Sensor Fusion and Pose Determination on an Autonomous Vehicle

GENG5511 / GENG5512 Engineering Research Project Report

Student:

Farhad Ahmed (22774116)
Master of Professional Engineering (Electrical & Electronic)
The University of Western Australia

Supervisor:

Prof Dr Thomas Bräunl
Robotics and Automation Lab
School of Electrical, Electronic and Computer Engineering
The University of Western Australia

(Word Count - Sections 1 to 7: 7168)

Table of Contents

Nomenclature 3

Abstract 4

Acknowledgements 5

List of Figures 6

1. Introduction 7

1.1. Purpose 7
1.2. Background 8
1.3. Scope 9

2. Literature Review 10

2.1. Pose Estimation using LiDAR 10
2.2. Pose Estimation using GPS and IMU 11
2.3. Sensor Fusion 12
2.4. Bounding Box Prediction 13

3. Design Process 15

3.1. Tools 15
3.2. Constraints 18
3.3. Requirements 19
3.4. Considered Design 19
3.5. Testing and Evaluation 23

4. Final Design 24

4.1. nUWAy LiDAR Pipeline 24
4.2. nUWAy Odometry Pipeline 24
4.3. second-ROS on nUWAy 24

5. Results 25

5.1. Point Cloud Operations 25
5.2. Odometry 25
5.3. Object Detection 29

6. Observations & Discussions 30

6.1. Point Cloud Operations 30
6.2. Odometry 30
6.3. Object Detection 32

7. Conclusion 33

7.1. Topics for Further Investigation 33

References 35

Nomenclature

2D	2-Dimensional
3D	3-Dimensional
AI	Artificial Intelligence
AV	Autonomous Vehicle
BEV	Bird's Eye View
CNN	Convolutional Neural Network
EKF	Extended Kalman Filter
GPS	Global Positioning System
GT	Ground Truth
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
LiDAR	Light Detection and Ranging
MOT	Moving Objects Tracking
Part A^2 Net	Part-Aware and Aggregation Neural Network
PL-ICP	Point-to-Line Iterative Closest Point
PSM	Polar Scan Matching
REV	Renewable Energy Vehicle (Project Team at UWA)
RGB	Red-Green-Blue (Camera Data)
RF2O	Range Flow-based 2D odometry
ROS	Robot Operating System
RTK	Real-Time Kinematic
SECOND	Sparsely Embedded Convolutional Detection
TSD	Traffic Signal Detection
URDF	Unified Robot Description Format
UWA	The University of Western Australia

Abstract

Safe and efficient operation of an autonomous vehicle (AV) is of exceptional importance, and it is possible only when the perception system of the vehicle is robust enough to represent the environment around it correctly. Light Detection and Ranging sensors (LiDARs), which provide a precise point cloud data of the surroundings, along with the software used to process this data, play a key role in determining perception accuracy. This research aimed to leverage the use of multiple LiDARs on a vehicle, merging their point cloud data with other sensors such as Global Positioning System (GPS) and Inertial Measurement Unit (IMU) to estimate the vehicle's pose. The vehicle on which these technologies were tested is the nUWAy, an electric shuttle intended to operate autonomously within the campus of the University of Western Australia (UWA). A processing pipeline of the sensor data was developed, which makes use of several open-source packages. Autonomous vehicle development in recent years has mostly focused on passenger cars operating in a road environment, and hence these packages required to be altered suitably before being implemented on the nUWAy. A self-written package was developed to merge separate sections of the point cloud data effectively and have two clouds - one for localisation and the other for obstacle avoidance. The perception data, both in 2D and 3D, along with the pose estimation data, was made available to the vehicle's decision-making systems to enable safe autonomous point-to-point navigation and obstacle avoidance. The entire pipeline was developed on Robot Operating System (ROS); all the data processed and generated follow all standard protocols and specifications, which provides the benefit of being easily integrable or modifiable in the future for further enhancements. The development was initially carried out in a simulated environment with real data gathered from the nUWAy, before being implemented on the actual vehicle. An attempt was also made on using deep learning on LiDAR data for the detection of objects, and potential was identified for the effective use of the same for detection of objects such as cars, cyclists, and pedestrians.

Acknowledgements

I would like to thank my supervisor and the director of the REV Project, Prof Dr Thomas Bräunl, for giving me this fantastic opportunity to work on the nUWAy shuttle and for his guidance and support throughout the project. I also extend my thanks to Dr Kai Li Lim, REV Project Coordinator along with Mr Thomas Drage, Mr Craig Brogle, Mr Chao Zhang and the rest of the team of REV Autonomous, for their continued support and knowledge sharing.

List of Figures

Figure 1 Typical hierarchical architecture of AVs, from [2] 8
Figure 2 nUWAy, from [5] 9
Figure 3 Comparison of RF2O with other methods, from [6] 11
Figure 4 Output of "robot_localization" (yellow/green) using only wheel encoders (left), encoders with 2 IMUs (centre), encoders with 2 IMUs and 2 GPSs (right) against GT shown in red, from [11] 12
Figure 5 Sample 3D detection on KITTI dataset, from [13] 13
Figure 6 Sample 3D detection (cars: green, cyclists: yellow & pedestrians: blue) on KITTI dataset, from [14] 14
Figure 7 Sample 3D detection on KITTI dataset, from [15] 14
Figure 8 ROS architecture 15
Figure 9 Coordinate frames, adapted from [19] 16
Figure 10 Coordinate frames, adapted from [20] 16
Figure 11 Velodyne LiDAR point-cloud, from [22] 17
Figure 12 Modified architecture for the nUWAy 19
Figure 13 Coordinate frames used in this research 20
Figure 14 LiDARs on nUWAy and their coordinate frames 20
Figure 15 Point cloud transform and merge algorithm 21
Figure 16 Definition of the pose 22
Figure 17 Odometry using GPS and IMU 22
Figure 18 nUWAy LiDAR pipeline 24
Figure 19 nUWAy odometry pipeline 24
Figure 20 second-ROS on nUWAy 24
Figure 21 Merged point cloud 25
Figure 22 LiDAR odometry results 26
Figure 23 GPS + IMU odometry results 26
Figure 24 Fusion odometry results 27
Figure 25 Odometry x-axis comparison 28
Figure 26 Odometry y-axis comparison 28
Figure 27 Odometry orientation comparison 28
Figure 28 second-ROS output 29
Figure 29 Inaccurate orientation in odometry results 31

1. Introduction

Both hardware and software of AVs are fast developing in recent times. While some car manufacturers have started making headway in autonomous driving through the development of proprietary software, many research centres and universities are carrying out open source development which is beneficial to the community at large [1, 2]. All research is aimed at developing safe and efficient operation of AVs with minimum to negligible human intervention.

While AV development started with passenger cars, they have since expanded to other market areas such as transit and tourism. Hence, parallelly to the marketing of autonomous cars, it is expected that autonomous shared-taxis and driverless transit vehicles such as shuttles would also evolve [3]. A great example of this is Singapore's Gardens by the Bay, who are operating autonomous shuttles for its guests, with a particular focus to making it easy for people with limited accessibility to explore the spread-out gardens [4].

With research mostly focussing AV operation on city roads and highways, there is very little work done on driving in cluttered environments, such as a Shuttle Bus operating within a University Campus or a Tourist Attraction. The internal roads in such a location would limit the speed at which a vehicle can operate, but at the same time, the dynamic environment with pedestrians and cyclists would pose many challenges to the driving systems in terms of determining obstacles and driveable areas. The nUWAy, an electric drive-by-wire shuttle bus at UWA provides an excellent platform for AV development in such a dynamic setting and the Renewable Energy Vehicle (REV) Project team is currently undertaking the task of making nUWAy autonomous [5].

1.1. Purpose

The purpose of and motivation behind carrying out this research were as follows:

1. Safety: Enabling safe & efficient navigation of the AV, especially in the highly dynamic environment of UWA.
2. Obstacle Detection: Make the perception system of the AV as robust as possible, so that it can avoid obstacles such as cyclists and pedestrians who obstruct the paths at most of the times during university working hours.

3. Integration: Leveraging the use of LiDARs and carrying out a unique kind of odometry fusion with GPS and IMU data.
4. Intelligence: Exploring the use of Artificial Intelligence (AI) to detect obstacles, which can help in avoiding obstacles during path planning.

1.2. Background

A brief discussion on the hierarchical architecture of an AV and the nUWAy vehicle is presented below.

1.2.1. AV Architecture

Figure 1 shows the standard architecture of an AV. The range of sensors can include specialised hardware such as LiDARs, GPS, IMU, Cameras and Wheel Encoders. The blocks represent various vital software algorithms and packages such as Mapping, Localisation, Object Detection, Traffic Sign Detection (TSD), Moving Object Tracking (MOT), Path Planning, Obstacle Avoidance and Drive Controls [2]. Software methods may make use of traditional numerical methods or more recent machine learning or deep learning methods which are made possible with increased processing capabilities of newer hardware.

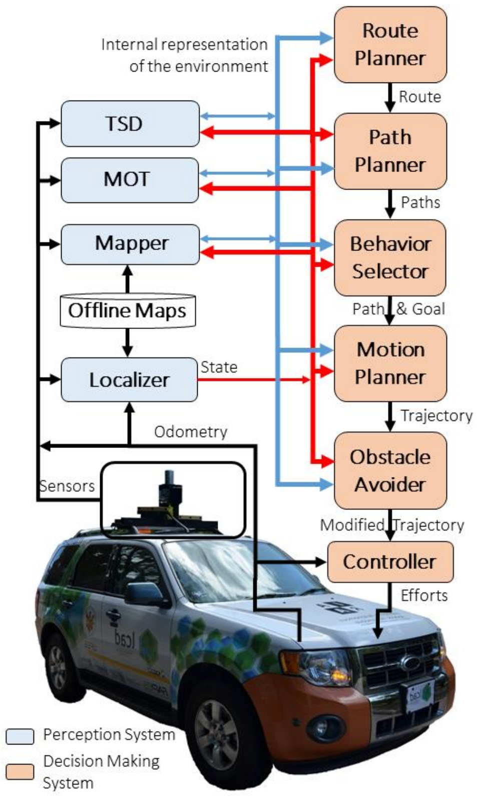

Figure 1 Typical hierarchical architecture of AVs, from [2]

1.2.2. nUWAy

As described above, nUWAy, shown in Figure 2, is the electric shuttle being made autonomous by the REV project team at UWA [5]. It has a unique sensor layout which necessitates a new data processing pipeline to enable practical autonomous driving. A key objective of this research was to develop this pipeline leveraging the use of multiple LiDARs and the fusion of its data with GPS and IMU data.

Figure 2 nUWAy, from [5]

1.3. Scope

The scope of this research project was to develop and test a software pipeline to carry out sensor data fusion and determine the pose of the vehicle accurately without the provision of a pre-fed map. This data could, in turn, enable more informed and safe autonomous vehicle control decisions.

The project objectives were separated into the following stages:

1. Use point cloud data from multiple LiDAR sensors and merge them in a meaningful manner into a single cloud using geometric transforms linking the vehicle base to each of the LiDAR positions.
2. Estimate the pose of the moving vehicle at any point of time using only the LiDAR data and only the GPS and IMU data.
3. Fuse the LiDAR data with the GPS and IMU data to give a more accurate estimation of the vehicle pose.
4. Test the point cloud data on a state-of-the-art neural network framework to detect obstacles and present them as bounding boxes around the vehicle, which can help in identification of driveable space.

2. Literature Review

This section discusses the various state-of-the-art frameworks which were reviewed for this project. Firstly, pose estimation methods have been explored, both using LiDARs and using GPS and IMU. Then, a framework suitable for a fusion of the pose estimation methods has been reviewed. Finally, the bounding box prediction on LiDAR data has been explored. The idea was to identify existing best practices, adapt them to work on the nUWAy and realistically combine them to arrive at the desired software pipeline.

2.1. Pose Estimation using LiDAR

Estimation of an AV's pose is essential for localisation, and it can be done by calculating the odometry, i.e., determining how far it has moved from a reference location. Traditionally, IMUs and wheel encoder odometry was relied on for pose estimation, but these are highly inaccurate when used alone due to accumulated drift [6]. The use of LiDARs for odometry has been evolving recently, and even mapping solely with LiDARs is possible now [7]. A few methods of determining odometry and pose using only LiDAR data has been review below.

2.1.1. Censi – Point-to-Line Iterative Closest Point (PL-ICP)

Iterative Closest Point (ICP) algorithm solves surface matching problems by iteratively minimising the guessed distance of a set of points from a reference surface [8]. PL-ICP is a variant of ICP which uses a Point-to-Line metric and tries to minimise that metric. This method requires a smaller number of iterations than traditional ICP for convergence and hence is a prevalent method of motion estimation. However, it makes use of a 3D point cloud and is computationally expensive. Although the results of PL-ICP for odometry is excellent, the slow computation makes it difficult for real-time implementation in an AV.

2.1.2. Diosi & Kleeman – Polar Scan Matching (PSM)

PSM was developed as a quick method to determine the odometry by simple point-to-point matching of consecutive LiDAR scans in their polar form [9]. It was designed to operate in the laser scanner's polar coordinate frame and thus vastly decrease computation time in comparison to ICP. However, it has been observed to have significant drifts, especially in areas lacking significant features, and hence may not be highly suitable for an AV, where even a slight deviation can result in highly undesirable results.

2.1.3. Jaimez, Monroy & Gonzalez-Jimener – Range Flow-based 2D Odometry (RF2O)

RF2O is an approach which uses consecutive LiDAR scans over time and imposes range flow constraint equation on them to estimate 2D motion [6]. It has been made open source by the researchers and has been proved to be more accurate than other forms of odometry such as PSM and standard wheel encoder odometry. It is also much faster than PL-ICP as it is not highly computing expensive and hence makes it an ideal candidate for selection in the pipeline being developed in this research.

A sample processing of odometry estimation by different methods in a simulated environment from the paper has been shown in Figure 3, which indicates RF2O to be the most accurate in following the Ground Truth (GT).

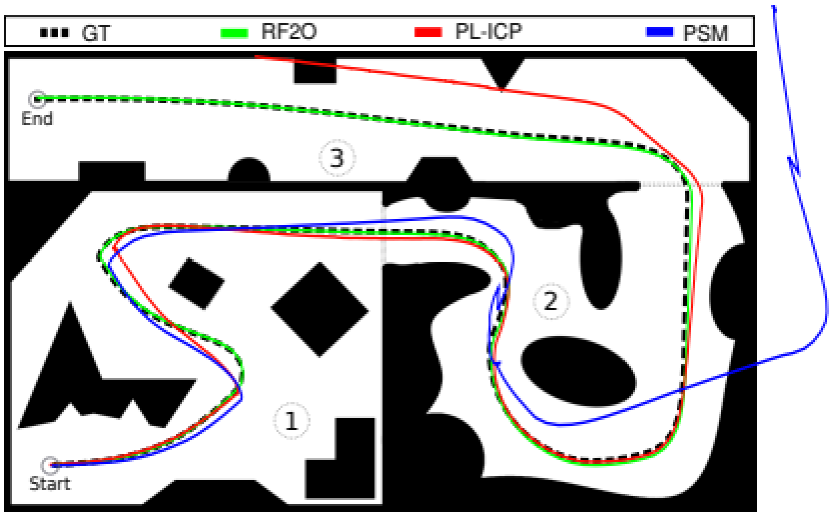

Figure 3 Comparison of RF2O with other methods, from [6]

2.2. Pose Estimation using GPS and IMU

As discussed in section 2.1, IMUs are not preferred to be used alone for pose estimation. However, new sensors are now available which integrate both GPS and IMU into an Inertial Navigation System (INS) solution, which has the capability of internally using filters to provide an accurate position and orientation output [10].

Hence, it is now possible to carry out more precise pose estimation by merely using the sensor output. A solution available for converting the sensor data from a GPS-aided INS solution has been explored below.

2.2.1. Bingham – "geonav_transform"

The package called "geonav_transform" was developed in ROS to integrate geographic navigation with sensor-frame orientation and velocities [10]. It can process any standard GPS and IMU messages to derive the vehicle odometry and express it in terms of a fixed reference frame. The primary advantage of integrating GPS is that there would be a correction of drift over time.

2.3. Sensor Fusion

As the objective of the project was to fuse different odometry data obtained from various sets of sensors, some review was done on the packages available for sensor fusion and have been discussed below.

2.3.1. Moore and Stouch – "robot_localization"

These researchers have developed and open-sourced a generalised Extended Kalman Filter (EKF) implementation for ROS [11]. It is capable of being customised for different sensors and parameters to solve the state estimation problem in various robots and AVs. The inputs that can be fed into the package include wheel odometry, IMU data and GPS data. The researchers have shown, using real-world tests, that using a more significant number of sensors results in a highly accurate localisation. This package can work in both 2D and 3D modes.

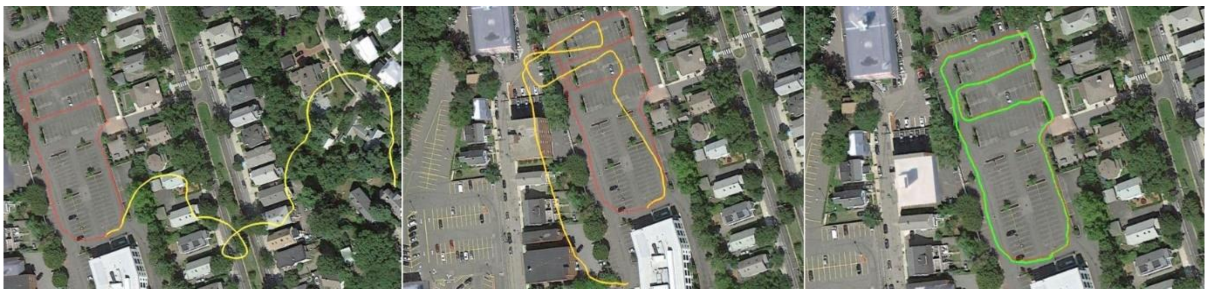

Figure 4 Output of "robot_localization" (yellow/green) using only wheel encoders (left), encoders with 2 IMUs (centre), encoders with 2 IMUs and 2 GPSs (right) against GT shown in red, from [11]

2.4. Bounding Box Prediction

Object detection is of considerable interest in AVs as it improves decision-making capability and enables safe, successful, and efficient navigation. Non-identification of obstructions such as pedestrians, cars and bicyclists may pose a severe safety hazard.

Object detection has been carried out extensively through camera image processing. However, these methods struggle to be reliable in low light or severe weather conditions. Reliability can be improved by instead using LiDAR data, which is less affected by external light or weather conditions. Further, it is easy to process spatial characteristics as the data incoming from the LiDAR is in 3D, in comparison to camera images where the depth information needs to be computed.

However, there is a lack of substantial research on using LiDAR for object detection as it is a comparatively newer technology than the camera. Further, the use of LiDAR data with neural networks is much tricky due to the randomness and irregularity of point cloud data. Suitable pre-processing of the point cloud data is necessary before feeding them into neural networks, and this has been attempted by a few researchers [12]. A few neural networks of vital interest, using LiDAR, have been reviewed below.

2.4.1. Sindagi, Zhou & Tuzel – MVXNet

MVXNet is an approach implementing early-fusion of LiDAR and Camera Red-Green-Blue (RGB) data leveraging the VoxelNet architecture [13]. The method computes bounding boxes on objects using a voxel-based 3D Convolutional Neural Network (CNN). The experimental results of this method, shown in Figure 5, are promising. However, it is computationally expensive, and the real-time AV use is untested.

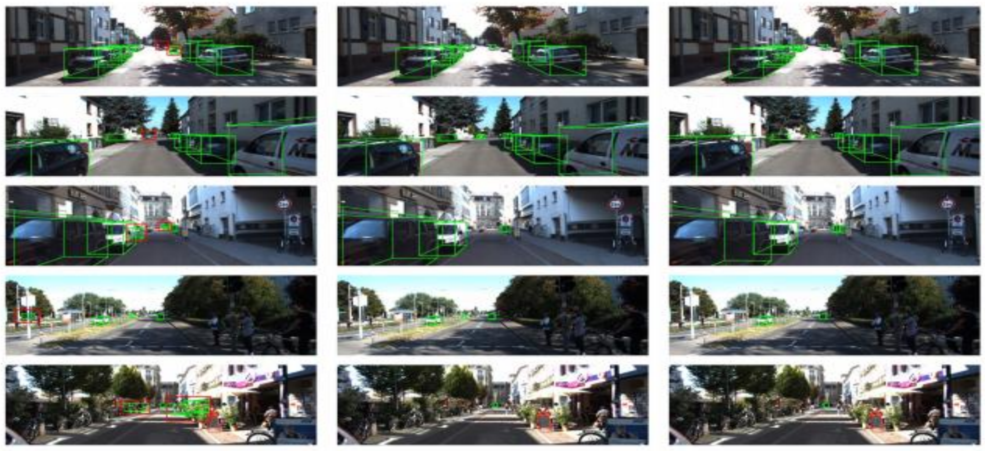

Figure 5 Sample 3D detection on KITTI dataset, from [13]

2.4.2. Shi et al. – Part-Aware & Aggregation (Part A2) Neural Network

Part A2 net is a 3D detection framework where the raw point cloud is processed in two stages [14]. Stage-I is a part-aware stage, which estimates the intra-object part locations and then uses a point cloud pooling scheme, while parallelly generating 3D proposals. Next comes Stage-II, which is the part-aggregation stage, and it better captures the geometric information of the objects to predict the bounding box locations accurately. Experimental results of this two-stage detection have been shown in Figure 6. However, there is no evidence of testing this framework in real-time on AVs.

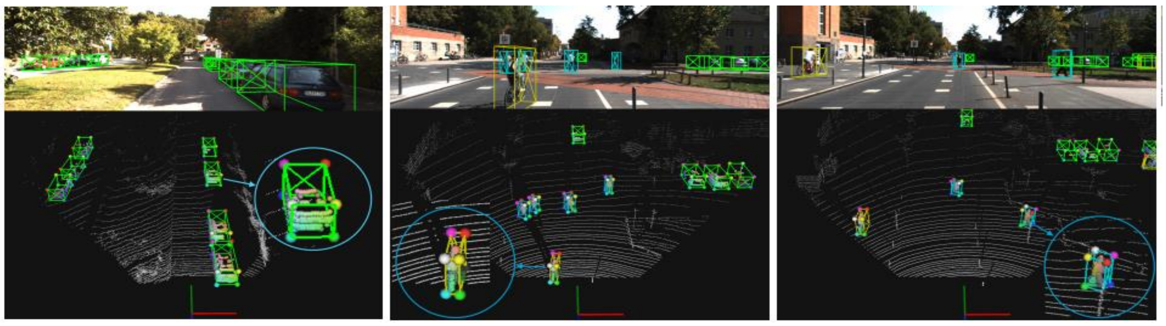

Figure 6 Sample 3D detection (cars: green, cyclists: yellow & pedestrians: blue) on KITTI dataset, from [14]

2.4.3. Yan, Mao & Li – Sparsely Embedded Convolutional Detection (SECOND)

SECOND was developed to solve the problem of slow speed and low orientation estimation problem prevalent in voxel-based CNNs by implementing sparse convolution [15]. Such a method vastly increases the speed of operation while retaining a high level of performance. It has been proved to produce state-of-the-art results on KITTI dataset. An integration of SECOND into ROS is also available as an open-source package, which makes it a potential candidate for selection in this project [16]. Experimental results of this method have been shown in Figure 7.

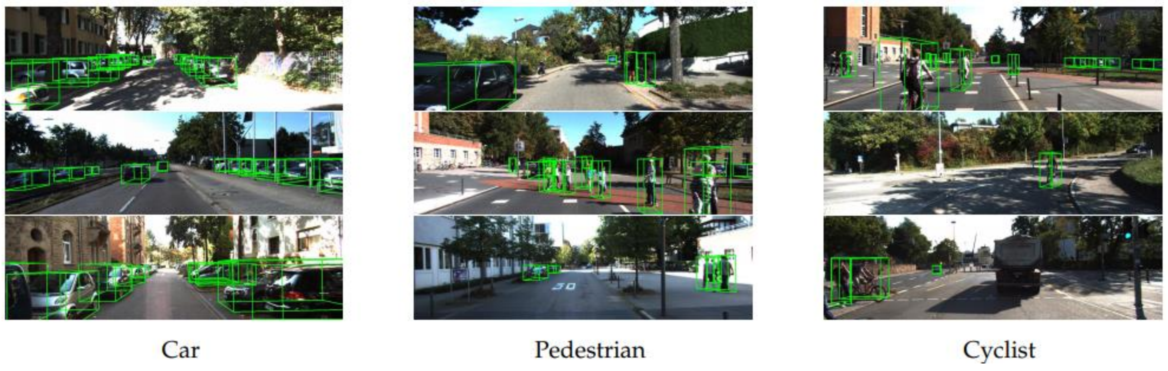

Figure 7 Sample 3D detection on KITTI dataset, from [15]

3. Design Process

3.1. Tools

3.1.1. ROS

ROS is an open-source framework for robotics software development [17]. It implements a graph architecture consisting of several processes called nodes. These nodes are capable of publishing streams of data called topics to which other nodes can subscribe. Some of the popular packages which are essential for the operation of an AV through ROS are listed below:

- Coordinate Frame Representation
- Localisation
- Mapping
- Navigation and Path Planning

Figure 8 shows a simple representation of nodes and topics in ROS and how some of the nodes are designed to interface with hardware inputs and outputs. There are also a lot of simulation packages available which enable testing of any software in a simulated environment rather than experimentation on an actual AV. The advantage of using ROS is that it is easily integrable and modifiable due to its simple architecture. However, the communications take place over IP ports, and this may introduce considerable delay, which might hinder usage on an AV which requires real-time processing. However, for development purposes and for the use case of nUWAy, which is a slow-moving vehicle, this system can suffice.

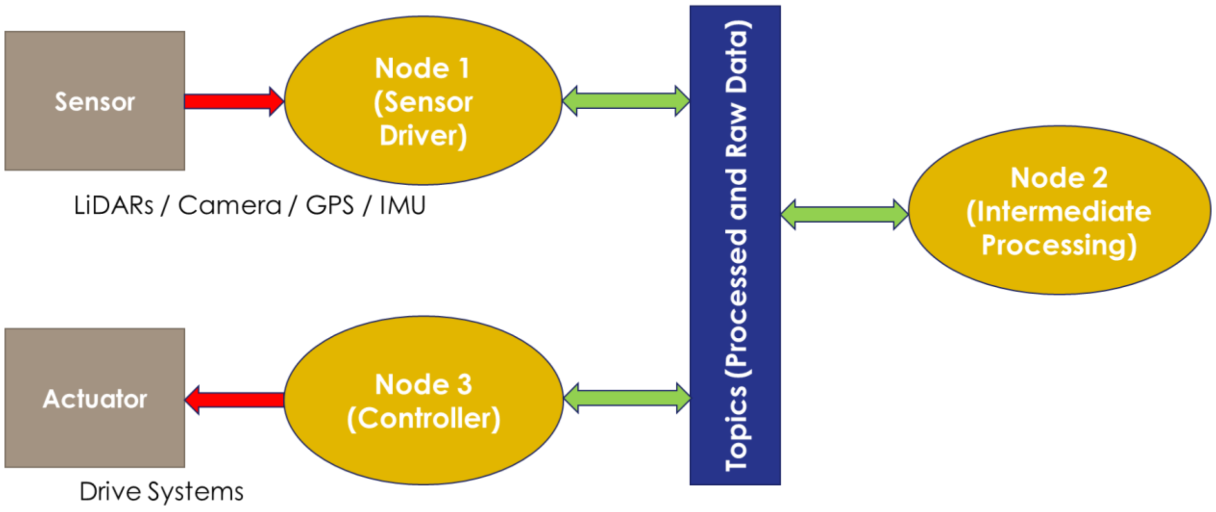
Sensor
Node 1 (Sensor Driver)
LiDARs / Camera / GPS / IMU
Topics (Processed and Raw Data)
Node 2 (Intermediate Processing)
Actuator
Node 3 (Controller)
Drive Systems

Figure 8 ROS architecture

3.1.2. Coordinate Frame Representation

Coordinate frames in ROS are represented in a tree structure [18]. Every object has a frame which is linked to a parent frame through a geometric transform.

Figure 9 shows a standard layout of coordinate frames used in a mobile robot. The “earth/utm” is an actual point on the earth and a fixed reference. Another earth-fixed framed is the “map”, which acts as a global reference. It follows that there is usually a fixed transform between “earth/utm” and “map”. The “odom” frame is also earth-fixed; however, it is a short-term fixed reference. Hence, although ideally, this frame would have a fixed transform with “map”, due to drift, this may slightly change over time, as shown in Figure 10. The “base_link” is the frame attached to the base of the AV, and it is a moving frame.

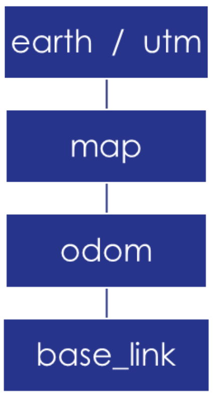

Figure 9 Coordinate frames, adapted from [19]

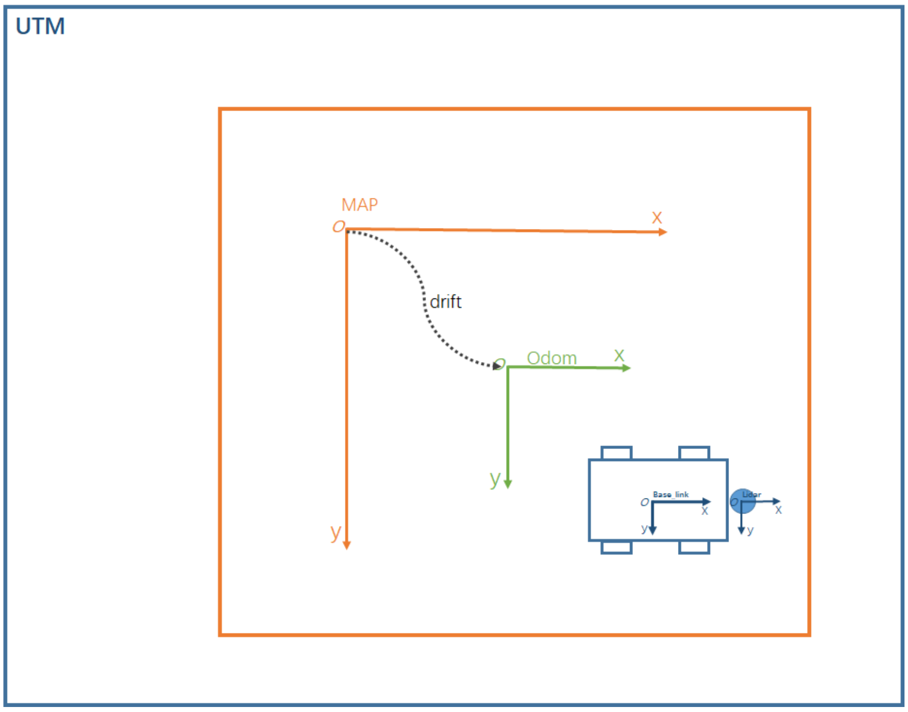

Figure 10 Coordinate frames, adapted from [20]

All the sensors on the AV have their respective coordinate frames which are defined through fixed transforms with the “base_link” in the vehicle’s Unified Robot Description Format (URDF) file [21].

3.1.3. LiDARs

LiDAR is a critical sensor which generates a 2D scan or a 3D point cloud of the environment around it using laser light pulses [22]. This data is used in Localisation, Mapping, Obstacle Avoidance and Road Keeping. Figure 11 shows a point cloud from a high definition Velodyne make LiDAR.

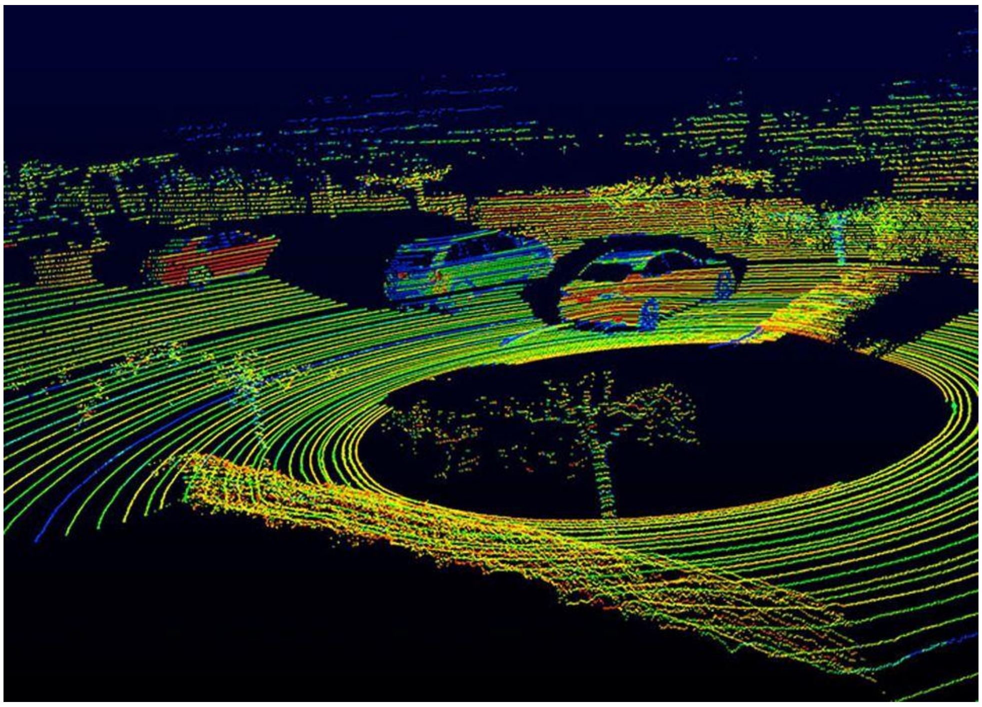

Figure 11 Velodyne LiDAR point-cloud, from [22]

The nUWAy is equipped with 3 different types of LiDAR sensors:

1. Sick 2D LMS – 4 nos.: These are termed “Safety” LiDARs and mounted at the bottom of the nUWAy on all 4 corners. These are a single layer LiDAR sensor with coverage of 270° and a maximum range of 20m. These sensors are available to both the High-Level systems and the Low-Level PLC. The Low-Level Safety PLC triggers an emergency stop in case of a close obstruction in the path of the shuttle. This sensor is the most critical one for the safety system but can provide only a 2D environment information and misses any obstacles which are overhanging or at a height.

2. Sick 3D LD-LRS – 2 nos.: These are termed "Localisation" LiDARs and mounted on the top of the shuttle, one facing the front and the other facing the rear. These provide a 4-layer, 3.2° vertical and 85° horizontal field of view and have a maximum range of 300m. Due to the long-range and height of mounting, these LiDARs are suited for localisation.
3. Velodyne VLP-16 – 2 nos.: These are the best class of LiDARs available on the shuttle with 16 layers, 30° vertical and 140° horizontal field of view. Apart from generating a point cloud, these sensors are also capable of indicating the intensity of reflection, and hence the possibility of detecting road markings and signs. The range of these sensors is up to 100m.

3.1.4. GPS and IMU

There are 2 GPS devices available on the nUWAy. The first one is NovAtel's PwrPak6 series receiver which is coupled with a high-performance antenna mounted on top of the shuttle. This receiver can also be configured to receive Real-Time Kinematic (RTK) positioning data to enhance the precision of the GPS fix [23].

The second one is integrated with an IMU into a GPS-aided INS solution, the XSens MTi-G-710. It can provide geographic position along with velocity, acceleration and orientation data [24]. The data provided by this device is already filtered and fused internally before being read in by ROS.

3.1.5. ROS Packages

The ROS packages for a few frameworks reviewed in Section 2, namely "rf2o_laser_odometry" (Section 2.1.3), "geonav_transform" (Section 2.2.1), "robot_localization" (Section 2.3.1) and "second-ROS" (Section 2.4.3) have been used in this design. Another package, "pointcloud_to_laserscan", has been utilised to convert a 3D LiDAR point-cloud into a 2D scan as the localisation is done in 2D [25].

3.2. Constraints

There were a few constraints in this project as listed below:

1. The computing power available of the bus was minimal, and the system struggled to keep up with the real-time processing when intensive software was used, as multiple projects were run parallelly on the nUWAy.

2. The LiDAR data had large blind spots on the sides, which made it difficult for LiDAR-based odometry methods to determine the motion accurately.
3. The neural network for the object detection was trained on KITTI dataset, which used high definition 64 channel LiDARs. As such, using it with LiDAR data from the nUWAy, which was in comparison very sparse, did not yield favourable results on the pre-trained model.
4. Training the neural network requires labelling of ground truth, which is a challenging and time-consuming task for 3D LiDAR point-cloud data, and hence had to be left out of the scope of this project.

3.3. Requirements

The following have been identified as the critical requirements of this project:

1. The processing pipeline should be able to process data in real-time on the limited available hardware.
2. The odometry and object detection must be accurate so that the Mapping and Decision-Making Systems can use it.
3. The outputs of the developed pipeline should be in standard ROS messages so that it can be easily integrated with the other packages.

3.4. Considered Design

3.4.1. Architecture

The nUWAy lacks input from wheel encoders, which means that odometry is not available inherently. Hence, it was proposed to use one or more of the frameworks described in Sections 2.1 and 2.2 and derive the odometry from sensors alone and use the fusion method described in Section 2.3 to fuse them. Accordingly, the architecture was altered from the standard one shown in Figure 1 to the one shown in Figure 12.

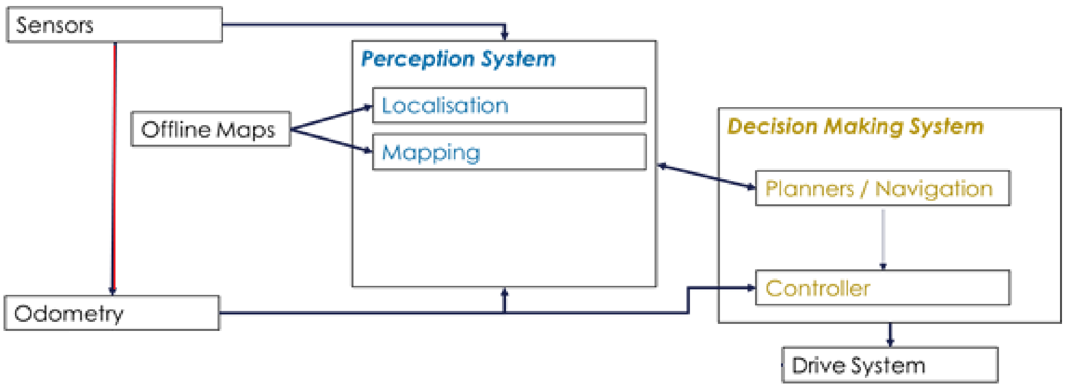

Figure 12 Modified architecture for the nUWAy

3.4.2. Coordinate Setup

A vital purpose of this project is to be able to obtain accurate localisation data without the provision of a map. Hence, it is required to continuously derive and update the transform between “odom” and “base_link” which were discussed in Section 3.1.2, and as such, this link would be the primary area of interest in the coordinate frame system. The “map” frame is only used as a reference to plot and find out how accurate the localisation technique is. Hence “odom” and “map” have both been linked to the global reference frame “utm” through a fixed transform. The standard transform tree in Figure 9 has been altered accordingly to the one shown in Figure 13.

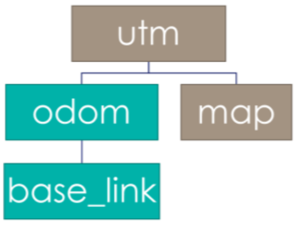

Figure 13 Coordinate frames used in this research

3.4.3. LiDARs in URDF

A description of the LiDAR sensors layout is provided in Section 3.1.3. By carrying out actual measurements, the translation and rotation of each sensor frame from the base frame called the “base_link” were determined. As described in section 3.1.2, these were then incorporated as fixed transforms into the URDF for the AV. The arrangement of all the LiDAR sensors mentioned above on the shuttle has been shown in Figure 14.

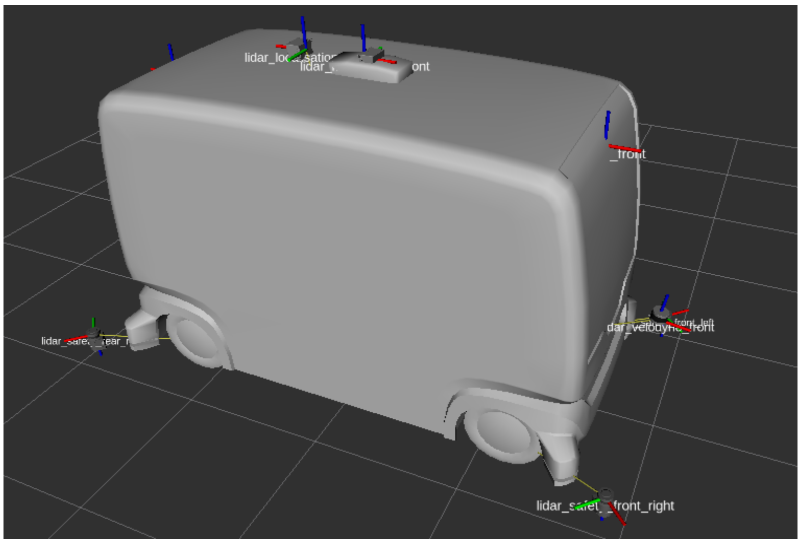

Figure 14 LiDARs on nUWAy and their coordinate frames

3.4.4. Point Cloud Operations

Once the transforms were defined in the vehicle's URDF, it became possible to visualise the output of all the sensors in 3D. However, most of the packages available for use on ROS such as those mentioned in Section 3.1.5 require the LiDAR data to be published on only one topic. Hence, it was necessary to carry out a transformation of all the incoming point clouds from their frames to the "base_link" frame and subsequently merge them.

A node called "nuway_tf_mg_clouds" was developed specifically for this task. This package also provided with the ability to selectively choose the sensors from which to merge the point cloud data as well as to define height cut-offs for the incoming point clouds. The output of this node was published in a standard ROS data structure for LiDAR point clouds, and it served as the input for several subsequent processes.

The algorithm for the package developed has been shown in Figure 15. As is necessary for any embedded application, an infinite loop has been incorporated. Mathematical functions were used to transform points in the coordinate axes of the individual LiDARs into the "base_link" coordinate frame.

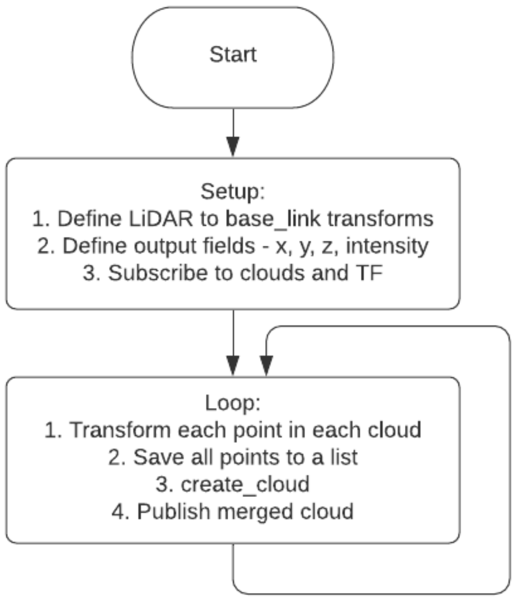
Start
Setup:
1. Define LiDAR to base_link transforms
2. Define output fields - x, y, z, intensity
3. Subscribe to clouds and TF
Loop:
1. Transform each point in each cloud
2. Save all points to a list
3. create_cloud
4. Publish merged cloud

Figure 15 Point cloud transform and merge algorithm

All systems do not use the 3D point-cloud data, and hence, a 2D scan was created using the "pointcloud_to_laserscan" package, mentioned in Section 3.1.5. This package flattens the cloud, creating a Bird's Eye View (BEV).

3.4.5. Pose Estimation

The pose is the combination of Position and Orientation. The position is merely 3D point coordinates in space, while the orientation defines how an object is orientated in the 3 dimensions relative to the coordinate frame.

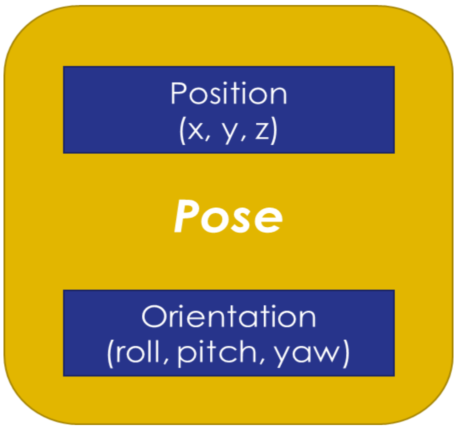

Figure 16 Definition of the pose

As discussed in Sections 3.4.1 and 3.4.2, a vital goal of this project is to determine the transform between the "base_link" and the "odom" frame using various sensors. In other words, it is a vehicle pose or odometry estimation, which can subsequently be fed to perception and decision-making systems.

3.4.6. GPS and IMU Odometry

The first method of pose estimation was carried out merely using GPS and IMU data and merging them. A node called "nuway_xsens_odom" was used to rewrap the GPS and IMU data provided by the XSens INS device into a message structure favourable to be fed into the "geonav_transform" package discussed in Section 2.2.1. The latitude, longitude and altitude data from the GPS was translated into x, y, and z with a fixed reference frame and then the orientation and velocity data from the IMU was merged with it to give out the final odometry message, as depicted in Figure 17.

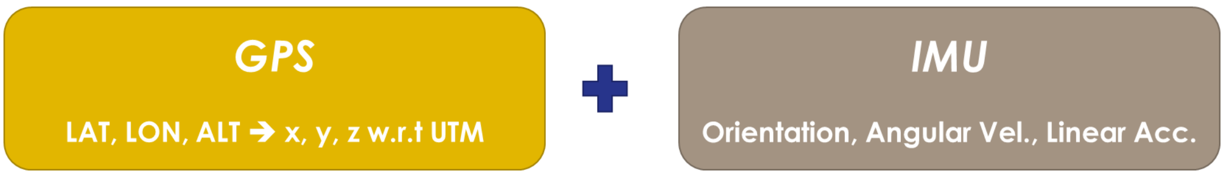
GPS
LAT, LON, ALT → x, y, z w.r.t UTM
IMU
Orientation, Angular Vel., Linear Acc.

Figure 17 Odometry using GPS and IMU

3.4.7. LiDAR Odometry

The “rf2o_laser_odometry” package discussed in Section 2.1.3 was employed with necessary tweaking of parameters to carry out progressive comparisons of LiDAR scans and determine the vehicle odometry. As this package takes in a single 2D LiDAR scan, the BEV obtained as described in Section 3.4.4 was used.

3.4.8. Sensor Fusion

Two different odometry messages were obtained, as discussed in Sections 3.4.6 and 3.4.7. To leverage the use of all sensor data available to determine the odometry, fusion using EKF of “robot_localisation”, as discussed in Section 2.3.1, was implemented. The parameters were tuned according to the available sensor data on the nUWAy and the desired output.

3.4.9. Object Detection

As a secondary goal, object detection using “second-ROS” framework discussed in Section 2.4.3 was attempted. The target was to bound cars, cyclists, and pedestrians in discrete boxes and to make this information available to other decision-making systems.

3.5. Testing and Evaluation

The testing of point-cloud operations and pose estimation using 3 different methods, viz., GPS + IMU, LiDAR, and Fusion were carried out on the nUWAy. The object detection using “second-ROS” was only attempted on recorded data from the shuttle on an external processor as the computational abilities of the on-board computer was limited.

Evaluation of the success of the odometry models was performed by manually driving the shuttle on a predefined route and then comparing the pose over time. For the object bounding box detection, an evaluation was not carried due to a lack of labelled ground truth data.

4. Final Design

4.1. nUWAy LiDAR Pipeline

The LiDAR pipeline for the nUWAy, which generates both a 3D point cloud and a 2D scan, was developed using nodes described in Section 3.4.4, as shown in Figure 18.

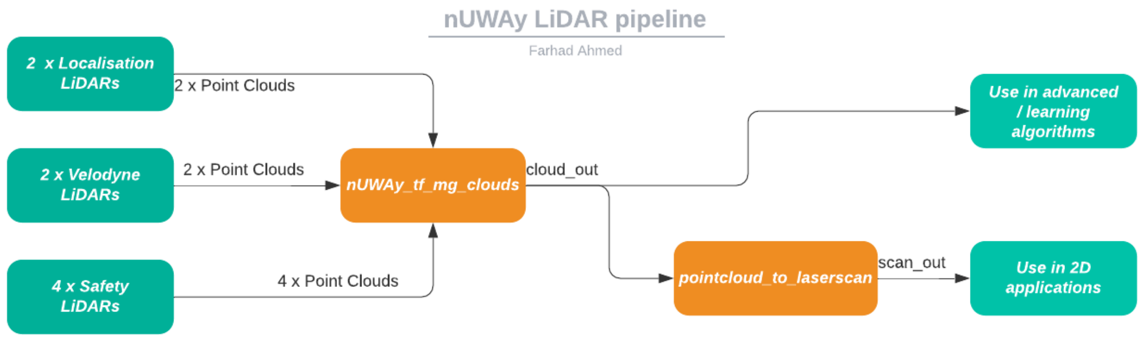

Figure 18 nUWAy LiDAR pipeline

4.2. nUWAy Odometry Pipeline

The odometry messages were separately derived from the GPS + IMU unit and the LiDAR pipeline. As described in Section 3.4.8, the two odometry messages were fused, as shown in Figure 19, and was then made available to decision-making packages.

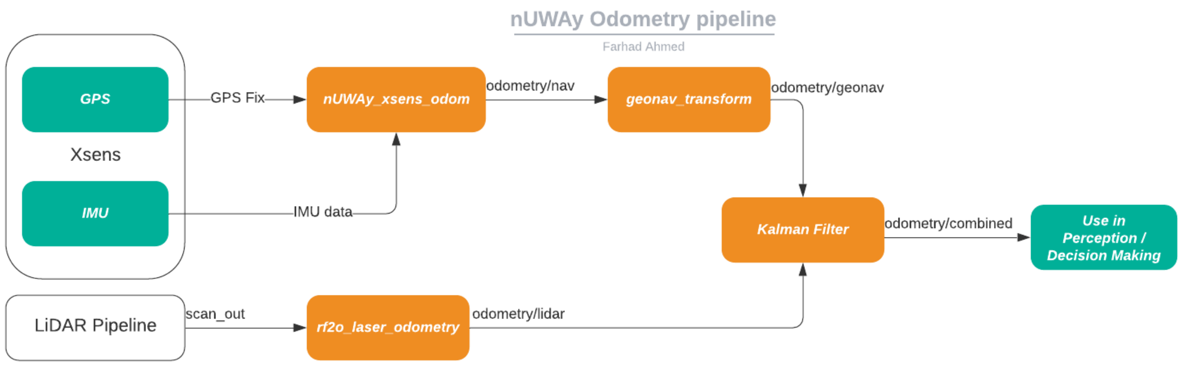

Figure 19 nUWAy odometry pipeline

4.3. second-ROS on nUWAy

The 3D merged point cloud from Section 4.1 was fed into the second-ROS framework described in Section 3.4.9, as shown in Figure 20. This node generates bounding box detections which can be further used in decision-making systems.

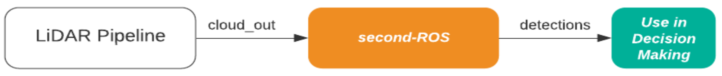

Figure 20 second-ROS on nUWAy

5. Results

5.1. Point Cloud Operations

The LiDAR pipeline was successfully set up on the nUWAy. Figure 21 shows the 3D visualisation of a merged point cloud. The white points at the top are from the “Localisation” LiDARs, the red points surrounding the bus close to the ground level are from the “Safety” LiDARs, while the multi-coloured scans are from the “Velodyne” LiDAR which also shows the intensity. Green represents a highly reflective surface, while the red colour represents a minimally reflective surface.

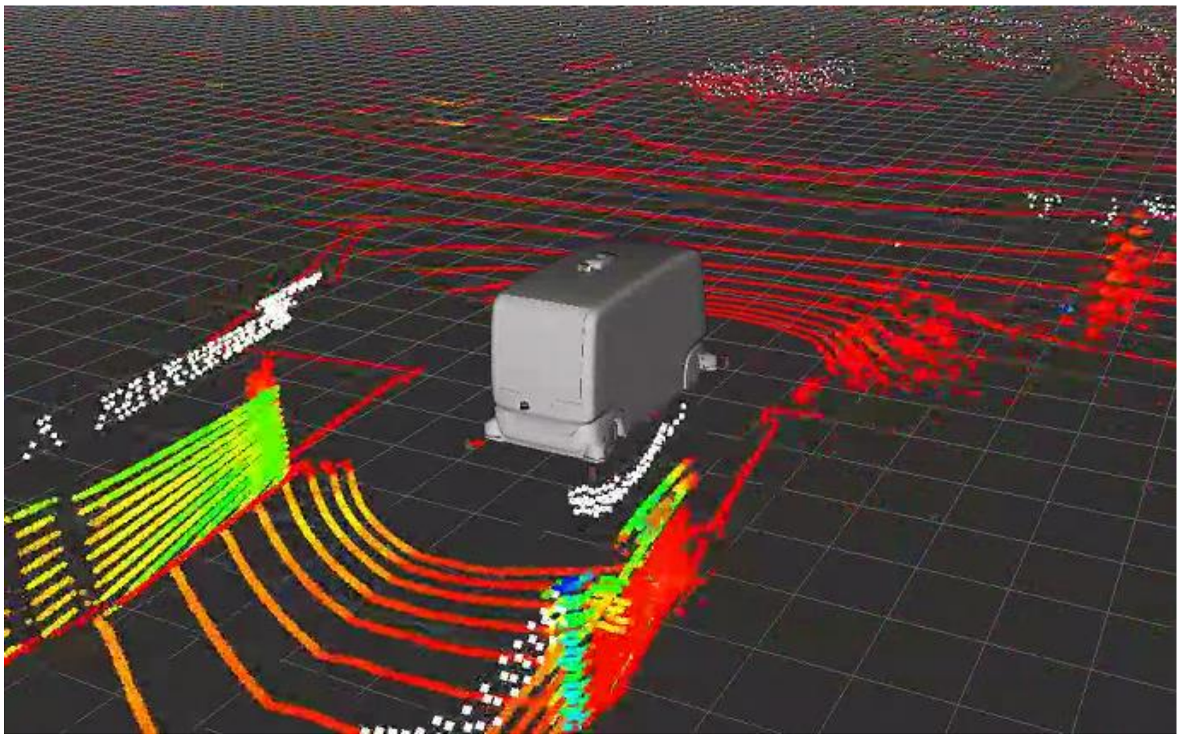

Figure 21 Merged point cloud

5.2. Odometry

The nUWAy was driven in the parking lot south of EZONE Central building at UWA to test for the accuracy of the different 2D odometry methods. The vehicle was made to complete one full loop around the area including a few turns including 180° turns and driven both forward and reverse.

5.2.1. LiDAR Odometry

RF2O LiDAR odometry was successfully implemented, and the plot of the vehicle's trajectory from this test is shown in Figure 22. A few deviations were observed and have been marked on the figure with red ovals. Both the deviations were observed after a significant turn or a change in the direction of motion was made.

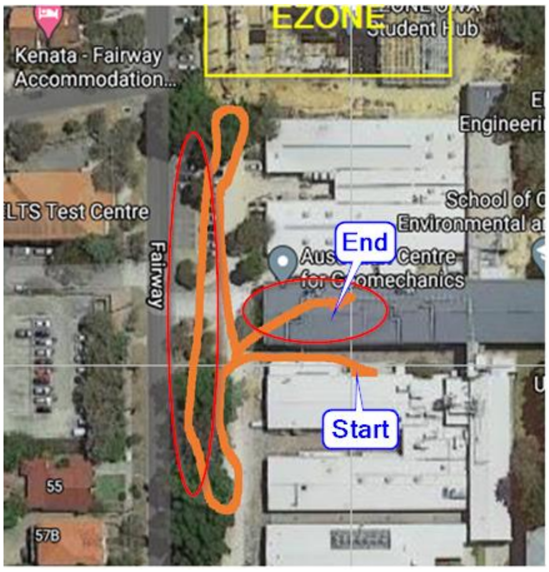

Figure 22 LiDAR odometry results

5.2.2. GPS + IMU Odometry Results

Odometry using GPS and IMU was also successfully implemented, and the plot of the vehicle's trajectory is shown in Figure 23. The tracking was very accurate due to strong GPS signals; however, the localisation was affected too near to the EZONE. The weak GPS signals caused due to the vehicle being surrounded by dense trees and tall buildings led to this deviation.

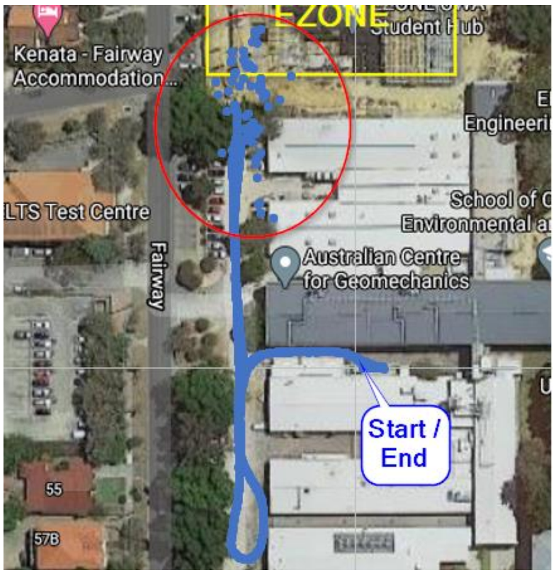

Figure 23 GPS + IMU odometry results

5.2.3. Fusion Odometry Results

The fusion of the two odometry signals was implemented with moderate success, and the plot of the vehicle's trajectory is shown in Figure 24. It can be observed that this plot mostly tracks the GPS generated plot from Figure 23, except for the area where GPS signals are weak. In such a case, it tries to extrapolate the odometry with the help of LiDAR. The result is not ideal but better than using one of the above methods alone.

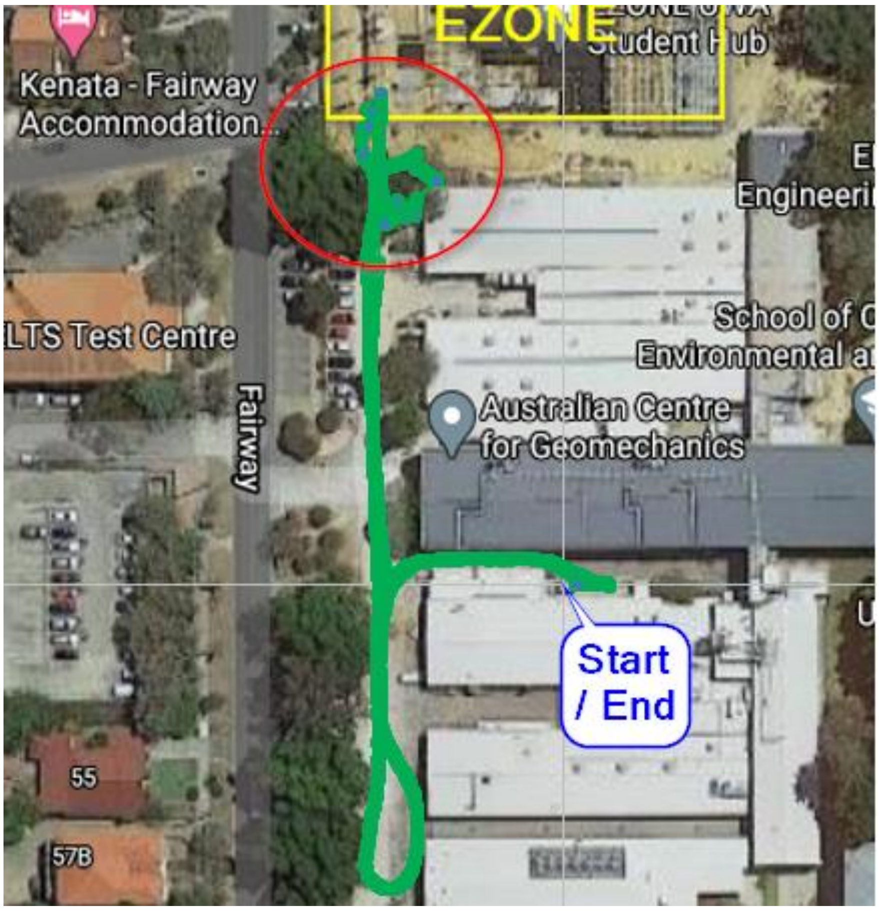

Figure 24 Fusion odometry results

5.2.4. Comparative Study

The x-axis position, y-axis position and the orientation over the time of the test has been plotted in Figure 25, Figure 26 and Figure 27, respectively. Due to a problem with the LiDAR odometry sampling, there is a varying offset on the plot over time. However, the plots can still be read meaningfully, and inferences can be drawn from them.

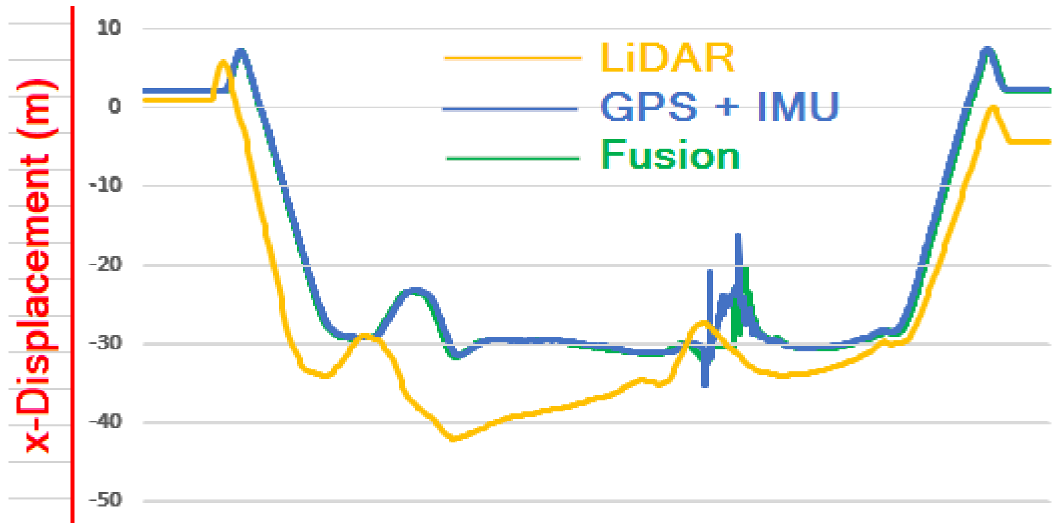

Figure 25 Odometry x-axis comparison

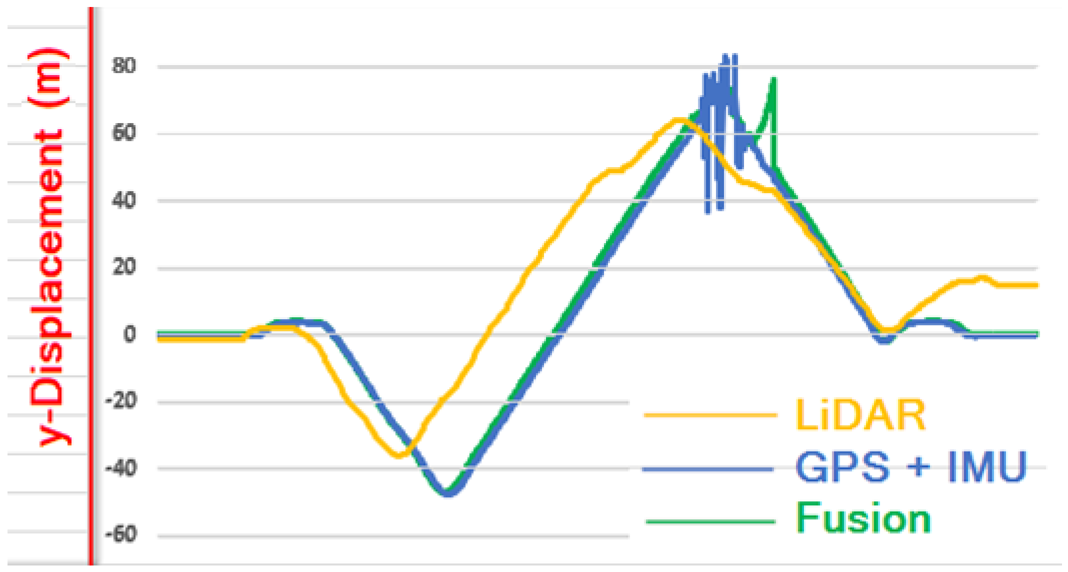

Figure 26 Odometry y-axis comparison

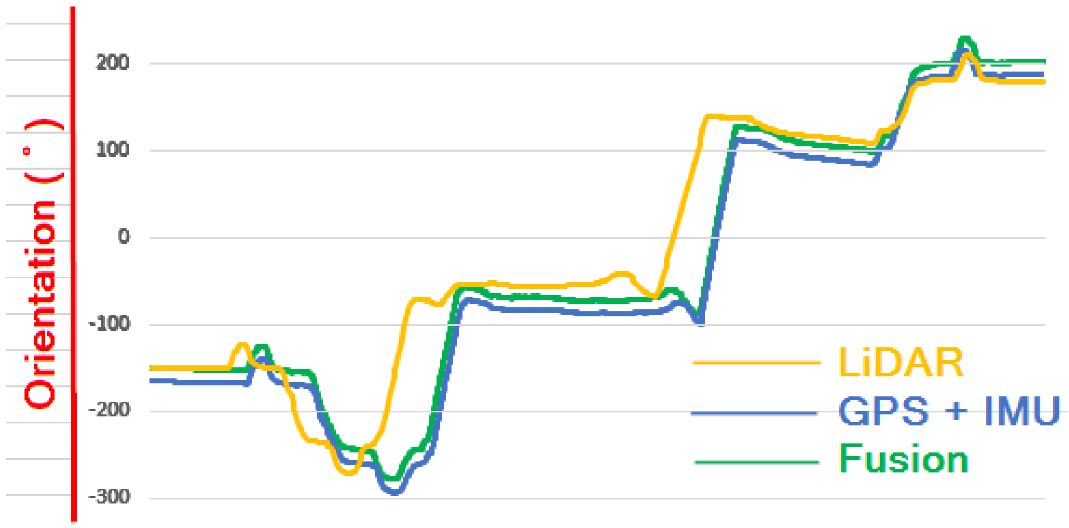

Figure 27 Odometry orientation comparison

5.3. Object Detection

Although object detection using second-ROS was attempted, the results were not satisfactory. The detections were random and were not continuously tracked, i.e., flickering over time. A screenshot of the detections has been shown in Figure 28.

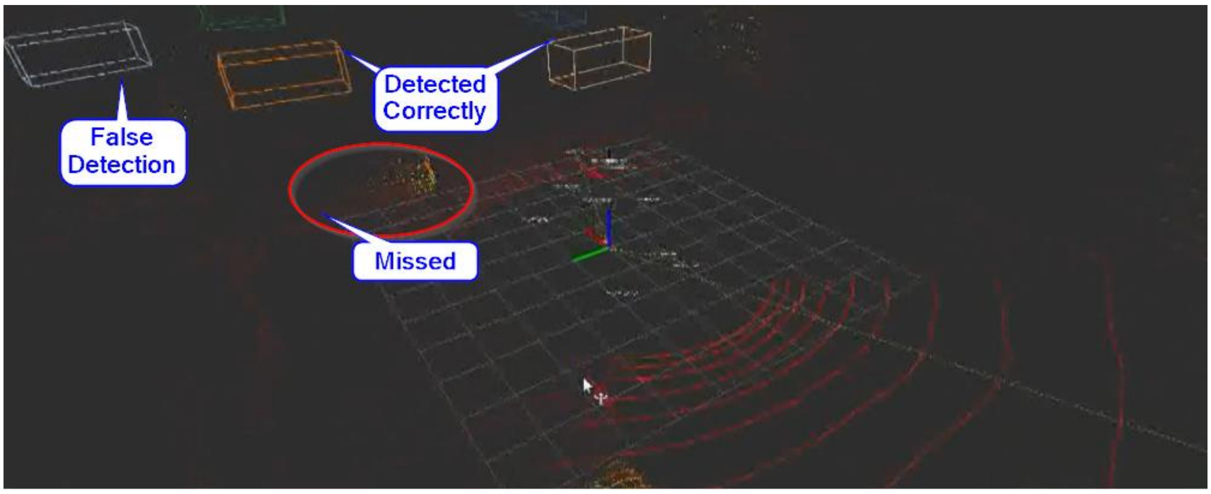

Figure 28 second-ROS output

6. Observations & Discussions

6.1. Point Cloud Operations

The following were observed regarding the point cloud operations:

- Creation of the LiDAR pipeline, using "nUWAy_tf_mg_cloud" to merge data from multiple point-clouds, was essential because none of the LiDARs on the shuttle gives a full 360° scan which is crucial for many navigation packages.
- Although developed specifically for the nUWAy, the package can be easily modified to be used on other sets of point-clouds.

6.2. Odometry

6.2.1. GPS + IMU Odometry

The following were observed about the odometry using GPS and IMU:

- Position estimate is highly accurate in open areas but suffers in areas with substantial tree cover or tall buildings, where the positional values start doing discrete and abrupt jumps.
- There is no accumulated drift of position over time, as the current position measurement is not related to the previous.
- The absolute orientation on the earth is initially inaccurate but corrects itself as more movement is made by the vehicle.
- GPS + IMU data can be used as an excellent initial estimate to the mapping systems.
- This form of odometry barely uses any resources as not much computation is involved, and is highly suited for an AV.

6.2.2. LiDAR Odometry

The following were observed about the odometry using LiDARs:

- The estimation is smooth, and there are no discrete jumps in position or orientation as can be seen from the smoothness of the curves in Section 5.2.
- Errors are accumulated over time and tend to derail the odometry, especially the orientation, entirely.
- There is no means of finding out the initial pose, and it needs to be fed externally to this algorithm.

- As it is dependent on comparing consecutive LiDAR scans, a dense scan is required for accurate computation, and hence suffers in open areas.
- This method is compute-intensive and should ideally be operated on high-performance systems only. Significant processing delays were observed in the test as it was observed that the data record rate was much lesser than the programmed scan rate.

6.2.3. Fusion Odometry

From Sections 6.2.1 and 6.2.2, it is evident that no single source of odometry on the nUWAy is capable of being sufficiently accurate and hence, the fusion method helped to improve the pose estimation better.

The following are the advantages of Sensor Fusion:

- Fused data provides a balance between the two individual methods, thus mitigating the ill effects of both and following the right pose very closely.
- The outcome is smooth and does not have any scattered points.
- It can provide a reasonable position estimate in both open and obstructed areas because LiDARs and GPS+IMU complement each other at their weak spots.
- GPS ensures no accumulated drift and auto pick-up of the initial position.

However, there are a few areas of concern, as listed below:

- The orientation determined by the fused odometry was slightly offset from the actual orientation. As seen in Figure 29, the arrows indicating orientation must be aligned towards the direction of motion, but they are aligned slightly away.
- Another problem is the high resource usage by the LiDAR processing nodes, which necessitates high computing power.

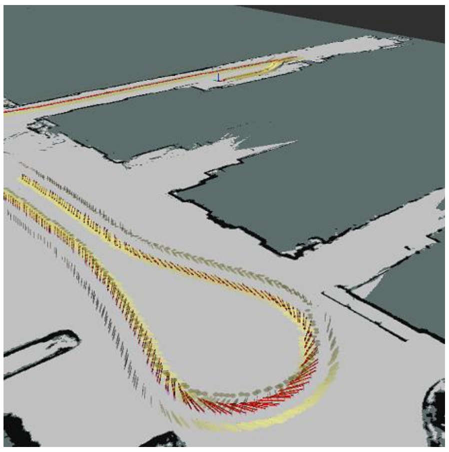
Figure 29 Inaccurate orientation in odometry results

6.3. Object Detection

Object detection using neural networks posed considerable challenges:

- The existing implementations, including "second-ROS", have been trained and tested on KITTI dataset which makes use of a High Definition 64 channel Velodyne LiDAR as shown in Figure 11. The LiDAR output from the nUWAy is significantly less dense, and hence the implementations fail on it.
- Unlike camera vision, LiDAR point cloud data is much complicated. Hence, the trained neural networks are not easily portable, i.e., the networked trained on a specific structure or configuration of LiDAR point cloud may not be able to operate correctly with input from a differently configured LiDAR.
- Training a neural network would require preparation of ground truth. In the case of LiDARs, it means, there is a need for labelling each 3D point-cloud frame which is a huge task and could not be done within the resource limits of this research project.

7. Conclusion

A pipeline for the LiDAR and another one for the odometry was developed. The unique requirements of nUWAy were kept in mind while developing these pipelines. The packages were successfully installed and tested on the nUWAy, and an analysis of the experimental results was carried out.

While both GPS+IMU based and LiDAR-based odometry had their advantages and disadvantages, the fusion method ensured that the best of both were picked up and fused. In particular, the LiDAR odometry provided smooth updates to the vehicle position while the GPS corrected any drifts over time. There are some concerns, however, over the computational resource requirement and the absolute orientation accuracy.

Object detection using neural networks was not of any significant success, but the causes for the same were determined, and the potential solutions were identified.

A few topics for further investigation have been identified in Section 7.1.

7.1. Topics for Further Investigation

This research project paves the way for further development, not only for the nUWAy platform but also for the robotics community in general. The following are a few ways in which this research can be expanded:

- The odometry message outputted by the pipeline can be implemented into the navigation stack.
- Instead of continuous fusing of sensor data, the signal quality information of the GPS can be used to determine when the GPS fix is accurate. Then the localisation may purely be done by the GPS and only when the fix is not accurate may the LiDAR be allowed to take over for short periods. This method would also save on computational resources.
- The "nUWAy_tf_mg_clouds" can be extended into a generic program which can take in LiDAR and transform information automatically from the vehicle description and effectively work on any sensor combination.

- Ground truth data may be prepared for:
    - Validation / Quantitative Analysis
    - Training of the Neutral Networks
- Other state-of-the-art CNNs like Part-A2 may be ported to ROS using "second-ROS" ROS implementation as a reference.
- An end-to-end pipeline may be developed integrating the object detection framework with local path planner to avoid obstacles.

All of these would contribute to the aim of making the operation of AVs safe and efficient.

References

[1] Y. Tian *et al.*, "DeepTest," *International Conference on Software Engineering*, ICSE '18, ACM, 2018, pp. 303-314.

[2] C. Badue *et al.*, "Self-Driving Cars: A Survey," 2019.

[3] J. Lutin, "Not If, but When: Autonomous Driving and the Future of Transit," *Journal of Public Transportation,* vol. 21, no. 1, pp. 92-103, 2018.

[4] "Singapore : Enhanced mobility at Gardens by the Bay with new wheelchair accessible shuttle fleet sponsored by SMRT," 2016.

[5] "Autonomous Shuttle Bus - The REV Project," 12/12/2020, 2020; http://revproject.com/vehicles/nuway.php.

[6] M. Jaimez, J. G. Monroy, and J. Gonzalez-Jimenez, "Planar odometry from a radial laser scanner. A range flow-based approach," IEEE, 2016, pp. 4479-4485.

[7] J. Zhang, and S. Singh, "Low-drift and real-time lidar odometry and mapping," *Autonomous robots,* vol. 41, no. 2, pp. 401-416, 2016.

[8] A. Censi, "An ICP variant using a point-to-line metric," IEEE, 2008, pp. 19-25.

[9] A. Diosi, and L. Kleeman, "Fast Laser Scan Matching using Polar Coordinates," *The International journal of robotics research,* vol. 26, no. 10, pp. 1125-1153, 2007.

[10] "geonav_transform - ROS.org," 12/12/2020, 2020; http://wiki.ros.org/geonav_transform.

[11] T. Moore, and D. Stouch, "A Generalized Extended Kalman Filter Implementation for the Robot Operating System," vol. 302, pp. 335-348, 01/01, 2016.

[12] C. R. Qi *et al.*, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," 2017.

[13] V. A. Sindagi, Y. Zhou, and O. Tuzel, "MVX-Net: Multimodal VoxelNet for 3D Object Detection," 2019.

[14] S. Shi *et al.*, "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.

[15] Y. Yan, Y. Mao, and B. Li, "SECOND: Sparsely Embedded Convolutional Detection," *Sensors,* vol. 18, pp. 3337, 10/06, 2018.

[16] "Training Deep Neural Networks with Synthetic Data using LGSVL Simulator - Github," 12/12/2020, 2020; https://github.com/lgsvl/second-ros.

[17] M. Quigley *et al.*, *ROS: an open-source Robot Operating System*, 2009.

[18] T. Foote, "tf: The transform library," IEEE, 2013, pp. 1-6.

[19] "REP-105 - Coordinate Frames for Mobile Platforms - ROS.org," 12/12/2020, 2020; https://www.ros.org/reps/rep-0105.html.

[20] "The Coordinate System Transformation Relationship when SLAM is Built - CSDN," 12/12/2020, 2020; https://blog.csdn.net/supengufo/article/details/88561273.

[21] "urdf - ROS.org," 12/12/2020, 2020; http://wiki.ros.org/urdf.

[22] E. Ackerman. "Cheap Lidar: The Key to Making Self-Driving Cars Affordable," 12/12/2020, 2020; https://spectrum.ieee.org/transportation/advanced-cars/cheap-lidar-the-key-to-making-selfdriving-cars-affordable.

[23] "Real-Time Kinematic (RTK) - NovAtel," 12/12/2020, 2020; https://novatel.com/an-introduction-to-gnss/chapter-5-resolving-errors/real-time-kinematic-rtk.
[24] "GNSS/INS - XSens," 12/12/2020, 2020; https://www.xsens.com/gnss.
[25] "pointcloud_to_laserscan - ROS.org," 12/12/2020, 2020; http://wiki.ros.org/pointcloud_to_laserscan.