3D Object Detection For Intelligent Vehicles

MSc thesis

J.R. van der Sluis

3D Object Detection For Intelligent Vehicles

MSc thesis

by

J.R. van der Sluis

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Monday October 26, 2020 at 14:00.

Student number: 4444426
Project duration: August 26, 2019 – September 22, 2020
Thesis committee:

Prof. Dariu M. Gavrila,	TU Delft, supervisor
Ir. Ewoud A.I. Pool,	TU Delft, daily supervisor
Dr. Wei Pan	TU Delft
Dr. Jan C. van Gemert	TU Delft

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Preface

This master thesis is performed at the Delft University of Technology, the Cognitive Robotics department, the section of Intelligent Vehicles. The goal of this thesis is to identify the current limitations of 3D object detection methods inside the domain of the intelligent vehicles.

Although it was a period of change because of the COVID-19 pandemic, I enjoyed doing research at Intelligent Vehicles section.

I would like to thank Ir. Ewoud A.I. Pool and Prof. Dr. Dariu M. Gavrila for their great supervision and appreciated constructive feedback during the whole graduation process. Besides my supervisors, I would like to express my gratitude towards my family, roommates, and girlfriend for their unconditional support during the past years. I would like to thank all the staff of the Intelligent Vehicle section for their support, the lunches together, and the nice conversations.

Last but not least, as a co-creator of the MSc. Robotics, I would like to wish all the new students, now and in the coming years, all the best.

J.R. van der Sluis
Delft, October 2020

Abstract

This master thesis presents an experimental study on 3D person localization (*i.e.*, pedestrians, cyclists) in traffic scenes, using monocular vision and Light Detection And Ranging (LiDAR) data. The performance of two top-ranking methods is analyzed on the 3D object detection KITTI dataset. In this evaluation, the effect of the Intersection over Union (IoU) threshold on the performance in terms of 3D bounding box location, size, and orientation is analysed.

Since the KITTI 3D object detection dataset contains relatively few 3D person instances, the analysis will is to the EuroCity Persons 2.5D (ECP2.5D) datasets (both day and night), which is one order of magnitude larger. Using both datasets, additional experiments are performed to evaluate the influence of distance, the number of LiDAR points, occlusion, and intensity on the performance. Domain transfer experiments between the KITTI and ECP2.5D datasets are performed, to examine how these datasets generalize with respect to each other.

Furthermore, Part-A^2 net is used to evaluate the detection score which is given to the ground truth pedestrians. The relationship between the detection score and the distance, the number of LiDAR points, and occlusion is analyzed. Some objects are not detected although their ground truth detection score is high. This creates the potential to detect these pedestrians.

Lastly, this thesis presents a method that uses the detections from the previous frame to increase the performance in the subsequent frame by adding the previous detections to the 3D proposals coming from the Region Proposal Network (RPN).

Contents

1 Introduction **1**
- 1.1 Research questions . . . 2
- 1.2 Submitted articles . . . 3

2 Related work **5**
- 2.1 Theoretical foundation of 3D object detectors . . . 5
  - 2.1.1 Neural Network basis . . . 5
  - 2.1.2 Optimizing a Neural network . . . 6
  - 2.1.3 Convolutional layers . . . 7
  - 2.1.4 General object detection pipelines . . . 8
  - 2.1.5 Non-maximum suppression . . . 10
- 2.2 Single-sensor modality networks . . . 10
- 2.3 Multi-sensor modality networks . . . 11
- 2.4 PointPillars and AVOD . . . 12
- 2.5 Datasets . . . 12
- 2.6 Part-A^2 net . . . 13
- 2.7 Temporal fusion . . . 13
- 2.8 Contributions . . . 14

3 Methodology **15**
- 3.1 Intersection over Union (IoU) . . . 15
- 3.2 Performance metrics . . . 16
- 3.3 Adaption to Part-A^2 net . . . 16
- 3.4 KITTI RAW 3D sequence dataset . . . 17
  - 3.4.1 KITTI RAW dataset . . . 17
  - 3.4.2 Relation KITTI RAW and KITTI 3D dataset . . . 18
  - 3.4.3 KITTI RAW 3D sequence dataset . . . 18

4 Experiments **19**
- 4.1 Datasets overview . . . 19
- 4.2 Effect of IoU on performance and error analysis . . . 22
- 4.3 Effect of distance and LiDAR points on performance . . . 24
- 4.4 Effect of occlusions on performance . . . 25
- 4.5 Cross-dataset evaluations . . . 28
- 4.6 Qualitative analysis ECP2.5D . . . 28
- 4.7 Detection score of ground truth pedestrians . . . 31
- 4.8 Temporal fusion . . . 33

5 Discussion **37**

6 Conclusion **41**
- 6.1 What are the limitations of current SoA 3D object detectors? . . . 41
  - 6.1.1 What is the effect of the IoU constraint on the performance and error analysis? . . . 41
  - 6.1.2 How does the performance change over distance? . . . 41
  - 6.1.3 What is the relationship between the number of LiDAR points on an object and the performance? . . . 42
  - 6.1.4 What is the effect of occlusion on the performance? . . . 42
  - 6.1.5 How does a network perform if cross-dataset evaluated? . . . 42
  - 6.1.6 How confident is a detection head about ground truth pedestrians? . . . 42

6.2 What is the effect on the performance if detections from the previous frame(s) are added to the list of proposals coming from the RPN? 42
6.3 Future work 42
6.3.1 Future work thesis 43
6.3.2 General future work of the field of 3D object detection 43

A Published and Submitted articles **47**
A.1 Article published at the IEEE Intelligent Vehicles Symposium 2020 47
A.2 Article to be submitted to the IEEE Transactions on Intelligent Vehicles 54

Bibliography **69**

1

Introduction

More than 1.35 million people died in a road traffic accident in 2016 [71]. Pedestrians and cyclists are the most susceptible to injuries in traffic, since they are not protected by any active safety mechanisms, like airbags. World wide 54% of the deaths were related to Vulnerable Road Users (VRUs) (*i.e.*, pedestrians, cyclists and riders). Thomas *et al*. [60] identified that most of these accidents are caused by human error. 51 % of these accidents where cars were involved, were related to bad timing of the car driver. Automation could help reduce the number of traffic deaths since it does not have the same weaknesses as a human. It will always have the same level of alertness since it will not get tired or distracted, nor will it be under the influence of alcohol. In order to protect the VRUs, intelligent vehicles need to detect them first.

An intelligent vehicle can have several levels of automation ranging from 0 to 5. Level 0 indicates a car without any driving automation and level 5 indicates full driving automation [51]. In order to reach higher levels of automation, the intelligent vehicle needs to take over more aspects of the dynamic driving task. Research into intelligent vehicles is increasingly becoming a topic of interest. For example, in the United States of America (USA), where Waymo already drove over 5 million autonomous miles over the past 9 years [69]. Besides companies, research institutes are also investigating several aspects of autonomous driving. Examples are the Intelligent Systems Lab (LSI) at Universidad Carlos III de Madrid [1] and the Intelligent Vehicles Group of the Delft University of Technology. An example of a research-setup of an automated car can be seen in fig. 1.1. This figure shows the test vehicle of the TU Delft which is equipped with Light Detection And Ranging (LiDAR) (on top), stereo cameras (behind the windshield), and radar sensors (around the car) to perceive the environment.

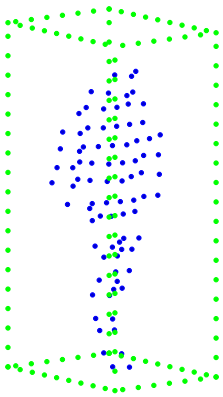
In order to protect VRUs, the car needs to detect them first. This detection can either be performed in 2D or 3D. The task of a 2D detector is to draw a 2D bounding box around the objects of interest. The goal of a 3D detector is, instead of drawing a 2D box, to draw a 3D bounding box around the objects of interest as shown in fig. 1.2a. In fig. 1.2 one can observe a LiDAR point cloud with a 3D box around it, and an image of the same person.

Currently, State-of-the-Art (SoA) 3D object detectors are compared based on AP_{3D} (this will be explained in section 3.2). Although it is good that 3D object detectors are made comparable by a performance metric, this number does not tell us the weak spots of the detectors. Therefore, the metric and the potential weak spots of SoA detectors will be investigated (research question 1).

Research question 1 is divided into six sub research questions that together answer the main question. First, the influence of one of the parameters of the AP_{3D} on the performance metric will be identified, namely the Intersection over Union (IoU). This parameter represents the overlap between an object and its prediction (research question 1.1.). Secondly, the relationship between the performance of the detectors over the distance (research question 1.2.) and the number of LiDAR points (research question 1.3.) will be investigated. During this investigation, a closer look will be taken at the effect of occlusion (research question 1.4.) on the relationship between the performance over distance and the number of LiDAR points. The performance of the detectors will be investigated by evaluating them on another dataset than they are trained on (research question 1.5.).

Some networks are explicitly divided into two stages: the first stage is called a region proposal network, it searches for regions of interest. These regions of interest are then sent to the second stage,

Figure 1.1: The TU Delft Toyota Prius. This car is equipped with different types of sensors, one can observe a LiDAR on top of the car. The car is also equipped with stereo cameras and radars. (Adapted from Palffy [41])



(a) A point cloud of a pedestrian (blue) with a ground truth bounding box around it (green)

(b) An image of a pedestrian

Figure 1.2: An point cloud and image of a pedestrian. The point cloud and image are extracted from the EuroCity Persons 2.5D (ECP2.5D) [8]

which does the actual detection. For a two-stage network, the efficacy of the detection head will be evaluated by supplying it with the ground truth bounding boxes, instead of the region proposals (research question 1.6.). The second stage also leads to a way for potentially improving the performance by adding additional regions of interest.

Ground truth is of course not available during inference, but detections from the previous frame are. These detections could potentially improve performance (research question 2).

Lastly, this thesis will contain an extensive discussion on the future work of 3D object detectors in section 6.3.2. This section is based on my view and ideas of the 3D object detection field resulting from my literature study, published/submitted articles, and additional thesis work.

1.1. Research questions

This thesis will look into the performance of SoA 3D object detectors and will investigate current bottlenecks of 3D object detectors. Besides the limitations of the detectors, there will be looked into improving 3D object detection.

1. What are the limitations of current SoA 3D object detectors?

   1.1. What is the effect of the IoU constraint on the performance and error analysis?
   1.2. How does the performance change over distance?
   1.3. What is the relationship between the number of LiDAR points on an object and the performance?
   1.4. What is the effect of occlusion on the performance?
   1.5. How does a network perform if cross-dataset evaluated?
   1.6. How confident is a detection head about ground truth pedestrians?
2. What is the effect on the performance if detections from the previous frame(s) are added to the list of proposals coming from the Region Proposal Network (RPN)?

1.2. Submitted articles

During this master thesis, two articles were written. The first article will be presented at the 2020 IEEE Intelligent Vehicles Symposium (IV2020) titled “An Experimental Study on 3D Person Localization in Traffic Scenes” by Joram R. van der Sluis, Ewoud A.I. Pool, and Dariu M. Gavrila [62].

The second article will be submitted to IEEE Transactions on Intelligent Vehicles (T-IV) titled “An Experimental Study on 3D Person Localization in Traffic Scenes” by Joram R. van der Sluis, Ewoud A.I. Pool, and Dariu M. Gavrila [63]. This second article is an extension of the first article. Since this article is not submitted yet, it could be that the appended version will contain some differences regarding the final submitted version.

This master thesis will contain large parts of these articles, especially in parts regarding research question 1, including sub research questions 1.1.-1.5.. Research question 1.6. and 2 are a follow-up to both articles and therefore not discussed in these articles. Both articles are appended to this master thesis.

2

Related work

This related work will focus on previous 3D object detection methods that use neural network architectures, as they are the current best performers in the various benchmarks. One way to categorize these detectors is by sensor modality, *i.e.*, either a single modality or a fusion of multiple modalities. This related work discusses the existing networks in detail using this categorization. The commonly used sensors used are (monocular) camera and Light Detection And Ranging (LiDAR). However, the RGB-only methods (*e.g.* Shift R-CNN [40]) are generally outperformed by methods that use LiDAR information instead. Therefore, only single-sensor methods using LiDAR information will be reviewed. Although recently a 3D object detector based on a camera image and radar was published [39, 42], the radar modality will not be taken into account in this thesis since it's very challenging to compare LiDAR and radar because there is no benchmark available containing both the modalities.

Another way of categorizing 3D object detection methods is by the number of stages used by the network. Two-stage approaches utilize a Region Proposal Network (RPN) to generate bounding boxes which are individually evaluated (*e.g.* STD [74] and Part-A^2 net[52]). Single-state approaches instead evaluate predetermined bounding boxes (*e.g.* PointPainting [64] and 3DSSD[75]), also called anchor boxes. Before discussing these categories, first some basic concepts about the Neural Networks behind the 3D object detectors will be explained.

2.1. Theoretical foundation of 3D object detectors

All State-of-the-Art (SoA) 3D object detectors are based on Neural Networks (NNs). Therefore, this chapter will explain the theoretical foundation of 3D object detectors.

3D object detectors often use NNs. These NNs need an input (e.g. an image) and a *ground truth* to compare its output to. During *training*, the *weights* in the *(convolutional) layers* are optimized in order to predict the correct output. To do so, one needs to define a *loss function* which is used by the NN to know where the NN should focus on. This loss is based on the detections of the network and the ground truth. All of the italic terms in this section will be explained.

This chapter will mainly focus on Feedforward Neural Networks (FNNs). An example of such a network is shown in fig. 2.1. FNNs are different from Recurrent Neural Networks (RNNs) in the sense that RNNs takes in sequences and can keep track of the past inputs using an internal state. In this thesis, there will be referred to FNNs as NNs. The goal of this chapter is to investigate the relevant NN basics for 3D object detection networks.

2.1.1. Neural Network basis

A Neural Network (NN) consists of several layers; an input layer, hidden layer(s), and an output layer. An example of a NN is shown in fig. 2.1. A FNN can consist of multiple hidden layers. Inside each hidden layer, there are several neurons. The number of hidden layers in fig. 2.1 is two and the number of neurons (shown in blue) in each hidden layer equals five. These neurons transform an input using learned weights and a learned bias. To the output of neurons is often referred as features. Neurons can be described by a mathematical function that takes in array of numbers and outputs features based on the input, weights and biases (see eqs. (2.1) and (2.2)). In eq. (2.1), one can observe that each neuron

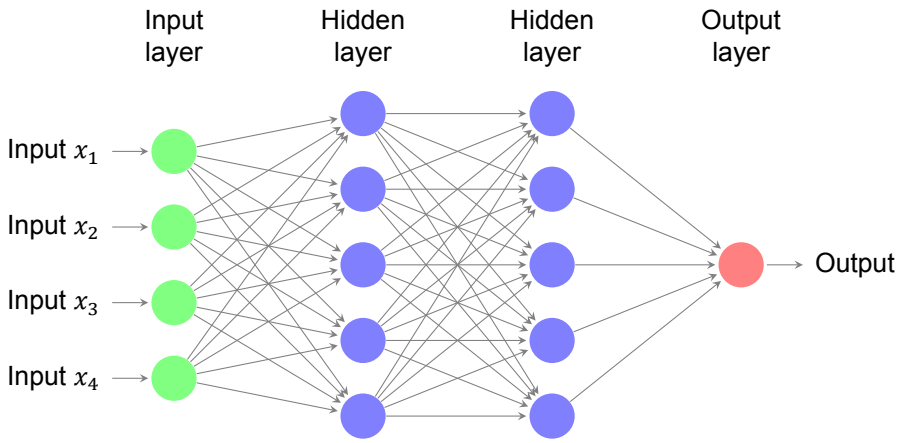


Figure 2.1: An example of a feedforward neural network. The green dots represent the input of the network (e.g. RGB values). The blue dots represent the nodes inside a hidden layer, where the input is multiplied with some trainable weights and biases. Here two hidden layers are shown. The red dot is the output layer where all the output of each hidden layer is summed.

takes an array of inputs (x), this array is multiplied with the weights (w) of the neuron and a bias (b) of the neuron is added. To introduce non-linearity, one can apply one of several activation functions (g) to the output of each node. The mathematical representation is shown in eq. (2.2), which shows that for each output of a node (z), an activation function($g(z)$) is applied. The output of an activation function is denoted as h. Examples of activation functions are a sigmoid, hyperbolic tangent (tanh) or Rectified Linear Unit (ReLU) as can be found in respectively equation eqs. (2.5) to (2.7) and shown in fig. 2.2. The combination of eqs. (2.1) and (2.2) will result in eq. (2.3), which describes one layer. These hidden layers can be placed in sequence after each other as done in fig. 2.1 which can be written as eq. (2.4).

$$z = \left(\sum_{i=0}^{n} w_i x_i\right) + b \tag{2.1}$$

$$h = g(z) \tag{2.2}$$

$$f(x) = g\left(\left(\sum_{i=0}^{n} w_i x_i\right) + b\right) \tag{2.3}$$

$$y = f^{(2)} f^{(1)}(x) \tag{2.4}$$

$$g(z) = \frac{1}{1 + \exp(-z)} \tag{2.5}$$

$$g(z) = \tan(z) \tag{2.6}$$

$$g(z) = \max(0, z) \tag{2.7}$$

2.1.2. Optimizing a Neural network

The goal of the NN is to solve a problem based on the given input. An example of a problem is 3D object detection where a point cloud is fed to the network and the network's goal is to predict the location, dimensions and orientation of the pedestrians. To succeed, the network needs to be optimized to predict the best output for each input. The process of optimizing a NN is called training, where example inputs are provided to the network and the output of the network is compared to the ground truth. The comparison (based on a loss function) between the ground truth and the output of the NN causes the NN to adapt the weights to decrease the loss and thereby the error of the network (also called backpropagation). The adaptation is done by modifying the weights of the network, or so-called

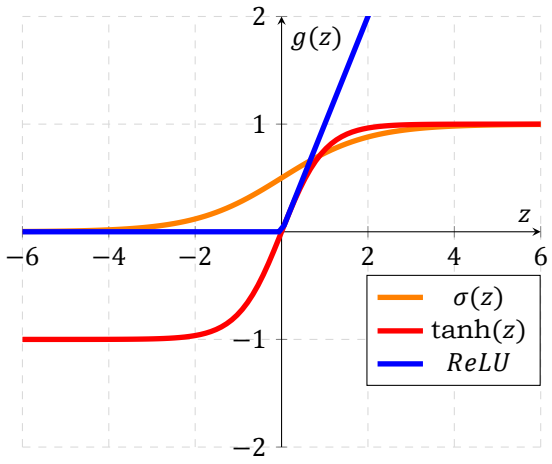


Figure 2.2: This plot shows three different types of activation functions ($g(z)$) which can be applied to the output of a node (z). The x-axis show the output of a node before the activation function is applied and the y-axis show the result after applying the activation function.

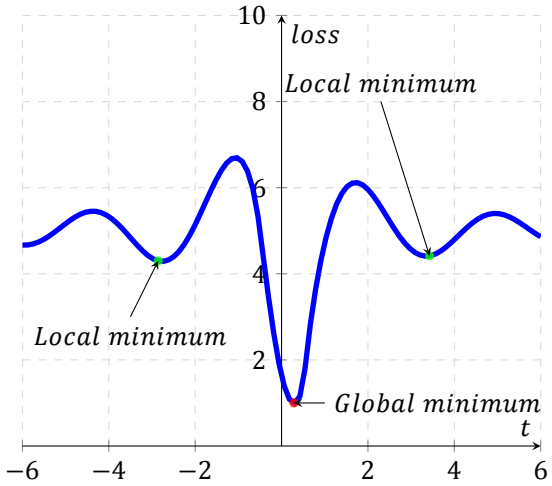


Figure 2.3: A example of a loss function with a global minimum and local minimum. The local minima are shown in green while the global minimum is shown in red.

updating of the weights. Since most SoA networks have more than one million weights, it is no option to adapt them by hand. The solution is to use deep learning to learn these weights. Therefore, one needs to define a loss function (also called loss or objective function) which determines the performance of the network.

To optimize the weights, one can use backpropagation (BP) to compute the gradient of each of the weights. To calculate the BP, the chain rule is used to determine the gradients efficiently. After the gradients are determined, a gradient descent algorithm (e.g. Momentum [47], Adam [23], RMSprop [61]) can be used to find the local or ideally global minimum of the loss function. An example of a loss function with a global and local minimum is shown in fig. 2.3.

2.1.3. Convolutional layers

In image classification and object detection, the number of parameters (weights) in the fully connected layers can be large en cause the NN to overfit. A solution to decrease the number of parameters is the use of convolutions. The layers that apply convolutions are called convolutional layers. They do not have regular weights but they instead have a matrix of weights which they shift over a matrix of input data. This input data is most often consisting of image data. In fig. 2.4 one can observe an example of a 2D convolution. In this thesis, the sizes of a filter used for 2D convolutions are defined as in eq. (2.8) where the 2D filter size is defined as a multiplication of the filter Height (H), filter Width(W),

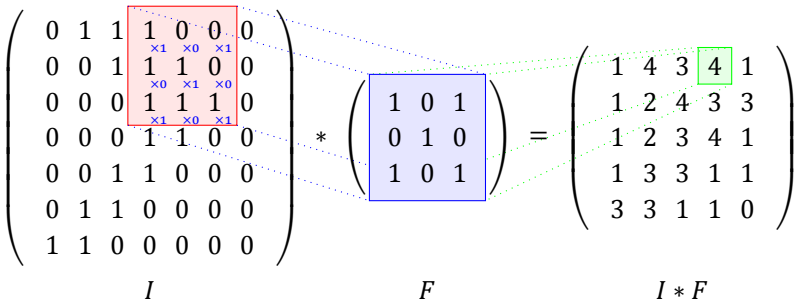


Figure 2.4: An example of a 2D convolution with matrix I and a $3 \times 3 \times \times C_I \times C_O$ filter F. One can observe the result of this convolution on right ($I * F$)

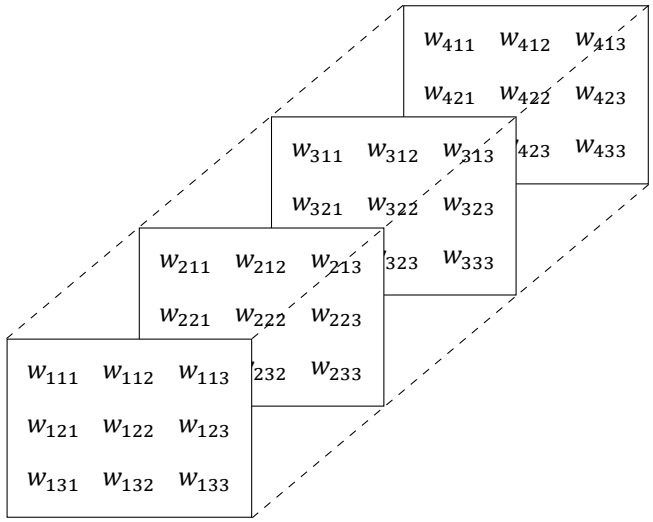


Figure 2.5: An example of a 3D filter $4 \times 3 \times 3 \times \times C_I \times C_O$, where w denote the weights.

input channels (C_I) and output channels/kernels (C_O). Here, $3 \times 3 \times \times C_I \times C_O$ filter F is convolved with Matrix I. If one talks about 3D convolution, the Depth of the filter (D) is added to the filter used for 3D convolution compared to 2D as can be found in eq. (2.9). One can look at such a 3D filter as multiple stacked 2D filters, in the case of fig. 2.5, 4 filters are stacked together.

$$2D\ filter\ size = H \times W \times C_I \times C_O \tag{2.8}$$

$$3D\ filter\ size = D \times H \times W \times C_I \times C_O \tag{2.9}$$

2.1.4. General object detection pipelines

To detect objects (e.g. pedestrians, riders) one needs an object detector. Object detectors can be subdivided into two main network architectures: single-stage and two-stage. A single-stage network has a certain amount of proposed bounding boxes at fixed locations where every proposal is evaluated using a bounding box regressor. Two-stage networks do not have these fixed bounding boxes, they propose their bounding boxes based on the output of the hidden layers, this is also called a Region Proposal Network (RPN). An example of a RPN is shown in fig. 2.6. The proposed bounding boxes are evaluated by a bounding box regressor one can observe in fig. 2.7. The bounding box regressor is an important development in the field of object detection, originally proposed by Girshick *et al.* [18]. A bounding box regressor uses the output of a hidden layer, also called feature layer, to refine the location and dimensions of the proposed bounding boxes.

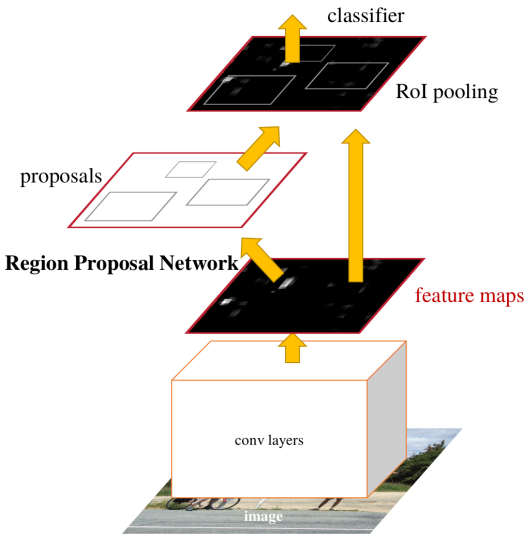


Figure 2.6: This is an example of a Region Proposal Network (RPN). The classifier only classify the features of the proposals generated by the RPN. (Adapted from Faster R-CNN [48])

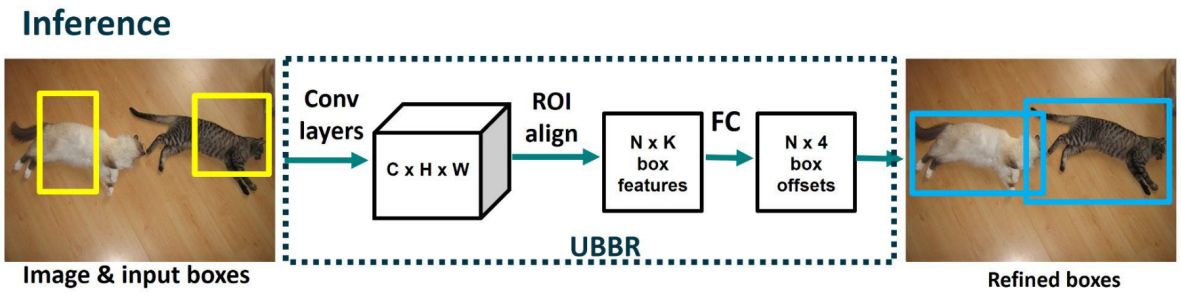


Figure 2.7: This is an example of a bounding box regressor, here a network refines the input bounding boxes during inference to predict refined bounding boxes. (Obtained from Lee *et al*. [29])

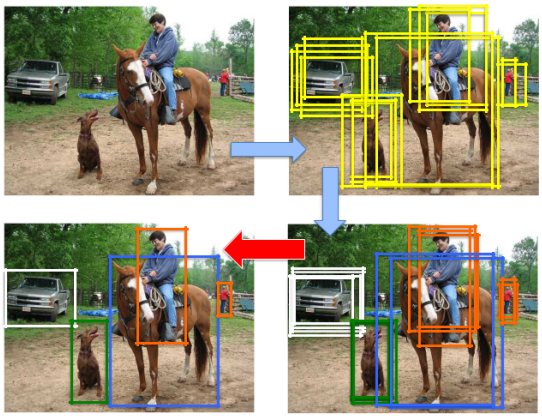


Figure 2.8: This is an example of applied 2D NMS. One can distinguish three stages: the proposal generation, class assignment, and finally the red arrow shows the NMS step. (from Bodla *et al*. [7])

2.1.5. Non-maximum suppression

Non-maximum suppression (NMS) is often applied at the end of an object detector. The goal of NMS is to reduce the number of proposals for the same object, an example of NMS is shown in fig. 2.8. The proposals are provided by an object detector and can be seen as the unfiltered detections. NMS, as published by Bodla *et al*. [7], generally consists of three steps. First, the proposal with the highest confidence (detection score) on the original detection list is added to the final list of outputs and removed from the original proposal list. Secondly, the overlap between this highest-ranked proposal and the other proposals is calculated (also referred to as Intersection over Union (IoU), which is explained in section 3.1). Thirdly, All proposals with an overlap higher than a preset threshold are removed from the original proposal list and thereby discarded. This process repeats until there are no more detections left on the original proposal list. The final detections will consist of all the detections on the final list of outputs.

2.2. Single-sensor modality networks

LiDAR-only networks map the point cloud to either a 2D or a 3D representation. Arnold *et al*. [2] compare different 2D and 3D LiDAR representations. They state that if the LiDAR point cloud is mapped to 2D, it suffers from a loss of information compared to 3D methods. A benefit of mapping to 2D is that conventional CNN architectures used for image detection can be used as a basis to obtain the 3D bounding boxes. Examples of 2D representations are Birds Eye View (used by *e.g*. HDNet [73]) and Range View (*e.g*. LaserNet [38]).

One group of methods that exploits the 3D information in a point cloud do so by dividing the data into voxels, and then afterwards apply 3D convolutions (*e.g*. Voxelnet [78] , TANet [33] and PV-RCNN [54]). The downside of applying 3D convolutions is the reduced efficiency when processing empty voxels and the generally higher computational costs for 3D convolutions. PointPillars [28] address this by creating “pillars”. The are a discretization of the point cloud into an equally spaced grid of pillars in the x-y plane. PointPillars additionally exploits the sparseness of a LiDAR scene by limiting both the number of pillars (P) in each sample and the number of LiDAR points per pillar (N). If a sample contains more than P non-empty pillars or a pillar contains more than N points, a random subset of respectively the pillars or points is taken in order to match P or N. Other 3D representations are Stixels (*e.g*. SCNet [66]) or Graph representations (*e.g*. Point-GNN [55]).

An alternative to dividing a point cloud into voxels is PointNet [44]. PointNet takes a fixed number of points as input, which are transformed by an input and feature transformation. Global features are extracted using max-pooling which results in an output score per class. PointNet can also be used for segmentation, in this case the features of each point are concatenated with the global features to output a score per point. Although the results seem promising for object classification, this method alone is not suitable to use for object detection since it requires one single object per point cloud. PointNet++[46] and Frustum PointNet [45] both address this shortcoming. PointNet++ has a hierarchical network structure that applies PointNet recursively on parts of the point cloud. This results in the

Table 2.1: Comparison of AVOD and PointPillars.

	AVOD	PointPillars
Modality	LiDAR + image	LiDAR
Stages	Two-stage	Single-stage
Bounding box regression	Four corners, heights, orientation	3D center point, length, width, height, orientation

Table 2.2: Overview of traffic-related 3D persons datasets. "Unknown" denotes that the information could not be determined. The number between brackets specifies the number of cameras/LiDAR sensors with that resolution/number of planes. [*] The lyft dataset uses 2 types of vehicles. These have a resolution of 1224×1024 (6) + 2048×864 (1) and 1920×1080 (7). The number of LiDAR planes are 40 (3) and 40 (2) + 64 (1).

Dataset	**Waymo** [59]	**nuScenes** [9]	**Argoverse** [10]	**Lyft** [22]	**KITTI** [16]	**ECP2.5D** [8]
# Countries	1	2	1	1	1	12
# Cities	2	2	2	1	1	30
# Frames	800K	34K	350K	55K	15K	46K
Image resolution	1920×1280 (3) + 1920×1040 (2)	1600×900 (6)	1920×1200 (7) + 2056×2464 (2)	* (7)	1240×376 (1)	1920×1024 (1)
# LiDAR planes	unknown (5)	32 (1)	32 (2)	* (3)	64 (1)	64 (1)
# Peds	2.8M	222K	132K	25K	9.4K	123K
# Riders	67K	24K	11K	22K	3.3K	13K
# Seasons	unknown	unknown	1	1	1	4
Weather	dry, rain	dry, rain	dry	dry	dry	dry, rain
Time of day	day, night	day, night	day, night	unknown	day	day, night
Unblurred	✗	✗	✗	✗	✓	✓

ability to capture local features. Frustum PointNet exploits PointNets for 3D object detection, an image detector is used to select regions of the point cloud, the subsets of the point cloud are passed into a PointNet for classification and prediction of 3D bounding boxes.

2.3. Multi-sensor modality networks

Multi-sensor modality networks, or fusion networks, use both camera and LiDAR. Hence, all the previously mentioned LiDAR mappings can be used to fuse with the camera data. The fusion types can be categorised in three categories

The first category is early fusion, where the modalities are concatenated before being passed into a neural network. An example of early fusion is MVX-Net PointFusion [58] where the point cloud is projected onto an RGB-image and then concatenated.

Secondly, deep fusion networks fuse the modalities after they have already been processed by a part of the network, for example, PointFusion[72]. Here, the features from a PointNet [44] and a ResNet-50 are concatenated. With deep fusion, it is also possible to fuse the various modalities at multiple stages, as is done with AVOD [25]. Within such a deep fusion network, the performance is dependent on the feature encoder used [50]. Because the different modalities are already processed by their part of the network, the data is transferred info features. When fusing the features, a decision can be made based on data from multiple modalities.

Thirdly, late fusion takes the output of two or more independent networks and fuses the class probabilities [3]. Since late fusion takes the output of multiple independent models, it has high flexibility because the models can be different for every data modality, trained independently, and more sensors can be added if they become available. The downside of late fusion is that only the outcome of the networks is fused. The internal features which are used by the network to make the decision are not shared with the other networks.

A specific form of late fusion is sequential fusion, which processes the sensor modalities in sequence. For example, Frustum PointNets [45] and Frustrum Convnet [68] use a 2D image detector to select frustums in a point cloud, which is then processed separately. The benefit of sequential fusion is that both networks can be trained independently. The downside is the strong dependence of the different parts upon each other, if one part fails the other will not be able to recover the loss of information.

Feng *et al*. [14] discuss the benefits and downsides of different fusion methods. They state that if the modalities are fused before entering the network, the network can fully exploit all the information from the modalities if the data is properly aligned. Another benefit is the low computational cost because the network does not need separate networks that process these modalities, but instead shares a part of the computational load. A benefit of late fusion is the added flexibility of selecting a different, more optimal architecture for each sensor modality. Additionally, as each model can be trained separately, there is no need for one dataset that combines all required sensor modalities [49].

2.4. PointPillars and AVOD

PointPillars [28] and AVOD [25], two of the best performing LiDAR and fusion networks, respectively, with code available at the time of writing, will be used later in the experiments and are therefore explained in detail. PointPillars first converts the point cloud into a pillar representation. Then, a simplified version of PointNet [44] is applied to each pillar, resulting in a sparse pseudo image. In the second step, the sparse pseudo image is passed onto a backbone which extracts top-down features at different upsampling factors by 2D convolutions. The final step includes a Single Shot Detector [32] setup which outputs the detections.

AVOD exploits the information from both images and LiDAR point clouds. AVOD uses the same method as MV3D [12] to generate a Birds Eye View (BEV) representation of the point cloud. The point cloud is divided into 3D cells by slicing the height dimension into 5 equal parts between 0 and 2.5 meter and divide the point cloud into a 2D grid with a resolution of 0.1 meter in the horizontal plane. Each of the 3D cells is encoded with a height feature that corresponds to the maximum height of the LiDAR point inside this cell, together with a feature that represents the number of points inside a cell. This results in a 6 channel BEV. The difference between AVOD and MV3D is that AVOD neglects intensity.

The image and BEV representation are both passed into a feature extractor that consists of an encoder and a decoder. The encoder is an adapted version of a VGG-16 [57] and the decoder is similar to the Feature Pyramid Network [31]. The output of the feature extractor is used by both the RPN and the second stage of the detection network. Based on the proposals of the RPN the features from the image and point cloud are fused, and subsequently used for detection. Table 2.1 summarizes the differences between PointPillars [28] and AVOD [25].

2.5. Datasets

In terms of existing datasets, one of the first 3D object detection benchmarks was an extension to KITTI [16], released in 2017, which contains around 9.4K pedestrians (of which half in the publicly available training set). Since then, KITTI has become the de facto standard for 3D object detection. However, because of the relatively small dataset size, performances can differ a lot on the validation and test set. In this thesis, KITTI will refer to the KITTI 3D Object Detection Evaluation dataset. More recent dataset additions to KITTI are significantly larger and more diverse, see table 2.2. Waymo, nuScenes, Argoverse, and Lyft also contain sequences rather than independent scenes and have annotated all objects over time. Additionally, they have annotated pedestrians in 360 degrees around the vehicle, and provide enough camera images to see all around the vehicle. The EuroCity Persons 2.5D (ECP2.5D) [8] dataset is, while lacking annotations overtime, unique due to the diversity in its urban scenes recorded in different European cities in various weather conditions. Another observation is the difference between the location of pedestrians in European cities compared to the United States of America (USA). In the USA, annotated pedestrians are most commonly located on either the sidewalk or a pedestrian crossing at an intersection as shown in fig. 2.9. Pedestrians in Europe are located at a variety of locations (*e.g.* on the road). Both the KITTI and ECP2.5D, which are used in the experiments, are recorded in Europe and have occlusion levels labeled per object. Occlusion levels are split into three categories: fully visible, partly occluded, and largely occluded. Objects that are partly and largely occluded will be referred to as occluded objects.

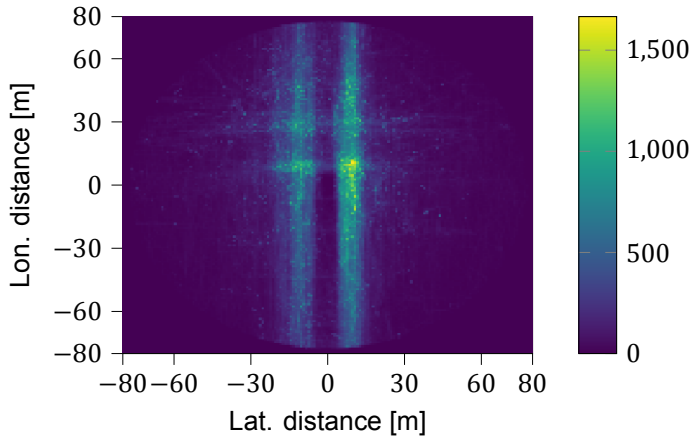


Figure 2.9: The overall distribution over location of pedestrians in the Waymo dataset [59]. The egovehicle is positioned at $(0, 0)$, looking upwards. Each pixel in the image corresponds to a 1×1 square meter area.

2.6. Part-A^2 net

Part-A^2 net [52] is, at the time of writing, the best performing two-stage network with code available, containing 3D bounding box proposals at a stage where both the feature extraction and the regression still needs to take place. Part-A^2 net is a LiDAR-only method that consists of two stages, the part-aware stage and the part-aggregation stage. The network is unique because it learns to predict the location of a LiDAR point within a 3D bounding box and performs segmentation to distinguish objects (foreground) from background points. This 3D object detector is described more in detail because it is used in the experiments of this thesis.

Part-A^2 net builds upon a PointRCNN [53], another 3D object detection method. PointRCNN is a LiDAR-only network which uses an encoder-decoder structure to extract point-wise features from a raw point cloud. The first stage of PointRCNN uses the point-wise extracted features to generate 3D bounding boxes. The second stage of the network refines the 3D proposals using the point-wise features and the LiDAR points location for each 3D proposal.

The first stage of Part-A^2 net first voxelizes the point cloud into voxels and extracts the features of all nonempty voxels. The voxelized point cloud is then passed to an encoder-decoder structure as can be seen in fig. 2.10. The encoder produces a down-sampled 2D BEV feature map. This feature map is used by RPN to predict 3D proposals. The encoder-decoder outputs the features (shown in light blue, at the right of the decoder in fig. 2.10) which are used for intra-object part prediction and the segmenting of the foreground points.

After the generation of the 3D proposals and the prediction of the intra-part information, the 3D proposals are given their final location, size, and orientation in the Part-aggregation stage. The added 3D proposals are then evaluated in the Part-aggregation using their corresponding features of the encoder-decoder structure, the intra-object part prediction, and the segmentation of the foreground points (referred to as semantic segmentation in fig. 2.10). The output of the network is a set of 3D boxes with a confidence rating (also referred to as a detection score).

2.7. Temporal fusion

Bergmann *et al*. [5] published a temporal fusion method for 2D object detection. Their method exploits the bounding box regression of a 2D object detector. Two types of methods were developed: Tracktor and Tracktor++, which is an extension of Tracktor. Tracktor uses the resulting bounding boxes from the previous frame and it uses bounding box regression to evaluate those in the current frame. As an addition to Tracktor, Bergmann *et al*. released Tracktor++ which also incorporates a motion model and a re-identification algorithm.

Another example of a 3D object detection network using temporal fusing is FaF (Fast and Furious) [35], it fuses the LiDAR voxel representations from the past five timesteps together using 3D convolutions. They show, using their own, not publicly available dataset, an increase in performance of 6% (AP_{3D}) looking at the car class for an IoU of 70%.

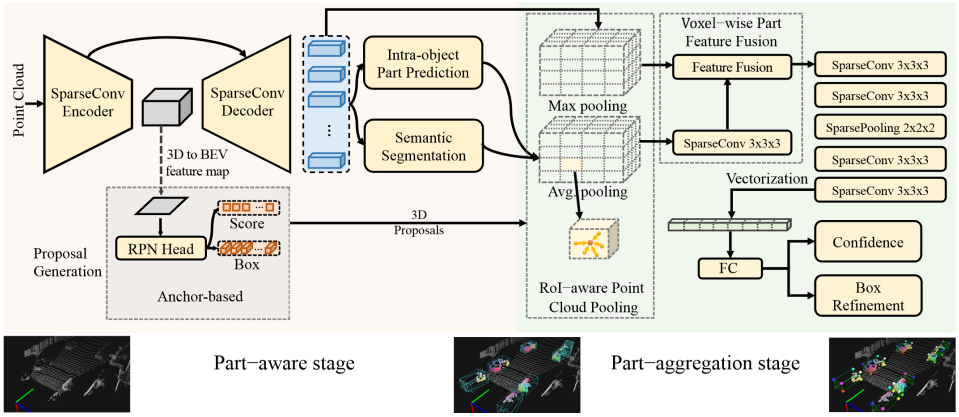


Figure 2.10: The two-stage network of Part-A^2 net. Image adapted from Part-A^2 net [52].

Instead of fusing the raw data, one could also exploit the information in the resulting features of previous point clouds as done Huang *et al*. [20], they fed the features into an Long Short-Term Memory (LSTM) to predict 3D objects. Another example is McCrae *et al*. [37], who recently published a method based on PointPillars where an LSTM is implemented after the 2D backbone of the network. They show increasing performance on the nuScenes validation dataset.

RV-FuseNet[26] uses the range view representation of multiple point clouds to detect the location of an object and predict the location in the future.

2.8. Contributions

This master thesis presents an experimental study on monocular and LiDAR-based 3D person detection. Its specific contributions are:

- A performance analysis of two state-of-the-art methods (PointPillars and AVOD) on KITTI, with respect to varying IoU and the underlying parameters of 3D bounding box location, extent, and orientation.
- An analysis of the effect of distance, number of LiDAR points, occlusion, and LiDAR intensity on the performance of these two methods on KITTI and ECP2.5D.
- Results from domain transfer experiments between KITTI and ECP2.5D.
- An analysis of the detection score that is given by the network to each ground truth object.
- A method that increases the performance of a SoA two-stage detector by adding the detections of the previous frame to the 3D proposals.

3

Methodology

The goal of 3D person detection in the KITTI benchmark is to detect the 3D bounding boxes of Vulnerable Road Users (VRUs) in the scene. The bounding boxes have seven degrees of freedom (see fig. 3.1). The 3D position is given in a coordinate system with respect to the ego-vehicle, where x is the position of the bounding box center lateral to the vehicle, z is the position longitudinal to the vehicle (*i.e.*, depth), and y determines the altitude of the bounding box center. The bounding box dimensions are specified by a width w, length l, and height h. Furthermore, each bounding box has a yaw rotation θ. The top and bottom face of the bounding box are assumed to be parallel to the $y = 0$ plane. The predicted bounding boxes will also have a detection score d related to them.

3.1. Intersection over Union (IoU)

To evaluate the performance of an object detector, each predicted bounding box is counted either as a true positive or false positive. In 3D (as well as 2D) object detection, the method to assess if a proposed bounding box is a true- or false-positive is based on IoU.

IoU is defined as the intersection (or overlap) of a 3D bounding box prediction (B_p) and ground truth (B_{gt}) divided by the union of the prediction and ground truth. When both bounding boxes only have a yaw rotation, this can be written as [77]:

$$\text{IoU} = \frac{B_p \cap B_{gt}}{B_p \cup B_{gt}} = \frac{A_o \times h_o}{V_{gt} + V_p - A_o \times h_o} \tag{3.1}$$

Where V_p and V_{gt} are the volumes of the predicted and ground truth bounding box. The overlap of volumes can be computed from the overlapping top-view area A_o and the overlapping height (h_o), see

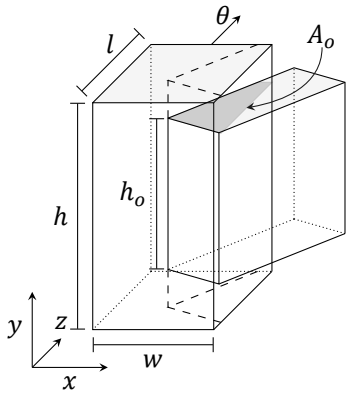


Figure 3.1: A visualization of the parameters relevant for computing the Intersection over Union (IoU) of a ground truth and predicted bounding box. The darker shaded area indicates the overlap area A_o. In this figure, the overlapping height h_o is equal to the height of the smaller bounding box.

fig. 3.1. In KITTI, a predicted bounding box is seen as a true positive if it has an IoU of more than 0.5. Only one predicted bounding box can be marked as a true positive for any ground truth bounding box.

3.2. Performance metrics

After the true positives have been determined, it is possible to compute the two metrics as defined in the KITTI benchmark for 3D object detection: 3D Average Precision (AP_{3D}) and Average Orientation Similarity (AOS) [16].

The AP_{3D} averages the maximum attained precision s with at least a recall r for a fixed range of recall values [56]:

$$\mathrm{AP}_{3D} = \frac{1}{40} \sum_{r \in \{\frac{1}{40}, \frac{2}{40}, \ldots, 1\}} \max_{\tilde{r}: \tilde{r} \geq r} s(\tilde{r}) \tag{3.2}$$

Recall is defined as the number of true positives divided by the number of ground truth annotations (*i.e.*, true positives and false negatives). Precision is defined as the number of true positives divided by the number of detections (*i.e.*, true positives and false positives). As precision and recall both depend on the number of true positives, the AP_{3D} strongly depends on the IoU threshold.

Where the AP_{3D} verifies whether the bounding boxes are in the correct place, the AOS additionally verifies the correctness of their orientations:

$$AOS = \frac{1}{40} \sum_{r \in \{\frac{1}{40}, \frac{2}{40}, \ldots, 1\}} \max_{\tilde{r}: \tilde{r} \geq r} \tilde{s}(\tilde{r}) \tag{3.3}$$

$$\tilde{s}(r) = \frac{1}{|\mathcal{D}(r)|} \sum_{i \in \mathcal{D}(r)} \frac{1 + \cos \Delta_{\theta}^{(i)}}{2} \delta_i \tag{3.4}$$

Where $\mathcal{D}(r)$ denotes the set of all objects at a specific recall rate r and $\Delta_{\theta}^{(i)}$ the difference between the estimated and the real orientation. The indicator δ_i is one if the predicted bounding box is seen as a true positive and zero otherwise. If each true positive predicted bounding box has an orientation error of 0, eq. (3.4) reduces to the precision at that recall rate.

3.3. Adaption to Part-A^2 net

To evaluate how confident a detection head is about ground truth pedestrians, Part-A^2 net was adapted. Part-A^2 net is generally discussed in section 2.6. For the adaptation of Part-A^2 net the codebase from PCDet[1] is used as base code. The focus will lay on the second part of the network, the part-aggregation stage where the 3D proposals are regressed.

The ground truth annotations are appended to the 3D proposals. Figure 3.2 shows the 3D proposals resulting from the Region Proposal Network (RPN) Head and the append 3D proposals in blue. The 3D proposals are in Light Detection And Ranging (LiDAR) coordinate frame and defined by their three coordinates (x, y, z), their size (w, l, h), and orientation (r). The added 3D proposals are processed identical to the proposals from the RPN to generate a final detection and its corresponding detection score. Since the 3D proposals are added during evaluation, there is no additional training of the network needed. The detection scores can be seen as how confident a network is about its prediction.

Because Part-A^2 net limits the range from -40 to 40, -3 to 3, and 0 to 70.4 in respectively the x-, y- and z-direction, the detection score of two pedestrians in the KITTI 3D object detection dataset can not be investigated because these pedestrians are outside this range.

After the investigation regarding the replacement of the 3D proposals from the RPN with the 3D ground truth boxes, the possible performance increase caused by adding the detections of the previous frame will be examined, as questioned in research question 2. During the examination, the 3D bounding boxes of the previous frame(s) are appended to the list of proposals coming from the RPN instead of the 3D ground truth. Since the detections of the previous frame are appended to the list of Regions Of Interest (ROIs), it could increase performance if the object is detected in the previous frame, but not

[1] https://github.com/sshaoshuai/PCDet

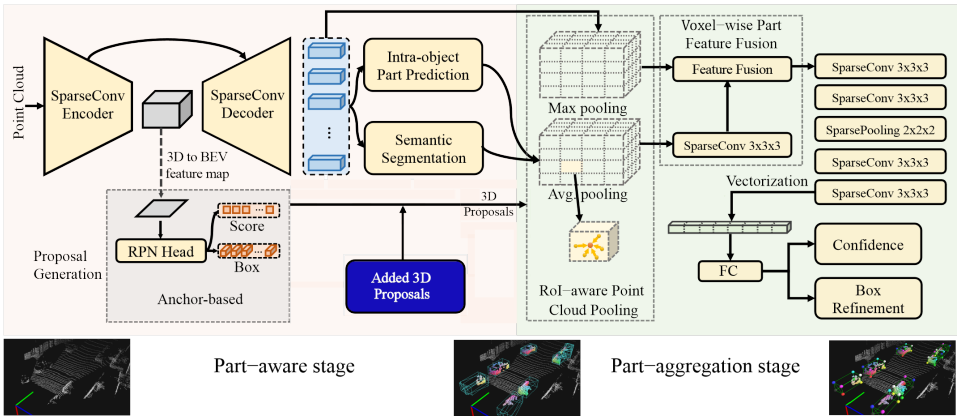


Figure 3.2: The adaption of Part-A^2 net, one can observe the addition of the 3D proposals to the outputted 3D proposals from the RPN. Image adapted from Part-A^2 net [52].

selected as a ROI in the current frame. By adding the detections of the previous frame to the list of 3D proposals coming from the RPN, the recall could potentially be improved.

During the sequences, the ego-vehicle is driving through a scene. When adding the detections of the previous frame to the list of proposals coming from the RPN, it can be important (especially at higher speeds) to compensate for this motion. The compensation for the motion of the ego-vehicle is called Ego-Motion Compensation (EMC). When applying EMC, the location (x, y, and z coordinate) of the detections are corrected using the known ego-motion transformation of the vehicle. An example of the applied ego-motion transformation is shown in eq. (3.5), x, y, and z denote the coordinates of a detection form the previous frame. The transformed coordinates are denoted as x', y', and z'. R denotes the rotation matrix and T the translation matrix.

.

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = R \begin{bmatrix} x \\ y \\ z \end{bmatrix} + T \tag{3.5}$$

3.4. KITTI RAW 3D sequence dataset

In order to evaluate the performance of Part-A^2 net on sequences, the KITTI RAW [17] is converted to a dataset containing camera, LiDAR, pose and 3D label information (if available).

During the experiments, when appending the detections of the previous frame to the ROIs of the current frame, there will be evaluated on the KITTI 3D object detection dataset, as will be further elaborated in section 3.4.1.

3.4.1. KITTI RAW dataset

The original KITTI RAW dataset [17] contains raw sequences of data recordings as can be downloaded on the KITTI vision benchmark suite website. The KITTI RAW dataset contains raw and processed images in both color and grayscale. The processed images have been rectified, distortions are removed, and the data frame numbers are corresponding to all other sensor streams. The dataset also contains 3D point clouds, 3D GPS/IMU data, calibration files, and 3D object labels. Only for a part of the data, labels are available. Opposed to the KITTI 3D object detection benchmark, the KITTI RAW dataset does not contain labels per frame. The KITTI RAW dataset contains labels per tracked object in the sequence. The labels of each tracked object contain information about the object class, 3D bounding box sizes, the 3D location and rotation, occlusion, and truncation level. The object class and 3D bounding box sizes (height, width, and length) are the same in all frames in which the object is annotated. As a consequence of tracking specific objects (*e.g.* pedestrians) in the sequence, not all objects in each frame are labeled. Another reason for the absence of 3D labels is that not all sequences are annotated and only the 3D labels are provided that passed their quality control.

3.4.2. Relation KITTI RAW and KITTI 3D dataset

To evaluate on the KITTI 3D object detection dataset, there is a need to determine which 3D object detection frame corresponds to a frame in The KITTI RAW 3D sequence dataset. The relationship between both datasets are summarized by eqs. (3.6) and (3.7), where $KITTI3D$ and $KITTIRAW$ respectively refer to the KITTI 3D object detection dataset and the KITTI RAW 3D sequence dataset. The i refers to the index of the frame in the unordered $KITTI3D$ dataset and t to the index of the corresponding frame in the ordered $KITTIRAW$ dataset. The index $t-1$ refers to the previous frame in the sequence of the $KITTIRAW$ dataset.

To find the corresponding mapping between the files, the images of both datasets are hashed and compared to each other which solves eq. (3.6). Now the corresponding file is known, the previous frame can be looked-up which solves eq. (3.7). The detections of the previous frame can then be added to the list of 3D proposals.

$$f^{i}_{KITTI3D} \rightarrow f^{t}_{KITTIRAW} \tag{3.6}$$

$$f^{t}_{KITTIRAW} \rightarrow f^{t-1}_{KITTIRAW} \tag{3.7}$$

3.4.3. KITTI RAW 3D sequence dataset

This thesis presents a conversion tool that can create two datasets. This tool converts the KITTI RAW dataset into the 3D object detection format, with and without labels. If label information is available, existing data is used to generate a tracking ID that is unique for each labeled object. The pose of the ego-vehicle is determined in each frame. Here the pose transformation to the start, the pose transformation to the previous frame, and the current pose are stored. The pose information can be used for so-called Ego-Motion Compensation (EMC), where one can use the position of the car to rotate and translate 3D points from one frame to another. During this thesis, EMC is used to compensate for the motion of the car between the current and the previous frame, when adding a 3D proposal of a previous frame to the list of 3D proposals from the RPN.

One can create two types of datasets using the code released with this thesis. The first dataset contains all the frames, this includes frames without labels. To this dataset will be referred to as the KITTI RAW 3D sequence dataset, containing 47885 frames. The second converts the KITTI RAW dataset into a dataset containing only sequences with labels. this dataset contains 10852 frames with labels. The KITTI 3D Object Detection Evaluation 2017 is a subset of KITTI RAW in terms of camera, LiDAR, and calibration data. The difference between these datasets is in the provided labels as explained in section 3.4.1. Therefore, the KITTI RAW dataset can not replace the KITTI 3D object detection dataset since not all objects are labeled in the KITTI RAW dataset. The dataset containing only sequences with labels is not used in this thesis but only mentioned for completeness.

The code for the conversion from KITTI RAW to either of these two datasets can be found at GitHub at joramvdsluis/KITTIRAW2KITTI3DSEQ.

4

Experiments

Experiments were performed with the codebase of the authors of AVOD [25] [1] and the codebase recommended by the authors of PointPillars [28] [2] as is, using the best performing network configuration as reported in their papers. Thus for AVOD, AVOD-FPN is used, and for PointPillars, a spatial resolution of 0.16×0.16 m^2 is used. For the following experiments, each network is trained 10 times. The reported results in sections 4.2 to 4.3 will be from the network closest to the average AP_{3D} of all 10 networks. Section 4.5 will analyze the variation between the 10 trained networks in more detail.

The experiments in section 4.7 and section 4.8 and are performed with the adapted version of Part-A^2 net as described in section 3.3.

4.1. Datasets overview

Each dataset contains data from two sensors: RGB images from a front-facing camera and point cloud data that includes the intensity of the reflected beam. Both datasets use the Velodyne HDL-64E (Light Detection And Ranging (LiDAR)) sensor. The ground truth contains annotations of pedestrians and cyclists. KITTI also provides annotations for vehicles, but these are ignored in this study.

Figure 4.1 shows the distribution of the Vulnerable Road Users (VRUs) locations relative to the vehicle, for the publicly available part of both KITTI and ECP2.5D. The bulk of the detections in the KITTI dataset lies within 30 m distance of the ego-vehicle. Both datasets have a bias towards VRUs being on the right side of the ego-vehicle. Figure 4.2 shows the relation between LiDAR points reflected off a VRU and distance for both datasets. The number of LiDAR points is found by counting all LiDAR points within the annotated 3D bounding box. The number of points scales inversely with the distance on average. The number of LiDAR points has a large spread for any given distance however, caused by partial occlusions as well as the orientation of the VRU. As ECP2.5D has more VRUs at a distance, the dataset contains relatively more annotations with few LiDAR points.

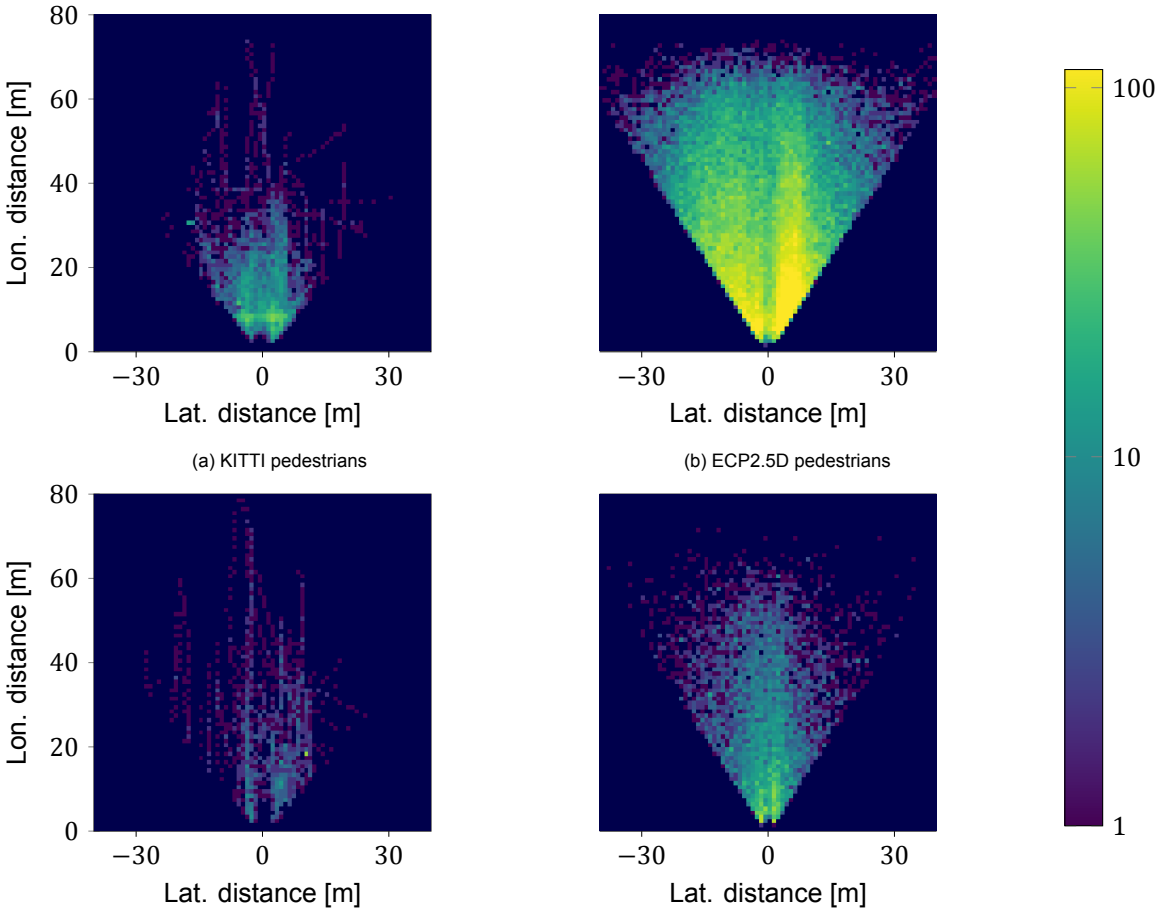
The KITTI dataset is not divided into an official train and validation split. However, both the AVOD and PointPillar codebase specify an identical train/validation split. The data is divided equally over the train and validation splits. Each split is also divided into an “easy”, “moderate” and “hard” part, as defined by KITTI (see table 4.1). These parts are subsets of each other, where the easy part is a subset of the moderate part and the moderate is a subset of the hard part.

The ECP2.5D dataset has a larger number of annotations for the 3D position and orientation but lacks width, length, and height annotation. Therefore, the median bounding box dimensions of the train split of the KITTI dataset will be used so both networks can still regress a full bounding box. The data is split into a “Day” and “Night” subset. If not specifically mentioned, this paper uses the “Day” subset. Additionally, some of the underlying EuroCity Persons (ECP) dataset annotations misses an orientation label. These are set to “Don’t Care”. The exact numbers can be found in table 4.2.

The test set ground truth annotations of both datasets are not made public, so all evaluations done in the following sections of this paper are done using the validation splits of either dataset. For KITTI, this means using the unofficial validation split described in this section.

[1] https://github.com/kujason/avod

[2] https://github.com/traveller59/second.pytorch



(a) KITTI pedestrians (b) ECP2.5D pedestrians

(c) KITTI cyclists (d) ECP2.5D cyclists

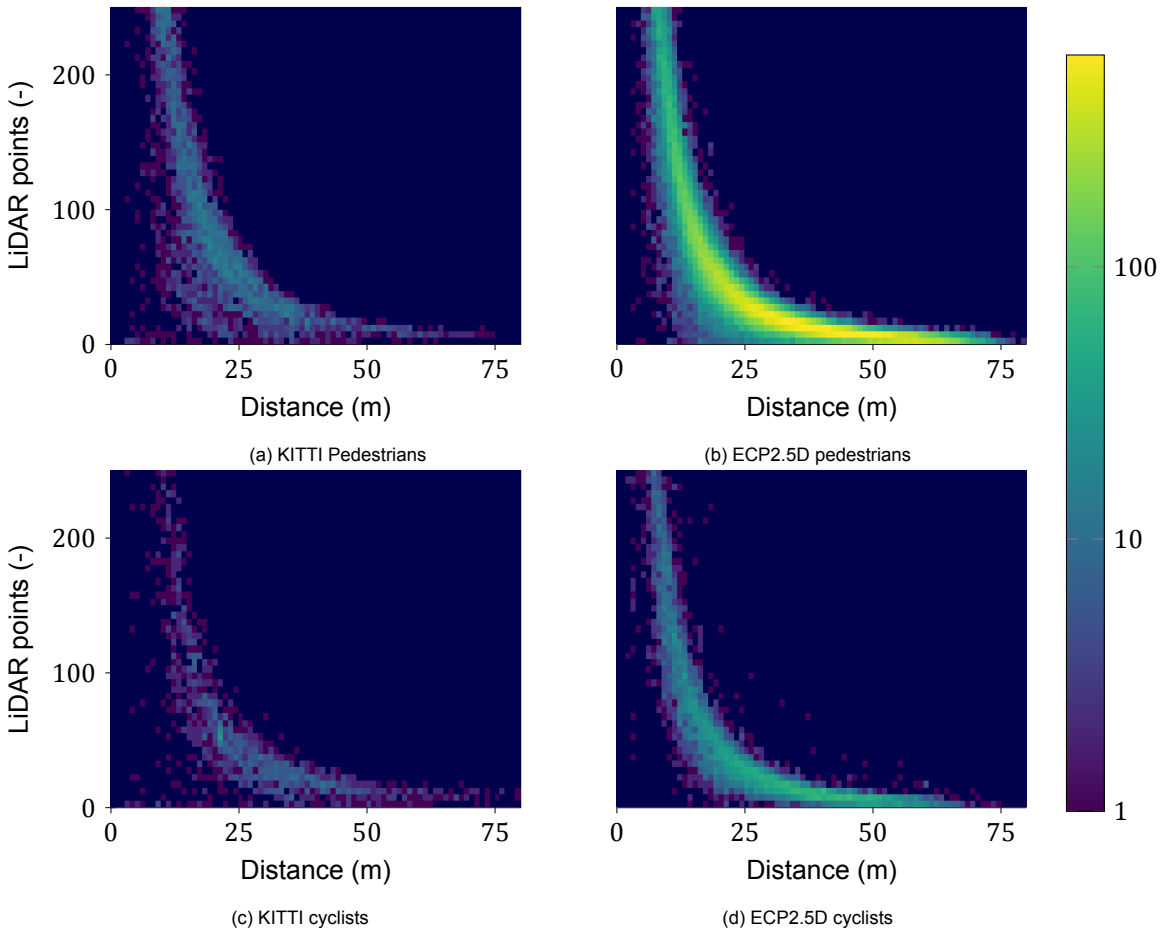
Figure 4.1: The overall distribution over location of pedestrians and cyclists in both the KITTI and EuroCity Persons 2.5D (ECP2.5D) dataset as a log plot. In all figures, the egovehicle is positioned at $(0, 0)$, looking upwards. Each pixel in the image corresponds to a 1×1 square meter area. The darkest blue region indicates areas with zero pedestrians.

Table 4.1: Difficulty levels as defined by KITTI [16]. Truncation refers to the percentage of the bounding box that is outside the image.

Difficulty	Min. bounding box height	Max. occlusion level	Max. truncation
Easy	40 Pixels	Fully visible	15 %
Moderate	25 Pixels	Partly occluded	30 %
Hard	25 Pixels	Difficult to see	50 %

Table 4.2: The number of pedestrians and cyclists the splits of each dataset. Values indicate the number used in this paper/number without orientation.

Dataset	Pedestrians	Cyclists
Train split :		
ECP2.5D Day	62.3K/318	7.3K/111
ECP2.5D-Night	13.4K/14	0.7K/1
KITTI	2.2K/0	0.7K/0
Validation split :		
ECP2.5D Day	12.6K/68	1.3K/33
ECP2.5D-Night	2.6K/2	0.1K/1
KITTI	2.3K/0	0.9K/0



(a) KITTI Pedestrians (b) ECP2.5D pedestrians

(c) KITTI cyclists (d) ECP2.5D cyclists

Figure 4.2: The number of VRUs at a given distance and with a given number of LiDAR points reflecting off them, for both the KITTI and ECP2.5D dataset as a log plot. The darkest blue region indicates areas with zero pedestrians/cyclists.

Table 4.3: AOS and AP_{3D} performance of PointPillars (PP) and AVOD, trained on KITTI and evaluated on the moderate part of the KITTI validation split.

	Pedestrian		Cyclist	
IoU	AP_{3D}	AOS	AP_{3D}	AOS
PP				
0.5	51.3	23.0	62.4	3.1
0.4	70.4	31.5	65.7	3.9
0.3	74.3	33.1	66.8	4.0
0.2	75.0	33.1	67.4	4.0
0.1	75.2	33.4	67.4	4.0
AVOD				
0.5	36.4	28.3	27.3	27.2
0.4	44.4	34	29.6	29.4
0.3	46.8	35.7	29.6	29.5
0.2	47.0	35.9	29.8	29.6
0.1	47.1	36.0	29.9	29.7

4.2. Effect of IoU on performance and error analysis

Performance with lower Intersection over Union (IoU) constraints

Table 4.3 shows the performance of PointPillars and AVOD on KITTI for the cyclist and the pedestrian classes. As mentioned, this shows the performance of the network closest to the average average AP_{3D} of the 10 trained networks of each type. PointPillars has a higher AP_{3D} than AVOD, even though their scores on the moderate test split on the online KITTI benchmark differ less than one percent. However, the results found for AVOD are comparable to those found on the validation split in the comparison study of Roth *et al*. [50]. For both networks, lowering the IoU threshold increases the AP_{3D} by a large margin. For example, the AP_{3D} of PointPillars on pedestrians increases from 51.3 % to 75.2 %.

The increase in AP_{3D} caused by the lowering the IoU is further visualized in fig. 4.3, which shows a histogram of the IoU found for all true positive detections at an IoU threshold of 0.1. The histogram shows that for pedestrians more than 20 % of the detections of PointPillars and 11 % of the detections of AVOD had an IoU between 0.4 and 0.5, just outside the normal IoU threshold. A similar effect is seen for cyclists, albeit less strongly.

The upper bound of the Average Orientation Similarity (AOS) is the AP_{3D}, as mentioned in section 3.2. Table 4.3 shows that even though the general detection accuracy of AVOD is lower than PointPillars, its AOS is very close to that upper bound, especially for cyclists. The AOS of PointPillars is worse than the AOS noted on the online KITTI benchmark. A closer inspection of the distribution of the orientation error (fig. 4.4) shows that for PointPillars, the orientation error peaks around 0 or 180 degrees. In the paper of PointPillars [28], the authors state that the orientation loss used cannot distinguish between flipped boxes, for which they use an additional binary classification loss. This seems to indicate that while the original overall orientation loss works as expected, there might be an implementation issue with the binary classification loss in the codebase of SECOND. As for AVOD, most of the orientation estimates indeed have an error close to 0 degrees as was expected from their AOS.

Error analysis of bounding box estimation

Figure 4.5 shows the error made in the position and size of the predicted bounding boxes on pedestrians by PointPillars. The smallest errors are made on the x and the z estimation: the lateral and longitudinal position. The largest error is made on the width and length estimation. These estimations have the largest increase in error at lower IoU thresholds. These depend on the stride of a pedestrian, as well as the location of their arms, which can be difficult to infer.

The relatively small error in x and the z position (essentially a top-down position estimate) is visualized in fig. 4.6. It shows the x and z position error made for the true positive detections for the original

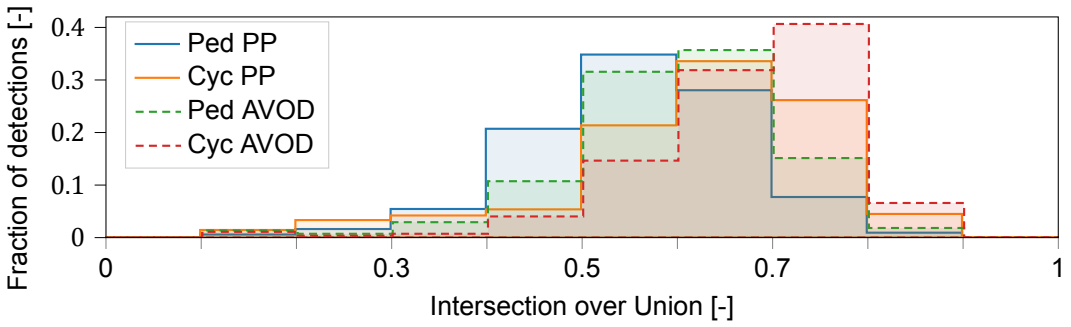


Figure 4.3: PointPillars and AVOD trained on KITTI: a histogram of what fraction of true positive detections had what IoU (IoU threshold of 0.1).

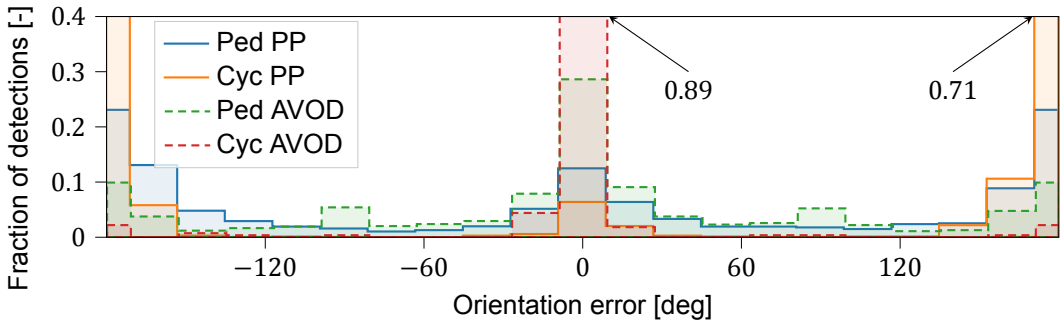


Figure 4.4: PointPillars and AVOD trained on KITTI: A histogram of the orientation error. The arrows indicate the fraction of detections of the two bars outside of the y axis range. Most orientation errors lie either between -40 and 40 degrees, or between 140 and -140 degrees.

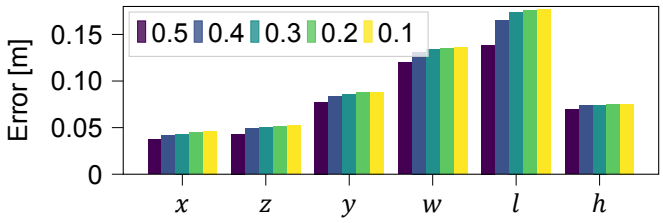


Figure 4.5: PointPillars trained on KITTI: the average error between the prediction and the ground truth for the pedestrian detections on x, z, y, w, l and h, at different IoUs thresholds. The largest error is made on the altitude estimation, together with the bounding box width and length.

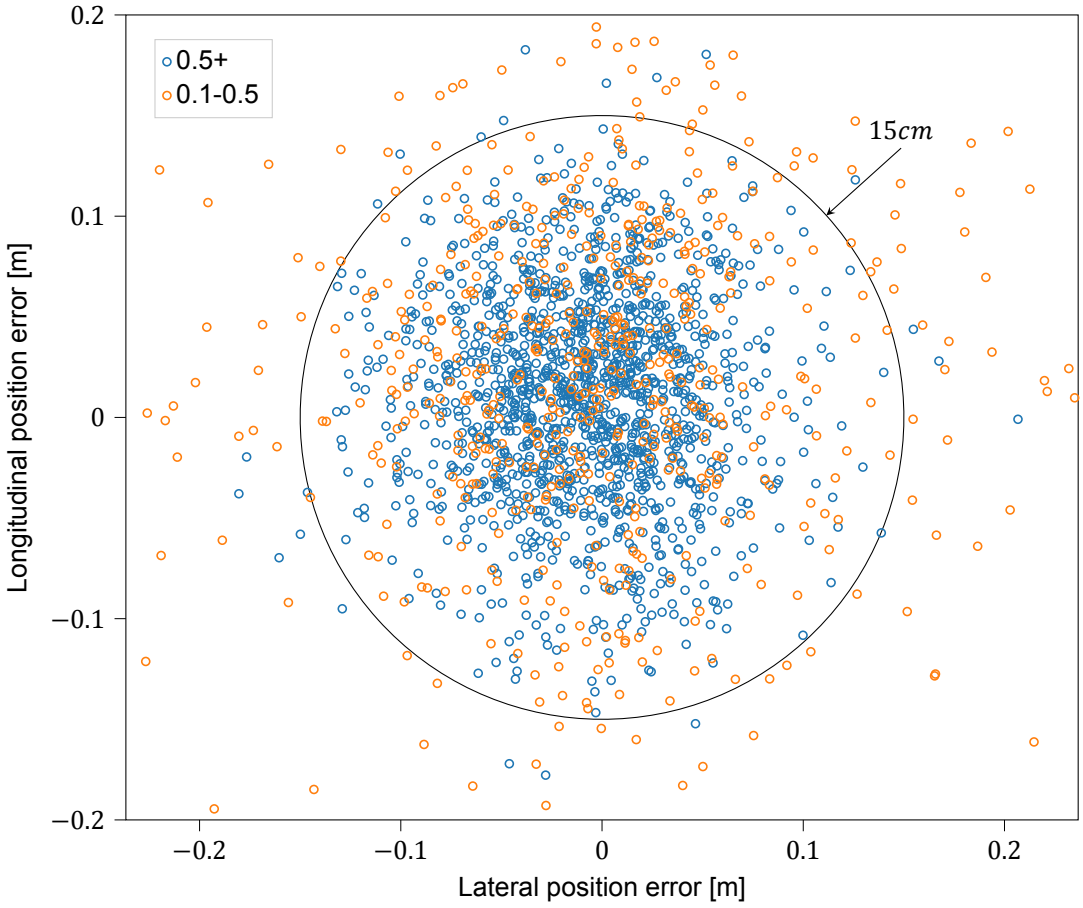


Figure 4.6: PointPillars trained on KITTI: The localization error made by true positive detections of **pedestrians**, from a bird's eye viewpoint. Of the true positive detections with an IoU of over 0.5, 1462 out of 1494 detections lie within a radius of 15 cm. Of the true positive detections with an IoU between 0.1 and 0.5, 349 out of 478 lie within that radius.

IoU threshold, as well as the detections between an IoU of 0.1 and 0.5. A lot of the detections with an IoU below 0.5 are still accurate at estimating the position. For an IoU threshold of 0.5, nearly all of the true positive detections (1423 out of 1445) lie within a radius of 15 cm. When looking at the detections found with an IoU threshold of 0.1, a total of 1850 detections lie within a radius of 15 cm. In other words, using a radius of 15 cm as a metric to determine true positives instead of an IoU of at least 0.5 shows an increase in detections of 23 %. The same data is put more succinctly in fig. 4.7, with cyclists added as well. It shows the number of true positive detections that fall below a specific Euclidean position error. Cyclists see a smaller benefit: because their annotated bounding boxes are usually larger, it is possible to make a larger position error without affecting the IoU as much.

Accuracy evaluation using fixed bounding boxes during training

The relatively large errors in width and length suggest that AVOD and PointPillars are not able to properly estimate bounding box dimensions. To investigate the influence of the bounding boxes dimensions, a network is trained on the version of the KITTI dataset train split where the dimensions of each VRU are fixed. The dimensions are fixed to the median dimensions of their respective class. Table 4.4 shows the AP_{3D} of those detectors evaluated on the original KITTI dataset validation split with the correct dimensions. In all cases except the PointPillars pedestrian case, the performance increases when the bounding box dimensions are disregarded during training.

4.3. Effect of distance and LiDAR points on performance

Focusing on PointPillars, figs. 4.8a to 4.8c show the precision and recall based on the distance for KITTI, ECP2.5D, and ECP2.5D-Night for an overall precision of 0.5. On both datasets, the performance

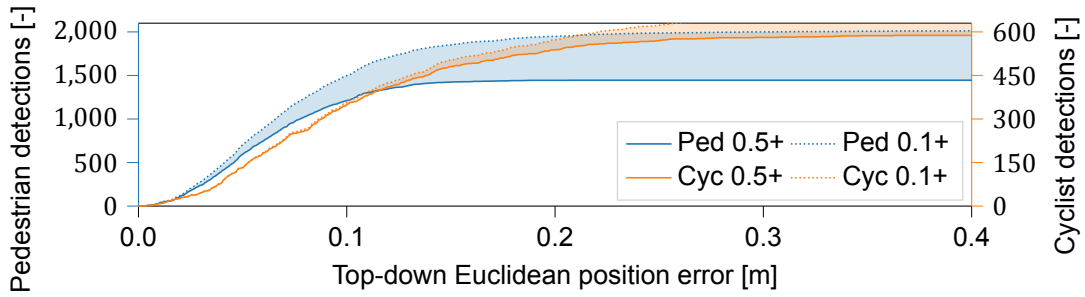


Figure 4.7: PointPillars trained on KITTI: showing how many detections would be inside a certain Euclidean position error threshold. The solid line shows what Euclidean error is made by detections using the current default IoU threshold of 0.5. The dotted line shows the number of detections within a given Euclidean error for an IoU threshold of 0.1. The shaded area then shows the number of detections added.

Table 4.4: AP_{3D} performance of PointPillars (PP) and AVOD for two IoU thresholds, evaluated on the moderate part of the KITTI validation split. The networks were trained on the original KITTI ground truth (Orig.) or the ground truth with fixed bounding box dimensions (Fixed).

	Pedestrian		Cyclist	
IoU	Orig.	Fixed	Orig.	Fixed
PP				
0.5	51.3	54.6	62.4	62.6
0.1	75.2	73.3	67.4	68.1
AVOD				
0.5	36.4	46.0	27.3	35.5
0.1	47.1	59.6	29.9	38.8

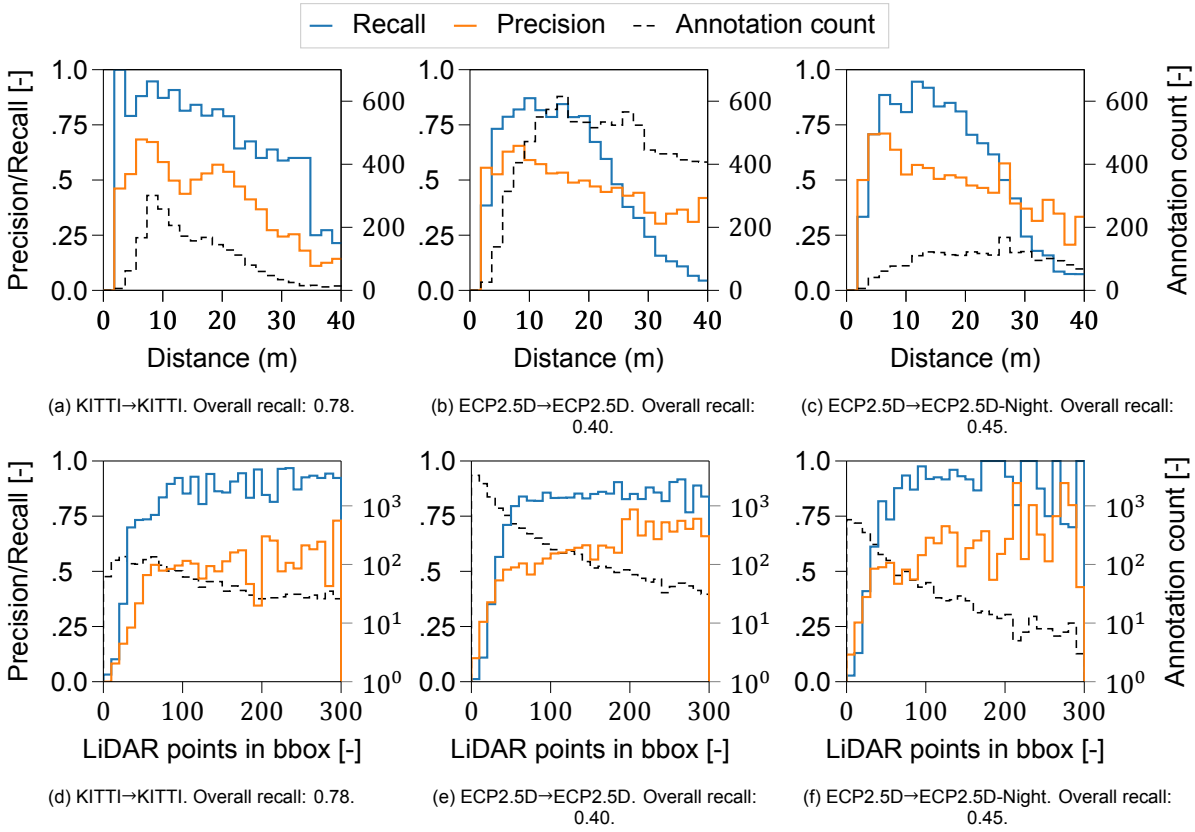
starts to degrade around 20 m distance. However, because ECP2.5D has a larger proportion of distant targets (see dashed line in fig. 4.1) the performance reduction on distant targets affects the total AP_{3D} more.

Figures 4.8d to 4.8f show the precision and recall as a function of LiDAR points for KITTI and ECP2.5D, at the same overall precision of 0.5 as figs. 4.8a to 4.8c. On all datasets, both recall and precision decrease as the number LiDAR points reflected by the target goes down; targets with fewer LiDAR points are more challenging to detect. Both ECP2.5D and ECP2.5D-Night have a larger proportion of pedestrians with a small number of points inside their bounding box compared to KITTI, ECP2.5D more so than ECP2.5D-Night. The network trained on ECP2.5D shows a similar overall trend on ECP2.5D and ECP2.5D-Night, except that the precision on ECP2.5D-Night starts to fluctuate on targets with more than 200 points in the detected bounding box. This is most likely due to the low sample count, as there are less than 10 ground truth annotations in this area, and even fewer true positives.

4.4. Effect of occlusions on performance

Figures 4.9a to 4.9c show the recall over distance separately for occluded and non-occluded pedestrians. Figure 4.9 does not contain any precision, as precision is a function of false positives and it is ambiguous whether a false positive should belong to the occluded or non-occluded group. As expected, the performance is higher on non-occluded pedestrians. The number of occluded and non-occluded pedestrians in KITTI (fig. 4.9a) is almost equal after 12 meters. ECP2.5D (fig. 4.9b) and ECP2.5D-Night (fig. 4.9c), have a higher fraction of non-occluded pedestrians over the full distance range.

The recall of occluded and non-occluded pedestrians as a function of the amount of LiDAR points in the ground truth bounding box is shown in figs. 4.9d to 4.9f. In these figures, the gap between occluded and non-occluded recall is smaller than when comparing performance over distance (figs. 4.9a to 4.9c). At a high number of LiDAR points in the ground truth bounding box, the occluded recall starts to deviate



(a) KITTI→KITTI. Overall recall: 0.78.

(b) ECP2.5D→ECP2.5D. Overall recall: 0.40.

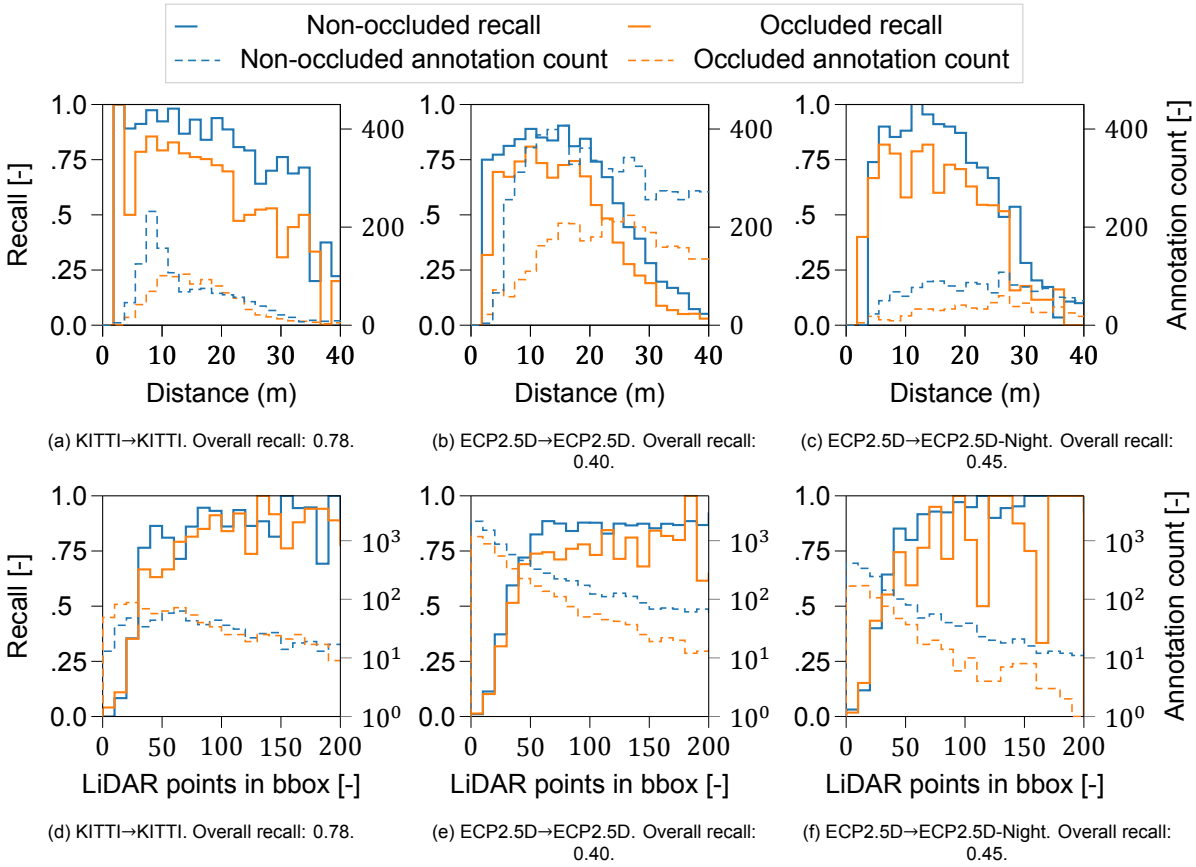
(c) ECP2.5D→ECP2.5D-Night. Overall recall: 0.45.

(d) KITTI→KITTI. Overall recall: 0.78.

(e) ECP2.5D→ECP2.5D. Overall recall: 0.40.

(f) ECP2.5D→ECP2.5D-Night. Overall recall: 0.45.

Figure 4.8: PointPillars: the precision and recall histogram, grouped by distance (figs. 4.8a to 4.8c) and by the number of LiDAR points on bounding box (figs. 4.8d to 4.8f). For recall, which depends on true positives, the ground truth bbox is used to compute the distance/LiDAR point count. For precision, which depends on false positives, the detected bbox is used to compute the distance/LiDAR point count. The subcaption shows on which dataset the network is trained→evaluated. The operating point of each network is selected at an overall precision of 0.5. True positives are selected using an IoU of 0.1. The dashed line indicates the total number of ground truth annotations for each bin on a linear scale for the distance, and on a logarithmic scale for LiDAR points.



(a) KITTI→KITTI. Overall recall: 0.78.

(b) ECP2.5D→ECP2.5D. Overall recall: 0.40.

(c) ECP2.5D→ECP2.5D-Night. Overall recall: 0.45.

(d) KITTI→KITTI. Overall recall: 0.78.

(e) ECP2.5D→ECP2.5D. Overall recall: 0.40.

(f) ECP2.5D→ECP2.5D-Night. Overall recall: 0.45.

Figure 4.9: PointPillars: the recall histogram for occluded and unoccluded pedestrians, grouped by distance (figs. 4.9a to 4.9c) and by the number of LiDAR points on bounding box (figs. 4.9d to 4.9f). For recall, which depends on true positives, the ground truth bbox is used to compute the distance of/LiDAR points count. Precision is not shown, as it is ambiguous whether false positives belong to the occluded or non-occluded category. The subcaption shows on which dataset the network is trained→evaluated. The operating point of each network is selected at an overall precision of 0.5. True positives are selected using an IoU of 0.1. The dashed line indicates the total number of ground truth annotations for each bin on a linear scale for the distance, and on a logarithmic scale for LiDAR points.

Table 4.5: AP_{3D} performance of PointPillars (PP) and AVOD for an IoU of 0.1 on the moderate validation split of KITTI, ECP2.5D and ECP2.5D-Night. Bold indicates highest performance in that column. The network are trained 10 times each, next to the AP_{3D}, the standard deviation is shown.

	AP_{3D}		
Trained network	KITTI	ECP2.5D	ECP2.5D-Night
$with\ intensity:$			
PP on ECP2.5D	49.4 $\pm$ 4.4	**33.0** $\pm$ **1.4**	37.4 $\pm$ 1.8
PP on KITTI	**75.2** $\pm$ **1.5**	9.4 $\pm$ 1.6	9.0 $\pm$ 1.8
$w/o\ intensity:$			
PP on ECP2.5D	54.1 $\pm$ 2.9	32.9 $\pm$ 1.4	**38.0** $\pm$ **1.4**
PP on KITTI	66.8 $\pm$ 2.3	23.4 $\pm$ 2.5	26.5 $\pm$ 2.9
AVOD on ECP2.5D	34.8 $\pm$ 8.3	27.5 $\pm$ 5.3	23.2 $\pm$ 7.6
AVOD on KITTI	47.6 $\pm$ 5.9	5.3 $\pm$ 2.2	2.8 $\pm$ 1.7

from the non-occluded recall. However, this part of the graph has a small number of samples, less than 10, making it difficult whether this is due to the training, the network, or a statistical anomaly. Regardless, for all datasets, it seems that a better distinction for which pedestrians are difficult to detect is that amount of LiDAR points in the ground truth bounding box, rather than whether a pedestrian is occluded or not.

4.5. Cross-dataset evaluations

To see how well each dataset generalizes, ten network are trained on KITTI and ECP, and evaluate on all three datasets. Because the original PointPillars uses the intensity information of the points in the point cloud, the experiments are performed once *with* this intensity information present, and once *without*. AVOD does not use the intensity information, and therefore only needs to be trained once on each dataset. To ensure the datasets are compatible, the LiDAR intensity KITTI values are rescaled linearly from $[0, 1]$ to $[255, 0]$ and vice versa. Linear scaling is also performed to match the modes of the intensity distributions, but this resulted in a significantly worse performance. For example, one of the networks trained on ECP2.5D and evaluated on KITTI attains a AP_{3D} of 18.9 instead of 46.7.

Table 4.5 shows the resulting average AP_{3D} of the ten networks, together with the standard deviation. For both KITTI and ECP, PointPillars using LiDAR intensity data and training on that dataset attains the best performance. When not using LiDAR intensity data, PointPillars' performance slightly drops on KITTI, but still outperforms AVOD. AVOD also has a higher variance than PointPillars. The performance of both methods is significantly lower on ECP2.5D vs. KITTI.

When training on one dataset and testing on another, performances degrade significantly for either methods, both when moving from KITTI to ECP2.5D, and vice versa. The resulting performance degradation for PointPillars is less severe when the LiDAR intensity data is not used, which can also be observed in the precision-recall curves.

Figure 4.10 shows that both cross-evaluated networks perform better if they are trained and evaluated without intensity. Networks that are trained and evaluated on the same dataset still perform better with intensity. However, as it does not outperform PointPillars with intensity, this concludes that there is relevant information in the intensity. As a final note, the highest performance on ECP2.5D-Night is higher than the highest performance on ECP. As fig. 4.8e and fig. 4.8f show, this is likely due to the relatively smaller number of pedestrians with a low number of LiDAR points inside their bounding box.

4.6. Qualitative analysis ECP2.5D

To perform a qualitative analysis of the type of errors made by PointPillars, 30 frames were randomly sampled from the validation part of the ECP2.5D dataset. These frames contained a total of 58 false positives, 35 true positives, and 55 false negatives. An example of such a frame is shown in fig. 4.11

Table 4.6 shows a categorization of different objects which are classified as a pedestrian and are not matched with a ground truth. The category poles contains traffic poles, traffic signs, trees, and

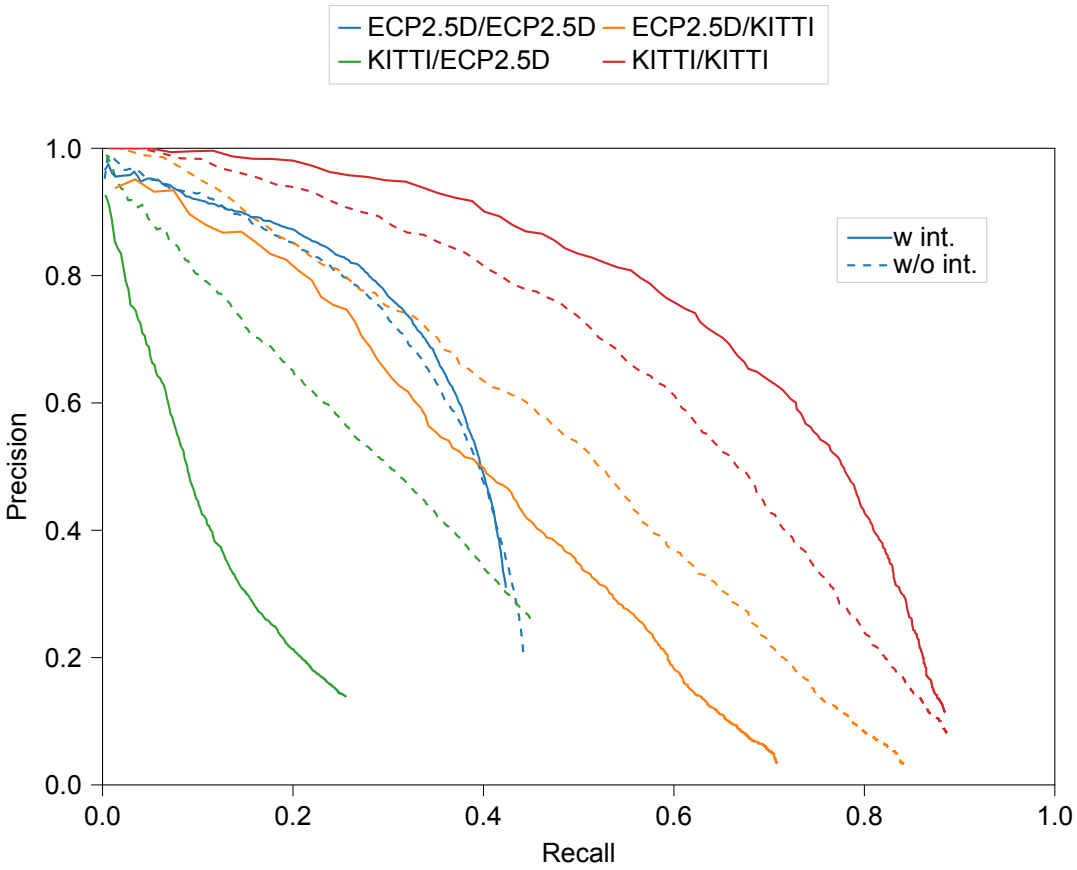


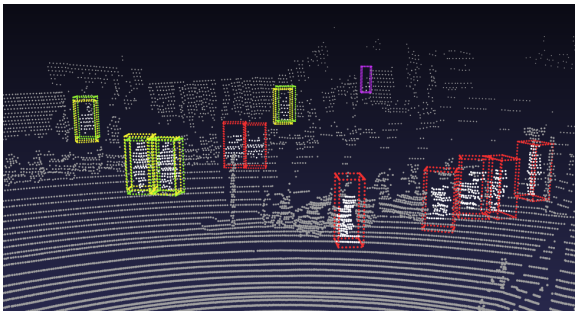
Figure 4.10: PointPillars: the precision-recall curve for PointPillars trained on ECP2.5D and KITTI at an IoU threshold of 0.1. The legend shows on which dataset each line was trained/evaluated, respectively. The solid/dashed line show the networks trained with/without intensity, respectively.

Table 4.6: Categorization of the false positives made by PointPillars in 30 frames of the validation part of the ECP2.5D dataset.

Type of error	Occurrences
Pole	21
Not annotated	15
Bicycle / scooter	2 / 9
Fence / wall / door	3 / 1 / 1
Small objects	3
Too low IoU	2
Unknown	1

Table 4.7: Categorization of the false negatives made by PointPillars in 30 frames of the validation part of the ECP2.5D dataset.

Type of error	Occurrences
Low number of LiDAR points	47
Occluded in LiDAR space	4
Too low IoU	2
Near wall	1
Unknown	1



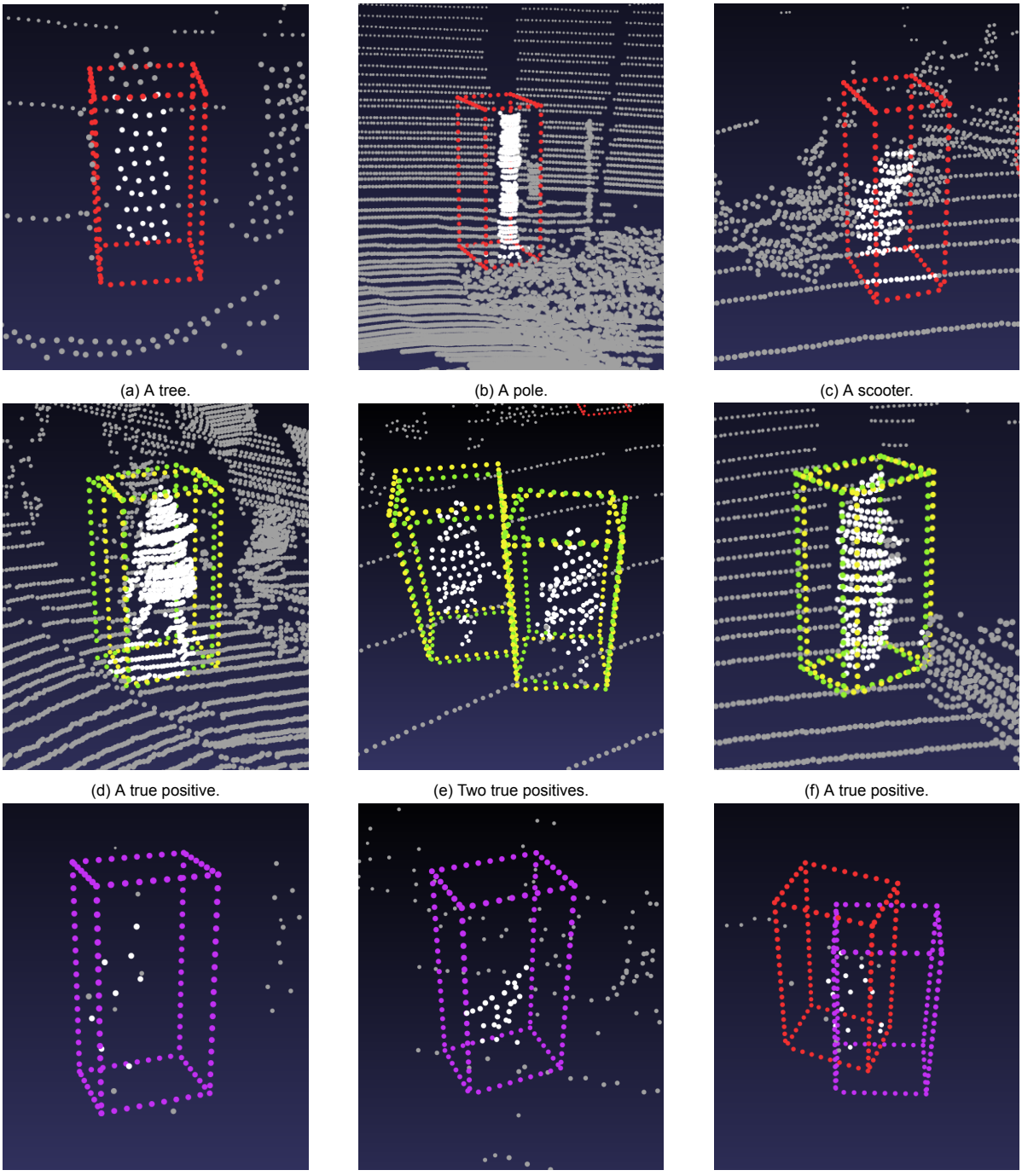
(a) An example of a point cloud including the 3D boxes representing the false positives, true positives and the false negatives.

(b) Corresponding image to point cloud.

Figure 4.11: Overview of a frame from the validation part of the ECP2.5D dataset. False positives, false negatives, the detection and ground truth of a true positive are respectively shown in red, purple, green and yellow. LiDAR points inside a 3D box are white, other points are grey.

large flag poles. The small object category contains a small advertisement board, pram, and trashcan. Not annotated refers to correctly classified objects which are not annotated because they are either occluded in the image(11) or just outside the image frame(4). There is one unknown object of which it was not possible to identify what kind of object it was based on the LiDAR point cloud or image information. There is a large number of true positives that are caused by poles, therefore one could argue that the detector is not able to capture enough of the shape information to distinct a human from a tree/pole.

Out of the 55 false negatives, 47 objects had very few LiDAR points on them. Table 4.7 shows the categorization of the reasons why the object was not detected. 31 false negatives have less than or equal to 10 LiDAR points in their 3D box and 47 false negatives have fewer or equal to 25 LiDAR points in their 3D box. The main reason for not detecting the pedestrians is the low number of LiDAR points on them. At large distances, the inclination of the road can result in pedestrians with no points on their upper body as visible in fig. 4.12h. Two false positives intersect the true negatives with a too low IoU to be a true positive. There was one object which was near a wall and therefore missed. Furthermore, Of one object it was not clear why it was not detected by the network, the reason why this objects is missed is called unknown.



(a) A tree. (b) A pole. (c) A scooter.

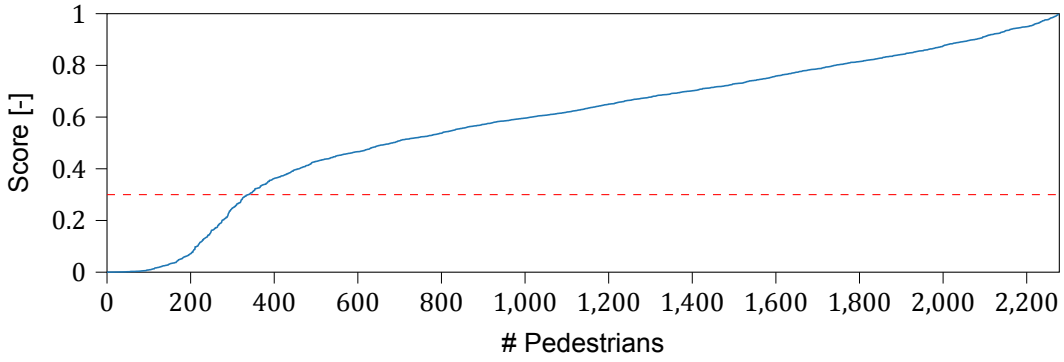
(d) A true positive. (e) Two true positives. (f) A true positive.

(g) An object with few points. (h) An object where only the bottom part is visible. (i) A false negative and a detection with a too low IoU to be a true positive.

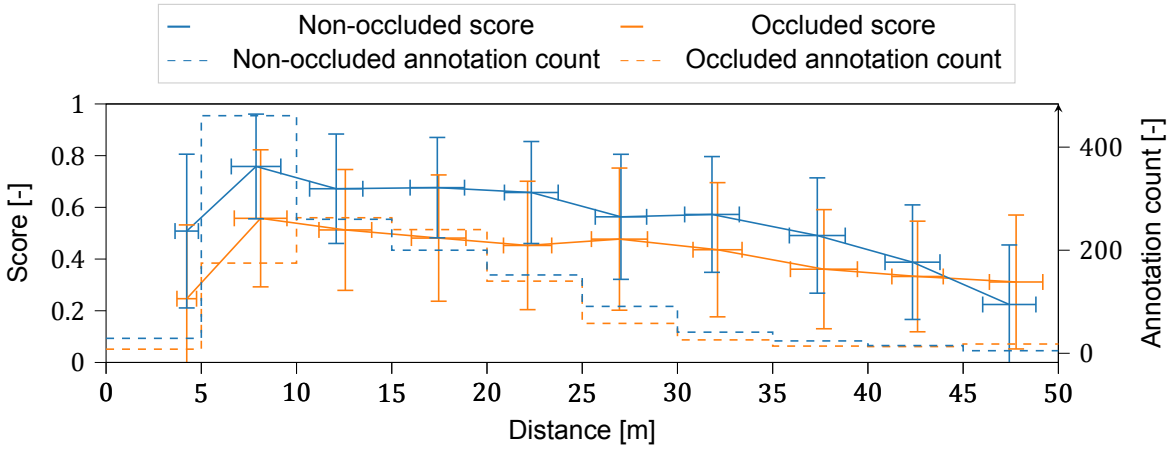
Figure 4.12: An example of false positives (a-c), true positives (d-f) and false negatives (g-h). The false positives, false negatives, the detection and ground truth of a true positive are respectively shown in red, purple, green and yellow. LiDAR points inside a 3D box are white, other points are grey.

4.7. Detection score of ground truth pedestrians

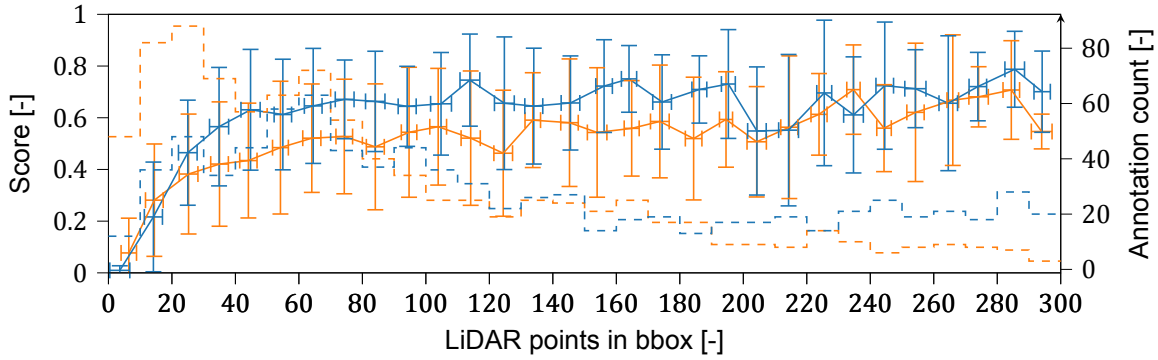
To investigate how confident a network is about ground truth pedestrians, Part-A^2 net is adapted as described in section 3.3. The two-stage network is adapted to create an input where the 3D ground truth can be added to the 3D proposals coming from the Region Proposal Network (RPN). These proposals are evaluated by the second stage of the network as described in section 3.3. Therefore, the resulting detection score is investigated to examine how confident the detection head is about ground



(a) All scores sorted from low to high. The dashed red line shows the default cut-off point of Part-A^2 net, indicating when pedestrians are discarded. 339 out of 2278 pedestrians have a score lower than 0.3.



(b) The thick line shows the score per distance, divided in bins of 5 meter. The dashed line shows the annotation count. The error bars show the standard deviation for both the score and the distance direction.



(c) The thick line shows the score per distance, divided in bins of 10 LiDAR points. The dashed line shows the annotation count. The error bars show the standard deviation for both the score and the number of LiDAR points within a 3D bounding box.

Figure 4.13: Detection scores of ground truth pedestrians as predicted by Part-A^2 net trained on KITTI.

truth pedestrians.

When a network is used in practice, a minimal detection score is set, also called a detection threshold. This is done to balance precision and recall. Due to this set threshold, some detections will be filtered out. The online implementation of Part-A^2 net comes with a default detection threshold of 0.3. This implies that objects with detection scores lower than 0.3 are discarded by the Part-A^2 net implementation. Figure 4.13a shows the ground truth pedestrians sorted by detection score. The red line indicates the cut-off detection score at which objects are disregarded. Out of the 2280 pedestrians in the KITTI validation dataset, 2 are not detected because they are outside the domain of the detector as mentioned in section 3.3, and therefore have no detection score. 339 out of 2280 have a detection score lower than 0.3. This indicates that even if the RPN would propose all the ground truth objects, 14.9% would be discarded by the regressor based on its given detection score.

To further pinpoint in which cases the network gives objects low scores, the relation between the average detection score and both the distance and number of LiDAR points on a pedestrian are shown in respectively fig. 4.13b and fig. 4.13c. Figure 4.13b shows a low detection score when close to the car which could be either caused by the low number of samples of objects between 0-5 m. From 10 meters there is a slow decay of the score. As expected, the score of occluded objects is lower. This shows that even if the RPN proposes the location of all ground truth pedestrians, the network is not confident about pedestrians close-by, increasingly less confident for objects further away than 10 meters, and is less confident about occluded pedestrians.

The relation between the average detection score and the number of LiDAR points in the 3D box of a ground truth pedestrian is shown in fig. 4.13c. For non-occluded pedestrians, there is a strong increase in detection score up to 50 LiDAR points. From 50 LiDAR points the detection score stabilizes. For occluded objects, the score keeps growing with the number of LiDAR points up to 200 LiDAR points where it shows similar trends as the non-occluded objects. This shows that the network is increasingly confident for non-occluded pedestrians up to 50 LiDAR points, after 50 LiDAR points the detection score is stable. For occluded pedestrians, the confidence increases up to 200 LiDAR points after which the performance is stable and similar to that of non-occluded objects. Although an occluded and non-occluded pedestrian have the same number of LiDAR points, the average confidence of the network will be lower for occluded pedestrians.

Figure 4.14 shows the precision-recall curve of Part-A^2 net evaluated with and without 3D GT boxes added to the list of proposals coming from the RPN. Ideally, one would like to detect all objects (recall of 100%) without having any false positive (100% precision). The ideal prevision-recall curve would be a curve which is as close to this point as possible. The difference between these evaluations shows the maximum possible performance increase of the network if the RPN would have a recall of 100%. When the GT is added to the 3D proposals coming from the RPN, the maximum recall does not go to 100% when applying non-maximum suppression (NMS). If NMS is disabled, one can observe that 100% recall is reached when the 3D GT boxes are added to the list of proposals coming from the RPN. Where the goal of NMS is to suppress overlapping 3D proposals that have a lower detection score as explained in section 2.1.5. Removing the NMS results in multiple detections per object and more detections with a low feature score, since they are not suppressed. This can be seen when the 3D GT boxes are added to the list of proposals coming from the RPN without applying NMS. Without applying NMS, there will be more 3D proposals of similar pedestrians, since those are not suppressed. Therefore, more detections will be seen as FP because there is only one detection matched with a GT, decreasing the precision. Applying NMS results in a decrease of the maximum recall from 100% to 93.7%, which is caused by the higher detection scores given by the network to objects close-by a pedestrian.

4.8. Temporal fusion

As described in section 3.3, Part-A^2 net is adapted to append 3D proposals to the list of proposals coming from the RPN. In this section, the detections of the previous frame(s) are appended to the list of proposals coming from the RPN. As shown in section 4.7, the performance of the network can be improved when the 3D GT boxes are added to the list of proposals coming from the RPN. Therefore, adding the detections of the previous frame instead of the ground truth could potentially improve performance.

Two types of detections can be added to the list of proposals coming from the RPN. The first type of

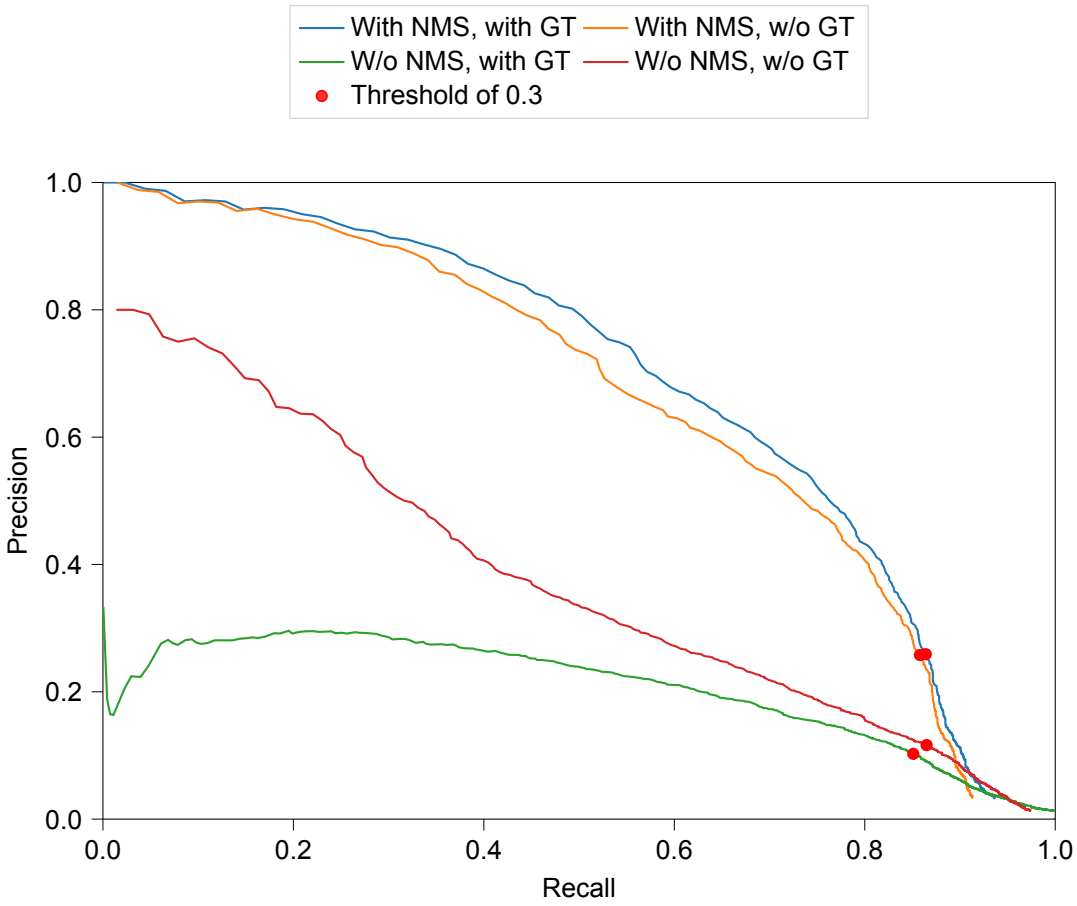


Figure 4.14: Precision-recall curves for Part-A^2 net, trained and evaluated on KITTI for an IoU of 0.1. Each of these networks is evaluated with or without non-maximum suppression (NMS), and ground truth (GT). The red dot denotes the default score threshold of Part-A^2 net.

Table 4.8: AP_{3D} for different IoU thresholds. EMC denotes if the location of the bounding boxes in the previous frame are ego-motion compensated for the motion of the car. Prev refers to the addition of the detections of the previous frame to the 3D proposals. Seq denotes that the detections from the previous frames are used, the detections of these frames are propagated through the whole sequence. Both the max recall and the recall at 50% precision are calculated with an IoU of 0.1.

	AP_{3D} for different IoUs			$Recall$	
Temporal methods	0.5	0.3	0.1	$recall_{max}$	$recall_{0.5precision}$
Original	54.43	68.64	70.09	91.36	73.55
$w/o\ EMC:$					
Prev	54.56	69.18	70.62	92.32	74.17
Seq	54.44	69.26	70.72	92.63	74.43
$with\ EMC:$					
Prev	**54.94**	**69.33**	**71.03**	92.63	**75.22**
Seq	54.03	**69.33**	70.85	**92.98**	74.96

detections are the detections from the previous frame only, to this kind of detections will be referred as 'Prev'. For the second type of detections, the detector generates detections for the whole KITTI RAW 3D sequence dataset, sequentially adding the detections of the previous frame to the list of proposals coming from the RPN. These detections, therefore, consist of propagated detections of all the previous frames up to the point where the detections are extracted. To these kind of detections is referred as 'Seq'. Both Seq and Prev are compared to investigate if the aggregation of previous detections though a whole sequence can result in higher performance compared to only using the detections of the previous frame. Additionally, the performances with and without Ego-Motion Compensation (EMC) applied to the detections from the previous frame(s) are compared.

Table 4.8 shows that all methods outperform the original method except for the methods using EMC and sequential detections. This method could suffer from aggregated false positives and detections of the previous frame which are slightly off but get a higher detection score, suppressing the detection with a lower score. Figure 4.15 shows the corresponding precision-recall curves. Here, one can observe an improvement of the existing network as well, which is indicated with a blue line.

One can observe that the temporal method using EMC and only the previous frame detection outperforms all other methods based on AP_{3D} (except for an IoU of 0.3, where it performs equal to the Seq method) and the recall at 50% precision. The maximum recall is reached by the method using EMC and aggregation of the detections. Due to the aggregation of detection, the method can use the information of the previous frames, which is the cause for the highest recall. The downside is that false positives also get aggregated, therefore this method has a lower AP_{3D} and recall at 0.5 precision. Methods that do not use EMC, lose more information and therefore perform worse compared to the methods using EMC. An exception is the sequential method without EMC, this method probably does not only has fewer true positives, but it also has fewer false positives because there is no aggregation of detections.

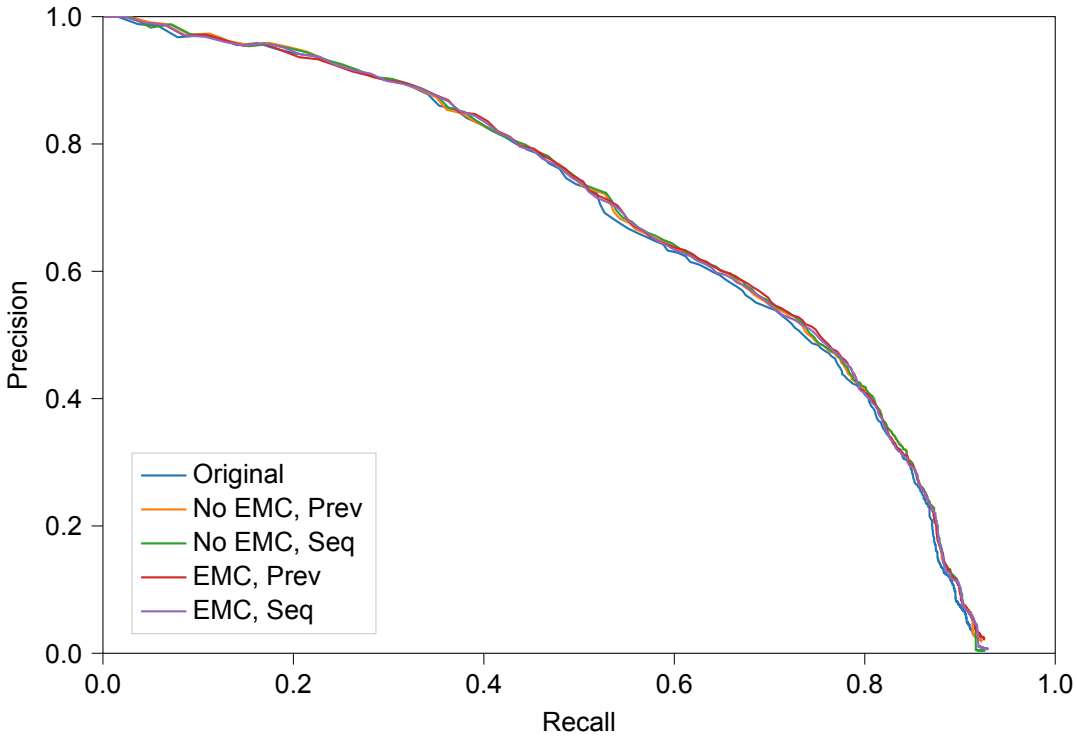


Figure 4.15: Precision-recall curves for Part-A^2 net, trained and evaluated on KITTI for an IoU of 0.1. The blue line is the original network, the four others have 3D ground truth boxes from the previous frame(s) added to their list of proposals coming from the RPN. EMC refers to ego-motion compensation, a correction to the location of the bounding boxes from the previous frames is applied based on the motion of the car. Prev and seq refer to which kind of detections are added to the 3D proposals. If only the detections of the previous frame are added to the list of proposals coming from the RPN, this is denoted with prev. If the detections of the previous frames are used, by sequentially pass the detections from each of the previous frames up to that point, it is denoted as seq.

5

Discussion

This thesis presented an analysis of three existing 3D object detectors in the context of person detection for self-driving vehicles in an urban setting. In this context, I believe the trade-off between safety and driving efficiency skews strongly towards safety; a false positive detection can result in a slower ride, whereas a false negative can result in injury or worse. For 3D person detection, the KITTI dataset is the de facto standard, and the three methods examined have been tested on this dataset by the original authors as well, making it an obvious choice for the initial experiments.

Using this dataset, a discrepancy is found between 2D and 3D object detection Average Precision (AP) which is reduced when the Intersection over Union (IoU) at which a detection is deemed a true positive is lowered (table 4.3). More importantly, this thesis showed that a large contributor to a low IoU is not the localization of the bounding box center, but the size estimation (fig. 4.5). Most of the true positives added by lowering the IoU threshold have a localization error similar to the original true positives (fig. 4.6). Additionally, the first and foremost requirement for driving safely is to localize the Vulnerable Road Users (VRUs). Therefore, I suggest that the metric for determining true positives should be independent of the bounding box dimension estimation, which can be evaluated separately as done in this thesis. This is analogous to how the orientation is evaluated with a separate performance metric, the Average Orientation Similarity (AOS).

The proposal given in this thesis is to consider a maximum top-down localization error, for example, the 15 cm from fig. 4.6. This is similar to the method for nuScenes [9], although their performance metric averages over various thresholds from 0.5 to 4 meter, which is large for pedestrian detection. Additionally, instead of categorizing a dataset based on the expected difficulty of detecting targets, it would be more useful to categorize it based on how safety-critical it is to detect a target, for example, based on their distance to the ego-vehicle, or based on their (human-annotated) importance to the traffic situation. An alternative, if the ability to detect most road users is considered the main concern for self-driving vehicles, is to invert the performance metric: what top-down localization error threshold results in an AP_{3D} of 0.99? The resulting error threshold can then be interpreted as a measure of the uncertainty of every detection given by the detector.

The accuracy of PointPillars is evaluated on both KITTI and EuroCity Persons 2.5D (ECP2.5D) as a function of distance and Light Detection And Ranging (LiDAR) points on ground truth to investigate up to what distance this method already functions well, and how much it depends on current limitations of LiDAR resolution. The high recall up to 20 m (figs. 4.8a to 4.8c) suggests that self-driving vehicles outfitted with a current State-of-the-Art (SoA) detector are not barred to drive safely in urban areas at low velocities because of their abilities to detect VRUs. This is not the same as ensuring that self-driving vehicles will be safe altogether, as this does not verify whether tracking, path prediction, and motion planning are also sufficiently capable. Zhou *et al*. [79] confirms the drop in performance over distance in the Waymo Open Dataset [59]. They compared the performance of their detector on the pedestrian and class over four categories: Overall, 0 - 30m, 30 - 50m, and 50m - infinity. Showing a decrease in performance for their multi-view fusion (MVF) detector over distance. Wang *et al*. [67] proposed a cross-range adaptation framework as an addition to existing detectors. They evaluate the effect of distance on the performance (AP_{3D}) in the car class of the KITTI dataset for three distance categories; near-range (0-40m), full-range (0-70m), and far-range (60-70m), showing decreasing performance for

an increasing distance.

The strong correlation between recall and LiDAR points in ground truth (figs. 4.8d to 4.8f) shows that increasing the number of LiDAR points on a target from less than 20 to more than 50 greatly improves performance. At the same time, an increase in LiDAR resolution would not lead to a similar increase in the computational load of voxel-based networks such as PointPillars, as they consider a fixed amount of LiDAR points per voxel. Furthermore, the number of persons with less than 50 LiDAR points in their annotated bounding box also outweighs the number of persons with more than 50 LiDAR points by almost an order of magnitude in each dataset. Something which is also supported by the authors of Frustum PointNets [45], who state that one of the common failure of their detector is due to inaccurate pose and size estimation in a sparse point cloud, where they urge that some objects have less then 5 LiDAR points on them. This indicates that the challenge for future work is to make person detectors that can accurately detect with lower resolutions.

Additionally, the correlation between recall and LiDAR points in ground truth is similar for occluded and non-occluded pedestrians. (figs. 4.9d to 4.9f). This indicates that detection difficulty is better based on LiDAR points in the ground truth bounding box rather than occlusion or distance. Such categorization has been mentioned in earlier work (e.g. [49]), and this paper shows it is indeed a useful approach.

The domain transfer experiment (table 4.5) indicates that switching to another sensor or dataset will negatively influence the performance of a network. When using a network trained on one of these datasets on a self-driving vehicle with a different LiDAR, this thesis showed that right now best practice is to disregard the intensity information of each LiDAR point. However, the same experiment also shows that the best performing network evaluated on the same dataset as trained on does use the intensity information, which implies that there is relevant information available. Therefore, a method that can effectively utilize the intensity information of a LiDAR while it is trained on another dataset is an important avenue of future work. These are not the only challenges in LiDAR domain transfer, but the ones that could be identified as the two datasets used in this paper were recorded with the same type of LiDAR sensor. Different LiDAR sensors, whose planes are not necessarily distributed with the same density, add an additional layer of complexity [49].

This thesis does not go into detail about how and which information is stored in the intensity values of each 3D point. Since not much research is done into the use for 3D object detection and the differences per dataset, this could be an interesting future research direction. Therefore, the influencing factors on the intensity value will be briefly discussed. Kashani *et al*. [21] subdivides the influencing factors on intensity into four categories. The first category is the effect of the characteristics of the surface from which the laser pulse is reflected (*e.g*. reflectance and roughness). The second category is the effects related to the acquisition (*e.g*. range, angle of incidence, and multiple returns from a laser beam). The third category includes instrumental effects (*e.g*. include transmitted energy, intensity bit depth and scaling of this digital number to a dynamical range, amplifiers for low reflected surfaces (if used), automatic gain control, brightness reducer for near distances, and aperture size). The last category consists of effects caused by the environment (*e.g*. atmospheric transmittance or wetness of an object). Since many different factors influencing the intensity measurement, there will be a constant fluctuation. Therefore, the same pedestrian will not always have the same intensity value. The intensity will especially fluctuate while driving since there can be effects of for example range and angle of incidence. Bijelic *et al*. [6] investigated the intensity values for detection in foggy weather. For this research, they used the same sensor which is used during the data collection of both KITTI and ECP2.5D (Velodyne HDL-64E). They show a correlation between fog density and intensity values. BirdNet [4] analyzed the performance of their network on the KITTI validation part with three different features used in their Birds Eye View (BEV) map; intensity, normalized density, and height. The performance on the moderate pedestrian class is 30.4%, 32.7%, and 34.0% for respectively using the intensity, normalized density, and height features. If all features are combined the network has a performance of 39.5% showing the relevance of including all features. Since there are many influences on the intensity value, it could be interesting to further analyze the intensity in order to understand how useful this information can be for 3D object detection.

The qualitative analysis of PointPillars on 30 frames of the ECP2.5D dataset suggests that the main reason for not detecting the pedestrians is the low number of LiDAR points on the pedestrian. Out of the 55 false negatives, 31 false negatives have less than or equal to 10 LiDAR points in their 3D box and 47 false negatives have fewer or equal to 25 LiDAR points in their 3D box (table 4.7). Since PointPillars does not use camera information, it is depended on the information in the point cloud. If the point cloud

does not contain enough information (*i.e.*, only a few LiDAR points on a pedestrian), the detector is not able to recognize the pedestrian. When looking at the false positives, 21 out of the 58 false positives are poles (table 4.6). These poles do have a relatively large number of LiDAR points on them. This could be related to the ability of PointPillars to capture enough of the shape information since PointPillars is not able to distinct a pedestrian from a tree/pole.

Part-A^2 net evaluated on KITTI shows similar trends for the detection score over distance and LiDAR points compared to the experiments concerning the recall of PointPillars. The detection score decreases for pedestrians which are further away than 10 meter and occluded pedestrians have a lower detection score. The is a large gap between the detection score for occluded and non-occluded pedestrians over the number of LiDAR points (fig. 4.13c). The detection score for non-occluded pedestrians stabilizes around 50 LiDAR points. For occluded pedestrians, the detection score keeps increasing up to 200 LiDAR points, from that point, the detection score shows similar trends as the non-occluded detection score. This indicates that although the number of LiDAR points is the same, Part-A^2 net is less confident about pedestrians which are occluded. One of the limitations of this analysis is the use of the KITTI dataset, since this dataset contains relatively few pedestrians, especially at a large distance, the results considering only a few pedestrians might be inconclusive. To further analyse the effect of distance and number of LiDAR points, a dataset containing more pedestrians (*e.g.* ECP2.5D) could be used.

The temporal methods proposed in this thesis outperform the original method except for the method using Ego-Motion Compensation (EMC) and sequential detections (table 4.8). The method using EMC and sequential detections has the highest recall which is related to the aggregation of the detections. The downside is that this method also suffers from a lower AP_{3D} due to the aggregation of false positives. Methods that do not use EMC, lose more information since the previous detections can not be transferred to the next frame. This can become increasingly problematic if the ego-vehicle has a high speed. Currently, the EMC methods are limited to the correction of the motion of the ego-vehicle. This could be extended, in future work, by applying a motion model to the detected objects.

6

Conclusion

This thesis presented an experimental study on 3D person localization in traffic scenes, based on monocular vision and Light Detection And Ranging (LiDAR) data. There are two main research directions investigated. The first investigates the limitations of current State-of-the-Art (SoA) 3D object detectors. The second main research question is regarding the addition of detections from the previous frame to the list of 3D proposals coming from the Region Proposal Network (RPN) evaluated in the current frame.

The performance of two 3D object detection methods is compared, AVOD and PointPillars on the KITTI and EuroCity Persons 2.5D (ECP2.5D) dataset. After this comparison, a more detailed analysis on the performance of PointPillars is presented about the relationship between the performance and distance, the number of LiDAR points, and occlusion. The performance of networks is analysed during cross-evaluation.

Furthermore, the two-stage network Part-A^2 net is used to investigate how confident a network is about the ground truth. Finally, the possibility to improve the network by adding detections from the previous frame to the list of 3D proposals from the RPN in the current frame is investigated .

6.1. What are the limitations of current SoA 3D object detectors?

To address the limitation of current SoA 3D object detectors the performance of PointPillars and AVOD on the KITTI and ECP2.5D dataset is investigated.

PointPillars outperformed AVOD AP_{3D} of 67 % vs. 48 % and 33 % vs. 28 %, for KITTI and ECP2.5D respectively, when not using LiDAR intensity information as can be seen in table 4.5. The performance of both methods was significantly lower on ECP2.5D vs. KITTI, which could be attributed to a larger prevalence of distant persons with fewer LiDAR points in ECP2.5D.

The identified limitations of current SoA 3D object detectors are presented in the following subquestions. The answer to research question 6.1 will be the sum of sub questions 6.1.1 to 6.1.6

6.1.1. What is the effect of the Intersection over Union (IoU) constraint on the performance and error analysis?

If the IoU constraint is lowered from 50 % overlap to 10 % overlap between the ground truth and the detection, the performance is increased from 51.3 to 75.2 % from PointPillars on the KITTI dataset (table 4.3). In experiments on KITTI, it is found that whereas headline results (AP_{3D}) might seem low, the 3D box center localization accuracies are quite high. The errors lowering AP_{3D} are mostly related to the estimates of the bounding box extents, especially width and length (fig. 4.5).

6.1.2. How does the performance change over distance?

When comparing the performance influence of distance, a strong decay in performance on both KITTI and ECP2.5D datasets past 20 m can be observed (fig. 4.8).

6.1.3. What is the relationship between the number of LiDAR points on an object and the performance?

The relationship between the performance over the number of LiDAR points per annotation is even stronger compared to the distance. Targets with fewer LiDAR points are harder to detect, especially pedestrians which have fewer than 50 LiDAR points on them. An strong increase in performance can be observed from 0 to 50 LiDAR points where the recall increases from 0 to around 75% (fig. 4.8).

6.1.4. What is the effect of occlusion on the performance?

When a pedestrian is occluded, it is annotated as not fully visible, this includes occlusion levels as partly occluded and largely occluded. As expected, the performance on occluded objects is lower compared to non-occluded objects. Where there is a clear gap in performance over distance, this effect is less strong for the performance over the number of LiDAR points in the 3D bounding box (fig. 4.9).

6.1.5. How does a network perform if cross-dataset evaluated?

Domain transfer experiments indicated that the KITTI and ECP2.5D have quite different biases, in the sense that training on one and testing on another leads to significantly degraded performance. The differences in AP_{3D} between networks that are evaluated on the same dataset but trained on either the KITTI or the ECP2.5D dataset range from 9.5 % to 25.8 % (table 4.5). These performance differences are smaller if the networks are trained and evaluated without intensity.

6.1.6. How confident is a detection head about ground truth pedestrians?

Out of the 2278 ground truth pedestrians in the KITTI dataset, 339 pedestrians got a detection score lower than 0.3. The detection score of the ground truth pedestrians are further examined by evaluating the relation over the distance, the number of LiDAR points on a pedestrian, and the effect of occlusion (fig. 4.13).

Past 10 meters, a decay of the average detection score is visible. There is a strong relationship between average detection score and occlusion since the average detection score of non-occluded pedestrians is always higher than for occluded pedestrians.

There is a relation between the score of a pedestrians and the number of LiDAR points. For non-occluded pedestrians, there is a strong increase in detection score up to 50 LiDAR points, from that point on the score is generally stable. For occluded pedestrians, the average detection score keeps increasing up to around 200 LiDAR points. The average detection score is lower for occluded pedestrians compared to non-occluded pedestrians up to 200 LiDAR points.

Conclusively, it is shown that if the RPN proposes the location of all ground truth pedestrians, the network is increasingly less confident for pedestrians further away than 10 meters, increasingly confident for an increasing number of LiDAR points, and is less confident about occluded pedestrians.

6.2. What is the effect on the performance if detections from the previous frame(s) are added to the list of proposals coming from the RPN?

To examine the effect on the performance if detections from the previous frame(s), the detections from the previous frame(s) are added to the list of 3D proposals of the RPN. The addition of the detections resulted in an increase in performance. When only using the previous frame, the performance increased with 0.9% for an IoU of 10 %, and the recall at 50 % precision increased with 1.7 % compared to the original method (table 4.8). When aggregating detections over a sequence, the performance is lower because it aggregates more false than true positives. Therefore, the maximum recall increased with 1.6 % compared to the original method.

6.3. Future work

The future work will consist of two sections. The first section will consist of future work related to this thesis specific (section 6.3.1).

Secondly, the general future work of the field of 3D object detection will be presented, which is based on my literature study, the published and submitted papers, and my experiences during the past

year (section 6.3.2).

6.3.1. Future work thesis

The scope of this thesis is mostly limited to pedestrians. Future work will include the extension of the experiments to other Vulnerable Road User (VRU) instead of pedestrians only.

The hyperparameters of the networks which are trained on ECP2.5D are not fine-tuned since these hyperparameters are assumed part of the network. In future work, the parameters could be fine-tuned to increase performance.

An interesting direction for research would be to focus on increasing the performance of 3D object detectors on objects with a low number of LiDAR points, specifically since both the performance and detection scores are low for objects with less than 50 LiDAR points on them.

The domain transfer experiments show a better domain adaptation if the network is trained without intensity. Although the best performing PP networks evaluated on the same dataset as trained on are networks using intensity information, which implies that there is relevant information available. Therefore, future research could be to investigate how to effectively utilize the intensity information of a LiDAR while it is trained on another dataset.

The detection score (confidence) of Part-A^2 net is evaluated using the KITTI dataset. This analysis could be extended to a dataset containing more pedestrians (*e.g.* ECP2.5D). The analysis of the performance of Part-A^2 net could be invested by evaluating precision and recall over the distance, the number of LiDAR and occlusion level to see if similar results are shown as for PointPillars.

A qualitative study could be performed into the objects with a low detection score (*e.g.* lower than 0.1) to determine the reason why Part-A^2 net is not confident about these ground truth objects and what these objects have in common.

In the KITTI and ECP2.5D dataset, the objects are annotated in the camera perspective only, while LiDAR is capable of 360-degrees detection, future work could therefore be to extend this analysis to a dataset containing 360-degree labels and adapting the network to predict pedestrians for the whole 360-degrees.

The ego-motion compensated detections of the previous frame are appended to the output of the RPN. Future work could benefit from an adaptation to this method, for example by adding more weight to the previous detections (artificially increasing the detection score), although this could also result in more false positives.

Inspired by Bergmann *et al*. [5], one could extend our temporal fusion 3D object detection method by a motion model and a re-identification algorithm.

6.3.2. General future work of the field of 3D object detection

This section is mostly based on my view and ideas of the 3D object detection field. The ideas and views arose during my literature study, the writing of the articles, and further thesis work. These ideas are substantiated with sources where possible.

Future research directions can be divided into five different categories; fusing multiple sensor modalities, a detector which can use whole point clouds consisting of a variable number of LiDAR points, further improving the fusion of data over time, the improvement of tracking methods based on the output of detectors and the efficiency of 3D object detectors.

6.3.2.1. Sensor fusion

Currently, the KITTI 3D object benchmark is the most used benchmark for 3D object detectors. When looking at the performance on the top ten detectors on the pedestrian class, one can notice that the best four are detectors are based on the LiDAR modality only. As can be seen in table 6.1, there are only five detectors using data fusion in the top fifteen.

I argue that detectors making use of multiple modalities should perform better compared to single modality detectors. The information from a camera and LiDAR is complementary. A LiDAR is strong at determining the distance, while a camera image is denser in information but suffers from lightning and weather conditions. LiDAR data is critical for a 3D detector since this detector is judged on its ability to accurately predict objects in 3D space. By combining LiDAR point clouds and camera images, a detector could benefit from the shape and texture information in the image [2, 19] and the depth in

Table 6.1: The top 15 on the moderate part of the KITTI 3D object detection benchmark on the class pedestrian accessed on 16-08-2020. Only methods with published articles are listed. Score refers to the moderate performance (AP_{3D}) of an object detector on the pedestrian class. Methods containing a *(star) are modules that can be added to an already existing 3D object detector.

Network	Modalities	Fusion type	Stages	Score (%)	Runtime (s)
HotSpotNet [11]	LiDAR	-	Single	45.37	0.04
TANet [33]	LiDAR	-	Single	44.34	0.035
3DSSD [75]	LiDAR	-	Single	44.27	0.04
Point-GNN [55]	LiDAR	-	Single	43.77	0.6
F-ConvNet [68]	LiDAR + Camera	Sequential	Single	43.38	0.47
PartA^2 [52]	LiDAR	-	Two	43.35	0.08
PV-RCNN [54]	LiDAR	-	Two	43.29	0.08
VMVS* [24]	LiDAR + Camera	Deep	*	43.27	0.25
STD [74]	LiDAR	-	Two	42.47	0.08
AVOD-FPN [25]	LiDAR + Camera	Deep	Two	42.27	0.1
F-PointNet [45]	LiDAR + Camera	Sequential	Single	42.15	0.17
PointPillars [28]	LiDAR	-	Single	41.92	0.016
PointPainting [64]	LiDAR + Camera	Deep	Single	40.97	0.4
MMLab-PointRCNN [53]	LiDAR	-	Two	39.37	0.1
ARPNET [76]	LiDAR	-	Two	39.31	0.08

the LiDAR point cloud. Liang *et al*. [30] states that detectors can benefit from the information in high-resolution images especially at large distances. This could be beneficial for a 3D detector because the LiDAR modality suffers from extreme data sparsity at large distances. Shi *et al*. [53] developed a LiDAR-only method, in which they consider that leveraging image information could increase performance, especially for pedestrians since they have a relatively small size in a point cloud where an image captures more details.

There are four general ways to combine the information from multiple heterogeneous sensors: early, deep, sequential, and late fusion (as explained in section 2.3). Sequential fusion and late fusion are considered to be sub-optimal. Since sequential fusion heavily depends on multiple modalities, it is therefore not robust to missing data. When applying late fusion, only the output probabilities of several separate networks are fused, compared to the information fusion which takes place in the other methods this method is considered inferior. To fuse information and have a robust detector which can handle missing modalities, early and deep fusion are considered as most promising future research directions. Therefore, for the remaining part of this section, some pitfalls and possible solutions for early and deep fusion are discussed.

For early fusion, the data needs to be spatially aligned to fuse them together. This alignment can be challenging because different sensors have different resolutions. When looking at the fusion of LiDAR and image data, one can observe the mismatch in sensor density. An example is the fusion of the camera and LiDAR modality where a point cloud is sparser compared to an image. Due to the sparseness of the LiDAR modality, not every image pixel has its own distance value. If an image and LiDAR point cloud are fused using early fusion, not every pixel will have its own LiDAR point. Therefore, some information will be removed during fusion which causes an information loss. A remedy for the information loss caused by early fusion could be to decrease the sparsity of a point cloud. The sparsity of point clouds could be decreased by either using depth completion (can be based on a sparse point cloud and an image, *e.g*. NLSPN [43]) or depth prediction (based on an image *e.g*. DORN [15]). A more dense point cloud could be fused with an image using early fusing where less RGB values have to removed due to the sparsity of a LiDAR point could. On top of adding the RGB image features, one could also add the class predicted by pixel-level image segmentation (*e.g*. MSeg [27]) to each point in the point cloud. An example of a network that adds the image class information to a sparse point cloud is PointPainting [64], this concept could also be applied to point cloud which is made denser by depth prediction or depth completion.

For deep fusion, the features of multiple modalities needs to be fused. Some depth completion methods perform deep fusion of an image and a sparse depth map (*e.g*. CSPN++ [13]). These depth

completion methods could serve as an inspiration for 3D object detectors about how to spatially align the features.

6.3.2.2. Variable input LiDAR network

Point clouds are unordered sets of points. In order use to use these point clouds one needs to pre-process them. Currently, the voxelization (*e.g*. PointPillars [28]), grouping in sets (*e.g*. PointNet [44]), and the creation of graphs (*e.g*. Point-GNN [55]) of the LiDAR point cloud are often used pre-processing methods. All these pre-processing steps remove data which could be the bottleneck for current 3D object detection networks.

Regarding the variable input of a network using a LiDAR point cloud, there are two directions for future work.

First, in the pursuit to find a pre-processing free network, one possible research direction could be to improve current pre-processing methods, an example is reconfigurable voxels [65] a method which creates voxels based on the local spatial distribution of points in the point cloud.

Secondly, developing a 3D object detection network that does not need any ordering of the point cloud which reduces the need for pre-processing steps. Guo *et al*. [19] states that the transformation of data has consequences in terms of information loss. This loss of information could be reduced when pre-processing steps are no longer needed.

6.3.2.3. Temporal fusion data

Another future research direction could be the temporal fusion of data. There are four interesting research directions regarding temporal fusion.

The first direction is to create a detector where multiple point clouds are separately passed into the 3D object detection network to detect VRUs. An example is FaF [35], used for the 3D detection of cars which concatenates the past 5 point clouds. This method could be taken as inspiration to create a network for 3D VRUs detection.

Another possibility is to fuse the point clouds of the previous frames into one point cloud (after compensating for the motion of the car) resulting in a denser point cloud. This would especially benefit static objects which were at some point close and are moving further away since they would have had only a few points on them and now they can have a large number of points on them.

One could also use the differences between the previous and the current point clouds to determine which objects are moving and thus could be more interesting.

Lastly, one could also take the features of previous point clouds into account by inserting them into a Recurrent Neural Network (RNN) which is able to use the information of previous point clouds.

6.3.2.4. Temporal fusion predictions (tracking)

In this thesis a temporal fusion method for 3D object detection is proposed that is inspired by Bergmann *et al*. [5]. In our work, the 3D bounding boxes from the previous frame are regressed in the current frame. Future work could be to extend this base method by applying a motion model and a re-identification algorithm as done in 2D by Bergmann *et al*.. One could also extend this work by applying a 3D tracker onto the resulting detections. An example of such a 3D tracker could be the tracker of Weng *et al*. [70]. Their tracker uses a combination of a 3D Kalman filter and the Hungarian algorithm for state estimation and data association.

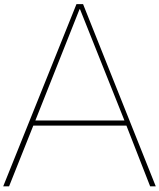
6.3.2.5. Improving efficiency

In the area of intelligent vehicles, it is important to have a short inference time to pass the observed information quickly to the brain of the vehicle. The inference time should therefore be real-time which is defined by Mauri *et al*. [36] as less than 100 ms or greater than 10 FPS.

To improve the inference speed of 3D object detectors, many different directions can be taken. One possible direction is decreasing the computational time for convolutions on voxels. PVCNN [34] presented a study where Voxel-based models and point-based models are compared. They conclude that a lot of time is lost on handling the irregularity of the point-based representation. As can be seen in table 6.1, only three out of the top 15 detection networks on the KITTI dataset run with more or equal to 20 frames per second.

One important note has to be made regarding PointPillars [28] with a run-time of 16 milliseconds which equals 62.5 frames per second. They claim their speedup is mostly related to three aspects. First, their encoding only takes 1.3 ms which is 2 orders of magnitude faster compared to the VoxelNet encoder (190 ms) [78]. Secondly, they attribute their speedup to their slim design. They used one PointNet instead of the two sequential PointNets as suggested by VoxelNet. Besides using only one PointNet, they also lowered the dimensions of the first block and reduced the output dimension of the up-sampled feature layers by half to 128. These steps decreased the run-time by 10.9 ms. Lastly, although their experiments were performed in PyTorch, they build their GPU kernels for the encoding, backbone, and detection head in NVIDIA TensorRT which speedup the inference time with 45.5%.

Future work could identify what the performance drop will be if the number of weights of a detector are lowered. Another direction to decrease the run-time could be to build certain modules of the 3D object detector in NVIDIA TensorRT. Lastly, research could be done into how to decrease the computational time for convolutions on voxels.



Published and Submitted articles

A.1. Article published at the IEEE Intelligent Vehicles Symposium 2020

This section contains the article which is published at the IEEE Intelligent Vehicles Symposium 2020.

An Experimental Study on 3D Person Localization in Traffic Scenes

Joram R. van der Sluis*,1, Ewoud A.I. Pool*,1 and Dariu M. Gavrila1

***Abstract*— This paper presents an experimental study on 3D person localization (i.e. pedestrians, cyclists) in traffic scenes, using monocular vision and LiDAR data. We first analyze the detection performance of two top-ranking methods (PointPillars and AVOD) on the KITTI benchmark, with respect to varying Intersection over Union (IoU) settings and the underlying parameters of 3D bounding box location, extent and orientation. Given that the KITTI dataset contains relatively few 3D person instances, we also consider the new EuroCity Persons 2.5D (ECP2.5D) dataset, which is one order of magnitude larger. We perform domain transfer experiments between the KITTI and ECP2.5D datasets, to examine how these datasets generalize with respect to each other.**

I. Introduction

According to a recent report of the World Health Organisation, about half of the 1.3 million people killed yearly in traffic worldwide involve Vulnerable Road Users (VRUs), i.e. pedestrians, cyclists and other riders. For much of the past two decades, vision was the dominant sensor modality for intelligent vehicles to detect VRU. Strong progress has been made on 2D image-based VRU detection facilitated by novel (deep learning) methods, faster processors and more data (including benchmarks, e.g. [1], [2], [3]). 3D localization from 2D detections can subsequently be achieved by back-projection, disparity computation [4], and/or association with radar targets. Vision-based VRU detection is meanwhile incorporated in active safety systems of various premium vehicles on the market.

Still, current active VRU safety systems are deployed in the context of driver assistance. With the advent of fully self-driving vehicles, performance needs to be significantly upped, as a driver is no longer available as a back-up. The LiDAR sensor is an attractive sensor for self-driving vehicles, stemming from its capabilities to directly and accurately measure distances and to deal with low-light environments. KITTI [5] meanwhile offers a 3D object detection benchmark, including one for pedestrians. The latter leader-board currently lists a 3D Average Precision (AP_{3D}) of 51% and around 40% for the *easy* and *all* targets, respectively (in contrast, the state-of-the-art in 2D object detection attains an Average Precision (AP) of 75% overall).

This paper presents an experimental study on 3D person localization (i.e. pedestrians, cyclists) in traffic scenes, using monocular vision and LiDAR data. We consider two 3D object detection methods, PointPillars [6] and AVOD [7], which are among the top performers on the KITTI benchmark, see fig. 1. We investigate the effect of the varying IoU setting on detection performance and quantify the various errors in terms of 3D bounding box location, extent and orientation. Given that the KITTI benchmark contains relatively few 3D person instances, we also perform experiments on a large subset of the new EuroCity Persons 2.5D (ECP2.5D) dataset [8]. Apart from being one order of magnitude larger than KITTI, ECP2.5D has advantages in terms of diversity (e.g. geographical coverage, time of day/season, weather conditions) and by being devoid of privacy-driven image blurring. This makes ECP2.5D also attractive when compared to other recent dataset additions, see table II. Finally, we perform domain transfer experiments between KITTI and ECP2.5D, to examine how these datasets relate to each other.

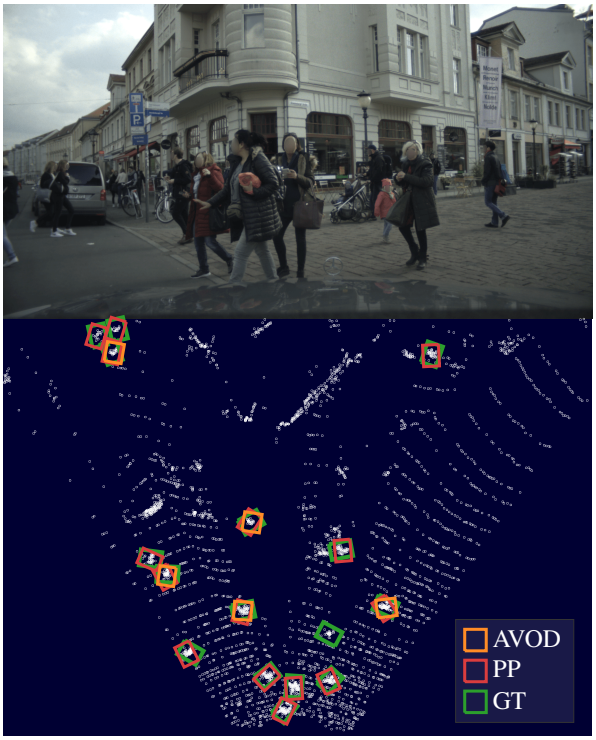


Fig. 1. An example of the predicted bounding boxes of PointPillars [6] (PP) and AVOD [7] on a scene from the EuroCity Persons 2.5D [8], along with the annotated ground truth (GT).

II. Previous Work

We focus our discussion on previous 3D object detection methods that use neural network architectures, as they are the current best performers in the various benchmarks.

One way to categorize this work is by sensor modality, i.e. either a single modality or a fusion of multiple modalities. The commonly used sensors used are (monocular) camera

*) Authors contributed equally
1) Intelligent Vehicles group, TU Delft, The Netherlands

TABLE I
COMPARISON OF AVOD AND POINTPILLARS.

	AVOD	PointPillars
Modality	LiDAR + image	LiDAR
Stages	Two-stage	Single-stage
Bounding box regression	four corners, heights, orientation	3D center point, length, width, height, orientation

and LiDAR. However, the RGB-only methods (e.g. Shift R-CNN [9]) are generally outperformed by methods that instead use LiDAR information. These LiDAR-only networks map the point cloud to either a 2D or a 3D representation. Examples of 2D representations are Birds Eye View (used by *e.g.* HDNet [10]) and Range View (*e.g.* LaserNet [11]). Networks can also map the point cloud to 3D representations like Voxels (*e.g.* Voxelnet [12]), Pillars (*e.g.* PointPillars [6]), or Stixels (*e.g.* SCNet [13]).

Multi-sensor modality networks, or fusion networks, use both camera and LiDAR. Here, all the previously mentioned LiDAR mappings can be used to fuse with the camera data. How they are fused exactly falls into four categories. The first category is early fusion, where the modalities are concatenated before being passed into a neural network. An example of early fusion is MVX-Net PointFusion [14] where the pointcloud is projected onto a RGB-image and then concatenated. Secondly, deep fusion networks fuse the modalities after they have already been processed by a part of the network, for example PointFusion [15]. Here, the features from a PointNet [16] and a ResNet-50 are concatenated. With deep fusion, it is also possible to fuse the various modalities at multiple stages, as is done with AVOD [7]. Within such a deep fusion network, the performance is dependent on the feature encoder used [17]. Thirdly, late fusion takes the output of two or more independent networks and fuses the class probabilities [18]. Lastly, sequential fusion processes the sensor modalities in sequence. For example, Frustum PointNets [19] and Frustrum Convnet [20] use a 2D image detector to select frustums in a pointcloud, which is then processed separately.

Another way of categorizing previous 3D object detection methods is by the number of stages used by the network. Two-stage approaches utilize a Region Proposal Network (RPN) to generate bounding boxes which are individually evaluated (*e.g.* STD [21]). Single-state approaches instead evaluate predetermined bounding boxes (*e.g.* PointPainting [22]), also called anchor boxes.

Table I highlights the differences between PointPillars [6] and AVOD [7], two of the best performing LiDAR and fusion networks, respectively, with code available at the time of writing. These will be used later in the experiments.

In terms of existing datasets, one of the first 3D object detection benchmarks was an extension to KITTI [5], released in 2017, which contains around 9400 pedestrians (of which half in the publicly available training set). Since then, KITTI has become the de facto standard for 3D object detection. However, because of the relatively small dataset size, performances can differ a lot on the validation and test set. More recent dataset additions to KITTI are significantly larger and more diverse, see table II.

TABLE II
OVERVIEW OF TRAFFIC-RELATED 3D PERSONS DATASETS. A DASH DENOTES THAT THE INFORMATION COULD NOT BE DETERMINED.

Dataset	**Waymo** [23]	**nuScenes** [24]	**Argoverse** [25]	**Lyft** [26]	**KITTI** [5]	**ECP2.5D** [8]
# Countries	1	2	1	1	1	12
# Cities	2	2	2	1	1	30
# Imgs	800K	34K	350K	55K	15K	46K
# Peds	2.8M	222K	132K	25K	9.4K	123K
# Riders	67K	24K	11K	22K	3.3K	13K
# Seasons	-	-	1	1	1	4
Weather	dry, rain	dry, rain	dry	dry	dry	dry, rain
Time of day	day, night	day, night	day, night	-	day	day, night
Unblurred	✗	✗	✗	✗	✓	✓

This paper presents an experimental study on monocular and LiDAR-based 3D person detection. Its specific contributions are:

- A performance analysis of two state-of-the-art methods (PointPillars and AVOD) on KITTI, with respect to varying IoU and the underlying parameters of 3D bounding box location, extent and orientation.
- Results from domain transfer experiments between KITTI and ECP2.5D.

III. METHODOLOGY

The goal of 3D person detectors is to detect the bounding boxes of VRUs in the scene. In KITTI, these bounding boxes have seven degrees of freedom (fig. 2). The 3D position is given in a coordinate system with respect to the egovehicle, where x is the position of the bounding box center lateral to the vehicle, z is the position longitudinal to the vehicle (i.e. depth), and y determines the altitude of the bounding box center. The bounding box dimensions are specified by a width w, length l and height h, and finally each bounding box has a yaw rotation θ. The top and bottom face of the bounding box are assumed to be parallel to the $y = 0$ plane. The predicted bounding boxes will also have a detection score d related to them.

A. Intersection over Union (IoU)

To evaluate the performance of an object detector, one needs to count a predicted bounding box as valid or non-valid (i.e. true positive or false positive). In 3D (as well as 2D) object detection, the method to assess if a proposed bounding box is a true- or false-positive is based on IoU. It is defined as the intersection (or overlap) of a 3D bounding box prediction (B_p) and ground truth (B_{gt}) divided by the union of the prediction and ground truth. When both bounding boxes only have a yaw rotation, this can be written as [27]:

$$\text{IoU} = \frac{B_p \cap B_{gt}}{B_p \cup B_{gt}} = \frac{A_o \times h_o}{V_{gt} + V_p - A_o \times h_o} \tag{1}$$

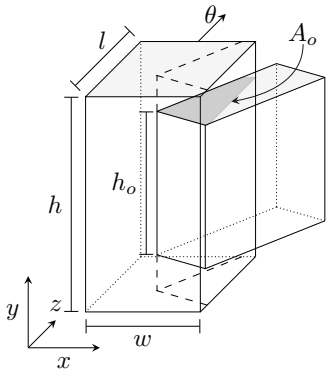


Fig. 2. A visualization of the parameters relevant for computing the IoU of a ground truth and predicted bounding box. The darker shaded area indicates the overlap area A_o. In this figure, the overlapping height h_o is equal to the height of the smaller bounding box.

Where V_p and V_{gt} are the volume of the predicted and ground truth bounding box. The overlap of volumes can be computed from the overlapping top-view area A_o and the overlapping height (h_o), see fig. 2. In KITTI, a predicted bounding box is seen as a true positive if it has an IoU of more than 0.5. Only one predicted bounding box can be marked as a true positive for any ground truth bounding box.

B. Performance metrics

After the true positives have been determined, it is possible to compute the two metrics as defined in the KITTI benchmark for 3D object detection: 3D Average Precision (AP_{3D}) and Average Orientation Similarity (AOS) [5].

The AP_{3D} averages the maximum attained precision s with at least a recall r for a fixed range of recall values [28]:

$$\mathrm{A}P_{3D} = \frac{1}{40} \sum_{r \in \{\frac{1}{40}, \frac{2}{40}, \ldots, 1\}} \max_{\tilde{r}:\tilde{r} \geq r} s(\tilde{r}) \tag{2}$$

As precision and recall both depend on the amount of true positives, the AP_{3D} strongly depends on the IoU threshold.

Where the AP_{3D} verifies whether the bounding boxes are in the correct place, the AOS additionally verifies the correctness of their orientations:

$$AOS = \frac{1}{40} \sum_{r \in \{\frac{1}{40}, \frac{2}{40}, \ldots, 1\}} \max_{\tilde{r}:\tilde{r} \geq r} \tilde{s}(\tilde{r}) \tag{3}$$

$$\tilde{s}(r) = \frac{1}{|\mathcal{D}(r)|} \sum_{i \in \mathcal{D}(r)} \frac{1 + \cos \Delta_\theta^{(i)}}{2} \delta_i \tag{4}$$

Where $\mathcal{D}(r)$ denotes the set of all objects at a specific recall rate r and $\Delta_\theta^{(i)}$ the difference between the estimated and the real orientation. The indicator δ_i is one if the predicted bounding box is seen as a true positive, and zero otherwise. If every true positive predicted bounding box has an orientation error of 0, eq. (4) reduces to the precision at that recall rate.

IV. Experiments

Experiments were performed with the codebase of the authors of AVOD[1] and the codebase recommended by the authors of PointPillars[2] as is, using the best performing network as reported in their papers. Thus for AVOD, we use AVOD-FPN, and for PointPillars, we use a spatial resolution of 0.16×0.16 m^2.

A. Datasets overview

Figure 3 shows the distribution of the VRUs locations relative to the vehicle, for the publicly available part of both KITTI and ECP2.5D. The bulk of the detections in the KITTI dataset lies within 30 m distance of the ego-vehicle. Both datasets have a bias towards VRUs being on the right side of the ego-vehicle.

We are using the same KITTI 1:1 train/validation split as specified by the AVOD and PP codebases. The KITTI dataset contains 2.2K/0.7K and 2.3K/0.9K pedestrian/cyclist annotations for the train and validation split respectively. The validation split can be divided in three parts which are “easy”, “moderate” and “hard”, as defined by KITTI. The ECP2.5D dataset has a larger amount of annotations for the 3D position and orientation, but lacks width, length and height annotation. We will use the median bounding box dimensions of the train split of the KITTI dataset so both networks can still regress a full bounding box. This paper uses the ”Day” subset of ECP2.5D as its basis. Additionally, the underlying EuroCity Persons (ECP) dataset misses an orientation label for 386 pedestrians and 144 riders, these are set to “Don’t Care”. This results in 62.3K/7.3K pedestrian/cyclist annotations in the training split, and 12.6K/1.3K pedestrian/cyclist annotations in the validation split. The test set ground truth annotations of both datasets is not made public, so all evaluations done in the rest of this paper are done using the validation splits of either dataset as mentioned here.

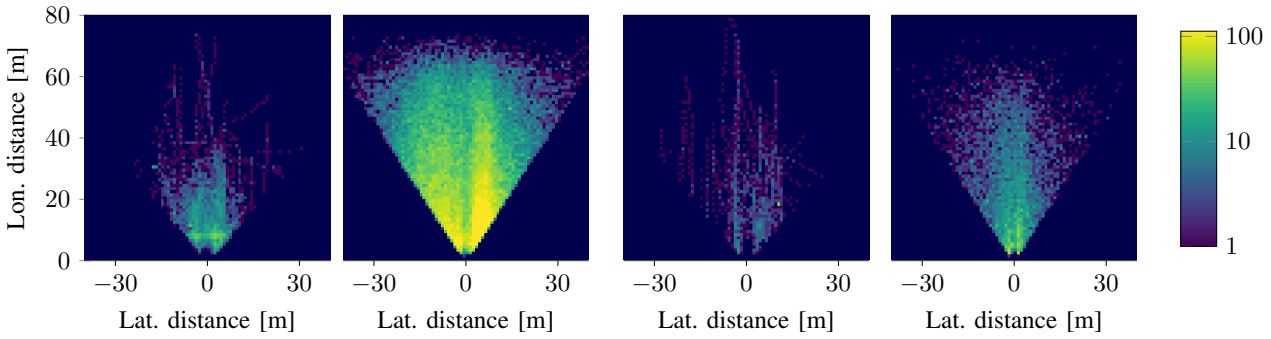
Both datasets use the Velodyne HDL-64E (LiDAR) sensor. The intensity of the LiDAR points in KITTI fall in 100 discrete bins of between 0 and 1. ECP2.5D has an intensity on a continuous range between 1.0 and 255.

B. Effect of IoU on performance and error analysis

Performance with lower IoU constraints: Table III shows the performance of PointPillars and AVOD on KITTI for the cyclist and the pedestrian classes. PointPillars has a higher AP_{3D} than AVOD, even though their scores on the moderate test split on the KITTI benchmark differ less than one percent. However, the results we find for AVOD are comparable to those found on the validation split in the comparison study of [17]. Lowering the IoU threshold increases the AP_{3D} by a large margin. For example, the AP_{3D} of PointPillars on pedestrians increases from 55.8 to 77.2 (21 %).

This is further visualized in fig. 4, which shows a histogram of the IoU found for all true positive detections

[1]https://github.com/kujason/avod
[2]https://github.com/traveller59/second.pytorch



(a) KITTI pedestrians (b) ECP2.5D pedestrians (c) KITTI cyclists (d) ECP2.5D cyclists

Fig. 3. The overall distribution over location of pedestrians and cyclists in both the KITTI and ECP2.5D dataset as a log plot. In all figures, the egovehicle is positioned at $(0, 0)$, looking upwards. Each pixel in the image corresponds to a $1x1$ square meter area. The darkest blue region indicates areas with zero pedestrians.

TABLE III
AOS AND AP_{3D} PERFORMANCE OF POINTPILLARS (PP) AND AVOD, TRAINED ON KITTI AND EVALUATED ON THE MODERATE PART OF THE KITTI VALIDATION SPLIT.

	Pedestrian		Cyclist	
IoU	AP_{3D}	AOS	AP_{3D}	AOS
PP				
0.5	55.8	27.0	58.5	5.8
0.4	71.5	34.5	63.7	6.9
0.3	76.5	37.1	64.9	7.1
0.2	77.1	37.4	66.0	7.2
0.1	77.2	37.5	66.0	7.2
AVOD				
0.5	41.2	32.3	35.1	34.8
0.4	50.0	38.3	36.3	35.9
0.3	52.5	40.1	36.3	35.9
0.2	52.7	40.2	36.3	35.9
0.1	52.7	40.3	36.3	35.9

at an IoU threshold of 0.1. This histogram shows that for pedestrians more than 15% of the detections of PointPillars and 10% of the detections of AVOD had an IoU between 0.4 and 0.5, just outside the normal IoU threshold. A similar effect is seen for cyclists, albeit less strongly.

The upper bound of the AOS is the AP_{3D}, as mentioned in section III-B. Table III shows that even though the general detection accuracy of AVOD is lower than PointPillars, its AOS is almost perfect, especially for cyclists. The AOS of PointPillars is far worse than the AOS noted on the online KITTI benchmark. A closer inspection of the distribution of the orientation error (fig. 5) shows that for PointPillars, the orientation error peaks around 0 or 180 degrees. In the paper of PointPillars, the authors state that the orientation loss used cannot distinguish between flipped boxes, for which they use an additional binary classification loss. The orientation errors of PointPillars shown in fig. 5 seem to indicate that while the original overall orientation loss works as expected, there might be an implementation issue with the binary classification loss in the codebase of SECOND. As for

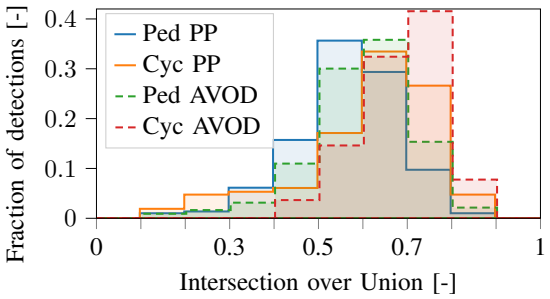


Fig. 4. PointPillars and AVOD trained on KITTI: a histogram of what fraction of true positive detections had what IoU (IoU threshold of 0.1).

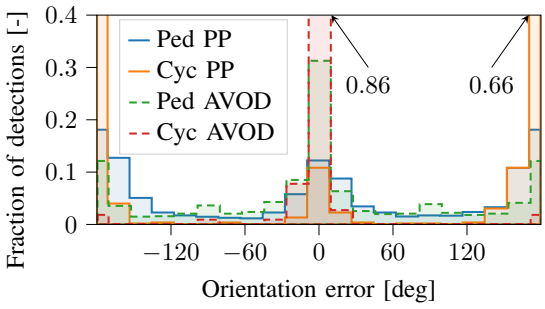


Fig. 5. PointPillars and AVOD trained on KITTI: A histogram of the orientation error. The arrows indicate the fraction of detections of the two bars outside of the y axis range. Most orientation errors lie either between -40 and 40 degrees, or between 140 and -140 degrees.

AVOD, almost all of the orientation estimates indeed have an error closer to 0 degrees as was expected from their AOS.

Error analysis of bounding box estimation: Figure 6 shows the error made in position and size of the predicted bounding boxes on pedestrians by PointPillars. The smallest errors are made on the x and the z estimation: the lateral and longitudinal position. The largest error is made on the width and length estimation. These depend on the stride of a pedestrian, as well as the location of their arms, which can be difficult to infer at larger distances.

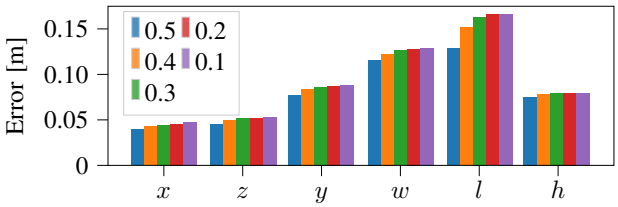


Fig. 6. PointPillars trained on KITTI: the average error between the prediction and the ground truth for the pedestrian detections on x, z, y, w, l and h, at different IoUs thresholds. The largest error is made on the altitude estimation, together with the bounding box width and length.

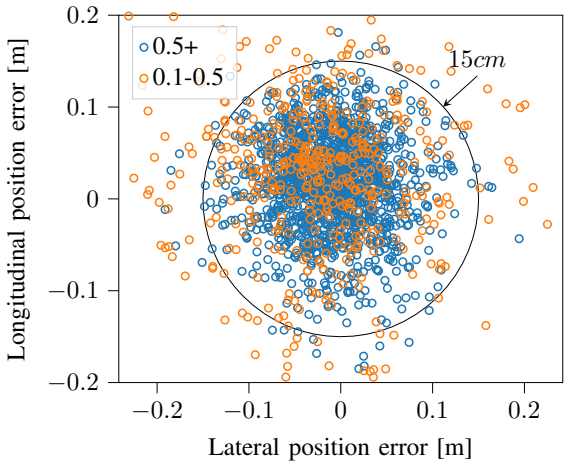


Fig. 7. PointPillars trained on KITTI: The localization error made by true positive detections of **pedestrians**, from a bird's eye viewpoint. Of the true positive detections with an IoU of over 0.5, 1462 out of 1494 detections lie within a radius of 15 cm. Of the true positive detections with an IoU between 0.1 and 0.5, 349 out of 478 lie within that radius.

The relatively small error in x and the z position (essentially a top-down position estimate) is visualized in fig. 7. It shows the x and z position error made for the true positive detections for the original IoU threshold, as well as the detections between an IoU of 0.1 and 0.5. A lot of the detections with an IoU below 0.5 are still accurate at estimating the position. For an IoU threshold of 0.5, nearly all of the true positive detections (1462 of the 1494) lie within a radius of 15 cm. When looking at the detections found with an IoU threshold of 0.1, a total of 1811 detections lie within a radius of 15 cm. In other words, using a radius of 15 cm for as a metric to determine true positives instead of an IoU of at least 0.5 shows an increase in detections of 23 %. The same data is put more succinctly in fig. 8, with cyclists added as well. It shows the amount of true positive detections which fall below a specific Euclidean position error. Cyclists see a smaller benefit, but as their annotated bounding boxes are larger, it is possible to make a larger position error without affecting the IoU as much.

Accuracy evaluation using fixed bounding boxes during training: The relatively large errors in width and length suggest that these two 3D object detectors are not able to properly estimate these. To investigate the influence of the

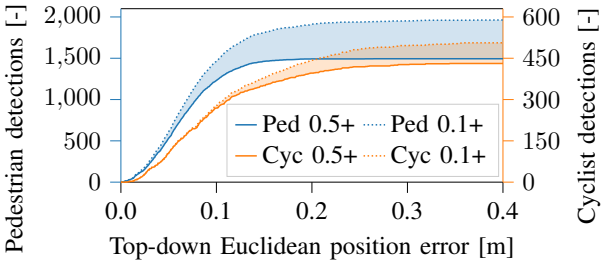


Fig. 8. PointPillars trained on KITTI: given a certain Euclidean position error threshold, how many detections would be inside. The solid line shows what Euclidean error is made by detections using the current default IoU threshold of 0.5. The dotted line shows the amount of detections within a given Euclidean error for an IoU threshold of 0.1. The shaded area then shows the amount of detections added.

TABLE IV

AP_{3D} PERFORMANCE OF POINTPILLARS (PP) AND AVOD FOR TWO IOU THRESHOLDS, EVALUATED ON THE MODERATE PART OF THE KITTI VALIDATION SPLIT. THE NETWORKS WERE TRAINED ON THE ORIGINAL KITTI GROUND TRUTH (ORIG.) OR THE GROUND TRUTH WITH FIXED BOUNDING BOX DIMENSIONS (FIXED).

	Pedestrian		Cyclist	
IoU	Orig.	Fixed	Orig.	Fixed
PP				
0.5	55.8	54.6	58.5	62.6
0.1	77.2	73.3	66.0	68.1
AVOD				
0.5	41.2	46.0	35.1	35.5
0.1	52.7	59.6	36.3	38.8

bounding boxes dimensions, we train on a version of the KITTI dataset train split where we fix the dimensions of each VRU. The dimensions are fixed to the median dimensions of their respective class. Then, we evaluate on the original KITTI dataset validation split with the correct dimensions. See Table IV. Where at an IoU of 0.5, the performance of PointPillars on the pedestrian class drops with 1.2 %, the performance of the cyclist class even increases with 3.9 %. Next to that, AVOD shows an increase for both the pedestrian and the cyclist class.

C. Cross-dataset Evaluations

To see how well each dataset generalizes, we train both networks on the one dataset and evaluate them on the other. Because the original PointPillars uses the intensity information of the points in the point cloud as well, we train it once *with* this intensity information present, and once *without*. AVOD does not use the intensity information, and therefore only needs to be trained once on each dataset. To ensure the datasets are compatible, we linearly rescaled the LiDAR intensity values in each dataset to the same range.

Table V shows the resulting AP_{3D}. PointPillars using LiDAR intensity data and the ("native") training sets, corresponding to the datasets tested, has the best performance on both datasets. When not using LiDAR intensity data, PointPillars' performance slightly drops, but still clearly

TABLE V

AP_{3D} PERFORMANCE OF POINTPILLARS (PP) AND AVOD FOR AN IOU OF 0.1 ON THE MODERATE VALIDATION SPLIT OF KITTI AND ECP2.5D. BOLD INDICATES HIGHEST PERFORMANCE IN THAT COLUMN.

	AP_{3D}	
Trained network	ECP2.5D	KITTI
with intensity :		
PP on ECP2.5D	**34.1**	46.7
PP on KITTI	6.9	**77.2**
w/o intensity :		
PP on ECP2.5D	32.8	55.4
PP on KITTI	26.0	67.5
AVOD on ECP2.5D	26.8	34.0
AVOD on KITTI	5.0	52.7

outperforms AVOD on both datasets, when using the native training sets. Performance of both methods was significantly lower on ECP2.5D vs. KITTI,

When non-native training sets are used, performances degrade significantly for both methods, both when moving from KITTI to ECP2D, and vice versa. The resulting performance degradation for PointPillars is less severe when the LiDAR intensity data is not used. More research is needed to improve cross-domain adaptation (e.g. [29]).

V. CONCLUSION

This paper presented an experimental study on 3D person localization in traffic scenes, on the basis of monocular vision and LiDAR data. In experiments on KITTI, we found that whereas headline results (AP_{3D}) results might seem low, the 3D box center localization accuracies are in fact quite high. The errors lowering AP_{3D} are mostly related to the estimates of the bounding box extents (especially, width and length).

PointPillars clearly outperformed AVOD (AP_{3D} of 68% vs. 53% and 33% vs. 27%, for KITTI and ECP2.5D respectively, when not using LiDAR intensity information). Performance of both methods was significantly lower on ECP2.5D vs. KITTI, we attribute this to a larger prevalence of distant persons with fewer LiDAR points in ECP2.5D.

Domain transfer experiments indicated the two datasets have quite different biases, in the sense that training on one and testing to the other leads to significantly degraded performance (upwards of AP_{3D} of 6.8%). Further research is needed on cross-domain adaptation.

ACKNOWLEDGMENT

This work received support from the Dutch Science Foundation NWO-TTW within the SafeVRU project (nr. 14667).

REFERENCES

[1] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE Trans. on PAMI*, vol. 34, no. 4, pp. 743–761, 2011.

[2] X. Li, F. Flohr, Y. Yang, H. Xiong, M. Braun, S. Pan, K. Li, and D. M. Gavrila, "A new benchmark for vision-based cyclist detection," in *Proc. of the IEEE IV*, 2016, pp. 1028–1033.

[3] M. Braun, S. Krebs, F. B. Flohr, and D. M. Gavrila, "EuroCity Persons: A novel benchmark for person detection in traffic scenes," *IEEE Trans. on PAMI*, vol. 41, no. 8, pp. 1844–1861, Aug 2019.

[4] C. G. Keller, T. Dang, H. Fritz, A. Joos, C. Rabe, and D. M. Gavrila, "Active pedestrian safety by automatic braking and evasive steering," *IEEE Trans. on ITS*, vol. 12, no. 4, pp. 1292–1304, 2011.

[5] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. of the IEEE CVPR*, 2012.

[6] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast encoders for object detection from point clouds," in *Proc. of the IEEE CVPR*, 2019, pp. 12 697–12 705.

[7] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3D Proposal Generation and Object Detection from View Aggregation," *IEEE IROS*, pp. 5750–5757, October 2018.

[8] M. Braun, S. Krebs, and D. M. Gavrila, "ECP2.5D - Person localization in traffic scenes," in *Proc. of the IEEE IV*, 2020.

[9] A. Naiden, V. Paunescu, G. Kim, B. Jeon, and M. Leordeanu, "Shift R-CNN: Deep monocular 3D object detection with closed-form geometric constraints," in *Proc. of the ICIP*, Sep. 2019, pp. 61–65.

[10] B. Yang, M. Liang, and R. Urtasun, "HDNET: Exploiting HD maps for 3D object detection," in *Proc. of the CoRL*, vol. 87, Oct 2018, pp. 146–155.

[11] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington, "LaserNet: An efficient probabilistic 3D object detector for autonomous driving," in *Proc. of the IEEE CVPR*, June 2019.

[12] Y. Zhou and O. Tuzel, "VoxelNet: End-to-end learning for point cloud based 3D object detection," in *Proc. of the IEEE CVPR*, June 2018.

[13] Z. Wang, H. Fu, L. Wang, L. Xiao, and B. Dai, "SCNet: Subdivision coding network for object detection based on 3D point cloud," *IEEE Access*, vol. 7, pp. 120 449–120 462, 2019.

[14] V. A. Sindagi, Y. Zhou, and O. Tuzel, "MVX-Net: Multimodal voxelnet for 3D object detection," in *IEEE ICRA*, May 2019, pp. 7276–7282.

[15] D. Xu, D. Anguelov, and A. Jain, "PointFusion: Deep Sensor Fusion for 3D Bounding Box Estimation," in *Proc. of the IEEE CVPR*, June 2018.

[16] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and Segmentation," in *Proc. of the IEEE CVPR*, July 2017.

[17] M. Roth, D. Jargot, and D. M. Gavrila, "Deep end-to-end 3D person detection from camera and lidar," in *Proc of the IEEE ITSC*. IEEE, 2019, pp. 521–527.

[18] A. Asvadi, L. Garrote, C. Premebida, P. Peixoto, and U. J. Nunes, "Multimodal vehicle detection: fusing 3D-LIDAR and color camera data," *Pattern Recognition Letters*, vol. 115, pp. 20–29, 2018.

[19] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum PointNets for 3D object detection from RGB-D data," in *Proc. of the IEEE CVPR*, June 2018, pp. 918–927.

[20] Z. Wang and K. Jia, "Frustum ConvNet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection," in *IEEE IROS*. IEEE, 2019.

[21] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia, "STD: Sparse-to-dense 3D object detector for point cloud," in *Proc. of the IEEE ICCV*, October 2019.

[22] S. Vora, A. H. Lang, B. Helou, and O. Beijbom, "PointPainting: Sequential fusion for 3D object detection," 2019.

[23] Waymo, "Waymo Open dataset: An autonomous driving dataset," 2019.

[24] H. Caesar *et al.*, "nuScenes: A multimodal dataset for autonomous driving," *arXiv preprint arXiv:1903.11027*, 2019.

[25] M.-F. Chang *et al.*, "Argoverse: 3D tracking and forecasting with rich maps," in *Proc. of the IEEE CVPR*, 2019.

[26] R. Kesten *et al.*, "Lyft level 5 AV dataset 2019," https://level5.lyft.com/dataset/, 2019.

[27] D. Zhou *et al.*, "IoU loss for 2D/3D object detection," in *International Conference on 3D Vision*. IEEE, 2019, pp. 85–94.

[28] A. Simonelli, S. R. R. Bulò, L. Porzi, M. López-Antequera, and P. Kontschieder, "Disentangling monocular 3D object detection," *arXiv preprint arXiv:1905.12365*, 2019.

[29] C. Rist, M. Enzweiler, and D. M. Gavrila, "Cross-sensor deep domain adaptation for LiDAR detection and segmentation," in *Proc. of the IEEE IV*, 2019.

A.2. Article to be submitted to the IEEE Transactions on Intelligent Vehicles

This section contains the article which will be submitted to the IEEE Transactions on Intelligent Vehicles.

IEEE TRANSACTIONS ON INTELLIGENT VEHICLES, VOL. X, NO. X, MONTH YEAR 1

An Experimental Study on 3D Person Localization in Traffic Scenes

Joram R. van der Sluis*,1, Ewoud A.I. Pool*,1 and Dariu M. Gavrila[1]

***Abstract*—This paper presents an experimental study on 3D person localization (*i.e.*, pedestrians, cyclists) in traffic scenes, using monocular vision and Light Detection And Ranging (LiDAR) data. We first analyze the detection performance of two top-ranking methods (PointPillars and AVOD) on the KITTI benchmark, with respect to varying Intersection over Union (IoU) settings and the underlying parameters of 3D bounding box location, extent and orientation. Given that the KITTI dataset contains relatively few 3D person instances, we furthermore conduct experiments analyzing how the detection performance varies over distance, number of LiDAR points, occlusion, intensity and include results on the new EuroCity Persons 2.5D (ECP2.5D) dataset, which is one order of magnitude larger. We perform domain transfer experiments between the KITTI and ECP2.5D datasets, to examine how these datasets generalize with respect to each other.**

***Index Terms*—**

I. Introduction

ACCORDING to a recent report of the World Health Organisation, about half of the 1.3 million people killed yearly in traffic worldwide involve Vulnerable Road Users (VRUs), *i.e.*, pedestrians, cyclists, and other riders. For much of the past two decades, vision was the dominant sensor modality for intelligent vehicles to detect VRU. Strong progress has been made on 2D image-based VRU detection facilitated by novel (deep learning) methods, faster processors, and more data (including benchmarks, *e.g.* [1], [2], [3]). 3D localization from 2D detections can subsequently be achieved by back-projection, disparity computation [4], and/or association with radar targets. Vision-based VRU detection is meanwhile incorporated in active safety systems of various premium vehicles on the market.

Still, current active VRU safety systems are deployed in the context of driver assistance. With the advent of fully self-driving vehicles, performance needs to be significantly upped, as a driver is no longer available as a back-up. Additionally, the shift in responsibility from driver to autonomous vehicle will result in a shift from aiming to minimize false positives to aiming to minimize false negatives. A promising sensor to achieve this higher performance for self-driving vehicles is the LiDAR sensor, stemming from its capabilities to directly and accurately measure distances and to deal with low-light environments. KITTI [5] meanwhile offers a 3D object detection benchmark, including one for pedestrians. The latter leader-board currently lists a 3D Average Precision (AP_{3D}) of 54 % and around 40 % for the *easy* and *all* targets, respectively. In contrast, the state-of-the-art in 2D object detection attains an Average Precision (AP) of 75 % overall, leaving the question why LiDAR is not performing as expected.

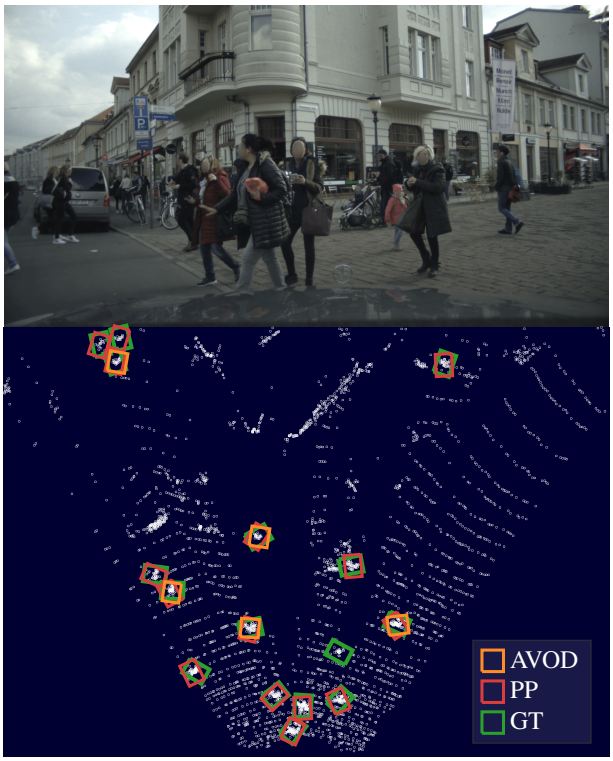


Fig. 1. An example of the predicted bounding boxes of PointPillars [6] (PP) and AVOD [7] on a scene from the EuroCity Persons 2.5D [8], along with the annotated ground truth (GT).

This paper presents an experimental study on 3D person localization (*i.e.*, pedestrians, cyclists) in traffic scenes, using monocular vision and LiDAR data. We consider two 3D object detection methods, PointPillars [6] and AVOD [7], which are among the top performers on the KITTI benchmark, see fig. 1, for which we investigate their usability in self-driving vehicles. We investigate the effect of the varying IoU setting on detection performance and quantify the various errors in terms of 3D bounding box location, extent, and orientation. We furthermore analyze how the detection performance varies over distance, the number of LiDAR points on the target, occlusion level, and the use of intensity. In the last years, many datasets much larger than KITTI have been published, and so we also perform experiments on a large subset of the new EuroCity Persons 2.5D (ECP2.5D) dataset [8]. Apart from being one order of magnitude larger than KITTI, ECP2.5D has advantages in terms of diversity (*e.g.* geographical coverage,

∗) Authors contributed equally
1) Intelligent Vehicles group, TU Delft, The Netherlands

time of day/season, weather conditions) and by being devoid of privacy-driven image blurring. This makes ECP2.5D also attractive when compared to other recent dataset additions, see table II. Finally, we perform domain transfer experiments between KITTI and ECP2.5D, to examine how these datasets relate to each other.

II. Related work

We focus our discussion on previous 3D object detection methods that use neural network architectures, as they are the current best performers in the various benchmarks. One way to categorize these detectors is by sensor modality, *i.e.*, either a single modality or a fusion of multiple modalities. This related work discuss the existing networks in detail using this categorization. The commonly used sensors used are (monocular) camera and LiDAR. However, the RGB-only methods (*e.g.* Shift R-CNN [9]) are generally outperformed by methods that instead use LiDAR information, and so we focus on those when reviewing single-sensor modality networks..

Another way of categorizing 3D object detection methods is by the number of stages used by the network. Two-stage approaches utilize a Region Proposal Network (RPN) to generate bounding boxes which are individually evaluated (*e.g.* STD [10] and Part-A^2 net [11]). Single-state approaches instead evaluate predetermined bounding boxes (*e.g.* PointPainting [12] and 3DSSD [13]), also called anchor boxes.

A. Single-sensor modality networks

LiDAR-only networks map the point cloud to either a 2D or a 3D representation Arnold *et al.* [14] compare different 2D and 3D LiDAR representations. They state that if the LiDAR point cloud is mapped to 2D, it suffers from a loss of information compared to 3D methods. A benefit of mapping to 2D is that conventional CNN architectures used for image detection can be used as a basis to obtain the 3D bounding boxes. Examples of 2D representations are Birds Eye View (used by *e.g.* HDNet [15]) and Range View (*e.g.* LaserNet [16]).

One group of methods that exploits the 3D information in a point cloud do so by dividing the data into voxels, and then afterwards apply 3D convolutions (*e.g.* Voxelnet [17] , TANet [18] and PV-RCNN [19]). The downside of applying 3D convolutions is the reduced efficiency when processing empty voxels and the generally higher computational costs for 3D convolutions. PointPillars [6] addresses this by creating "pillars". These pillars are a discretization of the point cloud into an equally spaced grid of pillars in the x-y plane. PointPillars additionally exploits the sparseness of a LiDAR scene by limiting both the number of pillars (P) in each sample and the number of LiDAR points per pillar (N). If a sample contains more than P non-empty pillars or a pillar contains more than N points, a random subset of respectively the pillars or points is taken in order to match P or N. Other 3D representations are, Stixels (*e.g.* SCNet [20]) or Graph representations (*e.g.* Point-GNN [21]).

An alternative to dividing a point cloud into voxels is PointNet [22]. PointNet takes a fixed number of points as input, which are transformed by an input and feature transformation. Global features are extracted using max-pooling which results in an output score per class. PointNet can also be used for segmentation, in this case the features of each point are concatenated with the global features to output a score per point. Although the results seem promising for object classification, this method alone is not suitable to use for object detection since it requires one single object per point cloud. PointNet++ [23] and Frustum PointNet [24] both address this shortcoming. PointNet++ has a hierarchical network structure that applies PointNet recursively on parts of the point cloud. This results in the ability to capture local features. Frustum PointNet exploits PointNets for 3D object detection, an image detector is used to select regions of the point cloud, the subsets of the point cloud are passed into a PointNet for classification and prediction of 3D bounding boxes.

TABLE I
Comparison of AVOD and PointPillars.

	AVOD	PointPillars
Modality	LiDAR + image	LiDAR
Stages	Two-stage	Single-stage
Bounding box regression	Four corners, heights, orientation	3D center point, length, width, height, orientation

B. Multi-sensor modality networks

Multi-sensor modality networks, or fusion networks, use both camera and LiDAR. Here, all the previously mentioned LiDAR mappings can be used to fuse with the camera data. How they are fused exactly falls into three categories.

The first category is early fusion, where the modalities are concatenated before being passed into a neural network. An example of early fusion is MVX-Net PointFusion [25] where the point cloud is projected onto an RGB-image and then concatenated.

Secondly, deep fusion networks fuse the modalities after they have already been processed by a part of the network, for example, PointFusion [30]. Here, the features from a PointNet [22] and a ResNet-50 are concatenated. With deep fusion, it is also possible to fuse the various modalities at multiple stages, as is done with AVOD [7]. Within such a deep fusion network, the performance is dependent on the feature encoder used [31]. Because the different modalities are already processed by their own part of the network, the data is transferred info features. When fusing these features, one decision can be made based on data from multiple modalities.

Thirdly, late fusion takes the output of two or more independent networks and fuses the class probabilities [32]. Since late fusion takes the output of multiple independent models, it has high flexibility because the models can be different for every data modality, trained independently, and more sensors can be added if they become available. The downside of late fusion is that only the outcome of the networks is fused, the internal features which are used by the network to make the decision are not shared with the other networks.

TABLE II
OVERVIEW OF TRAFFIC-RELATED 3D PERSONS DATASETS. "UNKNOWN" DENOTES THAT THE INFORMATION COULD NOT BE DETERMINED. THE NUMBER BETWEEN BRACKETS SPECIFIES THE NUMBER OF CAMERAS/LIDAR SENSORS WITH THAT RESOLUTION/NUMBER OF PLANES. [*] THE LYFT DATASET USES 2 TYPES OF VEHICLES. THESE HAVE A RESOLUTION OF 1224 × 1024 (6) + 2048 × 864 (1) AND 1920 × 1080 (7). THE NUMBER OF LIDAR PLANES ARE 40 (3) AND 40 (2) + 64 (1).

Dataset	**Waymo** [26]	**nuScenes** [27]	**Argoverse** [28]	**Lyft** [29]	**KITTI** [5]	**ECP2.5D** [8]
# Countries	1	2	1	1	1	12
# Cities	2	2	2	1	1	30
# Frames	800K	34K	350K	55K	15K	46K
Image resolution	1920 × 1280 (3) + 1920 × 1040 (2)	1600 × 900 (6)	1920 × 1200 (7) + 2056 × 2464 (2)	* (7)	1240 × 376 (1)	1920 × 1024 (1)
# LiDAR planes	unknown (5)	32 (1)	32 (2)	* (3)	64 (1)	64 (1)
# Peds	2.8M	222K	132K	25K	9.4K	123K
# Riders	67K	24K	11K	22K	3.3K	13K
# Seasons	unknown	unknown	1	1	1	4
Weather	dry, rain	dry, rain	dry	dry	dry	dry, rain
Time of day	day, night	day, night	day, night	unknown	day	day, night
Unblurred	✗	✗	✗	✗	✓	✓

A specific form of late fusion is sequential fusion, which processes the sensor modalities in sequence. For example, Frustum PointNets [24] and Frustrum Convnet [33] use a 2D image detector to select frustums in a point cloud, which is then processed separately. The benefit of sequential fusion is that both networks can be trained independently. The downside is the strong dependence of the different parts upon each other, if one part fails the other will not be able to recover the loss of information.

Feng *et al*. [34] discuss the benefits and downsides of different fusion methods. They state that if the modalities are fused before entering the network, the network can fully exploit all the information from the modalities if the data is properly aligned. Another benefit is the low computational cost because the network does not need separate networks that process these modalities, but instead shares a part of the computational load. A benefit of late fusion is the added flexibility of selecting a different, more optimal architecture for each sensor modality. Additionally, as each model can be trained separately, there is no need for one dataset that combines all required sensor modalities [35].

C. PointPillars and AVOD

PointPillars [6] and AVOD [7], two of the best performing LiDAR and fusion networks, respectively, with code available at the time of writing, will be used later in the experiments and are therefore explained in detail. PointPillars first converts the point cloud into a pillar representation. Then a simplified version of PointNet [22] is applied to each pillar, resulting in a sparse pseudo image. In the second step, the sparse pseudo image is passed onto a backbone which extracts top-down features at different upsampling factors by 2D convolutions. The final step includes a Single Shot Detector [36] setup which outputs the detections.

AVOD exploits the information from both images and LiDAR point clouds. AVOD uses the same method as MV3D [37] to generate a Birds Eye View (BEV) representation of the point cloud. The point cloud is divided into 3D cells by slicing the height dimension into 5 equal parts between 0 and 2.5 meter and divide the point cloud into a 2D grid with a resolution of 0.1 meter in the horizontal plane. Each of these 3D cells is encoded with a height feature that corresponds to the maximum height of the LiDAR point inside this cell, together with a feature that represents the number of points inside a cell. This results in a 6 channel BEV. The difference between AVOD and MV3D is that AVOD neglects intensity.

The image and BEV representation are both passed into a feature extractor that consists of an encoder and a decoder. The encoder is an adapted version of a VGG-16 [38] and the decoder is similar to the Feature Pyramid Network [39]. The output of the feature extractor is used by both the RPN and the second stage of the detection network. Based on the proposals of the RPN the features from the image and point cloud are fused, and subsequently used for detection. Table I summarizes the differences between PointPillars [6] and AVOD [7].

D. Datasets

In terms of existing datasets, one of the first 3D object detection benchmarks was an extension to KITTI [5], released in 2017, which contains around 9.4K pedestrians (of which half in the publicly available training set). Since then, KITTI has become the de facto standard for 3D object detection. However, because of the relatively small dataset size, performances can differ a lot on the validation and test set. More recent dataset additions to KITTI are significantly larger and more diverse, see table II. Waymo, nuScenes, Argoverse, and Lyft also contain sequences rather than independent scenes and have annotated all objects over time. Additionally, they have annotated pedestrians in 360 degrees around the vehicle, and provide enough camera images to see all around the vehicle. ECP2.5D [8], while lacking annotations over time, is unique due to the diversity in its urban scenes recorded in different European cities in various weather conditions. Another observation is the difference between the location of pedestrians in European cities compared to the United States of America (USA). In the USA, annotated pedestrians are most commonly located on either the sidewalk or a pedestrian crossing at an intersection, the pedestrians in Europe are located at a variety of locations (*e.g*. on the road).

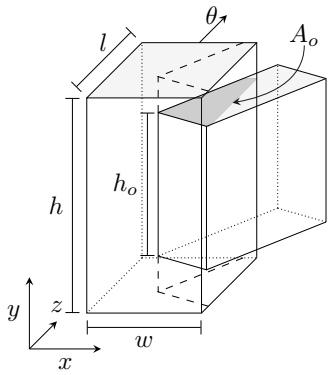


Fig. 2. A visualization of the parameters relevant for computing the IoU of a ground truth and predicted bounding box. The darker shaded area indicates the overlap area A_o. In this figure, the overlapping height h_o is equal to the height of the smaller bounding box.

E. Contributions

This paper presents an experimental study on monocular and LiDAR-based 3D person detection. Its specific contributions are:

- A performance analysis of two state-of-the-art methods (PointPillars and AVOD) on KITTI, with respect to varying IoU and the underlying parameters of 3D bounding box location, extent, and orientation.
- An analysis of the effect of distance, number of LiDAR points, occlusion, and LiDAR intensity on the performance of these two methods on KITTI and ECP2.5D.
- Results from domain transfer experiments between KITTI and ECP2.5D.

III. Experimental Methodology

The goal of 3D person detection in the KITTI benchmark is to detect the 3D bounding boxes of VRUs in the scene. These bounding boxes have seven degrees of freedom (see fig. 2). The 3D position is given in a coordinate system with respect to the ego-vehicle, where x is the position of the bounding box center lateral to the vehicle, z is the position longitudinal to the vehicle (*i.e.*, depth), and y determines the altitude of the bounding box center. The bounding box dimensions are specified by a width w, length l, and height h. Finally, each bounding box has a yaw rotation θ. The top and bottom face of the bounding box are assumed to be parallel to the $y = 0$ plane. The predicted bounding boxes will also have a detection score d related to them.

A. Intersection over Union (IoU)

To evaluate the performance of an object detector, each predicted bounding box is counted either as a true positive or false positive. In 3D (as well as 2D) object detection, the method to assess if a proposed bounding box is a true- or false-positive is based on IoU.

It is defined as the intersection (or overlap) of a 3D bounding box prediction (B_p) and ground truth (B_{gt}) divided by the union of the prediction and ground truth. When both bounding boxes only have a yaw rotation, this can be written as [40]:

$$\text{IoU} = \frac{B_p \cap B_{gt}}{B_p \cup B_{gt}} = \frac{A_o \times h_o}{V_{gt} + V_p - A_o \times h_o} \tag{1}$$

Where V_p and V_{gt} are the volumes of the predicted and ground truth bounding box. The overlap of volumes can be computed from the overlapping top-view area A_o and the overlapping height (h_o), see fig. 2. In KITTI, a predicted bounding box is seen as a true positive if it has an IoU of more than 0.5. Only one predicted bounding box can be marked as a true positive for any ground truth bounding box.

B. Performance metrics

After the true positives have been determined, it is possible to compute the two metrics as defined in the KITTI benchmark for 3D object detection: 3D Average Precision (AP_{3D}) and Average Orientation Similarity (AOS) [5].

The AP_{3D} averages the maximum attained precision s with at least a recall r for a fixed range of recall values [41]:

$$\text{A}P_{3D} = \frac{1}{40} \sum_{r \in \{\frac{1}{40}, \frac{2}{40}, \ldots, 1\}} \max_{\tilde{r}:\tilde{r} \geq r} s(\tilde{r}) \tag{2}$$

Recall is defined as the number of true positives divided by the number of ground truth annotations (*i.e.*, true positives and false negatives), and precision is defined as the number of true positives divided by the number of detections (*i.e.*, true positives and false positives) As precision and recall both depend on the number of true positives, the AP_{3D} strongly depends on the IoU threshold.

Where the AP_{3D} verifies whether the bounding boxes are in the correct place, the AOS additionally verifies the correctness of their orientations:

$$AOS = \frac{1}{40} \sum_{r \in \{\frac{1}{40}, \frac{2}{40}, \ldots, 1\}} \max_{\tilde{r}:\tilde{r} \geq r} \tilde{s}(\tilde{r}) \tag{3}$$

$$\tilde{s}(r) = \frac{1}{|\mathcal{D}(r)|} \sum_{i \in \mathcal{D}(r)} \frac{1 + \cos \Delta_\theta^{(i)}}{2} \delta_i \tag{4}$$

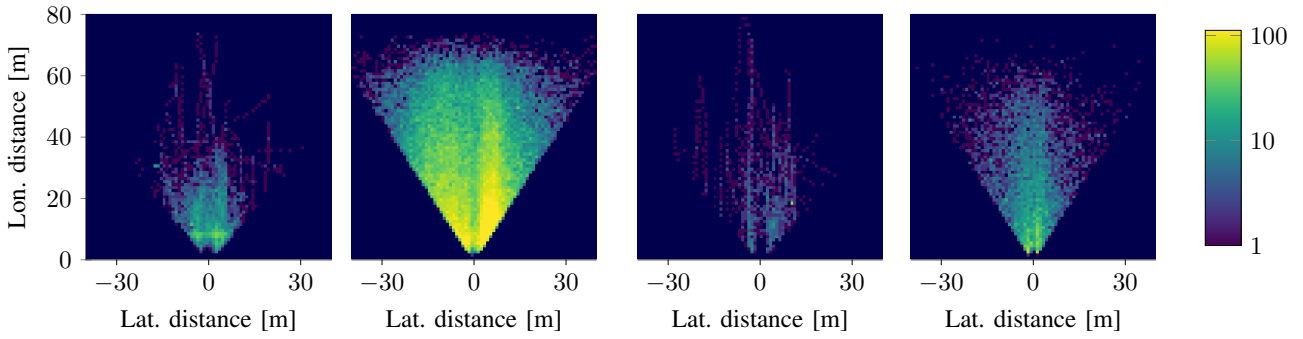
Where $\mathcal{D}(r)$ denotes the set of all objects at a specific recall rate r and $\Delta_\theta^{(i)}$ the difference between the estimated and the real orientation. The indicator δ_i is one if the predicted bounding box is seen as a true positive, and zero otherwise. If every true positive predicted bounding box has an orientation error of 0, eq. (4) reduces to the precision at that recall rate.

IV. Experiments

Experiments were performed with the codebase of the authors of AVOD[1] and the codebase recommended by the authors of PointPillars[2] as is, using the best performing network configuration as reported in their papers. Thus for AVOD, we use AVOD-FPN, and for PointPillars, we use a spatial resolution of 0.16×0.16 m^2. For the following experiments,

[1] https://github.com/kujason/avod

[2] https://github.com/traveller59/second.pytorch



(a) KITTI pedestrians (b) ECP2.5D pedestrians (c) KITTI cyclists (d) ECP2.5D cyclists

Fig. 3. The overall distribution over location of pedestrians and cyclists in both the KITTI and ECP2.5D dataset as a log plot. In all figures, the egovehicle is positioned at $(0, 0)$, looking upwards. Each pixel in the image corresponds to a 1×1 square meter area. The darkest blue region indicates areas with zero pedestrians.

we have trained each network 10 times. We then use the network closest to the average AP_{3D} of all 10 networks to report our results in sections IV-B to IV-D. Section IV-E will analyze the variation between the 10 trained networks in more detail.

A. Datasets overview

Each dataset contains data from two sensors: RGB images from a front-facing camera, and point cloud data that includes the intensity of the reflected beam. Both datasets use the Velodyne HDL-64E (LiDAR) sensor. The ground truth contains annotations of pedestrians and cyclists. KITTI also provides annotations for vehicles, but these are ignored in this study.

Figure 3 shows the distribution of the VRUs locations relative to the vehicle, for the publicly available part of both KITTI and ECP2.5D. The bulk of the detections in the KITTI dataset lies within 30 m distance of the ego-vehicle. Both datasets have a bias towards VRUs being on the right side of the ego-vehicle. Figure 4 shows the relation between LiDAR points reflected off a VRU and distance for both datasets. The number of LiDAR points is found by counting all LiDAR points within the annotated 3D bounding box. The number of points scales inversely with the distance on average. The number of LiDAR points has a large spread for any given distance however, caused by partial occlusions as well as the orientation of the VRU. As ECP2.5D has more VRUs at a distance, the dataset contains relatively more annotations with few LiDAR points.

The KITTI dataset is not divided into an official train and validation split. However, both the AVOD and PointPillar codebase specify an identical train/validation split. The data is divided equally over the train and validation splits. Each split is also divided into an "easy", "moderate" and "hard" part, as defined by KITTI (see table III). These parts are subsets of each other, where the easy part is a subset of the moderate part and the moderate is a subset of the hard part.

The ECP2.5D dataset has a larger number of annotations for the 3D position and orientation but lacks width, length, and height annotation. We will use the median bounding box dimensions of the train split of the KITTI dataset so both networks can still regress a full bounding box. The data is split into a "Day" and "Night" subset. If not specifically mentioned, this paper uses the "Day" subset. Additionally, some of the underlying EuroCity Persons (ECP) dataset annotations misses an orientation label. these are set to "Don't Care". The exact numbers can be found in table IV.

The test set ground truth annotations of both datasets are not made public, so all evaluations done in the following sections of this paper are done using the validation splits of either dataset. For KITTI, this means using the unofficial validation split described in this section.

TABLE III
DIFFICULTY LEVELS AS DEFINED BY KITTI [5]. TRUNCATION REFERS TO THE PERCENTAGE OF THE BOUNDING BOX THAT IS OUTSIDE THE IMAGE.

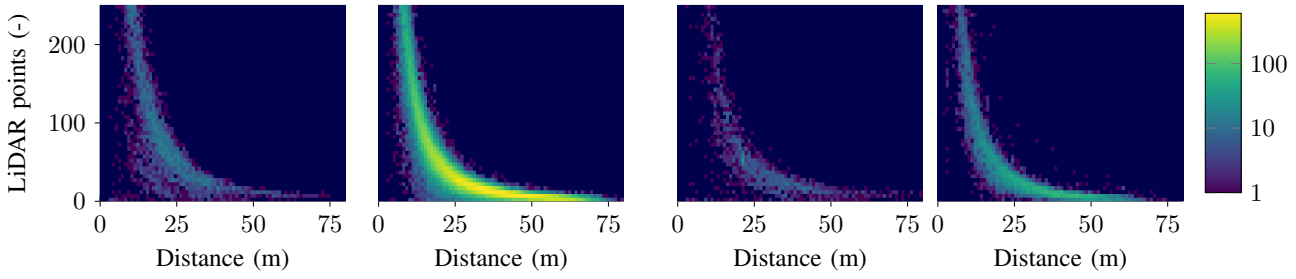
Difficulty	Min. bounding box height	Max. occlusion level	Max. truncation
Easy	40 Pixels	Fully visible	15 %
Moderate	25 Pixels	Partly occluded	30 %
Hard	25 Pixels	Difficult to see	50 %

TABLE IV
THE NUMBER OF PEDESTRIANS AND CYCLISTS THE SPLITS OF EACH DATASET. VALUES INDICATE THE NUMBER USED IN THIS PAPER/NUMBER WITHOUT ORIENTATION.

Dataset	Pedestrians	Cyclists
Train split :		
ECP2.5D Day	62.3K/318	7.3K/111
ECP2.5D-Night	13.4K/14	0.7K/1
KITTI	2.2K/0	0.7K/0
Validation split :		
ECP2.5D Day	12.6K/68	1.3K/33
ECP2.5D-Night	2.6K/2	0.1K/1
KITTI	2.3K/0	0.9K/0

B. Effect of IoU on performance and error analysis

Performance with lower IoU constraints: Table V shows the performance of PointPillars and AVOD on KITTI for the cyclist and the pedestrian classes. As mentioned, this shows the performance of the network closest to the average average



(a) KITTI Pedestrians (b) ECP2.5D pedestrians (c) KITTI cyclists (d) ECP2.5D cyclists

Fig. 4. The number of VRUs at a given distance and with a given number of LiDAR points reflecting off them, for both the KITTI and ECP2.5D dataset as a log plot. The darkest blue region indicates areas with zero pedestrians/cyclists.

TABLE V
AOS AND AP_{3D} PERFORMANCE OF POINTPILLARS (PP) AND AVOD, TRAINED ON KITTI AND EVALUATED ON THE MODERATE PART OF THE KITTI VALIDATION SPLIT.

	Pedestrian		Cyclist	
IoU	AP_{3D}	AOS	AP_{3D}	AOS
PP				
0.5	51.3	23.0	62.4	3.1
0.4	70.4	31.5	65.7	3.9
0.3	74.3	33.1	66.8	4.0
0.2	75.0	33.1	67.4	4.0
0.1	75.2	33.4	67.4	4.0
AVOD				
0.5	36.4	28.3	27.3	27.2
0.4	44.4	34.0	29.6	29.4
0.3	46.8	35.7	29.6	29.5
0.2	47.0	35.9	29.8	29.6
0.1	47.1	36.0	29.9	29.7

AP_{3D} of the 10 trained networks of each type. PointPillars has a higher AP_{3D} than AVOD, even though their scores on the moderate test split on the online KITTI benchmark differ less than one percent. However, the results we find for AVOD are comparable to those found on the validation split in the comparison study of [31]. For both networks, lowering the IoU threshold increases the AP_{3D} by a large margin. For example, the AP_{3D} of PointPillars on pedestrians increases from 51.3 to 75.2.

This is further visualized in fig. 5, which shows a histogram of the IoU found for all true positive detections at an IoU threshold of 0.1. This histogram shows that for pedestrians 20 % of the detections of PointPillars and 11 % of the detections of AVOD had an IoU between 0.4 and 0.5, just outside the normal IoU threshold. A similar effect is seen for cyclists, albeit less strongly.

The upper bound of the AOS is the AP_{3D}, as mentioned in section III-B. Table V shows that even though the general detection accuracy of AVOD is lower than PointPillars, its AOS is very close to that upper bound, especially for cyclists. The AOS of PointPillars is far worse than the AOS noted on the online KITTI benchmark. A closer inspection of the distribution of the orientation error (fig. 6) shows that for PointPillars, the orientation error peaks around 0 or 180 degrees. In the paper of PointPillars, the authors state that the

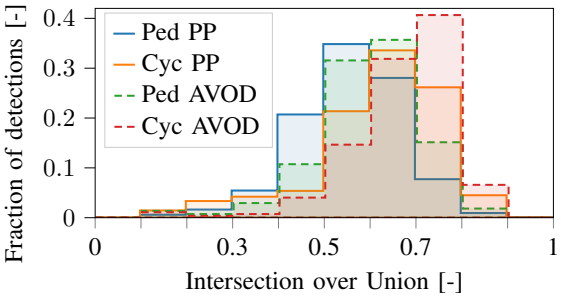


Fig. 5. PointPillars and AVOD trained on KITTI: a histogram of what fraction of true positive detections had what IoU (IoU threshold of 0.1).

orientation loss used cannot distinguish between flipped boxes, for which they use an additional binary classification loss. This seems to indicate that while the original overall orientation loss works as expected, there might be an implementation issue with the binary classification loss in the codebase of SECOND. As for AVOD, almost all of the orientation estimates indeed have an error close to 0 degrees as was expected from their AOS.

Error analysis of bounding box estimation: Figure 7 shows the error made in the position and size of the predicted bounding boxes on pedestrians by PointPillars. The smallest errors are made on the x and the z estimation: the lateral and longitudinal position. The largest error is made on the width and length estimation. These also have the largest increase in error at lower IoU thresholds. These depend on the stride of a pedestrian, as well as the location of their arms, which can be difficult to infer.

The relatively small error in x and the z position (essentially a top-down position estimate) is visualized in fig. 8. It shows the x and z position error made for the true positive detections for the original IoU threshold, as well as the detections between an IoU of 0.1 and 0.5. A lot of the detections with an IoU below 0.5 are still accurate at estimating the position. For an IoU threshold of 0.5, nearly all of the true positive detections (1423 out of 1445) lie within a radius of 15 cm. When looking at the detections found with an IoU threshold of 0.1, a total of 1850 detections lie within a radius of 15 cm. In other words, using a radius of 15 cm as a metric to determine true positives instead of an IoU of at least 0.5

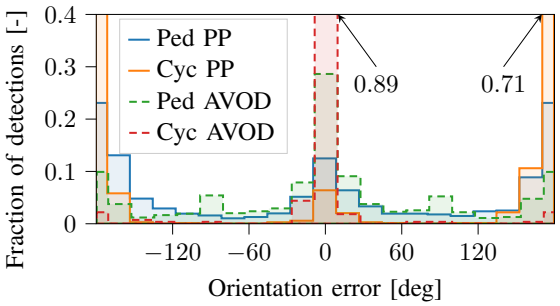


Fig. 6. PointPillars and AVOD trained on KITTI: A histogram of the orientation error. The arrows indicate the fraction of detections of the two bars outside of the y axis range. Most orientation errors lie either between -40 and 40 degrees, or between 140 and -140 degrees.

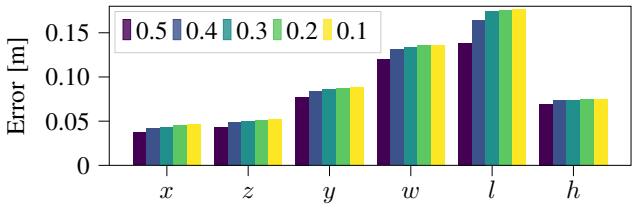


Fig. 7. PointPillars trained on KITTI: the average error between the prediction and the ground truth for the pedestrian detections on x, z, y, w, l and h, at different IoUs thresholds. The largest error is made on the altitude estimation, together with the bounding box width and length.

shows an increase in detections of 28 %. The same data is put more succinctly in fig. 9, with cyclists added as well. It shows the number of true positive detections that fall below a specific Euclidean position error. Cyclists see a smaller benefit: because their annotated bounding boxes are usually larger it is possible to make a larger position error without affecting the IoU as much.

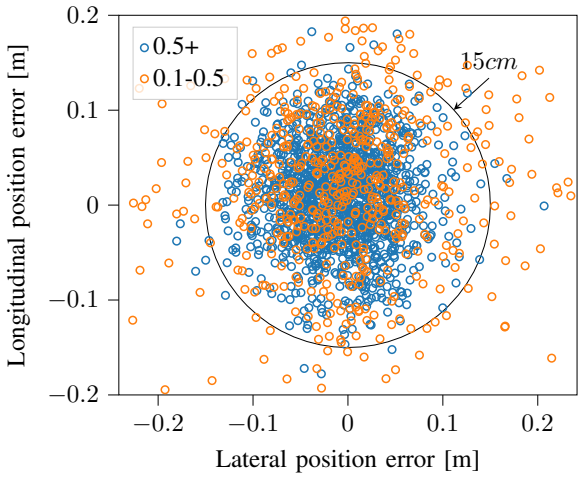


Fig. 8. PointPillars trained on KITTI: The localization error made by true positive detections of **pedestrians**, from a bird's eye viewpoint. Of the true positive detections with an IoU of over 0.5, 1423 out of 1445 detections lie within a radius of 15 cm. Of the true positive detections with an IoU between 0.1 and 0.5, 427 out of 573 lie within that radius.

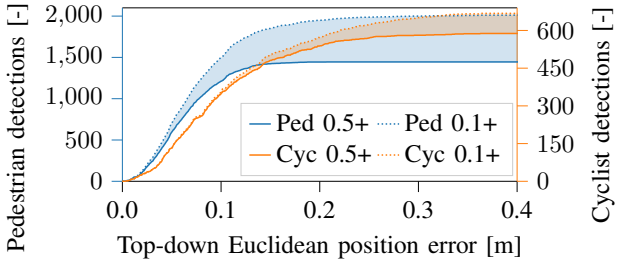


Fig. 9. PointPillars trained on KITTI: given a certain Euclidean position error threshold, how many detections would be inside. The solid line shows what Euclidean error is made by detections using the current default IoU threshold of 0.5. The dotted line shows the number of detections within a given Euclidean error for an IoU threshold of 0.1. The shaded area then shows the number of detections added.

TABLE VI
AP_{3D} PERFORMANCE OF POINTPILLARS (PP) AND AVOD FOR TWO IOU THRESHOLDS, EVALUATED ON THE MODERATE PART OF THE KITTI VALIDATION SPLIT. THE NETWORKS WERE TRAINED ON THE ORIGINAL KITTI GROUND TRUTH (ORIG.) OR THE GROUND TRUTH WITH FIXED BOUNDING BOX DIMENSIONS (FIXED).

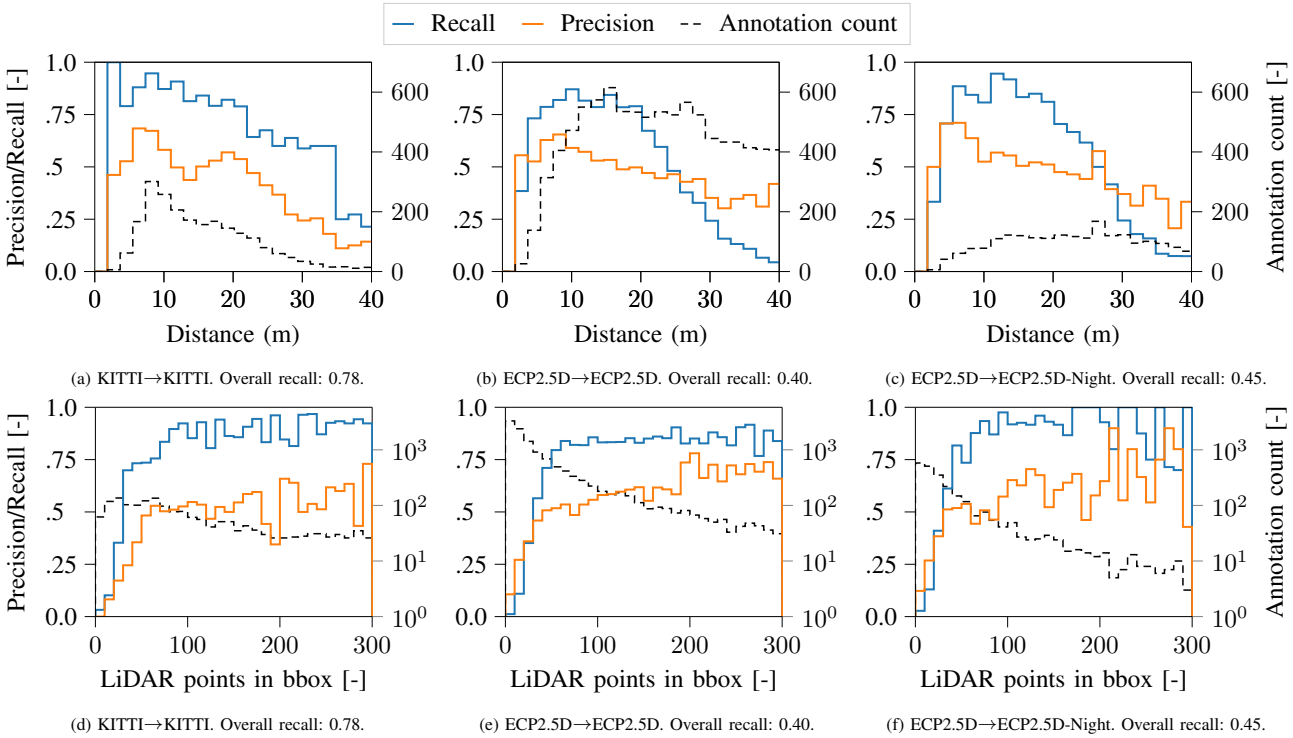
	Pedestrian		Cyclist	
IoU	Orig.	Fixed	Orig.	Fixed
PP				
0.5	51.3	54.6	62.4	62.6
0.1	75.2	73.3	67.4	68.1
AVOD				
0.5	36.4	46.0	27.3	35.5
0.1	47.1	59.6	29.9	38.8

Accuracy evaluation using fixed bounding boxes during training: The relatively large errors in width and length suggest that these two 3D object detectors are not able to properly estimate bounding box dimensions. To investigate the influence of the bounding boxes dimensions, we train on a version of the KITTI dataset train split where we fix the dimensions of each VRU. The dimensions are fixed to the median dimensions of their respective class. Table VI shows the AP_{3D} of those detectors evaluated on the original KITTI dataset validation split with the correct dimensions. In all cases except the PointPillars pedestrian case, the performance increases when the bounding box dimensions are disregarded during training.

C. Effect of distance and LiDAR points on performance

Focusing on PointPillars, figs. 10a to 10c show the precision and recall based on the distance for KITTI, ECP2.5D, and ECP2.5D-Night for an overall precision of 0.5. On both datasets, the performance starts to degrade around 20 m distance. However, because ECP2.5D has a larger proportion of distant targets (see dashed line in fig. 3) the performance reduction on distant targets affects the total AP_{3D} more.

Figures 10d to 10f show the precision and recall as a function of LiDAR points for KITTI and ECP2.5D, at the same overall precision of 0.5 as figs. 10a to 10c. On all datasets, both recall and precision decrease as the number LiDAR points reflected by the target goes down: targets with few LiDAR



(a) KITTI→KITTI. Overall recall: 0.78.
(b) ECP2.5D→ECP2.5D. Overall recall: 0.40.
(c) ECP2.5D→ECP2.5D-Night. Overall recall: 0.45.
(d) KITTI→KITTI. Overall recall: 0.78.
(e) ECP2.5D→ECP2.5D. Overall recall: 0.40.
(f) ECP2.5D→ECP2.5D-Night. Overall recall: 0.45.

Fig. 10. PointPillars: the precision and recall histogram, grouped by distance (figs. 10a to 10c) and by the number of LiDAR points on bounding box (figs. 10d to 10f). For recall, which depends on true positives, the GT bbox is used to compute the the distance/LiDAR point count. For precision, which depends on false positives, the detected bbox is used to compute the distance/LiDAR point count. The subcaption shows on which dataset the network is trained→evaluated. The operating point of each network is selected at an overall precision of 0.5. True positives are selected using an IoU of 0.1. The dashed line indicates the total number of ground truth annotations for each bin on a linear scale for the distance, and on a logarithmic scale for LiDAR points.

points are hard targets. Both ECP2.5D and ECP2.5D-Night have a larger proportion of pedestrians with a small number of points inside their bounding box than KITTI, ECP2.5D more so than ECP2.5D-Night. The network trained on ECP2.5D shows a similar overall trend on ECP2.5D and ECP2.5D-Night, except that the precision on ECP2.5D-Night starts to fluctuate on targets with more than 200 points in the detected bounding box. This is most likely due to the low sample count, as there are less than 10 ground truth annotations in this area, and even less true positives.

D. Effect of occlusions on performance

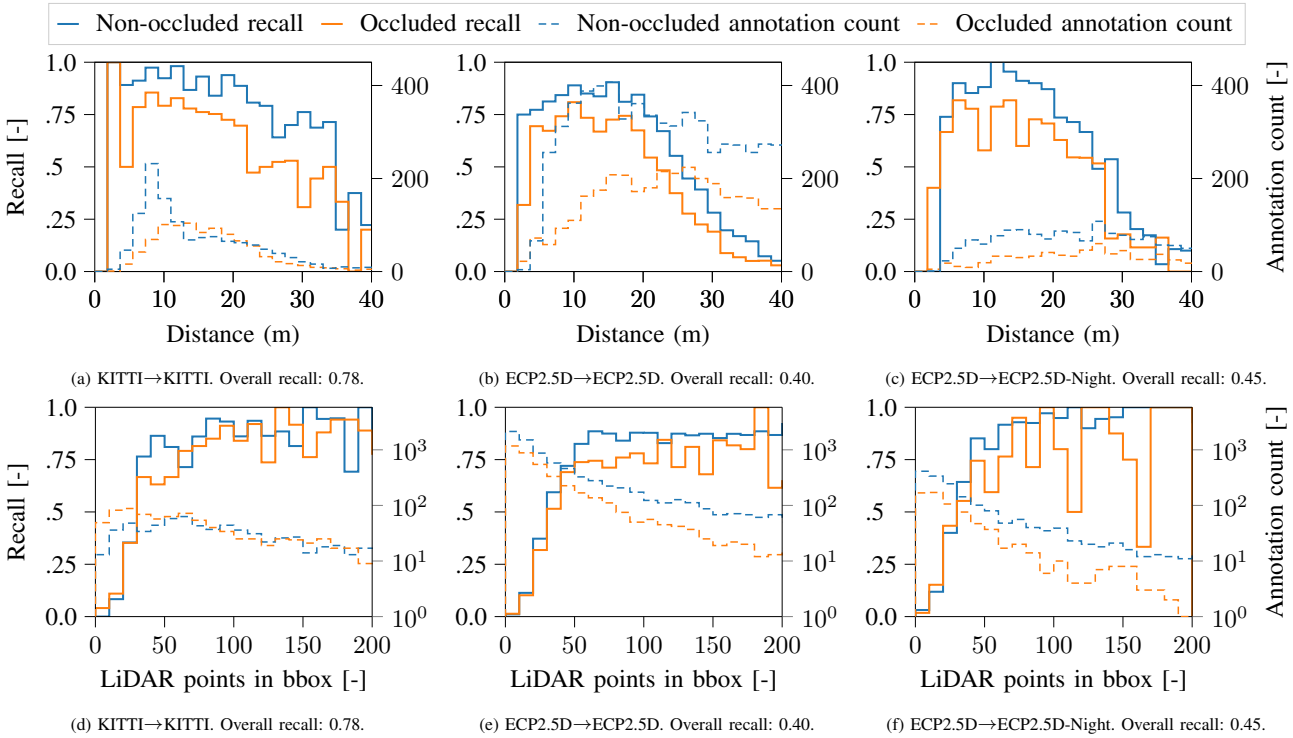
Figures 11a to 11c show the recall over distance separately for occluded and non-occluded pedestrians. This graph does not contain any precision, as precision is a function of false positives and it is ambiguous whether a false positive should belong to the occluded or non-occluded group . As expected, the performance is higher on non-occluded pedestrians. The number of occluded and non-occluded pedestrians in KITTI (fig. 11a) is almost equal after 12 meters. ECP2.5D (fig. 11b) and ECP2.5D-Night (fig. 11c), have a higher fraction of non-occluded pedestrians over the full distance range.

The recall of occluded and non-occluded pedestrians as a function of the amount of LiDAR points in the ground truth bounding box is shown in figs. 11d to 11f. In these figures, the gap between occluded and non-occluded recall is smaller than when comparing performance over distance (figs. 11a to 11c). At a high number of LiDAR points in the ground truth bounding box, the occluded recall starts to deviate from the non-occluded recall. However, this part of the graph has a small number of samples, less than 10, making it difficult whether this is due to the training, the network, or a statistical anomaly. Regardless, for all datasets, it seems that a better distinction for which pedestrians are difficult to detect is that amount of LiDAR points in the ground truth bounding box, rather than whether a pedestrian is occluded or not.

E. Cross-dataset evaluations

To see how well each dataset generalizes, we train each network on KITTI and ECP ten times, and evaluate those networks on all three datasets. Because the original PointPillars uses the intensity information of the points in the point cloud as well, we perform the experiment once *with* this intensity information present, and once *without*. AVOD does not use the intensity information, and therefore only needs to be trained once on each dataset. To ensure the datasets are compatible, we linearly rescale the LiDAR intensity KITTI values from $[0, 1]$ to $[255, 0]$ and vice versa. We also performed a linear scaling where we match the modes of the intensity distributions, but this resulted in a significantly worse performance, For example, one of the networks trained on ECP2.5D and evaluated on KITTI attains a AP_{3D} of 18.9 instead of 46.7.

Table VII shows the resulting average AP_{3D} of the ten networks, together with the standard deviation. For both KITTI



(a) KITTI→KITTI. Overall recall: 0.78.
(b) ECP2.5D→ECP2.5D. Overall recall: 0.40.
(c) ECP2.5D→ECP2.5D-Night. Overall recall: 0.45.
(d) KITTI→KITTI. Overall recall: 0.78.
(e) ECP2.5D→ECP2.5D. Overall recall: 0.40.
(f) ECP2.5D→ECP2.5D-Night. Overall recall: 0.45.

Fig. 11. PointPillars: the recall histogram for occluded and unoccluded pedestrians, grouped by distance (figs. 11a to 11c) and by the number of LiDAR points on bounding box (figs. 11d to 11f). For recall, which depends on true positives, the GT bbox is used to compute the distance of/LiDAR points count. Precision is not shown, as it is ambiguous whether false positives belong to the occluded or non-occluded category. The subcaption shows on which dataset the network is trained→evaluated. The operating point of each network is selected at an overall precision of 0.5. True positives are selected using an IoU of 0.1. The dashed line indicates the total number of ground truth annotations for each bin on a linear scale for the distance, and on a logarithmic scale for LiDAR points.

and ECP, PointPillars using LiDAR intensity data and training on that dataset attains the best performance. When not using LiDAR intensity data, PointPillars' performance slightly drops on KITTI, but still outperforms AVOD. AVOD also has a higher variance than PointPillars. The performance of both methods is significantly lower on ECP2.5D vs. KITTI.

When training on the one dataset and testing on the other, performances degrade significantly for both methods, both when moving from KITTI to ECP2.5D, and vice versa. The resulting performance degradation for PointPillars is less severe when the LiDAR intensity data is not used.

This can also be observed in the precision-recall curves. Figure 12 shows that both cross-evaluated networks perform better if they are trained and evaluated without intensity. Networks that are trained and evaluated on the same dataset still perform better with intensity. However, as it does not outperform PointPillars with intensity, we conclude that there is relevant information in the intensity.

As a final note, the highest performance on ECP2.5D-Night is higher than the highest performance on ECP. As fig. 10e and fig. 10f show, this is likely due to the relatively smaller number of pedestrians with a low number of LiDAR points inside their bounding box.

F. Qualitative analysis ECP2.5D

To perform a qualitative analysis of the type of errors made by PointPillars, 30 frames were randomly sampled from

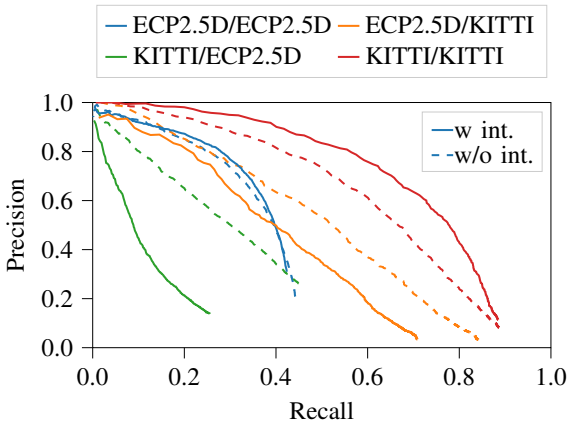


Fig. 12. PointPillars: the precision-recall curve for PointPillars trained on ECP2.5D and KITTI at an IoU threshold of 0.1. The legend shows on which dataset each line was trained/evaluated, respectively. The solid/dashed line show the networks trained with/without intensity, respectively.

the validation part of the ECP2.5D dataset. These frames contained a total of 58 false positives, 35 true positives, and 55 false negatives. An example of such a frame is shown in fig. 13.

Table VIII shows a categorization of different objects which are classified as a pedestrian and are not matched with a ground truth. The category poles contains traffic poles, traffic

TABLE VII
AP_{3D} PERFORMANCE OF POINTPILLARS (PP) AND AVOD FOR AN IOU OF 0.1 ON THE MODERATE VALIDATION SPLIT OF KITTI, ECP2.5D AND ECP2.5D-NIGHT. BOLD INDICATES HIGHEST PERFORMANCE IN THAT COLUMN. THE NETWORK ARE TRAINED 10 TIMES EACH, NEXT TO THE AP_{3D}, THE STANDARD DEVIATION IS SHOWN.

	AP_{3D}		
Trained network	KITTI	ECP2.5D	ECP2.5D-Night
with intensity :			
PP on ECP2.5D	49.4 ± 4.4	**33.0 ± 1.4**	37.4 ± 1.8
PP on KITTI	**75.2 ± 1.5**	9.4 ± 1.6	9.0 ± 1.8
w/o intensity :			
PP on ECP2.5D	54.1 ± 2.9	32.9 ± 1.4	**38.0 ± 1.4**
PP on KITTI	66.8 ± 2.3	23.4 ± 2.5	26.5 ± 2.9
AVOD on ECP2.5D	34.8 ± 8.3	27.5 ± 5.3	23.2 ± 7.6
AVOD on KITTI	47.6 ± 5.9	5.3 ± 2.2	2.8 ± 1.7

TABLE VIII
CATEGORIZATION OF THE FALSE POSITIVES MADE BY POINTPILLARS IN 30 FRAMES OF THE VALIDATION PART OF THE ECP2.5D DATASET.

Type of error	Occurrences
Pole	21
Not annotated	15
Bicycle / scooter	2 / 9
Fence / wall / door	3 / 1 / 1
Small objects	3
Too low IoU	2
Unknown	1

signs, trees, and large flag poles. The small object category contains a small advertisement board, pram, and trashcan. Not annotated refers to correctly classified objects which are not annotated because they are either occluded in the image(11) or just outside the image frame(4). There is one unknoqn object of which it was not possible to identify what kind of object it was based on the LiDAR point cloud and image. There is a large number of true positives that are caused by poles, therefore one could argue that the detector is not able to capture enough of the shape information to distinct a human from a tree/pole.

Out of the 55 false negatives, 47 objects had very few LiDAR points on them. Table IX shows the categorization of the reasons why the object was not detected. 31 false negatives have less than or equal to 10 LiDAR points in their 3D box and 47 false negatives have fewer or equal to 25 LiDAR points in their 3D box. The main reason for not detecting the pedestrians is the low number of LiDAR points on top of them. At large distances, the inclination of the road can result in pedestrians with no points on their upper body as visible in fig. 14h. Two false positives touch the true negatives with a too low IoU to be a true positive. There was one object which was near a wall and therefore missed. Of one object it was not clear why it was not detected by the network, the reason why this objects is missed is called unknown.

V. DISCUSSION

This paper presented an analysis of two existing 3D object detectors in the context of person detection for self-driving

TABLE IX
CATEGORIZATION OF THE FALSE NEGATIVES MADE BY POINTPILLARS IN 30 FRAMES OF THE VALIDATION PART OF THE ECP2.5D DATASET.

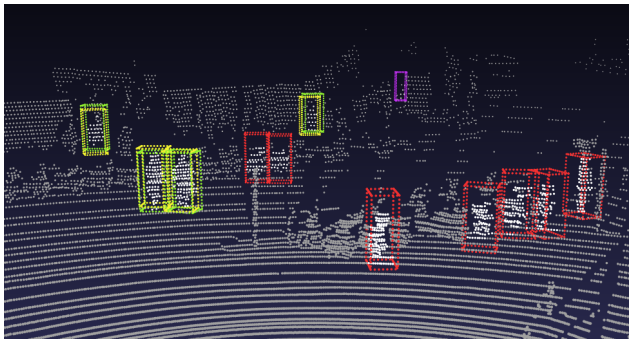
Type of error	Occurrences
Low number of LiDAR points	47
Occluded in LiDAR space	4
Too low IoU	2
Near wall	1
Unknown	1

vehicles in an urban setting. In this context, we believe the trade-off between safety and driving efficiency skews strongly towards safety: a false positive detection can result in a slower ride, whereas a false negative can result in injury or worse. For 3D person detection, the KITTI dataset is the de facto standard, and the two methods we examined have been tested on this dataset by the original authors as well, making it an obvious choice for our initial experiments.

Using this dataset, we found that the discrepancy between 2D and 3D object detection AP is reduced when we lower the IoU at which a detection is deemed a true positive (table V). More importantly, we showed that a large contributor to a low IoU is not the localization of the bounding box center, but the size estimation (fig. 7). Most of the true positives added by lowering the IoU threshold have a localization error similar to the original true positives (fig. 8). Additionally, the first and foremost requirement for driving safely is to localize the VRUs. Therefore, we suggest that the metric for determining true positives should be independent of the bounding box dimension estimation, which can be evaluated separately as we have done in this paper. This is analogous to how the orientation is evaluated with a separate performance metric, the AOS.

The proposal we give in this paper is to consider a maximum top-down localization error, for example, the 15 cm from fig. 8. This is similar to the method for nuScenes [27], although their performance metric averages over various thresholds from 0.5 to 4 meter, which is large for pedestrian detection. Additionally, instead of categorizing a dataset based on the expected difficulty of detecting targets, it would be more useful to categorize it based on how safety-critical it is to detect a target, for example, based on their distance to the ego-vehicle, or based on their (human-annotated) importance to the traffic situation. An alternative, if we consider the ability to detect most road users to be the main concern for self-driving vehicles, is to invert the performance metric: what top-down localization error threshold results in an AP_{3D} of 0.99? The resulting error threshold can then be interpreted as a measure of the uncertainty of every detection given by the detector.

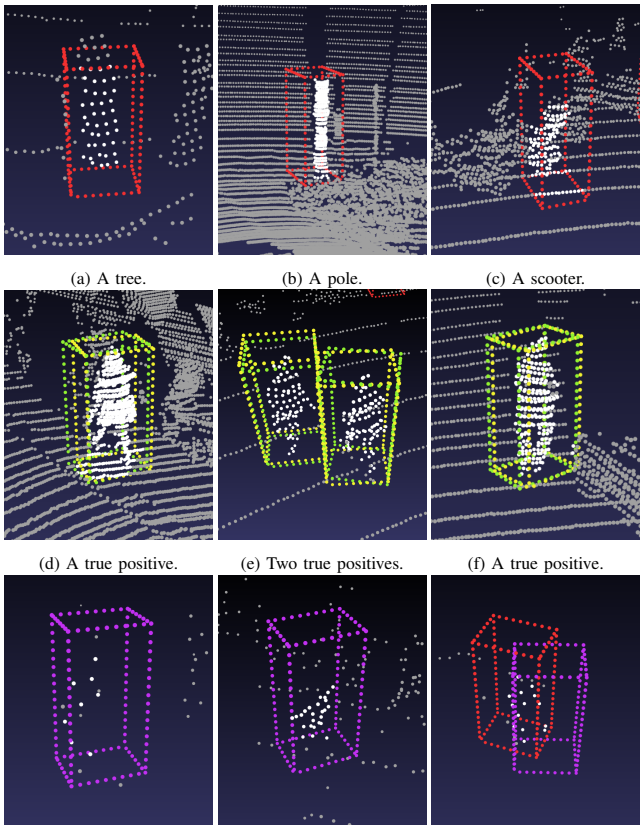
We evaluated the accuracy of PointPillars on both KITTI and ECP2.5D as a function of distance and LiDAR points on ground truth to investigate up to what distance this method already functions well, and how much it depends on current limitations of LiDAR resolution. The high recall up to 20 m (figs. 10a to 10c) suggests that self-driving vehicles outfitted with a current State-of-the-Art (SoA) detector are not barred to drive safely in urban areas at low velocities because of their



(a) An example of a point cloud including the 3D boxes representing the false positives, true positives and the false negatives.

(b) Corresponding image to point cloud.

Fig. 13. Overview of a frame from the validation part of the ECP2.5D dataset. False positives, false negatives, the detection and ground truth of a true positive are respectively shown in red, purple, green and yellow. LiDAR points inside a 3D box are white, other points are grey.



(a) A tree. (b) A pole. (c) A scooter.

(d) A true positive. (e) Two true positives. (f) A true positive.

(g) An object with few points. (h) An object where only the bottom part is visible. (i) A false negative and a detection with a too low IoU to be a true positive.

Fig. 14. An example of false positives (a-c), true positives (d-f) and false negatives (g-h). The false positives, false negatives, the detection and ground truth of a true positive are respectively shown in red, purple, green and yellow. LiDAR points inside a 3D box are white, other points are grey.

abilities to detect VRUs. This is not the same as ensuring that self-driving vehicles will be safe altogether, as we do not verify whether tracking, path prediction, and motion planning is also sufficiently capable. Zhou *et al*. [42] confirms the drop in performance over distance in the Waymo Open Dataset [26]. They compared the performance of their detector on the pedestrian and class over four categories: Overall, 0 - 30m, 30 - 50m, and 50m - infinity. Showing a decrease in performance for their multi-view fusion (MVF) detector over distance. Wang *et al*. [43] proposed cross-range adaptation framework as an addition to existing detectors. They evaluate the effect of distance on the performance (AP_{3D}) in the car class of the KITTI dataset, they identified three distance categories: near-range (0-40m), full-range (0-70m) and far-range (60-70m), showing decreasing performance for an increasing distance.

The strong correlation between recall and LiDAR points in ground truth (figs. 10d to 10f) shows that increasing the number of LiDAR points on a target from less than 20 to more than 50 greatly improves performance. At the same time, an increase in LiDAR resolution would not lead to a similar increase in the computational load of voxel-based networks such as PointPillars, as they consider a fixed amount of LiDAR points per voxel. Furthermore, the number of persons with less than 50 LiDAR points in their annotated bounding box also outweighs the number of persons with more than 50 LiDAR points by almost an order of magnitude in each dataset. Something which is also supported by the authors of Frustum PointNets [24], who state that one of the common failure of their detector is due to inaccurate pose and size estimation in a sparse point cloud, where they urge that some objects have less then 5 LiDAR points on them. This indicates that the challenge for future work is to make person detectors that can accurately detect with lower resolutions.

Additionally, the correlation between recall and LiDAR points in ground truth is similar for occluded and non-occluded pedestrians. (figs. 11d to 11f). This indicates that detection difficulty is better based on LiDAR points in the ground truth bounding box rather than occlusion or distance. Such a categorization has been mentioned in earlier work (e.g. [35]), and this paper shows it is indeed a useful approach.

Our domain transfer experiment (table VII) indicates that switching to another sensor or dataset will negatively influence the performance of a network. When using a network trained on one of these datasets on a self-driving vehicle with a different LiDAR, we show that right now best practice is to disregard the intensity information of each LiDAR point. However, the same experiment also shows that the best per-

forming network evaluated on KITTI does use the intensity information, which implies that there is relevant information available. Therefore, a method that can effectively utilize the intensity information of a LiDAR while it is trained on another dataset is an important avenue of future work. These are not the only challenges in LiDAR domain transfer transfer however, but the only ones we could identify as the two datasets used in this paper were recorded with the same type of LiDAR sensor. Different LiDAR sensors, whose planes are not necessarily distributed with the same density, add an additional layer of complexity [35].

Kashani *et al*. [44] presents an overview of the effective parameters for measuring intensity, underlying theory, and different methods to process intensity. The intensity values are influenced by many factors. As presented by Kashani *et al*., these factors can be subdivided into four categories. The first is the effect of the characteristics of the surface from which the laser pulse is reflected (*e.g*. reflectance and roughness). The second factors are the effects related to the acquisition (*e.g*. range, angle of incidence, and multiple returns from a laser beam). The third category is instrumental effects which include transmitted energy, intensity bit depth, and scaling of this digital number to a dynamical range, amplifiers for low reflected surfaces (if used), automatic gain control, brightness reducer for near distances and aperture size. The last category consists of effects caused by the environment like atmospheric transmittance or wetness of an object. Since many different factors influencing the intensity measurement, there will be a constant fluctuation. One can not guarantee that an object will have the same value while driving since there can be effects of for example range and angle of incidence. Because of these effects, LiDAR intensity values are preprocessed to reduce the variability caused by the aforementioned factors. One can split this processing into four different levels. The first level (level 0) is the raw intensity as provided by the manufacturer. When the intensity is corrected for one or more of the factors we refer to it as level 1. If the intensity is normalized through scaling to adjust the contrast / brightness, this is called level 2. The last level, level 3, the intensity values are corrected and calibrated using objects of which the intensity is known.

Bijelic *et al*. [45] investigated the intensity values for detection in foggy weather. For this research, they used the same sensor which is used during the data collection of both KITTI and ECP2.5D (Velodyne HDL-64E). They show a correlation between fog density and intensity values. BirdNet [46] analysed the performance of their network on the KITTI validation part with three different features used in their BEV map: intensity, normalized density, and height. The performance on the moderate pedestrian class is 30.4%, 32.7%, and 34.0% for respectively using the intensity, normalized density, and height features. If all features are combined the network has a performance of 39.5% showing the relevance of including all features.

When comparing the performance of detectors on the KITTI test benchmarks, it is noticeable that out of the top 5, 4 networks only use LiDAR information. If we look at the top 10, the number of fusion (image + LiDAR) networks grows to 4 and 2 for the pedestrian and car class respectively. The absence of image-only methods is not surprising since the this 2D data does not contain the 3D information needed to predict the 3D location of the objects. Since LiDAR methods do have this 3D information available, they have the means available to predict the 3D location of pedestrians. Surprisingly, the fusion of both LiDAR and image information does not result in SoA performance. Especially since the information from the sensors is complementary, where a LiDAR is strong at determining the distance, a camera image is denser in information but suffers from lightning and weather conditions. This is supported by Liang *et al*. [47] who state that detectors can benefit from information in high resolution images especially at large distances where the LiDAR modality suffers from extreme data sparsity. When speculating about the reason why current SoA detection methods are mostly LiDAR-only methods, I argue that this could be related to synchronization, or association issues between the different modalities. Another reason could be the capability of a detector to capture texture information from images as suggested by Guo *et al*. [48]. Shi *et al*. [49] also developed a LiDAR-only method, they consider that leverage image information could increase performance, especially for pedestrian because the pedestrians have a relative small size in a point cloud where an image captures more details. Therefore, future work could be improving the alignment of different sensor modalities and improving the capability of a image detector to exploit texture information in images.
Apart from fusion network, one could also try to improve the capability of LiDAR-only methods. Currently, a LiDAR point cloud can either be represented in 2D and 3D. Most 3D object detection networks transform the point cloud into a 2D (*e.g*. BEV) or 3D representations (*e.g*. Voxels). Guo *et al*. [48] claim that this transformation of the data has consequences in terms of information loss. An example is the maximum number of points per pillar in PointPillars. Therefore, future work could be to create a 3D object detection network which can take whole point cloud as input with the aim to exploit all the information in these point clouds as much as possible and eliminate the loss which occurs during transforming the data. Arnold *et al*. [14] also urges the need to explore the geometrical relationships between points.

There is an axis of evaluation that has not been touched upon in this paper however, that of time. The adverse effects of false positives depends a lot on whether or not they are randomly distributed in space, which would make it easy to filter them out. Similarly, these two datasets cannot show whether persons within 20 m are detected 80 % of the time, or whether only 80 % of all people are detected. Further study into existing detectors on datasets with track information (*e.g*. Waymo [26]) could provide more insight into this.

VI. Conclusion

This paper presented an experimental study on 3D person localization in traffic scenes, based on monocular vision and LiDAR data. In experiments on KITTI, we found that whereas headline results (AP_{3D}) might seem low, the 3D box center localization accuracies are quite high. The errors lowering AP_{3D} are mostly related to the estimates of the bounding box extents (especially, width and length).

PointPillars outperformed AVOD (AP_{3D} of 67 % vs. 48 % and 33 % vs. 28 %, for KITTI and ECP2.5D respectively, when not using LiDAR intensity information). The performance of both methods was significantly lower on ECP2.5D vs. KITTI, we attribute this to a larger prevalence of distant persons with fewer LiDAR points in ECP2.5D.

The performance on both KITTI and ECP2.5D decays past 20 m, this relation is even stronger for the number of LiDAR points per annotation. As expected the performance on occluded objects is lower compared to non-occluded objects.

Domain transfer experiments indicated the two datasets have quite different biases, in the sense that training on one and testing to the other leads to significantly degraded performance (upwards of AP_{3D} of 9.5 %). Further research is needed on cross-domain adaptation.

ACKNOWLEDGMENT

This work received support from the Dutch Science Foundation NWO-TTW within the SafeVRU project (nr. 14667).

REFERENCES

[1] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 743–761, 2011.

[2] X. Li, F. Flohr, Y. Yang, H. Xiong, M. Braun, S. Pan, K. Li, and D. M. Gavrila, "A new benchmark for vision-based cyclist detection," in *Proceedings of the IEEE Intelligent Vehicles Symposium*, 2016, pp. 1028–1033.

[3] M. Braun, S. Krebs, F. B. Flohr, and D. M. Gavrila, "EuroCity Persons: A novel benchmark for person detection in traffic scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1844–1861, Aug 2019.

[4] C. G. Keller, T. Dang, H. Fritz, A. Joos, C. Rabe, and D. M. Gavrila, "Active pedestrian safety by automatic braking and evasive steering," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1292–1304, 2011.

[5] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[6] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast encoders for object detection from point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 697–12 705.

[7] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3D Proposal Generation and Object Detection from View Aggregation," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, October 2018, pp. 5750–5757.

[8] M. Braun, S. Krebs, and D. M. Gavrila, "ECP2.5D - Person localization in traffic scenes," in *Proceedings of the IEEE Intelligent Vehicles Symposium*, 2020.

[9] A. Naiden, V. Paunescu, G. Kim, B. Jeon, and M. Leordeanu, "Shift R-CNN: Deep monocular 3D object detection with closed-form geometric constraints," in *Proceedings of the IEEE International Conference on Image Processing*, Sep. 2019, pp. 61–65.

[10] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia, "STD: Sparse-to-dense 3D object detector for point cloud," in *Proceedings of the IEEE International Conference on Computer Vision*, October 2019.

[11] S. Shi, Z. Wang, J. Shi, X. Wang, and H. Li, "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*, pp. 1–1, 2020.

[12] S. Vora, A. H. Lang, B. Helou, and O. Beijbom, "PointPainting: Sequential fusion for 3D object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2020.

[13] Z. Yang, Y. Sun, S. Liu, and J. Jia, "3DSSD: Point-based 3D single stage object detector," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2020.

[14] E. Arnold, O. Y. Al-Jarrah, M. Dianati, S. Fallah, D. Oxtoby, and A. Mouzakitis, "A survey on 3D object detection methods for autonomous driving applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3782–3795, 2019.

[15] B. Yang, M. Liang, and R. Urtasun, "HDNET: Exploiting HD maps for 3D object detection," in *Proceedings of the Conference on Robot Learning*, vol. 87, Oct 2018, pp. 146–155.

[16] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington, "LaserNet: An efficient probabilistic 3D object detector for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2019.

[17] Y. Zhou and O. Tuzel, "VoxelNet: End-to-end learning for point cloud based 3D object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2018.

[18] Z. Liu, X. Zhao, T. Huang, R. Hu, Y. Zhou, and X. Bai, "TANet: Robust 3D object detection from point clouds with triple attention," in *Proceedings of the AAAI Conference on Artificial Intelligence*, February 2020.

[19] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "PV-RCNN: Point-voxel feature set abstraction for 3D object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2020, pp. 10 529–10 538.

[20] Z. Wang, H. Fu, L. Wang, L. Xiao, and B. Dai, "SCNet: Subdivision coding network for object detection based on 3D point cloud," *IEEE Access*, vol. 7, pp. 120 449–120 462, 2019.

[21] W. Shi and R. Rajkumar, "Point-GNN: Graph neural network for 3D object detection in a point cloud," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2020.

[22] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and Segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, July 2017.

[23] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, 2017, pp. 5099–5108.

[24] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum PointNets for 3D object detection from RGB-D data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2018, pp. 918–927.

[25] V. A. Sindagi, Y. Zhou, and O. Tuzel, "MVX-Net: Multimodal voxelnet for 3D object detection," in *Proceedings of the IEEE International Conference on Robotics and Automation*, May 2019, pp. 7276–7282.

[26] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2020.

[27] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuScenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2020.

[28] M.-F. Chang *et al.*, "Argoverse: 3D tracking and forecasting with rich maps," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[29] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska, S. Omari, S. Shah, A. Kulkarni, A. Kazakova, C. Tao, L. Platinsky, W. Jiang, and V. Shet, "Lyft level 5 AV dataset 2019," https://level5.lyft.com/dataset/, 2019.

[30] D. Xu, D. Anguelov, and A. Jain, "PointFusion: Deep Sensor Fusion for 3D Bounding Box Estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2018.

[31] M. Roth, D. Jargot, and D. M. Gavrila, "Deep end-to-end 3D person detection from camera and LiDAR," in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, 2019, pp. 521–527.

[32] A. Asvadi, L. Garrote, C. Premebida, P. Peixoto, and U. J. Nunes, "Multimodal vehicle detection: fusing 3D-LIDAR and color camera data," *Pattern Recognition Letters*, vol. 115, pp. 20–29, 2018.

[33] Z. Wang and K. Jia, "Frustum ConvNet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, 2019.

[34] D. Feng, C. Haase-Schütz, L. Rosenbaum, H. Hertlein, C. Gläser, F. Timm, W. Wiesbeck, and K. Dietmayer, "Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–20, 2020.

[35] C. Rist, M. Enzweiler, and D. M. Gavrila, "Cross-sensor deep domain adaptation for LiDAR detection and segmentation," in *Proceedings of the IEEE Intelligent Vehicles Symposium*, 2019.

[36] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proceedings of the European Conference on Computer Vision*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., 2016, pp. 21–37.

[37] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3D object detection network for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6526–6534.

[38] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *International Conference on Learning Representations*, 2015.

[39] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, July 2017.

[40] D. Zhou, J. Fang, X. Song, C. Guan, J. Yin, Y. Dai, and R. Yang, "IoU loss for 2D/3D object detection," in *Proceedings of the International Conference on 3D Vision*, 2019, pp. 85–94.

[41] A. Simonelli, S. R. R. Bulò, L. Porzi, M. López-Antequera, and P. Kontschieder, "Disentangling monocular 3D object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, October 2019.

[42] Y. Zhou, P. Sun, Y. Zhang, D. Anguelov, J. Gao, T. Ouyang, J. Guo, J. Ngiam, and V. Vasudevan, "End-to-end multi-view fusion for 3D object detection in LiDAR point clouds," in *Proceedings of the Conference on Robot Learning*, vol. 100, October 2020, pp. 923–932.

[43] Z. Wang, S. Ding, Y. Li, M. Zhao, S. Roychowdhury, A. Wallin, G. Sapiro, and Q. Qiu, "Range adaptation for 3D object detection in LiDAR," in *Proceedings of the IEEE International Conference on Computer Vision*, Oct 2019.

[44] A. Kashani, M. Olsen, C. Parrish, and N. Wilson, "A review of LIDAR radiometric processing: From ad hoc intensity correction to rigorous radiometric calibration," *Sensors*, vol. 15, no. 11, p. 28099–28128, Nov 2015.

[45] M. Bijelic, T. Gruber, and W. Ritter, "A benchmark for lidar sensors in fog: Is detection breaking down?" in *Proceedings of the IEEE Intelligent Vehicles Symposium*, 2018, pp. 760–767.

[46] J. Beltrán, C. Guindel, F. M. Moreno, D. Cruzado, F. García, and A. De La Escalera, "BirdNet: A 3D object detection framework from LiDAR information," in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, 2018, pp. 3517–3523.

[47] M. Liang, B. Yang, S. Wang, and R. Urtasun, "Deep continuous fusion for multi-sensor 3D object detection," in *Proceedings of the European Conference on Computer Vision*, September 2018.

[48] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–27, 2020.

[49] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3D object proposal generation and detection from point cloud," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2019.

Joram R. van der Sluis received his Bachelor's degree in mechanical engineering at the TU Delft in 2018. Since then he is working on his Master's degree in mechanical engineering at TU Delft focusing on the perception of intelligent vehicles. His research interests include the field of robotics, especially the perception of the environment for automated devices, and machine learning.

Ewoud A. I. Pool received the Master's degree in System and Control engineering at the TU Delft in 2015. Since 2016, he is working on obtaining the Ph.D. degree at the TU Delft, focusing on path prediction of vulnerable road users for use in automated vehicles. His research interests include the modeling of human dynamics, perception for automated vehicles, and machine learning.

Dariu M. Gavrila received the Ph.D. degree in computer science from Univ. of Maryland at College Park, USA, in 1996. From 1997 until 2016, he was with Daimler R&D, Ulm, Germany, where he became a Distinguished Scientist. In 2016, he moved to TU Delft, where he since heads the Intelligent Vehicles group as a Full Professor. His research deals with sensor-based detection of humans and analysis of behavior, most recently in the context of the self-driving car in complex urban traffic. He was awarded the Outstanding Application Award 2014 and the Outstanding Researcher Award 2019, both from the IEEE Intelligent Transportation Systems Society.

Bibliography

[1] J. M. Armingol and A. de la Escalera. Intelligent systems lab (LSI): Mission its research lab. *IEEE Intelligent Transportation Systems Magazine*, 8(2):107–109, 2016.

[2] E. Arnold, O. Y. Al-Jarrah, M. Dianati, S. Fallah, D. Oxtoby, and A. Mouzakitis. A survey on 3D object detection methods for autonomous driving applications. *IEEE Transactions on Intelligent Transportation Systems*, 20(10):3782–3795, 2019.

[3] A. Asvadi, L. Garrote, C. Premebida, P. Peixoto, and U. J. Nunes. Multimodal vehicle detection: fusing 3D-LIDAR and color camera data. *Pattern Recognition Letters*, 115:20–29, 2018.

[4] J. Beltrán, C. Guindel, F. M. Moreno, D. Cruzado, F. García, and A. De La Escalera. BirdNet: A 3D object detection framework from LiDAR information. In *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, pages 3517–3523, 2018.

[5] P. Bergmann, T. Meinhardt, and L. Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 941–951, 2019.

[6] M. Bijelic, T. Gruber, and W. Ritter. A benchmark for LiDAR sensors in fog: Is detection breaking down? In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 760–767, 2018.

[7] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis. Soft-NMS – improving object detection with one line of code. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5562–5570, 2017.

[8] M. Braun, S. Krebs, and D. M. Gavrila. ECP2.5D - Person localization in traffic scenes. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, 2020.

[9] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuScenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11621–11631, 2020.

[10] M.-F. Chang et al. Argoverse: 3D tracking and forecasting with rich maps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8748–8757, 2019.

[11] Q. Chen, L. Sun, Z. Wang, K. Jia, and A. Yuille. Object as hotspots: An anchor-free 3D object detection approach via firing of hotspots. In *Proceedings of the European Conference on Computer Vision*, 2020.

[12] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-view 3D object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6526–6534, 2017.

[13] X. Cheng, P. Wang, C. Guan, and R. Yang. CSPN++: Learning context and resource aware convolutional spatial propagation networks for depth completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 10615–10622, 2020.

[14] D. Feng, C. Haase-Schütz, L. Rosenbaum, H. Hertlein, C. Gläser, F. Timm, W. Wiesbeck, and K. Dietmayer. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–20, 2020.

[15] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao. Deep ordinal regression network for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2002–2011, 2018.

[16] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, 2012.

[17] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research*, pages 1231–1237, 2013.

[18] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.

[19] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun. Deep learning for 3D point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–27, 2020.

[20] R. Huang, W. Zhang, A. Kundu, C. Pantofaru, D. A. Ross, T. Funkhouser, and A. Fathi. An LSTM approach to temporal 3D object detection in LiDAR point clouds. In *Proceedings of the European Conference on Computer Vision*, 2020.

[21] A. Kashani, M. Olsen, C. Parrish, and N. Wilson. A review of LIDAR radiometric processing: From ad hoc intensity correction to rigorous radiometric calibration. *Sensors*, 15(11):28099–28128, 2015.

[22] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska, S. Omari, S. Shah, A. Kulkarni, A. Kazakova, C. Tao, L. Platinsky, W. Jiang, and V. Shet. Lyft level 5 AV dataset 2019. `https://level5.lyft.com/dataset/`, 2019.

[23] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[24] J. Ku, A. D. Pon, S. Walsh, and S. L. Waslander. Improving 3D object detection for pedestrians with virtual multi-view synthesis orientation estimation. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 3459–3466, 2019.

[25] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander. Joint 3D Proposal Generation and Object Detection from View Aggregation. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 5750–5757, 2018.

[26] A. Laddha, S. Gautam, G. P. Meyer, and C. Vallespi-Gonzalez. RV-FuseNet: Range view based fusion of time-series LiDAR data for joint 3D object detection and motion forecasting. In *submission to Proceedings of the British Machine Vision Conference*, 2020.

[27] J. Lambert, Z. Liu, O. Sener, J. Hays, and V. Koltun. Mseg: A composite dataset for multi-domain semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2879–2888, 2020.

[28] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom. PointPillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12697–12705, 2019.

[29] S. Lee, S. Kwak, and M. Cho. Universal bounding box regression and its applications. In C. Jawahar, H. Li, G. Mori, and K. Schindler, editors, *Proceedings of the Asian Conference on Computer Vision*, pages 373–387, Cham, 2019.

[30] M. Liang, B. Yang, S. Wang, and R. Urtasun. Deep continuous fusion for multi-sensor 3D object detection. In *Proceedings of the European Conference on Computer Vision*, 2018.

[31] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017.

[32] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In B. Leibe, J. Matas, N. Sebe, and M. Welling, editors, *Proceedings of the European Conference on Computer Vision*, pages 21–37, 2016.

[33] Z. Liu, X. Zhao, T. Huang, R. Hu, Y. Zhou, and X. Bai. TANet: Robust 3D object detection from point clouds with triple attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, page 11677–11684, 2020.

[34] Z. Liu, H. Tang, Y. Lin, and S. Han. Point-voxel CNN for efficient 3D deep learning. In *Advances in Neural Information Processing Systems*, volume 31, pages 965–975, 2019.

[35] W. Luo, B. Yang, and R. Urtasun. Fast and furious: Real time end-to-end 3D detection, tracking and motion forecasting with a single convolutional net. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3569–3577, 2018.

[36] A. Mauri, R. Khemmar, B. Decoux, N. Ragot, R. Rossi, R. Trabelsi, R. Boutteau, J.-Y. Ertaud, and X. Savatier. Deep learning for real-time 3D multi-object detection, localisation, and tracking: Application to smart mobility. *Sensors*, 20(2):532, 2020.

[37] S. McCrae and A. Zakhor. 3D object detection using temporal LiDAR data. In *Proceedings of the IEEE International Conference on Image Processing*, 2020.

[38] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington. LaserNet: An efficient probabilistic 3D object detector for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12677–12686, 2019.

[39] M. Meyer and G. Kuschk. Deep learning based 3D object detection for automotive radar and camera. In *European Radar Conference*, pages 133–136, 2019.

[40] A. Naiden, V. Paunescu, G. Kim, B. Jeon, and M. Leordeanu. Shift R-CNN: Deep monocular 3D object detection with closed-form geometric constraints. In *Proceedings of the IEEE International Conference on Image Processing*, pages 61–65, 2019.

[41] A. Palffy. *TU Delft Prius with LiDAR*. 2018.

[42] A. Palffy, J. Dong, J. F. P. Kooij, and D. M. Gavrila. CNN based road user detection using the 3D radar cube. *IEEE Robotics and Automation Letters*, 5(2):1263–1270, 2020.

[43] J. Park, K. Joo, Z. Hu, C.-K. Liu, and I. S. Kweon. Non-local spatial propagation network for depth completion. In *Proceedings of the European Conference on Computer Vision*, 2020.

[44] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep learning on point sets for 3D classification and Segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 77–85, 2017.

[45] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas. Frustum PointNets for 3D object detection from RGB-D data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 918–927, 2018.

[46] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 5099–5108, 2017.

[47] N. Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12 (1):145–151, 1999.

[48] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015.

[49] C. Rist, M. Enzweiler, and D. M. Gavrila. Cross-sensor deep domain adaptation for LiDAR detection and segmentation. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 1535–1542, 2019.

[50] M. Roth, D. Jargot, and D. M. Gavrila. Deep end-to-end 3D person detection from camera and LiDAR. In *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, pages 521–527, 2019.

[51] SAE international. *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*, 2018.

[52] S. Shi, Z. Wang, J. Shi, X. Wang, and H. Li. 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*, 2020.

[53] S. Shi, X. Wang, and H. Li. PointRCNN: 3D object proposal generation and detection from point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–779, 2019.

[54] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li. PV-RCNN: Point-voxel feature set abstraction for 3D object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020.

[55] W. Shi and R. Rajkumar. Point-GNN: Graph neural network for 3D object detection in a point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1711–1719, 2020.

[56] A. Simonelli, S. R. R. Bulò, L. Porzi, M. López-Antequera, and P. Kontschieder. Disentangling monocular 3D object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1991–1999, 2019.

[57] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations*, 2015.

[58] V. A. Sindagi, Y. Zhou, and O. Tuzel. MVX-Net: Multimodal voxelnet for 3D object detection. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 7276–7282, 2019.

[59] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2446–2454, 2020.

[60] P. Thomas, A. Morris, R. Talbot, and H. Fagerlind. Identifying the causes of road crashes in europe. *Annals of advances in automotive medicine*, 57:13–22, 2013.

[61] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

[62] J. R. van der Sluis, E. A. I. Pool, and D. M. Gavrila. An Experimental Study on 3D Person Localization in Traffic Scenes. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, 2020.

[63] J. R. van der Sluis, E. A. I. Pool, and D. M. Gavrila. An Experimental Study on 3D Person Localization in Traffic Scenes. In *To be submitted to: IEEE Transactions on Intelligent Vehicles*, 2020.

[64] S. Vora, A. H. Lang, B. Helou, and O. Beijbom. PointPainting: Sequential fusion for 3D object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4604–4612, 2020.

[65] T. Wang, X. Zhu, and D. Lin. Reconfigurable voxels: A new representation for LiDAR-based point clouds. In *arXiv preprint arXiv:2004.02724*, 2020.

[66] Z. Wang, H. Fu, L. Wang, L. Xiao, and B. Dai. SCNet: Subdivision coding network for object detection based on 3D point cloud. *IEEE Access*, 7:120449–120462, 2019.

[67] Z. Wang, S. Ding, Y. Li, M. Zhao, S. Roychowdhury, A. Wallin, G. Sapiro, and Q. Qiu. Range adaptation for 3D object detection in LiDAR. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[68] Z. Wang and K. Jia. Frustum ConvNet: Sliding frustums to aggregate local point-wise features for amodal 3D object detection. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 1742–1749, 2019.

[69] L. Waymo. On the road to fully self-driving. *Waymo Safety Report*, pages 1–43, 2017.

[70] X. Weng, J. Wang, D. Held, and K. Kitani. 3D Multi-Object Tracking: A Baseline and New Evaluation Metrics. *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, 2020.

[71] World Health Organization. *Global status report on road safety*. 2018.

[72] D. Xu, D. Anguelov, and A. Jain. PointFusion: Deep Sensor Fusion for 3D Bounding Box Estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 244–253, 2018.

[73] B. Yang, M. Liang, and R. Urtasun. HDNET: Exploiting HD maps for 3D object detection. In *Proceedings of the Conference on Robot Learning*, volume 87, pages 146–155, 2018.

[74] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia. STD: Sparse-to-dense 3D object detector for point cloud. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1951–1960, 2019.

[75] Z. Yang, Y. Sun, S. Liu, and J. Jia. 3DSSD: Point-based 3D single stage object detector. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11040–11048, 2020.

[76] Y. Ye, C. Zhang, and X. Hao. ARPNET: attention region proposal network for 3D object detection. *Science China Information Sciences*, 62(12):220104, 2019.

[77] D. Zhou, J. Fang, X. Song, C. Guan, J. Yin, Y. Dai, and R. Yang. IoU loss for 2D/3D object detection. In *Proceedings of the International Conference on 3D Vision*, pages 85–94, 2019.

[78] Y. Zhou and O. Tuzel. VoxelNet: End-to-end learning for point cloud based 3D object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018.

[79] Y. Zhou, P. Sun, Y. Zhang, D. Anguelov, J. Gao, T. Ouyang, J. Guo, J. Ngiam, and V. Vasudevan. End-to-end multi-view fusion for 3D object detection in LiDAR point clouds. In *Proceedings of the Conference on Robot Learning*, volume 100, pages 923–932, 2020.