

# Survey and Systematization of 3D Object Detection Models and Methods

Moritz Drobnetzky<sup>a</sup>, Jonas Friederich<sup>b</sup>, Bernhard Egger<sup>c</sup>, Patrick Zschech<sup>c,a</sup>

<sup>a</sup>*Technische Universität Dresden, Münchner Platz 3, 01187 Dresden, Germany*

<sup>b</sup>*Mærsk Mc-Kinney Møller Institute, University of Southern Denmark, Campusvej 55, 5230 Odense, Denmark*

<sup>c</sup>*Friedrich-Alexander-Universität Erlangen-Nürnberg, Schloßplatz 4, 91054 Erlangen, Germany*

## ABSTRACT

This paper offers a comprehensive survey of recent developments in 3D object detection covering the full pipeline from input data, over data representation and feature extraction to the actual detection modules. We include basic concepts, focus our survey on a broad spectrum of different approaches arising in the last ten years and propose a systematization which offers a practical framework to compare those approaches on the methods level.

## 1. Introduction

Gaining a high-level and three dimensional understanding of digital pictures is one of the major challenges in the field of artificial intelligence. Applications like augmented reality, autonomous driving and other robotic navigation systems are pushing research in this field faster than ever. Participating in real-life road traffic, self-driving vehicles need to gain an absolute understanding of their surroundings. Hence, the vehicle not only needs to recognize other road users and other objects, but also comprehend their pose and location to avoid collisions. This objective is well-known as 3D object detection (3DOD) (Arnold et al., 2019).

Meanwhile, 2D object detection (2DOD) has obtained impressive results in terms of precision and inference time, and is able to compete with or even surpass human vision (Zhao et al., 2019). However, to fully grasp the scene in a real 3D world, 2D recognition and detection results alone are no longer sufficient. 3DOD now extends this approach into three-dimensional space by adding the desired parameters of dimension and orientation of the object to the established location and classification results.

The literature volume for 3DOD has increased significantly over the past years (Fernandes et al., 2021; Friederich and Zschech, 2020). Against the backdrop of increasingly sophisticated 2DOD models, it is apparent that the focus of research is shifting to 3DOD as the necessary hardware in terms of sensors and computing units becomes increasingly available.

Since 3DOD is a steadily growing field of investigation, there are several promising approaches and trends, including a large pool of various design options for the object detection pipeline. Providing an overview about relevant approaches and seminal achievements may offer orientation and can help to initiate further development in the research community. For this reason, we present a comprehensive review of 3DOD models and methods with exemplary applications and aim to conceptualize the full range of 3DOD approaches along a multi-stage pipeline.

With our work, we complement related surveys in the field (e.g., Arnold et al., 2019; Guo et al., 2021; Fernandes et al., 2021), which often focus on a particular domain (e.g., autonomous driving), specific data input (e.g., point cloud data), or a certain set of methods (e.g., deep learning techniques).

To carry out our review, we investigated papers that were published in a period from 2012 to 2021. In total, our literature corpus comprises more than hundred papers which we examined in detail to provide a classification of all approaches. Throughout our review, we describe representative examples along the 3DOD pipeline, while highlighting seminal achievements.

This survey is structured as follows. In Section 2, we provide all relevant foundations and subsequently refer to related work in Section 3. Thereafter, the identified literature is discussed and analyzed in detail. This constitutes the main part of our survey, for which we propose a structured framework along the 3DOD pipeline in Section 4. The framework consists of several stages with corresponding design options, which will be examined in individual Sections 5–9 due to their thematic depth. Afterwards, we leverage our framework and classify the examined literature corpus in Section 10. Finally, we draw a conclusion of this work and give an outlook for future work in Section 11. For better readability, we include a list of all abbreviations at the end of the manuscript.

## 2. Foundations

To give orientation in the following sections, we introduce major concepts of computer vision and in particular of 3DOD that are regarded as background and foundational knowledge.

### 2.1. Object Detection

A core task in the field of computer vision is to recognize and classify objects in images. This general task can further be subdivided into several sub-tasks as summarized in Table 1.

Object recognition	Recognise/Classify a specific object type in an image
Object localisation	Localise an object in an image
Object detection	Localise and recognise objects in an image
Object segmentation	Determine which pixels belong to an object in an image
Pose estimation	Determine the pose of an object in an image

**Table 1. Different tasks in computer vision that focus on object instances**

Following this distinction, object detection is the fusion of object recognition and localization. In detail, the approach tries to simultaneously classify and localize specific object instances in an image (Zhao et al., 2019). Detected objects are classified and usually marked with bounding boxes. These are imaginary boxes that describe the objects of interest. They are defined as the coordinates of the rectangular border that fully encloses the target object.

Object detection can be considered as a supervised learning problem which defines the process of learning a function that can map input data to known targets based on a given set of training data (Bishop, 2006). For this task, different kinds of machine learning algorithms can be applied.

Conventional machine learning approaches require to first extract representative image features based on feature descriptors, such as Viola-Jones method (Viola and Jones, 2004), scale-invariant feature transform (SIFT) (Lowe, 2004) or histogram of oriented gradients (HOG) (Dalal and Triggs, 2005). Those are low-level features which are manually designed for the specific use case. On their basis, prediction models such as support vector machines (SVMs) can be trained to perform the object recognition task (Sager et al., 2021). However, the diversity of image objects and use cases in form of pose, illumination and background makes it difficult to manually create a robust feature descriptor that can describe all kinds of objects (Janiesch et al., 2021).

For this reason, recent efforts are increasingly directed towards the application of artificial neural networks with deep network architectures, broadly summarized under the term *deep learning* (LeCun et al., 2015). Deep neural networks are able to perform object recognition without having to manually define specific features in advance. Their multi-layered architecture allows them to be fed with high-dimensional raw input data and then automatically discover internal representations at different levels of abstraction that are needed for recognition and detection tasks (LeCun et al., 2015).

A common type of deep neural network architecture, which is widely adopted by the computer vision community, is that of *convolutional neural networks* (CNNs). Due to their nested design, they are able to process high-dimensional data that come in the form of multiple arrays, such as given by color images that are composed of arrays containing pixel intensities in different color channels (Liu et al., 2020). These techniques proved to be superior in 2DOD offering more complex and robust features, while being applicable on any use case (Liu et al.,

2020; LeCun et al., 2015).

## 2.2. 3D Vision and 3D Object Detection

3D vision aims to extend the previously discussed concepts of object detection by adding data of the third dimension. This leads on the one hand to six possible degrees of freedom (6DoF)<sup>1</sup> instead of three, and on the other hand, to an accompanying increase in the number of scenery configurations. While methods in 2D space are good for simple visual tasks, more sophisticated approaches are needed to improve, for instance, autonomous driving or robotics applications. The full understanding of the environment composed of real 3D objects implies the interpretation of scenes in which items may show up in absolutely discretionary positions and directions related to 6DoF. This requires a substantial amount of computing power and increases the complexity of the performed operations (Davies, 2012).

3DOD transfers the task of object detection into the three-dimensional space. The idea of 3DOD is to output dimension and location of 3D bounding boxes and the corresponding class labels for all relevant objects within the sensors field of view. 3D bounding boxes are rectangular cuboids in the three-dimensional space. To ensure relevancy, their size should be minimal, while still containing all relevant parts of an object. One common way to parameterize a 3D bounding box is  $(x, y, z, h, w, l, c)$ , where  $(x, y, z)$  represent the 3D coordinates of the bounding box center,  $(h, w, l)$  refer to the height, width and length of the box, and  $c$  stands for the class of the box (Chen et al., 2017). Further, most approaches add an orientation parameter to the 3D bounding box defining the angle of each box (e.g., Shi et al., 2020b).

## 2.3. Sensing Technologies

To capture 3D scenes, commonly used monocular cameras are no longer sufficient. Therefore, special sensors have been developed to capture depth information. RGB-Depth (RGB-D) cameras like Intel's RealSense use stereo vision, while Light Detection and Ranging (LiDAR) sensors such as Velodyne's HDL-32E use laser beams to infer depth information. The data acquired by these 3D sensors can be converted to a more generic structure, the point cloud, which can be understood as a set of points in vector space. Further details on the different data inputs are provided in Section 5. By ordinary, 3DOD models resort on data captured by various active and passive optical sensor modalities with cameras and LiDAR-sensors as the most popular representatives.

### 2.3.1. Cameras

*Stereo cameras.* Stereo cameras are inspired by human ability to estimate depth of an object by capturing images with two eyes. Depth gets reconstructed by exploiting the disparity between two or more camera images that record the same scene from different points of view. To do so, stereoscopy leverages

---

<sup>1</sup>surge, heave, sway, yaw, pitch and roll

triangulation and epipolar geometry theory to create range information (Giancola et al., 2018). Acquired depth map normally gets appended to an RGB image as the fourth channel, together called RGB-D image. This sensor variant exhibits a dense depth map, though its quality is heavily dependent on depth estimation, which is also computationally expensive (Du et al., 2018).

*Time-of-Flight cameras.* Instead of deriving depth from different perspectives, the Time-of-Flight (TOF) principle can directly estimate the device-to-target distance. TOF systems are based on the LiDAR principle, sending light signals on the scene and measuring its time until receiving it back. The difference between LiDAR and camera based TOF is, that LiDAR build point clouds using a pulsed laser, whereas TOF-cameras capture depth maps through an RGB-like camera. The data captured by stereo and TOF cameras can either be transformed into 2.5D representation like RGB-D or into 3D representation by generating a point cloud (e.g., Song and Xiao, 2014; Qi et al., 2019; Sun et al., 2018; Ren and Suderth, 2020).

In general, camera sensors, such as stereo and TOF, possess the advantage of low integration costs and relatively low arithmetic complexity. However, all these methods experience considerable quality-volatility through environmental conditions like light and weather.

### 2.3.2. LiDAR Sensors

LiDAR sensors emit laser pulses on the scene and measure the time from the point of emission of the beam to receiving of the reflection. In combination with the constant of the speed of light, the measured time reveals the distance to the target. By assembling the 3D spatial information from the reflected laser in a 360° angle consequently, the sensor is constructing a full three-dimensional map of the environment. This map is a set of 3D points, also called *point cloud*.

The respective reflectance values represent the strength of the received pulses. Thereby LiDAR does not consider RGB (Lefsky et al., 2002). The HDL-64E3, as a common LiDAR system, outputs 120,000 points per frame, which adds up to a huge amount of data, namely 1,200,000 points per second on a 10 Hz frame rate (Arnold et al., 2019).

The advantages of LiDAR sensors are long-range detection abilities, high resolution compared to other 3D sensors and independence of lightning conditions, which are counterbalanced by its high costs and bulky devices (Arnold et al., 2019; Fernandes et al., 2021).

## 2.4. Domains

Looking at the extensive research on 3DOD in the past ten years, the literature can be roughly summarized into two main areas: *indoor applications* and *autonomous vehicle applications*. These two domains are the main drivers for the field, even though 3DOD is not strictly limited to these two specific areas, as there are also other applications conceivable and already in use, such as in retail, agriculture, and fitness.

Both domains face individual challenges and opportunities which led to this differentiation. However, it should also be

noted that research in both areas is not mutually exclusive, as some 3DOD models offer solutions that are sufficiently generic and therefore do not focus on a particular domain (e.g., Qi et al., 2018; Xu et al., 2018; Tang and Lee, 2019; Wang and Jia, 2019).

A fundamental difference between indoor and autonomous vehicle applications is that objects in indoor environments are often positioned one above the other. From this fact, possibilities arise with which inter-object relations between the target and the base/carrier object can be learned. In research, this is referred to as a holistic understanding of the inner scene, to ultimately enable better communication between service robots and people (Ren and Suderth, 2020; Huang et al., 2018). Challenges for indoor applications are that the scenes are often cluttered and that many objects occlude one another (Ren and Suderth, 2020).

Autonomous vehicle applications are characterized by long distances to potential objects and difficult weather conditions such as snow, rain and fog, which make the detection process more difficult (Arnold et al., 2019). Objects also occlude one another, but due to the observation that objects such as cars, pedestrians and traffic lights are unlikely to be positioned one above the other, techniques like the bird's-eye view projection can efficiently compensate for this disadvantage (e.g., Beltrán et al., 2018; Wang et al., 2018).

## 2.5. Datasets

As seen in 2DOD, a crucial prerequisite for continual development and fast progress of algorithms is the availability of publicly accessible datasets. They are needed to provide extensive data to train models. Further, they are used to apply benchmarks on them to compare one's own results with those of others. For instance, the availability of the dataset *ImageNet* (Deng et al., 2009) accelerated the development of 2D image classification and 2DOD models remarkably. The same phenomenon is observable in 3DOD and other tasks based on 3D data: more available data results in a greater coverage of possible scenarios.

Similar to the domain focus, the most commonly used datasets for 3DOD developments can be roughly divided into two groups, distinguishing between autonomous driving scenarios and indoor scenes. In the following, we describe some of the major datasets that are publicly available.

### 2.5.1. Autonomous Driving Datasets

*KITTI.* The most popular dataset for autonomous driving applications is KITTI (Geiger et al., 2012). It consists of stereo images, LiDAR point clouds and GPS coordinates, all synchronized in time. Recorded scenes range from highways, complex urban areas and narrow country roads. The dataset can be used for various tasks such as stereo matching, visual odometry, 3D tracking and 3D object detection. For object detection, KITTI provides 7,481 training- and 7,518 test frames including sensor calibration information and annotated 3D bounding boxes around the objects of interest in 22 video scenes. The annotations are categorized in easy, moderate and hard cases, depending on the object size, occlusion and truncation levels. Drawbacks of the dataset are the limited sensor configurations and

light conditions: all recordings have been made during daytime and mostly under sunny conditions. Moreover, the class frequencies are quite unbalanced. 75% belong to the class car, 15% to the class pedestrian and 4% to the class cyclist. In natural scenarios, the missing variety challenges the evaluation of the latest methods.

*nuScenes*. NuScenes comprises 1,000 video scenes of 20 seconds length in autonomous driving context. Each scene is presented through six different camera views, LiDAR and radar data with full 360° field of view (Caesar et al., 2020). It is significantly larger than the pioneering KITTI dataset with over seven times as many annotations and 100 times as many images. Further, the nuScenes dataset also provide night and bad weather scenarios, which is neglected in the KITTI dataset. On the downside the dataset has limited LiDAR sensor quality with 34,000 points per frame, as well as limited geographical diversity comparing to the Waymo open dataset covering an effective area of only five square kilometers.

*Waymo Open*. The Waymo Open dataset focuses on providing a diverse and large-scale dataset. It consists of 1,150 videos that are exhaustively annotated with 2D and 3D bounding boxes in images respectively LiDAR point clouds. The data collection was conducted by five cameras presenting a front and side view of the recording vehicle and one LiDAR sensor for 360° view. Further, the data is recorded in three different cities with various light and weather conditions, offering a diverse scenery (Sun et al., 2020).

### 2.5.2. Indoor Datasets

*NYUv2 & SUN RGB-D*. NYUv2 (Silberman et al., 2012) and its successor SUN RGB-D (Song et al., 2015) are datasets commonly used for indoor applications. The goal of these is to encourage methods focused on total scene understanding. The datasets got recorded using four different RGB-D sensors to ensure the generalizability of applied methods for different sensors. Even though SUN RGB-D inherited the 1449 labeled RGB-D frames from the NYUv2 dataset, NYUv2 is still occasionally used by nowadays methods. SUN RGB-D consists of 10,335 RGB-D images that are labelled with about 146,000 2D polygons and around 64,500 3D bounding boxes with accurate object orientation measures. Additionally, there is a room layout and scene category provided for every image. To improve the image quality, short videos of every scene have been recorded. Multiple frames of this videos were then used to create a refined depth map.

*Objectron*. Recently, Google released the Objectron dataset (Ahmadyan et al., 2021), which is composed of object centric video clips capturing nine different objects categories in indoor and outdoor scenarios. The dataset consists of 14,819 annotated video clips containing over four million annotated images. Each video is accompanied by a sparse-point cloud representation.

## 3. Related Reviews

As of today, there are to the best of our knowledge only a limited set of reviews aiming to arrange and classify the most important methods and pipelines for 3DOD.

Arnold et al. (2019) were some of the first to propose a classification for 3DOD approaches with a particular focus on autonomous driving applications. Based on the input data that is passed into the detection model, they divide the approaches into (i) *monocular image-based methods*, (ii) *point cloud methods* and (iii) *fusion-based methods*. Furthermore, they break down the point cloud category into three subcategories of data representation: (ii-a) *projection-based*, (ii-b) *volumetric representations*, and (ii-c) *Point Nets*. The data representation states which kind of input the model consumes and which information this input contains to allow the subsequent stage to process it more convenient according to the design choice.

While regarding various applications, such as 3D object classification, semantic segmentation and 3DOD, Liu et al. (2019b) focus on feature extraction methods which constitutes the properties and characteristics that the model derives from the passed data. They classify deep learning models on point clouds into (i) *point-based methods* and (ii) *tree-based methods*. The former directly uses the raw point cloud and the latter first employs a  $k$ -dimensional tree to preprocess the corresponding data representation.

Griffiths and Boehm (2019) consider object detection as a special type of classification and thus provide relevant information for 3DOD in their review on deep learning techniques for 3D sensed data classification. They differentiate the approaches on behalf of the data representation into (i) *RGB-D methods*, (ii) *volumetric approaches*, (iii) *multi view CNNs*, (iv) *unordered point set processing methods*, and (v) *ordered point set processing techniques*.

Huang and Chen (2020) touch lightly upon 3DOD in their review paper about autonomous driving technologies using deep learning methods. They suggest a similar classification of methods as Arnold et al. (2019) by distinguishing between (i) *camera-based methods*, (ii) *LiDAR-based methods*, (iii) *sensor-fusion methods*, and additionally (iv) *radar-based methods*. While giving a coarse structure for 3DOD, the conference paper is waiving an explanation for their classification.

Bello et al. (2020) consider the field from a broader perspective by providing a survey of deep learning methods on 3D point clouds. The authors organize and compare different methods based on a structure that is task-independent. Subsequently, they discuss the application of exemplary approaches for different 3D vision tasks, including classification, segmentation, and object detection.

Addressing likewise the higher-level topic of deep learning for 3D point clouds, Guo et al. (2021) give a more detailed look into 3DOD. They structure the approaches made for handling point clouds into (i) *region proposal-based methods*, (ii) *single shot methods*, and (iii) *other methods* by categorizing them on account of their model design choice. Additionally, the region proposal-based methods are split along their data representation into (i-a) *multi-view*, (i-b) *segmentation*, (i-c) *frustum-based* and again (i-d) *other methods*. Likewise, the single shot cate-

gory inherits the subcategories (ii-a) *bird's-eye view*, (ii-b) *discretization*, and (ii-c) *point-based* approaches.

Most recently, Fernandes et al. (2021) presented a comprehensive survey which might be the most similar to this work. They developed a detailed taxonomy for point cloud-based 3DOD. In general, they divide the detection models along their pipeline into three stages, namely *data representation*, *feature extraction* and *detection network modules*.

The authors note that in terms of data representation existing literature either follows the approach to transform the point cloud data into voxels, pillars, frustums, or 2D-projections or to directly consume the raw point cloud.

Feature extraction gets emphasized as the most crucial part of the 3DOD pipeline. Suitable features are essential for an optimal feature learning which in turn has a great impact on the appropriate object localization and classification in later steps. The authors classify the extraction methods into pointwise, segmentwise, objectwise and CNNs which are further divided into 2D-CNN and 3D-CNN backbones.

The detection network module consists of the multiple output task of object localization and classification, as well as the regression of 3D bounding box and orientation. Same as in 2DOD, these modules are categorized into the architectural design principles of single-stage and dual-stage detectors.

Although all preceding reviews provide some systematization for 3DOD, they move – with exception of Fernandes et al. (2021) – on a high level of abstraction. They tend to lose some of the information which is crucial to fully map relevant trends in this vivid research field.

Moreover, as mentioned above, all surveys are limited to either domain-specific aspects (e.g., autonomous driving applications) or focus on a subset of methods (e.g., point cloud-based approaches). Monocular-based methods, for example, are neglected in almost all existing review papers.

## 4. 3D Object Detection Pipeline

Intending to structure the research field of 3DOD from a broad perspective, we propose a systematization that enables to classify current 3DOD approaches at an appropriate abstraction level, by neither losing relevant information caused by a high level of abstraction nor being too specific and complex by a too fine-granular perspective. Likewise, our systematization aims at being sufficiently robust to allow a classification of all existing 3DOD pipelines and methods as well as of future works without the need of major adjustments to the general framework.

Figure 1 provides an overview of our systematization. It is structured along the general stages of an object detection pipeline, with several design choices at each stage. It starts with the choice of *input data* (Section 5), followed by the selection of a suitable *data representation* (Section 6) and corresponding approaches for *feature extraction* (Section 7). For the latter steps, it is possible to apply *fusion approaches* (Section 8) to combine different data inputs and take advantage of multiple feature representations. Finally, the *object detection module* is defined (Section 9).

The structuring along the pipeline enables us to order and understand the underlying principles of this field. Furthermore, we can compare the different approaches and are able to outline research trends in different stages of the pipeline. To this end, we carry out a qualitative literature analysis of proposed 3DOD approaches in the following sections along the pipeline to examine specific design options, benefits, limitations and trends within each stage.

## 5. Input Data

In the first stage of the pipeline, a model consumes the input data which already restricts the further processing. Common inputs for 3DOD pipelines are (i) *RGB images* (Section 5.1), (ii) *RGB-D images* (Section 5.2), and (iii) *point clouds* (Section 5.3). 3DOD models using RGB-D images are often referred to as 2.5D approaches (e.g., Deng and Latecki, 2017; Sun et al., 2018; Maisano et al., 2018), whereas 3DOD models using point clouds are regarded as true 3D approaches.

### 5.1. RGB Images

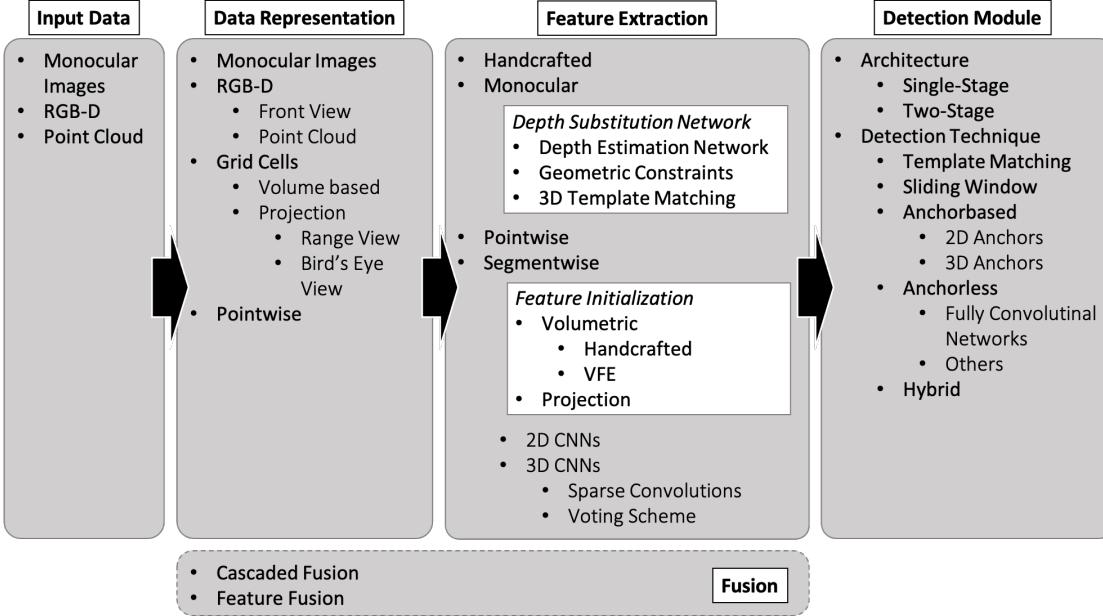
Monocular or RGB images provide a dense pixel representation in form of texture and shape information (Liang et al., 2018; Giancola et al., 2018; Arnold et al., 2019). A 2D image can be seen as a matrix, containing the dimensions of height and width with the corresponding color values.

Especially for subtasks of 3DOD applications such as lane line detection, traffic light recognition or object classification monocular based approaches enjoy the advantage of real time processing by 2DOD models. Likely the most severe disadvantage of monocular images is the lack of depth information. 3DOD benchmarks has shown that depth data are essential to achieve precise 3D localization (KITTI, 2021). Additionally, by only presenting a single view perspective monocular images faces the problem of object occlusion.

### 5.2. RGB-D Images

RGB-D images can be produced through stereo or TOF cameras, providing depth information aside the color one, as described in Section 2.3. RGB-D images consist of an RGB image with an additional depth map (Du et al., 2018). The depth map is comparable to a grayscale image, except that each pixel represents the actual distance between the sensor and the surface of the scene object. RGB image and depth image hold ideally a one-to-one correspondence between pixels (Wang and Ye, 2020).

Also called range images, RGB-D is convenient to use with the majority of 2DOD methods and depth information can be treated likewise to the three channels of RGB (Giancola et al., 2018). However, same as monocular images, RGB-D faces the problem of occlusion since the scene is only presented through a single perspective. In addition, it presents objects in different scales dependent on its position in the spatial space.



**Fig. 1. Systematization of a 3D object detection pipeline with its individual fine-branched design choices**

### 5.3. Point Cloud

The data acquired by 3D sensors can be converted to a more generic structure, the point cloud. It is a three-dimensional set of points that has an unorganized spatial structure (Otepka et al., 2013). The point cloud is defined by its points which encloses the spatial coordinates of an object sampled surface. However, other geometrical and visual attributes can be added to each point (Giancola et al., 2018).

As described in Section 2.3, point clouds can be obtained from LiDAR sensors or transformed RGB-D images. Yet, point clouds obtained from RGB-D images are typically noisier and sparser due to low resolution and perspective occlusion in comparison to LiDAR generated point clouds (Luo et al., 2020).

The point cloud offers a fully three-dimensional reconstruction of the scene, providing rich geometric, shape and scale information. This enables to extract meaningful features boosting the detection performance. Nevertheless, point clouds face severe challenges which are based on its nature and processability. Common deep learning operations, which have proven to be the most effective techniques for object detection, require data to be organized in a tensor with a dense structure (e.g., images, videos) which is not fulfilled by point clouds (Zhou and Tuzel, 2018). In particular, point clouds exhibit *irregular*, *unstructured* and *unordered* data characteristics (Bello et al., 2020).

- *Irregular* means that the points of a point cloud are not evenly sampled across the scene. Hence, some of the regions have denser point distribution than others. Especially faraway objects are usually represented sparsely by very few points because of the limited range recording ability of current sensors.
- *Unstructured* means that points are not on a regular grid. Accordingly, the distances between neighboring points can vary. In contrast, pixels in an image always have a

fixed position to their neighbors, which is evenly spaced throughout the image.

- *Unordered* means that the point cloud is just a set of points that is *invariant to permutations* of its members. Particularly, the order in which the points are stored does not change the scene that it represents. In other formats, an image for instance, data usually gets stored as a list (Qi et al., 2017a; Bello et al., 2020). Permutation invariance, however, means that a point cloud of  $N$  points has  $N!$  permutation and the subsequent data processing must be invariant to each of these different representations.

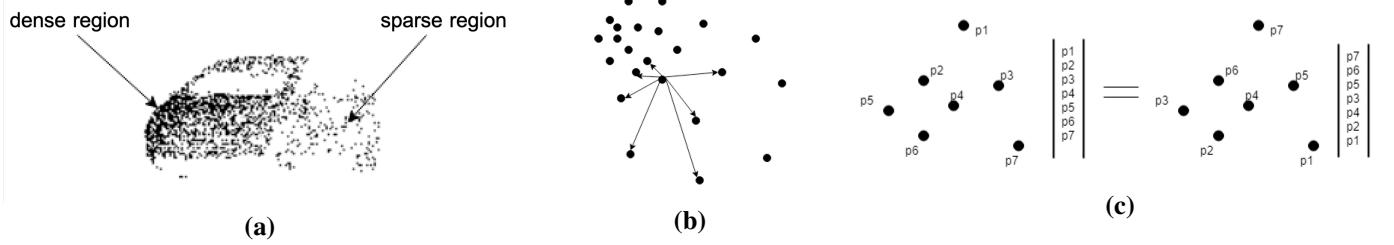
Figure 2 provides an illustrative overview of three challenging characteristics of point cloud data.

## 6. Data Representation

In order to ensure a correct processing of the input data by 3DOD models, it must be available in a suitable representation. Due to the different data formats, 3DOD data representation can be generally classified into 2D and 3D representations. Beyond that, we assign 2.5 representations to either 2D if they come in an image format, regardless of the number of channels, or 3D if the data gets described in a spatial structure. 2D representations generally cover (i) *monocular representations* (Section 6.1) and (ii) *RGB-D front views* (Section 6.2). 3D representations, on the other hand, cover (iii) *grid cells* (Section 6.3) and (iv) *pointwise representations* (Section 6.4).

### 6.1. Monocular Representation

Despite lacking the availability of range information, monocular representation enjoys a certain popularity among 3DOD methods due to its efficient computation in addition to being affordable and simple to set up with a single camera. Hence,



**Fig. 2. Characteristics of point cloud data:** (a) Irregular collection with sparse and dense regions, (b) Unstructured cloud of independent points without a fixed grid, (c) Unordered set of points that are invariant to permutation (Bello et al., 2020)

monocular representations are attractive for applications where resources are limited (Ku et al., 2019; Jørgensen et al., 2019).

The vast majority of monocular representation is using the well-known frontal view which is limited by the viewing angle of the camera. Other than that, Payen de La Garanderie et al. (2018) tackle monocular 360° panoramic imagery using equirectangular projections instead of rectilinear projection of conventional camera images. To access true 360° processing, they fold the panorama imagery into a 360° ring by stitching left and right edges together with a 2D convolutional padding operation.

Only a small proportion of 3DOD monocular approaches exclusively use image representation (e.g., Jørgensen et al., 2019) for 3D spatial estimations. Most models leverage additional data and information to substitute the missing depth information (more details in section 7.2.2). Additionally, representation fusion techniques are quite popular to compensate disadvantages. For instance, 2D candidates get initially detected from monocular images before predicting a 3D bounding box for the spatial object on basis of these proposals. In general, the latter step processes an extruded 3D subspace derived from the 2D bounding box. In case of representation fusion, monocular representation is usually not used for full 3DOD but rather as a support for increasing efficiency through limiting the search space for heavy three-dimensional computations or delivering additional features such as texture and color. These methods are described in depth in Section 8 (Fusion Approaches).

## 6.2. RGB-D Front View

RGB-D data can either be transformed to a point cloud or kept in its natural form of four channels. Therefore, we can distinguish between an *RGB-D (3D)* representation (e.g., Chen et al., 2018; Tang and Lee, 2019; Ferguson and Law, 2019), which exploits the depth information in its spatial form of a point cloud, and an *RGB-D (2D)* representation (e.g., Chen et al., 2015; He et al., 2017; Li et al., 2019c; Rahman et al., 2019; Luo et al., 2020), which holds an additional 2D depth map in an image format.

Thus, as mentioned in section 5.2, RGB-D (2D) images represent monocular images with an appended fourth channel of the depth map. The data is compacted along the  $z$ -axis generating a dense projection in the frontal view. The 2D depth image can be processed similar to RGB-channels by high performant 2D-CNN models.

The RGB-D (2D) representation is often referred to as *front view* (FV) in 3DOD research. However, front and *range view*

(RV) are occasionally equated with each other in current research. To clarify, this work considers the FV as RGB-D image generated by a TOF, stereo or likewise camera, whereas the RV gets predefined as the natural frontal projection of a point cloud (see also Section 6.3.2).

## 6.3. Grid Cells

The challenges of processing a point cloud (cf. section 5.3) are of particular difficulty for CNNs, since convolutional operations require a structured grid which is lacking in point cloud data (Bello et al., 2020). Thus, to resort on advanced deep learning methods and leverage high informative point clouds, they must first be transformed into a suitable representation.

Current research presents two ways to handle point clouds. The first and more natural solution is to fit a regular grid onto the point cloud, producing a grid cell representation. Many approaches do so by either quantizing point clouds into 3D *volumetric grids* (Section 6.3.1) (e.g., Song and Xiao, 2014; Zhou and Tuzel, 2018; Shi et al., 2020b) or by discretizing them to (*multi view*) *projections* (Section 6.3.2) (e.g., Li et al., 2016; Chen et al., 2017; Beltrán et al., 2018; Zheng et al., 2020).

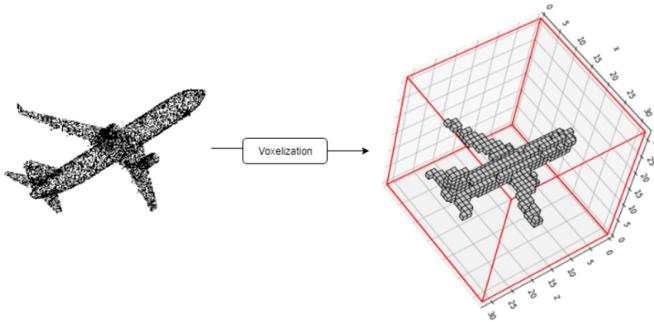
The second and more abstract way to solve the point cloud representation problem is to process the point cloud directly by grouping points into point sets. This approach does not require convolutions and thus enables to process the point cloud without transformation in a *pointwise representation* (Section 6.4) (e.g., Qi et al., 2018; Shi et al., 2019; Huang et al., 2020).

Along these directions, several state-of-the-art methods have been proposed for 3DOD pipelines, which we are going to describe exemplary in the following

### 6.3.1. Volumetric Grids

The main idea behind the volumetric representation is to subdivide the point cloud into equally distributed grid cells, called voxels, enabling a further processing on a structured form. Therefore, the point cloud is converted into a 3D fixed-size voxel structure of dimension  $(x, y, z)$ . The resulting voxels either contain raw points or already encode the occupied points into a feature representation such as point density or intensity per voxel (Bello et al., 2020). Figure 3 illustrates the transformation of a point cloud into a voxel-based representation.

Usually, the voxels are of cuboid shape (e.g., Li, 2017; Zhou and Tuzel, 2018; Ren and Sudderth, 2020). However, there are also approaches applying other forms such as pillars (e.g., Lang et al., 2019; Lehner et al., 2019).



**Fig. 3. Voxelization of a point cloud (Bello et al., 2020)**

Feature extraction networks utilizing voxel-based representations are computationally more efficient and reduce memory needs. Instead of extracting low- and high-dimensional features for each point individually, clusters of points (i.e., voxels) are used to extract such features (Fernandes et al., 2021). Despite reducing the dimensionality of point cloud through discretization, the spatial structure is kept and allows to make use of the geometric features of the scene.

Volumetric approaches using cuboid transformation of the point cloud scene have been used for example by Song and Xiao (2014), Li (2017), Engelcke et al. (2017), Zhou and Tuzel (2018), Ren and Sudderth (2020), and Kuang et al. (2020).

To speed up computation, Lang et al. (2019) propose a pillar-based voxelization of the point cloud, instead of the conventional cubical quantization. The vertical column representation allows to skip expensive 3D convolutions in the following steps, since pillars have unlimited spatial extent in  $z$ -direction and therefore can be directly projected to 2D pseudo images. All feature extraction operations are therefore processable by efficient 2D-CNNs.

### 6.3.2. Projection-based Representation

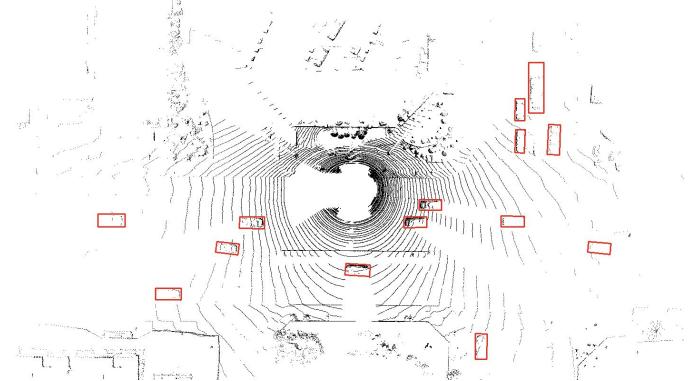
In addition to the need to transform the point cloud into a processable state, several approaches seek to leverage the expertise and power of 2DOD processing. Especially for inference time reasons of point cloud models projection approaches became popular. They are projecting the point cloud into an image plane whilst preserving the depth information. Then the representation can be consequently processed by efficient 2D extractors and detectors. Commonly used representations are the previously mentioned *range view*, using an image plane projection, and the *bird's eye view*, projecting the point cloud onto the ground plane.

**Range View.** Whereas the 2D FV corresponds to monocular and stereo cameras, the RV is a native 2D representation of the LiDAR data. The point cloud is projected onto a cylindrical 360° panoramic plane exactly as the data is captured by the LiDAR sensor. Since the LiDAR projection is still neither in a processable state nor contains any discriminative features such as RGB information, the projected RV is partitioned into a fine-grained grid and encoded in the successive feature initialization step (see Section 7.4.1).

Meyer et al. (2019b) and Liang et al. (2020) emphasize that the naturally compact RV results in a more efficient computation in comparison to other projections. In addition, the information loss of projection is considerably small since the RV constitutes the native representation of a rotating LiDAR sensor (Liang et al., 2020). At the same time, RV suffers from distorted object size and shape on account of its cylindrical image character (Yang et al., 2018b). Inevitably, RV representations face the same problem as camera images, in that the size of objects is closely related to their range and occlusions may occur due to perspective (Zhou et al., 2019). Liang et al. (2020, p.2) argue that “*range image is a good choice for extracting initial features, but not a good choice for generating anchors*”. Moreover, its performance in 3DOD models does not match with state-of-the-art *bird's eye view* (BEV) projections. Nevertheless, RV enables to produce more accurate detections on small objects (Meyer et al., 2019a,b).

Exemplary approaches using RV are proposed by Li et al. (2016), Chen et al. (2017), Zhou et al. (2019), and Liang et al. (2020).

**Bird's Eye View.** While RV discretize the data to a panoramic image plane, the BEV is an orthographic view of the point cloud scene projecting the points onto the ground plane. Therefore, data gets condensed along the  $y$ -axis. Figure 4 shows an exemplary illustration of a BEV projection.



**Fig. 4. Visualization of bird's eye view projection of a point cloud without discretization into grid cells (Yang et al., 2018a)**

Chen et al. (2017) were some of the first introducing BEV to 3DOD in their seminal work MV3D. They organized the point cloud as a set of voxels and then transformed each voxel column through an overhead perspective into a 2D grid with a specific resolution and encoding for each cell. As a result, a dense pseudo-image of the ground plane is generated which can be processed by standard image detection architecture.

Unlike RV, BEV offers the advantage to preserve the object scales regardless of range. Further, BEV perspective eases the typical occlusion problem of object detection since objects are displayed separately from each other in a free-standing position (Zhou et al., 2019). These advantages let the networks exploit priors about the physical dimension of objects, especially for anchor generation (Yang et al., 2018b).

On the other hand, BEV data gets sparse in at distance, which makes it unfavorable for small objects (Xu et al., 2018; Lang

et al., 2019). Furthermore, the assumption that all objects lie on one mutual ground plane often turns out to be infeasible in reality, especially in indoor scenarios (Zhou et al., 2019). Also the often coarse voxelization of BEV may remove fine granular information leading to inferior detection at small object sizes (Meyer et al., 2019a).

Exemplary models using BEV representation can be found in the work from Wang et al. (2018), Beltrán et al. (2018), Liang et al. (2018), Yang et al. (2018b), Simon et al. (2019b), Zeng et al. (2018), Li et al. (2019b), Ali et al. (2019), He et al. (2020), Zheng et al. (2020), Liang et al. (2020), and Wang et al. (2020).

*Multi View Representation.* Often BEV and RV are not used as single data representation but as a multi view approach, meaning that RV and BEV but also images of monocular nature are combined to represent the spatial information of a point cloud.

Chen et al. (2017) were the first to integrate this concept into a 3DOD pipeline, followed by many other models adapting to fuse 2D representations from different perspectives (e.g., Ku et al., 2018; Li et al., 2019b; Wang et al., 2019, 2020; Liang et al., 2020).

Although the representations of BEV and RV are compact and efficient, they are always limited by the loss of information, originated during the discretization to a fixed number of grid-cells and the respective feature encoding of the cell’s points.

#### 6.4. Pointwise Representation

Either way, discretizing the point cloud into a projection or volumetric representation inevitably leads to information loss. Against this backdrop, Qi et al. (2017a) introduced *PointNet*, and thus a new way to consume the raw point in its unstructured nature having access to all the recorded information.

In pointwise representations points are available isolated and sparsely distributed in a spatial structure representing the visible surface, while preserving precise localization information. PointNet handles this representation by aggregating neighboring points and extracting a compressed feature from the low-dimensional point features of each set, making raw point based representation possible for 3DOD. A more detailed description of PointNet and its successor *PointNet++* is given in Sections 7.3.1 and 7.3.2, respectively.

However, PointNet was developed and tested on point clouds containing 1,024 points (Qi et al., 2017a), whereas realistic point clouds captured by a standard LiDAR sensor such as Velodyne’s HDL-64E3 usually consist of 120,000 points per frame. Thus, applying PointNet on the whole point cloud is a time- and memory-consuming operation. De facto, point clouds are rarely consumed in total. As a consequence, it requires further techniques to improve efficiency, such as cascading fusion approaches (see Section 8.1) that crop point clouds to its region of interest, only handing subsets of the point cloud to the pointwise feature extraction stage.

In general it can be stated that point-based representations retain more information than voxel- or projection-based methods. But on the downside point-based methods are inefficient when the number of points is large. Yet, a reduction of the point clouds like in cascading fusion approaches always comes with a decrease in information. In summary, it can be said that

maintaining both efficiency and performance is not achievable for any of the representations to date.

## 7. Feature Extraction

Feature extraction gets emphasized as the most crucial part of the 3DOD pipeline that research is focusing on (Fernandes et al., 2021). It follows the paradigm to reduce dimensionality of the data representation with the intention of representing the scene by a robust set of features. Features generally depict the unique characteristics of the data used to bridge the semantic gap, which denotes the difference between the human comprehension of the scene and the model’s prediction. Suitable features are essential for an optimal feature learning which in turn has a great impact on the detection in later steps. Hence, the goal of feature extraction is to provide a robust semantic representation of the visual scene that ultimately leads to the recognition and detection of different objects (Zhao et al., 2019).

As with 2DOD, feature extraction approaches can be roughly divided into (i) *handcrafted feature extraction* (Section 7.1) and (ii) *feature learning* via deep learning methods. Regarding the latter, we can distinguish the broad body of 3DOD research depending on the respective data representation. Hence, feature learning can be performed either in a (ii-a) *monocular* (Section 7.2), (ii-b) *pointwise* (Section 7.3), (ii-c) *segmentwise* (Section 7.4), or in a (ii-d) *fusion-based* approach (Section 8).

### 7.1. Handcrafted Feature Extraction

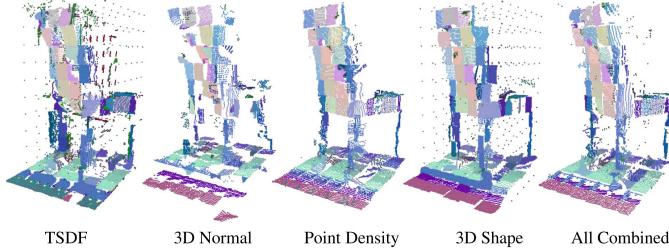
While the vast majority is shifted to hierarchical deep learning which is able to produce more complex and robust features, there are still cases where features are manually crafted.

Handcrafted feature extraction is distinguishing itself from feature learning in that the features are individually selected and usually directly serve for final determination of the scene. The features like edges or corners are tailored by hand and are the ultimate characteristics for object detection. There is no algorithm that independently learns how these features are built-up or how they can be combined as it is the case for CNNs. The feature initialization depicts already the feature extraction step. Often these handcrafted features are then scored by SVMs or random forest classifiers that are exhaustively deployed over the entire image respectively scene.

A few exemplary 3DOD models using handcrafted feature extraction shall be introduced in the following. For instance, Song and Xiao (2014) use four types of 3D features which they exhaustively extract from each voxel cell, namely point density, 3D shape feature, 3D normal feature and truncated signed distance function feature, to handle problems of self-occlusion (see Figure 5).

Wang and Posner (2015) also use a fixed-dimensional feature vector containing the mean and variance of the reflectance values of all points that lie within a voxel and an additional binary occupancy feature. The features are not further processed and directly used for detection purposes in a voting scheme.

Ren and Sudderth (2016) introduce their discriminative cloud of oriented gradients (COG) descriptor, which they further develop in their subsequent work when proposing LSS (latent support surfaces) (Ren and Sudderth, 2018) and COG 2.0 (Ren and



**Fig. 5. Visualization of manually specified features (Song and Xiao, 2014)**

Sudderth, 2020). Additionally, the approach got also adopted by Liu et al. (2018b). In general, the COG feature is able to describe complex 3D appearances within every orientation, as it consists of a gradient computation, 3D orientation bins and a normalization. For each proposal they compute point cloud density, surface normal features and COG-feature in a sliding window fashion, which they then score with pre-trained SVMs.

In addition, manually configured features are typically used in combination with matching algorithms for detection purposes. For example, Yamazaki et al. (2018) create gradient-based image features by applying a principal component analysis to the point cloud which are then used to compute normalized cross-correlation. The main idea is to use the spatial relationships among image projection directions to discover the optimal template matching for detection. Similarly, Teng and Xiao (2014) use handcrafted features, namely color histogram and key point histogram for template matching purposes. Another example is the approach proposed by He et al. (2017). The authors create silhouette gradient orientations from RGB and surface normal orientations from depth images.

## 7.2. Monocular Feature Learning

Probably the most difficult form of feature extraction for 3DOD is exercised in monocular representation. Since there is no direct depth data, well-informed features are difficult to obtain, and thus most approaches attempt to compensate for this lack of information with various depth substitution techniques.

### 7.2.1. Solely Monocular Approaches

In our literature corpus there is only one paper which take up the challenge of performing 3DOD exclusively on monocular imagery (Jørgensen et al., 2019). The authors use a single shot detection (SSD) framework (Liu et al. (2016), see also Section 9.3.2) to generate per-object canonical 3D bounding box parameters. They start from a classical bounding box detector and add new output heads for 3DOD features, specifically distance, orientation, dimension and 3D corners to the already available class score and 2D bounding box heads. The conducted feature extraction is the same as it is in the 2D framework.

Models using solely monocular inputs follow the straightforward way to predict spatial parameter directly, without any range information. While the objectives of dimension and pose estimation are relatively easy to fulfill as they rely heavily on the given feature of appearance in 2D images, they face the challenge of difficult location estimation since monocular in-

puts do not own a natural feature for spatial location (Gupta et al., 2019; Liu et al., 2019b).

### 7.2.2. Informed Monocular Approaches

The absence of depth information cannot be fully compensated in solely monocular approaches. Therefore, many state-of-the-art monocular models supplement the 2D representation by an auxiliary depth estimation network or additional external data and information. The idea is to use prior knowledge of the target objects and scene, such as shape, context and occlusion patterns to compensate missing depth data.

These depth substitution techniques can be roughly divided into (i) *depth estimation networks*, (ii) *geometric constraints*, and (iii) *3D template matching*. Often, they are used in combination.

Whereas depth estimation networks are directly applied onto the original representation to generate a new input for the model, geometric constraints and 3D model matching make restitution for the lack of range information in the later detection steps of the pipeline.

*Depth Estimation Networks.* Depth estimation networks generate informed depth features or even entirely new representations that possess range information such as point clouds from monocular images. These new representations are then subsequently exploited by depth-aware models. A representative model is Mono3D (Chen et al., 2016), which uses additional instance and semantic segmentation along with further features to reason about the pose and location of objects in 3D space.

Srivastava et al. (2019) modify the generative adversarial network of BirdNet (Beltrán et al., 2018) to create a BEV projection from the monocular representation. All following operations such as feature extractions and predictions are then performed on this new representation.

Similarly, Roddick et al. (2018) transform monocular representation to BEV perspective. They introduce an orthographic feature transformation network that maps the features from the RGB perspective to a 3D voxel map. The features of the voxel map are eventually reduced to the 2D orthographic BEV feature map by consolidation along the vertical dimension.

Payen de La Garanderie et al. (2018) adapt the monocular depth recovery technique of Godard et al. (2017), called Mono Depth, for their special case of monocular 360° panoramic processing. They predict a depth map by training a CNN on left-right consistency inside stereo image pairs. However, at the time of inference, the model requires only single monocular images for the estimation of a dense depth map.

Even more advanced, a few approaches devote themselves to generate a point cloud from monocular images. Xu and Chen (2018) use a stand-alone network for disparity prediction which is similarly based on Mono Depth to predict a depth map. Unlike Payen de La Garanderie et al. (2018), they further process the depth map to estimate a LiDAR-like point cloud.

An almost identical procedure is presented by Ma et al. (2019). First, they generate a depth map through a self-defined CNN and then proceed to generate a point cloud by using the camera calibration files. While Xu and Chen (2018) mainly take the depth data as auxiliary information of RGB features,

Ma et al. (2019) focus to take the generated depth as the core feature and make explicitly use of its spatial information.

Similarly, Weng and Kitani (2019) adapt the deep ordinal regression network (DORN) by Fu et al. (2018) for their pseudo LiDAR point cloud generation and then exploit a Frustum-PointNet-like model (Qi et al., 2018) for the object detection task. Further information on the Frustum PointNet approach is given in Section 8.1.

Instead of generating the whole scene to a point-based representation, Ku et al. (2019) primarily reduce the space by lightweight predictions and then only transform the candidate boxes to point clouds, preventing redundant computation. To do so, they exploit instance segmentation and available LiDAR data for training to reconstruct a point cloud in a canonical object coordinate system. A similar approach for instance depth estimation is pursued by Qin et al. (2019a).

*Geometric Constraints.* Depth estimation networks give the advantage of closing the gap of missing depth in a direct way. Yet, errors and noise occur during depth estimation, which may lead to biased overall results and contributes to a limited upper-bound of performance (Brazil and Liu, 2019; Barabau et al., 2020). Hence, various methods try to skip the naturally ill-posed depth estimation and tackle monocular 3DOD as a geometrical problem of mapping 2D into 3D space.

Especially in autonomous driving applications, the 3D box proposals are often constrained by a flat ground assumption, namely the street. It is assumed that all possible targets are located on this plane, since automotive vehicles do not fly. Therefore, these approaches force the bounding boxes to lay along the ground plane. In indoor scenarios, on the other hand, objects are located on various height levels. Hence, the ground plane constraint does not get the attention as in autonomous driving application models. Nevertheless, plane fitting is frequently applied in indoor scenarios to get the room orientation.

Zia et al. (2015) were one of the first to assume a common ground plane in their approach helping them to extensively reconstruct the scene. Further, a ground plane drastically reduces the search space by only leaving two degrees of freedom for translation and one for rotation. Other representative examples for implementing ground plane assumptions in monocular representations are given by Chen et al. (2016), Du et al. (2018) and Gupta et al. (2019). All of them leverage the random sample consensus approach by Fischler and Bolles (1981), a popular technique that is applied for ground plane estimation.

A different geometrical approach used to recover the under-constrained monocular 3DOD problem, is to establish a consistency between the 2D and 3D scene. Mousavian et al. (2017), Li et al. (2019a), Liu et al. (2019a) and Naiden et al. (2019) do so by projecting the 3D bounding box into a previously ascertained 2D bounding box. The core notion is that the 3D bounding box should fit tightly to at least one side of its corresponding 2D box detection. Naiden et al. (2019) use, for instance, a least square method for the fitting task.

Other methods deploy 2D-3D consistency by incorporating geometric constraints such as room layout and camera pose estimations through an entangled 2D-3D loss function (e.g., Huang et al., 2018; Simonelli et al., 2019; Brazil and Liu,

2019). For example, Huang et al. (2018) define the 3D object center through corresponding 2D and camera parameters. Then a physical intersection between 3D objects and 3D room layout gets penalized. Simonelli et al. (2019) first disentangle 2D and 3D detection loss to optimize each loss individually, then they also leverage correlation between 2D and 3D in a combined multi-task loss.

Another approach is presented by Qin et al. (2019b), who exploit triangulation, which is well known for estimating 3D geometry in stereo images. They use 2D detection of the same object in a left and right monocular image for a newly introduced anchor triangulation, where they directly localize 3D anchors based on the 2D region proposals.

*3D template matching.* An additional way of handling monocular representation in 3DOD is matching the images with 3D object templates. The idea is to have a database of object images from different viewpoints and their underlying 3D depth features. For instance, that could be rendering synthetic images from computer-aided design (CAD) models, producing images from all sides of the object. Then the monocular input image is searched and matched using this template database. Hence, the object pose and location can be concluded.

Fidler et al. (2012) address the task of monocular object detection with the representation of an object as a deformable 3D cuboid. The 3D cuboid consists of faces and parts, which are allowed to deform in accordance to their anchors of the 3D box. Each of these faces is modelled by a 2D template that matches the object appearance from an orthogonal point of view. It is assumed that the 3D cuboid can be rotated, so that the image view can be projected from a defined set of angles on the respective cuboids' face and subsequently scored by a latent SVM.

Chabot et al. (2017) initially use a network to generate 2D detection results, vehicle part coordinates and a 3D box dimension proposal. Thereafter, they match the dimensions to a 3D CAD dataset composed by a fixed number of object models to associate corresponding 3D shapes in form of manually annotated vertices. Those 3D shapes are then used for performing 2D-to-3D pose matching in order to recover 3D orientation and location. Barabau et al. (2020) reason their approach on sparse but salient features, namely 2D key points. They match CAD templates on account of 14 key points. After assigning one of five distinct geometric classes, they execute an instance depth estimation by a vertical plane passing through two of the visible key points to lift the predictions to a 3D space.

3D templates provide a potentially powerful source of information, yet, a sufficient quantity of models is not always available for each object class. Thus, these methods tend to focus on a small number of classes. The limitation on shapes covered by the selection of 3D templates, make the 3D template matching approach hard to generalize respectively extend to classes where models are unavailable (Ku et al., 2019).

Summarizing, monocular 3DOD achieves promising results. Nevertheless, the missing depth data prevents monocular 3DOD from reaching state-of-the-art results. Further, the depth substitution techniques may limit the detection performance because errors in depth estimation, geometrical assumptions or

template matching are propagated to the final 3D box prediction.

### 7.3. Pointwise Feature Learning

As described above, the application of deep learning techniques on point cloud data is not straightforward due to the irregularity of the point cloud (cf. Section 5.3). Many existing methods try to leverage the expertise of convolutional feature extraction by projecting point clouds either into 2D image views or convert them into regular grids of voxels. However, the projection to a specific viewpoint discards valuable information, which is particularly important in crowded scenes, while the voxelization of the point cloud leads to heavy computation on account of the sparse nature of point clouds and supplementary information loss in point-crowded cells. Either way, manipulating the original data may have a negative effect.

To overcome the problem of irregularity in point clouds with an alternative approach, Qi et al. (2017a) propose *PointNet*, which is able to learn pointwise features directly from the raw point cloud. It is based on the assumption that points which lie close to each other can be grouped together and compressed as a single point. Shortly after, Qi et al. (2017b) introduced its successor *PointNet++*, adding the ability to capture local structures in the point cloud. Both networks are originally designed for classification of the whole point cloud, next to being able of predicting semantic classes for each point of the point cloud. Thereafter, Qi et al. (2018) introduced a way to implement PointNet into 3DOD by proposing Frustum-PointNet. By now, many of the state-of-the-art 3DOD-methods are based on the general PointNet-architectures. Therefore, it is crucial to understand the underlying architecture and how it is used in 3DOD methods.

#### 7.3.1. PointNet

PointNet is built out of three key modules: (i) a max-pooling layer serving as a symmetric function, (ii) a local and global information combination structure in form of a multi-layer perceptron (MLP), and (iii) two joint alignment networks for the alignment of input points and point features, respectively (Qi et al., 2017a).

To face the permutation invariance of point clouds, PointNet is made up with symmetric functions in form of max-pooling operations. Symmetric functions have the same output regardless of the input order. The max-pooling operation results in a global feature vector that aggregates information from all points of the point cloud. Since the max pooling function operates as a “winner takes it all” paradigm, it does not consider local structures, which is the main limitation of PointNet.

Following, PointNet accommodates an MLP that uses the global feature vector subsequently for the classification tasks. Other than that, the global features can also be used in a combination with local point features for segmentation purposes.

The joint alignment networks ensure that a single point cloud is invariant to geometric transformations (e.g., rotation and translation). PointNet uses this natural solution to align the input points of a set of point clouds to a canonical space by pose normalization through spatial transformers, called T-Net. The

same operation is deployed again by a separate network for feature alignment of all point clouds in a feature space. Both operations are crucial for the networks’ predictions to be invariant of the input point cloud.

#### 7.3.2. PointNet++

PointNet++ is built in a hierarchical manner on several set abstraction layers to address the original PointNet’s missing ability to consider local structures. At each level, a set of points is further abstracted to produce a new set with fewer elements, in fact summarizing the local context. The set abstraction layers are composed again of three key layers: (i) a sampling layer, (ii) a grouping layer, (iii) and a PointNet layer (Qi et al., 2017b). Figure 6 provides an overview of the architecture.

The sampling layer is employed to reduce the resolution of points. PointNet++ uses farthest point sampling (FPS) which only samples the points that are the most distant from the rest of the sampled points (Bello et al., 2020). Thereby, FPS identifies and retains the centroids of the local regions for a set of point.

Subsequently, the grouping layers are used to group the representative points, which are obtained from the sampling operation, into local patches. Moreover, it constructs local region sets by finding neighboring points around the sampled centroids. These are further exploited to compute the local feature representation of the neighborhood. PointNet++ adopts ball query, which searches a fixed sphere around the centroid point and then groups all within laying points.

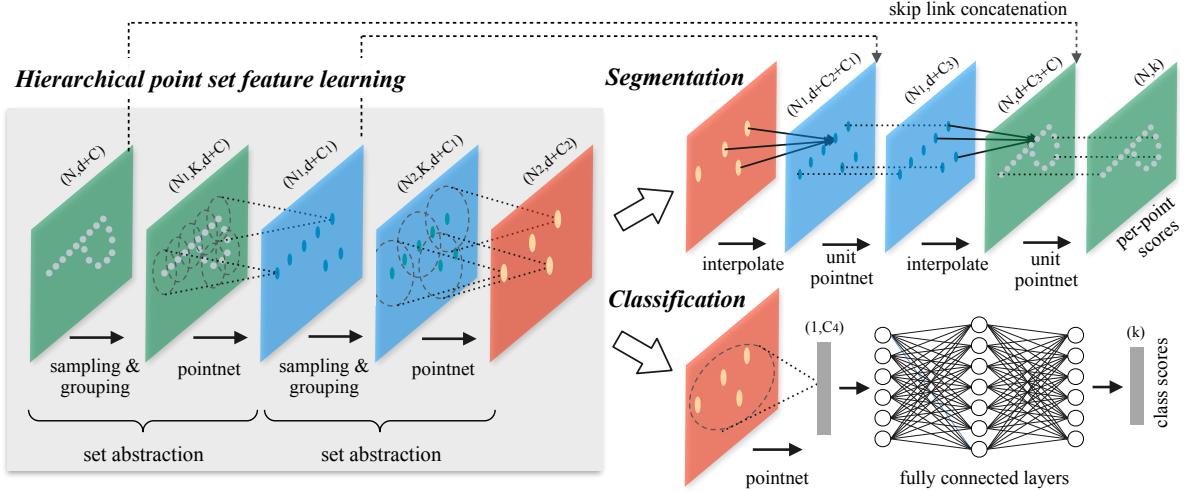
The grouping and sampling layers represent preprocessing tasks to capture local structure before giving the abstracted points to the PointNet layer. This layer consists of the original PointNet-architecture and is applied to generate a feature vector of the local region pattern. The input for the PointNet layer is the abstraction of the local regions, meaning the centroids and local features that encode the centroids’ neighborhood.

The process of grouping, sampling and applying PointNet is repeated in a hierarchical manner, which down-sample the points further and further until the last layer produces one final global feature vector (Bello et al., 2020). This allows PointNet++ to work with the same input data at different scales and produce higher level features at each set abstraction layer and thereby capturing local structures.

#### 7.3.3. PointNet-based Feature Extraction

In the following, no explicit distinction is made between PointNet and PointNet++. Instead, we summarize both under the term PointNet-based approaches, considering that we primarily want to imply that pointwise methods for feature extraction or classification purpose are used.

Many state-of-the-art models use PointNet-like networks in their pipeline for feature extraction. An exemplary selection of seminal works include the proposals from Qi et al. (2018), Yang et al. (2018c), Zhou and Tuzel (2018) Xu et al. (2018), Shi et al. (2019), Shin et al. (2019), Pamplona et al. (2019), Lang et al. (2019), Wang and Jia (2019), Yang et al. (2020), Li et al. (2020), Yoo et al. (2020), Zhou et al. (2020), and Huang et al. (2020).



**Fig. 6. Architecture of PointNet++ (Qi et al., 2017b)**

Very little is changed in the way PointNet is used from the models adopting it for feature extraction, indicating that it is already a well-designed and mature technique. Yet, Yang et al. (2020) examine in their work that the up-sampling operation in the feature propagation layers and refinement modules consume about half of the inference time of existing PointNet approaches. Therefore, they abandon both processes to drastically reduce inference time. However, predicting only on the surviving representative points on the last set abstraction layer leads to huge performance drops. Therefore, they propose a novel sampling strategy based on feature distance and fuse this criterion with the common euclidean distance sampling for meaningful features.

PointNet-based 3DOD models generally show superior performances in classification as compared to models using other feature extraction methods. The set abstraction operation brings the crucial advantage of flexible receptive fields for feature learning through setting different search radii within the grouping layer (Shi et al., 2019). Flexible receptive fields can better capture the relevant content respectively features as it adapts to the input. Fixed receptive fields such as convolutional kernels are always limited to their dimensionality which may result in that features and objects of different sizes cannot be captured so well. However, PointNet operations, especially set abstractions, are computationally expensive which manifests in long inference times compared to convolutions or fully connected layers (Yang et al., 2019, 2020).

#### 7.4. Segmentwise Feature Learning

Segmentwise feature learning follows the idea of a regularized 3D data representation. In comparison to pointwise feature extraction, it does not take the points of the whole point cloud into account as in the previous section, but processes an aggregated set of grid-like representations of the 3D scene (cf. Section 6.3), which are therefore considered as *segments*.

In the following, we describe central aspects and operations of segmentwise feature learning, including (i) *feature initialization* (Section 7.4.1), (ii) *2D and 3D convolutions* (Sec-

tion 7.4.2), *sparse convolution* (Section 7.4.3), and *voting scheme* (Section 7.4.4).

##### 7.4.1. Feature Initialization

Volumetric approaches discretize the point cloud into a specific volumetric grid during preprocessing. Projection models, on the other hand, typically lay a relatively fine-grained grid on the 2D mapping of the point cloud scene. In either case, the representation is transformed into segments. In other words, segments can be either volumetric grids like voxels and pillars (cf. Section 6.3.1) or discretized projections of the point cloud like RV and BEV projections (cf. Section 6.3.2).

These generated segments enclose a certain set of points that does not yet possess any processable state. Therefore, an encoding is applied on the individual segments, aggregating the points enclosed by it.

The intention of the encoding is to fill the formulated segments or grids with discriminative features giving information about the set of points that lie in each individual grid. This process is called feature initialization. Through the grid these features are now available in a regular and structured format. Other than in handcrafted feature extraction methods (cf. Section 7.1), these grids are not yet used for detection but made accessible to CNNs or other extraction mechanisms to further condense the features

For volumetric approaches, current research can be divided into two streams of feature initialization. The first and probably more intuitive approach is to encode the voxels manually. However, the handcrafted feature encoding introduces a bottleneck that discards spatial information and may prevent these approaches from effectively taking advantage of 3D shape information. This led to the latter one, where models apply a lightweight PointNet on each voxel to learn pointwise features and assign them in an aggregated form as voxel features, called *voxel feature encoding* (VFE) (Zhou and Tuzel, 2018).

*Volumetric Feature Initialization.* Traditionally, voxels get encoded into manually chosen features, so that each voxel contains one or more values consisting of statistics computed from

the points within that voxel cell. The feature choice is already a crucial task since it should capture the important information within a voxel and describe the corresponding points in a sufficiently discriminative way.

Early approaches like Sliding Shapes (Song and Xiao, 2014) uses a combination of four types of 3D features to encode the cells, namely point density, a 3D shape feature, a surface normal features and a specifically developed feature to handle the problem of self-occlusion, called truncated signed distance function. Others, however, rely only on statistical encoding like Wang and Posner (2015) as well as their adaption by Engelcke et al. (2017) that propose three shape factors, the mean and variance of the reflectance values of points as well as a binary occupancy feature.

Much simpler approaches are pursued by Li (2017) and Li et al. (2019e) using a binary encoding to express whether a voxel contains points or not. In order to prevent too much loss of information through a rudimentary binary encoding, the voxel size is usually set comparable small to generate high-resolution 3D grids (Li et al., 2019e).

More recently, voxel feature initialization shifted more and more towards deep learning approaches, for similar reasons as seen at feature extraction in other computer vision tasks where manually chosen features are not as performant as learned ones.

While segmentwise approaches turn out to be comparably efficient 3D feature extraction methods, pointwise models demonstrate impressive results in detection accuracy, since they have recourse to the full information of each point of the point cloud. By encoding the voxel in standard features by hand, normally a lot of information gets lost.

As a response, Zhou and Tuzel (2018) introduced the seminal idea of VFE and a corresponding deep neural network which moves the feature initialization from hand-crafted voxel feature encoding to deep-learning-based encoding. More specifically, they proposed VoxelNet, which is able to extract pointwise features from each segment of a voxelized point cloud through a lightweight PointNet-like network. Subsequently, the individual point features get stacked together with a locally aggregated feature at voxel-level. Finally, this volumetric representation is fed into 3D convolutional layers for further feature aggregation.

The use of PointNet allows VoxelNet to capture inter-point relations within a voxel and therefore hold a more discriminative feature than normal encoded voxel consisting of statistical values. A schematic overview of VoxelNet’s architecture is given in Figure 7.

The seminal idea of VFE, is – in modified versions – used in many subsequent approaches. Several works focus on improving performance and efficiency of VoxelNet. Performance-wise, Kuang et al. (2020) developed a novel feature pyramid extraction paradigm. For speeding up the model, Yan et al. (2018) and Shi et al. (2020b) combined VFE with more efficient sparse convolutions. Other than that, Sun et al. (2018) as well as Chen et al. (2019b) reduced the original architecture of VFE for faster inference times.

*Projection-based Feature Initialization.* For projection approaches, feature initialization of the cells are mostly hand-crafted. Both RV and BEV utilize fine-grained grids that pri-

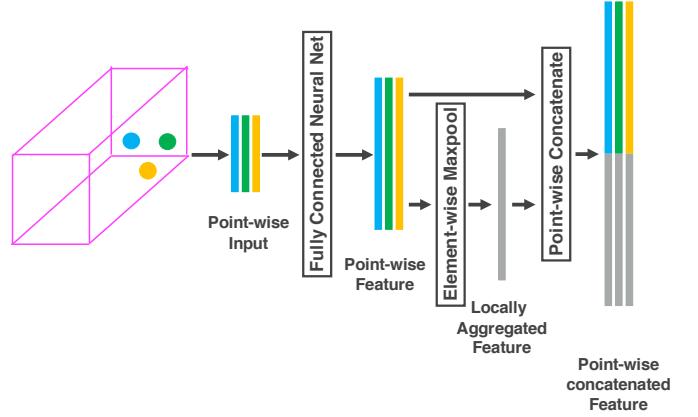


Fig. 7. Architecture of VFE module in VoxelNet (Zhou and Tuzel, 2018)

marily get filled with statistical quantities of the within lying points. Only more recently, first models started to encode the features by deep learning methods (e.g., Lehner et al., 2019; Wang et al., 2020; Zheng et al., 2020; Liang et al., 2020).

For RV, it is most popular to encode the projection map into three-channel features, namely height, distance, and intensity (Chen et al., 2017; Zhou et al., 2019; Liang et al., 2020). Instead, Meyer et al. (2019b) and Meyer et al. (2019a) are building a five-channel image with range, height, azimuth angle, intensity, and a flag indicating whether a cell contains a point. In contrast to manual encoding, Wang et al. (2020) use a point-based fully connected layer to learn high-dimensional point features of the LiDAR point cloud and then apply a max-pooling operation along the  $z$ -axis to obtain the cells’ features in RV.

Similar to RV, Chen et al. (2017) encode each cell of a BEV representation to height, intensity, and density. To increase significance of this representation, the point cloud gets divided into  $M$  slices along the  $y$ -axis, which results in a BEV map with  $M + 2$  channel features.

After the introduction to 3DOD by Chen et al. (2017), BEV representation became quite popular and many other approaches followed their proposal of feature initialization (e.g., Beltrán et al., 2018; Liang et al., 2018; Wang et al., 2018; Simon et al., 2019b; Li et al., 2019b). Yet, others choose a simpler set up such as only encoding maximum height and density without slicing the point cloud (Ali et al., 2019) or even use a binary occupancy encoding (Yang et al., 2018b; He et al., 2020).

To avoid information loss, more recent approaches initially extract features by means of deep learning approaches and then project these features into BEV (e.g., Wang et al., 2020; Zheng et al., 2020; Liang et al., 2020).

For example, Zheng et al. (2020) first initialize features of a voxelized point cloud by calculating the mean coordinates and intensities of points in each voxel. Then they apply a sparse convolution and transform the representation to a dense feature map before condensing it on the ground plane to produce a BEV feature map. Analogous to their RV approach, Wang et al. (2020) again use the learned point-wise features of the point cloud, but now apply the max pooling operation along the  $y$ -axis for feature aggregation in BEV. Liang et al. (2020), on the other hand, first extract features in RV and then transform

these to a BEV representation, adding more high-level information in comparison to directly projecting point cloud to BEV.

After the feature initialization of either volumetric grids or projected cells, segmentwise solutions usually utilize 2D/3D convolutions to extract features of the global scene.

#### 7.4.2. 2D and 3D Convolutions

Preprocessed 2D representations, such as feature initialized projections, monocular images, and RGB-D (2D) images, have the advantage that they can all leverage mature 2D convolutional techniques to extract features.

Volumetric voxelwise representations, on the other hand, present the spatial space in a regular format which are accessed by 3D convolutions. However, directly applying convolutions in 3D space is a very inefficient procedure due to the multiplication of space.

Early approaches to extend the traditional 2D convolutions to 3D were applied by Song and Xiao (2016) as well as Li (2017) by putting 3D convolutional filter in 3D space and performing feature extraction in an exhaustive operation. Since the search space increases dramatically from 2D to 3D, this procedure is associated with immense computation costs.

Further examples using conventional 3D-CNNs can be found in the models of Chen et al. (2015), Sun et al. (2018), Zhou and Tuzel (2018), Sindagi et al. (2019), and Kuang et al. (2020).

Despite delivering state-of-the-art results, the conventional 3D-CNN lacks efficiency. Given the fact that the sparsity of point clouds leads to many empty and non-discriminative voxels in a volumetric representation, the exhausting 3D-CNN operations perform a huge amount of redundant computations. This issue can be addressed by a sparse convolution.

#### 7.4.3. Sparse Convolution

Sparse convolution was proposed by Graham (2014) and Graham (2015). At first a ground state is defined for the input data. The ground state expresses whether the spatial location is active or not. A spatial location in the input representation is active if it has a non-zero value, which in the case of a regularized point cloud is a voxel that encloses at least a certain threshold of points. Additionally, a site in the following layers is active if any of the spatial location from the foregoing layer, from which it receives its input, is active. Therefore, sparse convolution must only process the sites which differ from the ground state of the preceding convolution layer, focusing computation power on the meaningful and new information present.

To lower resource cost and speed up feature extraction, processing of point clouds needs to skip irrelevant regions. Yan et al. (2018) were the first to apply sparse convolutions in 3DOD, which do not suffer but take advantage of sparsity.

Yet, the principle of sparse convolution has the disadvantage that it continuously leads to dilation of the data as it discards any non-active sites. The deeper a network gets the more the sparsity gets reduced and the data is diluted. For that reason, Graham et al. (2018) introduced submanifold sparse convolution, where the input first gets padded so that the output retains the same dimensions. Further, the output is only active if the central site of the receptive field is active, keeping the efficiency

of sparse convolution while simultaneously preserving the sparsity.

Relevant works using sparse 3D convolutions are proposed by Yan et al. (2018), Chen et al. (2019b), Shi et al. (2020b), Pang et al. (2020), Yoo et al. (2020), He et al. (2020), Zheng et al. (2020), and Deng et al. (2021).

#### 7.4.4. Voting Scheme

Another approach to exploit sparsity of point clouds is to apply a voting scheme as exemplified by Wang and Posner (2015), Engelcke et al. (2017) and Qi et al. (2019). The idea behind voting is, to let each non-zero site in the input declare a set of votes to its surrounding cells in the output layer. The voting weights are calculated by flipping the convolutional weights of the filter along its diagonal (Fernandes et al., 2021). Equivalent to sparse convolution, processing only needs to be conducted on non-zero sites. Hence, computational cost is proportional to the number of occupied sites rather than the dimension of the scene. Facing the same problem of dilation as sparse convolution, Engelcke et al. (2017) argue to select non-linear activation functions. In this case, rectified linear units help to maintain sparsity since only features with values higher than zero and not just non-zero are allowed to cast votes.

Mathematically, the feature centric voting-operation is equivalent to the submanifold sparse convolution as Wang and Posner (2015) proof in their work.

## 8. Fusion Approaches

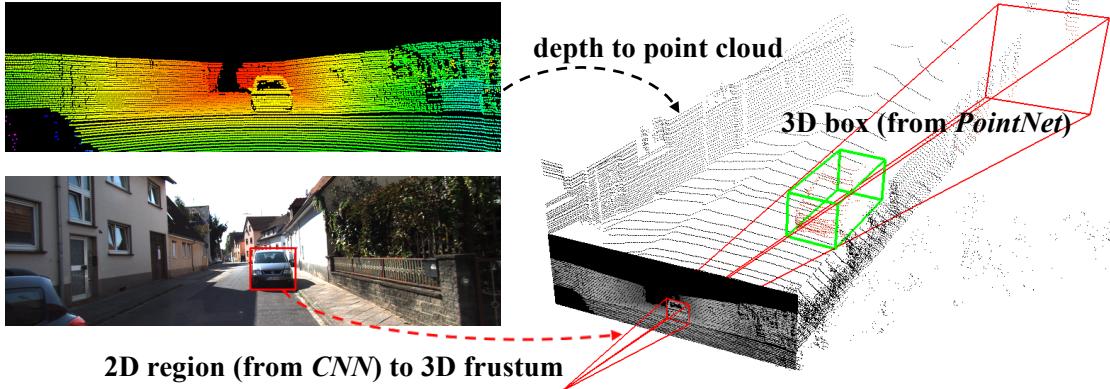
Single modality pipelines for 3DOD have developed well along the last years and have shown remarkable results. Yet, unimodal models still reveal shortcomings preventing them to reach full maturity and human-like performance. For instance, camera images are lacking depth information and suffer from truncation and occlusion, while point clouds cannot offer any texture information as well as being exposed by sparsity in longer range. To overcome these problems the attention of more recent research is increasingly directed towards fusion models that try to leverage the combination of information from different modalities.

The main challenges of fusion approaches are the synchronization of the different representations and the preservation of the relevant information during the fusion process. Further, holding the additional complexity at a computationally reasonable level must also be taken into account.

Fusion methods can be divided into two classes according to the orchestration of the modality integration, namely (i) *cascaded fusion* (Section 8.1) and (ii) *feature fusion* (Section 8.2). The former combines different sensor data and their individual features or predictions across different stages, whereas the latter jointly reasons about multi-representation inputs.

### 8.1. Cascaded Fusion

Cascaded fusion methods use consecutive single-modality detectors to restrict second-stage detection by the results of the first detector. Usually, monocular-based object detectors are leveraged in the first stage to define a restricted subset of the



**Fig. 8. Cascaded fusion scheme of Frustum-PointNets (Qi et al., 2018)**

point cloud, that is only including 3D points which are likely to be defining an object. Hence, in the second stage, 3D detectors only need to reason over a delimited 3D search space.

Two seminal works in this regard are the fusion frameworks proposed by Lahoud and Ghanem (2017) and Qi et al. (2018). The approaches use the detection results from the 2D image to extrude a corresponding frustum into 3D space. For each 2D proposal, a frustum is generated. The popular Frustum-PointNets by Qi et al. (2018) then processes the frustum with PointNet for instance segmentation. Finally, the amodal 3D box gets predicted on basis of the frustum and the extracted foreground points (see Figure 8). Lahoud and Ghanem (2017), on the other hand, first estimates the orientation for each object within the frustum. In the last step, they apply an MLP regressor for the 3D boundaries of the object.

Several approaches follow this main principle idea in a similar way. For example, Yang et al. (2018c) and Ferguson and Law (2019) both make use of 2D semantic segmentation and then project these foreground pixels of the image into a point cloud. The selected points are subsequently exploited for proposal generation through PointNet or convolution operations. Du et al. (2018) leverage the restricted 3D space by applying a model matching algorithm for detection purpose. In contrast, Shin et al. (2019) try to improve the 3D subset generation by creating point cloud region proposals with the shape of standing cylinders instead of frustums, which is more robust to sensor synchronization.

While the described models above mainly focus on the frustum creation process, Wang and Jia (2019), Zhang et al. (2020) as well as Shen and Stamos (2020) seek to advance the processing of the frustums.

By its modular nature, Frustum-PointNet is not able to provide an end-to-end prediction. To overcome that limitation, Wang and Jia (2019) subdivide the frustums to eventually make use of a fully convolutional network allowing a continuous estimation of oriented boxes in 3D space. They generate a sequence of frustums by sliding along the frustum axis and then aggregate the grouped points of each respective sections into local pointwise features. These features on frustum-level are arrayed as a 2D feature map enabling the use of a subsequent fully convolutional network.

Other than that, Shen and Stamos (2020) aim to integrate the advancements of voxelization into frustum approaches by transforming regions of interests (ROIs) within the point frustums into 3D volumetric grids. Thus, they only voxelize relevant regions, allowing a high resolution that improves the representation while still being efficient. In this case, the voxels are then fed to a 3D fully convolutional network.

More recently, Zhang et al. (2020) observed that point-based 3DOD does not perform well in longer range because of an increasing sparsity of point clouds. Therefore, they take advantage of RGB images which contain enough information to recognize faraway objects with mature 2D detectors. While following the idea of frustum generation, the estimated location of objects that are considered to be faraway are recognized. Taking into account that very few points define these objects in a point cloud, they do not possess sufficient discriminative information for neural networks so that 2D detectors are applied on corresponding images. Otherwise, for near objects, Zhang et al. (2020) use conventional neural networks to process the frustum.

## 8.2. Feature Fusion

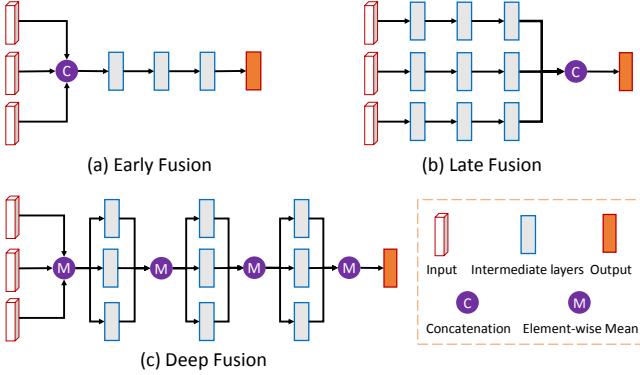
Since the performance of the cascaded fusion models is always limited by the accuracy of the detector at each stage, some researchers try to increase performance by arguing that the models should infer more jointly over the different modalities.

To this end, feature fusion methods first concatenate the information of different modalities before reasoning over the combined features, trying to exploit the diverse information in its combination. Within feature fusion, it can be further distinguished between (i) early, (ii) late, and (iii) deep fusion approaches (Chen et al., 2017), constituting at which stage of the 3DOD pipeline fusion occurs. Figure 9 provides an illustrative overview of the different fusion schemes.

*Early fusion* merges multi view features in the input stage before any feature transformation takes place and proceeds with a single network to predict the results. *Late fusion* in contrast uses multiple subnetworks that process the individual inputs separately up until the last state of the pipeline, where they get concatenated in the prediction stage. Beyond that, *deep fusion* allows an interaction of different input modalities at several stages

in the architecture and alternately performs feature transformation and feature fusion.

Although the following approaches let themselves all be categorized as feature fusion methods, the classification between the various subclasses of early, late, and deep fusion is not trivial and can be fluid. Nevertheless, the concepts help to convey a better understanding of feature fusion processes.



**Fig. 9. Early, late and deep feature fusion scheme (Chen et al., 2017)**

The pioneers among 3DOD fusion approaches are Chen et al. (2017), introducing the multi view approach. They take multiple perspectives, specifically RV, BEV and FV as input representations. The BEV representation gets utilized to generate 3D candidates, following a region-wise feature extracting by projecting these 3D proposals onto the respective feature maps of the individual views. A deep fusion scheme is then used to combine the information elementwise over several intermediate stages.

Ku et al. (2018) use 3D anchors which are mapped on both BEV representation and 2D image. Subsequently, a crop and resize operation is applied on every projected anchor and the feature crops from both views are fused via an elementwise mean operation at an intermediate convolutional layer. Unlike Chen et al. (2017), Ku et al. (2018) not only merge features in the late refinement stage, but already in the region proposal phase to generate positive proposals. They specifically use the full resolution feature map to improve prediction quality, particularly for small objects. Similar approaches to Chen et al. (2017) and Ku et al. (2018) are performed by Chen et al. (2018), Li et al. (2019b) and Wang et al. (2019), who also fuse the region of interests of the entered data representations elementwise. Yet, Chen et al. (2018) use segmentwise 3D detection for box proposals in the first stage.

In contrast, Rahman et al. (2019) not only fuse the region of interests, but combine the whole feature map of processed monocular images and FV-representations already within the region proposal stage.

Further, Ren et al. (2018) want to leverage not only object detection information but also context information. Therefore, they simply concatenate the processed features of 2D scene classification, 2D object detection and 3D object detection of the voxelized scene before feeding them to a conditional random field model for a joint optimization.

Instead of only combining regular representations through

multi view fusion models, some approaches also aim at merging raw point features with other representations.

For example, Xu et al. (2018) process the raw point cloud with PointNet. Thereafter, they concatenate each pointwise feature with a global scene feature and the corresponding image feature. Each point is then used as a spatial anchor to predict the offset to the 3D bounding box. Likewise, Yang et al. (2019) use the features of a PointNet++ backbone to extract semantic context features for each point. The sparse features subsequently get condensed by a point pooling layer to take advantage of a voxel-wise representation applying VFE.

Shi et al. (2020a) first use 3D convolution on a voxel representation to summarize the scene into a small set of key points. In the next step, these voxel-feature key points are fused with the grid-based proposals for refinement purposes.

The previously presented models all perform late or deep fusion procedures. Instead of fusing multi-sensor features per object after the proposal stage, Wang et al. (2018) follow the idea of early fusing BEV and image views by sparse non-homogenous pooling layers over the full resolution. Similarly, Meyer et al. (2019a) also deploy an early fusion but they use RV images for the point-cloud-related representation. The approach associates the LiDAR point cloud with camera pixels by projecting the 3D points onto the 2D image. The image features are then concatenated and further processed by a fully connected convolutional network.

Furthermore, several works apply a hybrid approach of early and late fusion schemes. For example, Sindagi et al. (2019) first project LiDAR to an RV representation and concatenate image features with the corresponding points in an early fusion fashion. Then they apply a VFE layer to the voxelized point cloud and append the corresponding image features for every non-empty voxel. While early concatenation already fuses features, late fusion aggregates image information for volume-based representation, where voxels may contain low-quality information due to low point cloud resolution or distant objects.

Akin, Liang et al. (2019) initially conduct feature fusion to RGB images and BEV feature maps. Thereby they incorporate multi-scale image features to augment the BEV representation. In the refinement stage, the model fuses image and augmented BEV again, but other than in the first stage, the fusion occurs elementwise for the regions of interest. They further add ground estimation and depth estimation to the fusion framework to advance the fusion process.

Simon et al. (2019a) extend with Complexer You Only Look Once (YOLO) their earlier approach Complex YOLO (Simon et al., 2019b) by exchanging the input of a BEV map for a voxel representation. To leverage all inputs, they first create a semantic segmentation of the RGB image and then fuse this picture pointwise on the LiDAR-frame to generate a semantic voxel grid.

*Continuous Fusion.* LiDAR points are continuous and sparse, whereas cameras capture dense features at a discrete state. Fusing these modalities is not a trivial task due to the one-to-many projection. In other words, not for every image pixel exists an observable corresponding LiDAR point in any projection, and vice versa.

To overcome this discontinuous mapping of images into point-cloud-based representations such as BEV, Liang et al. (2018) propose a novel continuous convolution that is applied to create a dense feature map through interpolation. It proposes to project the image feature map onto a BEV space, and then fuse the original LiDAR BEV map in a deep fusion manner through continuous convolution over multiple resolutions. The fused feature map is then further processed by a 2D-CNN, solving the discrepancy between image and projection representations.

*Attention Mechanism.* A common challenge among fusion approaches is the occurrence of noise and the propagation of irrelevant features. Previous approaches fuse multiple features simply by concatenations or elementwise summation and/or mean operations. Thereby, noise such as truncation and occlusion gets inherited to the resulting feature maps. Thus, inferior point features will be obtained through fusion. Attention mechanisms can cope with these difficulties by determining the relevance of each feature in order to only fuse features that improve the representation.

Lu et al. (2019) use a deep fusion approach for BEV and RGB images in an elementwise fashion but additionally incorporate attention modules over both modalities to leverage the most relevant features. Spatial attention adapts pooling operations over different feature map scales, while the channel-wise fusion applies global pooling. Both create an attention map that expresses the importance of each feature. Those attention maps are then multiplied with the feature map and finally fused.

Analog to this, Wang et al. (2020) use an attentive point-wise fusion module to estimate the channel-wise importance of BEV, RV, and image features. In contrast, they deploy the attention mechanism after the concatenation of the multi view feature maps to consider the mutual interference and the importance of the respective features. They specifically address the issue of ill-posed information introduced by the front view of images and RV. To compensate the inevitable loss of geometric information through the projection of LiDAR points, the authors finally enrich the fused point features with raw point features through an MLP network.

Consecutive to the attention fusion, Yoo et al. (2020) use first stage proposals from the attentive camera-LiDAR feature map to extract the single modality LiDAR and camera features of the proposal regions. Using a PointNet encoding, these are subsequently fused elementwise with the joint feature map for refinement.

Other than that, Huang et al. (2020) operate directly on the LiDAR point cloud introducing a pointwise fusion. In their deep fusion approach, they process a PointNet-like geometric - as well as a convolution-based - image stream in parallel. Between each abstraction stage, the point features get fused with the semantic image features of the corresponding convolutional layer by applying an attention mechanism.

Moreover, Pang et al. (2020) observed that elementwise fusion takes place after non-maximum suppression (NMS), which can have the consequence of mistakenly suppressing useful candidates of the individual modalities. NMS is used to suppress duplicate candidates after proposal and prediction stage, respectively. Therefore, Pang et al. (2020) use a much-reduced thresh-

old for proposal generation for each sensor and combine detection candidates before NMS. The final prediction is based on a consistency operation between 2D and 3D proposals in a late fusion fashion.

Other representative examples exploiting attention mechanism for effective fusion of features are proposed by Chen et al. (2019b) and Li et al. (2020).

A totally different approach to combine different inputs is presented by Chen et al. (2019a). For the special case of autonomous vehicles, they propose to connect surrounding vehicles with each other and combine their sensor measurements. More specifically, LiDAR data collected from different positions and angels of the connected vehicles gets fused together to provide the vehicles with a collective perception of the scene.

In summary, it is notable that the fusion of different modalities is a vivid research area within 3DOD. With continuous convolutions and attention mechanism, potential solutions for common issues, such as image-to-point-cloud discrepancies and/or noisy data representations, are already introduced. Nevertheless, fusion approaches are still exposed to several unsolved challenges. For example, 2D-driven fusion approaches like cascaded methods are always constrained by the quality of the 2D detection during the first stage. Therefore, they might fail in cases that can only be observed properly from the 3D space. Feature fusion approaches, on the other hand, generally face the difficulty to fuse different data type structures. Take the example fusing images and LiDAR data. While images provide a dense, high-resolution structure, LiDAR point clouds show a sparse structure with a comparably low resolution. The workaround to transform point clouds to another representation inevitably leads to a loss of information. Another challenge for fusion approaches is that crop and resize operations to fuse proposals of different modalities may break the feature structure derived from each sensor. Thus, a forced concatenation of a fixed feature vector size could result in imprecise correspondence between the different modalities.

## 9. Detection Module

The detection module depicts the last stage of the pipeline. It uses the extracted features to perform the multi-task consisting of classification, localization along with the bounding box regression, and object orientation determination.

Early 3DOD approaches either relied on (i) *template and keypoint matching algorithms*, such as matching 3D CAD models to the scene (Section 9.1), or (ii) suggested handcrafted SVM-classifiers using *sliding window approaches* (Section 9.2).

More recent research mainly focuses *detection frameworks based on deep learning* due to their flexibility and superior performance (Section 9.3). Detection techniques of this era can be further classified into (i) *anchor-based detection* (Section 9.4), (ii) *anchorless detection* (Section 9.5), and (iii) *hybrid detection* (Section 9.6).

### 9.1. Template and Keypoint Matching Algorithms

A natural approach to classify objects is to compare and match them to a template database. These approaches typically

leverage 3D CAD models to synthesize object templates that guide the geometric reasoning in inference. Applied matching algorithms use parts or whole CAD models of the objects to classify the candidates.

Teng and Xiao (2014) follow a surface identification approach. Therefore, they accumulate a surface object database of RGB-D images from different viewpoints. Then they match the specific 3D surface segment obtained by segmentation of the current scene to the surface segments of the database. For the pose estimation, they further match key points between the matched surface and the observed surface.

Crivellaro et al. (2015) initially perform a part detection. For each part, they project seven, so called, 3D control points which represent the pose of the object. Finally, a bounding box is estimated out of a small set of learned objects, matching the part and key point constraints.

Kehl et al. (2016) create a codebook of local RGB-D patches from synthetic CAD models. Consisting of patches from a variety of different views, those patches get matched against the feature descriptors of the scene to classify the object.

Another matching approach is designed by He et al. (2017) extending LINE-MOD (Hinterstoisser et al., 2011), which combines surface normal orientations from depth images and silhouette gradient orientations from RGB images to represent object templates. LINE-MOD is first used to produce initial detection results based on lookup tables for similarity matching. To exclude the many false positive and duplicate detections, He et al. (2017) cluster templates that matched with a similar spatial location and only then score the matchings.

Further, Yamazaki et al. (2018) applied a template matching in point cloud projections. The key novelty is to use constraints imposed by the spatial relationship among image projection directions which are linked through the shared point cloud. This allows to achieve a consistency of the object throughout the multi-viewpoint images, even in cluttered scenes.

Another approach is proposed by Barabanau et al. (2020). The authors introduce a compound solution of key point and template matching. They observe that depth estimation on monocular 3DOD is naturally ill-posed. For that reason, they propose to make use of sparse but salient key point features. They initially regress 2D key points and then match these with 3D CAD models for object dimension and orientation prediction.

## 9.2. Sliding Window Approaches

The sliding window technique was largely adopted from 2DOD to 3DOD. In fact, an object detector slides in form of a specified window over the feature map and directly classify each window position. For 3DOD pipelines, this idea is extended by replacing the 2D window with a spatial rectangular box that drives through a discretized 3D space. However, tested solutions have revealed that running a window over the entire 3D room is a very exhaustive task, leading to heavy computation and large inference times.

One of the popular pioneers in this area were Song and Xiao (2014) with their Sliding Shapes approach. They use prior trained SVM classifiers to run exhaustively over a voxelized 3D space.

Similarly, Ren and Sudderth (2016, 2018, 2020) use in all of their works a sliding window approach with extensively leveraging pre-trained SVMs. In COG 1.0, for example, they use SVMs along with a cascaded classification framework used to learn contextual relationships among objects in the scene. Therefore, they train SVMs for each object category with hand-crafted features such as surface orientation. Furthermore, they integrate a Manhattan room layout, which assumes an orthogonal room structure to estimate walls, ceilings and floors for a more holistic understanding of the 3D scene and to restrict the detection. Finally, the contextual information is used in a Markov random field representation problem to consider object relationship in detection (Ren and Sudderth, 2016).

In the successor model, namely LSS, Ren and Sudderth (2018) observe that the height of the support surface for many object categories is the primary cause of style variation. Therefore, they further add support surfaces as a latent part for each object which they use in combination with an SVM and additionally constraints from the predecessor.

Even in their latest work, COG 2.0, Ren and Sudderth (2020) still apply an exhaustive sliding window search for 3DOD, laying their focus on robust feature extraction rather than detection techniques.

Alike, Liu et al. (2018a) also use SVMs learned for each object class based on the feature selection proposed by Ren and Sudderth (2016). Through a pruning of candidates, by comparing the cuboid size of the bounding boxes with the distribution of the physical size of the objects, they further reduce inference time of detection.

However, an exhaustive sliding window approach tends to be computationally expensive, since the third dimension increases the search space significantly. Therefore, Wang and Posner (2015) leverage the sparsity of 3D representations by adding a voting scheme, that only activates on occupied cells, reducing the computation while maintaining mathematical equivalence. Whereas the sliding window approach of Song and Xiao (2014) operates linear to the total number of cells in 3D grids, voting by Wang and Posner (2015) reduces the operations exclusively to the occupied cells. The voting scheme is explained in further detail in Section 7.4.4.

Engelcke et al. (2017) tie on the success of Wang and Posner (2015) and propose to exploit feature-centric voting to detect objects in point clouds in even deeper networks to boost performance.

## 9.3. Detection Frameworks based on Deep Learning

Presenting solid solutions within their specific use cases, all the above detection techniques rely on manually designed features and are difficult to transfer. Thus, to exploit more robust features and improve detection performance, most modern detection approaches are based on deep learning models.

As with 2DOD, detection networks for 3DOD relying on deep learning can be basically grouped into two meta frameworks: (i) *two-stage detection frameworks* (Section 9.3.1) and (ii) *single-stage detection frameworks* (Section 9.3.2).

To provide a basic understanding of these two concepts, we will briefly revisit the major developments for 2D detection frameworks in the following subsections.

### 9.3.1. Two-Stage Detection Frameworks

As the name indicates, two-stage frameworks perform the object detection task in two stages. In the first stage, spatial sub-regions of the input image are identified that contain object candidates, commonly known as region proposals. The proposed regions are coarse predictions that are scored based on their "objectness". Regions with a high probability to occupy an object will achieve a high score and get passed as input to the second stage. These unrefined predictions often lack localization precision. Therefore, the second stage mainly improves the spatial estimation of the object through a more fine-grained feature extraction. The following multi-task head then outputs the final bounding box estimation and classification score.

A seminal work following this central idea is that of Girshick et al. (2014), who introduced *region-based CNN* (R-CNN). Instead of working with a huge amount of region proposals via an exhaustive sliding window procedure, R-CNN integrates the selective search algorithm (Uijlings et al., 2013) to extract just about 2,000 category-independent candidates. More specifically, selective search is based on a hierarchical segmentation approach which recursively combines smaller regions into larger ones based on color, texture, size and fill similarity. Subsequently, the 2,000 generated region proposals are cropped and warped into fixed size images in order to be fed into a pre-trained and fine-tuned CNN. The CNN acts as feature extractor to produce a feature vector with a fixed length, which can then be consumed by binary SVM classifiers that are trained for each object class independently. At the same time, the CNN features are used for the class-specific bounding box regression.

The original R-CNN framework turned out to be expensive in both time and disk space due to a lack of shared computations between the individual training steps (i.e., CNN, SVM classifiers, bounding box regressors). To this end, Girshick (2015) developed an extension, called *Fast R-CNN*, in which the individual computations were integrated into a jointly trained framework. Instead of feeding the region proposals generated by selective search to the CNN, both operations are swapped, so that the entire input image is now processed by the CNN to produce a shared convolutional feature map. The region proposals are then projected onto the shared feature map and a feature vector of fixed-length is extracted from each region proposal using a region-of-interest pooling layer. Subsequently, the extracted features are consumed by a sequence of fully connected layers to predict the results for the final object classes and the bounding box offset values for refinement purposes (Girshick, 2015). This approach saves memory and improves both, the accuracy and efficiency of object detection models (Zhao et al., 2019; Liu et al., 2020).

Both R-CNN and Fast R-CNN have the drawback that they rely on external region proposals generated by selective search, which is a time-consuming process. Against this backdrop, Ren et al. (2017) introduced another extension, called *Faster R-CNN*. As an innovative enrichment, the detection framework consists of a *region proposal network* (RPN) as a sub-network for nominating regions of interest. The RPN is a CNN by itself and replaces the functionality of the selective search algorithm. To classify objects, the RPN is connected to a Fast

R-CNN model, with which it shares convolutional layers and the resulting feature maps.

It initializes multiple reference boxes, called *anchors*, with different sizes and aspect ratios at each possible feature map position. These anchors are then mapped to a lower dimensional vector, which is used for "objectness" classification and bounding box regression via fully connected layers. These are in turn passed to the Fast R-CNN for bounding box classification and fine tuning. Due to the convolutional layers used simultaneously by the RPN and the Fast R-CNN, the architecture provides an extremely efficient solution for the region proposals (Ren et al., 2017). Furthermore, since Faster R-CNN is a continuous CNN, the network can be trained end-to-end using backpropagation iteratively and hand-crafted features are no longer necessary (Zhao et al., 2019; Liu et al., 2020).

### 9.3.2. Single-Stage Detection Frameworks

Single-stage detectors present a simpler network by transforming the input into a structured data representation and employ a CNN to directly estimate bounding box parameters and class score in a fully convolutional manner.

Object detectors based on region proposals are computationally intensive and have high inference times especially on mobile devices with limited memory and computational capacities (Redmon et al., 2016; Liu et al., 2020). Therefore, single-stage frameworks with significant time advantages have been designed, while having acceptable drawbacks in performance in comparison to the heavyweight two-stage region proposal detectors of the R-CNN family. The speed improvement results from the elimination of bounding box proposals as well as of the feature resampling phase (Liu et al., 2016). Two popular approaches which launched this development are *YOLO* (you only look once) (Redmon et al., 2016) and *SSD* (single shot multibox detector) (Liu et al., 2016).

The basic idea of YOLO is to divide the original input image into an  $S \times S$  grid. Each grid cell is responsible for both, classifying the objects within it and predicting the bounding boxes and their confidence value. However, they use features of the entire input image and not only those of proposed local regions. The use of only a single neural network by omitting the RPN allows YOLO-successor FAST YOLO to run in real-time at up to 155 frames per second. However, YOLO exhibits disadvantages in a comparable lower quality of results such as more frequent localization errors especially for smaller objects (Redmon et al., 2016).

The SSD framework also has real-time capability while not suffering as severe performance losses like YOLO. Similarly, the model consists of a single continuous CNN but uses the idea of anchors from RPN. Instead of fixed grids as in YOLO, anchor boxes of various sizes are used to determine the bounding boxes. In order to detect objects of different sizes, the predictions of several generated feature maps of descending resolution are combined. In this process, the front layers of the SSD network are increasingly used for classification due to their size, and the back layers are used for detection (Liu et al., 2016).

Thanks to regressing bounding boxes and class scores in one stage, single-stage networks are faster than two-stage ones. However, features are not learned from predicted bounding-box

proposals, but from predefined anchors. Hence, resulting prediction typically are not as accurate as those of two-stage frameworks.

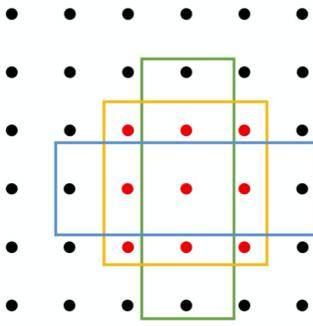
Compared to single-stage approaches, proposal-based models can leverage finer spatial information at the second stage, only focusing on the narrowed down region of interest, predicted by the first stage. Features get re-extracted for each proposal which achieves more accurate localization and classification, but in turn increases the computational costs.

The single- and two-stage paradigm can be transferred from 2DOD to 3DOD. Other than that, we want to further distinguish between the detection techniques described in the following.

#### 9.4. Anchor-based Detection

Many modern object detectors make use of anchor boxes, which serve as the initial guess of the bounding box prediction. The main idea behind anchor boxes is to define a certain set of boxes with different scales and ratios that are mapped densely across the image. This exhaustive selection should be able to capture all relevant objects. The boxes containing and fitting the objects best are finally retained.

Anchor boxes are boxes of predefined width and length. Both depict important hyperparameter to choose as they need to match those of the objects in the dataset. To consider all variation of ratios and scales, it is common to choose a collection of several sized anchor boxes (see Figure 10).



**Fig. 10.** Visualization of several anchors of different scales and ratios centering on the same feature point (Li et al., 2019d)

Anchor boxes are proposed in high numbers across the image. Typically, they are initialized at the center of each cell of the final feature map subsequent of the feature extraction stage. For the purpose of localization and classification, a network uses these anchor boxes to learn predicting the offsets between the anchors and the ground truth. Typically, by a combination of the classification confidence of each box and its overlapping to the ground truth box, called Intersection-over-Union, it is chosen which anchor boxes are discarded and which are kept for refinement purposes (Zhong et al., 2020).

In Faster R-CNN, three aspect ratios and three scales are used by default, resulting in nine anchors per location. The approach significantly reduces the amount of anchors in comparison to existing solutions. This is of particular importance as it enables a computationally acceptable integration of region proposals into the huge search space of 3DOD candidates (Song and Xiao, 2016).

#### 9.4.1. Two-Stage Detection: 2D Anchors

For 2D representations of the spatial space (e.g., BEV, RV, and RGB-D), the previously described R-CNN frameworks can be directly implemented and de facto are frequently used. Most models first predict 2D candidates from monocular representations. These predictions are then given to a subsequent network which transforms these proposals into 3D bounding boxes by applying various constraints such as depth estimation networks, geometrical constraints or 3D template matching (cf. Section 7.2.2).

As a representative example of a 2D anchor-based detection approach in 3DOD, Chen et al. (2017) use a 2D RPN at considerably sparse and low-resolution input data such as FV or BEV projections. As this data may not have enough information for proposal generation, Chen et al. (2017) assign four 2D anchors - per frame and class - to every pixel in BEV, RV and image feature map, and combine these crops in a deep fusion scheme. The 2D anchors are derived from representative 3D boxes which were obtained through clustering the ground truth objects in the training set by size and restricting the orientation to  $0^\circ$  and  $90^\circ$ . Leveraging sparsity, they only compute non-empty anchors of the last convolution feature map.

For the purpose of 3DOD, the 2D anchors are then re-projected to their original spatial dimensionality, that were derived from the ground truth boxes. Following, these 3D proposals serve the ultimate refinement regression of the bounding boxes.

Further examples are proposed by Deng and Latecki (2017), Zeng et al. (2018), Maisano et al. (2018) and Beltrán et al. (2018).

#### 9.4.2. Two-Stage Detection: 3D Anchors

Offering a relatively fast detection, 2D anchor-based detectors are not as suitable for high precision detection. Therefore, a part of research dedicates itself to the novel and more complex 3D anchor-based detection.

An initial attempt to deploy region proposals with 3D anchors was made by Chen et al. (2015) when introducing 3DOP based on handcrafted features and priors. 3DOP uses depth features in RGB-D point clouds to score candidate boxes in spatial space. A little later, Song and Xiao (2016) exploit more powerful deep learning features for candidate creation in their seminal work of deep sliding shapes. Inspired by Faster R-CNN, deep sliding shapes divides the 3D scene, obtained from RGB-D data, into voxels and then designs a 3D convolutional RPN to learn objectness for spatial region proposals. The authors define the anchor boxes for each class based on statistics. For each anchor with non-square ground planes, they define an additional anchor with the same size but rotated  $90^\circ$ . This results in a set of 19 anchors for their indoor scenarios. Given their experiments on the SUN RGB-D (Song et al., 2015) and NYUv2 (Silberman et al., 2012) datasets, the total number of anchors per image is about 1.4 million, in comparison to 2,000 anchors per RGB image frame through the selective search algorithm in an R-CNN. Thus, the huge number of anchors leads to extreme computation costs.

Apart of regular spatial representations in form of a voxelized spatial space, Xu et al. (2018) leverage a pointwise representation for anchor-based detection. The input 3D points are used

as dense spatial anchors and a prediction is performed on each of the points with two connected MLPs. Similarly, Yang et al. (2018c) define two anchors to propose a total of six 3D candidates on each point of the point cloud. To reduce the number of proposals, they apply a 2D semantic segmentation network which is mapped into the 3D space and eliminates all proposals made on background points. Subsequently, the proposals are refined and scored by a lightweight PointNet prediction network.

#### 9.4.3. Single-Stage Detection: 2D Anchors

Similar to two-stage architectures, the 2D anchor-based single-stage detector framework can be applied straightforward onto 2D-based representations. Exemplary representatives can be in the works by Liang et al. (2018), Meyer et al. (2019a), Ali et al. (2019), and He et al. (2020).

Ali et al. (2019), for instance, use the average box dimensions for each object class from the ground truth dataset as 3D reference boxes and derive 2D anchors. Then a single-stage YOLO framework is adapted on a BEV representation and two regression branches are added to produce the  $z$ -coordinate of the center of the proposal as well as the height of the box.

To enhance prediction quality, He et al. (2020) perform the auxiliary detection tasks of pointwise foreground segmentation prior to exploiting anchor-based detection. Subsequently, they estimate the object center through a 3D-CNN and only then reshape the feature maps to BEV and employ anchor-based 2D detection.

For even further improvement, Gustafsson et al. (2021) design a differentiable pooling operator for 3D to be able to extend the SA-SSD approach of He et al. (2020) by an conditional energy-based regression approach instead of the commonly used Gaussian-model.

#### 9.4.4. Single-Stage Detection: 3D Anchors

Single-stage detection networks using anchors based on 3D representations are especially committed to extract meaningful and rich features in the first place, since the 3D detection methods are not as matured as their equivalent 2D ones. Likewise, the missing performance boost has to be compensated due to the lack of the additional refinement stage.

An as exemplary approach, Zhou and Tuzel (2018) introduce the seminal VFE as discriminative feature extraction (cf. Section 7.4.1). As of today, it is the state-of-the-art encoding for voxel-wise detection. Having access to these meaningful features, they only use a simple convolutional middle layer in combination with a slightly modified RPN for a single-stage detection purpose.

Observing the problem of high inference times by 3D-CNNs in volumetric representations, Sun et al. (2018) introduce a single-stage 3D-CNN, treating detection and recognition as one regression problem in a direct manner. Therefore, they develop a deep hierarchical fusion network capturing rich contextual information.

Further exemplary representatives for single-stage 3D anchor detection are proposed by Yan et al. (2018), Li et al. (2019e), and Kuang et al. (2020) which mainly build upon the success of VoxelNet.

### 9.5. Anchorless Detection

Anchorless detection methods are commonly based on point- or segmentwise detection estimates. Instead of generating candidates, the whole scene is densely classified, and the individual objects and their respective position are derived in a direct fashion. Beside a bigger group of approaches using *fully convolutional networks (FCNs)* (Section 9.5.1) there exist several *other individual solutions* (Section 9.5.2 proposing anchor-free detection models).

#### 9.5.1. Approaches Based on Fully Convolutional Networks

Rather than exploiting anchor-based region proposal networks, Li et al. (2016) were pioneers by extending the idea of fully convolutional networks (FCNs) (Long et al., 2015) to 3DOD. The proposed 3D FCN does not need candidate regions for detection but implicitly predicts objectness over the entire image. Instead of generating multiple anchors over the feature map, the bounding box then gets directly determined over the objectness regions. Li (2017) further extend this approach in their successor model by going from depth map data to a spatial volumetric representation derived from a LiDAR point cloud.

Kim and Kang (2017) use a two-stage approach, initially predicting candidates in a projection representation based on edge filtering. Leveraging the edge detection, objects get segmented and unique box proposals are generated based on the edge boundaries. In the second stage, the authors then apply a region-based FCN to the region of interest.

Meyer et al. (2019a,b) both employ a mean shift clustering for detection. They use an FCN to predict a distribution over 3D boxes for each point of the feature map independently. In conclusion, points on the same object should predict a similar distribution. To eliminate natural noise of the prediction, they combine per-point-prediction through a mean shift clustering. Since all distributions are class-dependent and multimodal, the mean shift has to be performed for each class and modality separately. For efficiency reasons, mean shift clustering is carried out over box centers instead of box corners, reducing the dimensionality.

Further representatives using FCNs are Yang et al. (2018a,b) who use hierarchical multi-scale feature maps, and Wang and Jia (2019) who apply a sliding-window-wise application of FCN. These networks output pixelwise predictions at a single stage, with each prediction corresponding to a 3D object estimate.

#### 9.5.2. Other Approaches

Since point-based representation do not admit convolutional networks, models processing the raw point cloud need to find other solutions to apply detection mechanisms. In the following, we summarize some of the innovative developments.

A seminal work that offers a pipeline for directly working on raw point clouds was proposed by Qi et al. (2019) when introducing *VoteNet*. The approach integrates the synergies of 3D deep learning models for feature learning, namely PointNet++ (Qi et al., 2017b), and Hough voting (Leibe et al., 2008). Since the centroid of a 3D bounding box is most likely far from any surface point, the regressing of bounding box parameters that

is based solely on point clouds is a difficult task. By considering a voting mechanism, the authors generate new points that are located close to object centroids which are used to produce spatial location proposals for the corresponding bounding box. They argue that a voting-based detection is more compatible with sparse point sets as compared to RPNs, since RPNs have to carry out extra computations to adjust the bounding box without having an explicit object center. Furthermore, the center is likely to be in an empty space of the point cloud.

Figure 11 illustrates the approach. First, a backbone network based on PointNet++ is used to learn features on the points and derive a subset of points (seeds). Each seed proposes a vote for the centroid by using Hough voting (votes). The votes are then grouped and processed by a proposal module to provide refined proposals (vote clusters). Eventually, the vote clusters are classified and bounding boxes are regressed.

*VoteNet* provides accurate detection results even though it relies solely on geometric information. To further enhance this approach, Qi et al. (2020) propose *ImVoteNet*. The successor complements *VoteNet* by utilizing the high resolution and rich texture of images in order to fuse 3D votes in point clouds with 2D votes in images.

Another voting approach is proposed by Yang et al. (2020). The authors first use a voting-scheme similar to Qi et al. (2019) to generate candidate points as representatives. The candidate points are then further treated as object centers and the surrounding points are gathered to feed an anchor-free regression head for bounding box prediction.

Pamplona et al. (2019) propose an on-road object detection method where they eliminate the ground plane points and then establish an occupation grid representation. The bounding boxes are then extracted for occupation regions containing a threshold number of points. For classification purposes, PointNet is applied.

Shi et al. (2019) use PointNet++ for a foreground point segmentation. This contextual dense representation is further used for a bin-based 3D box regression and then refined through point cloud region pooling and a canonical transformation.

Besides their anchor-based solution, Shi et al. (2020b) also conduct experiments on an anchor-free solution, reusing the detection head of PointRCNN (Shi et al., 2019). They examine, that while the anchorless application being more memory efficient, the anchor-based strategy results in a higher object recall.

Similar to Shi et al. (2019), Li et al. (2020) also exploit a foreground segmentation of the point cloud. For each foreground point, an intersection-over-union (IoU)-sensitive proposal is produced, which leverages the attention mechanism. This is done by only taking the most relevant features into account, as well as further geometrical information of the surroundings. For the final prediction, they add a supplementary IoU-perception branch to the commonly used classification and bounding box regression branch for a more accurate instance localization.

Other than that, Zhou et al. (2020) introduce spatial embedding-based object proposals. A pointwise semantic segmentation of the scene is used in combination with a spatial embedding for instance segmentation. The spatial embedding

consists of an assembling of all foreground points into their corresponding object centers. After clustering, a mean bounding box is derived for each instance that is again further refined by a network based on PointNet++.

### 9.6. Hybrid Detection

Next to representation and feature extraction fusion (cf. Section 8), there are also approaches for fusing detection modules. Two-stage detection frameworks, especially representation-fusion-driven models, generally prefer to exploit 3D detection methods such as anchor-based 3D CNNs, 3D FCNs, or PointNet-like architectures for the refinement stage, after an originally lightweight 2D-based estimation was performed in the first place. This opens the advantage of a precise prediction using the spatial space by 3D detection frameworks, which are otherwise too time-consuming to be applied on the whole scene. In the following, we classify models that use multiple detection techniques as hybrid detection modules.

Exemplary, Wang and Jia (2019) apply multiple methods to finally give a prediction. First, they use an anchor-based 2D detection for proposal generation which are then extruded as frustums into 3D space. Following, an anchorless FCN detection technique gets applied to classify the frustum in a sliding window fashion along the frustum axis to output an ultimate prediction.

A frequently exercised approach is to extrude 2D proposals into spatial space of point clouds. The pioneer for this technique is Frustum-PointNet (Qi et al., 2018), enabling PointNet for the task of object detection. Since PointNet is able to effectively segment the scene, but not to produce location estimations, the authors use preceding 2D anchor-based proposals, which then are classified by PointNet.

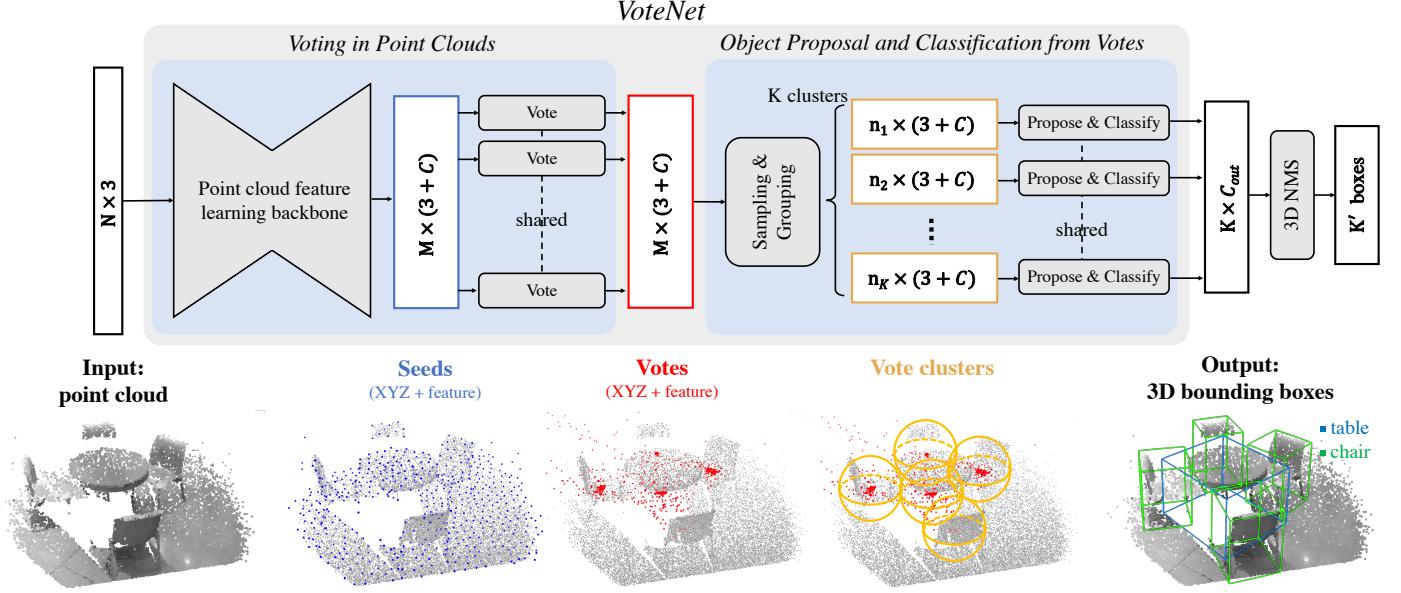
Likewise, Ferguson and Law (2019) as well as Shen and Stamos (2020) propose a quite similar idea. They first reduce the search space extruding a frustum from a 2D region proposals into 3D space and then use a 3D CNN for the detection task.

Apart of extruding frustums of the initial 2D candidates, proposals in projection representations are often converted to fully specified 3D proposals into spatial space. This is possible as projection occupy depth information enabling the mapping of the 2D representation to 3D.

For example, Zhou et al. (2019), Shi et al. (2020a), and Deng et al. (2021) transfer the proposals of the 2D-representation into a spatial representation not by extruding an unrestricted search space along the  $z$ -axis but as fully defined anchor-based 3D proposals by a 2D to 3D conversion of the projections. 3D-compatible detection methods such as anchorless and anchor-based 3D-CNNs or PointNets can then be deployed for the refinement.

Chen et al. (2019b) first generate voxelized candidate boxes that are further processed in a pointwise representation during a second stage. 3D- and 2D-CNNs are stacked upon a VFE-encoded representation for proposals in the first stage, and only then a PointNet-based network is applied for the refinement of the proposals.

Ku et al. (2019) use an anchor-based monocular 2D detection to estimate the spatial centroid of the object. An object instance



**Fig. 11. Illustration of the architecture and the steps performed by VoteNet (Qi et al., 2019)**

is then reconstructed and laid on the proposal in the point cloud, helping to regress the final 3D bounding box.

In contrast to anchor-based detection, Gupta et al. (2019) do not execute a dense pixelwise regression of the bounding box, but initially estimates key points in form of the bottom center of the object. Only those key points and their nearest neighbors are then used to produce a comparable low number of positive anchors, accelerating the detection process.

Similar to other techniques, matching algorithms are likewise fused in hybrid frameworks. For example, Chabot et al. (2017) use a network to output 2D bounding boxes, vehicle part coordinates, and 3D box dimensions. Then they match the dimensions and parts derived in the first step with CAD templates for final pose estimation. Further, Du et al. (2018) score the 3D frustum region proposals by matching them with a predefined selection consisting of three car model templates. Alike, Wang and Jia (2019) use a combination of PointNet and 2D-3D consistency constraints within the frustum to locate and classify the objects.

Instead of commonly fusing detection techniques in a hierarchical way, Pang et al. (2020) use 2D and 3D anchor-based detection in a parallel fashion to fuse the candidates in a IoU-sensitive way. There is no NMS performed before fusing the proposals because the 2D-3D consistency constraint between the proposals eliminates most elements.

In summary, hybrid detection approaches try to compensate for inferiorities of a single detection framework. While some of these models reach remarkable results, the harmonization of two different systems represents a major challenge. Next to the advantages also the disadvantage of the specific technique needs to be handled. In case of a compound solution between a 2D- and PointNet-like technique for example, the result offers an obvious improvement in inference speed, as the initial prediction is usually performed in a lightweight 2D detection framework limiting the search space for the PointNet. Yet, the accuracy and precision of detection is less favorable in com-

parison to full 3D region proposal networks, since the possible uncertainties of a 2D detection are inherited to the hierarchical next step.

## 10. Classification of 3D Object Detection Pipelines

In the previous sections, we gave a comprehensive review of different models and methods along the 3DOD pipeline and emphasized representative examples for every stage with their corresponding design options. In the following, we use our proposed pipeline framework from Figure 1 (Section 4) to classify each 3DOD approach of our literature corpus to derive a thorough systematization.

For better comparability, we distinguish all 3DOD models according to their data representation. That is, we provide separate classification schemes for (i) *monocular models* (Table 2), (ii) *RGB-D front-view-based models* (Table 3), (iii) *projection-based models* (Table 4), (iv) *volumetric grid-based models* (Table 5), (v) *point-based models* (Table 6), and (vi) *fusion-based models* (Table 7).

For each 3DOD approach, we provide information on the authors, the year and the name, classify the underlying domain and benchmark dataset(s), and categorize the specified design choices along the 3DOD pipeline.

The resulting classification can help researchers and practitioners alike to get a quick overview about the field, spot developments and trends over time, and identify comparable approaches for further development and benchmark purposes. As such, our classification delivers an overview about different design options, provides structured access to knowledge in terms of a 3DOD pipeline catalog, and offers a setting to position individual configurations of novel solutions on a more comparable basis.

Author	Year	Model	Domain	Benchmark	Sensor	Rep.	Depth Substitution	FE	Det Model	Archit.
			Outdoor	Indoor	KITTI	SUN RGB-D	others	Mono	Stereo	
								Mono		Geometric Constraint
Fidler et al.	2012	3D DPM	x	x	x			x		x
Crivellaro et al.	2015	Cluttered3DPose		x		x	x	2D-3D	x	x
Chen et al.	2016	Mono3D	x		x		x	x	GP	x
Chabot et al.	2017	Deep MANTA	x		x		x		x	x
Mousavian et al.	2017	Deep3DBox	x	x	x		x	2D-3D	x	x
Payen de La Garanderie et al.	2018	360Panoramic	x		x		x	360°	x	x
Huang et al.	2018	GGN-LON		x		x	x	2D-3D	x	x
Gupta et al.	2019	KeypointCBF	x		x		x	GP	x	x
Jørgensen et al.	2019	SS3D	x		x		x		x	x
Simonelli et al.	2019	MonoDIS	x		x		x	2D-3D	x	x
Li et al.	2019a	GS3D	x		x		x	2D-3D	x	x
Qin et al.	2019a	MonoGRNet	x		x		x	x	x	x
Naiden et al.	2019	Shift R-CNN	x		x		x	2D-3D	x	x
Liu et al.	2019a	FQNet	x		x		x	2D-3D	x	x
Brazil and Liu	2019	M3D-RPN	x		x		x	2D-3D	x	x
Barabanau et al.	2020	Keypoint3D	x		x		x	x	x	x
Qin et al.	2019b	TLNet	x		x		x	2D-3D	x	x
Ku et al.	2019	MonoPSR	x		x		x	PC	x	x

Table 2. Classification of monocular 3DOD models (360°: 360°-monocular image, PC: point cloud, 2D-3D: 2D-3D consistency, GP: groundplane)

Author	Year	Model	Domain	Benchmark	Sen.	Rep.	FE Model	Det Model	Archit.
			Outdoor	Indoor	KITTI	SUN RGB-D	NYUV2	other	
							Stereo		Handcrafted
Teng and Xiao	2014	SB-3D		x			x	x	x
Kehl et al.	2016	Patch3D		x			x	x	x
He et al.	2017	3DTemplateMatch		x			x	x	x
Yamazaki et al.	2018	3D-CORR		x			x	x	x
Li et al.	2019c	Stereo R-CNN	x		x		x	x	x
Luo et al.	2020	3D-SSD Stereo	x		x	x	x	x	x

Table 3. Classification of RGB-D front view-based 3DOD models

Author	Year	Model	Domain	Benchmark	Sensor	Rep.	Feature Initialization	FE Model	Detection Model	Archit.				
			Outdoor	KITTI	LiDAR	Bird's Eye View	Range View	Handcrafted	3D Sparse Conv	2D CNN	Anchored (2D)	Anchorless (FCN)	Single-Stage	Two-Stage
Li et al.	2016	VeloFCN	x		x	x	x	x		x	x	x	x	
Yang et al.	2018a	UberATG-HDNET	x		x	x	x	x		x	x	x	x	
Yang et al.	2018b	UberATG-PIXOR	x		x	x	x	x		x	x	x	x	
Beltrán et al.	2018	BirdNet	x		x	x	x	x		x	x	x	x	
Zeng et al.	2018	RT3D	x		x	x	x	x		x	x	x	x	
Ali et al.	2019	YOLO3D	x		x	x	x	x		x	x	x	x	
Simon et al.	2019a	Complex-YOLO	x		x	x	x	x		x	x	x	x	
Meyer et al.	2019b	LaserNet	x		x	x	x	x		x	x	x	x	
Zheng et al.	2020	CIA-SSD	x		x	x	x	x	x	x	x	x	x	

Table 4. Classification of projection-based 3DOD models

Author	Year	Model	Domain	Benchmark	Sensor	Rep.	Feature Initialization	Feature Extraction Model			Detection Model	Archit.													
								Outdoor	Indoor	KITTI	NuScenes	SUN RGB-D	NYUv2	LiDAR	Stereo	Voxel	Pillars	Handcrafted	2D CNN	3D CNN	3D Sparse CNN	Sliding Window	Anchored-based	Anchorless (FCN)	Single-Stage
Song and Xiao	2014	Sliding Shapes			x							x	x	x	x	x	x	x	x	x	SVM	3D		x	x
Chen et al.	2015	3DOP	x	x						x	x	x	x	x	x	x	x	x	x	x			x	x	
Wang and Posner	2015	Vote3D	x		x					x	x	x	x	x	x	x	x	x	x	x	Vote			x	x
Ren and Sudderth	2016	COG 1.0		x						x		x	x	x	x	x	x	x	x	x	SVM			x	x
Engelcke et al.	2017	Vote3Deep	x	x	x					x	x	x	x	x	x	x	x	x	x	x	Vote			x	x
Li	2017	3D-FCN	x	x	x					x	x	x	x	x	x	x	x	x	x	x	FCN			x	x
Yan et al.	2018	SECOND	x	x	x					x	x	x	x	x	x	x	x	x	x	x	2D	x		x	x
Zhou and Tuzel	2018	VoxelNet	x	x	x					x	x	x	x	x	x	x	x	x	x	x	3D	x		x	x
Liu et al.	2018a	3D SelSearch		x						x	x	x	x	x	x	x	x	x	x	x	SVM			x	x
Ren and Sudderth	2018	LSS	x	x	x					x	x	x	x	x	x	x	x	x	x	x	SVM			x	x
Sun et al.	2018	3D CNN	x	x	x					x	x	x	x	x	x	x	x	x	x	x	3D	x		x	x
Li et al.	2019e	3DBN	x	x	x					x	x	x	x	x	x	x	x	x	x	x	3D	x		x	x
Lang et al.	2019	PointPillars	x	x	x					x	x	x	x	x	x	x	x	x	x	x	2D	x		x	x
Ren and Sudderth	2020	COG 2.0		x						x	x	x	x	x	x	x	x	x	x	x	SVM			x	x
Shi et al.	2020b	Part-A^2 Net	x	x	x	x				x	x	x	x	x	x	x	x	x	x	x	other			x	x
Kuang et al.	2020	Voxel-FPN	x	x	x					x	x	x	x	x	x	x	x	x	x	x	3D	x		x	x

Table 5. Classification of volumetric grid-based models 3DOD models (SVM: support vector machine, Vote: voting scheme, FCN: fully convolutional network)

Author	Year	Model	Domain	Benchmark	Sensor	Rep.	FE Model	Det.	Archit.					
									Point-based	PointNet	PointNet ++	Anchorless (others)	Single-Stage	Two-Stage
Pamplona et al.	2019	On-Road3D	x	x	x	x	x	x	x	x	x	x	x	x
Shi et al.	2019	PointRCNN	x	x	x	x	x	x	x	x	x	x	x	x
Qi et al.	2019	VoteNet		x		x	x	x	x	x	x	x	x	x
Yang et al.	2020	3D-SSD LiDAR	x	x	x	x	x	x	x	x	x	x	x	x
Li et al.	2020	3D IoU-Net	x	x	x	x	x	x	x	x	x	x	x	x
Zhou et al.	2020	Joint3D Instance Seg	x	x	x	x	x	x	x	x	x	x	x	x

Table 6. Classification of point-based 3DOD models

Author	Year	Model	Domain	Benchmark	Sensor	Fusion	Representation	Depth Sub. (Mono)	Feature Extraction Model			Pointbased	Anchored-based (2D)	Anchored-based (3D)	Anchorless (FCN)	PointNet	Matching	Single-Stage	Two-Stage
									Mono	RGB-D (FV)	Projection	Segmentwise	Pointbased	Geometric Constraint	Depth Estimation Network	RGB-D	Projection (Feature Init.)	Segmentwise (Feature Init.)	
Song and Xiao	2016	DSS	x		x	x	x	x	x	x	x	x	x	x	2D CNN	3D CNN (H)		x	x
Kim and Kang	2017	Edge Affinity	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Chen et al.	2017	3M3D	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Deng and Latecki	2017	Amodial3D	x		x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Lahoud and Ghannem	2017	2D-driven	x		x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Liang et al.	2018	Uber/TG-ConfFuse	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Wang et al.	2018	FusionEV-FV	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Roddick et al.	2018	OPT-Net	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Xu and Chen	2018	MF3D	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Kui et al.	2018	AVOD/AVOD-FPN	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Du et al.	2018	PC-CNN-V2	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Yang et al.	2018c	IPOD	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Maisano et al.	2018	MobileNet-3D	x		x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Y. Ren et al.	2018	C3D		x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Xu et al.	2018	Fusion3D-PointNet	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
X. Chen et al.	2018	3DOP	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Shin et al.	2018	RoarNet	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Sindagi et al.	2019	MVN-Net	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	3D CNN (VFE)		x	x
Chen et al.	2019b	Fast Point-R-CNN	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	3D CNN (VFE)		x	x
Meyer et al.	2019a	LaserNet++	x		x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Li et al.	2019b	ComplexRetina	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Simo et al.	2019	Complexer-YOLO	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	3D CNN (DL)		x	x
Lu et al.	2019	SCANet	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Zhou et al.	2019	FVNet	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Ma et al.	2019	AM3D	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Weng and Kitani	2019	Mono3D-PLIDAR	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Liang et al.	2019	Uber/TG-MMF	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Wang et al.	2019	3D MC-CNN	x		x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Han and Jia	2019	FC-PointNet	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Tang and Lee	2019	BoxPC Fit	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)		x	x
Ferguson and Law	2019	Object R-CNN	x		x	x	x	x	x	x	x	x	x	x	2D CNN	3D CNN (H)		x	x
Lechner et al.	2019	Patches	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	3D CNN (VFE)		x	x
Yang et al.	2019	STD	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	3D CNN (VFE)		x	x
Rahman et al.	2019	ScaledDownBB	x		x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN	P++	x	x
He et al.	2020	SA-SSD	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)	P	x	x
Shen and Stamos	2020	Ray3D	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	3D CNN (H)	P	x	x
Li et al.	2020	Ray3DRCNN	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	3D CNN (H)	P	x	x
Wang et al.	2020	MFAP	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)	P	x	x
Zhang et al.	2020	FF-Net	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	2D CNN (H)	P	x	x
Pang et al.	2020	CLOC	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	3D S.-CNN (VFE)	P	x	x
Yoo et al.	2020	3D-CVF	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	3D S.-CNN (VFE)	P	x	x
Huang et al.	2020	EPENet	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	3D S.-CNN (VFE)	P	x	x
Qi et al.	2020	ImVoteNet	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	3D S.-CNN (H)	P	x	x
Shi et al.	2020a	PV-RCNN	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	3D S.-CNN (H)	P	x	x
Gustafsson et al.	2021	EBM SA-SSD	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	3D S.-CNN (H)	P	x	x
Deng et al.	2021	Voxel R-CNN	x	x	x	x	x	x	x	x	x	x	x	x	2D CNN	3D S.-CNN (H)	P	x	x

Table 7. Classification of fusion-based 3DOD models (BEV: bird's eye view, RV: range view, V: voxel, PC: point cloud, 2D-3D: 2D-3D consistency (geometrical constraint), H: handcrafted, VFE: voxel feature encoding, DL: deep learning, P: PointNet, P++: PointNet++, 3D S.-CNN: 3D sparse CNN, Vote: voting scheme)

## 11. Concluding Remarks and Outlook

3D object detection is a vivid research field with a great variety of different approaches. The additional third dimension compared to 2D vision forces to explore completely new methods, while mature 2DOD solutions can only be adopted to a limited extent. Hence, new ideas and data usage are emerging to handle the advanced problem of 3DOD resulting in a fastly growing research field that is finely branched in its trends and approaches.

From a broader perspective, we could observe several global trends within the field. For instance, a general objective of current research is clearly the efficiency optimization of increased computation and memory resource requirements due to the extra dimension of 3DOD, with the ultimate goal to finally reach real-time detection.

Additionally, it can be seen that more recent approaches increasingly focus on fully leveraging pointwise representations, since it promises the best conception of spatial space. As of the literature body of this work, PointNet-based approaches remain the only method so far that can directly process the raw point representation.

Furthermore, we observe that the fusion of feature extraction and detection techniques as well as of data representation are the most popular approaches to challenge common problems of object detection, such as amodal perception, instance variety, and noisy data. For feature fusion approaches, the development of attention mechanisms to efficiently fuse features based on their relevance is a major trend. Additionally, the introduction of continuous convolutions facilitates the complex modality mapping. In general, hybrid detection models enjoy popularity for exploiting lightweight proposals to restrict the search space for more performant but heavy refinement techniques.

Summarizing, this work aimed to complement previous surveys, such as those from Arnold et al. (2019), Guo et al. (2021) and Fernandes et al. (2021), by closing a gap of not only focusing on a single domain and/or specific methods of 3D object detection. Therefore, our search was narrowed only to the extent that the relevant literature should provide a design of an entire pipeline for 3D object detection. We purposely included all available approaches independent from the varieties of data inputs, data representations, feature extraction approaches, and detection methods. Therefore, we reviewed an exhaustively searched literature corpus published between 2012 and 2021, including more than 100 approaches from both indoor applications as well as autonomous driving applications. Since these two application areas cover the vast majority of existing literature, our survey may not be subject to the risk of missing major developments and trends.

Furthermore, a particular goal of this survey was to give an overview over all aspects of the 3DOD research field. Therefore, we provided a systematization of 3DOD methods along the model pipeline with a proposed abstraction level that is meant to be neither to coarse nor being too specific. As a result, it was possible to classify all models within our literature corpus in order to structure the field, highlight emerging trends, and guide future research.

At the same time, however, it should be noted that the several stages of the 3DOD pipeline can be designed with much broader variety and that each stage therefore deserves a much closer investigation in subsequent studies. Fernandes et al. (2021), for instance, go into much further details for the feature extraction stage by aiming to organize the entanglement of different extraction paradigms. Yet, we belief that a full conception and systematization of the entire field has not been reached.

In addition, we would like to acknowledge that, in individual cases, it might be difficult to draw a strict boundary for the classification of models regarding design choices and stages. 3D object detection is a highly complex and multi-faceted field of research, and knowledge from 2D and 3D computer vision as well as continuous progress in artificial intelligence and machine learning are getting fused.

Thus, our elaboration of the specific stages marks a broad orientation within the configuration of a 3DOD model and should be rather seen as a collection of possibilities than an ultimate and isolated choice of design options. Especially modern models often jump within these stages and do not follow a linear way along the pipeline, making a strict classification challenging.

For future research, we suggest to look into combination of methods along all stages of the 3DOD pipeline. We recommend examining these aspects independent of the pipeline, since fusion of techniques often occurs in a non-linear way.

Finally, this work could support a practical creation of individual modules or even a whole new 3DOD model, since the systematization along the pipeline can serve as an orientation of design choices within the specific stages.

## Abbreviations

2DOD	2D object detection
3DOD	3D object detection
6DoF	Six degrees of freedom
BEV	Bird's eye view
CAD	Computer-aided design
CNN	Convolutional neural networks
COG	Cloud of oriented gradients
DORN	Deep ordinal regression network
FCN	Fully convolutional networks
FPS	Farthest point sampling
FV	Front view
HOG	Histogram of oriented gradients
IoU	Intersection-over-union
LiDAR	Light Detection and Ranging
LSS	Latent support surfaces
MLP	Multi-layer perceptron
NMS	Non-maximum suppression
R-CNN	Region-based CNN
RGB-D	RGB-Depth
ROI	Regions of interest
RPN	Region proposal network
RV	Range view
SIFT	Scale-invariant feature transform
SSD	Single shot detection
SVM	Support vector machine
TOF	Time-of-Flight
VFE	Voxel feature encoding
YOLO	You Only Look Once

## References

- Ahmadyan, A., Zhang, L., Ablavatski, A., Wei, J., Grundmann, M., 2021. Objectron: A Large Scale Dataset of Object-Centric Videos in the Wild with Pose Annotations, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Nashville, TN, USA. pp. 7818–7827. doi:10.1109/CVPR46437.2021.00773.
- Ali, W., Abdelkarim, S., Zidan, M., Zahran, M., Sallab, A.E., 2019. YOLO3D: End-to-End Real-Time 3D Oriented Object Bounding Box Detection from LiDAR Point Cloud, in: Leal-Taixé, L., Roth, S. (Eds.), Computer Vision – ECCV 2018 Workshops, Springer International Publishing. pp. 716–728. doi:10.1007/978-3-030-11015-4\_54.
- Arnold, E., Al-Jarrah, O.Y., Dianati, M., Fallah, S., Oxtoby, D., Mouzakitis, A., 2019. A survey on 3d object detection methods for autonomous driving applications. *IEEE Transactions on Intelligent Transportation Systems* 20, 3782–3795. doi:10.1109/TITS.2019.2892405.
- Barabanan, I., Artemov, A., Burnaev, E., Murashkin, V., 2020. Monocular 3D Object Detection via Geometric Reasoning on Keypoints, in: Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, SCITEPRESS - Science and Technology Publications, Valletta, Malta. pp. 652–659. doi:10.5220/0009102506520659.
- Bello, S.A., Yu, S., Wang, C., Adam, J.M., Li, J., 2020. Review: Deep Learning on 3D Point Clouds. *Remote Sensing* 12, 1729. doi:10.3390/rs12111729.
- Beltrán, J., Guindel, C., Moreno, F.M., Cruzado, D., García, F., De La Escalera, A., 2018. BirdNet: A 3D Object Detection Framework from LiDAR Information, in: 2018 21st International Conference on Intelligent Transportation Systems (ITSC), pp. 3517–3523. doi:10.1109/ITSC.2018.8569311.
- Bishop, C.M., 2006. Pattern recognition and machine learning. Information science and statistics, Springer.
- Brazil, G., Liu, X., 2019. M3D-RPN: Monocular 3D Region Proposal Network for Object Detection, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 9286–9295. doi:10.1109/ICCV.2019.00938.
- Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Lioung, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O., 2020. nuScenes: A Multimodal Dataset for Autonomous Driving, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Seattle, WA, USA. pp. 11618–11628. doi:10.1109/CVPR42600.2020.01164.
- Chabot, F., Chaouch, M., Rabarisoa, J., Teuliere, C., Chateau, T., 2017. Deep MANTA: A Coarse-to-Fine Many-Task Network for Joint 2D and 3D Vehicle Analysis from Monocular Image, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Honolulu, HI. pp. 1827–1836. doi:10.1109/CVPR.2017.198.
- Chen, Q., Tang, S., Yang, Q., Fu, S., 2019a. Cooper: Cooperative Perception for Connected Autonomous Vehicles Based on 3D Point Clouds, in: 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), pp. 514–524. doi:10.1109/ICDCS.2019.00058.
- Chen, X., Kundu, K., Zhang, Z., Ma, H., Fidler, S., Urtasun, R., 2016. Monocular 3D Object Detection for Autonomous Driving, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2147–2156. doi:10.1109/CVPR.2016.236.
- Chen, X., Kundu, K., Zhu, Y., Berneshawi, A.G., Ma, H., Fidler, S., Urtasun, R., 2015. 3D Object Proposals for Accurate Object Class Detection, in: Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 28. Curran Associates, Inc., pp. 424–432.
- Chen, X., Kundu, K., Zhu, Y., Ma, H., Fidler, S., Urtasun, R., 2018. 3D Object Proposals Using Stereo Imagery for Accurate Object Class Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40, 1259–1272. doi:10.1109/TPAMI.2017.2706685.
- Chen, X., Ma, H., Wan, J., Li, B., Xia, T., 2017. Multi-view 3D Object Detection Network for Autonomous Driving, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6526–6534. doi:10.1109/CVPR.2017.691.
- Chen, Y., Liu, S., Shen, X., Jia, J., 2019b. Fast point r-CNN, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), IEEE. pp. 9774–9783. doi:10.1109/ICCV.2019.00987.
- Crivellaro, A., Rad, M., Verdie, Y., Yi, K.M., Fu, P., Lepetit, V., 2015. A Novel Representation of Parts for Accurate 3D Object Detection and Tracking in Monocular Images, in: 2015 IEEE International Conference on Computer Vision (ICCV), pp. 4391–4399. doi:10.1109/ICCV.2015.499.
- Dalal, N., Triggs, B., 2005. Histograms of Oriented Gradients for Human Detection, in: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), IEEE. pp. 886–893. doi:10.1109/CVPR.2005.177.
- Davies, E.R., 2012. Computer and machine vision: theory, algorithms, practicalities. 4th ed., Elsevier.
- Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L., 2009. Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255. doi:10.1109/CVPR.2009.5206848.
- Deng, J., Shi, S., Li, P., Zhou, W., Zhang, Y., Li, H., 2021. Voxel R-CNN: Towards High Performance Voxel-based 3D Object Detection. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 1201–1209.
- Deng, Z., Latecki, J.L., 2017. Amodal Detection of 3D Objects: Inferring 3D Bounding Boxes from 2D Ones in RGB-Depth Images, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 398–406. doi:10.1109/CVPR.2017.50.
- Du, X., Ang, M.H., Karaman, S., Rus, D., 2018. A general pipeline for 3d detection of vehicles, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE. pp. 3194–3200. doi:10.1109/ICRA.2018.8461232.
- Engelcke, M., Rao, D., Wang, D.Z., Tong, C.H., Posner, I., 2017. Vote3Deep: Fast object detection in 3D point clouds using efficient convolutional neural networks, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 1355–1361. doi:10.1109/ICRA.2017.7989161.
- Ferguson, M., Law, K., 2019. A 2D-3D Object Detection System for Updating Building Information Models with Mobile Robots, in: 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1357–1365. doi:10.1109/WACV.2019.00149.
- Fernandes, D., Silva, A., Névoa, R., Simões, C., Gonzalez, D., Guevara, M., Novais, P., Monteiro, J., Melo-Pinto, P., 2021. Point-cloud based 3d object

- detection and classification methods for self-driving applications: A survey and taxonomy. *Information Fusion* 68, 161–191. doi:10.1016/j.inffus.2020.11.002.
- Fidler, S., Dickinson, S., Urtasun, R., 2012. 3D Object Detection and Viewpoint Estimation with a Deformable 3D Cuboid Model, in: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, Curran Associates Inc., USA. pp. 611–619.
- Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM* 24, 381–395. doi:10.1145/358669.358692.
- Friederich, J., Zschech, P., 2020. Review and systematization of solutions for 3d object detection, in: 2020 15th International Conference on Wirtschaftsinformatik (WI), pp. 1699–1711. doi:10.30844/wi\_2020\_r2-friederich.
- Fu, H., Gong, M., Wang, C., Batmanghelich, K., Tao, D., 2018. Deep Ordinal Regression Network for Monocular Depth Estimation, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2002–2011. doi:10.1109/CVPR.2018.00214.
- Geiger, A., Lenz, P., Urtasun, R., 2012. Are we ready for autonomous driving? The KITTI vision benchmark suite, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3354–3361. doi:10.1109/CVPR.2012.6248074.
- Giancola, S., Valenti, M., Sala, R., 2018. A Survey on 3D Cameras: Metrological Comparison of Time-of-Flight, Structured-Light and Active Stereoscopy Technologies. *SpringerBriefs in Computer Science*, Springer International Publishing. doi:10.1007/978-3-319-91761-0.
- Girshick, R., 2015. Fast R-CNN, in: 2015 IEEE International Conference on Computer Vision (ICCV), pp. 1440–1448. doi:10.1109/ICCV.2015.169.
- Girshick, R., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, IEEE. pp. 580–587. doi:10.1109/CVPR.2014.81.
- Godard, C., Aodha, O.M., Brostow, G.J., 2017. Unsupervised Monocular Depth Estimation with Left-Right Consistency, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6602–6611. doi:10.1109/CVPR.2017.699.
- Graham, B., 2014. Spatially-sparse convolutional neural networks. arXiv:1409.6070 [cs].
- Graham, B., 2015. Sparse 3D convolutional neural networks, in: Proceedings of the British Machine Vision Conference 2015, British Machine Vision Association, Swansea. pp. 150.1–150.9. doi:10.5244/C.29.150.
- Graham, B., Engelcke, M., Maaten, L.v.d., 2018. 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9224–9232. doi:10.1109/CVPR.2018.00961.
- Griffiths, D., Boehm, J., 2019. A Review on Deep Learning Techniques for 3D Sensed Data Classification. *Remote Sensing* 11, 1499. doi:10.3390/rs11121499.
- Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., Bennamoun, M., 2021. Deep Learning for 3D Point Clouds: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43, 4338–4364. doi:10.1109/TPAMI.2020.3005434.
- Gupta, I., Rangesh, A., Trivedi, M., 2019. 3D Bounding Boxes for Road Vehicles: A One-Stage, Localization Prioritized Approach Using Single Monocular Images, in: Leal-Taixé, L., Roth, S. (Eds.), *Computer Vision – ECCV 2018 Workshops*. Springer International Publishing, Cham. volume 11133 of *Lecture Notes in Computer Science*, pp. 626–641. doi:10.1007/978-3-030-11021-5\_39.
- Gustafsson, F.K., Danelljan, M., Schon, T.B., 2021. Accurate 3D Object Detection using Energy-Based Models, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE, Nashville, TN, USA. pp. 2849–2858. doi:10.1109/CVPRW53098.2021.00320.
- He, C., Zeng, H., Huang, J., Hua, X.S., Zhang, L., 2020. Structure aware single-stage 3d object detection from point cloud, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE. pp. 11870–11879. doi:10.1109/CVPR42600.2020.01189.
- He, R., Rojas, J., Guan, Y., 2017. A 3D object detection and pose estimation pipeline using RGB-D images, in: 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 1527–1532. doi:10.1109/ROBIO.2017.8324634.
- Hinterstoesser, S., Holzer, S., Cagniart, C., Illic, S., Konolige, K., Navab, N., Lepetit, V., 2011. Multimodal templates for real-time detection of textureless objects in heavily cluttered scenes, in: 2011 International Conference on Computer Vision, IEEE. pp. 858–865. doi:10.1109/ICCV.2011.6126326.
- Huang, S., Qi, S., Xiao, Y., Zhu, Y., Wu, Y.N., Zhu, S.C., 2018. Cooperative Holistic Scene Understanding: Unifying 3D Object, Layout, and Camera Pose Estimation, in: Proceedings of the 32nd International Conference on Neural Information Processing Systems, Curran Associates Inc., USA. pp. 206–217.
- Huang, T., Liu, Z., Chen, X., Bai, X., 2020. EPNet: Enhancing Point Features with Image Semantics for 3D Object Detection, in: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (Eds.), *Computer Vision – ECCV 2020*. Springer International Publishing, Cham. volume 12360, pp. 35–52. doi:10.1007/978-3-030-58555-6\_3.
- Huang, Y., Chen, Y., 2020. Survey of state-of-art autonomous driving technologies with deep learning, in: 2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C), IEEE. pp. 221–228. doi:10.1109/QRS-C51114.2020.00045.
- Janiesch, C., Zschech, P., Heinrich, K., 2021. Machine learning and deep learning. *Electronic Markets* 31, 685–695. doi:10.1007/s12525-021-00475-2.
- Jørgensen, E., Zach, C., Kahl, F., 2019. Monocular 3D Object Detection and Box Fitting Trained End-to-End Using Intersection-over-Union Loss. arXiv:1906.08070 [cs], 1–10.
- Kehl, W., Milletari, F., Tombari, F., Ilic, S., Navab, N., 2016. Deep Learning of Local RGB-D Patches for 3D Object Detection and 6D Pose Estimation, in: Leibe, B., Matas, J., Sebe, N., Welling, M. (Eds.), *Computer Vision – ECCV 2016*. Springer International Publishing. pp. 205–220. doi:10.1007/978-3-319-46487-9\_13.
- Kim, J.U., Kang, H., 2017. LiDAR Based 3D Object Detection Using CCD Information, in: 2017 IEEE Third International Conference on Multimedia Big Data (BigMM), pp. 303–309. doi:10.1109/BigMM.2017.59.
- KITTI, 2021. Kitti 3dod benchmark. URL: [http://www.cvlibs.net/datasets/kitti/eval\\_object.php?obj Benchmark=3d](http://www.cvlibs.net/datasets/kitti/eval_object.php?obj Benchmark=3d).
- Ku, J., Mozifian, M., Lee, J., Harakeh, A., Waslander, S.L., 2018. Joint 3D Proposal Generation and Object Detection from View Aggregation, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1–8. doi:10.1109/IROS.2018.8594049.
- Ku, J., Pon, A.D., Waslander, S.L., 2019. Monocular 3D Object Detection Leveraging Accurate Proposals and Shape Reconstruction, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 11859–11868. doi:10.1109/CVPR.2019.01214.
- Kuang, H., Wang, B., An, J., Zhang, M., Zhang, Z., 2020. Voxel-FPN: Multi-scale voxel feature aggregation for 3d object detection from LIDAR point clouds. *Sensors* 20, 704. doi:10.3390/s20030704.
- Payen de La Garanderie, G., Atapour Abarghouei, A., Breckon, T.P., 2018. Eliminating the Blind Spot: Adapting 3D Object Detection and Monocular Depth Estimation to 360° Panoramic Imagery, in: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (Eds.), *Computer Vision – ECCV 2018*. Springer International Publishing, Cham. pp. 812–830. doi:10.1007/978-3-030-01261-8\_48.
- Lahoud, J., Ghanem, B., 2017. 2D-Driven 3D Object Detection in RGB-D Images, in: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 4632–4640. doi:10.1109/ICCV.2017.495.
- Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O., 2019. PointPillars: Fast encoders for object detection from point clouds, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE. pp. 12689–12697. doi:10.1109/CVPR.2019.01298.
- LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. *Nature* 521, 436–444. doi:10.1038/nature14539.
- Lefsky, M.A., Cohen, W.B., Parker, G.G., Harding, D.J., 2002. Lidar remote sensing for ecosystem studies. *BioScience* 52, 19. doi:10.1641/0006-3568(2002)052[0019:LRSFES]2.0.CO;2.
- Lehner, J., Mittrecker, A., Adler, T., Hofmarcher, M., Nessler, B., Hochreiter, S., 2019. Patch refinement – localized 3d object detection. arXiv:1910.04093 [cs].
- Leibe, B., Leonardis, A., Schiele, B., 2008. Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision* 77, 259–289. doi:10.1007/s11263-007-0095-3.
- Li, B., 2017. 3d fully convolutional network for vehicle detection in point cloud, in: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE. pp. 1513–1518. doi:10.1109/IROS.2017.

- 8205955.
- Li, B., Ouyang, W., Sheng, L., Zeng, X., Wang, X., 2019a. GS3D: An Efficient 3D Object Detection Framework for Autonomous Driving, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1019–1028. doi:10.1109/CVPR.2019.00111.
- Li, B., Zhang, T., Xia, T., 2016. Vehicle detection from 3d lidar using fully convolutional network, in: Robotics: Science and Systems XII, Robotics: Science and Systems Foundation. pp. 1–8. doi:10.15607/RSS.2016.XII.042.
- Li, J., Luo, S., Zhu, Z., Dai, H., Krylov, A.S., Ding, Y., Shao, L., 2020. 3d IoU-net: IoU guided 3d object detector for point clouds. arXiv:2004.04962 [cs].
- Li, M., Hu, Y., Zhao, N., Qian, Q., 2019b. One-stage multi-sensor data fusion convolutional neural network for 3d object detection. Sensors 19, 1434. doi:10.3390/s19061434.
- Li, P., Chen, X., Shen, S., 2019c. Stereo R-CNN Based 3D Object Detection for Autonomous Driving, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7636–7644. doi:10.1109/CVPR.2019.00783.
- Li, S., Yang, L., Huang, J., Hua, X.S., Zhang, L., 2019d. Dynamic Anchor Feature Selection for Single-Shot Object Detection, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 6608–6617. doi:10.1109/ICCV.2019.00671.
- Li, X., Guivant, J.E., Kwok, N., Xu, Y., 2019e. 3D Backbone Network for 3D Object Detection. arXiv:1901.08373 [cs].
- Liang, M., Yang, B., Chen, Y., Hu, R., Urtasun, R., 2019. Multi-task multi-sensor fusion for 3d object detection, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE. pp. 7337–7345. doi:10.1109/CVPR.2019.00752.
- Liang, M., Yang, B., Wang, S., Urtasun, R., 2018. Deep Continuous Fusion for Multi-sensor 3D Object Detection, in: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (Eds.), Computer Vision – ECCV 2018, Springer International Publishing. pp. 663–678. doi:10.1007/978-3-030-01270-0\_39.
- Liang, Z., Zhang, M., Zhang, Z., Zhao, X., Pu, S., 2020. RangeRCNN: Towards fast and accurate 3d object detection with range image representation. arXiv:2009.00206 [cs].
- Liu, J., Chen, H., Li, J., 2018a. Faster 3D Object Detection in RGB-D Image Using 3D Selective Search and Object Pruning, in: 2018 Chinese Control And Decision Conference (CCDC), pp. 4862–4866. doi:10.1109/CCDC.2018.8407973.
- Liu, L., Lu, J., Xu, C., Tian, Q., Zhou, J., 2019a. Deep Fitting Degree Scoring Network for Monocular 3D Object Detection, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1057–1066. doi:10.1109/CVPR.2019.00115.
- Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., Pietikäinen, M., 2020. Deep learning for generic object detection: A survey. International Journal of Computer Vision 128, 261–318. doi:10.1007/s11263-019-01247-4.
- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C., 2016. SSD: Single Shot MultiBox Detector, in: Leibe, B., Matas, J., Sebe, N., Welling, M. (Eds.), Computer Vision – ECCV 2016, Springer International Publishing, Cham. pp. 21–37. doi:10.1007/978-3-319-46448-0\_2.
- Liu, W., Sun, J., Li, W., Hu, T., Wang, P., 2019b. Deep learning on point clouds and its application: A survey. Sensors 19, 4188. doi:10.3390/s19194188.
- Liu, Y., Xu, Y., Li, S.b., 2018b. 2-D Human Pose Estimation from Images Based on Deep Learning: A Review, in: 2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), IEEE, Xi'an. pp. 462–465. doi:10.1109/IMCEC.2018.8469573.
- Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks for semantic segmentation, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3431–3440. doi:10.1109/CVPR.2015.7298965.
- Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60, 91–110. doi:10.1023/B:VISI.0000029664.99615.94.
- Lu, H., Chen, X., Zhang, G., Zhou, Q., Ma, Y., Zhao, Y., 2019. Scanet: Spatial-channel Attention Network for 3D Object Detection, in: ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1992–1996. doi:10.1109/ICASSP.2019.8682746.
- Luo, Q., Ma, H., Tang, L., Wang, Y., Xiong, R., 2020. 3D-SSD: Learning hierarchical features from RGB-D images for amodal 3D object detection. Neurocomputing 378, 364–374. doi:10.1016/j.neucom.2019.10.025.
- Ma, X., Wang, Z., Li, H., Zhang, P., Ouyang, W., Fan, X., 2019. Accurate Monocular 3D Object Detection via Color-Embedded 3D Reconstruction for Autonomous Driving, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 6850–6859. doi:10.1109/ICCV.2019.00695.
- Maisano, R., Tomaselli, V., Capra, A., Longo, F., Puliafito, A., 2018. Reducing Complexity of 3D Indoor Object Detection, in: 2018 IEEE 4th International Forum on Research and Technology for Society and Industry (RTSI), pp. 1–6. doi:10.1109/RTSI.2018.8548514.
- Meyer, G.P., Charland, J., Hegde, D., Laddha, A., Vallespi-Gonzalez, C., 2019a. Sensor Fusion for Joint 3D Object Detection and Semantic Segmentation, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 1230–1237. doi:10.1109/CVPRW.2019.00162.
- Meyer, G.P., Laddha, A., Kee, E., Vallespi-Gonzalez, C., Wellington, C.K., 2019b. LaserNet: An Efficient Probabilistic 3D Object Detector for Autonomous Driving, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 12669–12678. doi:10.1109/CVPR.2019.01296.
- Mousavian, A., Anguelov, D., Flynn, J., Košeká, J., 2017. 3D Bounding Box Estimation Using Deep Learning and Geometry, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5632–5640. doi:10.1109/CVPR.2017.597.
- Naiden, A., Paunescu, V., Kim, G., Jeon, B., Leordeanu, M., 2019. Shift R-CNN: Deep Monocular 3D Object Detection With Closed-Form Geometric Constraints, in: 2019 IEEE International Conference on Image Processing (ICIP), pp. 61–65. doi:10.1109/ICIP.2019.8803397.
- Otepka, J., Ghuffar, S., Waldhauser, C., Hochreiter, R., Pfeifer, N., 2013. Georeferenced point clouds: A survey of features and point cloud management. ISPRS International Journal of Geo-Information 2, 1038–1065. doi:10.3390/ijgi2041038.
- Pamplona, J., Madrigal, C., de la Escalera, A., 2019. PointNet Evaluation for On-Road Object Detection Using a Multi-resolution Conditioning, in: Vera-Rodriguez, R., Fierrez, J., Morales, A. (Eds.), Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, Springer International Publishing. pp. 513–520. doi:10.1007/978-3-030-13469-3\_60.
- Pang, S., Morris, D., Radha, H., 2020. CLOCs: Camera-LiDAR Object Candidates Fusion for 3D Object Detection, in: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, Las Vegas, NV, USA. pp. 10386–10393. doi:10.1109/IROS45743.2020.9341791.
- Qi, C.R., Chen, X., Litany, O., Guibas, L.J., 2020. ImVoteNet: Boosting 3D Object Detection in Point Clouds With Image Votes, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Seattle, WA, USA. pp. 4403–4412. doi:10.1109/CVPR42600.2020.00446.
- Qi, C.R., Hao, S., Mo, K., Leonidas, J.G., 2017a. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Honolulu, HI. pp. 77–85. doi:10.1109/CVPR.2017.16.
- Qi, C.R., Litany, O., He, K., Guibas, L., 2019. Deep Hough Voting for 3D Object Detection in Point Clouds, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 9276–9285. doi:10.1109/ICCV.2019.00937.
- Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J., 2018. Frustum PointNets for 3D Object Detection from RGB-D Data, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 918–927. doi:10.1109/CVPR.2018.00102.
- Qi, C.R., Yi, L., Su, H., Guibas, L.J., 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space, in: Advances in Neural Information Processing Systems, Curran Associates, Inc.. pp. 1–10.
- Qin, Z., Wang, J., Lu, Y., 2019a. MonoGRNet: A geometric reasoning network for monocular 3d object localization. Proceedings of the AAAI Conference on Artificial Intelligence 33, 8851–8858. doi:10.1609/aaai.v33i01.33018851.
- Qin, Z., Wang, J., Lu, Y., 2019b. Triangulation Learning Network: From Monocular to Stereo 3D Object Detection, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7607–7615. doi:10.1109/CVPR.2019.00780.
- Rahman, M.M., Tan, Y., Xue, J., Shao, L., Lu, K., 2019. 3d object detection: Learning 3d bounding boxes from scaled down 2d bounding boxes in RGB-d

- images. *Information Sciences* 476, 147–158. doi:10.1016/j.ins.2018.09.040.
- Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You only look once: Unified, real-time object detection, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE. pp. 779–788. doi:10.1109/CVPR.2016.91.
- Ren, S., He, K., Girshick, R., Sun, J., 2017. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39, 1137–1149. doi:10.1109/TPAMI.2016.2577031.
- Ren, Y., Chen, C., Li, S., Kuo, C.C.J., 2018. Context-Assisted 3D (C3D) Object Detection from RGB-D Images. *Journal of Visual Communication and Image Representation* 55, 131–141. doi:10.1016/j.jvcir.2018.05.019.
- Ren, Z., Sudderth, E.B., 2016. Three-Dimensional Object Detection and Layout Prediction Using Clouds of Oriented Gradients, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1525–1533. doi:10.1109/CVPR.2016.169.
- Ren, Z., Sudderth, E.B., 2018. 3D Object Detection with Latent Support Surfaces, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 937–946. doi:10.1109/CVPR.2018.00104.
- Ren, Z., Sudderth, E.B., 2020. Clouds of Oriented Gradients for 3D Detection of Objects, Surfaces, and Indoor Scene Layouts. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42, 2670–2683. doi:10.1109/TPAMI.2019.2923201.
- Roddick, T., Kendall, A., Cipolla, R., 2018. Orthographic Feature Transform for Monocular 3D Object Detection. arXiv:1811.08188 [cs].
- Sager, C., Janiesch, C., Zschech, P., 2021. A survey of image labelling for computer vision applications. *Journal of Business Analytics* 4, 91–110. doi:10.1080/2573234X.2021.1908861.
- Shen, X., Stamos, I., 2020. Frustum VoxNet for 3d object detection from RGB-d or depth images, in: 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE. pp. 1687–1695. doi:10.1109/WACV45572.2020.9093276.
- Shi, S., Guo, C., Jiang, L., Wang, Z., Shi, J., Wang, X., Li, H., 2020a. PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Seattle, WA, USA. pp. 10526–10535. doi:10.1109/CVPR42600.2020.01054.
- Shi, S., Wang, X., Li, H., 2019. PointRCNN: 3D Object Proposal Generation and Detection From Point Cloud, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–779. doi:10.1109/CVPR.2019.00086.
- Shi, S., Wang, Z., Shi, J., Wang, X., Li, H., 2020b. From Points to Parts: 3D Object Detection from Point Cloud with Part-aware and Part-aggregation Network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–1doi:10.1109/TPAMI.2020.2977026.
- Shin, K., Kwon, Y.P., Tomizuka, M., 2019. RoarNet: A Robust 3D Object Detection based on RegiOn Approximation Refinement, in: 2019 IEEE Intelligent Vehicles Symposium (IV), pp. 2510–2515. doi:10.1109/IVS.2019.8813895.
- Silberman, N., Hoiem, D., Kohli, P., Fergus, R., 2012. Indoor Segmentation and Support Inference from RGBD Images, in: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (Eds.), Computer Vision – ECCV 2012, Springer, Berlin, Heidelberg. pp. 746–760. doi:10.1007/978-3-642-33715-4\_54.
- Simon, M., Amende, K., Kraus, A., Honer, J., Samann, T., Kaulbersch, H., Milz, S., Gross, H.M., 2019a. Complexer-YOLO: Real-Time 3D Object Detection and Tracking on Semantic Point Clouds, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE, Long Beach, CA, USA. pp. 1190–1199. doi:10.1109/CVPRW.2019.00158.
- Simon, M., Milz, S., Amende, K., Gross, H.M., 2019b. Complex-YOLO: An Euler-Region-Proposal for Real-Time 3D Object Detection on Point Clouds, in: Leal-Taixé, L., Roth, S. (Eds.), Computer Vision – ECCV 2018 Workshops, Springer International Publishing. pp. 197–209. doi:10.1007/978-3-030-11009-3\_11.
- Simonelli, A., Bulò, S.R., Porzi, L., Lopez-Antequera, M., Kortscheder, P., 2019. Disentangling Monocular 3D Object Detection, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1991–1999. doi:10.1109/ICCV.2019.00208.
- Sindagi, V.A., Zhou, Y., Tuzel, O., 2019. MVX-Net: Multimodal VoxelNet for 3D Object Detection, in: 2019 International Conference on Robotics and Automation (ICRA), pp. 7276–7282. doi:10.1109/ICRA.2019.8794195.
- Song, S., Lichtenberg, S.P., Xiao, J., 2015. SUN RGB-D: A RGB-D scene understanding benchmark suite, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 567–576. doi:10.1109/CVPR.2015.7298655.
- Song, S., Xiao, J., 2014. Sliding Shapes for 3D Object Detection in Depth Images, in: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (Eds.), Computer Vision – ECCV 2014, Springer International Publishing. pp. 634–651. doi:10.1007/978-3-319-10599-4\_41.
- Song, S., Xiao, J., 2016. Deep Sliding Shapes for Amodal 3D Object Detection in RGB-D Images, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 808–816. doi:10.1109/CVPR.2016.94.
- Srivastava, S., Jurie, F., Sharma, G., 2019. Learning 2D to 3D Lifting for Object Detection in 3D for Autonomous Vehicles, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4504–4511. doi:10.1109/IROS40897.2019.8967624.
- Sun, H., Meng, Z., Du, X., Ang, M.H., 2018. A 3D Convolutional Neural Network Towards Real-Time Amodal 3D Object Detection, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 8331–8338. doi:10.1109/IROS.2018.8593837.
- Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., Vasudevan, V., Han, W., Ngiam, J., Zhao, H., Timofeev, A., Ettinger, S., Krivokon, M., Gao, A., Joshi, A., Zhang, Y., Shlens, J., Chen, Z., Anguelov, D., 2020. Scalability in Perception for Autonomous Driving: Waymo Open Dataset, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2443–2451. doi:10.1109/CVPR42600.2020.00252.
- Tang, Y.S., Lee, G.H., 2019. Transferable Semi-Supervised 3D Object Detection From RGB-D Data, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1931–1940. doi:10.1109/ICCV.2019.00202.
- Teng, Z., Xiao, J., 2014. Surface-based general 3D object detection and pose estimation, in: 2014 IEEE International Conference on Robotics and Automation (ICRA), pp. 5473–5479. doi:10.1109/ICRA.2014.6907664.
- Uijlings, J.R.R., van de Sande, K.E.A., Gevers, T., Smeulders, A.W.M., 2013. Selective search for object recognition. *International Journal of Computer Vision* 104, 154–171. doi:10.1007/s11263-013-0620-5.
- Viola, P., Jones, M.J., 2004. Robust Real-Time Face Detection. *International Journal of Computer Vision* 57, 137–154. doi:10.1023/B:VISI.000013087.49260.fb.
- Wang, D.Z., Posner, I., 2015. Voting for voting in online point cloud object detection, in: Robotics: Science and Systems XI, Robotics: Science and Systems Foundation. pp. 1–9. doi:10.15607/RSS.2015.XI.035.
- Wang, G., Tian, B., Zhang, Y., Chen, L., Cao, D., Wu, J., 2020. Multi-view adaptive fusion network for 3d object detection. arXiv:2011.00652 [cs].
- Wang, L., Li, R., Shi, H., Sun, J., Zhao, L., Seah, H.S., Quah, C.K., Tandianus, B., 2019. Multi-Channel Convolutional Neural Network Based 3D Object Detection for Indoor Robot Environmental Perception. *Sensors* 19, 1–14. doi:10.3390/s19040893.
- Wang, Y., Ye, J., 2020. An overview of 3d object detection. arXiv:2010.15614 [cs].
- Wang, Z., Jia, K., 2019. Frustum ConvNet: Sliding Frustums to Aggregate Local Point-Wise Features for Amodal, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1742–1749. doi:10.1109/IROS40897.2019.8968513.
- Wang, Z., Zhan, W., Tomizuka, M., 2018. Fusing Bird's Eye View LiDAR Point Cloud and Front View Camera Image for 3D Object Detection, in: 2018 IEEE Intelligent Vehicles Symposium (IV), pp. 1–6. doi:10.1109/IVS.2018.8500387.
- Weng, X., Kitani, K., 2019. Monocular 3D Object Detection with Pseudo-LiDAR Point Cloud, in: 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), pp. 857–866. doi:10.1109/ICCVW.2019.00114.
- Xu, B., Chen, Z., 2018. Multi-level Fusion Based 3D Object Detection from Monocular Images, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2345–2353. doi:10.1109/CVPR.2018.00249.
- Xu, D., Anguelov, D., Jain, A., 2018. PointFusion: Deep Sensor Fusion for 3D Bounding Box Estimation, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 244–253. doi:10.1109/CVPR.2018.00033.
- Yamazaki, T., Sugimura, D., Hamamoto, T., 2018. Discovering Correspon-

- dence Among Image Sets with Projection View Preservation For 3D Object Detection in Point Clouds, in: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3111–3115. doi:10.1109/ICASSP.2018.8461677.
- Yan, Y., Mao, Y., Li, B., 2018. SECOND: Sparsely Embedded Convolutional Detection. *Sensors* 18, 1–17. doi:10.3390/s18103337.
- Yang, B., Liang, M., Urtasun, R., 2018a. HDNET: Exploiting HD Maps for 3D Object Detection, in: Proceedings of The 2nd Conference on Robot Learning, PMLR, pp. 146–155.
- Yang, B., Luo, W., Urtasun, R., 2018b. PIXOR: Real-time 3D Object Detection from Point Clouds, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7652–7660. doi:10.1109/CVPR.2018.00798.
- Yang, Z., Sun, Y., Liu, S., Jia, J., 2020. 3DSSD: Point-Based 3D Single Stage Object Detector, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Seattle, WA, USA. pp. 11037–11045. doi:10.1109/CVPR42600.2020.01105.
- Yang, Z., Sun, Y., Liu, S., Shen, X., Jia, J., 2018c. IPOD: Intensive Point-based Object Detector for Point Cloud. arXiv:1812.05276 [cs].
- Yang, Z., Sun, Y., Liu, S., Shen, X., Jia, J., 2019. STD: Sparse-to-dense 3d object detector for point cloud, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), IEEE, pp. 1951–1960. doi:10.1109/ICCV.2019.00204.
- Yoo, J.H., Kim, Y., Kim, J., Choi, J.W., 2020. 3D-CVF: Generating Joint Camera and LiDAR Features Using Cross-view Spatial Feature Fusion for 3D Object Detection, in: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (Eds.), Computer Vision – ECCV 2020. Springer International Publishing, Cham. volume 12372, pp. 720–736. doi:10.1007/978-3-030-58583-9\_43.
- Zeng, Y., Hu, Y., Liu, S., Ye, J., Han, Y., Li, X., Sun, N., 2018. RT3D: Real-Time 3-D Vehicle Detection in LiDAR Point Cloud for Autonomous Driving. *IEEE Robotics and Automation Letters* 3, 3434–3440. doi:10.1109/LRA.2018.2852843.
- Zhang, H., Yang, D., Yurtsever, E., Redmill, K.A., Özgüner, U., 2020. Faraway-frustum: Dealing with lidar sparsity for 3d object detection using fusion. arXiv:2011.01404 [cs].
- Zhao, Z.Q., Zheng, P., Xu, S.T., Wu, X., 2019. Object detection with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems* 30, 3212–3232. doi:10.1109/TNNLS.2018.2876865.
- Zheng, W., Tang, W., Chen, S., Jiang, L., Fu, C.W., 2020. CIA-SSD: Confident IoU-aware single-stage object detector from point cloud. arXiv:2012.03015 [cs].
- Zhong, Y., Wang, J., Peng, J., Zhang, L., 2020. Anchor Box Optimization for Object Detection, in: 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE, Snowmass Village, CO, USA. pp. 1275–1283. doi:10.1109/WACV45572.2020.9093498.
- Zhou, D., Fang, J., Song, X., Liu, L., Yin, J., Dai, Y., Li, H., Yang, R., 2020. Joint 3d instance segmentation and object detection for autonomous driving, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE. pp. 1836–1846. doi:10.1109/CVPR42600.2020.00191.
- Zhou, J., Tan, X., Shao, Z., Ma, L., 2019. FVNet: 3D Front-View Proposal Generation for Real-Time Object Detection from Point Clouds, in: 2019 12th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), pp. 1–8. doi:10.1109/CISP-BMEI48845.2019.8965844.
- Zhou, Y., Tuzel, O., 2018. VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4490–4499. doi:10.1109/CVPR.2018.00472.
- Zia, M.Z., Stark, M., Schindler, K., 2015. Towards Scene Understanding with Detailed 3D Object Representations. *International Journal of Computer Vision* 112, 188–203. doi:10.1007/s11263-014-0780-y.