Implementation of a LiDAR system via Raspberry Pi for object detection and avoidance

By

Max Calcroft

School of Engineering

Faculty of Science and Engineering

University of Plymouth

Honours project submitted in partial fulfilment of the

requirements for the degree of

BEng (Hons) in Mechanical Engineering

May 2019

Supervisor Dr Asiya Khan

Abstract

Autonomous vehicles are redefining the transport industry but public perception regarding safety and the programming in of morality presents issues. Due to their infancy, sensor technology for the vehicles is still developing and an ideal solution is still contested. This report aims to investigate the implementation of a Hokuyo URG-04LX laser scanner with a Raspberry Pi micro-computer and demonstrate it's effectiveness for object detection and avoidance in varying conditions. The original vehicle was a disassembled mobility scooter equipped with ultrasonic sensors. These have now been replaced with a singular laser scanner with existing code being altered to allow laser scanner integration, resulting in autonomous driving of the buggy. This report presents a detailed analysis of the accuracy and effectiveness of laser scanner data for variable object detection, computational speed and comparison with existing ultrasonic sensor data. It was found that laser scanners perform well for basic object detection but did not respond well to reflective or dark surfaces. Laser scanners also outperform ultrasonic sensors when stopping distance and latency are compared but could benefit from a filtering algorithm to decrease the risk of false positives. Overall the laser scanner acts as an effective sensor for the buggy, showing it’s viability in detecting objects and acting as a small scale representation of autonomous technology, aiding with public perception and understanding.

Acknowledgements

This project was made possible through the guidance and support of these individuals:

Asiya Khan - my supervisor, for always providing helpful information and for being genuinely interested in my work being of the best quality

Rob Calcroft - for providing help on understanding Python and common coding issues

Karen and Chris Calcroft - for their grammatical feedback and support

Sahand Murad - For helping with work on the buggy and the collection of data

Katherine Hart - For her positive support and assistance with Python scripts

Trevor Bevan - For all the assistance within Brunel and help with the buggy

Contents

Abstract 1
Acknowledgements 1
List of Figures 3
List of Tables & Equations 3
Nomenclature 4
1.0 - Introduction 5
2.0 - Project planning 7
2.1 - Aim of project 7
2.2 - Objectives 7
3.0 - Literature review 8
3.1 - Understanding the basics of LiDAR 8
3.2 - Classification of objects 10
3.3 - Applications in industry 11
3.3.1 - Autonomous vehicle applications and comparison 11
3.3.2 - Aerial and other applications of LiDAR 14
3.3.3 - Plymouth University autonomous buggy 15
4.0 - Methodology 17
4.1 - Investigation 20
5.0 - Results and Discussion 21
5.1 - Stopping distance Results 21
5.2 - Clock time results 22
5.3 - Ultrasonic vs LiDAR results comparison 23
5.4 - RViz scenario comparison 25
5.5 - Discussion 26
6.0 - Conclusion 28
6.1 - Recommendations 28
References 29
Peer reviewed papers and Journals 29
Online articles and websites 30
Image sources 31
Books & manuals 32
Background reading 32
Appendices 33
Appendix A - LiDAR range values for all tests conducted 33
Appendix B - Autonomous Python script for buggy 35
Appendix C - Existing autonomous script for buggy 36
Appendix D - Risk Assessment 36
Appendix E - Gantt chart 37
Appendix F - Interim Report 38

List of Figures

Figure 1 - *The 6 levels of driving automation* - Obtained from (Vox, 2016)

Figure 2 - *Raw Point cloud data from LiDAR* - Obtained from (Velodyne, n.d.)

Figure 3 - *LiDAR point cloud* - Obtained from (Dwivedi, 2017)

Figure 4 - *Diagram of the LiDAR optics and encoders* - Obtained from (Renishaw, n.d.)

Figure 5 - *Tesla range detection capabilities* - Obtained from (Tesla, 2018)

Figure 6 - *How a trained machine sees* - Obtained from (pirzada, 2015)

Figure 7 - *MobilEye free-space representation* - Obtained from (pirzada, 2015)

Figure 8 - *Point cloud map of terrain* - Obtained from (OREFIND, 2013)

Figure 9 - *Hovermap scan of radio tower* - Obtained from (LiDAR-UK, n.d)

Figure 10 - *Autonomous Buggy Diagram - Obtained from (McEachen and Day, 2018)*

Figure 11 - *Raspberry Pi Model 3B* - Obtained from (Raspberrypi, n.d)

Figure 12 - *Object detection function* - Obtained from (Day, 2018)

Figure 13 - *I2C addresses for X&Y DAC's* - Obtained from (Day, 2018)

Figure 14 - *RViz running on the Raspberry Pi showing point cloud data for the test environment*

Figure 15 - *Simplified logic of autonomous script for buggy object detection*

Figure 16 - *LiDAR range values from a single 240° scan*

Figure 17 - *Updated internals of buggy with LiDAR integration*

Figure 18 - *Buggy & testing environment*

Figure 19 - *Graph of clock time results against number of measurements*

Figure 20 - *Graph of results from sensor stopping distance comparison*

Figure 21 - *RViz Reflective test image*

Figure 22 - *RViz Pedestrian test image*

Figure 23 - RViz Angular Reflective test image

Figure 24 - RViz Stationary test image

Figure 25 - RViz Low light test image

List of Tables & Equations

Table 1 - *Table comparing LiDAR's provided for project* (Data obtained from hokuyo-aut.jp and orbbec3d.com)

Table 2 - *Table of results from LiDAR stopping distances*

Table 3 - *Table of clock time results*

Table 4 - *Table of results from sensor clock time comparison*

Equation 4 - LiDAR range calculation - Obtained from (LiDAR-UK, n.d)

Nomenclature

ADS - Automated Driving System

DAC - Digital to Analogue Converter

DARPA - Defense Advanced Research Projects Agency

DDT - Dynamic Driving Task

ECU - Electronic Control Unit

FoV - Field of View

FPS - Frames Per Second

GPiO - General Purpose input Output

GPS - Global Positioning System

GPU - Graphics Processing Unit

HOG - Histogram of Orientated Gradients

IMU - Inertial Measurement Unit

IR - Infrared

I2C - Inter-Integrated Circuit

LiDAR - Light Detection And Ranging

NTSB - National Transport Safety Board

PCB - Printed Circuit Board

'psk' - Pre Shared Key (password)

Raspberry Pi - A low cost, credit-card sized computer that plugs into a computer monitor or TV (Raspberry Pi, n.d.)

ROS - Robot Operating System

RViz - ROS Visualization

SAE - Society of Automotive Engineers

SLAM - Simultaneous Localisation And Mapping

SSID - Service Set Identifier (name of a wireless network)

UAV - Unmanned Aerial Vehicle

USB - Universal Serial Bus

WHO - World Health Organisation

3D-DSF - 3 Dimensional - Double Stage Filter

1.0 - Introduction

Autonomous vehicles have seen a rapid development over the past decade and as technology continues to improve, the more prevalent they become. Autonomy in vehicles is dictated by the Society of Automotive Engineers (S.A.E) levels of driving automation (Fig.1) which characterises how much human intervention is needed to get the vehicle from A to B. Levels 0-2 require the user to perform all of the Dynamic Driving Task (DDT) whereas levels 3-5 expect the ADS (Automated Driving System) to perform the entire DDT while engaged (Kelechava, 2018). Companies such as Tesla and Waymo (a subsidiary of Google) have a range of vehicles out on the road today that are somewhat autonomous. Currently, Tesla has level 3 autonomy in all its vehicles requiring some form of human intervention and Waymo's vehicles have reached level 4 but are still in the pilot study stage. The ultimate aim is to create a self-driving car that has level 5 autonomy and can therefore handle any road/route without the need for human intervention which has been promised by Tesla for 2020.

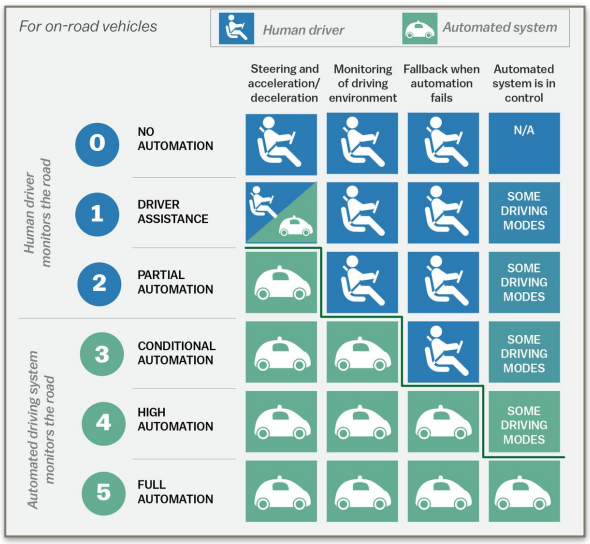


Figure 1 - *The 6 levels of driving automation* - Obtained from (Vox, 2016)

Waymo's vehicles have such accurate perception of the environment due to the use of Light Detection And Ranging (LiDAR) technology. LiDAR has been used primarily since the invention of Global Positioning Systems (GPS) in the 1980's for aircraft and satellite use in mapping terrain. LiDAR can now be seen in applications within areas such as

building restoration, meteorology, gaming and geology (Lidar-uk.com, n.d.). LiDAR uses point cloud data (Fig.2) to build up a 3D image of the environment through echo location via laser, similar to how a bat identifies its surroundings.

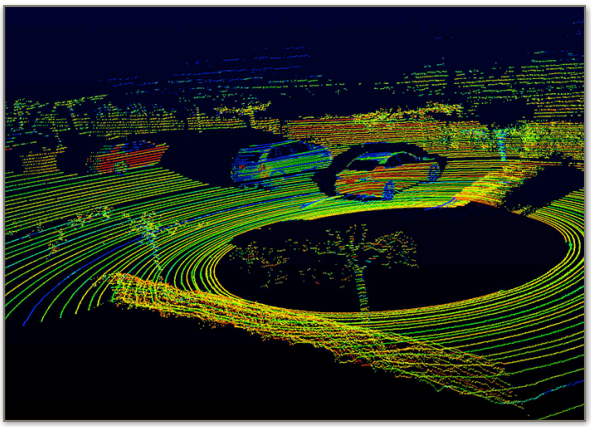


Figure 2 - *Raw Point cloud data from LiDAR* - Obtained from (Velodyne, n.d.)

This report aims to investigate the implementation of a LiDAR on a Raspberry Pi model 3B and the sensing of objects for avoidance. The LiDAR will be placed onto an autonomous buggy and subjected to a pre-determined route using the programming language Python. The buggy itself is a previous final year project built and run by two students investigating pedestrian recognition and the relaying of data to a wireless infrastructure hub using ultrasonic sensors . The buggy will therefore avoid objects in its path and provide a direct comparison between the two types of sensor for object detection.

According to statistics from the World Health Organisation (WHO), around 1.35M road traffic deaths occur every year (World Health Organisation, 2018) with 72% of crashes in Europe being down to road user errors (Thomas et al., 2013). The widespread use of autonomous vehicles could rapidly reduce that number however statistics such as 71% of people being concerned about loss of driving skills due to self driving cars and more than 60% being concerned about job losses implicates a hesitance by the general populous (Pettigrew, Fritschi and Norman, 2018). The integration of a small scale computer like the Raspberry Pi with LiDAR technology should demonstrate the effectiveness and accessibility of this technology, aiding the understanding and public perception of autonomous vehicles as they become more prevalent in everyday life. The consideration of morality and focus on social and ethical impacts during the design process is a problem that has never been faced before in engineering and this project acts as an important talking point that will hopefully encourage the conversation.

2.0 - Project planning

2.1 - Aim of project

The primary aim of this project is to implement a LiDAR sensor onto a Raspberry Pi for object detection. The secondary aim is to implement the LiDAR via Raspberry Pi onto the existing autonomous buggy set up by two BEng (Hons) Mechanical Engineering students, to expand upon it's object detection and avoidance capabilities when subjected to a pre-determined route using the programming language Python. By writing Python scripts, the object detection capabilities of the LiDAR should be able to integrate with the motion scripts already written for the buggy.

The simplicity of the Raspberry Pi when combined with the buggy and LiDAR should provide a good visual example of the accessibility and simplicity of the technology used in autonomous cars today.

The objectives below have been selected to reflect the aims and help to achieve them.

2.2 - Objectives

- Conducting a literature review around LiDAR basic principles and applications to understand the scope of LiDAR today and how it works
- Communication between LiDAR and Raspberry Pi reached to allow sensing of environment
- Learn the programming language Python for the understanding of existing code for operation of the buggy
- Achieving communication between the Raspberry Pi and Autonomous Buggy by running pre-existing code
- Writing of new code to integrate LiDAR sensing functionality with existing Python scripts for buggy

- Implementation of a LiDAR onto autonomous buggy allowing sensing of objects and collision avoidance when subjected to a predetermined route
- Testing of autonomous buggy with LiDAR installed
- Evaluation of results and comparison of existing ultrasonic sensors against LiDAR system

3.0 - Literature review

3.1 - Understanding the basics of LiDAR

LiDAR is essentially a form of technology used for sensing the world around it. It works by sending out green or near Infrared (IR) light, scanning the surrounding area (Fig.3). The time taken between the laser being sent out and bouncing off the object and being received again allows the distance of the object to be determined. This is done by the simple equation seen in Equation 1.

Figure 3 - *LiDAR point cloud* - Obtained from (Dwivedi, 2017)

$$[(traveltime) * (speedoflight)]/2 = Distance$$

Equation 1 - *LiDAR range calculation* - Obtained from (LiDAR-UK, n.d)

When the echo beam from the laser returns back it is reflected by a rotating mirror that matches the angle of each beam as it returns. This echo beam is then fed through an infrared transmitter diode to a photo diode receiver allowing distance to be calculated (Fig.4).

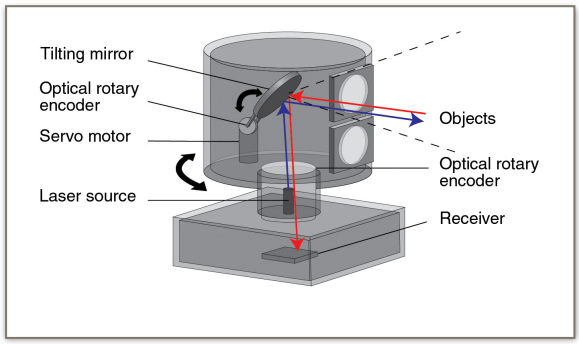


Figure 4- *Diagram of the LiDAR optics and encoders* - Obtained from (Renishaw, n.d.)

As LiDAR's are usually implemented on moving vehicles, a system known as an Inertial Measurement Unit (IMU) allows the movement of the vehicle to be taken into account for distance calculations or allows orientation to be accounted for aerial

applications. A GPS allows the exact position to be known. The combination of data gathered from the IMU and GPS allows a 3D static point to be determined. The IMU, GPS and LiDAR are connected to an onboard computer and produce 3D point cloud data from the echo beam pulses that can be either stored in mapping/ topography applications or act as an input to determine a vehicles movement via an Electronic Control Unit (ECU) (Agarwal, n.d.).

For ground based purposes two spectrums of wavelength can be used for the range detecting lasers. 600-1000nm at lower power levels are considered 'eye-safe', an important factor due to the nature of the typical application. Lasers around 1550nm can also be used and deemed 'eye-safe' at much higher power levels however they tend towards higher range, lower accuracy purposes (LiDAR-UK, n.d).

Two LiDAR's were provided for use in this project. The models provided were a Hokuyo URG-04LX and a ORBBEC Astra S and both are suited towards student research projects due to the power over Universal Serial Bus (USB). As seen in Table 1, the Hokuyo model has clear benefits and was chosen as the LiDAR used for the project. Other LiDAR's include the LiDAR lite 3, a common cheap LiDAR that can be used on small robotic applications. The RoboPeak LIDAR -A2M8 is a more expensive option but has a 360° Field of View (FoV) allowing advanced peripheral vision (RobotShop, n.d.).

Type of LiDAR	Range	Accuracy	Angular Resolution	Scanning time	Price
Hokuyo URG-04LX	60-4095mm (240° FoV)	60-1000mm ±10mm 1000-4095mm ±1%	0.36°	100ms/scan	$1080
ORBBEC Astra S	400mm-2000mm	60° horizontal 49.5° vertical	Data not available	Data not available	$149.99

Table 1 - *Table comparing LiDAR's provided for project* (Data obtained from hokuyo-aut.jp and orbbec3d.com)

3.2 - Classification of objects

The classification of objects by any autonomous system allows the world to be interpreted around it into known factors, reducing uncertainty. The accuracy of this is an important factor in bringing autonomous vehicles into everyday use as the consequences when dealing with humans in an uncontrolled environment can be very severe.

Many different algorithms are used for classification of objects such as Real AdaBoost or 3D - Double Stage Filter (3D-DSF) however the accuracy of identifying dynamic objects and their paths along with the processing time required for all of this calculation has room for improvement. Processing times of 0.002s for each object when compared to average human reaction times of 0.25s (Jain et al., 2015) may seem quick but this time is only for one object and ideally needs to be as low as possible if autonomous cars are to become the norm replacing human drivers. (Yoshioka et al., 2017) (Wu et al., 2018).

Data obtained from a LiDAR is presented as 3D point cloud data. Each point has an x, y and z coordinate and can be clustered together based on proximity to formulate objects such as pedestrians, cyclists, cars or background. Lower density clusters are typically identified as background, however accuracy usually drops off around 30m. Methods such as Density-Based Spatial Clustering of Applications with Noise (DBSCAN) can be used to appropriately filter out unwanted digital noise (Yoshioka et al., 2017).

Objects are clustered together using 4 steps. First the individual points are localised relative to each other within the environment surrounding the LiDAR sensor. Filtering occurs at this stage to identify false positives and isolated points that are then removed. Objects are then segmented from the background and points are assigned to either nodes or edges forming the basis for the next stage of feature extraction. This stage assigns shapes to objects and gives context relative to their environment such as cars in a line down a street, or lampposts being distributed along the pavement allowing the type of object to be identified. Features are then classified

based on context and clustering of points. This is achieved by the process of manually training a neural network to recognise these objects based on past experiences (Golovinskiy, Kim and Funkhouser, 2009)

Wu et al's study of near pedestrian crashes used back propagation via a neural network to classify objects similar to Tesla's approach with their neural net. Other strategies include giving each cluster of point cloud data a score for which the probability of what class of object (e.g pedestrian, car, cyclist) is determined via logistic regression analysis. This is a predictive analysis tool used to describe data by explaining the relationship between one variable and one or more independent variables (Yoshioka et al., 2017).

3.3 - Applications in industry

3.3.1 - *Autonomous vehicle applications and comparison*

With advancements in technology, autonomy is becoming more prevalent within many industries. Autonomy has many benefits which have led to it becoming more widespread across many disciplines. Reduction of costs, less risk due to lack of human interaction, higher efficiency, increased quality are all factors that are driving this change.

Two companies leading the push towards an autonomous vehicle future are Tesla and Waymo. Tesla currently has several semi-autonomous models on the road with iterations in software being pushed to the car every few months. These cars are not without their limitations and several accidents over the past few years such as collisions with concrete barriers, tractors that could not be distinguished by 'Tesla Vision' and a stationary firetruck (Stewart, 2018) while in autopilot have made the public and authorities such as the National Transport Safety Board (NTSB) question the safety of autonomous driving further slowing it's progress (Goggin, 2018).

Tesla cars use a system known as 'Tesla Vision' to sense the world around them. The latest version of this system has 8 cameras around the car coupled with 12 ultrasonic

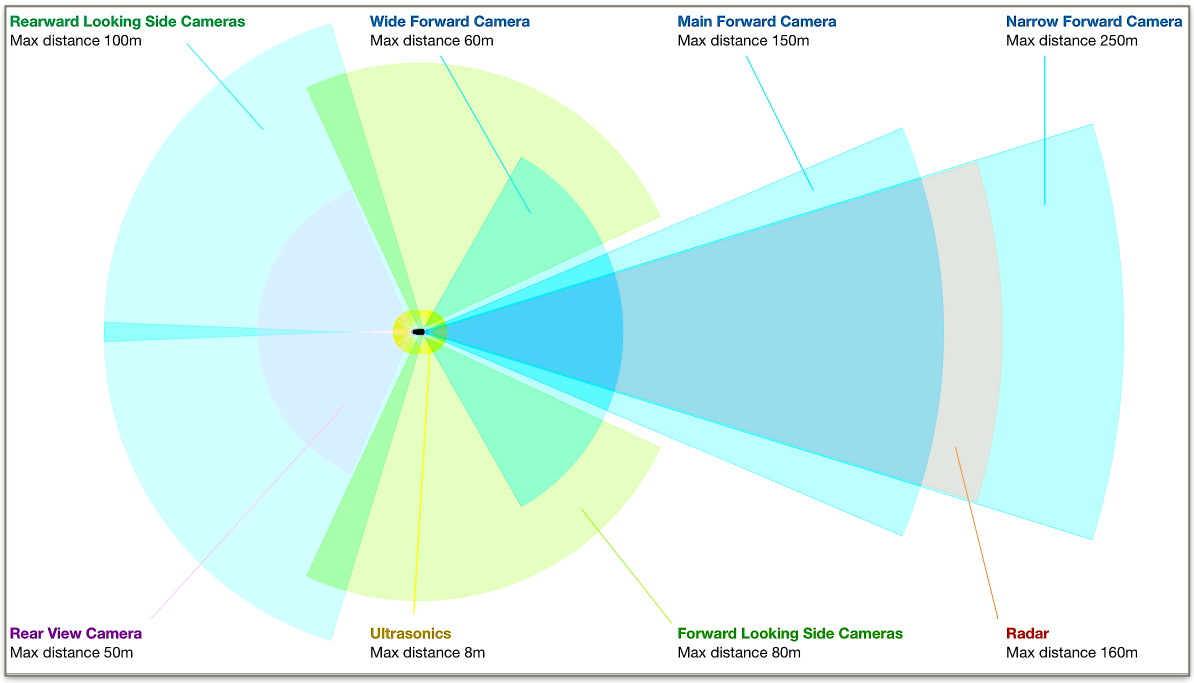


Figure 5 - *Tesla range detection capabilities* - Obtained from (Tesla, 2018)

sensors and radar for a 360° coverage with range up to 250m (Lambert, 2016) as seen in Fig.5.

'Tesla Vision' as previously mentioned is Tesla's own neural network. A neural network is vaguely based upon the human brain and works upon the process of an input, a hidden layer and an output layer with weights connecting these layers that are to be determined. Tesla Vision uses two systems, a deep neural network capable of learning by identifying only edges of objects and then being trained through images and videos to build neurons in order to perceive objects in the environment as seen in Fig.6. The second is MobilEye, a software chip capable of identifying road

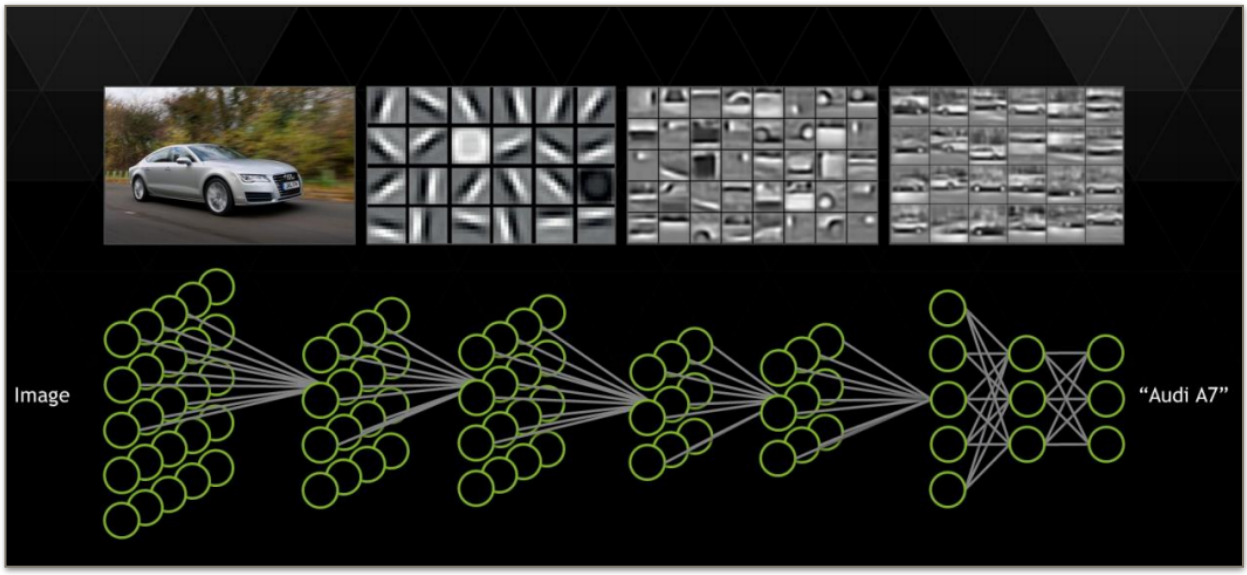


Figure 6 - *How a trained machine sees* - Obtained from (pirzada, 2015)

signs, lane markings, free space where the vehicle can freely move without collision (Fig.7) and even adjust steering and electronic stability depending on the road surface. The downside is many of these features are available but haven't yet been pushed in new software updates for Tesla vehicles. (pirzada, 2015)

Figure 7 - *MobilEye free-space representation* - Obtained from (pirzada, 2015)

Waymo's fleet of autonomous vehicles use LiDAR technology in order to build up an image of the surrounding areas. This forms part of their new vision system designed to take high resolution pictures of the world and analyse them while moving. Multiple sensors are installed around the vehicle, similar to Tesla, with a forward facing high resolution multi-sensor providing 360° coverage. The LiDAR, sensors and radar provide an unprecedented level of computer vision which has yet to be matched by other car manufacturers.

Waymo's cars use 3 LiDAR's, high resolution cameras and radar to perceive the world around the vehicle. The 3 LiDAR's are front mounted and are the Laserbear Honeycomb model, Waymo's own proprietary technology with a 360° FoV. Before Waymo's vehicles are released in an area, detailed maps of lanes, stop signs and traffic signals are built up which then feed into the algorithms that help drive the car using chips from Intel, similar to MobilEye in Tesla's (Waymo, 2019).

Other vehicles such as Volkswagen bring in other forms of data for cognition algorithms via car2x-communication, allowing the communication of car-to car as well as integration of 3D maps, weather conditions and on-road construction. Tesla currently uses cloud connectivity via the internet for data transfer but has no car to car communication as of yet (Silver, 2018)

Hirz and Walzel highlight the benefits of both sensor technologies. Camera based technology provides good geometric and object detail information and when combined with radar, provides accurate information in poor weather conditions much like LiDAR however, camera/radar based systems still suffer from light reflections and diverse edges. LiDAR provides accurate geometric data, functions well in low light and poor weather conditions and has a lower risk of false positives from reflections/light etc. however LiDAR sensors are still costly compared to camera/radar sensors and are largely obtrusive to a vehicles profile (Hirz and Walzel, 2018).

3.3.2 - *Aerial and other applications of LiDAR*

LiDAR technology has been predominantly used for aerial use until recently. LiDAR units are often attached to the underside of the airplane and are ideal for scanning terrain and mapping. The adversity to brightness and weather conditions allows detailed mapping (Fig. 8) regardless of conditions and cloud coverage and the type of light used reflects strongly off of vegetation. The nature of the beams from the LiDAR also allows several returns of the light when passing through trees. Both of these factors allow accuracy at high altitudes (Neon Science, 2014).

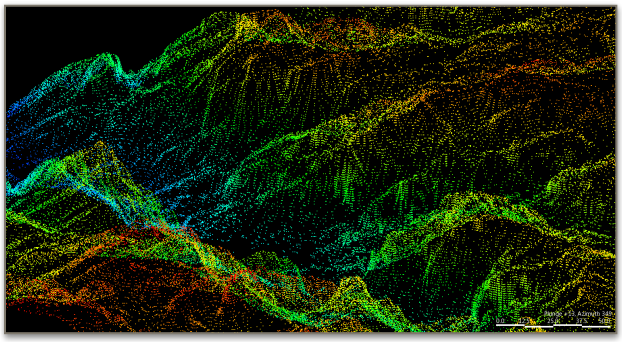


Figure 8 - *Point cloud map of terrain* - Obtained from (OREFIND, 2013)

Because of LiDARs mapping capabilities it can be used for applications in architecture, sewer & manhole maintenance, military and law enforcement for surveillance or via Unmanned Aerial Vehicle (UAV). Research has been conducted looking at the combination of ultrasonic and infrared sensors for simultaneous object detection and collision avoidance when implemented on a UAV (Gageik, Benz and Montenegro, 2015). Tests were carried out measuring reliability of distance information measured from both sensors and the ability to respond to a collision

course with a dynamic object and adjust flight path accordingly. The conclusions drawn from this experiment proved the effectiveness of low cost sensors and suggested the replacement of the IR/Ultrasonic sensors with a lightweight, cheaper sensor such as the RoboPeak-LiDAR.

3.3.3 - *Plymouth University autonomous buggy*

The autonomous buggy set up by two BEng Hons Mechanical Engineering students, Charlie Day and Liam McEachen, is the basis for this project and to be possibly expanded upon via LiDAR integration. The basis of the buggy was originally a mobility scooter that has been repurposed and the programming language Python has been used to write code for the buggy. The buggy has 5 preset speeds dictated by voltage levels that are manipulated by Digital to Analogue Converters (DAC's). Speeds 4 and 5 were used for tests conducted for this report.

The buggy uses 4 ultrasonic sensors seen in Fig.10 to detect objects using basic Python logic to execute a stop (2.5V) or reverse (1.13V) command if the distance between the object and the sensor is less than a certain distance. The ultrasonic sensors are pulsed every second taking distance measurements based on the time it takes the echo to come back. The Pi (Fig.11) connects into the Printed Circuit Board (PCB) via the General Purpose input Output (GPiO) pins and sends signals to the onboard DAC's to emulate the voltage the buggy would receive for omnidirectional motion. These voltages are sent to the buggy's micro-controller via the hijacked joystick connection which are then transmitted to the motors. Each DAC has an Inter-Integrated Circuit address (I^2C) which acts as a slave device to the Raspberry Pi and allows component to micro-controller communication via transfer of 8-bit packets (I2C Bus, Interface and Protocol, n.d.). A simplified version of the existing autonomous code found in Appendix C can be seen in Fig.12 demonstrating the object detection function of the buggy when using the ultrasonic sensors.

Using Pythons clocking functions, latency can be determined allowing the reaction time of the buggy to obstacles to be determined and accounted for in results. The

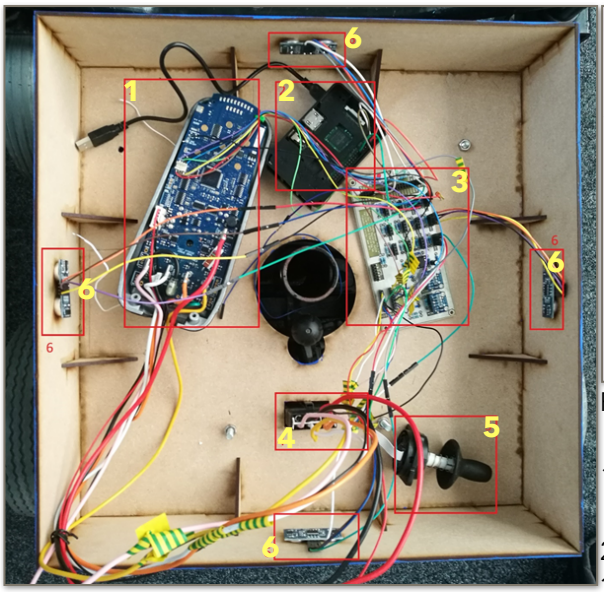


Figure 10 - *Autonomous Buggy Diagram - Obtained from (McEachen and Day, 2018)*

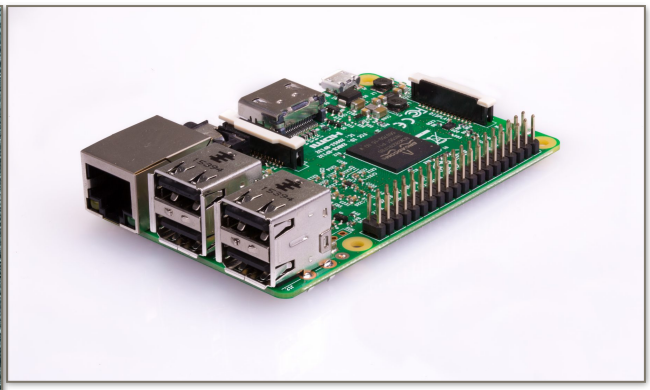


Figure 11 Key:

1. Buggy Control Panel
2. Raspberry Pi
3. Sensor PCB
4. Main Buggy Connection
5. Joystick
6. Ultrasonic Sensors

Figure 11 - *Raspberry Pi Model 3B* - Obtained from (Raspberrypi, n.d)

Adafruit MCP4725 module is also used for the interfacing between Pi and DAC's by providing a Python library similar to OpenCV.

Tests were carried out measuring accuracy of sensors, stopping distance and recognition time for collision avoidance when subjected to a path between two points. A range of speeds and distances were programmed and tested on the buggy and its ability to recognise objects and pedestrians was also tested. The Raspberry Pi struggled in terms of performance resulting in low Frames Per Second (FPS) and the Histogram of Orientated Gradients (HOG) classifier for computer vision was found to be superior (Day, 2018). Ultrasonic and processing time measurements were also found to be inconsistent.

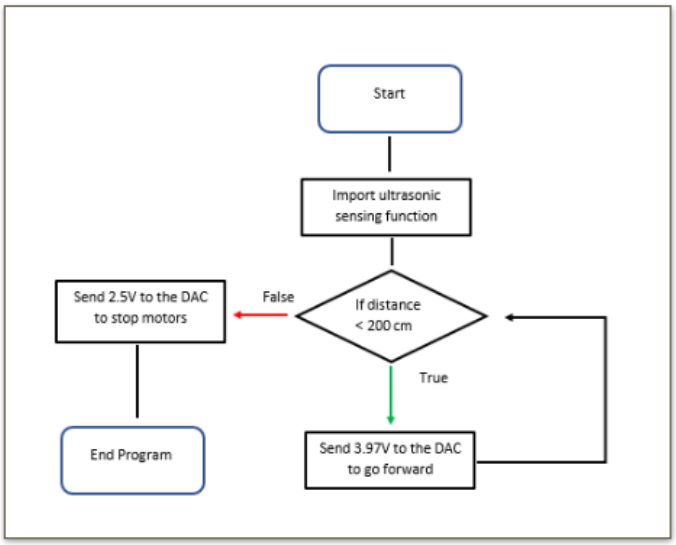


Figure 12 - *Object detection function* - Obtained from (Day, 2018)

4.0 - Methodology

Due to the buggy being a project from the year before, documentation created by the previous students already existed on how to set up the buggy and possible errors that could occur. Existing scripts seen in Appendix C were tested and modified for getting the buggy functioning as intended and implementing the LiDAR.

For the buggy to integrate with the LiDAR, the existing scripts for operating the buggy with the ultrasonic sensors would have to be altered to incorporate the LiDAR's range information. The programming language Python was used to write the code for the buggy due to all other scripts being written in this language. The time taken for these scripts to be executed was recorded using the inbuilt clocking function within Python.

Both Lubuntu, a lightweight version of the Ubuntu operating system and Robot Operating System (ROS) Kinetic, a version of the Robot Operating System that allows LiDAR integration, were imaged to the SD card of the Pi as the Hokuyo LiDAR only worked with Linux & Windows based systems. The Rospy package was imported for Python/ROS communication and the Adafruit module allowed Python code to change the DAC voltages.
The Y DAC and X DAC allowed for forwards/backwards and left/right movement respectively through using the I2C bus addresses. The use of separate addresses seen in Fig.13 allowed unrestricted omnidirectional movement.

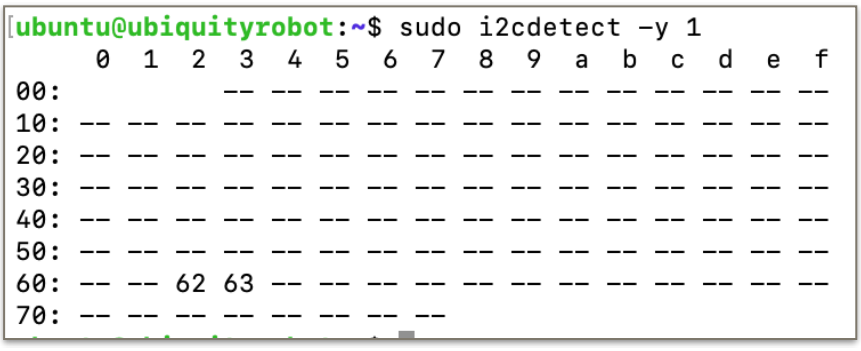


Figure 13 - *I2C addresses for X&Y DAC's* - Obtained from (Day, 2018)

As the buggy was being essentially 'hijacked' by the PCB, 2.5V had to be sent through the DAC by executing a Python script in order to trick the micro-controller into thinking it was correctly connected and as a result the joystick had to always be plugged in;

allowing the controller to power up and receive inputs but further complicating the buggy internals.

With Lubuntu installed on the Pi several packages had to be installed for the LiDAR to interact with and display range information. The package URG NODE for the Hokuyo URG-04LX LiDAR had to be installed to interact with the device. Ros Visualization (RViz), (Fig.14), shows a visual representation of the range data that the LiDAR sees in a 2D scan as point cloud data at 100ms intervals. The middle value of the range data in a 240° scan from the ROS laserscan topic was used as the reference value within the autonomous script and was acquired through an altered version of the script from Quigley, Gerkey and Smart (2015, p.105) and implemented within the simplified block

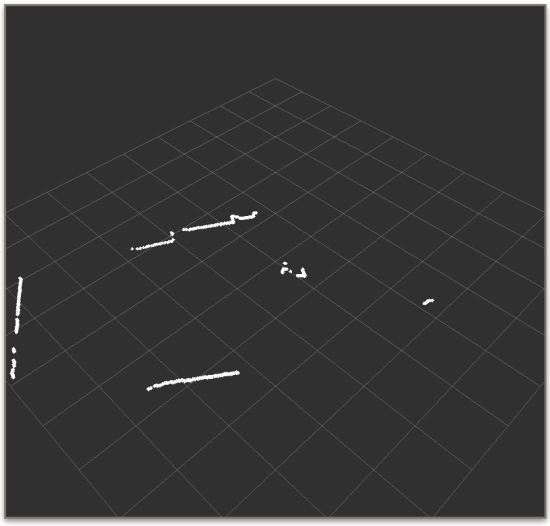


Figure 14 - *RViz running on the Raspberry Pi showing point cloud data for the test environment*

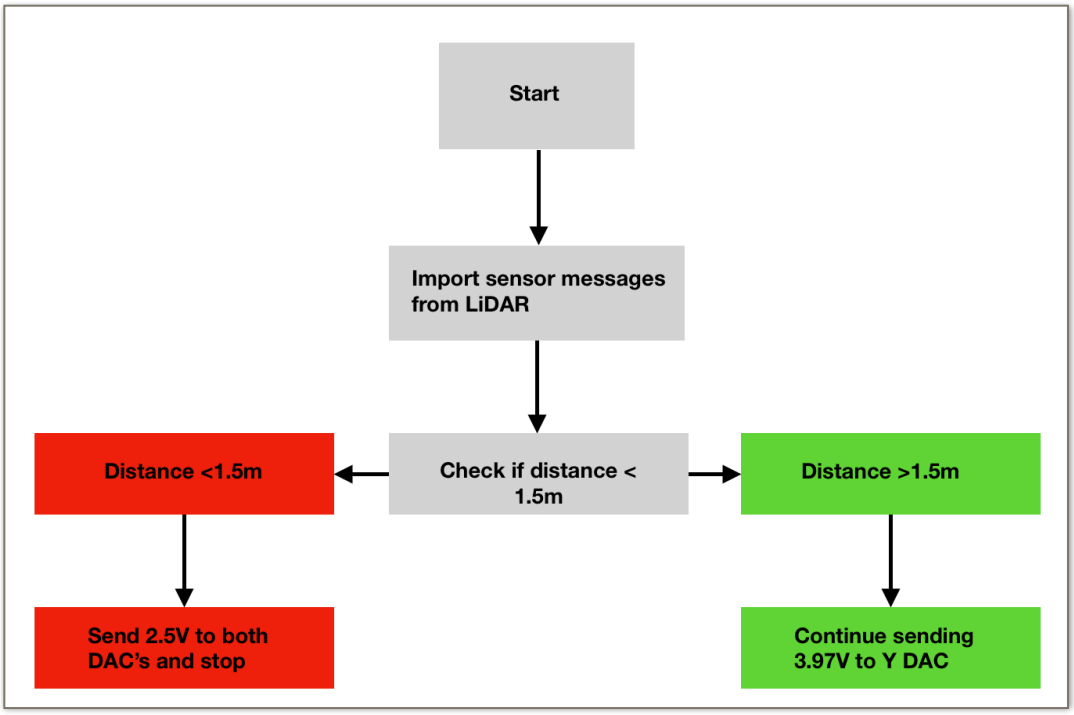


Figure 15- *Simplified logic of autonomous script for buggy object detection*

diagram as seen in Fig.15. The full script can be found in Appendix B. This middle range value takes the distance to the nearest object directly in front of the buggy but the entire spectrum of live range values can be displayed by using the inbuilt echo command within ROS as seen in Fig.16. The script imported only allows one or all range values to be imported and as the buggy was located within the housing, only the middle range value was used to prevent false positives.

The LiDAR was secured onto the buggy housing (Fig.17) and used the existing holes of the ultrasonic sensors for forwards vision as only the middle range value was acquired. The LiDAR connects into the Pi via power and data transfer over USB and a 5V portable battery provides power for both the LiDAR and Pi. The packages mentioned above are run along with the autonomous script. If the range imported from the LiDAR is less than a predefined value (e.g 1.0m), a stop function is executed sending 2.5V to the DAC for a set amount of time, otherwise the buggy will continue in a forward trajectory at a set speed defined by the voltage passing through the Y DAC.

New boards and connections could have been made for LiDAR integration, improving connections and functionality however due to the knowledge on the existing technology within the buggy and short timescale for this project, this method was the most feasible and effective.

1. Raspberry Pi
2. Joystick connection
3. Hokuyo LiDAR
4. Micro-controller
5. Printed Circuit Board
6. 5V portable battery

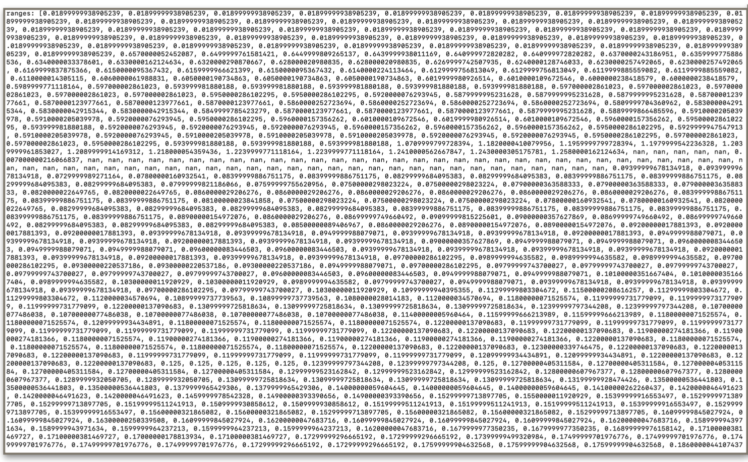


Figure 16 - *LiDAR range values from a single 240° scan*

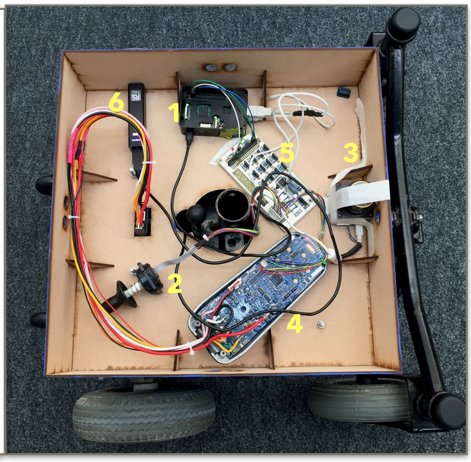


Figure 17 - *Updated internals of buggy with LiDAR integration*

4.1 - Investigation

The investigation of this project mainly focuses on the ability of the LiDAR to detect objects and its effect on the buggy's ability to avoid collisions. Various scenarios with low light conditions, reflective surfaces at multiple angles, simple stopping tests and different motion paths at varied speeds were tested in order to analyse the LiDAR's object detection capabilities and it's effectiveness when dealing with scenarios that cause errors in Tesla and Waymo's technology. Dynamic objects such as pedestrians would also be tested to emulate real world scenarios.

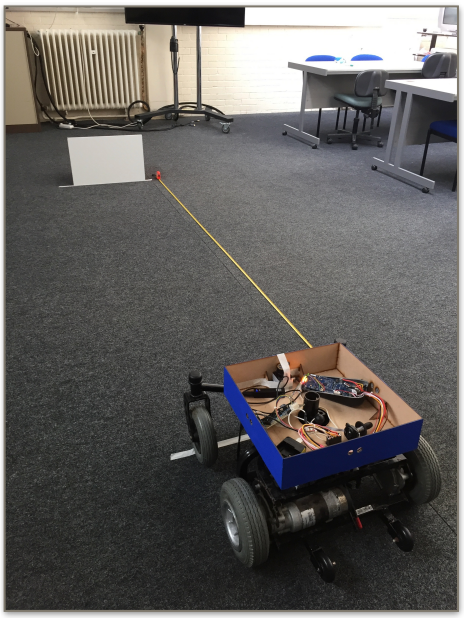


Figure 18 - B*uggy & testing environment*

LiDAR's generally do not respond well to reflective surfaces as the laser light is deflected and doesn't always find it's way back to the receiver resulting in varied distance measurements, providing diverse results for analysis (Yang and Wang, 2011). All tests were run at either speed 4 or 5 on the buggy microcontroller which corresponded to DAC voltages of 3.97V and 5.17V respectively.

The variables measured for this test were LiDAR recorded stopping distance, actual stopping distance, accuracy of range measurements and clock time of the Pi. Clock time and stopping distance were both compared using results from the previous iteration of the buggy allowing a direct comparison of LiDAR and ultrasonic detection capabilities. The LiDAR range information was recorded using 'rosbag', a feature of ROS to track all range measurements, diagnostic messages and errors over the time the test took place. A 3m track was used for the buggy with objects placed at the 3m mark as seen in Fig.18. 3m was chosen as quoted range accuracy for the LiDAR drops off after 1000mm to ±1% between 1000mm-4095mm. 1.0m was used as the distance threshold for when the buggy motors would stop due to a command issued by the Pi.

By measuring clock time using the inbuilt Python function, it could be observed if the variables implemented had any effect on processing times for the Pi.

This investigation is designed to emulate scenarios within the real world with objects that autonomous vehicles would encounter daily. Due to the limitations of working inside a controlled environment, these results give a general idea of LiDAR's capabilities on a small scale that are applicable to real world scenarios.

5.0 - Results and Discussion

In all tests, an object would be used for the LiDAR to detect and the surface of the object, speed and environment would be altered. A full table of the last 10 range values recorded before stopping are shown in Appendix A.

5.1 - Stopping distance Results

Stopping distance tests (Table 2) measured the final recorded distance by the LiDAR to the object that was to be detected. Angular reflective and reflective tests used a sheet of aluminium for detection set at 45° and 90° to the buggy's path respectively . All angular reflective tests were errors with incorrect range values recorded

LiDAR Stopping Distances							
		Final Recorded Distance to Object (m)					
Test Type	**Speed**	Test 1	Test 2	Test 3	Test 4	Test 5	Range Threshold (m)
Angular Reflective	4	2.3	0	2.9	2.5	3.0	1
Pedestrian	4	0.5	0.3	0.1	0.3	0.5	1
High Voltage	5	0.6	0.6	0.6	0.6	0.6	1
Low light	4	0.4	0.1	0.5	0.6	0.4	1
Reflective	4	1	2.1	1	1	3.5	1
Stationary	4	0.9	0.8	0.8	0.8	0.8	1

Table 2 - *Table of results from LiDAR stopping distances*

throughout resulting in the buggy crashing. Reflective tests performed better with only one crash and successful stops but recorded range values were still incorrect at times. Pedestrian tests involved an individual walking in front of the buggy during motions and this resulted in low recognition times with the final stopping distances to the pedestrian being significantly below the range threshold. Low light tests involved a dark object for detection in a blacked out room and showed a diverse range of final distances to the object. As seen in Appendix A the object was often not recorded until around 0.5m away. Stationary and high voltage tests both used a wooden square for detection and produced uniform results across all tests with minor variation.

High Voltage Tests	**Final Distance Recorded to Object (m)**				
Actual	0.45	0.47	0.49	0.46	0.46
LiDAR Recorded	0.4	0.4	0.4	0.4	0.4
% Error	11.1	14.9	18.4	13.0	13.0

Table 3 - *Table of high voltage stopping distance results*

High voltage tests (Table 3) used a voltage of 5.17V to run the buggy at the preset speed setting 5. This test was run in order to test stopping distances at higher speeds and show the actual vs recorded range measurements. Due to the range measurements only being displayed to 2 significant figures, it can be assumed that the LiDAR is reasonably accurate at recording the actual distance to the object, subject to ±1% as quoted previously but round ups to the nearest 0.1m. ROS automatically rounds range values up when not viewed in their raw format seen in Fig. 16 therefore reducing accuracy.

5.2 - Clock time results

Clock time was measured against stopping distance and the number of range values recorded. As seen below in Fig.19 there is very little variation in results as only one message is being imported into the script making it not very computationally intense.

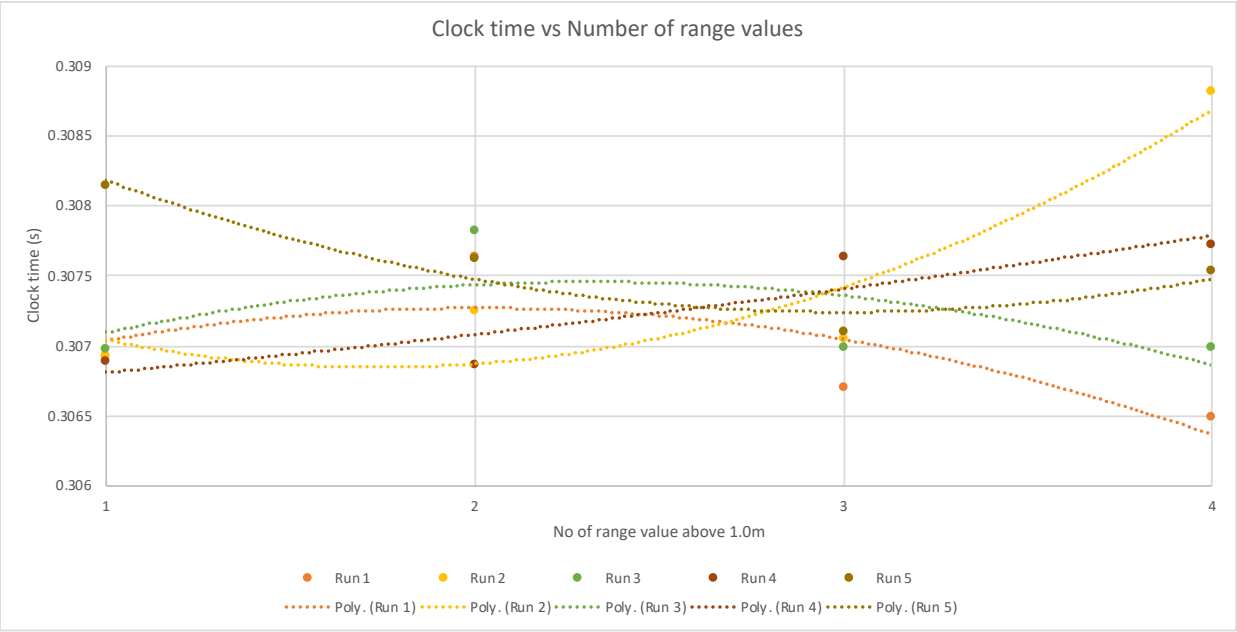


Figure 19 - *Graph of clock time results against number of measurements*

However earlier range values show more diverse results in terms of clock time. Differences in clock time values are very small however when put into the context of how long the Pi takes to complete these operations, they display significant variation.

5.3 - Ultrasonic vs LiDAR results comparison

Results from the previous iteration of the buggy with ultrasonic sensors can be seen above compared with results from the LiDAR at speeds 4 and 5 over 3 different tests (Fig.20). The stopping distance results show the distance taken to stop when given a range threshold of 1.0m. Points that are on the black line would count as crashes. Ultrasonic results can be seen to take further to come to a complete stop once the range threshold of 1.0m has been reached when compared to LiDAR data and both the 4 and 5 speed tests show more variation in stopping distances for ultrasonic data. Pedestrian tests show more variation for LiDAR data and performs worse when compared to ultrasonic tests possibly due to the use of only one laser being used for LiDAR range measurements. Clock times were also compared between these tests in Table 4 showing a significant difference between LiDAR and ultrasonic times with the latter taking almost 7x longer with little variation between tests using the same sensor.

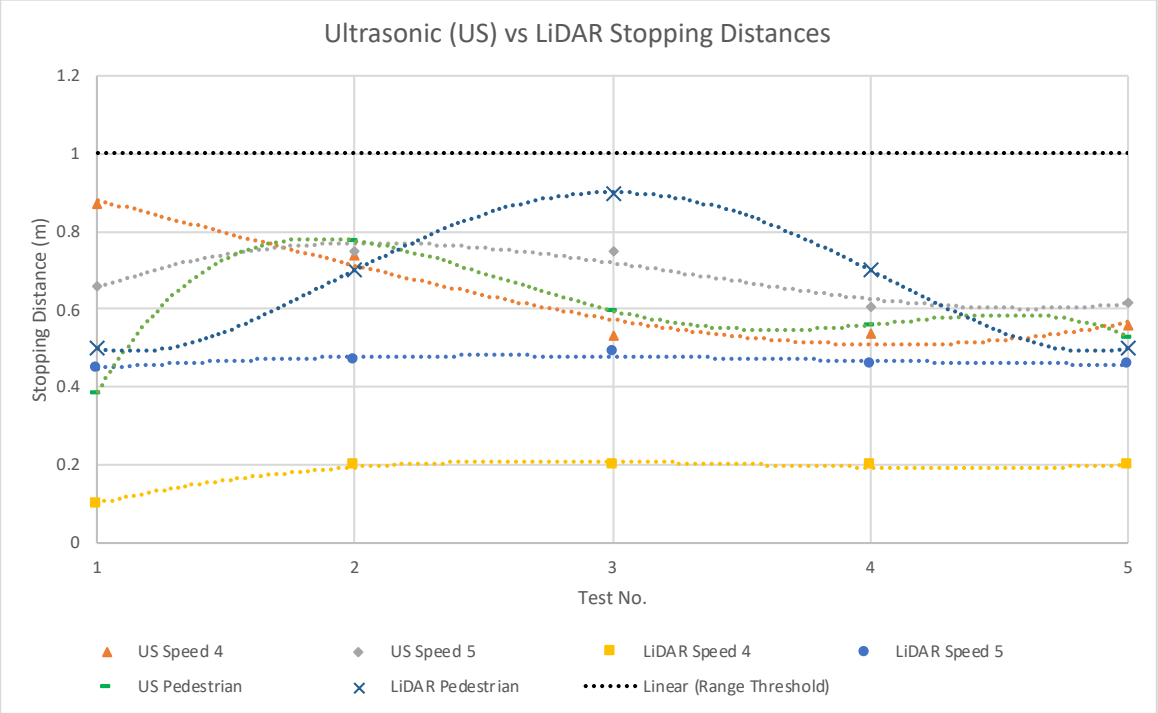


Figure 20 - *Graph of results from sensor stopping distance comparison*

Ultrasonic vs LiDAR Clock Times							
			Clock Times (s)				
Test Type	**Speed**	**Sensor**	**Test 1**	**Test 2**	**Test 3**	**Test 4**	**Test 5**
Stationary	4	Ultrasonic	2.334	2.33	2.086	2.349	2.313
Stationary	4	LiDAR	0.307	0.307	0.308	0.307	0.297
Stationary	5	Ultrasonic	2.349	2.339	2.087	2.711	2.399
Stationary	5	LiDAR	0.306	0.307	0.308	0.307	0.306
Pedestrian	4	Ultrasonic	2.57	2.351	2.121	2.526	2.921
Pedestrian	4	LiDAR	0.308	0.307	0.308	0.308	0.308

Table 4 - *Table of results from sensor clock time comparison*

5.4 - RViz scenario comparison

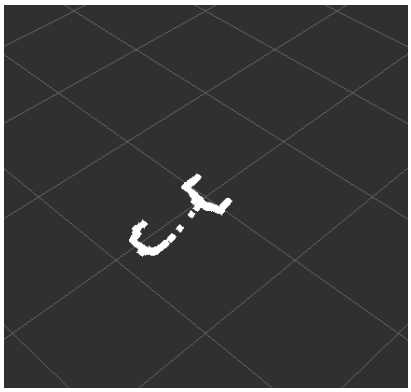


Figure 21 - RViz Reflective test image

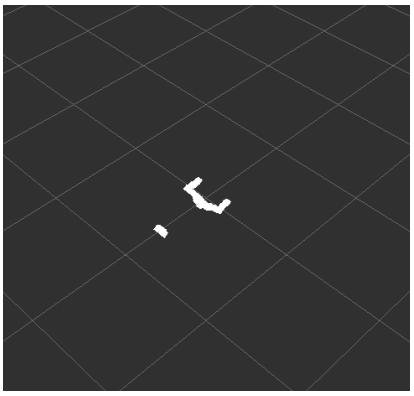


Figure 22 - RViz Pedestrian test image

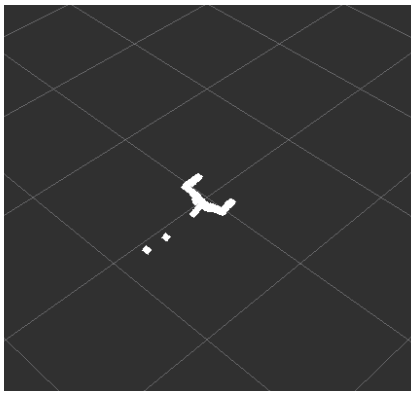


Figure 23 - RViz Angular Reflective test image

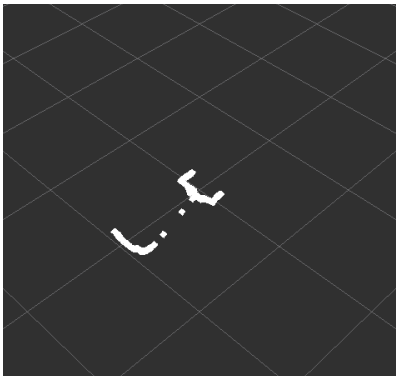


Figure 24- RViz Stationary test image

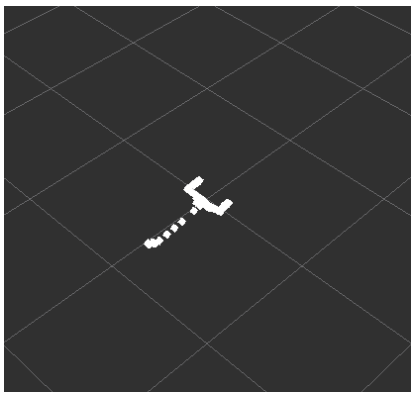


Figure 25 - RViz Low light test image

RViz images were captured for each scenario with the corresponding object for detection placed at a distance of 0.5m from the buggy to show differences in how the LiDAR perceived each object. On each image the U shaped point cloud distribution is the inside of the buggy housing being picked up by the LiDAR.

Pedestrians (Fig.22) show few point cloud points, but they are grouped together allowing the LiDAR to easily detect the object. Reflective tests (Fig.21) have some interference between the object and the buggy displaying point cloud data where no object is present. Angular reflective (Fig.23) and stationary tests (Fig.24) also show the same trend, possibly due to reflections or dust. Low light

tests (Fig.25) show some anomalous point cloud points and the object isn't picked up by the LiDAR.

5.5 - Discussion

Testing the LiDAR's effectiveness in a range of conditions adequately displayed the sensors advantages and weaknesses when compared with other technologies. The LiDAR was fairly accurate at recording range measurements and subsequently issuing stop commands to the motors however this could be improved. The buggy suffers with only being able to stop the motors and not apply brakes when an object comes within the range threshold leading to an average distance of 0.34m required to come to a complete stop amongst the speed 4 tests. Average stopping distance based on data from the RAC (Rac.co.uk, 2017) when scaled for the buggy equates to an average stopping distance of 0.48m. This data however does not take into account weight, ground surface etc. but still provides a good benchmark for comparison.

Errors were mostly seen in the reflective and low light tests (Appendix A). Figures 21, 23 and 25 show the LiDAR has difficulty with accurately locating and portraying these objects in point cloud format. This could be possibly due to the lasers deflecting and not reaching the receiver or being absorbed by the dark object due to the laser signal weakening with distance from the LiDAR, as range measurements were only ever recorded below 0.6m for these tests. Point cloud data suffers from errors mainly due to reflections, an uncalibrated LiDAR possibly due to vibrations on the buggy and mixed pixels whereby the depth coordinate of point cloud data is incorrectly displayed resulting in the effect seen in Fig.25 (Anil et al., 2013).

A peer reviewed study using a 2D LiDAR for the Defense Advanced Research Projects Agency (DARPA) grand challenge proved LiDAR's effectiveness in low light conditions when detecting road markings (Kammel and pitzer, 2008), but due to the use of only one datapoint and the type of LiDAR used, the buggy range information in these conditions was not viable and did not perform as expected. Other peer reviewed papers looking at the effect of dust particulates on LiDAR data found that information

transmittance of point cloud data was as low as 2% for high reflective surfaces and 6% for black surfaces (Phillips, Guenther and McAree, 2017). These studies used a more advanced LiDAR so these figures may not be directly transferrable but show significant limitations of LiDAR for certain surfaces. Similar studies have alleviated the issue via filtering techniques that do not rely on trajectory data, using processes of segmentation, clustering and feature recognition for reflective road signs (Riveiro et al., 2016).

Small percentage errors were observed between actual and recorded high voltage range measurements but this is due to LiDAR range measurements only publishing to 2 significant figures.

Fluctuations in clock time seen in Fig.20 could be due a plethora of factors such as resistance within the wires, vibrations from the buggy during motion exacerbating the loose connections via the GPiO pins or the Pi running slow. The latter is the probable cause as LiDAR's computational requirements vary for the Pi during operation.

LiDAR and ultrasonic sensor results show remarkably lower stopping distances and clock times as well as more uniformity in stopping distances for LiDAR. The LiDAR runs at a high refresh rate, did not have RViz running during testing and dealt with only one range value compared to the ultrasonic sensor which had to trigger multiple sensors resulting in more computational processing time and therefore an increased time and distance to stop. Data streaming via USB for the LiDAR is more reliable than the pins used for the ultrasonic sensor connections that would often come loose resulting in slow or no data transfer. LiDAR does struggle with pedestrian recognition compared to ultrasonic sensors but it is believed this is due to the fact that only one laser signal was measured.

These results agree with similar studies (Llorens et al., 2011) comparing ultrasonic and LiDAR sensors demonstrating LiDAR's superior accuracy and high efficiency, however it is also noted that ultrasonic sensors are generally more user-friendly and cheaper than LiDAR due to it's software and hardware requirements. Most common errors encountered with LiDAR technology appear to be alleviated via filtering

algorithms. Filtering algorithms would increase processing times but could possibly be reduced by upgrading or overlocking the computer used.

6.0 - Conclusion

In this project a Hokuyo LiDAR has been successfully implemented as a range sensor for the Raspberry Pi through the appreciation and understanding of the technology. The development of existing code through research allowed new Python scripts to be written allowing the buggy to be tested across a range of scenarios avoiding collisions with objects. LiDAR proved to be a superior technology when compared to ultrasonic sensors demonstrating high accuracy and clock time but was limited by fidelity of point cloud data and it's perception of reflective or dark surfaces. The use of only one range value from the LiDAR hindered its performance when detecting pedestrians but it still managed to maintain a low overall stopping distance. LiDAR has proved to be a viable technology for a vehicle of this scale and appropriately meets the function of a scalable testbed for autonomous technologies, but could benefit from filtering algorithms and other forms of sensors to improve it's overall detection performance.

6.1 - Recommendations

Further recommendations include the use of a 3D LiDAR with filtering algorithms applied to improve detection and segmentation of objects. The Sick MRS1000 provides 3D scanning with easy configuration so would be ideal for student research projects but does come with a high cost (Sick.com, n.d.). Simultaneous Localisation And Mapping (SLAM) technology and rotation of the buggy once the distance threshold is reached would allow it to be left to map out an area providing a point cloud map of its environment.

Several tests were run but all used the same range threshold value of 1.0m. By altering this the buggy's ability to come to a stop for both close and distant objects could be tested across a range of speeds.

Improved connections between Pi's GPiO pins and the PCB board and a housing for the LiDAR to prevent disconnections and reduce vibrations would increase accuracy of results. A larger portable battery is needed as the 5V battery has to be constantly recharged making testing difficult.

Processing times could be improved by overclocking the Pi as it's hardware does not allow upgrading of the GPU and code run on the buggy could be simplified, as currently several windows have to be open to run all necessary nodes and processes for the LiDAR.

These recommendations could improve the buggy's functionality, aid with understanding of the technology and make it's results more relevant for the ever-changing environment of autonomous vehicles.

References

Peer reviewed papers and Journals

Anil, E., Tang, P., Akinci, B. and Huber, D. (2013). Deviation analysis method for the assessment of the quality of the as-is Building Information Models generated from point cloud data. *Automation in Construction*, 35, pp.507-516.

Bonnefon, J., Shariff, A. and Rahwan, I. (2016). The social dilemma of autonomous vehicles. *Science*, 352(6293), pp.1573-1576.

Day, C. (2018). An investigation into the application of a Raspberry Pi as a microcontroller to an autonomous buggy with object avoidance capabilities and a comparison of computer visionmachine learning algorithms for pedestrian recognition. Undergraduate. Plymouth University.

Gageik, N., Benz, P. and Montenegro, S. (2015). Obstacle Detection and Collision Avoidance for a UAV With Complementary Low-Cost Sensors. *IEEE Access*, 3, pp.599-609.

Golovinskiy, A., Kim, V. and Funkhouser, T. (2009). Shape-based Recognition of 3D Point Clouds in Urban Environments. *International Conference on Computer Vision (ICCV)*, [online] pp.2-5. Available at: https://gfx.cs.princeton.edu/pubs/Golovinskiy_2009_SRO/index.php [Accessed 8 Mar. 2019].

Hirz, M. and Walzel, B. (2018). Sensor and object recognition technologies for self-driving cars. *Computer-Aided Design and Applications*, 15(4), pp.501-508.

Jain, A., Bansal, R., Kumar, A. and Singh, K. (2015). A comparative study of visual and auditory reaction times on the basis of gender and physical activity levels of medical first year students. *International Journal of Applied and Basic Medical Research*, 5(2), p.124.

Kammel, S. and pitzer, B. (2008). Lidar-based lane marker detection and mapping. *2008 IEEE Intelligent Vehicles Symposium*.

Llorens, J., Gil, E., Llop, J. and Escolà, A. (2011). Ultrasonic and LIDAR Sensors for Electronic Canopy Characterization in Vineyards: Advances to Improve Pesticide Application Methods. *Sensors*, 11(2), pp.2177-2194.

Pettigrew, S., Fritschi, L. and Norman, R. (2018). The Potential Implications of Autonomous Vehicles in and around the Workplace. *International Journal of Environmental Research and Public Health*, 15(9), p.3.

Phillips, T., Guenther, N. and McAree, P. (2017). When the Dust Settles: The Four Behaviors of LiDAR in the Presence of Fine Airborne Particulates. *Journal of Field Robotics*, 34(5), pp.985-1009.

Riveiro, B., Diaz-Vilarino, L., Conde-Carnero, B., Soilan, M. and Arias, P. (2016). Automatic Segmentation and Shape-Based Classification of Retro-Reflective Traffic Signs from Mobile LiDAR Data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(1), pp. 295-303.

Thomas, P., Morris, A., Talbot, R. and Fagerlind, H. (2013). Identifying the causes of road crashes in Europe. [online] Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3861814/ [Accessed 7 Mar. 2019].

World Health Organization (2018). *Global Status Report on Road Safety 2018*. [online] Geneva: World Health Organisation, pp.pg's 21-22. Available at: https://www.who.int/violence_injury_prevention/road_safety_status/2018/en/ [Accessed 6 Mar. 2019].

Wu, J., Xu, H., Zheng, Y. and Tian, Z. (2018). A novel method of vehicle-pedestrian near-crash identification with T roadside LiDAR data. Reno: Elsevier.
Yang, S. and Wang, C. (2011). On Solving Mirror Reflection in LIDAR Sensing. *IEEE/ASME Transactions on Mechatronics*, 16(2), pp.255-265.

Yoshioka, M., Suganuma, N., Yoneda, K. and Aldibaja, M. (2017). Real-Time Object Classification for Autonomous Vehicle using LIDAR. Okinawa: Kanazawa University.

Online articles and websites

(Where possible, multiple sources have been cited for online references excluding manufacturer websites. Due to the current nature of some information, journals have not yet been published regarding certain subjects)

Agarwal, T. (n.d.). *LIDAR System (Light Detection And Ranging) Working and Applications*. [online] ElProCus - Electronic Projects for Engineering Students. Available at: https://www.elprocus.com/lidar-light-detection-and-ranging-working-application/ [Accessed 14 Nov. 2018].

Goggin, B. (2018). *After Several Deaths, Tesla Is Still Sending Mixed Messages About Autopilot*. [online] Digg.com. Available at: http://digg.com/2018/tesla-crash-autopilot-investigation [Accessed 17 Nov. 2018].

Hokuyo-aut.jp. (n.d.). *URG-04LX Product Details*. [online] Available at: https://www.hokuyo-aut.jp/search/single.php?serial=165 [Accessed 29 Nov. 2018].

I2C Info – I2C Bus, Interface and Protocol. (n.d.). *I2C Bus, Interface and Protocol*. [online] Available at: https://i2c.info [Accessed 14 Apr. 2019].

Kelechava, B. (2018). *SAE Levels of Driving Automation - ANSI Blog*. [online] The ANSI Blog. Available at: https://blog.ansi.org/2018/09/sae-levels-driving-automation-j-3016-2018/#gref [Accessed 9 Nov. 2018].

Lambert, F. (2016). *A look at Tesla's new Autopilot hardware suite: 8 cameras, 1 radar, ultrasonics & new supercomputer*. [online] Electrek. Available at: https://electrek.co/2016/10/20/tesla-new-autopilot-hardware-suite-camera-nvidia-tesla-vision/ [Accessed 16 Nov. 2018].

Lidar-uk.com. (n.d.). *A brief history of LiDAR*. [online] Available at: http://www.lidar-uk.com/a-brief-history-of-lidar/ [Accessed 9 Nov. 2018].

Neon Science (2014). *How Does LiDAR Remote Sensing Work? Light Detection and Ranging*. [video] Available at: https://www.youtube.com/watch?v=EYbhNSUnIdU [Accessed 8 Nov. 2018].

Orbbec3d.com. (n.d.). *Astra S Technical Specs*. [online] Available at: https://orbbec3d.com/product-astra/ [Accessed 29 Nov. 2018].

pirzada, U. (2015). *An In-Depth Look At The Technology Behind the Engineering Marvel – Wccftech*. [online] Wccftech.
Available at: https://wccftech.com/tesla-autopilot-story-in-depth-technology/5/ [Accessed 9 Mar. 2019].

Rac.co.uk. (2017). *Stopping distances made simple | RAC Drive*. [online] Available at: https://www.rac.co.uk/drive/advice/learning-to-drive/stopping-distances/ [Accessed 30 Apr. 2019].

Raspberry Pi. (n.d.). *Raspberry Pi 3 Model B*. [online] Available at: https://www.Raspberrypi.org/products/Raspberry-pi-3-model-b/ [Accessed 29 Nov. 2018].

RobotShop. (n.d.). *LiDAR, Laser Scanners and Rangefinders*. [online] Available at: https://www.robotshop.com/uk/lidar.html [Accessed 29 Nov. 2018].

Sick.com. (n.d.). *MRS1000 | SICK*. [online] Available at: https://www.sick.com/gb/en/detection-and-ranging-solutions/3d-lidar-sensors/mrs1000/c/g387152 [Accessed 1 May 2019].

Silver, D. (2018). *Connected Teslas*. [online] Medium. Available at: https://medium.com/self-driving-cars/connected-teslas-e0201c70ec94 [Accessed 9 Mar. 2019].

Stewart, J. (2018). *Why LiDAR might have prevented the Tesla Autopilot firetruck crash*. [online] Wired.co.uk. Available at: https://www.wired.co.uk/article/tesla-fire-truck-crash-autopilot-accident [Accessed 17 Nov. 2018].

Tesla (2018). *Full self driving hardware on all cars*. [online] Available at: https://www.tesla.com/en_GB/autopilot [Accessed 17 Nov. 2018].

Waymo. (2019). *Lidar - Laser Bear Honeycomb – Waymo*. [online] Available at: https://waymo.com/lidar/ [Accessed 10 Mar. 2019].

Image sources

Dwivedi, P. (2017). *LiDAR point cloud*. [image] Available at: https://towardsdatascience.com/tracking-pedestrians-for-self-driving-cars-ccf588acd170 [Accessed 29 Nov. 2018].

OREFIND (2013). *Original LiDAR point data*. [image] Available at: http://www.orefind.com/blog/orefind_blog/2013/04/02/rest-in-peace-topographic-contours---part-2 [Accessed 21 Nov. 2018].

Renishaw (n.d.). *Diagram of the LiDAR optics and encoders*. [image] Available at: https://www.renishaw.com/en/optical-encoders-and-lidar-scanning--39244 [Accessed 29 Nov. 2018].

Velodyne (n.d.). *Raw data from HDL-64E*. [image] Available at: https://velodynelidar.com/hdl-64e.html [Accessed 7 Mar. 2019].

Vox (2016). *The 5 levels of driving automation*. [image] Available at: https://www.vox.com/2016/9/19/12966680/department-of-transportation-automated-vehicles [Accessed 19 Nov. 2018].

Books & manuals

McEachen, L. and Day, C. (2018). *Autonomous Buggy Manual*.

Quigley, M., Gerkey, B. and Smart, W. (2015). *Programming robots with ROS*.

Background reading

(A list of material not directly referenced but used to gain a broader understanding of autonomous vehicles)

Carbuyer. (2018). *Best self-parking cars*. [online] Available at: https://www.carbuyer.co.uk/reviews/recommended/best-self-parking-cars [Accessed 17 Nov. 2018].

CleanTechnica. Available at: https://cleantechnica.com/2018/06/01/waymo-orders-62000-autonomous-chrysler-pacifica-hybrid-vans/ [Accessed 11 Nov. 2018].

DARPA. (n.d.). *DARPA Urban Challenge*. [online] Available at: http://archive.darpa.mil/grandchallenge/ [Accessed 11 Nov. 2018].

Field, K. (2018). *Tesla Director Of AI Discusses Programming A Neural Net For Autopilot (Video) | CleanTechnica*. [online] CleanTechnica. Available at: https://cleantechnica.com/2018/06/11/tesla-director-of-ai-discusses-programming-a-neural-net-for-autopilot-video/ [Accessed 17 Nov. 2018].

Hall, D. (2017). *128 Lasers on the Car Go Round and Round: David Hall on Velodyne's New Sensor*. [online] Velodyne

Hanley, S. (2018). *Waymo Orders 62,000 Autonomous Chrysler Pacifica Hybrid Vans | CleanTechnica*. [online]

Heater, B. (2016). *Technology is killing jobs, and only technology can save them*. [online] TechCrunch. Available at: https://techcrunch.com/2017/03/26/technology-is-killing-jobs-and-only-technology-can-save-them/ [Accessed 21 Nov. 2018].

Higgins, S. (2017). *Hovermap: Powerful SLAM for Drone Autonomy and Lidar Mapping - SPAR 3D*. [online] SPAR 3D. Available at: https://www.spar3d.com/news/uav-uas/hovermap-powerful-slam-drone-autonomy-lidar-mapping/ [Accessed 20 Nov. 2018].

Korosec, K. (2017). *5 Things to Know About the Future of Google's Self-Driving Car Company: Waymo*. [online] Fortune. Available at: http://fortune.com/2017/01/08/waymo-detroit-future/ [Accessed 8 Nov. 2018].

O'Kane, S. (2018). *Consumer Reports reverses course and now recommends the Tesla Model 3*. [online] The Verge. Available at: https://www.theverge.com/2018/5/30/17409782/consumer-reports-tesla-model-3 [Accessed 26 Apr. 2019].

Venugopal, V. and Kannan, S. (2013). Accelerating real-time LiDAR data processing using GPUs. *2013 IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS)*

Appendices

Appendix A - LiDAR range values for all tests conducted

	Range Value No										Result
Test type	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	
Angular Reflective 1	2.5	2.5	2.5	2.5	2.4	2.4	2.4	2.3	2.3	2.3	Crashed
Angular Reflective 2	nan	nan	nan	nan	nan	nan	nan	nan	nan	0	Error
Angular Reflective 3	nan	3.3	3.2	3.2	nan	nan	3.0	3.0	2.9	2.9	Crashed
Angular Reflective 4	2.8	2.8	2.7	2.7	2.6	2.6	2.6	2.5	2.5	2.5	Crashed
Angular Reflective 5	nan	nan	nan	nan	nan	nan	3.2	3.1	3.0	3.0	Crashed
Dynamic 1	nan	nan	nan	nan	nan	nan	1.9	1.9	1.7	0.5	Stopped
Dynamic 2	2.9	2.9	2.9	2.8	2.8	2.7	2.7	2.6	2.5	0.3	Stopped
Dynamic 3	nan	nan	nan	nan	nan	nan	nan	3.2	nan	0	Error
Dynamic 4	nan	nan	nan	nan	nan	nan	nan	nan	nan	0.3	Stopped
Dynamic 5	nan	nan	nan	nan	nan	nan	nan	2.1	1.9	0.5	Stopped
High Voltage 1	nan	nan	1.8	1.6	1.5	1.3	1.2	1.0	0.9	0.6	Stopped
High Voltage 2	nan	2.0	1.8	1.7	1.5	1.3	1.2	1.0	0.9	0.6	Stopped
High Voltage 3	2.2	2.0	1.8	1.6	1.5	1.3	1.2	1.0	0.9	0.6	Stopped
High Voltage 4	2.1	2.0	1.8	1.7	1.5	1.4	1.2	1.0	0.9	0.6	Stopped

	Range Value No										Result
Test type	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	
High Voltage 5	nan	nan	1.8	1.7	1.5	1.3	1.2	1.1	0.9	0.6	Stopped
Low light 1	nan	nan	nan	nan	nan	nan	nan	nan	nan	0.4	Stopped
Low light 2	nan	nan	nan	nan	nan	nan	nan	nan	nan	0	Crashed
Low light 3	nan	nan	nan	nan	nan	nan	nan	nan	nan	0.5	Stopped
Low light 4	nan	nan	nan	nan	nan	nan	nan	nan	nan	0.6	Stopped
Low light 5	nan	nan	nan	nan	nan	nan	nan	nan	nan	0.4	Stopped
Reflective 1	nan	nan	nan	nan	nan	nan	1.1	1.1	1.0	1.0	Stopped
Reflective 2	nan	nan	nan	nan	nan	nan	0	2.1	2.1	2.1	Crashed
Reflective 3	0	1.8	1.7	1.6	1.5	1.4	1.3	1.2	1.1	1.0	Error + Stopped
Reflective 4	1.6	1.5	1.4	1.4	1.3	1.3	1.2	1.2	1.1	1	Stopped
Reflective 5	1.5	1.4	1.4	1.3	1.2	1.2	1.1	1.0	1.0	3.5	Error + Stopped
Stationary 1	1.5	1.4	1.4	1.3	1.2	1.2	1.1	1.0	1.0	0.9	Stopped
Stationary 2	1.4	1.4	1.3	1.2	1.2	1.1	1.1	1.0	1.0	0.8	Stopped
Stationary 3	nan	1.4	1.3	1.3	1.2	1.2	1.1	1.0	1.0	0.8	Stopped
Stationary 4	nan	1.3	1.3	1.2	1.2	1.1	1.1	1.0	1.0	0.8	Stopped
Stationary 5	1.4	1.3	1.3	1.2	1.2	1.1	1.1	1.0	0.9	0.8	Stopped

Appendix B - Autonomous Python script for buggy

```
Set as interpreter
#!/usr/bin/env python
#import necessary modules
import time
import Adafruit_MCP4725
import rospy
#import LiDAR range messages
from sensor_msgs.msg import LaserScan

#Defining function for going forward using DAC Address and time
def Y(icAdd, t):
    dac = Adafruit_MCP4725.MCP4725(icAdd)
    #maximum DAC voltage
    piV = 5.17
    #set voltage for DAC
    arr = [3.97]
    print "Going forward"
    #conversion of pi voltage to 12 bit DAC voltage
    for x in range(0, len(arr)):
    result = arr[x] / piV
    result = result * 4096
    print int(result)
    dac.set_voltage(int(result))
    time.sleep(0.5)
#Defining the function for setting the X DAC (left&right) to 2.5V
#therefore the buggy only goes forwards/backwards

def X(icAdd, t):
    dac = Adafruit_MCP4725.MCP4725(icAdd)
    #maximum DAC voltage
    piV = 5.17
    #set voltage for DAC
    arr = [3.97]
    print "Going forward"
    #conversion of pi voltage to 12 bit DAC voltage
    for x in range(0, len(arr)):
    result = arr[x] / piV
    result = result * 4096
    print int(result)
    dac.set_voltage(int(result))
    time.sleep(0.5)

#Defining stop function
def stop(icAdd, maxV):

    dac = Adafruit_MCP4725.MCP4725(icAdd)
    #2.5V is the voltage at which the buggy stops
    arr = [2.5]

    print "Stopping"

    for x in range(0, len(arr)):
        voltage = (2.5 / maxV) * 4096
        dac.set_voltage(int(voltage))
        time.sleep(0.5)
#Defining the LiDAR range variable
def scan_callback(msg):
    global g_range_ahead
    #taking the middle value from the Laserscan messages
    g_range_ahead=msg.ranges[len(msg.ranges)/2]

#setting the default range value and initiating the nodes for ROS
g_range_ahead=1
#subscribing to the laser scan messages (See publishing and subscribing on roswiki)
scan_sub=rospy.Subscriber('scan',LaserScan,scan_callback)
rospy.init_node('wander')
#setting the refresh rate to 10Hz
rate=rospy.Rate(10)

#The while loop for autonmous driving
while not rospy.is_shutdown():

        if(g_range_ahead<1.0):
            #calling the stop function at the X&Y DAC address with their Max voltages
            stop(0x62, 5.17)
            stop(0x63, 4.24)
            #stopping the buggy for 3 seconds
            time.sleep(3.0)
            print ("range ahead %0.1f"%g_range_ahead)
        else:
            #calling the forward function at X&Y DAC's for 5 seconds
            Y(0x62, 5)
            X(0x63, 5)
            time.sleep(0.5)
            print ("range ahead %0.1f"%g_range_ahead)
            #sleep constantly adjusted by ROS to keep rate at 10Hz based on speed of computer
            rate.sleep()
```

Appendix C - Existing autonomous script for buggy

```
# Import required packages

import time
from FourSensors import fourSensors
from Mobile_Test_1 import forward
from STOP import stop
import RPi.GPIO as gpio
import cv2


# Declare the file and max distance value globally
global f, maxVal


# Declare sensor distance (e.g. 50cm, 100cm, 150cm etc.)

maxVal = int(300)

def test_run():

    # Call the sensor values

    fourSensors()

    # Get a first count

    e1 = cv2.getTickCount()

    # If the first index of the sensor array is greater than the maxVal

    if int(fourSensors()[0])>maxVal:
        print fourSensors()

        # Go forward for half a second

        forward(0x62, 0.5)

        # Get a second count

        e2 = cv2.getTickCount()

        # Find difference between counts to determine program speed

        myTime = (e2 - e1) / cv2.getTickFrequency()
        print "The distance for the first sensor is " + str(fourSensors()[0]) + "cm"
        print "The speed of the program is " + str(myTime)

    elif int(fourSensors()[0]) < maxVal:
        e2 = cv2.getTickCount()
        myTime = (e2 - e1) / cv2.getTickFrequency()

        # Stop motors

        stop(0x62, 5.17)
        stop(0x63, 4.24)
        print "The first sensor is too close...exiting"
        e2 = cv2.getTickCount()
        myTime = (e2 - e1) / cv2.getTickFrequency()
        print "The speed of the program is " + str(myTime)

        # Exit/Kill program

        exit()

# Call function

while True:
    test_run()
```

Appendix D - Risk Assessment

General Risk Assessment Form

Date:	11/11/18	Assessed by:	Asiya Khan	Activity/Location	Brunel Lab W11	ENGINEERING WITH PLYMOUTH UNIVERSITY

Hazard and likely consequences	No. at Risk	Uncontrolled Risk			How is the hazard controlled; (E.g. CoPs – Guidance Notes – mechanical measures – supervision – training etc.)	Residual Risk			Responsible Person
		L	S	L x S		L	S	L x S	
Soldering of printed circuit boards	1	2	3	6	Care taken when soldering, allowing parts to cool and solidify before touching	1	3	3	M Calcroft
Keeping workspace tidy	3	3	2	6	Putting away of buggy and all chairs etc. to original conditions	1	2	2	M Calcroft
Harm from battery used on buggy	1	2	2	4	Care taken when dealing with battery, making sure not to expose wires etc.	2	2	4	M Calcroft

Appendix E - Gantt chart

Appendix F - Interim Report

Implementation of a LIDAR system via Raspberry Pi for object detection and avoidance

By
Max Calcroft

School of Engineering
Faculty of Science and Engineering
University of Plymouth

Interim Progress Report

Honours project submitted in partial fulfilment of the
requirements for the degree of
BEng (Hons) in Mechanical Engineering

December 2018

Supervisor Asiya Khan

Contents

1.0 - Nomenclature 1
2.0 - Introduction 2
3.0 - Project planning
3.1 - Aim of project 3
3.2 - Objectives 4
3.3 - Project deliverables 4
4.0 - Literature review
4.1 - Understanding the basics of LiDAR 3
4.2 - Classification of objects 5
4.3 - S.A.E levels of autonomy 5
4.4 - Applications in industry 6
4.4.1 - Autonomous vehicle applications and comparison 6
4.4.2 - Aerial and other applications of LiDAR 8
4.4.3 - Social aspects of autonomy 8
4.4.4 - Plymouth University autonomous buggy 9
5.0 - Methodology 10
6.0 - Progress to date
6.1 - Work completed 11
6.2 - Obstacles 11
6.3 - Plan of future work 11
7.0 - Appendices
7.1 - Appendix A: Declarations 12
7.2 - Appendix B: Gantt charts 13
7.3 - Appendix C: Risk Assessment Form 13
7.4 - Appendix D: References 13

1.0 - Nomenclature

LiDAR - Light Detection And Ranging
GPS - Global Positioning System
IMU - Inertial Moment Unit
ECU - Electronic Control Unit
Raspberry Pi - A low cost, credit-card sized computer that plugs into a computer monitor or TV (Raspberry Pi, n.d.)
PCB - Printed Circuit Board
GPiO - General Purpose Input Output

DAC - Digital to Analogue Converter
SSID - Service Set IDentifier (name of a wireless network)
'psk' - Pre Shared Key (password)
FoV - Field of View

2.0 - Introduction

This reports primary aim is to look at the implementation of a LiDAR on a Raspberry Pi model 3B and the identification/ classification of objects for avoidance. The secondary aim, if time permits is to implement the LiDAR onto an autonomous buggy alongside several ultrasonic sensors and subject it to a pre-determined route, avoiding objects. The buggy itself is a previous final year project built and run by two students investigating pedestrian recognition using the Raspberry Pi and the relaying of data to a wireless infrastructure hub.

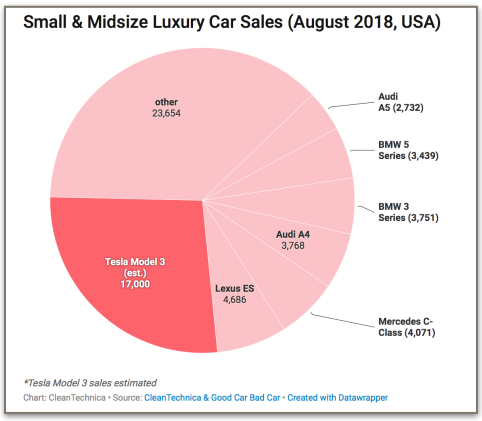


Figure 1 (Shahan, 2018)

With the rapid proliferation of technology within the past decade, autonomy is being introduced into areas such as transport, infrastructure, agriculture and manufacturing reducing the need for human interaction. This is becoming increasingly common within the vehicle market, with automobiles from Tesla and Waymo, previously Google's attempt at a self driving vehicle, taking up a significant portion of the market share. As seen in August 2018 (Figure 1), the new Tesla Model 3 outsold the competition by a large margin for small and midsize luxury cars in the US. Public transport can also be seen heading in the same direction with Waymo aiming to have a fleet of roughly 82,600 self driving vehicles operational in the near future (Hanley, 2018).

This fleet of autonomous vehicles rely on a LiDAR for sensing the world around them. LiDAR stands for Light Detection And Ranging and has been used primarily since the invention of GPS in the 1980's for aircraft and satellite use in mapping terrain. LiDAR can now be seen in applications within areas such as building restoration, meteorology, gaming and geology (Lidar-uk.com, n.d.).

Within the autonomous vehicle market there is much contention between the use of LiDAR versus a combination of cameras and sensors in terms of accuracy and object recognition. It is unknown which technology will be adopted in the future as both of which are still in the early stages of development, with implementation and reductions in price being one of the main contending factors.
The integration of a small scale computer like the Raspberry Pi with LiDAR technology should demonstrate the effectiveness of LiDAR and accessibility of this technology, providing a good small scale working example that aids with the understanding and public perception of autonomous vehicles and their development in the modern era. The social and ethical implications of autonomy and designing in morality is a problem

that has never been faced before in engineering and this project acts as an important talking point that will hopefully encourage the conversation.

3.0 - Project planning

3.1 - Aim of project

The primary aim of this project is to implement a LiDAR sensor onto a Raspberry Pi for object detection. The secondary aim (if time permits) is to implement the LiDAR via Raspberry Pi onto the existing autonomous buggy set up by two BEng (Hons) Mechanical Engineering students, to expand upon it's object detection and avoidance capabilities when subjected to a pre-determined route using the programming language Python.

3.2 - Objectives

- Learn the programming language Python for the understanding and writing of existing and new code for operation of the buggy
- Conducting a literature review around LiDAR basic principles and applications to understand the scope of LiDAR today and how it works
- Some working form of communication between LiDAR and Raspberry Pi reached to allow identification of objects
- Achieving communication between the Raspberry Pi and Autonomous Buggy by running pre-existing code
- If possible, implementation of a LiDAR onto autonomous buggy allowing identification of objects and avoidance when subjected to a predetermined route
- Testing of autonomous buggy with LiDAR installed
- Evaluation of results and comparison of existing ultrasonic sensors and cameras against LiDAR system

3.3 - Project deliverables

By the end of this project, the primary deliverable should be a working Raspberry Pi equipped with a LiDAR for range and object identification. The secondary deliverable, if time permits, is an autonomous buggy able to perform avoidance manoeuvres and come to a complete stop upon facing a stationary or moving object within a controlled environment when subjected to a series of movement based instructions.

A report detailing the basics of LiDAR, the vehicle, how the implementation of the LiDAR was carried out and the issues associated with the process will also be part of the deliverables of the project.

4.0 - Literature review

4.1 - Understanding the basics of LiDAR

LiDAR is essentially a form of technology used for sensing the world around it. It works by sending out green or near infrared light, scanning the surrounding area (Figure 3). The time taken between the laser being sent out and bouncing off the object and being received again allows the distance of the object to be determined. This is done by the simple equation seen in Figure 2.

$$[(traveltime) * (speedoflight)]/2 = Distance$$

Figure 2 (LiDAR-UK, n.d)

When the echo beam from the laser returns back it is reflected by a rotating mirror that matches the angle of each beam as it returns. This echo beam is then fed through an infrared transmitter diode to a photo diode receiver allowing distance to be calculated (Figure 4).

Figure 3 - *LiDAR point cloud* (Dwivedi, 2017)

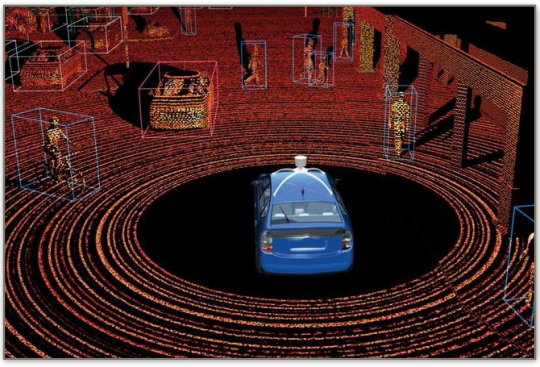
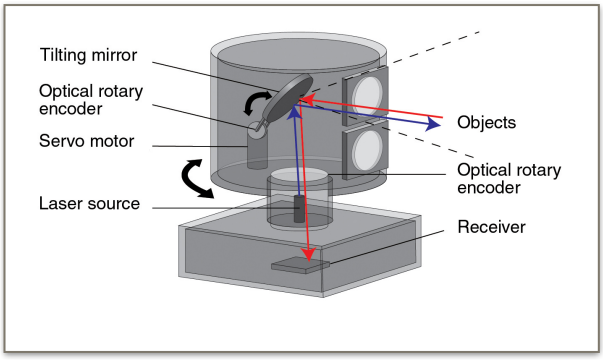


Figure 4 - *Diagram of the LiDAR optics and encoders* (Renishaw, n.d.)



As LiDAR's are usually implemented on moving vehicles, a system known as an IMU (Inertial Moment Unit) allows the movement of the vehicle to be taken into account for distance calculations or allows orientation to be accounted for aerial applications. A GPS allows the exact position to be known. The combination of data gathered from the IMU and GPS allows a 3D static point to be determined. The IMU, GPS and LiDAR are connected to an onboard computer and produce 3D point cloud data from the echo beam pulses that can be either stored in mapping/ topography applications or act as an input to determine a vehicles movement via an ECU (Agarwal, n.d.).

For ground based purposes two ranges of wavelength can be used for the range detecting lasers. 600-1000nm at lower power levels are considered 'eye-safe', an important factor due to the nature of the typical application. Lasers at 1550nm can also be used and deemed 'eye-safe' at much higher power levels however they tend towards higher range, lower accuracy purposes (LiDAR-UK, n.d).

Two LiDAR's have been provided for use in this project with one being chosen for final implementation on the Raspberry Pi. The models provided are a Hokuyo URG-04LX and a ORBBEC Astra S and both are suited towards student research projects due to the power over USB. As seen in Figure 5, the Hokuyo model has clear benefits and may be chosen as the LiDAR used for the project however implementation carried out by the previous students proved to be difficult. This may result in the Astra being used due to ease of setup.
There are many different types of LiDAR available that can be used on small scale projects. The LiDAR lite 3 is a common, cheap LiDAR that can be used on small robotic applications. The RPLIDAR -A2M8 is a more expensive option but has a 360° FoV allowing advanced peripheral vision. The Benewake TFMINI Micro LiDAR is a cheap option at £35 however with an decrease in price, units often have limited range, accuracy and FoV (RobotShop, n.d.).

Type of LiDAR	Range	Accuracy	Angular Resolution	Scanning time	Price
Hokuyo URG-04LX	60-4095mm (240° FoV)	60-1000mm ±10mm 1000-4095mm ±1%	0.36°	100ms/scan	$1080
ORBBEC Astra S	400mm-2000mm	60° horizontal 49.5° vertical	Data not available	Data not available	$149.99

Figure 5 - *Table comparing LiDAR's provided for project* (Data taken from hokuyo-aut.jp and orbbec3d.com)

4.2 - Classification of objects

The classification of objects by any autonomous system allows the world to be interpreted around it into known factors, reducing uncertainty. The accuracy of this is an important factor in bringing autonomous vehicles into everyday use as the consequences when dealing with humans in an uncontrolled environment can be very severe.

Many different algorithms are used for classification of objects such as Real AdaBoost or 3D-DSF however the accuracy of identifying dynamics objects and their paths along with the processing time required for all of this calculation has room for improvement with processing times of 0.002s for each object. Quick processing times within the real world are vital due to the nature of dynamic pedestrians, cyclists, cars etc. that need to be avoided (Yoshioka et al., 2017) (Wu et al., 2018).

Data obtained from a LiDAR is presented as 3D point cloud data. Each point has an x, y and z coordinate and can be clustered together based on proximity to formulate objects such as pedestrians, cyclists, cars or background. Lower density clusters are typically identified as background however accuracy usually drops off around 30m. Methods such as DBSCAN (Density-based spatial clustering of applications with noise) can be used to appropriately filter out unwanted noise.

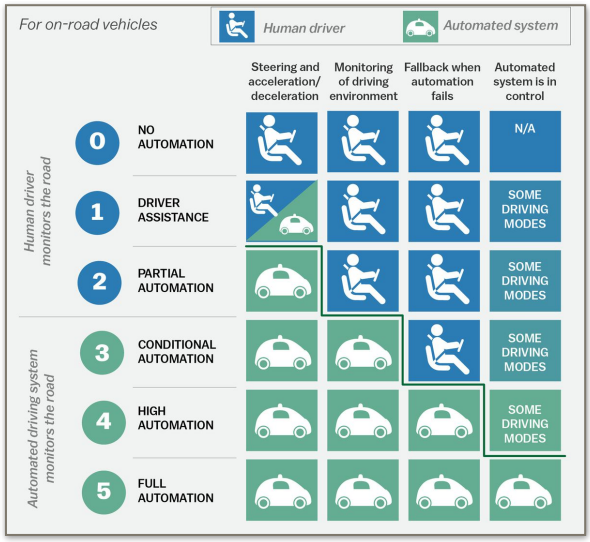
Wu et al's study of near pedestrian crashes used back propagation via a neural network to classify objects similar to Tesla's approach with their neural net. Other strategies include giving each cluster of point cloud data a score for which the probability of what class of object (e.g pedestrian, car, cyclist) is determined via logistic regression analysis. This is a predictive analysis tool used to describe data by explaining the relationship between one variable and one or more independent variables (Yoshioka et al., 2017).

4.3 - S.A.E levels of autonomy

The S.A.E (Society of Automotive Engineers) have 6 levels of autonomy for measuring a cars autonomous capabilities while on the road. The levels range from Level 0, no driving automation, to level 5, full driving automation whereby the vehicle takes control of the entire DDT (Dynamic Driving Task) without the need for any form of human intervention (Kelechava, 2018).

As you can see in Figure 7, as levels increase there is less need for human involvement on most areas of the DDT.

Figure 7 - *The 5 levels of driving automation* (Vox, 2016)



Companies like Waymo are aiming for level 4 autonomy but with the rapid advancement and competition driving this that we've seen in the past decade, the mainstream integration of level 5 autonomy may be sooner than expected.

4.4 - Applications in industry

4.4.1 - *Autonomous vehicle applications and comparison*

With advancements in technology, autonomy is becoming more prevalent within many industries. Autonomy has many benefits which have led to it becoming more widespread across many disciplines. Reduction of costs, less risk due to lack of human interaction, higher efficiency, increased quality are all factors the are driving this change.

The availability of the technology that powers these vehicles allows not just industrialists to develop autonomous vehicles. The DARPA grand challenge, run every year, pits teams often from universities together to develop a completely autonomous vehicle that is tasked with completing complex manoeuvres i.e merging, passing, negotiating intersections (DARPA, n.d.). Technology that is now in full commercial use such as Velodyne's spinning LIDAR sensor was developed leading up to the 2005 DARPA grand challenge, which is why events such as these are so beneficial to the industry, pushing for continuous improvement in the sector (Hall, 2017).

One of the main integrations of autonomy into everyday life for most people has been the introduction of autonomous vehicles. Many cars such as the Mercedes C-Class Saloon, Peugeot 3008 SUV and the Citroën C4 SpaceTourer MPV all have some form of autonomy in the form of self parking modes with Mercedes boasting a system that automatically scans for available parking spaces (Carbuyer, 2018).

Two companies leading the push towards an autonomous vehicle future are Tesla and Waymo. Tesla already have several models on the road with the Model S and Model X and with the Model 3, Roadster and Tesla truck due in the next few years. All cars currently promise 'full self-driving capability at a safety level substantially greater than that of a human driver.' (Tesla, 2018). These cars are not without their limitations and several accidents over the past few years such as collisions with concrete barriers, tractors that could not be distinguished by 'Tesla Vision' and a stationary firetruck while in autopilot have made the public and authorities such as the NTSB (National Transport Safety Board) question the safety of autonomous driving (Goggin, 2018).

Tesla cars use a system known as 'Tesla Vision' to sense the world around them. The latest version of this system has 8 cameras around the car coupled with 12 ultrasonic sensors and radar for a 360 degree coverage with range up to 250m (Lambert,

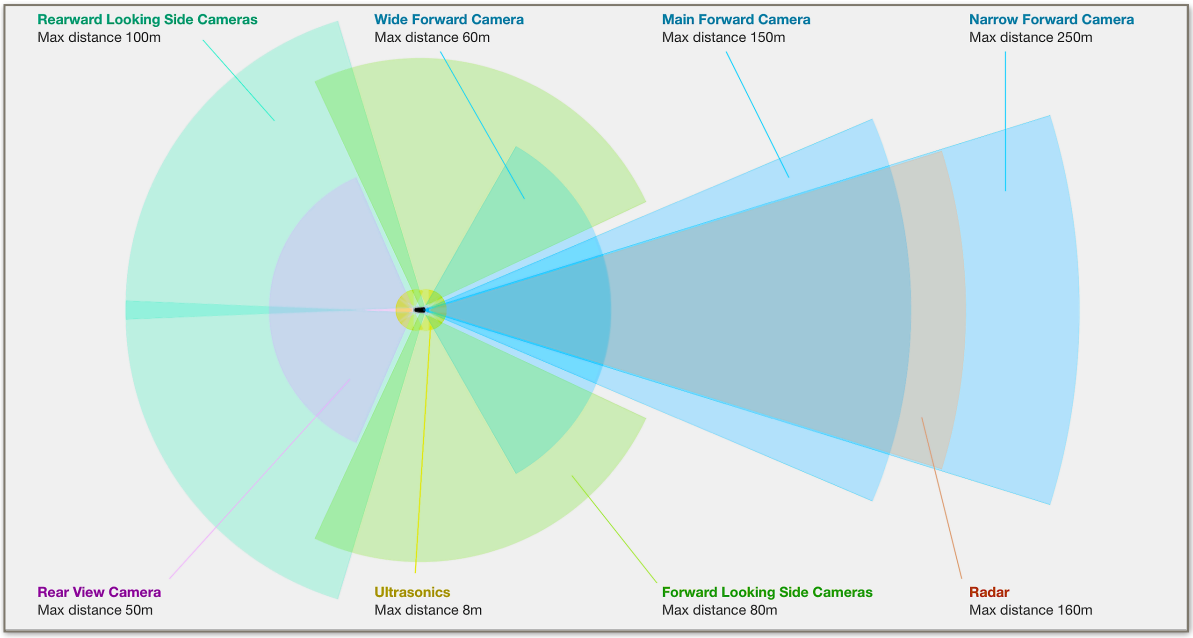


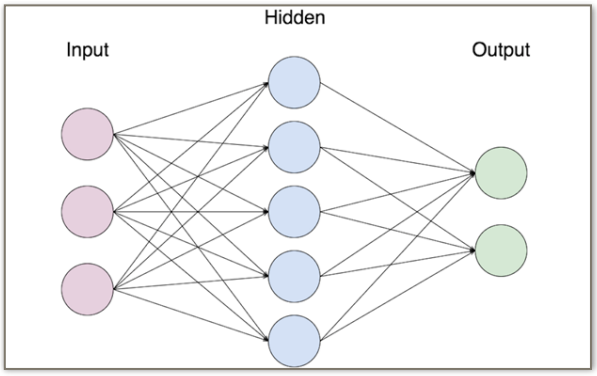
Figure 8 - *Tesla range detection capabilities* (Tesla, 2018)

2016) as seen in Figure 8. The cheapest model available is currently the Model S starting at £65,000 however the Model 3, due to be released in 2019 starts at $35,000 USD, a much more affordable price (Tesla, 2018)

'Tesla Vision' as previously mentioned is Tesla's own neural network. A neural network is vaguely based upon the human brain and works upon the process of an input, a hidden layer and an output layer with weights connecting these layers that are to be determined (Figure 9). For a Tesla this means the network is able to learn from previous driving situations and modify it's solutions based on these past experiences. Andrej Karpathy, Tesla's head of AI went into detail on how for programming, the network is shown several labelled images that it then learns from in order to better deal with abnormalities when driving (Field, 2018).

Waymo are currently working on producing a fully autonomous vehicle to be released this December acting as more of a shuttle to get people from A to B. Waymo are more in direct competition with the likes of Uber and Lyft in order to get an autonomous taxi style service to market however the comparison of technologiesbetween Tesla and Waymo are of great significance and interest. Waymo's fleet of autonomous vehicles use LiDAR technology in order to build up an image of the surrounding area. One of the main advantages of using LiDAR is the fact that it responds well to high brightness levels and sudden changes of light (Wu et al., 2018). Arguments have been made that use of a LiDAR could have prevented accidents such as the firetruck crash and tractor collision

Figure 9 - *Basic Neural Network Diagram* (Simple NN Working, 2018)



mentioned above due to this feature (Stewart, 2018).

Waymo's vehicles have a new vision system designed to take high resolution pictures of the world and analyse them while moving. Multiple sensors are installed around the vehicle, similar to Tesla, with a forward facing high resolution multi-sensor providing 360 degree coverage. The LiDAR, sensors and radar provide an unprecedented level of computer vision which has yet to be matched by other car manufacturers. Part of the reason for Waymo integrating multiple vision systems is because they are aiming for level 4 autonomy as stated by the SAE. Their aim is for vehicles to be fully autonomous in certain areas such as cities which already have detailed mapping and grid systems.

This commitment to full autonomy has led to them developing everything in house, having complete control over the entire manufacturing process and this has resulted in the development of two new types of LiDAR. Medium range LiDAR has been used by the company since 2012 with their units coming from Velodyne, one for the leading suppliers of LiDAR units, however now short and long range LiDAR is also being used on Waymo's vehicles resulting in 3 separate LiDAR units for all ranges.

Waymo's vehicles are yet to hit the roads so a direct comparison over errors is not yet feasible, however Waymo have released numbers stating that errors in the driverless system have now decreased form 0.8 to 0.2 over 1000miles during testing.

LiDAR is still very much in the early stages of development within the automotive industry and as a result units can cost around $75,000 (Korosec, 2017). This is one of the reasons holding LiDAR back slightly, however Waymo have stated they have reduced the cost of LiDAR by up to 90% making it much more viable. More research and development is needed within the field of LiDAR to help it develop and understand more about the limitations of the technology. This is one of the reasons why development of this autonomous buggy is important, showing the advantages and disadvantages on a smaller scale.

4.4.2 - *Aerial and other applications of LiDAR*

LiDAR technology has been predominantly used for aerial use until recently. LiDAR units are often attached to the underside of the plane and are ideal for scanning terrain and mapping. The adversity to brightness and weather conditions allows detailed mapping (Figure 11) regardless of conditions and cloud coverage and the type of light used reflects strongly off of vegetation. The nature of the beams from the LiDAR also allows several returns of the light when passing through trees. Both of these factors allow accuracy at high altitudes (Neon

Figure 10 -*Hovermap scan of radio tower* (LiDAR-UK, n.d)

Science, 2014).

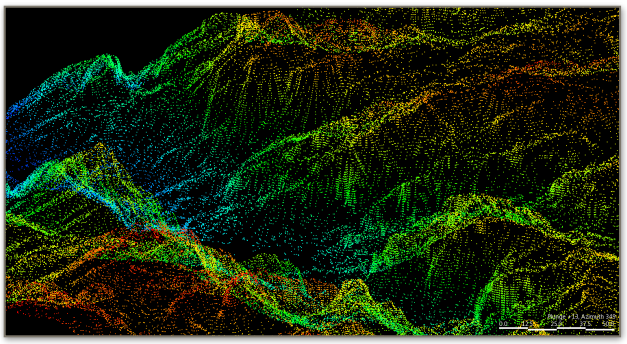


Figure 11 - *Point cloud map of terrain* (OREFIND, 2013)

Because of LiDARs mapping capabilities it can be used for applications in architecture, sewer & manhole maintenance, military and law enforcement for surveillance or via UAV.
Certain technologies have been developed that allow UAV's to fly autonomously and scan an area. Hovermap, from DATA61 and CSIRO brings advanced SLAM-based mapping (Simultaneous Localisation and Mapping) and collision avoidance to unmanned drones. Using LiDAR, a 3D point cloud map of the surrounding area can be developed, reliably reducing the need for human intervention (Figure 10). Much like the autonomous buggy and Waymo vehicles, a map can be developed in real-time, however the data gathered is more for industrial purposes and generally stored instead of for immediately being used for avoidance (Higgins, 2017).

4.4.3 - *Social aspects of autonomy*

With the increase in automated processes, the impact this will have on the job market and the question of ethics in replicating human decision making processes presents a difficult challenge.

With increased autonomy in industry mundane, labour intensive, dangerous jobs have become more automated. This not only reduces cost for a company but also reduces liability, risk related to human based injury and the knowledge that the process will be completed efficiently. At the turn of the century 41% of jobs in the US were agriculture based. That number a decade later was 1.9% showing a significant decline as agriculture processes are now mostly robotic.

Colin Parris, VP of software at GE research stated that there will be job losses, however he explains there are benefits that will be produced from this change: "It allows us to reduce cost. If I reduce cost, I have more money that I can use for innovation. The more money I have, the more new products I can create. The more products I create, the more workforce I can hire" (As quoted in Heater, 2016) . Whether this will be true in the future or not is to remain unseen however the likely probability may just be an initial re-shuffle of jobs from hands on to more management/overseeing of processes similar to what has occurred in the last century (Heater, 2016).

The ethical and moral dilemma that faces autonomous vehicles hinders their adoption into mainstream culture. The issue of programming morality into a vehicle, deciding whether it preserves it's passenger or saves a pedestrians life for example

is a challenging topic. Manufacturers will need to prove to consumers that their technology is reliable, safe and looks after their best interests.

Studies show that people generally agree that the sacrifice of one life for many is an appropriate decision process for an algorithm to possess. However the introduction of family members skewed the results as participants cared more about self-preservation and of those close to them when presented with these scenarios. Purchase intention was also found to be low proving that participants were happy to have utilitarian autonomous vehicles on roads but expressed no interest in buying one. This reveals another hurdle manufacturers will need to overcome through marketing and public perception in order to achieve a sufficient adoption rate for these new vehicles, however more studies and research need to be conducted to gather a better overall consensus (Bonnefon, Shariff and Rahwan, 2016).

4.4.4 - *Plymouth University autonomous buggy*

The autonomous buggy set up by two BEng Hons Mechanical Engineering students, Charlie Day and Liam McEachen, is the basis for this project and to be possibly expanded upon via LiDAR integration. The basis of the buggy was originally a mobility scooter that has been repurposed and the programming language Python has been used to write code for the buggy. The generic controller for the buggy had several voltages that could be emulated by the Pi to change direction and speed. Using the GPiO (General Purpose Input Output) Pins and Adafruit MCP4725 DAC converter, a connection with the Raspberry Pi can be formed.

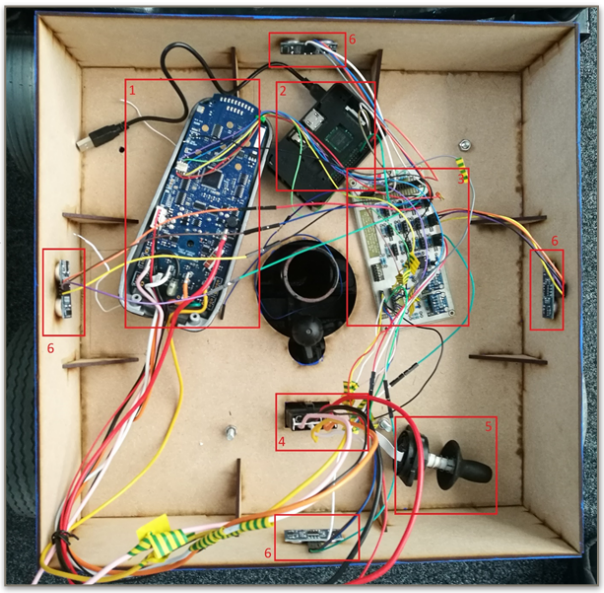


Figure 12 - *Autonomous Buggy Diagram (McEachen and Day, 2018)*

Figure 12 Key:

1. Buggy Control Panel
2. Raspberry Pi
3. Sensor PCB
4. Main Buggy Connection
5. Joystick
6. Ultrasonic Sensors

The buggy uses 4 ultrasonic sensors to detect objects using basic Python logic (Figure 13) to execute a stop or reverse command if the distance between the object and the sensor is less than a certain distance. A photo of all the components of the buggy can be seen in Figure 12.

The Open CV module is used for computer vision on the Pi and allows object detection by using a pre-trained Haar Classifier that can identify pedestrians and other objects. Using Pythons clocking functions, latency can be determined allowing the reaction time of the buggy to obstacles to be determined and accounted for in results.

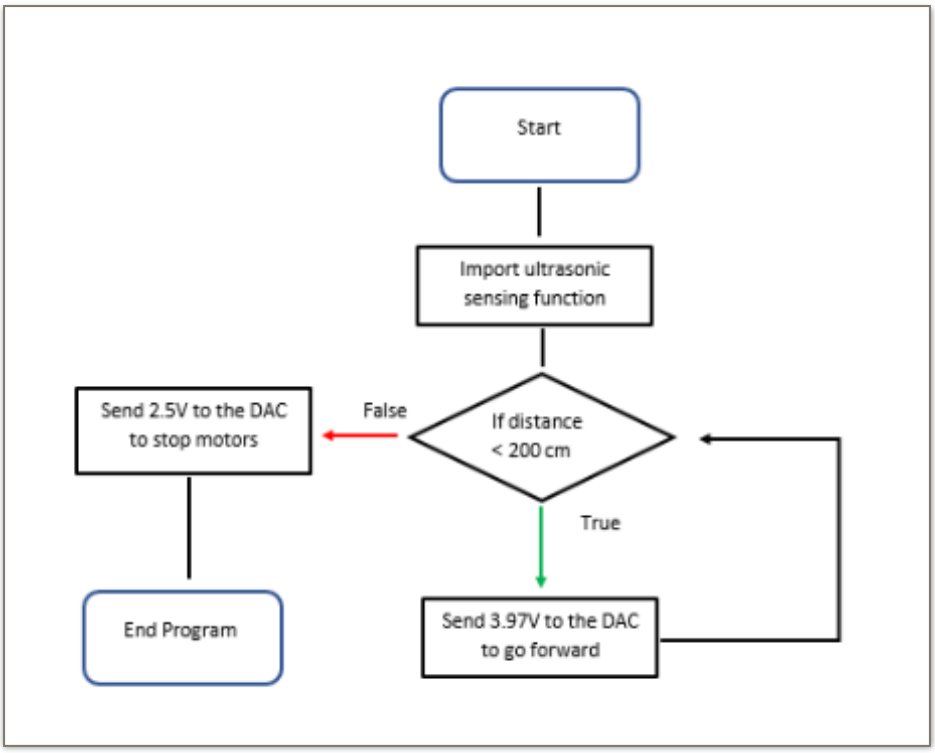


Figure 13 - *Object detection function* (Day, 2018)

5.0 - Methodology

The autonomous buggy currently uses a Raspberry Pi Model 3B (Figure 15) with either a Hokuyo URG-04LX LiDAR or ORBBEC Astra S LiDAR being used to sense the surrounding environment. A Raspberry Pi is ideal for this type of application due to it's size, ease of use and multiple ports. It can also run a number of operating systems such as Linux, Raspbian and Ubuntu. The ease of integration with the Pi and usability will determine what LiDAR will be chosen and it will act as an additional sensor to the buggy alongside the ultrasonic sensors as seen in Figure 14 if Raspberry Pi/LiDAR integration is successful and time allows.

As seen in Figure 14 the sensors currently feed through distance data to the Raspberry Pi along with a video input from the camera for identification of objects. The LiDAR will act as another sensor in the network and affect the voltages sent through the DAC's to affect the micro-controller and subsequent movement of the buggy. This strategy is similar to the way the ultrasonic sensors currently work. The LiDAR will use the equation seen in Figure 2 to calculate distance from an object and will use this to issue Python commands if the value becomes lower than a predetermined distance. Issues with this technique may require a workaround or different strategy further along in the buggy's development.

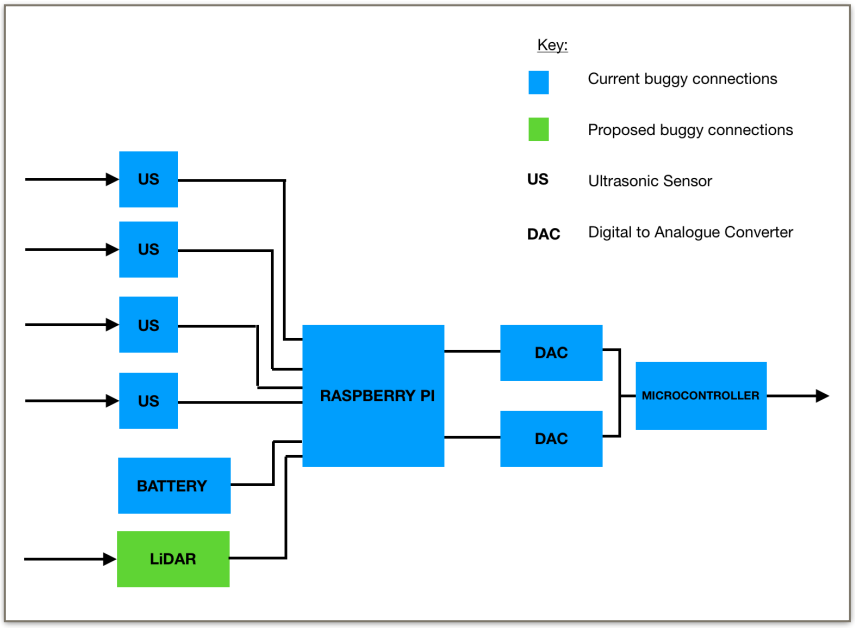


Figure 14 - *Autonomous Buggy Block Diagram*

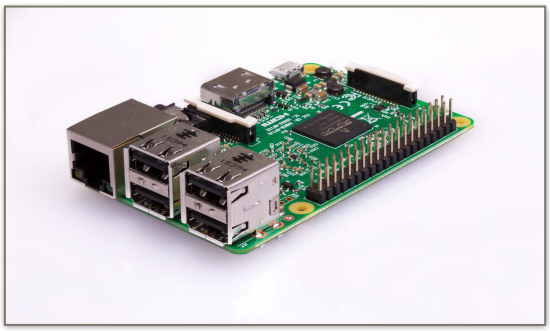


Figure 15 - *Raspberry Pi Model 3B* - (RaspberryPi, n.d)

6.0 - Progress to date

6.1 - Work completed

As previously mentioned the basis for the project is an already built buggy with Raspberry Pi integration. I have already spent extensive time studying the buggy and manual provided, understanding how the buggy works as well as developing a basic knowledge of the Python programming language. The buggy still has yet to be set up with the new Pi and testing can then proceed using existing Python movement scripts to ensure the buggy is still fully operational. The Raspberry Pi has been set up and can connect to a wireless network with a given SSID and 'psk' with the beginning of LiDAR/ Pi integration having taken place. Relevant literature surrounding the subject has been reviewed online however no communication has been made as of yet.

6.2 - Obstacles

Originally, learning of Python as a language proved difficult as I had no previous coding experience, however using lynda.com for tutorials allowed me to gain a basic understanding and start creating my own scripts.

The Raspberry Pi provided originally would not connect to a wireless network and after extensive testing, the microSD card was deemed corrupted. A second Pi was used and experienced a similar issue. After researching online and implementing code on the Pi's configuration file, the issue was still not resolved and so the microSD card was re-formatted with Raspbian (One of Raspberry Pi's Operating systems) being re-installed resulting in a fully functioning device.

Deadlines from other modules coursework, particularly HYFM322 and MFRG311 delayed the writeup of the interim report, research for literature review and setup of the Pi as seen in Appendix B. This did not prove to be particularly problematic and sufficient time was left to complete the report alongside working on the Pi and LiDAR.

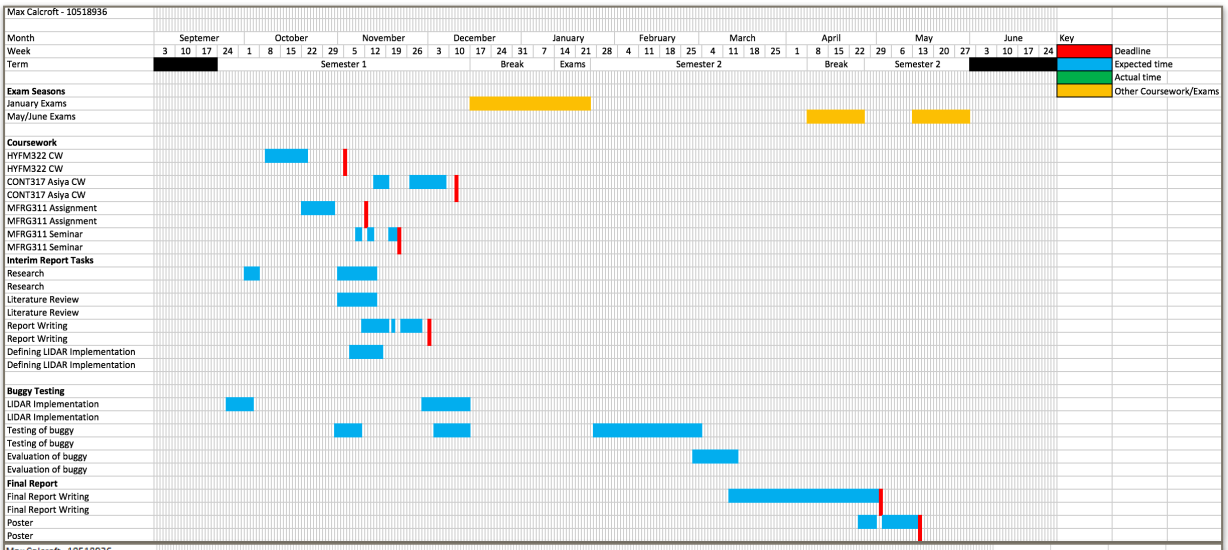
The original objectives detailed having full LiDAR and autonomous buggy integration however due to time constraints and the complexity/ unreliability of LiDAR software and the autonomous buggy this has now been changed to a more realistic goal of just LiDAR and Pi integration with implementation on the autonomous buggy being an optional extra objective if time permits. This decision was made due to the length of time and amount of errors dealt with already in the project as the software and methods can be unpredictable and not always work to plan. As a result the title of the project has changed to reflect this as it originally detailed implementation onto the buggy. The Gantt chart has been updated with new tasks and expected times for task completion as seen below.

6.3 - Plan of future work

Over the next few months there will be more of a focus on progressing with LiDAR integration. So far the objective of achieving some form of communication between LiDAR and personal computer/ Pi by end of semester 1 seems realistic with only one other deadline between now and then. The Pi integration with the autonomous buggy is now possible however a greater understanding of the connections and GPIO pins is needed before I engage in this area of the project. Due to upcoming exams and being away from the physical buggy, motivation over this period will be ensured by researching LiDAR/ autonomous buggy integration more, continuing with implementation while creating an outline for the final report and revising for January exams.

7.0 - Appendices

7.1 - Appendix A: Gantt charts



7.2 - Appendix B: Risk Assessment Form

General Risk Assessment Form

Date:	11/11/18	Assessed by:	Asiya Khan	Activity/Location	Brunel Lab W11	ENGINEERING WITH PLYMOUTH UNIVERSITY

Hazard and likely consequences	No. at Risk	Uncontrolled Risk			How is the hazard controlled; (E.g. CoPs – Guidance Notes – mechanical measures – supervision – training etc.)	Residual Risk			Responsible Person
		L	S	L x S		L	S	L x S	
Soldering of printed circuit boards	1	2	3	6	Care taken when soldering, allowing parts to cool and solidify before touching	1	3	3	M Calcroft
Keeping workspace tidy	3	3	2	6	Putting away of buggy and all chairs etc. to original conditions	1	2	2	M Calcroft
Harm from battery used on buggy	1	2	2	4	Care taken when dealing with battery, making sure not to expose wires etc.	2	2	4	M Calcroft

7.3 - Appendix C: Declarations

- I will require no use of the Marine Building laboratory facilities throughout the duration of the reported final year project.
- I will not be undertaking any work requiring the use of human or animal tissue throughout the duration of the reported final year project.
- I will not require any manufacturing throughout the duration of the reported final year project.

- I will not require no use of the rapid prototyping throughout the duration of the reported final year project.

7.4 - Appendix D: References

Lidar-uk.com. (n.d.). *A brief history of LiDAR*. [online] Available at: http://www.lidar-uk.com/a-brief-history-of-lidar/ [Accessed 9 Nov. 2018].

Shahan, Z. (2018). *7 Charts — Tesla Model 3 vs The Competition (US Sales) | CleanTechnica*. [online] CleanTechnica. Available at: https://cleantechnica.com/2018/08/06/7-charts-tesla-model-3-vs-the-competition-us-sales/ [Accessed 11 Nov. 2018].

Agarwal, T. (n.d.). *LIDAR System (Light Detection And Ranging) Working and Applications*. [online] ElProCus - Electronic Projects for Engineering Students. Available at: https://www.elprocus.com/lidar-light-detection-and-ranging-working-application/ [Accessed 14 Nov. 2018].

Yoshioka, M., Suganuma, N., Yoneda, K. and Aldibaja, M. (2017). Real-Time Object Classification for Autonomous Vehicle using LIDAR. Okinawa: Kanazawa University.

Wu, J., Xu, H., Zheng, Y. and Tian, Z. (2018). A novel method of vehicle-pedestrian near-crash identification with T roadside LiDAR data. Reno: Elsevier.

Carbuyer. (2018). *Best self-parking cars*. [online] Available at: https://www.carbuyer.co.uk/reviews/recommended/best-self-parking-cars [Accessed 17 Nov. 2018].

Goggin, B. (2018). *After Several Deaths, Tesla Is Still Sending Mixed Messages About AutoPilot*. [online] Digg.com. Available at: http://digg.com/2018/tesla-crash-autoPilot-investigation [Accessed 17 Nov. 2018].

Tesla (2018). *Full self driving hardware on all cars*. [online] Available at: https://www.tesla.com/en_GB/autoPilot [Accessed 17 Nov. 2018].

Field, K. (2018). *Tesla Director Of AI Discusses Programming A Neural Net For AutoPilot (Video) | CleanTechnica*. [online] CleanTechnica. Available at: https://cleantechnica.com/2018/06/11/tesla-director-of-ai-discusses-programming-a-neural-net-for-autoPilot-video/ [Accessed 17 Nov. 2018].

Stewart, J. (2018). *Why LiDAR might have prevented the Tesla AutoPilot firetruck crash*. [online] Wired.co.uk. Available at: https://www.wired.co.uk/article/tesla-fire-truck-crash-autoPilot-accident [Accessed 17 Nov. 2018].

Korosec, K. (2017). *5 Things to Know About the Future of Google's Self-Driving Car Company: Waymo*. [online] Fortune. Available at: http://fortune.com/2017/01/08/waymo-detroit-future/ [Accessed 8 Nov. 2018].

Kelechava, B. (2018). *SAE Levels of Driving Automation - ANSI Blog*. [online] The ANSI Blog. Available at: https://blog.ansi.org/2018/09/sae-levels-driving-automation-j-3016-2018/#gref [Accessed 9 Nov. 2018].

Vox (2016). *The 5 levels of driving automation*. [image] Available at: https://www.vox.com/2016/9/19/12966680/department-of-transportation-automated-vehicles [Accessed 19 Nov. 2018].

DARPA. (n.d.). *DARPA Urban Challenge*. [online] Available at: http://archive.darpa.mil/grandchallenge/ [Accessed 11 Nov. 2018].

Hall, D. (2017). *128 Lasers on the Car Go Round and Round: David Hall on Velodyne's New Sensor*. [online] Velodyne LiDAR. Available at: https://velodynelidar.com/newsroom/128-lasers-car-go-round-round-david-hall-velodynes-new-sensor/ [Accessed 15 Nov. 2018].

Neon Science (2014). *How Does LiDAR Remote Sensing Work? Light Detection and Ranging*. [video] Available at: https://www.youtube.com/watch?v=EYbhNSUnIdU [Accessed 8 Nov. 2018].

Higgins, S. (2017). *Hovermap: Powerful SLAM for Drone Autonomy and Lidar MapPing - SPAR 3D*. [online] SPAR 3D. Available at: https://www.spar3d.com/news/uav-uas/hovermap-powerful-slam-drone-autonomy-lidar-mapPing/ [Accessed 20 Nov. 2018].

OREFIND (2013). *Original LiDAR point data*. [image] Available at: http://www.orefind.com/blog/orefind_blog/2013/04/02/rest-in-peace-topographic-contours---part-2 [Accessed 21 Nov. 2018].

Heater, B. (2016). *Technology is killing jobs, and only technology can save them*. [online] TechCrunch. Aavailable at: https://techcrunch.com/2017/03/26/technology-is-killing-jobs-and-only-technology-can-save-them/ [Accessed 21 Nov. 2018].

Bonnefon, J., Shariff, A. and Rahwan, I. (2016). The social dilemma of autonomous vehicles. *Science*, 352(6293), pp.1573-1576.

Day, C. (2018). An investigation into the application of a Raspberry Pi as a microcontroller to an autonomous buggy with object avoidance capabilities and a comparison of computer visionmachine learning algorithms for pedestrian recognition. Undergraduate. Plymouth University.

McEachen, L. and Day, C. (2018). *Autonomous Buggy Manual*.

Orbbec3d.com. (n.d.). *Astra S Technical Specs*. [online] Available at: https://orbbec3d.com/product-astra/ [Accessed 29 Nov. 2018].

Hokuyo-aut.jp. (n.d.). *URG-04LX Product Details*. [online] Available at: https://www.hokuyo-aut.jp/search/single.php?serial=165 [Accessed 29 Nov. 2018].

RobotShop. (n.d.). *LiDAR, Laser Scanners and Rangefinders*. [online] Available at: https://www.robotshop.com/uk/lidar.html [Accessed 29 Nov. 2018].

Dwivedi, P. (2017). *LiDAR point cloud*. [image] Available at: https://towardsdatascience.com/tracking-pedestrians-for-self-driving-cars-ccf588acd170 [Accessed 29 Nov. 2018].

Renishaw (n.d.). *Diagram of the LiDAR optics and encoders*. [image] Available at: https://www.renishaw.com/en/optical-encoders-and-lidar-scanning--39244 [Accessed 29 Nov. 2018].

Raspberry Pi. (n.d.). *Raspberry Pi 3 Model B*. [online] Available at: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/ [Accessed 29 Nov. 2018].

Lambert, F. (2016). *A look at Tesla's new Autopilot hardware suite: 8 cameras, 1 radar, ultrasonics & new supercomputer*. [online] Electrek. Available at: https://electrek.co/2016/10/20/tesla-new-autopilot-hardware-suite-camera-nvidia-tesla-vision/ [Accessed 16 Nov. 2018].

Hanley, S. (2018). *Waymo Orders 62,000 Autonomous Chrysler Pacifica Hybrid Vans | CleanTechnica*. [online] CleanTechnica. Available at: https://cleantechnica.com/2018/06/01/waymo-orders-62000-autonomous-chrysler-pacifica-hybrid-vans/ [Accessed 11 Nov. 2018].