Technical University of Crete
School of Electrical and Computer Engineering
Intelligent Systems Laboratory

Autonomous Navigation of an Electric Vehicle

Diploma Thesis by

Nikolaos M. Sarantinoudis

Submitted in partial fulfilment of the requirements for the Diploma of Electrical and Computer Engineer at the Technical University of Crete

September 2018

To be defended publicly on Wednesday, 26th of September at 14:00
against the thesis committee consisting of

Associate Professor Michail G. Lagoudakis
School of Electrical and Computer Engineering

Professor Konstantinos C. Kalaitzakis
School of Electrical and Computer Engineering

Professor Nikolaos C. Tsourveloudis
School of Production Engineering and Management

This page was intentionally left blank

Abstract

Autonomous driving is one of the major areas of interest for the automotive industry. This constantly evolving field requires the involvement of a wide range of engineers with complementary skills. The education of these engineers is a key issue for the further development of the field. Currently in the engineering curricula, there is a lack of related platforms that can assist the engineers to train in and further develop the required dexterities. The current practice is to use either small robotic devices or full scale prototypes in order to understand and experiment in autonomous driving principles. Each approach has disadvantages ranging from the lack of realistic conditions to the cost of the platforms and sensors being used. In this thesis we present a low-cost and open-source modular electric vehicle platform, consisting from off-the-shelf components, which can be used for experimentation and research in the area of autonomous cars. This proposed platform, an urban concept vehicle, aims to tackle the problems of realistic conditions and cost respectively. Equipped with perception sensors, such as camera, lidar and ultrasonics, as well as navigation sensors such as GPS and IMU, provides the ideal foundation for anyone dealing with autonomy - from beginners to experts. The motivation for this work was to construct and provide a functioning platform for research purposes in the domain. The functionality of the suggested system is verified by extensive experimentation in very-close-to-real traffic conditions proving reliability, robustness and easy adaptability in diverse test cases.

Διπλωματική εργασία του φοιτητή

Νικόλαου Μ. Σαραντινούδη

με θέμα

Αυτόνομη Πλοήγηση Ηλεκτρικού Οχήματος

Περίληψη

Η αυτόνομη πλοήγηση οχημάτων είναι μια από τις κύριες περιοχές ενδιαφέροντος στην αυτοκινητοβιομηχανία. Η συνεχής εξέλιξη του πεδίου αυτού προϋποθέτει την ενασχόληση ενός εύρους μηχανικών με αλληλοσυμπληρούμενες ικανότητες, η εκπαίδευση των οποίων αποτελεί κλειδί για την περαιτέρω ανάπτυξη του τομέα αυτού. Ωστόσο, πλατφόρμες ικανές να βοηθήσουν στην ανάπτυξη των απαραίτητων δεξιοτήτων δεν είναι διαθέσιμες. Συνήθως χρησιμοποιούνται είτε ρομποτικά οχήματα υπό κλίμακα, είτε πραγματικών διαστάσεων πρωτότυπα, για την κατανόηση των βασικών αρχών της αυτόνομης πλοήγησης. Σε κάθε περίπτωση υπάρχουν μειονεκτήματα όπως είναι η έλλειψη ρεαλιστικών συνθηκών ή το κόστος των οχημάτων και των αισθητήρων που χρησιμοποιούνται αντίστοιχα. Στην παρούσα εργασία, παρουσιάζεται μια πλατφόρμα ηλεκτρικού οχήματος, που μπορεί να χρησιμοποιηθεί για έρευνα στον τομέα των αυτόνομων οχημάτων. Η προτεινόμενη πλατφόρμα, ένα πρωτότυπο όχημα πόλης, αποσκοπεί στο να αντιμετωπίσει τα προβλήματα των μη ρεαλιστικών συνθηκών και του κόστους αντίστοιχα. Είναι εξοπλισμένη με αισθητήρες αντίληψης του περιβάλλοντος, όπως κάμερες, λέιζερ σαρωτή και αισθητήρες υπερήχων αλλά και αισθητήρες πλοήγησης όπως μονάδα παγκόσμιας στιγματοθέτισης αλλά και μονάδα αδρανειακής μέτρησης, και προσφέρει τα απαραίτητα εφόδια σε οποιονδήποτε σκοπεύει να ασχοληθεί με τον τομέα της αυτονομίας. Ο σκοπός της παρούσας εργασίας ήταν η κατασκευή μιας λειτουργικής πλατφόρμας που θα υποστηρίξει της ερευνητικές προσπάθειες στον τομέα αυτό. Η λειτουργικότητά της έχει επαληθευτεί μέσα από εκτεταμένα πειράματα σε -σχεδόν- πραγματικές συνθήκες αποδεικνύοντας την αξιοπιστία της και την δυνατότητα άμεσης προσαρμογής σε διαφορετικά σενάρια.

Acknowledgements

Many colleagues have participated and helped in the completion of this thesis document, each and every one of them with their unique way. It feels appropriate to dedicate this part of my thesis to them in order to express my acknowledgements.

- Firstly, I'd like to thank my supervisor Assoc. Professsor Michail G. Lagoudakis for his support, his advice and help to the completion of this Thesis Document.

- Prof. Konstantinos C. Kalaitzakis, for being a member of the thesis committee and accepting to evaluate my work.

- Prof. Nikolaos C. Tsourveloudis,Dean of the Production Enginnering and Management School, for his confidence and trust to allow me undertake such an ambitious and costly project and exploit the resources of the Machine Tools Laboratory and the Intelligent Systems and Robotics Laboratory of the School of Engineering and Management.

- Dr. Polychronis Spanoudakis, head of the TUCer Team, for the flawless cooperation and unreserved help, not only for this thesis, but also the previous two years on TUCer team and the trust on me for past and upcoming projects.

- Dr. Lefteris Doitsidis, Associate Professor in the Department of Electronic Engineering, Technological Educational Institute of Crete for his support in every aspect of this thesis from the beginning until the final presentation.

- Dr. Savvas Piperidis, member of TUcer Team, head of the Electronics Department, for his help in electronics design and wiring, as well as for equipment and sensors selection.

- TUCer Team Members, Gerasimos Moschopoulos, Eftichis Papadokokolakis and Thodoris Stefanoulis for their help in mechanical construction and maintenance of the vehicle used in my thesis.

- And finally my family and friends, who supported me during this thesis and backed up my choice. Especially, I'd like to thank Eva Zymari and Kostas Tsakos who played a vital role in the final testing of the platform.

Αφιερώνεται στους γονείς μου,

Κατερίνα και Μιχάλη.

Contents

Abstract **i**

Acknowledgements **iii**

List of Tables **viii**

List of Figures **ix**

1 Introduction **3**

2 Background **5**

2.1 Classification of Autonomy Levels 7

2.2 Sensors 9

2.2.1 LiDAR 9

2.2.2 Camera 11

2.2.3 Ultrasonic Sensors 13

2.2.4 GPS 14

2.2.5 IMU 15

3 Related Work **17**

3.1 Commercial Approach to Autonomous Driving 17

3.1.1 Waymo (Google Self-Driving Car Project) 17

3.1.2 Tesla 18

3.1.3 Audi 19

3.1.4 General Motors 20

3.1.5 Various Car Manufactures 21

3.2 Educational Approach to Autonomous Driving 21

3.2.1 Duckietown 22

3.2.2 Donkey Car 22

3.2.3 BARC - Berkeley Autonomous Race Car 23

3.2.4 MIT Racecar 24

3.2.5 F1/10 24

4 Our Approach **26**

5 Platform Description **30**

5.1 Platform Architecture 30

5.1.1 Power Supply 30

5.1.2 Motor & Motor Controller 33

5.1.3 Transmission 33

5.1.4 Steering and Braking System 34

5.1.5 Processing Unit 36

5.1.6 Sensors 36

5.1.7 Various Electronic Components 42

5.2 Software Development 44

5.2.1 Embedded Code Development 45

5.2.2 External Code Development 45

6 Implementation and Experimentation **47**

6.1 Preliminary Testing - Platform Validation 47

6.2 GPS Waypoints Path Following 51

6.3 Object Detection and Avoidance 59

7 Conclusion **63**

8 Bibliography **65**

Appendices **70**

A Platform User Manual **70**

B Hardware Specification Sources **79**

List of Tables

4.1 Comparison of Various Educational Platforms 28

5.1 Sensors Classification 37

List of Figures

2.1 Simplified workflow of autonomy 5

2.2 Various approaches on autonomous driving 6

2.3 SAE Autonomy Level 8

2.4 Automotive Grade LiDAR with 128 layers 9

2.5 Separation of two LiDAR points at a given range 10

2.6 Pointcloud of a LiDAR reading 11

2.7 Automotive Camera Modules 12

2.8 Influence of external conditions on camera systems 13

2.9 Ultrasonic Sensors operation principle 14

2.10 GPS Satellite constellation 14

2.11 GPS Receiver Connectivity 15

2.12 The nine Degrees of Freedom of an IMU 16

3.1 Waymo's Current Architecture 18

3.2 Waymo's Firefly and Chrysler Pacifica Hybrid 18

3.3 Tesla's Current Architecture 19

3.4 Audi's Architecture 20

3.5 General Motor's Architecture 21

3.6 Duckiebot and Duckietown 22

3.7 Donkey Car Platform 23

3.8 BARC Platform 23

3.9 MIT Racecar Platform 24

3.10 F1/10 25

4.1 The experimentation platform 27

4.2 Platform Overview 28

4.3 Urban Mobility Vehicles 29

5.1 Vehicle Power Supplies 31

5.2 DC/DC Buck Converters 32

5.3 Simplistic Power Supply Schematic 32

5.4 Motor and Motor Controller 33

5.5 Transmission Set-Up 34

5.6 Steering Mechanism 35

5.7 Braking Mechanism 35

5.8 Nvidia Jetson TX2 36

5.9 Sensors Overview 37

5.10 ZED Stereo Camera 38

5.11 LiDAR Sensor 39

5.12 Maxbotix Ultrasonic Sensors 39

5.13 Perception Sensors Subsystem 40

5.14 Adafruit GPS Module 41

5.15 SparkFun Razor IMU Module 41

5.16 Localization Sensors Subsystem 42

5.17 Autonomous Navigation System Overview 44

5.18 Hand-Soldered Prototyping Boards 44

5.19 External Code Development Cycle 45

6.1 Instance of the Control Logging `.csv` file 48

6.2 Instance of the Sensors Logging `.csv` file 49

6.3 Instance of the LiDAR Logging `.csv` file 50

6.4 Screenshot from the camera recording 51

6.5 Pure Pursuit Geometry 52

6.6 Flowchart of the Steering Control Procedure 55

6.7 Simulation of the Pure Pursuit Method on MATLAB 56

6.8 Vehicle's trajecory inside the TUC Campus 57

6.9 Alternative trajecory inside the TUC Campus 58

6.10 Target Coordinates and Actual Path followed 59

6.11 Data acquired from the LiDAR sensor 60

6.12 Pedestrian Detection with vision 61

6.13 Flowchart of the Steering Control and Object Avoidance Procedure 62

A.1 CPU and USB hub power supply 71

A.2 Jetson TX2 and USB hub power led and button 71

A.3 Running state of Jetson TX2 and USB Hub 72

A.4 Command to connect to Jetson TX2 through terminal 72

A.5 Screen after successful connection 73

A.6 Command to enable Maximum Performance mode 73

A.7 Connection Error Common Problem 74

A.8 Jetson TX2 Reset Button 74

A.9 12V - 40Ah battery 75

A.10 Main Power Supply Holder and Fuse 75

A.11 Main Power Supply Connection 76

A.12 Safety Buttons 76

A.13 Status Leds 77

Acronyms and Abbreviations

2D — Two Dimensional

3D — Three Dimensional

A — Ampere

ACC — Adaptive Cruise Control

Ah — Ampere Hour

AI — Artificial Intelligence

ARM — Advanced RISC Machine

AV — Autonomous Vehicle

CAN — Controller Area Network

CPU — Central Processing Unit

CSV — Comma Separated Value

DC — Direct Ccurrent

eMMC — embedded Multi Media Card

FOV — Field of View

FPS — Frames Per Second

GPIO — General Purpose Input Output

GPS — Global Positioning System

GPU — Graphics Processing Unit

I/O — Input Output

I2C — Inter-Integrated Circuit

I2S — Inter-IC Sound

IEC — International Electrotechnical Commission

IMU — Inertia Measurement Unit

LIDAR — Light Detection and Ranging

LPDDR4 — Low Power Double Data Rate 4

LRR — Long Range Radar

MEO — Medium Earth Orbit

MRR — Medium Range Radar

Nm — Newton Meter

PC — Personal Computer

PNT — Positioning, Navigation and Timing

RC — Radio Controlled

ROS — Robotics Operating System

SAE — Society of Automotive Engineers

SATA — Serial Advanced Technology Attachment

SD Card — Secure Digital Card

SDK — Software Development Kit

SLAM — Simultaneous Localization and Mapping

SPI — Serial Peripheral Interface

SRR — Short Range Radar

SSD — Solid State Disk

TUC — Technical University of Crete

TUCer Team — Technical University of Crete Eco Racing Team

UART — Universal Asynchronous Receiver Transmitter

UAV — Unmanned Aerial Vehicle

US — United States of America

USB — Universal Serial Bus

V — Volts

W — Watts

Chapter 1

Introduction

In recent years, autonomous driving has emerged as a major innovation in the automotive industry. Currently the technology has matured and commercialisation is expected in the following years. Autonomous driving has significant internal (user) and external (external actors) impact. It provides a stressful environment for the driver and increases its productivity during the time spent inside the car, as well as it provides the option to non-drivers (or incapable to drive people) to commute. On the external side, increased safety (reducing crash risks and high-risk driving), increased road capacity and reduced costs, increased fuel efficiency and reduced pollution are equally important [1]. Even though the related technology has made significant advances in this domain, the cost of developing and experimenting with autonomous cars still remains high. Apart from the platform itself, high quality sensors and powerful processing units, which are essential for autonomous capabilities, are very expensive. The aforementioned cost makes it difficult for a platform to be adopted in the educational and research activities of higher education institutes. There is always the option of small autonomous robotic vehicles, which are adequate for understanding the basic concepts of autonomous driving, or even inexpensive RC-car based autonomous car testbeds, but few attempts have been made to provide low cost realistic platforms for research and education. Providing a realistic platform for experimentation is significant, since it will prepare future engineers for the transition to full-scale prototypes and it will simulate the response and behaviour of an every-day vehicle much more accurately, providing an insight on difficulties and safety concerns of the full-scale prototypes.

This thesis proposes an approach that will allow the educational process on the domain of autonomous driving to become viable and inexpensive, providing students the ability to apply the principles related to autonomous driving on a realistic low-cost platform using tools that will minimise the development time and maximise the efficiency, gaining important hands-on experience. The proposed platform has been extensively tested at the Technical University of Crete, in close-to-real traffic conditions, in the campus parking spaces and has been proven reliable, robust and easily adaptable to the various test cases imposed. It is important to point out here that all our tests exhibit basic functionality and don't aim to provide any novelty or new findings in autonomous car navigation, since this will be the target of the users of the proposed platform.

The rest of this thesis is organised as follows. In the second chapter, the concept of autonomous driving is briefly explained and basic background theory about the sensors used is presented. In the third chapter, related commercial and academic work is listed and briefly explained as well as our own approach. The fourth chapter describes the platform itself, every component used for this thesis completion and the software development procedures. The fifth chapter refers to the experiments conducted for proving robustness and proper functionality of the platform. The final chapter concludes the present thesis and proposes future improvements.

This diploma thesis work was partially presented at the 10^{th} Workshop on Planning, Perception and Navigation for Intelligent Vehicles (PPNIV'18), in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2018), Madrid, Spain on October 1st.

N. Sarantinoudis, P. Spanoudakis, L. Doitsidis, T. Stefanoulis, N. Tsourveloudis, "Enhancing the educational process related to autonomous driving," in 10^{th} Workshop on Planning, Perception and Navigation for Intelligent Vehicles, Madrid, Spain, October, 2018.

Chapter 2

Background

Autonomous navigation, refers to the ability of a vehicle to perceive its surroundings with various attached sensors, evaluate the conditions and navigate to an exact location safely without the interference of a human driver, taking into consideration the ever-changing and unpredictable environment. Explaining autonomy further, like every robotic system, autonomous vehicles utilize the “sense-plan-act“ design. Using a combination of sensors to perceive the environment, they continuously gather information in real time [2]. The gathered data are then analyzed and fused together, allowing the vehicle to identify its surroundings. Finally, decisions are taken and the vehicle is controlled accordingly. In Figure 2.1 this simplified explanation of autonomy is visualized.

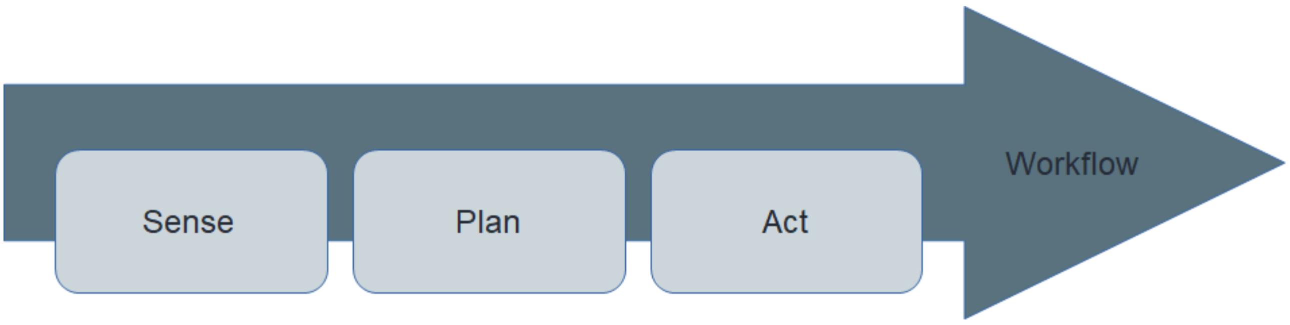


Figure 2.1: Simplified workflow of autonomy

Sensing is the basis of autonomous driving. If the environment around the vehicle is totally unknown then there is no chance of getting anywhere. The level of detail and accuracy required for autonomous driving dictates the use of multiple sensors in tandem. Two major categories can be distinguished in the domain of sensors. The perception sensors and the localization sensors. Both are equally important. Perception sensors used in autonomous driving are usually LiDAR (Laser Detection and Ranging), Radar, Ultrasonic Sensors and Cameras. LiDAR sensor bounces a laser beam off surfaces and measures the reflection time to determine the vehicle’s distance from various objects. Because LiDAR is highly accurate, an autonomous vehicle can rely on it as the main sensor to produce high-definition maps, localize itself while moving (using the map as a reference), and detect obstacles. Cameras are used mostly for object recognition and tracking, for example, to detect other cars, lanes, traffic lights, and pedestrians. To enhance safety, existing implementations usually mount five or more cameras around the car, so that they can detect, recognize, and track objects in front of, behind, and on both sides of the vehicle. Radar and ultrasonic sensor system is used primarily as the last line of defense in obstacle avoidance generating data showing the distance to the nearest object in front of the vehicle's path.

Localization is handled by GPS (Global Positioning System), IMU (Inertia Measurement Unit) and odometers. The GPS/IMU combination reports both a global position estimate and inertial updates at a high rate and is thereby essential in helping the autonomous vehicle localize itself. A GPS provides fairly accurate localization, but its update rate is only 10 Hz, rendering it incapable of providing real-time updates. Conversely, an IMU's accuracy degrades with time, so it cannot provide reliable positioning over long periods. It can, however, furnish more frequent updates at 200 Hz or higher, which satisfies the real-time requirement. Combining GPS and IMU provides both accurate and real-time updates for vehicle localization.

While all major and minor pioneers on autonomous driving are persuaded that the approach to localization is correct, perception is an entirely different story. There are many different approaches to perception and they don't differ only in the number of sensors and the placement around the vehicle. In Figure 2.2 three seperate approaches from three different Original Equipment Manufacturers (OEM) are presented.

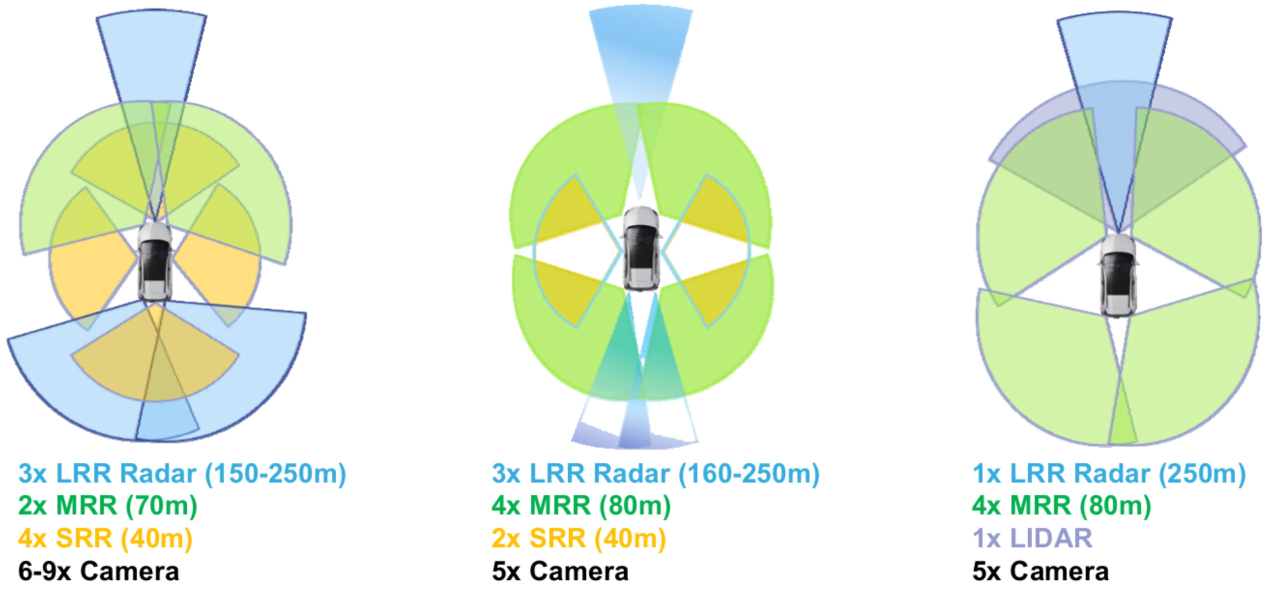


Figure 2.2: Various approaches on autonomous driving
Source: Embedded Vision Summit - NXP Semiconductors

Examining the previous figure we can see that the main difference between these approaches is the use or not of a LiDAR sensor. LiDAR sensors are quite robust and deliver a high level of detail, however LiDAR measurements can be extremely noisy, for example when raindrops and dust are in the air. Adding to LiDAR disbelief, is the pricing also, with a 64-beam LiDAR costing at around $ 80,000 while in contrast cameras are much more cheaper.

Planning includes interpretation of all the gathered data from the various sensors followed by decision-making. Data streams alone say nothing to a computer. Various algorithms, such as object recognition, object tracking and path planning, are applied to sensors outputs. Object recognition and tracking is performed in order to allow the vehicle to recognize what lies around it and how its position is changing relative to it. The main goal is to ensure that the vehicle does not collide with a moving object, whether a vehicle or a person crossing the road. Path planning is the process of selecting a path to safely navigate to the target position, accounting for every expected (stop signs, traffic lights) or unexpected (pedestrian, road accident) event. It is very important to mention that these procedures must run without any interruption and in real-time, explaining the need for extreme computational capabilities in autonomous navigation [3]. Acting is the translation of the previous step to actual movement of the car. Every major system responsible for driving (steering wheel, throttle, brake) is controlled by the main processing unit. Thanks to drive-by-wire (meaning that everything on a car is controlled electronically) this is possible to occur. The processing unit communicates with each system's local controller, passing real-time commands just like a human driver would have done.

2.1 Classification of Autonomy Levels

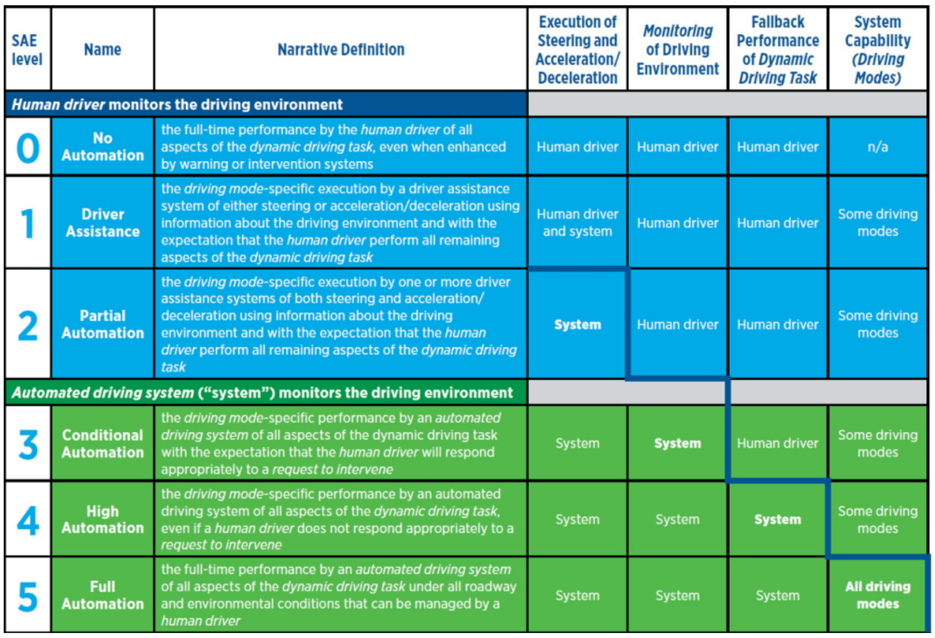
Due to the broadness of the term autonomy, SAE International, an automotive standardization body, has explicitly defined the levels of automation to characterize the extent to which a vehicle can drive autonomously. A classification system based on six different levels (ranging from fully manual to fully automated systems) was published in 2014, as J3016, Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems. This classification system is based on the degree of driver intervention and attentiveness required, rather than the vehicle capabilities, although these are very closely related. In 2016, SAE revised its classification now called J3016_201609 [4].
Following is an explanation of SAE's taxonomy paired with examples in order to be easily comprehended:

- Level 0: Human intervention is required at all times. Automated system issues warnings and may momentarily intervene, but has no sustained vehicle control.
- Level 1 ("hands on"): The driver and the automated system share control of the vehicle. Examples are Adaptive Cruise Control (ACC), where the driver controls steering and the automated system controls speed and Parking Assistance, where steering is automated while speed is under manual control. The driver must be ready to retake full control at any time.
- Level 2 ("hands off"): The automated system takes full control of the vehicle (accelerating, braking, and steering). The driver must monitor the driving and be prepared to intervene

immediately at any time if the automated system fails to respond properly. The shorthand “hands off“ is not meant to be taken literally. In fact, contact between hands and steering wheel is often mandatory during SAE Level 2 automated driving, to confirm that the driver is ready to intervene.

- Level 3 (“eyes off“): The driver can safely turn their attention away from the driving tasks, e.g. the driver can text or watch a movie. The vehicle will handle situations that call for an immediate response, like emergency braking. The driver must still be prepared to intervene within some limited time, specified by the manufacturer, when called upon by the vehicle to do so.
- Level 4 (“mind off“): As level 3, but no driver attention is ever required for safety, i.e. the driver may safely go to sleep or leave the driver’s seat. Self driving is supported only in limited spatial areas (geofenced) or under special circumstances, like traffic jams. Outside of these areas or circumstances, the vehicle must be able to safely abort the trip, i.e. park the car, if the driver does not retake control.
- Level 5 (“steering wheel optional“): No human intervention is required at all.

In Figure 2.3 the formal description of SAE Autonomy Levels can be seen. The last column “driving mode“ means “a type of driving scenario with characteristic dynamic driving task requirements (e.g., expressway merging, high speed cruising, low speed traffic jam, closed-campus operations, etc.)“ that are explicitly stated in J3016_201609 [4].



SAE level	Name	Narrative Definition	Execution of Steering and Acceleration/ Deceleration	*Monitoring* of Driving Environment	Fallback Performance of *Dynamic Driving Task*	System Capability *(Driving Modes)*
***Human driver* monitors the driving environment**						
0	No Automation	the full-time performance by the *human driver* of all aspects of the *dynamic driving task*, even when enhanced by warning or intervention systems	Human driver	Human driver	Human driver	n/a
1	Driver Assistance	the *driving mode*-specific execution by a driver assistance system of either steering or acceleration/deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	Human driver and system	Human driver	Human driver	Some driving modes
2	Partial Automation	the *driving mode*-specific execution by one or more driver assistance systems of both steering and acceleration/ deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	**System**	Human driver	Human driver	Some driving modes
***Automated driving system* (“system”) monitors the driving environment**						
3	Conditional Automation	the *driving mode*-specific performance by an *automated driving system* of all aspects of the dynamic driving task with the expectation that the *human driver* will respond appropriately to a *request to intervene*	System	**System**	Human driver	Some driving modes
4	High Automation	the *driving mode*-specific performance by an automated driving system of all aspects of the *dynamic driving task*, even if a *human driver* does not respond appropriately to a *request to intervene*	System	System	**System**	Some driving modes
5	Full Automation	the full-time performance by an *automated driving system* of all aspects of the *dynamic driving task* under all roadway and environmental conditions that can be managed by a *human driver*	System	System	System	**All driving modes**

Figure 2.3: SAE Autonomy Level
Source: “Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles,” SAE, Tech. Rep., rev.2016.

2.2 Sensors

Multiple sensors have been previously referred as a very important part of autonomous navigation. Two main categories have been distinguished, the perception sensors and the navigation sensors. Following there is some important background information about the basic working principles of the sensors for easier and better comprehension of the referenced technology.

2.2.1 LiDAR

A LiDAR (Light Detection and Ranging) is an active optical ranging sensor. Some of the applications are obstacle detection, pedestrian and vehicle detection [5],[6] lane recognition [7], and determination of the exact position of the vehicle [8]. It emits light pulses at a high frequency using a laser beam. If the emitted light hits an object, the reflected light is measured and used to calculate the distance between the sensor and the object. Since the laser beam can only measure a single point at a time, the LiDAR has to measure at very high frequencies to build a high resolution depth image at the desired update rate. Meanwhile, the laser beam is aimed using rotating mirrors or by rotating the entire sensor unit. Automotive LiDARs typically scan in a horizontal direction which is separated in layers. A large number of layers allows the LiDAR to compensate for the pitch angle of the vehicle easier [6] and reduces the influence of occlusions [9]. Currently available LiDAR sensors use up to 128 layers simultaneously (Figure 2.4). The range of a LiDAR can be over 250 *meters* and the FOV (Field of View) as large as 360 *degrees*. Unlike radar sensors, LiDARs can have large FOV and range at the same time.

Figure 2.4: Automotive Grade LiDAR with 128 layers
Source: Velodyne

The LIDAR generates a point cloud, where every point represents a single distance measurement (Figure 2.6). This point cloud has to be processed in order to extract object information. Classification algorithms can be used to classify the detected objects [10]. To detect an object reliably, it is necessary to have at least two LiDAR returns from that object. Given an angular resolution α and a range r , the distance between two LiDAR points δ (illustrated in Figure 2.5) can be calculated using the following equation:

$$\delta = 2r\sin(\alpha/2)$$

Since $r \gg \delta$, this reduces to:

$$\delta = \alpha r$$

In order to be able to see an object of size l at range r, the object has to be at least 2δ wide to guarantee two LiDAR returns, assuming single layer scanning. This means that larger objects can be detect further away compared to smaller objects. LiDARs typically have a high angular resolution, which allows even small objects to be detected at fairly large distances. The angular resolution required to detect an object of size l at range r can be described using the equation below, which is confirmed in [11]:

$$\frac{l}{2r}$$

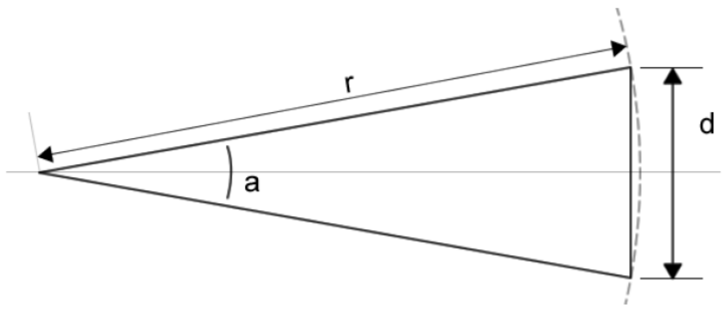


Figure 2.5: Separation of two LiDAR points at a given range
Source: Sensing requirements for an automated vehicle for highway and rural environments, MSc Thesis, K.J.Bussemaker

LiDARs can't detect velocity information directly. Tracking algorithms have to be used to analyze the position of an object over time, which allows the velocity to be estimated [12]. This is done by comparing two (or more) consecutive LiDAR readouts. Like vision systems, LiDARs are affected by adverse weather and lighting conditions [6], however "multi-echo" technology can improve performance in these conditions by evaluating multiple LiDAR returns [11]. A clear line-of-sight is required, which means the sensor has to be placed in the open air or behind a transparent surface. Under good conditions, LiDARs can have a large range (over 200 m). Another important factor that influences the range is the reflectivity of an object. An object with high reflectivity can be detected at full range, but objects that reflect poorly can be detected at a significantly lower range. Most LiDARs have a very high resolution and accuracy specifications on paper, however manufacturers are very unspecific about whether these numbers apply to the point cloud data or to the interpreted object detection data (if there's any on board processing). The quality of the LiDAR data is only as good as its object recognition algorithms, and some accuracy is likely to be lost during this stage. Due to the sheer volume of data typically generated by LiDAR sensors, concessions often have to be made when it comes to processing, possibly decreasing the accuracy even further. Lasers can be harmful to the humans and animals, especially to the eyes. Like all laser products, LiDARs are subjected to regulations defined by IEC 60825-1 [13]. This puts a limit on the LiDAR range, which (among other factors) depends on the power of the emitted pulse [14]. Most LiDARs are class 1 laser products, which means they are completely safe under normal operating conditions.

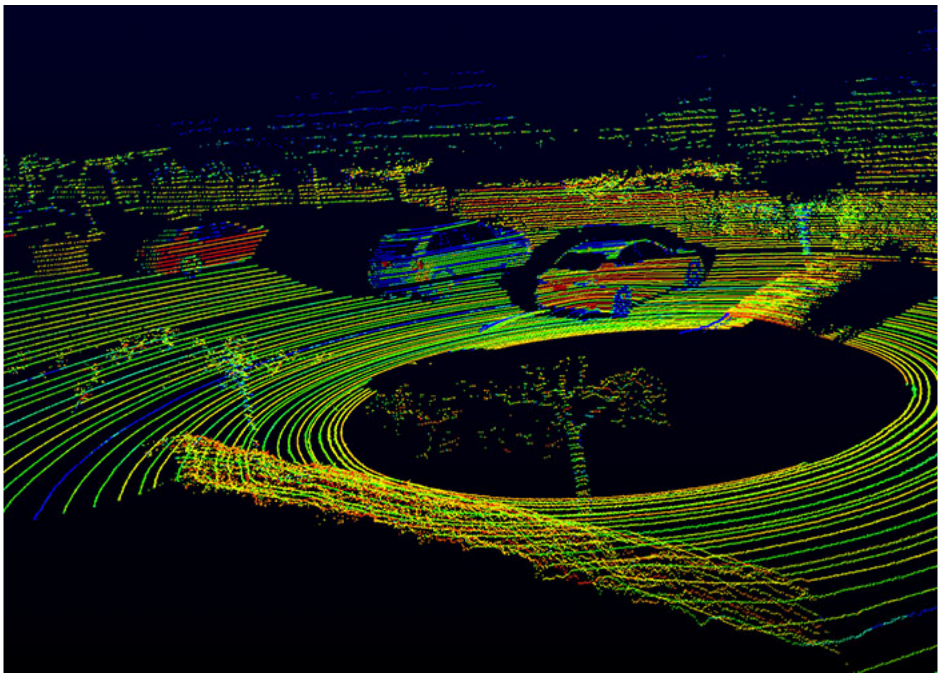


Figure 2.6: Pointcloud of a LiDAR reading
Source: Velodyne

2.2.2 Camera

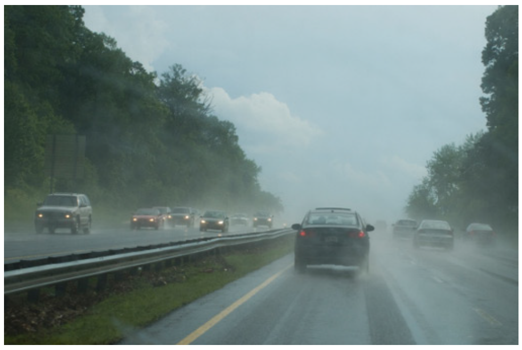
A vision system uses a light-sensitive sensor to form an image of the surroundings. Some highly important aspects of driving can only be sensed through a vision system, for example a sign could be detected by a number of different sensors, but the actual image on the sign can only be perceived using a vision system. Vision systems take advantage of the visual light spectrum (although some sensors are also able to pick up parts of the infrared spectrum, which allows for night vision [15]). A vision sensor requires an unobstructed line of sight, which means it has to be placed in open air or behind a translucent surface (i.e. the wind shield). A vision system relies on image processing algorithms to detect and classify objects. Image processing is a demanding process, but the information that can be extracted from images is very useful and can be used for many tasks like mapping and navigation, object detection and recognition, collision detection and more [16]. The processing algorithms used heavily influence the performance of a vision system. If we consider the pinhole camera model, an object's width and height in pixels is inversely proportional to its distance to the camera [17]. The work of [17] has described the relationship between a collection of state-of-the-art pedestrian recognition algorithms and the height of a pedestrian in pixels. The further away a pedestrian is, the smaller its size in pixels will be, hence the chance of a pedestrian going undetected increases. An obvious conclusion is that a high resolution is beneficial. A slightly more disturbing conclusion is that pedestrian detection using vision alone is not very reliable, even when using the best algorithms currently available. Fusion of vision systems with LiDAR sensors has the potential to improve pedestrian detection performance [18], but this is still an open area of research [15]. Two main types of vision systems can be distinguished: monoscopic and stereoscopic (Figure 2.7). Monoscopic vision systems simply use one optical sensor. Stereoscopic vision systems consist of two optical sensors with a small distance between them. Similar to a pair of human eyes, this allows depth perception. Stereo vision uses disparity (the difference in position of a point or area between the left and right image) to calculate a depth image. The accuracy of this depth image decreases greatly with distance since the disparity is much lower for points that are far away from the sensor [16].

(a) Automotive Mono Camera

(b) Automotive Stereo Camera

Figure 2.7: Automotive Camera Modules
Source: Continental.

Visions systems are passive, which means their performance depends on external lighting conditions. Low lighting conditions occurring at night can degrade the performance of the sensor. High-intensity lighting conditions, such as a setting sun or bright headlights of an oncoming vehicle can also blind a vision sensor. Another major influence on a vision sensor's performance is the weather. Heavy rain, snow and especially fog can greatly decrease the effective range of the image processing algorithms (Figure 2.8). Most of the automotive vision systems currently available are multi-purpose units, and come with a built-in image processor. The unit is usually equipped with a number of algorithms for different detection and classification purposes, such as Traffic Sign Recognition, Pedestrian Detection and General Object Detection. As mentioned earlier, the detection and recognition performance is heavily influenced by the quality of the imaging sensor and the algorithms used by the processing unit.


(a) Rain influence

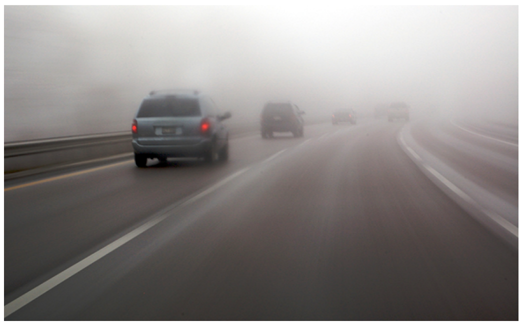

(b) Fog influence

Figure 2.8: Influence of external conditions on camera systems
Source: Sensing requirements for an automated vehicle for highway and rural environments, MSc Thesis, K.J.Bussemaker

2.2.3 Ultrasonic Sensors

Ultrasonic sensors operate by emitting a high-frequency audio signal, which will reflect off any object in front of the sensor. This reflected signal is detected, and the time difference between sending the signal and receiving the reflected signal is used to calculate the distance to the object. Ultrasonics are commonly used as parking sensors in modern cars. Since ultrasonics use an audio signal, their performance could be degraded if the air carrying the signal is moving. The faster the vehicle is travelling, the faster the airflow around the car body will be, possibly creating vortices as well. In [19], an ultrasonic sensor is used to measure the distance of a car under body. Their system was tested with speeds up to 120 km/h, and performed well for all typical driving manoeuvres, so we can assume that ultrasonic sensors can operate at least up to highway speed limits. Ultrasonic sensors are active, they rely on a internally generated audio signal to perform their measurements. This would mean that they could also pick up signals from other ultrasonic sensors. There are various ways of filtering, which can reduce this interference that are utilized in the ultrasonic sensor systems.

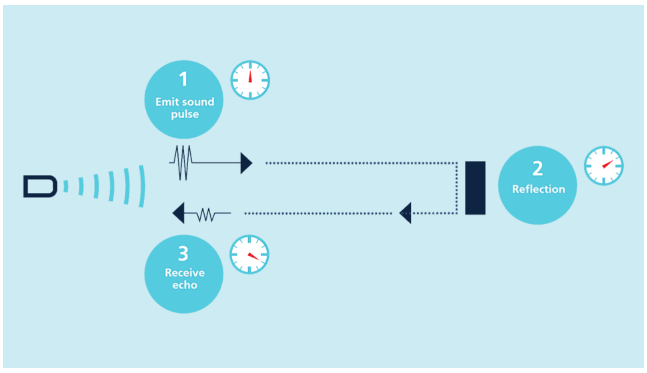


Figure 2.9: Ultrasonic Sensors operation principle
Source: Microsonic GmbH

2.2.4 GPS

The Global Positioning System (GPS) is a U.S.-owned utility that provides users with positioning, navigation, and timing (PNT) services. This system consists of three segments: the space segment, the control segment, and the user segment. The U.S. Air Force develops, maintains, and operates the space and control segments. The GPS space segment consists of a constellation of satellites transmitting radio signals to users. GPS satellites fly in Medium Earth Orbit (MEO) at an altitude of approximately 20,200 km (12,550 miles). Each satellite circles the Earth twice a day. The satellites in the GPS constellation are arranged into six equally-spaced orbital planes surrounding the Earth. Each plane contains four “slots“ occupied by baseline satellites. This 24-slot arrangement ensures users can view at least four satellites from virtually any point on the planet, which are required for accurate localization [20].(Figure 2.10).

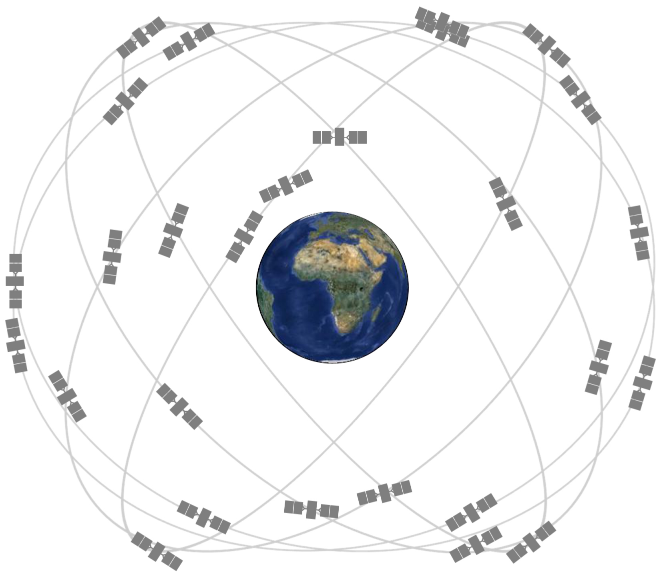


Figure 2.10: GPS Satellite constellation
Source:gps.gov

The GPS control segment consists of a global network of ground facilities that track the GPS satellites, monitor their transmissions, perform analyses, and send commands and data to the constellation [20]. The user segment consists of a GPS receiver placed on the object of interest that remains at all times connected with at least 4 satellites in order to be able to provide an accurate location. The operation of Global Positioning System is based on the 'trilateration' mathematical principle. The GPS receiver takes the information from the satellites and uses this method to determine the user's exact position. The position is determined from the distance measurements to satellites. Three satellites are used to trace the location. A fourth satellite is used to confirm the target location of each of those space vehicles [21]. (Figure 2.11).

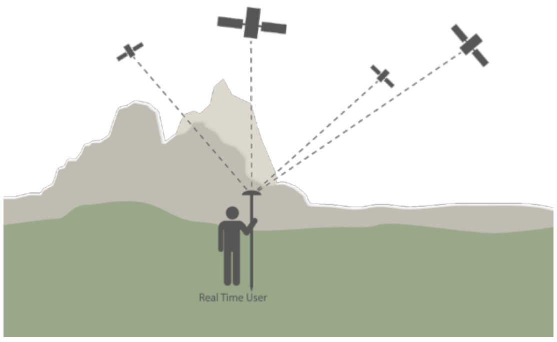


Figure 2.11: GPS Receiver Connectivity
Source: Federal Office of Topography, Switzerland

2.2.5 IMU

The Inertia Measurement Unit (IMU) is an electronic device that measures and reports a body's specific force, angular rate and surrounding magnetic field, using a combination of accelerometers, gyroscopes and magnetometers. They are self-contained systems that measure linear and angular motion of an object or vehicle. Measurements are summed over a time period to determine the instantaneous position, velocity, orientation, and direction of movement. IMUs usually measure nine degrees of freedom. This includes the measurement of linear motion over three perpendicular axes (surge, heave, and sway), as well as rotational movement about three perpendicular axes (roll, pitch, and yaw) and magnetic field strength over the same three axes of rotation. This yields nine independent measurements that together define the movement of an object or vehicle. It is comprised of at least two dedicated sensors, one or more linear accelerometers and one or more gyroscopes or angular accelerometers. An optional magnetometer may be integrated into the unit to calibrate against orientation drift. Accelerometers detect the direction and magnitude of change in velocity. Simple accelerometers measure linear motion, while biaxial and triaxial accelerometers detect a change in velocity over a plane or three-dimensional space, respectively. The IMU possesses a triaxial (sometimes referred to as a triad) accelerometer, or otherwise uses multiple accelerometers that are aligned across perpendicular axes. Gyroscopes detect the angular rate or orientation about a given directional vector. The angular rate is relative to a reference surface. The IMU uses multi-axis gyros to provide measurements in three orthogonal directions. These angular movements must be aligned with those of the accelerometer. Magnetometers measures the strength and, in some cases, the direction of the magnetic field at a point in space and give the values of magnetic field intensity along the 3-axes of the IMU, which are aligned with those of the accelerometer and gyroscope [22]. In Figure 2.12 we can see the 9 Degrees of Freedom of an IMU sensor.

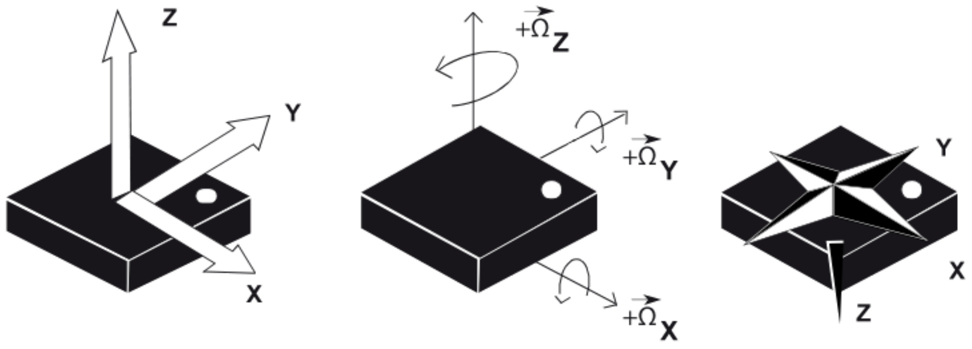


Figure 2.12: The nine Degrees of Freedom of an IMU
Source: Sparkfun

Chapter 3

Related Work

During the last decade there have been multiple companies that have engaged themselves with autonomous driving. And this extensive occupation has led to exceptional results. Some of the most influential and pioneering efforts can be found below.

3.1 Commercial Approach to Autonomous Driving

3.1.1 Waymo (Google Self-Driving Car Project)

One of the first companies working on the autonomous driving domain was Waymo [23]. Starting from 2009, as Google Self-Driving Project, it has experimented with multiple vehicles (Toyota Prius, Lexus Rx450h, Chrysler Pacifica Hybrid) and even "Firefly", a new reference vehicle designed by them. More than 7 million miles of testing, mostly on city streets have been conducted in multiple locations around the United States of America [24]. Currently the early rider program is conducted in Phoenix, Arizona where autonomous vehicles have been put to trial by the residents for their everyday commute. Future plans include the release of the first autonomous consumer vehicle for everyday use, as well as the construction of the the world's first premium autonomous electric car (together with Jaguar). Waymo's vehicle are the only SAE Level 4 vehicles that are used by commuters today. It is important to point out that Waymo is a firm supporter of the use of LiDAR technology in its vehicles. In Figure 3.1 Waymo's platform currently in use is briefly explained. In Figure 3.2 the left vehicle is Waymo's custom design (Firefly).

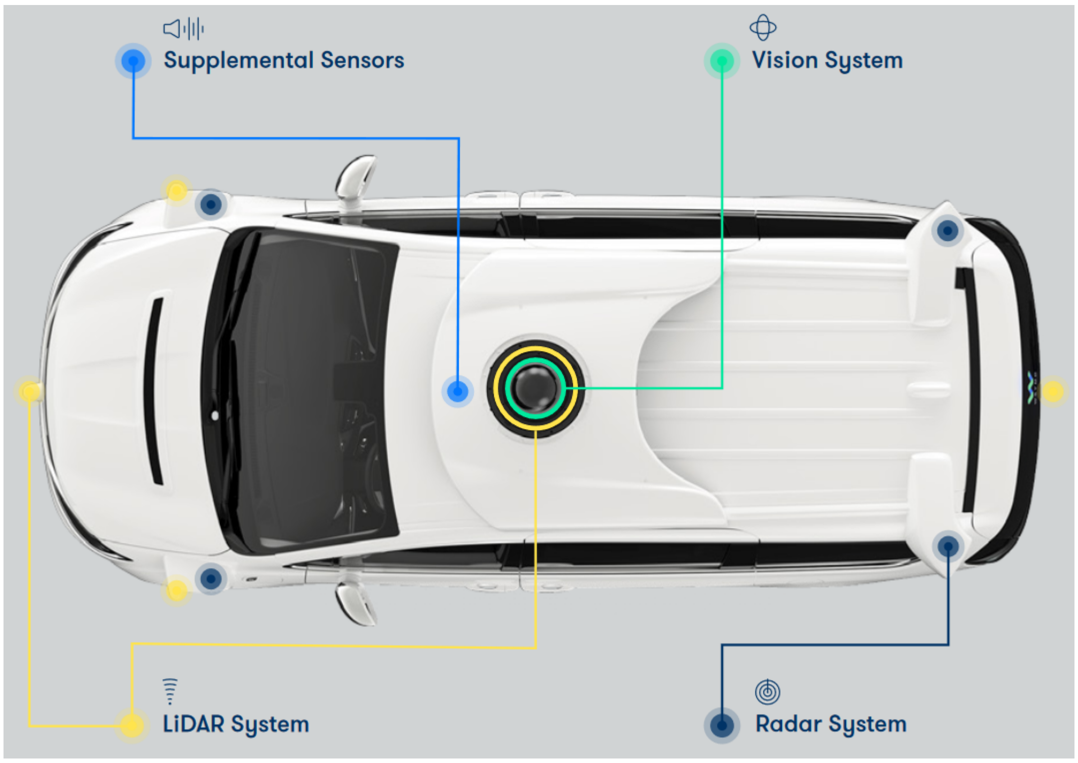


Figure 3.1: Waymo's Current Architecture
Source: Waymo

Figure 3.2: Waymo's Firefly and Chrysler Pacifica Hybrid
Source: Waymo

3.1.2 Tesla

Tesla introduced autonomous features in its cars in late 2014, and with a major update in 2016 states that they are capable of fully autonomous driving (Level 5) when the proper firmware will be released. Currently, Tesla is one of the few companies offering a vehicle to consumers that can drive itself, even if it is only on highway [25], aiming to unveil the fully autonomous driving capabilities in late-2018 and become the first company selling cars that drive themselves. Tesla's Autopilot achieves SAE Level 2 automation, however it is quite popular, since it was the first manufacturer introducing it in production vehicles. In contrast to Waymo, Tesla is against the use of LiDAR and incorporate multiple cameras and radars. In Figure 3.3 sensors onboard Tesla's production vehicle are listed.

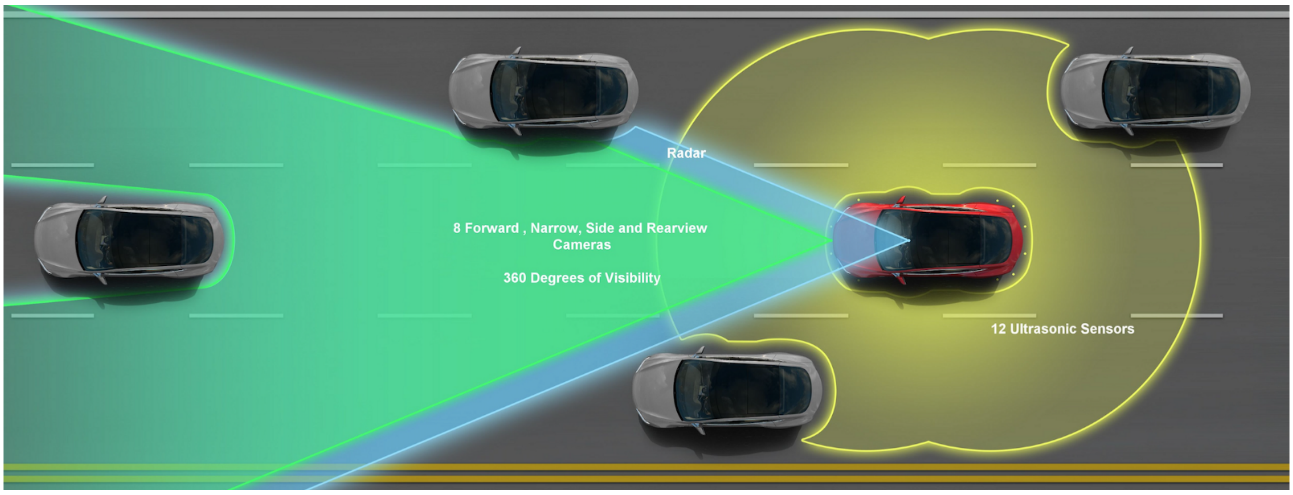

Radar
8 Forward , Narrow, Side and Rearview Cameras
360 Degrees of Visibility
12 Ultrasonic Sensors

Figure 3.3: Tesla's Current Architecture
Source: Tesla

3.1.3 Audi

Audi is the first and only car manufacturer that has introduced Level 3 autonomy in a production vehicle [26]. The 2019 A8 is equipped with Audi AI, that is capable of controlling the car on highways (like Tesla's Autopilot) and in potential traffic jams with speeds up to 60 km/h. The main difference from Tesla's Autopilot is that this system is also capable of handling unexpected events, such as accidents or other's careless driving. However, in every day city commute driver is required to have the total control of the vehicle. In Figure 3.4 Audi's sensor topology is depicted.

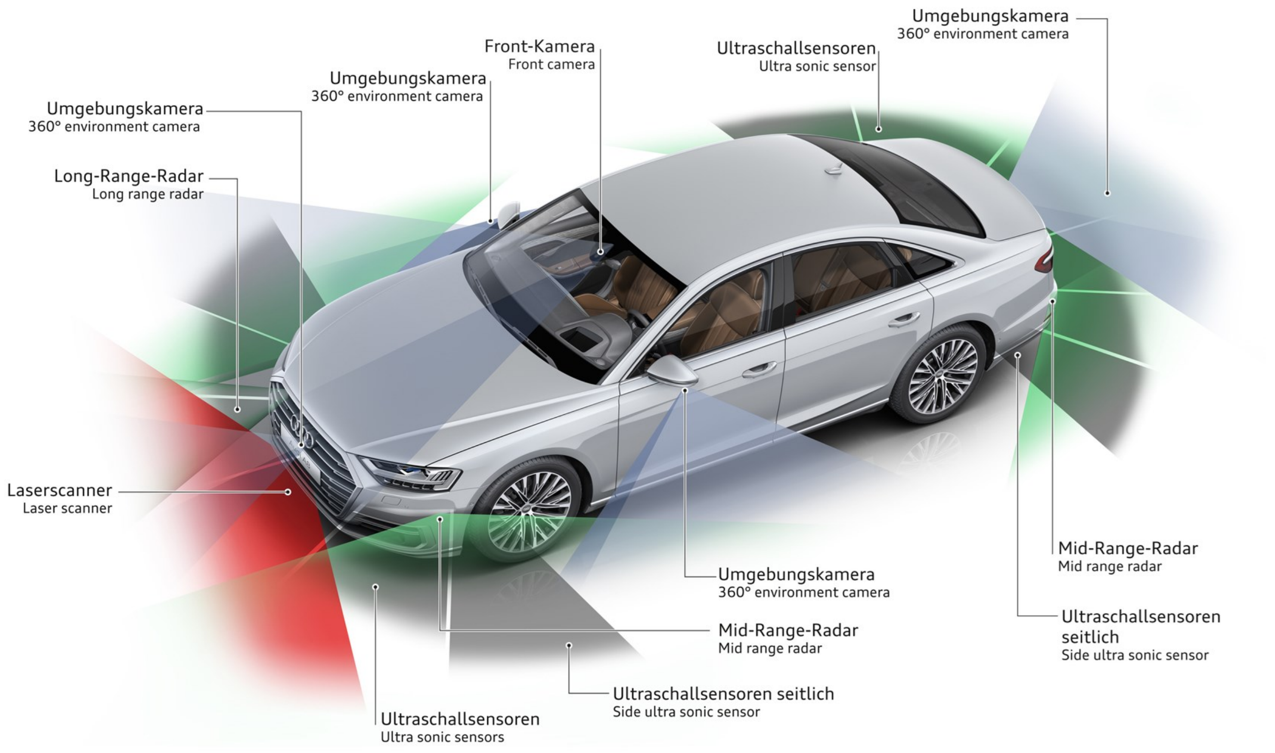


Figure 3.4: Audi's Architecture
Source: Audi

3.1.4 General Motors

General Motors has also made extensive progress in autonomous driving. They have just announced their CruiseAV, the first vehicle without steering wheel and pedals, aiming purely to autonomous driving [27]. Cruise AV is a SAE Level 4 autonomous vehicle. They are also planning to commission a fleet of autonomous taxis in large cities by 2019 for testing purposes in real conditions.

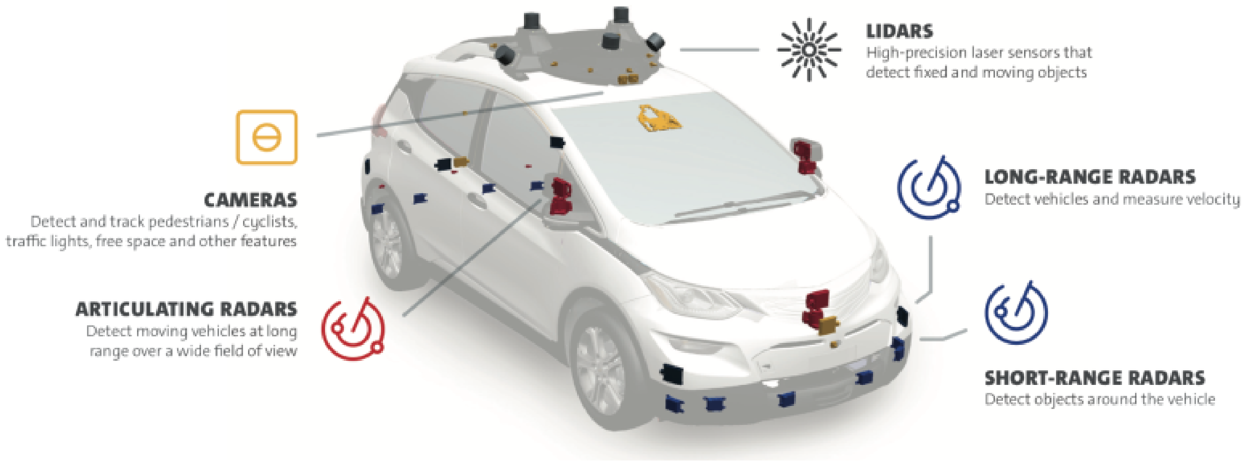

LIDARS
High-precision laser sensors that detect fixed and moving objects
CAMERAS
Detect and track pedestrians / cyclists, traffic lights, free space and other features
LONG-RANGE RADARS
Detect vehicles and measure velocity
ARTICULATING RADARS
Detect moving vehicles at long range over a wide field of view
SHORT-RANGE RADARS
Detect objects around the vehicle

Figure 3.5: General Motor's Architecture
Source: General Motors

3.1.5 Various Car Manufactures

Many more car manufactures have invested money in autonomous driving. Mercedes has introduced autonomous capabilities in their luxury models that were presented in 2017 and 2018. Japanese manufactures, such as Toyota, have conducted extensive research and experimentation in the domain of autonomous driving. Also many automotive companies have formed cooperations with electronics giants, such as the BMW-Intel-Fiat Group cooperation, or the Daimler-Bosch cooperation, due to the fact that automated driving involves such a diverse range of technologies.

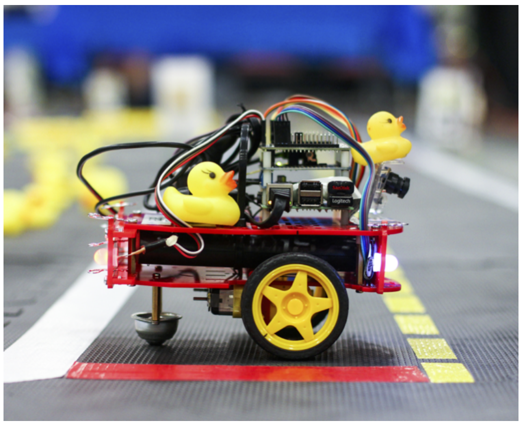
Unfortunately, since all the above regard commercial products, information about architecture and sensors is very limited and held back until the official presentation of the products.

3.2 Educational Approach to Autonomous Driving

Even though the related technology has made significant advances in the domain, the cost of developing and experimenting with autonomous cars still remains high. Apart from the standard platform cost, there is a significant cost related to the actual sensors and processing units which are essential for autonomous capabilities. The aforementioned cost makes it difficult for a platform to be adopted in the research activities of higher education institutes. There is always the option of small autonomous robotic vehicles which are adequate for understanding the basic concepts of autonomous driving or even inexpensive RC-car based autonomous car testbeds. Some of the most influential education-oriented platforms are described in the following sections:

3.2.1 Duckietown

Duckietown [28] is an open, inexpensive and flexible platform for autonomy education and research. The platform comprises small autonomous vehicles ("Duckiebots") built from off-the-shelf components, and cities ("Duckietowns") complete with roads, signs, traffic lights, obstacles, and citizens (duckies) in need of transportation. The Duckietown platform offers a wide range of functionalities at a low cost. Duckiebots sense the world with only one monocular camera and perform all processing on board with a Raspberry Pi 2, yet are able to: follow lanes while avoiding obstacles, pedestrians (duckies) and other Duckiebots, localize within a global map, navigate a city, and coordinate with other Duckiebots to avoid collisions. Duckietown is a useful tool, since educators and researchers can save money and time by not having to develop all of the necessary supporting infrastructure and capabilities [29]. All materials are available as open source, and the hope is that others in the community will adopt the platform for education and research.


(a) Side view of the Duckiebot Platform

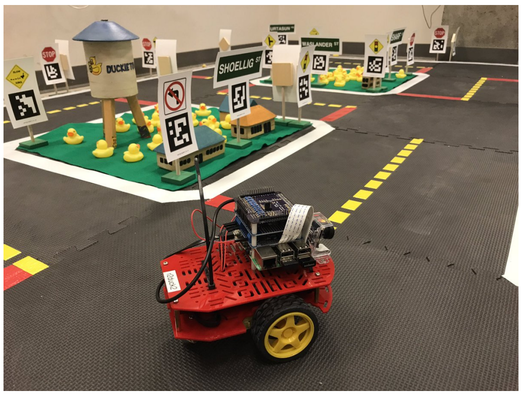

(b) Duckietown environment

Figure 3.6: Duckiebot and Duckietown
Source: Duckietown

3.2.2 Donkey Car

Donkey Car [30] is a 1/10 scale RC car converted for autonomous navigation. Donkey Car adopts a somewhat similar architecture with Duckietown using a monocular camera and a RaspberryPi. It demonstrates the power of Convolutional Neural Networks in autonomous navigation. Open-Source, easy to build and maintain and continuously evolving through the community, is a well-known and proven solution for autonomous navigation experimentation and research.

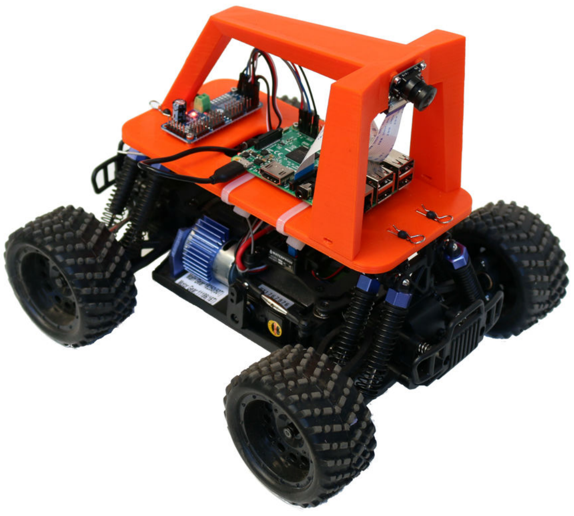


Figure 3.7: Donkey Car Platform
Source: Donkey Car

3.2.3 BARC - Berkeley Autonomous Race Car

The Berkeley Autonomous Race Car [31] is a development platform for autonomous driving to achieve complex maneuvers such as drifting, lane changes, and obstacle avoidance. A 1/10 scale RC car and an embedded Linux computer (Odroid XU4), together with a monocular camera and an IMU make up the hardware platform of the project. This project aims to be fully open-source. The data collection process is cloud-based and brings new dimensions to the Vehicle Dynamics and Control Theory teaching and research world [32].

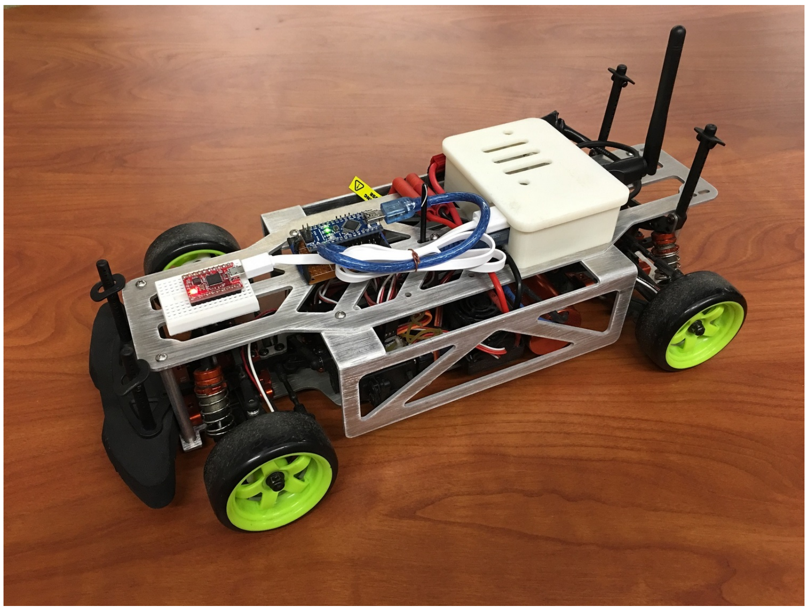


Figure 3.8: BARC Platform
Source: Barc-project.com

3.2.4 MIT Racecar

MIT Racecar [33] is a powerful platform for robotics research and teaching. Based on a 1/10 RC-car and housing state-of-the-art sensors and computing hardware is one of the most intelligent platforms. Originally build from BeaverWorks Initiative of the Lincoln Laboratory, the Department of Aeronautics & Astronautics, and the Laboratory for Information and Decision Systems at the Massachusetts Institute of Technology, the fact that it is open-source and publicly available boosted its fame and made it one of the best solutions when it comes to autonomous navigation research. MIT Racecar steps-up the sensing capabilities, bearing a stereo camera, a 2D Lidar sensor, an IMU and an odometer. Simultaneously, in order to cope with the extensive computational load, a Jetson TK1 is used on-board as the main processing unit, packing much more computational power than the RaspberryPi or Odroid used in the previously mentioned platforms.

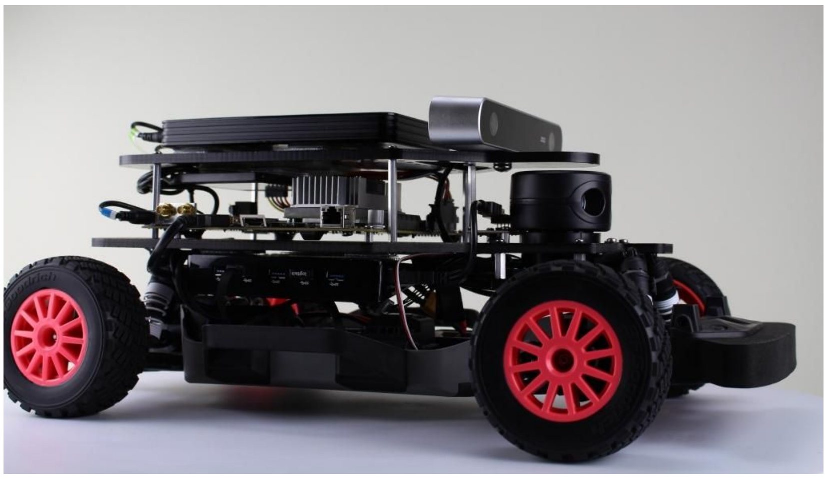


Figure 3.9: MIT Racecar Platform
Source: MIT Racecar

3.2.5 F1/10

The F1/10 autonomous platform [34] is the building block and the vehicle for educating tomorrow's engineers on the interlocking concerns of performance, control, and safety for autonomous systems, in particular autonomous vehicles. The F1/10 platform enables research in agile autonomy. The research focus here is on testing the limits of control with algorithms for path planning, aggressive maneuvers and overtaking strategies at high speeds. Pushing the car to its limits in a controlled environment, such as a racetrack is the best way to stress test the breaking points of the perception, planning, and control pipeline of the autonomous vehicle, and what it takes to get the car straightened out from such extreme situations. F1/10 also uses a stereo camera and a 2D Lidar, as well as an IMU. However, the newest Jetson TX1 is used on board, in contrary to MIT Racecar.

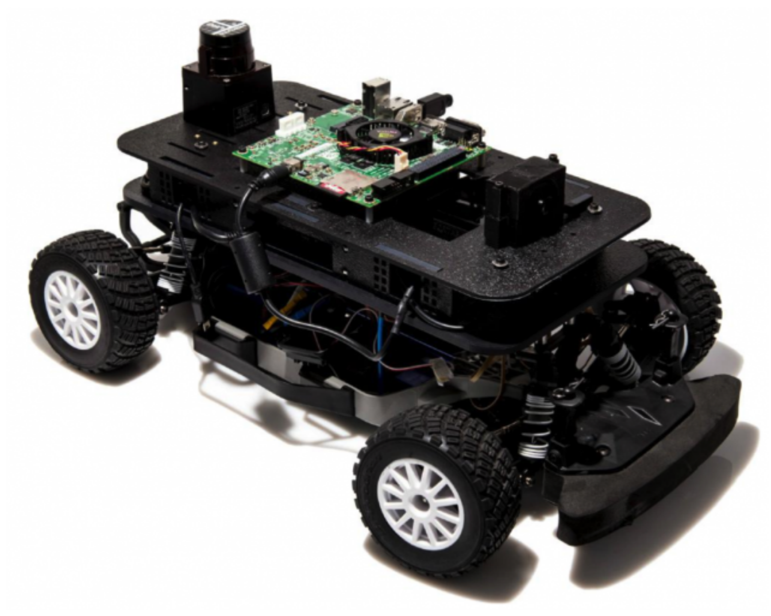


Figure 3.10: F1/10
Source: f1tenth.org

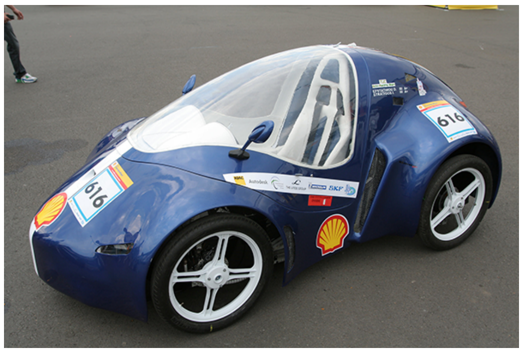
Chapter 4

Our Approach

The aforementioned platforms have all proven their abilities and robustness on research and development. We can point out here that there are several different approaches on the platforms, however we can round them up in two main categories. One that uses purely vision sensors and one that uses vision sensors and LiDAR sensors. The analogy between the commercial and the educational approach is more than obvious. However, there is a significant scale difference.

A lot of effort has already been put in autonomous vehicles, however it is easy to distinguish two main disadvantages in the above stated approaches. In the automotive industry approach the biggest disadvantage is the cost, not only for the vehicle itself, but also for the on-board sensors and the main processing units , with the cost of a 64 beam 3D-LiDAR reaching \$80000 and a processing unit up to \$10000. On the contrary, the educational approach, although it is low-cost, it lacks the resemblance of an every day vehicle. The small size makes it easy to experiment, however traditional robotics techniques, used in these vehicle are impossible to be used in reality.

Understanding the needs on the field, we selected a different approach on the educational domain. Our platform is based on an urban concept vehicle, that has previously competed in the Shell Eco Marathon Europe competition [35], with the TUCer Team [36] from Technical University of Crete (TUC). TUCer team develops and builds urban concept vehicles powered by hydrogen and competes in Shell Eco Marathon [35] since 2008. In those 10 years the team has won multiple international awards, such as four times the Safety Award, the Energy Challenge Award and the Peoples Choice Award as well as consecutive 4^{th} places in the overall consumption challenge, emerging as one of the most successful European teams and remaining until today the only Greek Institution participating in the Urban Concept category. The aforementioned platform has been in constant development and has been used as the testbed for research in automotive engineering [37], [38], [39], [40]. The specific platform used in our project, has participated in Shell Eco Marathon 2009, 2010 and 2011. Robustness, proper functionality and reliability have been proven by the challenging conditions of the competition, and one of the main reasons for selecting this vehicle as our base platform was this. In Figure 4.1, the platform competing in Shell Eco Marathon can be seen.


(a) Side view of the platform

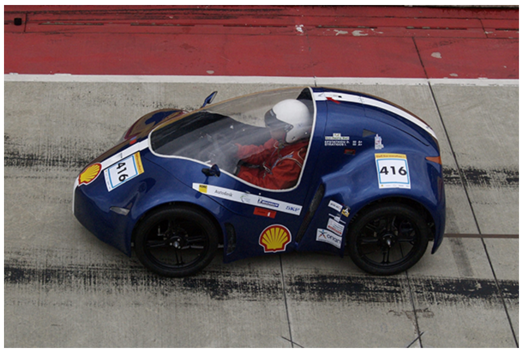

(b) Top view of the platform

Figure 4.1: The experimentation platform
Source: TUCer Team

It has been modified from a hydrogen fuel cell powered car to a battery-powered autonomous vehicle. In order to achieve this goal a series of hardware and software solutions were adopted and several devices were installed. Stepper motors have been fitted for steering and braking control as well as various sensors for perception (Stereo Camera, Lidar, Ultrasonic) and localization (GPS, IMU) along with an embedded computer, as seen in Figure 4.2. All of the above sensors are off-the-shelf components, easy to acquire and use and suitable for entry-level approach on the autonomous vehicle domain. Multiple tests to prove proper functionality have been conducted and the results are being presented.

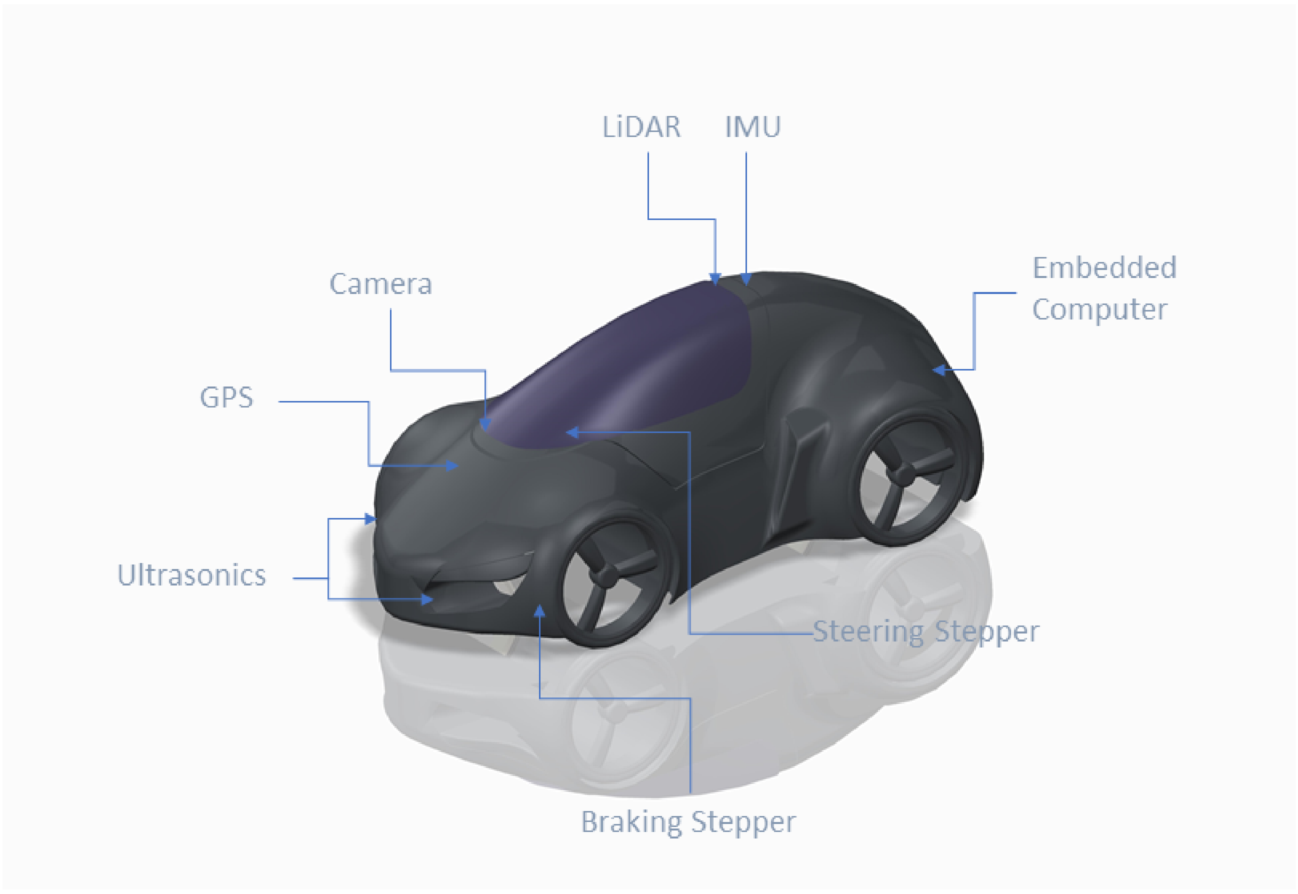


Figure 4.2: Platform Overview

Comparing our approach with the aforementioned education oriented platforms, the main advantage is the simulation of a real-life vehicle. This allows to better understand the limitations introduced in autonomous vehicles, simply by the size of the platform. Additionally, various sensors have been incorporated into the vehicle, allowing researchers to select the approach they would like to follow (pure vision or vision and LiDAR) and the extent of engagement with autonomous vehicle experimentation. In Table 4.1, a comparison between the educational platforms in use is presented. By examining this table, we can clearly see the advantages of the proposed platform.

Comparison of Educational Platforms						
Platform	Camera	LiDAR	Ultrasonic	GPS	IMU	Scale
Duckietown	✓	✗	✗	✗	✗	Radio Controlled (1:20 Scale)
Donkey Car	✓	✗	✗	✗	✗	Radio Controlled (1:10 Scale)
BARC	✓	✗	✗	✗	✓	Radio Controlled (1:10 Scale)
MIT Racecar	✓	✓	✗	✗	✓	Radio Controlled (1:10 Scale)
F1/10	✓	✓	✗	✗	✓	Radio Controlled (1:10 Scale)
TUCer	✓	✓	✓	✓	✓	Urban Mobility Vehicle

Table 4.1: Comparison of Various Educational Platforms

It is a single-seater vehicle, designed and constructed in the Machine Tools Laboratory and the Intelligent Systems and Robotics Laboratory of the School of Production Engineering and Management, TUC, build by students and professors in a voluntary effort. Its dimensions measure 2.5 × 1.25 × 1 m (L × W × H) and its curb weight is 110 kg. The chassis consists of aluminium tubing and the vehicle cover is made from carbon fiber for lightness and strength. Hydraulic bicycle disc brakes are installed on every wheel. The steering mechanism follows the Ackermann Steering Geometry [41]. It is build to simulate an urban mobility vehicle, an emerging category in the evolving vehicle market due to the rise of autonomous driving [42]. Commercial representatives of urban mobility are the Renault Twizy, the Eli Zero and the Smart ForTwo electric drive which can be seen in Figure 4.3.

(a) Renault Twizy
Source: Groupe Renault

(b) Eli Zero
Source: Eli Electric Vehicles

(c) Smart ForTwo Electric
Source: Daimler AG

Figure 4.3: Urban Mobility Vehicles

In order for the vehicle to be used as our experimentation platform, it has totally been stripped out of anything that has been on-board. From the hydrogen fuel cell, to the wiring everything was removed, in order to be rebuild in the best possible way to serve our purpose.

Chapter 5

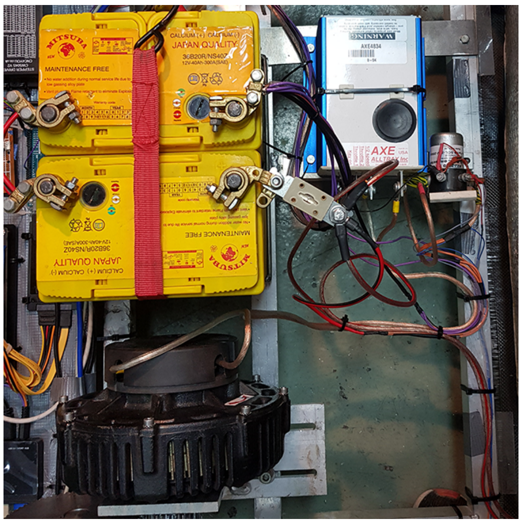
Platform Description

5.1 Platform Architecture

In this section, the architecture of the platform is described in detail, as well as the selected and installed sensors. Multiple figures accompany the description for easier and better understanding. Firstly, the mechanical aspects of the platform are described, and afterwards the sensors and the various electronic components.

5.1.1 Power Supply

In its current form the vehicle is purely electric. It has been converted from hydrogen powered by removing the hydrogen fuel cell and the super-capacitors bank that powered it before. This swap was made for simplicity and easy of use in every day tests, as hydrogen is more difficult to handle and refuel, than rechargeable batteries. The main propulsion system is supplied by 2 $12V$-$40Ah$ rechargeable lead-acid batteries (main power supply), providing adequate power for propulsion to urban mobility speeds (50 km/h) with sufficient range (for testing purposes) before recharging. Additionally, a separate battery pack (auxiliary power supply), consisting by 2 $12V$-$7.2Ah$ rechargeable lead-acid batteries supplies the on-board computer and the various sensors. The reason for the separate supplies, even though the voltage is the same (24 V), lies to the fact that the common supply of motors and electronic devices, adds substantial electromagnetic noise to the circuit disrupting the integrity of sensor readings. Furthermore, motor's inrush current or sudden load changes (acceleration or inclination) could potentially cause voltage drops disrupting the constant power application needed by the electronic devices on board the vehicle. Even though it adds complexity to the architecture, it is a common practice in robots with large DC Motors, and was adopted in our solution too. In Figure 5.1 the main and auxiliary power supplies can be seen.


(a) Main Power Supply

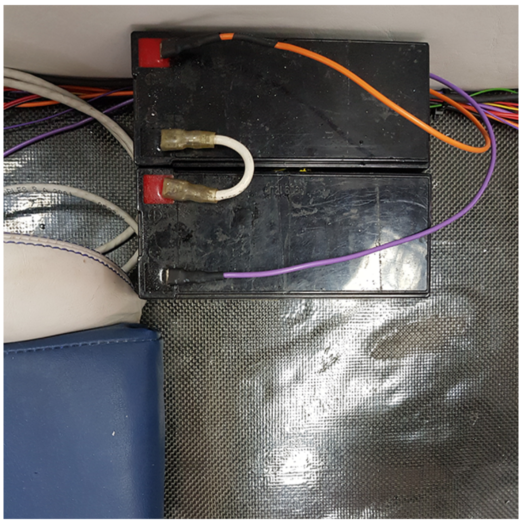

(b) Auxiliary Power Supply

Figure 5.1: Vehicle Power Supplies
Source: Google Images

The main power supply system is placed in the rear compartment of the vehicle, close to the propulsion system that powers, for easier wiring. It should be mentioned that the main power supply, except from the motor, it powers also the stepper drivers and motors that actuate the steering and braking mechanisms. Both the motor controller and the stepper drivers have wide voltage input, thus making it possible to connect directly to the main power source without the need of a DC/DC converter. The auxiliary power supply is placed in the driver's compartment for easier access and recharging. It supplies the Processing Unit and a 7-port powered 3.0 USB hub responsible for driving all the sensors on-board. In contrast to the main power supply that is connected directly to the components, for maintaining a stable DC voltage and eliminating oscillations that widely affect electronic components, between the auxiliary power supply and the target devices DC/DC converters have been used. For the CPU an XL4015 5 A Buck DC/DC converter has been selected. With a wide input from 8 V to 36 V and an adjustable output from 1.25 V to 32 V it is suitable for various applications. The selected output at 19 V is capable of delivering up to 100 W of power. In our implementation, these power levels are not required, however a future implementation where computational loads will be heavier may require maximum performance. For the powered USB hub an LM2596S 3 A Buck DC/DC converter has been selected. Similarly, it has a wide input range up to 40 V and an adjustable output from 1.25 V to 37 V. A different converter has been used because 12 V are required for the hub in contrast to the 19 V for the CPU. In Figure 5.2 the converter modules used are depicted.

(a) LM2596S Buck Converter

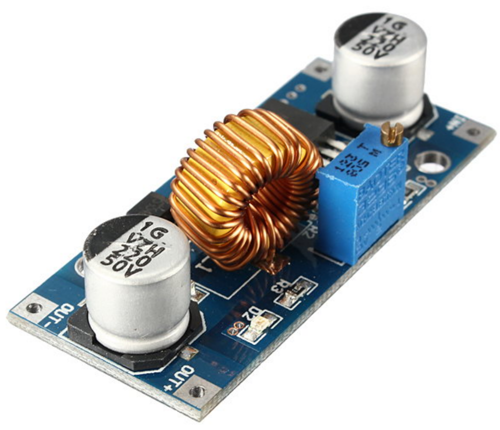

(b) XL4015 Buck Converter

Figure 5.2: DC/DC Buck Converters
Source: Google Images

A simplistic schematic of the vehicle power supply can also be seen in Figure 5.3. It is important to point out here that, although we have two different power supplies, the negative branch (ground) is common in the two systems, and thus they are not isolated from each other.

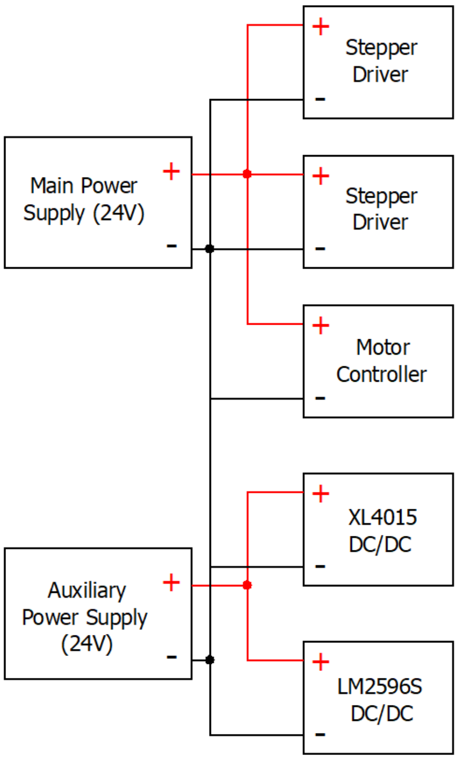


Figure 5.3: Simplistic Power Supply Schematic

5.1.2 Motor & Motor Controller

The vehicle is equipped with PGM132, a permanent magnet direct current brushed motor from Heinzmann. At 24 V it outputs 1.8 kW at 1100 rpm with rated torque of 15 Nm (peak 38 Nm) and 90 A current. This motor is capable of inputs up to 60 V, raising power to 5.1 kW adding to our platform various performance profiles. This specific motor is usually installed in golf carts, so power and torque provided is adequate for our application. The motor is controlled by an Alltrax AXE 4834 permanent magnet motor controller delivering 135 Amps rated and 300 Amps peak, more than enough for the motor paired with. The motor controller has also the ability to be programmed according to the user's specifications in terms of throttle response and throttle input. We have selected a moderate throttle response with a smooth and linear curve and an analog voltage throttle input between 0 V and 5 V, serving our control policy. The accompanying software can also provide multiple operation key values, such as $Voltage$ and $Current$ as well as throttle position and operating temperatures, very helpful for calibrating and debugging in case of unexpected failures. For instant failure observation, a led with blink codes is also installed on the controller helping to reassure proper operation or not on the fly. In Figure 5.4 the motor and the motor controller installed on this vehicle can be seen.

(a) PGM 132 PMDC Motor
Source: Heinzmann GmbH & Co. KG

(b) AXE 4834 PMDC Motor Controller
Source: Alltrax Inc

Figure 5.4: Motor and Motor Controller

5.1.3 Transmission

The rotational power of the motor is transmitted via a fixed gear ratio with one gear directly attached to the motor's rotating axle and the other to the wheel axle, without the need for a gearbox. Worth mentioning is that for simplicity in construction and maintenance, the drive wheel is only the rear right, thus eliminating the need of differentials and axles along the width of the vehicle. This set-up saves up space and weight; it was perceived and utilized for the Shell

Eco Marathon competition, therefore it is used and in our case. It usually rises the question about the car having the tendency to veer towards the drive wheel (right in our case) and being unable to maintain straight line heading, however the fact that speeds don't exceed in any case 60 km/h eliminate this possibility. In Figure 5.5 this set-up is depicted.

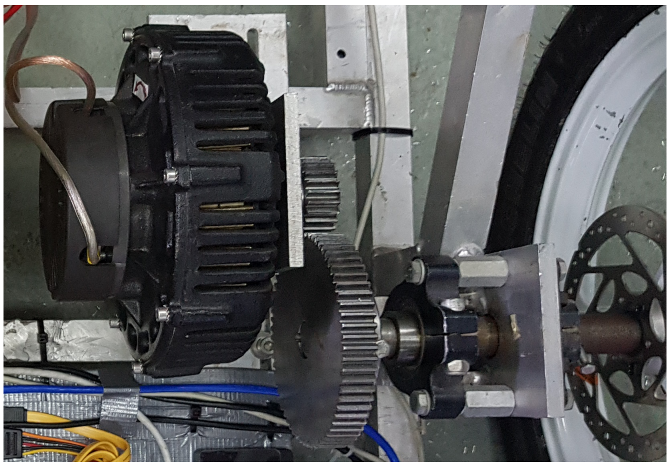


Figure 5.5: Transmission Set-Up

5.1.4 Steering and Braking System

Steering and braking control has been utilized with stepper motors, incorporated in the already existing steering and brake mechanisms of the vehicle, providing the ability to the driver to take control in a moment's notice quiting the autonomous navigation, if necessary, or not engage it at all and drive the vehicle manually. Both braking and steering steppers are powered by the main power supply, as mentioned before, at 24 V and are directly controlled by the on-board computer with appropriately developed software.

Steering Control

The manual steering mechanism on this car is a simple implementation of the Ackerman steering geometry. Ackerman steering refers to the principle that the inner wheel and the outer wheel on a vehicle need to trace out circles of different radii in order to eliminate sideways tyre slip. It is important to point out this fact as it is going to have an important effect on the software development for the steering control. Regarding the steering mechanism developed for autonomous navigation, a stepper motor is placed on the steering rack, rotating it directly to the desired steering angle. The hybrid stepper motor from Motech Motors (MT-2305HS280AW-C) with 1.26 Nm of torque and steps of 1.8 degrees is rotating the steering angle via a gearbox of fixed ratio (Figures5.6) that multiplies the torque exercised to the steering rack and divides the steps for better accuracy. It is controlled by a Wantai Microstepping Driver (DQ542MA) which is set up to further enhance the precision of the motor resulting in steps of just 0.225 degrees or 1600 steps for a 360^o rotation. In the bottom end of the steering rack, a rotary magnetic modular encoder (RM22) from RLS is attached. This encoder provides constant information about the steering's exact position and angle and is essential for the steering control procedure.

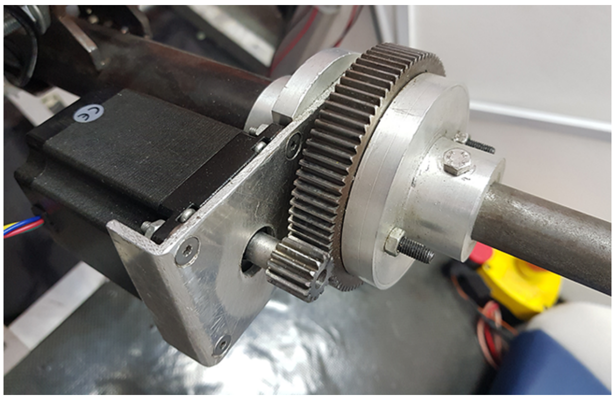


Figure 5.6: Steering Mechanism

Braking Control

This vehicle is equipped with Shimano hydraulic bicycle piston callipers and disk brakes on the front wheels. The bicycle hydraulic pumps are also used in the braking mechanism, however they are installed vertically and not horizontally as in a bicycle. The two independent pumps are linked together mechanically to be pressed as a single unit, forming the braking pedal of the vehicle. The challenge in converting this system to be controlled autonomously lies in the fact that it needs to maintain the manual operation. A pull-lever was utilized for actuating the brakes, while retaining the ability to be used as a normal foot pedal(Figure 5.7). The autonomous braking mechanism utilizes a somewhat similar design with the steering mechanism. A Nema 23 stepper motor (60BYGH401-03) with 4 Nm of torque and 1.8 degree steps actuates the brake pedal as described before. The microstepping driver (CW5045) controlling the stepper motor is set to the same level of precision (1600 steps / 360^o) for ease of programming and coherence between braking and steering steps. Braking stepper motor packs higher torque due to the needs of the lever pulling design for brake actuation and due to the fact that no gears are used to multiply the torque provided.

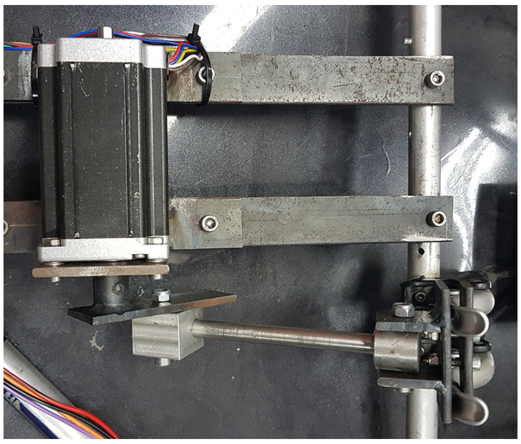


Figure 5.7: Braking Mechanism

5.1.5 Processing Unit

The central processing unit installed on the vehicle is an NVIDIA Jetson TX2 Developer Kit (Figure 5.8a). It is an embedded ARM architecture computer with exceptional Artificial Intelligence (AI) capabilities. It is equipped with a Quad ARM A57 processor plus a Dual Denver processor, 8 GB of LPDDR4 memory and 256 Cuda cores of NVIDIA's Pascal Architecture, able to deal with computational intensive processes. It supports CAN, UART, SPI, I2C, I2S protocols as well as plain GPIO. For higher level communication WLAN, Gigabit Ethernet and Bluetooth is supported together with USB 3.0 and USB 2.0 ports. Internal memory is 32 GB eMMC, but SATA connectivity and SD Card port are available too. In our setup a 240 GB Sandisk Extreme II Solid State Disk is used for speed and extended storage capabilities. Additionally, a 7-port TP-Link powered USB 3.0 hub is attached to the USB 3.0 port, for easier multiple peripherals connectivity and uninterrupted power supply, since built-in ports are able to deliver up to 900 mA of current, not enough for the attached USB devices. Powered USB hubs is a common practice when multiple devices need to operate simultaneously. Except from the USB interface for the sensors, the GPIO ports have also been used for direct control of the stepper drivers and motors from the embedded computer. The Jetson TX2 was selected for two main reasons: (i) the extended capabilities that provides for a reasonably low cost and (ii) the ability to rapidly prototype solutions and test them in a real testbed using standard tools which are widely used in the education process (i.e. Maltab, libraries provided by NVIDIA or other open-source software). In Figure 5.8b the Jetson TX2 installed on the vehicle can be seen. We can distinguish the 7-port USB hub on the right, the Sandisk SSD and the wiring for the stepper controllers on the left of the board.

(a) Jetson TX2 Processing Unit
Source: NVIDIA Corporation

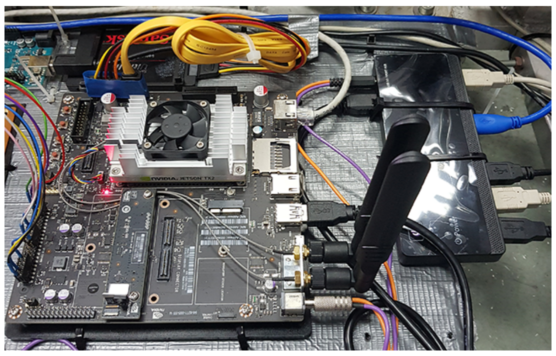

(b) Jetson TX2 and peripherals on vehicle

Figure 5.8: Nvidia Jetson TX2

5.1.6 Sensors

Sensors are the most important aspect of an autonomous vehicle. Without them any level of autonomy won't be possible to be achieved. From simple ultrasonic sensors used today in almost every car (for parking assist purposes) to sophisticated 3D Lidar costing more than the vehicle itself, every sensor plays a vital role in the pursuit of autonomy. We can distinguish two main categories of sensors that are incorporated in autonomous vehicles as seen in Table 5.1. One category is perception sensors, which are the sensors that recognize the surrounding environment of the vehicle. The other category is localization sensors, referring to sensors able to provide position and orientation.

Sensors	
Perception Sensors	Navigation Sensors
Monocular Cameras Stereo 3D Cameras Infrared Cameras Ultrasonic Sensors Laser Range Finders LiDAR Radars	GPS IMU Odometers

Table 5.1: Sensors Classification

In our platform a stereo camera, a LiDAR and a pair of ultrasonic sensors have been selected for perception. For localization respectively, a GPS module and an IMU module have been chosen. Sensor overview and installation can be seen in Figure 5.9

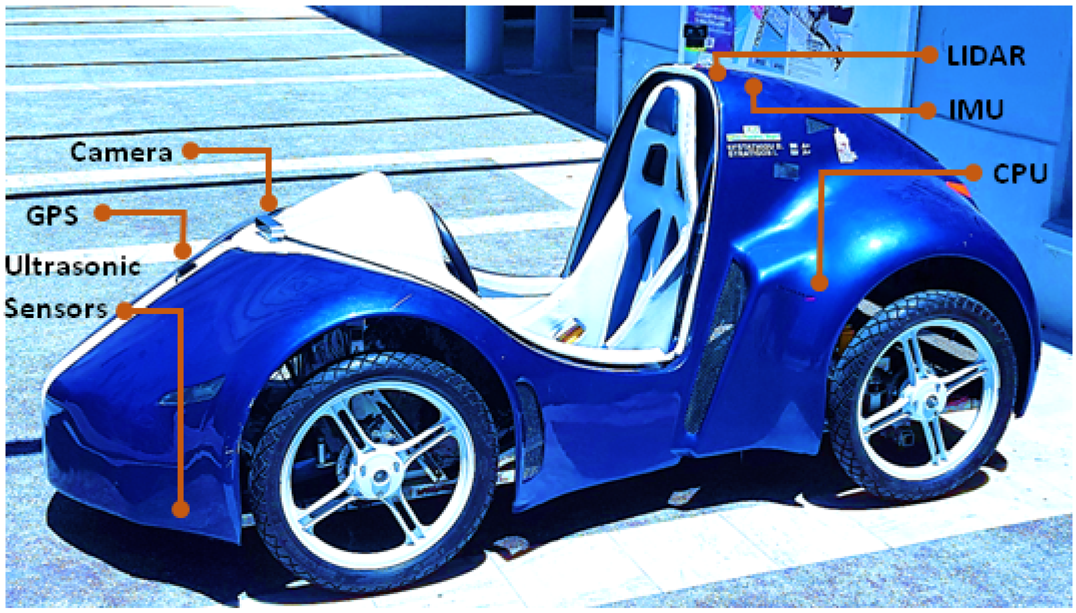


Figure 5.9: Sensors Overview

Following up is a detailed description of the sensors used in our vehicle.

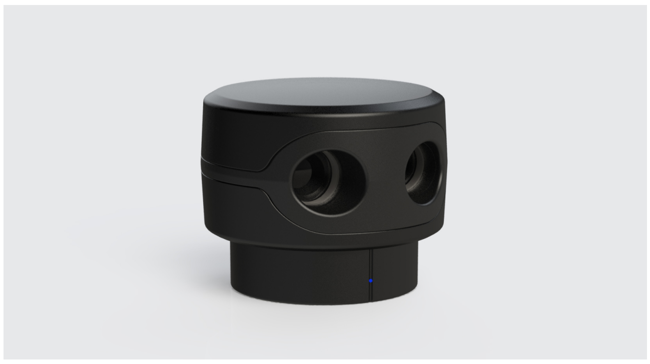
Perception - Stereo Camera

Forward mounted at the center of the vehicle, in a height of 65 *cm* there is a ZED stereo camera from Stereolabs (Fig.5.10). It consists of two 4 Megapixels cameras, with wide-angle dual lenses, field of view at 90 degrees × 60 degrees × 110 degrees (H × V ×D) and $f/2.0$ aperture. Sensors are 1/3″ with backside illumination capable of high low-light sensitivity and have a native 16:9 format for a greater horizontal field of view. USB 3.0 connectivity allows for high resolutions (up to 2.2K @ 15 FPS). Equally important are the depth recognition capabilities of the ZED camera, achieving the same resolution as the video with range from 0.5m to 20 m and the motion sensing with 6-axis Pose Accuracy (Position:± 1 *mm* and orientation: 0.1 degrees) using real-time depth-based visual odometry and SLAM (Simultaneous Localization and Mapping). Our choice was further backed by the fact that Stereolabs provide an SDK for the Jetson TX2, our main processing unit. In order to select the camera module for our platform, multiple tests have been conducted. At first a regular web camera was tested and rejected due to poor performance in outdoor environments. Our second attempt was an action camera, however this was rejected too due to the fact that action cameras output video in mp4 format which is difficult to process in real-time due to encoding. A Microsoft Kinect sensor, that has depth capabilities was tested too. Depth sensing uses an infra-red projector in the Kinect sensor, and the interference from the sun rays in outdoor conditions was rendering the readings useless. ZED Stereo Camera incorporates dual camera sensors for depth sensing, capable of performing flawlessly in both indoor and outdoor conditions making it the perfect choice for us.

Figure 5.10: ZED Stereo Camera
Source: Stereolabs Inc.

Perception - Lidar

Mounted on top of the vehicle there is a Scanse Sweep lidar (Figure 5.11a). It consists of a LIDAR-lite v3 sensor from Garmin (Figure 5.11b), in a rotating head, thus providing 360 *degrees* scans of the surroundings. It has a range of 40 *meters* and a maximum sample rate of 1000 samples per second. It scans on a single plain (2 dimension LIDAR scanning) and is suitable for Robotics and UAVs. The counterclockwise rotating head emits a beam with 12.7 *mm* diameter which expands by approximately 0.5 *degrees*. Scanse Sweep pairs it with an SDK for easy implementation to projects, as well as a visualizer app for graphical representation of the measurements. This specific lidar sensor was selected among many others since it was a great value for money. Already tested in autonomous navigation on UAVs [43] it was the perfect candidate for our application.


(a) Scanse Sweep
Source: Scanse LLC

(b) Lidar Lite v3
Source: Garmin Ltd.

Figure 5.11: LiDAR Sensor

Perception - Ultrasonic Sensors

Forward mounted on the vehicle's body at 40 *cm* from the ground, there are two Maxbotix MB1010 Lv-MaxSonar-EZ1 ultrasonic sensors. With a range from 15 *cm* to 645 *cm* and a refresh rate of 20 Hz, they are capable of accurately detect and measure the distance from objects in front of the vehicle. They have multiple connectivity protocols such as analog, pulse width and serial output for easier sensor integration to different platforms. Due to cross-talk issues between the two sensors, a chaining between them is required, triggering sensors one after another eliminating the interference on the measurements of each individual sensor. The ultrasonic sensors serve the purpose of detecting objects in front of the car that pose an immediate threat for collision.

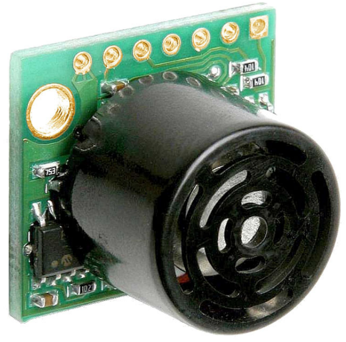


Figure 5.12: Maxbotix Ultrasonic Sensors
Source: Maxbotix Inc.

Both the Stereo Camera and the LiDAR are connected through the USB hub directly to Jetson TX2. For the ultrasonic sensors, the analog output was selected to be used, since it required the least amount of wiring. Since Jetson TX2 has no analog inputs, an Arduino Mega, one of the most popular microcontrollers, is used as middleware between the sensors and the processing unit improving input-output capabilities of the system. An overview of the perception sensors sub-system can be seen in Fig.5.13.

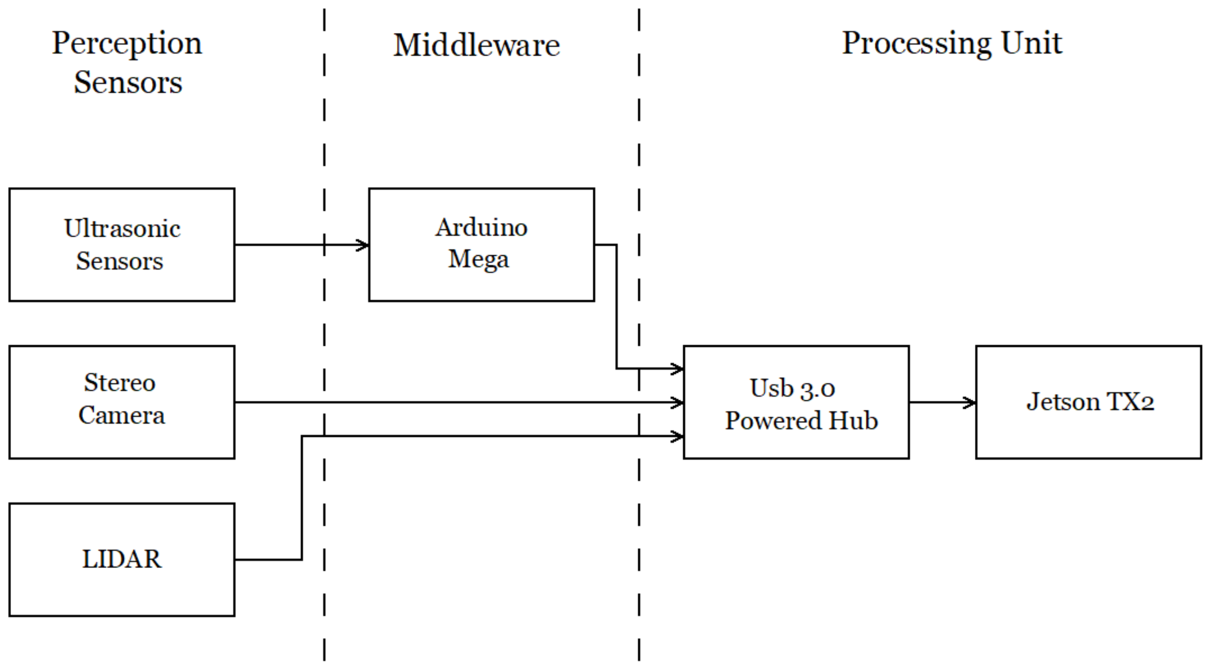


Figure 5.13: Perception Sensors Subsystem

Localization - Global Positioning System (GPS)

For localization purposes an Adafruit Ultimate GPS Version 3 is used (Figure 5.14). Capable of using up to 66 channels and having -165 dBm sensitivity, the achieved accuracy is adequate to position our vehicle in the real world for navigation purposes. For more robustness and better satellite reception, an external GPS antenna is used (instead of the built-in) with the u.FL connector provided in this board. The serial output of the GPS sensor, similarly to the ultrasonic sensors, uses the same Arduino Mega mentioned before as a middleware, which converts the raw data from the GPS module into readable format and then passes them on to the Jetson TX2.

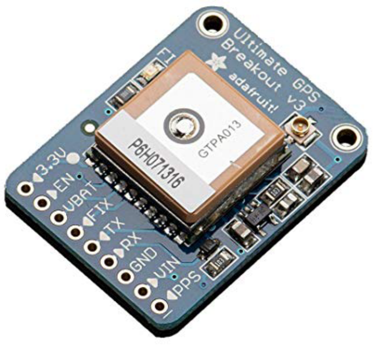


Figure 5.14: Adafruit GPS Module
Source: Adafruit

Localization - Inertia Measurement Unit (IMU)

For better navigation, in tandem with the GPS, an IMU module is also used. The Sparkfun Razor IMU is a 9 Degrees-of-Freedom sensor, able to measure the absolute orientation, angular and linear acceleration, gravity vector and magnetic field strength. Absolute orientation, i.e. the current heading of the vehicle relative to the magnetic north, is the most important value for our application, as it is the key value for the calculation of the steering angle. The Razor IMU, connects directly to the Jetson TX2 via USB without the need of a middleware unlike the GPS module, since it has an embedded microcontroller that is programmed according to user's needs.

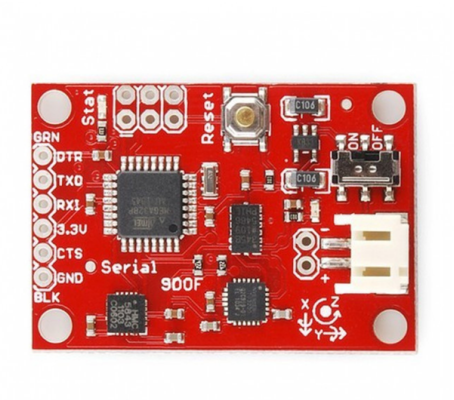


Figure 5.15: SparkFun Razor IMU Module
Source: SparkFun Electronics

In Figure 5.16 the architecture of the localization sensors sub-system implemented on the platform can be seen.

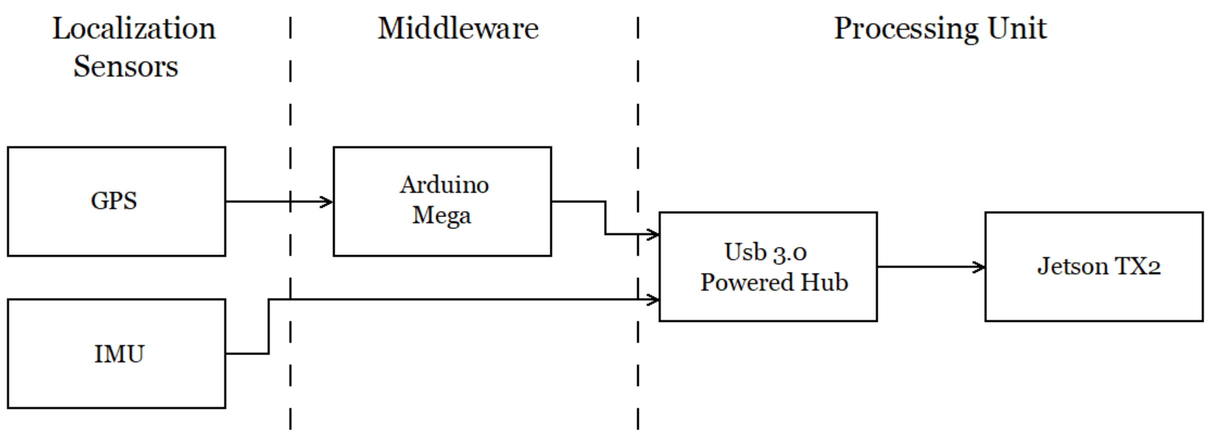


Figure 5.16: Localization Sensors Subsystem

5.1.7 Various Electronic Components

In addition to all specified sensors above, there are some extra hardware components used on the car for easier implementation of the needed circuits.

Arduino Mega

As mentioned before an Arduino Mega is used as a middleware between the sensors and the Jetson TX2. This is due to the fact that the Arduino microcontroller adds serial port and analog signal inputs needed for the GPS and the Ultrasonic sensors. The Arduino Mega 2560 Rev.3 is designed for complex projects with 54 I/O and 16 analog inputs. Not only the ease of Arduino programming, but also the fact that Adafruit and Maxbotix offer drivers and schematics for using their sensors with Arduino, help speed up the integration of the sensors to our platform. Communication with the Jetson TX2 is done via the USB interface.

Arduino Nano

Also an Arduino Nano is used in the vehicle. This is dedicated for the steering wheel angle sensor. The Nano is a more compact Arduino board with smaller I/O capabilities from the Mega, but powerful enough to process the incoming data from the magnetic shaft encoder. A separate Arduino board is used and not the Mega, due to the fact that the steering wheel angle readings occur in different frequencies than the GPS and the Ultrasonics. Connection to the Jetson TX2 is again made possible over USB interface.

Magnetic Shaft Encoder

The magnetic shaft encoder from RLS used for tracking the steering wheel angle, is the RM22, an extremely accurate sensor, reading steps of 0.35 degrees with ±0.5 degrees accuracy. Our variation has a linear voltage output with the ability to measure 360 degrees of rotation at 30000 *rpm*. Connectivity to the Jetson TX2 occurs via the Arduino Nano, as previously described.

Digital to Analog Converter

An MCP4725 digital to analog converter from Adafruit is used as throttle control. With 12-bit analysis and I2C connectivity is directly controlled from Jetson TX2. It receives a digital input from 0 to 255 (12 bit) and converts it in the appropriate DC voltage between 0 and 5 Volts to control the throttle. The Alltrax controller is set to receive analog throttle input between 0 and 5 volts as described earlier.

Logic Level Converter

Two Sparkfun Logic Level Converters are also used in the setup. These small breakout boards are capable of level shifting, i.e. to change the logic level of true value (1) from 3.3 V to 5 V and the opposite. The need to use them derives from the fact that Jetson TX2 GPIOs operate at 3.3 V where the stepper motor controllers need 5 V signals and since the stepper control signals are generated from Jetson TX2, this simple board makes communication and control possible.

All of the above sensors and components have been properly positioned into the vehicle to serve the intended purpose in the best possible way. Wiring has been laid out from scratch as well as prototyping boards for the electronic circuits and the sensors have been constructed. All the safety measures have been taken, with fuses installed in the main and auxiliary power supplies and safety push buttons for instantaneous immobilisation of the car. In Figure 5.17 an overview of the system can be seen while in Figure 5.18 some of the prototyping boards constructed for this platform.

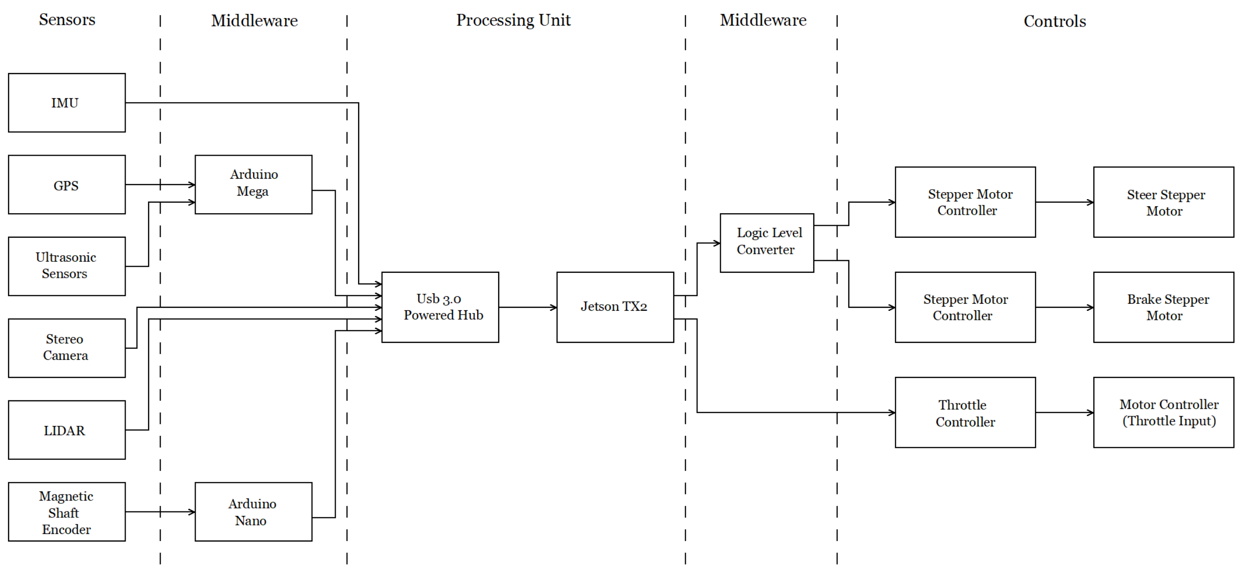


Figure 5.17: Autonomous Navigation System Overview

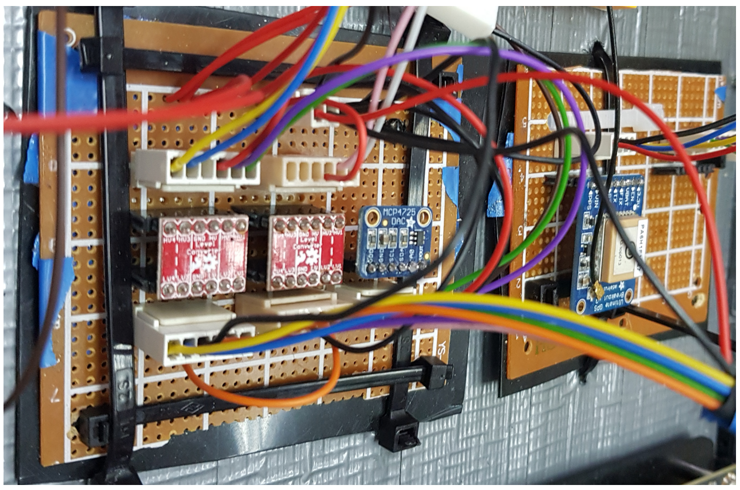


Figure 5.18: Hand-Soldered Prototyping Boards

5.2 Software Development

Equally important with the sensors and the platform architecture described above is the software development for the processing of the sensors input and the control of the vehicle. The software development procedure for this platform can be distinguished into two major classes, the embedded code development and the external code development. The first refers to the development of programs and functions on-board the Jetson TX2. The language of choice for this is *C++*, as it is one of the most popular for embedded systems. The external code development is performed using Matlab on a remote device and downloaded directly into the board.

5.2.1 Embedded Code Development

All the main modules for vehicle control and sensors logging have been developed in *C++*. One of the main reasons behind this choice was the Software Development Kits provided for the LiDAR and Camera. Also, Jetson's GPIO and I2C manipulation libraries have been developed in *C++*, making it the easiest way for faster and reliable code development. In addition execution speed (due to pre-compilation of the programs), wide support and direct use (without dedicated compilers and developer's tool installation) in Linux environments make it even more suitable. Besides, *C++* ranks second in IEEE Spectrum Top Programming Ranking [44] as the language of choice for embedded systems, only behind the *Python* language. Multiple third-party libraries have been also used for serial communication between the Arduinos, the sensors and the main program, as well as keyboard input.

5.2.2 External Code Development

The external code development is performed using Matlab and by adopting the GPU Coder tool, installed in the latest versions, we have the ability to generate optimized CUDA code from MATLAB code for deep learning, embedded vision, and autonomous navigation. The generated code is portable across NVIDIA GPUs and can be compiled and executed on an NVIDIA Jetson platform (Fig.5.19). This approach is user friendly since CUDA and GPU programming are much more demanding tasks than Matlab coding. This approach allows to exploit the maximum potentials of our processing unit (GPU applications) in the least amount of time. With Matlab also the data acquired from experimentation can be easily extracted and used for post processing analysis and further development. In addition, simulations for the vehicle control procedure can be evaluated before tested in real world conditions, eliminating errors and bugs in the developed software.

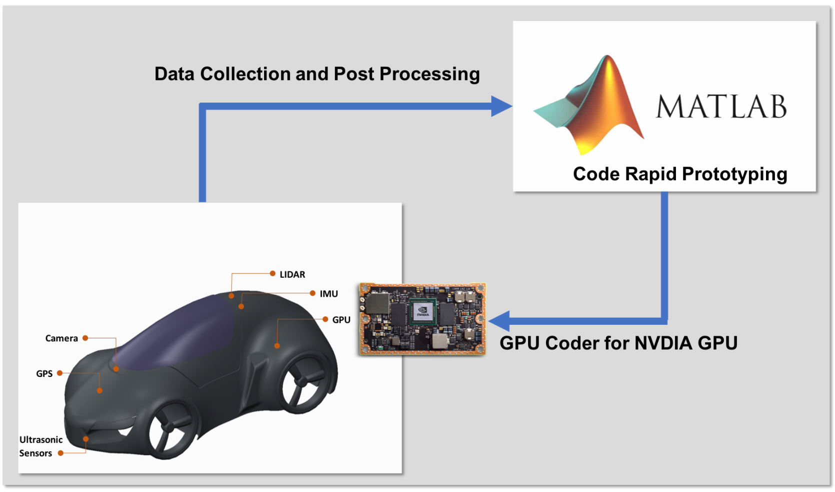


Figure 5.19: External Code Development Cycle

The control procedure of the vehicle as well as the GPU code used for pedestrian detection are going to be described in detail in the next chapter. What is important to point out here is the fact, that during the development phase, many different sub-programs have been developed and tested. Theses sub-programs have been put together, with minor or major tweaks, to form the main control program of the vehicle.

Chapter 6

Implementation and Experimentation

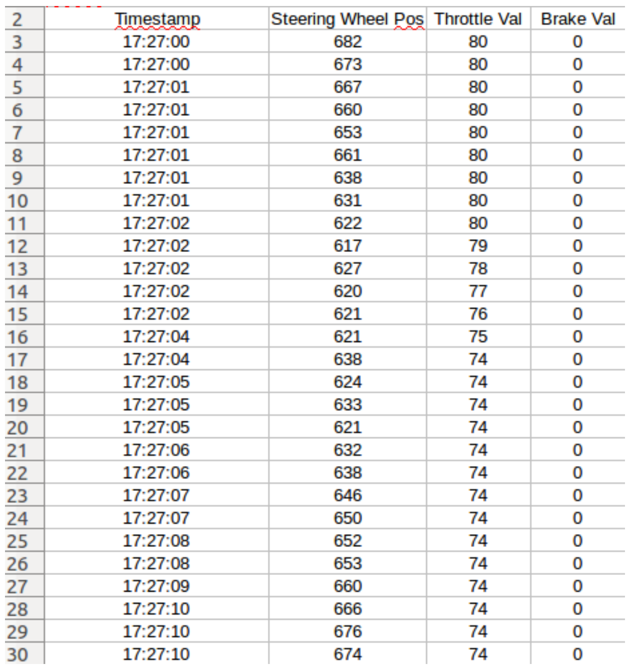
Platform construction and software development procedures have been described in detail in the previous chapter. However, the target of this project isn't only the construction of the platform. In this chapter we are going to discuss the steps that our development effort followed to reach the desired target of the autonomous navigation.

6.1 Preliminary Testing - Platform Validation

The first target for the platform in development was to prove proper functionality and robustness. Due to the complexity of autonomous navigation, the first experiment was targeting to define that steering, throttle and braking mechanisms are properly working as well as that sensor logging is successful and with the appropriate frequency. For ease of testing, the car was going to be controlled by keyboard, simulating a full-scale RC. Multiple programs have been developed for this experiment. Since it was a preliminary approach every module attached to the car required an extensive development time. At last, four different programs, running simultaneously, have been used in the final experiment. One that controls steering, speed and braking, one that logs sensor data (GPS, IMU, Ultrasonics), one that logs the LIDAR data and one that records the video stream from the camera.

Vehicle Control Software

The first part of the software development was concerning the control of the vehicle. The vehicle is controlled via keyboard commands send to the on-board computer from an external PC, either connected via ethernet cable or Wi-Fi network. The program constantly monitors the keys pressed and depending on the user's input it responds accordingly. A game-control logic was used, with W being the throttle increase command, S the throttle decrease command, A the left steer and D the right steer. C key was used to return the wheel to the center point, while B was the brake increase , V the brake decrease and P the parking brake (fully applied brakes). Lastly, Q was used for program quit and exit from vehicle control loop. As mentioned earlier, throttle is controlled by MCP4725 Digital-to-Analog converter. The communication between Jetson TX2 and MCP4725 uses the I2C protocol. With the help of libraries developed fot Jetson TX2 board, the C++ programm controls the connected board and writes a digital value between 0 and 255 to its input buffer. Digital to Analog converter translates this value into the appropriate voltage in order to control the vehicle. Similarly, the Stepper Drivers for steering and braking are controlled by the Jetson TX2. Libraries for controlling Jetson TX2 GPIO ports are used just like the I2C libraries. Steppers require two signals for control, one that sets its direction and one pulse that drives the motor for one step. The frequency of the pulses control the speed of the stepper, and therefore the response of steering and braking. This software except for control, it records in a `.csv` file with explicit timestamps every change that occurs in steering angle, throttle and braking position, allowing to later process and evaluate the response of these systems.



2	Timestamp	Steering Wheel Pos	Throttle Val	Brake Val
3	17:27:00	682	80	0
4	17:27:00	673	80	0
5	17:27:01	667	80	0
6	17:27:01	660	80	0
7	17:27:01	653	80	0
8	17:27:01	661	80	0
9	17:27:01	638	80	0
10	17:27:01	631	80	0
11	17:27:02	622	80	0
12	17:27:02	617	79	0
13	17:27:02	627	78	0
14	17:27:02	620	77	0
15	17:27:02	621	76	0
16	17:27:04	621	75	0
17	17:27:04	638	74	0
18	17:27:05	624	74	0
19	17:27:05	633	74	0
20	17:27:05	621	74	0
21	17:27:06	632	74	0
22	17:27:06	638	74	0
23	17:27:07	646	74	0
24	17:27:07	650	74	0
25	17:27:08	652	74	0
26	17:27:08	653	74	0
27	17:27:09	660	74	0
28	17:27:10	666	74	0
29	17:27:10	676	74	0
30	17:27:10	674	74	0

Figure 6.1: Instance of the Control Logging `.csv` file

In Figure 6.1 an instance of the logged control data can be seen. Steering wheel position, throttle value and brake value are being recorded. Regarding steering value, in order to be translated in degrees we have to use the following conversion

$$(SteeringValue - 690) \times 0.35$$

where 690 is the value when the steering wheel is centered and 0.35 the degrees per step of the encoder. Positive values refer to right steering, while negative values to left steering. Throttle value, as mentioned earlier, ranges between 0 and 254 so for easier comprehension the following conversion

$$ThrottleValue \times 0.3937$$

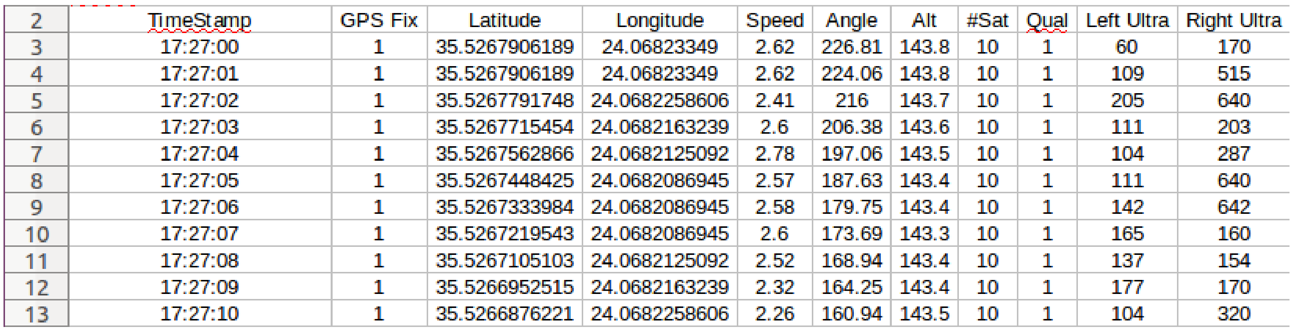
returns the value in a percentage from 0 to 100. Similarly for the brake value the conversion

$$BrakeValue \times 0.3125$$

returns the brake value between a 0 to 100 percentage. It is also obvious that every second has more than one recordings. This is due to the fact that steering, throttle and brake adjustments may occur many times in a single second, and one of the main reasons that control and sensors are being logged in different files.

Sensors Logging Software

The above software was responsible for vehicle control. However equally important with controls is sensor logging. This program, also developed in *C++* is responsible for reading sensor input and recording data in a `.csv` file. Communication between sensors and the program was achieved via the serial communication protocol. An external library that handles serial communication was used and through this program data requests have been transferred to the sensors and responses are being received. The ease of file operations in *C++* helped for logging into a file for easier post processing. Timestamps have been attached in every separate set of data and the frequency was set at $1Hz$, since the GPS module refreshes once every second. The advantage of this approach is that with simple software modifications only the important data can be transferred from the sensors to the main program, relieving computational sources that may be needed elsewhere.



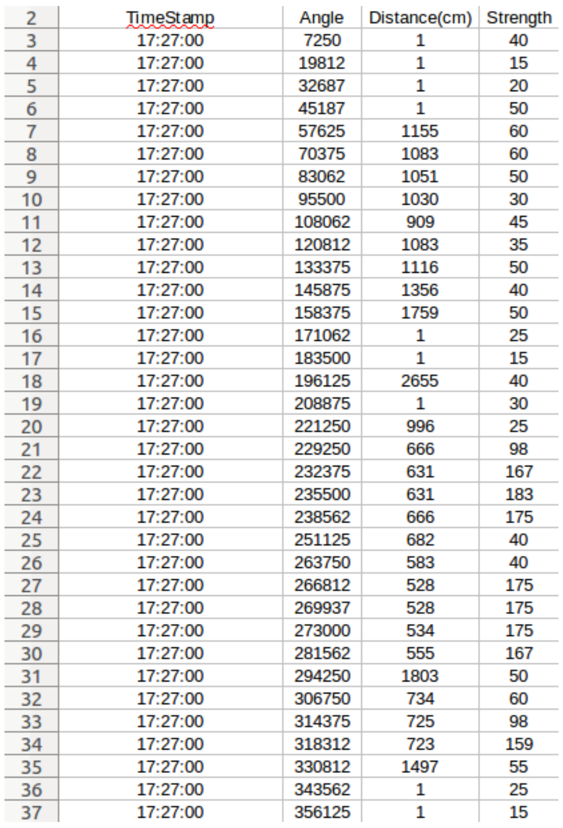
2	TimeStamp	GPS Fix	Latitude	Longitude	Speed	Angle	Alt	#Sat	Qual	Left Ultra	Right Ultra
3	17:27:00	1	35.5267906189	24.06823349	2.62	226.81	143.8	10	1	60	170
4	17:27:01	1	35.5267906189	24.06823349	2.62	224.06	143.8	10	1	109	515
5	17:27:02	1	35.5267791748	24.0682258606	2.41	216	143.7	10	1	205	640
6	17:27:03	1	35.5267715454	24.0682163239	2.6	206.38	143.6	10	1	111	203
7	17:27:04	1	35.5267562866	24.0682125092	2.78	197.06	143.5	10	1	104	287
8	17:27:05	1	35.5267448425	24.0682086945	2.57	187.63	143.4	10	1	111	640
9	17:27:06	1	35.5267333984	24.0682086945	2.58	179.75	143.4	10	1	142	642
10	17:27:07	1	35.5267219543	24.0682086945	2.6	173.69	143.3	10	1	165	160
11	17:27:08	1	35.5267105103	24.0682125092	2.52	168.94	143.4	10	1	137	154
12	17:27:09	1	35.5266952515	24.0682163239	2.32	164.25	143.4	10	1	177	170
13	17:27:10	1	35.5266876221	24.0682258606	2.26	160.94	143.5	10	1	104	320

Figure 6.2: Instance of the Sensors Logging `.csv` file

In Figure 6.2 an instance of the logged sensor data can be seen. Except for the data used for navigation, such as Latitude, Longitude and Heading and the left and right ultrasonic sensors readings, speed and altimeter is also recorded from the GPS data. Also, for easier evaluation of data integrity and accuracy, a GPS Fix value is stored (0 when no fix is achieved, 1 when fix is ok) along with the number of satellites connected and the signal quality.

Lidar Logging Software

LiDAR logging software is distinct from the other sensors. Although it implements the same mechanism with the serial communication protocol, the difference frequency between the GPS and the LiDAR ($5Hz$) dictated the use of separate software for fully exploiting the potential of the sensor. The same principle was followed in the logging procedure with the data recorded into a texttt.csv with timestamps on every different occurrence. However, the post processing of this file was more challenging, since a single LiDAR recording consists of multiple point in a 360^o circle.



2	TimeStamp	Angle	Distance(cm)	Strength
3	17:27:00	7250	1	40
4	17:27:00	19812	1	15
5	17:27:00	32687	1	20
6	17:27:00	45187	1	50
7	17:27:00	57625	1155	60
8	17:27:00	70375	1083	60
9	17:27:00	83062	1051	50
10	17:27:00	95500	1030	30
11	17:27:00	108062	909	45
12	17:27:00	120812	1083	35
13	17:27:00	133375	1116	50
14	17:27:00	145875	1356	40
15	17:27:00	158375	1759	50
16	17:27:00	171062	1	25
17	17:27:00	183500	1	15
18	17:27:00	196125	2655	40
19	17:27:00	208875	1	30
20	17:27:00	221250	996	25
21	17:27:00	229250	666	98
22	17:27:00	232375	631	167
23	17:27:00	235500	631	183
24	17:27:00	238562	666	175
25	17:27:00	251125	682	40
26	17:27:00	263750	583	40
27	17:27:00	266812	528	175
28	17:27:00	269937	528	175
29	17:27:00	273000	534	175
30	17:27:00	281562	555	167
31	17:27:00	294250	1803	50
32	17:27:00	306750	734	60
33	17:27:00	314375	725	98
34	17:27:00	318312	723	159
35	17:27:00	330812	1497	55
36	17:27:00	343562	1	25
37	17:27:00	356125	1	15

Figure 6.3: Instance of the LiDAR Logging `.csv` file

Since LiDAR data consists of multiple lines as mentioned before, only an instance of a second was selected to be presented. Angle values represent the degrees from the LiDAR center point, in counterclockwise rotation, of the current reading. Distance is the position of the detected object from the center point in centimeters. Value 1 that can be found in some cases depict an error in reading. Strength is a pointer of how accurate are the readings and is usually taken into account in order to reject very weak readings (< 125) that usually tend to be wrong.

Camera Recording

For video recording, rather than developing a software, an already existing solution was used. Gstreamer [45] was chosen for the ease of use, since a simple command with some variables for video formatting, video resolution and saving path, are enough for the recording. An *mp*4 video format was chosen, for optimal compression and quality simultaneously. The camera resolution was selected to be at Full High Definition (1920 × 1080 pixels). Also a timestamp was inserted in the top left corner of the captured video, in order to be possible to relate video frames with the recorded `.csv` files mentioned before.

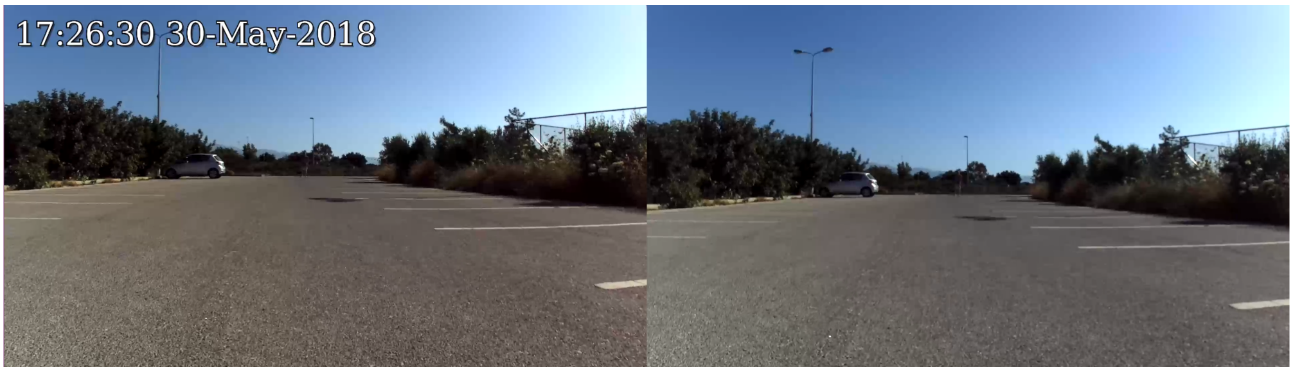


Figure 6.4: Screenshot from the camera recording

As shown in Figure 6.4 the recorded video from the stereo camera consist of two side by side videos of each camera lens. On the top left corner the timelapse overlay on the video can be seen.

After the development of the above software, the platform was tested in real conditions in the parking lot of the Technical University of Crete. A human driver was at all times on board, ready to take control in an unexpected event for safety reasons. The vehicle performed flawlessly over extended periods of experimentation and testing, proving robust and functional for the leap to autonomy. The logged data have been post processed for validation of the data recorded and minor adjustments to the software have been made for improved functionality and speed.

6.2 GPS Waypoints Path Following

After proving the proper functionality of the constructed platform and having established that every sensor is working properly, the next step was to implement a basic level of autonomy. Initially we developed the ability of the vehicle to move in a predefined path based on the readings from the GPS sensor, emulating the operation of a real autonomous vehicle in a urban environment which has to operate and navigate in an unknown area using the GPS coordinates.

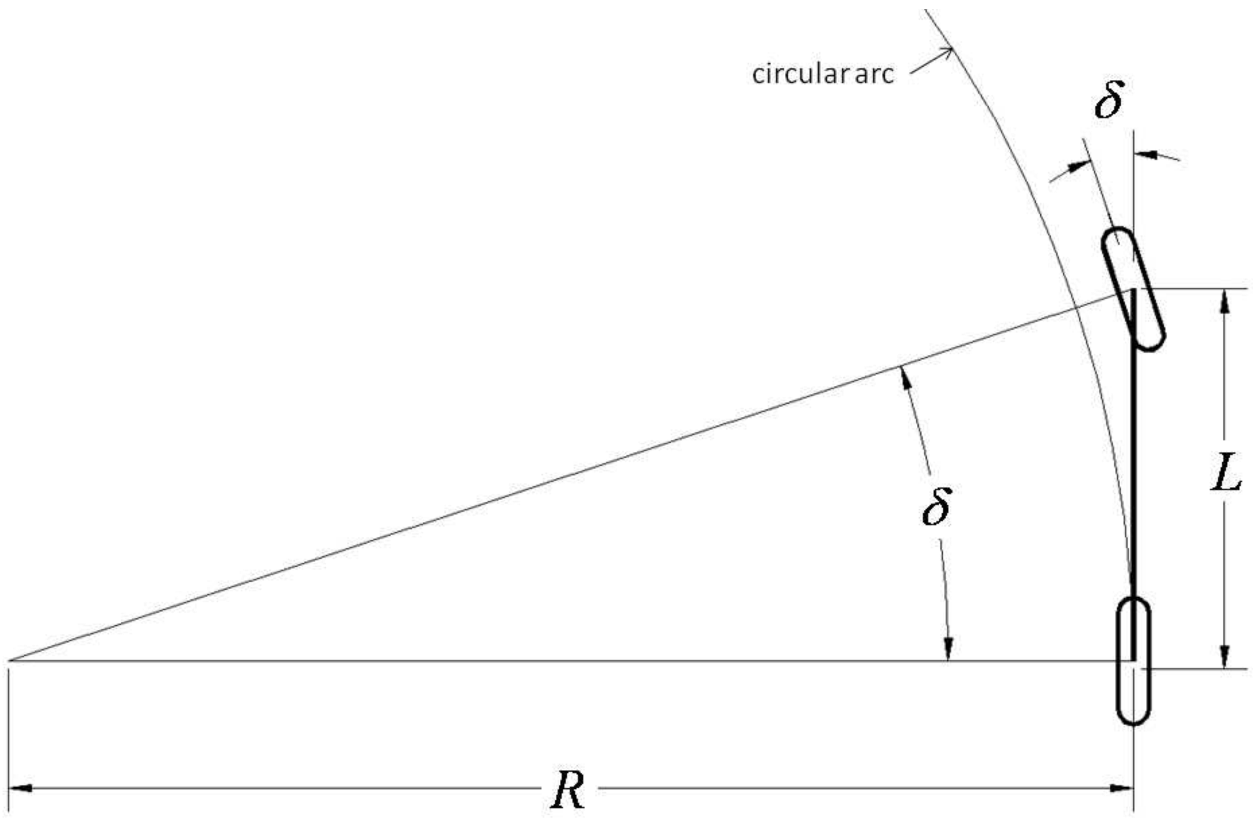


Figure 6.5: Pure Pursuit Geometry
Source: Carnegie Mellon University

In order to develop the appropriate software some simplifications have been accepted regarding the vehicle model. Our vehicle, as mentioned before, incorporates the Ackerman Steering Mechanism. However, this specific steering model poses significant difficulties to the control procedures. So, a simplification of an Ackerman steered vehicle commonly known as the bicycle model was used. With that in mind, path following was based on the pure pursuit method [46], a simplistic but robust and accurate method. The pure pursuit method and variations of it are among the most common approaches to the path tracking problem for mobile robots. The pure pursuit method consists of geometrically calculating the curvature of a circular arc that connects the rear axle location to a goal point on the path ahead of the vehicle. The goal point is determined from a look-ahead distance l_d from the current rear axle position to the desired path. The goal point (g_x,g_y) is illustrated in Figure 6.5. The vehicle's steering angle δ can be determined using only the goal point location and the angle α between the vehicle's heading vector and the look-ahead vector. Applying the law of sines to Figure 6.5 results in

$$\frac{l_d}{\sin(2\alpha)} = \frac{R}{\sin\left(\frac{\pi}{2} - \alpha\right)}$$

$$\frac{l_d}{2\sin(\alpha)\cos(\alpha)} = \frac{R}{\cos(\alpha)}$$

$$\frac{l_d}{\sin(\alpha)} = 2R$$

or

$$\kappa = \frac{2\sin(\alpha)}{l_d} \tag{6.1}$$

where κ is the curvature of the circular arc. Using the simple geometic bicycle model of an Ackermann steered vehicle, the steering angle can be written as

$$\delta = \tan^{-1}(\kappa L) \tag{6.2}$$

Using Eq.6.1 and 6.2, the pure pursuit control law is given as

$$\delta(t) = \tan^{-1}\left(\frac{2L\sin(\alpha(t))}{l_d}\right)$$

meaning that in every given time (t), the steering angle on an Ackerman Steered Vehicle is depending on the angle between the steering vector and the look-ahead vector (α) and the look-ahead distance (l_d).

For the calculation of the look-ahead vector angle the function $atan2$ is used, which returns the four-quadrant inverse tangent of y, x. The function $atan2(y, x)$ is defined as the angle in the Euclidean plane, given in radians, between the positive x-axis and the ray to the point $(x, y) \neq (0, 0)$. The angles are signed, with counter-clockwise angles being positive, and clockwise being negative. In other words, $atan2(y, x)$ is in the interval $[0, \pi]$ when $y \geq 0$, and in $(-\pi, 0)$ when $y < 0$. The y and x values are derived from the subtraction of the current GPS coordinates (Longitude, Latitude) from the target coordinates. The steering vector angle is obtained directly from the IMU sensor on-board the vehicle. Subtracting those two values, α angle is calculated. The look ahead distance is calculated by the help of the distance formula deriving from the Pythagorean Theorem where

$$l_d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

with (x_1, y_1) being the current and $(x_2, y_2$ the target latitude and longitude respectively, converted from GPS format (in degrees) to a Cartesian 2-D plain with reference point $(0, 0)$ located where the equator crosses the prime meridian. The conversion formula is

$$x = rx$$

where x is in radians and r is the average earth radius at $6371000km$. This conversion assumes that earth is a perfect sphere and uses an average radius and not the exact radius of the current position. Although these simplifications inserts a calculation error, it is negligible for our application and is therefore ignored.

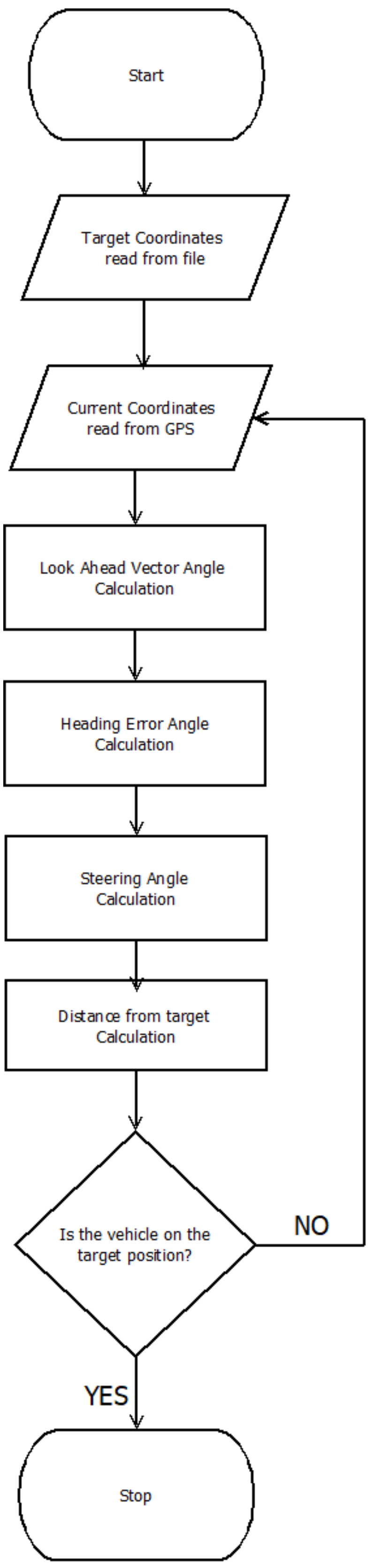

Start
Target Coordinates read from file
Current Coordinates read from GPS
Look Ahead Vector Angle Calculation
Heading Error Angle Calculation
Steering Angle Calculation
Distance from target Calculation
Is the vehicle on the target position?
NO
YES
Stop

Figure 6.6: Flowchart of the Steering Control Procedure

In order to validate the proper functionality of the aforementioned “Pure Pursuit“ method, we first simulated the control loop in MATLAB. Using real data that have been gathered from the preliminary testing described above, and with the help of the Robotics Toolbox from Peter

Corke [47] the code was debugged before the real world experiments. In Figure 6.7 an instance of the simulation can be seen. The path in red colour is the recorded path that a human driver followed controlling the vehicle as described above. The path in black colour is the path deriving from the simulation in MATLAB with the “Pure Pursuit“ method utilized for steering angle calculation.

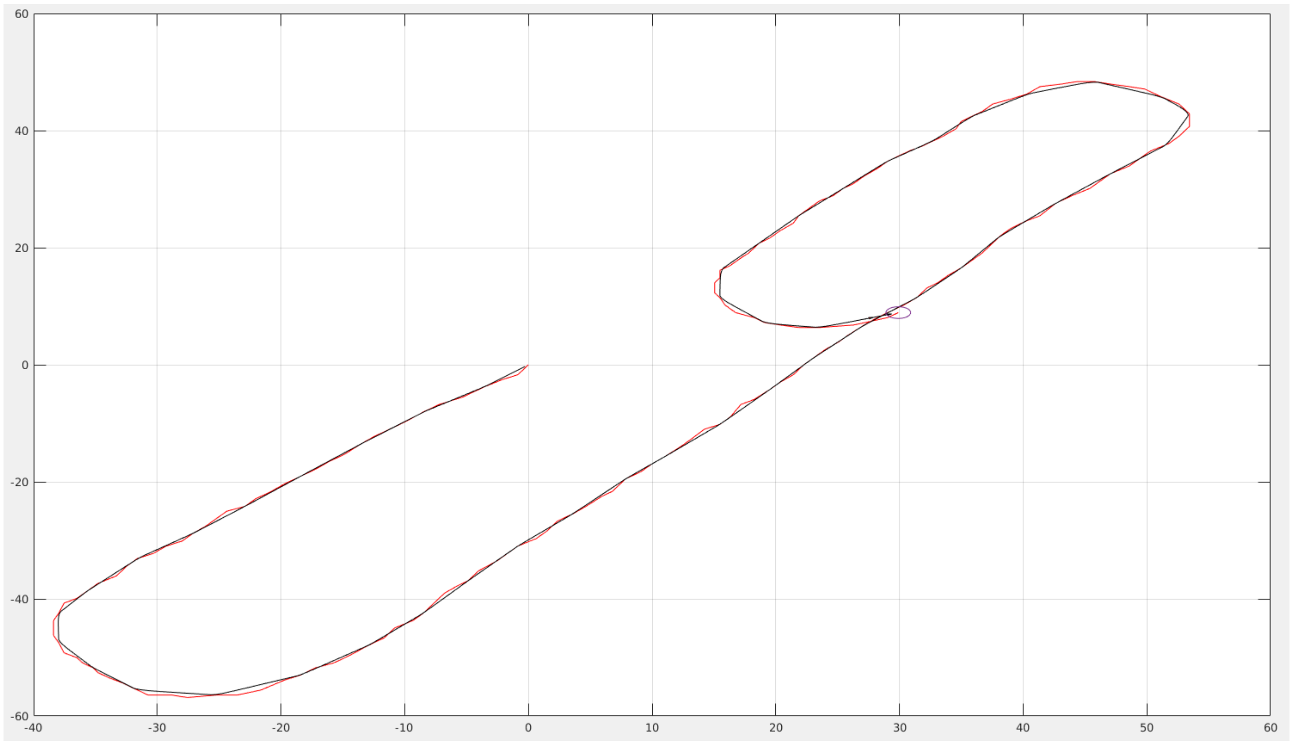


Figure 6.7: Simulation of the Pure Pursuit Method on MATLAB

The steering control procedure follows a simple but proven sequential logic. At first, the target position is read from the input file consisting of GPS coordinates pairs and the current position is acquired by reading the GPS sensor directly. As described earlier both current and target data are converted in a Cartesian format for easier manipulation. Look-ahead vector angle is then calculated with the help of the $atan2$ function, and by subtracting this value from the current heading angle (acquired by the IMU) heading angle error is calculated. By using the pure pursuit formula δ value, the angle for the wheels to steer, is generated. Limits are imposed for maximum steering lock and before applying steering value to the wheels previous steering command is accounted. Lastly, the distance from the target position is calculated. The aforementioned steps are repeated until target position has been reached (within a certain radius). In Figure 6.6 the block diagram depicts the control procedure for the steering wheel. Regarding throttle and braking in this test case we did not actuate them autonomously. Throttle is set before the control loop at a steady value that propels the car at a walking pace and is not altered until the target position is reached, where it is set to zero and the brakes are set to full in order to stop the vehicle. The loop iteration is continuous as vehicle's position is continuously changing and steering corrections occur in the fastest possible rate in order to have smooth and precise direction changes. In Figure 6.8 one of the multiple test conducted for path following is presented. The recorded path that the vehicle then autonomously followed can be seen.

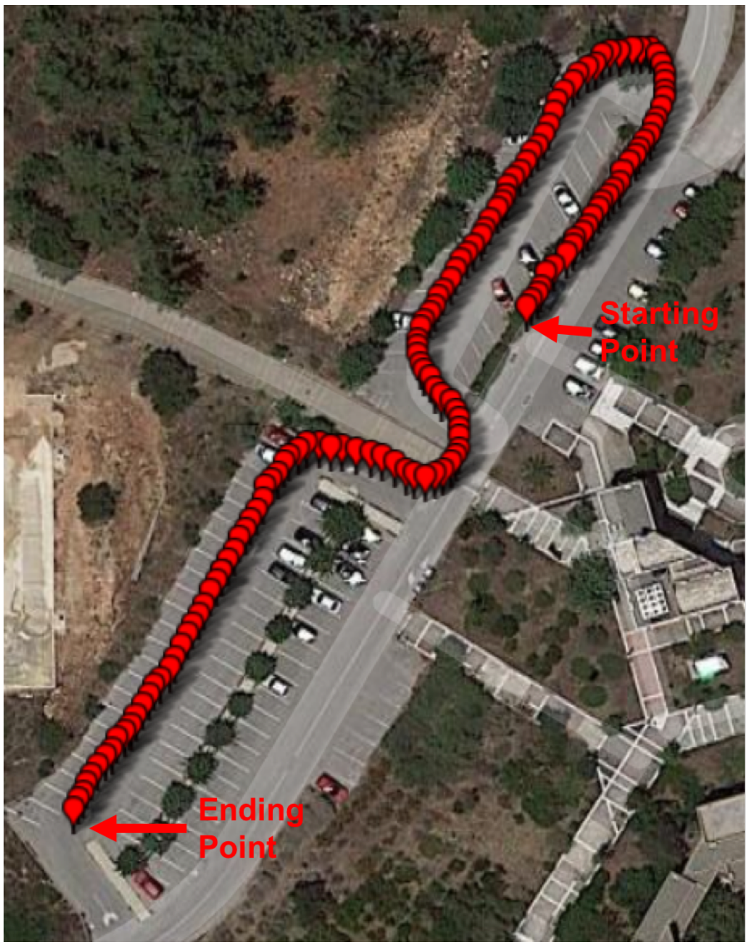


Figure 6.8: Vehicle's trajecory inside the TUC Campus

In order to further prove the proper functionality of the proposed algorithm, additional experiments conducted in the parking spaces of the Technical University of Crete. In Figure 6.9 an alternative route followed is presented, proving that regardless the test area, the functionality remains the same.

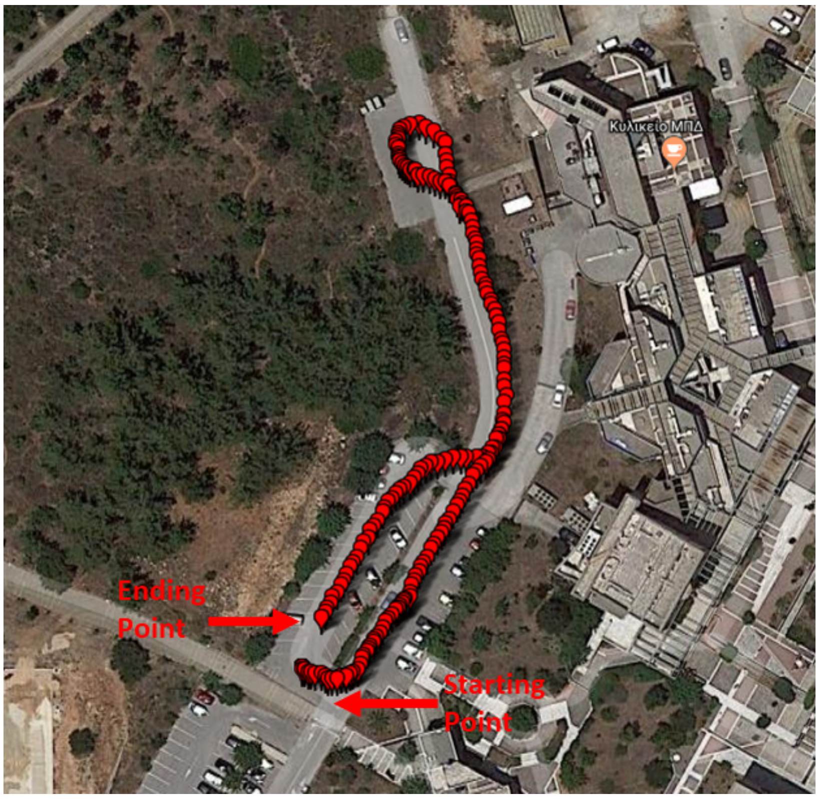


Figure 6.9: Alternative trajecory inside the TUC Campus

In order to validate the accuracy of the “Pure Pursuit“ method we visualized the data gathered from a third experiment. In Figure 6.10 you can see the path that the vehicle autonomously followed with the blue line and the target coordinates that were given as input with the red stars. There is a deviation visible, however the targets are not meant to reach exactly, but in a radius of 3 meters, since the vehicle has a considerable volume. The red circle simulates the radius of the final target coordinates.

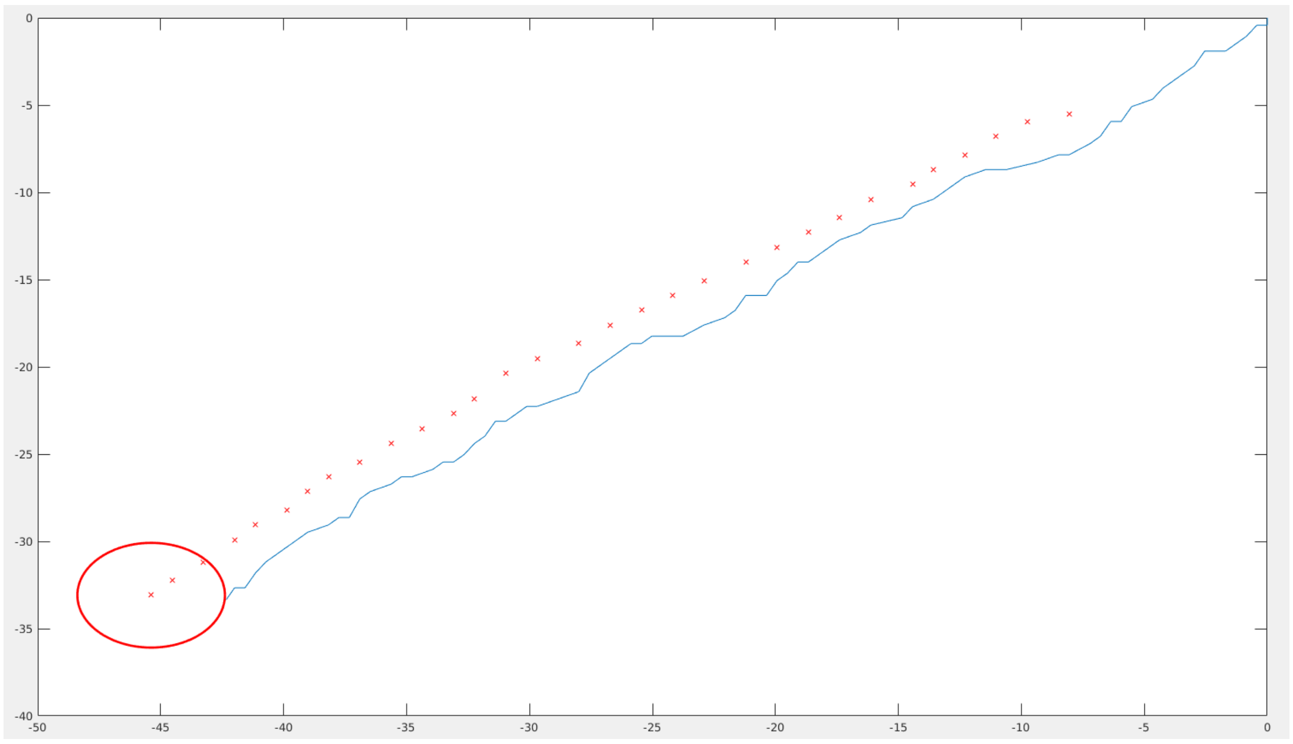


Figure 6.10: Target Coordinates and Actual Path followed

6.3 Object Detection and Avoidance

Since the ability of the vehicle to navigate autonomously in a predefined path is established and thoroughly tested, the next step was to establish an autonomous functionality of perceiving the environment and avoiding a moving obstacle, in our case a pedestrian. As mentioned earlier, a 2D 360^o LiDAR is mounted on top of the vehicle scanning the surrounding environment. Exploiting LiDAR readings, we developed the appropriate software for pedestrian detection and avoidance. The LiDAR continuously scans the environment of the vehicle and as soon as it detects an obstacle (pedestrian) in a distance smaller that a certain threshold the vehicle stops its operation. Due to space limitations on the experimentation field, the avoidance is restricted to the immobilization of the vehicle at its path, and not a path recalculation. A snapshot of the LiDAR readings is depicted in Figure 6.11, where the arrow represents the actual heading of the vehicle, while the readings inside the red eclipse correspond to the pedestrian detected. This approach also simplistic demonstrates the ability of the testing platform to handle unexpected events as they occur during the autonomous operation of the vehicle.

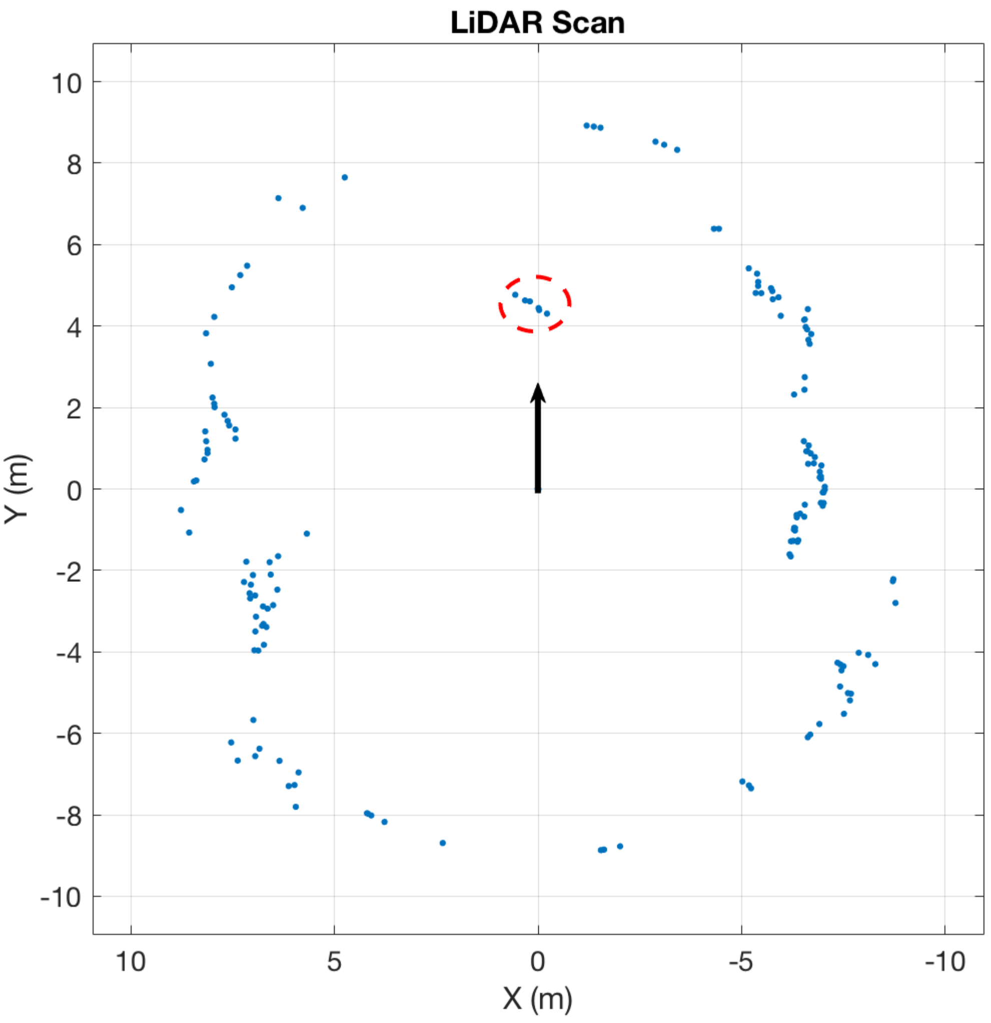


Figure 6.11: Data acquired from the LiDAR sensor

The developed software was based on the GPS Waypoints following software developed for the previous experiments. The functionality of detecting an object and actuating the brakes was added to implement this experiment. In the control loop that was previously described, a condition was introduced about possible objects blocking the vehicle's path. An explicit region of interest was selected in front of the car covering a distance of 5 meters ahead of the sensor and an angle of $\pm 25^o$ from the center point of the vehicle. The region of interest was introduced in order to avoid false detections or detections that won't affect the current path of the vehicle, as they may be on the sides or behind the vehicle. In the flowchart af Figure 6.13 the control and pedestrian detection procedure is visualized. In this experiment only the throttle control maintained the non-autonomous aspect, since braking was also controlled by the vehicle now. Regarding pedestrian detection we have also implemented a method of visual detection. Using the Pedestrian Detection Network provided by Matlab, and by the help of the GPUCoder provided in the latest Matlab toolbox, this network was converted to appropriate CUDA code for the Jetson TX2. As seen in Figure 6.12 the results are quite satisfactory. We have to point out here that this experiment was purely conducted to establish the ability of the Jetson TX2 to execute successfully image processing and detection tasks running neural networks. Due to the complexity of the vision tasks and the sensor fusion with the LiDAR readings required vehicle control with vision was planned on the future improvements of the platform.

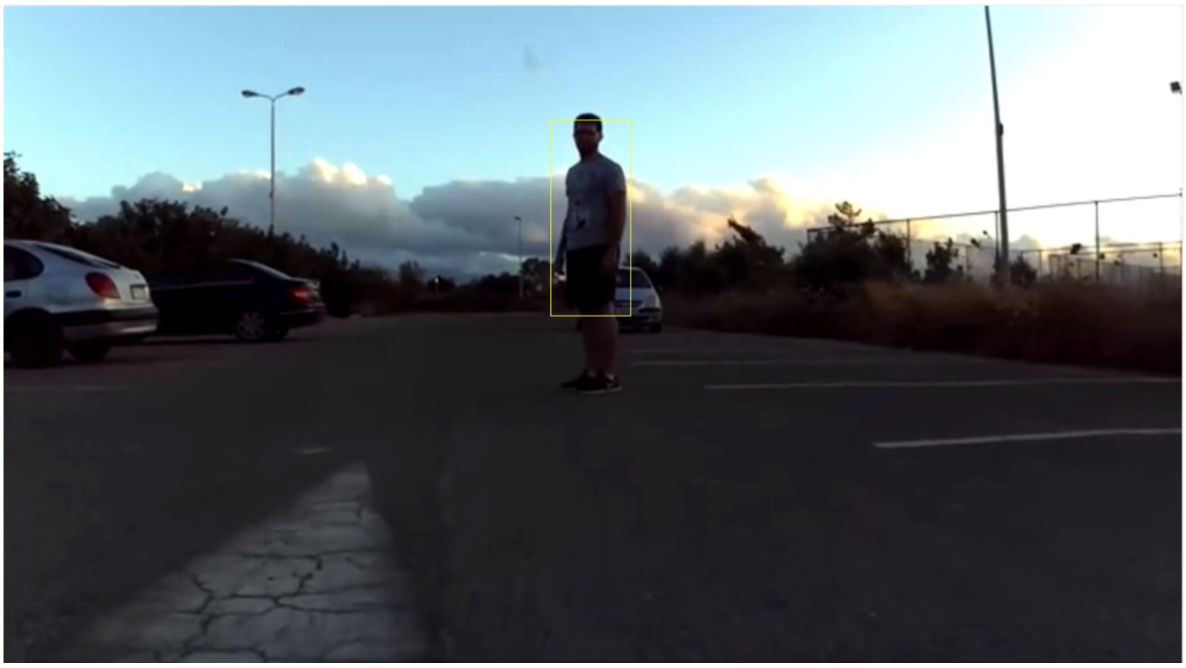


Figure 6.12: Pedestrian Detection with vision

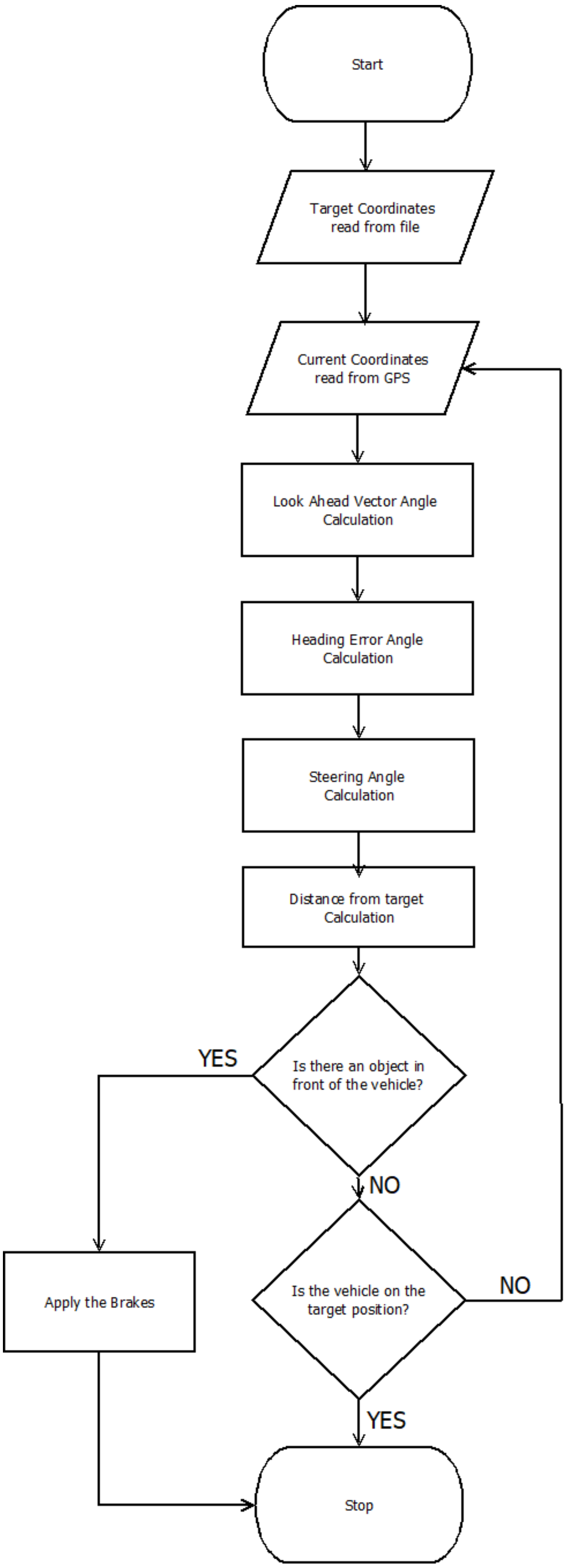

Start
Target Coordinates read from file
Current Coordinates read from GPS
Look Ahead Vector Angle Calculation
Heading Error Angle Calculation
Steering Angle Calculation
Distance from target Calculation
Is there an object in front of the vehicle?
YES
NO
Apply the Brakes
Is the vehicle on the target position?
NO
YES
Stop

Figure 6.13: Flowchart of the Steering Control and Object Avoidance Procedure

Chapter 7

Conclusion

This thesis has described the detailed assembly, development and testing of an autonomous navigation platform targeting to enhance the educational process in the domain of autonomous driving. It has described the difficulties and drawbacks of similar platforms and the advantages and innovations that we offer through comparison with related work in the domain. There has been a detailed description of every component and sensor used in the platform, as well as the software development procedure. Through extensive testing and experimentation, the ability of this platform to serve its purpose has been proven.

The proposed testbed, as mentioned before, aims to promote the educational procedure in the domain of autonomous vehicles. Having that in mind, we are planning to provide public access to the software in its final form. In addition, electronic schematics, mechanical blueprints and detailed bill of materials, to replicate the platform, will also be available. Our long term vision is to provide a robust open platform for education and research purposes to the community. Regarding software development and sensor easier installation and communication, we are also planning to implement different modules of our platform on Robot Operating System (ROS), gaining significant advantages in development and simulation with the provided tools. However, it should be left to the choice of the researcher to select the middle-ware of his preference and build upon that.

An updated version of the proposed system is under development, where we are also considering the need for additional computational power so that we can implement more complicated strategies. The Jetson TX2, although proven robust enough for our purposes, can't keep up with the increasing demands of an autonomous vehicle navigating in a dense environment performing complex functions. For that reason we have already considered the Drive PX2 which apart from the additional computational power and extended connection capabilities, can also accommodate the already developed CUDA code with minor tweaks and modifications and also bears a dedicated software package for autonomous navigation tasks. As far as perception is concerned we are planning on upgrading from the 2D LiDAR currently incorporated into the vehicle to a 3D one, capable of scanning at greater distances (more than $250 meters$) and providing a 3D representation of the real world.

Finally, one should not forget that autonomous automotive industry have seen tremendous improvements during the last decade, and will continue to improve. Many companies are investing in the field of automated driving, which fuels the development of new technology. Sensors and CPUs are reducing in prices and growing in capabilities, making them more accessible to every researcher. An easily adaptable platform to accommodate these new sensors and experiment with their capabilities is going to be more crucial than ever.

Chapter 8

Bibliography

[1] T. Litman, "Autonomous vehicle implementation predictions: Implications for transport planning," Victoria Transport Policy Institute, Tech. Rep., 2018.

[2] J. M. Anderson, N. Kalra, K. D. Stanley, P. Sorensen, C. Samaras, and T. A. Oluwatola, "Autonomous vehicle technology: A guide for policymakers," RAND Corporation, Tech. Rep., 2016.

[3] S. Liu, J. Tang, Z. Zhang, and J. L. Gaudiot, "Computer architectures for autonomous driving," *Computer*, vol. 50, no. 8, pp. 18–25, 2017.

[4] "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles," SAE, Tech. Rep., 2018.

[5] C. Premebida, G. Monteiro, U. Nunes, and P. Peixoto, "A lidar and vision-based approach for pedestrian and vehicle detection and tracking," in *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*, 2007, pp. 1044–1049.

[6] K. C. Fuerstenber, K. C. Dietmayer, , and V. Willhoeft, "Pedestrian recognition in urban traffic using a vehicle based multilayer laserscanner," in *Intelligent Vehicle Symposium,2002.IEEE*, vol. 1, 2002, pp. 31–35.

[7] T. Ogawa and K. Takagi, "Lane recognition using on-vehicle lidar," in *Intelligent Vehicles Symposium, 2006 IEEE*, 2006, pp. 540–545.

[8] Z. Chong, B. Qin, T. Bandyopadhyay, M. Ang, E. Frazzoli, and D. Rus, "Synthetic 2d lidar for precise vehicle localization in 3d urban environment," in *Robotics and Automation (ICRA), 2013 IEEE International Conference*, 2013, pp. 1554–1559.

[9] T. Miyasaka, Y. Ohama, , and Y. Ninomiya, "Ego-motion estimation and moving object tracking using multi-layer lidar," in *Intelligent Vehicles Symposium, 2009 IEEE*, 2009, pp. 151–156.

[10] M. Himmelsbach, A. Müller, T. Lüttel, and H.-J. Wünsche, “Lidar-based 3d object perception,” in *1st International Workshop on Cognition for Technical Systems*, vol. 1, 2008.

[11] C. Urmson, J. Anhalt, J. A. D. Bagnell, C. R. Baker, R. E. Bittner, J. M. Dolan, D. Duggins, D. Ferguson, T. Galatali, H. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. Howard, A. Kelly, D. Kohanbash, M. Likhachev, N. Miller, K. Peterson, R. Rajkumar, P. Rybski, B. Salesky, S. Scherer, Y.-W. Seo, R. Simmons, S. Singh, J. M. Snider, A. T. Stentz, W. R. L. Whittaker, and J. Ziglar, “Tartan racing: A multi-modal approach to the darpa urban challenge,” Carnegie Mellon University, Pittsburgh, PA, Tech. Rep., April 2007.

[12] A. Mendes, L. C. Bento, and U. Nunes, “Multi-target detection and tracking with a laser scanner,” in *IEEE Intelligent Vehicles Symposium, 2004*, June 2004, pp. 796–801.

[13] “Safety of laser products-part 1: Equipment classification and requirements,” International Elechtrotechnical Commission et al., Tech. Rep., 2007.

[14] P. McCormack, “Lidar system design for automotive/industrial/military applications,” National Semiconductor Corporation, Tech. Rep., 2006.

[15] D. Geronimo, A. M. Lopez, A. D. Sappa, and T. Graf, “Survey of pedestrian detection for advanced driver assistance systems,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 7, pp. 1239–1258, July 2010.

[16] J. Lidholm, “Stereo vision algorithms in reconfigurable hardware for robotics applications,” Mälardalen University, School of Innovation, Design and Engineering, Tech. Rep. 141, 2011.

[17] P. Dollar, C. Wojek, B. Schiele, and P. Perona, “Pedestrian detection: An evaluation of the state of the art,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 743–761, April 2012.

[18] F. García, A. de la Escalera, J. M. Armingol, J. G. Herrero, and J. Llinas, “Fusion based safety application for pedestrian detection with danger estimation,” in *14th International Conference on Information Fusion*, July 2011, pp. 1–8.

[19] A. Carullo and M. Parvis, “An ultrasonic sensor for distance measurement in automotive applications,” *IEEE Sensors Journal*, vol. 1, no. 2, pp. 143–147, Aug 2001.

[20] “Gps: The global positioning system.” [Online]. Available: https://www.gps.gov/systems/gps/

[21] “Elprocus, the budding electronics’ engineer knowledge space.” [Online]. Available: https://www.elprocus.com/how-gps-system-works/

[22] “Engineering 360,powered by ieee globalspec.” [Online]. Available: https://www.globalspec.com/learnmore/sensors_transducers_detectors/ orientation_position_sensing/inertial_measurement_units_imu

[23] "Waymo." [Online]. Available: https://waymo.com/

[24] "On the road to fully self-driving," Waymo, Tech. Rep., 2017.

[25] "Tesla autopilot." [Online]. Available: https://www.tesla.com/en_EU/autopilot?redirect=no

[26] "Audi." [Online]. Available: https://www.audi-mediacenter.com/en/on-autopilot-into-the-future-the-audi-vision-of-autonomous-driving-9305/the-new-audi-a8-conditional-automated-at-level-3-9307

[27] "2018 self-driving safety report," General Motors, Tech. Rep., 2018.

[28] L. Paull, J. Tani, H. Ahn, J. Alonso-Mora, L. Carlone, M. Cap, Y. F. Chen, C. Choi, J. Dusek, Y. Fang, D. Hoehener, S. Liu, M. Novitzky, I. F. Okuyama, J. Pazis, G. Rosman, V. Varricchio, H. Wang, D. Yershov, H. Zhao, M. Benjamin, C. Carr, M. Zuber, S. Karaman, E. Frazzoli, D. D. Vecchio, D. Rus, J. How, J. Leonard, and A. Censi, "Duckietown: An open, inexpensive and flexible platform for autonomy education and research," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 1497–1504.

[29] J. Tani, L. Paull, M. T. Zuber, D. Rus, J. How, J. Leonard, and A. Censi, "Duckietown: An innovative way to teach autonomy," in *Educational Robotics in the Makers Era*, D. Alimisis, M. Moro, and E. Menegatti, Eds. Cham: Springer International Publishing, 2017, pp. 104–121.

[30] "Donkey car." [Online]. Available: http://www.donkeycar.com/

[31] "Barc - berkeley autonomous race car." [Online]. Available: http://www.barc-project.com/

[32] J. Gonzales, F. Zhang, K. Li, and F. Borrelli, "Autonomous drifting with onboard sensors: Proceedings of the 13th international symposium on advanced vehicle control (avec' 16), munich, germany, 13–16 september 2016," 12 2016, pp. 133–138.

[33] "Mit racecar." [Online]. Available: https://mit-racecar.github.io/

[34] "F1/10." [Online]. Available: http://f1tenth.org/

[35] "Shell eco-marathon europe." [Online]. Available: https://www.shell.com/energy-and-innovation/shell-ecomarathon/europe.html

[36] "Tucer team." [Online]. Available: http://www.tucer.tuc.gr/el/archiki/

[37] P. Spanoudakis, N. C. Tsourveloudis, G. Koumartzakis, A. Krahtoudis, T. Karpouzis, and I. Tsinaris, "Evaluation of a 2-speed transmission on electric vehicle's energy consumption," in *2014 IEEE International Electric Vehicle Conference (IEVC)*, Dec 2014, pp. 1–6.

[38] P. Spanoudakis and N. C. Tsourveloudis, “On the efficiency of a prototype continuous variable transmission system,” in *21st Mediterranean Conference on Control and Automation*, June 2013, pp. 290–295.

[39] D. S. Efstathiou, A. K. Petrou, P. Spanoudakis, N. C. Tsourveloudis, and K. P. Valavanis, “Recent advances on the energy management of a hybrid electric vehicle,” in *20th Mediterranean Conference on Control Automation (MED)*, July 2012, pp. 896–901.

[40] A. K. Petrou, D. S. Efstathiou, and N. C. Tsourveloudis, “Modeling and control of the energy consumption of a prototype urban vehicle,” in *19th Mediterranean Conference on Control Automation (MED)*, June 2011, pp. 44–48.

[41] W. C. Mitchell, A. Staniforth, and I. Scott, “Analysis of ackermann steering geometry,” in *SAE Technical Paper*. SAE International, 12 2006. [Online]. Available: https://doi.org/10.4271/2006-01-3638

[42] I. A. of Public Transport, “Autonomous vehicles: A potential game changer for urban mobility,” Internatioal Association of Public Transport, Tech. Rep., 2017.

[43] N. Smolyanskiy, A. Kamenev, J. Smith, and S. Birchfield, “Toward low-flying autonomous mav trail navigation using deep neural networks for environmental awareness,” 05 2017.

[44] IEEE, “The top programming languages 2018,” IEEE Spectrum, Tech. Rep., 2018.

[45] “Gstreamer.” [Online]. Available: https://gstreamer.freedesktop.org/

[46] J. M. Snider, “Automatic steering methods for autonomous automobile path tracking,” Robotics Institute, Carnegie Mellon University, Tech. Rep. CMU-RI-TR-09-08, 2009.

[47] P. I. Corke, *Robotics, Vision & Control: Fundamental Algorithms in Matlab*. Springer, 2011, iSBN 978-3-642-20143-1.

Appendices

Appendix A

Platform User Manual

Following there is a description on how to get started with the basic platform functionalities and familiarise with the procedures of using the platform with safety. This is also an introduction on how to develop your own test cases and execute them successfully.

Before starting the following procedure please take every precaution listed below: Wear a pair of antistatic gloves and protective glasses. Remove metal jewellery that you may wear, such as watches, rings, earrings and neck chains. Inspect visually the wiring of the vehicle for any damage. If something seems out of order, check with a digital multimeter for short circuits. Don't proceed unless you are absolutely sure about your safety. Always remember to check tyre pressure of the vehicle before use. Friendly reminder to keep the multimeter with you, as it is going to be needed further on.

Quick Start Guide

This quick start guide is organised in steps that must explicitly followed in the given order for system power up and use:

Step 1: CPU and USB Hub Power Supply Connection and Power Up
First step is to power up the Central Processing Unit and the USB hub installed in the vehicle. Two 12V batteries, like the ones seen in Figure A.1a are required. The capacity of the battery is not relevant, however the nominal voltage must be 12V. We can point out here that fully charged lead acid batteries measure $\sim$ 13.5 V and are completely suitable for use. Bare in mind not to over-discharge the batteries (below 12 V) or their life expectancy will be greatly reduced. The batteries have to be connected in series, starting from the negative pole of the first battery and reaching the positive pole of the other. The white connecting wire shown in Figure A.1b is provided for this reason. Place the two batteris on the left side of the driver's compartment and verify (with a multimeter) more than 24 V on output. Locate in the same position an orange and purple pair of wires having battery terminals attached. Connect purple wire to negative ($-$) and orange wire to positive (+) like in Figure A.1b. Always remember to

connect the negative pole first.

(a) 12 V - 7.2 Ah battery pair

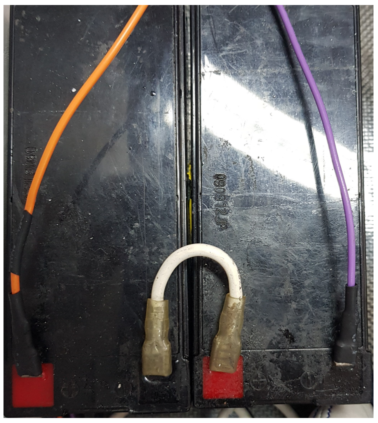
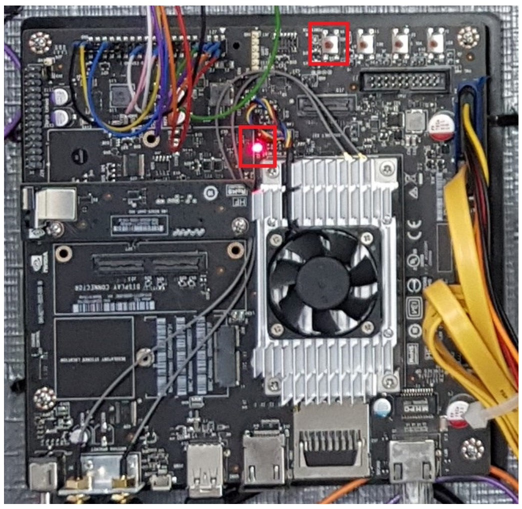
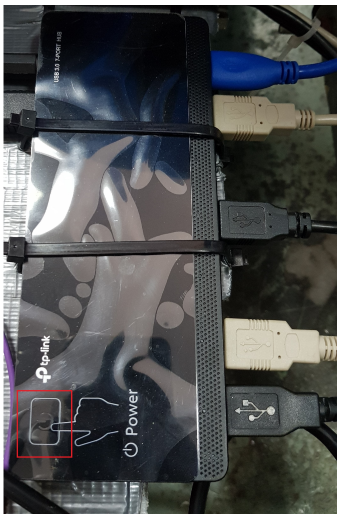

(b) Batteries on-board the vehicle

Figure A.1: CPU and USB hub power supply

Now the CPU and USB hub are powered on. Check that the power led on each device is on, verifying the power application. Next step is to press the power button on the Jetson TX2 (the USB Hub powers up itself when connection occurs). Note that the fan on the Jetson TX2 is not going to start, this is not a defect. In Figure A.2 the power led and buttons on the devices can be seen.

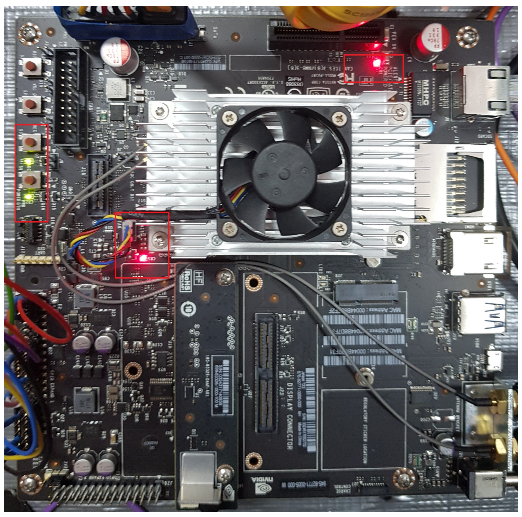

(a) Jetson TX2 Power Led and Power Button

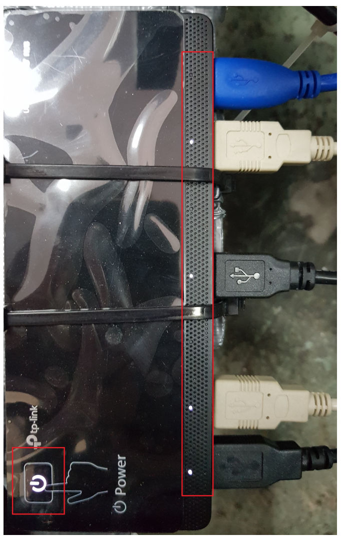


(b) USB Hub Power Led and Power Button

Figure A.2: Jetson TX2 and USB hub power led and button

After pressing the button on the Jetson TX2, multiple more leds indicating the board's state are going to light, as well as in the USB hub, indicating established connection via the hub between the devices and Jetson TX2. Check that the board and the hub are in the condition shown in Figure A.3 before moving to the next step. Also verify that 5 USB cables are connected into the hub.


(a)



(b)

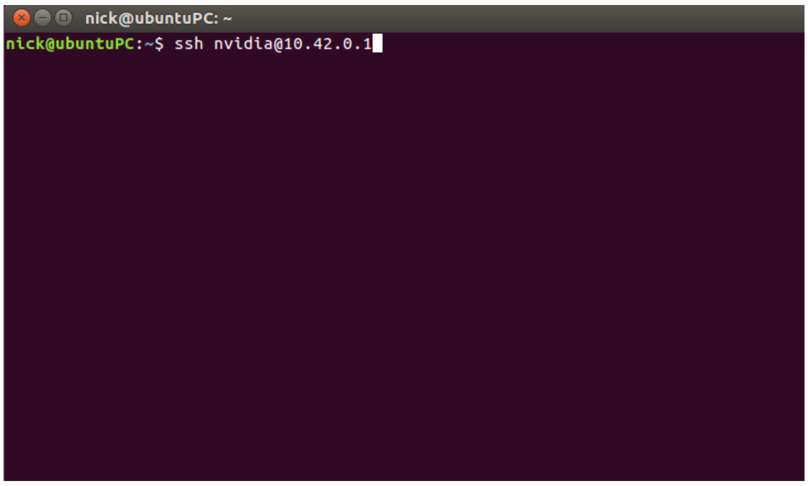
Figure A.3: Running state of Jetson TX2 and USB Hub

<u>IMPORTANT NOTICE:</u> When connecting the positve pole of the battery there might be a spark. Don't be afraid, this is happening due to the instantaneous charging of the capacitors located in the Dc/Dc converters powering the electronics. This doesn't pose a threat to these devices as they are internally protected.

<u>Step 2: Connect to Jetson TX2 and verify normal boot</u>

After powering up sequence connect the ethernet cable located on the left side of the driver's compartment to a Linux (or Mac) computer and through a terminal connect to the Jetson TX2 with the following command (Figure A.4):

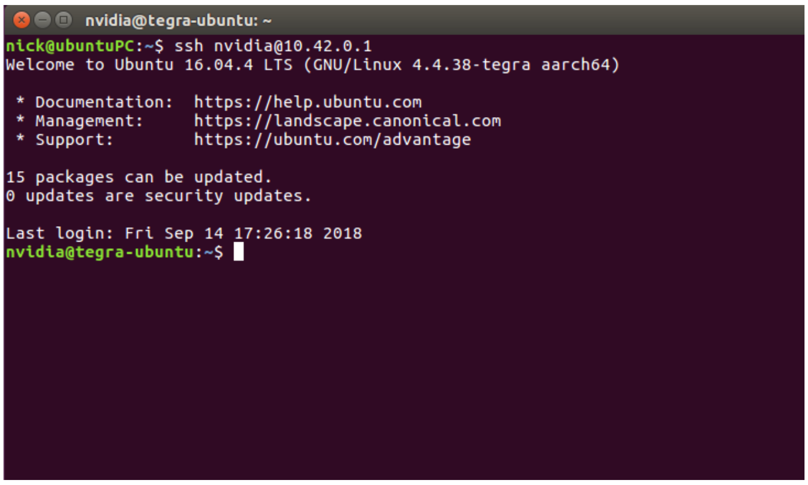
```
$ ssh nvidia@10.42.0.1
```



```
nick@ubuntuPC: ~
nick@ubuntuPC:~$ ssh nvidia@10.42.0.1
```

Figure A.4: Command to connect to Jetson TX2 through terminal

When prompted for password enter "nvidia" and wait for the connection to be established. Successful connection will lead to a screen like Figure A.5.

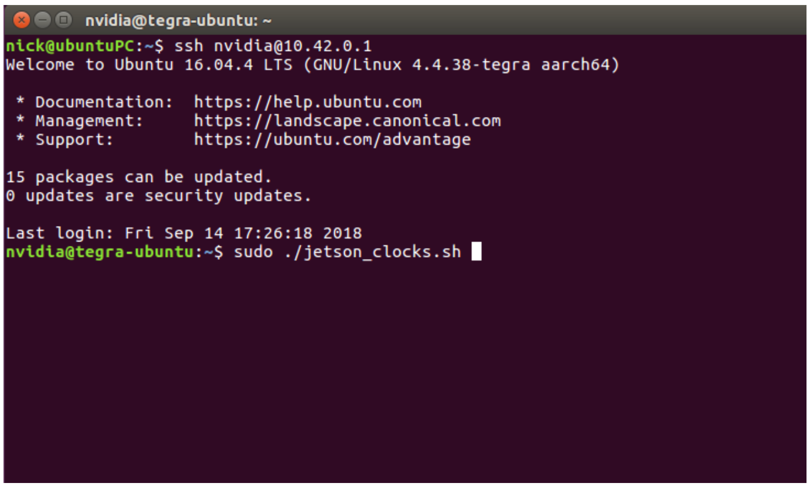


```
nvidia@tegra-ubuntu: ~
nick@ubuntuPC:~$ ssh nvidia@10.42.0.1
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.38-tegra aarch64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

15 packages can be updated.
0 updates are security updates.

Last login: Fri Sep 14 17:26:18 2018
nvidia@tegra-ubuntu:~$
```

Figure A.5: Screen after successful connection

Last but of extreme importance is to enable the Jetson's TX2 Maximum Performance mode. In the same terminal execute the following command (Figure A.6):

```
$ sudo ./jetson_clocks.sh
```

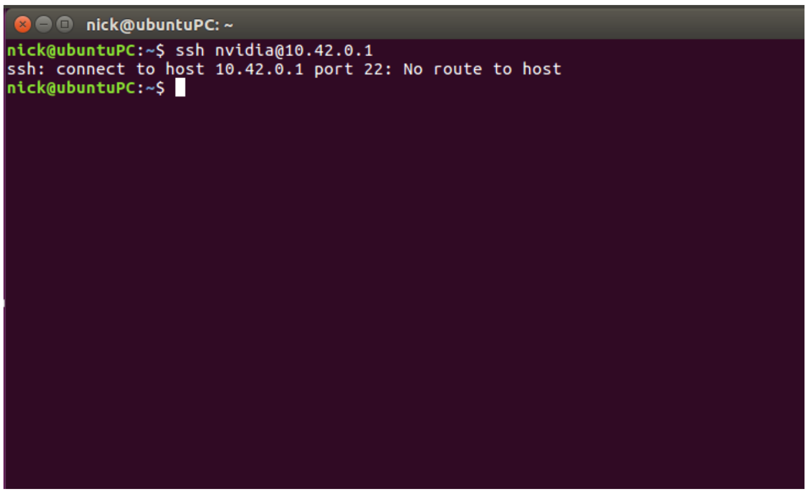


```
nvidia@tegra-ubuntu: ~
nick@ubuntuPC:~$ ssh nvidia@10.42.0.1
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.38-tegra aarch64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

15 packages can be updated.
0 updates are security updates.

Last login: Fri Sep 14 17:26:18 2018
nvidia@tegra-ubuntu:~$ sudo ./jetson_clocks.sh
```

Figure A.6: Command to enable Maximum Performance mode

An easy way to determine if the Maximum Performance mode is active is by the cooling fan, that should now be rotating at full speed.

If the connection hangs out and you have been prompted that the connection can't be established,like Figure A.7, you have to reset the board as something has interrupted the normal boot procedure. The reset button can be seen in Figure A.8 below. Pressing once is going to reset the board and start boot sequence again. If the problem persists, better consult the boards troubleshooting guide or the manufacturer.



```
nick@ubuntuPC: ~
nick@ubuntuPC:~$ ssh nvidia@10.42.0.1
ssh: connect to host 10.42.0.1 port 22: No route to host
nick@ubuntuPC:~$ 
```

Figure A.7: Connection Error Common Problem

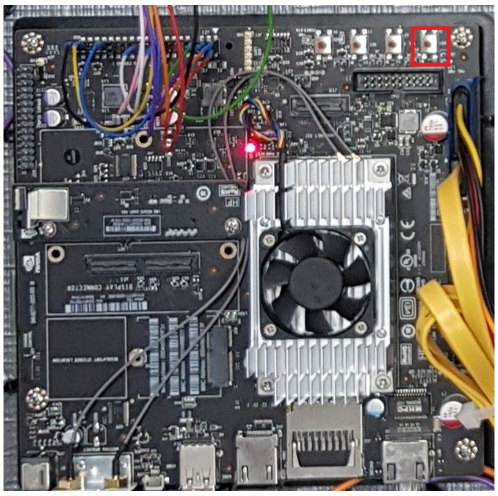


Figure A.8: Jetson TX2 Reset Button

For those of you not familiar with Linux (or Mac) computers (the easy and direct way of connection) you can also use Windows, however Putty (a program for establishing Secure Shell (SSH) connections) must be used as a middle ware (or any other similar software of your preference).

Step 3: Connect Motor and Stepper Motors power supply

Now that the main computer is powered on and booted, the main power supply can be connected. Two 12V batteries are also needed in this case, however due to the fact that these motors demand more power, the capacity is greater than before. Also the terminals of these batteries are different following the automotive standards, as seen in Figure A.9 .Blue and Red plastic caps must be removed from the poles of the battery to be used, but is recommended to be kept and put on when batteries are stored away.

Figure A.9: 12V - 40Ah battery

In order to be able to connect these batteries you are going to need a 10 *mm* spanner as the connectors have a nut and bolt fastening system. Similarly to Step 1, after placing the batteries on the dedicated place in the rear compartment of the vehicle and fastening with the safety strap provided (Figure A.10a), you have to connect the two batteries in series with the provided wire. These batteries are fitted with different width positive and negative terminals so the wire fits in a specific way. Don't forget to securely tight the bolts with the spanner on the connectors. Then, the negative terminal (with the black and purple wires attached) must be connected first and secured and finally the positive terminal (with the white 250 Amps fuse before the wiring Figure A.10b). The final connection should be like Figure A.11.

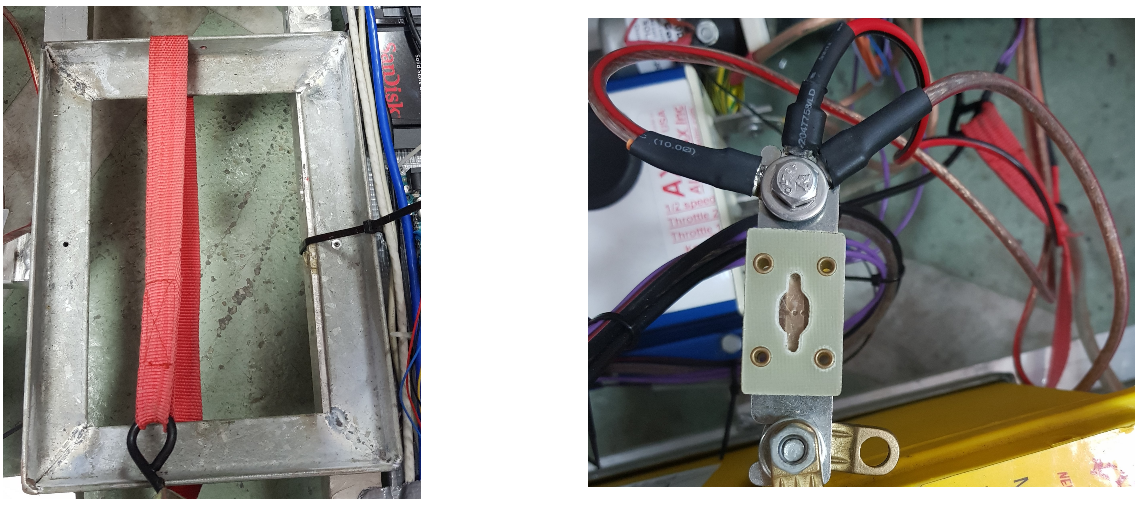

(a) (b)

Figure A.10: Main Power Supply Holder and Fuse

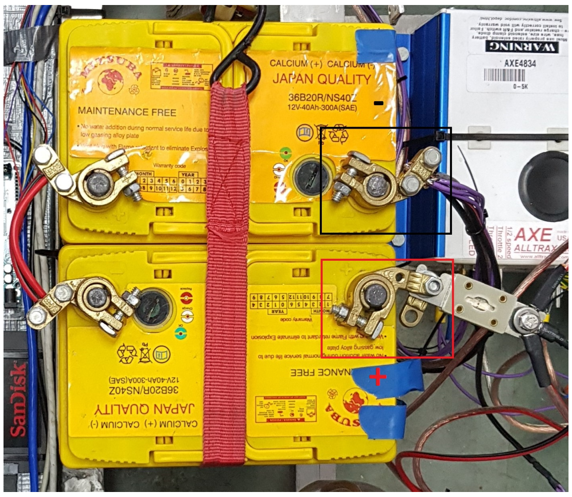


Figure A.11: Main Power Supply Connection

Now the power connections have been properly done, but still power hasn't been applied. To do so, in the right side of the driver's compartment, two safety buttons (Figure A.12) are located. Disengaging them will apply power to the motor and the stepper motors. You can visually verify the power applied by checking that the led on the controllers are stable and green. The led state can be seen in Figure A.13. After visually verifying proper power application, make sure that the terminals on the battery are tightened adequately, and the batteries properly seated and fastened with the strap. Only then you are ready to proceed to the next step.

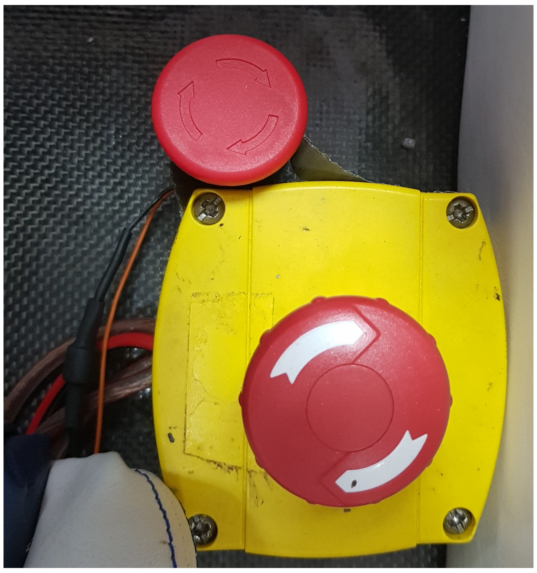
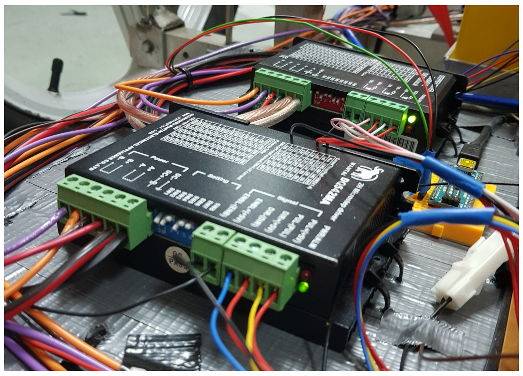


Figure A.12: Safety Buttons


(a)

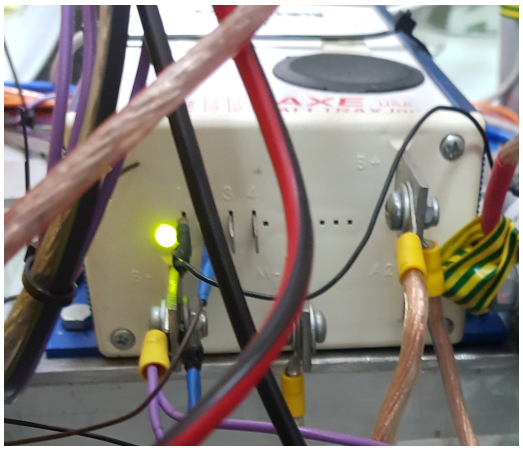

(b)

Figure A.13: Status Leds

IMPORTANT NOTICE: When connecting the positve pole of the battery there might be a spark. Don't be afraid, this is happening due to the instantaneous charging of the capacitors located in the motor and stepper motor controller. This doesn't pose a threat to these devices. However the spark, if occurs, might be greater than the previous step, due to the different batteries.

Step 4: Select the appropriate software to execute
The last step is to execute the appropriate program already installed on the vehicle. Their names are extremely descriptive and detailed instructions can also be read on the terminal during the execution of the program. Every software developed in the process of completing this thesis is located in the following folder:

```
/home/Drivers
```

For developing your own test cases Steps 1 to 3 are compulsory, as they describe the sequence to start the vehicle. Step 4 can be used as reference in order to understand the already developed software and create yours. Every SDK and library used in our development is already installed on the board itself. The user can install every software package needed for development and is also encouraged to update the already installed software.
Important files to be read and understood are Keyboard_Control, Camera, Lidar and Logging for the basic functionality and logging, the Self_Nav_multiple_points for GPS Waypoint following and the Self_Nav_with_avoidance. These files correspond to the test cases described in Chapter 5.

Step 5: System shut down and storage
After experimentation has ended, in order to safely shut down the platform you have to make the steps in the opposite direction. The safety buttons should first be engaged to isolate the power from the stepper motors and the motor itself. The positive pole of the main power supply can now safely removed. Then close every program running on the Jetson TX2 and through

terminal type the following command for proper termination:

```
$ sudo shutdown -hP now
```

The main computer will be now off (check that the fan has stopped) and then the positive pole from the auxiliary power supply can be removed. Now there is no power on the system so negative poles can be disconnected too. The batteries should be removed from the vehicle and charged (if needed) in order to be ready for the next use. Store the batteries away with the poles always protected with at least electrical tape. A visual inspection of the vehicle is also required. When in storage please cover the electronics compartment and remove the sensitive sensors to protect them from dust, humidity and accidental damage. Do not store in a place with high or low temperatures, direct sunlight or exposure to the elements.

This short guide can help a user with basic knowledge, use the platform and conduct experimentation and testing on the domain of autonomous navigation. It also provides some safety hints. It is very important to always take every needed precaution and safety measure when using this platform, and have in mind to maintain it properly as mechanical and electrical failures may occur unexpectedly.

Appendix B

Hardware Specification Sources

The following websites can be really useful to someone trying to utilize its own experiments in this platform. Please refer to them if needed.

`https://developer.nvidia.com/embedded-computing`
The Nvidia Portal for Embedded Computing. Manuals, Software Updates, Drivers, Presentations and Tutorials regarding Jetson TX2 are freely available.

`https://devtalk.nvidia.com/default/board/188/jetson-tx2/`
Jetson TX2 Community Forum, maintained by NVIDIA.

`https://www.jetsonhacks.com/`
A website dedicated to the use of Jetson platform. Multiple tutorials can be found here as well as software for executing very common tasks. Saves a lot of time and helps in the first steps of Jetson development.

`https://www.stereolabs.com/`
ZED stereo camera website. Newer software and firmware versions, manuals and tutorials can be found here.

`http://scanse.io/`
Scanse Sweep lidar website. Newer software and firmware versions, manuals and tutorials can be found here.

`https://www.maxbotix.com/Ultrasonic_Sensors/MB1010.htm`
Maxbotix ultrasonic sensors website. Manuals and application diagrams can be found here.

`https://alltraxinc.com/axe-products/`
Axe motor controller website. Manuals, application diagrams and programming software can be found here.

`https://learn.adafruit.com/adafruit-ultimate-gps/overview` Adafruit Ultimate GPS website. Manuals, application diagrams, electrical schematics and prrgramming tutorials can be found here.

`https://www.sparkfun.com/products/retired/10736` Sparkfun Razor IMU website. Manuals, application diagrams, electrical schematics and prrgramming tutorials can be found here.