Point Cloud Clustering

Shubham Suresh Patil
Carnegie Mellon University

Task : To design a system that accepts as input a 3D point cloud and outputs a segmented point cloud. Every point cloud cluster has to be represented by a bounding box.

Pre-Requisites: For data loading and visualization, I have used Open3D [link]. You can install Open3D using the following

pip3 install open3d-python

Dataset : I have used KITTI dataset (sequence 2011_09_26_drive_0052) frames as sample

Why KITTI?
Frankly, I wanted the coding assignment to be as close to the work being done at luminar. KITTI data set is a great resource in understanding the actual perception through sensors especially LiDAR. I wanted to broaden my knowledge on the same and saw this assignment as a great learning opportunity.

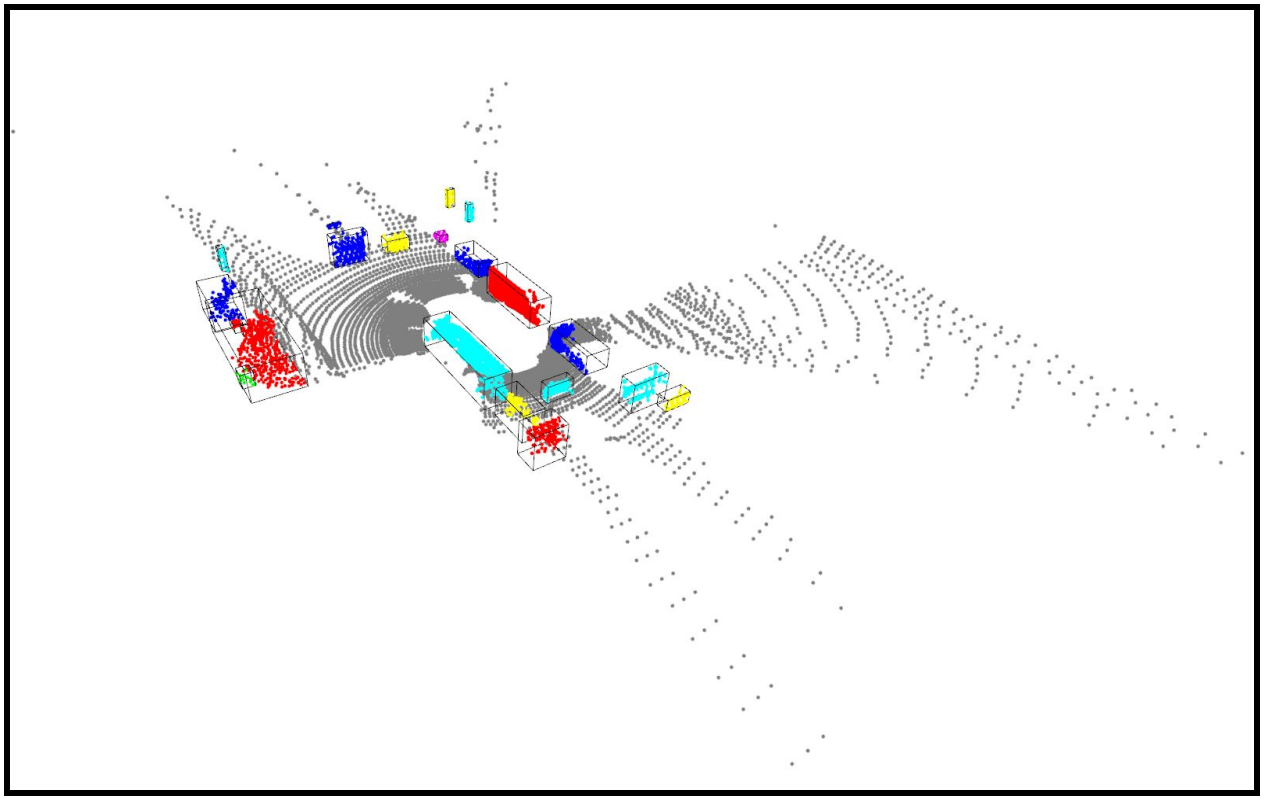
Methodology

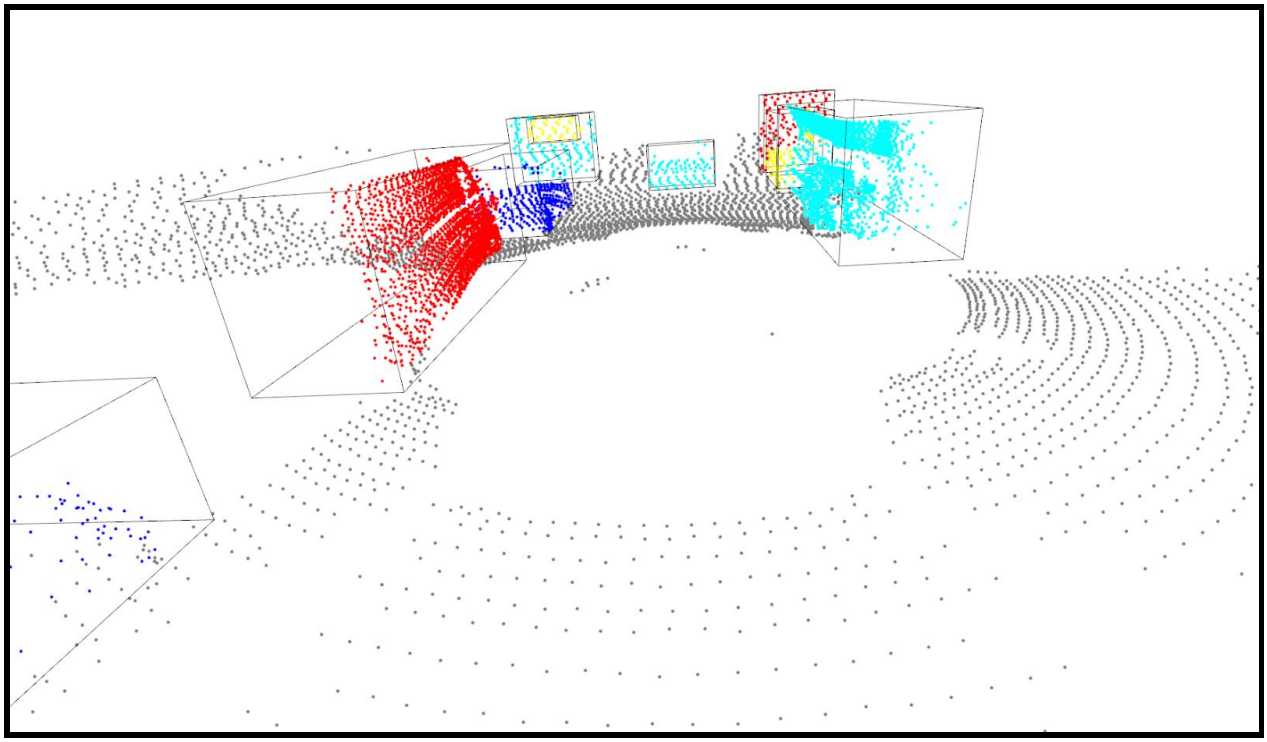
1. Extract the ground plane out of the point cloud as it might interfere with the clustering and segmentation. visualization. The ground plane is assumed to be at 1.5 m below the LiDAR source.
2. Segment the point cloud data using DBSCAN Method. To make the clustering more efficient, kd Tree, a spatial partitioning data-structure was implemented.
3. On getting the clusters, we determine the maximum and minimum coordinates to fit a cube around the cluster.
4. Append the ground plane removed from step 1 back with the segmented data for visualization.

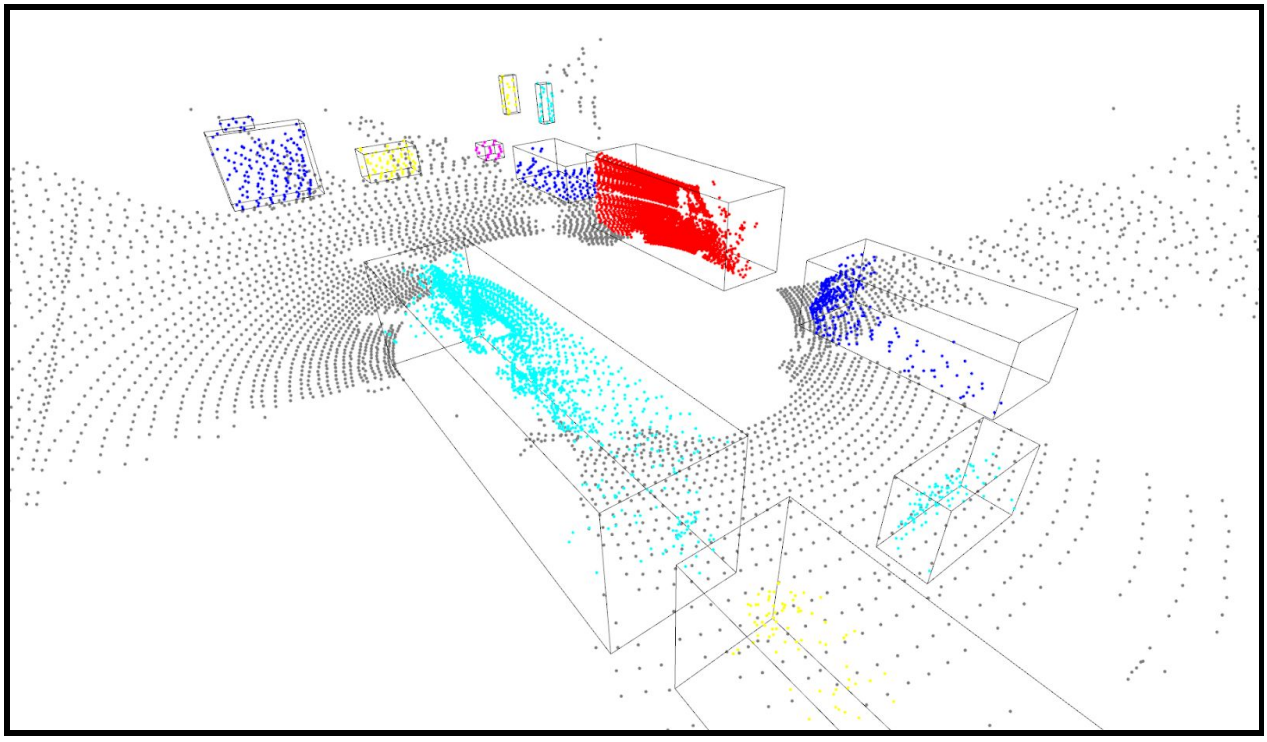
Note:
I have some issues with integrating kdtree with DBSCAN. At unit level, kdtree is working fine but I ran out of time for integrating it properly with DBSCAN, I understood the problem with clustering huge point clouds (50,000 plus) and hence moved to kdtree for spatial clustering. Earlier I was using a brute search which was giving good results for less number of points (~1000). I have used the SKlearn kdTree for the task. I will be working on it in my spare time to make it work. I am sorry for not abiding by the instructions

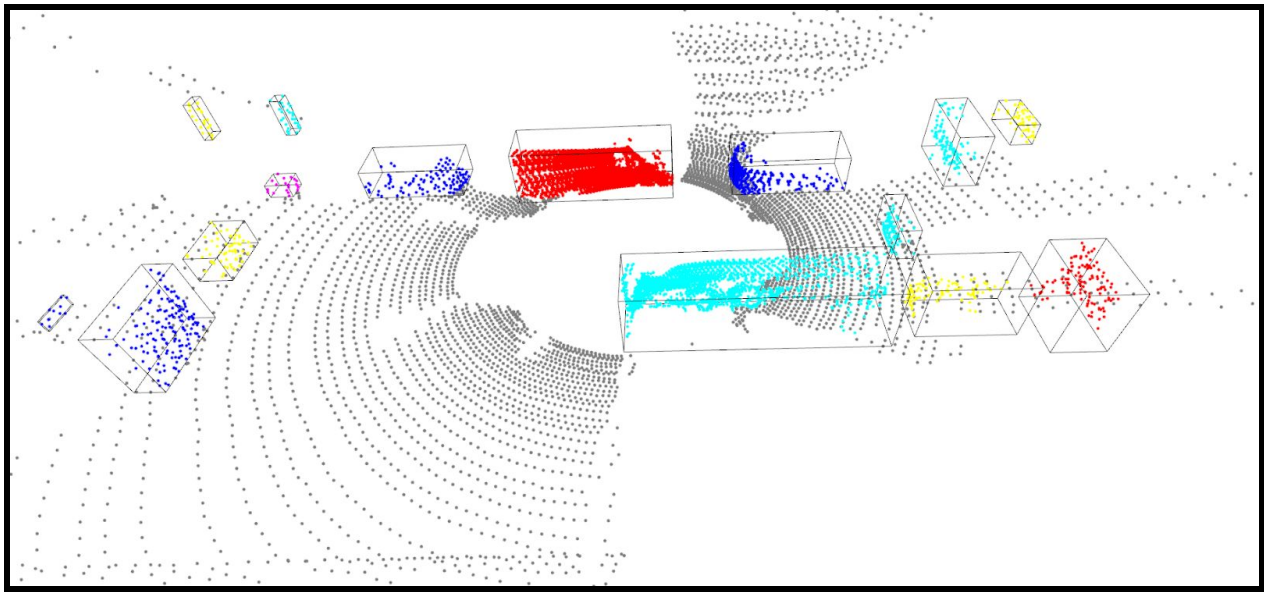
Results:

Frame 1

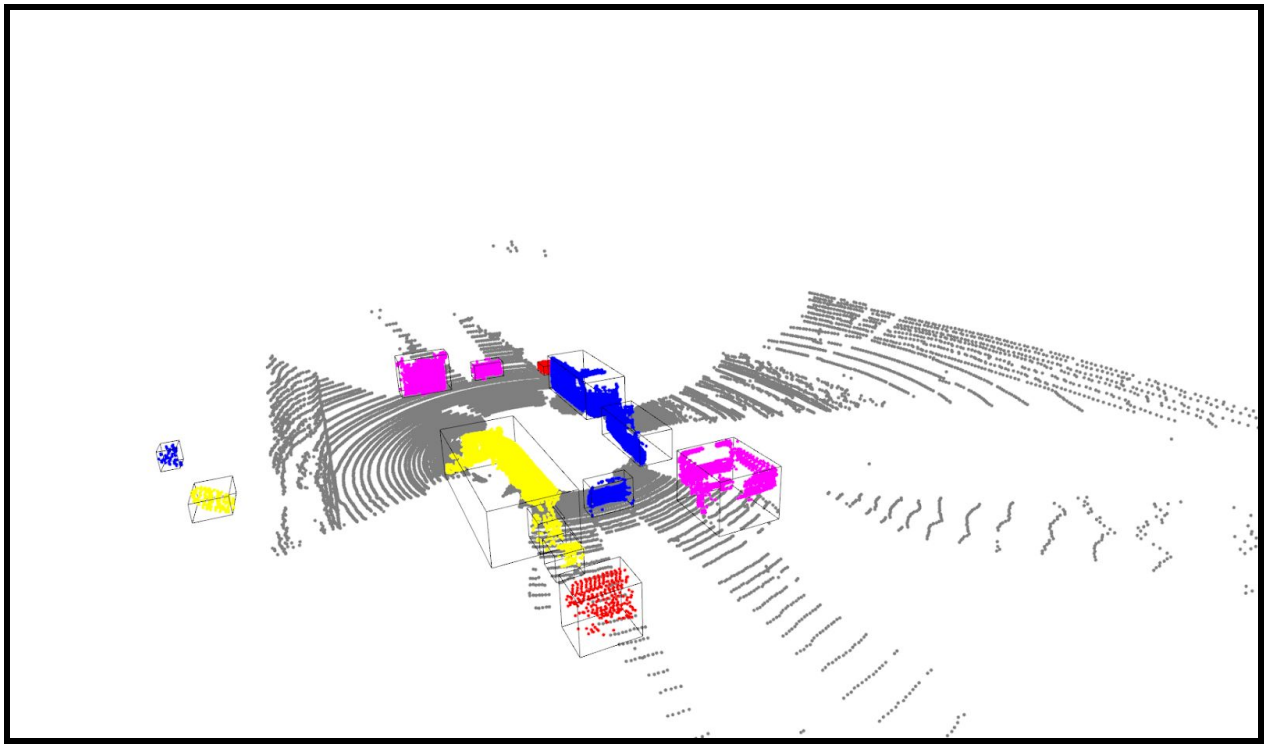


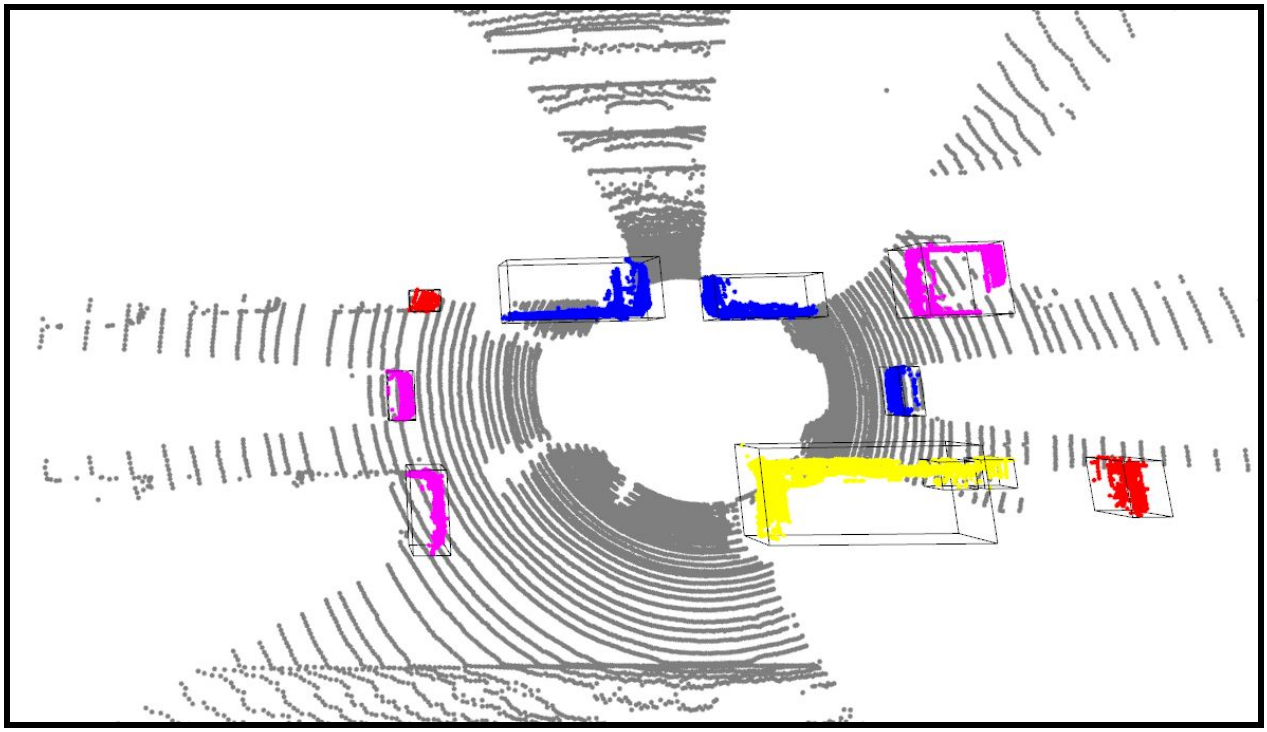




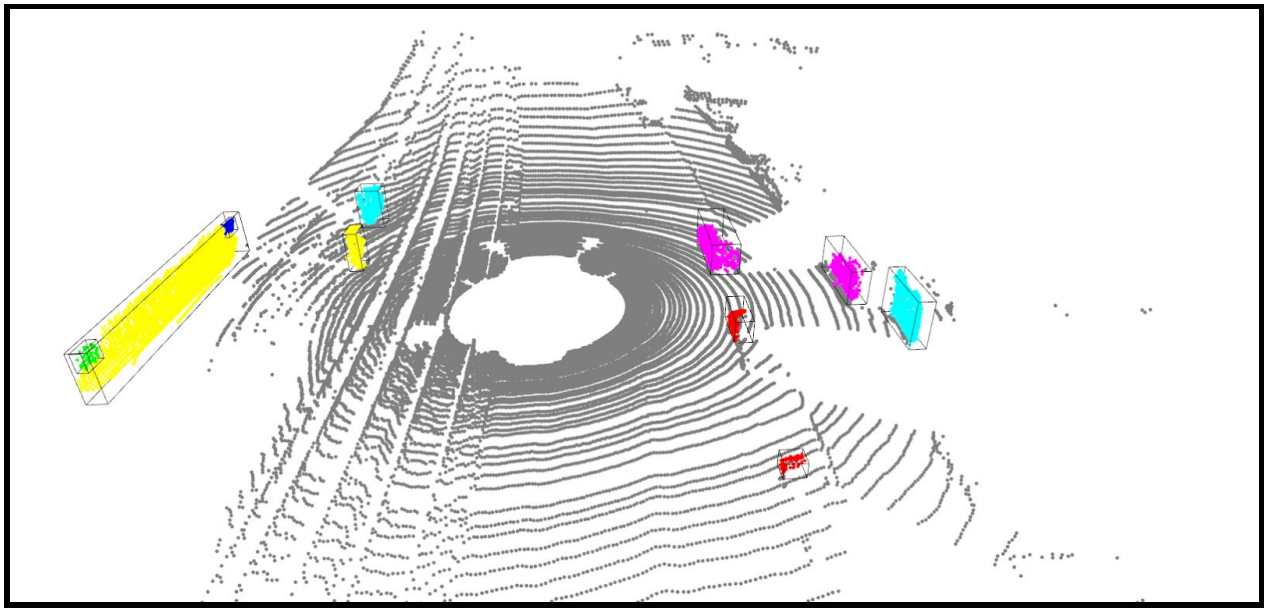


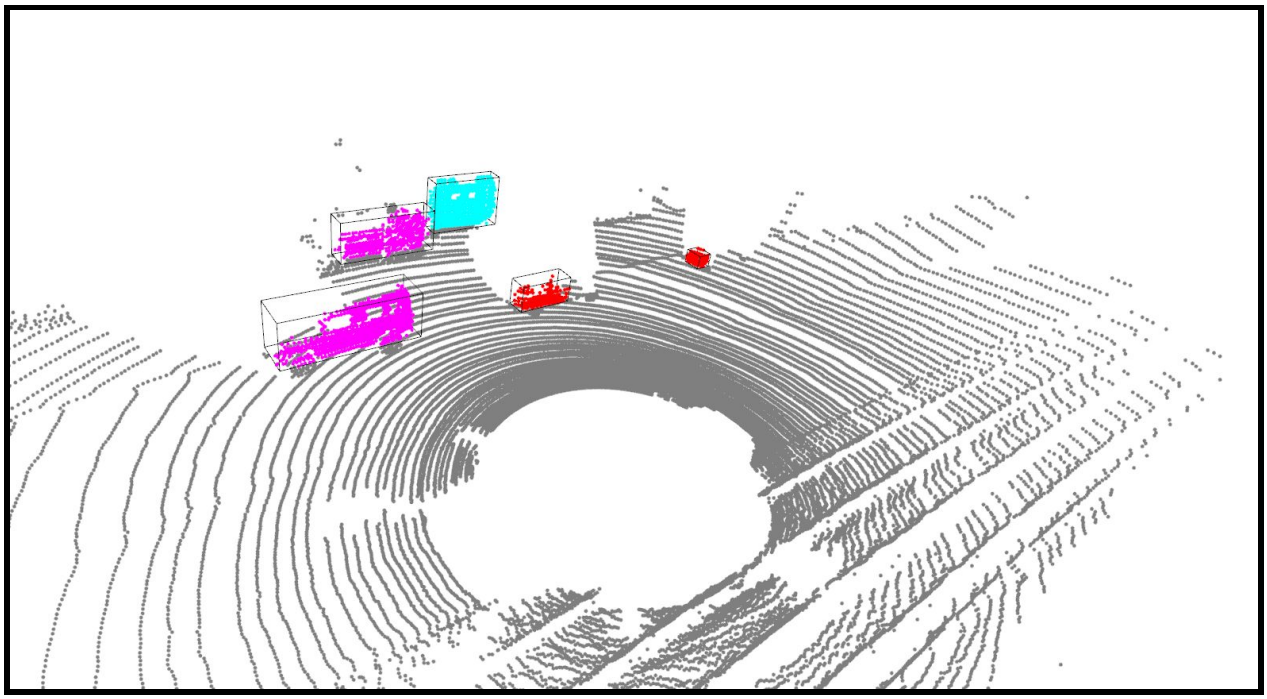
Frame 2 (23 bin)



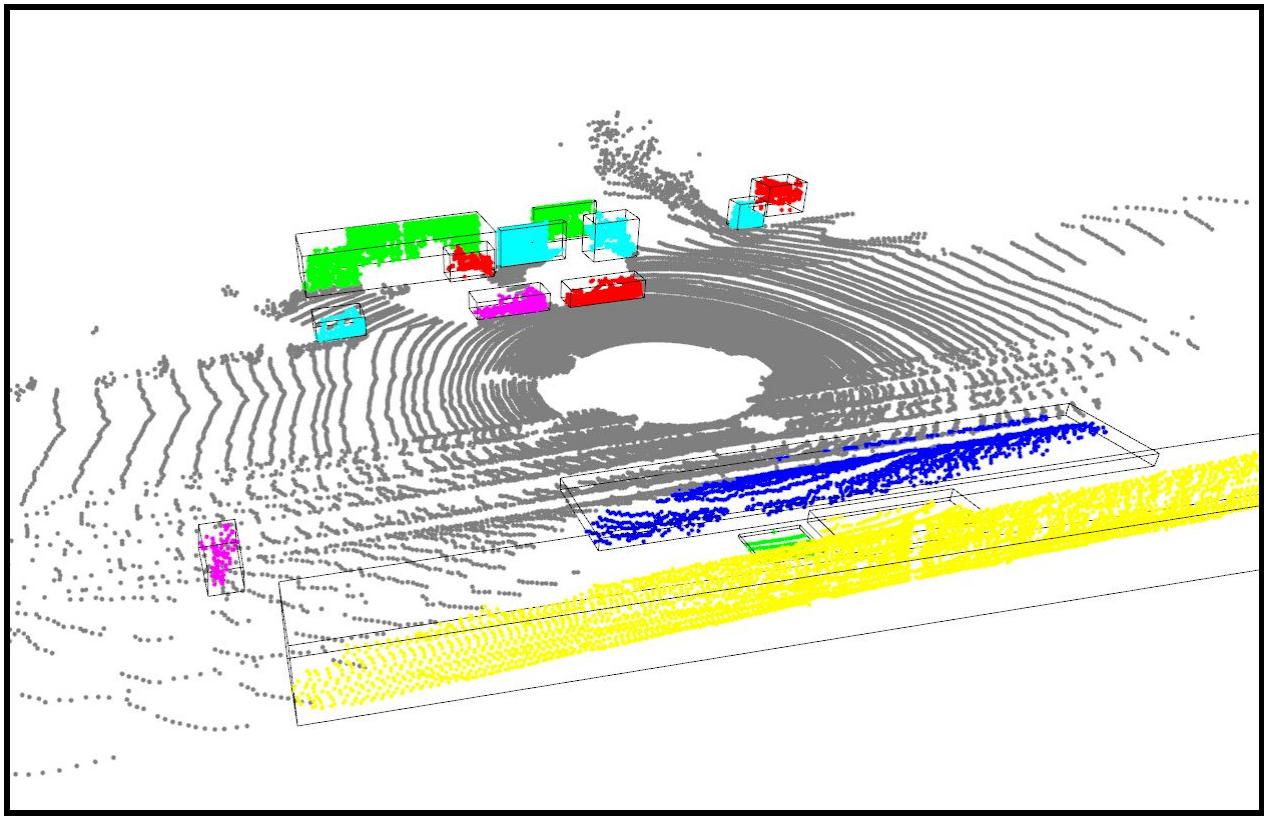


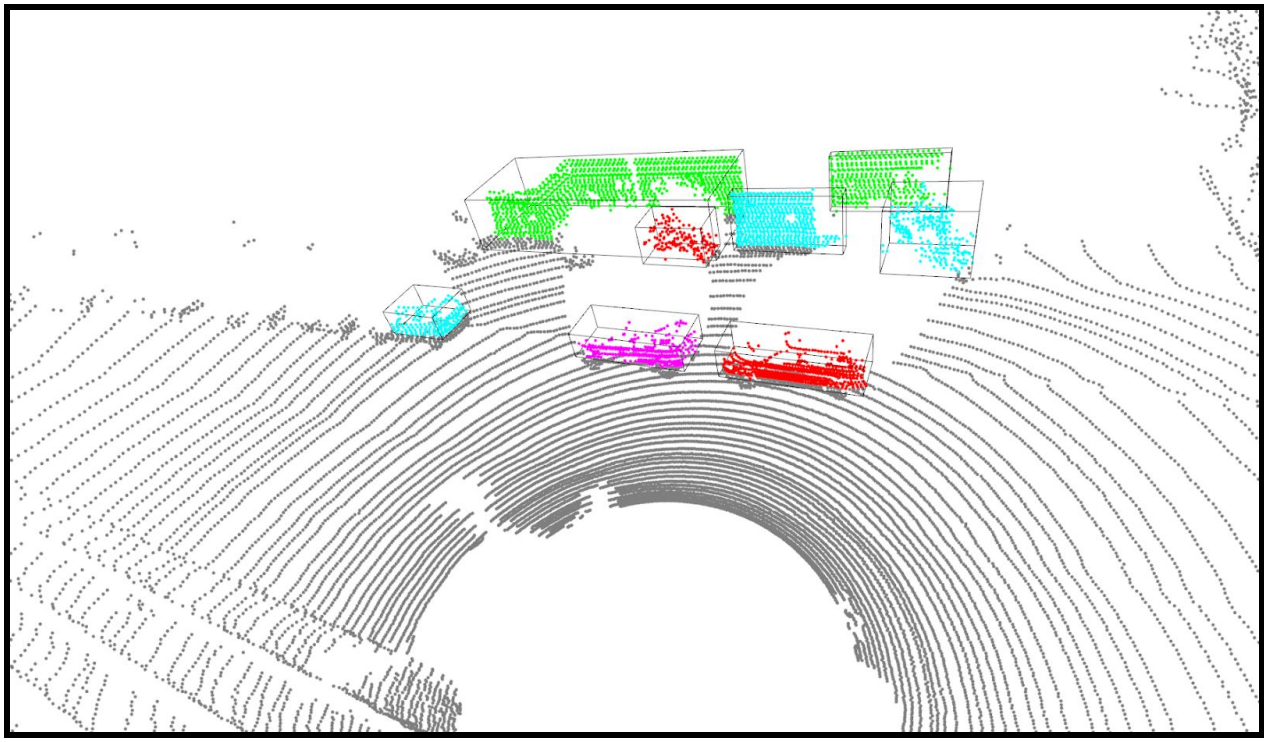
Frame 3:



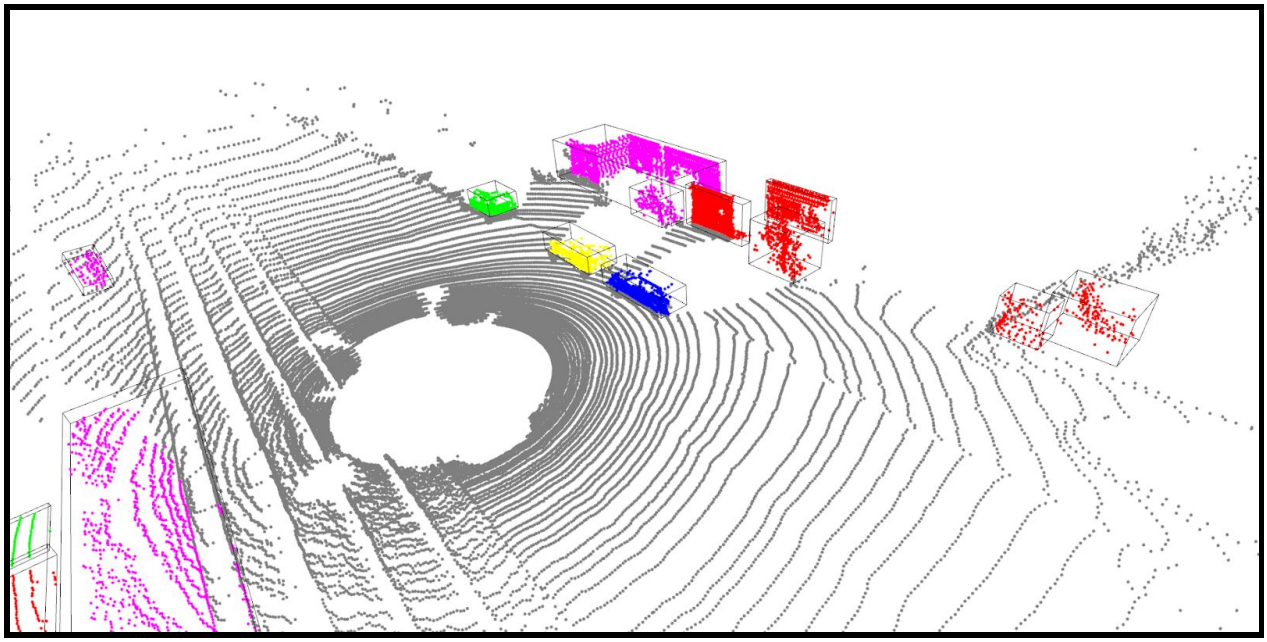


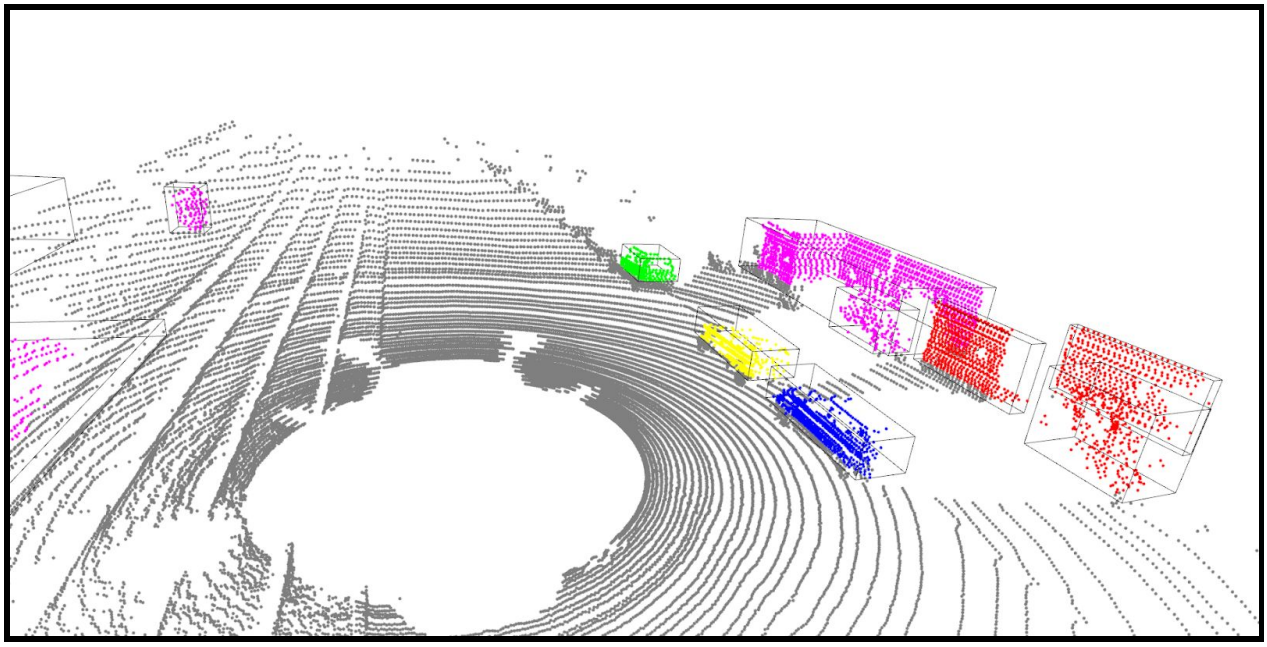
Frame 4:



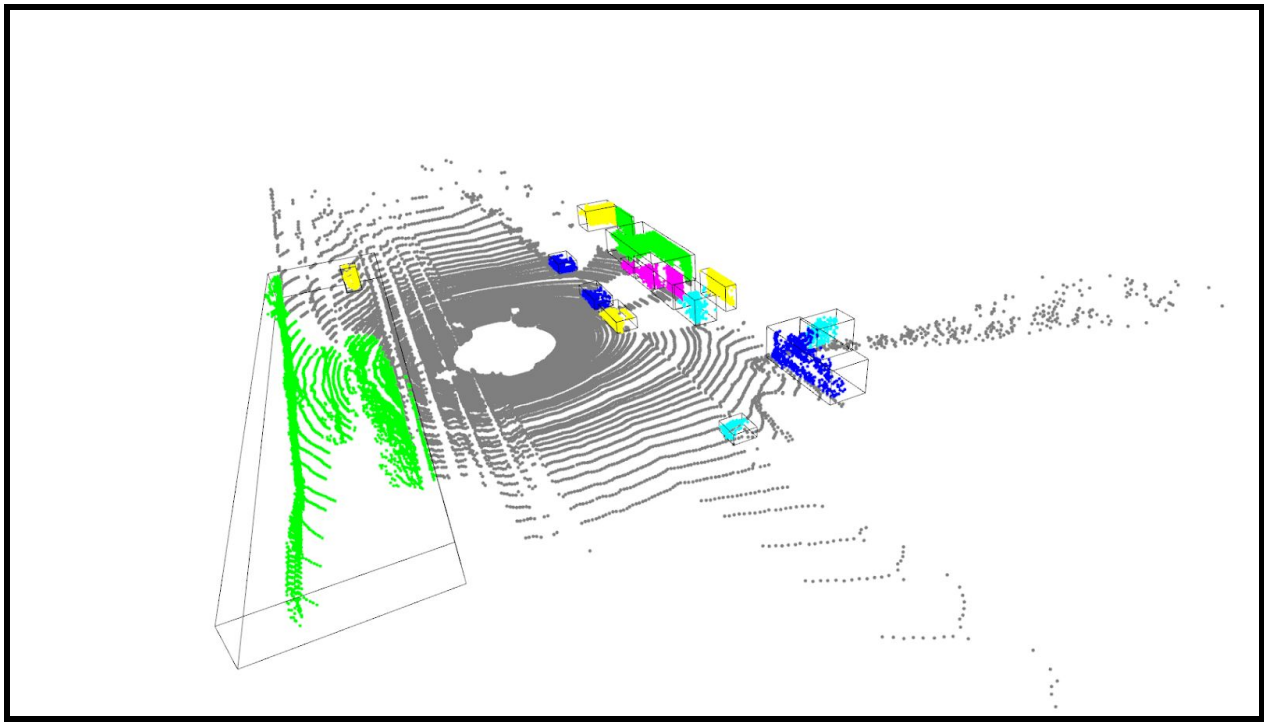


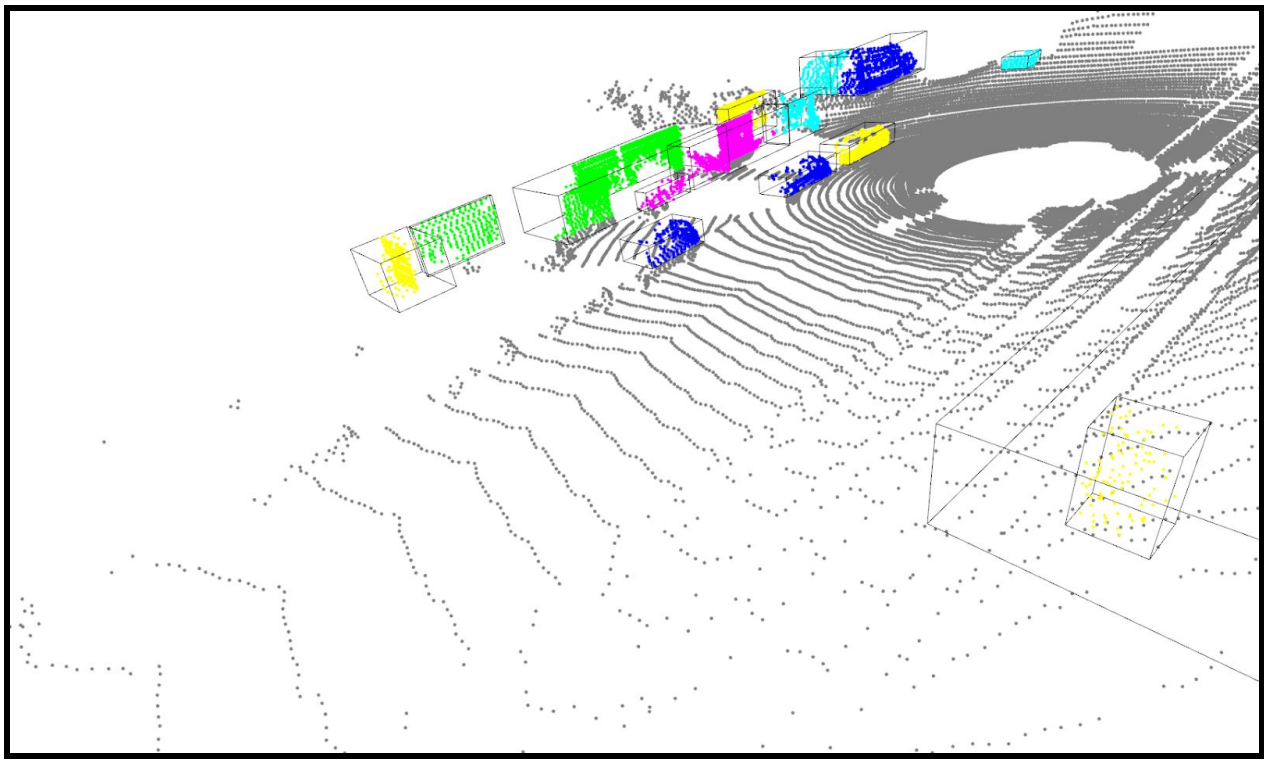
Frame 5:



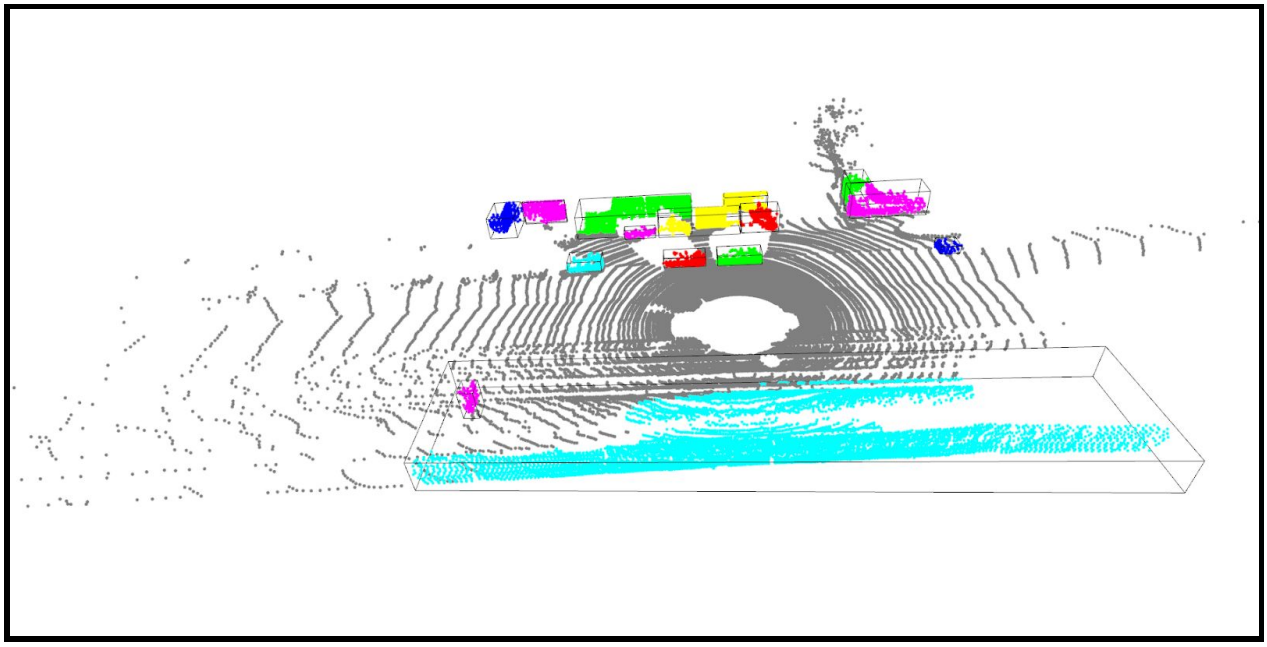


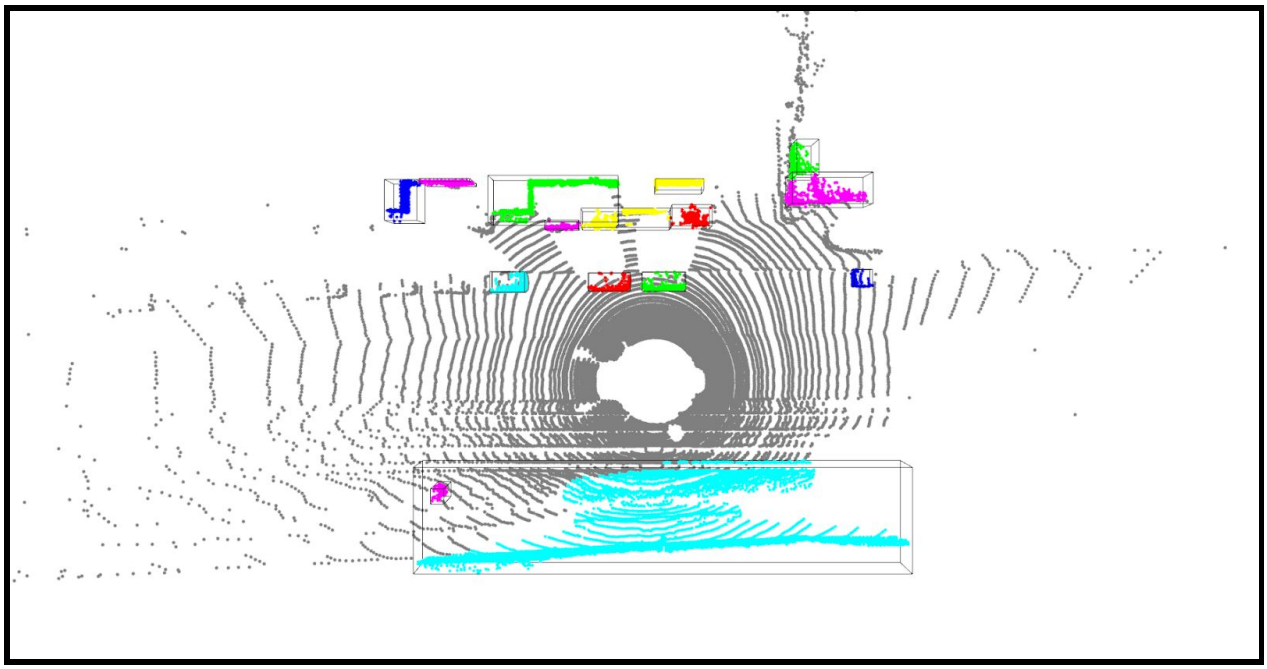
Frame 6

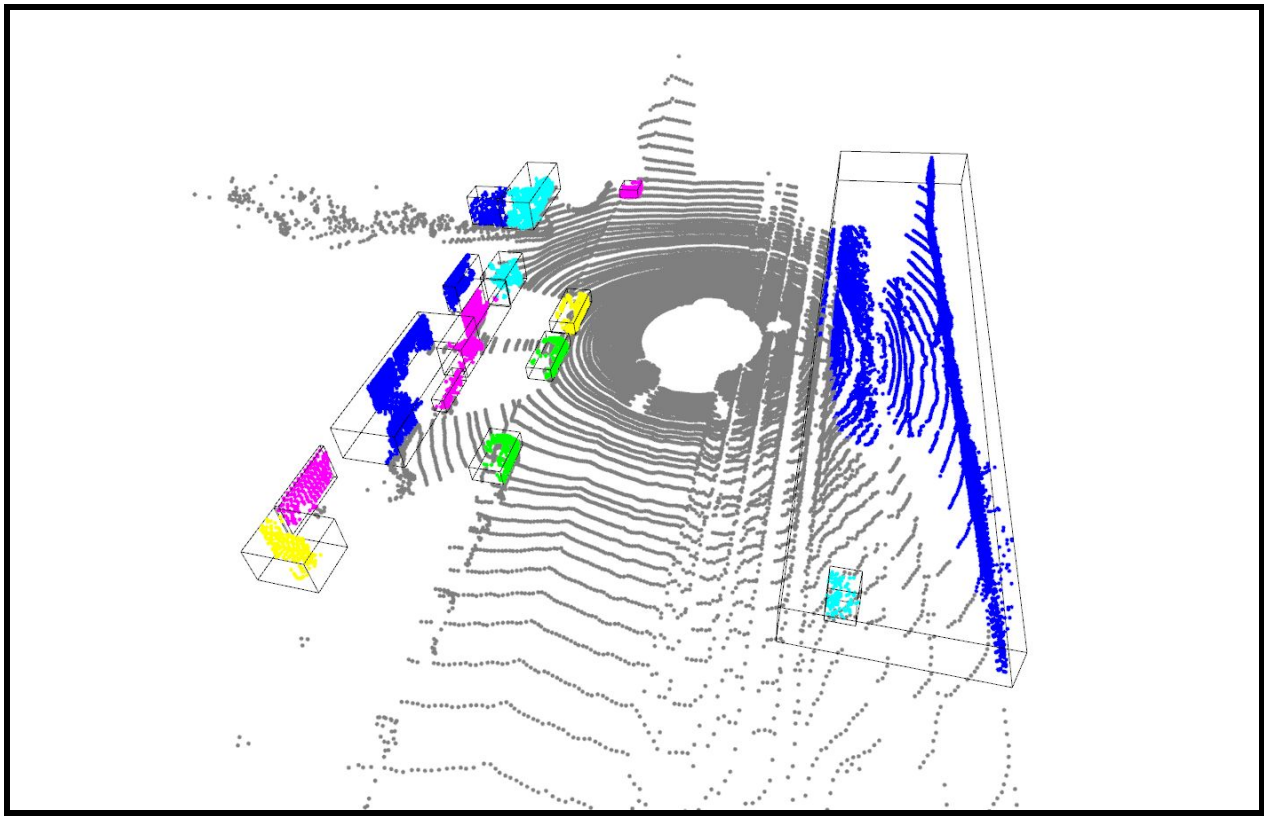




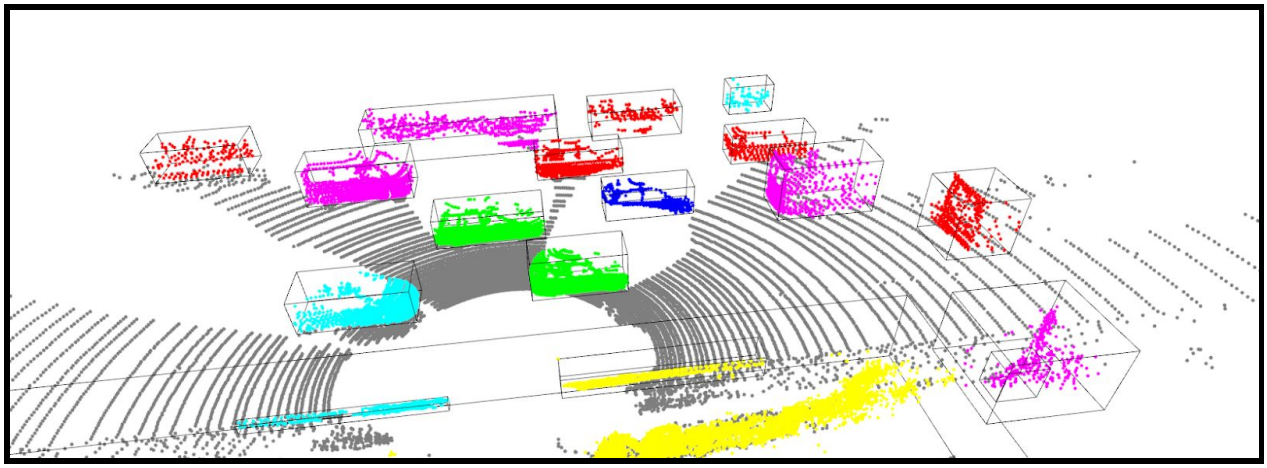
Frame 7

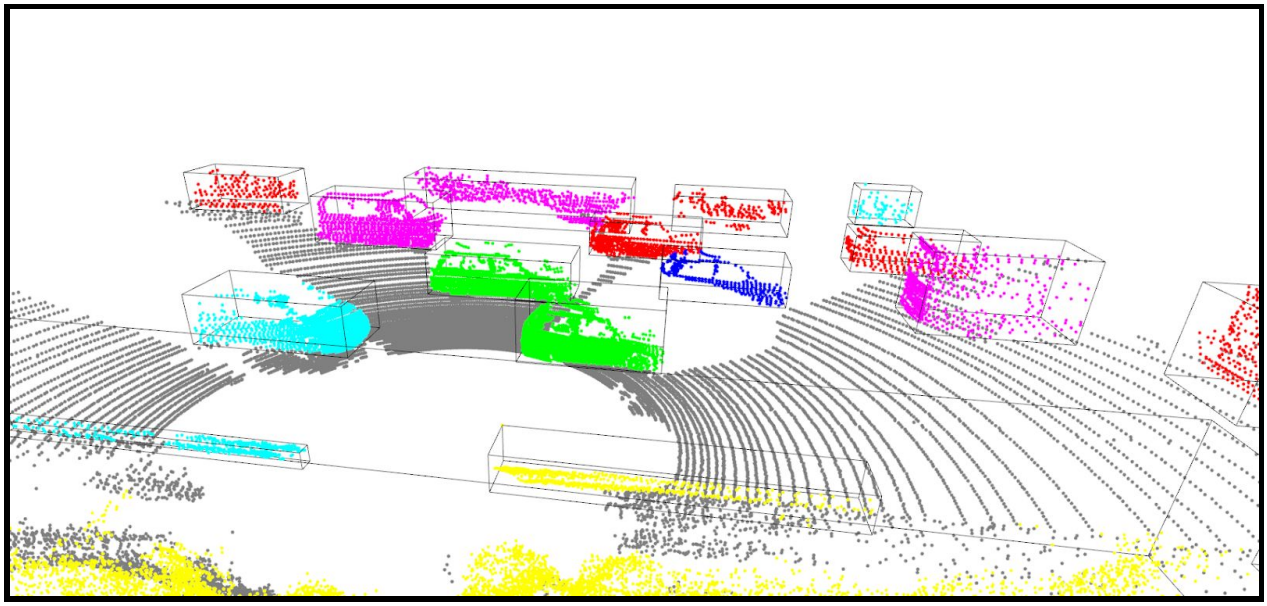


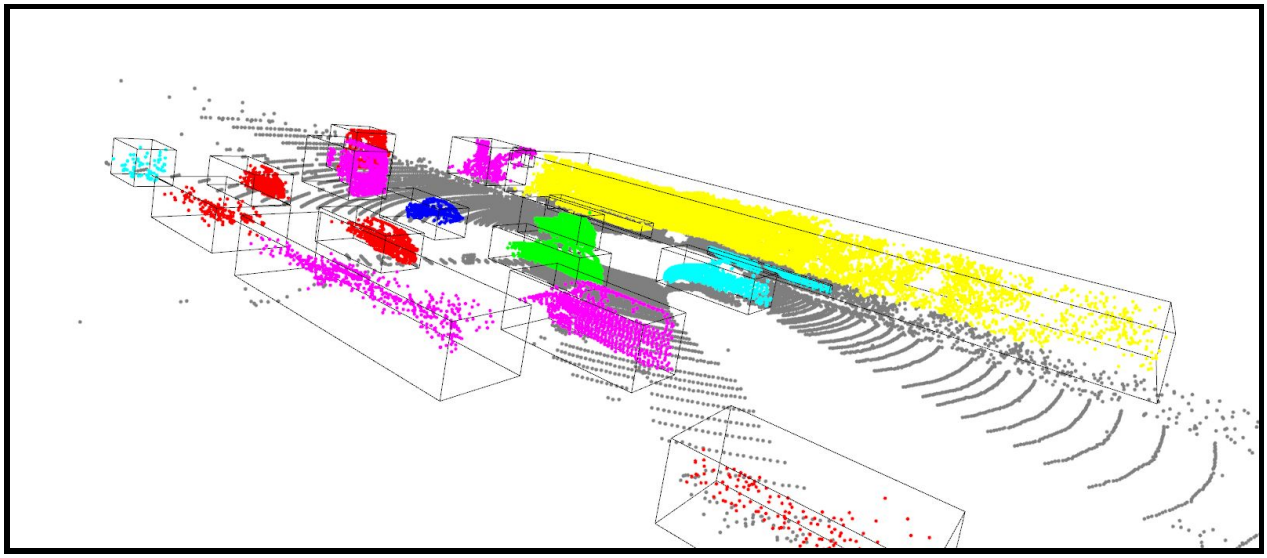


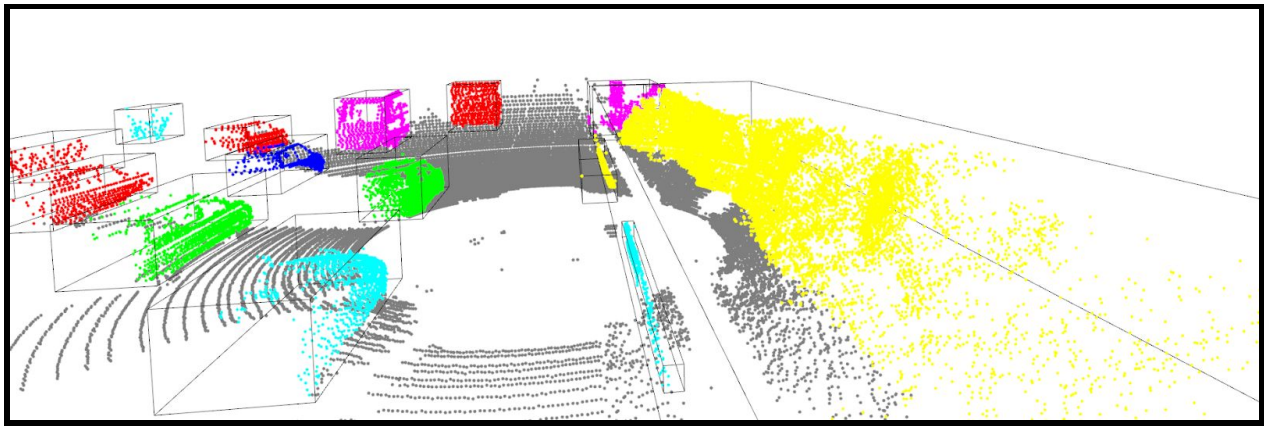


Frame 8

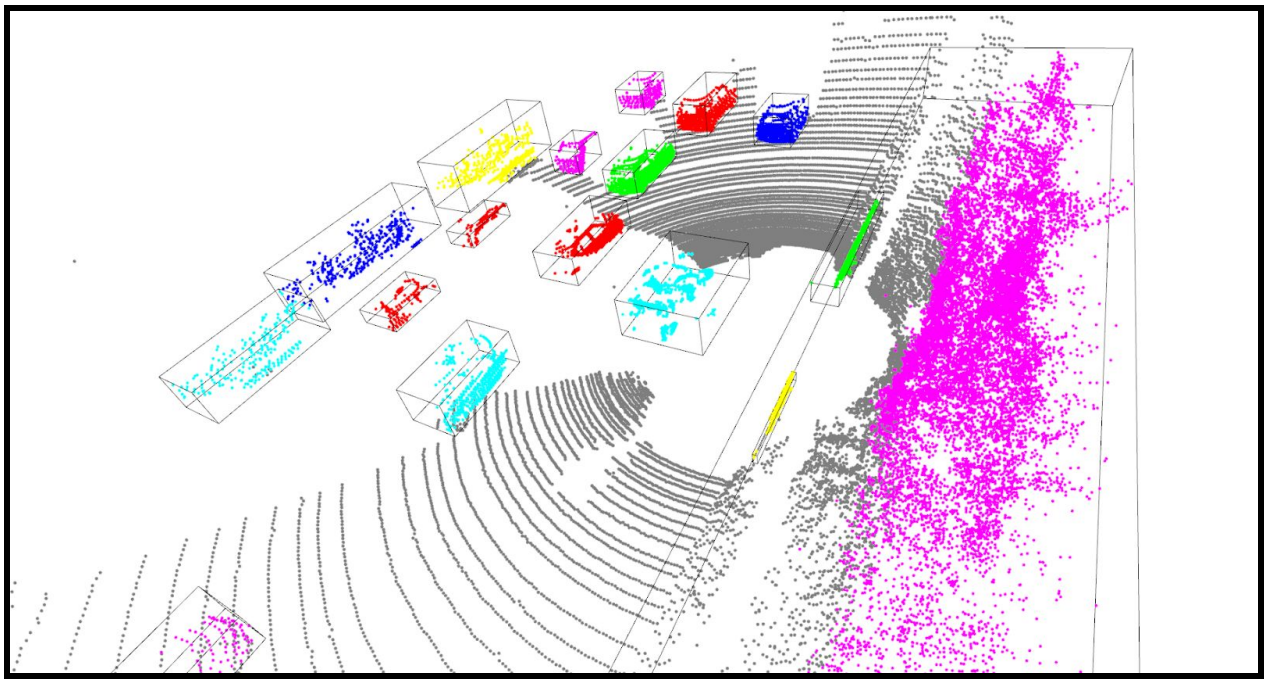


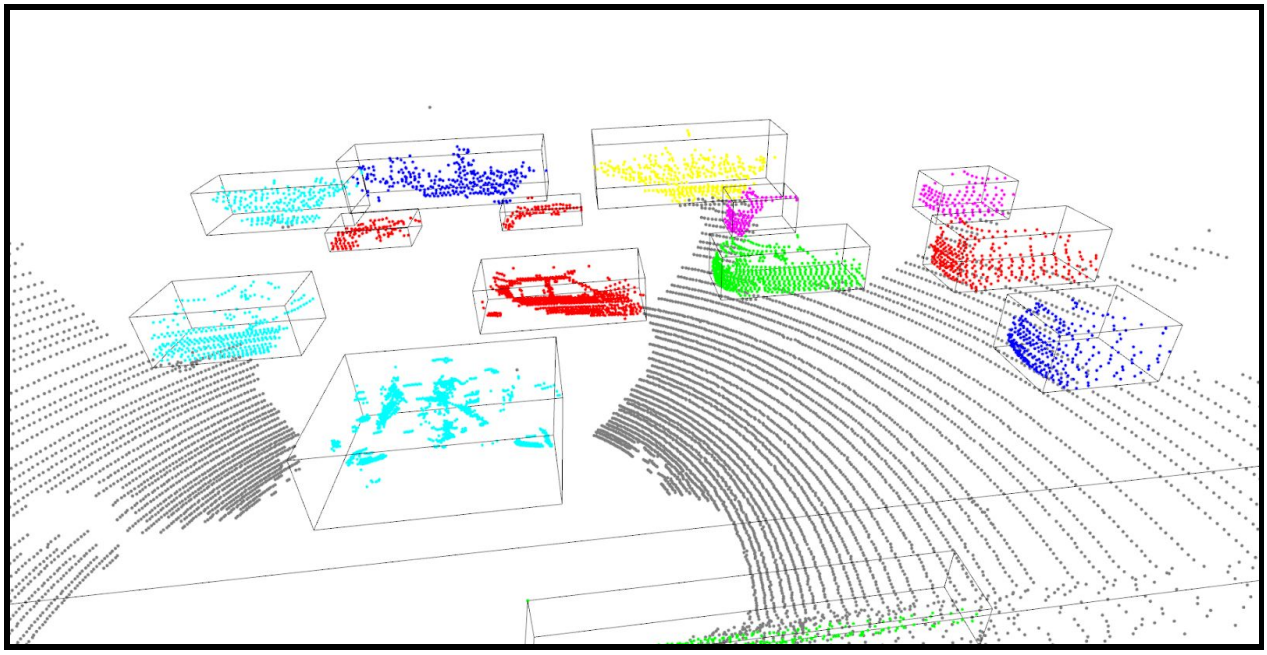


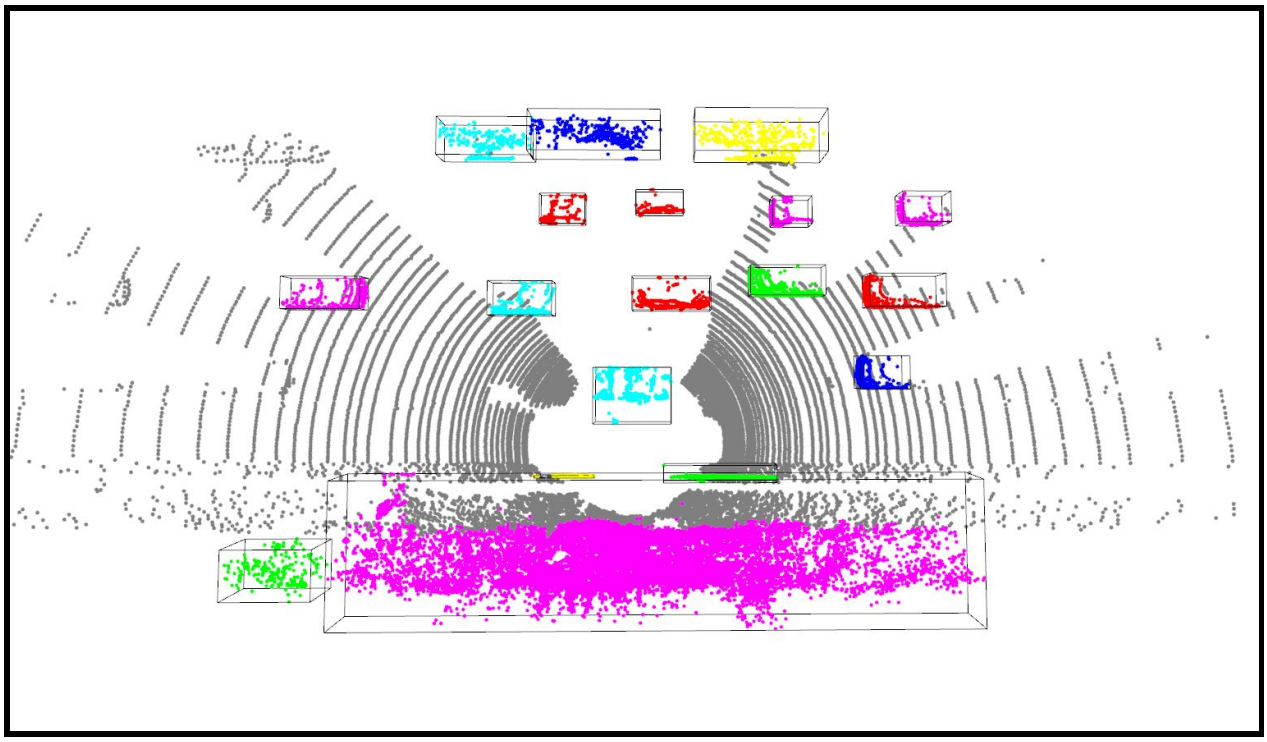




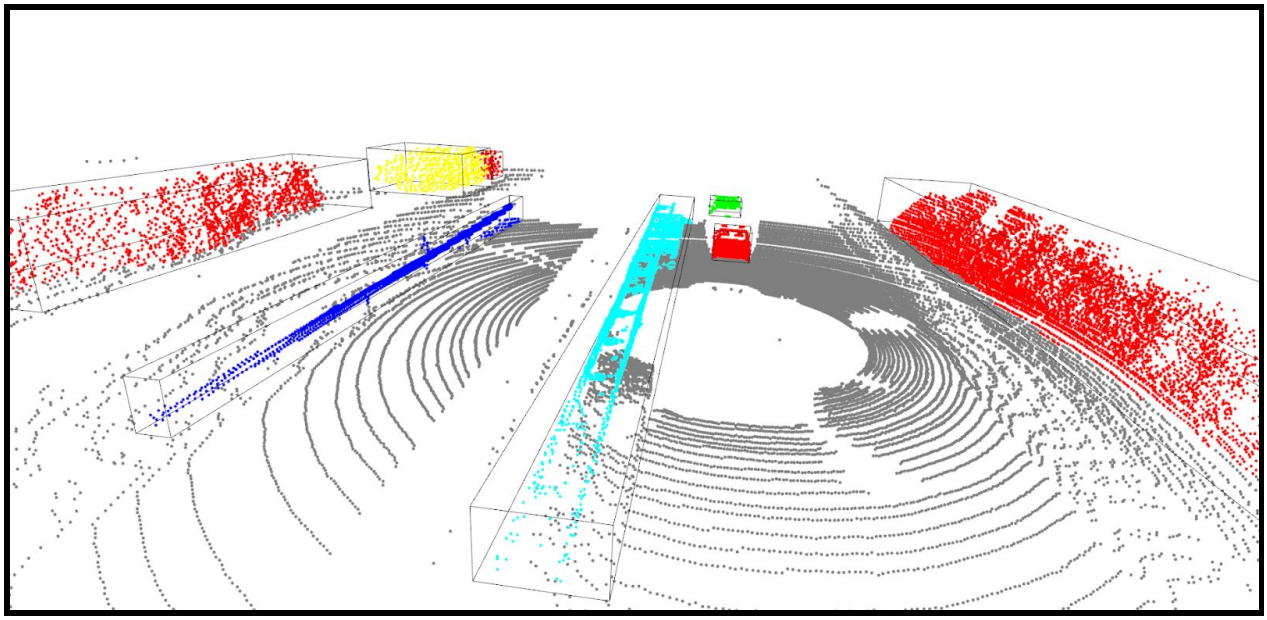
Frame 9

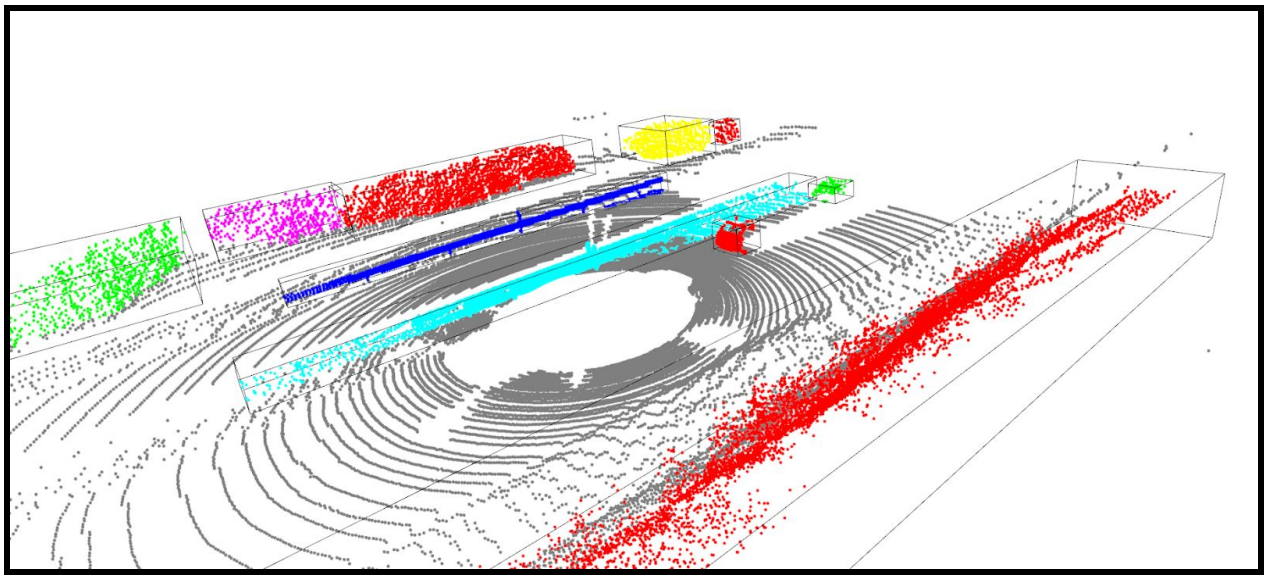


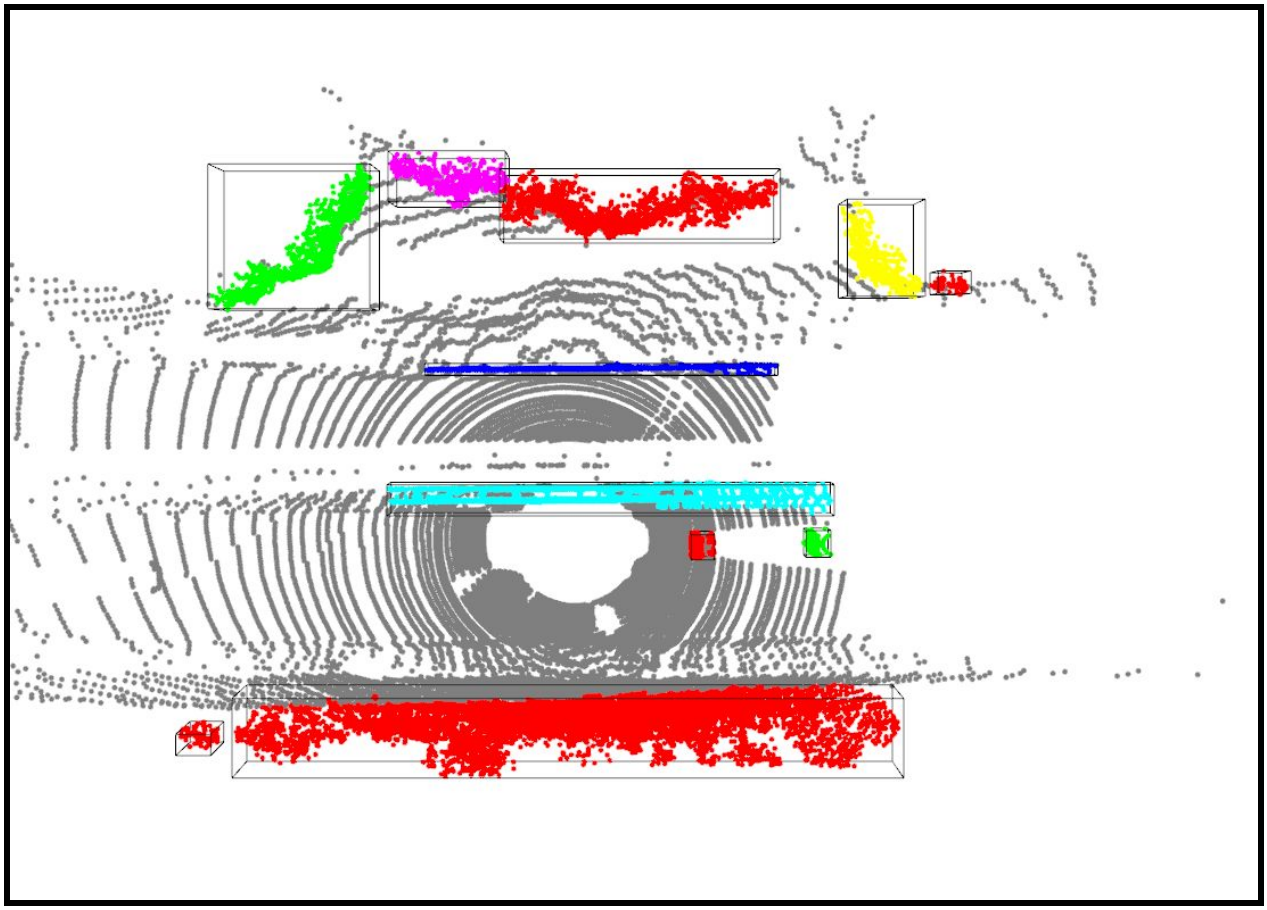




Frame 10:







Input files:

1. Please load a '*.pcd*' file containing three data channels viz X,Y,Z through the args parser.

2. To run the file run 'python3 main.py --file test.pcd --eps 1 --minPts 50 --kitti True --sklearn True' in the terminal, preferably with a KITTI dataset file. Replace the 'test' with the file name.