Chapter 5
3D Object Detection in Autonomous Driving

Peng Yun, Yuxuan Liu, Xiaoyang Yan, Jiahang Li, Jiachen Wang, Lei Tai, Na Jin, Rui Fan, and Ming Liu

Abstract 3D object detection is an important perception module in autonomous driving systems. It recognizes sensor observations and predicts locations, sizes and orientations of key objects, which provides both semantic and spatial information for high-level decision making. In this chapter, we first introduce and analyze the properties of commonly used perceptual sensors in autonomous vehicles: cameras, LiDARs and RADARs. Then we define the research problem, detail the assumptions and introduce evaluation metrics of 3D object detection in the context of autonomous

Peng Yun, Yuxuan Liu, Xiaoyang Yan—equal contribution.
Rui Fan, Ming Liu—co-corresponding author.

P. Yun (✉) · Y. Liu · X. Yan · N. Jin · M. Liu
Hong Kong University of Science and Technology, Hong Kong, Hong Kong
e-mail: pyun@ust.hk

Y. Liu
e-mail: yliuhb@ust.hk

X. Yan
e-mail: xyanaq@ust.hk

N. Jin
e-mail: njinab@ust.hk

M. Liu
e-mail: eelium@ust.hk

J. Li · R. Fan
Tongji University, Shanghai, China
e-mail: 2230745@tongji.edu.cn

R. Fan
e-mail: rfan@tongji.edu.cn

J. Wang
Jilin University, Changchun, China
e-mail: wangjc2119@mails.jlu.edu.cn

L. Tai
Huawei Autonomous Driving Solutions, Shanghai, China
e-mail: ltai@connect.ust.hk

R. Fan et al. (eds.), *Autonomous Driving Perception*, Advances in Computer Vision and Pattern Recognition, https://doi.org/10.1007/978-981-99-4287-9_5

driving. The main body reviews the state-of-the-art techniques and categorize them into camera-based, LiDAR-based, RADAR-based and multi-sensor fusion methods. For each method, we point out the main problems and their existing solutions. By analyzing the limitations of existing methods, we propose promising directions and open problems for future research.

5.1 Introduction

Perception is the primary component in an autonomous driving system. It takes charge of encoding the sensor readings to understand its surroundings. Accurate perception results act as the foundation of decision making. Object detection is one of the important problems in the perception of autonomous driving (Fig. 5.1). It allows vehicles to recognize and locate the key objects they are concerned about (like cars, pedestrians, and cyclists) in the 3D space from the sensor readings. Previously, object detection in autonomous driving was mostly implemented based on the images captured by vehicle-mounted cameras, called image-based object detection [1–4]. Compared to image-based object detection, 3D object detection is a more challenging problem. It is implemented based on various sensors, including cameras, LiDARs and RADARs. Its searching space is larger than image-based object detection due to the extra z-axis and the continuous coordinates. It requires more variables to estimate, including the z-axis position, height as well as the heading orientations. The extra estimation results of 3D object detection provide more information for decision-making in autonomous vehicles.

One challenge for 3D object detection arises from sensor limitations. Cameras suffer from foreshortening, flickering effects, and over-exposure problems; LiDARs and RADARs suffer from low-resolution and sparse data representations. Another challenge arises from environmental variations such as lighting and weather conditions. These factors significantly influence camera-based 3D object detection since the appearance information captured by cameras could be much different in various conditions. The density of the point clouds captured by LiDARs will be reduced by half on rainy days [5]. Besides, occlusion causes performance degradation of object

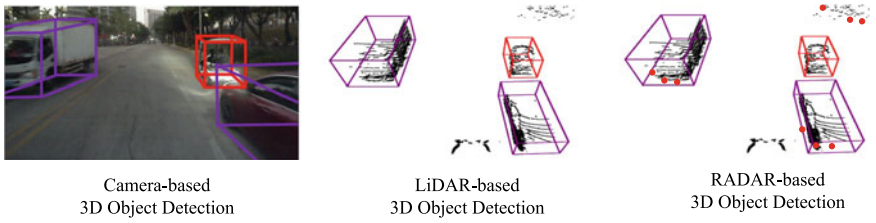

Fig. 5.1 3D object detection with various sensors. (The red dots in the RADAR-based 3D object detection sub-figure denote RADAR observations)

detection. When one object blocks the view of another, it results in partial or complete invisibility of the object.

There have been some survey papers published on 3D object detection in autonomous driving and its related fields [6–8]. However, they mainly focused on the camera- and the LiDAR-based methods and did not review the RADAR-based work. Recently, many state-of-the-art methods have emerged but none are covered in existing surveys. In this chapter, we consider the vehicle-mounted perceptual sensors: cameras, LiDARs, and RADARs, and discuss detection approaches based on uni-modal as well as multi-modal sensor observations. We also provide a more fine-grade classification framework for this problem, and point out crucial problems in each category to help researchers in this field. We claim the following contributions:

- This chapter provides a fine-grade classification framework for current 3D object detection methods in autonomous driving. It generally classifies the existing methods according to the data modal. Besides, it defines the crucial problems for each subcategory and further classifies the methods according to the solution they used.
- Compared to the earlier surveys, this chapter covers the most recent and advanced work. It provides the readers with an in-depth review of the state-of-the-art methods.
- This chapter proposes promising research directions in this field.

In this survey, we first introduce the background concepts and terminology of 3D object detection in Sect. 5.2. We classify the existing research work into camera-based methods, LiDAR-based methods, RADAR-based methods, and multi-sensor fusion methods according to the sensor modality in Sects. 5.3–5.6. We then propose promising research directions for 3D object detection in Sect. 5.7 and concludes this chapter in Sect. 5.8.

5.2 Background Concepts

In this section, we will first introduce the problem definition and assumptions of 3D object detection. Then we describe commonly used perceptual sensors, datasets as well as evaluation metrics.

5.2.1 Problem Definition and Assumptions

We define 3D object detection as a pattern recognition problem. The context of autonomous driving provides this problem with some assumptions which make it different from indoor or aerial applications. We denote $x \in X$ as an input, which can be a dense representation with shape $[W, H, C]$ like images, or a sparse representation with shape $[N, C]$. Let $y \in Y$ be a target, which is a set of 3D bounding

boxes $\{B_{3D}^i = [cls, x_c, y_c, z_c, l, w, h, \theta] | i = 1, ..., n\}$, where cls is the class of the bounding box, $[x_c, y_c, z_c]^T$ is the box center, w, h, l denote the box sizes along the x,y,z-axes respectively (x and y axes define the ground plane in the right-handed coordinate system), and θ denotes the box rotation angle along the z-axis in range of $[0, \pi)$.

We denote $\mathcal{L}$ as a loss function, and $R[f]$ as the expected risk, i.e. $R[f] = \mathbb{E}_{x,y\sim P(X,Y)}[\mathcal{L}(f(x), y)]$, where $P(X, Y)$ is the true data distribution. We denote $f_{A(S)} : X \rightarrow Y$ as a model learned by a learning algorithm A using a training dataset $S := S_m := \{(x_i, y_i)\}_{i=1}^m$ of size m. We adopt $R_S[f]$ as the empirical risk of f as $R_S[f] = \frac{1}{m} \sum_{i=1}^m \mathcal{L}(f(x_i, y_i))$ with $S = \{(x_i, y_i)\}_{i=1}^m$. The goal of 3D object detection is defined as the minimization of the expected risk $R[f_{A(S)}]$. We typically minimize the non-computable expected risk $R[f_{A(S)}]$ by minimizing the computable empirical risk $R_S[f_{A(S)}]$.

The context of autonomous driving implies multiple assumptions for 3D object detection. Firstly, the semantic categories should be frequent and meaningful on the road, like vehicles, pedestrians, cyclists, barriers, etc. The objects like tables and books are ignored. Secondly, the scenes of interest are large-scale outdoor driving scenes across hundreds of meters. The indoor scenes are not considered. Thus it suffers less from stacking problems, like a pile of books stacked on a table. Thirdly, only the rotation along the z-axis is considered. It is a reasonable assumption since most objects on the road we are concerned about (like vehicles, pedestrians, and cyclists) have only yaw heading angles. Some works [9–11] also include ground assumptions whereby key objects are located on the same ground as the ego vehicle to simplify this problem.

5.2.2 Sensors

The commonly used perceptual sensors in autonomous driving include cameras, LiDARs and RADARs. In this section, we will discuss both advantages and limitations of each sensor in 3D object detection.

5.2.2.1 Cameras

Cameras are commonly used in our daily life. They are passive sensors similar to human vision and provide appearance information of their perception regions. An image captured from a camera is digitally represented in a dense array I having a dimension of $[H, W, 3]$, where H and W denote the height and width of the image. Each entry of I is an integer ranging from 0 to 255, representing the pixel intensity. The properties of cameras are listed in the following paragraphs.

Rich textures: Textures refer to information about the spatial arrangement of color or intensities in an image, such as edges, lines, and color patches. Images provide rich texture information of the surrounding environments.

Implicit geometrical shape information: Geometrical shape information is helpful for object detection, and such information is implicitly embedded inside the images and can be extracted by analyzing the contours of the patches or the relationship to their neighborhoods.

No depth information: RGB cameras provide no depth information. Even though stereo camera setups can recover the depth information under geometrical constraints, the extra computation for depth recovery is costly [13]. With the booming of deep learning, the depth information can be estimated in a relatively accurate and fast manner with GPU acceleration [14, 15]. However, deep neural networks suffer from bad generalization ability and require fine-tuning if the data distributions are changed. Current RGB-D cameras can provide depth information, but they have short detection range (less than 10 m), and most of them cannot be used in outdoor scenes due to the sunlight effects, which make them unsuitable for autonomous driving applications.

Long detection Range: A common RGB camera can capture the appearance of key objects like cars and pedestrians 200 m away.

The sensor limitations of cameras are derived from the above-mentioned properties. On one hand, cameras suffer from a lack of depth information. 3D object detection requires estimating the accurate locations, scales, and orientations in a 3D coordinate system. The depth information can be used to directly project each pixel on the image plane to the 3D space. Given the 3D coordinates of each pixel, the location of the key objects is much easier to estimate [16, 17]. On the other hand, rich texture information brings both benefits and limitations. They help distinguish key objects but make the captured image significantly different in various lighting or weather conditions.

5.2.2.2 LiDARs

LiDARs are active sensors and provide the position as well as the reflection intensity of each detected point. A point cloud retrieved from a LiDAR is commonly represented as a point list $P = \{p_i | i = 1, ..., n\}$ (Fig. 5.2b), where each p_i is a vector representing its coordinates in the continuous 3D space and its reflection intensity. The properties of LiDARs are listed in the following paragraphs.

(a) (b) (c)

Fig. 5.2 A visualization of sensor data. **a** camera data; **b** LiDAR data; **c** RADAR data rendered from the nuScenes dataset [12]

Available spatial information: Each point is associated with its coordinates in the LiDAR frame, which helps estimate object size (dimension) and the six-degree-of-freedom (DoF) pose accurately.

Explicit structure and shape information: The points are naturally clustered and separated in a point cloud (Fig. 5.2b). The distribution of points in a local region preserves the structure and shape information.

Sparsity: There is a mass of no-point regions in the point cloud as shown in Fig. 5.2b. The points are distributed loosely in 3D space and are not arranged tightly side-by-side in a grid.

Unordered point list: A point cloud is represented as an unordered point list. The order of the points does not distinguish point clouds. For instance, if $P_1 = [p_a, p_b, p_c]$, $P_2 = [p_b, p_c, p_a]$, we still have $P_1 = P_2$.

Limited detection range: The detection range of LiDARs is always limited to under 100 m. The distant objects can only be detected with a few points.

The sensor limitations of LiDARs are derived from the above-mentioned properties. Firstly, it lacks appearance information and can only use geometrical information to recognize key objects. However, it is hard to recognize their semantic classes for those distant objects detected with only few points. Secondly, the sparsity and the unordered point list representation in the continuous space make point clouds unsuitable for convolution operations. The translation invariance of convolution operations has helped achieve state-of-the-art performance on image recognition tasks. Researchers have to convert the point clouds into a convolutional-friendly representation if they want to adopt convolution neural networks (CNNs) as feature extractors. Otherwise, new feature extractors need to be designed considering the properties above. Thirdly, the density of the point clouds captured by LiDARs might be reduced by half on rainy or snowy days, which may cause false negatives in extreme weather [5].

5.2.2.3 RADARs

In this chapter, we focus on automotive RADARs. The physical mechanisms of RADARs are similar to LiDARs, but they are implemented with microwaves instead of light waves. The on-chip processors of RADARs process the raw data, and finally, a point cloud can be retrieved from RADARs. Different from the point clouds obtained from LiDARs, they are in the 2D space and much sparser (Fig. 5.2c). Besides, the velocity of each point can be measured by RADARs through the Doppler effect. The properties of RADARs are similar to LiDARs except for the following points.

Low resolution: Because of the longer wavelength, the resolution of RADARs is lower than LiDARs. As a result, the point clouds of RADARs are much sparser than LiDARs' and are less precise in terms of localization.

Moderate detection range: The detection range of RADARs is longer than the LiDAR range but shorter than that of cameras. There are different types of RADARs suitable for different ranges: short-range RADARs (less than 30 m), moderate-range RADARs (less than 100 m), and long-range RADARs (less than 250 m).

Robust in extreme weathers: The point clouds of RADARs are much more robust in harsh environments (poor lighting and weather conditions, as well as extreme temperatures) than LiDARs [12], which allows it to be commonly used in blind-spot detection and collision avoidance.

The low resolution is one of the major limitations of RADARs in 3D object detection. It provides 2D locations of each point only (the 2D space defined by the x-axis and y-axis), which limits the possibility of height estimation. Besides, even though its detection range can be as high as 250 m, the automotive RADARs become unreliable at 10–15m when working in real-world scenes due to reflections from other objects [18]. The points of a RADAR projected onto a car 50m away can be less than 10 points [12]. Whether RADARs can accurately capture an object largely depends on its reflection strength. The reflections of radar signals from cars are always strong, while pedestrians and cyclists are smaller in size and have relatively few hard or metallic surfaces to reflect radar signals. For instance, a child standing next to a vehicle can be overlooked by a RADAR system. These are the reasons why 3D object detection algorithms based on automotive RADARs are uncommon.

5.2.2.4 Summary

The comparison between the properties of cameras, LiDARs, and RADARs are shown in Table 5.1. Due to the rich appearance or spatial information, cameras and LiDARs can be used independently for 3D object detection. There are numerous successful implementations solely based on camera or LiDAR data. In contrast, there are few 3D object detector based on automotive RADAR data only, since its data is too sparse and insufficient to estimate semantic classes.

Since different types of sensors have their pros and cons in different conditions, joint treatment of sensor data is essential for 3D object detection [12]. Multi-sensor fusion is a popular research direction in 3D object detection, which provides not only complementary but also redundancy in the face of sabotage, failures, adverse conditions and blind spots [19] (Fig. 5.3).

Table 5.1 Comparison between different types of sensors

	Cameras	LiDARs	RADARs
Information for object detection	Appearance	Spatial	Spatial & velocity
Data representations	Dense array	Unordered point list	Unordered point list
Robustness to extreme conditions	*	**	***
Detection range	$\geq 250m$	$\leq 100m$	$\leq 250m$

The number of '*'s represents its robustness level

Fig. 5.3 Visualization rendered from the KITTI 3D object detection dataset [20]. The bottom row contains the BEV images of the two scenes in the upper rows

5.2.3 Public Datasets

Data is indispensable to supervised learning algorithms. In this section, we will introduce three representative 3D object detection datasets: KITTI [20], nuScenes [12] and Waymo [21] datasets.

5.2.3.1 KITTI Dataset

KITTI dataset is one of the most commonly used datasets in autonomous driving [20]. Its recording platform is a standard station wagon with two color and two grayscale PointGrey Flea2 video cameras, a Velodyne HDL-64E 3D laser scanner, a GPS/IMU localization unit with RTK correction signals [20]. All these cameras were rectified and well-calibrated with LiDAR and the GPS/IMU unit. The data collection scenes include well-structured highways, complex urban areas, and narrow countryside roads.

3D object detection is one of the benchmarks provided in the KITTI challenges. The dataset includes 7,481 training frames and 7,518 test frames, all of which come

with sensor calibration information and annotated 3D boxes surrounding objects of interest. Along with the sensor data from LiDARs and cameras, a 3D object detection benchmark is provided for researchers to test their detection methods. Annotations are classified into three categories, namely easy, moderate, and hard cases, based on the size of objects, the degree of occlusion, and the level of truncation. The metrics that the KITTI dataset adopts consist of AP_{2D}, AP_{3D}, and AP_{BEV} for precision evaluation, and AOS for orientation evaluation, which will be discussed in Sect. 5.2.4.

Even though the KITTI dataset is well-known and popular, some limitations still exist. Firstly, all the measurements of KITTI were collected during the daytime and mostly under sunny conditions. It is hard to conduct research on the generalization ability and robustness against extreme conditions on the KITTI dataset. Secondly, RADARs were not considered in the KITTI recording platform, which limits the research output on RADAR-based 3D object detections. In addition, the class frequency is highly unbalanced: 75% car, 4% cyclist, and 15% pedestrian [22]. As a result, some 3D object detection algorithms were only tested on the car class due to insufficient samples of cyclists and pedestrians. Moreover, the majority of objects tend to have a dominant orientation, with their front facing towards the ego-vehicle.

5.2.3.2 NuScenes Dataset

The nuScenes dataset was specifically collected for object detection and tracking [12]. This dataset is designed for the driving context and contains sensory data from a fully autonomous vehicle, including 6 cameras, 5 RADARs, and 1 LiDAR, all with a complete 360-degree field of view. All the sensors are well-calibrated. They collected data using two Renault Zoe supermini electric cars equipped with an identical sensor layout, and the cars were driven in Boston and Singapore allowing researchers to evaluate their work across different geographic locations.

There are 40K frames annotated with 3D bounding boxes of 23 categories, including car, adult, child, barrier, and animal, which are in finer grain degrees compared to the KITTI dataset. On average, the nuScenes dataset has seven pedestrians and twenty vehicles per frame. Besides, they considered the effects of extreme conditions (nighttime and rainy days). The rainy and night frames account for 11.6% and 19.4% in the full 40k annotated frames. To evaluate the generalization capability of algorithms on such a dataset, they also collected data from different scene locations: Boston: 55%, Singapore-OneNorth: 21.5%, Singapore-Queenstown: 13.5%, Singapore-HollandVillage: 10%. They adopted AP_{BEV} and nuScenes detection score (NDS) as evaluation metrics. NDS is a new metric they proposed for evaluating the results on the nuScenes dataset. Half of the NDS is based on AP_{BEV}[1] while the remaining half of the measurements evaluates the accuracy of the detections based on box location, size, orientation, attributes, and velocity, calculated using their corresponding distances [12].

[1] It is noted that, different from the method calculating with intersection over union in the KITTI dataset, nuScenes calculate the AP_{BEV} with the 2D center distance on the ground plane.

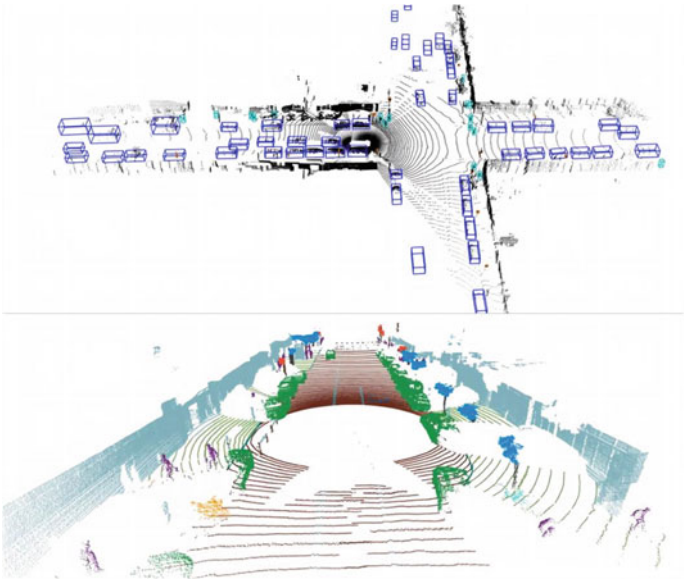

Fig. 5.4 Visualization rendered from Waymo Open Dataset [21]. The upper row is for object detection and tracking, while the lower row is for semantic segmentation

Compared to the KITTI dataset, the nuScenes dataset provides more data annotations covering more driving conditions. The available RADAR data will motivate researchers to contribute more on RADAR-based 3D object detection algorithms. The possible improvement from dataset aspects is two-fold. Firstly, more extreme conditions should be considered. There are some omitted conditions like snowy and stormy days. The ice layer formed on RADAR sensors may cause performance degradation, and the strong wind may cause physical vibration of sensors. Secondly, multi-LiDAR settings are also one type of common setups in autonomous vehicles, which is not considered in the current datasets (Fig. 5.4).

5.2.3.3 Waymo Open Dataset

Waymo dataset [21] is collected by the following sensors: 1 mid-range LiDAR, four short-range LiDARs, and five cameras (front and sides). The data is collected from various places, including Los Angeles, Detroit, Mountain View, San Francisco, Seattle, and Phoenix. This dataset also includes various environments, objects, and weather conditions.

The Waymo Open Dataset provides object detection and tracking capabilities with a collection of 12.6 million 3D bounding box labels, each with tracking IDs, for four object categories, namely vehicles, pedestrians, cyclists, and signs, on LiDAR data. Additionally, the dataset also contains 11.8 million 2D bounding box labels, also

Table 5.2 Comparison between KITTI [20], nuScenes [12], and Waymo [21] datasets

Dataset	# ann. frames	# 3D boxes	Night	Rain	Snow	Locations
KITTI	15k	200k	No	No	No	Karlsruhe
nuScenes	40k	1.4M	Yes	Yes	No	Boston, SG
Waymo	39k	12.6M	Yes	Yes	No	LA, Detroit, MV, SFO, Seattle, Phoenix

with tracking IDs, on camera data. These LiDAR labels are 3D 7-DoF bounding boxes with globally unique tracking IDs. The metrics used by Waymo Open Dataset include AP_{2D}, AP_{3D}, and APH for precision evaluation.

Being the most recent among these well-known open-source datasets, the Waymo dataset uses better sensors for data acquisition, resulting in denser LiDAR point clouds, better image data quality, and more objects in each frame. However, this is also a challenge for developers because larger data volumes bring more computational and reasoning pressure on processors.

5.2.3.4 Summary

The comparison between representative datasets is shown in Table 5.2. In summary, current datasets are sufficient to support the development of 3D object detection algorithms [11, 23–25] for basic classes (car, pedestrian, cyclist) in common scenes (sunny structured scenes). In the future, it is an urgent need to collect data in more challenging conditions, like snowy days, animal-on-the-road scenes.

5.2.4 Evaluation Metrics

In this section, we will discuss the metrics for 3D object detection evaluation. The evaluation metrics mainly contain two aspects: the precision metrics for localization and bounding box size evaluation (average precision), as well as the rotation similarity metrics for orientation evaluation (average orientation similarity).

5.2.4.1 Average Precision

For detection and classification tasks that output a probability y_i of the sample x_i belonging to positive samples in the given ground truth, recall rate is defined as the ratio of all positive samples ranked above a certain threshold t to the total number of positive samples:

$$r(t) = P(y_i \geq t | x_i \in C), \tag{5.1}$$

where C consists of all positive samples that exist in the ground truth. Precision is calculated as the ratio of all samples above the threshold t in the given ground truth to the total number of samples predicted above that threshold.

$$p(t) = P(x_i \in C | y_i \geq t). \tag{5.2}$$

By setting t, we can get a recall and precision rate accordingly, so that a precision-recall curve can be drawn by setting different $t \in [0, 1]$. The average precision (AP) is the area of the region below the precision-recall curve, which can be approximately computed as

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots, 1\}} p_{interp}(r), \tag{5.3}$$

where r denotes the recall rate, and $p_{interp}(r) = \max_{\hat{r}:\hat{r} \geq r} p(\hat{r})$ is an interpolated alternative of the precision for a recall r.

At the very beginning, researchers adopted a similar AP in image detection, called AP_{2D}. For AP_{2D}, samples were considered as true positives, if the intersection over unions (IoUs) of the estimated and ground-truth 2D boxes on the image plane exceeds a specific value. It was recommended to combine AP_{2D} and AOS (which will be detailed in the next section) to evaluate the 3D object detection results in both locations, box size, and orientation. However, the effects of localization and bounding box size estimation cannot be decoupled by these two metrics [26].

AP_{2D} also suffers from the foreshortening effects that two objects with different sizes may project to the same 2D bounding box on the image plane. To solve this problem, some researchers computed the IoU between the bird's-eye-view (BEV) projection of the estimated and the ground-truth 3D boxes on the BEV image plane and proposed AP_{BEV}. The drawback of AP_{BEV} is that height error is not considered in the evaluation. Therefore, researchers computed AP_{3D} with the IoU computed between the estimated and ground-truth 3D boxes in the 3D space. Specifically, the IoU thresholds are 0.5 for pedestrians and cyclists and 0.7 for cars in the KITTI 3D object detection Benchmark [20].

In the nuScenes benchmark [12], the AP is specially computed according to the 2D center distance on the ground plane instead of IoUs. They claimed it decoupled the metric AP from object size and orientation and suited well to the evaluation of small objects like pedestrians. To decouple all the locations, scale, and orientation evaluations, they proposed average translation error, average scale error, and average orientation error [12]. They used a linear combination of AP and the average errors to indicate the thorough performance of the 3D detectors. They called this metric as called nuScenes detection score (*NDS*). In contrast, Waymo benchmark [21] proposed to weigh the true positives by their heading accuracy [21]. The weight scalar is defined as $\min(|\tilde{\theta} - \theta|, 2\pi - |\tilde{\theta} - \theta|)/\pi$, and this weighted average precision is called *APH*.

5.2.4.2 Average Orientation Similarity (AOS)

Similar to the definition of AP, average orientation similarity (AOS) is defined as

$$AOS = \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots, 1\}} \max_{\hat{r}: \hat{r} \geq r} s(\hat{r}), \tag{5.4}$$

where the orientation similarity $s \in [0, 1]$ at recall r is a normalized variant of the cosine similarity defined as

$$s(r) = \frac{1}{|D(r)|} \sum_{i \in D(r)} \frac{1 + \cos(\Delta_\theta^{(i)})}{2} \delta_i \tag{5.5}$$

where $D(r)$ refers to all object detections at a given recall rate r, while $\Delta_\theta^{(i)}$ represents the difference in angle between the estimated and ground truth orientation of detection i [20]. In order to penalize the cases where multiple detections correspond to a single object, δ_i is set to 1 if detection i overlaps with a ground truth bounding box by at least 50% in 2D, indicating that it has been assigned to that object. If the detection does not overlap sufficiently with any ground truth bounding boxes, δ_i is set to 0 to indicate that it has not been assigned. Through a similar exploration as AP, some researchers modified AOS by replacing the IoU computation method. Instead of computing IoUs on the image plane, they computed it in 3D space, resulting in a metric the metric called average heading similarity (AHS) [11].

5.2.4.3 Summary

Table 5.3 shows that $AP_{2D} + AOS$ and AP_{3D} are good choices for generally evaluating a 3D object detector. In practice, AP_{3D} is a harsher metric than $AP_{2D} + AOS$, since the methods perform lower than 10% in AP_{3D} even if they get more than 90% in $AP_{2D} + AOS$ [9, 10, 27].

Even though AP_{3D} can be used as a good metric for evaluating the general performance of a 3D object detector, it fails to decouple the effects of localization, scale, and orientation estimation. As a result, it is hard for researchers to analyze and trace back

Table 5.3 Comparison among different metrics for 3D object detection

Metrics	Loc. (x,y)	Loc. (z)	Scale (x,y)	Scale (z)	Orientation
AP_{2D}	Yes	Yes	Yes	Yes	No
$AP_{2D} + AOS$	Yes	Yes	Yes	Yes	Yes
AP_{BEV}	Yes	No	Yes	No	Yes
AP_{3D}	Yes	Yes	Yes	Yes	Yes

the reasons for the bad performance of their detectors. Promising solutions include the metrics such as *NDS* and the set of average errors we mentioned in Sect. 5.2.4.1. Both of them not only consider general performance but also decoupled performance in each aspect.

5.3 Camera-Based Methods

Images captured by cameras can be used for 3D object detection due to their rich textures, but camera-based object detection suffers from the lack of depth information which is greatly helpful for accurate spatial information estimation, including locations, scales, and orientations of 3D boxes. Therefore, one major riddle for camera-based methods is: **how to project 2D input to 3D and learn depth information from object supervision?**

We observe that the final feature representation significantly affects the architecture, training process, and computation complexity in camera-based methods. Depending on the output feature representation, we divide camera-based methods into (1) result-lifting and (2) feature-lifting methods.

5.3.1 Result-Lifting Methods

Result-lifting methods refer to models that are based on 2D features. These methods first make predictions on the image plane using the 2D features, then estimate depth information, and finally lift the 2D detections into the 3D space. A typical example is shown in Fig. 5.5. Depending on the method of obtaining depth information, we further group these methods into three categories: direct depth prediction, depth from keypoints, and depth from priors.

Direct Depth Prediction: The network structure for direct depth prediction is concise and straightforward. Thus, there exist numerous works have emerged to follow in this direction. In MonoPair [29], object locations and spatial constraints between matched object pairs are computed together using uncertainty-aware spatial constraints to optimize the predicted 3D locations of the objects. It utilizes the pair-wise spatial constraint to model the relationship between two neighboring objects. During the prediction, it imposes aleatoric uncertainty on the detector, formulates the predicted 3D locations and their pair-wise spatial information into a nonlinear least squares problem, and finally predicts the 3D results. On the other hand, Transformer [30] has shown great potential in computer vision, and researchers recently tried to apply it to 3D detection. However, it is challenging to apply the learned object query [31] to fully represent the object property in the image-based 3D detection task. The reason is that the size of objects at far and close distance fluctuates significantly due to the perspective projection. As a solution, MonoDTR [32] proposed the first transformer-based fusion module, which globally merges the image and depth

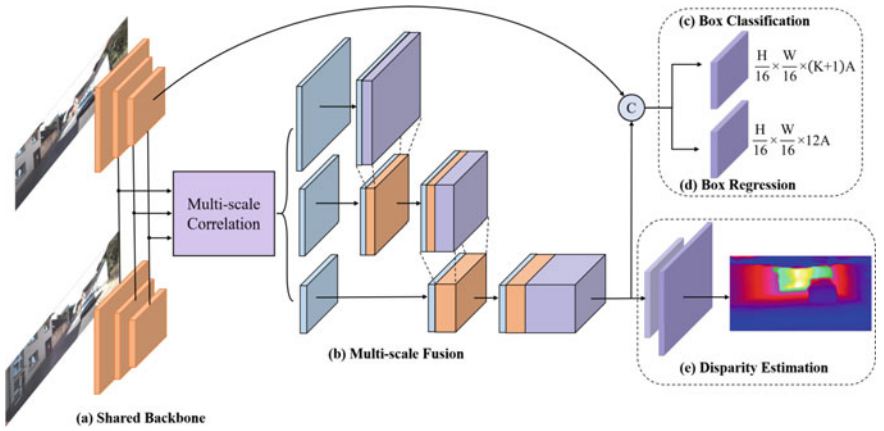

Fig. 5.5 YoloStereo3D architecture [28]. Backbones and multi-scale features both process features in 2D. 3D results are assembled from dense 2D predictions

information. The model employs a depth-aware feature enhancement (DFE) module that seeks to depth-aware features through implicit auxiliary supervision. These depth-aware features nicely assist the monocular 3D object detector, while preventing introducing high computational cost and inaccurate depth priors from using the off-the-shelf depth estimator.

In monocular 3D object detection tasks, getting an accurate depth estimate is always an ill-posed problem. To address the issue of inaccurate depth prior, the Depth-conditioned Dynamic Message Propagation (DDMP [33]) network presents a new method that utilizes a graph-based approach. This method utilizes contextual nodes in the image context, and predicts hybrid filter weights as well as affinities using aligned multi-scale depth features to propagate messages. In contrast, DD3D [34] proposes to use a wider range of data sources to address the problem of depth inaccuracy. The pseudo-lidar methods[35], have good scalability, while end-to-end detection networks [29] with simple structures and better generalization benefit less from large-scale depth data. DD3D aims to get the best of both worlds. It introduces a new 3D detection architecture that is fully convolutional and single-stage and effectively uses monocular depth estimation for pre-training. It leverages a large amount of unlabeled raw data, and its depth perception module benefits from pre-training on large labeled 2D detection datasets.

Depth from Keypoints: Keypoint-based methods usually make use of geometric constraints in the image for prediction. Stereo R-CNN [17] adopts the region-based framework and introduces another image to provide the spatial cues. Specifically, it simultaneously detects and associates objects from stereo pairs, uses the Stereo RPN module to produce left and right RoI proposals, then concatenates left-right RoI features to classify object categories and regress accurate 2D stereo boxes. Sparse constraints for 3D box estimation are developed using the results and keypoints. 3D box predictions are then developed based on projection relations between 3D box

corners and 2D left-right boxes and keypoints. KM3D [36] predicts the projection of 9 keypoints for each object in the image using the center-net baseline, and it predicts depth purely from the minimization of the projection error between keypoint predictions and instance predictions. MonoFlex [37] focuses on using only the 3D height and the visual height of the object to predict depth, fully utilizing the knowledge of autonomous driving scenes.

Depth from Priors: As mentioned before, monocular methods highly dependent on depth estimation due to the lack of depth perception. Researchers have attempted to improve the accuracy of depth prediction by mining the information implied in images through various methods. Geometry projection is a technique that can be used to detect 3D objects from 2D images. This method estimates depth by using the object's height, which is a mathematical prior that can be incorporated into a deep learning model. M3D-RPN [38], GAC [39], and YoloStereo3D [28] collect depth priors from the training dataset to help define each anchor. However, the process of projection can result in the amplification of errors, causing the estimated height to be inaccurately reflected at the output depth. This amplification of error can lead to challenges in controlling depth inferences and can negatively impact training efficiency. To solve this problem, GUPNet [40] introduces a geometric uncertainty projection approach. It proposes a GUP module to obtain the geometry-guided uncertainty of the inferred depth, and adopts uncertainty theory to model the projection process under the probability framework.

5.3.2 Feature-Lifting Methods

Feature-lifting methods transform intermediate features into world coordinates and produce final predictions in BEV or 3D volumes. We further categorize these methods into pseudo-lidar methods and feature mapping methods.

Pseudo-Lidar Methods: Pseudo LiDAR methods, stemmed from [35], correspond to a framework that explicitly produces point cloud from images and applies LiDAR-based 3D detection methods to produce the detection output. Pseudo-LiDAR++ [41] uses sparse LiDAR signals to optimize the point cloud from the depth estimation network locally. Refined-MPL [42] demonstrates that point cloud generation quality does matter in 3D object detection performance. We point out that the above methods usually contain two separate modules trained independently. As a result, depth prediction networks cannot receive supervision signals from object-level supervision. To solve this problem, Pseudo-LiDAR E2E [43] develops a differentiable voxelization module and makes the entire network pipeline end-to-end trainable. Such methods are generally robust to camera parameter changes because they decouple depth prediction from 3D detection. However, the explicit point cloud prediction ignores the uncertainty and errors in depth prediction, and only the expectation of the predicted depth distribution is mapped to the downstream 3D detection part, making the pipeline sub-optimal. Further, CG-Stereo[44] incorporates uncertainty and confidence maps into 3D detection and achieves better detection accuracy.

Feature-Mapping Methods: Feature-mapping methods learn a mapping between 2D image features and 3D volume features. They learn a depth distribution for each image pixel and project the entire 3D volume in camera coordinates to the world coordinates [45–49]. Lift-Splat-Shoot [45] shows that mapping 2D features into 3D makes it easier to form a unified representation for surround-view cameras and other sensors. CaDNN [49] learns a depth distribution for each image pixel and lifts the features as a camera frustum. The frustum in camera coordinates is sampled into a feature volume in the world coordinates. LIGAStereo [48] benefits from sharing a similar feature space with LiDAR detection and applies LiDAR distillation to the 3D volume produced by the stereo network. BEVFusion[50] introduces a BEV Pooling module specifically designed for a parallel acceleration of the sampling process under the condition that the cameras have stable parameters. A common properties of these methods is that their feature mapping is driven by geometric constraints.

In contrast, some recent works adopt semantic-driven feature mapping, where semantics in the images will also affect the mapping between two features [51–53]. They achieve this with the cross-attention mechanism. PETR[51] directly uses transformers with carefully-designed positional embedding to achieve 2D-3D mapping. Furthermore, BEVFormer[52] applies deformable attention modules [53], where the reference points are geometrically determined, while the offsets are semantically determined, and takes advantage of both streams of methods.

5.3.3 Summary

Table 5.4 shows the state-of-the-art results for camera-based methods in the KITTI dataset. Both result-lifting methods and feature-lifting methods remain popular in academics and the industry. Recent result-lifting methods are fast and easy to deploy because of simple operators, and many of them are robust to changes in camera parameters or scenes. Feature-lifting methods are also popular because it is found that 3D features are easier to merge with multi-camera features, LiDAR features, and temporal features. The future developments of camera-based methods of both types are drawing growing interest for computer vision researchers and autonomous driving engineers.

5.4 LiDAR-Based Methods

The geometry information captured by LiDARs can be used for perception, and accurate spatial information is helpful for precise 3D object location. However, LiDARs suffer from sparsity and CNN-incompatible representations. Researchers who intend to percept key objects from LiDAR scans have to deal with the problem: **How to extract features from point clouds?**

Table 5.4 Comparison among the performances of camera-based methods based on the KITTI benchmark (test set)

Modality	Category	Implementation	Car (AP_{3D})			Time(s)
			Easy	Mod	Hard	
Stereo Cameras	Result-Lifting	3DOP [9]	6.6	5.1	4.1	3
		Stereo RCNN [17]	47.6	30.2	23.7	0.3
		YoloStereo3D [28]	65.8	40.7	30.0	0.08
	Feature-Lifting	Pseudo-LiDAR [35]	54.5	34.1	28.3	0.4
		Pseudo-LiDAR++ [41]	61.1	42.4	37.0	0.4
		Pseudo-LiDAR E2E [41]	64.8	43.9	39.0	0.4
		DSGN [47]	73.5	52.2	45.1	0.67
		LIGAStereo [48]	81.4	64.7	57.2	0.4
Mono Camera	Result-Lifting	Deep3D Box [27]	5.9	4.1	3.8	–
		MonoPair [29]	13.0	10.0	8.7	0.06
		M3DRPN [38]	14.8	9.7	7.4	0.16
		KM3D [36]	16.7	11.5	9.9	0.04
		DDMP [33]	19.7	12.8	9.8	0.18
		GAC [39]	21.6	13.2	9.9	0.05
		MonoDTR [32]	22.0	15.4	12.7	0.04
		GUPNet [40]	22.3	15.0	13.1	–
		DD3D [34]	23.2	16.3	14.2	–
	Feature-Lifting	Pseudo-LiDAR[35] [10]	10.8	7.5	6.1	0.1
		AM3D [54]	16.5	10.7	9.5	0.4
		RefinedMPL [42]	18.1	11.1	8.9	0.15
		CaDDN[49]	19.2	13.4	11.5	0.64

5.4.1 Quantization+CNN-Based Methods

One solution is to convert the input point clouds into convolutional-friendly representations and then apply CNNs to extract features. The conversion process is called quantization. Common quantization processes include projecting the point cloud into bird's-eye-view images [22, 55–57] or voxelizing the point cloud into a grid [23, 58–63]. The 3D spatial information is encoded into the grids by hand-crafted features, like point density, distance, occupancy, etc. After quantization, high-level features

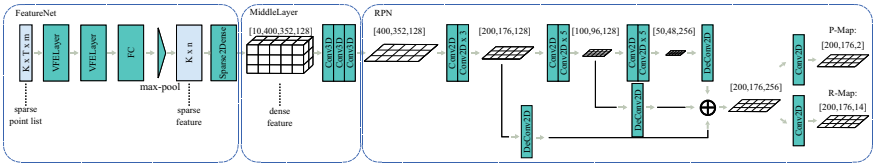

Fig. 5.6 VoxelNet architecture [23]. It consists of three main parts: FeatureNet (point-wise and voxel-wise feature transformation), MiddleLayer (3D dense convolution), and RPN (2D dense convolution)

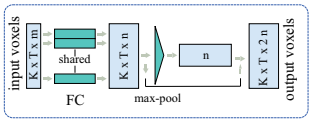

Fig. 5.7 VFELayer of VoxelNet [23]. The input is K voxels, within which there are T points with m feature channels. An MLP transforms the inputs into the feature space, and a max-pooling layer aggregates point-wise features into a global feature

can be extracted from the grid with CNNs and further used for classification and regression tasks.

However, the hand-crafted features in quantization are always sub-optimal to the visual tasks [56, 64]. Zhou et. al. proposed an end-to-end network for 3D object detection, called VoxelNet, where the voxel-wise features were learned from raw point clouds instead of hand-crafted by researchers [23]. The network architecture is shown in Fig. 5.6. To learn the voxel-wise features from point clouds, they first grouped the points according to their corresponding position in the grid. Then they applied a bunch of novel voxel feature encoding (VFE) layers to extract point-wise features from the points inside each voxel. The VFELayer contains multi-layer perceptrons (MLPs) and a max-pooling layer sequentially, as shown in Fig. 5.7. Finally, the point-wise features inside each voxel were aggregated into a voxel-wise feature with a max-pooling layer.

The MiddleLayer limited the running-time performance of VoxelNet and accounted for 50% computation of the whole network due to the 3D dense convolution operation. The grid and dense convolution in 3D are inefficient from both memory and computation aspects. Due to the sparsity of point clouds, the non-empty voxels only occupy $\leq$1% of the grid.[2] For dense 3D convolution, each voxel will incur addition-multiplication operations no matter if it is zero or not. Yan et al. adopted the spatially sparse convolution [65] to deal with the sparsity of point clouds [59]. As a result, the running-time performance of VoxelNet was improved by 4$\times$ (230 ms to 50 ms per frame).

One benefit of Quantization+CNN methods is that the success of vision tasks can be easily extended to LiDAR-based 3D object detection. Yin et al. claims

[2] If a point cloud is partitioned into a [10,400,352] dense grid, only around 5300 voxels are non-empty.

previous anchor-based detection unnecessarily increases the computational burden and tends to induce a huge number of false positives [60]. They extended the CenterNet [66, 67] from image-based 2D detection to LiDAR-based detection and proposed a center-based two-stage framework for 3D object detection. Compared to anchor-based parameterization, the center-based method reduces the search space since centers do not have an orientation dimension. Adopting auxiliary tasks to improve performance has achieved great success in vision tasks [1, 68, 69]. He et al. adopted the same idea and proposed to add auxiliary networks for estimating foreground points and centers to learn structure information [61]. Similarly, Ye et al. demonstrated that multi-task learning of segmentation and detection improves the overall performance of all these tasks [63].

Occlusion is a problem hindering both image- and LiDAR-based detection tasks. Xu et al. [70] explored the occlusion problem in 3D object detection and discussed three causes of occlusion and signal miss in LiDAR point clouds: external occlusion, signal miss, and self-occlusion. By considering these occlusion causes, they proposed BTCNet, which first estimates the occupancy of complete object shapes in the region affected by occlusion and signal miss, then applies a two-stage detector to integrate the occupancy information in generating proposals and finally refine the results. The limitation of their work is that it requires an additional occupancy map estimation network, and the computational effectiveness requires improvement.

The Quantization+CNN methods share two limitations. Firstly, information loss is induced in the stage of quantization. The accuracy highly depends on the quantization resolution. As resolution increases, information loss decreases, and positive samples are more distinguishable from negative ones. However, the improvement in accuracy is obtained at the cost of real-time performance. Secondly, the computational cost is not polynomial to the input size (i.e., the number of points in the point cloud). It is polynomial to the grid size determined by the size of RoI and the resolution.

5.4.2 Point-Based Methods

Quantization induces information loss, and results in unnecessarily voluminous data, and causes high computational costs when applied to large-scale scenes. To avoid quantization artifacts, some researchers designed special network structures to learn features from the input point clouds directly [24, 71–74].

Qi et al. [71] proposed PointNet for end-to-end learning representations directly from point clouds. They endowed PointNet with order invariance by applying a symmetric function. A symmetric function is a function that takes n vectors as input and produces a new vector that is invariant to the order of the input. Examples of symmetric functions include the $+$ and $\times$ operators. On account of the transformation invariance, they proposed T-Net, which was intended to align all input point clouds to a canonical space. A T-Net consists of MLPs and a max-pooling layer at the end and regresses the affine transformation 3×3 matrix. To force the estimated vector by

T-Net to coincide with an affine transformation matrix, they added a regularization term in their loss function:

$$L_{reg} = ||I - AA^T||^2, \tag{5.6}$$

where A is the matrix estimated by the T-Net, and I denotes the identity matrix.

Exploiting local structures has proven to be important for the success of convolutional architecture in visual tasks. PointNet can extract point-wise and global features, but the local features are hard to capture. In their following work [72], they proposed set-abstraction blocks based on PointNet to capture the local features of the point clouds, where each set-abstraction block sequentially samples center points, groups neighbor points and extracts features. Wang et al. [74] extended the spirit of graph neural networks to 3D point clouds to learn the local features from the point clouds which was based on the graph dynamically built with k-nearest neighbors (KNN). Li et al. [73] proposed a more efficient method to catch up with the local feature of point clouds. Instead of grouping the points with ball query methods and KNN methods like [72, 74], they learned the spatial distribution of the input point clouds with the self-organizing map in an unsupervised manner, so that the training time was largely reduced.

Shi et al. [75] extended PointNet architecture to 3D object detection and proposed PointRCNN. They adopted the PointNet++ network to implement foreground segmentation and point-wise bounding box regression. The estimated bounding boxes of the predicted foreground points were filtered out as 3D proposals, and the points inside each proposal were further used to refine the 3D bounding box. PointRCNN achieves better accuracy on the KITTI benchmark than VoxelNet on car class, which proves that the point-cloud-based learning representations can solve the information loss induced by quantization. Further, Qi et al. [76] adopted both PointNet architecture and hough voting to frame an end-to-end 3D object detection pipeline, called VoteNet. It estimates vote centers with the PointNet++ network and then clusters them into K groups. The K groups vote centers are further used for 3D bounding boxes regression. The above Point-based methods all consist of two stages, which have common limitations due to their slow inference time. Recently, Zhang et al. [77] proposed a one-stage point-based 3D object detector, and it runs at more than 80 FPS on the KITTI dataset. They claimed the bottleneck of point-based detectors is sampling resolution and proposed two learning-based instance-aware down-sampling strategies. With these two down-sampling strategies, their proposed approach largely improves the real-time performance of LiDAR-based object detectors with state-of-the-art accuracy.

5.4.3 Point-Voxel-Based Methods

To complement the shortcomings of Quantization+CNN and Point-based methods, some works combine them to balance accuracy and speed [78–80]. Yang et al. [79]

proposed a two-stage voxel-point-based 3D object detector. They discussed the problem where the classification score does not match the precision of location estimation, and proposed estimating IoU as an alternative to alleviate this problem. Another representative Point-Voxel-based method is HVPR [80], which is a one-stage detector and uses a memory module to augment point-based features, maintaining the efficiency of a single-stage method.

5.4.4 Summary

The performance of the methods mentioned above is compared in Table 5.5. The results are evaluated on the KITTI test set. It demonstrates that the 3D detection of small objects, like pedestrians and cyclists, is still an open problem. It is more challenging than vehicle detection because only few points are detected on these small objects, especially when they are far away. One possible solution is to utilize the help of other sensors like multi-LiDAR setups or LiDAR-camera setups. Besides, the research around Point-based 3D object detection is insufficient compared to quantization+CNN methods. It is a promising direction due to its excellent improvement in accuracy and there is substantial room for speed improvement.

5.5 RADAR-Based Methods

Similar to LiDARs, RADARs also provide spatial information on their perception fields. RADARs have already been widely used in automotive applications, such as collision avoidance and blind-spot detection, due to their robust performance in various weather and lighting conditions.

However, research of RADAR-based methods in 3D object detection is barely available. The reasons are: (1) the RADAR data is quite sparse due to the low resolution; (2) the automotive RADAR becomes unreliable at 10–15m when working in real-world scenes due to reflections from other objects; (3) the RADAR data is commonly in the 2D form, and height information is not available for 3D bounding box estimation. Due to the limited research work, we generally classified existing RADAR-based methods into *Spectrum-Image as Input* and *Point-Cloud as Input* methods.

Spectrum-Image as Input: The raw radar data contains the information of each location along a specific angle, which can be represented as a spectrum image. It is a dense representation similar to RGB images but contains spatial and reflection information. In the last decade, Bartsch et al. proposed a knowledge-based recognition system to detect pedestrians from the RADAR spectrum images [82]. They first segmented the spectrum images and handcrafted features from the patches. Then they classified the patches with a predefined empirical deterministic function. They claimed that it is hard to recognize pedestrians from solely RADAR data, especially

Table 5.5 Comparison between the performances of LiDAR-based methods based on the KITTI benchmark (test set)

Category	Method	Car(AP_{3D})			Ped(AP_{3D})			Cyc(AP_{3D})			Time(s)
		Easy	Mod	Hard	Easy	Mod	Hard	Easy	Mod	Hard	
Q+CNN	PointPillars [57]	79.0	74.9	68.3	52.0	43.5	41.4	75.7	59.0	52.9	0.15
Q+CNN	VoxelNet [23]	77.5	65.1	57.7	39.5	33.7	31.5	61.2	48.4	44.4	0.2
Q+CNN	SECOND [59]	83.1	73.6	66.2	51.0	42.5	37.2	70.5	53.8	46.9	0.05
Q+CNN	SA-SSD [61]	88.8	79.8	74.2	–	–	–	–	–	–	0.4
Q+CNN	CIA-SSD [62]	89.6	80.3	72.9	–	–	–	–	-	–	0.031
Q+CNN	BTCNet [70]	90.6	82.9	78.1	–	-	-	82.8	68.7	61.8	–
Q+CNN	VoxelRCNN [81]	90.9	81.6	77.1	–	—	–	–	–	–	0.04
Point	PointRCNN [75]	85.9	75.7	68.3	49.4	41.7	38.6	73.9	59.6	53.5	0.1
Point	IA-SSD [77]	88.3	80.1	75.0	46.5	39.0	35.6	78.4	61.9	55.7	0.013
Voxel-Point	STD [79]	88.0	79.7	75.1	53.3	42.5	38.4	78.7	61.6	55.3	0.08
Voxel-Point	HVPR [80]	86.4	77.9	73.0	53.5	44.0	40.6	–	–	–	0.028
Voxel-Point	PVRCNN [78]	90.3	81.4	76.8	52.2	43.3	40.3	78.6	63.7	57.7	0.08

in partial-vision or occluded conditions. In recent years, deep learning models have been used as powerful feature extractors and classifiers for perceptual tasks. Kanil et al. first generated proposals from the spectrum images and classified them with a CNN model [83]. Their work can be further improved if it is solved with the help of R-CNN framework [2] or one-stage detection framework [3].

Point-Cloud as Input: As mentioned in Sect. 5.2.2.3, point clouds can be obtained from RADAR raw data with clustering preprocessing. Scheiner et al. [84] proposed a straightforward solution that separates the input RADAR point clouds into multiple clusters with DBSCAN and further classifies them with LSTMs. Schumann et al. [85] went further in this direction and extracted features from static and dynamic RADAR points with CNNs and memory-aware PointNets, respectively. Danzer et al. proposed to estimate rotated 2D bounding boxes (no height) from RADAR point clouds [86]. They treated the RADAR data similarly to the LiDAR's and adopted the PoineNet architectures [71, 72] to extract features and regress 2D bounding boxes. Similar to LiDAR-based object detection in Sect. 5.4, RADAR point clouds can be quantified into 2D images and CNN can be adopted to extract features and detect objects [85, 87]. In [88], they proposed a RADAR-based detection benchmark and compared YOLOv3 [4], PointPillars [57], PointNet++ [72] as well as their variances. Their comparison shows YOLOv3 and PointNet++ perform better than PointPillars variances.

Section Summary: Compared to camera-based and LiDAR-based object detection, RADAR-based object detection is underexplored due to its sparse data observation. In Sect. 5.6, we will introduce some RADAR-based object detectors which utilize RADAR data to improve LiDAR/camera-based detector performance.

5.6 Multi-sensor-fusion Methods

Multi-sensor fusion is a popular research direction in 3D object detection, which provides not only complementary but also redundancy in the face of sabotage, failures, adverse conditions and blind spots [19]. However, different sensors provide different data representations from which features are extracted with different models. Therefore, one important problem for multi-sensor-fusion methods is: **How to fuse multi-sensor data for perception?**

5.6.1 Feature Fusion

The feature vectors extracted from different types of sensors can be fused using concatenation or addition (Fig. 5.8 top). Chen et al. [89] proposed a two-stage network, called MV3D, to combine the camera and LiDAR data. With a Quantization+CNN pipeline, they quantized the point cloud into a BEV image and a front-view image, and extracted features with CNNs from both. In the first stage, they generated 3D

proposals from the BEV image based on an RPN. In the second stage, they adopted multi-view ROI pooling to align the features from the BEV, front view, and RGB images. The features from different sensors were fused by concatenation and finally used to refine the proposal. A similar idea of fusing features on the image plane was also adopted by [11, 89] for camera-LiDAR fusion and adopted by [90] for camera-RADAR fusion.

In contrast to fusing pixel-wise features on the image plane, Sindaigi et al. [91] explored the performance of point-wise feature fusion in the 3D space and voxel-wise feature fusion in the grid space based on VoxelNet [23]. BEVFusion [50] uses two different pipelines to extract camera and LiDAR features, respectively. They introduce a BEV encoder to fuse the concatenated features. CenterFusion[92] proposed a frustum-based approach to complete representation fusion by projecting radar features onto the image plane. Objects in RADAR feature maps are associated by using information from the primary image bounding box. Then, they are projected into the image feature space and fused with the image feature. Decoder takes the extra velocity and depth information from the RADAR to generate prediction results.

5.6.2 Transformer Interaction Fusion

Vision transformer achieved good performance in 2D and 3D object detection in recent years. The architecture of transformer interaction fusion methods is shown in the middle row of Fig. 5.8. DeepFusion [93] uses the visual transformer to fuse features. PointPillars [57] is used as the feature extractor to obtain LiDAR features. The camera image serves as input to a 2D image feature extractor (such as ResNet) to obtain camera features. Then, the image feature is used as the key and value of the transformer decoder, and LiDAR feature is used to generate the query. The output of the decoder is concatenated with the LiDAR feature to complete feature fusion. Finally, it uses the voxel-based detection head to process the feature fusion to output the detection results.

LiDAR is hard to deal with object which is tiny and far away because of its low resolution compared to the camera. To address this problem, TransFusion [94] uses a dual transformer decoder. The first layer of the decoder generates initial prediction results by using the sparse 3D feature of point cloud queries. The query of the second layer fuses the initial result of the first layer to predict the final result on the image feature. Similar to TransFusion, DeepInteraction [95] introduces transformer encoders based on different modalities. In particular, the multi-modal predictive interaction layers take the image or the LiDAR representation as the input, the odd layer takes the camera representation, and even for LiDAR. Each layer boosts the perception strength of the corresponding modality.

A benefit of the unified representation is that there is no need to design a block for transforming feature space. MVDNet [96] can directly generate 2D proposals for both RADAR and LiDAR. The representations of the two modalities are then fused using the cross attention module. Transforming different modalities into a unified

representation space is also a solution. UVTR [97] unifies feature representation in voxel space. They exploit image features to generate voxel-space representations of image features through simple convolutional layers. The modalities are then fused using knowledge transfer learning. Finally, transformer decoder is used to predict the result.

5.6.3 Cascade Pipeline

Cascade pipeline means that the results of one detector act as the inputs to the next detector, as shown in the bottom part of Fig. 5.8. Qi *et al.* [24] proposed Frustum-PointNet to fuse camera and LiDAR data for 3D object detection. They first generated 2D bounding boxes from only RGB images with an off-the-shelf image-based object detector. The results were projected into 3D space as frustums, the points inside which were used to refine 3D bounding boxes by PointNet architectures [71, 72] in the second stage. Some researches focus on camera-LiDAR fusion algorithms mainly using enhancement point cloud to do detection tasks using the image feature information. PointPainting and CenterPointV2 [60, 98] are classical examples of such algorithms. They all fuse the segmentation information to the LiDAR points. They use semantic segmentation to get objects. According to the segmentation results, the corresponding point clouds are classified, and a point cloud-based 3D object detection framework is applied to predict the results. For the latter, they use voxel-based method to process the LiDAR points. In the cascade pipeline, the features of different detectors can also be fused by concatenation or addition [99].

Even though these cascade methods achieve state-of-the-art performances on the KITTI benchmark [20], they suffer from the following limitations: 1) The accuracy is upper bounded by the early-level detectors. The false negatives will never be recovered if they are missed in the early stages. 2) If the early stage is an image-based detector, the detections will be vulnerable to appearance changes even if LiDAR data is included. 3) The long detection range of cameras is limited by LiDARs if there is no point detected by the LiDAR in the camera-detected region.

5.6.4 Section Summary

The methods mentioned above are evaluated on the KITTI test dataset and demonstrated in Table 5.6. One common drawback of the fusion methods above is that the fusion pipelines are fixed for a specific setup ($1\times$ camera + $1\times$ LiDAR or $1\times$ camera $+1\times$ RADAR). They can hardly be adapted to other settings. Due to various sensor setups on autonomous vehicles, one promising direction is to design a general multi-sensor fusion framework for 3D object detection so that it can flexibly scale up or down according to the sensor settings. It should be more accurate when more sensors are considered.

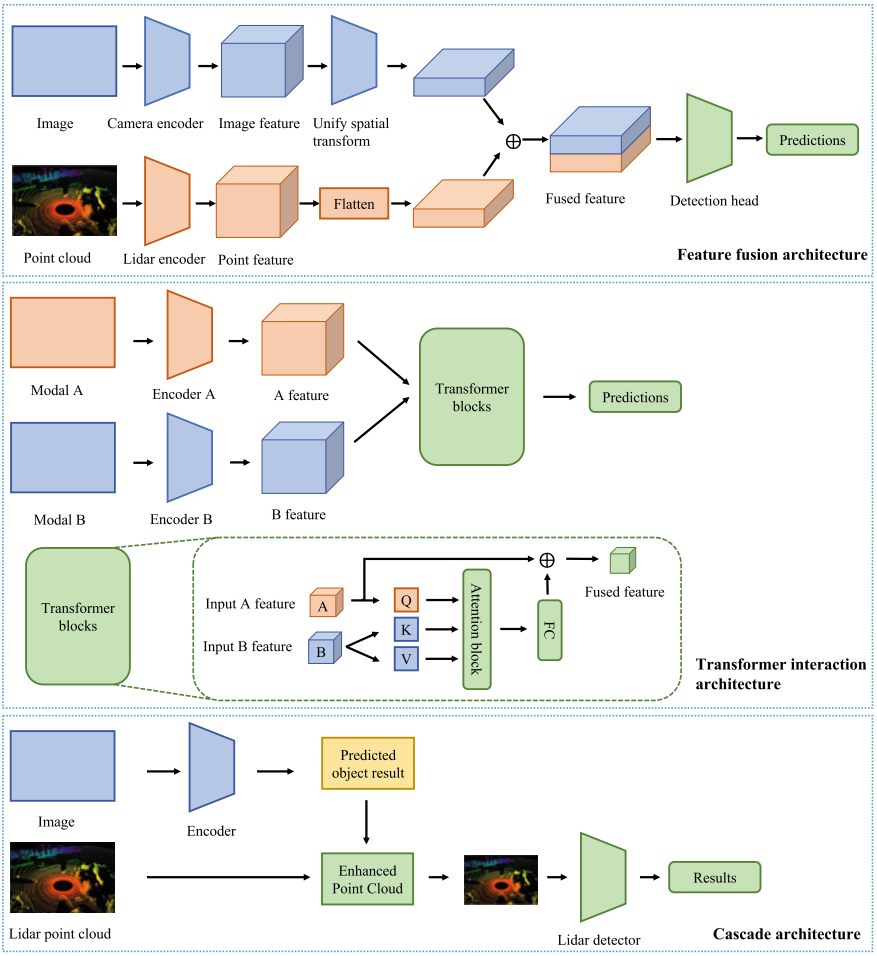

Fig. 5.8 Architectures of different multi-sensor-fusion-based methods. The top, middle, and bottom rows denote feature-fusion, transformer-based, and cascade architectures

5.7 Promising Directions

Besides continuously improving the accuracy and real-time performance, we have already introduced two open problems: RADAR-based 3D object detection methods in Sect. 5.5 and a flexible framework for multi-sensor-fusion 3D object detection in Sect. 5.6. In this section, we will introduce more interesting open problems in 3D object detection.

Table 5.6 Comparison between the performances of multi-sensor fusion methods. C, R, and L are short for camera, RADAR, and LiDAR, respectively

Solution	Implementation	Modality	KITTI ($AP_{Car,3D}$)			nuScenes		Time(s)
			Easy	Mod	Hard	NDS	mAP	
Feat.	MV3D [89]	C + L	71.2	62.6	56.5	–	–	0.4
Feat.	AVOD [11]	C + L	81.9	71.8	66.3	–	–	0.1
Feat.	MVX-Net [91]	C + L	83.2	72.7	65.2	–	-	0.06
Feat.	BEVFusion [50]	C + L	–	–	–	72.9	70.2	0.119
Feat.	CenterFusion [92]	R + L	–	–	-	44.9	32.6	–
Tran.	TransFusion [94]	C + L	–	–	–	71.7	68.9	0.266
Tran.	DeepInteraction [95]	C + L	–	–	–	76.3	75.6	0.204
Tran.	UVTR [97]	C + L	–	–	–	71.1	67.1	–
Cascade	F-PointNet [24]	C + L	81.2	70.4	62.2	–	–	0.2
Cascade	PointPainting [98]	C + L	92.4	88.1	83.3	46.4	33.9	0.016
Cascade	CenterPoint [60]	C + L	–	–	–	65.5	58.0	0.089

5.7.1 Extreme Conditions

The edge cases caused by extreme conditions are one of the major challenges in autonomous driving. As we claimed in Sect. 5.2.2, cameras and LiDARs are not robust to bad lighting conditions or extreme weather. The different appearance information in extreme weather and bad lighting conditions may cause a different data distribution from the training dataset for camera-based methods. Extreme weather, like rainy or snowy days, may worsen the quality of the point clouds of LiDAR scans.

One possible solution is to consider the data in extreme conditions when collecting the training data so that the data distribution will be similar when the 3D object detector is deployed in the real world. It can improve the performance of networks, but it is not the best option, since it is hard to enumerate all the extreme conditions in the real world. The unexpected conditions will put the passengers in a dangerous situation. Another solution is to adopt the multi-sensor setups since different sensor types have different failure modes during different conditions [12]. Especially, RADARs are famous for their robustness towards extreme conditions. Current multi-sensor fusion methods focus more on accuracy improvement than robustness improvement in extreme conditions, which needs more effort in the future. In addition, generative adversarial networks (GANs) [100], which have shown powerful ability, especially in domain adaptation in recent years, can be used to alleviate the difference between extreme conditions and normal conditions. For instance, the night-time image can be translated by GANs to daytime for 3D object detection. Similar works have gained attention in some visual tasks like place recognition [101, 102].

5.7.2 Uncertainty

Uncertainty is a natural part of any perception system's output. Knowing the confidence with which we can trust the 3D object detection output is crucial for decision making. To model the uncertainty of a perception output, a natural idea is to use the softmax output as the uncertainty. The higher output represents lower uncertainty.

Recently, some researchers modeled the perceptual uncertainty of a deep learning network output in a more detailed manner [103–105]. They classified the uncertainty into epistemic uncertainty which is caused by insufficient data, and aleatoric uncertainty which is caused by noisy annotation and model properties. In their work, they claimed that not only uncertainty was well modelled but also the accuracy of semantic segmentation networks was improved due to their consideration of uncertainty.

5.7.3 Summary

According to our discussion in this chapter, 3D object detection contains the following open problems:

How to improve the accuracy of 3D object detection, especially the objects in small sizes? For LiDAR- and RADAR-based methods, the cause of the bad accuracy when detecting small objects is due to the fact that there are few points detected by the perceptual sensors. One possible solution is to adopt more than one perceptual sensor, which provides ont only redundancy but also robustness. It could be the fusion of multiple LiDARs or RADARs, which naturally improve the number of points detected on the small objects. Besides, it could also be the fusion of different types of sensors, as we mentioned before, which also provides complimentary information in extreme cases.

How to implement 3D object detection based on RADAR data? As we mentioned in Sect. 5.5, current research of RADAR-related 3D object detection can be classified as point-cloud-based methods and spectral-image-based methods. However, all of them cannot estimate the properties of objects in the z-axis. It is limited by the sensor property, for instance, most of the current RADARs are 2D sensors. In the future, RADAR systems should be improved to provide 3D information as well.

How to design a flexible framework for multi-sensor fusion? As we mentioned in Sect. 5.6, current multi-sensor fusion methods are fixed and pre-designed for a specific sensor setup. The generalization to another setup is costly. A flexible framework of multi-sensor fusion 3D object detection is a promising research direction. One possible solution could be that all the sensors or groups of sensors can independently detect objects, and their results are fused by optimizing an objective function. More sensors provide more constraints and hence the results should be more accurate.

How to measure the robustness of 3D object detectors against extreme conditions? How to improve their robustness? A direct solution is to measure the difference in performance under common conditions and extreme conditions, like

$AP_{extreme}/AP_{common}$, where $AP_{extreme}$ and AP_{common} denote the average precision in the extreme and the common conditions. When the number of evaluation data is large enough, such a metric can measure the robustness of the algorithm in extreme conditions. One of the significant reasons for the bad performance under extreme conditions is sensor limitation. Therefore, the multi-sensor fusion methods could be a promising solution due to the availability of complementary information under different conditions.

How to model the uncertainty of 3D object detection? How to utilize the uncertain samples to enhance the performance of the detectors? As we mentioned in Sect. 5.7.2, the epistemic and aleatoric of [103, 104] can be extended to 3D object detection for measuring the uncertainty. One possible way to utilize the uncertain samples could include providing more weights to the uncertain samples in the training phase by adding a dynamic term in the loss function as demonstrated in [64, 106].

5.8 Conclusion

In this chapter, we reviewed the state-of-the-art 3D object detection methods in autonomous driving. We introduced the properties of three perceptual sensors in autonomous driving: cameras, LiDARs and RADARs, and analyzed their pros and cons for 3D object detection.

We reviewed the state-of-the-art 3D object detection methods in autonomous driving from the aspects of loss functions and architectures, and categorized them into camera-based, LiDAR-based, RADAR-based, and multi-sensor fusion methods. We introduced the problem definition, assumption, and evaluation metrics of 3D object detection in autonomous driving. The representative datasets were discussed, and their limitations were analyzed for future research. We pointed out the main problems and the existing solutions for each method. By analyzing the limitations of current solutions, we proposed some promising research directions and open problems. Quantitative results from the KITTI benchmark showed that 3D object detection is worth more effort, especially in small-size objects. Our analysis around autonomous driving system requirements showed that the robustness against extreme conditions, flexible multi-sensor fusion framework, and uncertainties in a perceptual system were all open problems in this field.

References

1. He K et al (2017) Mask R-CNN. In: Proceedings of the IEEE international conference on computer vision (ICCV), pp 2961–2969
2. Girshick R (2015) Fast R-CNN. In: IEEE international conference on computer vision (ICCV), pp 1440–1448
3. Liu W et al (2016) SSD: single shot multibox detector. In: European conference on computer vision (ECCV). Springer, pp 21–37

4. Redmon J et al (2018) YOLOv3: an incremental improvement. Computing research repository (CoRR). https://arxiv.org/abs/1804.02767
5. Zhang C, et al (2018) Robust LIDAR localization for autonomous driving in rain. In: IEEE/RSJ international conference on intelligent robots and systems (IROS), pp 3409–3415
6. Arnold E et al (2019) A survey on 3D object detection methods for autonomous driving applications. IEEE Trans Intell Transp Syst (TITS) 3782–3795
7. Guo Y et al (2020) Deep learning for 3D point clouds: a survey. IEEE Trans Pattern Anal Mach Intell (TPAMI) 43(12):4338–4364
8. Alaba SY et al (2022) A survey on deep-learning-based LiDAR 3D object detection for autonomous driving. Sensors 22(24):9577. https://www.mdpi.com/1424-8220/22/24/9577
9. Chen X et al (2018) 3D object proposals using stereo imagery for accurate object class detection. IEEE Trans Pattern Anal Mach Intell (TPAMI) 40(5):1259–1272
10. Xiaozhi C et al (2016) Monocular 3D object detection for autonomous driving. In: IEEE conference on computer vision and pattern recognition (CVPR), pp 2147–2156
11. Ku J et al (2018) Joint 3D proposal generation and object detection from view aggregation. In: IEEE/RSJ international conference on intelligent robots and systems (IROS), pp 1–8
12. Caesar H et al (2020) nuscenes: a multimodal dataset for autonomous driving. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 11 621–11 631
13. Fan R et al (2017) Real-time implementation of stereo vision based on optimised normalised cross-correlation and propagated search range on a GPU. In: IEEE international conference on imaging systems and techniques (IST), pp 1–6
14. Chang J-R et al (2018) Pyramid stereo matching network. In: IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 5410–5418
15. Mayer N et al (2016) A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: IEEE conference on computer vision and pattern recognition (CVPR), pp 4040–4048
16. Xiang Y et al (2015) Data-driven 3D voxel patterns for object category recognition. In: IEEE conference on computer vision and pattern recognition (CVPR), pp 1903–1911
17. Li P et al (2019) Stereo r-CNN based 3d object detection for autonomous driving. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 7644–7652
18. Geronimo D et al (2010) Survey of pedestrian detection for advanced driver assistance systems. IEEE Trans Pattern Anal Mach Intell (TPAMI) 32(7):1239–1258
19. Kim J et al (2018) Robust camera lidar sensor fusion via deep gated information fusion network. In: IEEE intelligent vehicles symposium (IV), pp 1620–1625
20. Geiger A et al (2012) Are we ready for autonomous driving? The KITTI vision benchmark suite. In: IEEE conference on computer vision and pattern recognition (CVPR), pp 3354–3361
21. Sun P et al (2020) Scalability in perception for autonomous driving: waymo open dataset. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 2446–2454
22. Simon M et al (2018) Complex-YOLO: an Euler-region-proposal for real-time 3D object detection on point clouds. In: European conference on computer vision (ECCV). Springer, pp 197–200
23. Zhou Y et al (2018) VoxelNet: end-to-end learning for point cloud based 3D object detection. In: IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 4490–4499
24. Qi CR et al (2018) Frustum point nets for 3D object detection from RGB-D data. In: IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 918–927
25. Liang M et al (2018) Deep continuous fusion for multi-sensor 3D object detection. In: Proceedings of the European conference on computer vision (ECCV), pp 641–656
26. Redondo-Cabrera CO (2016) Pose estimation errors, the ultimate diagnosis. In: European conference on computer vision (ECCV). Springer, pp 118–134
27. Mousavian A et al (2017) 3D bounding box estimation using deep learning and geometry. In: IEEE conference on computer vision and pattern recognition (CVPR), pp 5632–5640

28. Liu Y et al (2021) YOLOStereo3D: a step back to 2D for efficient stereo 3D detection. In: International conference on robotics and automation (ICRA). IEEE, pp 13 018–13 024
29. Chen Y et al (2020) MonoPair: monocular 3D object detection using pairwise spatial relationships. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 12 093–12 102
30. Vaswani A et al (2017) Attention is all you need. In: Advances in neural information processing systems (NeurIPS), vol 30
31. Carion N et al (2020) End-to-end object detection with transformers. In: European conference on computer vision (ECCV). Springer, pp 213–229
32. Huang K-C et al (2022) MonoDTR: monocular 3D object detection with depth-aware transformer. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 4012–4021
33. Wang L et al (2021) Depth-conditioned dynamic message propagation for monocular 3D object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 454–463
34. Park D et al (2021) Is pseudo-lidar needed for monocular 3D object detection? In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV), pp 3142–3152
35. Wang Y et al (2018) Pseudo-LiDAR from visual depth estimation: bridging the gap in 3D object detection for autonomous driving. Computing research repository (CoRR), vol abs/1812.07179. https://arxiv.org/abs/1812.07179
36. Li P et al (2021) Monocular 3D detection with geometric constraints embedding and semi-supervised training. IEEE Robot Autom Lett (RAL) 6(3):5565–5572
37. Zhang Y et al (2021) Objects are different: flexible monocular 3D object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 3289–3298
38. Brazil G et al (2019) M3D-RPN: monocular 3D region proposal network for object detection. In: Proceedings of the IEEE international conference on computer vision (ICCV), pp 9287–9296
39. Liu Y et al (2021) Ground-aware monocular 3D object detection for autonomous driving. IEEE Robot Autom Lett (RAL), pp 919–926
40. Lu Y et al (2021) Geometry uncertainty projection network for monocular 3D object detection. In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV), pp 3111–3121
41. You Y et al (2019) Pseudo-LiDAR++: accurate depth for 3D object detection in autonomous driving. Computing research repository (CoRR). https://arxiv.org/abs/1906.06310
42. Vianney JMU et al (2019) RefinedMPL: refined monocular PseudoLiDAR for 3D object detection in autonomous driving. Computing research repository (CoRR). https://arxiv.org/abs/1911.09712
43. Qian R et al (2020) End-to-end pseudo-LiDAR for image-based 3D object detection. In: Conference on computer vision and pattern recognition (CVPR), pp 5881–5890
44. Li C et al (2020) Confidence guided stereo 3D object detection with split depth estimation. In: IEEE/RSJ international conference on intelligent robots and systems (IROS), pp 5776–5783
45. Philion J et al (2020) Lift, splat, shoot: encoding images from arbitrary camera rigs by implicitly unprojecting to 3D. In: Proceedings of the European conference on computer vision (ECCV). Springer, pp 194–210
46. Chen Y et al (2022) DSGN++: exploiting visual-spatial relation for stereo-based 3D detectors. IEEE Trans Pattern Anal Mach Intell (TPAMI) 1–14
47. Chen Y et al (2020) DSGN: deep stereo geometry network for 3D object detection. In: Conference on computer vision and pattern recognition (CVPR), pp 12 536–12 545
48. Guo X et al (2021) LIGA-Stereo: learning LiDAR geometry aware representations for stereo-based 3D detector. In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV), pp 3153–3163
49. Reading C et al (2021) Categorical depth distribution network for monocular 3D object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 8555–8564

50. Liu Z et al (2022) BEVFusion: multi-task multi-sensor fusion with unified bird's-eye view representation. Computing research repository (CoRR). https://arxiv.org/abs/2205.13542
51. Liu Y et al (2022) Petr: position embedding transformation for multi-view 3d object detection. In: Proceedings of the European Conference on Computer Vision (ECCV). Springer, pp 531–548
52. Li Z et al (2022) Bevformer: learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In: Proceedings of the European conference on computer vision (ECCV). Springer, pp 1–18
53. Xia Z et al (2022) Vision transformer with deformable attention. Computing research repository (CoRR). https://arxiv.org/abs/2201.00520
54. Ma X et al (2019) Accurate monocular 3D object detection via color-embedded 3D reconstruction for autonomous driving. In: IEEE/CVF international conference on computer vision (ICCV), pp 6850–6859
55. Beltrán J et al (2018) BirdNet: a 3D object detection framework from LiDAR information. In: International conference on intelligent transportation systems (ITSC), pp 3517–3523
56. Yang B et al (2018) PIXOR: real-time 3D object detection from point clouds. In: IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 7652–7660
57. Lang AH et al (2019) Pointpillars: fast encoders for object detection from point clouds. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 12 697–12 705
58. Li B (2017) 3D fully convolutional network for vehicle detection in point cloud. In: IEEE/RSJ international conference on intelligent robots and systems (IROS), pp 1513–1518
59. Yan Y et al (2018) SECOND: sparsely embedded convolutional detection. Sensors 18(10):3337
60. Yin T et al (2021) Center-based 3D object detection and tracking. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 11 784–11 793
61. He C et al (2020) Structure aware single-stage 3D object detection from point cloud. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 11 873–11 882
62. Wu Z et al (2021) CIA-SSD: confident IoU-aware single-stage object detector from point cloud. Proc AAAI Conf Artif Intell (AAAI) 35(4):3555–3562
63. Ye D et al (2022) LidarMutliNet: unifying LiDAR semantic segmentation, 3D object detection, and panoptic segmentation in a single multi-task network. Computing research repository (CoRR). https://arxiv.org/abs/2206.11428
64. Lin T-Y et al (2017) Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision (ICCV), pp 2980–2988
65. Graham B et al (2017) Submanifold sparse convolutional networks. Computing research repository (CoRR). https://arxiv.org/abs/1706.01307
66. Zhou X et al (2020) Tracking objects as points. In: European conference on computer vision (ECCV). Springer, pp 474–490
67. Zhou X et al (2019) Objects as points. Computing research repository (CoRR). https://arxiv.org/abs/1904.07850
68. Teichmann M et al (2018) MultiNet: real-time joint semantic reasoning for autonomous driving. In: IEEE intelligent vehicles symposium (IV). IEEE, pp. 1013–1020
69. Gkioxari G et al (2019) Mesh R-CNN. In: Proceedings of the IEEE/CVF international conference on computer vision (CVPR), pp 9785–9795
70. Xu Q et al (2022) Behind the curtain: learning occluded shapes for 3D object detection. Proc AAAI Conf Artif Intell (AAAI) 36(3):2893–2901
71. Qi CR et al (2017) PointNet: deep learning on point sets for 3D classification and segmentation. In: IEEE conference on computer vision and pattern recognition (CVPR), pp 77–85
72. Qi C et al (2017) PointNet++: deep hierarchical feature learning on point sets in a metric space. In: Advances in neural information processing systems (NeurIPS), pp 5099–5108
73. Li J et al (2018) SO-Net: self-organizing network for point cloud analysis. In: IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 9397–9406

74. Wang Y et al (2019) Dynamic graph CNN for learning on point clouds. ACM Trans Graph (TOG) 38(5):1–12
75. Shi S et al (2019) PointRCNN: 3D object proposal generation and detection from point cloud. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 770–779
76. Qi CR et al (2019) Deep hough voting for 3d object detection in point clouds. In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV), pp 9277–9286
77. Zhang Y et al (2022) Not all points are equal: learning highly efficient point-based detectors for 3D LiDAR point clouds. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 18 953–18 962
78. Shi S et al (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 (CVPR), pp 10 529–10 538
79. Yang Z et al (2019) STD: sparse-to-dense 3D object detector for point cloud. In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV), pp 1951–1960
80. Noh J et al (2021) HVPR: hybrid voxel-point representation for single-stage 3D object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 14 605–14 614
81. Deng J et al (2021) Voxel R-CNN: towards high performance voxel-based 3D object detection. Proc AAAI Conf Artif Intell (AAAI) 35(2):1201–1209
82. Bartsch A et al (2012) Pedestrian recognition using automotive radar sensors. Adv Radio Sci **10**(B.2), 45–55
83. Patel K et al (2019) Deep learning-based object classification on automotive radar spectra. In: IEEE radar conference (RadarConf), pp 1–6
84. Scheiner N et al (2020) Off-the-shelf sensor vs. experimental radar How much resolution is necessary in automotive radar classification?. In: International conference on information fusion (FUSION), pp 1–8
85. Schumann O et al (2019) Scene Understanding With Automotive Radar. IEEE Trans Intell Veh (TIV) 5(2):188–203
86. Danzer A et al (2019) 2d car detection in radar data with pointnets. In: IEEE intelligent transportation systems conference (ITSC). IEEE, pp 61–66
87. Dreher M et al (2020) Radar-based 2D car detection using deep neural networks. In: International conference on intelligent transportation systems (ITSC). IEEE, pp 1–8
88. Scheiner N et al (2021) Object detection for automotive radar point clouds - a comparison. AI Perspect 3(1):1–23
89. Chen X et al (2017) Multi-view 3D object detection network for autonomous driving. In: IEEE conference on computer vision and pattern recognition (CVPR), pp 6526–6534
90. Zhang G et al (2019) Object detection and 3D estimation via an FMCW radar using a fully convolutional network. Computing research repository (CoRR). https://arxiv.org/abs/1902.05394
91. Sindagi VA et al (2019) Mvx-net: multimodal voxelnet for 3d object detection. In: International conference on robotics and automation (ICRA). IEEE, pp 7276–7282
92. Nabati R et al (2021) Center fusion: center-based radar and camera fusion for 3D object detection. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision (WACV), pp 1527–1536
93. Li Y et al (2022) DeepFusion: lidar-camera deep fusion for multi-modal 3D object detection. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp 17 182–17 191
94. Bai X et al (2022) TransFusion: robust LiDAR-camera fusion for 3D object detection with transformers. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 1090–1099
95. Yang Z et al (2022) DeepInteraction: 3D object detection via modality interaction. Computing research repository (CoRR). https://arxiv.org/abs/2208.11112

96. Qian K et al (2021) Robust multimodal vehicle detection in foggy weather using complementary lidar and radar signals. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 444–453
97. Li Y et al (2022) Unifying voxel-based representation with transformer for 3d object detection. In: Advances in neural information processing systems (NeurIPS). https://openreview.net/forum?id=XA4ru9mfxTP
98. Xu S et al (2021) Fusion painting: multimodal fusion with adaptive attention for 3D object detection. In: IEEE international intelligent transportation systems conference (ITSC). IEEE, pp 3047–3054
99. Xu D et al (2018) Point fusion: deep sensor fusion for 3D bounding box estimation. In: IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 244–253
100. Goodfellow I et al (2014) Generative adversarial networks. In: Neural Information processing systems (NeurIPS), pp 2672–2680
101. Porav H et al (2018) Adversarial training for adverse conditions: robust metric localisation using appearance transfer. In: IEEE international conference on robotics and automation (ICRA), pp 1011–1018
102. Latif Y et al (2018) Addressing challenging place recognition tasks using generative adversarial networks. In: IEEE international conference on robotics and automation (ICRA), pp 2349–2355
103. Kendall A et al (2017) What uncertainties do we need in bayesian deep learning for computer vision? In: Advances in neural information processing systems (NeurIPS), pp 5574–5584
104. Kendall A et al (2018) Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 7482–7491
105. Yun P et al (2023) Laplace approximation based epistemic uncertainty estimation in 3D object detection. In: Conference on robot learning (CoRL). PMLR, pp 1125–1135
106. Yun P et al (2019) Focal Loss in 3D Object Detection. IEEE Robot Autom Lett (RAL) 4(2):1263–1270