Senior Design II
Collaborative-Car-Driving
Authors: Melido Bello, Thierno Dicko

Table of contents

Automatic Driving Introduction

Autonomous driving technology heavily depends on driving simulators for their ability to generate diverse, labeled datasets and facilitate rapid algorithm training. These simulators provide a controlled and safe testing environment under a range of conditions, fast-tracking development and ensuring algorithm robustness. By enabling iterative refinement, simulators accelerate innovation, propelling us closer to the widespread adoption of fully autonomous vehicles.

Tools

CARLA 0.9.9 with python 3.7

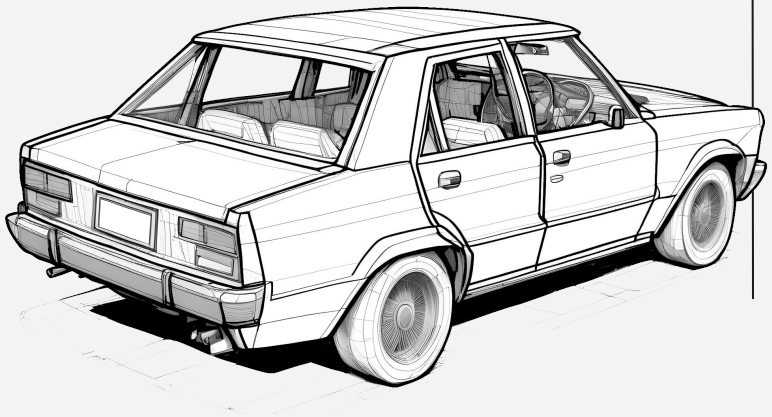
CARLA 0.9.9 is a simulation platform for autonomous driving development.
It offers a realistic environment for testing algorithms related to self-driving vehicles.
Python 3.7 integration allows easy interaction with the simulation.
Developers can prototype and experiment with perception, planning, and control algorithms.
Valuable for researchers and engineers working on autonomous vehicle technology.

Matlab with Simulink

MATLAB Automated Driving Toolbox: MATLAB provides a dedicated toolbox for developing perception, navigation, and control algorithms for autonomous driving applications. It includes functionalities for sensor fusion, object detection and tracking, path planning, and vehicle control.

Simulink: Simulink is a graphical programming environment for modeling, simulating, and analyzing multi domain dynamical systems. You can create plant models, controllers, and entire vehicle dynamics simulations using Simulink.

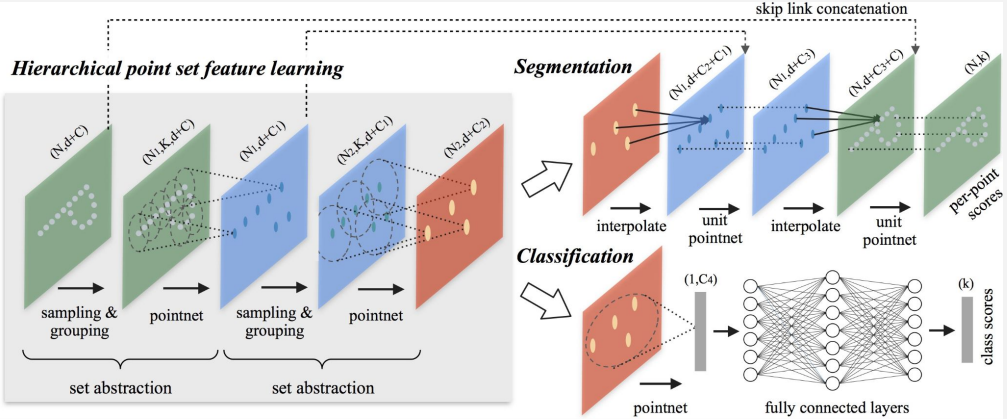
01 Open3D-LiDAR

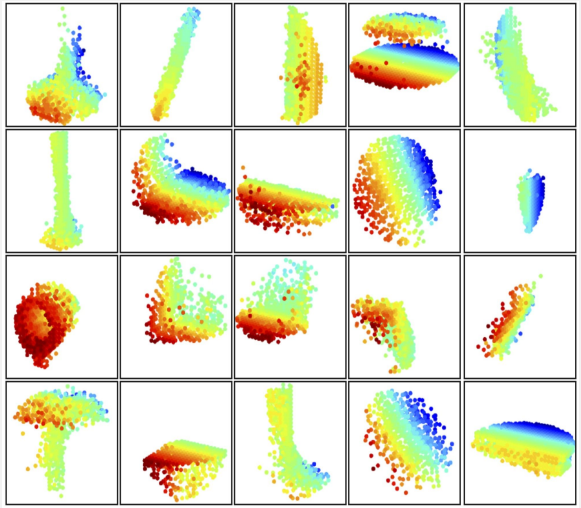


is a remote sensing method that uses a pulsed laser to measure the distance to the Earth.

Approach I - Open3D-LiDAR

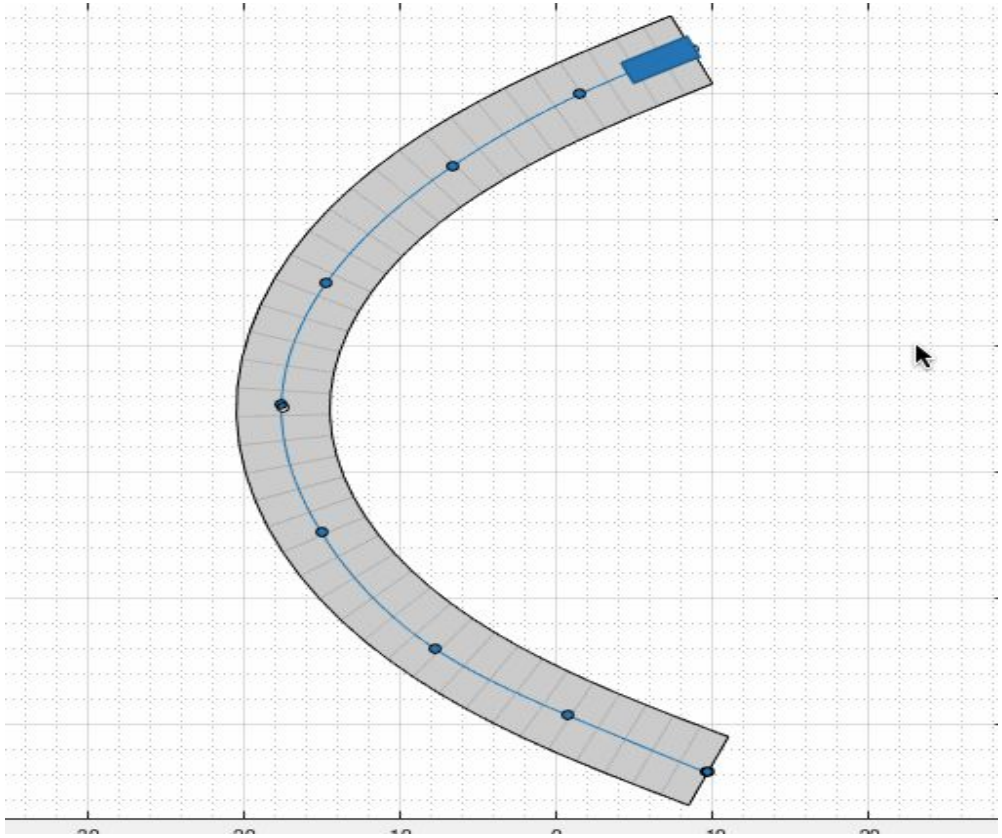
Open3D-LiDAR is a toolkit for processing 3D data from LiDAR sensors.
It employs neural network architectures like PointNet++ for understanding the environment.
Features are extracted from dense point clouds to distinguish objects and their characteristics.
Specialized modules for tasks like segmentation and object detection are utilized.
Results are refined for accuracy through post-processing techniques.
Visualization tools are provided to interpret the processed data effectively.





Simulation

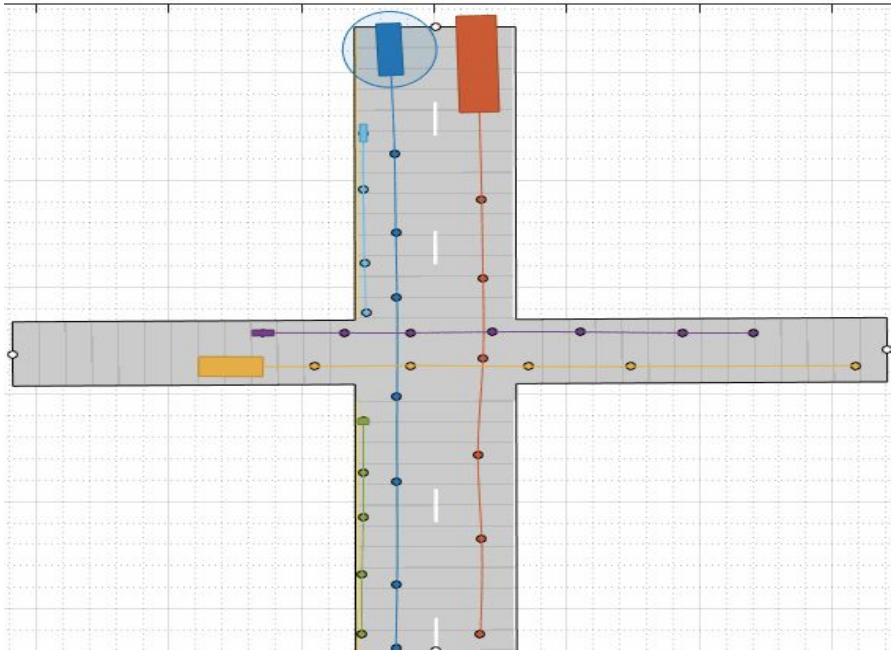
We generate our first simulation in a 2D format using only one car with a speed limit of 30mph, but we make sure that the speed of the car is reduces whenever the car go through a curve.

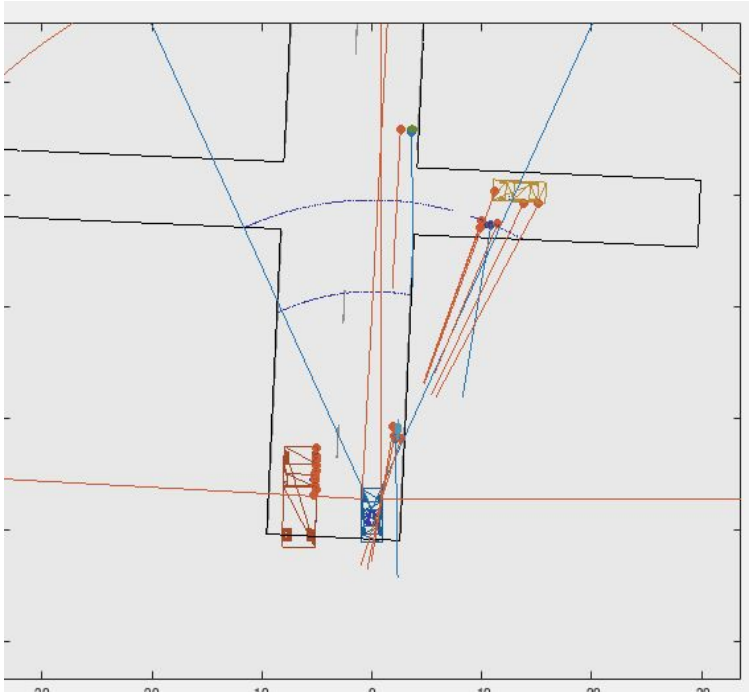


Waypoints, Speeds, Wait Times, and Yaw

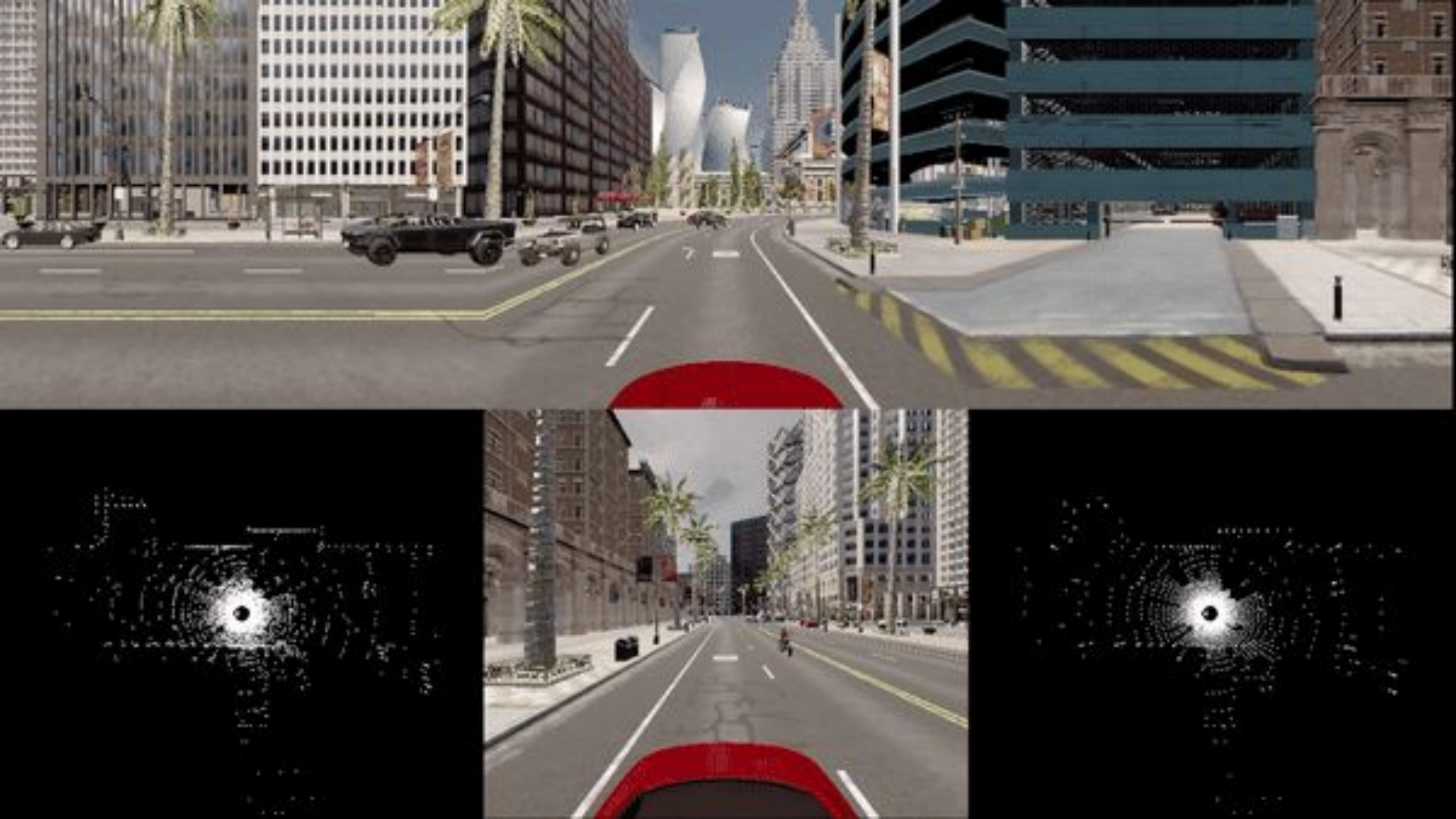
	x (m)	y (m)	z (m)	v (m/s)	wait (s)	yaw (°)
1	56.3200	-0.8665	0	30	0	120.1202
2	53.2000	4.4000	0	30	0	121.6898
3	47.3000	12.5000	0	30	0	132.2569
4	41.4000	17.8000	0	25	0	143.3577
5	32.6000	22.6000	0	23	0	161.4094
6	23.1000	23.7000	0	26	0	-174.12...
7	13.1000	20	0	28	0	-145.35...
8	6.2000	12.6000	0	30	0	-121.76...
9	2.4000	4	0	30	0	-107.32...
10	0.2000	-6.2000	0	30	0	-97.5082
11	-0.3000	-15.9000	0	30	0	-87.7187

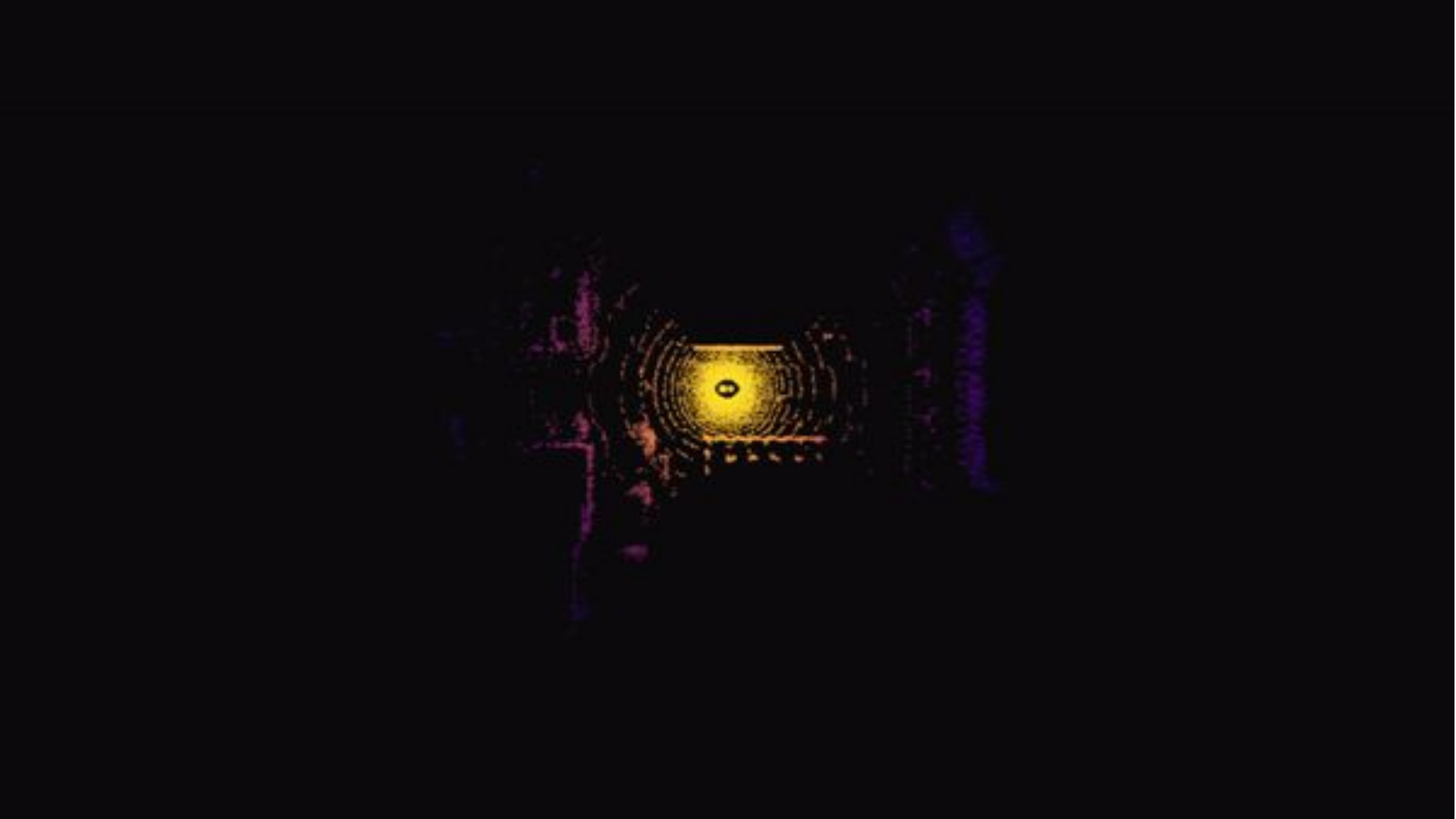
Second scenario consist of having different actors on the road including cars, truck, pedestrian, and bicycle. We generate also camera, sensor and radar to our ego vehicle to detect all the actors nearby.





As the ego vehicle get closer to other actors its make sure a safe distance is created in all sides.






Server: 14 FPS
Client: 97 FPS
Vehicle: Lincoln MKz 2017
Map: Town10HD_Opt
Simulation time: 0:04:49
Speed: 1 km/h
Compass: 269° W
Accelero: (0.1, 0.0, 9.8)
Gyroscop: (0.0, -0.0, -0.0)
Location: (-82.0, -65.2)
GNSS: (0.000586, -0.000215)
Height: 0 m
Throttle:
Steer:
Brake:
Reverse: 0
Hand brake: 0
Manual: 0
Gear: 1
Collision:
Number of vehicles: 41
Nearby vehicles:
9m Tesla Model3
81m Bmw Grandtourer
54m Mercedes Coupe 2020
42m Ford Mustang
85m Diamondback Century
78m Carlamotors Firetruck
80m Volkswagen T2 2021
81m Yamaha Yzf
82m Mercedes Sprinter
100m Tesla Cybertruck
CAPLA
CAPLA

Collaborative Driving idea explanation

Issues with method 1

Description: since the cars were not at a constant speed cause my computer was lagging but were calculating their next step everytime. This leaves room for errors.

In a simulation this may not be very relevant but imagine a normal car crash involving civilians

Figure: crash

News about West Chester, crash

Hamilton In-depth, Investi…

West Chester accident: Three-vehicle crash

46 minutes ago

WLWT

911 calls describe driver traveling wrong-way moments before triple-fatal crash

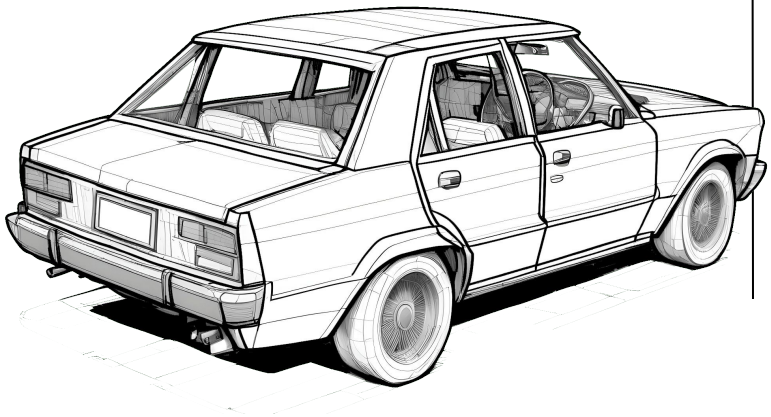
17 hours ago

WCPO 9 Cincinnati

Two killed in crash with wrong-way driver,

1 day ago

02 Object Detection



putting boxes around important objects:
Cars, traffic lights, etc.

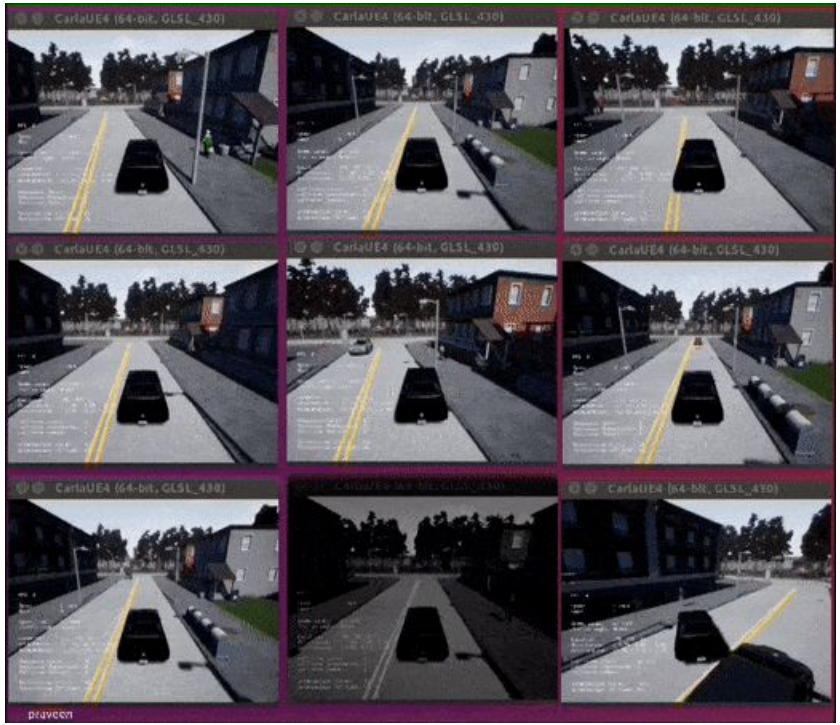
Method II - putting boxes around important objects: Cars, traffic lights, etc.

Why would this be different?
Cause it enhances visuals

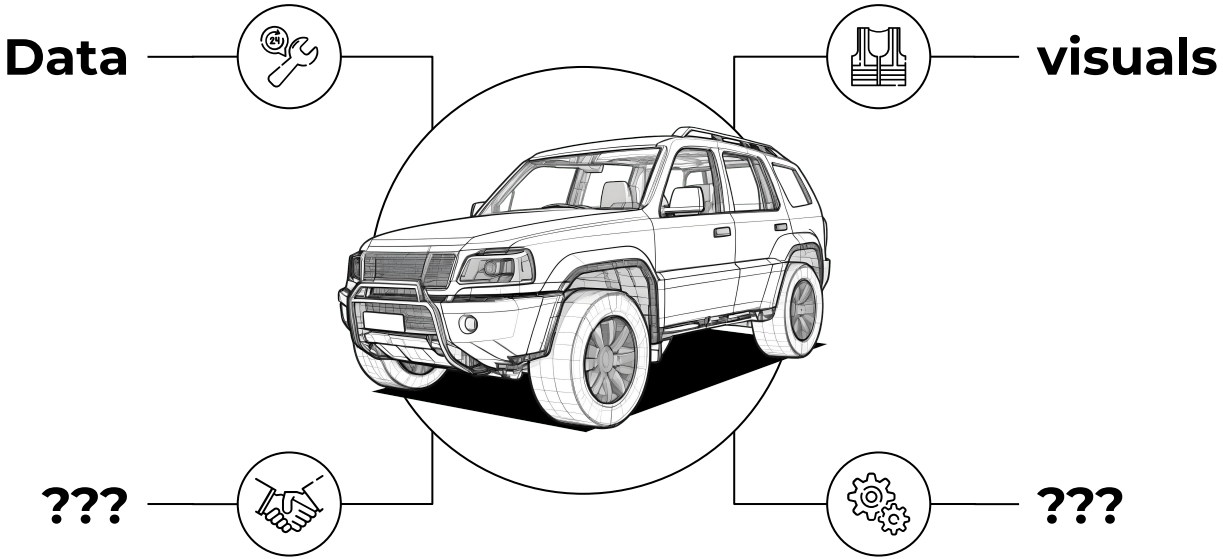
So now instead of just data now there is:

1) Data
2) visuals

This can be use for training models with machine learning or deep learning.

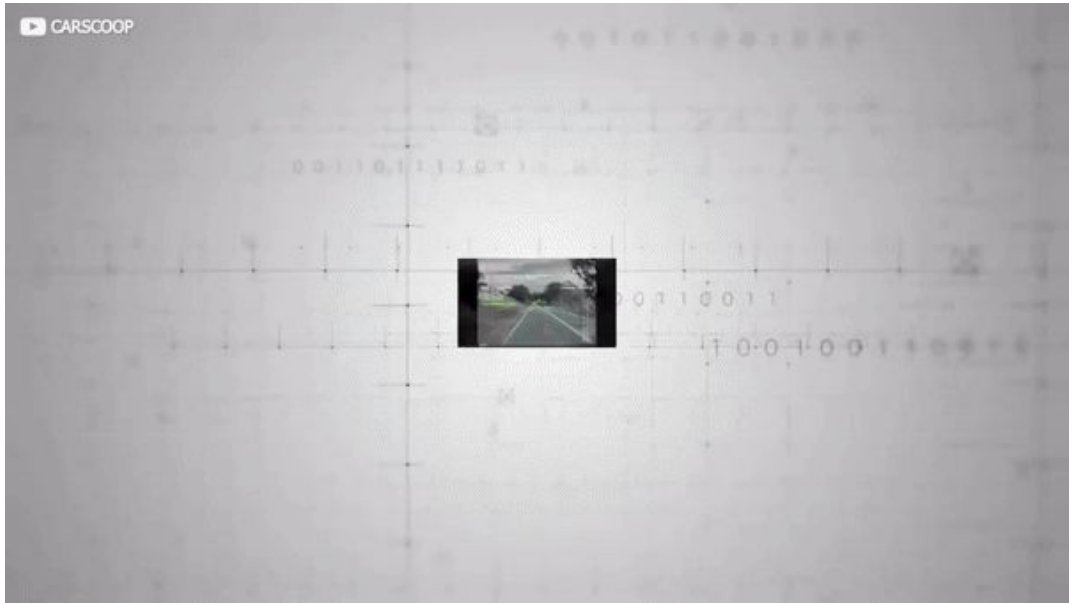


Infographics are always useful



Tesla approach

This is the exact tesla approach. Instead of using 1 method they incorporate a lot to increase accuracy and avoid crashes



Resources

Github repositories

- https://github.com/PacktPublishing/Hands-On-Intelligent-Agents-with-OpenAI-Gym.git
- https://github.com/carla-simulator/carla

Youtube videos

- https://youtu.be/ECmG0nNJE98
- https://www.youtube.com/playlist?list=PLQVvvaa0QuDeI12McNQdnTlWz9XlCa0uo
- https://www.youtube.com/watch?v=kS7OC7q0v3c&t=24s&ab_channel=MaisJamal

Documentation

- https://carla.readthedocs.io/en/latest/
- https://stanford.edu/~rqi/pointnet2/

Github repository

https://github.com/MelidoB/Senior-Design-II---Collaborative-Car-Driving.git