classification by the focal loss. Our fusion network is trained using stochastic gradient descent (SGD). We run the training for 12 epochs using Adam optimizer with an initial learning rate of 0.0025 and momentum of 0.9 along with weight-decay factor of 0.0001. The parameter 'momentum' adds a fraction of the previous update to the current update, which helps to smooth out the update trajectory and accelerate the convergence. A value of 0.0 means no momentum, while a value closer to 1.0 means more momentum. CLOC prioritizes 3D candidate objects based on the assumption that the number of objects missed by 3D detector but detected by 2D detector is negligible. As a result of this, they ignore some useful 2D candidate objects. On the other hand, our proposal refinement algorithm conserves some 2D candidate object which eventually help in identifying objects that are not detected by LiDAR sensor, especially objects at far distance. While CLOC usies the IoU approach for refining candidate object we use our own approach as discussed in section 4.2.

5 EXPERIMENTS

We evaluate Sniffer Faster R-CNN++ on the challenging KITTI Dataset [8] to experimentally prove the viability of our proposed framework. This section mainly focuses on our experimental setup, dataset configuration and implementation details. We will also evaluate the performance and impact of our framework in a real-world environment, and compare it against existing state-of-the-art multimodal fusion methods of 3D object detection.

Method	Modality	Car (IoU=0.7)			Pedestrian (IoU =0.5)		
		Easy	Mod	Hard	Easy	Mod	Hard
Early-Fusion							
PFF3D [35]	L+R	81.11	72.93	67.24	43.93	36.07	32.86
Painted PointRCNN [32]	L+R	82.11	71.70	67.08	50.32	40.97	37.87
PI-RCNN [37]	L+R	84.37	74.82	70.03	-	-	-
Complexer-YOLO [28]	L+R	55.93	47.34	42.60	17.60	13.96	12.70
MVX-Net(PF) [29]	L+R	83.20	72.70	65.20	-	-	-
Deep Fusion							
SECOND	L	83.34	72.55	65.82	51.07	42.56	37.29
PV-RCNN	L	87.45	80.28	76.21	47.30	39.42	36.97
PointRCNN	L	86.23	75.81	68.99	49.43	41.78	38.63
PointPillars	L	82.58	74.31	68.99	49.43	41.78	38.63
PointFusion [38]	L+R	77.92	63.00	53.27	-	-	-
RoIFusion [3]	L+R	88.09	79.36	72.51	42.22	35.14	32.92
Late Fusion							
CLOCs	L+R	89.16	82.28	77.23	52.10	42.72	39.08
SECOND+FRCNN	L+R	86.44	78.33	70.73	50.37	39.78	36.44
PointPillars+FRCNN	L+R	87.43	80.20	74.37	50.02	39.98	36.94
PointRCNNs+FRCNN	L+R	87.71	81.38	**77.26**	51.40	42.32	38.98
Sniffer Faster R-CNN++	L+R	88.82	81.47	76.79	51.88	42.56	39.04

Table 1. A comparision of the performance of Sniffer Faster R-CNN++ with the state of the art object detectors evaluated on the KITTI test set. The results are evaluated by the mean Average Precision with 40 recall positions.

5.1 Results on KITTI Dataset

The KITTI Dataset [8] is a popular dataset for 2D and 3D detection in autonomous driving and contains both LiDAR point clouds and camera images along with files for calibration. It contains 7,481 training samples and 7,518 test samples. For our experiments, we divided the official training set into two sets: a set1 with 3712 samples and another set2 with 3769 samples. We use the first training set to train existing 3D detectors and 2D detectors. Our Sniffer R-CNN++ network is trained on set2 training set. The KITTI benchmark requires detecting cars, pedestrians, and cyclists, but for the sake of convenience and accessibility of ground truth, we trained our model on only the car class based on this split. To showcase the versatility of our proposed approach, we utilize a fusion network that combines various 2D and 3D detectors. Specifically, for 2D detectors we incorporate Faster R-CNN , and Cascade R-CNN. Similarly, for 3D detectors we incorporate PV-RCNN, PointPillars, SECOND, PointPainting and PointRCNN. The result of our experiments demonstrate a notable enhancement in the performance of the detectors withe the incorporation of CLOCs and proposal refinement algorithm. Although, the accuracy is marginally below the original CLOC's fusion network we still manage to outperform other existing 2D and 3D detectors. For some instances, aided by our proposal refinement algorithm, we managed to get even better detection result for distant object in comparison to CLOC.

6 EVALUATION RESULTS

Table 1 showcase the evaluation result based on KITTI test set. We evaluate our model using the combination of different set of 2D and 3D detectors. Our Sniffer Faster R-CNN++ (PVR-CNN + Faster-RCNN) outperforms most

Method	2D Detector			3D Detector			Fusion Based	
	Name	GPU	Speed	Name	GPU	Speed	GPU	Speed
SFR + PVR	Faster RCNN	3.8GB	4.2Hz	PV-RCNN	3.5 GB	9.6Hz	1.1GB	3.8Hz
SFR + POR	Faster RCNN	3.8GB	4.2Hz	PointRCNN	4.5 GB	12Hz	1.1GB	3.9Hz

Table 2. Comparision of running speed and GPY memory usage between individual 2D and 3D detector and our fusion based method (Sniffer Faster R-CNN++). Here, the fusion of Sniffer Faster R-CNN and PV-RCNN methods is denoted as SFR + PVR, while the fusion of Sniffer Faster R-CNN and PointRCNN is represented as SFR + POR.

of the existing detection algorithms. Single Modality based methods such as SECOND, PV-RCNN, PointRCNN are improved by our fusion based methods. Significant improvement can be seen especially on the moderate and hard classes as compared to the easy class. In addition to this, our assumption that late fusion based methods are more accurate in comparison to the early-fusion and deep-fusion based methods is also proved by the evaluation result in Table 1. The accuracy of all easy, moderate and hard condition for both Car and Pedestrian class in late fusion based methods is higher then all the other methods. Similarly, our netowork perfroms better then CLOC's in terms of detecting distant object as seen in Fig. 7.

Table 2 illustrates a comparison between the operational speed and GPU memory consumption of our distinct 2D and 3D detectors, alongside the fused model (Sniffer Faster R-CNN++). Given our approach, which involves the concurrent execution of Faster R-CNN and a 3D detector, it becomes infeasible to execute both seamlessly on a solitary GPU system at the desktop level. Therefore, we proceed to assess the individual operational speeds of each model as well as the fused version. As depicted in the table, while the fusion model mandates a greater GPU capacity, it demonstrates superior speed.

Similarly, we also evaluate the effectiveness of our proposal refinement algorithm for detecting distant objects. Fig. 7 illustrates the outcomes of both our proposal refinement algorithm and its adapted version aimed at identifying distant objects. When observing objects in close proximity to the vehicle, specifically within a distance of 0 to 15 meters, the 3D Average Precision (AP) Gain exhibits a negative variance when compared to the baseline model, denoted as CLOC. However, with the progression of distance, there is a noticeable positive trend in the 3D AP Gain. This upward trend indicates an enhanced capability in detecting additional objects as the distance from the vehicle increases. This observation serves to underline that our proposal refinement algorithm, while not causing a substantial enhancement in distant object detection, does contribute to some degree of improvement in this regard. This observation substantiates the efficacy of our algorithm.

In addition to this, we also study the practicality of using alternative methods for generating 3D/2D object candidates. Table 3 showcases the result in terms of time, accuracy and number of candidate object generated during each method. Here, total clusters for 3D object detection refers to the total number of clusters formed based on each method. For instance, application of DBSCAN algorithm results in a total of 964632 clusters of point clouds in KITTI test set. For each frame or point cloud associated with the frame, the total number of clusters or candidate objects for DBSCAN is 128. As we can see, the accuracy is significantly low in comparison to existing 3D object detectors such as PV-RCNN. Although methods such as DBSCAN, CCL can be computationally inexpensive, our finding prove that, using these methods will lower the accuracy substantially.

7 ABLATION STUDIES

Since we are using different threshold for refining proposals, we conducted multiple experiments to get the optimal solution. Fig. 8 shows the effect of variation of beta (used for accommodating small boxes) and lambda values used as a thresholds for proposal refinement and also the IoU approach followed by CLOC's fusion network. As seen in the figure, for both lambda and beta value as the value increases the accuracy increases and

Method	Modality	Total Clusters	Candidate Objects/f	Accuracy	Time(s/f)
Result on 2D Object Detection					
Faster-RCNN	R	7518000	1000	88.97	0.2149
Cascade-RCNN	R	7518000	1000	86.40	0.3147
Snifer Faster R-CNN	L+R	3112452	414	83.71	0.1737
Snifer Faster R-CNN++	L+R	**1706586**	**227**	**93.20**	**0.1438**
Result on 3D Object Detection					
DBSCAN	L	964632	128	57.23	0.16
RANSAC	L	872243	116	52.11	0.16
CCL	L	563604	73	62.34	0.15
PV-RCNN	L	751800	100	87.45	0.15

Table 3. A study based on the practicality of using alternative methods for generating computationally inexpensive 3D candidate object on the KITTI test set with the state of art methods.

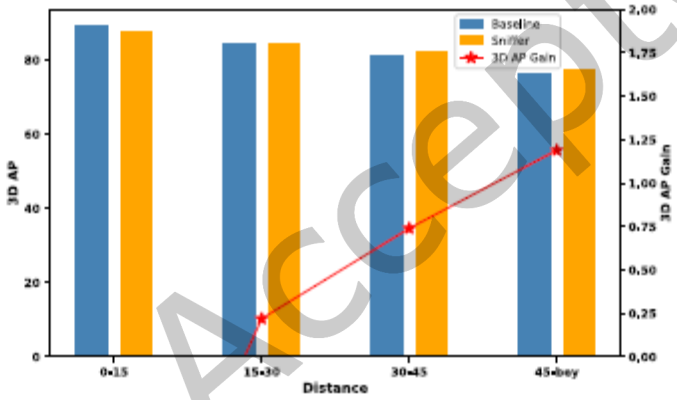


Fig. 7. 3D Average Precision (AP) and 3D AP Gain based on distance for our model (sniffer) and baseline model [17].

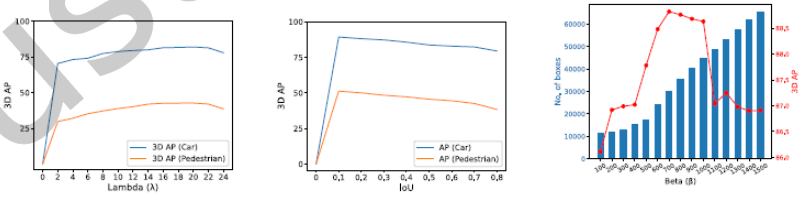


Fig. 8. The effect of variation of lambda and beta values to the Sniffer Faster R-CNN++ network along with the underlying IoU approach used by CLOC's fusion architecture.

becomes stable at a point and starts to decrease again. This is due to the fact that, the increasing values for both lambda and beta results in more boxes being included in the detection process. Although they contribute to the final detection result, they also introduce false negatives and hence the accuracy tend to decrease towards the

PR	IoU	S2d	S3d	focal loss	3D AP
		✓	✓	✓	83.04
✓		✓		✓	55.45
✓			✓	✓	85.67
	✓	✓	✓	✓	89.16
✓		✓	✓		87.37
✓		✓	✓	✓	88.82

Table 4. Effect of different combination of channels in our fusion network. The results are on the easy level car class of KITTI test set with AP calculated by 40 recall positions. PV-RCNN and Faster RCNN are fused in this experiment

end. This is one promising area we have for the future research purpose. We also study the effect of different combination of channels in our network which includes, proposal refinement (PR) for refining proposals using the proposal refinement algorithm, IoU approach used by the CLOC's fusion approach, confidence score from 2D object detection (s2D), confidence score from 3D object detectors (s3D) and focal loss. Each channel provides a major contribution to the network as seen in Table 4. Major contribution is seen from PR as comparison to other channels. Similarly, confidence scores are also fused together during the processing pipeline and therefore plays a major role in final detection. Focal loss on the other hand, addresses the issue of imbalance between positives and negatives among the detection candidates.

As stated earlier, we have trained our Sniffer Faster-RCNN++ model using refined boxes derived from a collaborative process involving candidate objects from a 3D detector and proposals from 2D detectors. Our approach operates in an offline manner, meaning that we initially execute these models to obtain candidate objects and proposals, respectively. However, we acknowledge that this offline nature could potentially create a disparity between our work and its real-world deployment. Consequently, deploying our model in real-world scenarios poses several challenges. To begin with, the offline nature of our method necessitates the parallel execution of the 2D and 3D detectors for real-time deployment. This introduces complexity, as the runtimes of these detectors can vary. Consequently, the generation of 2D proposals and 3D candidate objects might occur at different time intervals, resulting in alignment issues that could affect the performance and effectiveness of the model or overall system. Similarly in dynamic real time environments, where objects and scenes change rapidly, offline methods might struggle to provide accurate implementation. In such a dynamic setting, objects may exhibit real-time movement, alter their positions, or become obscured, factors that are not adequately addressed by offline methods like ours. This forms a component of the future research challenge we intend to address too.

8 CONCLUSION

In this paper, we presented Sniffer Faster R-CNN++, a novel sequential late fusion approach for object detection that inherently performs sensor fusion during the final detection stage, combining the candidate objects obtained from LiDAR with vision data to obtain faster and accurate object detection results. We used proposal refinement algorithm to refine candidate objects from both 2D and 3D detectors and significantly improve the accuracy. Most prominently, our approach aided by the rich texture information from camera image helps in detecting distant objects. We also presented a comprehensive study of different point cloud data clustering algorithms and studied their feasibility as a alternative for computationally expensive 3D detectors. While these methods have less computational overhead we showcase that the application of such methods is questionable due to their overall performance in detection result . Apart from this, we also perform extensive experiments on the KITTI dataset to show the superiority of our proposed architecture over the state of art in terms of both inference and accuracy.

Fig. 9. Qualitative analysis of KITTI results. We created two different comparison figures where green boxes represent accurate candidate boxes and red represent the missed one). For the first comparison, the fig (a), shows the projection of 3D object candidates on the point cloud, fig (b) shows the projection of useful 2D object proposals that complements the missed boxes from 3D lidar detector), fig (c) shows the final result. For the second comparison, the fig (d), shows the projection of 3D object candidates on the point cloud (green boxes represent accurate detection candidates that complements the missed boxes from 2D detector), fig (e) shows the projection of 2D object proposals, fig (f) shows the final result.

9 ACKNOWLEDGEMENT

The work is supported by the National Science Foundation grants CNS-2231519, OAC-2017564, and ECCS-2010332.

REFERENCES

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

[2] Zhaowei Cai and Nuno Vasconcelos. 2018. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 6154–6162.

[3] Can Chen, Luca Zanotti Fragonara, and Antonios Tsourdos. 2021. RoIFusion: 3D object detection from LiDAR and vision. *IEEE Access* 9 (2021), 51710–51721.

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

[5] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. 2021. Voxel r-cnn: Towards high performance voxel-based 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 1201–1209.

[6] Sudip Dhakal, Qi Chen, Deyuan Qu, Dominic Carillo, Qing Yang, and Song Fu. 2023. Sniffer Faster R-CNN: A Joint Camera-LiDAR Object Detection Framework with Proposal Refinement. In *2023 IEEE International Conference on Mobility, Operations, Services and Technologies (MOST)*. 1–10. https://doi.org/10.1109/MOST57249.2023.00009

[7] Sudip Dhakal, Deyuan Qu, Dominic Carrillo, Qing Yang, and Song Fu. 2021. OASD: An Open Approach to Self-Driving Vehicle. In *2021 Fourth International Conference on Connected and Autonomous Driving (MetroCAD)*. 54–61. https://doi.org/10.1109/MetroCAD51599.2021.00017

[8] Andreas Geiger, Philip Lenz, and Raquel Urtasun. 2012. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 3354–3361.

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

[10] Tengteng Huang, Zhe Liu, Xiwu Chen, and Xiang Bai. 2020. Epnet: Enhancing point features with image semantics for 3d object detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*. Springer, 35–52.

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

[12] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. 2019. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 12697–12705.

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

[14] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. 2017. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2117–2125.

[15] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. 2016. Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, 21–37.

[16] Anh Nguyen and Bac Le. 2013. 3D point cloud segmentation: A survey. In *2013 6th IEEE conference on robotics, automation and mechatronics (RAM)*. IEEE, 225–230.

[17] Su Pang, Daniel Morris, and Hayder Radha. 2020. CLOCs: Camera-LiDAR object candidates fusion for 3D object detection. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 10386–10393.

[18] AJ Piergiovanni, Vincent Casser, Michael S. Ryoo, and Anelia Angelova. 2021. 4D-Net for Learned Multi-Modal Alignment. arXiv:2109.01066 [cs.CV]

[19] Charles R Qi, Xinlei Chen, Or Litany, and Leonidas J Guibas. 2020. Imvotenet: Boosting 3d object detection in point clouds with image votes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 4404–4413.

[20] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. 2019. Deep hough voting for 3d object detection in point clouds. In *proceedings of the IEEE/CVF International Conference on Computer Vision*. 9277–9286.

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

[22] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 652–660.

[23] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.

[24] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems* 28 (2015).

[25] Erich Schubert, Jörg Sander, Martin Ester, Hans Peter Kriegel, and Xiaowei Xu. 2017. DBSCAN revisited, revisited: why and how you should (still) use DBSCAN. *ACM Transactions on Database Systems (TODS)* 42, 3 (2017), 1–21.

[26] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. 2020. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10529–10538.

[27] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. 2019. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 770–779.

[28] Martin Simon, Karl Amende, Andrea Kraus, Jens Honer, Timo Samann, Hauke Kaulbersch, Stefan Milz, and Horst Michael Gross. 2019. Complexer-yolo: Real-time 3d object detection and tracking on semantic point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 0–0.

[29] Vishwanath A Sindagi, Yin Zhou, and Oncel Tuzel. 2019. Mvx-net: Multimodal voxelnet for 3d object detection. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 7276–7282.

[30] Vishwanath A. Sindagi, Yin Zhou, and Oncel Tuzel. 2019. MVX-Net: Multimodal VoxelNet for 3D Object Detection. arXiv:1904.01649 [cs.CV]

[31] Alexander JB Trevor, Suat Gedikli, Radu B Rusu, and Henrik I Christensen. 2013. Efficient organized point cloud segmentation with connected components. *Semantic Perception Mapping and Exploration (SPME)* (2013), pp–1.

[32] Sourabh Vora, Alex H Lang, Bassam Helou, and Oscar Beijbom. 2020. Pointpainting: Sequential fusion for 3d object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 4604–4612.

[33] Chunwei Wang, Chao Ma, Ming Zhu, and Xiaokang Yang. 2021. PointAugmenting: Cross-Modal Augmentation for 3D Object Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 11794–11803.

[34] Zhixin Wang and Kui Jia. 2019. Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3d object detection. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 1742–1749.

[35] Li-Hua Wen and Kang-Hyun Jo. 2021. Fast and accurate 3D object detection for lidar-camera-based autonomous vehicles using one shared voxel-based backbone. *IEEE Access* 9 (2021), 22080–22089.

[36] Liang Xie, Chao Xiang, Zhengxu Yu, Guodong Xu, Zheng Yang, Deng Cai, and Xiaofei He. 2019. PI-RCNN: An Efficient Multi-sensor 3D Object Detector with Point-based Attentive Cont-conv Fusion Module. arXiv:1911.06084 [cs.CV]

[37] Liang Xie, Chao Xiang, Zhengxu Yu, Guodong Xu, Zheng Yang, Deng Cai, and Xiaofei He. 2020. 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*, Vol. 34. 12460–12467.

[38] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. 2018. Pointfusion: Deep sensor fusion for 3d bounding box estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 244–253.

[39] Yan Yan, Yuxing Mao, and Bo Li. 2018. Second: Sparsely embedded convolutional detection. *Sensors* 18, 10 (2018), 3337.
[40] Bin Yang, Wenjie Luo, and Raquel Urtasun. 2018. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 7652–7660.
[41] Jin Hyeok Yoo, Yecheol Kim, Jisong Kim, and Jun Won Choi. 2020. 3d-cvf: Generating joint camera and lidar features using cross-view spatial feature fusion for 3d object detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16*. Springer, 720–736.
[42] Yin Zhou and Oncel Tuzel. 2018. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4490–4499.