KTH Mechatronics Advanced Course

MF2059, HT 2021

FINAL REPORT

Automated Driving & ITS Testbed

AD-EYE

Marcus Akkila, Emilia Haltorp, Niclas Määttä,
Youchen Sun, Yaman Thif, Johannes Toma,
Alexander Wallén Kiessling, Caspar Westerberg,
Adam Westermark

---

19 December 2021

Abstract

Automated Driving & ITS Testbed

Automatic driving and intelligent transportation has the potential to mitigate many of the problems associated with increasing vehicle counts in modern society. To support the development of such technologies, vehicle and infrastructure communication and computing must be improved to tolerate increases in data flow, as well as computationally intensive tasks.

This project aims to design a Road Side Unit (RSU) which can be placed near roadsides. This unit should be able to perform computationally intensive tasks, whilst communicating with vehicles in two separate channels. One of these channels are for safety critical messages, the other for large volumes of sensor data. Moreover, it should be possible to reproduce the unit, so that several of them can be used to cover a larger testing area in the future. A demonstration of the RSU's performance should also be held.

To solve this task, the project team has designed a RSU which can be placed roadside. It fulfills nearly all major requirements set by the stakeholder, and can be expanded for future projects. The safety critical communication channel has been simplified in this project, and as such does not adhere to all existing standards. The project result is therefore mainly the prototype for the computational unit, while the safety communication element of the project still needs to be developed further.

Keywords: Automated Driving, ITS, Safety, RSU, Platooning

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The following chapter includes the background and description of the project, as well as the requirements and delimitations on the project. Lastly, a reader's guide details how the rest of the report is structured.

1.1 Background

In the modern age, transportation plays a prominent role in the development of national economies. As such, the global number of vehicles is gradually increasing. This accumulation threatens the sustainable development of modern society, with problems in the form of traffic accidents, pollution, energy requirements and congestion.

The emerging development of intelligent transportation and automatic driving shows the potential of mitigating such concerns. Automatic driving can remove the human factor causing many driving incidents, and vehicle platooning can reduce emissions as well as fuel consumption. However, to support this developing technology sensor information is required. Sensors can obtain vehicle environment data, which provides support for intelligent decision making in vehicles. By sharing the sensor data between vehicles, more intelligent decisions by the automotive systems can be made. However, since a large number of sensors and vehicles will produce a massive amount of sensing data, intelligent transportation is facing the issue of computational and communicative burdens. Edge computing units that are placed near roads can help alleviate this computational burden, by offloading intensive analysis of data from the vehicles. This infrastructure can also be placed at critical road segments, where traffic congestion or frequent accidents place a high demand on information. Vehicular communication can furthermore be

divided into separate channels. Safety critical messages can be broadcasted at a dedicated frequency, whereas large volumes of sensor data can be communicated separately.

The Automated Driving Simulation Platform (AD-EYE) is a co-simulation platform for Automated Driving Simulations, primarily for the purposes of functional safety. This platform is developed at the Royal Institute of Technology (KTH) in Stockholm. The platform provides simulation and testing possibilities for intelligent transportation solutions, with a particular focus on the safety aspect. They have asked for a Road Side Unit (RSU) to be designed, which can be used as an edge computing unit for intelligent transportation and automatic driving. Moreover this RSU should be capable of communicating with a dedicated safety critical message channel, and another channel for larger volumes of sensor data.

1.2 Project Description

This project aims to further develop the AD-EYE testbed by enabling vehicle communication and designing infrastructure prototypes for related edge computing.

1.2.1 Problem Formulation

The problem to be addressed is enabling vehicle communication on roads, by creating infrastructure that facilitates communication and edge computing. This task is divided into creating a computational unit which can be placed roadside, and a demonstration where the capabilities of this unit is shown through a platooning scenario. Moreover, the communication for this purpose should be divided into two separate channels. One of these channels should be dedicated to critical messages for automotive safety purposes, while the other channel is dedicated to a high volume flow of data.

The produced solution should be possible to extend for a more extensive testbed, using multiple computational units stationed at critical road sections. As such, different technologies for enabling communication must be investigated with respect to the requirements.

1.2.2 Goals

The goal of the project is to produce a computational unit that employs communication channels which can be used for testing automotive safety communication. This unit is commonly referred to as a Road Side Unit (RSU).

1.2.3 Motivation

There are many benefits for automotive safety from an increased level of communication. By sharing information, more intelligent decisions can be made by automotive systems. Smarter decisions have the potential of reducing driving casualties and fuel consumption, as well as optimizing traffic capacity.

Even if the produced solution is not the optimal choice for a future extended testbed, the project serves as an investigation into different technologies and their use case. Moreover, key problem points can be identified for future solutions to be produced.

1.2.4 Deliverables and Most Important Issues

The main deliverable of this project is the RSU alongside with all documentation required to construct more units. All developed software is also requested by the stakeholder with detailed documentation.

1.2.5 Team's Strengths

The team consists of nine second cycle mechatronics students from KTH. The students have knowledge in mechanical design, electronics, control theory and computer science. Besides this, each student has specialised knowledge in different topics such as deep learning, estimation theory, robotics and embedded systems. The collective knowledge and competence in these fields allowed for creating solutions to solve the desired deliverables. However, the team's knowledge was very limited regarding all topics related to radio communication such as ITS-G5, V2X communication, SDR and antennas. This meant that a lot of time needed to be spent researching this area.

1.3 Requirements

This chapter list all the given stakeholder- and technical requirements in the project. The requirements will serve as a basis for evaluating the prototypes during the next phases of the project.

1.3.1 Stakeholder Requirements

The given stakeholder requirements are listed below.

1. The RSU shall be able to transmit ROS messages wirelessly to on-road entities and the virtual vehicle.
2. The RSU, OBU and virtual vehicle shall be able to communicate over DSRC.
3. The RSU shall communicate to a backbone network.
4. The RSU shall be robustly constructed to withstand outer elements.
5. The RSU shall be constructed for placement on the ground.
6. The RSU shall be constructed with modularity in mind.
7. The RSU's computer shall be able to perform computationally intensive tasks.
8. The RSU shall be trackable.
9. The RSU shall be able to operate using an external power source other than the power grid.
10. The RSU shall be able to get manufactured for a cost less than 30000 SEK.
11. The OBU shall be able to transmit camera and LiDAR data wirelessly to the RSU.
12. The OBU shall be powered by both cars power outlet or an external power source.

1.3.2 Technical Requirements

The given technical requirements are listed below, these requirements are linked to the stakeholder requirements stated in section 1.3.1. As an example, technical requirement 4.2 is linked to stakeholder requirement 4. Technical requirement 10.1 was added a few weeks into project after existing of the shelf solutions were deemed too expensive for this project.

1.1 The communication shall support a a data-rate of at least 50 MBps.

2.1 The communication shall be done using ITS-G5 messages.

3.1 The communication shall take place over Ethernet.

4.1 The RSU shall be robust against vibration.

4.2 The RSU shall have IP54 or above rating in ingress protection.

4.3 The RSU shall be operational with little to no degradation between -15 to +35 degrees.

4.4 The RSU shall be constructed with a frame capable of carrying a load of at least 50 kilos without deformation for extended periods of time.

6.1 The RSU shall have the possibility of extending the sensors.

6.2 The RSU shall have a mount ontop of the RSU to secure position of a camera.

7.1 The RSU's PC shall have a processor with X86-64 architecture with at least 6 cores.

7.2 The RSU's PC shall have at least a single consumer grade GPU: GTX 1080 or above.

7.3 The RSU's PC shall have Ubuntu 16.04 and ROS kinetic installed.

8.1 The RSU shall be equipped with a GPS.

9.1 The RSU shall have the ability to run on battery power for at least an hour.

10.1 The ITS-G5 transceiver shall cost less than 750$ and be implemented from scratch since the existing ITS-G5 modules on the market is too expensive.

11.1 The communication shall support a data rate of at least 20 MBps.

12.1 The OBU shall be powered by standard automotive aux power outlet (12 V) or a battery with the same voltage.

1.4 Delimitations

The initial project aims were to deliver one RSU and two OBUs which are capable of fulfilling the requirements presented in the next section. Using this hardware and interfacing with communication and simulation software, two differential drive robots (more specifically, TurtleBot 3 Waffle Pi), are to perform platooning (see Appendix C) operations in cooperation with a virtual lead vehicle. This scenario is outlined in Figure 1.1. However, due to the disproportional workload in integrating the ITS-G5 solution as well as the architectural limitations of ROS 1, the OBUs were scrapped from this project.

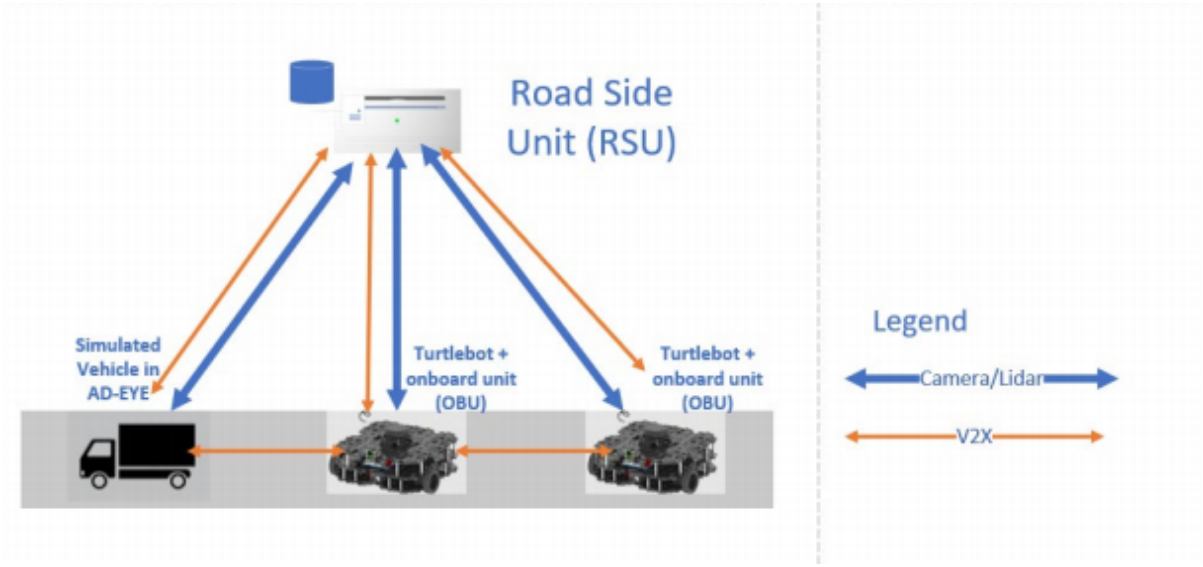


Figure 1.1: Outline of platooning test scenario

The project will make use of pre-existing software solutions to the fullest extent, and will only create new software when no existing solution fulfills the requirements. To this end, Robot Operating System (ROS) will be used extensively for interfacing between different systems. Moreover, off-the-shelf hardware will be used in the extent that is possible in the project.

The time allocated for the project is 39 weeks, running from spring to winter of 2021. The project team consists of nine second cycle students working a total of 18 credits each. This is combined with a third cycle student in the role of guidance, acting as a coach for the project.

Concerning deliverables, a pitch presentation and a state of the art analysis report is to be delivered during the spring. In the fall, a demonstration in the form of the previously mentioned platooning is expected, as well as a full project report.

1.5 Reader's Guide

This report is divided into 8 chapters excluding the appendices. Each of these aim to give the reader a deeper understanding of the topics in this project and further a better understanding for the decisions, results and conclusions presented in the end.

In chapter 2 the reader can find a literature review and state of the art. The intention is to give an introduction to current technologies that are found relevant for the project. This also provides some background information and the state of the art is used as a frame of reference for later design choices in the project.

Chapter 3 presents the methodology used in the project. This is which tools and techniques were used to solve problems, how the design choices were approached and how the project management were done.

In chapter 4, the implementation is described. Here, the reader can find a description about the hardware construction, as well as the implementation of electronics and software.

To assure fulfillment of the requirements, testing and evaluation was needed. A method and description for how this was executed can be found in chapter 5. The reader can find the tests that were performed in order to verify and validate the design here.

Following this is chapter 6 where the results of the project are presented. In this chapter it is also presented how well the requirements were met.

Chapter 7 reflects upon the results presented in the previous chapter. It includes a discussion about the results, whether they are reasonable, how well the requirements were met and what could have been done differently.

Lastly is chapter 8 which present the possible future work that could be done to the project. This includes anything that could be improved or that are still to be implemented.

The appendices present additional information produced during the course of this project, such as thermodynamic calculations and simulations, software documentation and project management documents.

Chapter 2

Literature Review and State of the Art

In order to make informed decisions about building the RSU and OBU as well as other parts of the project implementation, a State of the Art (SOTA) research was conducted. The findings of it are published in this chapter.

2.1 Vehicle to Everything Communication - V2X

Vehicle-To-Everything (V2X) communication has the potential to contribute both to better fuel efficiency in vehicles by its use in traffic management, but also to safety applications, leading to safer roads. Today there are two main technologies enabling V2X communication, namely Dedicated Short Range Communication (DSRC) and Cellular-V2X (C-V2X) [1]. These technologies enable V2X communication by supporting Day-1 basic V2X messages. Day-1 type messages are considered as vehicular safety applications that require an end-to-end transmission latency of about 100 ms; as long as the network load (vehicle density) is not too high. Moreover, in Europe, the V2X message types refer to Cooperative Awareness Messages (CAM), such as collision warnings and Decentralized Environmental Notification Messages (DENM). These are specified in the ETSI EN 302 637-3 standard, see [2]. The mentioned technologies and the corresponding message standards allow for vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication. A more in depth description of the mentioned technologies can be seen in section 2.1.1 and 2.1.2 below.

2.1.1 Dedicated Short Range Communication (ITS-G5)

DSRC technology is referred to as ITS-G5 in Europe and its ITS-G5 access layer with its corresponding physical- and data link layer is situated at the bottom of the ITS protocol stack. The ITS-G5 access layer is standardized according to the European standard ETSI EN 302 663 [3].

According to the ETSI EN 302 663, the ITS-G5 access layer technology is based on existing standards. The data link layer is divided into two sublayers; logical link control and medium access control layer. The logical link control layer is based on IEEE/ISO/IEC 8802-2-1998, while both the physical layer and the medium access layer are based on the IEEE 802.11p-2010 standard. The protocol of 802.11p was specifically made to allow wireless communication between vehicles outside the context of a Basic Service Set (BSS), allowing direct V2V communication in an ad hoc network. This type of communication requires a predefined frequency channel. The European Commission has allocated the 5.9 GHz (5855- 5925 GHz) frequency band for ITS non-safety- , ITS road safety- and future ITS applications. The Physical layer (PHY) in 802.11p is OFDM, further detailed in IEEE 802.11-2016, and allows for a 10 MHz bandwidth and a data rate of 3-27 Mbps at the 5.9 GHz frequency band. The medium access layer (MAC) in 802.11p, further detailed in IEEE 802.11-2016, utilizes the Enhanced Distributed Coordination Access (EDCA) algorithm. This is based on the basic Distributed Coordinates of Function (DCF) but adds Quality of Service (QoS). DCF is referred to as a Carrier Sense Multiple Access with Collisions Avoidance (CSMA/CA) algorithm which deals with collision of nodes in the network. Furthermore, the ITS-G5 standard also includes features for Decentralized Congestion Control (DCC) methods according to the ETSI TS 102 687 standard. DCC is an algorithm making sure that a single ITS station does not consume all the resources on the channel. For a more detailed description of ITS-G5 or for references to all the mentioned standards, the reader is referred to ETSI EN 302 663 [3].

ITS-G5 is able to communicate up to a distance of 1000 m. In [1] it is mentioned that studies have shown that DSRC technology based on IEEE 802.11p can reliably meet the requirements of Day-1 applications. The network load problem is also mentioned in a study conducted in [4] implying that IEEE 802.11p's CMSA/CA algorithm in the MAC layer has issues with high number of collision occurring during high network load leading to channel congestion inducing higher latencies in the network. Furthermore, the paper in [5] states that DSRC technology cannot always support high data rate transmission and reliable connection due to limited communication range and bandwidth. The paper in [1] concludes that the current state of DSRC technology cannot meet the QoS requirements of more advanced V2X applications such as remote driving and vehicle platooning.

2.1.2 C-V2X

C-V2X technologies combine direct communication between vehicles outside of the context of cellular coverage with existing cellular infrastructure technology to increase mobility, coverage and high network capability. This allows for V2V communication, V2I communication, vehicle-to-network (V2N) communication and vehicle-to-pedestrians (V2P) communication.

The evolution of Cellular-V2X technology can be divided into several stages. 3GPP released a standard in 2017 named Rel-14 that support basic V2X services such as CAM and DENM messages [5]. In Rel-14 C-V2X technology communicate through two different transmission modes, C-V2X sidelink mode 3 and 4. Mode 3 utilizes the LTE-UU air interface, corresponding to the traditional air interface between user equipment (UE) and an eNodeB. Mode 4 employs the PC5 air interface which allows direct communication between vehicles outside of cellular coverage. These two interfaces and modes are further explained in the papers [1] and [5]. Another standard has been conducted by 3GPP in Rel-15 which introduces support for more advanced use cases such as remote driving and platooning. Moreover, Rel-15 is backward compatible with Rel-14 [5]. The existing ready-to-purchase C-V2X modules today are Rel-14 compatible.

A comparison field test between DSRC- and C-V2X technology conducted by the 5GAA Automotive Association in [6] show that today's C-V2X technology outperforms DSRC technology in a number of areas including Line-of-Sight range, Non-Line-of-Sight blockage and Non-Line-of-Sight Intersection. The paper in [1] states that studies has shown that today's C-V2X technology reliably supports Day-1 applications, but that the performance in a high density vehicle scenario deteriorates, just as in the case with DSRC technology mentioned in section 2.1.1. The paper further explains that 3GPP started to work on a new evolutionary New Radio (NR) V2X standard for Rel-16 which will adopt the 5G NR technology introduced in Rel-15 but will support even more advanced use cases that require more stringent QoS guarantees, such as autonomous driving. The Rel-16 standard was finished in June 2020 [7]. However, it will take time before Rel-15 and Rel-16 compatible C-V2X technology hit the market.

2.2 Existing ITS-G5 Solutions

Any type of communication with radio waves relies on having two things, a transmitter and a receiver. The job of the transmitter is to take the information provided to it, modulate it, and send it out into the world with the help of antennas [8]. Modulating information into a signal can be done in several ways, which will affect qualities such as the transfer rate of information. ITS-G5 has eight different modulation schemes which affect the transfer rate of the data, as

well as the sensitivity needed for the receiver. The possible range of transfer speed rate is 3-27 Mbits/s [9], and the modulation can either be done by hardware components or by software.

Commercially available ITS-G5 transceivers, such as the UNEX SOM-301E[10], are too expensive (750$) to meet the cost requirement for the project. As such, two cheaper solutions were investigated. The following two sections explain them independently.

2.2.1 WiFi-based Radio

Because ITS-G5 is similar to other WLAN protocols, a group of researchers have been able to develop solutions for cheaper ITS-G5 radio devices in [11],[12]. The idea is to use WiFi network cards and have them operate according to the ETSI ITS-G5 standard [3]. This modification requires a computer with OpenWRT [13] as its operating system and a WiFi network card which can run the ath9k [14] Linux kernel driver. As these are the only requirements, the solution can be recreated using cheap hardware in comparison with the commercially available modules.

This solution has been tested and verified by researchers in Italy [15] who later published their work for others to use and build upon on GitHub. Their GitHub repository is called OpenWrt-V2X [16] and includes a pre-compiled image of their patch, ready to be flashed onto an OpenWRT compatible device. The image is specifically designed to be used in x86 64 bit systems which is something to consider when selecting hardware. It might be possible to change that and make it compatible with cheaper hardware.

OpenWRT is an operating system based on Linux that is aimed at embedded devices such as routers [13]. The OS enables fine-tuning features that are needed to use WiFi cards outside of their marketed operational capabilities. OpenWRT also enables users to change the transmission power, or TX power, for the network cards, which regulates how much energy the antenna sends out as radio waves and thus controls the transmission range. The study in [17] found that increasing the TX power comes at the cost of the TX rate. This means that there is a trade-off between the amount of data that can be transmitted between clients, such as speed, position, collision-warnings etc., versus the desired range between the clients.

2.2.2 Software Defined Radio

Software defined radio (SDR) is a relatively new invention, with its origin only dating back to 1987 [18]. Because of this, the market is not quite as developed as for traditional radio, but the demand is rising. Different sources for market research cite different market sizes for the SDR

market, but they all predict the market to grow in the following years [19][20][21]. SDR systems use software for the critical task of modulation and demodulation in the devices. This comes with both pros and cons. Since everything is controlled by software, it is easy to reconfigure and expand the functionality of an SDR as one simply has to change the software on the device. A disadvantage with using an SDR system is that since all the modulation is done digitally, a lot of computational power is needed. This can create a bottleneck which limits the throughput of the device [18].

Developing software for a SDR can be a daunting task. Thankfully, there are development tool-kits available to ease the load. One of them is GNU Radio which is both free and open-source. GNU Radio provides an easy to use environment with processing blocks and a graphical interface. It is used by everyone from hobbyists to industrial companies and there is an active community with a lot of help to be used [22].

A framework for a IEEE 802.11p (which is what the ITS-G5 protocol is based on) transceiver has been developed using GNU Radio and released under a Open Source licence. This transceiver is possible to implement on several SDR platforms, such as the HackRF One and the bladeRF. When using an Ettus Research N210 as the SDR platform, the transceiver was able to produce comparable results to a Cohda Wireless MK5 (which is a commercially available ITS-G5 unit) in regards to receiving and transmitting standard IEEE 802.11p frames [23].

2.3 Existing Solution OBU

The earliest OBUs were mainly used in Electronic Road Pricing System in order to decrease the congestion of traffic. Then it was gradually integrated with more functions, and satisfying the need of smart transportation and self-driving.

Siemens has Sitraffic OBU that integrates vehicles into cooperative ITS systems (CITS) and the deployment of infrastructure-driven C-ITS services and applications [24]. It can send hazard alerts to service vehicles or prioritize public transport and emergency vehicles, and enables road users to adapt their driving behavior at an early stage, thus increasing safety for pedestrians and cyclists. But the Sitraffic OBU only has ITS communication, and no WiFi module, so it does not support a large throughput such as camera or LiDAR data tranmission.

The C-ITS OBU from Q-Free, which is a new company from Norway, is integrated with other apps and third-parties thanks to open standards interface, and can connect to vehicle network via WiFi or Bluetooth [25]. Their products are mainly focused on Road Pricing System.

Unex has a series of OBU products based on ITS-G5, DSRC or C-V2X[26]. For example, OBU 301E offers multiple connectivity options to provide a rapid way to develop V2X applications and use-cases, such as on-board mPCIe socket that allows add-on LTE, WiFi, or Bluetooth. They could also provide ready-to-use V2X communication services with V2Xcast software. However, the communication signal frequency cannot satisfy the needs of this project, and Unex is too expensive, meaning this option has to be rejected.

2.4 Existing Solutions RSU

The Intelligent Roadside Unit by NXP [27] is an RSU designed to see and communicate with autonomous vehicles through V2X communication as well as non autonomous vehicles and pedestrians using traffic lights. Their goal is to schedule all traffic, autonomous and non autonomous, to reduce waiting time. It uses multiple radar and camera modules to recognize both vehicles and pedestrians, and a software defined radio processor for the V2X communication. A key difference compared to the AD-EYE RSU is that it does not utilize edge computing. It is specifically made to schedule the traffic at intersections, and thus has comparably limited computing power. However, overall it is similar to the one this project will develop in both size and function. Thus, some valuable comparisons can be made.

Another RSU comparable in size, and in some ways function, is the pole mounted server platform Outdoor Edge Systems (OES) by Supermicro [28]. OES has capabilities to be used for edge computing, as a 5G radio access network or for localized data storage, but the unit itself does not include the hardware necessary. It is instead a modular housing for hardware, made to function reliably under virtually any circumstances. The methods used in this RSU for both hardware and housing are advanced and expensive, but can likely still give some insight into possible RSU solutions.

There are plenty of RSUs using smaller, compact boxes. As the size of the required hardware for this project makes this approach impossible, these constructions can be largely ignored when it comes to the design of this project's RSU.

2.4.1 Enclosure Cooling Solutions

Air conditioning with heat pumps is sometimes used for roadside units. As the heat pumps make it possible to cool the air below the outside temperature, this makes for more powerful cooling. This can be necessary for RSU applications with large power consumption, like server platforms,

or ones that run in high temperature areas. This is an expensive solution not well suited for small projects.

A similar but more cost effective alternative is air conditioning without active cooling, simply forcing airflow through the enclosure using fans. By having the inflow of air far down on the enclosure side and the outflow higher up, this can ensure an inflow of cold air and an outflow of warm air. As a result the cooling capacity becomes quite effective.

In areas with high humidity or with other characteristics making the outside air possibly damaging to the inside of the RSU, it can be relevant to use cooling without airflow through the enclosure. One solution for this is using a heat exchanger. With this solution, outside and inside air are circulated through a heat exchanger, causing the outside air to cool the inside air without mixing. This is an expensive solution used for specific circumstances.

Another solution without airflow through the enclosure is a liquid cooling system. With liquid cooling, a liquid is passed through a tube in a path around the areas to be cooled, where the liquid absorbs heat and is then passed through a cooling system. At this point, the liquid releases heat. This is a rather expensive solution, and because of the risk of leaks inside the enclosure this method is seldom used in RSUs or other industrial purpose hardware enclosures.

2.4.2 Enclosure Heating Solutions

A cheap solution to ensure the necessary hardware is kept above minimum operating temperature is to use a digital thermometer and heating pads. The heating pad is simply a metal wire with its resistance evenly distributed throughout a non conductive pad, so that an applied current causes evenly distributed heat. These heating pads could be attached to hardware with a requirement for minimum operating temperature that is turned on when the digital thermometer reads temperatures close to the minimum.

A more expensive but certainly more robust solution is to use a compatible thermostat and enclosure heater available through online vendors. The thermostat can then make the heater start when the temperature drops too low. This will also be helpful to prevent condensation inside the box.

2.4.3 Uninterruptible Power Supply

An uninterruptible power supply (UPS) can be used to run a unit a short time on battery in case of a sudden power loss. It works similar to a laptop battery, where the laptop switches to battery

power if the supply from the grid cuts out. A benefit with the UPS is the simplicity. It is plugged into the wall plug and then the computer is plugged into the UPS. When the electricity is on the battery is charged and when the power cuts out the battery powers the computer. A UPS typically has a built-in battery with capacity of about 7 Ah [29]. To supply the RSU with power for the required hour long period, the capacity would have to be at least around 100 Ah with 12 V. To use a UPS for this application, the built-in battery would have to be replaced with a much larger one.

2.5 Wireless Sensor Data Transmission

The throughput requirement for the Camera and LiDAR transmissions cannot be met by ITS-G5 and therefore another transmission channel is needed between the RSU and the OBU for heavy data transmission, where cellular and WiFi ac/ax were feasible options to examine.

2.5.1 WiFi

WiFi or Wireless Fidelity is a radio transmission technology used for transmitting data wirelessly to connected devices through routers and wireless access points [30]. WiFi protocols are one of the most popular wireless communications for transmitting large amounts of data. When the OBU and RSU are connected to the router, they will be able to communicate with each other through a unique IP address, which the router creates, for the local area network. First WiFi was invented and released for consumers in 1997, which led to the creation of IEEE802.11 [31], and since then, WiFi has significantly improved from 2 Mbps link speeds for WiFi 1 (802.11a), up to 9.6 Gbps for WiFi 6 (802.11ax) [32]. As the range is the trade-off for WiFi in V2X communication, it has advantages over cellular solutions with its low latency for image transmission as is concluded in a study comparing WiFi 11.ac with 4G LTE cellular for V2X [33]. In the study they also measure the latency in a mobile condition for speeds under 20 m/s and the WiFi had more stable performance while 4G LTE had bigger variations in latency and performance degradation when the speed increases. Data from the experiment can be seen in table 2.1, which is end-to-end latency of sending two images, 91 kB and 401 kB, from RSU to OBU, and from OBU to RSU, using WiFi 5 and 4G LTE [33].

Table 2.1: End-to-end latency from RSU to OBU and from OBU to RSU taken from A Comparison of Communication Mechanisms in Vehicular Edge Computing [33].

RSU -> OBU (ms)	LTE	WiFi
Image 1	6174.32	**1733.90**
Image 2	20220.85	**1741.07**
OBU -> RSU (ms)	LTE	WiFi
Image 1	6012.8	**1301.38**
Image 2	21876.4	**1293.25**

2.5.2 WiFi 5 and WiFi 6

WiFi 5 (802.11ac) and WiFi 6 (802.11ax) are the latest IEEE802.11 standards respectively which have brought significant improvements in speed and efficiency. Both WiFi 5 and WiFi 6 support MU-MIMO (multi-user, multiple input, multiple output) which allows access points to send up to four streams for WiFi 5 and eight streams for WiFi 6 simultaneously. However, WiFi 5 operates in the 5 GHz frequency only and is limited to download link transmissions, where WiFi 6 operates in both 2.4 GHz and 5 GHz. It can also create MU-MIMO connections with download link and upload link MU-MIMO so that an access point can transmit concurrently to multiple receivers, and an endpoint is able to receive from multiple transmitters simultaneously. WiFi 6 promises up to 40% potential speed benefits compared to WiFi 5. The market is still in its early adoption phase for 802.11ax, but WiFi 6 technology is compatible with WiFi 5 in case the hardware does not support WiFi 6, and so both WiFi 5 and WiFi 6 are good candidates for heavy data transmitting and receiving from multiple on board units (OBU) [32],[34].The table 2.2 shows a complicit comparison between WiFi 5 and WiFi 6 [35]. The numbers in the table however are the maximum theoretical data rates and cannot be expected in reality. Benchmark tests are required in order to get more realistic data rates. However, finding benchmark test from reliable sources that is applicable to this project turned out to be difficult.

Table 2.2: WiFi 5 and WiFi 6 comparison from Microwaves and RF [35].

Parameter	Wi-Fi 5 (802.11ac)	Wi-Fi 6 (802.11ax)
Frequency	5 GHz	2.4 and 5.0 GHz
Bandwidths (channels)	20, 40, 80+80, 160 MHz	20, 40, 80+80, 160 MHz
Access	OFDM	OFDMA
Antennas	MU-MIMO (4 × 4)	MU-MIMO (8 × 8)
Modulation	256QAM	1024QAM
Maximum data rate	3.5 Gb/s	9.6 Gb/s
Maximum users/AP	4	8

2.5.3 Cellular Network

The other option for transferring media data wirelessly is through cellular network or mobile network, which is the most common connectivity method used with cell phones, smartphones, and dial-up devices. These devices connect through radio antennas to fixed-location transceiver stations, towers and/or satellites, which covers predetermined land areas called cells. The stations provide the devices, within its cell area, with network coverage for data transmission, and even when the connected devices move from a cell area to another, they still maintain their connectivity. Cellular range has an obvious advantage over WiFi range, but it can also be a cheaper one in terms of power consumption, because cellular has less power dissipation than WiFi. The same comparison study introduced in WiFi subsection 2.5.1 has concluded that the memory footprint and CPU utilization of 4G LTE cellular are less than the WiFi one for image transmission [33].

2.5.4 4G, 5G Cellular Standards

4G and 5G standards are the latest exponential leaps to provide the highest quality, and most reliable communication with minimum latency. The key technologies that have made 4G and 5G possible are MIMO and OFDM through standards like LTE in 4G. The theoretical speed of 4G reaches up to 1 Gbps with theoretical latency of 200 milliseconds. 5G on the other hand uses millimeter waves (mmWave), which means 5G signals can carry a lot more data much faster than 4G signals with ultra low latency. 5G can have a theoretical speed up to 20 Gbps for downlink and 10 Gbps for uplink with theoretical latency as low as one millisecond and is

estimated to be 10 to 20 times faster than the average 4G latency. However, shorter wavelength in 5G also means shorter range because any obstacle such as a tree can block 5G signals.

Even though the 5G theoretical data rate is high, those numbers cannot be expected in reality. A global benchmark 5G study conducted by Opensignal in [36] list the top 10 countries with the fastest download speed and upload speeds. The average download speeds range from 162.9 Mbps up to 361 Mbps with the corresponding highest peaks ranging from 573.6 Mbps up to 863 Mbps, while the average upload speeds range from 21 Mbps to 36.7 Mbps. Sweden was not a part of this study, but it still provides an overview of how well 5G technology perform in its current state globally. Implementing an open-source cellular software that includes the needed modems on a standard Linux-based machine to provide 4G or 5G connections to nearby vehicles could be a solution for transferring media data between the RSU and OBU and vice versa [33].

In addition to a low latency as previously mentioned, the LTE technologies can provide a substantial benefit in reliability compared to the 802.11 standards, as shown in Figure 2.1 from the study [37]. It can be seen that the packet receival ratio for LTE technologies degenerates over distance at a much slower rate.

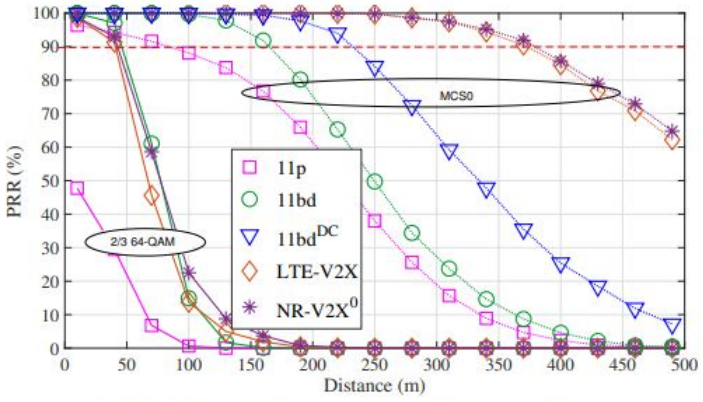


Figure 2.1: Packet reception ratio for 1500 bytes packet [37].

2.5.5 WiFi 6 vs 5G Cellular

A study was published on ScienceDirect [38] comparing WiFi to 5G seen in table 2.3, where the most crucial aspects of both WiFi 6 and 5G Cellular were included such as data rates, range, channel bandwidth and costs. However, this table only show theoretical values and might not be met in reality.

Table 2.3: WiFi 6 and 5G Cellular comparison from a study published on ScienceDirect [38].

Variable	3GPP 5G	Wi-Fi 6/Wi-Fi 6 E
Peak data rate	2 Gbps (DL), 1 Gbps (UL)	10 Gbps 8x8 (DL), 5 Gbps (UL)
MU-MIMO	128x128	8x8
Coverage range	100–300 m for small cells, up to tens of km for macro cells	<50 m indoor, up to 300 m outdoor
Carrier aggregation	Yes	Yes, 40, 80, 160 (or 80 + 80)
Inter-cell interference	Controlled	Mainly uncontrolled
Channel Access Scheme	OFDMA	OFDMA
License type	Mostly licensed	Unlicensed
General bands	Low, mid and high	Low and mid
Specific frequencies	Low-band (<1 GHz), mid-band (1–7 GHz) and high-band (−24–29 GHz)	2.4 GHz, 5 GHz, 6 GHz, 60 GHz
Channel Bandwidth	20, 40, 80, 100 MHz	20, 40, 80, 160 MHz
Revenue model	Pre- or post-pay billing for data services	Either a service, 'free', amenity, or pure WLAN without external connection
User equipment price	High	Low
Public versus private	Traditionally publicly provided by an MNO	Traditionally privately provided
Chip/modem cost	High	Low
Data cost	Monthly subscription ($5–20)	Free ('piggybacks' on fixed broadband)
Deployment approach	Controlled and managed	Uncontrolled and mostly unmanaged
Installation skill level	High	Low
Development skill level	High	Low

2.6 Antennas

The most important characteristics when it comes to antennas is radiation pattern, polarization, power gain and directivity [39].

One type of antenna is the omni-directional antenna, also known as the dipole antenna. Omni-direcitonal antennas have a torus shaped radio pattern with its power radiated 360 degrees around the antenna, pictured in Figure 2.2. This type of antenna is commonly used in mobility applications [39]. The uniform signal pattern comes with the advantage of being able to receive signals from multiple directions but this also contribute to the disadvantage of possible interference with its surroundings.

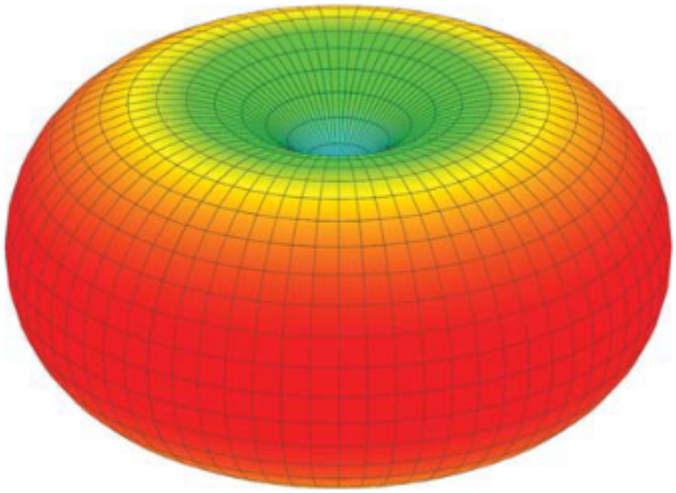


Figure 2.2: Radiation pattern of a omni-directional antenna [40]

Conversely, directional antennas use a more targeted coverage that, while providing the ability of extending the antenna range does limit the covered area in other directions. The benefits of the directional antenna is a greater radiation and reception of signals from the targeted direction, and reduced noise from interfering sources. Furthermore, directional antennas are usually considered as a group of antennas which send and receive signals in a rather narrow, focused beam. These are the antennas which can provide the greatest amount of one directional range. However, there are variations of directional antennas which are more aptly termed as sector antennas, which cover signals in a field of view (usually ranging from 60 to 180 degrees). This allows for certain compromises in coverage and range of radiated signals through using this type of antenna.

The area an antenna covers is directly related to the gain of the antenna. An antenna with 0dB gain is equivalent with a lossless antenna which broadcasts its signal equally in every direction (termed as isotropic). The more directed the antenna is, the higher gain it has. As such, omni-directional antennas generally has lower gain than directional antennas. Omni-directional antennas also differ in gain. This is due to the shape of their signal pattern that extends further in the horizontal plane than the vertical one, as well as leaving a blind spot directly above and below the antenna. High gain in an omni-directional antenna means that the signal will extend further in the horizontal plane in expense of less range in the vertical plane [41].

Different antennas have different polarization, which is the plane in which the electric field component of the signal is present. There are two main types of polarization, linear and circular. For a transmitter and a receiver antenna to connect, they have to have the same polarization. If the two do not match, a signal loss will occur [42]. For linear antennas, the orientation of the antennas also have to match. This is not required for circularly polarized antennas, that also

are better at dealing with multi-path interference [40]. The downside with circularly polarized antennas is that they are more complex than their linear counterparts, and are therefore more expensive.

2.7 Platooning Algorithms

Platooning is the idea of creating "linked" groups of vehicles that automatically follow a leading vehicle in a tightly spaced formation. An overview of large scale existing projects related to platooning can be seen in [43]. Furthermore, an overview of the technical details of platooning and existing literature is in [44]. Additionally, an overview of existing literature related to platooning is found in [45], where research is divided into separate categories.

To calculate the desired spacing between linked vehicles in platoons, two ubiquitous policies are used in research. The first is the constant distance policy, formulated in [46]. In this paper, the negative effect of communication delay on the minimum distance is also discussed. The second policy is of constant time to space vehicles in a platoon. A key difference between these policies is that the time spacing can retain string stability of a platoon even without the use of V2V communication, which is discussed and shown in [47]. However, the use of V2V communication yields additional benefits such as a reduced time headway, and increased safety.

The stability of the platoon requires both internal stability of each vehicle, and also the string stability of the entire group of vehicles. This can be condensed to the idea that errors shall not propagate through the platoon. A generic formulation of string stability in platooning is formulated in [48]. Furthermore, dynamics of vehicles are an important consideration when evaluating stability and designing control schemes for platoons. The dynamics of a platoon model are considered in [49] where it is found that the coupling between two vehicles can be modelled as two springs and a damper. In this report, it is also recommended to accelerate platoon vehicles individually, and to decelerate uniformly. Additionally, the dynamics of a differential drive wheeled robot are instead considered in [47].

A platoon contains one leading vehicle as well as n following vehicles, where $n \in \mathbb{N}$. Generally, the lead vehicle will be driven whilst the following vehicles use longitudinal and lateral automatic control to follow at the desired distance. A simplified study which considered only longitudinal control used a PID algorithm for platooning in [50]. A more modern approach is to use a variation of Model Predictive Control (MPC). Some core variations of the MPC method include the Look Ahead Control (LAC) which incorporates knowledge of future disturbances, as well as the distributed MPC (DMPC) which attempts to split the overall optimization problem into local MPC problems. This is done to reduce communication efforts, and computational complexity.

Both these variations are discussed in [51]. In [52], the method of DMPC is applied together with clustering, to alternately cluster platoon members into two groups. An optimization was then solved for each group respectively. This method attempts to add more flexibility to platooning, allowing vehicles to join and leave more easily. Moreover, the use of a DMPC scheme coupled with additional robustness constraints are outlined and compared to previous DMPC methods in [53]. Furthermore, a control matching model is used in [54], where an $\mathbb{H}_\infty$ controller is used to attain string stability, and an MPC controller which achieves given constraints. The MPC controller is then matched to the $\mathbb{H}_\infty$ controller. A novel optimal control scheme is also introduced in [55], which offers several benefits such as compensating for time delays in communication, and accommodating heterogeneous platoons.

The use of V2V communication can become unstable in some platooning applications, and a main reason is due to packet losses. Packet losses are induced by collision of packets [56]. An analysis of the stability of platoons with regards to packet losses is included in [57], along with a discussion on the bounds of the packet loss probability to retain stability. An adaptive control method is also introduced in [58], which switches between control schemes dependent on communication network status. However, collision of packets can also be reduced by controlling the rate of vehicle beaconing. The standard approach to V2V communication has been the static rate beaconing, shown in [46]. There are adaptive methods which attempt to limit communication rate by using event driven beaconing. Two such methods are outlined in [59, 60], and are experimentally shown to outperform the static rate method.

Chapter 3

Methodology

3.1 Project Management

This section outlines management elements of the project.

3.1.1 Organization

The project team consist of nine team members. In the beginning of the project several research topics were identified by the help of the ALM tool Polarion. Based on the findings in Polarion, the project team was then divided into three subgroups; one responsible for the RSU, one responsible for the controller and platooning algorithm and one responsible for the ITS-G5 implementation. These subgroups were established in the spring, during the research part of the project and was kept during the autumn. The members of the three groups were switched between spring och autumn in order for everyone to acquire experience and learn about different areas. The project team had two recurrent team meetings every week, one with the project coach and one with the stakeholder. During these meetings, research findings were presented, solutions were discussed and requirements for the project were refined.

3.1.2 Communication

For communication, the group used three different channels. This included a Messenger group for internal communication within the group, a permanent zoom channel for easy coordination of weekly meetings, and a mailing list for traceable communication with actors outside the

group. Other tools used in the project were Google Drive and GitHub. Google Drive was used to store and organize files so that everyone easily could access them all the time. GitHub was used for version control of code used in the project.

3.1.3 Budgeting and Costs

To keep track of costs associated with the project, a bill of materials was created and shared online with all members. This bill was passed to the stakeholder whenever new costs or components were suggested that needed approval. Moreover, feedback and changes were also kept in this document.

3.1.4 Time Plan

A Gantt chart was created, which covers the tasks of the project through the spring and the fall. This chart is included in section D of the appendices.

3.2 Engineering Approaches

A requirement based approach was used for the duration of the project. Since the project spanned several mechatronic domains, the validation of these requirements generally could be applied as tests with different levels of modularity. This fits into the V-model approach (see Figure 3.1), where the system design is layered according to the requirements of specific systems or components. Moreover, the system integration is performed as tests in the form of unit tests, integration tests, as well as system tests.

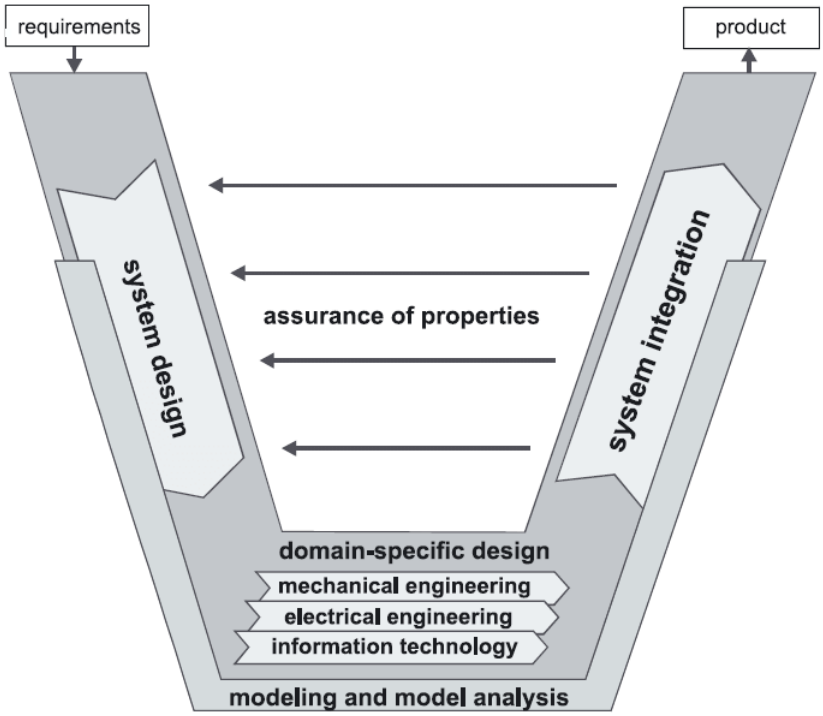


Figure 3.1: V-Model according to guideline VDI2206 [61].

3.3 Development Process

A large part of the project was deciding components. For each component decision several different alternatives were investigated and their relative pros and cons compared. These alternatives were then presented to the stakeholder during the weekly meeting. The stakeholder then chose the alternative that they preferred.

3.4 Engineering Tools

This section outlines the different tools employed in this project, first going into physical engineering tools and then software tools.

3.4.1 Physical Tools

Workshop tools have been used in the project for construction and electrical testing. These were mainly (but not limited to) pillar drill, water cutter, 3D printer as well as multimeter.

3.4.2 Software Tools

Third party software and libraries have been employed throughout the project as tools to solve tasks. These are presented in list format below.

1. **Robot Operating Software (ROS):** Open source robotics middleware with a collection of software libraries.
2. **Prescan:** Physics-based simulation platform used in the automotive industry for development of Advanced Driver Assistance Systems
3. **Matlab & Simulink:** Matlab is a programming and numeric computing platform, while Simulink is a software package for modeling, simulating, and analyzing dynamic systems.
4. **Solid Edge:** Software program for 3D Computer Aided Design (CAD) and solid modeling.
5. **Simcenter FLOEFD:** A 3D computational fluid dynamics analysis solution that can be used with Solid Edge.
6. **Ansible:** Open source IT automation system.
7. **GNU Radio:** Open-source software development toolkit that provides signal processing blocks to implement software radios.

Chapter 4

Implementation

4.1 Final Implementation OBU

The initial idea was to implement two On Board Units (OBU) that would be mounted ontop of the TurtleBots. These OBUs would have the capability to communicate through ITS-G5 between vehicles and the RSU as well as retrieve and pass forward camera and LiDAR data from the TurtelBots to the RSU. However, the OBUs did not end up getting developed in this project. The main reason for this was the stakeholders requirement of using Ubuntu 16.04 and ROS kinetic throughout the system. ROS kinetic, which is a part of ROS 1, does not support a distributed system with communication between several ROS cores. And since the implementation of separate OBU units would require several ROS cores throughout the system, it was decided together with the stakeholder that those were not to be implemented. Therefore in this project the TurtleBots can be seen as the OBUs. However, a more in depth discussion covering two different concepts of how separate OBU units could be implemented in the future can be seen in the future work section, see section 8.1.2.

4.2 Final Implementation RSU

The purpose of the Roadside Unit (RSU) in intelligent Transportation Systems (ITS) is to process all the sensor data and do all the necessary computations in order to improve traffic flow, safety and response time. The On Board Unit (OBU) does not have the computing power to analyze the sensor data, it should simply send the raw data forward to the RSU and receive commands

from the RSU to control the actuators. It is designed for outdoor use with IP54+ environmental protection.

4.2.1 RSU Edge Computation Capabilities

The main limiting factor for the computer hardware is the budget. For the motherboard, a full ATX, ASUS motherboard was chosen. It is of type ROG STRIX Z590-E GAMING WIFI. The board has an extra PCIe x16 to allow expansion of the GPU, and 4 DIMM 288 connections for the RAM[62]. It also has 6 USB ports which are required for extension of sensors. For the RAM, two 8 GB modules with 3200 MHz were chosen rather than one 16 GB as this allows higher performance. For the CPU a 10th generation Intel Core i7 10700K with 8 cores was opted for. As the RSU is not required to store much data, a small hard drive of 120 GB is used. It was decided to use a solid state drive (SSD) as it is faster, more durable and more reliable than a regular hard drive and can offer better performance.

4.2.2 Motherboard Compatibility

The motherboard used for the RSU is an ROG STRIX Z590-E. It was chosen for its compatibility with the CPU and because it has two Ethernet ports, which was necessary. An unforeseen problem with this motherboard was that the Ethernet ports are of 2.5 Gbps variant. This is a relatively new standard which was not supported by Ubuntu 16.04. In the end the only way to resolve this issue was to upgrade the kernel to version 5.1. Ubuntu 16.04 normally only supports kernels 4.4 to 4.15, but it is possible to upgrade further manually. Doing this is generally not recommended as an untested kernel might bring convoluted bugs. One of these bugs we encountered was that if an Ethernet cable was connected to an Ethernet port during startup of the computer, that Ethernet port could not detect any devices until it was restarted. This problem was solved by running automatic shell scripts that restarts the Ethernet ports after startup of the computer. The process for creating this script can be found in appendix E.3.
The SSD was not detected to begin with. It was later found that only one of the SSD slots on the motherboard is compatible with the SSD used.

4.2.3 RSU Enclosure

The initial decision regarding the RSU enclosure was to either build a suitable box specifically for this project or to order an IP54+ box online. It was quickly decided that the best option for

this project would be to order a box online. Not only would the quality of a professionally built IP54+ certified box no doubt be better than anything the group could build, but it would also save a lot of time. A steel control cabinet by Rittal with IP66 [63] rating was opted for. These are made for outdoors and are more than sufficiently robust to weather, hits and vibrations. They come with tools for mounting hardware inside the enclosure, rails for the sides of the box and a back plate. It was also decided to order a standard computer box to allow work on the computer hardware/software and the RSU enclosure hardware simultaneously. This allowed the team to assemble the computer parts and get started on the software quickly.

4.2.4 RSU Temperature Regulation

For the RSU to operate without issues in standard Swedish climate, roughly in the temperature range of -15 to 35 degrees Celsius, it requires temperature regulation through a cooling and heating mechanism. The solution opted for to cool the enclosure during the warm periods is air conditioning without active cooling. This was chosen as it was the most cost efficient of the possible solutions analyzed in the SOTA, section 2.4.1. For this solution, two IP54 rated air filters, one of them with an attached fan, will be placed on the side of the RSU. As mentioned in the SOTA, to maximize the cooling effect, the outflow of air will be placed high up on the side of the enclosure and the inflow lower down, ensuring that the inflow is as cold as possible and the outflow as warm as possible. As the calculations of appendix A show, using the selected cooling fan the and running the RSU at maximum power capacity during the worst case outdoor temperature of 35 °C, the temperature inside the RSU would be 41.4 °C. Although this temperature is a bit high, and might cause reduced efficiency in the CPU and GPU or decreased lifespan of some hardware, it is acceptable as a worst case scenario.

For the heating element the most cost efficient solution analyzed in the SOTA, Enclosure Heating Solutions 2.4.2, was opted for a compatible enclosure heater and thermostat. The heater selected is a 150 W Pfannenberg Enclosure Heater. As the calculations of appendix B show, the minimum temperature in the enclosure when the RSU is running idle, at worst case conditions, will be 0.3 °C. Although a lot of the used hardware does not specify a minimum temperature, 0.3 °C should be fine for computer hardware.

As the temperature inside the RSU will always be above the outside temperature, condensation will most likely not be an issue, even if humidity is high. The air that flows in will be heated up, its water carrying capacity increased, and thus its relative humidity reduced, making condensation unlikely if not impossible. If the RSU is standing idle at a temperature where the heater is off, outside humidity is high and the outside temperature is falling rapidly, then condensation could occur. To avoid this specific scenario a hygrometer could be installed in the RSU to activate the

heating element at high humidity, although this might require the whole heating system to be digitized. Having the heat element by the inflow, thus heating the air upon entry, further reduces the risk of condensation.

4.2.5 Placement of Heater

The heater is used to prevent condensation and keep the temperature in the enclosure at an appropriate level. It can be mounted in any position, but should preferably be placed vertically and in the lower part of the enclosure for best heat dissipation. All components and cables must have at least 50 mm clearance around the heater. The minimum clearance around the induction and exhaust area is 100 mm[64]. As space in the enclosure is limited, the vertical placement preference is not possible to fulfill.

4.2.6 Battery Powering

The RSU will most of the time be connected to the power grid. However, it should be able to run on battery for at least an hour for tests were access to the power grid is not available. This is made possible by using a 12 V car battery and a 12 VDC to 230 VAC inverter. The battery has a capacity of 100 Ah at 12 V, and with an inverter with efficiency above 90% [65], this gives a theoretical maximum power delivery of above 1080 Wh. The battery is placed in the bottom of the enclosure for increased stability, it is then fastened with straps as seen in Figure 4.1. As the battery is designed for a car, it works well in the required operating temperature, with a starting capacity of 900 CCA(EN) at -18 degrees [66]. The final product would always be connected to the power grid, as the use of battery power is intended for the prototype.

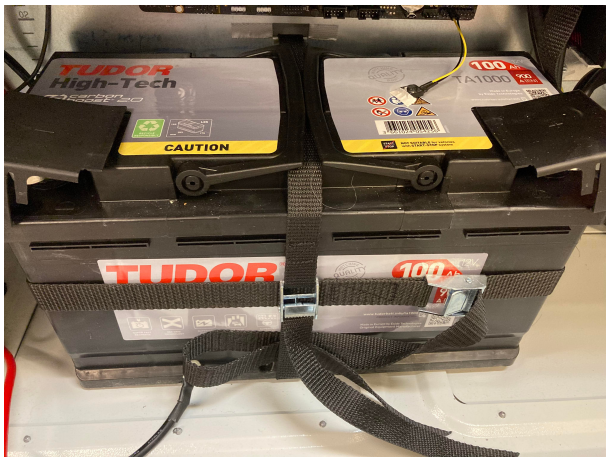


Figure 4.1: Battery mounting

4.2.7 Theoretical Battery Time

The lowest theoretical battery time would be in a scenario where all of the computer components are running on maximum power rating, the heater is off, and the fan is on. The reason the heater is not considered to be on is that even if the temperature outside is -15 °C, the RSU would reach a temperature above 0 °C without the heater. The max power rating for the components can be found in table 4.1.

Table 4.1: RSU component ratings

Component	Max wattage [W]	Source
CPU	250	[67]
GPU	180	[68]
Motherboard	100	[69]
RAM	30	[69]
SSD	25	[69]
PSU (efficency loss)	65	[70]
Heater (not needed in this calculation)	150	[64]
Fan	50	[71]
SDR	4,5	[72]
Router	10	[73]
DC-AC inverter (efficency loss)	86,6	[74]

The total amount of power needed in this worst case scenario is summed up to 786,6 W. We can get the maximum amount of watt hours, the battery with the formula:

$$\text{Watt hours} = \text{Voltage} * \text{Ampere hours}. \tag{4.1}$$

With the current battery the maximum amount of watt hours is 1200.

By dividing the available watt hours with the maximum power consumption the battery life the battery life for this worst case scenario is calculated as 1,53 h. In reality, the battery should not be fully drained. Even in this worst case scenario, there is a buffer against this.

4.2.8 DC-AC Inverter

In order to power the RSU with the 12 V car battery, a 12 VDC to 230 VAC inverter is used. This inverter increases the voltage and generates a pure sinusoidal wave, needed to power delicate electrical equipment on alternating current. It can supply a continuous output power of 1000 W

and a maximum output current of 8.7 A. The inverter has an efficiency over 90% [65]. It has built in protection against overheating, overloading and short-circuiting, as well as an inbuilt earth circuit breaker.

4.2.9 RSU Feet

The weight of the RSU is about 85 kg including the battery. The RSU should be able to be moved by one person using a trolley. To ensure this, it will stand on four feet with adjustable height and ability to support a static weight of 500 kg each. The height should not be higher than what is needed to load it on the trolley, as higher feet introduces instability. Another procedure made to increase stability is that the feet are mounted on steel beam supports as seen in Figure 4.2 to get the feet further away from the center of gravity for the enclosure. Each support is mounted with four M8 screws using rubber washers to keep the IP rating.

Figure 4.2: Mounting of RSU feet

4.2.10 Support for Power Supply Unit

The weight of the power supply unit is supported by a sheet metal piece mounted with four M6 screws in the back plate. The power supply unit rests on this support without any screws, but is fastened with straps as seen in Figure 4.3

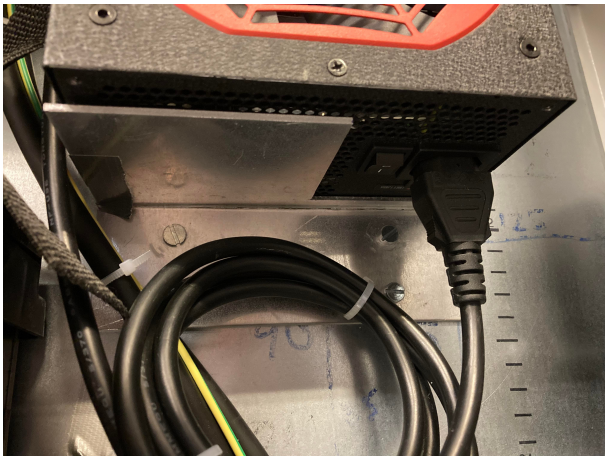


Figure 4.3: Support for power supply unit

4.2.11 Support for Graphics Processing Unit

It is crucial that the GPU is fastened properly. The unit is mounted on the motherboard, which is attached to the back plate. The GPU is perpendicular to the back plate. If no additional support is used, the whole weight of the unit is supported by the connection to the motherboard. The GPU is a sensitive component, especially the connections, which implies that an additional support must be used. The support used is shown in Figure 4.4. The support is mounted with two M6 screws on the back plate and the GPU is fastened to this support with a cable tie.

Figure 4.4: Support for GPU

4.2.12 Mounting Rails

The DC-AC converter, heater and WiFi router is mounted with mounting rails as seen in Figure 4.5. These are attached on the sides of the enclosure. They are fastened with the screw seen in the top right of Figure 4.5. When the screw is tightened, the rail expands, locking it in place in the enclosure.

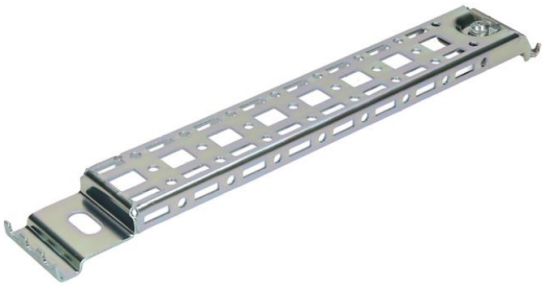


Figure 4.5: Mounting rail [75]

4.2.13 Support for the Power Switch

The support for the power switch used to switch between battery power and grid is 3D printed with polyactide and is shown in Figure 4.6 It is mounted on the back plate with M4 screws. The support has holes where the cables are inserted, adjusted to fit the cables. There are holes from the top of the support with a tight fit of M3 nuts. A 3D printed lid that the switch is attached to is mounted with the help of M3 screws and a nut inside each hole. The lid is mounted in this way to avoid having screws inside of the support that are conductive. A potential approach was to mount the switch with long screws from the switch to the back plate and isolate these screws with melted plastic. This approach was discarded because it seemed unsafe.

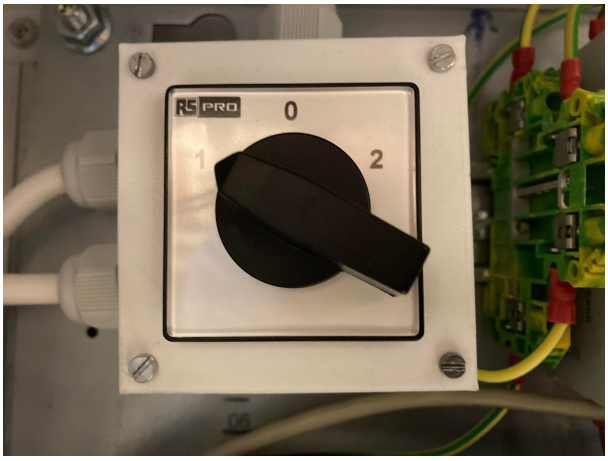


Figure 4.6: Mounting of power switch

4.2.14 Support for WiFi Antenna

The support for the antenna is 3D printed. The purpose of this support is to avoid drilling in the enclosure. It will be attached to the enclosure with adhesive. The cable to the antenna will be extracted through the cable passage on the side of the enclosure. The support is seen in Figure 4.7

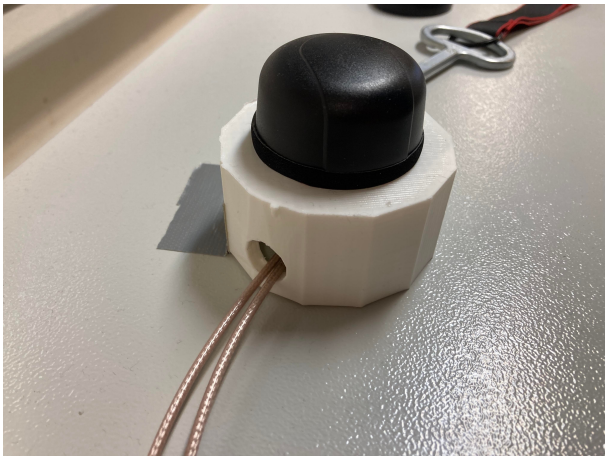


Figure 4.7: Support for antenna

4.2.15 Cable Entry

The Ethernet cable and the cables to the antennas and GPS have to be extracted out of the enclosure in a way that does not violate the requirement of IP54. This is done with a cable entry frame and cable grommets seen in Figure 4.8. This solution is IP65.

Figure 4.8: Cable entry with cable grommets [76]

4.2.16 The Electrical Wiring of the RSU

The electrical wiring of the RSU were done with safety and modularity in mind. Since one of the stakeholders requirements states that the RSU have to be able to run on both grid power as well as battery power a IEC power socket [77] was mounted on the side of the RSU and a DC-AC converter [74], to convert the 12 V DC outputted from the car battery to 230 V AC, was purchased. To be able to switch between the different power options or to completely turn of the electricity a three position rotary cam switch [78] was installed. Furthermore, to ensure personal safety and to protect the circuit a earth leakage circuit breaker with a inbuilt 10 A fuse were added to the circuit [79]. Additionally, to add modularity to the installation a DIN mounted power outlet [80] was added to which a branch outlet with 6 slots [81] could be connected. All the electrical apparatus inside the RSU as well as the heater and the enclosure fan was connected to the branch outlet. To control the heater and enclosure fan a normally closed and normally open thermostat was installed [82][83]. The addition of a joint box [84] and wago connectors [85] to connect cables together made the space management inside the RSU more efficient. The wires used were standard RKK 3x1.5 mm^2 cables [86] as well as single FQ 1.5 mm^2 cables and the colour code for a electrical installation was followed. Lastly the RSU box and the installation was grounded through the use of a earth plint [87]. To see how all the components were mounted inside the RSU, see section 6.1. The electrical schematic for the installation can be seen in appendix F.

4.3 Final Implementation Camera and LiDAR Data Transmission

This project had stakeholder requirements on both the throughput capabilities of the RSU as well as the OBU when it came to transferring camera and LiDAR data, see section 1.3.1 and 1.3.2 for the requirements. To meet these requirements different solutions were investigated. The camera and LiDAR data transmission solution was intended to include hardware that would be implemented in both the RSU and the OBU. To be able to evaluate different options for the camera and LiDAR transmission an evaluation was made on the existing options, see table 4.2. The scoring of the table is done by a point system with points ranging from 1-5, where a higher score equals a better performance for the corresponding feature. The reasoning behind the scoring is based on the information presented in the SOTA part of the report in section 2.5.

Table 4.2: Comparison between WiFi and cellular 5G

Feature	WiFi 11.ac (WiFi 5)	WiFi 11.ax (WiFi 6)	Cellular 5G
Throughput	3	4	3.5
Range	1	2	3
Reliability	2	3	4
Availability	5	3.5	2.5
Cost	5	4	2
Score	16	16.5	15

As can be seen in table 4.2 the final scoring between the different technologies were quite similar. The WiFi 11ax technology received the highest score, yet the most effective solution for this project when it comes to range and reliability would be to implement a cellular 5G solution. However, given the high cost of both hardware, implementation of infrastructure, as well as low availability of existing hardware together with modest average upload speeds, the 5G technology is scored lower. Even though the WiFi 11ax technology achieved the highest score a design solution for both WiFi 11ac and WiFi 11ax was developed, due to the high level of similarity between the two.

For the hardware that was going to be placed in the RSU the setup looked the same for both solutions. A WiFi 11ax-router with detachable antennas was intended to be purchased and placed inside the RSU enclosure. Originally an omni-directional Poynting PUCK-12 [88] antenna with vertical polarization was chosen to be connected to the router and mounted ontop of the RSU enclosure. The reasoning behind choosing an omni-directional antenna is based on section 2.6 in the state of the art chapter. An Omni-directional antenna with its donut shaped radio pattern allows for connectivity from several directions, which is preferable since the RSU will be communicating with moving vehicles. However, choosing an antenna for a specific application can be a difficult task and often involve testing before deciding for a antenna type. That this was the case was confirmed by reaching out to professor Mark T Smith at KTH that confirmed it would be hard to choose a certain antenna for a application without doing actual field tests. Mark also suggested that when field tests is not possible one should look at the antenna pattern and choose an antenna with as high gain as possible. The project group did not get an approval to buy either a WiFi 11ax router or the originally chosen antenna during the project. Instead a router was given by the stakeholder which was shown to not be efficient enough for the use case. Therefore one of the members of the project brought its own ASUS RT-AC68U WiFi 11ac router [89] that was used for the remainder of the project. However, during the final week of the project a low gain omni-directional antenna [90] was approved for purchase, mostly to be able to deliver a complete RSU in the end of the project.

For the hardware that was going to be placed inside the OBU the developed options differed a little. However, none of these options were implemented in the end since the OBU didn't end up being developed. Instead the camera and LiDAR data was sent directly from the TurtleBots Raspberry Pi. To increase the throughput Netgear A6100 WiFi 11ac USB adapters [91] was purchased and connected to the TurtleBots. To see the originally developed options for the OBU see appendix H.

4.4 Final Implementation Platooning Algorithm

Platooning refers to the action of vehicles following other vehicles in a controlled formation. In this project the robots needed to be able to follow a simulated vehicle, whilst retaining a safe distance between one another. This section begins by describing the overall approach, and follows up by describing some of the methods used to facilitate this approach.

4.4.1 Platooning Algorithm

To facilitate channels of communication, a ROS based approach was used. The lead vehicle communicated its position to the vehicle directly following it, and this vehicle relayed its own position to the next one. This allowed for a rather modular solution, which could be expanded easily. Since the communication occurs at a specified rate, the continuous path of a vehicle is communicated as discretized waypoints with positional information. When compared with the current position of a vehicle, these waypoints can be used to generate longitudinal and lateral error signals for control purposes. Once a vehicle reaches a position within a certain range from a waypoint, that waypoint is removed.

4.4.2 Localization

For the platooning algorithm to be possible, the robots require the ability to reason about their position in the environment. This allows them to calculate a relative position to the vehicle that was to be followed, in order to generate control error signals. Since the used robots came readily equipped with a 360 degree laser distance scanner, range measurement based localization was utilized. More specifically, this employed the Adaptive Monte-Carlo Localization (AMCL) method. As the algorithm exists in a finished state as a ROS package, the implementation was fairly simple. After installing this package, the only remaining task was to tune parameters. These parameters related to implementation framework specifications, such as update frequency,

sensor accuracy, motion noise and so on. To tune this package is a task in and of itself, and a more in depth discussion of how these parameters should be treated can be found in [92] as well as [93].

Localization methods compare measurements with a known environment to localize a robot. As such, they require a previous understanding of how the environment is structured in order to function. To this end the robots needed a map, which was provided by using Simultaneous Localization and Mapping (SLAM). A pre-existing ROS package was used for SLAM as well, called gmapping.

The application of localization methods can be considered reasonable for this project, as the testing and demonstration was limited to an indoor environment. Moreover, the next stage of the project is to continue testing outside within a select area which has also been mapped by laser to a high degree of accuracy. As such these methods can be used in further stages as well, and do not need to be used solely for the platooning demonstration. However, for other use cases an existing map might not be a reasonable prerequisite. In such a case the normal solution is to use some form of GNSS, or even online SLAM. It is important to note that the limitations of this project did not allow for either of these solutions. This was due to testing inside (which disallows use of normal GNSS services) and the inferiority of accuracy throughout all stages of SLAM compared to localization. While SLAM can create reasonable maps, this usually requires loop closure. As such, accurate positioning of robots while in unknown GNSS-denied environments is still an area of active research.

4.4.3 Controller and System Identification

In order for the TurtleBots to follow the trajectory of the simulated vehicle in Prescan, in a platoon, controllers are needed. As discussed in the SOTA part of the report in section 2.7 a more state of the art approach to ensure internal stability and string stability throughout the platoon would be to use some variation of a MPC controller or a MPC controller in combination with another controller such as the $\mathbb{H}_\infty$ controller. However, since the TurtleBots only reach up to a maximum of 0.26 m/s and have negligible inertia it was concluded that such a advanced controller was not needed in this projects use case and that a PI controller would be sufficient. Additionally, the use of a PI controller reduce the computational power needed compared to a more advanced controller. And since this project was under a strict budget reducing the cost of computational units were preferred. Furthermore, because of the mentioned low speed and small inertia it was also decided that there was no need to derive a dynamical model of the TurtleBot. Therefore, decoupled PI controllers for the longitudinal distance and angular difference to the preceding vehicle were implemented as part of the platooning package described in section

4.4.1. The plan was to hand tune the proportional- and integral gain of the controllers. However, it was found to be too time consuming to find appropriate controller gain values by hand that would allow for string stability throughout the platoon. To speed up the tuning process of the controllers it was instead decided to use system identification.

System identification is a method for building mathematical models of dynamic systems based on measured input and output signals of a system. Matlab includes a System Identification Toolbox and in particular this toolbox contain a PID Tuner which can be used for both plant identification and controller design in the same interface [94]. One way to retrieve the plant identification of the TurtleBots for both the longitudinal and lateral controllers using the PID Tuner is to use the software's inbuilt step response. The step response expects an output signal vector of the system being analyzed. Therefore, a ROS package was constructed with the functionality of publishing both a linear velocity as well as a angular velocity step output to the TurtleBots. The transnational and rotational velocities of the TurtleBot were then recorded and written to two separate text files using the same package. These textfiles could then be loaded into the Matlab workspace and then further imported to the PID Tuner to perform the plant identification and controller tuning for the longitudinal and lateral PI controller. Figure 4.9 and 4.10 show block diagrams of the longitudinal and lateral PI controllers, and how way-points and positional information of the vehicles are passed throughout the platoon.

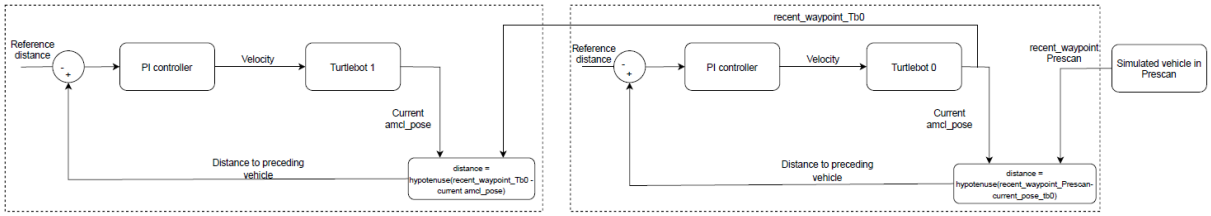


Figure 4.9: Block diagram of the longitudinal PI controller and information flow of way-points and positions of the vehicles throughout the platoon.

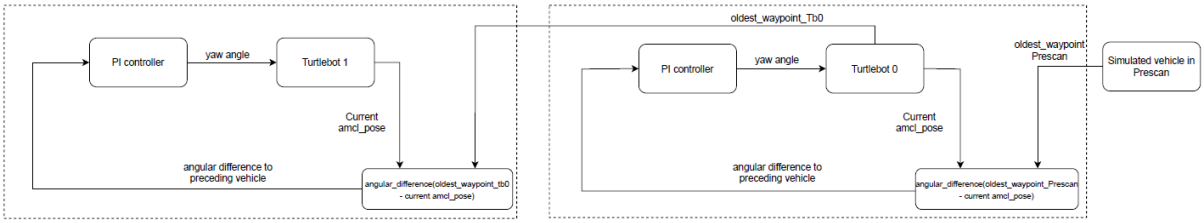


Figure 4.10: Block diagram of the lateral PI controller and information flow of way-points and positions of the vehicles throughout the platoon.

4.5 Final Implementation Object Detection

This section explain the purpose of implementing object detection in the project as well as how it was done.

4.5.1 Edge Computing and Object Detection

Edge computing refers to the ability to offload complex computations from OBU's with less computation power to the infrastructure. In this project RSU is the platform referred to as the infrastructure where the TurtleBots have the ability to send sensor data such as camera and LiDAR data to the RSU where it can be processed. To showcase the Edge computing capabilities of the RSU object detection was implemented on both of the TurtleBots which allowed them to detect objects such as cars or trucks along their trajectories in the platoon.

In order to realize the function of object detection and get the information of targets like categories and positions, an object detection algorithm was implemented which is called "You Only Look Once" (YOLO). This is a state-of-the-art real-time object detection system. YOLOv3 uses features learned by a deep convolutional neural network to detect an object [95]. Different pre-trained configurations, detection weights are available, which showed different performances when tested on the RSU. The best one was applied for the object detection in this project. Furthermore, OpenCV was also used to allow for real-time computer vision while CUDA and cuDNN was implemented to speed up calculations.

CUDA

In order to increase the identification speed of YOLOv3 in order to have a better performance in video source, GPU is used to accelerate the calculation of image matrix.

NVIDIA GPU needs Compute Unified Device Architecture driver for parallel computing and application programming interface which called CUDA [96]. Considering the existing NVIDIA x86_64 Graphic driver on Ubuntu of version 430.64, and the limitation of CUDA Toolkit on this NVIDIA driver version, the CUDA version of 10.1 was selected as the best choice [97].

cuDNN

NVIDIA provides NVIDIA CUDA Deep Neural Network library (cuDNN) for acceleration of deep neural network. This includes methods of convolution, pooling, normalization, and activation layers which are used in YOLOv3[98]. According to the NVIDIA cuDNN Archive[99], the version v7.6.5 of cuDNN matches with the other libraries used, and as such was employed in the project implementation.

OpenCV

OpenCV a library of programming functions mainly aimed at real-time computer vision developed by Intel [100]. YOLOv3 requires the functionality of OpenCV for quick image and video loading, and for realizing the real-time monitoring of detection results. Considering the versions of software used (specifically Python3, the Nvidia graphic driver and CUDA version) OpenCV 3.4.2 was selected as the most suitable version.

4.5.2 Ubuntu 16.04 with NVIDIA GTX 1080

To use the NVIDIA GTX 1080 (the GPU) correctly, initially it was attempted to use drivers from the NVIDIA website. After installing these, Ubuntu 16.04 would only boot to the login screen. After inserting the login credentials, the system returned to the login screen. Forum posts indicated that the NVIDIA drivers were simply not compatible with Ubuntu 16.04. As such, the problems have been solved by disabling the original graphics driver in Ubuntu system. The specific steps to install drivers can be seen in Appendix E.1

After doing software update on Ubuntu 16.04 to update the drivers, the graphics card worked when running GPU tests. This was not very apparent as the fans on the GPU did not start until it reached 60 degrees Celsius. These drivers were unfortunately insufficient as they did not work properly when using CUDA.

One thing to note when checking if the GPU drivers are working is to make sure that the attached screen is connected to the GPU itself. The screen will still work if attached to the motherboard connector for integrated graphics, but it will be impossible to log into the account. A loop will occur in which the log in screen will be repeated forever.

4.5.3 Object Detection Algorithm

The object detection algorithm use a ROS based approach and were built upon the darknet_ros package [101] which enables the use of YOLOv3 to detect objects. The darknet_ros package publish boundary box information of the detected object which could be used to calculate the coordinates of the detected object in the global coordinate system set by the map frame. To do this the midpoint of the boundary boxes were calculated and transformed into the corresponding angle in the TurtleBots camera's viewing angle. The angle were then used to retrieve the distance value published by the LiDAR sensor at that particular angle. By then using basic geometric formulas the coordinates of the detected object relative to the LiDAR could be calculated. Based on these coordinates a tag frame of the detected object could be published where then a transformation from the map frame to the tag frame could be done to retrieve the coordinates of the detected object in the global coordinate system. The explained algorithm were divided into three different scripts where each script correspond to a ROS node. Furthermore, an additional script were made to visualize the detection in ROS Visualization (RVIZ), based on the calculated global coordinate of the object, using basic shapes [102]. To see how the communication between the different nodes looks like see the RQT graph in section 6.4.1.

During the testing phase it was found that the global coordinates of the detection got less accurate when the midpoint of the boundary box were to close to the edge of the TurtleBot camera's viewing angle. Therefore an interval were implemented for which if the midpoint of the boundary box is to close to the edges of the camera's viewing angle, then the detection is discarded.

4.6 Implementation of the V2X Communication (ITS-G5 frequency band)

Two solutions for ITS-G5 communication were developed. However, due to some problems in ordering and delivery the one that was spent the most time on was the Software Defined Radio (SDR) solution. At first, a HackRF One was ordered. It was later discovered that the HackRF One, due to only supporting half duplex mode, could not be used as a transceiver for ITS-G5 specifically. It's possible to shift between transmitting and receiving but this would compromise the reliability of the communication channel which is intended to be robust and reliable for safety critical applications.

A bladeRF 2.0 micro xA4 [103] was then ordered as it is full duplex. This enabled the possibility to use it as a transceiver, i.e. transmitting and receiving at the same time.

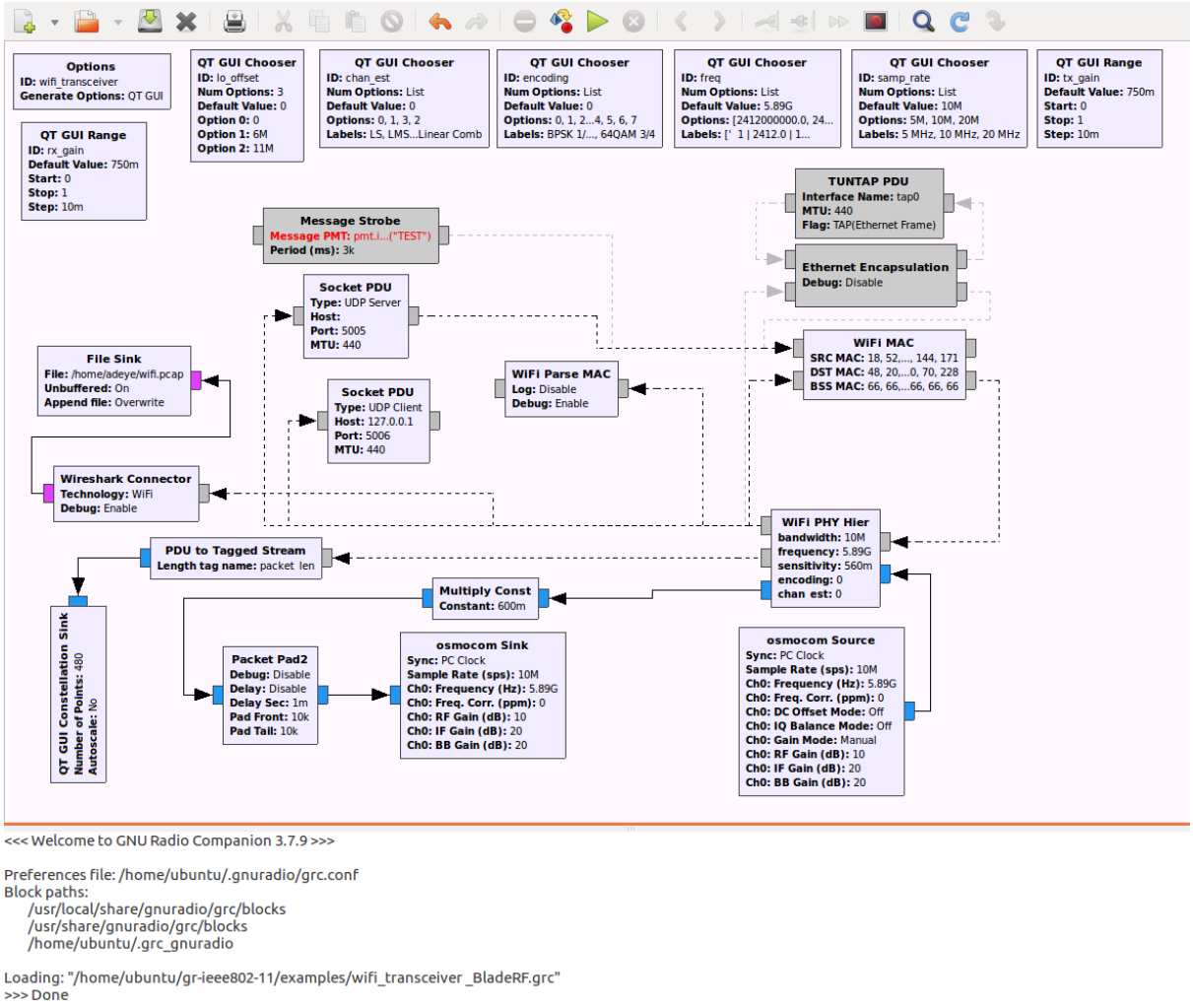
The bladeRF was then mounted inside the road side unit enclosure and powered by a 3.0 USB port from the RSU's motherboard and connected to two antennas. This will make the Road Side Unit able to send and receive simultaneously on the dedicated frequency band for the ITS-G5.

4.6.1 GNU Radio

The first step in developing an ITS-G5 solution with a SDR was to build a signal processing application in GNU Radio. GNU Radio is an open-source software development toolkit, it provides signal processing blocks for implementation of software-defined radios, among other [104]. In GNU Radio it is possible to set the desired frequency, channel, bandwidth and sampling rate. GNU Radio provides a message block that is able to send text as Ethernet packages, thereby, enabling transfer of critical messages from a back bone unit to demonstrate the functionality of the transceiver on the ITS-G5 frequency band. Since the Road Side Unit runs Ubuntu 16.04, Gnu-radio 3.7.9 is the latest version that works with this Ubuntu version.

4.6.2 Transceiver Software

For our implementation we are using an open source transceiver application Ieee802-11 for Gnu radio [105] written by B.Bloessl [106]. This open source code include GRC files which are GNU Radio flow graph files with the necessary signal processing blocks that can be re-modified and run by GNU Radio to work as a transmitter, receiver or transceiver on the bladeRF. Ubuntu 16.04 comes with CMake 3.5.1 and for installing the gr-ieee802-11, CMake version above 3.7.8 is needed. CMake is cross-platform free and open-source software for build automation, testing, packaging and installation of software by using a compiler-independent method. For more on ITS-G5 Software installation see Appendix E.1. In Figure 4.11 the transceiver flow graph can be seen with its necessary signal processing blocks to send/ receive data on the dedicated frequency band in a form of Ethernet packages and store the received data in a file sink as a PCAP (packet capture)files consist of network packet data, created by capturing live network activity. The pcap files can be captured and fetched through capture tools such as the open-sources packet analyzer Wireshark. The Message Strobe block in Figure 4.11 is disabled because the sending of the emergency stop is done from an external ITS-G5 source which is a laptop with the same ITS-G5 setup connected to a HackRF. In case the emergency stop needs to be generated internally, then enabling this block will make the SDR able to broadcast the emergency to other vehicles in range.




```
<<< Welcome to GNU Radio Companion 3.7.9 >>>

Preferences file: /home/ubuntu/.gnuradio/grc.conf
Block paths:
    /usr/local/share/gnuradio/grc/blocks
    /usr/share/gnuradio/grc/blocks
    /home/ubuntu/.grc_gnuradio

Loading: "/home/ubuntu/gr-ieee802-11/examples/wifi_transceiver _BladeRF.grc"
>>> Done
```

Figure 4.11: GNU Radio flow graph for the transceivers used with the bladeRF

4.6.3 Integration with ROS

If an obstacle occurs the TurtleBots are supposed to receive a message to stop. This is done by the ITS-G5 unit that sends a message to ROS, which then forwards the message to the TurtleBots. To do this, a message needs to be sent from GNU Radio to ROS. The flow graphs that are included in the Ieee802-11 transceiver have their own TUN/TAP interface that can be used to extract the information from them for a further integration with programs outside GNU Radio. However, since the integration is with ROS, where a UDP server node can be created, a block in GNU Radio called *Socket PDU* was therefore implemented in the flow graphs. This block starts a UDP socket as a client UDP inside GNU Radio. On the ROS side a UDP socket server node on the same port is opened. This results in the possibility to send messages between the two applications/programs.

As seen in Figure 4.11 both TUN/TAP PDU and Ethernet Encapsulation were disabled and a Socket PDU (UDP client) was added instead to capture any incoming messages coming out of the WIFI PHY Hier from the receiver port, which is the emergency stop message in this case, and publish it to the UDP socket on a specific port. This message will be picked up by ROS which listens to the same port through ServiceProxy and preform an instant and simultaneous stop to all real and simulated vehicles in the platoon.

4.6.4 Final Communication Implementation

The final implementation of the communication flow was slightly different than in early analysis of the system. Below a figure of the communication flow can be seen. The ITS-G5 message is sent from a backbone network to ROS in the RSU which then sends a message to stop the TurtleBots.

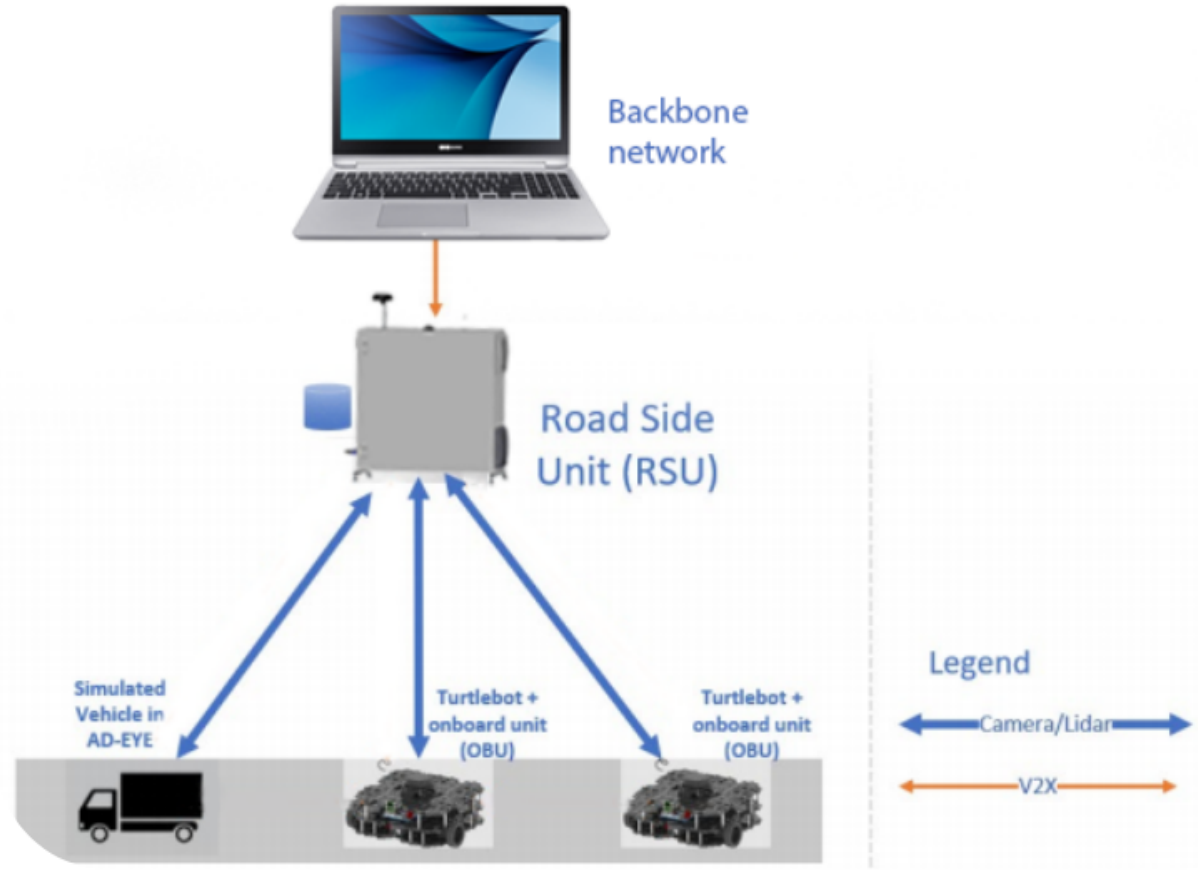


Figure 4.12: Showing the final communication setup throughout the system.

Chapter 5

Verification and Validation

In this chapter, the tests to verify and validate the project are described.

5.1 Unit Tests

Unit tests refer to single module tests, where the functionality of individual physical components or software can be tested.

5.1.1 Electrical Installation Verification of the RSU

To ensure that the electrical wiring of the RSU had been done in a correct and safe way all the connections were double checked against the electrical schematic, see appendix F. Furthermore, to be sure that all the wires were connected securely a pulling test of each cable was done to see that no cables were lose. Additionally a multimeter was used to ensure that there was 230 V between phase and zero as well as between phase and ground. It was also checked that there was 0 V between zero and ground. On top of this, a measurement was made to check that there was no current present in the RSU backplate/enclosure when measuring between the backplate/enclosure and true ground which ensures that a human will not get an electrical chock when touching the RSU while being in contact with ground. Lastly, a small load was connected between phase and ground using a small resistor to test that the earth leakage circuit breaker worked properly. The measurements/tests were conducted for both power grid mode and battery mode.

5.1.2 RSU Temperature Management

A test was conducted in a cooling room to test the minimum temperature requirement of -15 °C. The RSU was on inside of the room while the temperature in the cooling room was lowered. The motherboard was left in standby in order to test worst case scenario of minimum power consumption.

5.1.3 CPU and GPU Performance Degradation

To see if the computing components in the RSU suffered any degradation in performance for different outside temperatures, a test was made. First, the temperature of the room was recorded. Then computer was set to output the temperature readings of the CPU cores and GPU once every 2 seconds. The system test described in section 5.3 was then started on the computer. If any of the CPU cores reached 100 °C and then started dropping, it meant that it had thermal throttled and as such couldn't output its maximum performance. The throttling point for the GPU is 94 °C [68].

5.1.4 RSU Ingress Protection

All components mounted on the enclosure are rated IP54 or higher, so in theory the RSU should be IP54+. The standard way of verifying IPx4 protection is to spray the object with 10 l/min of water at a pressure of 80-100 kPa for 5 minutes all over the object [**IPx4 test**]. We did not have access to a proper water hose and as such used a bucket filled with an estimated 20 liters of water. The focus was on the parts that could take in water i.e. the filters, cable entries and antennas.

5.1.5 Camera and LiDAR Data Transmission

Since the OBU did not end up being developed, as mentioned in section 4.1, the Turtlebots is seen as the OBU in this verification of the throughput capabilities of the OBU.

To verify that the throughput requirements of the RSU and the turtlebots were met the ASUS RT-AC68U routers [89] inbuilt user interface was used. The platoon was run in a loop and both of the TurtleBots performed object detection of trucks and cars placed along their trajectory. At the same time the throughput from the two TurtleBots could be monitured through the mentioned ASUS routers user interface.

5.1.6 ITS-G5 Receiver Testing

The SDRs were tested with the Cohda Wireless MK2. The Cohda box has an inbuilt script for testing which sends random packages on the ITS-G5 frequency. These packages were captured by both the HackRF and the bladeRF which confirms the receiver capabilities for the SDRs on the ITS-G5 frequency band.

How to launch the test script from the MK2 is detailed in appendix G.

5.1.7 ITS-G5 Transmitter Testing

The SDRs were tested by communicating with each other on the ITS-G5 dedicated frequency band by sending from the HackRF and receiving on the bladeRF, and vice versa. Because we know both devices can reveive on the intended frequency bands, this also implies that transmitting on the ITS-G5 bands is confirmed to be working on both devices.

5.1.8 ITS-G5 Transceiver Testing

The bladeRF is a full-duplex SDR which can send and receive on the same time as opposed to the HackRF which is half-duplex. To verify that the full-duplex mode works with the ITS-G5 frequency, both the Cohda Wireless box and the HackRF were used. A transceiver flowgraph was used on the badeRF which sends and listens on the right frequency, the Cohda box sending its ITS-G5 messages and the HackRF in receiver mode listening to ITS-G5 frequency. By the received messages on the bladeRF and HackRF the full-duplex mode on the bladeRF can therefore be verified.

5.1.9 Performance of YOLOv3

In order to test the performance of GPU and related drivers on acceleration of object detection and computer vision, 2 pictures and 1 videos are tested on YOLOv3.

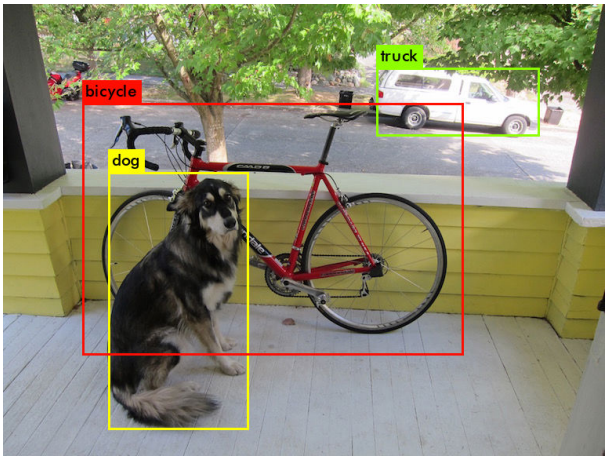


Figure 5.1: Testing image with dog

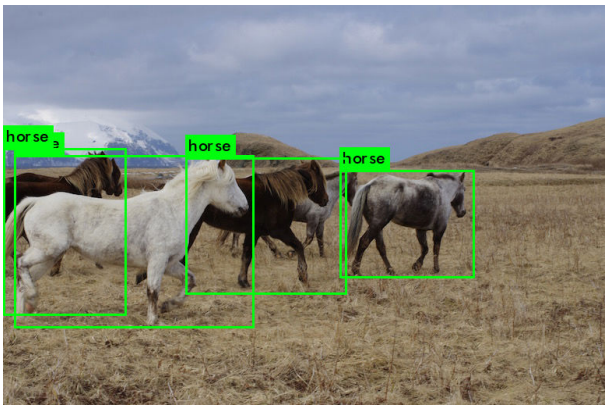


Figure 5.2: Testing image with horse

Figure 5.3: Testing video with car and truck

5.2 Integration Tests

In these integration tests, individual components are combined and tested in unison. This allows to measure the performance and trouble points when combining the functionality of several units.

5.2.1 Simulation of Platooning

To test the integration of several software components for platooning, the use of a simulated environment was employed. The ROS compatible simulation engine Gazebo provided a solid platform for performing functionality tests on the integration of controller with platooning logic on TurtleBot vehicles. Moreover, additional components could easily be added to the overall scheme, such as sensor fusing, localization methods, and other software modules directly related to the platooning. This served as a good indication of what issues between different software modules could arise.

5.2.2 Measuring of Platooning

The performance of the platooning is further verified in reality, through comparing the the trajectory of the simulated vehicle in Prescan with the following TurtleBot's path. To measure these trajectories, the simulated vehicle is discretized and used as ground truth. The first following robot's measured position with localization is then used as comparison. This measurement, being performed with the localization scheme AMCL, is much more accurate than using the odometry as comparison. However, it is still not a perfect measurement by any means, but serves as a benchmark for discussion. Moreover, the controller's performance in reality when combined with a discrete sample time and disturbances was difficult to simulate accurately, and these could be tested in reality by comparing the distances of the robots as well as their angular difference to the preceding robot. This was done at two different speeds. The distance was provided by their respective positions in the map, measured by AMCL.

5.2.3 Object Detection Accuracy

The focus in this project was not on how accurate the coordinates of the detected objects were in the global coordinate system. The focus was more to show the RSU's edge computing capabilities by using object detection. However, the detected objects in the TurtleBots environment were

output to RVIZ based on their global coordinates to see that the object detection worked and the that the coordinates of the detection seemed to be somewhat accurate.

5.2.4 GNU Radio - ROS Integration

As mentioned in the implementation section 4.6.3 a *Socket PDU* was implemented in GNU Radio to a UDP client that can be listened to on ROS by a UDP socket server node on the same port. This was tested and proven to work in one direction from a GNU Radio to ROS through multiple testes and demonstrations where emergency messages were send through the UDP socket to ROS on the same unit (RSU) and ROS in its turn stopped the platoon. However, Sending in the reverse direction meaning from ROS to GNU Radio through a UDP socket should work in theory but needs to be tested and verified in future work.

5.3 System Tests

The system as a whole was tested by performing a platooning scenario where the TurtleBots followed the simulated Vehicle in Prescan in a loop. Throughout the trajectory the TurtleBots also performed object detection which allowed them to detect object such as trucks and cars along the track and output the detections for visualization in RVIZ. In order to test ITS-G5 communication a stop message from a bladeRf unit positioned outside the RSU was sent to a HackRF unit inside the RSU. This showcased the capability to send messages on the allocated 5.9Ghz frequency band for V2X communication. The message received by the bladeRF unit was then interfaced with ROS by the use of a ROS service call to stop the TurtleBots in the platoon.

5.3.1 Flow Simulation

In order to get a rough understanding of how air and heat moves through the RSU, a flow simulation in Simcenter FLOEFD for Solid Edge was constructed.

Simplifications

In order to construct the simulation in a reasonable amount of time, some simplifications were made. This is a list of these:

- Our real motherboard is very complex with extremely many small heat generating chips, as well as heat sinks to counteract them. For this simulation, the motherboard was assumed to not generate any heat on its own, and had no heat-sinks.
- The motherboard is made out of several different layers, each with different thermal properties. The material was set to a custom pcb material from a FLOEFD tutorial.
- The CPU fan is a similar one to the one we have, but comes from FLOEFD tutorial and is as such not identical to the one in the real RSU. Small differences in the heat-sink and fan blades could make a sizeable difference. This fan was modeled to spin at the maximum speed of the real CPU fan.
- The GPU is a gross simplification of the real one, as it was modeled as a block with cutouts where the fans are supposed to be.
- The fan model is not official, the modeled one has no blades but other internal dimensions were matched as closely as possible.
- Since all components are static in FLOEFD simulations, there is no way to simulate the flaps of the outflow filter. Because of this, the inside of the outflow filter was considered to have environmental pressure. In reality, the pressure there would be a bit higher since a small pressure differential between inside and outside is needed to open the flaps.
- The power source was assumed to be the grid, and as such, the battery and DC-AC converter was considered "turned off".
- No heat was assumed to be generated by any components except the CPU, GPU, heater and RAM. In reality, the PSU has an efficiency of 90% and would be a significant enough heat source. The SDR also contributes with some heat in reality. However, since it is placed close to the outflow, this heat should be dissipated fairly easy.

Scenarios

Three different simulation scenarios were modeled.

1. Regular inside conditions.
   The initial conditions of every component, and the air around them was considered to be 20 °C. The computer components were considered to run at 80% of their maximum power rating, the inflow fan was assumed to have a 40% reduction in airflow compared to its maximum, in accordance with the data-sheet of the fan

2. Maximum temperature.
   All initial conditions were set to 35 °C, but otherwise it was identical to scenario one.

3. Minimum temperature.
   All initial condition were set to -15 °C and all electronics, including the fans, except the heater was turned off. The heater was modeled to output its maximum power.

The primary goal of scenario 1 and 2 was to model how air flows within the RSU, as well as checking how hot the CPU gets, since a CPU that gets too hot will throttle and lose performance. The primary goal of scenario 3 was to see how much the heater could warm the enclosure by itself, and as such ready the surrounding components for startup. It was also important to know how hot the closest surrounding components would become. The exit filter in particular is very close and could start to melt if it becomes to hot.

The results of this simulation can be found in section 6.5 and a more theoretical global heat estimation was done in section 4.2.4.

Chapter 6

Results

This chapter presents the hardware was built (RSU) and the results of the testing and verification described in the previous chapter. These results are used to confirm whether the requirements were fulfilled or not.

6.1 The Assembled RSU

The assembled RSU can be seen in Figure 6.1.

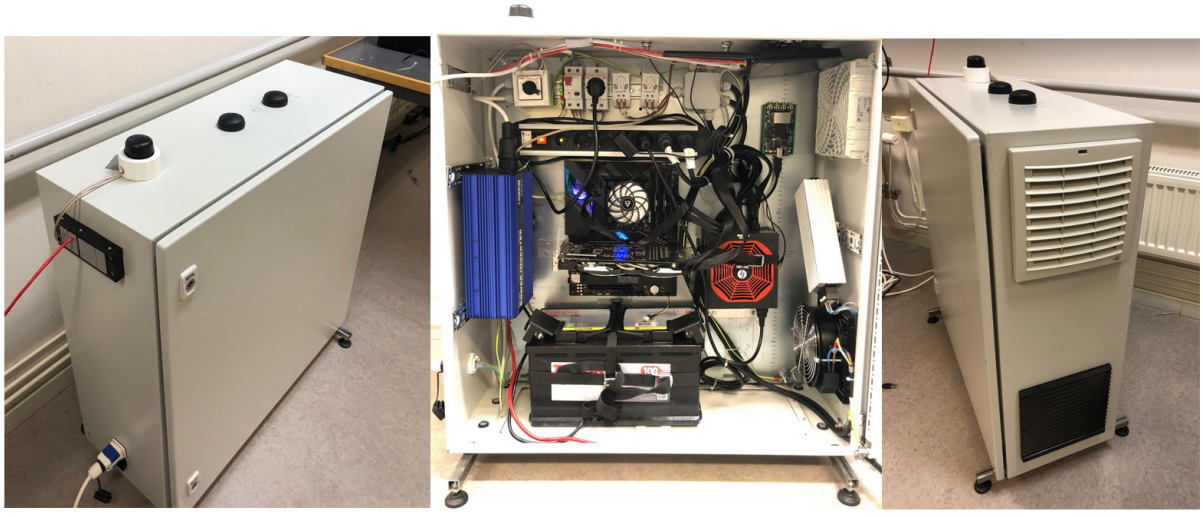


Figure 6.1: The assembled RSU

6.2 Unit Test Results

Results related to unit tests are presented here.

6.2.1 Electrical Installation Test Results

From the measurements and tests described in the previous chapter it could be concluded that when running on power grid power everything worked as expected and all the measurements and tests gave the expected results.

However, when running the RSU on car battery power through the DC-AC inverter the measurements did not give the expected results. Measuring between phase and zero showed 230 V which is expected. But measuring between phase and ground showed 180 V while measuring between zero and ground showed 50 V, which deviates from expectations. Since the DC-AC inverter does not output real AC power there can be some differences when it comes to the potentials between different wires compared to a normal AC power system. This was also discussed with a professor at KTH which also confirmed that these kind of measurements could be seen when using a inverter. Furthermore, when putting a load between phase and ground the external earth leakage circuit breaker in the RSU did not shut down the power, this is most likely because the different potentials that are present between the wires when using a inverter compared to a normal AC power system described previously. It can be noted though that since the inverter has an inbuilt protection against overheating, overloading, short-circuiting as well as an earth leakage circuit breaker the installation was still deemed safe. Lastly when measuring between the backplate/RSU enclosure to a real ground 0 V could be seen which means a human will not suffer from electric shock when touching the RSU while being in contact with ground. Overall, the car battery power solution was deemed safe and it also sufficiently delivered power to all the components inside the RSU.

6.2.2 Battery Testing

To test the battery time, the RSU was setup to run object detection for an hour as a real world use case. The RSU ended up running slightly longer than two hours before the DC-AC inverter started beeping, indicating a low battery voltage (low capacity). The test was then terminated, with a successful result.

6.2.3 RSU Temperature Management

The RSU was able to keep the inside temperature high, ranging from 7 °C to 12 °C at bottom and top, so an average of about 9 °C. This is quite a bit better than what was calculated in appendix B, which is likely because of a higher idle power consumption than predicted, as well as a lower air velocity in the cooling room than in the calculations, causing a lower convective heat transfer. It was powered through the power grid during the test. If it would have run on battery then more heat would have been generated because of the losses in the inverter. It is therefore concluded that the requirement of being operational with little to no degradation at -15 °C is fulfilled.

The other worst case scenario of 35 °C outside temperature has not been tested. As the current CPU fan is far too weak for the CPU, so not only could high strain damage the CPU, but power consumption will also be throttled before coming close to the maximum power used in the calculations.

6.2.4 CPU and GPU Performance Degradation

The CPU was not able to output its maximum performance in room temperature. After running the system test for a minute, the CPU cores reached the 100 °C mark and started to throttle. Because of this, no further test was deemed useful. The GPU was during the test able to keep a stable temperature of 71 °C.

6.2.5 RSU Ingress Protection

Each vulnerability had a quarter of the buckets content poured over it over a duration of 30 seconds. The interior was then examined after the test and no traces of water could be found. The amount and intensity of the water was more than could within reason be expected from Swedish weather conditions, and so water resistance is considered verified.

As for dust resistance, this is more difficult to verify. But seeing as the only place dust can possibly enter is the inflow of air which is equipped with a certified IP54 air filter, the dust resistance can be considered certain.

6.2.6 Camera and LiDAR Data Transmission

The results from monitoring the throughput using the ASUS router user interface can be seen in Figure 6.2. This was when the turtlebots had ran in the platoon for about 1.5 minutes while at the same time performing object detection. In the picture the IP adress of 192.168.2.200 corresponds to the TurtleBot behind the Prescan vechile in the platoon while 192.168.2.192 corresponds to the TurtleBot in the last position of the platoon. As can be seen both TurtleBots are able to both send and receive above a throughput of 260 Mbps. It was therefore concluded that the stakeholder requirements of a throughput of 50 MBps (400 Mbps) for the RSU and 20 MBps (160 Mbps) for the OBU's (TurtleBots) had been met.

Figure 6.2: Showing the throughput of both of the TurtleBots

6.2.7 Performance of YoloV3

The results of the tests are shown here.

Table 6.1: Testing results of CUDA performance

Source	Performance without CUDA	Performance with CUDA
Dog	17.409372 s	0.050922 s
Horse	16.913832 s	0.051932 s
Car	0.1 FPS	17.2 FPS

It can be concluded that the performance with CUDA increases obviously. And in the later test for object detection in ROS, the quality of video could reach to 60 FPS.

6.2.8 ITS-G5

The packages sent from Cohda MK2 could be received on the HackRF and bladeRF and through this, it could be verified that the SDRs could receive on the dedicated frequency for ITS-G5. The bladeRF full-duplex was verified by running a transceiver flow graph on GNU Radio to receive and send from/to the Cohda Wireless and The HackRF respectively at the same time.

6.3 Integration Test Results

Results related to integration tests are presented here.

6.3.1 Simulation of Platooning Test Results

The testing in simulation revealed early on that there were issues with implementing sensor fusion packages for the Turtlebot. This was purely a software issue, resulting from the existing software framework of the Turtlebot in ROS being unsuitable for sensor fusion in multiple robots. The namespace which separated the robots did not work as intended, due to the limitations of this existing framework. This resulted in abandoning the sensor fusion package of *robot_localization*, and instead focusing purely on localization through LiDAR based methods.

6.3.2 Measuring of Platooning Test Results

The recorded discretized trajectory of the simulated Prescan vehicle as well as the TurtleBot's following path can be seen in Figure 6.3. Here the starting position of the TurtleBot is at (0.5,0) while the Prescan vehicle start at (1,0).

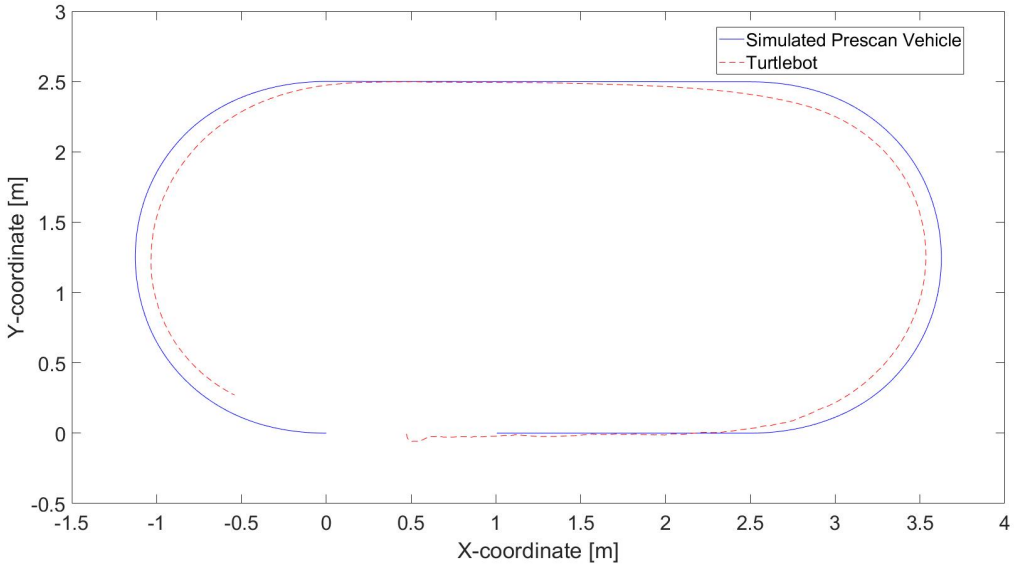


Figure 6.3: Showing how well the TurtleBot follow the trajectory of the simulated Prescan vehicle.

6.3.3 Controllers

The result from the plant identification and initial tuning for the longitudinal and lateral PI controller, which was further explained in section 4.4.3, can be seen in Figure 6.4 and 6.5, respectively. To perfect the performance of the controllers some additional hand tuning had to be done, but the use of system identification and the PID Tuner provided a indication of feasible values of the proportional- and integral gains.

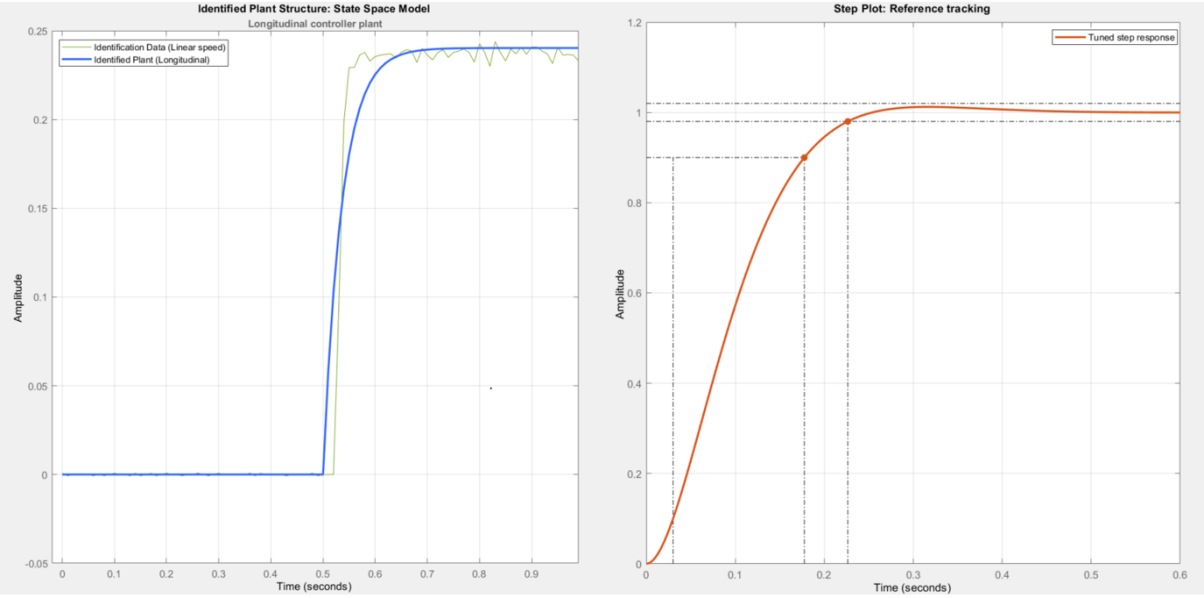


Figure 6.4: To the left: Identification data and identified plant for the longitudinal PI controller. To the right: Step response of the longitudinal PI controller, tuned by PID Tuner.

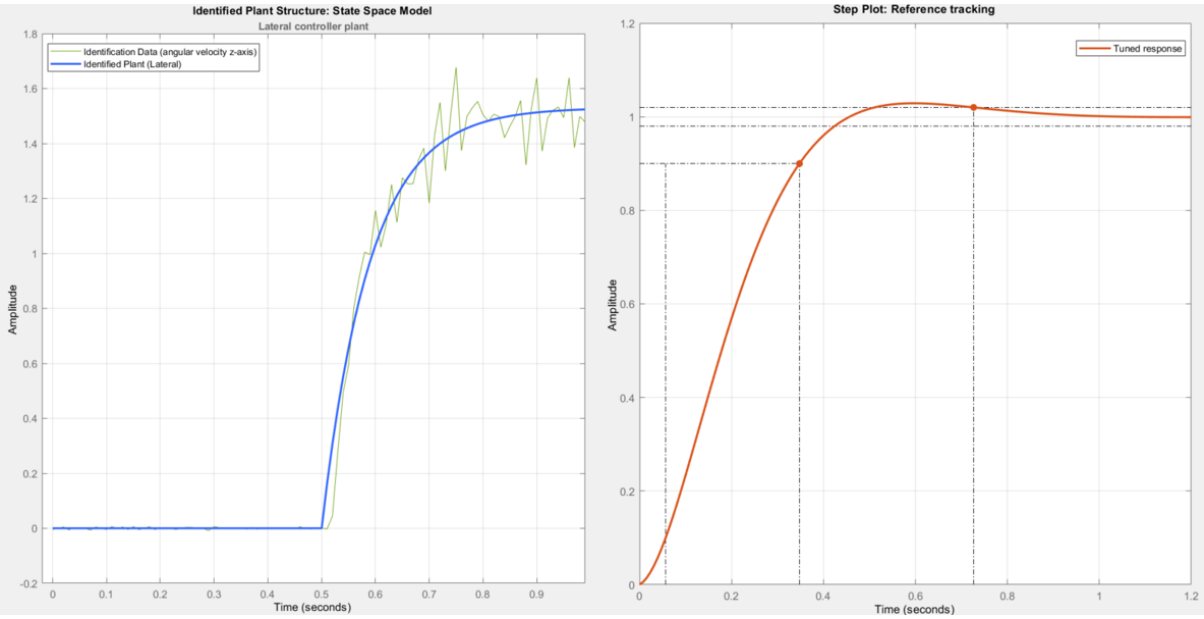


Figure 6.5: To the left: identification data and identified plant for the lateral PI controller. To the right: step response of the lateral PI controller, tuned by PID Tuner.

Furthermore, the performance of the longitudinal PI controller which was tested by plotting recorded longitudinal error signals of the TurtleBots throughout their trajectories, at a speed of both 0.15 m/s and 0.23 m/s. The result can bee seen in Figure 6.6 and 6.7, respectively. As can be

seen the error signal does fluctuate a bit around the reference with a slightly better performance at the lower speed of 0.15 m/s.

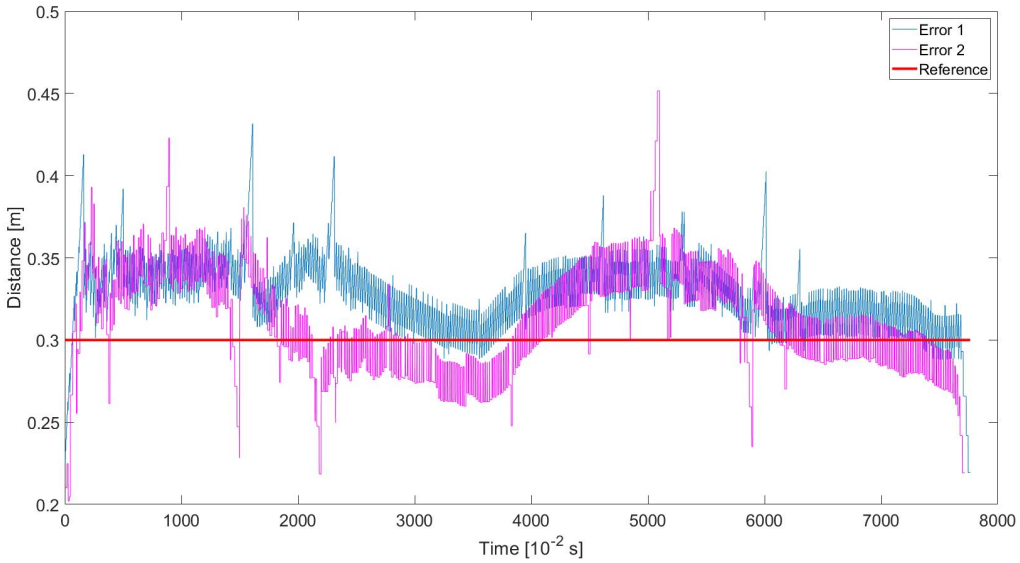


Figure 6.6: Showing the longitudinal error signal produced by each TurtleBot throughout their continuous trajectory at a speed of 0.15 m/s.

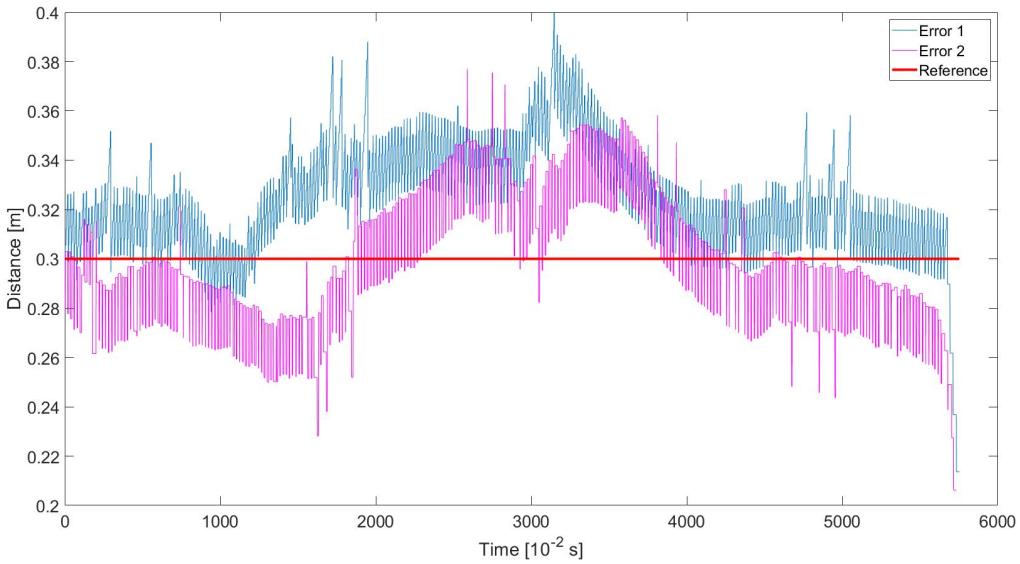


Figure 6.7: Showing the longitudinal error signal produced by each TurtleBot throughout their continuous trajectory at a speed of 0.23 m/s

Additionally, the performance of the lateral PI controllers was tested by recording the angular difference between TurtleBots throughout their trajectories, at a speed of both 0.15 m/s and 0.23 m/s. The result can be seen seen in Figure 6.8 and 6.9.

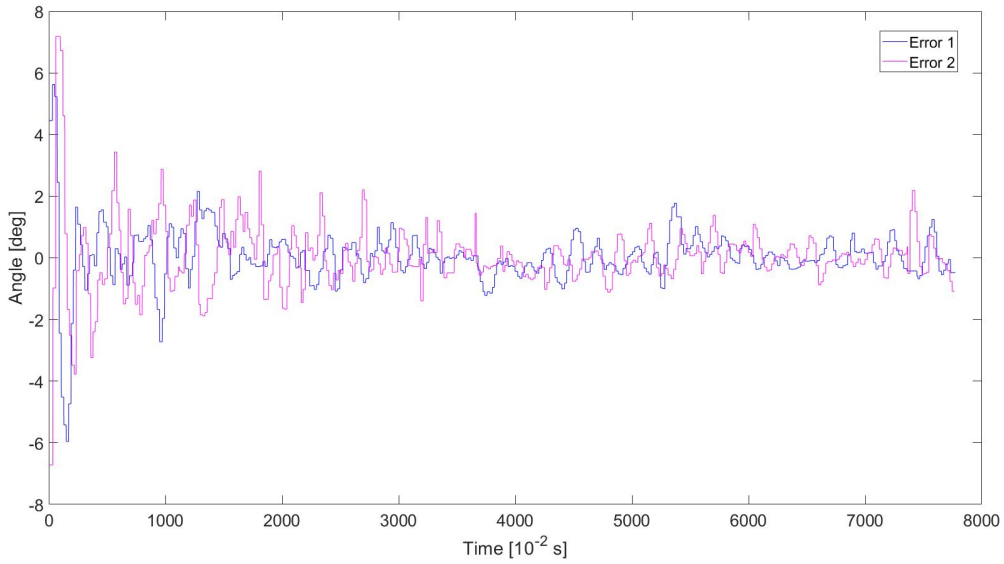


Figure 6.8: Showing the angular difference to the preceding vehicle for each TurtleBot throughout their continuous trajectory at a speed of 0.15 m/s

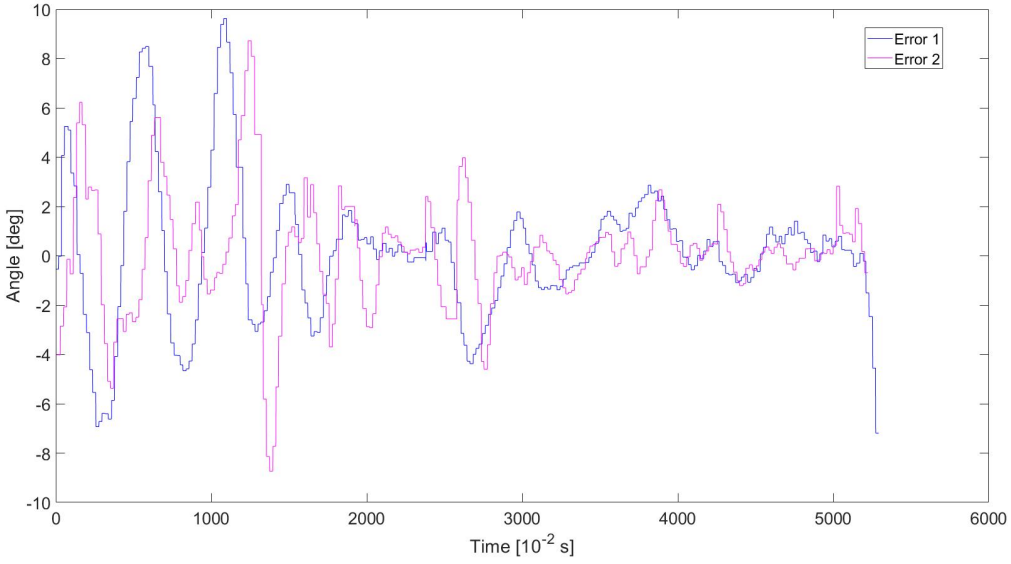


Figure 6.9: Showing the angular difference to the preceding vehicle for each TurtleBot throughout their continuous trajectory at a speed of 0.23 m/s

6.3.4 Object Detection Accuracy

From running several platooning tests with object detection while outputting the detections in RVIZ it was shown that the global coordinates of the detected object most of the time was somewhat accurate. In Figure 6.10 a snapshot from one of these test runs can be seen. Here the red sphere is the Prescan Vehicle which the TurtleBots follow, while the blue square is the detection of a truck and the green sphere is a detection of a car. In the top left corner of the picture the live feed from the TurtleBots cameras can be seen. It can be mentioned that it looks like the last TurtleBot in the platoon is running behind but this is just an bug in RVIZ, the base scan frames that can be seen corresponds to the real position of the TurtleBots in real life.

Figure 6.10: Showing the platoon and the object detection in RVIZ. The red sphere is the Prescan Vehicle, the blue square is detection of a truck while the green sphere is a detection of a car.

6.3.5 Software Defined Radios

The SDRs needs a hardware amplifier to amplify the signals coming out from the SDR to the antennas in order to increase the range (transmitter power). In this project, due to challenges and time limitations, the range was not at a high priority and was left for a future work. See the suggested future work in section 8.2.3 for more on how the range can be increased.

The integration between GNU Radio and ROS through publishing to a UDP socket was tested in one direction from GNU Radio to ROS and was proven to work in. However, further testing is required for sending information from ROS to GNU Radio and adjusting GNU Radio flow graph behavior accordance to prove that the GNU Radio-ROS integration works in both directions.

6.4 System Test Results

The system test was performed and the results were satisfactory. The platoon could be done through a loop inside the testing space, objects could be detected with reasonable accuracy. Moreover the platoon successfully stopped when a safety message was applied.

The SDR concept for ITS-G5 communication is proved to be working with the Road Side Unit. The Road Side Unit is able to send and receive in transceiver mode in terms of the right frequency band for the ITS-G5.

6.4.1 Resulting RQT-graph of the Whole System

Figure 6.11 show the resulting RQT-graph while the whole system is running. This includes the platooning algorithm with AMCL for localization of the TurtleBOts and the controllers as well as the object detection package. What is not shown in the RQT-graph is the script client for the GNU-radio software which enables the possibility to retrieve a message on the allocated frequency band for V2X communication and then feed-forward the message using a ROS service to stop the TurtleBots in the platoon.

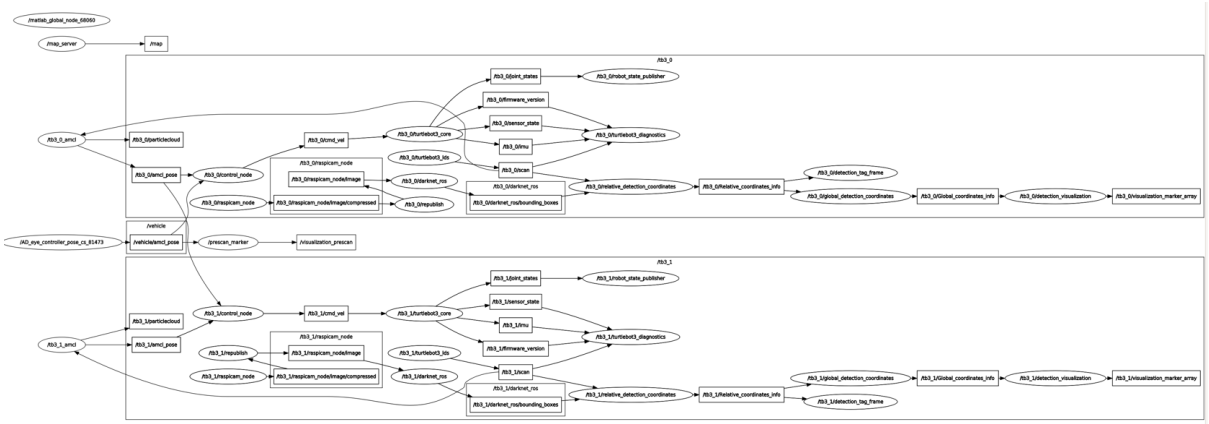


Figure 6.11: The resulting RQT graph of the whole system

6.5 Heat Flow Simulation Results

From the simulations in Simcenter FLOEFD, some findings could be made. The referenced scenarios in this section are described in section 5.3.1.

The airflow inside the RSU can be seen i Figure 6.12 and 6.13 These specific images are from scenario 1 where the outside temperature is 20 °C, but the velocity of the flow looks the same for scenario 2. Next, the specific local temperatures of the of the CPU and motherboard was found for all scenarios, an example of how it looks can be seen in Figure 6.14 The surface temperature of the RSU was also found, the spots with maximal temperature for scenario 3 can be found in Figure 6.15 Lastly, the surface temperature of the computing components as well as for those near the heater in scenario 3 can be found in Figure 6.16. More pictures of the simulations can be found in Appendix **??**.

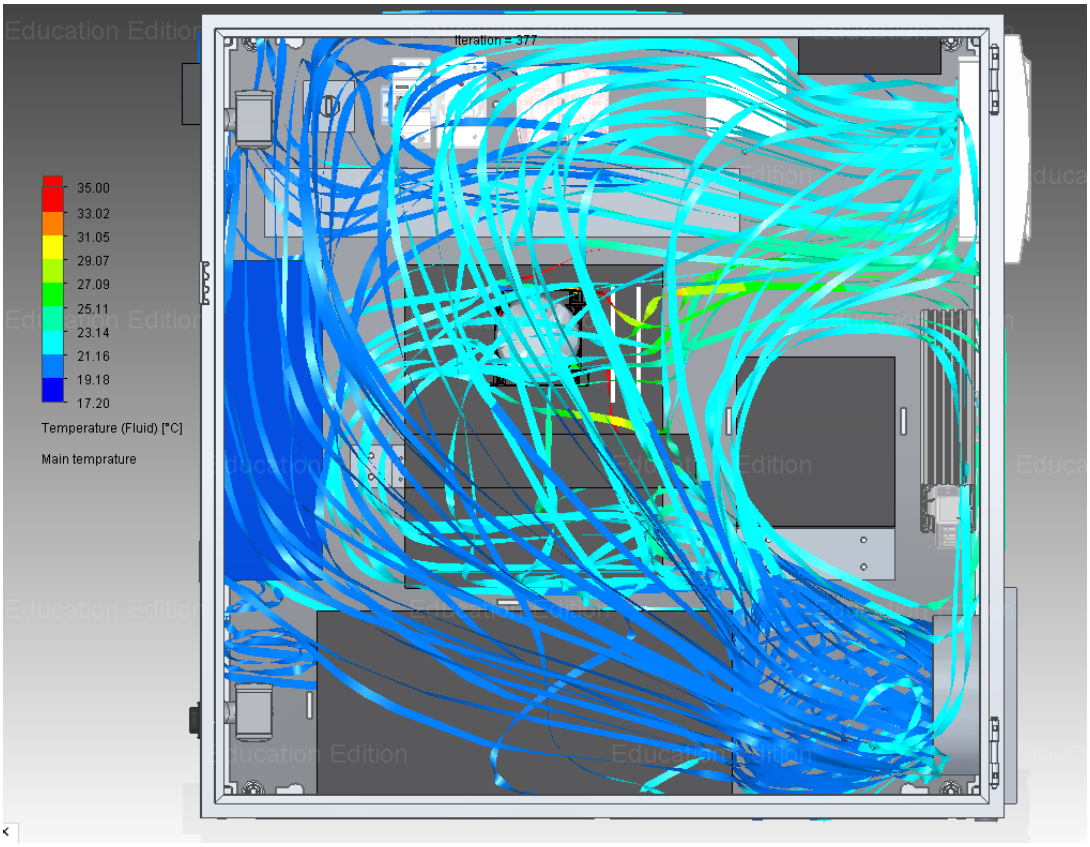


Figure 6.12: Airflow temperature in RSU scenario 1

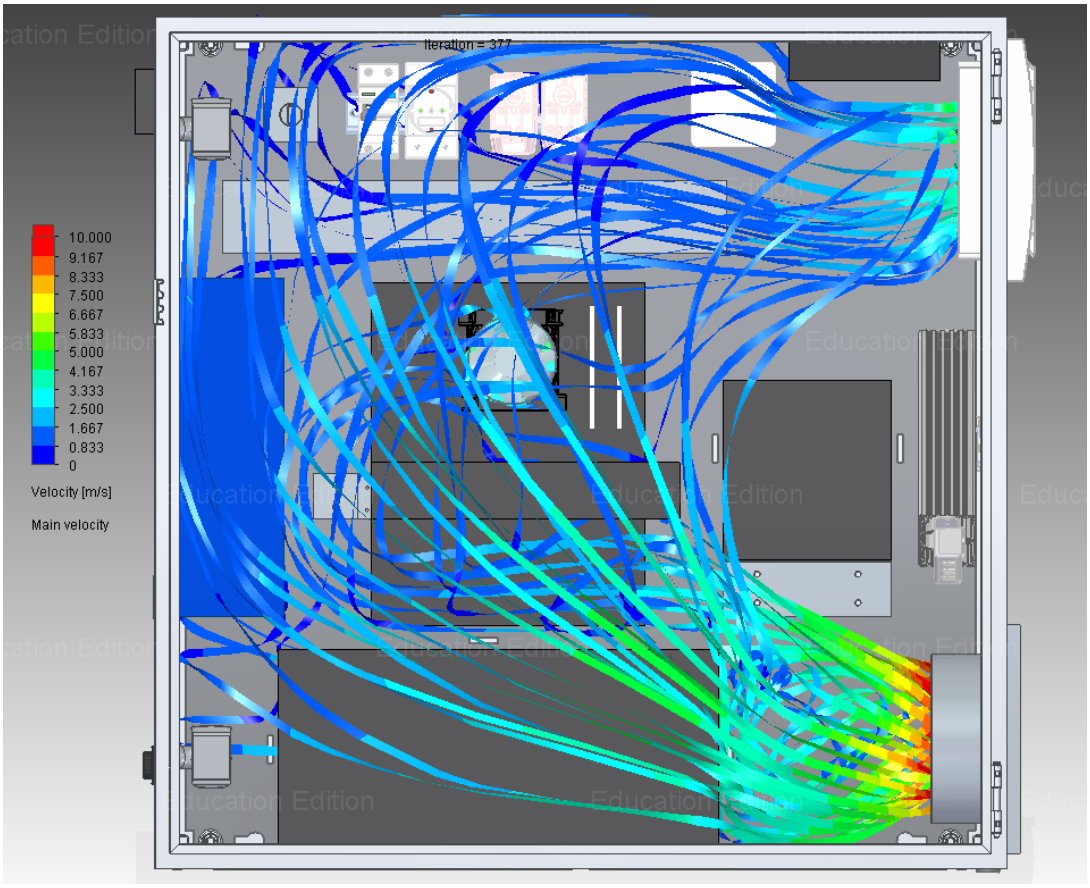


Figure 6.13: Airflow velocity in RSU scenario 1

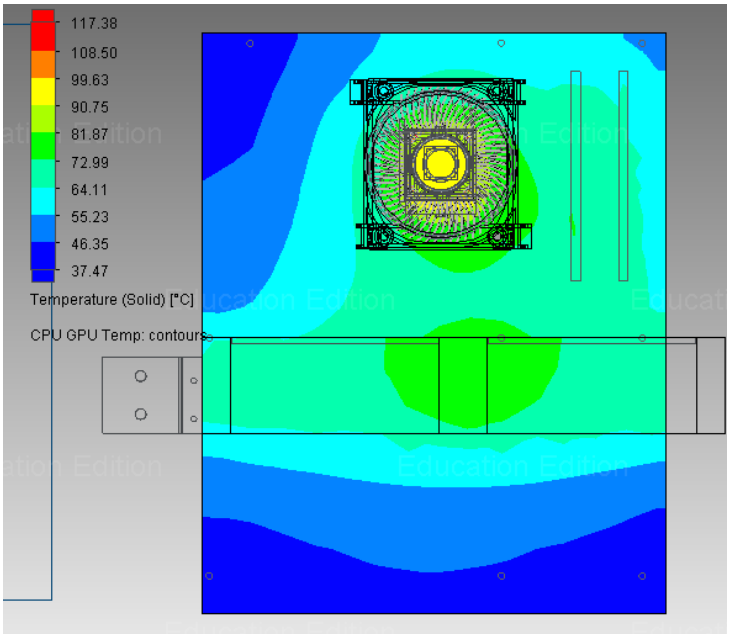


Figure 6.14: Motherboard + CPU temperature scenario 1

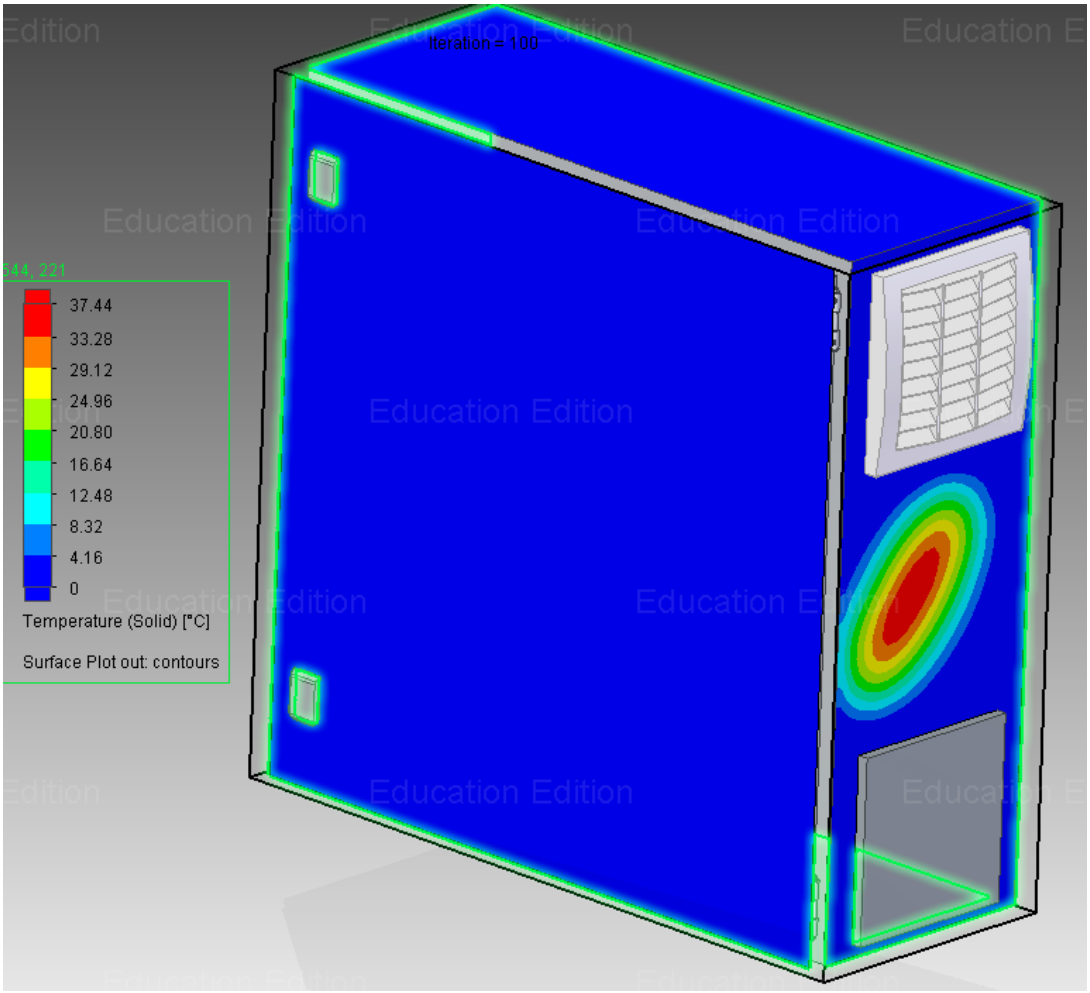


Figure 6.15: Hottest area on RSU surface in scenario 3

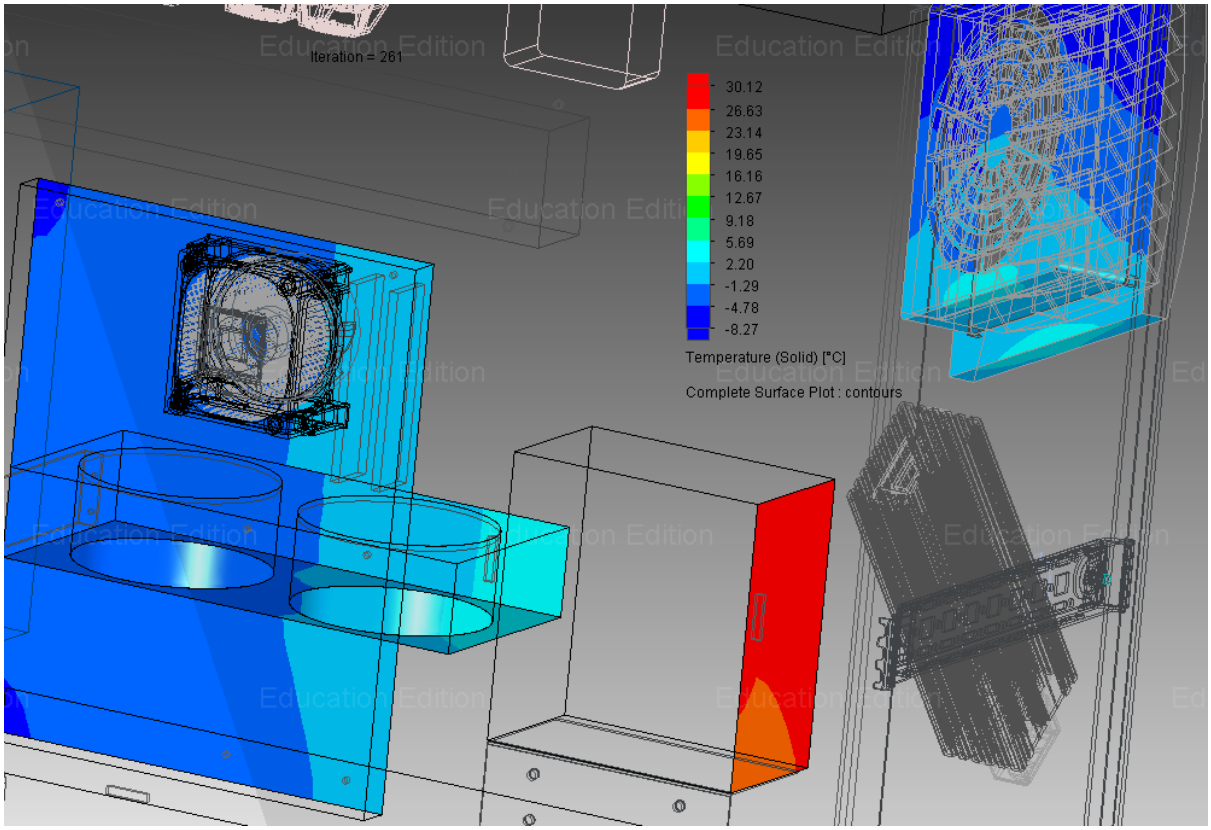


Figure 6.16: Surface temperature of critical components in scenario 3

From the simulations the max temperature of the motherboard and CPU was found. It was 110 °C in scenario 1, 122 °C in scenario 2 and 1 °C in scenario 3. As we can see, The CPU is as such overheating in scenario 1 and 2. The Reason for this is that the current CPU fan is under dimensioned. From scenario 3, we learn that the maximum temperature of the components surrounding the heater is 30 °C and occurs in the PSU, while the exit filter only reaches 3 °C. This is well within he acceptable range and should as such not be a cause of failure, as long as the heater is only running at sub zero temperatures. The thermostat responsible for controlling the heater will only reach -4 °C during scenario 3. The lowest setting for this thermostat is 0 °C, which means that the heater alone will not be able the heat the enclosure enough to turn itself off.

6.6 Fulfillment of Requirements

Table 6.2 shows the stakeholder requirements and whether they have been met or not and table 6.3 shows the technical requirements and whether they have been met or not.

Table 6.2: Fulfillment of stakeholder requirements

Requirement	**Fulfilled?**	**Comment**
1. The RSU shall be able to transmit camera and LiDAR data wirelessly to on-road entities and the virtual vehicle	Yes	
2. The RSU, OBU and virtual vehicle shall be able to communicate over DSRC	Partly	The communication could be established between the RSU and an external computer
3. The RSU shall communicate to a backbone network	Yes	
4. The RSU shall be robustly constructed to withstand outer elements	Yes	
5. The RSU shall be constructed for placement on the ground	Yes	
6. The RSU shall be constructed with modularity in mind	Yes	
7. The RSU's computer shall be able to perform computationally intensive tasks	Yes	
8. The RSU shall be trackable	Yes	
9. The RSU shall be able to operate using an external power source other than the power grid	Yes	
10. The RSU shall be able to get manufactured for a cost less than 25000 SEK	No	
11. The OBU shall be able to transmit camera and LiDAR data wirelessly to the RSU	Yes	If TurtleBots counts as OBU in this case
12. The OBU shall be powered by a cars power outlet or an external power source	No	No OBU

Table 6.3: Fulfillment of technical requirements

Requirement	**Fulfilled?**	**Comment**
1.1 The communication shall support a data-rate of at least 50 MBps	Yes	
2.1 The communication shall be done using ITS-G5 messages	Partly	The messages sent are not CAM
3.1 The communication shall take place over Ethernet	Yes	
4.1 The RSU shall be robust against vibration	Yes	
4.2 The RSU shall have IP54 or above rating in ingress protection	Yes	
4.3 The RSU shall be operational with little to no degradation between -15 to +35 degrees	Partly	A sufficiently strong CPU fan has to be installed to sustain high temperatures
4.4 The RSU shall be constructed with a frame capable of carrying a load of at least 50 kilos without deformation for extended periods of time	Yes	
6.1 The RSU shall have the possibility of extending the sensors	Yes	
6.2 The RSU shall have a mount on top of the RSU to secure position of a camera	No	
7.1 The RSU's PC shall have a processor with X86-64 architecture with atleast 6 cores	Yes	
7.2 The RSU's PC shall have at least a single consumer grade GPU: GTX 1080 or above	Yes	
7.3 The RSU's PC shall have Ubuntu 16.04 and ROS kinetic installed	Yes	
8.1 The RSU shall be equipped with a GPS	Yes	
9.1 The RSU shall have the ability to run on battery power for at least an hour	Yes	
10.1 The ITS-G5 transceiver shall cost less than 750$ and be implemented from scratch since the existing ITS-G5 modules on the market is too expensive	No	The cheaper WiFi card solution is left for future work
11.1 The communication shall support a data rate of at least 20 MBps	Yes	
12.1 The OBU shall be powered by standard automotive aux power outlet (12 V) or a battery with the same voltage	No	No OBU

Chapter 7

Discussions and Conclusions

7.1 Platooning

As can be seen from the results in section 6.3.2 the TurtleBots have a hard time following the simulated Prescan vehicles trajectory during curves. This is due to how the platooning algorithm is structured. The platooning algorithm store the waypoints of each vehicle in the platoon in lists and clean out waypoints if the euclidean distance between the oldest waypoint and the current position of the TurtleBot is below a certain threshold. Since the euclidean distance will become shorter in a curve compared to on a straight road more waypoints will be removed and therefore some information of the trajectory of the preceding will be lost which will lead to the inability of the TurtleBot to follow the trajectory accurately during curves. To combat this issue one could try to decrease the threshold for which the waypoints are cleaned out this would decrease the loss of information of the preceeding vehicles trajectory.

The performance of the longitudinal and lateral PI controllers, as can be seen in section 6.3.3, were not perfect. The error signals for the longitudinal PI controller and the angular difference to the preceding vehicle for the lateral PI controller fluctuate a bit around the reference values. One main problem when tuning the controllers was that the floor in the room where the platoon took place was quite slippery. Due to how the TurtleBots are constructed, as a differential robot, with small metal balls in the rear of the TurtleBots body, it caused the TurtleBot to either get stuck or randomly drift at some points throughout its trajectory. This also explains why the controllers did perform a little bit better at lower speeds, since lower speed usually caused the TurtleBots to drift less. For this project it was deemed sufficient to use a PI controller due to the low inertia and speed of the TurtleBots. However if a controller for a platooning scenario would be implemented on a real vehicle a more advanced controller would have to be implemented. For

this a more modern state of the art controller could be used such as a MPC controller, which have also been discussed in the SOTA chapter in section 2.7. There is some different variations of a MPC controller but the main advantage is that a MPC controller solve a optimization problem at every time step, which make it more robust against sudden disturbances. However, this also increase the computational power needed compared to a more simple PI or PID controller.

7.2 V2X (ITS-G5)

The ITS-G5 solution was in its nature a challenging part with many learning curves due to the lack of the necessary background knowledge in software radios, signal processing, antenna theory and other subjects. In addition, the development environment was the operating system Ubuntu 16.04, which introduced further complications and delays for the V2X setup installations to be completed successfully. This had a huge effect on the time plan for developing, implementation, and integrating the SDR V2X solution. These factors, paired with the knowledge that the implementation of separate OBU units would require several ROS cores throughout the system, lead to a decision together with the stakeholder that OBUs were not to be developed.

The range of the signals are limited due to the limited voltage signal coming out of the SDRs. The bladeRF 2.0 micro features bias-tee circuitry that allows the device to power amplifiers through the SMA ports before the antennas to enhance the signal voltage and increase the range in accordance, for more see Future Work for the Software Defined Radio 8.2.2.

The setup detailed in section 8.2.1 was successfully installed, but not tested. It is highly suggested that more work be put into this solution as opposed to the SDR as it is a lot cheaper. The price for one of these WiFi cards is roughly 300 SEK [107] and the price for one bladeRF xA4 is rougly 6600 SEK [108], and the SDR will need a more expensive external amplifier.

7.3 Battery time test

In section 6.2.2, the battery is presented as lasting for two hours, clearly passing technical requirement 9.1 in section 1.3.2. There's a limitation in the test however, in the CPU not being fully utilizable due to the underdimensioned CPU fan. The test is therefore only an indication of an acceptable battery time. Please see section 8.1.4.

7.4 Cost

The cost of developing the RSU was 32 811 SEK. The main part of this cost was allocated in computer components, costs associated with adding battery functionality, and the ITS-G5 module. Out of these, the easiest to reduce would be the ITS-G5, given that it is possible to fully develop the WiFi card solution. Implementing that instead of having the BladeRF would drop the cost with around 4000kr. Other then that, dropping the costs will be hard, as most components chosen were the cheapest alternative. It could be done by dropping some of the requirements. The most effective requirement to drop from a cost perspective would be the battery one, as the battery and DC-AC converter, and electricity switch could then be skipped. This would also free up a lot of space which could mean that a smaller and less expensive enclosure could be used. On top of that Less cables would be needed.

Chapter 8

Future Work

This chapter will cover the concepts not yet implemented and suggestions for future work in all areas of the project.

8.1 RSU

This section will cover the suggested improvements that can be done to the RSU. A combined hygrometer and thermostat should be added to the RSU to easily determine if the air inside is suitable.

8.1.1 Measuring the Battery Voltage

The car battery voltage can be measured using a Raspberry Pi pico[109] with a voltage divider circuit, the concept is shown in Figure 8.1. This circuit provides logic level voltages to the ADC on the Pi pico, which supports up to 3.3 volts. The circuit in 8.1 is to be connected in parallel to the battery and supports battery voltages of up to 19.8 volts which includes a safety margin to the expected maximum battery voltage. The Pi pico is powered by USB and can be connected directly to the motherboard in the RSU. This allows easy access to the device from a terminal window in the RSU, to see the measured voltages or run a script which prints the battery voltages in real time. Pin 31 is the first ADC pin, for a full pinout see Figure 2 in [109].

This concept lacks an important part however, and that is some safety mechanisms such as fuses. This should be considered when finalizing the concept, as the Pi pico is meant to be directly connected to the motherboard.

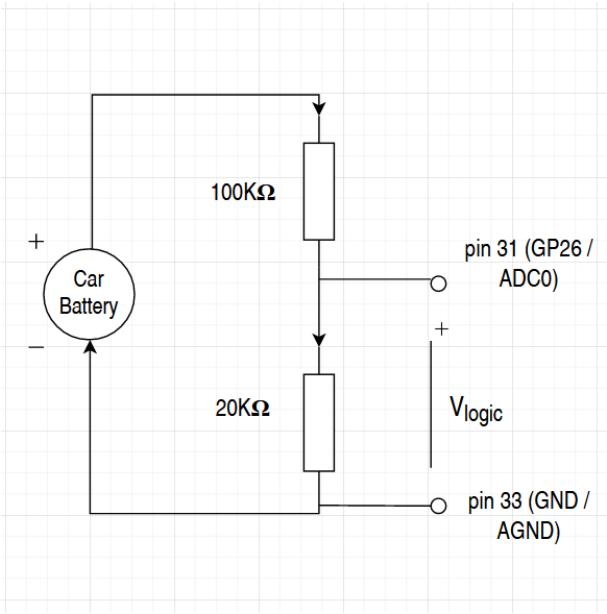


Figure 8.1: Voltage divider circuit for logic level voltages.

Further research is also needed in understanding the voltage changes for lead acid batteries, particularly under demanding loads, and the RSU is expected to pull up to 90 A. This knowledge is needed in order to convert the sampled voltage levels to an estimated battery charge percentage. Lead acid batteries are not intended to be deep cycled either, where more than 50% of the capacity is pulled [110].

The requirement for the battery powering was that it should be able to last for an hour. In theory, as seen in section 4.2.6, the energy used is equal to roughly 65% of the total battery capacity. Such deep cycles are not recommended as they reduce the lifespan of the battery [110].

The implication is that the Pi pico voltmeter should ideally be developed as part of a larger battery management system (BMS), which could be mostly software based if the behaviour of the lead acid battery is fully understood.

Finally, it was discovered during testing that the DC-AC inverter begins to beep loudly when it's been running for on battery power for a long time. This is very likely an indication of low battery voltage, meaning the battery capacity is low. It might be possible to read this signal from the circuitry instead, although it's not clear at what level the inverter begins to beep.

8.1.2 System Communication

The original system communication concept throughout the system consisted of two different setups. In general both concepts included two Onboard Units (OBUs), one for each TurtleBot, which would be powered through a standard automotive aux power outlet or by a battery. The OBUs would then be connected to the TurtleBots by an Ethernet cable. Each OBU would have the capability to both send and receive ITS-G5 messages between the TurtleBots and the RSU using the allocated frequency band for ITS-G5 communication, as well as retrieve LiDAR and camera data from the TurtleBots and feed forward the data to the RSU using WiFi. The RSU would include a WiFi router which is both connected to a backbone network and the RSU computer which run ROS kinetic. Furthermore, the RSU would also include an ITS-G5 module.

The main difference between the two concepts is the hardware configuration of the ITS-G5 modules. In the first concept, which can be seen in Figure 8.2, the ITS-G5 module consist of a (model) WiFi card and a APU2E0 router running the OpenWRT software. In this case the OBU would also include a raspberry pi with ROS kinetic installed, which would run the controllers for the TurtleBots as well as feedforward camera and LiDAR data to the RSU. To connect the Raspberry Pi and ITS-G5 module to the TurtleBot, a network switch would be used. In the second concept that can be seen in figure 8.3, the ITS-G5 solution is instead based on a software defined radio, or more precisely a bladeRF using the GNU-radio software. This would allow for the ITS-G5 module, the controllers scripts as well as the feed forwarding of camera and LiDAR data to be run from the same unit. However, this would require a more powerful minicomputer. In the OBU, a minicomputer such as the Intel NUC could be used with both ROS kinetic and GNU-radio installed. The bladeRF would then be connected through USB.

A current issue with both of the mentioned concepts is that this would require several ROS cores to be run throughout the system; one for each TurtleBot, OBU's and the RSU. This project has the requirement of using Ubuntu 16.04 and ROS kinetic throughout the system. In ROS kinetic, which is part of ROS 1, communication between several ROS cores are not supported. One could possibly try to get communication going by syncing the clocks between different ROS cores, but this is tedious work and not recommended. Therefore the mentioned concepts could not be implemented. However, if the Ubuntu operating system would be upgraded to a more recent version of Ubuntu in the future then ROS 2 could be used, which is developed to work in a distributed system that require communication between several ROS cores. This would then enable to implementation of separate OBU units.

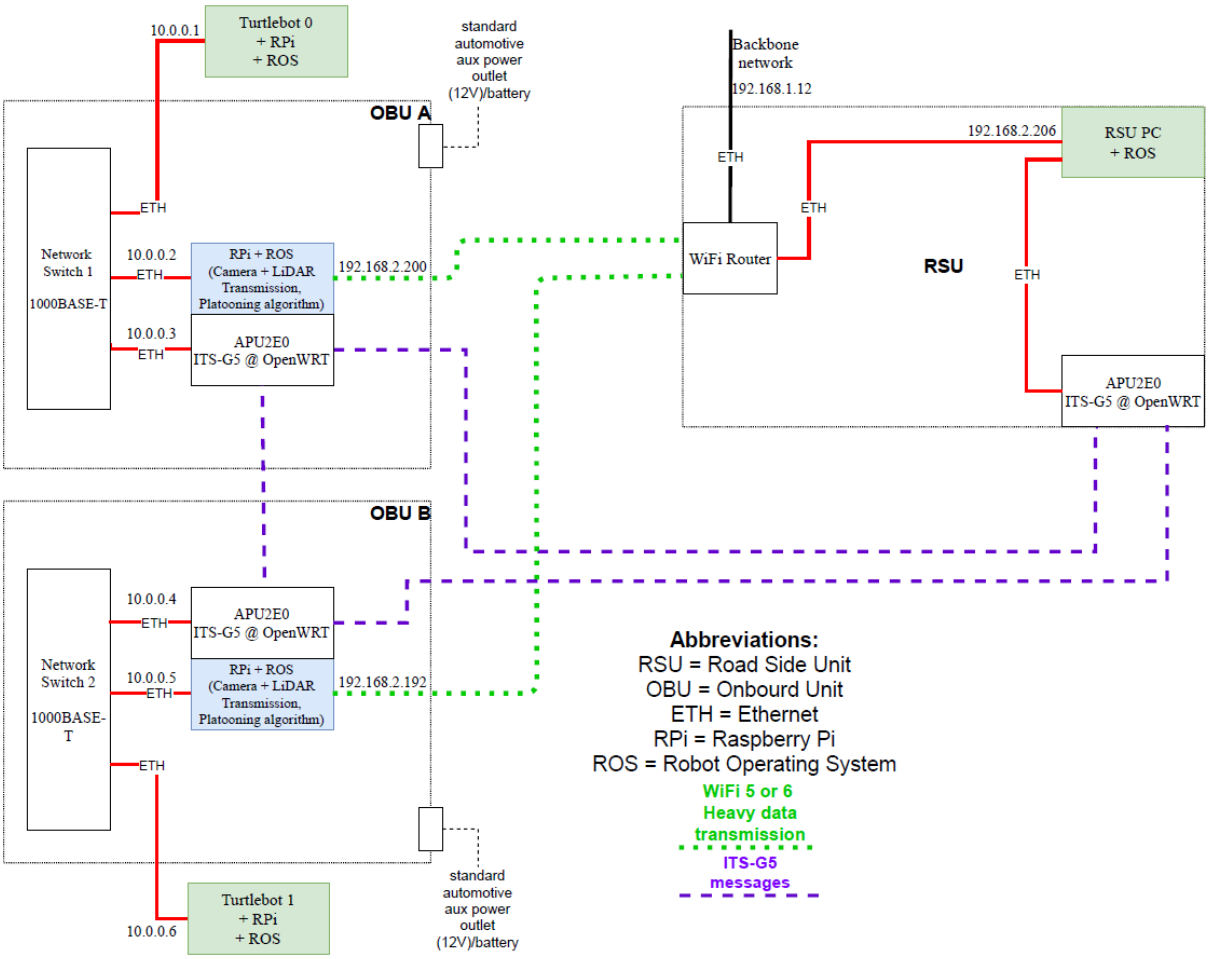


Figure 8.2: System communication throughout the system using a WiFi-card + router as ITS-G5 unit.

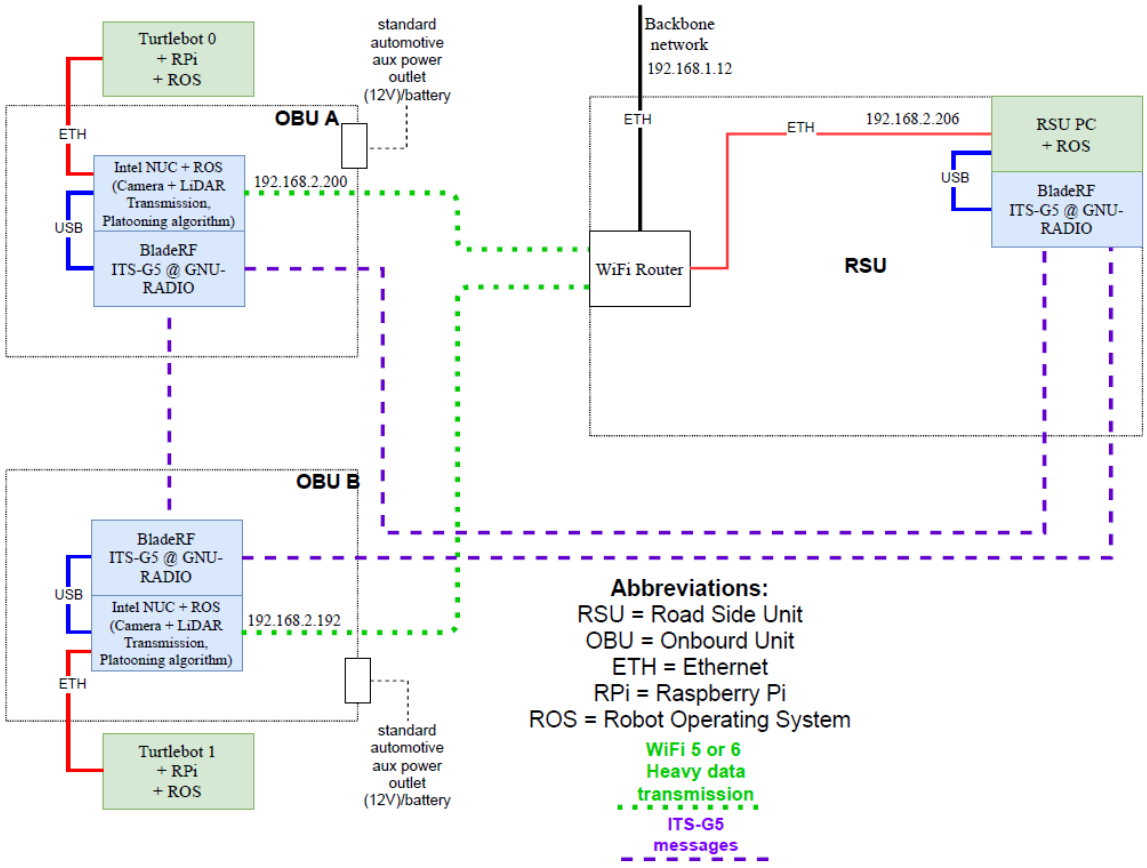


Figure 8.3: System communication throughout the system using a bladeRF + GNU Radio as ITS-G5 unit.

8.1.3 GPS

Technical requirement 8.1 in section 1.3.2 states that the RSU should include a GPS. A GPS device was purchased but not implemented. This means that the RSU is not trackable in its current state. The cable entry design in section 4.2.15 allows for easy installation of the device.

8.1.4 CPU fan

The CPU fan implemented is the Arctic Alpine 12 [**alpine_cooler**] which is dimensioned for CPUs with a TDP of up to 95 W. This is not a suitable cooler for the Intel core i7 10700K as it has been shown to boost upwards 200 W [67]. A different cooler is needed in order to fully utilize the CPU.

8.2 ITS-G5

This section will cover the suggested future work in developing the ITS-G5 solution used in the RSU and OBU. First of all, in the interest of verifying the functionality of any developed radio module, it is reasonable to suggest a purchase of at least one industrial grade ITS-G5 solution, complete with any necessary documentation and other support channels for properly setting up and using the device(s).

8.2.1 WiFi Card Solution

After the SOTA was done, a different implementation of the WiFi card solution, mentioned in section 2.2.1, was discovered [111]. This setup is based on modifying the ath9k driver on a Debian based system instead of using OpenWrt. A lot of modifications are also done to make the setup compatible with Linux tools such as iw and wireless-regdb. The entire implementation has been automated with Ansible, and it is highly suggested to use the automated version detailed in this repository [112]. It was designed to be used on Debian buster, but after submitting several support tickets on the Gitlab repository, the setup is now compatible with Debian Bullseye and Ubuntu focal as well, thanks to the generous help of repository contributor Lukas Pirl. Installation guidelines are provided in appendix E.4.

The suggested WiFi card to use with this setup is called MikroTik R11-5HnD [113], it's also the one acquired for this project. The card is capable of transmitting signals at powers of up to 500 mW (27 dBm) [113], and the authors of [112] have been able to use as much as 400 mW (26 dBm) in their implementation. This means that the MikroTik card is much more capable than the SDRs in terms of transmitting power, and would need a lot less amplification in comparison. This should be taken into consideration in the expansion of this project, as it allows for a cheaper amplifier to be used.

Finally, if this setup is preferable to the SDR, the OBU concept would need to be modified to allow for a full sized mini PCI-e card to be added. The card uses MMCX connectors for external antennas, these would need appropriate adapters.

8.2.2 Software Defined Radio (SDR)

The ITS-G5 solution using a SDR was never fully developed, this means lots of improvements can be made. Missing in the solution for now is transmitting and receiving between vehicles. For this to be done, a full duplex SDR needs to be installed in every vehicle that should communicate

with each other. A solution for receiving messages in GNU Radio needs to be developed, as for now it is only possible to send from GNU Radio to ROS and stop the vehicles that way.

8.2.3 Transmit Power (TX power)

The transmitting power of the HackRF at 5.9GHz is approximately 0.1 mW (or -10 dBm) according to the official documentation site [114]. An approximation for the bladeRF could not be found, but no remarkable difference in range performance could be found in testing alongside the HackRF.

The ETSI has provided different power limitations based on what the use case is. The strictest limit is 10 dBm or 10 mW which is the limit for coexistence mode A [115]. This allows the HackRF to be amplified by up to 20 dB without considering other limitations. The amplification can be done using a low-noise amplifier (LNA). Coexistence mode D allows TX powers of up to 33 dBm or 2000 mW [115], which would require an amplification of 43 dB.

In regards to the WiFi card, it fully supports coexistence mode D without an external amplifier. For other modes, it could be beneficial in terms of transmission range to amplify the signal further. In this case, some research is required to find the best signal to noise ratio that can be achieved by the combinations of card output power and external amplification.

Furthermore, it should be noted that external amplification requires external power, i.e. not from the WiFi card or SDR. This could introduce additional challenges in fitting an amplifier in an existing OBU or RSU.

8.2.4 ITS-G5 messages

The standardized messages intended for ITS-G5 communication were not implemented in this project. Further research is needed in understanding and generating these messages, presented here in [116]. The Cooperative Awareness Message (CAM) should be studied in particular, as it is the primary message type for inter-vehicle communication of parameters such as speed and position [117]. The other message type, Decentralized Environment Notification Message (DENM), is used to communicate special circumstances such as nearby roadwork or pressence of vehicles carrying dangerous goods [117].

It should be considered, if the project is to be redone with TurtleBots, to reconstruct the controllers to work directly from the messages received on the ITS-G5 channel. This increases

the validity of the test case since the ITS channels are intended for safety critical use, and decreases ROS dependency which increases overall robustness.

8.3 Platooning

This section discusses possible improvements in the implementation of any methods used for platooning in this project.

8.3.1 Control algorithm

The implementation of control algorithms were purposefully kept simple. The main reason was to retain low computational complexity, as the intended platform that these controllers would run on was the OBU. These OBUs' main task would be dealing with software radio and communication. As such, the project required light weight controllers that could deliver reasonable performance. However, as focus shifted away from building OBUs, this reasoning became obsolete for this current project.

Depending on the resources available for the controller platform, the algorithms could be extended to include more advanced techniques as described in 2.7. The benefits would include a better response to communication problems, disturbance mitigation, and better reference following.

8.3.2 Localization

Future testing in outdoor environments can definitely benefit by implementing GNSS into the project. The main benefits would be a better stability of localization (in areas of coverage), as well as gaining ground truth trajectories that the rest of the system can be compared to. For the latter, it would be especially beneficial to use some form of differential GPS for high accuracy evaluation.

8.3.3 Vehicles

The accuracy of testing was severely hampered by the limitations of the TurtleBot design. The rear of the robot moved on small metal balls, which tended to gather up all the dust and dirt on

the floor. Even after cleaning the testing surface thoroughly, the worn state of these balls resulted in them jamming often. This led to the robots getting stuck on nothing, particularly when turning and taking curves. Overall this became a problem for the localization and control scheme.

To support the project in future endeavours, either the robustness of the controller and localization schemes need to be attuned to such disturbances or a different vehicle could be selected for testing. Another option is to replace the parts of the TurtleBot.

8.4 Antennas

As mentioned in section 2.5 it is important to do field tests to find an appropriate antenna for a certain application. This holds for both WiFi antennas and ITS-G5 antennas. Since field testing of different antennas was not done it this project this is something that can be done in the future to ensure that the best possible range and throughput between different units throughout the system is achieved.

Bibliography

[1] Gaurang Naik, Biplav Choudhury, and Jung-Min Park. "IEEE 802.11bd 5G NR V2X: Evolution of Radio Access Technologies for V2X Communications". In: *IEEE Access* 7 (2019), pp. 70169–70184. DOI: 10.1109/ACCESS.2019.2919489.

[2] CAR 2 CAR Communication consortium. *Position Paper on Road Safety and Road Efficiency Spectrum Needs in the 5.9 Ghz for C-ITS and Cooperative Automated Driving*. [Online]. URL: https://www.car-2-car.org/fileadmin/documents/General_Documents/C2CCC_TR_2050_Spectrum_Needs.pdf (visited on May 4, 2021).

[3] ETSI. *ETSI EN 302 663 V1.3.1: Intelligent Transport Systems(ITS); ITS-G5 Access layer specification for Intelligent Transport Systems operating in the 5 GHz frequency band*. [Online]. URL: https://www.etsi.org/deliver/etsi_en/302600_302699/302663/01.03.01_60/en_302663v010301p.pdf (visited on May 4, 2021).

[4] Giammarco Cecchini et al. "Performance comparison between IEEE 802.11p and LTE-V2V in-coverage and out-of-coverage for cooperative awareness". In: *2017 IEEE Vehicular Networking Conference (VNC)*. 2017, pp. 109–114. DOI: 10.1109/VNC.2017.8275637.

[5] H. Zhou et al. "Evolutionary V2X Technologies Toward the Internet of Vehicles: Challenges and Opportunities". In: *Proceedings of the IEEE* 108.2 (2020), pp. 308–323. DOI: 10.1109/JPROC.2019.2961937.

[6] 5GAA Automotive Association. *V2X Technology Benchmark Testing September 2018*. [Online]. URL: https://ecfsapi.fcc.gov/file/109271050222769/5GAA%209.25.18%20Ex%20Parte%20Notice.pdf (visited on May 11, 2021).

[7] 3GPP. *Release 16*. [Online]. URL: https://www.3gpp.org/release-16 (visited on May 11, 2021).

[8] Marshall Brain. *How Radio Works*. Dec. 2000. URL: https://electronics.howstuffworks.com/radio.htm (visited on May 5, 2021).

[9] AMAN KUMAR GULIA. *A Simulation Study on the Performance Comparison of the V2X Communication Systems: ITS-G5 and C-V2X*. URL: http://kth.diva-portal.org/smash/get/diva2:1422828/FULLTEXT01.pdf (visited on May 5, 2021).

[10] Unex. *SOM-301E - ITS-G5 EU Stack - GEN.2 - V2X Module*. URL: https://www.unex.com.tw/products/v2x/v2xsolution/v2xmodule/v2x-subsystem-module/detail/som-301e (visited on May 23, 2021).

[11] Sven Laux et al. "Demo: OpenC2X — An open source experimental and prototyping platform supporting ETSI ITS-G5". In: *2016 IEEE Vehicular Networking Conference (VNC)*. 2016, pp. 1–2. DOI: 10.1109/VNC.2016.7835955.

[12] Florian Klingler et al. "Field Testing Vehicular Networks using OpenC2X". In: *15th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys 2017), Poster Session*. Niagara Falls, NY: ACM, June 2017, pp. 178–178. DOI: 10.1145/3081333.3089322.

[13] *Reasons to use OpenWrt*. Aug. 2020. URL: https://openwrt.org/reasons_to_use_openwrt (visited on May 23, 2021).

[14] Kalle Vallo. *ath9k*. Jan. 2017. URL: https://wireless.wiki.kernel.org/en/users/drivers/ath9k (visited on May 23, 2021).

[15] Francesco Raviglione, Marco Malinverno, and Claudio Casetti. "Characterization and Performance Evaluation of IEEE 802.11p NICs". In: *Proceedings of the 1st ACM MobiHoc Workshop on Technologies, MOdels, and Protocols for Cooperative Connected Cars*. TOP-Cars '19. Catania, Italy: Association for Computing Machinery, 2019, pp. 13–18. ISBN: 9781450368070. DOI: 10.1145/3331054.3331548. URL: https://doi.org/10.1145/3331054.3331548.

[16] Francesco Raviglione. *OpenWrt-V2X*. [commit: 91c733827a3126f3fe876bb3880944d95d7a2139]. 2019. URL: https://github.com/francescoraves483/OpenWrt-V2X (visited on May 23, 2021).

[17] Bernhard Kloiber et al. "Random Transmit Power Control for DSRC and its Application to Cooperative Safety". In: *IEEE Transactions on Dependable and Secure Computing* 13.1 (2016), pp. 18–31. DOI: 10.1109/TDSC.2015.2449845.

[18] *Software Defined Radio*. URL: https://www.sciencedirect.com/topics/engineering/software-defined-radio (visited on May 23, 2021).

[19] *Software Defined Radio Market Size: SDR Industry Report, 2027*. URL: https://www.grandviewresearch.com/industry-analysis/software-defined-radio-sdr-market (visited on May 23, 2021).

[20] *Software Defined Radio Market*. URL: https://www.marketsandmarkets.com/Market-Reports/software-defined-radio-market-138946173.html (visited on May 23, 2021).

[21] *Software Defined Radio Market: Global Industry Trends, Share, Size, Growth, Opportunity and Forecast 2021-2026*. URL: https://www.imarcgroup.com/software-defined-radio-market (visited on May 23, 2021).

[22] *Main Page*. URL: https://wiki.gnuradio.org/index.php/Main_Page (visited on May 23, 2021).

[23] Bastian Bloessl et al. "Performance Assessment of IEEE 802.11p with an Open Source SDR-Based Prototype". In: *IEEE Transactions on Mobile Computing* 17.5 (2018), pp. 1162–1175. DOI: 10.1109/TMC.2017.2751474.

[24] *Vehicle2X communication*. URL: https://www.mobility.siemens.com/global/en/portfolio/road/traffic-management/connected-mobility/vehicle2x.html (visited on May 23, 2021).

[25] *Tolling Solutions: Q-Free*. May 2021. URL: https://www.q-free.com/tolling/ (visited on May 23, 2021).

[26] *Unex OBU*. May 2021. URL: https://www.unex.com.tw/products/v2x/v2vsolution/on-board-unit/etsi-tc-its-stack-pr/detail/obu-301e (visited on May 23, 2021).

[27] NXP. *Intelligent Roadside Unit by NXP*. [Online]. URL: https://www.nxp.com/applications/automotive/connectivity/intelligent-roadside-unit:INTELLIGENTRSU (visited on May 22, 2021).

[28] Supermicro. *Outdoor Edge Systems*. [Online]. URL: https://www.supermicro.com/en/products/outdoor-edge (visited on May 22, 2021).

[29] Essential IT Solutions. *UPS battery*. [Online] Retrieved from https://essentialitsolutions.co.za/backup-power.html (visited on May 4, 2021).

[30] Adel Ismail Al-Alawi. "WiFi technology: Future market challenges and opportunities". In: *Journal of computer science* 2.1 (2006), pp. 13–18.

[31] "IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications". In: *IEEE Std 802.11-1997* (1997), pp. 1–445. DOI: 10.1109/IEEESTD.1997.85951.

[32] "IEEE Approved Draft Standard for Information technology– Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 1: Enhancements for High Efficiency WLAN". In: *IEEE P802.11ax/D8.0, October 2020* (2021), pp. 1–820.

[33] Liangkai Liu, Baofu Wu, and Weisong Shi. "A Comparison of Communication Mechanisms in Vehicular Edge Computing". In: *3rd USENIX Workshop on Hot Topics in Edge Computing (HotEdge 20)*. USENIX Association, June 2020. URL: https://www.usenix.org/conference/hotedge20/presentation/liu-liangkai.

[34] "IEEE Standard for Information technology–Telecommunications and information exchange between systems—Local and metropolitan area networks–Specific requirements–Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications–Amendment 4: Enhancements for Very High Throughput for Operation in Bands below 6 GHz." In: *IEEE Std 802.11ac(TM)-2013 (Amendment to IEEE Std 802.11-2012, as amended by IEEE Std 802.11ae-2012, IEEE Std 802.11aa-2012, and IEEE Std 802.11ad-2012)* (2013), pp. 1–425. DOI: 10.1109/IEEESTD.2013.7797535.

[35] Jack Browne. *What's the Difference Between Wi-Fi 5 and Wi-Fi 6?* URL: https://www.mwrf.com/technologies/systems/article/21849959/whats-the-difference-between-wifi-5-and-wifi-6 (visited on May 23, 2021).

[36] Ian Fogg. *Benchmarking the Global 5G Experience*. [Online]. URL: https://www.opensignal.com/2021/04/15/benchmarking-the-global-5g-experience-april-2021 (visited on May 20, 2021).

[37] Waqar Anwar et al. "PHY Abstraction Techniques for IEEE 802.11p and LTE-V2V: Applications and Analysis". In: *2018 IEEE Globecom Workshops (GC Wkshps)*. 2018, pp. 1–7. DOI: 10.1109/GLOCOMW.2018.8644470.

[38] Edward J. Oughton et al. "Revisiting Wireless Internet Connectivity: 5G vs Wi-Fi 6". In: *Telecommunications Policy* 45.5 (2021), p. 102127. ISSN: 0308-5961. DOI: https://doi.org/10.1016/j.telpol.2021.102127. URL: https://www.sciencedirect.com/science/article/pii/S030859612100032X.

[39] Anthony C. Caputo. "5 - Wireless Networked Video". In: *Digital Video Surveillance and Security (Second Edition)*. Ed. by Anthony C. Caputo. Second Edition. Boston: Butterworth-Heinemann, 2014, pp. 145–204. ISBN: 978-0-12-420042-5. DOI: https://doi.org/10.1016/B978-0-12-420042-5.00005-8. URL: https://www.sciencedirect.com/science/article/pii/B9780124200425000058.

[40] Steven Shichang Gao, Qi Luo, and Fuguo Zhu. *Circularly Polarized Antennas*. eng. 1st ed. Wiley - IEEE. New York: Wiley, 2013. ISBN: 9781118374412.

[41] antenna-theory.com. *Antenna Gain*. [Online]. URL: https://www.antenna-theory.com/basics/gain.php (visited on May 19, 2021).

[42] electronics-notes.com. *Antenna Polarization*. [Online]. URL: https://www.electronics-notes.com/articles/antennas-propagation/antenna-theory/polarisation-polarization.php (visited on May 19, 2021).

[43] Carl Bergenhem et al. “Overview of platooning systems”. In: *Proceedings of the 19th ITS World Congress, Oct 22-26, Vienna, Austria (2012)*. 2012.

[44] *Vehicle Platooning: A Brief Survey and Categorization*. Vol. Volume 3: 2011 ASME/IEEE International Conference on Mechatronic and Embedded Systems and Applications, Parts A and B. International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. Aug. 2011, pp. 829–845. DOI: 10.1115/DETC2011-47861. eprint: https://asmedigitalcollection.asme.org/IDETC-CIE/proceedings-pdf/IDETC-CIE2011/54808/829/2768062/829_1.pdf. URL: https://doi.org/10.1115/DETC2011-47861.

[45] Shengbo Eben Li et al. “An overview of vehicular platoon control under the four-component framework”. In: *2015 IEEE Intelligent Vehicles Symposium (IV)*. 2015, pp. 286–291. DOI: 10.1109/IVS.2015.7225700.

[46] S. Ellwanger and E. Wohlfarth. “Truck platooning application”. In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. 2017, pp. 966–971. DOI: 10.1109/IVS.2017.7995840.

[47] Gregor Klančar, Drago Matko, and Sašo Blažič. “A control strategy for platoons of differential drive wheeled mobile robot”. In: *Robotics and Autonomous Systems* 59.2 (2011), pp. 57–64. ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2010.12.002. URL: https://www.sciencedirect.com/science/article/pii/S0921889010001995.

[48] Jeroen Ploeg, Nathan van de Wouw, and Henk Nijmeijer. “Lp String Stability of Cascaded Systems: Application to Vehicle Platooning”. In: *IEEE Transactions on Control Systems Technology* 22.2 (2014), pp. 786–793. DOI: 10.1109/TCST.2013.2258346.

[49] Amr Farag et al. “Dynamics Platooning Model and Protocols for Self-Driving Vehicles”. In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. 2019, pp. 1974–1980. DOI: 10.1109/IVS.2019.8813864.

[50] Florin Catalin Braescu. “Basic control algorithms for vehicle platooning prototype model car”. In: *2017 21st International Conference on System Theory, Control and Computing (ICSTCC)*. 2017, pp. 180–185. DOI: 10.1109/ICSTCC.2017.8107031.

[51] Sebastian Thormann, Alexander Schirrer, and Stefan Jakubek. “Safe and Efficient Cooperative Platooning”. In: *IEEE Transactions on Intelligent Transportation Systems* (2020), pp. 1–13. DOI: 10.1109/TITS.2020.3024950.

[52] Peng Liu, Arda Kurt, and Umit Ozguner. “Distributed Model Predictive Control for Cooperative and Flexible Vehicle Platooning”. In: *IEEE Transactions on Control Systems Technology* 27.3 (2019), pp. 1115–1128. DOI: 10.1109/TCST.2018.2808911.

[53] Shengling Shi and Mircea Lazar. “On distributed model predictive control for vehicle platooning with a recursive feasibility guarantee”. In: *IFAC-PapersOnLine* 50.1 (2017). 20th IFAC World Congress, pp. 7193–7198. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2017.08.607. URL: https://www.sciencedirect.com/science/article/pii/S2405896317309849.

[54] Roozbeh Kianfar, Paolo Falcone, and Jonas Fredriksson. “A control matching model predictive control approach to string stable vehicle platooning”. In: *Control Engineering Practice* 45 (2015), pp. 163–173. ISSN: 0967-0661. DOI: https://doi.org/10.1016/j.conengprac.2015.09.011. URL: https://www.sciencedirect.com/science/article/pii/S0967066115300228.

[55] Şerban Sabău et al. “Optimal Distributed Control for Platooning via Sparse Coprime Factorizations”. In: *IEEE Transactions on Automatic Control* 62.1 (2017), pp. 305–320. DOI: 10.1109/TAC.2016.2572002.

[56] S. Kato et al. “Vehicle control algorithms for cooperative driving with automated vehicles and intervehicle communications”. In: *IEEE Transactions on Intelligent Transportation Systems* 3.3 (2002), pp. 155–161. DOI: 10.1109/TITS.2002.802929.

[57] Chengcheng Zhao, Lin Cai, and Peng Cheng. “Stability Analysis of Vehicle Platooning With Limited Communication Range and Random Packet Losses”. In: *IEEE Internet of Things Journal* 8.1 (2021), pp. 262–277. DOI: 10.1109/JIOT.2020.3004573.

[58] Youssef Abou Harfouch, Shuai Yuan, and Simone Baldi. “An Adaptive Switched Control Approach to Heterogeneous Platooning With Intervehicle Communication Losses”. In: *IEEE Transactions on Control of Network Systems* 5.3 (2018), pp. 1434–1444. DOI: 10.1109/TCNS.2017.2718359.

[59] Victor S. Dolk, Jeroen Ploeg, and W. P. Maurice H. Heemels. “Event-Triggered Control for String-Stable Vehicle Platooning”. In: *IEEE Transactions on Intelligent Transportation Systems* 18.12 (2017), pp. 3486–3500. DOI: 10.1109/TITS.2017.2738446.

[60] Michele Segata, Falko Dressler, and Renato Lo Cigno. “Jerk Beaconing: A dynamic approach to platooning”. In: *2015 IEEE Vehicular Networking Conference (VNC)*. 2015, pp. 135–142. DOI: 10.1109/VNC.2015.7385560.

[61] Vahid Salehi, Shirui Wang, et al. “Using point cloud technology for process simulation in the context of digital factory based on a systems engineering integrated approach”. In: *DS 87-3 Proceedings of the 21st International Conference on Engineering Design (ICED 17) Vol 3: Product, Services and Systems Design, Vancouver, Canada, 21-25.08. 2017*. 2017, pp. 011–020.

[62] ROG. *Manual for ROG STRIX Z590-E GAMING WIFI*. [Online]. URL: https://rog.asus.com/motherboards/rog-strix/rog-strix-z590-e-gaming-wifi-model/helpdesk_manual (visited on Dec. 19, 2021).

[63] RS components. *Rittal Ax, Steel Enclosure, 760 x 760 x 300mm*. [Online]. (Visited on Dec. 16, 2021).

[64] Pfannenberg. *Manual for Pfannenberg Enclosure Heater, 150W*. [Online]. URL: https://docs.rs-online.com/8f64/0900766b8111e126.pdf (visited on Dec. 19, 2021).

[65] *1000W Fixed Installation DC-AC Power Inverter, 12V / 230V | RS Components*. URL: https://se.rs-online.com/web/p/fixed-installation-dc-ac-power-inverters/1793330?fbclid=IwAR2Y47-WkHNh7gpOtDwSpRApao1FnxfNpkfRG9qZkdOjQxZuKQ_4NbKvDJM (visited on Dec. 19, 2021).

[66] Batteriexpressen. *Battery*. [Online]. URL: https://www.batteriexpressen.se/product.html/startbatteri-100ah-tudor-exide-ta1000-high-tech?category_id=243%5C&fbclid=IwAR1RquFvirPjIuaVwdvvtFVTmyEVvOvtCUTRDZUlGwMfbBtPtL4HDrxNVs (visited on Dec. 19, 2021).

[67] Paul Alcorn. *Intel Core i7-10700K Review: Taking the Gaming Shine Off Core i9*. June 2020. URL: https://www.tomshardware.com/reviews/intel-core-i7-10700k-cpu-review/2 (visited on Dec. 19, 2021).

[68] *GeForce GTX 1080 Graphics Cards from NVIDIA GeForce*. URL: https://www.nvidia.com/en-sg/geforce/products/10series/geforce-gtx-1080/ (visited on Dec. 19, 2021).

[69] Steve McDonnell. *How Many Watts for a Power Supply Is Enough?* Oct. 2016. URL: https://smallbusiness.chron.com/many-watts-power-supply-enough-71452.html (visited on Dec. 19, 2021).

[70] *Chieftronic PowerPlay Platinum 850W 1050W: Power On, Play On*. URL: https://www.chieftronic.com/powerplay-platinum (visited on Dec. 19, 2021).

[71] *RS PRO Filter Fan, AC Operation, 230 V ac, IP54*. URL: https://se.rs-online.com/web/p/filter-fans/1759782 (visited on Dec. 19, 2021).

[72] *Frequently Asked Questions*. URL: https://www.nuand.com/frequently-asked-questions/ (visited on Dec. 19, 2021).

[73] Hilbert Hagedoorn. *ASUS RT-AX88U (AX6000) router review*. URL: https://www.guru3d.com/articles-pages/asus-rt-ax88u-dual-band-ax6000-router,9.html.

[74] RS components. *1000W Fixed Installation DC-AC Power Inverter, 12V / 230V*. [Online]. URL: https://se.rs-online.com/web/p/fixed-installation-dc-ac-power-inverters/1793330?fbclid=IwAR2Y47-WkHNh7gpOtDwSpRApao1FnxfNpkfRG9qZkdOjQxZuKQ_4NbKvDJM (visited on Dec. 16, 2021).

[75] Rittal. *Mounting Rail*. [Online]. URL: https://www.automation24.co.uk/rails-for-interior-installation-4-pieces-rittal-ax-2393-300 (visited on Dec. 19, 2021).

[76] ELFA DISTRELEC. *Cable Entry*. [Online]. URL: https://www.elfa.se/sv/taetningsmodul-foer-kabelgenomfoering-kel-er-antal-genomfoeringar-10-112-36mm-polyamid-icotek-kel-er-24-10/p/11083550?origPos=3%5C&q=%5C&pos=3%5C&origPageSize=50%5C&track=true (visited on Dec. 19, 2021).

[77] Mouser Electronics. *Shurter IEC connector*. [Online]. URL: https://www.mouser.se/ProductDetail/Schurter/43120012?qs=5hdTdOtYSo85AQ0NepKyrw%5C%3D%5C%3D%5C&fbclid=IwAR17Pi7doD93MUtmBY-_gbwi65iRsQdchBZ5eIIXM3ZVf2tUPKPgLWS3YmU (visited on Dec. 16, 2021).

[78] RS components. *RS PRO, 2P 3 Position Rotary Cam Switch, 20A*. [Online]. URL: https://se.rs-online.com/web/p/rotary-cam-switches/2083910 (visited on Dec. 16, 2021).

[79] Kjell & Company. *earth leakage circuit breaker*. [Online]. URL: https://www.kjell.com/se/produkter/el-verktyg/el-produkter/starkstrom/din-produkter/personskyddsbrytare-10-a-p67133 (visited on Dec. 16, 2021).

[80] Kjell & Company. *6 way branch outlet*. [Online]. URL: https://se.rs-online.com/web/p/plug-sockets/8294559?cm_mmc=SE-PLA-DS3A-_-google-_-CSS_SE_EN_Connectors_Whoop-_-(SE%5C%3AWhoop!)%5C%20DIN%5C%20Rail%5C%20Terminal%5C%20Blocks-_-8294559%5C&gclid=CjwKCAjwkvWKBhB4EiwA-GHjFj_4Ub4xPMbutkyV_xj_rLclB7u1JRsn63zHQNuVwlg27lxyW74pGBoCAyIQAvD_BwE%5C&gclsrc=aw.ds%5C&matchtype=%5C&pla-477485958713 (visited on Dec. 16, 2021).

[81] Kjell & Company. *6 way branch outlet*. [Online]. URL: https://www.kjell.com/se/produkter/kontor/hemmakontor/grenuttag/grenuttag-med-overspanningsskydd-6-vags-p37814 (visited on Dec. 16, 2021).

[82] RS components. *Pfannenberg FLZ Enclosure Thermostat NC*. [Online]. URL: https://se.rs-online.com/web/p/thermostats/0103106/?relevancy-data=7365617263685F636173636164655F6F726465723D3126736561726368685F696E74657266616365
5C&searchHistory=%5C%7B%5C%22enabled%5C%22%5C%3Atrue%5C%7D (visited on Dec. 16, 2021).

[83] RS components. *Pfannenberg FLZ Enclosure Thermostat NO*. [Online]. URL: https://se.rs-online.com/web/p/thermostats/9048048 (visited on Dec. 16, 2021).

[84] ELFA DISTRELEC. *Skarvdosa, 2.5mm², 37x75x75mm*. [Online]. URL: https://www.elfa.se/sv/skarvdosa-5mm-37x75x75mm-kabelingangar-10-polypropylen-spelsberg-332-90701/p/13647092?q=%5C&pos=2%5C&origPos=2%5C&origPageSize=50%5C&track=true (visited on Dec. 16, 2021).

[85] RS components. *Wago 3-Way Terminal Block, 32A*. [Online]. URL: https://se.rs-online.com/web/p/standard-terminal-blocks/8837548?cm_mmc=SE-PLA-DS3A-_-google-_-CSS_SE_EN_Connectors_Whoop-_-(SE:Whoop!)+Standard+Terminal+Blocks-_-8837548%5C&matchtype=%5C&aud-827186183926:pla-341519951009%5C&gclid=Cj0KCQjw5oiMBhDtARIsAJi0qk2U3KrR7GhOiYZRElytM60YkweYcaUfDkHA_9dsaAv_OEALw_wcB%5C&gclsrc=aw.ds (visited on Dec. 16, 2021).

[86] Clas Ohlson. *Electrical cable RKK 3G1,5*. [Online]. URL: https://www.clasohlson.com/se/p/49-328-25 (visited on Dec. 16, 2021).

[87] RS components. *RS PRO Green, Yellow Earth Modular Terminal Block*. [Online]. URL: https://se.rs-online.com/web/p/din-rail-terminal-blocks/8724805 (visited on Dec. 16, 2021).

[88] Poynting. *PUCK-12*. [Online]. URL: https://poynting.tech/wp-content/uploads/downloads/product_documents/technical_specifications/puck_series/Technical-Specification-A-PUCK-0012-V1-01.pdf (visited on May 23, 2021).

[89] Dustin. *RT-AC68U Dual-Band Wireless AC1900 Gigabit Router*. [Online]. URL: https://www.dustin.se/product/5010755873/rt-ac68u-dual-band-wireless-ac1900-gigabit-router (visited on Dec. 16, 2021).

[90] Elfa. *MA510.C.CG.005 - Wi-Fi-antenn 2.4 ... 2.5 GHz/4.9 ... 5.9 GHz 3.9 dBi SMA/RP, hane, Taoglas*. [Online]. URL: https://www.elfa.se/sv/wi-fi-antenn-ghz-ghz-dbi-sma-rp-hane-taoglas-ma510-cg-005/p/30220267?q=%5C&pos=1%5C&origPos=15%5C&origPageSize=50%5C&track=true (visited on Dec. 16, 2021).

[91] Dustin. *A6100 WiFi USB Mini Adapter*. [Online]. URL: https://www.dustin.se/product/5010780210/a6100-wifi-usb-mini-adapter?ssel=false%5C&gclid=CjwKCAiA1aiMBhAUEiwACw25Mc4gbhgVnFqAJQx-sxekKrnXW5n3h4w9ifxCPfuwJkun4tZiPJabcBoCSUAQAvD_BwE%5C&_ga=2.114887879.1785133111.1636479024-1074048296.1636479024%5C&_gac=1.86273642.1636479166.CjwKCAiA1aiMBhAUEiwACw25Mc4gbhgVnFqAJQx-sxekKrnXW5n3h4w9ifxCPfuwJkun4tZiPJabcBoCSUAQAvD_BwE (visited on Dec. 16, 2021).

[92] Wallace Pereira Neves dos Reis et al. “An extended analysis on tuning the parameters of Adaptive Monte Carlo Localization ROS package in an automated guided vehicle”. In: *The International Journal of Advanced Manufacturing Technology* 117.5 (2021), pp. 1975–1995.

[93] Kaiyu Zheng. “Ros navigation tuning guide”. In: *arXiv preprint arXiv:1706.09068* (2017).

[94] MathWorks. *System Identification for PID Control*. [Online]. URL: https://se.mathworks.com/help/control/getstart/system-identification-of-plant-models.html (visited on Dec. 16, 2021).

[95] Joseph Redmon et al. “You Only Look Once: Unified, Real-Time Object Detection”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91.

[96] *CUDA Wikipedia*. URL: https://en.wikipedia.org/wiki/CUDA (visited on Dec. 19, 2021).

[97] *CUDA Toolkit Release Notes*. URL: https://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html (visited on Dec. 19, 2021).

[98] *cuDNN*. URL: https://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html (visited on Dec. 19, 2021).

[99] *cuDNN Archive*. URL: https://developer.nvidia.com/rdp/cudnn-archive (visited on Dec. 19, 2021).

[100] *OpenCV Wikipedia*. URL: https://en.wikipedia.org/wiki/OpenCV (visited on Dec. 19, 2021).

[101] leggedroboticss. *YOLO ROS: Real-Time Object Detection for ROS*. [Online]. URL: https://github.com/leggedrobotics/darknet_ros (visited on Dec. 16, 2021).

[102] FelixvonDrigalsk. *Markers: Sending Basic Shapes*. [Online]. URL: http://wiki.ros.org/rviz/Tutorials/Markers%5C%3A%5C%20Basic%5C%20Shapes (visited on Dec. 16, 2021).

[103] *BladeRF micro 2.0 xA4 Datasheet*. URL: https://www.elfa.se/Web/Downloads/_m/an/WRL-15043_eng_man.pdf (visited on Dec. 19, 2021).

[104] GNU Radio project. *About GNU Radio*. [Online]. URL: https://www.gnuradio.org/about/ (visited on Dec. 2, 2021).

[105] B.Bloessl. *gr-ieee802-11 Open-source transceiver software*. [GitHub repository]. URL: https://github.com/YamanDaif/gr-ieee802-11/projects?type=beta (visited on Dec. 18, 2021).

[106] B.Bloessl. *https://www.bastibl.net/*. [Home page]. (Visited on Dec. 18, 2021).

[107] *MikroTik R11e-5HnD nu 15% billigare R11E-5HND Wewnętrzny RF Wireless*. URL: https://www.senetic.se/product/R11E-5HND (visited on Dec. 19, 2021).

[108] URL: https://www.elfa.se/sv/bladerf-micro-xa4-programdefinierad-radiomodul-sparkfun-electronics-wrl-15043/p/30152854?q=bladerf&pos=1&origPos=1&origPageSize=50&track=true (visited on Dec. 19, 2021).

[109] *Raspberry Pi Pico Datasheet*. English. 2020. URL: https://datasheets.raspberrypi.org/pico/pico-datasheet.pdf (visited on Dec. 19, 2021).

[110] *Deep Cycle Battery FAQ*. en. URL: https://www.solar-electric.com/learning-center/deep-cycle-battery-faq.html/ (visited on Dec. 19, 2021).

[111] Daniel Richter et al. "Performance of Real-Time Wireless Communication for Railway Environments with IEEE 802.11p". In: *Proceedings of the 52nd Hawaii International Conference on System Sciences (HICSS)*. Jan. 8, 2019. ISBN: 978-0-9981331-2-6. DOI: 10.24251/HICSS.2019.907. URL: http://scholarspace.manoa.hawaii.edu/handle/10125/60190.

[112] Lukas Pirl. *IEEE 802.11p on Linux*. URL: https://gitlab.com/hpi-potsdam/osm/g5-on-linux/11p-on-linux (visited on Dec. 19, 2021).

[113] *MikroTik R11-5HnD specifications*. en. URL: https://mikrotik.com/ (visited on Dec. 19, 2021).

[114] *HackRF One - transmit power*. URL: https://hackrf.readthedocs.io/en/latest/hackrf_one.html#transmit-power (visited on Dec. 19, 2021).

[115] *ETSI TS 102 792 V1.2.1*. English. chapter 5. June 2015. URL: https://www.etsi.org/deliver/etsi_ts/102700_102799/102792/01.02.01_60/ts_102792v010201p.pdf (visited on Dec. 19, 2021).

[116] *ITS-stack*. en. URL: https://www.qualcomm.com/media/documents/files/c-v2x-its-stack.pdf (visited on Dec. 19, 2021).

[117] Bernhard Kloiber et al. "Performance of CAM based safety applications using ITS-G5A MAC in high dense scenarios". In: *2011 IEEE Intelligent Vehicles Symposium (IV)*. 2011, pp. 654–660. DOI: 10.1109/IVS.2011.5940461.

[118] Wikipedia. *Volumetric heat capacity*. [Online]. URL: https://en.wikipedia.org/wiki/Volumetric_heat_capacity (visited on May 22, 2021).

[119] The Engineering Toolbox. *Convective Heat Transfer*. [Online]. URL: https://www.engineeringtoolbox.com/convective-heat-transfer-d_430.html (visited on May 22, 2021).

[120] Wikipedia. *Table of specific heat capacities*. [Online]. URL: https://en.wikipedia.org/wiki/Table_of_specific_heat_capacities (visited on May 22, 2021).

Appendix A

Cooling calculations

At steady state temperature the sum of heat transfer from the enclosure will be equal to the power consumed by the hardware. To simplify the calculations the heat transfer to the ground and the difference in heat radiated to and from the enclosure are assumed to be negligible in the long term. This gives the equation below.

$$P_{hardware} = P_{ventilation} + P_{convection} \tag{A.1}$$

The heat transported out by ventilation is equivalent to the difference in energy of the air flowing in and out. Assuming the temperature of the outflow is the average temperature of the inside and that the pressure at the outflow is the same as the inflow, the heat loss is calculated by the isobaric volumetric heat capacity for air, c_{PV} [J/m^{3}K], multiplied by the volume flow (which will be the same at the outflow and inflow) and the difference in temperature of the outflow and inflow [118].

$$P_{ventilation} = c_{PV}\dot{V}(T_{in} - T_{out}) \tag{A.2}$$

The heat flow from convection is calculated with the below equation[119].

$$P_{convection} = h_c A(T_{enclosure} - T_{out}) \tag{A.3}$$

Here h_c is the convective heat transfer coefficient for air, A is the surface area for convection, and $T_{enclosure}$ is the temperature of the enclosure. The convective heat transfer coefficient for air has been experimentally approximated to the below equation[119].

$$h_c = 10.45 - v + 10\sqrt{v} \tag{A.4}$$

Here v is the velocity of the air. As this is the worst case scenario the velocity of the air will be very low, it can be assumed to be 1 m/s. Thus h_c is (roughly) the same for the inside and outside air, and so at steady state temperature, when the convective heat flow from the inside air to the enclosure and from the enclosure to the outside air is the same, the temperature of the enclosure will have the average of the inside and outside air temperature.

$$P_{convection} = h_c A(\frac{T_{out} + T_{in}}{2} - T_{out}) = h_c A \frac{T_{in} - T_{out}}{2} \tag{A.5}$$

Combining equations A.1, A.2 and A.5 gives the below equation.

$$P_{hardware} = c_{PV}\dot{V}(T_{in} - T_{out}) + h_c A \frac{T_{in} - T_{out}}{2} \tag{A.6}$$

This equation is rearranged to give the inside temperature T_{in}.

$$T_{in} = T_{out} + \frac{2P_{hardware}}{2c_{PV}\dot{V} + h_c A} \tag{A.7}$$

If T_{out} is set to the estimated worst case scenario temperature of 35 °C and $P_{hardware}$ to the maximum power consumption possible for the RSU as calculated in section 4.2.7, 786.6 W, T_{in} becomes the enclosure temperature at worst case scenario. A is set to 2.1 m^3 based on our current 0.76 m x 0.76 m x 0.3 m enclosure, the $\dot{V}$ of 0.085 m^3/s is given by the impeded airflow of the fan, h_c is calculated to 19.45 $W/(m^2K)$ by equation A.4 (assuming inside and outside wind velocity is 1 m/s) and c_{PV} is given for normal temperature air as 1210 J/m^3K[120]. The result is that the maximum temperature in the enclosure at worst case scenario is 41.4 °C.

A.1 Sources for errors

As the outflow of air is high up on the enclosure the assumption that the out flowing air is the average inside air temperature is not correct. The outflow temperature will be slightly higher, and so the cooling from the ventilation will be slightly better than calculated.

If the sun is shining on the enclosure the heat radiated to the enclosure will be greater than the heat radiated by the enclosure, despite the enclosure having a higher temperature than the

surroundings. Thus the heat from radiation that was earlier neglected could have a negative effect on the cooling. Although if the enclosure is in the shade it will likely have a higher temperature than surrounding structures, and so the net energy from radiation will likely be negative, and the cooling better than calculated.

Although there is circulation in the enclosure it is possible that the assumed 1 m/s of air velocity is too high. If that is the case c_{PV} will be lower than calculated and the cooling from convection is slightly worse than calculated.

Appendix B

Heating calculations

As the ventilation will be shut off and the heater turned on at low temperatures, equation A.1 can be rewritten for the case of heating to the below equation.

$$P_{hardware} + P_{heater} = P_{convection} \tag{B.1}$$

Unlike for the case of the cooling calculations, the worst case for the heating calculations includes strong winds, thus the convective heat transfer coefficients for heat flow from the inside to the enclosure and enclosure to the outside are different, and so A.5 cannot be applied. The convective heat flow from the inside to the enclosure and enclosure to the outside are calculated with the below equations[119].

$$P_{convection,in} = h_{c,in}A(T_{in} - T_{enclosure}) \tag{B.2}$$

$$P_{convection,out} = h_{c,out}A(T_{enclosure} - T_{out}) \tag{B.3}$$

Here $h_{c,in}$ and $h_{c,out}$ are the convective heat transfer coefficients for the inside air and outside air respectively. The convective heat transfer coefficient for air has been experimentally approximated to the below equation[119].

$$h_c = 10.45 - v + 10\sqrt{v} \tag{B.4}$$

Here v is the velocity of the air. The circulation inside the enclosure is very low and the velocity is approximated to 0 as all fans will be turned of at the lowest temperatures. As this is the worst

case scenario for heating the outside air will not only be cold, but also windy. The outside air velocity is approximated to 10 m/s. At steady state temperature $P_{convection,in}$ and $P_{convection,out}$ will be the same, giving the below equation.

$$h_{c,in}A(T_{in} - T_{enclosure}) = h_{c,out}A(T_{enclosure} - T_{out}) \tag{B.5}$$

This is rearranged to give the temperature of the enclosure as shown below.

$$T_{enclosure} = \frac{T_{out}h_{c,out} + T_{in}h_{c,in}}{h_{c,out} + h_{c,in}} \tag{B.6}$$

Combining equations B2 (or B3) and B6 gives the convective heat flow as shown below.

$$P_{convection} = h_{c,in}A(T_{in} - \frac{T_{out}h_{c,out} + T_{in}h_{c,in}}{h_{c,out} + h_{c,in}}) = h_{c,in}h_{c,out}A\frac{T_{in} - T_{out}}{h_{c,out} + h_{c,in}} \tag{B.7}$$

Equation B.7 can be combined with B.1 and rearranged to give the inside temperature.

$$T_{in} = T_{out} + (P_{hardware} + P_{heater})\frac{h_{c,out} + h_{c,in}}{h_{c,in}h_{c,out}A} \tag{B.8}$$

If T_{out} is set to the worst case scenario temperature of -15 °C, $P_{hardware}$ to a low estimate of the idle power consumption, 100 W, and P_{heater} to 150 W based on the current heater, T_{in} becomes the inside temperature at worst case scenario. A is set to 2.1 m^3 based on our current 0.76 m x 0.76 m x 0.3 m enclosure and $h_{c,out}$ and $h_{c,in}$ are approximated to 32.1 and 10.45 respectively based on equation 10 (assuming inside wind velocity is 0 as all fans are off, and inside wind velocity is 10 m/s). The result is that the minimum temperature in the enclosure at worst case scenario is 0.3 °C.

B.1 Sources for errors

As the surroundings are colder than the enclosure the heat radiated from the enclosure will be larger than the heat radiated to the enclosure. Although the radiated heat from a treated steel surface at low temperatures is low, this will cause an increase in required heating in certain circumstances.

If the sun is shining on the enclosure the heat radiated to the enclosure will likely be greater that the heat radiated by the enclosure, despite the enclosure having a higher temperature than the surroundings. Thus the heat from radiation that was earlier neglected could have a negative effect on the cooling.

Although there is circulation in the enclosure it is possible that the assumed 1 m/s of air velocity is too high. If that is the case c_{PV} will be lower than calculated and the cooling from convection is slightly worse than calculated.

Appendix C

Test case demonstrations

The stakeholder of the project has given the project team specific demonstrations that the final implementation of the OBU and RSU need to be able to perform. These demonstration will be used to measure the success of the project. A more detailed description of the demonstrations can be seen below.

C.1 Demonstration 1

This demonstration involves a platooning scenario between a simulated vehicle and two physical entities. The simulated world within the AD-EYE testbed, contains a simulated vehicle and some simulated sensors, communicates sensor data and V2X messages from the ITS G5 to the first TurtleBot (TurtleBot1). The goal of the first TurtleBot is to follow the simulated vehicle in the real world. The goal of the second TurtleBot (TurtleBot2) is to follow TurtleBot1 in the real world with its onboard sensors, to complete the platoon. The platoon has thus the following entities in order: AD-EYE simulated vehicle -> TurtleBot1 -> TurtleBot2.

C.1.1 Scenario 1:

The platoon runs across a large loop in the simulated world (in an empty area in the real world) Sending a safety critical message over V2X should cause every vehicle (real and simulated) in the platoon to stop independently (and without crashing).

- Find out experimentally the smallest distance possible between the turtle bots, given the limitations of latency, lack of reliability of communication channels etc.

C.1.2 Scenario 2:

The platoon runs across a large loop in the simulated world with known types of objects placed around the track at known positions. The RSU is the only entity performing object detection and classification across all the sensor data streams available in the experiment. It then communicates back a list of objects and locations detected across to the AD-EYE platform using a protocol decided by the testbed owner where the simulation world is updated.

- Experimentally demonstrate how accurate the final positioning of the real objects can be within the simulation world

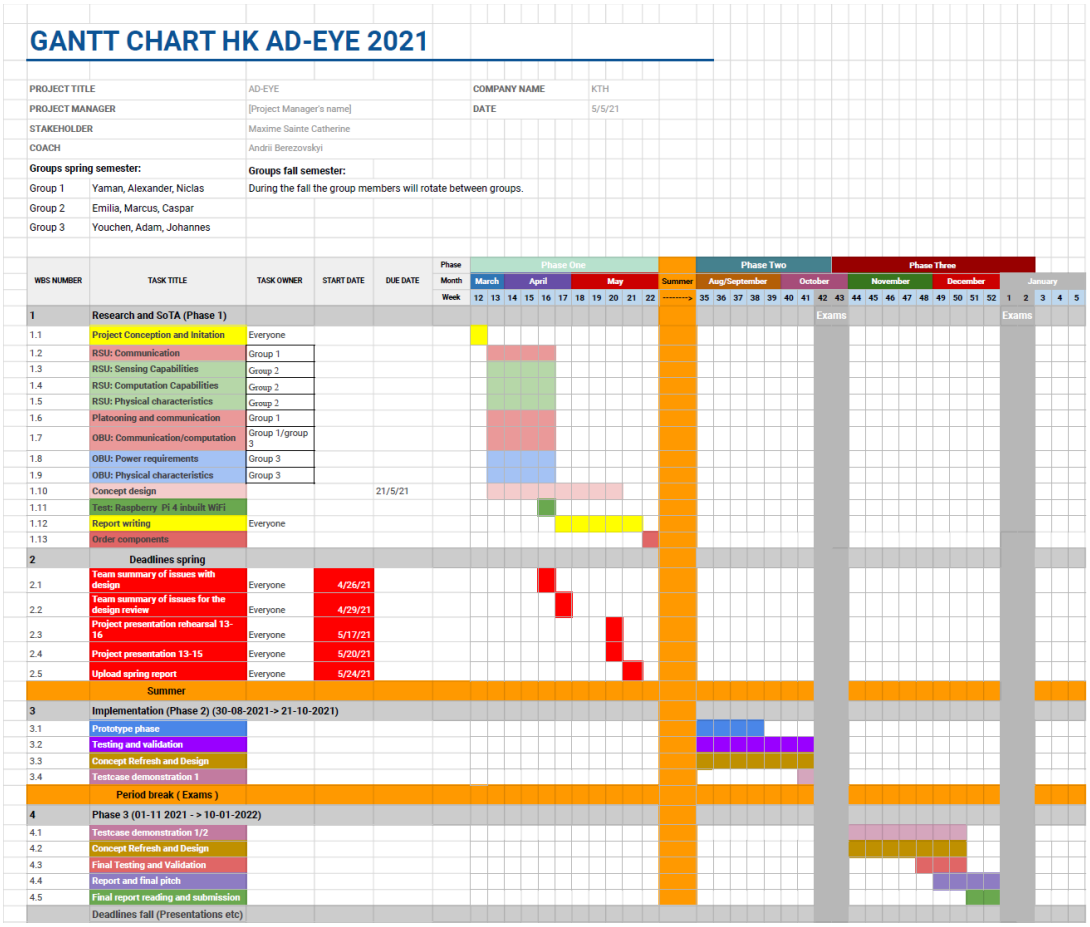
C.2 Demonstration 2

This demonstration is not decided yet. Can be as complex or simple as needed based on the number of students allocated, and if design choices made for Demonstration 1 leaves unfulfilled learning goals for some of the students.

Appendix D

GANTT chart

This appendix include a time plan for the project in form of a GANTT chart for the already elapsed spring semester and the coming fall semester. See Figure D.1



GANTT CHART HK AD-EYE 2021

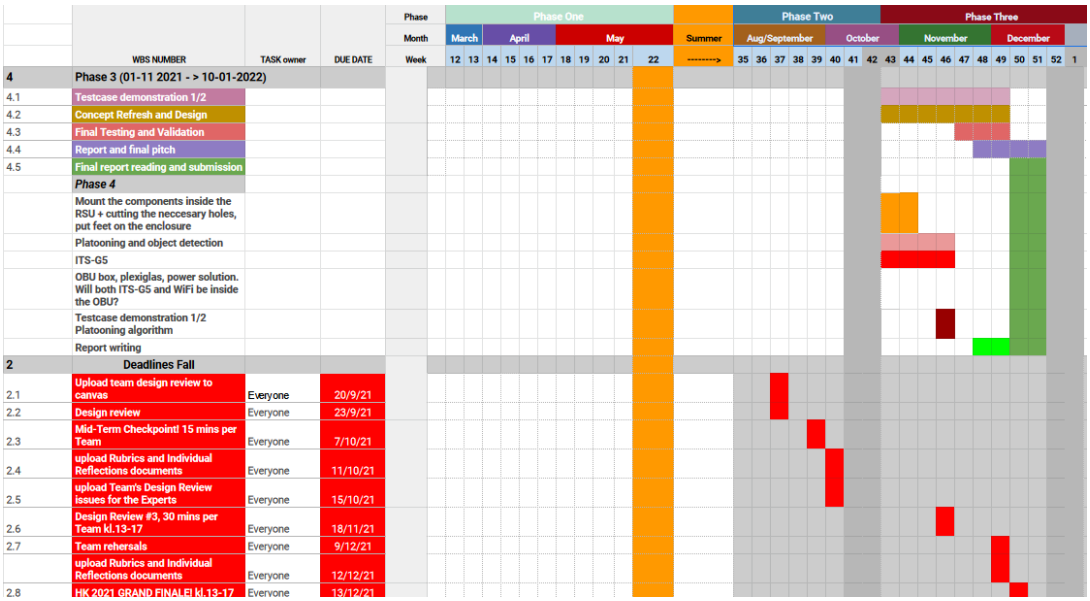
PROJECT TITLE	AD-EYE	COMPANY NAME	KTH
PROJECT MANAGER	[Project Manager's name]	DATE	5/5/21
STAKEHOLDER	Maxime Sainte Catherine		
COACH	Andrii Berezovskyi		

Groups spring semester:		Groups fall semester:
Group 1	Yaman, Alexander, Niclas	During the fall the group members will rotate between groups.
Group 2	Emilia, Marcus, Caspar	
Group 3	Youchen, Adam, Johannes	

WBS NUMBER	TASK TITLE	TASK OWNER	START DATE	DUE DATE
	Phase	Phase One		
	Month	March / April / May	Summer	Aug/September / October / November / December / January
	Week	12 13 14 15 16 17 18 19 20 21 22	------>	35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 1 2 3 4 5
1	Research and SoTA (Phase 1)			
1.1	Project Conception and Initation	Everyone		
1.2	RSU: Communication	Group 1		
1.3	RSU: Sensing Capabilities	Group 2		
1.4	RSU: Computation Capabilities	Group 2		
1.5	RSU: Physical characteristics	Group 2		
1.6	Platooning and communication	Group 1		
1.7	OBU: Communication/computation	Group 1/group 3		
1.8	OBU: Power requirements	Group 3		
1.9	OBU: Physical characteristics	Group 3		
1.10	Concept design			21/5/21
1.11	Test: Raspberry Pi 4 inbuilt WiFi			
1.12	Report writing	Everyone		
1.13	Order components			
2	Deadlines spring			
2.1	Team summary of issues with design	Everyone	4/26/21	
2.2	Team summary of issues for the design review	Everyone	4/29/21	
2.3	Project presentation rehearsal 13-16	Everyone	5/17/21	
2.4	Project presentation 13-15	Everyone	5/20/21	
2.5	Upload spring report	Everyone	5/24/21	
	Summer			
3	Implementation (Phase 2) (30-08-2021-> 21-10-2021)			
3.1	Prototype phase			
3.2	Testing and validation			
3.3	Concept Refresh and Design			
3.4	Testcase demonstration 1			
	Period break (Exams)			
4	Phase 3 (01-11 2021 - > 10-01-2022)			
4.1	Testcase demonstration 1/2			
4.2	Concept Refresh and Design			
4.3	Final Testing and Validation			
4.4	Report and final pitch			
4.5	Final report reading and submission			
	Deadlines fall (Presentations etc)			

Figure D.1: GANTT chart for spring and fall semester.

The updated GANTT chart for the fall can be seen in Figure D.2



	WBS NUMBER	TASK owner	DUE DATE
4	Phase 3 (01-11 2021 - > 10-01-2022)		
4.1	Testcase demonstration 1/2		
4.2	Concept Refresh and Design		
4.3	Final Testing and Validation		
4.4	Report and final pitch		
4.5	Final report reading and submission		
	Phase 4		
	Mount the components inside the RSU + cutting the neccesary holes, put feet on the enclosure		
	Platooning and object detection		
	ITS-G5		
	OBU box, plexiglas, power solution. Will both ITS-G5 and WiFi be inside the OBU?		
	Testcase demonstration 1/2 Platooning algorithm		
	Report writing		
2	Deadlines Fall		
2.1	Upload team design review to canvas	Everyone	20/9/21
2.2	Design review	Everyone	23/9/21
2.3	Mid-Term Checkpoint! 15 mins per Team	Everyone	7/10/21
2.4	upload Rubrics and Individual Reflections documents	Everyone	11/10/21
2.5	upload Team's Design Review issues for the Experts	Everyone	15/10/21
2.6	Design Review #3, 30 mins per Team kl.13-17	Everyone	18/11/21
2.7	Team rehersals	Everyone	9/12/21
	upload Rubrics and Individual Reflections documents	Everyone	12/12/21
2.8	HK 2021 GRAND FINALE! kl.13-17	Everyone	13/12/21

Figure D.2: Updated GANTT chart for thefall semester.

Appendix E

Setup and Installations

E.1 ITS-G5 Software installation

For the ITS-G5 solution we are using SDR and GNU Radio for programming the SRDs and an open source transceiver polygraphs on GNU Radio software to send and receive on the ITS-G5 frequency band

E.1.1 Installing GNU radio 3.7.9

Because we are using Ubuntu 16.04 GNU Radio 3.7.9 is the latest version that works with our system, for the installation we used the official wiki GNU Radio link:
`https://wiki.gnuradio.org/index.php/InstallingGR`

E.1.2 Installing SDR HackRF One

HackRF is installed and configured using apt-get install command:

```
$sudo apt-get update
$sudo apt-get install hackrf
```

E.1.3 Installing SDR bladeRF 2.0 micro xA4

Nuand/bladeRF (github):
https://github.com/Nuand/bladeRF/wiki#Getting_Started
Easy installation for Ubuntu: The bladeRF PPA:
https://github.com/Nuand/bladeRF/wiki/Getting-Started%3A-Linux#Easy_installation_for_Ubuntu_The_bladeRF_PPA

```
$ sudo add-apt-repository ppa:nuandllc/bladerf
$ sudo apt-get update
$ sudo apt-get install bladerf
```

If you plan to build GNU Radio, gr-osmosdr, etc, you will also need the header files:

```
$ sudo apt-get install libbladerf-dev
```

Firmware and FPGA images can be installed from this PPA as well. Firmware should be manually updated using:

```
$bladeRF-cli --flash-firmware /usr/share/Nuand/bladeRF/bladeRF_fw.img
```

But the FPGA image will be automatically loaded by libbladeRF when you open your device.

```
$sudo apt-get install bladerf-firmware-fx3    #for all bladeRF models
$sudo apt-get install bladerf-fpga-hostedxa4 #BladeRF 2.0 micro xA4
```

It's important to have matching versions between libbladeRF, FPGA and FX3. The matching versions can be found under "realeses" on the official github repo.

https://github.com/Nuand/bladeRF/releases

N.B.
If you've installed libbladerf through apt-get as shown above, it's likely you can't use the latest FPGA and FX3 release. How to check your library version and other useful commands can be found in the manpage for bladeRF-cli, linked below. Take care to download the matching versions from the releases section linked above.

http://manpages.ubuntu.com/manpages/bionic/man1/bladeRF-cli.1.html

E.1.4 Installing gr-osmosdr

We have to install Gr-osmosdr to get the Sink and Source blocks for the software defined radio (SDR) in GNU Radio. The official link to the installation is:

https://osmocom.org/projects/gr-osmosdr/wiki
Use the following commands:

```
$git clone git://git.osmocom.org/gr-osmosdr
$cd gr-osmosdr/
$mkdir build
$cd build/
$cmake ../
```

If the previous method didn't woke try apt-get command for the installation:

```
$sudo apt install gnuradio gr-osmosdr
```

Troubleshooting:

During our installations we needed to change the compiler properties to build gr-osmosdr by adding to the Cmakefile the below code line after line number 147:

```
set(CMAKE_CXX_FLAGS '${CMAKE_CXX_FLAG} -std=c++11')
```

E.1.5 Upgrading the Cmake

CMake is cross-platform free and open-source software for build automation, testing, packaging and installation of software by using a compiler-independent method. For installing the gr-ieee802-11(in the next step), CMake version above 3.7.8 is needed. Ubuntu 16.04 comes with CMake 3.5.1 to upgrade the CMake there are plenty of methods but one worked for our setup:
https://programmer.group/cmake-install-higher-version.html
Follow the instruction in the link and make sure you don't remove the previous CMake if you have already installed ROS. You might face sourcing problems if you install CMake without removing the previous one and you will need to fix this issue and create a new links when you want to run the newest version. However, the ultimate solution is to upgrade the CMake before installing ROS.

E.1.6 Installing gr-ieee802-11

This an IEEE 802.11 a/g/p transceiver by B.Bloessl on GNU Radio that is modified to work with our SDR namely the bladeRF and the HackRF. This will make the SDR able to send and/or receive on the ITS-G5 frequency band.
This forked repository is accustomed to the HackRF and bladeRF used in this project and includes all the

modifications needed for the integration with ROS. `https://github.com/YamanDaif/gr-ieee802-11/tree/maint-3.7`
Follow the information for the installation in the link and make sure you clone the maint-3.7 branch, which is the dedicated branch for the GNU Radio (3.7).

Troubleshooting

For the gr-foo and gr-ieee802-11 Swig package is needed if it's not already installed. For installation use the following commands:

```
$sudo apt-get update
$sudo apt-get install swig
```

E.2 GPU driver installation and system adaption

E.2.1 Install Nvidia graphics driver

Normally, there are three ways to install Nvidia graphics driver for Nvidia GPU.

1. Use the existing drivers in Ubuntu system
   If system installation package is complete, it has existing drivers for Nvidia GPU in the system. Find it in "System Setup/ System/ Software & Update/ Affiliate drivers" But by this way you couldn't choose the suitable driver that you want.

2. Download the driver from Nvidia website
   After installation, use the following order:

   ```
   $sudo apt-get remove --purge nvidia* / Remove old drivers
   $sudo gedit /etc/modprobe.d/blacklist.conf
   ```

   Then Add the following in the end of the file:

   ```
   blacklist vga16fb
   blacklist nouveau
   blacklist rivafb
   blacklist rivatv
   blacklist nvidiafb
   ```

   Close graphic interface

   ```
   $sudo update-initramfs -u
   $sudo service lightdm stop
   ```

Enter the order line mode with pressing Ctrl+Shift+F1 on keyboard on make the file executable. Then install it.

```
$sudo chmod +x NVIDIA-Linux-x86_64-430.64.run
$sudo ./NVIDIA-Linux-x86_64-430.64.run -no-opengl-files
```

After installation, try this order to test if the driver works. It mean success if it shows information of hardware and software.

```
$nvidia-smi
```

3. Enable the system find the driver by itself
   In this method, it needs to be set "System Setup/ System/ Software & Update/ Download from" with choosing all choices except "Source code". And the following steps is as same as the previous one.

E.2.2 Install CUDA driver

The method to install CUDA is much easier.

1. Download a suitable CUDA driver on Nvidia website.
   Run the driver .run file:

   ```
   $sudo sh cuda_<version>_linux.run
   ```

   Do some choices:

   ```
   accept/decline/quit: accept
   Install NVIDIA Accelerated Graphics Driver for Linux-x86_64 430.64?
   (y)es/(n)o/(q)uit: n
   Install the CUDA 10.1 Toolkit?
   (y)es/(n)o/(q)uit: y
   Enter Toolkit Location
   [ default is /usr/local/cuda-10.1 ]:
   Do you want to install a symbolic link at /usr/local/cuda?
   (y)es/(n)o/(q)uit: y
   Install the CUDA 10.1 Samples?
   (y)es/(n)o/(q)uit: n
   Installing the CUDA Toolkit in /usr/local/cuda-10.1 ...
   ```

   Setup the environment path:

   ```
   $sudo gedit ~/.bashrc
   ```

   Add the following in the end:

```
export PATH=/usr/local/cuda-10.1/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda/lib64:$LD_LIBRARY_PATH
```

Update the source path:

```
$source ~/.bashrc
```

Finally check the version of CUDA is it has been installed successfully:

```
$nvcc -V
```

E.3 Ethernet connection bug

This section describes how to fix the specific Ethernet connection bug described in section 4.2.2. Before doing these steps, make sure that the Ethernet port starts to work when they are restarted manually. This will not make the 2.5 Gbps ports work with any of the standard Ubuntu 16.14 kernels, so upgrade to kernel 5.1 first. The steps to solve the bug are as following:

1. Manually resart the network connection.
2. Type ifconfig in the command line. The devices listed that start with e is the names of the Ethernet ports, note them.
3. Create a file in the folder /etc called rc.local
4. Edit the file to contain the following lines:

```
#!/bin//sh -e

sudo ifconfig {Ethernet port 1 name} down
sudo ifconfig {Ethernet port 2 name} down

sleep 3

sudo ifconfig {Ethernet port 1 name} up
sudo ifconfig {Ethernet port 2 name} up

exit 0
```

5. Allow the file to run as a program by typing the following line into the command line:

```
sudo chmod +x /etc/rc.local
```

E.4 Wi-Fi card as an ITS-G5 radio

This section will be short as the instructions provided on the repository are good. The prerequisites are also mentioned, the setup uses the software Ansible. You should familiarize yourself with this software first, using this source for example:

`https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-ansible-on-ubu`

The automatic setup is recommended from the repository [112]:
`https://gitlab.com/hpi-potsdam/osm/g5-on-linux/11p-on-linux`

It could be of interest to see the issues submitted, or creating your own as the contributors are very helpful.

Another repository could be useful to gain some deeper knowledge, as a thesis is available to read on the performance of a similar setup.

`https://github.com/jfpastrana/802.11p/blob/master/Documentation/802.11p_standard_and_V2X_applications.pdf`

Appendix F

Electrical Schematic

The electrical schematic of the RSU can be seen in figure F.1

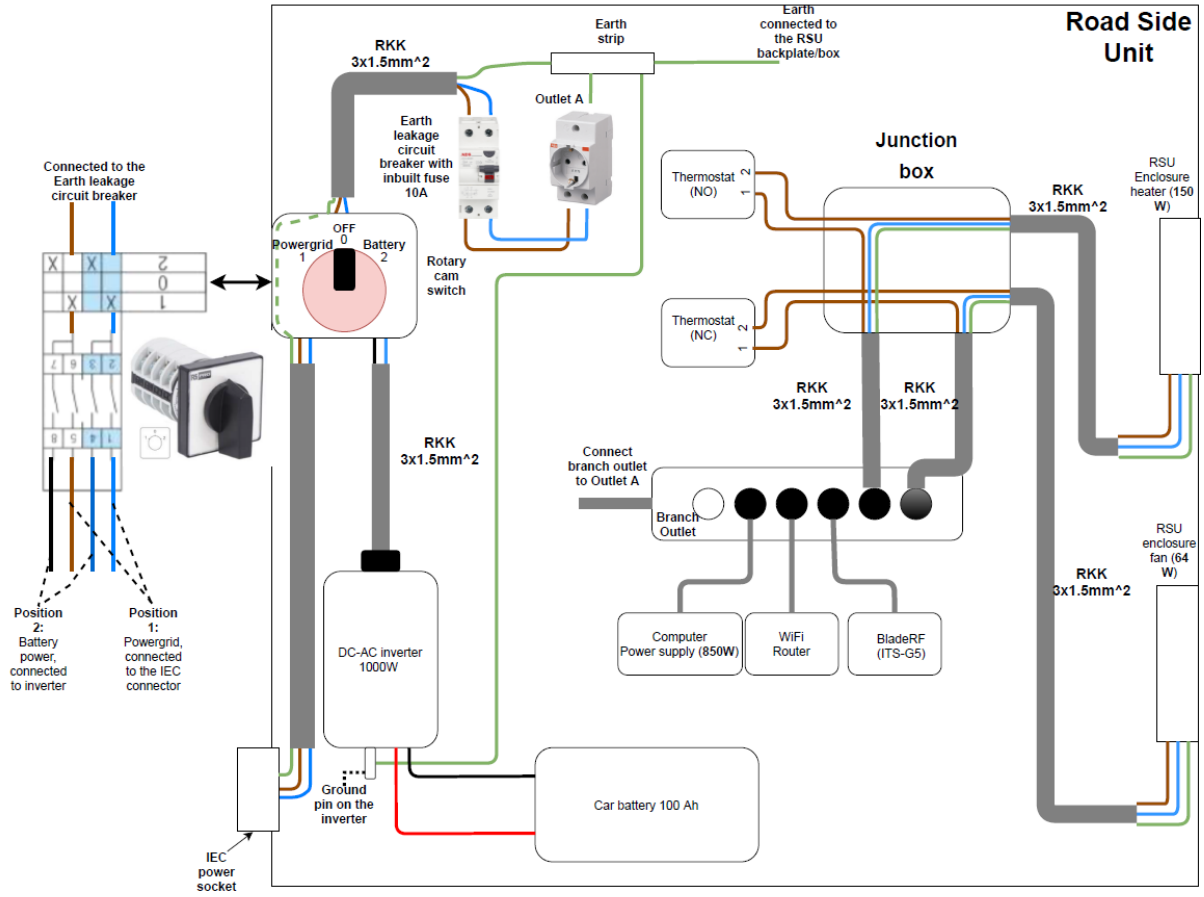


Figure F.1: The electrical schematic of the RSU

Appendix G

Using Cohda MK2

The physical connection to the Cohda box must be an Ethernet cable to a router. We tried directly connecting to them with both patch and crossover cables and it didn't work*.

When your PC and the Cohda unit are connected to the same router, you can ssh into the Cohda unit. From inside the SDK in vmware, running on your PC, you ssh to the local network ip of the Cohda like normal "ssh user@192.168.xyz.pqr"

ssh from windows successful with this line in powershell:
ssh -oKexAlgorithms=+diffie-hellman-group1-sha1 user@legacyhost

Where legacy host is the ip of the Cohda device, provided by the router. This ip can be found in the client table of your router, which is easily viewed by accessing your router from your browser [1].

user = user
password = user

If you become root@MK2 then you're in, congrats. Next thing could be to test the communication. Go into the directory with the test scripts: cd /opt/cohda/test

Helpful abbreviations:
tx = transmit
rx = receive
cch = control channel.
sch = service channel.

Depending on the channel number you want to use, you should use different scripts.

Run a script:
`./runtest_cch_tx.sh 178 target TXlog.txt`

178 is the channel number in this case. TXlog.txt is the output log file which will contain the results of the test.

Now you are sending from the cohda box. To recieve the packets, make sure you use the same channel in your receiver.

[1] help with this is easily found on the internet, for example:
https://uk.pcmag.com/wireless-routers/83093/how-to-access-your-wi-fi-routers-settings

*The very first time we tried with a regular patch cable, it did work. The connection could not be done again however, for no apparent reason. We tried directly connecting (PC to Cohda box) with both patch and crossover cables and it didn't work with either. From this point, we only managed to connect using a router in between.

Appendix H

Two different concepts for the camera and LiDAR transmission can be seen in figure **??**.

Solution 1 (WiFi 11ac):		**Cost included in the RSU:**
Dualband WiFi-6 Router	ASUS RT-AX55	679
RSU WiFi Antenna	Poynting Puck-12 Rundstrål 2X2 Mimo	990
		Total cost included in RSU:
		1669
		Cost included in one OBU:
WiFi 11ac dongle	Alfa Long-Range Dual-Band AC1200	480
Raspberry pi 4 4GB	Raspberry Pi 4 Model B	543
Power solution	[Added later on, will be included in	
External Antenna		
		Total cost included in one
		1023
Solution 2 (WiFi 11ax):		**Cost included RSU:**
RSU WiFi Antenna	Poynting Puck-12 Rundstrål 2X2 Mimo	990
Dualband WiFi 6 Router	ASUS RT-AX55	679
		Total Cost included in RSU:
		1669
		Cost included in one OBU:
Raspberry Pi CM4	Raspberry Pi Compute Module 4 (CM4)	562
Raspberry Pi CM4 IO board	Raspberry Pi Compute Module 4 (CM4)	318
WiFi 6 (11ax) PCIe card	Ziyituod Intel AX200 Wifi 6 PCIe card	416
Power solution	[Added later on, will be included in	
External Antenna		
		Total cost included in one
		1296

Figure H.1: Showing two different setups for the LiDAR and camera transmission