Hochschule Ravensburg-Weingarten

University of Applied Sciences

Lidar and Radar Systems

Task: Evaluation of an 3D Object Detector

Sr. No.	Student Name	Matriculation Numbers	Study Programm
1	Arjun Rameshbhai Beladiya	28744163	Mechatronics
2	Dhaval Jagdish Mistry	20644015	Mechatronics

Guided by:
Prof. Dr. rer. nat. Stefan Elser

Contents

1 Introduction

This task focuses on detecting cars in LiDAR points when they are projected on 2D image as well as mapped in 3D space. To perform this task, KITTI 360 sample (3) dataset was used; which is sub Dataset of KITTI 360(2). Further in this report, we will discuss Methodology, Results and their Evaluation.

2 Methodology

Before starting this project, we have to import the libraries open3d, cv2, numpy, os, and json to perform essential tasks such as 3D point cloud visualization, image processing, numerical computations, file management, and data visualization. The detailed approach to our solution is explained below.

2.1 Loading the KITTI-360 Dataset

In this task, The used KITTI-360 sample dataset (3) contains all necessary data such as camera images, LiDAR scans, calibration file, and additionally contains 3D ground truth bounding boxes. To utilize these data effectively, we first need to understand the content of these files.

- **data_2d_raw/**: Contains the raw camera data. Specifically, the `image_00/data_rect/` subfolder contains 20 images from the left camera.
- **data_3d_raw/**: Contains raw 3D LiDAR point cloud data (`.bin` file) captured by veodyne sensor.
- **calibration/**: Includes all necessary calibration files such as `calib_cam_to_velo`, `perspective`.
- **bboxes_3D_cam0/**: Contains the annotated ground truth 3D bounding boxes in JSON format.

2.2 YOLO Semantic Segmentation on Images

First we need to mask the cars in 2D image which is later used for finding LiDAR points corresponding to cars. For this semantic segmentation, we have chose the YOLOv8x-seg (1) model because it provides a good balance between performance and accuracy, producing precise pixel-wise masks. For this task, we will only focus on cars. Therefore, all other objects (Pedestrians, cyclists, Truck, Traffic Signs, etc) are filtered out. we have used different colors for different segmentated cars.

2.3 Projection of LiDAR Points onto the Image

Now to accurately project 3D LiDAR points onto 2D camera images, first we need to align the coordinate systems of the LiDAR sensor and the camera. which we have done with the help of `calib_cam_to_velo` and `perspective` file given KITTI 360 subdataset (3).

2.3.1 Transformation from LiDAR to Camera Frame

`calib_cam_to_velo` file contains 12 numerical values of 3×4 transformation matrix which maps coordinates from the camera frame to the LiDAR frame. but we want to project LiDAR points into the camera frame, so we took inverse of the transformation matrix to convert LiDAR coordinates into the camera coordinate system:

$$T_{\text{velo}\to\text{cam}} = (T_{\text{cam}\to\text{velo}})^{-1}$$

Now we have all 3D points from the LiDAR's world into the same spatial frame as camera.

2.3.2 Conversion Projection Matrix

The LiDAR sensor captures points in its local 3D coordinate frame, which is represented as:

$$\mathbf{P}_{\text{lidar}} = \begin{bmatrix} x & y & z \end{bmatrix}^T$$

To apply transformation, we have to convert 3D points to homogeneous coordinates by adding an extra 1 to make it a 4x1 vector.

$$\mathbf{P}_{\text{hom}} = \begin{bmatrix} x & y & z & 1 \end{bmatrix}^T$$

Now, we can do both rotation and translation and by multiplying transformation matrix to $\mathbf{P}_{\text{hom}}$ we can convert points in the camera frame:

$$\mathbf{P}_{\text{cam}} = T_{\text{velo}\to\text{cam}} \cdot \mathbf{P}_{\text{hom}}$$

2.3.3 Using `P_rect_00` to Project 3D Points into 2D Image Plane

Now, with the help of projection matrix `P_rect_00` we will project 3D points from the camera coordinate frame onto the 2D image plane, which is provided in the `perspective.txt` calibration file. This matrix contains both the intrinsic camera parameters and rectification transformation to ensure the projected points align with the undistorted (rectified) image.

$$P_{\text{rect}} = \begin{bmatrix} f_x & 0 & c_x & t_x \\ 0 & f_y & c_y & t_y \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Where:

- f_x, f_y are the focal lengths in pixels (scaling factors),
- c_x, c_y are the optical center coordinates (principal point),
- t_x, t_y are translation offsets (often 0 or adjusted for stereo calibration).

Now we have all required parameters to project each 3D points $\mathbf{P}_{\text{cam}} = \begin{bmatrix} x & y & z & 1 \end{bmatrix}^T$ into image plane using this equation:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = P_{\text{rect\_00}} \cdot \mathbf{P}_{\text{cam}}$$

In the next step we have applyed perspective division to convert result into valid 2D pixel coordinates (u', v')

$$u' = \frac{u}{w}, \quad v' = \frac{v}{w}$$

These pixel coordinates represent where each 3D LiDAR point appears on the 2D image.

2.4 Matching Projected Points with YOLO Masks

Now, the LiDAR points are being projected onto the image plane. But which point is corresponds to which car is still remains a problem. To solve this pixel-wise segmentation masks generated by the YOLOv8x-seg model (1) is used. Every detected car has a binary mask, which tells us which specific pixel belongs to that specific car. These masks are stored as 2D arrays with values of 1 indicating the presence of the object and 0 elsewhere. Now, if the pixel value at (u', v') is 1, then the LiDAR point is inside the corresponding car region and to differentiate every car points. A unique RGB have been assigned color to every LiDAR point laying inside every car. If the point is outside the car, then it considered background and is colored black.

2.5 Projecting Lidar points into 3D Space

To visualize the LiDAR points in 3D space, we have used the Open3D library. The projected points are represented as a point cloud, which can be visualized in a 3D coordinate system. This point cloud was created by taking the projected 2D pixel coordinates and mapping them back to their corresponding 3D coordinates using the inverse of the projection matrix. After that we have reapplied colors(which was saved during Yolo segmentation) to the points corresponding to cars.

2.6 Comparing LiDAR Points with Ground Truth Bounding Boxes

To evaluate the accuracy of the LiDAR points against the ground truth bounding boxes, we have used the 3D bounding box annotations provided in the KITTI 360 Sample Dataset (3). The ground truth bounding boxes are defined in the 3D space of LiDAR and contain the 3D coordinates of the corners of each box. To evaluate the projected LiDAR points with the ground truth bounding boxes, first we have checked if the projected points fall within the boundaries of these boxes.

2.6.1 Precision and Recall

To evaluate the performance of our LiDAR–image fusion and segmentation pipeline, we use two primary formulae: Precision and Recall
Precision: It measures the accuracy of the detections made by the model. It is the ratio of (TP) to (TP + FP). Formula:

$$\text{Precision} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}} \tag{1}$$

Recall: It quantifies the model's ability to detect all relevant objects in the scene. It is the ratio of (TP) to (TP + FN). Formula:

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}} \tag{2}$$

Where:

- **True Positive (TP):** A point correctly assigned to the car it belongs to, i.e., it lies within the predicted 3D GT bounding box and has the correct segmentation label.
- **False Positive (FP):** A point that is included in a predicted 3D bounding box but does not belong to that car.
- **False Negative (FN):** A point belonging to a car (based on segmentation) that was missed by the predicted 3D bounding box.

By the help of Recall and Precision values, we can evaluate the performance of the object detection model. The more we get the value close to 1, the better the performance. Below [1]i have attached the result of Precision and Recall of our images in the form of graph.

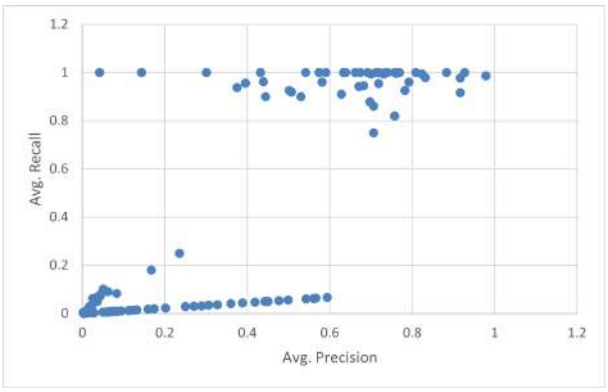


Figure 1: Avg. Precision vs Avg. Recall graph

2.7 Flow chart

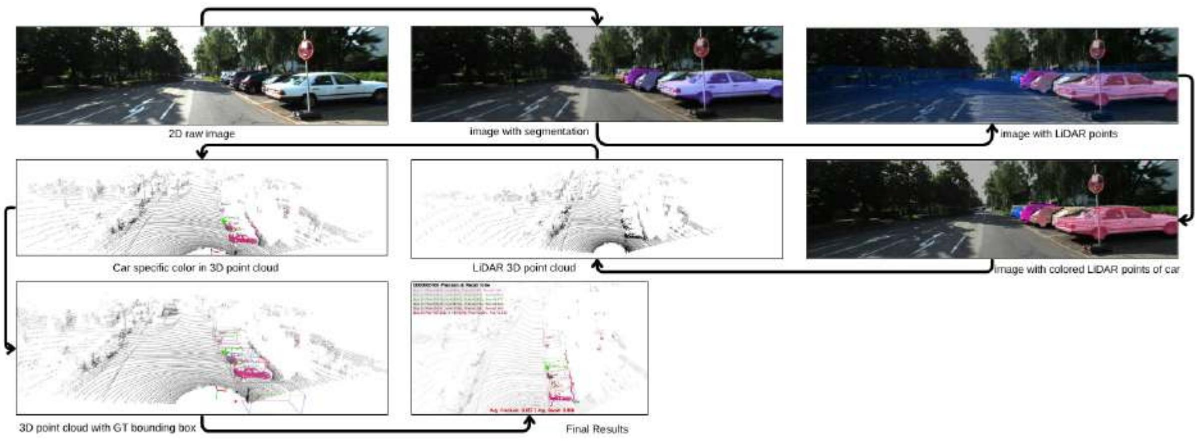

2D raw image
image with segmentation
image with LiDAR points
Car specific color in 3D point cloud
LiDAR 3D point cloud
image with colored LiDAR points of car
3D point cloud with GT bounding box
Final Results

Figure 2: Flowchart

3 Results

Final result of images, which contains points belong to a car, points inside bounding box (TP), precision, recall, Avg. precision and Avg. recall values.

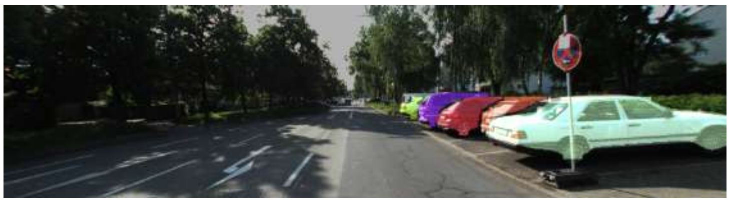


Figure 3: 2D View: 0000000100.png

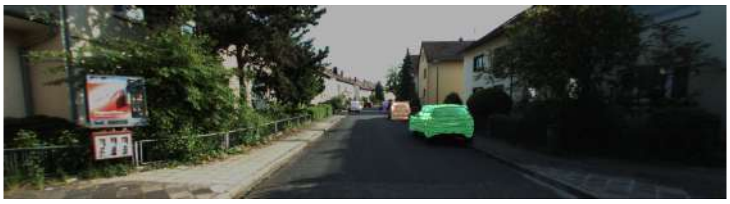
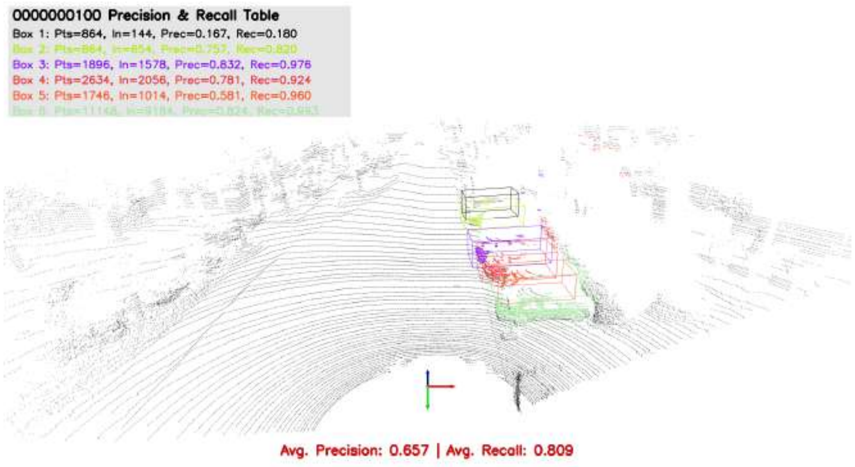


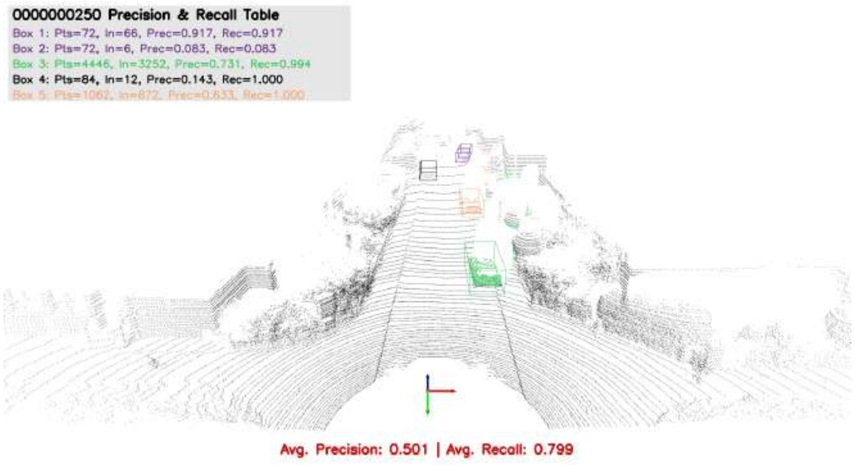
Figure 4: 2D View: 0000000250.png



```
0000000100 Precision & Recall Table
Box 1: Pts=864, In=144, Prec=0.167, Rec=0.180
Box 2: Pts=864, In=654, Prec=0.757, Rec=0.820
Box 3: Pts=1896, In=1578, Prec=0.832, Rec=0.976
Box 4: Pts=2634, In=2056, Prec=0.781, Rec=0.924
Box 5: Pts=1746, In=1014, Prec=0.581, Rec=0.960
Box 6: Pts=11148, In=9184, Prec=0.824, Rec=0.953
```

```
Avg. Precision: 0.657 | Avg. Recall: 0.809
```

Figure 5: 3D Point cloud: 100.png



```
0000000250 Precision & Recall Table
Box 1: Pts=72, In=66, Prec=0.917, Rec=0.917
Box 2: Pts=72, In=6, Prec=0.083, Rec=0.083
Box 3: Pts=4446, In=3252, Prec=0.731, Rec=0.994
Box 4: Pts=84, In=12, Prec=0.143, Rec=1.000
Box 5: Pts=1062, In=672, Prec=0.633, Rec=1.000
```

```
Avg. Precision: 0.501 | Avg. Recall: 0.799
```

Figure 6: 3D Point cloud: 250.png

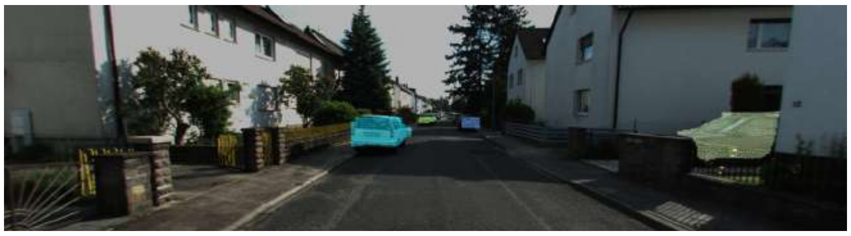


Figure 7: 2D View: 0000000360.png

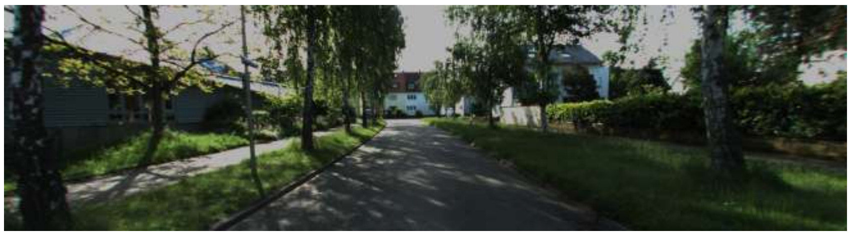
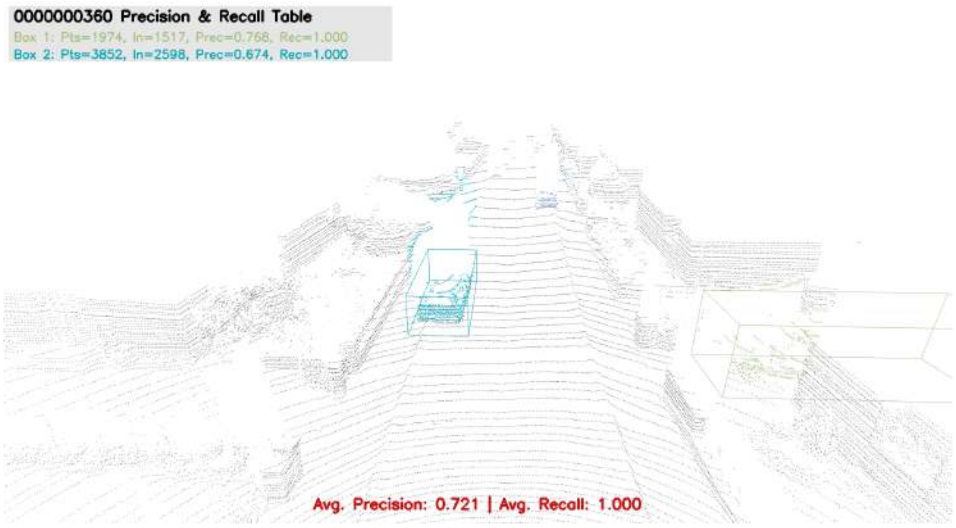
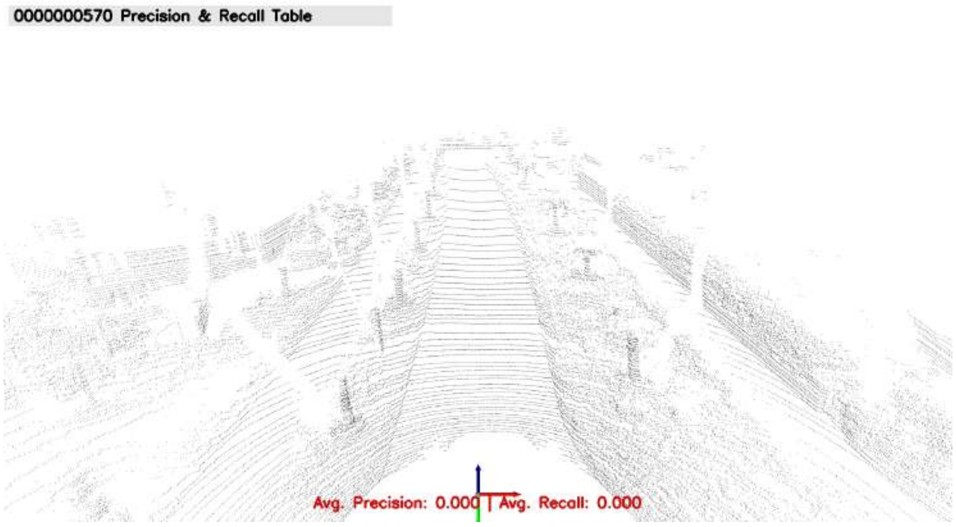


Figure 8: 2D View: 0000000570.png



```
0000000360 Precision & Recall Table
Box 1: Pts=1974, In=1517, Prec=0.768, Rec=1.000
Box 2: Pts=3852, In=2598, Prec=0.674, Rec=1.000
Avg. Precision: 0.721 | Avg. Recall: 1.000
```

Figure 9: 3D Point cloud: 360.png



```
0000000570 Precision & Recall Table
Avg. Precision: 0.000 | Avg. Recall: 0.000
```

Figure 10: 3D Point cloud: 570.png

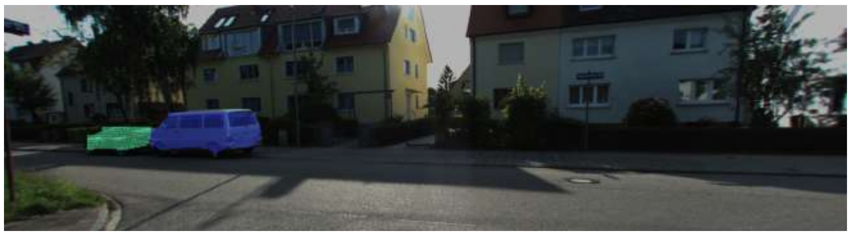


Figure 11: 2D View: 0000000644.png

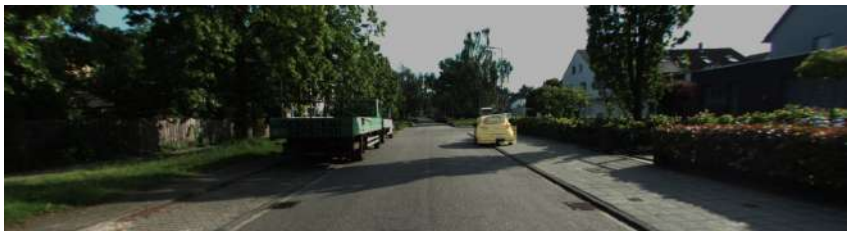
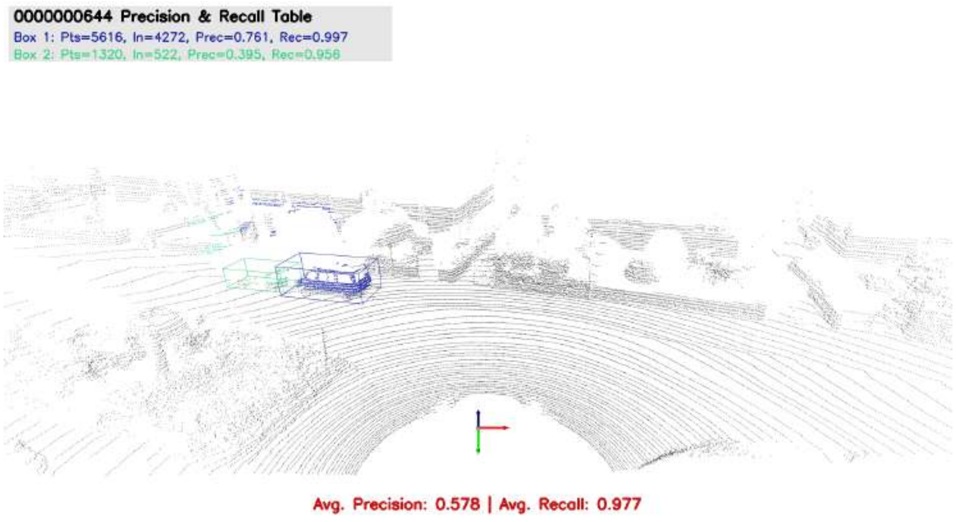


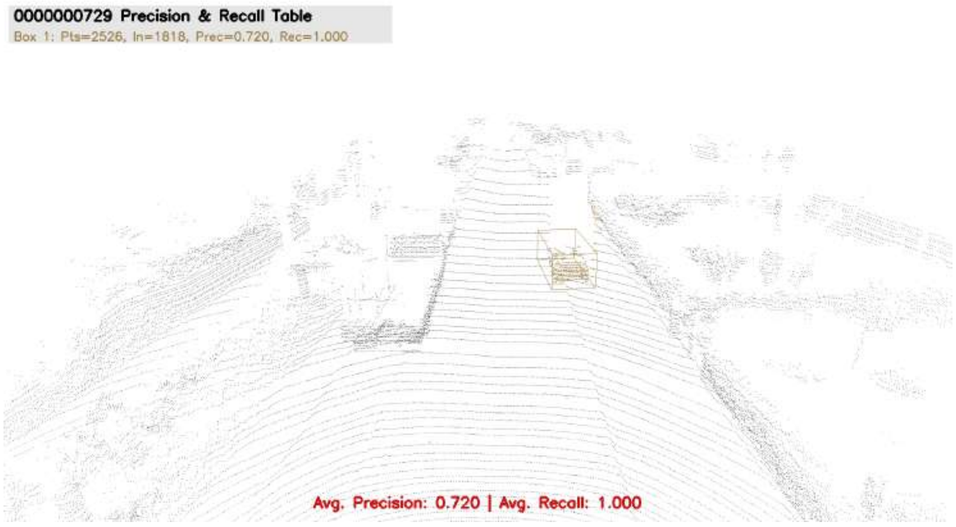
Figure 12: 2D View: 0000000729.png



```
0000000644 Precision & Recall Table
Box 1: Pts=5616, In=4272, Prec=0.761, Rec=0.997
Box 2: Pts=1320, In=522, Prec=0.395, Rec=0.956
```

```
Avg. Precision: 0.578 | Avg. Recall: 0.977
```

Figure 13: 3D Point cloud: 644.png



```
0000000729 Precision & Recall Table
Box 1: Pts=2526, In=1818, Prec=0.720, Rec=1.000
```

```
Avg. Precision: 0.720 | Avg. Recall: 1.000
```

Figure 14: 3D Point cloud: 729.png

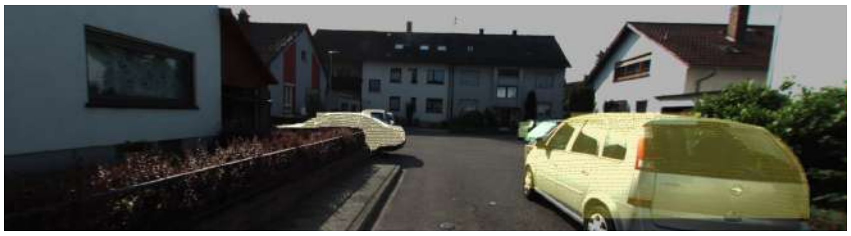


Figure 15: 2D View: 0000000947.png

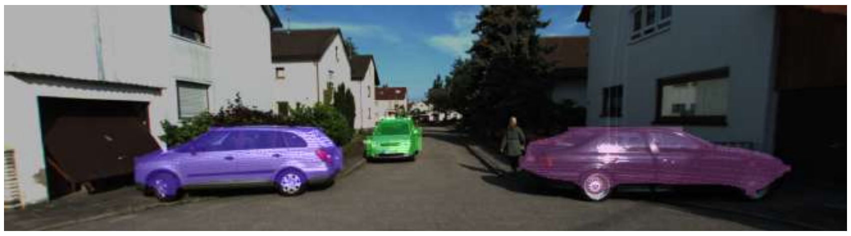
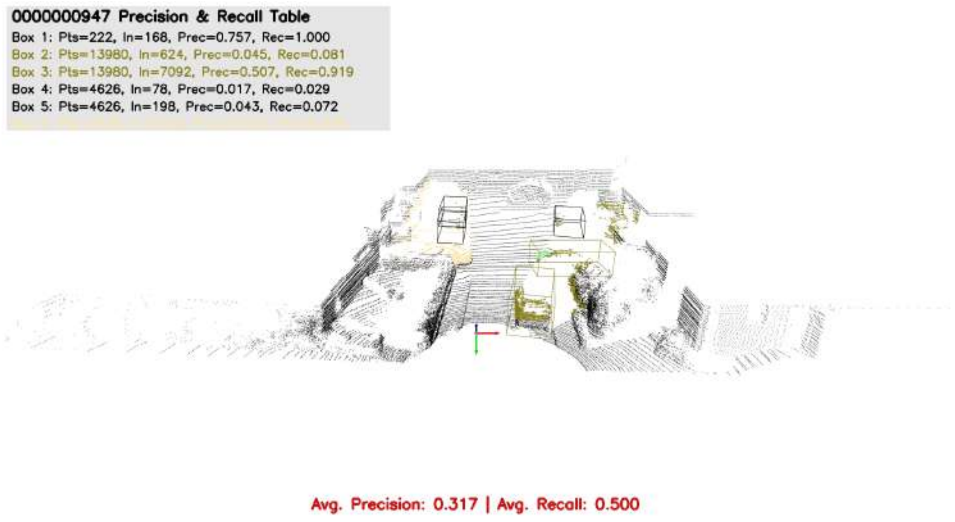


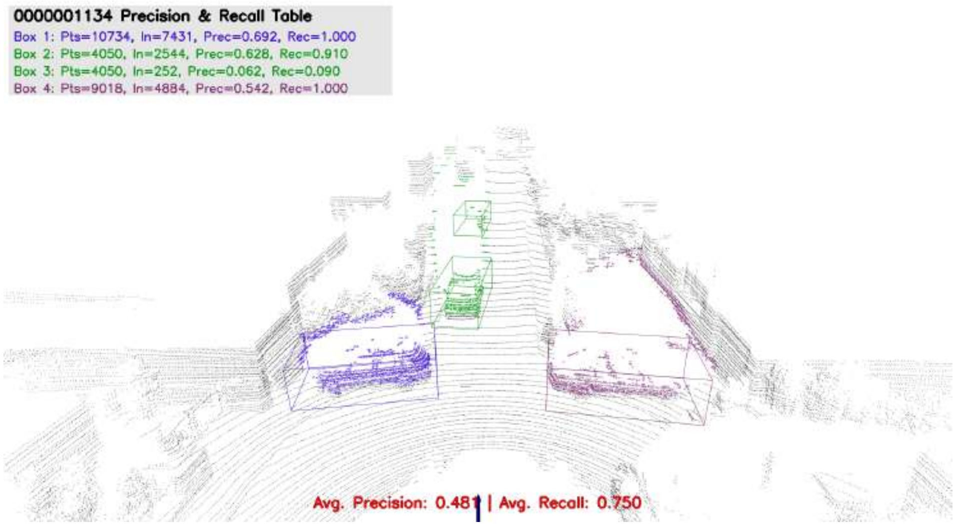
Figure 16: 2D View: 0000001134.png



```
0000000947 Precision & Recall Table
Box 1: Pts=222, In=168, Prec=0.757, Rec=1.000
Box 2: Pts=13980, In=624, Prec=0.045, Rec=0.081
Box 3: Pts=13980, In=7092, Prec=0.507, Rec=0.919
Box 4: Pts=4626, In=78, Prec=0.017, Rec=0.029
Box 5: Pts=4626, In=198, Prec=0.043, Rec=0.072
```

```
Avg. Precision: 0.317 | Avg. Recall: 0.500
```

Figure 17: 3D Point cloud: 947.png



```
0000001134 Precision & Recall Table
Box 1: Pts=10734, In=7431, Prec=0.692, Rec=1.000
Box 2: Pts=4050, In=2544, Prec=0.628, Rec=0.910
Box 3: Pts=4050, In=252, Prec=0.062, Rec=0.090
Box 4: Pts=9018, In=4884, Prec=0.542, Rec=1.000
```

```
Avg. Precision: 0.481 | Avg. Recall: 0.750
```

Figure 18: 3D Point cloud: 1134.png

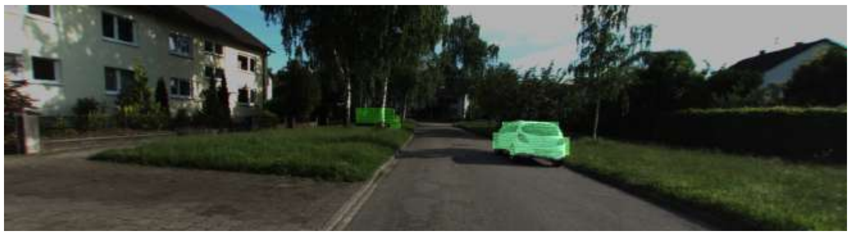


Figure 19: 2D View: 0000001280.png

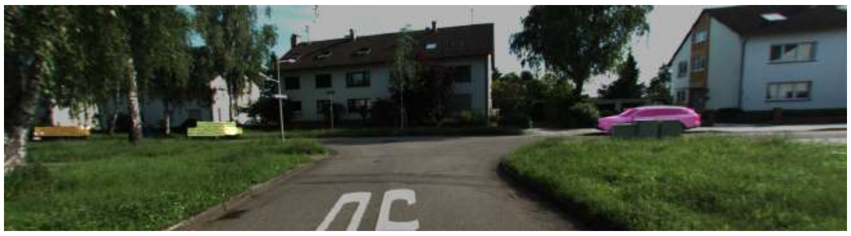
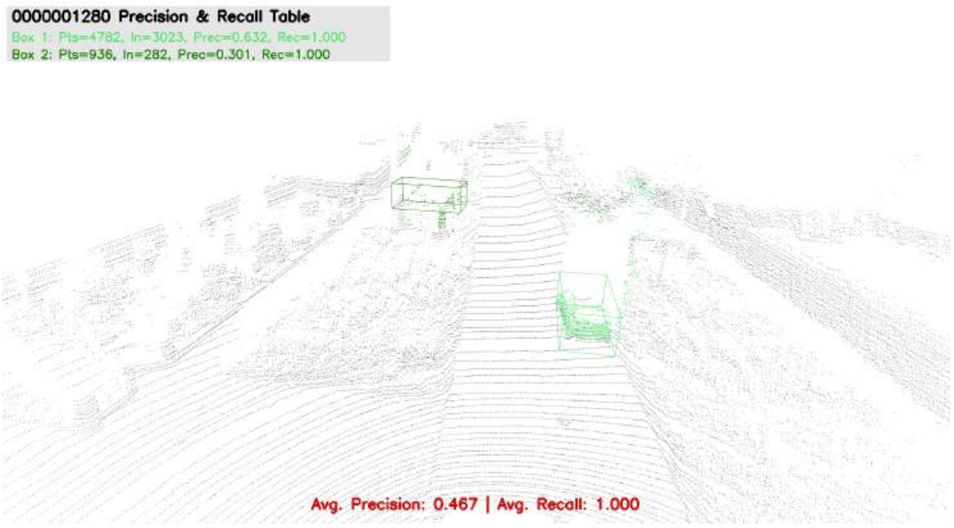


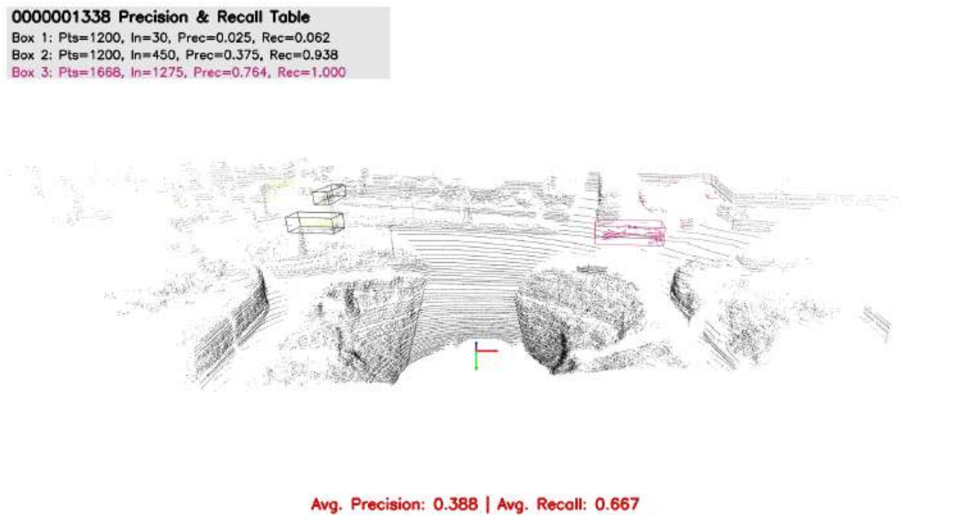
Figure 20: 2D View: 0000001338.png



```
0000001280 Precision & Recall Table
Box 1: Pts=4782, In=3023, Prec=0.632, Rec=1.000
Box 2: Pts=936, In=282, Prec=0.301, Rec=1.000
```

```
Avg. Precision: 0.467 | Avg. Recall: 1.000
```

Figure 21: 3D Point cloud: 1280.png



```
0000001338 Precision & Recall Table
Box 1: Pts=1200, In=30, Prec=0.025, Rec=0.062
Box 2: Pts=1200, In=450, Prec=0.375, Rec=0.938
Box 3: Pts=1668, In=1275, Prec=0.764, Rec=1.000
```

```
Avg. Precision: 0.388 | Avg. Recall: 0.667
```

Figure 22: 3D Point cloud: 1338.png

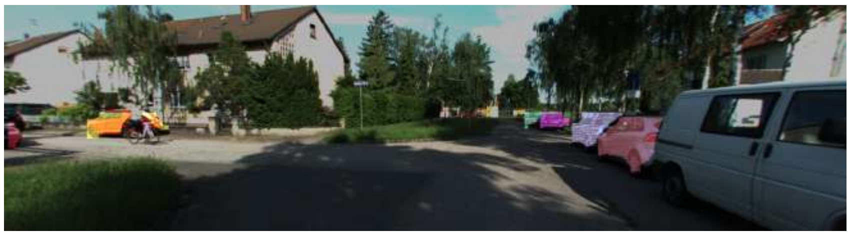


Figure 23: 2D View: 0000001461.png

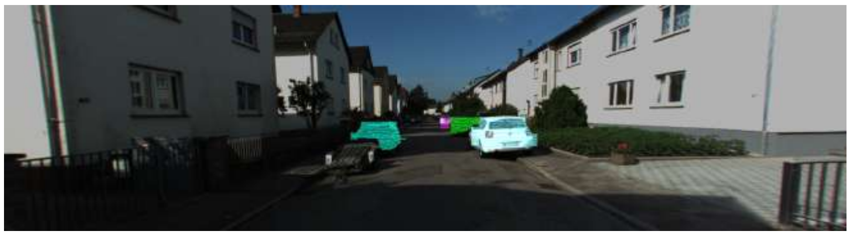
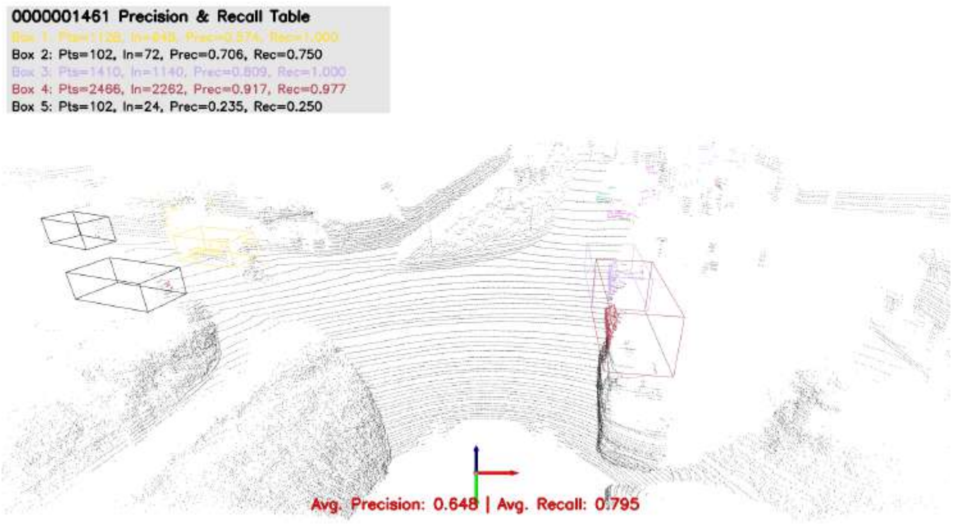


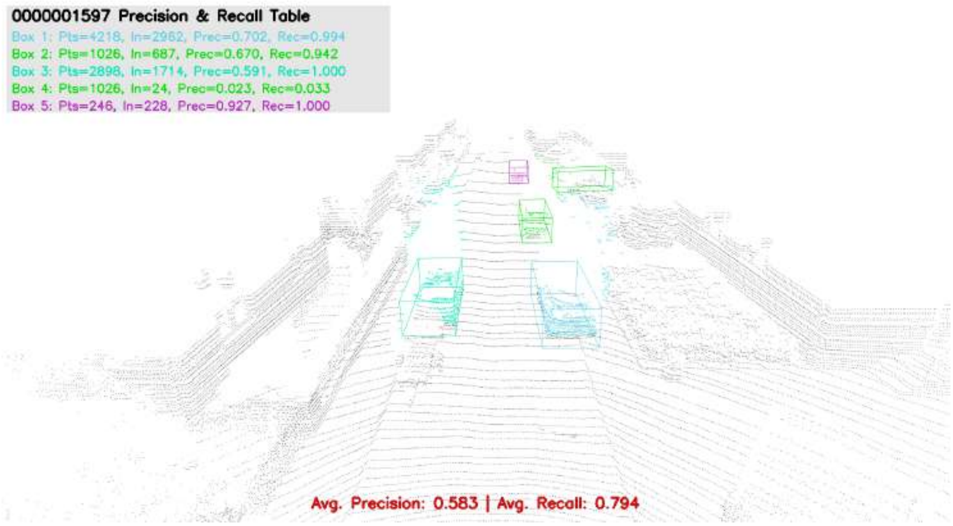
Figure 24: 2D View: 0000001597.png



```
0000001461 Precision & Recall Table
Box 1: Pts=1128, In=648, Prec=0.574, Rec=1.000
Box 2: Pts=102, In=72, Prec=0.706, Rec=0.750
Box 3: Pts=1410, In=1140, Prec=0.809, Rec=1.000
Box 4: Pts=2466, In=2262, Prec=0.917, Rec=0.977
Box 5: Pts=102, In=24, Prec=0.235, Rec=0.250
```

```
Avg. Precision: 0.648 | Avg. Recall: 0.795
```

Figure 25: 3D Point cloud: 1461.png



```
0000001597 Precision & Recall Table
Box 1: Pts=4218, In=2962, Prec=0.702, Rec=0.994
Box 2: Pts=1026, In=687, Prec=0.670, Rec=0.942
Box 3: Pts=2898, In=1714, Prec=0.591, Rec=1.000
Box 4: Pts=1026, In=24, Prec=0.023, Rec=0.033
Box 5: Pts=246, In=228, Prec=0.927, Rec=1.000
```

```
Avg. Precision: 0.583 | Avg. Recall: 0.794
```

Figure 26: 3D Point cloud: 1597.png

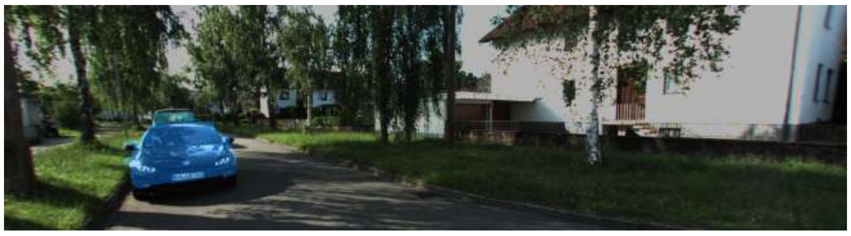


Figure 27: 2D View: 0000001791.png

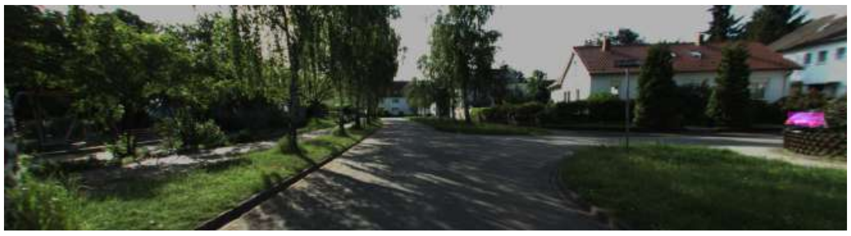


Figure 28: 2D View: 0000001866.png

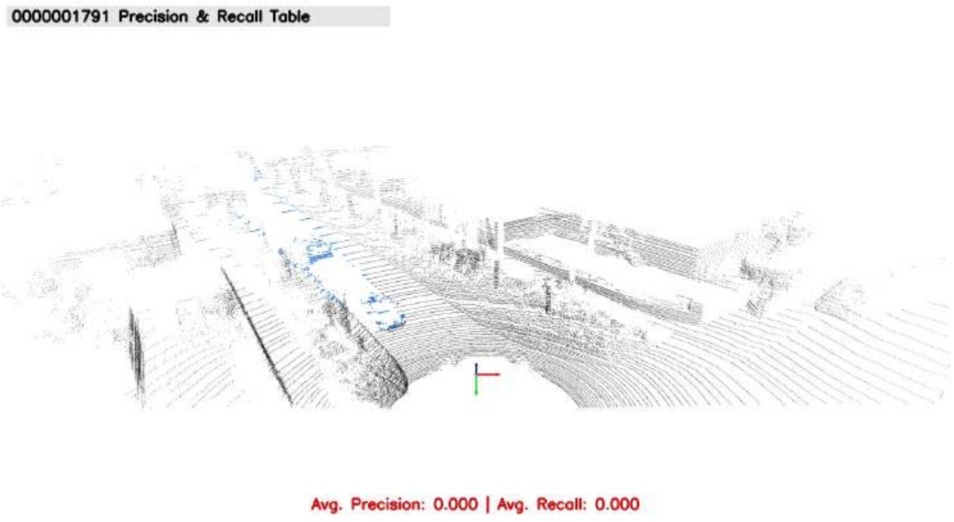

0000001791 Precision & Recall Table
Avg. Precision: 0.000 | Avg. Recall: 0.000

Figure 29: 3D Point cloud: 1791.png

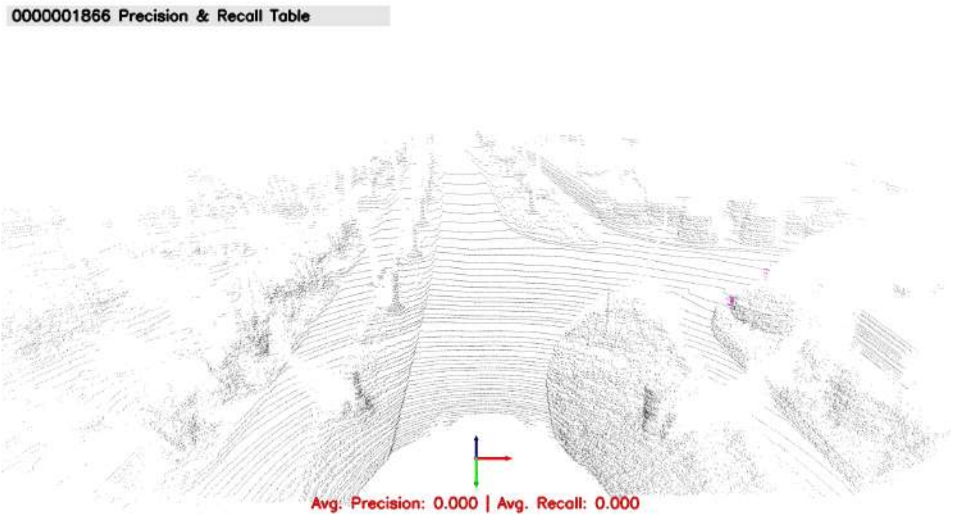

0000001866 Precision & Recall Table
Avg. Precision: 0.000 | Avg. Recall: 0.000

Figure 30: 3D Point cloud: 1866.png

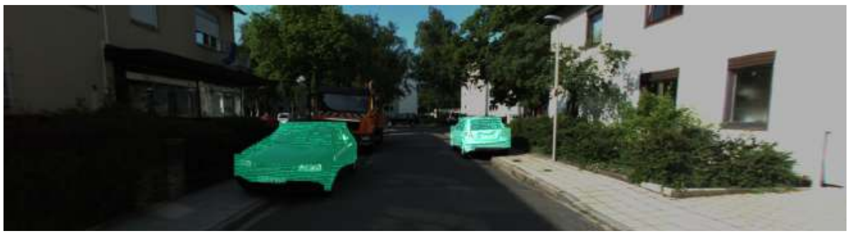


Figure 31: 2D View: 0000002033.png

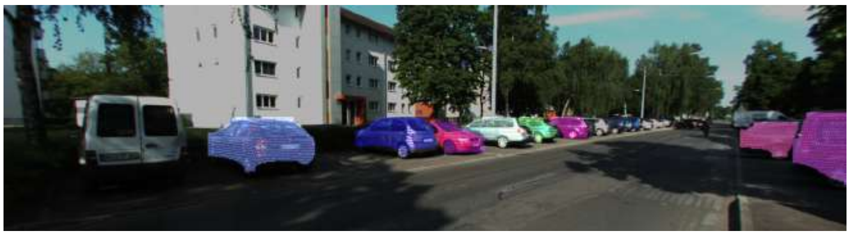
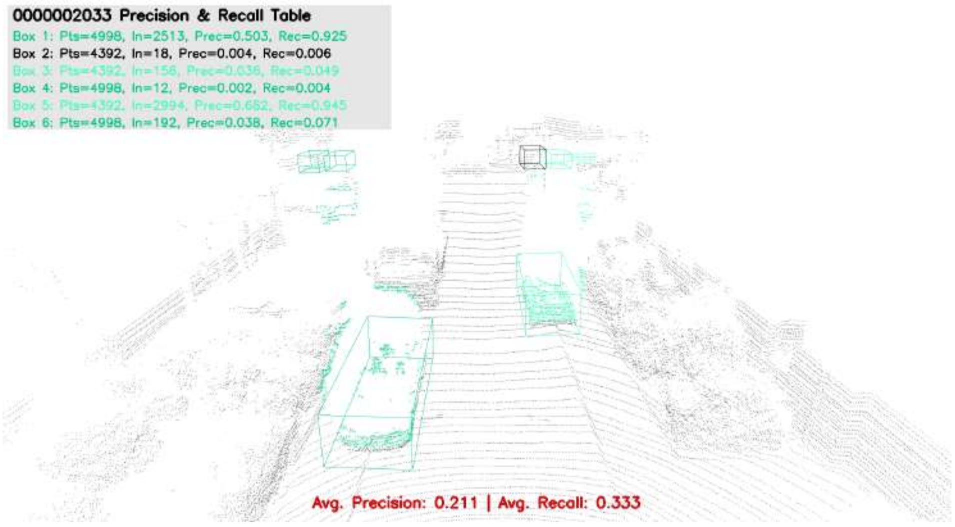


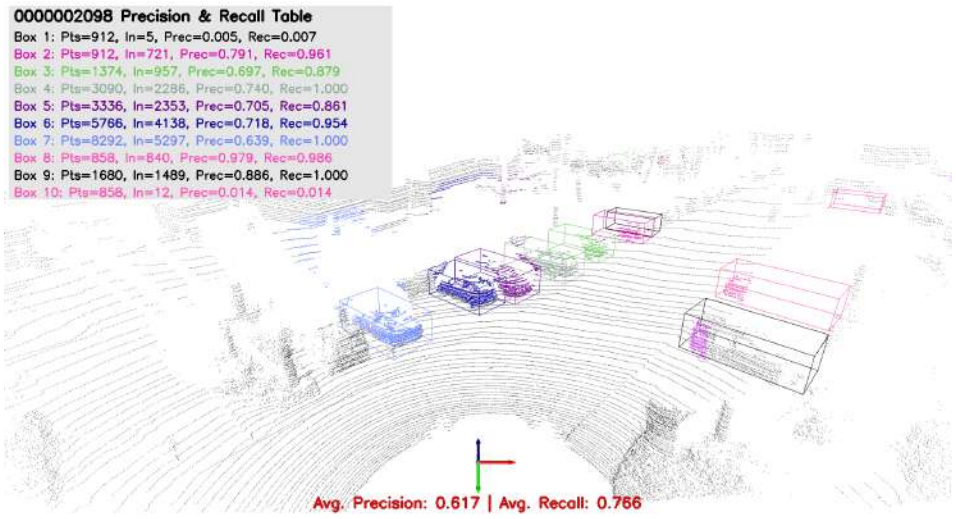
Figure 32: 2D View: 0000002098.png



```
0000002033 Precision & Recall Table
Box 1: Pts=4998, In=2513, Prec=0.503, Rec=0.925
Box 2: Pts=4392, In=18, Prec=0.004, Rec=0.006
Box 3: Pts=4392, In=156, Prec=0.036, Rec=0.049
Box 4: Pts=4998, In=12, Prec=0.002, Rec=0.004
Box 5: Pts=4392, In=2994, Prec=0.682, Rec=0.945
Box 6: Pts=4998, In=192, Prec=0.038, Rec=0.071
```

Avg. Precision: 0.211 | Avg. Recall: 0.333

Figure 33: 3D Point cloud: 2033.png



```
0000002098 Precision & Recall Table
Box 1: Pts=912, In=5, Prec=0.005, Rec=0.007
Box 2: Pts=912, In=721, Prec=0.791, Rec=0.961
Box 3: Pts=1374, In=957, Prec=0.697, Rec=0.879
Box 4: Pts=3090, In=2286, Prec=0.740, Rec=1.000
Box 5: Pts=3336, In=2353, Prec=0.705, Rec=0.861
Box 6: Pts=5766, In=4138, Prec=0.718, Rec=0.954
Box 7: Pts=8292, In=5297, Prec=0.639, Rec=1.000
Box 8: Pts=858, In=840, Prec=0.979, Rec=0.986
Box 9: Pts=1680, In=1489, Prec=0.886, Rec=1.000
Box 10: Pts=858, In=12, Prec=0.014, Rec=0.014
```

Avg. Precision: 0.617 | Avg. Recall: 0.766

Figure 34: 3D Point cloud: 2098.png

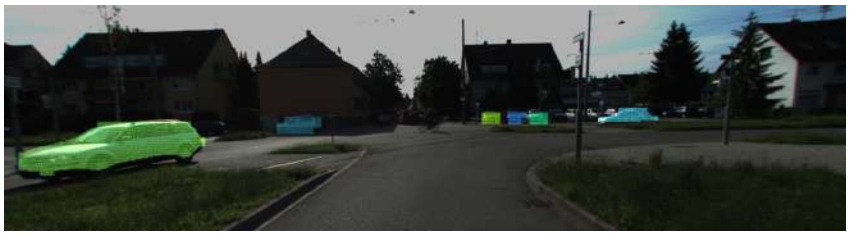


Figure 35: 2D View: 0000002449.png

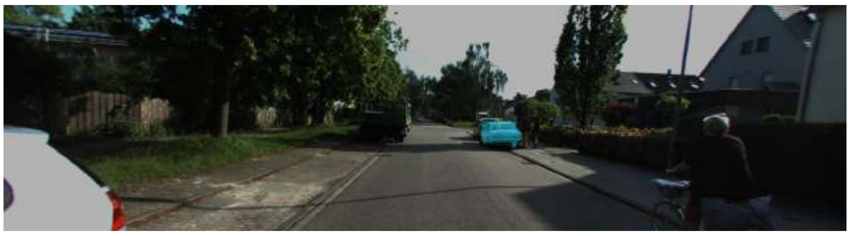
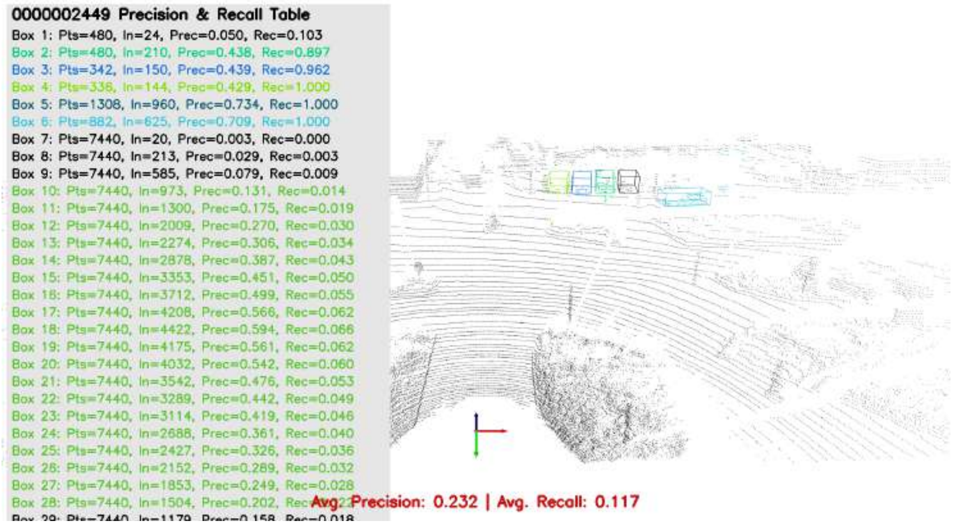
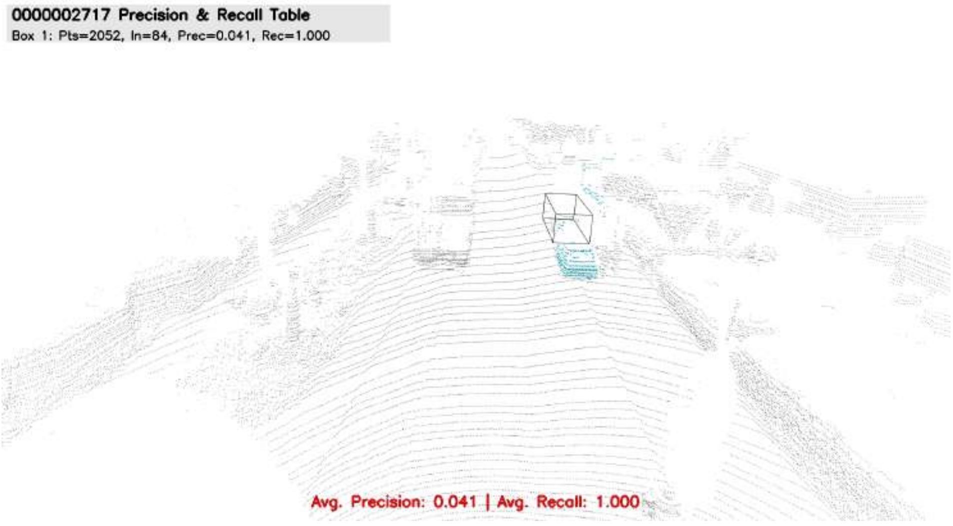


Figure 36: 2D View: 0000002717.png



```
0000002449 Precision & Recall Table
Box 1: Pts=480, In=24, Prec=0.050, Rec=0.103
Box 2: Pts=480, In=210, Prec=0.438, Rec=0.897
Box 3: Pts=342, In=150, Prec=0.439, Rec=0.962
Box 4: Pts=336, In=144, Prec=0.429, Rec=1.000
Box 5: Pts=1308, In=960, Prec=0.734, Rec=1.000
Box 6: Pts=882, In=625, Prec=0.709, Rec=1.000
Box 7: Pts=7440, In=20, Prec=0.003, Rec=0.000
Box 8: Pts=7440, In=213, Prec=0.029, Rec=0.003
Box 9: Pts=7440, In=585, Prec=0.079, Rec=0.009
Box 10: Pts=7440, In=973, Prec=0.131, Rec=0.014
Box 11: Pts=7440, In=1300, Prec=0.175, Rec=0.019
Box 12: Pts=7440, In=2009, Prec=0.270, Rec=0.030
Box 13: Pts=7440, In=2274, Prec=0.306, Rec=0.034
Box 14: Pts=7440, In=2878, Prec=0.387, Rec=0.043
Box 15: Pts=7440, In=3353, Prec=0.451, Rec=0.050
Box 16: Pts=7440, In=3712, Prec=0.499, Rec=0.055
Box 17: Pts=7440, In=4208, Prec=0.566, Rec=0.062
Box 18: Pts=7440, In=4422, Prec=0.594, Rec=0.066
Box 19: Pts=7440, In=4175, Prec=0.561, Rec=0.062
Box 20: Pts=7440, In=4032, Prec=0.542, Rec=0.060
Box 21: Pts=7440, In=3542, Prec=0.476, Rec=0.053
Box 22: Pts=7440, In=3289, Prec=0.442, Rec=0.049
Box 23: Pts=7440, In=3114, Prec=0.419, Rec=0.046
Box 24: Pts=7440, In=2688, Prec=0.361, Rec=0.040
Box 25: Pts=7440, In=2427, Prec=0.326, Rec=0.036
Box 26: Pts=7440, In=2152, Prec=0.289, Rec=0.032
Box 27: Pts=7440, In=1853, Prec=0.249, Rec=0.028
Box 28: Pts=7440, In=1504, Prec=0.202, Rec
Box 29: Pts=7440, In=1179, Prec=0.158, Rec=0.018
Avg. Precision: 0.232 | Avg. Recall: 0.117
```

Figure 37: 3D Point cloud: 2449.png



```
0000002717 Precision & Recall Table
Box 1: Pts=2052, In=84, Prec=0.041, Rec=1.000
Avg. Precision: 0.041 | Avg. Recall: 1.000
```

Figure 38: 3D Point cloud: 2717.png

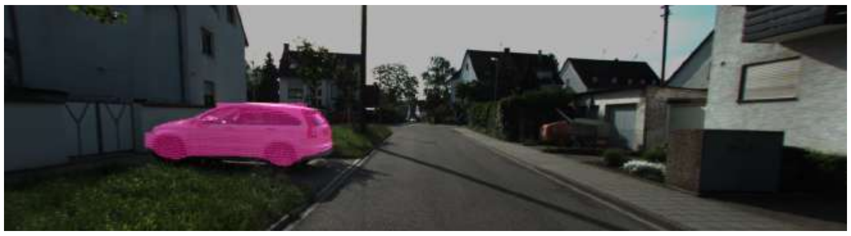


Figure 39: 2D View: 0000002903.png

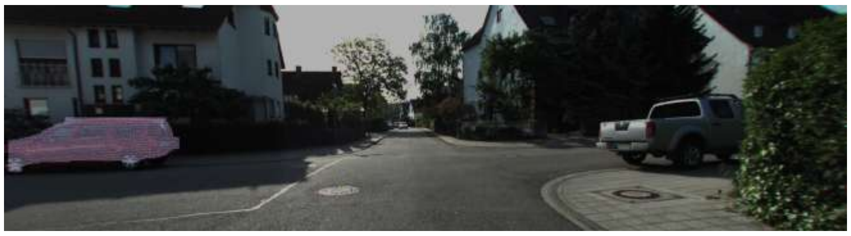
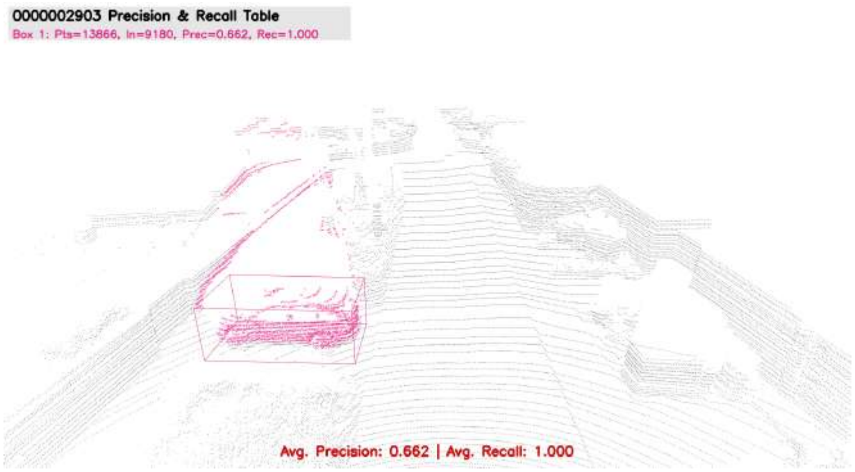


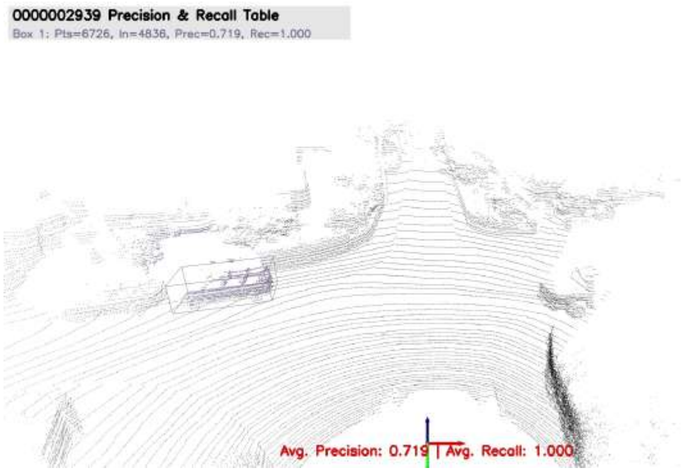
Figure 40: 2D View: 0000002939.png



```
0000002903 Precision & Recall Table
Box 1: Pts=13866, In=9180, Prec=0.662, Rec=1.000
```

```
Avg. Precision: 0.662 | Avg. Recall: 1.000
```

Figure 41: 3D Point cloud: 2903.png



```
0000002939 Precision & Recall Table
Box 1: Pts=6726, In=4836, Prec=0.719, Rec=1.000
```

```
Avg. Precision: 0.719 | Avg. Recall: 1.000
```

Figure 42: 3D Point cloud: 2939.png

4 Evaluation

To evaluate the result, we will use the relative amount of points. Since point density depends on object distance and occlusion, absolute counts can be misleading. so precision and recall normalize these differences, providing a fairer quality measure. As a result we will evaluate this model based on the precision and recall values, obtained from the YOLOv8x-seg (1) model.

Detection Category	Precision	Count
Total number of detections		110
Total number of good detections	(Precision > 0.6)	34
Total number of poor detections	($0.6 >$ Precision < 0.35)	25
Total number of bad detections	(Precision < 0.35)	51

Table 1: Summary of detection evaluation results based on precision thresholds.

4.1 Precision Threshold Justification

If the value of precision is greater than 0.6, we considered it as a correct detection. This threshold ensures that the majority of the points inside the predicted 3D bounding box truly belong to the car, while allowing a small margin of error due to real-world factors such as noise, occlusions, or segmentation inaccuracies. A precision above 0.6 strikes a practical balance between strict accuracy and robustness, making the detection reliable enough for downstream tasks such as tracking or decision-making.

4.2 Image specific errors

4.2.1 Point Cloud Color Leakage into Multiple Boxes

In this scenes [Image Ids: 947(17), 1134(18), 1461(25), 1597(26), 2033(33), 2098(34), 2449(37)], the same color (i.e., segmented car) point cloud was partially spread across two or more bounding boxes. If the second bounding box contains only leaked points from the first car, it gets falsely counted as TP for both boxes, which leads to artificially inflated precision for incorrect boxes. This occurs because the segmentation color (representing car ID) is not uniquely tied to a single bounding box.
A potential solution is to modify the evaluation logic: if two boxes share the same color, the system should count those points only for the box that contains the majority of that color's points. In this way we can encounter this problem. see image (43) and (44)

4.2.2 Missing Ground Truth Bounding Boxes

In this scenes [Image ids:360(9), 1461(25), 1791 (29), 2712(38)], the segmented cars were visible and colored, but corresponding ground truth bounding boxes were absent. As a result, no precision or recall was calculated for those instances.

4.2.3 Too many GT boxes on one car

In the image Id 2449(37), we observed one problem, which was the presence of multiple ground truth (GT) bounding boxes overlapping a single physical car. As a result, the model interpreted this as multiple distinct cars and generated several detections in the same area. However, these detections corresponded to only one actual vehicle. we have solve this problem with the help of above proposed solution (in section 4.2.1).

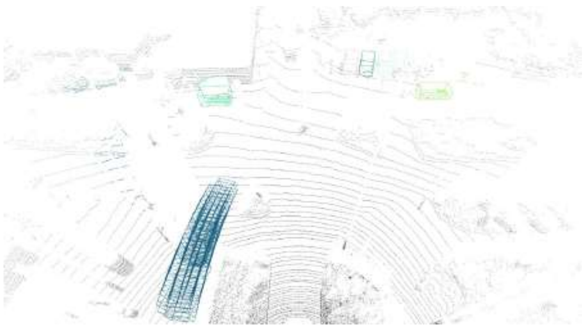
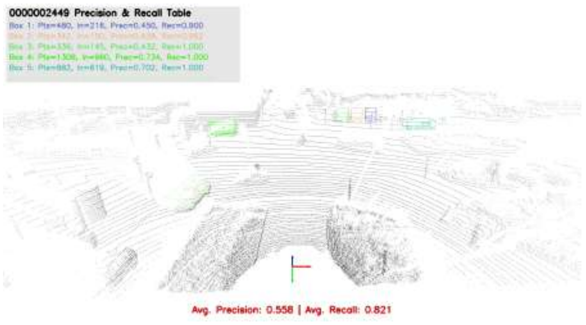


Figure 43: Too many BB on one car



0000002449 Precision & Recall Table
Box 1: Pts=480, In=216, Prec=0.450, Rec=0.900
[illegible]
Box 3: Pts=336, In=145, Prec=0.432, Rec=1.000
Box 4: Pts=1306, In=960, Prec=0.734, Rec=1.000
Box 5: Pts=882, In=619, Prec=0.702, Rec=1.000

Avg. Precision: 0.558 | Avg. Recall: 0.821

Figure 44: updated result

5 Conclusion

The proposed solution demonstrates promising results; however, several limitations were identified, as detailed in the image-specific error section. These errors impact the overall reliability of the solution. with further work and error correction, the method has potential for real-time applications. However, at its current stage, the system is not yet reliable enough for deployment in real-world automotive environments.

6 References

1. YOLOv8x-seg "Accessed June 13, 2025" https://docs.ultralytics.com/models/yolov8/overview

2. KITTI 360 : https://www.cvlibs.net/datasets/kitti-360/index.php

3. KIITI 360 SUB Data set https://elearning.rwu.de/mod/resource/view.php?id=213354