

# 3D Object Detection for Autonomous Driving: A Survey

Rui Qian<sup>a</sup>, Xin Lai<sup>b</sup> and Xirong Li<sup>a,\*</sup>

<sup>a</sup>Key Lab of Data Engineering and Knowledge Engineering, Renmin University of China, Beijing 100872, China.

<sup>b</sup>School of Mathematics, Renmin University of China, Beijing 100872, China

## ARTICLE INFO

### Keywords:

3D object detection  
Point clouds  
Autonomous driving

## ABSTRACT

Autonomous driving is regarded as one of the most promising remedies to shield human beings from severe crashes. To this end, 3D object detection serves as the core basis of perception stack especially for the sake of path planning, motion prediction, and collision avoidance *etc.* Taking a quick glance at the progress we have made, we attribute challenges to visual appearance recovery in the absence of depth information from images, representation learning from partially occluded unstructured point clouds, and semantic alignments over heterogeneous features from cross modalities. Despite existing efforts, 3D object detection for autonomous driving is still in its infancy. Recently, a large body of literature have been investigated to address this 3D vision task. Nevertheless, few investigations have looked into collecting and structuring this growing knowledge. We therefore aim to fill this gap in a comprehensive survey, encompassing all the main concerns including sensors, datasets, performance metrics and the recent state-of-the-art detection methods, together with their pros and cons. Furthermore, we provide quantitative comparisons with the state of the art. A case study on fifteen selected representative methods is presented, involved with runtime analysis, error analysis, and robustness analysis. Finally, we provide concluding remarks after an in-depth analysis of the surveyed works and identify promising directions for future work.

© 2022 Elsevier Ltd. All rights reserved.

## 1. INTRODUCTION

Dream sheds light on reality. It is a dream that autonomous vehicles hit the roads legally, functioning wisely as good as human drivers or even better, responding timely to various unconstrained driving scenarios, and being fully free of the control of human drivers, a.k.a. Level 5 with “driver off” in Fig. 1. Let the dream be realized, thousands of new employment opportunities shall be created for those physically impaired (*Mobility*), millions of lives shall be rescued from motor vehicle-related crashes (*Safety*), and billions of dollars shall be saved from disentangling traffic accidents and treating the wounded (*Economics*). It is a reality that there is still no universal consensus on where we are now and how we shall go next. As illustrated in Fig. 1, We are largely above level 2 but under or infinitely close to level 3 by taking into account the following three social concerns: (1) Safety and Security. Rules and regulations are still blank, which shall be developed by governments to guarantee the safety for an entire trip. (2) Law and Liability. How to define the major responsibility and who will take that responsibility shall be identified both ethically and clearly. (3) Acceptance. Long-term efforts shall be made to establish the confidence and trust for the whole society, before autonomous driving can be finally accepted. This survey paper will take a structured glance at 3D object detection, one of the core techniques for autonomous driving.

Perception in 3D space is a prerequisite in autonomous driving. A fully understanding of what is happening right

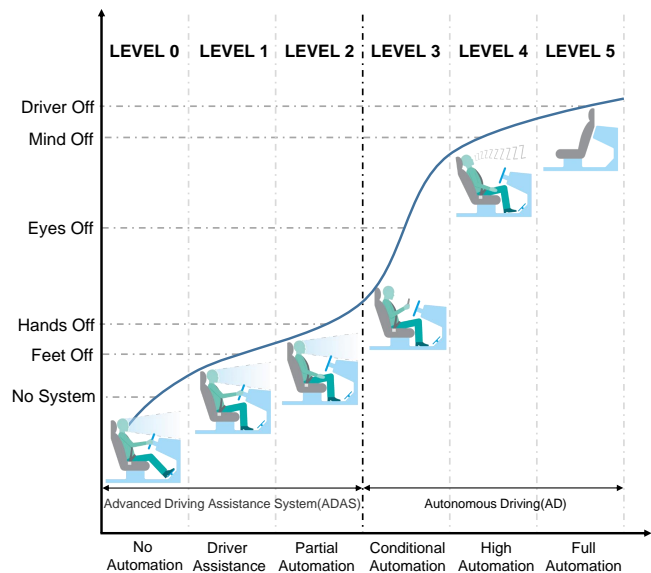


Figure 1: Levels of autonomous driving proposed by SAE (Society of Automotive Engineers) International [1]. Where are we now?

now in front of the vehicle will facilitate downstream components to react accordingly, which is exactly what 3D object detection aims for. 3D object detection perceives and describes what surrounds us via assigning a label, how its shape looks like via drawing a bounding box, and how far away it is from an ego vehicle via giving a coordinate. Besides, 3D detection even provides a heading angle that indicates orientation. It is these upstream information from perception stack that enables downstream planning model to make decisions.

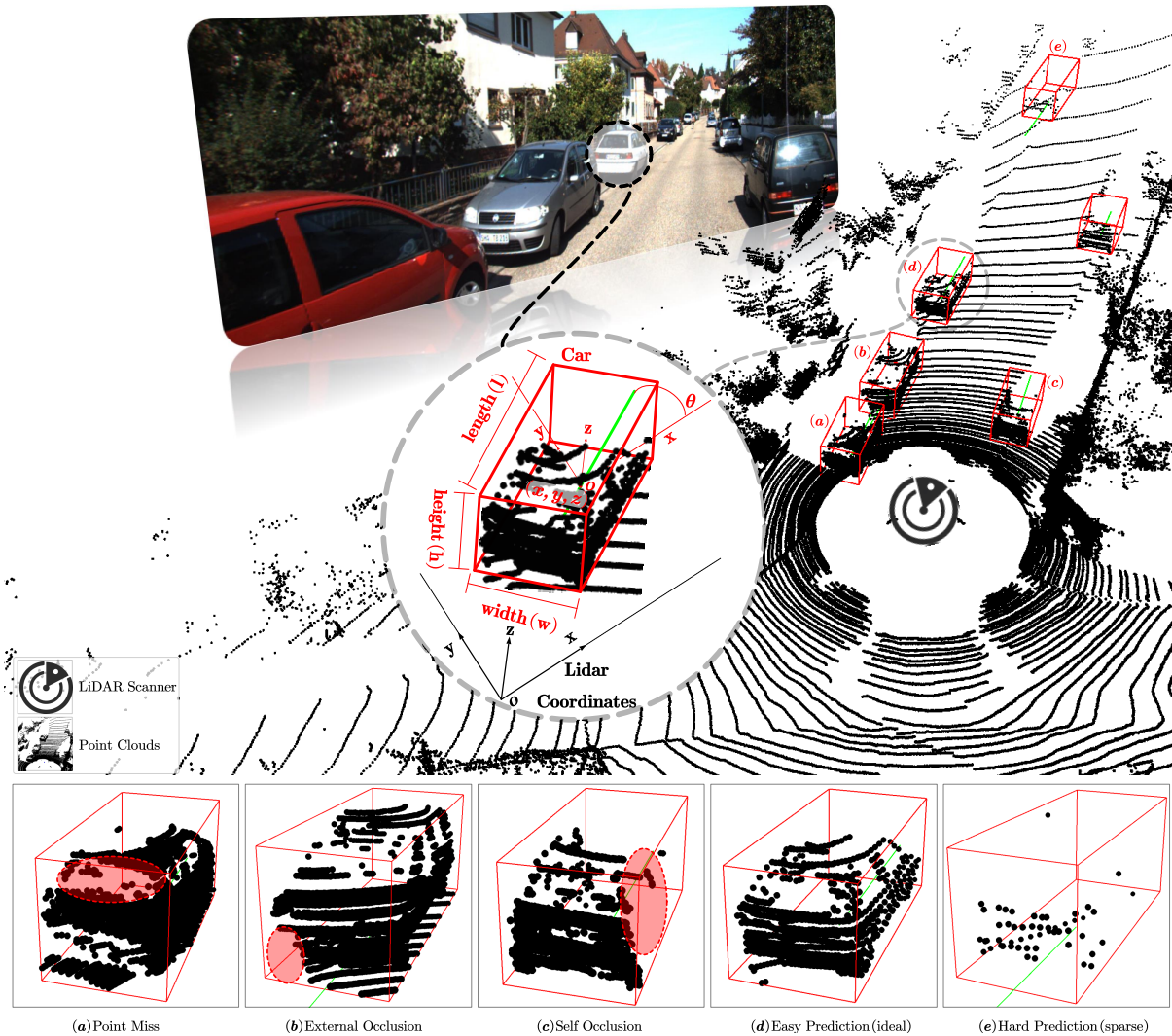
\*Corresponding author

[rui-qian@ruc.edu.cn](mailto:rui-qian@ruc.edu.cn) (R. Qian); [laixin@ruc.edu.cn](mailto:laixin@ruc.edu.cn) (X. Lai);

[xirong@ruc.edu.cn](mailto:xirong@ruc.edu.cn) (X. Li)

<https://doi.org/10.1016/j.patcog.2022.108796>

© 2022 Elsevier Ltd. All rights reserved.



**Figure 2: An overview of 3D object detection task from images and point clouds.** Typical challenges: (a) Point Miss. When LiDAR signals fail to return back from the surface of objects. (b) External Occlusion. When LiDAR signals are blocked by occluders in the vicinity. (c) Self Occlusion. When one near side of the object blocks the other, which makes point clouds 2.5D in practice. Note that bounding box prediction in (d) is much easier than that in (e) due to the sparsity of point clouds at long ranges.

### 1.1. Tasks and Challenges

Fig. 2 presents an overview of 3D object detection task from images and point clouds. The whole goal of 3D object detection is to recognize the objects of interest by drawing an oriented 3D bounding box and assigning a label. Consider two commonly used 3D object detection modalities, *i.e.* images and point clouds, the key challenges of this vision task are strongly tied to the way we use, the way we represent, and the way we combine. With only images on hand, the core challenge arises from the absence of depth information. Although much progress has been made to recover depth from images [2, 3], it is still an ill-posed inverse problem. The same object in different 3D poses can result in different visual appearance in the image plane, which is not conducive to the learning of representation [4]. Besides, given that camera is passive sensor (see Sec. 2.2.1), images are naturally vulnerable to illumination (*e.g.*, nighttime) or rainy weather conditions. With only point clouds on hand,

the key difficulty stems from the representation learning. Point clouds are sparse by nature, *e.g.* in works [5, 6], non-empty voxels normally account for approximately 1%, 3% in a typical range setup on Waymo Open [7] dataset and KITTI [8] dataset respectively. Point clouds are irregular and unordered by nature. Directly applying convolution operator to point clouds will incur “desertion of shape information and variance to point ordering” [9]. Besides, point clouds are 2.5D in practice as they are point miss in (a), external-occlusion in (b), self-occlusion in (c) [10], as indicated in Fig. 2. Without the umbrella of convolutional neural networks, one way is to present point clouds as voxels. The dilemma is that scaling up voxel size will loss resolution and consequently degrade localization accuracy while scaling down its size will cubically increase the complexity and memory footprints as the input resolution grows. Another way is to present point clouds as point sets. Nevertheless, around 80% of runtime is occupied by point retrieval, say,

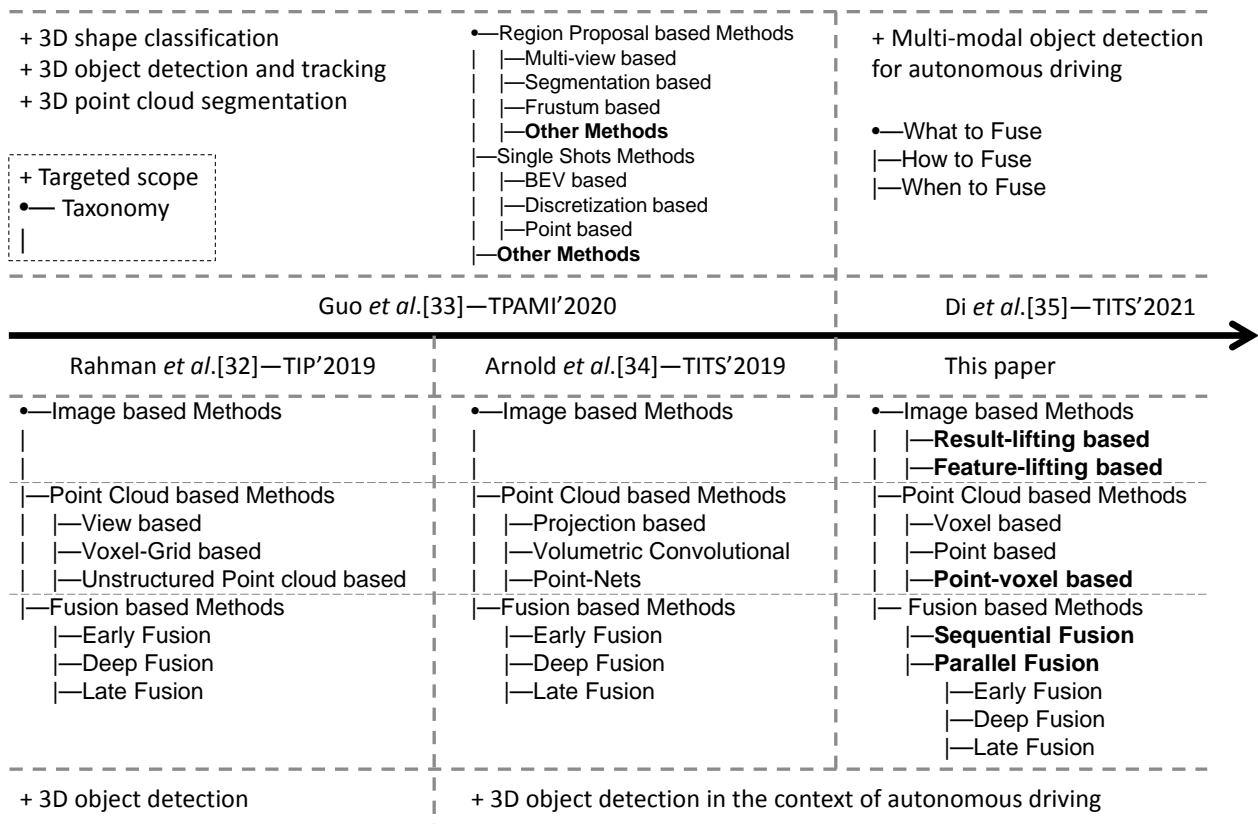


Figure 3: A summary showing how this survey differs from existing ones on 3D object detection. Vertically, targeted scope concisely determines where the boundary is located among their investigations. Horizontally, hierarchical branches of this paper reveal a good continuity of existing efforts [13, 14] while adapt new branches (indicated in bold font) for dynamics, which importantly contributes to the maturity of the taxonomy on 3D object detection.

ball query operation, in light of the poor memory locality [11]. With both images and point clouds on hand, the tricky obstacle often derives from semantic alignments. Images and point clouds are two heterogeneous media, presented in camera view and real 3D view, finding point-wise correspondences between LiDAR points and image pixels results in “feature blurring” [12].

### 1.2. Targeted Scope, Aims, and Organization

We review literature that are closely related to 3D object detection in the context of autonomous driving. Depending on what modality we use, existing efforts are divided into the following three subdivisions: (1) image based [15, 16, 17, 18, 19, 20, 21, 22], which is relatively inaccurate but several orders of magnitude cheaper, and more interpretable under the guidance of domain expertise and knowledge priors. (2) point cloud based [23, 24, 25, 10, 6, 26, 27, 28, 29, 5], which has a relatively higher accuracy and lower latency but more prohibitive deployment cost compared with its image based counterparts. (3) multimodal fusion based [12, 30, 31, 32, 33, 34], which currently lags behind its point cloud based counterparts but importantly provides a redundancy to fall-back onto in case of a malfunction or outage.

Fig. 3 presents a summary showing how this survey differs from existing ones on 3D object detection. 3D object detection itself is an algorithmic problem, whereas involved

with autonomous driving makes it an application issue. As of this main text in June 2021, we notice that rather few investigations [13, 35, 14, 36] have looked into this application issue. Survey [13] focuses on 3D object detection, also taking into account indoor detection. Survey [36] involves with autonomous driving but it concentrates on multi-modal object detection. Survey [35] covers a series of related subtopics of 3D point clouds (e.g., 3D shape classification, 3D object detection and tracking, and 3D point cloud segmentation etc.). Note that survey [35] establishes its taxonomy based on network architecture, which fails to summarize the homogeneous properties among methods and therefore results in overlapping in the subdivisions, e.g. multi-view based and BEV based are the same in essence in terms of learning strategies. As far as we know, only survey [14] is closely relevant to this paper, but it fails to track the latest datasets (e.g., nuScenes [37], and Waymo Open [7]), algorithms (e.g., the best algorithm it reports on KITTI [8] 3D detection benchmark is AVOD [31][IROS'18: 71.88] vs. BtcDet [10][AAAI'22: 82.86] in this paper), and challenges, which is not surprising as much progress has been made after 2018.

The aims of this paper are threefold. First, notice that no recent literature exists to collect the growing knowledge concerning 3D object detection, we aim to fill this gap by starting with several basic concepts, providing a glimpse

of evolution of 3D object detection, together with comprehensive comparisons on publicly available datasets being manifested, with pros and cons being judiciously presented. Witnessing the absence of a universal consensus on taxonomy with respect to 3D object detection, our second goal is to contribute to the maturity of the taxonomy. To this end, we are cautiously favorable to the taxonomy based on input modality, approved by existing literature [13, 14]. The idea of grouping literature based on their network architecture derives from 2D object detection, which fails to summarize the homogeneous properties among methods and therefore results in overlapping in the subdivisions, *e.g.*, multi-view based and BEV based are the same representation in essence. Another drawback is that several plug-and-play module components can be integrated into either region proposal based (two-stage) or single shot based (one-stage). For instance, VoTr [5] proposes voxel transformer which can be easily incorporated into voxel based one stage or two stage detectors. Notice that diverse fusion variants consistently emerge among 3D object detection, existing taxonomy in works [13, 14] needs to be extended. For instance, works [38, 38] are sequential fusion methods (see Sec. 3.3.1), which are not well suited to existing taxonomy. We therefore define two new paradigms, *i.e.* sequential fusion and parallel fusion, to adapt to underlying changes and further discuss which category each method belongs to explicitly, while works [13, 14] not. Also, we analyze in deep to provide a more fine-grained taxonomy above and beyond the existing efforts [13, 14] on image based methods, say, *result-lifting based* and *feature-lifting based* depending on intermediate representation existence. Finally, we open point-voxel branch to classify newly proposed variants, *e.g.* PV-RCNN [39]. By contrast, survey [35] directly groups PV-RCNN into “Other Methods”, leaving the problem unsolved. Our third goal aims to present a case study on fifteen selected models among surveyed works, with regard to runtime analysis, error analysis, and robustness analysis closely. We argue that what mainly restricts the performance of detection is 3D location error based on our findings. Taken together, this survey is expected to foster more follow-up literature in 3D vision community.

The rest of this paper is organized as follows. Section 2 introduces background associated with foundations, sensors, datasets, and performance metrics. Section 3 reviews 3D object detection methods with their corresponding pros and cons in the context of autonomous driving. Comprehensive comparisons of the state-of-the-arts are summarized in Section 4. We conclude this paper and identify future research directions Section 5. We also set up a regularly updated project page on: <https://github.com/rui-qian/SoTA-3D-Object-Detection>.

## 2. BACKGROUND

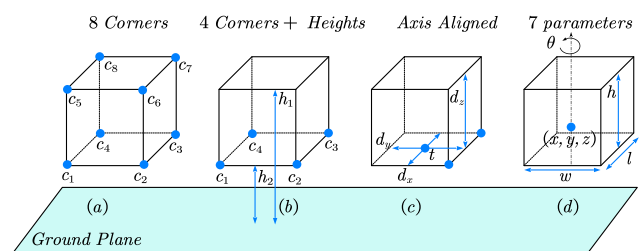
### 2.1. Foundations

Let  $\mathcal{X}$  denote input data, say, LiDAR signals or monocular images,  $\mathcal{F}$  denote a detector parameterized by  $\Theta$ . Consider an  $(F+1)$ -dimensional result subset with  $n$  predictions,

denoted by  $\{\mathbf{y}_1, \dots, \mathbf{y}_n\} \subseteq \mathbb{R}^{F+1}$ , we have

$$\Theta_{MLE} = \arg \max_{\Theta} \mathcal{F} \left( \{\mathbf{y}_1, \dots, \mathbf{y}_n\} \mid \mathcal{X}, \Theta \right), \quad (1)$$

where  $\mathbf{y}_i = (\mathcal{B}_i, s_i)$  denotes a certain prediction of detector  $\mathcal{F}(\mathcal{X}; \Theta)$  with bounding box  $\mathcal{B}_i \in \mathbb{R}^F$  and its probabilistic score  $s_i \in [0, 1]$ . In the context of autonomous driving,  $\mathcal{B}_i$  is usually parameterized as portrayed in Fig. 4, which indicates the volume of the object of interest and its position relative to a reference coordinate system that can be one of the sensors equipped on a ego-vehicle. We notice that attributes encoded by (d) in Fig. 4 are orthogonal and therefore result in a more lower information redundancy compared with (a), (b), (c). In this paper, we adopt the form of (d) as most previous works [25, 24, 6, 40, 41] do.



**Figure 4: Comparisons of the 3D bounding box parameterization**, between 8 corners proposed in [30], 4 corners with heights proposed in [31], the axis aligned box encoding proposed in [42], and the 7 parameters for an oriented 3D bounding box adopted in [25, 24, 6, 40, 41].

### 2.2. Sensors

We human beings leverage visual and auditory systems to perceive the real world when driving, so how about autonomous vehicles? If they were to drive like a human, then to identify what they see on the road constantly is the way to go. To this end, sensors matter. It is sensors that empower vehicles a series of abilities: obstacles perception, automatic emergency braking, and collision avoidance *etc.* In general, the most commonly used sensors can be divided into two categories: passive sensors and active sensors [43]. The on going debate among industry experts is whether or not to just equip vehicles with camera systems (no LiDAR), or deploy LiDAR together with on-board camera systems. Given that camera is considered to be one of the typical representatives of passive sensors, and LiDAR is regarded as a representative of active ones, we first introduce the basic concepts of passive sensors and active sensors, then take camera and LiDAR as examples to discuss how they serve the autonomous driving system, together with pros and cons being manifested in Table 1.

#### 2.2.1. Passive Sensors

Passive sensors are anticipated to receive natural emissions emanating from both the Earth’s surface and its atmosphere. These natural emissions could be natural light or infrared rays. Typically, a camera directly grabs a bunch of color points from the optics in the lens and arranges them

**Table 1**  
Advantages and disadvantages of different sensors.

|         | Sensors           | Advantages   | Disadvantages   | Publications  |
|---------|-------------------|--|---|---|
| Passive | Monocular Camera  | <ul style="list-style-type: none"> <li>•cheap and available for multiple situations</li> <li>•informative color and texture attributes</li> </ul>                          | <ul style="list-style-type: none"> <li>•no depth or range detecting feature</li> <li>•susceptible to weather and light conditions</li> </ul>  | [44], [15], [16], [45], [17], [41]  |
|         | Stereo Camera     | <ul style="list-style-type: none"> <li>•depth information provided</li> <li>•informative color and texture attributes</li> <li>•high frame rate</li> </ul>                 | <ul style="list-style-type: none"> <li>•computationally expensive</li> <li>•sensitive to weather and light conditions</li> <li>•limited Field-of-View</li> </ul>                    | [18], [46], [47]  |
| Active  | LiDAR             | <ul style="list-style-type: none"> <li>•accurate depth or range detecting feature</li> <li>•less affected by external illumination</li> <li>•360° Field-of-View</li> </ul> | <ul style="list-style-type: none"> <li>•high sparseness and irregularity by nature</li> <li>•no color and texture attributes</li> <li>•expensive and critical deployment</li> </ul> | [48], [49], [39], [40], [50], [51], [52], [53], [25], [54], [6], [55], [24], [38], [56], [57] |
|         | Solid State LiDAR | <ul style="list-style-type: none"> <li>•more reliable compared with surround view sensors</li> <li>•cost decrease</li> </ul>   | <ul style="list-style-type: none"> <li>•error increase when different points of view are merged in real time</li> <li>•still under development and limited Field-of-View</li> </ul> | n.a.  |

**Table 2**  
A summary of publicly available datasets for 3D object detection in the context of autonomous driving. \*: Numbers in brackets indicate classes evaluated in their official benchmarks.

| Dataset        | Year | Size      |          |            |        |         | Diversity |           |       |      | Modality |          |       | Benchmark | Cites |
|----------------|------|-----------|----------|------------|--------|---------|-----------|-----------|-------|------|----------|----------|-------|-----------|-------|
|                |      | #Train    | #Val     | #Test      | #Boxes | #Frames | #Scenes   | #Classes* | Night | Rain | Stereo   | Temporal | LiDAR |           |       |
| KITTI [8, 58]  | 2012 | 7,418x1   | -        | 7,518 × 1  | 200K   | 15K     | 50        | 8 (3)     | No    | No   | Yes      | Yes      | Yes   | Yes       | 5011  |
| Argoverse [37] | 2019 | 39,384x7  | 15,062x7 | 12,507 × 7 | 993K   | 44K     | 113       | 15        | Yes   | Yes  | Yes      | Yes      | Yes   | Yes       | 88    |
| Lyft L5 [59]   | 2019 | 22,690x6  | -        | 27,460 × 6 | 1.3M   | 46K     | 366       | 9         | No    | No   | No       | Yes      | Yes   | No        | -     |
| H3D [60]       | 2019 | 8,873x3   | 5,170x3  | 13,678 × 3 | 1.1M   | 27K     | 160       | 8         | No    | No   | No       | Yes      | Yes   | No        | 31    |
| Applo [61, 62] | 2019 | -         | -        | -          | -      | 140K    | 103       | 27        | Yes   | Yes  | Yes      | Yes      | Yes   | Yes       | 78    |
| nuScenes [37]  | 2019 | 28,130x6  | 6,019x6  | 6,008 × 6  | 1.4M   | 40K     | 1,000     | 23 (10)   | Yes   | Yes  | No       | Yes      | Yes   | Yes       | 225   |
| Waymo [7]      | 2020 | 122,200x5 | 30,407x5 | 40,077 × 5 | 112M   | 200K    | 1,150     | 4 (3)     | Yes   | Yes  | No       | Yes      | Yes   | Yes       | 31    |

into an image array that is often referred to as a digital signal for scene understanding. Primarily, a monocular camera lends itself well to informative color and texture attributes, better visual recognition of text from road signs, and high frame rate at a negligible cost *etc.* Whereas, it is lack of depth information, which is crucial for accurate location estimation in the real 3D world. To overcome this issue, stereo cameras use matching algorithms to align correspondences in both left and right images for depth recovery [2]. While cameras have shown potentials as a reliable vision system, it is hardly sufficient as a standalone one given that a camera is prone to degrade its accuracy in cases where luminosity is low at night-time or rainy weather conditions occur. Consequently equipping autonomous vehicles with an auxiliary sensor, say active counterparts, to fall-back onto is necessary, in case that camera system should malfunction or disconnect in hazardous weather conditions.

### 2.2.2. Active Sensors

Active sensors are expected to measure reflected signals that are transmitted by the sensor, which are bounced by the Earth's surface or its atmosphere. Typically, Light Detection And Ranging (LiDAR) is a point-and-shoot device with three basic components of lens, lasers and detectors, which spits out light pulses that will bounce off the surroundings in the form of 3D points, referred to as "point clouds". High sparseness and irregularity by nature and the absence of texture attributes are the primary characteristics of a point cloud, which is well distinguished from image array. Since

we have already known how fast light travels, the distance of obstacles could be determined without effort. LiDAR system emits thousands of pulses that spin around in a circle per second, with a 360 degree view of surroundings for the vehicles being provided. For example, Velodyne HDL-64L produces 120 thousand points per frame with a 10 Hz frame rate. Obviously, LiDAR is less affected by external illumination conditions (*e.g.*, at night-time), given that it emits light pulses by itself. Although LiDAR system has been hailed for high accuracy and reliability compared with camera system, it does not always hold true. Specifically, wavelength stability of LiDAR is susceptible to variations in temperature, while adverse weather (*e.g.*, snow or fog) is prone to result in poor signal-to-noise ratio in the LiDAR's detector. Another issue with LiDAR is the high cost of deployment. A conservative estimate according to Velodyne, so far, is about \$75,000<sup>1</sup> [14]. In the foreseeable future of LiDAR, how to decrease cost and how to increase resolution and range are where the whole community is to march ahead. As for the former, the advent of solid state LiDAR is expected to address this problem of cost decrease, with the help of several stationary lasers that emit light pulses along a fixed field of view. As for the latter, the newly announced Velodyne VLS-128 featuring 128 laser pulses and 300m radius range has been on sale, which is going to significantly facilitate better recognition and tracking in terms of public safety.

<sup>1</sup><http://www.woodsdecap.com>

Table 3

Instance distribution on nuScenes *train* split. Here, "TC" and "Cons.Veh." denote traffic cone and construction vehicle respectively.

| #Classes | Car     | Pedestrian | Barrier | TC     | Truck  | Trailer | Bus    | Cons.Veh. | Motocycle | Bicycle |
|----------|---------|------------|---------|--------|--------|---------|--------|-----------|-----------|---------|
| Number   | 413,318 | 185,847    | 125,095 | 82,362 | 72,815 | 20,701  | 13,163 | 11,993    | 10,109    | 9,478   |

Table 4

Instance distribution on KITTI *train* split. *Car* category accounts for 82.99% of the three (*i.e.*, *Car*, *Pedestrian*, and *Cyclist*).

| #Classes | Car    | Ped.  | Van   | Cyclist | Truck | Misc | Tram | Person.sit. |
|----------|--------|-------|-------|---------|-------|------|------|-------------|
| Number   | 14,357 | 2,207 | 1,297 | 734     | 448   | 337  | 224  | 56          |

### 2.2.3. Discussion

Fatalities occurred with autonomous vehicles have already increased the society's grave concern about safety. If autonomous vehicles were to hit the road legally, they at least need to satisfy three basic requirements: high accuracy, high certainty, and high reliability. To this end, sensor fusion incorporating the merits of two worlds (camera vs. LiDAR) is going to be necessary. From a sensor standpoint, LiDAR provide depth information close to linearity error with a high level of accuracy, but it is susceptible to adverse weather (*e.g.*, snow or fog). Camera is intuitively much better at visual recognition in cases where color or texture attributes are available, but they are not sufficient as a standalone system as aforementioned. Note that certainty is still an important yet largely unexplored problem. A combination of LiDAR and camera is anticipated to ensure detection accuracy and improve prediction certainty. With regard to reliability, two facets should be considered: sensor calibration and system redundancy. Sensor calibration undoubtedly increases the difficulty of deployment and directly affects the reliability of the whole system. Studies [63, 64] have looked into calibrating sensors to avoid drift over time. System redundancy is to have a secondary sensor to fall-back onto in case of a malfunction or outage in extreme scenarios. Although balancing affordability and safety has been a long-term ethical dilemma, the community should be keenly aware of the safety risk of over-reliance on a single sensor.

### 2.3. Dataset

The availability of large-scale datasets has been continuously fostering the community with the advent of data-driven era. As regards 3D object detection, we summarize publicly available datasets [8, 58, 37, 59, 60, 61, 62, 65, 7] in the context of autonomous driving in Table 2, out of which the KITTI [8], nuScenes [37], and Waymo Open [7] are the typical representatives. In a sequel, we selectively introduce these three datasets with regard to their size, diversity, pros and cons accordingly.

**Dataset size.** The KITTI manually annotates 200K boxes among 15K frames, with 7,481, 7,518 samples for training and testing respectively. Rather, the training set is subdivided into 3,712 and 3,769 samples for *train*, *val* split as a common practice initially introduced by 3DOP [66]. The nuScenes manually labels 1.4M boxes among 40K frames, with 28,130, 6,019 and 6,008 frames for training, validation,

and testing accordingly. Waymo Open, encouragingly annotates 112M boxes among 200K frames, with 122,200, 30,407 and 40,077 for training, validation, and testing. Note that only the labels for training/validation are available, whereas none of them provide annotations for testing. Competitors are required to submit predictions to the online leaderboard for fairly assessing on *test* set.

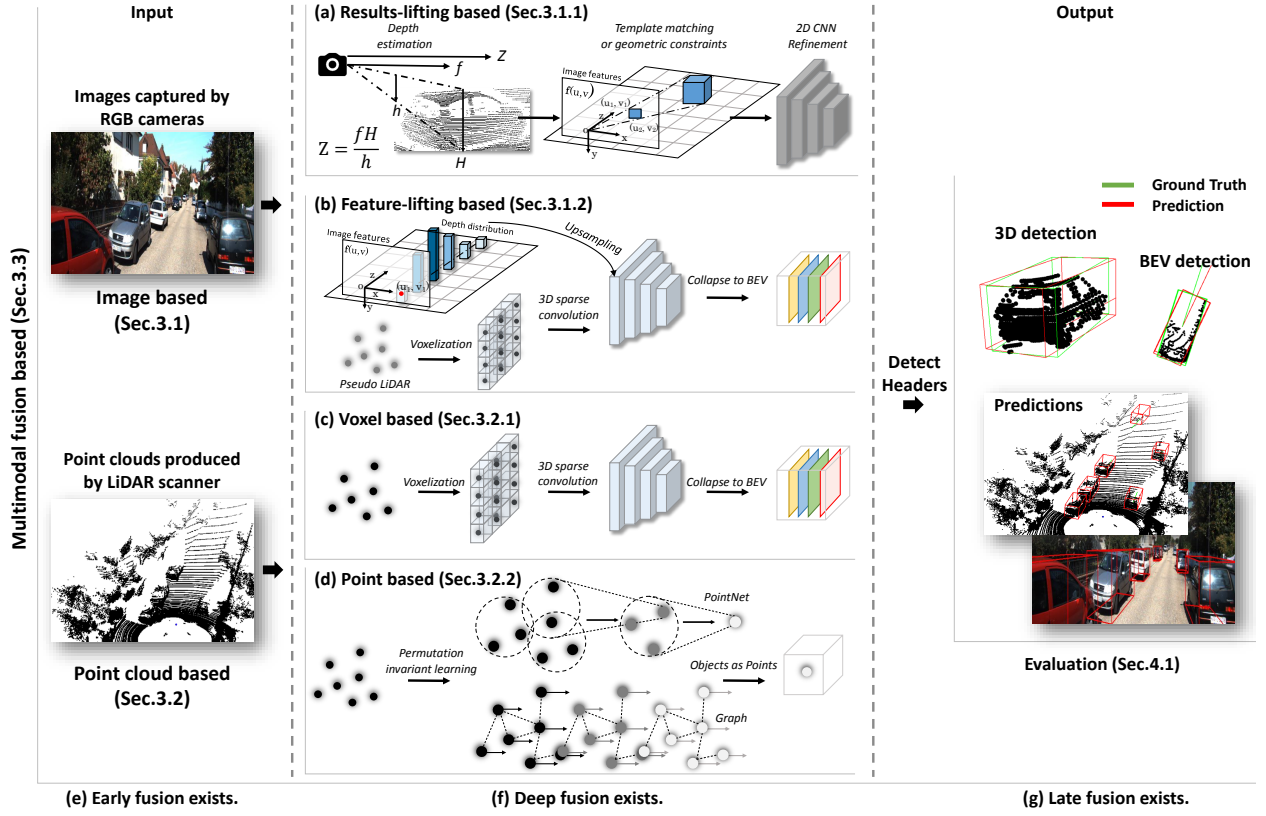
**Diversity.** The KITTI provides 50 scenes over 8 classes in Karlsruhe, Germany, out of which only *Car*, *Pedestrian*, and *Cyclist* are considered for online evaluation. Three difficulty levels (*i.e.*, *Easy*, *Moderate*, and *Hard*) for its protocol are introduced depending on the height of 2D bounding boxes, the level of occlusion and truncation. The nuScenes collects 1,000 sequences over 23 classes in Boston and Singapore, out of which only 10 classes are considered for 3D object detection. The Waymo Open consists of 1,150 sequences over 4 classes in Phoenix and San Francisco, out of which only three classes are assessed similar to KITTI. It is worth mentioning that both nuScenes and Waymo Open acquire their data under multiple weather (*e.g.*, rainy, foggy, and snowy *etc.*) and lighting (*e.g.*, daytime and nighttime) conditions throughout a day, whereas KITTI only captures its dataset on sunny days.

**Pros and cons.** These three datasets manifested above indeed catalytically foster the academics. KITTI, as a pioneer, has profoundly influences the whole community in terms of data acquisition, protocol and benchmark. Nevertheless, as we mentioned above, KITTI is recorded in daytime on sunny days, without taking lighting and weather conditions into account, resulting a relative lower diversity compared with nuScenes and Waymo Open. Real dataset tends to suffer from class imbalance as it is true to life. According to our statistics in Table 3, 50% categories account for only 6.9% of total annotations, which clearly reflects a long-tail distribution on nuScenes. This phenomenon also holds true for KITTI as indicated in Table 4.

### 2.4. Performance Metrics

3D object detection adopts the Average Precision (*AP*) as its primary performance metrics, all of which stem from the same ideology as its 2D counterparts [67]. We first revisit the vanilla form of *AP* metric, and then recognize subtle connections and differences of dataset-specific *AP* adopted among commonly used benchmarks.

**Revisiting.** Consider a prediction subset  $\{y_1, \dots, y_n\}$  in descending order by confidence score  $s_i$ , a prediction  $y_i$



**Figure 5: Pipeline of 3D object detection in general.** *Image based*, which either lifts estimated 2D results into 3D space via template matching, geometric constraints in (a), or directly lifts 2D image features into 3D space via computing a Pseudo LiDAR, learning a latent depth distribution in (b). *Point clouds based*, which either voxelizes an irregular point cloud into regular voxel grids and then learn feature representation in an explicit way in (c), or leverage PointNet-like block, GNNs to learn permutation-invariant representations in an implicit fashion in (d). *Multimodal fusion based*, which is likely to fuse cross-modalities at early phase in (e), middle phase in (f), and late phase in (g) during the forward propagation.

is considered as true positive if the ratio of overlapping area between  $B_i$  and its assigned ground-truth, namely, the Intersection over Union (IoU), exceeds a certain threshold, otherwise false positive. The zigzag-like precision-recall curve can easily be plotted and  $AP$  is just the area under the curve. Notice that accurately calculating the related area is difficult, PASCAL VOC [67] established an alternative instead.

*Interpolated  $AP|_{R_N}$  Metric* is formulated as the mean precision calculated at a recall subset  $R$  of which  $N$  evenly spaced recall levels are composed, that is

$$AP|_{R_N} = \frac{1}{N} \sum_{r \in R} P_{interpolate}(r), \quad (2)$$

where  $R = \left[ r_0, r_0 + \frac{r_1 - r_0}{N-1}, r_0 + \frac{2(r_1 - r_0)}{N-1}, \dots, r_1 \right]$ . For each recall level  $r$ , its corresponding precision is *interpolated* by taking into account the maximum precision of which recall value is greater than or equal to  $r$ , denoted by

$$P_{interpolate}(r) = \max_{\tilde{r}: \tilde{r} \geq r} P(\tilde{r}). \quad (3)$$

**KITTI Benchmark.** KITTI adopts standard *Interpolated  $AP|_{R_{11}}$  Metric* as its official metric for assessing detector  $\mathcal{F}(\mathcal{X}; \Theta)$ . Usually two leaderboard is considered, i.e.

3D detection and Bird's Eye View (BEV) detection. 3D detection evaluates  $AP_{3D}|_{R_{11}}$  with a rotated  $IoU_{3D}$  threshold of 0.7, 0.5, 0.5 for *Car*, *Pedestrian* and *Cyclist* accordingly. The principles of 3D detection largely holds true for BEV detection, except for the calculation of  $IoU_{BEV}$ , which is calculated by projecting the bounding box  $B_{3D}$  from 3D space into the ground plane. Note that from 08.10.2019, KITTI turns to 40 recall levels  $[1/40, 2/40, 3/40, \dots, 1]$  instead of 11 recall levels  $[0, 1/10, 2/10, \dots, 1]$  as suggested in [68], with recall level 0 being reasonably removed.

**nuScenes Benchmark.** nuScenes uses NuScenes Detection Score (NDS) as its official metric for evaluating detector  $\mathcal{F}(\mathcal{X}; \Theta)$ . Consider a mean average error subset  $\mathcal{E}$  of which translation, size, orientation, attribute and velocity are composed, denoted by  $\mathcal{E} = \{mATE, mASE, mAOE, mAAE, mAVE\}$ , we have

$$NDS = \frac{1}{10} \left[ 5mAP + \sum_{err \in \mathcal{E}} (1 - \min(1, err)) \right], \quad (4)$$

where  $mAP$  indicates mean Average Precision. NDS jointly justifies a weighted average of  $mAP$  and mean average errors of set  $\mathcal{E}$  among 10 classes. It is worth noting that  $mAP$  is calculated by a bird-eye-view center distance of thresholds  $\{0.5m, 1m, 2m, 4m\}$  rather than standard box-overlap.

**Waymo Benchmark.** Waymo Open leverages *Interpolated AP*<sub>R<sub>21</sub></sub> Metric and Average Precision weighted by Heading (APH) as its primary metric for evaluating detector  $\mathcal{F}(\mathcal{X}; \Theta)$ . To compute AP, Waymo evaluates on 21 equally spaced recall levels  $[0, 1/20, 2/20, \dots, 1]$  in Equation 2 with an IoU threshold of 0.7, 0.5 for vehicles, pedestrians respectively. To compute APH, the heading accuracy is incorporated into true positives, each of which is weighted by  $\min(|\theta - \theta^*|, 2\pi - |\theta - \theta^*|) / \pi$ , where  $\theta$  and  $\theta^*$  are subject to  $[-\pi, \pi]$ , indicating the predicted azimuth and its assigned ground truth accordingly. Waymo breaks down its difficulty into two levels: LEVEL\_1 lends itself to boxes with at least five LiDAR signals, while LEVEL\_2 is suited for all non-empty ones.

### 3. TAXONOMY AND REVIEW

Fig. 5 presents a pipeline of 3D object detection in general. Having only images on hand, the core challenge arises from the absence of depth information. Usually two lines exist: one is to break down this 3D vision task into 2D object detection [82, 83, 84, 85, 86, 87, 88] and depth estimation [2, 3], which lifts these estimated results into 3D space via geometric properties and constraints. The other line is to directly lift 2D image features into 3D space via computing a point cloud [47, 45, 41, 71, 73, 75], termed as *Pseudo LiDAR* or learning a latent depth distribution [4, 72, 74, 70]. Having only point clouds on hand, the key difficulty stems from the representation learning over sparse, irregular, and unordered point clouds. Also two ways exist mainly: one is to first voxelize an irregular point cloud into regular voxel grids and then learn feature representations in an explicit way. Nevertheless, the other is to leverage PointNet-like block [89] or Graph Neural Networks (GNNs) [90] to learn permutation-invariant representations in an implicit fashion. Nowadays, combining the merits of these two lines reveals a new fashion trend. What if having both of them on hand? The tricky obstacle often derives from semantic representation (what to fuse), alignment (how to fuse), and consistency (when to fuse) for multimodal fusion based on what we have already learnt from these two preceding modalities.

Depending on what we feed to  $\mathcal{F}(\mathcal{X}; \Theta)$  internally during inference, we frame our taxonomy along three dimensions: (1) image based, (2) point cloud based, and (3) multimodal fusion based, ordered chronologically in which each method emerges. Table 5 selectively manifests several core literature structured along each dimension, which is expected to leave the readers a clear picture in his or her mind. In what follows, we present image based in Section 3.1, and point cloud based in Section 3.2. Multimodal fusion based is addressed in Section 3.3.

#### 3.1. Image based Methods

Depth estimation from images is still an ill-posed problem that are not fully understood yet [2, 3]. Errors from depth recovery inherently contribute to the major factor of the performance gap between image based and point cloud

based [19]. Focusing on the inverse issue, it is the way of recovering depth and the way of use that collected knowledge that determines how the intermediate representation is lifted. Depending on intermediate representation existence, we divide this group into the following two subdivisions: (1) result-lifting based, (2) feature-lifting based. Table 5 selectively lists several significant contributions concerning the subject.

##### 3.1.1. Result-lifting based Methods

Works in this group break down  $\mathcal{F}(\mathcal{X}; \Theta)$  into two vision tasks: 2D object detection and depth estimation [15, 16, 17, 18, 19, 20, 21, 22]. The underlying principle is that the spatial location of the associated objects can be empirically inferred with regard to the visual appearance. To that end, Mono3D [44] scores proposals with location prior, object shape, size, and semantics with the hypothesis that objects are close to the ground plane via minimizing energy function. Deep3DBox [16] leverages the geometric properties that the perspective projection of 3D corners should tightly touch at least one side of the 2D bounding box. GS3D [17] relies on the observation that the top center of a 3D bounding box should be close to the top midpoint of the 2D bounding box when projected onto the image plane. Stereo R-CNN [18] exploits geometric alignment with keypoints, yaw angle, and object shape using left and right proposal pairs. Literature [69, 21, 19] fully exploits semantic properties as well as dense geometric constraints in monocular imagery. We notice that these methods are over-reliance on feature engineering and hinder them from further extending to general scenarios. For instance, works [44, 15, 20] require external data for training. Keypoint constraints in works [18, 19, 69] are more likely to be vulnerable to slim and tall objects, say, Pedestrians. To what extent works in this group rely on domain expertise determines to what extent efforts in this group can be generalized largely.

##### 3.1.2. Feature-lifting based Methods

Works in this group develop  $\mathcal{F}(\mathcal{X}; \Theta)$  via lifting 2D image features into 3D space via computing a point cloud intermediately [47, 45, 41, 71, 73, 75] or learning a categorical depth distribution directly [4, 74]. To this end, works [71, 45, 41] first uses a stand-alone depth estimation network to obtain disparity map, then back-projects 2D coordinates associated with each pixel in image plane into 3D space, and finally lends itself to independent point cloud based detectors. Notice that the depth estimation error grows quadratically at long ranges, Pseudo-LiDAR++ [73] uses sparse but accurate real LiDAR signals as “landmarks” to correct and de-bias depth errors. Observe that preceding networks are trained separately, Pseudo-LiDAR E2E [75] goes one step further by making the entire pipeline trainable end to end. We note that Pseudo-LiDAR signals in works [71, 73, 41] are internally accompanied by noises that stem from errors in depth estimation, which reflect in two aspects: (1) Pseudo-LiDAR signals are slightly off relative to the real LiDAR ones with a local misalignment. (2) depth artifacts



**Table 5**  
**A taxonomy of 3D object detection for autonomous driving.**

| Input modality   | Subdivision                      | Network architecture  |   |
|--|----------------------------------|---|---|
|  |                                  | Two-stage   | One-stage   |
| <i>Depending on intermediate representation existence:</i>           |                                  |   |   |
| Image<br>(Sec.3.1)   | result-lifting<br>(Sec.3.1.1)    | <i>with extra data:</i><br>+ Mono3D [44], CVPR'16   | <i>with extra data:</i><br>+ DD3D [20], ICCV'21   |
|  |                                  | <i>without extra data:</i><br>+ Deep MANTA [15], CVPR'17<br>+ Deep3DBox [16], CVPR'17<br>+ GS3D [17], CVPR'19<br>+ Stereo R-CNN [18], CVPR'19<br>+ MonoRCNN [19], ICCV'21   | <i>without extra data:</i><br>+ Monodle [21], CVPR'21<br>+ MonoFlex [69], CVPR'21<br>+ YOLOStereo3D [22], ICRA'21   |
| Point cloud<br>(Sec.3.2)   | feature-lifting<br>(Sec.3.1.2)   | <i>with extra data:</i><br>+ MF3D [45], CVPR'18<br>+ Mono3D-PLiDAR [41], ICCVW'19<br>+ Pseudo-LiDAR [71], CVPR'19<br>+ Pseudo-LiDAR++ [73], ICLR'20<br>+ Pseudo-LiDAR E2E [75], CVPR'20                                       | <i>with extra data:</i><br>+ OFT-Net [70], BMVC'19<br>+ DSGN [4], CVPR'20<br>+ LIGA-Stereo [72], ICCV'21<br>+ CaDDN [74], CVPR'21   |
|  |                                  | <i>without extra data:</i><br>+ 3DOP [47], NeurIPS'15   |   |
| <i>Depending on representation learning strategies:</i>              |                                  |   |   |
| Point cloud<br>(Sec.3.2)   | voxel based<br>(Sec.3.2.1)       | <i>single-scale voxelization:</i><br>+ TANet [76], AAAI'20<br>+ SPG [77], ICCV'21<br>+ CenterPoint [23], CVPR'21<br>+ BtcDet [10], AAAI'22  | <i>single-scale voxelization:</i><br>+ VeloFCN [48], RSS'16<br>+ PIXOR [49], CVPR'18<br>+ VoxelNet [24], CVPR'18<br>+ SECOND [25], Sensors'18<br>+ PointPillars [6], CVPR'19<br>+ CIA-SSD [26], AAAI'21<br>+ SE-SSD [27], CVPR'21<br>+ Voxel R-CNN [28], AAAI'21<br>+ CT3D [29], ICCV'21<br>+ VoTr [5], ICCV'21 |
|  |                                  | <i>multi-scale voxelization:</i><br>+ Part-A <sup>2</sup> [56], T-PAMI'21   | <i>multi-scale voxelization:</i><br>+ HVNet [57], CVPR'20<br>+ Voxel-FPN [55], Sensors'20   |
|  | point based<br>(Sec.3.2.2)       | + PointRCNN [40], CVPR'19   | + 3DSSD [51], CVPR'20<br>+ Point-GNN [52], CVPR'20  |
|  | point-voxel based<br>(Sec.3.2.3) | + Fast PointRCNN [78], ICCV'19<br>+ STD [50], ICCV'19<br>+ PV-RCNN [39], CVPR'20<br>+ BADet [79], PR'22   | + SA-SSD [53], CVPR'20  |
| <i>Depending on to what extent these two modalities are coupled:</i> |                                  |   |   |
| Multimodal<br>(Sec.3.3)  | sequential fusion<br>(Sec.3.3.1) | + Frustum-PointNets [80], CVPR'18<br>+ Frustum-ConvNet [38], IROS'19  |   |
|  | parallel fusion<br>(Sec.3.3.2)   | <i>early fusion:</i><br>+ PointPainting [12], CVPR'20<br><i>deep fusion:</i><br>+ MV3D [30], CVPR'17<br>+ AVOD [31], IROS'18<br>+ MMF [81], CVPR'19<br>+ 3D-CVF [33], ECCV'20<br><i>late fusion:</i><br>+ CLOCs [34], IROS'20 | <i>deep fusion:</i><br>+ ContFuse [32], ECCV'18   |

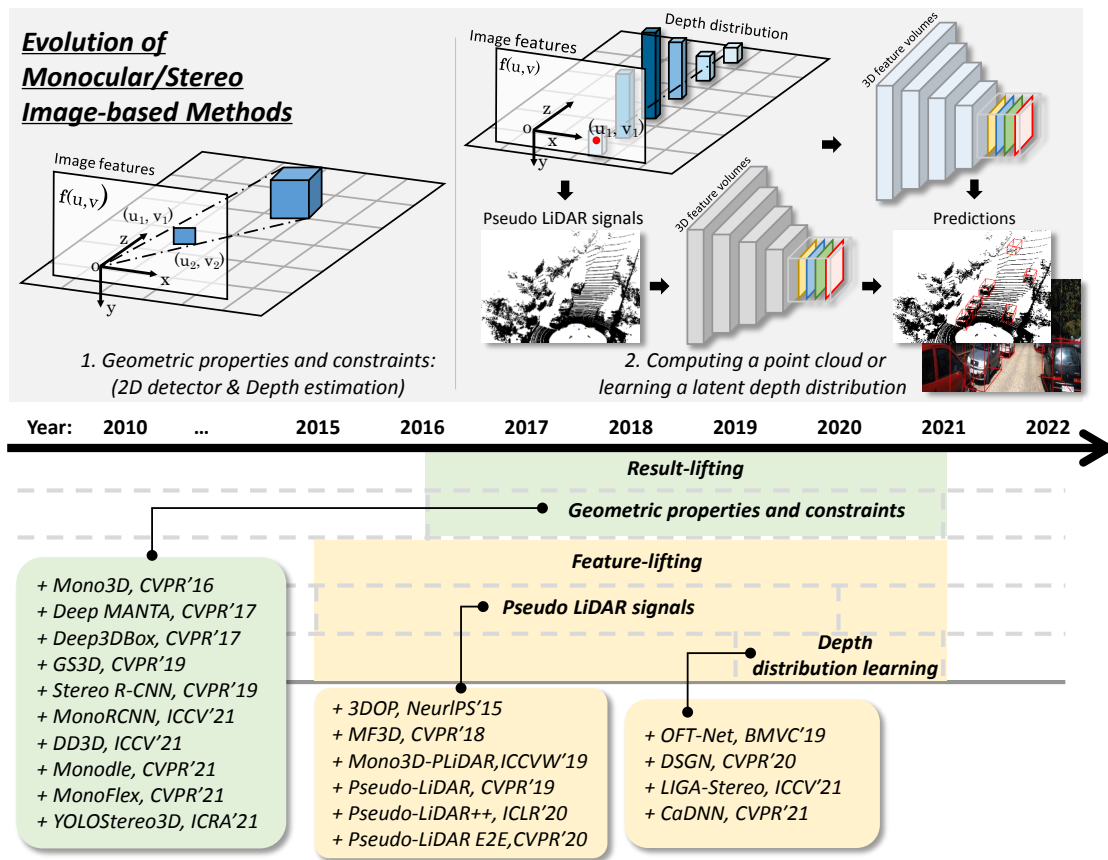


Figure 6: Evolution of monocular or stereo image-based methods.

commonly exist in the vicinity of an object along with a long tail [41]. Works [4, 74] also point out that the independent networks, say depth estimation, make this 3D representation sub-optimal concerning the non-differentiable transformation from 2D coordinates into 3D space. Consequently, instead of computing an intermediate point cloud, the other scheme wants to incorporate 3D geometry into image based networks and learn a latent depth distribution [4, 72, 74] explicitly or implicitly [70] in an end-to-end manner. DSGN [4] and CaDNN [74] encode visual appearance into 3D feature volumes, jointly optimizing depth and semantic cues for 3D detection. LIGA-Stereo [72] turns to real LiDAR signals for the high-level geometry-aware supervisions and beyond under the guidance of a teacher network.

### 3.1.3. Summary

Fig. 6 illustrates the evolution of image based. Works in this group are divided into result-lifting based and feature-lifting based accordingly. Result-lifting based depends on domain expertise to design 3D representation and resorts to prior knowledge for template matching [44] or geometric constraints for recovery [16, 17, 69]. Feature-lifting based wants to compute an intermediate representation, say Pseudo LiDAR [71, 45, 41] or 3D feature volumes [4, 72, 74]. As the absence of depth information, we note that works [15, 74, 20, 4, 41, 71, 73, 75, 44] rely on additional data for training. For an autonomous system, redundancy is indispensable to

guarantee safety apart from economic concerns, so image based methods are poised to make a continuing impact over the next few years.

## 3.2. Point Cloud based Methods

Convolutional Neural Networks (CNNs) has always been applied in computer vision for its capability of exploiting spatially-local correlations in dense regular grids [91] (e.g., images). Whereas, point clouds are sparse in distribution, irregular in structure, and unordered in geometry by nature. Consequently, convolving against a point cloud directly will lead to a gross distortion of representation [9]. To address issues above, works [24, 25, 6] voxelize their inputs to lend themselves to convolution operator adaptively, while works [51, 51, 40] turns to the recent advances of learning over point sets [89, 92, 93, 90] for help instead. Depending on representation learning strategies, we therefore divide existing works into the following three groups: (1) voxel based, (2) point based, and (3) voxel-point based.

### 3.2.1. Voxel based Methods

Works in this group voxelize irregular point clouds to 2D/3D compact grids and then collapse it to a bird's-eye-view 2D representation on which CNNs effectively convolve against [76, 48, 49, 77, 23, 24, 25, 10, 6, 26, 27, 28, 29, 5, 56, 57, 55]. Typically, VoxelNet [24] rasterizes point clouds into volumetric dense grids, followed by 3D CNNs

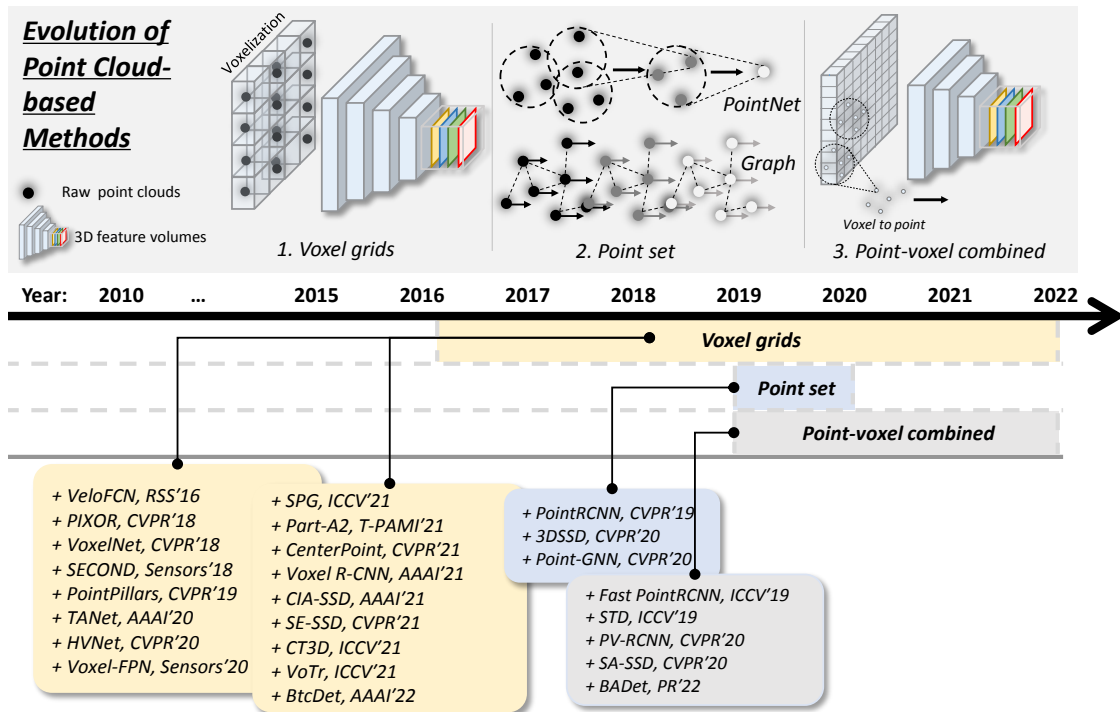


Figure 7: Evolution of point cloud-based methods.

that convolve along each dimension. Notice that the computation overheads and memory footprints grow cubically in VoxelNet, SECOND [25] leverages sparse convolution operation to get rid of unnecessary computation squandered by unavailing zero-padding voxels. To thoroughly remove 3D CNNs layers, PointPillars [6] elongates voxels into pillars that are arrayed in a BEV perspective. Note that PointPillars decreases the resolution in vertical axis in essence which is detrimental to the learning of representation. As a trade-off, noticeable efforts stretch back to voxels. Voxel R-CNN [28] aggregates voxel-wise features from 3D convolutional volumes for proposal refinement. SE-SSD [27] jointly supervises a student network under the guidance of the distilled knowledge injected by its teacher. Works [56, 57, 55] incorporate multi-scale voxelization strategies into feature aggregation. Works [29, 5, 76] exploit long-range contextual dependencies among voxels inspired by recent success of Transformers [94] in computer vision. Works [10, 77] integrate shape learning strategy to the deteriorating point cloud quality that caused by adverse weather, occlusions, and truncations *etc.* It is worth mentioning that voxel based approaches benefit from bird-view representation, which possess less scale ambiguity and minimal occlusions [48, 49].

### 3.2.2. Point based Methods

Works in this group utilize permutation invariant operators to implicitly capture local structures and fine-grained patterns without any quantization in order to retain the original geometry of a raw point cloud [51, 52, 40]. To this end, PointRCNN [40] leverages PointNet-like block [92] to

learn semantic cues associated with foreground points over which 3D proposals are generated in a bottom-up fashion. Notice that PointNet series are exhausted with the process of upsampling and broadcasting semantic cues back into the relevant points, 3DSSD [51] closely revisits the sampling strategy with feature distance, compounded by Euclidean distance, to safely remove it. Encouraged by the promising performance on classification and semantic segmentation from point clouds [90, 95, 96], Point-GNN [52] reasons on local neighborhood graphs constructed from point clouds, on which each node iteratively summarizes semantic cues from intermediate reaches of its neighbors. Point based approaches are internally time-consuming with a ball query complexity of  $\mathcal{O}(k \cdot |\mathcal{X}|)$ . Note that works [51, 40] leverage multi-scale and multi-resolution grouping to achieve an expanding receptive field, which make latency even severe as points in  $\mathcal{X}$  grow.

### 3.2.3. Point-voxel based Methods

Works in this group reveal a new fashion trend to integrate the merits of both worlds together: voxel based [24, 25, 6] approaches are computationally effective but the desertion of fine-grained patterns degrades further refinement, while point based methods [52, 51, 40] have relatively higher latency but wholly preserve the irregularity and locality of a point cloud [78, 50, 39, 79, 53]. STD [50] applies PointNet++ to summarize semantic cues for sparse points, each of which is then voxelized to form a dense representation for refinement. PV-RCNN [39] deeply integrates the effectiveness of 3D sparse convolution [25] and the flexible receptive fields of PointNet-like set abstraction [92] to learn

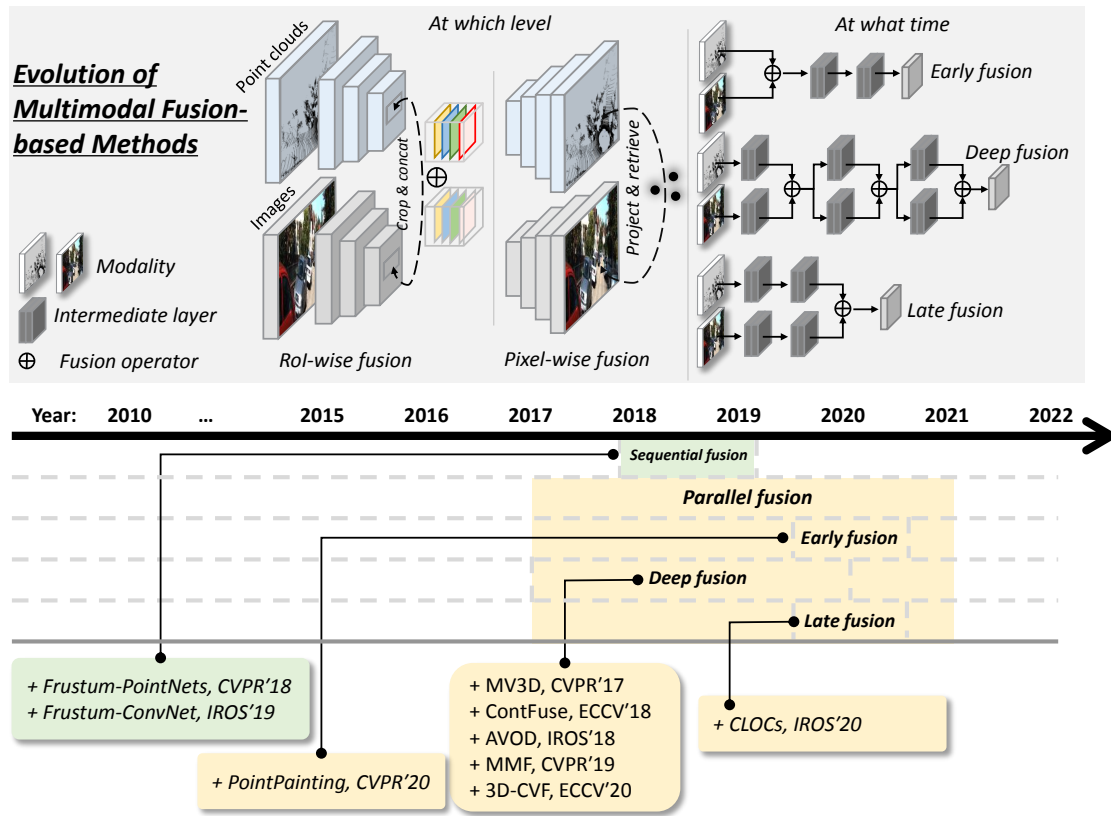


Figure 8: Evolution of multimodal fusion-based methods.

more discriminative semantics. SA-SSD [53] interpolates 3D sparse convolution features for raw point clouds on which an auxiliary network is applied to endow voxel features with structure-aware capability. BADet [79] exploits long-range interactions iteratively among detection candidates, wherein local neighborhood graphs are constructed to facilitate a boundary-aware receptive field in a coarse-to-fine manner.

### 3.2.4. Summary

Fig. 7 illustrates the evolution of point cloud based. Works in this group are divided into voxel based, point based, and point-voxel based respectively, among which voxel based appears to be dominant, as evidenced by Fig. 7. Voxel based is easily amenable to efficient hardware implementations with distinguished accuracy and relatively lower latency. Point based easily retains the spatially-local structure of a point cloud at the cost of taking longer feedforward time than those based on voxels. Comparing the three subdivisions, voxel based is still the most promising direction currently existing concerning real-time applications, say autonomous driving.

## 3.3. Multimodal Fusion based Methods

Intuitively, single modality itself has its own defects, a joint treatment appears to be a potential opportunity to failure cases. Nevertheless, multimodal fusion based has still trailed its point cloud based counterparts thus far. What, how,

and when to fuse have not been full understood yet [36]. Depending on to what extent these two modalities are coupled, we divide existing efforts into sequential fusion based and parallel fusion based. As the names suggest, if data flow has more than one independent path through the networks, then it is considered as *parallel*, otherwise *sequential*. The former emphasizes that the data flow of different modalities can pass through networks concurrently, while the latter emphasizes that the data flow of different modalities only has one single path to flow through networks successively. In what follows, we analyze these two paradigms as regards fusion evolutions, connections, and concerns to meet the requirements.

### 3.3.1. Sequential Fusion based Methods

Works in this group are characterized by 2D driven 3D pipelines in a sequential manner, wherein the input of downstream depends heavily on the output of upstream [80, 38]. Frustum PointNets [80] first leverages a mature 2D CNN object detector to predict 2D proposals, each of which are then transformed to 3D space in order to crop corresponding frustum candidates, followed by point cloud based detectors for segmentation and detection. Frustum-ConvNet [38] generates a sequence of frustums that slide along the optical axis perpendicular to 2D image plane, which are summarized as frustum-wise features by PointNet blocks and arrayed as a BEV feature map for fully convolutional network. Works in this group resort to off-the-shelf 2D

detectors for prior knowledge, with 3D search space being dramatically reduced. Whereas, they also indicate a fatal risk that the failure of 2D detectors is bound to deteriorate all subsequent pipelines, which violates the original intention of providing redundancy during difficult conditions rather than just complementary and we therefore argue that sequential fusion based may not be well-suited for autonomous driving for the sake of safety.

### 3.3.2. Parallel Fusion based methods

Works in this group are decoupled from the single modality. Suppose that one of the branches is cut off, the network will still work if adjusted appropriately, say hyper-parameters. Depending on at what point these semantic representations from single modality are fused, a further subdivision for parallel fusion based can be identified: (1) early fusion, (2) deep fusion, and (3) late fusion.

*Early fusion* fuses different modalities at the data pre-processing point, which is not commonly used in light of the big noise of feature alignment at low level semantics. PointPainting [12] takes as inputs raw point clouds, compounded by segmentation scores that are predicted from the associated images, with the help of an independent segmentation network. *Deep fusion* fuses different modalities at the intermediate point. MV3D [30] takes as inputs the bird-view, front view, and images, fusing intermediate convolution features at stage two. AVOD [31] further extends fusion strategy to stage one to enrich more informative semantics for proposal generation. Observe that works [30, 31] fuse futures in a roi-wise manner, ContFuse [32] further fuses multi-scale convolutional features via continuous convolution in a pixel-wise fashion. Notice that pixel-wise fusion is vulnerable to faraway objects due to the difficulty of finding corresponding pixels for the sparse LiDAR signals at long ranges, MMF [81] exploits multiple related subtasks (*e.g.*, ground estimation, depth completion, and 2D object detection *etc.*) to enhance the learning of cross-modality fusion. Note that existing efforts treat semantics from different modalities equally, 3D-CVF [33] employs attention mechanism to adaptively fuse semantics from point clouds and the associated images. *Late fusion* fuses the outputs of different modalities at the decision point. CLOCs [34] fuses the outputs of 2D and 3D detections in decision level via exploiting the consistency of geometries and semantics.

### 3.3.3. Summary

Fig. 8 illustrates the evolution of multimodal fusion based. What to use. Consider the most commonly used sensors for autonomous driving: camera and LiDAR. Preceding works [80, 38, 12, 30, 31, 81, 33, 34, 32] all take images and point clouds as inputs. How to fuse. Efforts have been made to align semantics of different levels at different scales. Works [30, 31, 80, 38, 34] exploit roi-wise fusion. Works [12, 81, 33, 32] leverage point-wise fusion. When to use. Cross-modality fusion can happen at any time during the forward propagation: say early fusion [12], deep fusion [30, 32, 31, 81, 33], and late fusion [34]. In retrospect of

the evolution of this group, we notice that multimodal fusion based still lags far behind its point cloud based counterparts. We attribute the performance gap to the difficulty of semantic alignments. First, semantics from images and point clouds are heterogeneous as they are presented in different views [33] (*i.e.*, camera view vs. real 3D view). Consequently, finding point-wise correspondences is difficult as the transformation from LiDAR points to image pixels is a many-to-one mapping and vice versa. Such an ambiguity is referred as “feature blurring” in PointPainting [12]. Second, images are arrayed in dense grids, while point clouds are distributed in sparse points. To what extent semantics are aligned by forcing these two modalities to have the same size for aggregation remains untouched. Finally, operations that we use to crop and resize features may not be as accurate as expected [34].

## 4. EVALUATION

This section holds an apples-to-apples comparison of the state-of-the-arts on the widely used KITTI dataset, more recent nuScenes and Waymo dataset (Sec. 4.1). A case study based on fifteen selected models is given in Sec. 4.2, with respect to runtime analysis (Sec. 4.2.2), error analysis (Sec. 4.2.3), and robustness analysis (Sec. 4.2.4).

### 4.1. Comprehensive Comparison of the State-of-the-Arts

Table 6 summarizes the 3D detection performance on the KITTI Dataset. Image based methods have currently trailed the performance of point clouds counterparts thus far, which should be ascribed to depth ambiguity. Point clouds based methods still predominate KITTI benchmark due to low latency and high accuracy by resorting to 3D sparse convolution. Multimodal fusion based methods are closing the gap with point clouds counterparts somewhat. As mentioned above, fusing these two modalities together is non-trivial due to view misalignment. Noticeably, on the one hand, monocular or stereo cameras indeed bring extra source information as a supplementary, which circumvent the risk of over-reliance on a single sensor. On the other hand, multiple sensors hinder the runtime and make it hard to deploy given the requirement of continuous synchronization. Table 7 and Table 8 present the 3D detection performance on the more recent nuScenes dataset and Waymo dataset, respectively. Although works that report performances on nuScenes or Waymo are not as common as KITTI currently, assessing the effectiveness of detectors on these two large-scale datasets shall be necessary in the foreseeable future, regardless of dataset size or diversity as evidenced in Table 2.

### 4.2. Case Study

#### 4.2.1. Experimental Setup

Fifteen models are selected from the surveyed works depending on whether the official source code and the corresponding pretrained parameters are available. Notice that all

**Table 6**

**Comparisons of the state-of-the-art 3D detection  $AP_{3D}|_{R_{40}}$  on KITTI *test split*, by submitting to official test server. All these methods follow the official KITTI evaluation protocol, i.e. the rotated  $IoU_{3D}$  of 0.7, 0.5, and 0.5 is for the categories of *Car*, *Cyclist*, and *Pedestrian*, respectively. ‘-’ means the results are unavailable.**

| Input               | Methods                  | Speed (fps)            | Cars  |          |       | Pedestrians |          |       | Cyclists |          |       |       |
|---------------------|--------------------------|------------------------|-------|----------|-------|-------------|----------|-------|----------|----------|-------|-------|
|                     |                          |                        | Easy  | Moderate | Hard  | Easy        | Moderate | Hard  | Easy     | Moderate | Hard  |       |
| Images              | Deep MANTA [15]          | -                      | -     | -        | -     | -           | -        | -     | -        | -        | -     |       |
|                     | Deep3DBox [16]           | -                      | -     | -        | -     | -           | -        | -     | -        | -        | -     |       |
|                     | Mono3D [44]              | -                      | 2.53  | 2.31     | 2.31  | -           | -        | -     | -        | -        | -     |       |
|                     | GS3D [17]                | -                      | 4.47  | 2.90     | 2.47  | -           | -        | -     | -        | -        | -     |       |
|                     | Result-lifting           | Mono3D-PLiDAR [41]     | -     | 1.76     | 7.50  | 6.10        | -        | -     | -        | -        | -     | -     |
|                     |                          | Monodle [21]           | 25    | 17.23    | 12.26 | 10.29       | 9.64     | 6.55  | 5.44     | 4.59     | 2.66  | 2.45  |
|                     |                          | MonoRCNN [19]          | -     | 18.36    | 12.65 | 10.03       | -        | -     | -        | -        | -     | -     |
|                     |                          | MonoFlex [69]          | -     | 19.94    | 13.89 | 12.07       | 9.43     | 6.31  | 5.26     | 4.17     | 2.35  | 2.04  |
|                     |                          | DD3D [20]              | -     | 23.19    | 16.87 | 14.36       | 16.64    | 11.04 | 9.38     | 7.52     | 4.79  | 4.22  |
|                     |                          | Stereo R-CNN [18]      | -     | 47.58    | 30.23 | 23.72       | -        | -     | -        | -        | -     | -     |
|                     |                          | YOLOStereo3D [22]      | 10    | 65.68    | 41.25 | 30.42       | 28.49    | 19.75 | 16.48    | -        | -     | -     |
|                     | Feature-lifting          | 3DOP [47, 66]          | -     | -        | -     | -           | -        | -     | -        | -        | -     | -     |
|                     |                          | MF3D [45]              | -     | -        | -     | -           | -        | -     | -        | -        | -     | -     |
|                     |                          | OFT-Net[70]            | -     | 2.50     | 3.28  | 2.27        | -        | -     | -        | -        | -     | -     |
|                     |                          | Mono3D-PLiDAR [41]     | -     | 10.76    | 7.50  | 6.10        | -        | -     | -        | -        | -     | -     |
|                     |                          | CaDDN [74]             | -     | 19.17    | 13.41 | 11.46       | 12.87    | 8.14  | 6.76     | 7.00     | 3.41  | 3.30  |
|                     |                          | Pseudo-LiDAR [71]      | -     | 54.53    | 34.05 | 28.25       | -        | -     | -        | -        | -     | -     |
|                     |                          | Pseudo-LiDAR++ [73]    | -     | 61.11    | 42.43 | 36.99       | -        | -     | -        | -        | -     | -     |
|                     |                          | Pseudo-LiDAR E2E [75]  | -     | 64.75    | 43.92 | 38.14       | -        | -     | -        | -        | -     | -     |
|                     |                          | DSGN [4]               | -     | 73.50    | 52.18 | 45.14       | 20.53    | 15.55 | 14.15    | 27.76    | 18.17 | 16.21 |
| LIGA-Stereo [72]    |                          | -                      | 81.39 | 64.66    | 57.22 | 40.46       | 30.00    | 27.07 | 54.44    | 36.86    | 32.06 |       |
| Point clouds        | VeloFCN [48]             | 1.0                    | -     | -        | -     | -           | -        | -     | -        | -        | -     |       |
|                     | PIXOR [49]               | 28.6                   | -     | -        | -     | -           | -        | -     | -        | -        | -     |       |
|                     | HVNet [57]               | 31                     | -     | -        | -     | -           | -        | -     | -        | -        | -     |       |
|                     | VoxelNet[24]             | 2.0                    | 77.82 | 64.17    | 57.51 | -           | -        | -     | -        | -        | -     |       |
|                     | CenterPoint [23]         | -                      | 81.17 | 73.96    | 69.48 | 47.25       | 39.28    | 36.78 | 73.04    | 56.67    | 50.60 |       |
|                     | PointPillars [6]         | 62.0                   | 82.58 | 74.31    | 68.99 | 51.45       | 41.92    | 38.89 | 77.10    | 58.65    | 51.92 |       |
|                     | TANet [76]               | -                      | 84.39 | 75.94    | 68.82 | 53.72       | 44.34    | 40.49 | 75.70    | 59.44    | 52.53 |       |
|                     | SECOND [25]              | 26.3                   | 84.65 | 75.96    | 68.71 | -           | -        | -     | -        | -        | -     |       |
|                     | Voxel-FPN [55]           | 50                     | 85.48 | 76.70    | 69.44 | -           | -        | -     | -        | -        | -     |       |
|                     | Part-A <sup>2</sup> [56] | 12.5                   | 87.81 | 78.49    | 73.51 | 53.10       | 43.35    | 40.06 | 79.17    | 63.52    | 56.93 |       |
|                     | CIA-SSD [26]             | 32                     | 89.59 | 80.28    | 72.87 | -           | -        | -     | -        | -        | -     |       |
|                     | Voxel R-CNN [28]         | 25.2                   | 90.90 | 81.62    | 77.06 | -           | -        | -     | -        | -        | -     |       |
|                     | CT3D [29]                | -                      | 87.83 | 81.77    | 77.16 | -           | -        | -     | -        | -        | -     |       |
|                     | VoTr [5]                 | -                      | 89.90 | 82.09    | 79.14 | -           | -        | -     | -        | -        | -     |       |
|                     | SPG [77]                 | -                      | 90.50 | 82.13    | 78.90 | -           | -        | -     | -        | -        | -     |       |
|                     | SE-SSD [27]              | 32                     | 91.49 | 82.54    | 77.15 | -           | -        | -     | -        | -        | -     |       |
|                     | BtcDet [10]              | -                      | 90.64 | 82.86    | 78.09 | -           | -        | -     | 82.81    | 68.68    | 61.81 |       |
|                     | Point based              | PointRCNN [40]         | 10    | 86.96    | 75.64 | 70.70       | 47.98    | 39.37 | 36.01    | 74.96    | 58.82 | 52.53 |
|                     |                          | Point-GNN [52]         | 1.7   | 88.33    | 79.47 | 72.29       | 51.92    | 43.77 | 40.14    | 78.60    | 63.48 | 57.08 |
|                     | Point-voxel based        | 3DSSD [51]             | 25.0  | 88.36    | 79.57 | 74.55       | 54.64    | 44.27 | 40.23    | 82.48    | 64.10 | 56.90 |
| Fast PointRCNN [78] |                          | 16.7                   | 85.29 | 77.40    | 70.24 | -           | -        | -     | -        | -        | -     |       |
| STD [50]            |                          | 12.5                   | 87.95 | 79.71    | 75.09 | 53.29       | 42.47    | 38.35 | 78.69    | 61.59    | 55.30 |       |
| SA-SSD [53]         |                          | 25.0                   | 88.75 | 79.79    | 74.16 | -           | -        | -     | -        | -        | -     |       |
| PV-RCNN [39]        |                          | 12.5                   | 90.25 | 81.43    | 76.82 | 52.17       | 43.29    | 40.29 | 78.60    | 63.71    | 57.65 |       |
| BADet [79]          |                          | 7.1                    | 89.28 | 81.61    | 76.58 | -           | -        | -     | -        | -        | -     |       |
| Sequential fusion   |                          | Frustum-PointNets [80] | 5.9   | 82.19    | 69.79 | 60.59       | 50.53    | 42.15 | 38.08    | 72.27    | 56.12 | 49.01 |
|                     | Frustum-Convnet [38]     | 2.1                    | 87.36 | 76.39    | 66.69 | 52.16       | 43.38    | 38.80 | 81.98    | 65.07    | 56.54 |       |
| Multimodal          | MV3D [30]                | 2.8                    | 74.97 | 63.63    | 54.00 | -           | -        | -     | -        | -        | -     |       |
|                     | AVOD [31]                | 12.5                   | 76.39 | 66.47    | 60.23 | 36.10       | 27.86    | 25.76 | 57.19    | 42.08    | 38.29 |       |
|                     | ContFuse [32]            | 16.7                   | 83.68 | 68.78    | 61.67 | -           | -        | -     | -        | -        | -     |       |
|                     | PointPainting [12]       | 2.5                    | 82.11 | 71.70    | 67.08 | 50.32       | 40.97    | 37.87 | 77.63    | 63.78    | 55.89 |       |
|                     | MMF [81]                 | 12.5                   | 88.40 | 77.43    | 70.22 | -           | -        | -     | -        | -        | -     |       |
|                     | 3D-CVF [33]              | -                      | 89.20 | 80.05    | 73.11 | -           | -        | -     | -        | -        | -     |       |
|                     | CLOCs [34]               | -                      | 88.94 | 80.67    | 77.15 | -           | -        | -     | -        | -        | -     |       |

**Table 7**

**Comparisons of the state-of-the-art 3D detection on nuScenes *test set*. “TC” and “Cons.Veh.” denote traffic cone and construction vehicle respectively.**

| Methods            | mAP  | NDS  | Car  | Truck | Bus  | Trailer | CV   | Pedestrian | Motorcycle | Bicycle | TC   | Barrier |
|--------------------|------|------|------|-------|------|---------|------|------------|------------|---------|------|---------|
| PointPillars [6]   | 30.5 | 45.3 | 68.4 | 23.0  | 28.2 | 23.4    | 4.1  | 59.7       | 27.4       | 1.1     | 30.8 | 38.9    |
| PointPainting [12] | 46.4 | 58.1 | 77.9 | 35.8  | 36.2 | 37.3    | 15.8 | 73.3       | 41.5       | 24.1    | 62.4 | 60.2    |
| CenterPoint [23]   | 58.0 | 65.5 | 84.6 | 51.0  | 60.2 | 53.2    | 17.5 | 83.4       | 53.7       | 28.7    | 76.7 | 70.9    |

**Table 8**  
Comparisons of the state-of-the-art 3D detection on Waymo *val* set.

| Methods          | LEVEL_1     | LEVEL_2     | LEVEL_1 3D mAP/mAPH by Distance |             |             |
|------------------|-------------|-------------|---------------------------------|-------------|-------------|
|                  | 3D mAP/mAPH | 3D mAP/mAPH | 0-30m                           | 30-50m      | 50m-Inf     |
| PointPillars [6] | 63.30/62.70 | 55.20/54.70 | 84.90/84.40                     | 59.20/58.60 | 35.80/35.20 |
| Voxel R-CNN [28] | 75.59/-     | 66.59/-     | 92.49/-                         | 74.09/-     | 53.15/-     |
| SECOND [25]      | 67.94/67.28 | 59.46/58.88 | 88.10/87.46                     | 65.31/64.61 | 40.36/39.57 |
| PV-RCNN [39]     | 71.69/71.16 | 64.21/63.70 | 91.83/91.37                     | 69.99/69.37 | 46.26/45.41 |
| VoTr [5]         | 74.95/74.25 | 65.91/65.29 | 92.28/91.73                     | 73.36/72.56 | 51.09/50.01 |
| BtcDet [10]      | 78.58/78.06 | 70.10/69.61 | 96.11/-                         | 77.64/-     | 54.45/-     |

**Table 9**

**Runtime Analysis.** Numbers in 1<sup>st</sup> row indicate 3D detection on moderate difficulty for *Car* category sorted in ascending order by  $AP_{3D}|_{R_{11}}$ .

| Methods                | CaDDN[74] | PointPillars[6] | TANet[76] | Point-GNN[52] | SECOND[25] | PointRCNN[78] | 3DSSD[51] | Part-A <sup>2</sup> [56] | SA-SSD[53] | CIA-SSD[26] | PV-RCNN[39] | Voxel R-CNN[28] | CT3D[29] | BADet[79] | SE-SSD[27] |
|------------------------|-----------|-----------------|-----------|---------------|------------|---------------|-----------|--------------------------|------------|-------------|-------------|-----------------|----------|-----------|------------|
| AP                     | 19.19     | 77.28           | 77.54     | 78.34         | 78.62      | 77.38         | 79.23     | 79.38                    | 79.80      | 79.74       | 83.56       | 84.54           | 85.47    | 86.21     | 86.25      |
| FLOPS                  | 27.33G    | 3.72G           | 13.54G    | 4.45M         | 3.49G      | 1.53G         | 31.44G    | 2.95G                    | 155.48G    | 24.48G      | 2.97G       | 0.87G           | 3.17G    | 158.74G   | 24.45G     |
| Params                 | 67.55M    | 4.83M           | 6.5M      | 1.49M         | 5.33M      | 4.04M         | 7.56M     | 63.81M                   | 5.34M      | 3.81M       | 13.12M      | 7.59M           | 7.83M    | 5.79M     | 3.81M      |
| FPS <sub>default</sub> | 3.17      | 23.85           | 14.36     | 1.89          | 15.00      | 6.25          | 10.03     | 9.51                     | 20.50      | 33.40       | 7.19        | 16.97           | 7.13     | 7.10      | 32.40      |
| FPS <sub>unified</sub> | 3.17      | 23.98           | 14.36     | -             | 15.10      | 6.27          | 10.09     | 9.47                     | 20.40      | 23.90       | 7.20        | 17.28           | 8.21     | 7.20      | 23.70      |

our experiments are conducted on KITTI dataset by directly reloading official pretrained parameters with default settings. We follow KITTI protocol to evaluate on the *val* split for the most important *Car* category of moderate difficulty based on  $AP_{3D}|_{R_{11}}$  metric with an IoU threshold 0.7. For the ease of reproducibility, all materials are available online: <https://github.com/rui-qian/SoTA-3D-Object-Detection>.

#### 4.2.2. Runtime Analysis

To assess the real latency of detectors, we report runtime analysis in Table 9. Instead of just citing the numbers claimed in the papers, we conduct new experiments by ourselves. We argue that it is necessary as these numbers are obtained under different hardware resources in various settings. Some of them may use Tesla P40 GPU (*e.g.*, Fast PointRCNN [78]) whereas others may use TITAN Xp GPU (*e.g.*, CIA-SSD [26]). Some of them may ignore the time of data processing while others may use multiple process. Hence, directly comparing against these numbers inevitably leads to controversy. By contrast, we report two versions of runtime on a single GTX 1080Ti GPU, termed as FPS<sub>default</sub> and FPS<sub>unified</sub>. FPS<sub>default</sub> means the numbers are obtained with official settings, which are almost consistent with the ones claimed in the papers. FPS<sub>default</sub> reveals that our environment can reproduce the claimed results. FPS<sub>unified</sub> justifies all models in a unified criteria with 1 batch size and 4 multiple processes, fully eliminating other irrelevant factors. Table 9 indicates CIA-SSD [26] and SE-SSD [27] drop a lot under our paradigm. Considering that FLOPS is hardware independent, we also provide their FLOPS in the 2<sup>nd</sup> row that have never been reported before among existing surveys to the best of our knowledge. SE-SSD [27] shows a superior performance of speed-accuracy tradeoff, as evidenced by the 1<sup>st</sup> and 2<sup>nd</sup> rows of Table 9.

#### 4.2.3. Error Analysis

To identify the key parameters affecting the performance of detectors, we report error analysis in Table 10. As mentioned in Sec. 2.1, we adopt 7 parameters for an oriented 3D bounding box. These 7 parameters are treated equally when we regress variables. However, what mainly restricts 3D detection performance remains unexplored largely. Inspired by Monodle [21], we therefore conduct an errors analysis by replacing part of predicted 3D bounding box parameters with their corresponding ground truth values. A prediction will be assigned with a ground truth if the ratio of overlapping area exceeds a certain threshold. We set 0.6 in this paper. As shown in Table 10, we achieve a significant  $AP_{3D}$  gain among all selected models in the 2<sup>nd</sup> row if the predicted 3D location is replaced by ground truth, where the maximum gain is 13.19%. According to Table 10, we observe that 3D location error plays the leading role of error contribution, followed by depth and 3D size error.

#### 4.2.4. Robustness Analysis

To understand to what extent detectors are resilient to LiDAR sparsity, we report robustness analysis in Table 11. As mentioned in Sec. 2.2, LiDAR, typically an HDL-64E Velodyne LiDAR, is several orders of magnitude more expensive than camera, which leads to an exorbitant cost for deployment. Therefore, resorting to a less dense point cloud for 3D detection is encouraging. In this paper, we use algorithms proposed in Pseudo-LiDAR++ [73] to sparsify KITTI LiDAR signals from 64 to 32, 16, 8 accordingly. As shown in Table 11, Point-GNN [52], SECOND [25], 3DSSD [51] Part-A<sup>2</sup> [56], SA-SSD [53] and CIA-SSD [26] maintain a reasonable accuracy when LiDAR signals are reduced from 64 to 32. We also observe an obvious performance drop among all models when LiDAR signals are reduced from 32 to 16.

**Table 10**

**Error Analysis.** We replace the predicted 3D bounding boxes partially with their corresponding ground truth values. Numbers in 1<sup>st</sup> row indicate 3D detection on moderate difficulty for *Car* category sorted in ascending order by  $AP_{3D}|_{R_{11}}$ .

| Methods           | CaDDN[74] | PointPillars[6] | TANet[76] | Point-GNN[52] | SECOND[25] | PointRCNN[78] | 3DSSD[51] | Part-A <sup>2</sup> [56] | SA-SSD[53] | CIA-SSD[26] | PV-RCNN[39] | Voxel R-CNN[28] | CT3D[29] | BADet[79] | SE-SSD[27] |
|-------------------|-----------|-----------------|-----------|---------------|------------|---------------|-----------|--------------------------|------------|-------------|-------------|-----------------|----------|-----------|------------|
| baseline          | 19.19     | 77.28           | 77.54     | 78.34         | 78.62      | 77.38         | 79.23     | 79.38                    | 79.80      | 79.74       | 83.56       | 84.54           | 85.47    | 86.21     | 86.25      |
| w/ gt 3D location | 32.20     | 87.20           | 87.08     | 88.79         | 88.31      | 87.31         | 88.72     | 88.11                    | 89.45      | 89.15       | 88.40       | 88.53           | 88.65    | 89.06     | 89.09      |
| w/ gt depth       | 30.32     | 78.68           | 78.66     | 79.02         | 82.59      | 78.08         | 85.97     | 83.47                    | 86.88      | 86.58       | 84.18       | 84.80           | 86.19    | 86.70     | 86.56      |
| w/ gt 3D size     | 19.36     | 78.42           | 77.88     | 78.80         | 79.16      | 78.07         | 85.69     | 83.36                    | 86.26      | 79.80       | 83.95       | 84.47           | 85.86    | 86.34     | 86.25      |
| w/ gt orientation | 19.38     | 77.52           | 77.73     | 78.46         | 78.72      | 77.56         | 79.24     | 79.42                    | 86.23      | 79.78       | 83.68       | 84.32           | 85.58    | 86.28     | 86.33      |

**Table 11**

**Robustness Analysis.** Numbers in 1<sup>st</sup> row indicate 3D detection on moderate difficulty for *Car* category sorted in ascending order by  $AP_{3D}|_{R_{11}}$ .

| LiDAR beams   | CaDDN[74] | PointPillars[6] | TANet[76] | Point-GNN[52] | SECOND[25] | PointRCNN[78] | 3DSSD[51] | Part-A <sup>2</sup> [56] | SA-SSD[53] | CIA-SSD[26] | PV-RCNN[39] | Voxel R-CNN[28] | CT3D[29] | BADet[79] | SE-SSD[27] |
|---------------|-----------|-----------------|-----------|---------------|------------|---------------|-----------|--------------------------|------------|-------------|-------------|-----------------|----------|-----------|------------|
| 64 (Baseline) | n.a.      | 77.28           | 77.54     | 78.34         | 78.62      | 77.38         | 79.23     | 79.38                    | 79.80      | 79.74       | 83.56       | 84.54           | 85.47    | 86.21     | 86.25      |
| 32            | n.a.      | 68.32           | 67.09     | 75.85         | 75.58      | 70.11         | 77.27     | 76.06                    | 76.10      | 76.32       | 77.60       | 76.43           | 78.15    | 76.47     | 76.11      |
| 16            | n.a.      | 57.26           | 46.23     | 58.75         | 59.09      | 57.06         | 59.65     | 59.41                    | 57.06      | 56.22       | 63.34       | 58.61           | 60.82    | 57.39     | 57.27      |
| 8             | n.a.      | 32.49           | 24.25     | 32.56         | 31.58      | 34.26         | 31.53     | 34.22                    | 30.55      | 29.50       | 35.80       | 34.05           | 38.96    | 30.29     | 31.11      |

## 5. RETROSPECT AND PROSPECT

### 5.1. Concluding Remarks

This research presents a survey on 3D object detection in the context of autonomous driving, for the sake of holding potential interest and relevance for 3D visual data analysis, and consequently facilitating a mature taxonomy for the interested audience to either form a structured picture quickly or start their own research from scratch easily.

Depending on what modalities are actually fed into networks during inference, we structure existing literature along three dimensions: (1) image based, (2) point cloud based, and (3) multimodal fusion based, allowing us to clarify the key challenge that stems from nature properties of modality itself. We attribute challenges to visual appearance recovery in the absence of depth information from images, representation learning from partially occluded unstructured point clouds, and semantic alignments over heterogeneous features from cross modalities. Having taken a glimpse of evolution of 3D object detection, an apples-to-apples comparison of the state-of-the-arts is presented. We notice that there is a growing tendency for point clouds based methods to further broaden accuracy advantages over their image based counterparts. A case study on the basis of fifteen selected models is conducted to justify the state-of-the-arts, in terms of runtime analysis, error analysis, and robustness analysis. We observe that what mainly restricts the performance of detection is 3D location error.

In retrospect of what has been done, we draw concluding remarks for the surveyed works. *Seminal works are profound.* VoxelNet [24] takes the first lead to propose an end-to-end trainable network via learning an informative 3D volumetric representation instead of manual feature engineering as most previous works do. Subsequently SECOND [25] exploits sparse convolution operation to only convolve against

non-empty voxels, mitigating unnecessary computation incurred by unavailing zero-padding voxels in light of the sparsity of point clouds. VoxelNet-like pipelines incorporated with sparse convolution have been continuously inspiring their successors ever since. *Auxiliary network learning is artful.* SA-SSD [53] explores an auxiliary network to endow voxel features with structure-aware capability. SE-SSD [27] and LIGA-Stereo [72] exploit distilled intermediate representation and beyond from a teacher network, which subtly differs from multi-task learning [81] as they are detachable in the phase of inference artfully. *Transformers are promising.* A new paradigm of applying transformers [94] to object detection has recently evolved as the effectiveness of acquiring long-range interactions for faraway objects and learning spatial context-aware dependencies for false negatives, wherein CT3D [29] and VoTr [5] have achieved a remarkable performance gain.

### 5.2. Reflections on Future Work

In prospect of what remains to be done, we identify the avenues for future work. *Safety is nothing without security.* Reasoning under uncertainty matters. 3D visual data holds potentials of uncertainty by nature, regardless of LiDAR signals or images. Therefore, all critical decisions that autonomous driving system makes should be under the guidance of uncertainty constraints, *e.g.* the system needs to recognize a higher uncertainty in foggy weather than that under sunny conditions. Whereas, how to trade off the concerns between “minimizing risks” and “completing tasks” is still an imperative yet largely unexplored problem [36]. We notice MonoFlex [69] and MonoRCNN [19] take certainty into account in their works among our surveyed literature. Adversarial attack matters. Note that modern autonomous driving system relies heavily on deep learning. Whereas, deep learning methods have already been proved to be



vulnerable to visually imperceptible perturbations [97] and therefore poses an inherent security risk. With sabotage and threats of blind spots on the rise, adversarial attacks on 3D object detection should arouse enough attention of 3D vision community. *Rethink what we have on hand*. Representation matters. Whether the data representation or the discrepancy in depth estimation mainly results in the performance gap between images and LiDAR signals remains open. Pseudo-LiDAR series [71, 73] break the stereotype that images can only be leveraged in the form of 2D representation, which remind us to rethink the off-the-shelf 3D sensor data. DSGN [4] and CaDDN [74] try to learn a latent depth distribution for an intermediate 3D representation directly instead of resorting to empirical geometric constraints. These findings also provide a new idea for multimodal fusion as they remove the requirement of point-wise correspondences retrieval for semantic alignments. Shape learning matters. A point cloud is self-occluded by nature, which makes itself 2.5D in practice. Shape learning from partially occluded sparse point clouds seems to be necessary, as evidenced in SE-SSD [27], SPG [77], and BtcDet [10]. In summary, we hope that this survey will shed light on 3D object detection and inspire more follow-up literature in this area.

**Acknowledgements.** This work was supported by the National Natural Science Foundation of China under Grant 62172420 and Beijing Natural Science Foundation under Grant 4202033.

## References

- [1] S. International, Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles, accessed in 2021, [https://www.sae.org/standards/content/j3016\\_202104](https://www.sae.org/standards/content/j3016_202104).
- [2] J. Chang, Y. Chen, Pyramid stereo matching network, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2018, pp. 5410–5418.
- [3] H. Fu, M. Gong, C. Wang, K. Batmanghelich, D. Tao, Deep ordinal regression network for monocular depth estimation, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2018, pp. 2002–2011.
- [4] Y. Chen, S. Liu, X. Shen, J. Jia, DSGN: deep stereo geometry network for 3D object detection, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2020, pp. 12533–12542.
- [5] J. Mao, Y. Xue, M. Niu, H. Bai, J. Feng, X. Liang, H. Xu, C. Xu, Voxel transformer for 3D object detection, in: International Conference on Computer Vision, IEEE, 2021, pp. 3164–3173.
- [6] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, O. Beijbom, PointPillars: Fast encoders for object detection from point clouds, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2019, pp. 12697–12705.
- [7] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, D. Anguelov, Scalability in perception for autonomous driving: Waymo open dataset, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2020, pp. 2443–2451.
- [8] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the KITTI vision benchmark suite, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2012, pp. 3354–3361.
- [9] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, B. Chen, PointCNN: Convolution on X-transformed points, in: Advances in Neural Information Processing Systems, 2018, pp. 828–838.
- [10] Q. Xu, Y. Zhong, U. Neumann, Behind the curtain: Learning occluded shapes for 3D object detection, in: AAAI Conference on Artificial Intelligence, Vol. 36, AAAI Press, 2022.
- [11] Z. Liu, H. Tang, Y. Lin, S. Han, Point-Voxel CNN for efficient 3D deep learning, in: Advances in Neural Information Processing Systems, 2019, pp. 963–973.
- [12] S. Vora, A. H. Lang, B. Helou, O. Beijbom, Pointpainting: Sequential fusion for 3D object detection, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2020, pp. 4603–4611.
- [13] M. M. Rahman, Y. Tan, J. Xue, K. Lu, Recent advances in 3D object detection in the era of deep neural networks: A survey, in: Transactions on Image Processing, Vol. 29, IEEE, 2019, pp. 2947–2962.
- [14] E. Arnold, O. Y. Al-Jarrah, M. Dianati, S. Fallah, D. Oxtoby, A. Mouzakitis, A survey on 3D object detection methods for autonomous driving applications, in: Transactions on Intelligent Transportation Systems, Vol. 20, IEEE, 2019, pp. 3782–3795.
- [15] F. Chabot, M. Chaouch, J. Rabarisoa, C. Teulière, T. Chateau, Deep MANTA: A coarse-to-fine many-task network for joint 2D and 3D vehicle analysis from monocular image, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2017, pp. 1827–1836.
- [16] A. Mousavian, D. Anguelov, J. Flynn, J. Kosecka, 3D bounding box estimation using deep learning and geometry, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2017, pp. 5632–5640.
- [17] B. Li, W. Ouyang, L. Sheng, X. Zeng, X. Wang, GS3D: An efficient 3D object detection framework for autonomous driving, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2019, pp. 1019–1028.
- [18] P. Li, X. Chen, S. Shen, Stereo R-CNN based 3D object detection for autonomous driving, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2019, pp. 7644–7652.
- [19] X. Shi, Q. Ye, X. Chen, C. Chen, Z. Chen, T.-K. Kim, Geometry-based distance decomposition for monocular 3D object detection, in: International Conference on Computer Vision, IEEE, 2021, pp. 15172–15181.
- [20] D. Park, R. Ambrus, V. Guizilini, J. Li, A. Gaidon, Is Pseudo-Lidar needed for monocular 3D object detection?, in: International Conference on Computer Vision, IEEE, 2021, pp. 3142–3152.
- [21] X. Ma, Y. Zhang, D. Xu, D. Zhou, S. Yi, H. Li, W. Ouyang, Delving into localization errors for monocular 3D object detection, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2021, pp. 4721–4730.
- [22] Y. Liu, L. Wang, M. Liu, YOLOStereo3D: A step back to 2D for efficient stereo 3D detection, in: International Conference on Robotics and Automation, IEEE, 2021, pp. 13018–13024.
- [23] T. Yin, X. Zhou, P. Krähnenbühl, Center-based 3D object detection and tracking, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2021, pp. 11784–11793.
- [24] Y. Zhou, O. Tuzel, VoxelNet: End-to-end learning for point cloud based 3D object detection, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2018, pp. 4490–4499.
- [25] Y. Yan, Y. Mao, B. Li, SECOND: sparsely embedded convolutional detection, in: Sensors, Vol. 18, Multidisciplinary Digital Publishing Institute, 2018, p. 3337.
- [26] W. Zheng, W. Tang, S. Chen, L. Jiang, C. Fu, CIA-SSD: confident iou-aware single-stage object detector from point cloud, in: AAAI Conference on Artificial Intelligence, AAAI Press, 2021, pp. 3555–3562.
- [27] W. Zheng, W. Tang, L. Jiang, C.-W. Fu, SE-SSD: Self-Ensembling single-stage object detector from point cloud, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2021, pp. 14494–14503.
- [28] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, H. Li, Voxel R-CNN: towards high performance voxel-based 3D object detection, in: AAAI

- Conference on Artificial Intelligence, AAAI Press, 2021, pp. 1201–1209.
- [29] H. Sheng, S. Cai, Y. Liu, B. Deng, J. Huang, X.-S. Hua, M.-J. Zhao, Improving 3D object detection with channel-wise transformer, in: International Conference on Computer Vision, IEEE, 2021, pp. 2743–2752.
- [30] X. Chen, H. Ma, J. Wan, B. Li, T. Xia, Multi-view 3D object detection network for autonomous driving, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2017, pp. 1907–1915.
- [31] J. Ku, M. Mozifian, J. Lee, A. Harakeh, S. L. Waslander, Joint 3D proposal generation and object detection from view aggregation, in: International Conference on Intelligent Robots and Systems, IEEE, 2018, pp. 1–8.
- [32] M. Liang, B. Yang, S. Wang, R. Urtasun, Deep continuous fusion for multi-sensor 3D object detection, in: European Conference on Computer Vision, Vol. 11220 of Lecture Notes in Computer Science, Springer, 2018, pp. 663–678.
- [33] J. H. Yoo, Y. Kim, J. S. Kim, J. W. Choi, 3D-CVF: Generating joint camera and lidar features using cross-view spatial feature fusion for 3D object detection, in: European Conference on Computer Vision, Springer, 2020, pp. 720–736.
- [34] S. Pang, D. Morris, H. Radha, CLOCs: Camera-LiDAR object candidates fusion for 3D object detection, in: International Conference on Intelligent Robots and Systems, IEEE, 2020, pp. 10386–10393.
- [35] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, M. Bennamoun, Deep learning for 3D point clouds: A survey, in: Transactions on Pattern Analysis and Machine Intelligence, Vol. 43, IEEE, 2020, pp. 4338–4364.
- [36] D. Feng, C. Haase-Schütz, L. Rosenbaum, H. Hertlein, C. Glaeser, F. Timm, W. Wiesbeck, K. Dietmayer, Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges, in: Transactions on Intelligent Transportation Systems, Vol. 22, IEEE, 2021, pp. 1341–1360.
- [37] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, O. Beijbom, nuScenes: A multimodal dataset for autonomous driving, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2020, pp. 11618–11628.
- [38] Z. Wang, K. Jia, Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection, in: International Conference on Intelligent Robots and Systems, 2019, pp. 1742–1749.
- [39] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, H. Li, PV-RCNN: Point-Voxel feature set abstraction for 3D object detection, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2020, pp. 10526–10535.
- [40] S. Shi, X. Wang, H. Li, PointRCNN: 3D object proposal generation and detection from point cloud, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2019, pp. 770–779.
- [41] X. Weng, K. Kitani, Monocular 3D object detection with pseudo-lidar point cloud, in: International Conference on Computer Vision Workshops, IEEE, 2019, pp. 857–866.
- [42] S. Song, J. Xiao, Deep sliding shapes for amodal 3D object detection in RGB-D images, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2016, pp. 808–816.
- [43] T. Mai, What are passive and active sensors?, accessed in 2017, [https://www.nasa.gov/directorates/heo/scan/communications/outreach/funfacts/txt\\_passive\\_active.html](https://www.nasa.gov/directorates/heo/scan/communications/outreach/funfacts/txt_passive_active.html).
- [44] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, R. Urtasun, Monocular 3D object detection for autonomous driving, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2016, pp. 2147–2156.
- [45] B. Xu, Z. Chen, Multi-level fusion based 3D object detection from monocular images, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2018, pp. 2345–2353.
- [46] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, R. Urtasun, 3D object proposals for accurate object class detection, in: Advances in Neural Information Processing Systems, 2015, pp. 424–432.
- [47] X. Chen, K. Kundu, Y. Zhu, H. Ma, S. Fidler, R. Urtasun, 3D object proposals using stereo imagery for accurate object class detection, in: Transactions on Pattern Analysis and Machine Intelligence, Vol. 40, IEEE, 2018, pp. 1259–1272.
- [48] B. Li, T. Zhang, T. Xia, Vehicle detection from 3D lidar using fully convolutional network, in: Robotics: Science and Systems, 2016.
- [49] B. Yang, W. Luo, R. Urtasun, PIXOR: Real-time 3D object detection from point clouds, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2018, pp. 7652–7660.
- [50] Z. Yang, Y. Sun, S. Liu, X. Shen, J. Jia, STD: Sparse-to-dense 3D object detector for point cloud, in: International Conference on Computer Vision, IEEE, 2019, pp. 1951–1960.
- [51] Z. Yang, Y. Sun, S. Liu, J. Jia, 3DSSD: Point-based 3D single stage object detector, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2020, pp. 11037–11045.
- [52] W. Shi, R. Rajkumar, Point-GNN: Graph neural network for 3D object detection in a point cloud, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2020, pp. 1708–1716.
- [53] C. He, H. Zeng, J. Huang, X. Hua, L. Zhang, Structure aware single-stage 3D object detection from point cloud, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2020, pp. 11870–11879.
- [54] Y. Zhou, P. Sun, Y. Zhang, D. Anguelov, J. Gao, T. Ouyang, J. Guo, J. Ngiam, V. Vasudevan, End-to-end multi-view fusion for 3D object detection in lidar point clouds, in: Conference on Robot Learning, PMLR, 2020, pp. 923–932.
- [55] H. Kuang, B. Wang, J. An, M. Zhang, Z. Zhang, Voxel-FPN: Multi-scale voxel feature aggregation for 3D object detection from LIDAR point clouds, in: Sensors, Vol. 20, 2020, p. 704.
- [56] S. Shi, Z. Wang, J. Shi, X. Wang, H. Li, From points to parts: 3D object detection from point cloud with part-aware and part-aggregation network, in: Transactions on Pattern Analysis and Machine Intelligence, Vol. 43, IEEE, 2021, pp. 2647–2664.
- [57] M. Ye, S. Xu, T. Cao, HVNet: Hybrid voxel network for lidar based 3D object detection, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2020, pp. 1628–1637.
- [58] A. Geiger, P. Lenz, C. Stillér, R. Urtasun, Vision meets robotics: The KITTI dataset, in: The International Journal of Robotics Research, Vol. 32, Sage Publications Sage UK, 2013, pp. 1231–1237.
- [59] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, S. B. Low, A. Jain, P. Ondruska, S. Omari, S. Shah, A. Kulkarni, A. Kazakova, C. Tao, L. Platinsky, W. Jiang, V. Shet., Lyft level 5 AV dataset 2019, 2019.
- [60] A. Patil, S. Malla, H. Gang, Y. Chen, The H3D dataset for full-surround 3D multi-object detection and tracking in crowded urban scenes, in: International Conference on Robotics and Automation, ICRA, IEEE, 2019, pp. 9552–9557.
- [61] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, D. Manocha, Trafficpredict: Trajectory prediction for heterogeneous traffic-agents, in: AAAI Conference on Artificial Intelligence, AAAI Press, 2019, pp. 6120–6127.
- [62] X. Huang, P. Wang, X. Cheng, D. Zhou, Q. Geng, R. Yang, The apolloscape open dataset for autonomous driving and its application, in: Transactions on Pattern Analysis and Machine Intelligence, Vol. 42, IEEE, 2020, pp. 2702–2719.
- [63] B. Halloran, P. Premaratne, P. J. Vial, Robust one-dimensional calibration and localisation of a distributed camera sensor network, in: Pattern Recognition, Elsevier, 2020, p. 107058.
- [64] F. Dornaika, Self-calibration of a stereo rig using monocular epipolar geometries, in: Pattern Recognition, Elsevier, 2007, pp. 2716–2729.
- [65] M. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, J. Hays, Argoverse: 3D tracking and forecasting with rich maps, in: Conference on Computer Vision and Pattern Recognition, IEEE, 2019, pp. 8748–8757.
- [66] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, R. Urtasun, 3D object proposals for accurate object class detection, in: Advances in Neural Information Processing Systems, 2015, pp. 424–432.

- [67] M. Everingham, L. V. Gool, C. K. I. Williams, J. M. Winn, A. Zisserman, The pascal visual object classes (VOC) challenge, *International Journal of Computer Vision* 88 (2) (2010) 303–338.
- [68] A. Simonelli, S. R. Bulò, L. Porzi, M. Lopez-Antequera, P. Kotschieder, Disentangling monocular 3D object detection, in: *International Conference on Computer Vision, IEEE, 2019*, pp. 1991–1999.
- [69] Y. Zhang, J. Lu, J. Zhou, Objects are different: Flexible monocular 3D object detection, in: *Conference on Computer Vision and Pattern Recognition, IEEE, 2021*, pp. 3289–3298.
- [70] T. Roddick, A. Kendall, R. Cipolla, Orthographic feature transform for monocular 3d object detection, in: *British Machine Vision Conference, BMVA Press, 2019*, p. 285.
- [71] Y. Wang, W.-L. Chao, D. Garg, B. Hariharan, M. Campbell, K. Q. Weinberger, Pseudo-lidar from visual depth estimation: Bridging the gap in 3D object detection for autonomous driving, in: *Conference on Computer Vision and Pattern Recognition, IEEE, 2019*, pp. 8445–8453.
- [72] X. Guo, S. Shi, X. Wang, H. Li, LIGA-Stereo: Learning lidar geometry aware representations for stereo-based 3D detector, in: *International Conference on Computer Vision, IEEE, 2021*, pp. 3153–3163.
- [73] Y. You, Y. Wang, W. Chao, D. Garg, G. Pleiss, B. Hariharan, M. E. Campbell, K. Q. Weinberger, Pseudo-lidar++: Accurate depth for 3D object detection in autonomous driving, in: *International Conference on Learning Representations, OpenReview.net, 2020*.
- [74] C. Reading, A. Harakeh, J. Chae, S. L. Waslander, Categorical depth distribution network for monocular 3D object detection, in: *Conference on Computer Vision and Pattern Recognition, IEEE, 2021*, pp. 8555–8564.
- [75] R. Qian, D. Garg, Y. Wang, Y. You, S. Belongie, B. Hariharan, M. Campbell, K. Q. Weinberger, W.-L. Chao, End-to-end pseudo-lidar for image-based 3D object detection, in: *Conference on Computer Vision and Pattern Recognition, IEEE, 2020*, pp. 5881–5890.
- [76] Z. Liu, X. Zhao, T. Huang, R. Hu, Y. Zhou, X. Bai, TANet: Robust 3D object detection from point clouds with triple attention, in: *AAAI Conference on Artificial Intelligence, AAAI Press, 2020*, pp. 11677–11684.
- [77] Q. Xu, Y. Zhou, W. Wang, C. R. Qi, D. Anguelov, SPG: Unsupervised domain adaptation for 3D object detection via semantic point generation, in: *International Conference on Computer Vision, IEEE, 2021*, pp. 15446–15456.
- [78] Y. Chen, S. Liu, X. Shen, J. Jia, Fast Point R-CNN, in: *International Conference on Computer Vision, IEEE, 2019*, pp. 9774–9783.
- [79] R. Qian, X. Lai, X. Li, BADet: Boundary-aware 3D object detection from point clouds, in: *Pattern Recognition, Vol. 125, Elsevier, 2022*, p. 108524.
- [80] C. R. Qi, W. Liu, C. Wu, H. Su, L. J. Guibas, Frustum pointnets for 3D object detection from RGB-D data, in: *Conference on Computer Vision and Pattern Recognition, IEEE, 2018*, pp. 918–927.
- [81] M. Liang, B. Yang, Y. Chen, R. Hu, R. Urtasun, Multi-task multi-sensor fusion for 3D object detection, in: *Conference on Computer Vision and Pattern Recognition, IEEE, 2019*, pp. 7345–7353.
- [82] R. B. Girshick, Fast R-CNN, in: *International Conference on Computer Vision, IEEE, 2015*, pp. 1440–1448.
- [83] K. He, X. Zhang, S. Ren, J. Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition, in: *European Conference on Computer Vision, Vol. 8691 of Lecture Notes in Computer Science, Springer, 2014*, pp. 346–361.
- [84] S. Ren, K. He, R. B. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, in: *Advances in Neural Information Processing Systems, 2015*, pp. 91–99.
- [85] J. Redmon, S. K. Divvala, R. B. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: *Conference on Computer Vision and Pattern Recognition, IEEE, 2016*, pp. 779–788.
- [86] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, A. C. Berg, SSD: single shot multibox detector, in: *European Conference on Computer Vision, Lecture Notes in Computer Science, 2016*, pp. 21–37.
- [87] T. Lin, P. Goyal, R. B. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: *International Conference on Computer Vision, IEEE, 2017*, pp. 2999–3007.
- [88] K. He, G. Gkioxari, P. Dollár, R. B. Girshick, Mask R-CNN, in: *International Conference on Computer Vision, IEEE, 2017*, pp. 2980–2988.
- [89] C. R. Qi, H. Su, K. Mo, L. J. Guibas, PointNet: Deep learning on point sets for 3D classification and segmentation, in: *Conference on Computer Vision and Pattern Recognition, IEEE, 2017*, pp. 652–660.
- [90] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: *International Conference on Learning Representations, OpenReview.net, 2017*.
- [91] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, in: *Nature, 2015*, pp. 436–444.
- [92] C. R. Qi, L. Yi, H. Su, L. J. Guibas, PointNet++: Deep hierarchical feature learning on point sets in a metric space, in: *Advances in Neural Information Processing Systems, 2017*, pp. 5099–5108.
- [93] X. Qi, R. Liao, J. Jia, S. Fidler, R. Urtasun, 3D graph neural networks for RGB-D semantic segmentation, in: *International Conference on Computer Vision, IEEE, 2017*, pp. 5209–5218.
- [94] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: *Advances in Neural Information Processing Systems, 2017*, pp. 5998–6008.
- [95] W. L. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: *Advances in Neural Information Processing Systems, 2017*, pp. 1024–1034.
- [96] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, J. M. Solomon, Dynamic graph CNN for learning on point clouds, in: *Transactions on Graphics, ACM, 2019*, pp. 1–12.
- [97] J. Tu, M. Ren, S. Manivasagam, M. Liang, B. Yang, R. Du, F. Cheng, R. Urtasun, Physically realizable adversarial examples for lidar object detection, in: *Conference on Computer Vision and Pattern Recognition, IEEE, 2020*, pp. 13713–13722.