Virtual Simulation Platform for Autonomous Vehicle

1st Ajit Bijapur,Neha C. Panji,Akshata Ambure,Dr N.C.Iyer
dept. of Electronics and Communication
KLE Technological University
Hubli, India
ajitbijapur1996@gmail.com

***Abstract*— This paper is a document that delineates the autonomous functionalities on a simulation platform.A Self-driving car is a combination of different sensors like camera lidar, radar, and artificial intelligence using this sensor module that as to reach its destination without any human conduction and it has some functionalities like recognition, judgment, and operation.**

So offline testing remains a very important method permitting affordable and economical validation of car performance and vehicle management algorithms in multiple virtual eventualities.to achieve the autonomous functionality we started Autoware virtual simulation platform which will provide the necessary modules. Autoware is an Open Source software based on ROS, which is a complete set of self-driving modules like 3-D map generation, localization, detection, prediction, planning, and control. We can generate a vector map which is necessary for path planning. Object detection is achieved using the existing algorithms embedded into project, Localization of the autonomous vehicle is also achieved in our research using NDT mapping by the means of it the position of the car is obtained with the less error rate. Using programming tools multiple test cases are generated through which validation of the functionalities is achieved on the virtual platform

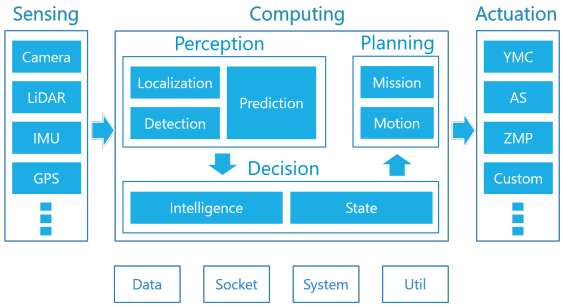
***Index Terms*—Autoware, localization, ROS, Vector map,Object detection.**

I. INTRODUCTION

Autonomous vehicles and Electric vehicles are the next generation of mobility infrastructure in the Automotive industry. The industrialists and other companies are working on achieving the SAE Levels of autonomy. The SAE stands for "Society of Automotive Engineers" which defines 6 levels of driving automation. Level 0 with no automation to level 5 with complete automation. Autonomous driving which is variably known as self-driving vehicle or robotic car is one of the most emerging technologies that has gained interest from industrialists and researchers because of its good deeds to the society and environment as well. These cars are a major advantage which reduces the accident rates, labor-cost savings and reduction in carbon dioxide emissions through optimized driving. But the autonomous vehicles are restricted to operate on roads because of the safety regulations which are held by the Government. Automotive Safety Integrity Level is a scheme defined by the ISO 26262 for the functional Safety for Road Vehicles standard that needs to be passed by the automotive companies[1].

To build an autonomous vehicle, it is achieved in four main processes of perception, localization, control, and planning. In brief, perception is the first stage of the computational pipeline where a car has to detect vehicles ahead with its sensors and continuously track their movements. Lo- localization, when self-driving cars use maps to figure out what the world is supposed to look like but it is not familiar with the car to understand. So to minimize that error, localization is done. The car has to later plan a path in finding a safe, comfortable and efficient path to maneuver around the traffic. All of the above can also be achieved on simulation platforms and programming tools. We can develop hands-on experience working with software programs. Simulations provide opportunities to engage developers, as they get to learn in an interactive environment that is close to the real-life situations The computation also requires high-end GPU's and CPU's for faster processing and high power. There are many open-source platforms for self-driving are Apollo, Airsim, Neurojs, Carla, and Autoware. Autoware is a software that allows you to virtually simulate and show how an autonomous car works virtually. So, to plan motion and maneuver the vehicle we need to generate an ADAS map and guide the vehicle through waypoints. To generate this ADAS Map it takes software to tackle the problem of map generation for Autoware. The vector map states information of road structure and path planner uses a vector map to create the path to guide the vehicle[2].

A. AUTOWARE Architectural Diagram



The above diagram depicts the Architectural diagram of Autoware. Autoware is all in one open source software for autonomous vehicles which provides modules for our dataset. The sensors on the left hand side gives input data after sensing.

During the computation it undergoes perception, planning and decision making. These computations and decision making controls the actuators like steering, throttle etc. During computation it has to localize to know the location of the vehicle. And also it has to detect the objects to take decisions accordingly[5].

II. Literature survey

The history of Autonomous vehicles started from General Motors in 1930s who created the first Self Driving car, which was guided by radio controlled electromagnetic fields. By 1958, GM had made this concept a reality by using sensors called pick-up coils that would detect the current flowing through a wire embedded in road. Later, Japanese improved and brought the concept of using cameras to process the road images. They tested this on a vehicle which only travelled at speed below 20mph. This later was improved by Germans who outfitted their vehicle with cameras and vehicle could safely at 56mph. The increase in the technology by decades has improved the ability of vehicle to detect and as well as the computational speed has been improved along with implementation of new sensors[1].

Technology have reached such an extent that Stanford university's racing team in cooperation with volkswagen created an autonomous vehicle which is most well known "Stanley" which won 2005 DARPA Grand Challenge. It also had made use of 5 roof mounted sick AG lidars to build the 3D map of environment. To process and compute the sensor's data to make decision, stanley had made use of Intel pentium M based computers running on various versions of LINUX OS. It also had made use of 5 roof mounted sick AG lidars to build the 3D map of environment. To process and compute the sensor's data to make decision, stanley had made use of Intel pentium M based computers running on various versions of LINUX OS.

Fig. 1. Stanley autonomous car

At present, though there are no fully autonomous vehicles seen on roads , but they are semi-autonomous due to ADAS features like assisted parking, cruise control system, lane detection. Many are able to achieve the SAE levels upto 2 where as Audi has made it to SAE level 3 but yet aren't on roads.

A. Localisation

It is important for an autonomous vehicle to locate itself. To achieve that there are different techniques been implemented i.e Odometry, Kalman Filter, Particle Filter etc. Most of the recently manufactured vehicles have inbuilt localization systems that have been implemented using satellite navigation with a digital map. This also requires sensors like GPS and IMU. The IMU gives the data which later provides a vector that includes coordinates and orientations.

B. Object Detection

To solve this challenge there are several techniques available. It was traditionally implemented with HOG and SHIFT method. Later, the technology of deep learning brought out many algorithms of neural network that increased the accuracy of detection. The algorithms that are evolved are : R-CNN, SPP, RetinaNet, YOLO Framework- yolo,yolo2,yolo3. Among all these algorithms Yolo has improved in terms of both speed and accuracy and considered to be provide pointed enhancement that address faster processing. The R-CNN which is deep learning based algorithm with highest accuracy but the biggest drawback of this algorithm was it obtained only 5FPS. Then there was an optimised algorithm of R-CNN itself called fast R-CNN and faster R-CNN, which were optimised to increase the accuracy and computation speed for detection.The R-CNN which is deep learning based algorithm with highest accuracy but the biggest drawback of this algorithm was it obtained only 5FPS.Then there was an optimised algorithm of R-CNN itself called fast R-CNN and faster R-CNN, which were optimised to increase the accuracy and computation speed for detection.

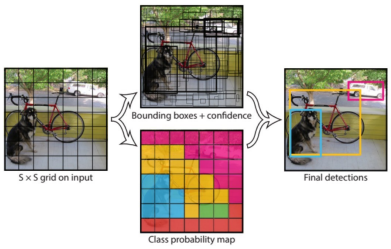


Fig. 2. The model

The below image depicts the process of object detection.At very first it divides the image into grids i.e S X S grid. Each grid looks for bounding boxes B along with class probability C. The whole prediction can be expressed in a single equation i.e S X S X (B * 5 + C)tensor[7].

C. Vector Map Builder

To generate a map we need the data from Lidar. The lidar gives the data in the format of " Point cloud data" also known as PCD file. This pcd file is important to generate the 3D map. In autoware, the map is divided into several subcategories like lane, crosswalk, edge, sidewalk, and many more. These categories define a road so that the vehicle learns on its own from the given input. SVMT is a simple vector map tool that provides an automatic lane connection. This requires a way point to generate a vector map to load the map on Autoware

Rviz using Open Planner for the simulation[5].There are several simulators available to test the autonomous functionalities like Carla, LGSVL, Autoware.

III. Implementation

The implementation involves both software and hardware requirements set up to perform the task. The implementation was achieved on Ubuntu 18.04v Platform which also requires libraries and other dependencies that need to be installed further. As Autoware is based on ROS we need to install Robot operating system (ROS) as well. So on the basis of compatibility and requirements, the installation varies. Ubuntu 18.04v requires ROS melodic as it is not compatible with ROS kinetic. To install ROS we need to configure the Ubuntu repositories and set up the system to accept the packages from ROS.

Robot operating system (ROS) is designed to be a loosely coupled system where a process is called a node, and each node is responsible for one task. Nodes interact with each other using messages that are exchanged through the logical channels called subjects. Using the publish / subscribe model each node can send or get data from the other node.

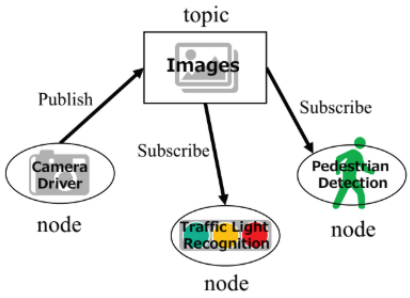


Fig. 3. The publish/subscribe model in ROS.

Nvidia-Docker builds a bridge between GPU and repository. Basically, this repository has all the libraries and packages. So, Nvidia-Docker acts as a wrapper around the docker command line that funds a repository with the necessary dependencies to execute code on the GPU. As developing a docker speeds up the application that is why it is necessary. There are basically two versions Docker CE (Community Edition) and Docker EE (Enterprise Edition). For small application computation we need community edition.

To install docker we need to initially update the software repositories and uninstall the older version of docker. Later, install docker.io which is a docker image. Later, we also need cuda, which is a computational platform and model developed Nvidia for computing on its own GPU. It speeds up computation of intensive application. Cuda toolkits gives a development environment,which includes libraries, debugging and optimization tools, compiler and runtime library to deploy the application. At last, we need to install autoware which is built on ROS for the research and development purpose for autonomous driving technology. It is all in one open source platform where you can achieve localisation, detection, prediction and planning through various algorithms with respective to those.

A. Virtual Simulator in Autoware Environment

So, the applications achieved with these software's are Visualisation of Point Cloud Data(PCD) files from lidar, Visualisation of ROSBAG from lidar, Localisation through NDT Mapping using inbuilt GPS and IMU from lidar, interacting the lidar point cloud data with the camera data. Object detection is also achieved on this platform with other software dependencies like darknet and yolov3 with Nvidia GPU and CUDA installedWe first visualize the ROSBAG and Point Cloud Data from Velodyne lidar on Autoware which we are able to see 3D map of the college. To plan the motion and to maneuver the vehicle, we need to create a predefined path to guide the vehicle so that it understands the roads shape and structure.To generate such a map we need a tool. The tool available as an open source is Autoware's tools.tierIV that can create ADAS map and waypoints to maneuver the vehicle. This tool requires point cloud data from the lidar using which we can view the 3D map of the college as well. This tool can define road shape,road surface, roadside, waypoints, roadstructure. In these, we can individually draw fences,road curves, edges, street light, footpath, crosswalk, lanes and stopline as well. Vector map builders are quite difficult as they do not generate the map automatically. Basically, this is a sort of stitching from one node to another node, where each node has its own coordinates that x,y,z. We need to export the ADAS map, which produces the .csv file. This file has the coordinates values of each node. These file are required to generate the launch file for the Autoware to view the map on Autoware. The only purpose is to generate a vector map to guide the vehicle.

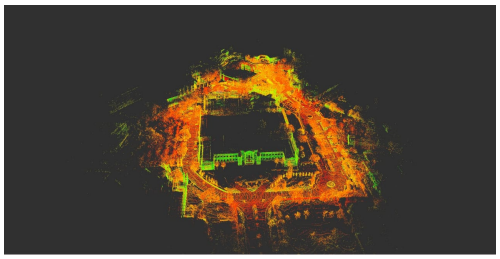


Fig. 4. 3d view of college map in pcd

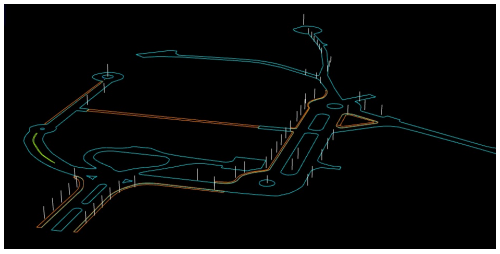


Fig. 5. 3d view of college map in pcd

The first application on this platform, Localisation which is achieved using normal distribution transform(NDT) algorithm. The position of the vehicle can be located by scan matching with the 3D map of PCD files. Here, we break the point cloud map into 3D boxes called Voxes.

NDT follows 3 steps i.e Initialisation, Matcing step, Transformation Determination.At first stage, the environment of the PCD is divided into voxes which are also known as grids. It also initialises the grid size. Second stage, it generates a Gaussian distribution function of each voxel. For each point in the model,we search the voxel that verifies the distribution. Third stage, to determine the transformation, we need to define objective function . From this we obtain x,y,z,yaw,pitch,roll values and transformation matrix. The position error is around 10cm.

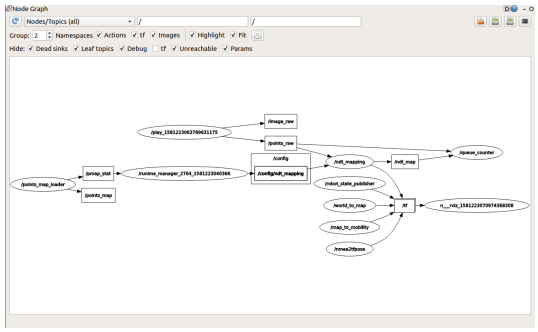


Fig. 6. NDT algorithm graph

Object detection, using an algorithm named YOLO version 3. This is the updated version with increase in accuracy and its speed. It uses darknet neural network framework for training the images.This model processes images in real-time at speed of 45 frames per second. This algorithm is a concept of bounding box which helps in prediction of the object. Yolo predicts the bounding box using dimensional clusters in which the network predicts 4 coordinates for each box that is T_x, T_y, T_w, T_z with width and height of P_w and P_h, and if the cell is offset from top left corner of the image by C_x,C_y The mathematical representation of the prediction is obtained using following equations.

B. Connecting Autoware with LGSVL Simulator

Using this software we have created a basic scenario like Environmental conditions for example rain,fog,summer and also we can create pedestrians,traffic light control,collision test by using python scripts we can generate and validate it and this also provide real-time outputs from sensors including camera, LIDAR, RADAR, GPS, and IMU.

Autoware communicates with the simulator using the rosbridge-suite, which provides JSON interfacing with ROS publishers/subscribers. The official autoware docker containers have rosbridge-suite included.

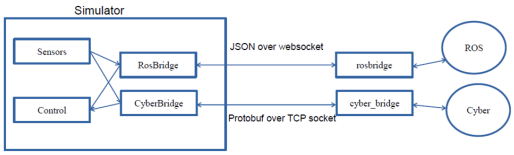


Fig. 7. working flow of ROS bridge connection

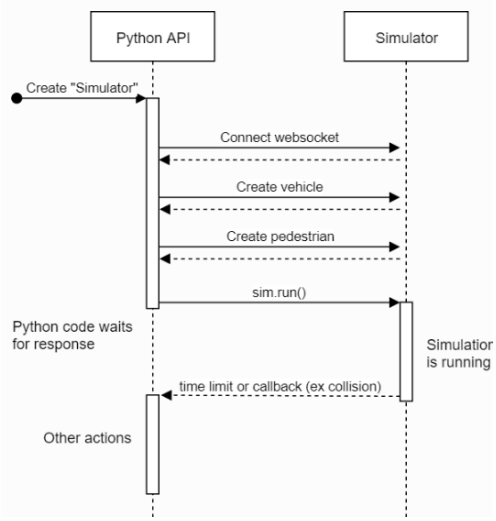


Fig. 8. TCP connection in LGSVL simulator for python scripts

IV. Result Analysis

A. Results of Localization in Autoware

The position of the vehicle can be found using the NDT algorithm.As we can see the output of this algorithm for localisation using NDT algorithm. This algorithms gives us the coordinates in the form of (x, y, z, roll, pitch, yaw). It also gives the transformation matrix. This position error of this localisation is in and around 10cm.

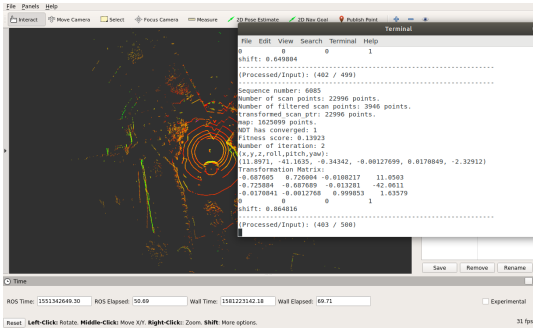


Fig. 9. NDT Mapping in autoware

B. Results of object detection

We can detect vehicles,pedestrian, traffic lights, person and variety of objects in Autoware with maximum accuracy. The algorithm we have used is YOLOv3.In figure 10 The lines with different colour represents different object. For example,

the blue circles represent the vehicles where as the green circles represent the pedestrians and yellow represents the curve edges. Figure 13 the graph of object detection that is generated on Autoware if we run rqt.

Fig. 10. car detection in Autoware

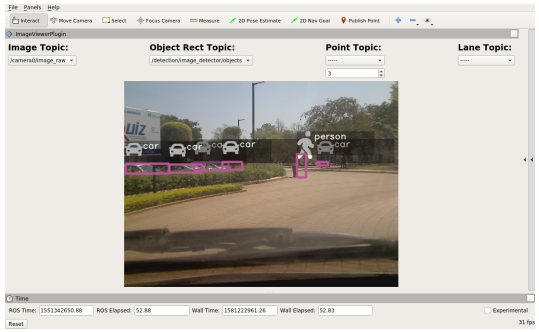


Fig. 11. car detection in Autoware

C. Results of LGSVL simulator

LGSVL Simulator exposes runtime functionality to a Python API which you can use to manipulate object placement and vehicle movement in a loaded scene, retrieve sensor configuration and data, control weather, time state, and more.

Fig. 12. Results of LGSVL simulator

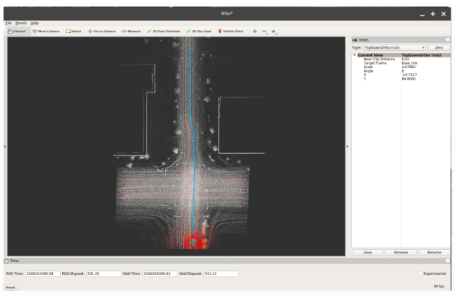


Fig. 13. ROS bridge connection between autoware and LGSVL

V. Future scope

Some of the issues that can be further explored are

- In future we will try to increase the computation speed and implement real-time service.
- The simulations can be extended further to explore and to achieve more of the parameters of autoware.
- The simulations can be tested on many other simulation platforms as well.
- We can also use other development kits to deploy our simulations on to like Jetson AGX drivers.
- The future work may also involve creation of HD Maps on LGSVL simulator.

VI. Conclusion

Autonomous electric vehicles are the next generation of the automotive industry. We will get to see autonomous cars on roads as well. But, before real time implementation we need to test all the test cases on simulation platform to see and analyse how things work virtually than taking risks by bringing them on roads directly. The sensor's data like lidar(velodyne),radar and camera data is for sensing the environment and act accordingly. The lidar data is major requirement in this project. The final simulations have met all the goals and objectives of the project. The localisation has given the position coordination and in case of object detection the detection is real fast with maximum accuracy. As well as the vector map is generated as expected. The ADAS map is bea By the means of virtual simulation, we are able to successfully present the autonomous functionalities on the simulation platform.

References

[1] title=Structure of intelligence for an autonomous vechile, author=Chavez, R and Meystel, Alex, booktitle=Proceedings. 1984 IEEE International Conference on Robotics and Automation, volume=1, pages=584–591, year=1984, organization=IEEE

[2] title=Simulation accelerates development of autonomous driving, author=Sovani, Sandeep, journal=ATZ worldwide, volume=119, number=9, pages=24–29, year=2017, publisher=Springer

[3] title=Autonomous vehicle testing and validation platform: Integrated simulation system with hardware in the loop, author=Chen, Yu and Chen, Shitao and Zhang, Tangyike and Zhang, Songyi and Zheng, Nanning, booktitle=2018 IEEE Intelligent Vehicles Symposium (IV), pages=949–956, year=2018, organization=IEEE

[4] title=Training Engineers in Autonomous Driving Technologies using Autoware, author=Carballo, Alexander and Wong, David and Ninomiya, Yoshiki and Kato, Shinpei and Takeda, Kazuya, booktitle=2019 IEEE Intelligent Transportation Systems Conference (ITSC), pages=3347–3354, year=2019, organization=IEEE

[5] title=Open-Source Tool of Vector Map for Path Planning in Autoware Autonomous Driving Software, author=Tun, Wai Nwe and Kim, Sangho and Lee, Jae-Woo and Darweesh, Hatem, booktitle=2019 IEEE International Conference on Big Data and Smart Computing (BigComp), pages=1–3, year=2019, organization=IEEE

[6] title=YOLO-LITE: a real-time object detection algorithm optimized for non-GPU computers, author=Huang, Rachel and Pedoeem, Jonathan and Chen, Cuixian, booktitle=2018 IEEE International Conference on Big Data (Big Data), pages=2503–2510, year=2018, organization=IEEE

[7] title=YOLO based Detection and Classification of Objects in video records, author=Jana, Arka Prava and Biswas, Abhiraj and others, booktitle=2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), pages=2448–2452, year=2018, organization=IEEE

[8] title=Generic NDT mapping in dynamic environments and its application for lifelong SLAM, author=Einhorn, Erik and Gross, Horst-Michael, journal=Robotics and Autonomous Systems, volume=69, pages=28–39, year=2015, publisher=Elsevier