KTH Mechatronics Advanced Course

MF2059, HT 2022

FINAL REPORT

Towards Autonomous Driving on KTH Campus

AD-EYE

Malin Dyberg, Axel Hedvall, Johan Hultenheim
Óscar Poveda Ruiz, Shaya Rahmanian, Gowtham Rangaraju
Elvira Troillet, Ehab Ziad Raheem

December 18, 2022

Abstract

AD-EYE is an ongoing collection of projects at Kungliga Tekniska Högskolan (KTH) in Sweden, aiming to develop a platform for researchers to test autonomous driving solutions in real-world conditions. The future goal of AD-EYE is to create a smart infrastructure at KTH Campus with autonomous driving vehicles. This iteration of the project covers the foundation of implementing the AD-EYE software on a real car, thus leading to the shift from a simulation environment to the real world.

This shift is accomplished by providing real-world data through perception sensors interfaced mechanically and electrically to the vehicle and logically to the AD-EYE software. The vehicle would then utilise the Controller Area Network (CAN) to send information from onboard sensors and react autonomously to the commands received from the AD-EYE software.

This report covers the solutions introduced pertaining to the goal of this project. Nearly all major stakeholder requirements have been fulfilled and can be expanded for future projects. Finally, a demonstration of the solutions was presented to the stakeholders proving the feasibility of the project while highlighting the considerations for further implementation.

Keywords: Autonomous, Driving, AD-EYE, OBU, On Board Unit, Mechanical Mounting, Power Electronics, LiDAR, Camera, GNSS, CAN, Controller Area Network.

Contents

List of Figures

List of Tables

List of Abbreviations

ABS	Acrylonitrile Butadiene Styrene
AC	Alternating Current
ACC	Adaptive Cruise Control
ADAS	Autonomous Driving Assistance Systems
ADI	Autonomous Driving Intelligence
AI	Artificial Intelligence
APU	Application Processing Unit
BOM	Bill Of Material
C-V2X	Cellular V2X
CAD	Computer Aided Design
CAN	Controller Area Network
CAN-FD	CAN with Flexible Data Rate
CAV	Connected and Autonomous Vehicles
CPU	Central Processing Unit
CU	Controller Unit
DC	Direct Current
DGPS	Differential GPS
DNN	Deep Neural Network
DSRC	Dedicated Short-Range Communications
ECU	Electronic Control Unit
EMI	ElectroMagnetic Interference
EU	European Union
FDX	Fast Data Exchange
FoV	Field Of View
FSD	Full Self-Driving
GLONASS	Global Navigation Satellite System
GMSL	Gigabit Multimedia Serial Link
GNSS	Global Navigation Satellite Systems
GPS	Global Positioning System
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HIL	Hardware-In-The-Loop
IEEE	Institute of Electrical and Electronics Engineers
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
ITRL	Integrated Transport Research Lab

ITU	International Telecommunications Union
KTH	Kungliga Tekniska Högskolan
LAN	Local Area Network
LiDAR	Light Detection And Ranging
MU-MIMO	Multi-User Multiple-Input Multiple-Output
NMEA	National Marine Electronics Association
OBD	On-Board Diagnostics
OBU	On Board Unit
OFDMA	Orthogonal Frequency Division Multiple Access
OSI	Open Systems Interconnection
PLA	PolyLactic Acid
radar	Radio Detection and Ranging
RCV	Research Concept Vehicle
ROS	Robot Operating System
RSU	Road Side Unit
RTK	Real-Time Kinematics
RTOS	Real-Time Operating System
SDK	Software Development Kit
SoC	System on a Chip
SOTA	State of the Art
UART	Universal Asynchronous Receiver-Transmitter
USB	Universal Serial Bus
V2X	Vehicle-To-Everything
VCC	Volvo Cars Corporation
WLAN	Wireless Local Area Network

1 Introduction

This chapter includes the background, project description, stakeholder requirements, and report disposition.

1.1 Background

Since 2017, researchers at Kungliga Tekniska Högskolan (KTH) have been developing the AD-EYE software for research in autonomous driving. The AD-EYE software supports a virtual simulation environment, enabling the simulation of a vehicle driving autonomously in different environments, e.g., the KTH Campus. The AD-EYE software is built upon the middleware Robot Operating System (ROS), the simulation environment Prescan, the nominal channel Autoware and a safety channel developed by KTH, the KTH Safety Channel. The perception system in the simulation environment is enabled through different sensors such as cameras, Light Detection And Ranging (LiDAR), Radio Detection and Ranging (radar), Inertial Measurement Unit (IMU), and Global Navigation Satellite Systems (GNSS). The core decision-making algorithm, which relies upon the information obtained from the perception system to plan upcoming actions, exists in Autoware, which is an open source ROS-based software stack for autonomous vehicles.

1.2 Project Description

The task was to replace the simulation environment with a physical vehicle and to develop a testing platform for further development and research in autonomous driving at KTH. This involved integration and implementation of external sensors and the existing AD-EYE software on a commercial vehicle. The integrated sensors consisted of a camera, a LiDAR, and a GNSS, which enabled the perception and localisation system on the vehicle. Furthermore, the testing platform should be able to send and receive control commands for enabling different functions of the vehicle, e.g., turning the wheels, activating the hazard lights, or shifting the gears. An On Board Unit (OBU) should be used for interfacing the communication between the sensors, the AD-EYE software, and the Controller Area Network (CAN) system of the vehicle.

1.3 Initial stakeholder requirements

This section covers the initial stakeholder requirements.

1. Initial milestones:
   1.1. CAN shall be interfaced with the vehicle and verify vehicle controls.
   1.2. All provided sensors shall be integrated optimally on-vehicle, that is, mounting, power, and electrical interface.

1.3. All provided sensors shall be logically interfaced to AD-EYE.

1.4. Vehicle-To-Everything (V2X) communication shall be enabled between the Road Side Unit (RSU) and the vehicle using 5G and a selected wireless communication protocol.

1.5. AD-EYE shall be interfaced towards the simulated vehicle, that is, AD-EYE on NVIDIA DRIVE™ PX 2 within a simulator environment.

1.6. AD-EYE shall be interfaced towards vehicle in a Hardware-In-The-Loop (HIL) mode, that is, AD-EYE on NVIDIA DRIVE™ PX 2 with CAN interfaced to the vehicle.

2. Intermediate scenarios

2.1. The AD-EYE Graphical User Interface (GUI) shall be developed to visualise the vehicle state, and environment through external sensors and communications.

2.2. The vehicle shall be controlled via "open loop controls" through the AD-EYE GUI by a cabled connection.

2.3. Localisation shall be tested by mapping the driving area and integrating it so that both the localisation map and road network map are available.

2.4. The vehicle must be able to drive autonomously along a short predefined trajectory.

2.5. The vehicle must be able to drive autonomously to a given goal.

3. End scenarios

3.1. The vehicle must safely stop when an emergency signal is received from the RSU over V2X.

3.2. The vehicle must be able to react and navigate around a non-existent object introduced into the vehicle's perception by the RSU.

3.3. The vehicle must be able to enter follow mode and match its speed to a non-existent leading vehicle introduced into the vehicle's perception by the RSU.

1.4 Final stakeholder requirements

After careful consideration and discussion, the stakeholder requirements were condensed and refined in order to facilitate the process. The final stakeholder requirements are listed below:

1. Initial scenarios:

1.1. Demonstrating CAN interfacing (regular CAN and/or CAN-FD) to the vehicle and verifying the vehicle controls.

i. The vehicle should have safety procedures to disable the vehicle in case of an emergency stop (e.g., stop button).

ii. The vehicle should allow the safety driver to override the commands sent by the AD-EYE software to CAN (e.g., if the safety driver tries to brake or steer the steering wheel during AD mode, the lower level should allow it).

1.2. Sensor integration on-vehicle (mounting, power, electrical interfaces).
    i. The mechanical installation of the vehicle should be planned, documented, and implemented for the provided hardware.
    ii. The electrical installation of the vehicle should be designed for the proposed hardware and capable to accommodate one additional embedded computer, camera, and LiDAR.
    iii. The vehicle should contain an electrical switch as a safety procedure to disable the vehicle in case of an emergency stop (e.g., stop button).
    iv. All the sensors installed in the vehicle must be calibrated and the process must be clearly documented (e.g., code used and README).
    v. The vehicle should be able to localise itself and map the environment using a LiDAR and the GNSS-RTK.

1.3. Demonstrating AD-EYE interfacing towards simulated vehicle (AD-EYE on an on-board computer).

1.4. Demonstrating AD-EYE interfacing towards vehicle in HIL mode, integrating (1.1) and (1.3).

2. Demo requirements:

2.1. Point number (1.4) address testing in open loop. All the above steps must be tested, presented and documented for the stakeholders in the final demo.

3. Optional requirements:

3.1. AD-EYE GUI – visualisation of vehicle state, environment (external sensors) and any other vehicle information.

3.2. Vehicle open loop "controls" through the AD-EYE GUI – cabled connection – request control from GUI.

3.3. Mapping of the driving area (or reuse of existing map) and map integration: both localisation map and road network map.

Throughout the project, the aim was to accomplish all of the stakeholder requirements. The final stakeholder requirements will be referred to as *the stakeholder requirements* throughout the entire report.

1.5 Report disposition

This report is presented over 8 chapters excluding the appendices. Chapters 2 and 3 provide relevant background information and project execution strategy respectively. Each chapter from chapter 4 builds on the previous chapters, providing a detailed perspective on each aspect of the implemented solutions with conclusions and future work presented at the end.

To make the report reader-friendly, all sub-sections from chapter 4 are presented in the following order: On Board Unit and AD-EYE Software, Camera, LiDAR, GNSS-RTK, CAN Communication,

Mechanical Mounting, and Power Electronics. Chapter 6 includes an extra sub-section discussing the fulfilment status of the stakeholder requirements and Chapter 8 includes an extra sub-section discussing the considerations necessary from the overall system perspective.

The On Board Unit and AD-EYE Software section discuss the OBU used to host the AD-EYE software while also highlighting information about the AD-EYE software necessary to implement other parts of the project. The camera, LiDAR, and GNSS-RTK sections discuss the key components that provide localisation data and the interfacing methodology towards the AD-EYE software. The CAN Communication section discusses the CAN network and exchanging of information between the OBU and the vehicle. Finally, the Mechanical Mounting and Power Electronics sections discuss the mechanical and electrical interfacing of the OBU, LiDAR, camera and the GNSS-RTK to the vehicle.

The appendices provide key information such as the commands and tools used to initialise the AD-EYE software and sensors, camera and LiDAR calibration information, and Bill Of Material (BOM) of the mechanical and electrical systems.

2 Literature Review

2.1 Existing Solutions

The existing solutions for autonomous vehicles will be generalised into two different segments:

1. *Partially Autonomous Vehicles*, which refers to the commercial availability of partially autonomous vehicles that exist on the market today.
2. *Testbeds*, which refers to vehicles developed for testing, verifying, and validating technologies used for autonomous driving.

Both partially autonomous vehicles and testbeds may be equipped with different levels of automated driving. In order to distinguish these different levels, the grading scale Levels of Driving Automation may be used, as shown in Figure 1. The figure shows a range of increasing levels of driving automation, ranging between level 0 (L0) and level 5 (L5). The figure also clearly states the meaning of each level [1].

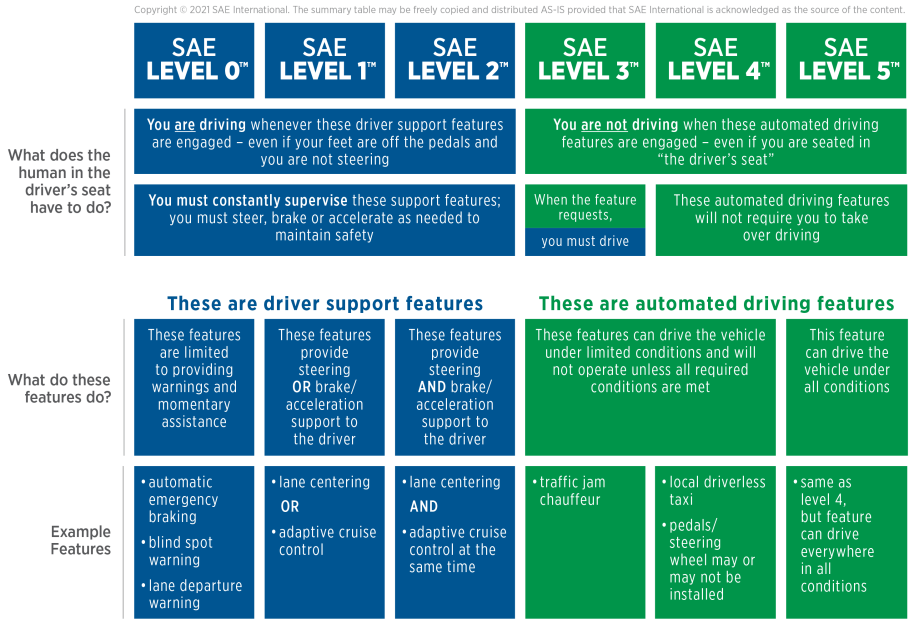


Figure 1: The various Levels of Driving Automation, ranging between level 0 (L0) and level 5 (L5) [1], edited with [2].

2.1.1 Partially Autonomous Vehicles

Different driver assistance systems are becoming increasingly ubiquitous in modern vehicles. These systems are used to increase performance and safety in various car related activities, such as parking, lane keeping, traffic sign identification, and collision avoidance. By combining Autonomous Driving Assistance Systems (ADAS) with some form of Artificial Intelligence (AI), the automotive industry is developing automated technologies that are evolving towards becoming more autonomous and requiring less user interaction [3].

Currently, many well-known car manufacturers are involved in the development of vehicles equipped with some level of driving automation. Several manufacturers are already supplying vehicles equipped with L1 and L2 capability for automated driving, such as Volvo with the Pilot Assist [4], BMW with the Driver Assistance Package [5] and Ford with the Co-Pilot360™ [6]. Features such as lane keeping assistance, collision avoidance, evasive steering assist, hill descent control, and cruise control are categorised as L1 and L2 capabilities. These features make use of sensors, microcontrollers, and algorithms to produce the desired output [6].

The Tesla line of vehicles is yet another example of vehicles with L2 capability for automated driving. The Tesla AutoPilot™ technology takes input from surrounding environments using eight cameras, one radar, and twelve sonars [7]. The vehicle is also equipped with Tesla's self developed computer, the Full Self-Driving (FSD) chip. The chip has been developed with the aim of eventually equipping their line of vehicles with higher levels of capability for automated driving [8]. According to Tesla, their line of cars provide a fully self-driving experience. However, the technology in Tesla cars still requires overseeing by a driver and erratic behaviour during driving has been reported by their own users [9].

Currently, there exists manufacturers that offer vehicles with L3 capability, such as Audi and Honda [11]. However, reaching higher levels of driving automation is still limited by safety regulations, performance issues in adverse conditions and a multitude of technical difficulties that inhibit the research and commercialisation of fully L4 or L5 capability [12]. Figure 2 shows an example of how a research vehicle may be designed.

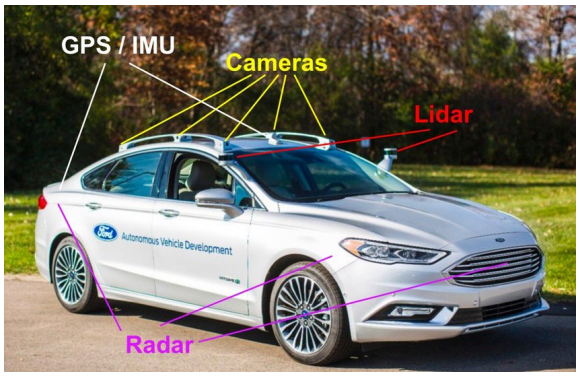


Figure 2: Ford's research autonomous vehicle [10].

A few companies provide a robotaxi experience [13], such as Waymo and Autox [14]. The Waymo line of vehicles is equipped with a LiDAR, about 19 to 29 cameras (depending on the vehicle model), radar, and an onboard computer responsible for decision making, trajectory planning, and processing sensor data. Waymo has been granted the permit in the US to offer a robotaxi service in a few states to a limited amount of users [15].

2.1.2 Testbeds

Testbeds are used during the development of autonomous vehicles for implementing rigorous and comprehensive testing. Simulation and physical testing in specific conditions are crucial for ensuring proper behavior and quality of both hardware and software. The vehicle must go through extensive testing in order to assure that it complies with the existing demands, laws, and regulations related to autonomous driving. Although testbeds for autonomous driving are rare in general, there are a few establishments that are developing testbeds for autonomous driving. Development of testbeds is being driven forward in order to satisfy the need for a safe environment where testing can be executed with accuracy and close similarities to real world situations [16].

AstaZero is a testbed located near Göteborg in Sweden. It has been stated to provide a rural road lane, a high-speed area, and an indoor dry zone for testing in optimal conditions. The testbed provides bi-directional, right-hand traffic and a road width of 7 meters. The infrastructure has been developed to support a myriad of modules for the test vehicles, such as communication resources, Differential GPS (DGPS), dummies, and wireless coverage [17].

Another testbed of note is Millbrook, located in Culham in the UK. Development of Connected and Autonomous Vehicles (CAV) technologies and its implementation using 5G is emphasised at Millbrook to test and validate ADAS. The testbed also contains a facility called the Autonomous Driving Village, where the development of the aforementioned CAV takes place [18].

There are many more upcoming testbeds, like Tesla's Gigaberlin [19], TiHaN-IIT in India [20] and Hyundai in Namyang R&D Centre in South Korea [21]. Universities are expanding their research into autonomous vehicles as well, with Oxford creating the Selenium software system with the goal of uploading it into any vehicle with the right type of sensors. The University of Michigan is creating a large model of a city that will serve as a testbed to evaluate how autonomous driving will affect urban planning [22]. As the future of autonomous vehicles approaches, there is a growing demand to create environments where vehicles can be safely tested in simulations that are close to reality.

2.2 On Board Resources

This section covers the resources available on board the car, along with their integration strategy.

2.2.1 CAN

The CAN protocol is the internationally dominating serial network protocol for embedded control systems in passenger cars [23]. The protocol covers the lower layers of the Open Systems Interconnection (OSI) seven-layer model for data communication, including the data link layer and the physical layer. However, the medium of data transportation and the interfaces depend on the application [24]. In a vehicle, the various electrical systems are controlled by the Electronic Control Unit (ECU)s. These devices receive information from e.g., sensors and command the various actuating systems to perform the proper action [25]. The CAN bus connects these ECUs in a physical

network, resulting in all ECUs being able to broadcast and receive information on a single bus [26]. For the data communication, the CAN protocol uses a non-destructive arbitration scheme, where higher priority messages can be transmitted on the bus without interruption or notion of collisions [27]. Altogether, this makes the CAN system extremely robust and efficient and hence the de facto standard for communication in the majority of the vehicles today [26].

CAN provides the base of the communication in a vehicle, but higher-layer protocols are necessary in order to further specify how the data is transmitted over the network and how to interpret and use the data. Several standard higher-level protocols exist, of which On-Board Diagnostics (OBD) is the most commonly used in cars [28].

However, the original CAN protocol has limitations in terms of performance and communication bandwidth. The serial network protocol CAN with Flexible Data Rate (CAN-FD) was hence developed to meet the demands of complex electrical systems in the automotive industry. CAN-FD has, compared to the original CAN protocol, increased size of data field and higher communication bandwidth. CAN-FD supports a data field up to 64 bytes and a bandwidth of up to 8 Mbps [29]. The protocol is internationally standardised, as well as the original CAN protocol. The CAN-FD protocol enables higher speeds compared to CAN, and is thus more suitable for meeting the increasing complexity of requirements on bandwidth and higher data transmission rates. In order to obtain faster performance, the CAN-FD protocol will increase the bitrate when only one node is transmitting. The nodes are then re-synchronised before the transmission is resumed [30].

2.2.2 On Board Computer

To handle the additional sensor inputs and calculations required for an autonomous vehicle, a dedicated computer is required. The NVIDIA DRIVE™ PX 2 is a computer from NVIDIA Corporation designed with this in mind. The NVIDIA DRIVE™ PX 2 is the second generation of hardware for NVIDIA DRIVE™, a computing platform for autonomous driving. NVIDIA DRIVE™ leverages artificial intelligence and deep learning for the heavy computational tasks required for autonomous driving [32]. For the rest of the report, NVIDIA DRIVE™ PX 2 will be referred to as 'PX2' only.

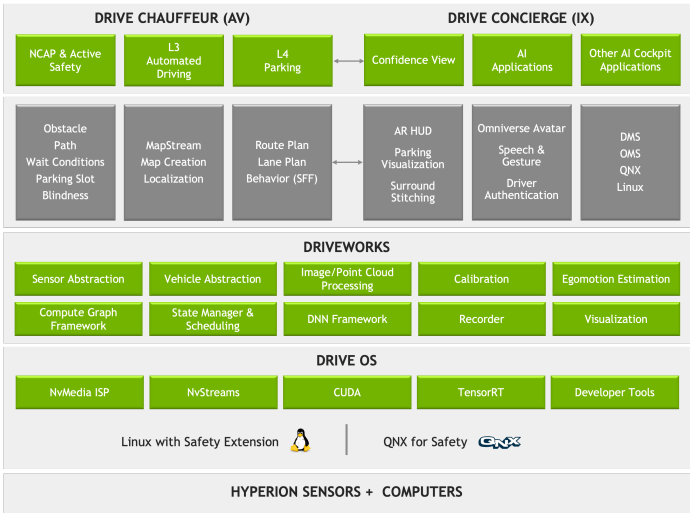


Figure 3: The software layers of NVIDIA DRIVE™ [31].

The PX2 is based on the Pascal microarchitecture and is available in two configurations:

- AutoCruise with one Tegra X2 System on a Chip (SoC)
- AutoChauffeur with two Tegra X2 SoCs

Each Tegra X2 SoC contains a Denver 2 dual-core Central Processing Unit (CPU), an ARM Cortex-A57 quad-core CPU, and a Pascal Graphics Processing Unit (GPU) [33]. The PX2 includes several relevant hardware interfaces including CAN, Universal Asynchronous Receiver-Transmitter (UART), Gigabit Multimedia Serial Link (GMSL) camera inputs, and HSAutoLink II. Interfaces found on personal computers are also available on the PX2s, such as display outputs, Ethernet, and Universal Serial Bus (USB) [34]. The less powerful of the two, AutoCruise, is mainly designed to aid in driving by handling automated driving on highways and mapping. The more powerful, AutoChauffeur, can handle point-to-point driving. If more computational power is required, multiple systems can be interconnected [35].

NVIDIA DRIVE™ does not only offer hardware solutions but also software, ranging from a simulation environment [36] to a virtual assistant and fully autonomous point-to-point driving [37]. The software is built of layers on top of each other, as can be seen in Figure 3. At the core is DRIVE OS, an embedded Real-Time Operating System (RTOS) that is designed to be safe to use in-vehicle. DRIVE OS includes libraries to interface with the NVIDIA DRIVE™ hardware, as well as different deep learning libraries. The middleware DriveWorks is implemented on top of DRIVE OS in order to interface with sensors using different tools and libraries. DriveWorks also provides a Deep Neural Network (DNN) framework. The next layer in the NVIDIA DRIVE™ software is DRIVE AV. DRIVE AV can handle the rest of the tasks required for point-to-point autonomous driving. This includes pre-trained DNNs for mapping, including perception, and planning. DRIVE AV also includes NVIDIA Safety Force Field™, a computational framework for safety on the road. The NVIDIA DRIVE™ software also provides DRIVE IX, an interior AI for occupant monitoring, and DRIVE Concierge, a virtual assistant [31].

2.3 V2X Communication

V2X communications refer mainly to the group of technologies and standards that can be used to establish communication between a vehicle and the surrounding environment including other vehicles, pedestrians, networks, and infrastructures such as RSUs[1]. Using V2X enables different vehicles to share various sensor data in order to avoid collisions or traffic congestion, which creates a safer and more efficient transport environment. Depending on the chosen set of standards of V2X, different wireless communication technologies can be implemented. Latency, coverage, reliability, and power consumption are examples of factors that should be taken into consideration when choosing the proper communication technology [38]. However, the need of transmitting heavy amounts of data narrows down the choices to either Wireless Local Area Network (WLAN) or cellular technologies.

[1] See section 2.6.4

2.3.1 WLAN

WLAN represents a group of devices and computers that are connected to each other using radio transmissions. One of the most known WLAN technologies is WiFi, which refers to a class of certified wireless networking solutions that comply with the Institute of Electrical and Electronics Engineers (IEEE) 802.11b industry-standard [39]. However, due to the fact that the previous WiFi versions are no longer able to meet the latency and coverage requirements for autonomous vehicles, only the latest WiFi technologies, namely WiFi 5 and WiFi 6, are taken into consideration.

WiFi 5 is a user-friendly word for the IEEE 802.11ac standard. WiFi 5 uses a frequency of 5 GHz, and it offers a maximum throughput of 7 Gbps across different channels. On the other hand, IEEE 802.11ax standard, known as WiFi 6, uses a frequency of either 2.4 or 5 GHz and is optimised for better performance in large outdoor implementations. Furthermore, WiFi 6 offers reduced power consumption and up to 9.6 Gbps data rate through multiple channels by using more advanced features such as Orthogonal Frequency Division Multiple Access (OFDMA) and Multi-User Multiple-Input Multiple-Output (MU-MIMO) [40].

Put to the test, WiFi 6 provided up to 75% less latency resulting in significantly higher speed than WiFi 5 [41]. However, the performance of the used WiFi technology varies strongly depending on the location of the user, using devices, and the local radio environment.

2.3.2 Cellular Telecommunication

Cellular Telecommunication is based on the usage of a large number of radio stations, where each station is able to communicate with a large number of devices in a certain small area. Furthermore, these stations have the ability to communicate with each other, which establishes a huge amount of communication between different devices [42].

The latest cellular telecommunication technology, namely 5G, is expected to be able to address the problem of reliability, latency, and peak throughput per connection. Due to its high use of virtualisation of network infrastructure, 5G provides a reliability of almost 100% and an end-to-end latency of less than 1 ms with a data rate of up to 100 Mbps. This is done by making better use of edge computing technologies compared to previous cellular telecommunication technologies [40].

While the 5G technologies provide a solution to faster communication, it requires access to large amounts of frequency spectrum meaning that both coverage and speed can vary strongly depending on the available frequency spectrum. A frequency spectrum between 1 GHz to 6 GHz allows for transmitting a good amount of data through a significant distance. On the other hand, a frequency spectrum from 24 GHz, called the millimeter wave spectrum, is able to transmit huge amounts of data, being more suitable for vehicle communication. However, millimeter waves can easily be absorbed by gases and rain, and it can be impeded by buildings and trees which makes the coverage distance for this spectrum short [43].

2.3.3 ITS-G5 and C-V2X

There are two existing solutions for implementing wireless communication technologies to establish reliable communication for vehicles. The first solution is Dedicated Short-Range Communications (DSRC), also known as ITS-G5 and based on the IEEEs 802.11p technologies. This solution introduces the standards to exchange wireless short-range information messages between the vehicle and the RSU at a certain location. The other solution is based on cellular communication technologies and is known as the Cellular V2X (C-V2X). Using this standard, a vehicle is able to communicate with a base station to connect to the surrounding environment as well as to information bases. Both of these technologies use a radio spectrum band of 5.9 GHz which is allocated by the European Union (EU) commission to be exclusively used for V2X based applications [44].

ITS-G5 uses the radio spectrum in a simple and efficient way to obtain a robust V2X short-range communication. However, it suffers from a great deal of challenges including limited mobility support, poor coverage range, latency, and reliability. This has raised the need to develop C-V2X to address these problems by leveraging existing cellular network infrastructures. C-V2X provides a reliable, high-speed communication by combining short-range direct communication and long-range network communication [44].

2.4 Sensor

In the following subsections, relevant sensors for the project will be presented.

2.4.1 LiDAR

A LiDAR, see Figure 4, is a device that determines the distance to an object or a surface by using a laser emitter and a pulsing laser beam. The distance to the object is determined by measuring the time delay between the emission of the beam from the laser emitter to the detection through the reflected signal.

With the information of the pulsing laser beam and the use of a differential GNSS system and an Inertial Navigation System (INS), tens of thousands of points per second are obtained, forming point clouds [2], which are useful for navigating in surrounding environments [46].

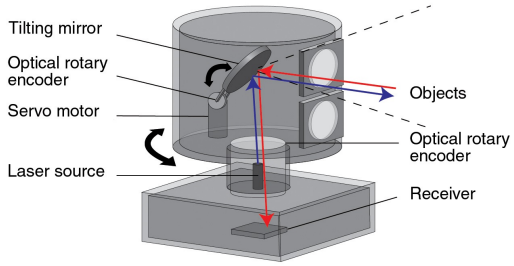


Figure 4: LiDAR components and operation [45].

[2] See section 2.6.2

2.4.2 Camera

Cameras are widely used on autonomous vehicles to detect surrounding objects. 3D cameras are capable of capturing data at high resolution. Two types of cameras are commonly used in autonomous vehicles: mono-cameras and stereo-cameras. A camera with only one sensor capturing video that has to be processed is called a monocular system. If the system has two separated cameras it is a stereo-vision system [47].

The distance to an object can be obtained by determining the projection difference between the object of two stereo cameras. The maximum distance range and depth that a camera is capable to detect depends on different parameters, such as the focal length of the camera, the baseline, and the density of pixels of the camera. In addition to defining the distance to the objects, the images of the cameras have to be processed to interpret the detected object, and these computations are typically very heavy. Moreover, the processing has to occur in real-time which adds difficulty to the implementation of cameras in a system [48].

2.4.3 Radar and Ultrasonic Sensors

Radars are sensors that send and receive electromagnetic waves to detect and locate objects in the environment. As vehicles are good reflectors, radars can be used to obtain the position and distance of the other surrounding vehicles.

The International Telecommunications Union (ITU) defines two categories of radar systems for vehicles. The first one enables an Adaptive Cruise Control (ACC) and collision avoidance in ranges up to 250m. The second category adds functionalities for passive and active safety like pedestrian detection in a range between 50 and 100m. It is possible to find three types of radars depending on the maximum distance range and the field of view. In order to obtain 360° coverage, radars should be placed on several sides of the car [49].

Ultrasonic sensors use echolocation by transmitting high-frequency sound waves to obtain the distance to nearby objects. They can combine information with multiple sensors such as LiDARs or cameras of the vehicle to obtain full data information of the objects around the car. Ultrasonic sensors are useful for detecting objects at close distances that are moving at low velocities. They are important for increasing visibility in the close surroundings of the car. [50].

2.4.4 Inertial Measurement Unit

An IMU is a device that tracks the orientation, position, and acceleration of an object. This is done with accelerometers, gyroscopes, and sometimes also magnetometers, which track the change of the Earth's magnetic field in order to perceive its movement. IMUs are often used in inertial positioning systems where the sensor data from the IMU is used to determine its position relative to a global coordinate system. One specific use case could be to aid the GNSS system of a car when it enters a tunnel and the GNSS signal is blocked, with an IMU the car can still track its position with dead reckoning [51]. A disadvantage of the IMU compared to other positioning techniques is

that the positional error is accumulated over time. This is why it is best used together with another positioning technology that can correct the sensor when it is drifting out of position [52]. The IMU should be placed as close to the center of gravity as possible [53].

2.4.5 Global Navigation Satellite Systems

GNSS is a collective name for satellite-based navigation and positioning systems [54]. There are several different GNSS systems such as Global Positioning System (GPS), Global Navigation Satellite System (GLONASS), and Galileo [55]. In order to use GNSS systems, a free line of sight to the satellites and a GNSS receiver is required [56]. A GNSS system may be used to position a vehicle in a global coordinate system along with a IMU mentioned above. The GNSS antennas should be placed at the top of a vehicle, so it always is in line of sight with the satellites.

2.5 Sensor Placement

Deciding what type of sensors to be used in an autonomous driving project is crucial for gathering the right type of data. However, choosing the right spot to place these sensors is of equal importance as it determines the amount of information that can be used to perceive the surrounding environment. This section presents different placement strategies that are used by autonomous vehicle manufacturers.

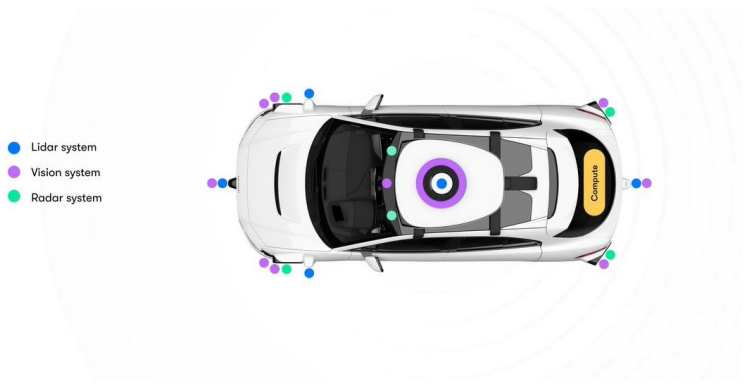


Figure 5: Sensors configuration in a Waymo vehicle [57], edited with [2].

2.5.1 LiDAR Placement

Most manufacturers, including Waymo, choose to equip their autonomous research vehicles with a main LiDAR at the top of the vehicle in order to create a full 360° map of its surroundings, see Figure 5, where the blue dots represent the placement of the LiDAR. However, LiDARs can also be placed on the sides of the vehicle in order to cover the blind spots that the main LiDAR is unable to see.

2.5.2 Camera Placement

According to Wang et al., a perception system needs to be equipped with at least six cameras in order to fulfill the basic requirements for autonomous driving and create a 360° coverage for the car's surroundings [48]. For that purpose, three main types of cameras are being widely used by autonomous vehicle manufacturers. These cameras are front-view, rear-view, and side-view cameras. The camera distribution for the Apollo vehicle, shown in Figure 6b, is an example of how these three types of cameras can be placed in an autonomous vehicle. In this vehicle, the minimal number of required cameras is being used, one front-view camera, one rear-view camera, and four side-view cameras. On the other hand, Tesla's Model 3 vehicle, seen in Figure 6a, has a different placement of these cameras, where four side cameras are being used. Two of them are pointed forward and the other two are pointed backward. In addition, the vehicle is equipped with triple front-view cameras to ensure full coverage of the surrounding environment [48].

(a) Tesla's Model 3 vehicle.

(b) Apollo vehicle.

Figure 6: Cameras configurations in autonomous vehicles [48].

2.5.3 Radar Placement

A vehicle equipped with a LiDAR and a set of cameras would be able to detect objects and understand the surrounding environment. However, radars are still required to complement these properties by providing instant object detection and accurate velocity measurements. It is therefore important to install radars in positions where their electromagnetic waves do not get interrupted. Furthermore, radars should be placed in positions that allow them to detect obstacles and cover the blind spots that the LiDAR and cameras might miss [58]. In Figure 5, the green dots represent the radars' placement in a Waymo vehicle. By distributing the radars around the vehicle in this fashion, the detection can cover the complete surrounding area,

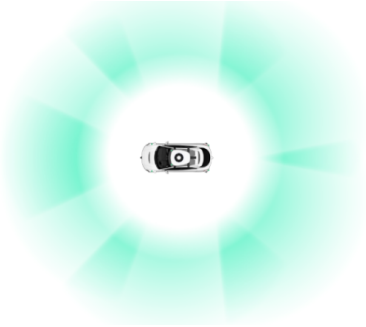


Figure 7: Radars' coverage in a Waymo vehicle [58].

see Figure 7. This is important in order to minimise the presence of blind spots.

2.6 Related AD-EYE Projects

Under this section, other related AD-EYE projects are presented along with a short summary of the AD-EYE software stack.

2.6.1 Software Stack

A schematic view of the AD-EYE software stack can be seen in Figure 8 along with the PX2[1] in the bottom right corner and the Research Concept Vehicle (RCV)[2] to the bottom left. The software stack can roughly be divided into two parts, the Autonomous Driving Intelligence (ADI) and the simulation environment. The ADI system is based on Autoware [59] together with a safety channel developed by KTH, see the right part of Figure 8. Both Autoware and the safety channel run in a middleware called ROS. The simulation is done in Prescan which is a simulation tool from Siemens [60]. Simulink [61] is used as an interface between ROS [62] and the Prescan environment which is not natively compatible. The simulation environment makes it safer and easier to test new sensor models, vehicle models and driving scenarios.

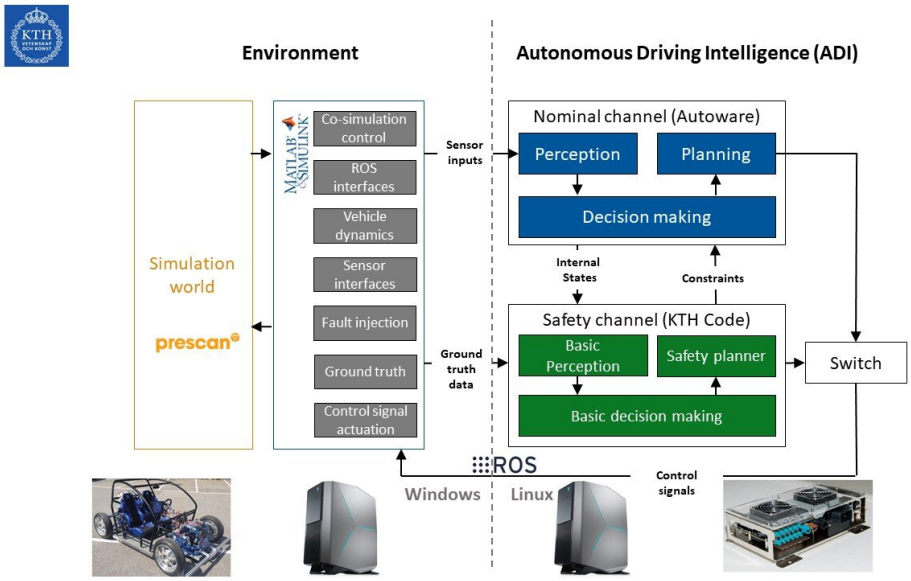


Figure 8: Schematic view of AD-EYE software stack [63].

[1] See section 2.2.2
[2] See section 2.6.3

2.6.2 Point Cloud Maps Implementation

In order to test the AD-EYE software in real-life applications, highly detailed point cloud maps need to be generated and integrated into the system. This would introduce a real-world reference to the platform, which with the use of real-time data from sensors, will enable trajectory planning and autonomous driving [64].

During a previous research work held by students at KTH, several point cloud maps were established. This was done by using a LiDAR and an IMU to generate and record the point cloud map into several ROS-bag files which were then processed by the Autoware channel into a point cloud data file. Furthermore, the *VectorMapBuilder* tool was used to generate vector maps that could be transformed to the PX2 to be used during autonomous driving [64].

2.6.3 AD-EYE On Research Concept Vehicle

The RCV and RCV-E are research platforms developed by the Integrated Transport Research Lab (ITRL) lab at KTH. The purpose of these research platforms is to provide vehicles where research results can be tested and verified in the real world. The RCVs are developed in collaboration with students and multiple departments at KTH [65]. The RCVs are equipped with technology that enables them to be controlled via an API or manually [66].

In 2020, students from the mechatronics capstone project used the RCV-E to test the AD-EYE software in a real environment. The AD-EYE software stack was installed on a PX2 and integrated into the car. The onboard hardware and software from the RCVs were interfaced with the AD-EYE software to actuate the vehicle. The sensor setup consisted of a LiDAR, odometers, and an IMU. The team achieved autonomous driving for a short distance on campus. However, the AD-EYE software would lose track of the RCV's position after a short while. The authors discussed contributing factors to this particular performance issue. The used solution did not implement GNSS, and consequently, the software struggled to navigate when moving objects, such as trees moving with the wind, was present [64].

2.6.4 Road Side Unit

A roadside unit is a communication relay, designed to be an access point for a vehicle. The RSU connects with the onboard unit of a vehicle, over a DSRC protocol [67]. By doing so, the vehicle and the RSU can share information wirelessly, such as GNSS, automatic toll collection, or help schedule traffic light timing. By incorporating multiple RSUs, the vehicles on the road can effectively communicate with each other, sharing important data, such as location and speed. The RSU is intended to act as a remote agent supplying vehicles with V2X-data, in order to improve traffic management.

The design of the RSUs differs depending on the type of functions they need to fulfill. Regardless of the functions, some key components are to be expected to have mandatory features, such as wireless capabilities, message caching, an antenna to receive GNSS signals and communicate wirelessly,

an Application Processing Unit (APU) to run applications, a Controller Unit (CU) to direct the operations and a modem to communicate with a data center [68]. Moreover, the placement of a RSU in the roadside infrastructure is important for establishing effective coverage for the vehicular network.

A RSU was designed to be used with the AD-EYE software by the students at KTH during 2021. This RSU houses a WiFi router, a WiFi antenna, an ITS-G5 router, and an antenna, GNSS, a car battery, a motherboard, a GPU, and a thermostat. The aim was to relieve the OBU computing by using Edge Computing, as the project made use of Raspberry Pi's and Turtlebots. The RSU would then process data from the aforementioned units. Furthermore, two algorithms of note were developed, an object detection algorithm and a platooning algorithm [69].

2.7 Concept Design

In the following section, a concept design is proposed, based on the information presented in the State of the Art (SOTA), and the Current Design section.

2.7.1 Sensors

For the autonomous car to be able to perceive obstacles on the road and locate itself, different sensors are necessary. To fully define the concept design for the sensors, it is necessary to explore the possible placement of the sensors and the required mounting of them. One LiDAR, two to three cameras, one radar, and a GNSS are expected to be used.

2.7.2 Sensor Placement

A single 3D-LiDAR could be placed in the middle of the vehicle roof, as is shown in Figure 9. This setup obtains only one point cloud, that does not require alignment with other point clouds of other LiDARs. Moreover, with this placement, a 360°coverage would be obtained. The main disadvantage of this placement would be that the roof of the vehicle would create blind areas around it, as can be seen in Figure 9. One way to reduce these blind spots could be to elevate the LiDAR to give it a greater range of view. However, this would increase the need for a more robust mounting system, since the sensor would be more exposed.

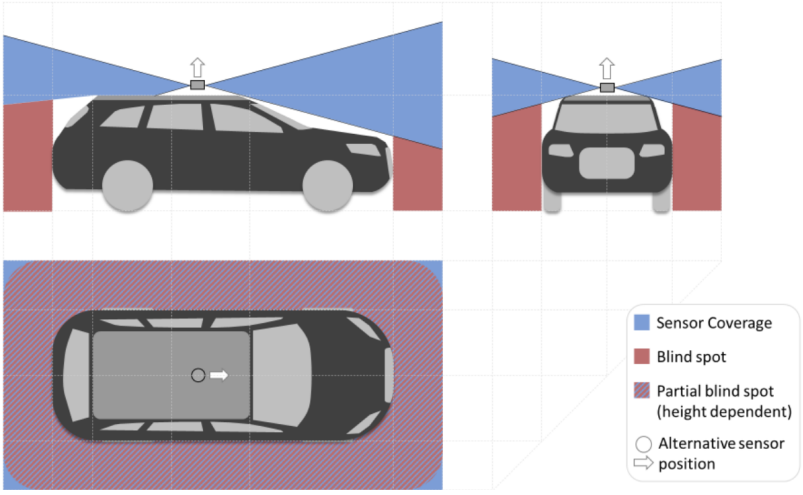


Figure 9: LiDAR placement [70].

If two cameras are provided, the optimal choice would be to place one in the front and one in the rear of the car as can be seen in Figure 10. By placing one camera in the front, the detection of traffic lights, lanes in the road, and obstacles would be enabled. The main purpose of the placement of the rear camera would be to perform auxiliary reversing for safe parking while detecting static and dynamic obstacles. The drawback of this sensor placement, however, would be limited detection of objects that are located on the sides of the vehicle.

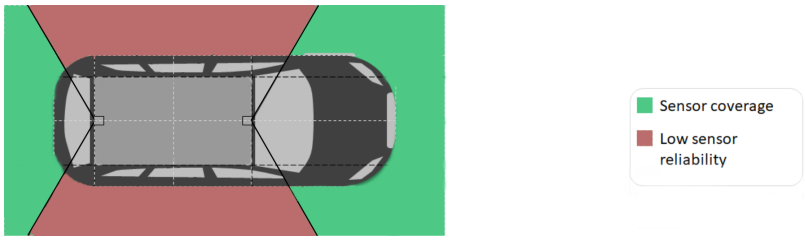


Figure 10: Two cameras placement [70], edited with [2].

If three cameras are provided, one placement possibility would be to have two cameras in the front and one in the back, as is illustrated in Figure 11. This setup would allow the complete view of the front and increment the view of both sides around the vehicle. This placement would result in an increased field of view, that could help detect vehicles and other objects around the car and evidently improve the features for parking safely.

If the vertical field of view of the front camera turns out to not be wide enough to correctly detect traffic lights, one of the front cameras could be placed on the side tilted upwards. This placement would help to correctly detect traffic lights and road signs. In Table 1 a comparison between the

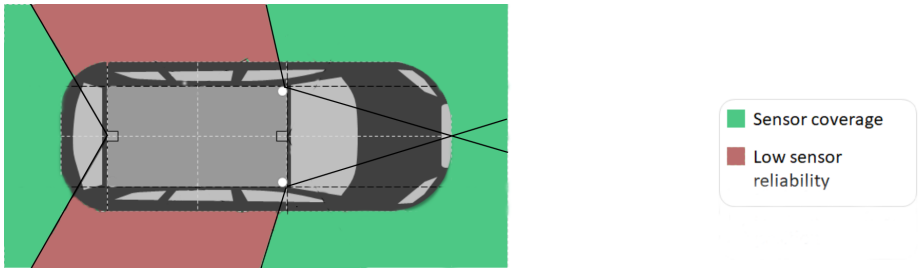


Figure 11: Three cameras placement [70], edited with [2].

different camera placement options can be seen.

	2 cameras	3 cameras	
		One on each side	**One in the center, one tilted**
Pros	Simple mounting, simple sensor fusion	More visible parts on the sides	Ability to detect high traffic lights and signals
Cons	Shorter field of view, blind spots in the side, difficulty detecting signs and traffic lights	Complexity in data fusion, difficulty detecting signs and traffic lights	Complexity in data fusion, shorter field of view horizontally

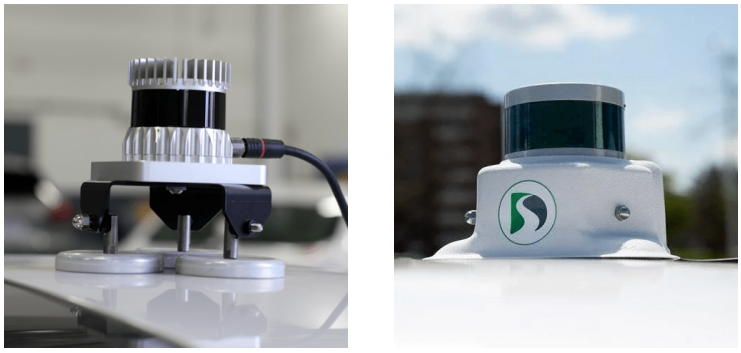
Table 1: Camera placement options comparison.

In case that one radar or one ultrasonic sensor would be provided, it could then be placed in the front of the vehicle. As commented in section 2.4.3, radars can be helpful in detecting the surroundings while ultrasonic sensors can be important in poor visibility conditions.

Referring to the localisation sensors, if the GNSS has a receiver and its own antennas, the receiver would preferably be placed near the center of gravity of the car and the antennas should be placed in an uncovered place. The IMU can be installed in a shielded container, deep into the vehicle's chassis. By choosing this placement, the IMU is completely protected from weather and other environmental conditions [71].

2.7.3 Sensor Mounting

As mentioned in section 2.7.2, the position of the LiDAR could be on the roof of the vehicle. This placement would cause some blind spots as illustrated in Figure 9. In order to reduce the blind spots, the sensor could be elevated by using the mounting gadget shown in Figure 12. That mount consists of three magnetic plates that could be used on slightly curved surfaces.



(a) Without shroud (b) With shroud

Figure 12: Magnetic LiDAR mount [72].

Another option that could be used to mount both LiDAR and cameras is the roof rack of the car with load bars. If the vehicle has a naked roof, a load carrier feet could be used to replace the roof rack. These could be installed to the side of the car, in the front and in the back, connected by load bars that could be used to mount LiDAR and cameras.

A third option for the perception sensors mounting is to use suction cups. Suction cups use negative pressure to adhere to the nonporous surface. It is a cheap solution and easy to install. However, vibrations would produce a lot of noise on the data. For that reason, additional vibration dampers could be used [73].

2.8 Communication

The vehicle has several components that need to communicate with the PX2, which in turn has to communicate with the RSU, see Figure 13. The design should encompass these components so that the line of communication is fast, efficient and constructed with regards to minimising risks such as delay, package loss or a faulty connection, which in turn could impair time-critical functions.

The sensors could be connected through a combination of the CAN, Ethernet, and USB hardware to the PX2, which is the main CPU. The PX2 could in turn be connected to the RSU through the use of a wireless protocol, e.g., the ITS-G5 or any other wireless protocol that fulfills the performance needs.

The PX2 contains two Tegra cores, which are already connected to each other but do not share operations like other multi-core processors do. Instead, to allow the burden of operations to be relieved, the two cores connection should be harnessed to improve the performance or add redundancy in order to increase safety.

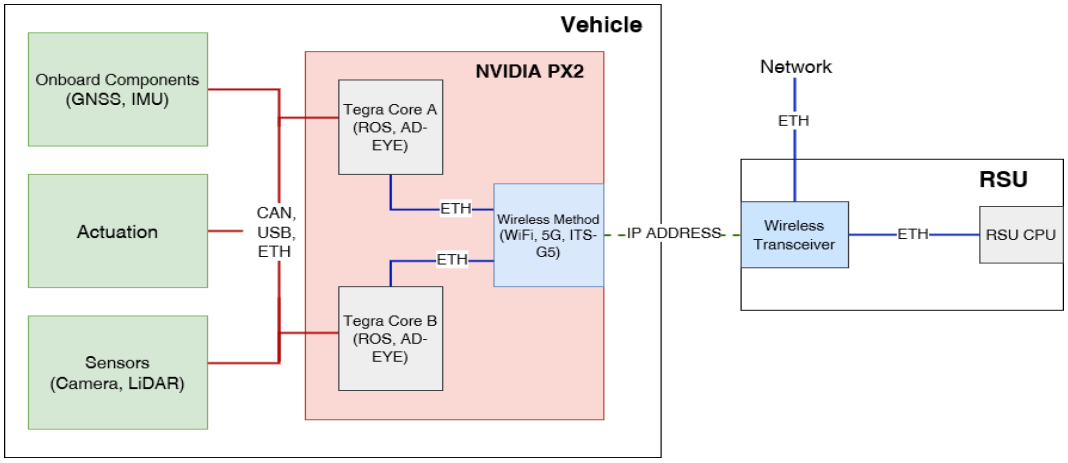


Figure 13: Schematic of the communication interface, created with [74].

2.9 Interfacing

As stated in 2.2.2 On Board Computer, the PX2 includes several relevant interfaces for an autonomous vehicle. The easiest approach would be to choose sensors with these interfaces. However, as per the initial stakeholder requirements, the choice of sensors will be decided and provided by the stakeholder. If the group is provided sensors that are not equipped with these interfaces, additional adapters and software might be required for the connection.

As stated in the initial stakeholder requirement 1.1, and further in 1.6, the PX2 shall be connected to the existing CAN bus of the vehicle. This could be done either by connecting to the OBD port of the vehicle, or directly to the wires for the CAN bus. The easiest approach would be to connect to the OBD port since it is accessible from inside the vehicle without any modifications. However, there is no guarantee that the OBD port includes a connection to the CAN bus, and if the connection is there, the information and control available might be limited. This means that to ensure access to the full CAN bus, the PX2 has to be connected directly through wiring to the vehicle's CAN bus. The wires are not easily accessible and hidden behind interior panels, hence an interior panel has to be removed or modified to give access to the vehicle's internal wires. Since the PX2's CAN interfaces use DB9 connectors [34], an adapter is required to connect directly to the wires.

Another required connection is the power connection to the PX2 and the sensors. The PX2 has a 12 V connector for power which could be connected directly to the vehicle's 12 V power system. The installation guide for the PX2 recommends an 80 A fuse in series and a diode in parallel to suppress transient voltages [75]. If the sensors are compatible with 12 V input then they could also be connected to the 12 V system. Otherwise, power regulators are required to step down or up the voltage to a level the sensors can handle. The sensors also requires appropriate fuses in series to ensure safe operations. The PX2 also requires a connection to an antenna or network card to handle the wireless communication with the RSU, since the PX2 only includes wired networking.

3 Methodology

This section describes the methodology applied for this project.

3.1 Project Divisions

The project was divided into different divisions to optimise the work process and ensure a structured work environment. The divisions are listed below:

- AD-EYE software
- Sensors
- CAN communication
- Mechanical mounting
- Power electronics

The primary responsibility for each division was allocated to different team members. However, the team had a good dynamic throughout the project and all team members contributed with their knowledge, advice, and aid in all the different divisions.

3.2 Work Process

In order to establish a clear goal and ensure that the team worked using similar implementation strategies, the V-model was chosen as the project management method. The V-model is suitable for small to medium-sized projects that take place during a limited time frame, which was the case for this project. Figure 14 shows the process of the V-model [76].

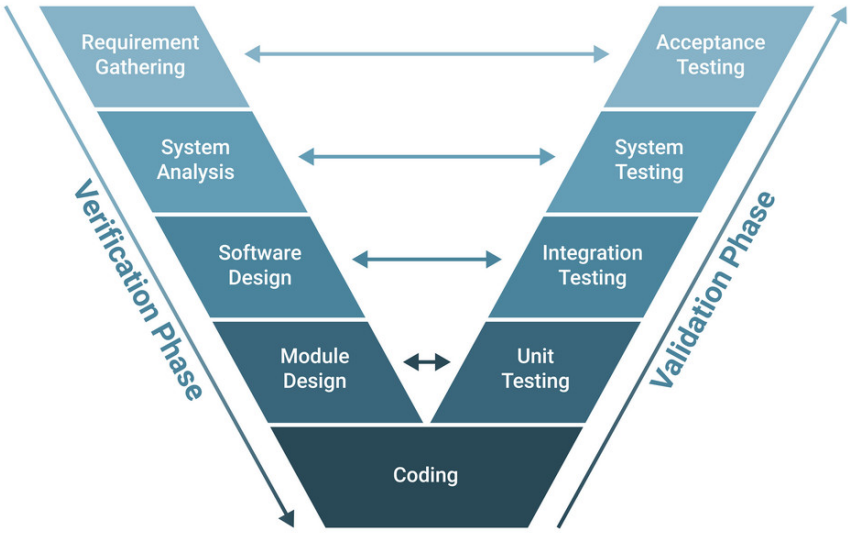


Figure 14: Illustration of the implementation process using the V-model [77].

Since the project had well-defined requirements and each phase of the project had well-defined milestones, the V-model method was a reliable choice of method. Continuous verification and validation throughout the project avoided errors from emerging in the late stages of the project.

To ensure that all group members were kept up to date regarding issues and the current state of the project, weekly group meetings were held. During the group meetings, all of the group members reported what had been accomplished, what problems were encountered, and what the next steps should be. This structure for the weekly group meetings was efficient for identifying problems early and sharing ideas for how to solve them. Different milestones for each of the divisions were set in order to work strategically towards accomplishing the stakeholder requirements. A GANTT chart was made for establishing the timeline for the project. Furthermore, the tool Trello was used for creating a Kanban board. The Kanban board is a great tool for visualising the milestones and getting a comprehensive overview of the state of the project.

Additionally, weekly coach meetings were held in order to get support and input from the coach. These meetings were a great opportunity for the group to discuss problems and maintain the right direction for the project. Frequent meetings with the stakeholder and product owner were also held throughout the autumn, which was useful for fast communication and quickly assigning action to problems that arose.

4 Implementation

This section covers the implementation processes in this project.

4.1 On Board Unit and AD-EYE Software

Initially, the PX2 was intended as the OBU for the vehicle. However, some problems arose, such as compatibility and memory issues, and the PX2 was eventually exchanged with a more powerful computer, namely an Alienware desktop computer. The OBU runs Ubuntu 16.04, ROS Kinetic and CUDA 10.0. It has the AD-EYE software installed as well as a virtual Windows environment for simulations in Prescan and Simulink.

The main objective of the AD-EYE software was to exchange the simulation in Prescan and Simulink with the real-world testing platform. The simulation was however used frequently to identify parameters and to verify and validate the work with real-world sensors. To know what type of data should be sent from and received by the real vehicle, the input and output messages to and from the simulated vehicle were identified, see Table 2.

Table 2: Input to and outputs from the simulation.

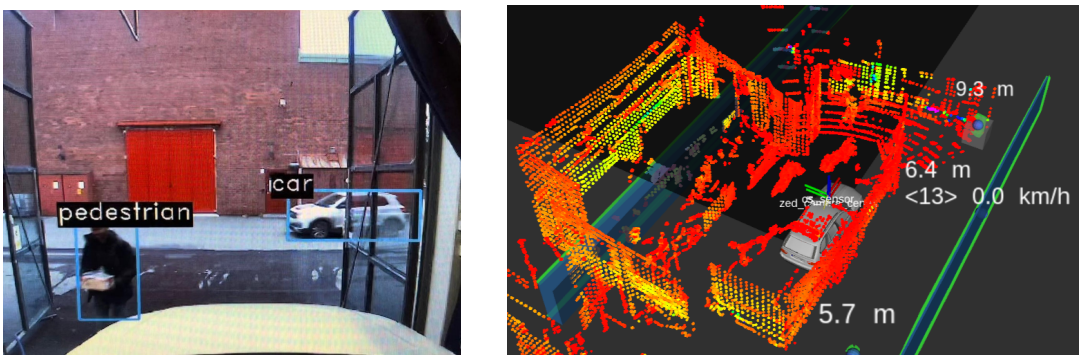
	Topic	**Message type**
1	/TwistS	geometry_msgs/TwistStamped
2	/camera_1/image_raw	sensor_msgs/Image
3	/camera_2/image_raw	sensor_msgs/Image
4	/tl/image_raw	sensor_msgs/Image
5	/radarDetections	std_msgs/Float32MultiArray
6	/points_raw_float32	std_msgs/Float32MultiArray
7	/ground_truth_pose	geometry_msgs/PoseStamped
8	/gnss_pose_simulink	geometry_msgs/PoseStamped
9	/initial_checks	std_msgs/Bool
10	/Features_state	std_msgs/Int32MultiArray
11	/fault	std_msgs/Bool
12	/activation_request	std_msgs/Bool
13	/current_velocity	geometry_msgs/TwistStamped
14	/adeye/goals	geometry_msgs/PoseStamped

The interesting topics for the work of this project were */TwistS* (topic 1) and the sensor related topics (topics 2-7). */TwistS* is the sole input to the vehicle from the AD-EYE software. The messages published on this topic contain the information that determines how the car should move. These messages have to be further interpreted and processed in order to send the correct control signals to the car via the CAN network, see section 4.5.

The topics related to the sensors had to be identified as these are the topics that the real sensors

should publish to. The proper nodes should be activated by the real sensors, i.e., the camera and the LiDAR, and the topics that the sensors publish their data to had to be remapped to the respective topic in the AD-EYE software, see sections 4.2 and 4.3. Not all of the sensors in the simulation were used on the real vehicle, so each node related to a sensor should be inactivated in the AD-EYE software so that no inputs are expected from these nodes. Attempts were made to inactivate these nodes in the AD-EYE software. However, due to the time limit and limited knowledge of the AD-EYE software, no final solution was found.

For the remaining topics, the messages from the simulation were used for the initialisation of AD-EYE. In order to start AD-EYE without the simulation, the messages sent to the different topics in the simulation were recorded with a rosbag. When running AD-EYE, the rosbag could then be played so that the required initialisation messages were received by the software. The topics related to the nodes that were replaced by real sensors, namely the camera and the LiDAR, could then be removed from the rosbag. In this way, the real data from the sensors could instead be integrated into AD-EYE. The resulting image from the ZED camera and point cloud from the Ouster LiDAR can be seen in Figure 15. The commands for running AD-EYE without the simulation can be found in Appendix A.



(a) ZED camera output

(b) Ouster LiDAR point cloud

Figure 15: Outputs from ZED camera and Ouster LiDAR with the car positioned in LabHallen at KTH Campus.

4.2 Camera

As part of the perception system of the vehicle, a ZED stereo camera from Stereolabs Inc. was used, see Figure 16. The stereo camera was chosen because it was already available to the group and fulfilled the essential performance requirements for its intended purpose. However, only the left hand side lens was used in this project.

Figure 16: ZED stereo camera [78].

Stereolabs provides Software Development Kit (SDK)'s for their cameras for easy software integration. A few initial attempts to install the appropriate SDK on the PX2 were made. However, these attempts failed as the ZED SDK is not compatible with NVIDIA PX2. The ZED SDK version 3.5 was instead successfully installed on the OBU. This SDK version was chosen as it supports Ubuntu 16 and CUDA 10.0. The SDK could then be used for an initial intrinsic calibration of the ZED camera.

To facilitate the use of ROS with the ZED camera, a ZED ROS wrapper provided by Stereolabs Inc. was installed on the OBU. The ROS topic that the ZED camera publishes its raw data to was then remapped to the corresponding topic in the AD-EYE software. In this way, the ZED camera was integrated with the AD-EYE software without creating an unnecessary amount of new ROS nodes. To further integrate the camera towards the vehicle, the coordinates of the camera relative to the rear axle of the car were written into the code for the ZED camera. The commands for the installations and for running the ZED camera as well as the code for the remapping and coordinate transform can be found in Appendix B.

4.3 LiDAR

The LiDAR used in the project was an *Ouster OS1-32*, see Figure 17. For the project, it was decided that the official driver for the LiDAR was the best option. However, the LiDAR was delivered with the latest firmware which only has support for ROS Melodic and Noetic [79]. Since the AD-EYE software runs on ROS Kinetic, the driver and it are not natively compatible. However, the driver includes a dockerfile which can be used to build and run a Docker container. The docker container runs a virtual environment of ROS melodic and the driver. This allows the use of the official driver, while still running ROS Kinetic as the master. A more detailed walk-through of the process is available in Appendix C. The communication between the LiDAR and OBU was handled by a router where both of them were connected via Ethernet.

Figure 17: Ouster OS1 LiDAR with heatsink cap [80].

For the project, it was not desirable to run the default settings of the driver. Therefore, the ROS launch file was replaced with a new one with new settings. This launch file also included remapping the relevant topics to be compatible with the AD-EYE software. This launch file is available in Appendix D.

When the ZED camera and Ouster LiDAR had been installed on the vehicle, an extrinsic calibration was made with Autoware's camera-LiDAR calibration package. This calibration converts the data from the two sensors to the same coordinate system, which is necessary for enabling fusion of the data [81]. The steps for the calibration can be found in Appendix E.

4.4 GNSS-RTK

The GNSS receiver used in the project was a *SimpleRTK2B Pro* with an integrated *ZED-F9P RTK* receiver. The *SimpleRTK2B Pro* is a GNSS receiver with Real-Time Kinematics (RTK) capabilities using the PointPerfect service, which gives it a centimeter accuracy [82]. The receiver outputs National Marine Electronics Association (NMEA) sentences via serial over USB. The sentences were read and converted into a position fix by an existing ROS package [83]. The position fix contains, among other things, longitude, latitude, and altitude. This was then read by Autoware to get the position of the vehicle in X, Y, and Z.

4.5 CAN Communication

The device used to read CAN frames from the vehicle was Vector VN1630A, which provides fast access to both CAN and CAN-FD networks. This device was connected to the CANoe software which is only available to be installed on Windows operating system. Therefore a laptop that runs the windows operating system was added to the setup. The CANoe is a software tool developed by Vector to analyse and test ECU networks. In this project, CANoe was used to receive, visualise, modify and send signals to the vehicle. In order for the CANoe to be able to translate CAN frames from the vehicle into visualised data, databases were obtained from Volvo Cars Corporation (VCC) and linked to the software. However, modifying and sending CAN frames to the vehicle required a

specific communication protocol to ensure safe and protected data exchange in order to make sure that safety-related data does not get affected by other signals. This protocol was also provided by VCC and installed on the CANoe which enabled full control over the vehicle via the software.

In order to establish communication with the ROS topics that are published by the AD-EYE software, C++ scripts needed to be programmed into the Linux operating system on the OBU to subscribe to the AD-EYE topics. Furthermore, there was a need to extract information about the CAN frames from the CANoe software and exchange this information with the OBU. For that, Fast Data Exchange (FDX) was used, which is a UDP-based protocol on the CAONoe software that enables fast, simple, and real-time data exchange with other systems via Ethernet communication [84]. The flow of the information in the system can be seen in Figure 18, and can be described as follows: A ROS listener node on the OBU subscribes to the ROS topic published by the AD-EYE software, the listener node receives messages and translates these messages into CAN frames which can then be sent via FDX to CANoe in order to control the vehicle.

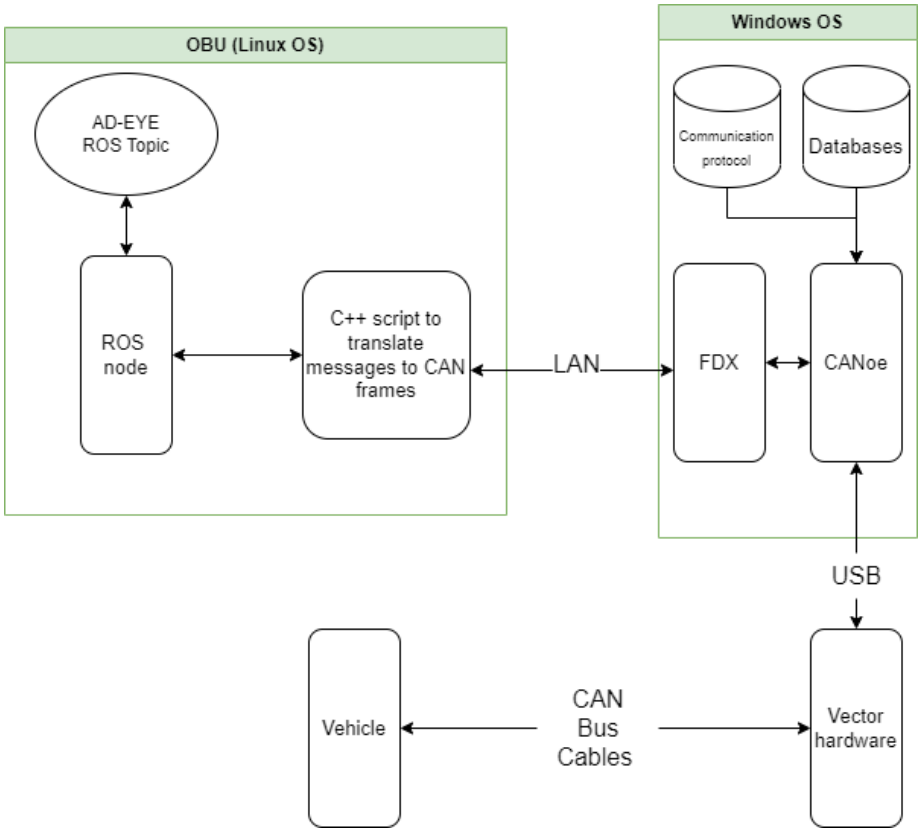


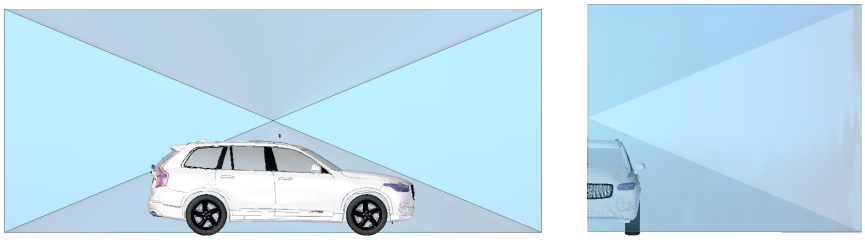
Figure 18: Flowchart of the CAN communication interface, created with [74].

4.6 Mechanical Mounting

In this section, the mechanical mounting of sensors and hardware inside the vehicle will be discussed.

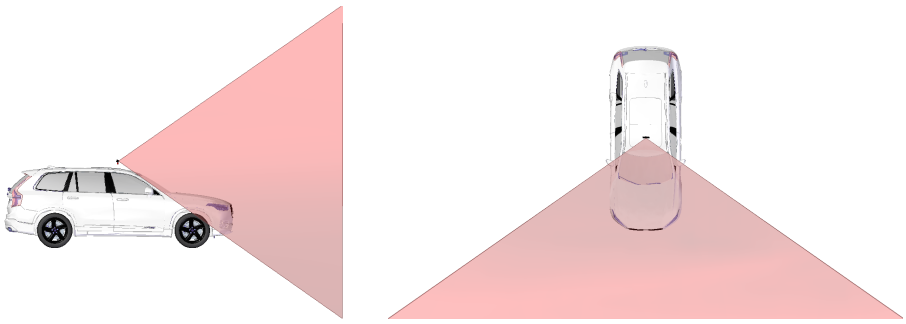
4.6.1 Sensor Mounting On The Roof

As mentioned in section 2.7.2 and seen in Figure 9, the LiDAR's optimal position is at the top of the vehicle as that provides the best possible coverage for the sensor. Only one camera was used in this project, and the position considered most suitable was the front-facing camera that can be observed in Figure 10. In order to adapt the concept design to the car used in the project, a Computer Aided Design (CAD) model of the car was used. With the aid of the CAD model, the Field Of View (FoV) of both the camera and the LiDAR could be reviewed in order to place them optimally. In Figure 19 a picture of the FoV for the LiDAR is shown from the side and from the front. In Figure 20 the FoV for the camera from the side and above can be seen.



(a) Side view (b) Front view

Figure 19: LiDAR FoV together with the car, generated in Solid Edge [85].



(a) Side view (b) Above view

Figure 20: Camera FoV together with the car, generated in Solid Edge [85].

In order to mount the sensors at the locations decided with aid of the CAD software, extruded aluminum bars were used to build a frame where the sensors could be fixed. The use of the above-

mentioned bars makes the solution modular and easy to build upon as these are standard products. The frame is shown in Figure 21. The frame also holds an antenna mounted outside of the FoV of the LiDAR in order to keep disturbances to a minimum.

The BOM for the entire rooftop mounting system can be found in Appendix F.

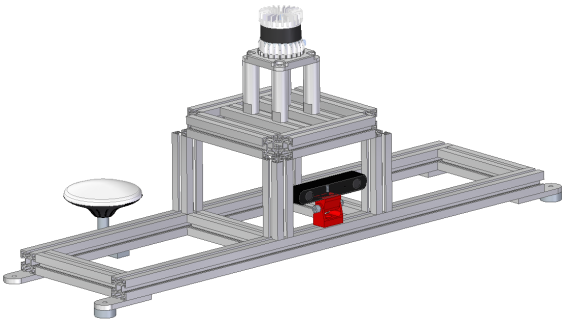


Figure 21: CAD model of the rooftop mounting system, made in Solid Edge [85].

4.6.2 Hardware mounting inside the car

All hardware installed in the car was securely mounted to avoid any movement of the components and possible damage to them while driving. The components are stored in the boot of the car. In order to secure their position, they were screwed to a wooden board, which was then fastened to the car. In this way, there is no damage to the car while fastening the components. Some hardware did not have mounting holes, for these, custom-built brackets were 3D modeled and printed. The result can be seen in Figure 26.

4.7 Power Electronics

According to stakeholder requirement 1.2.2, the electrical system should be able to supply two computers, two cameras, two LiDARs, one GNSS, and one router. In order to dimension the electrical system some assumptions about their power consumption had to be made, they are listed below, per item:

- Computer - 500W
- Camera - 50W
- LiDAR - 50W
- GNSS - 30W
- Router - 30W

When possible, components should be powered directly via Direct Current (DC) as that would create fewer sources of ElectroMagnetic Interference (EMI) than an approach where their standard Alternating Current (AC)/DC adapter is used. However, some components are not possible to power directly via DC such as the computer as this would require substantial modifications of the computer. Therefore, a pure sine wave converter was used in order to convert the 12V supplied by the car into 220V 50Hz. The LiDAR requires 24V which is why a DC/DC power supply was used. To protect the LiDAR from possible power surges, an insulated power supply was used.

Further, the stakeholder requirement 1.2.2 state that there should be a button to disconnect the components from the car. This is solved by the use of a control box that connects to relays that can connect or disconnect the electrical components. As the DC/AC power supply will draw a lot of current, a contactor had to be used instead of a relay.

The resulting circuit of the whole electrical system can be seen in Figure 22.

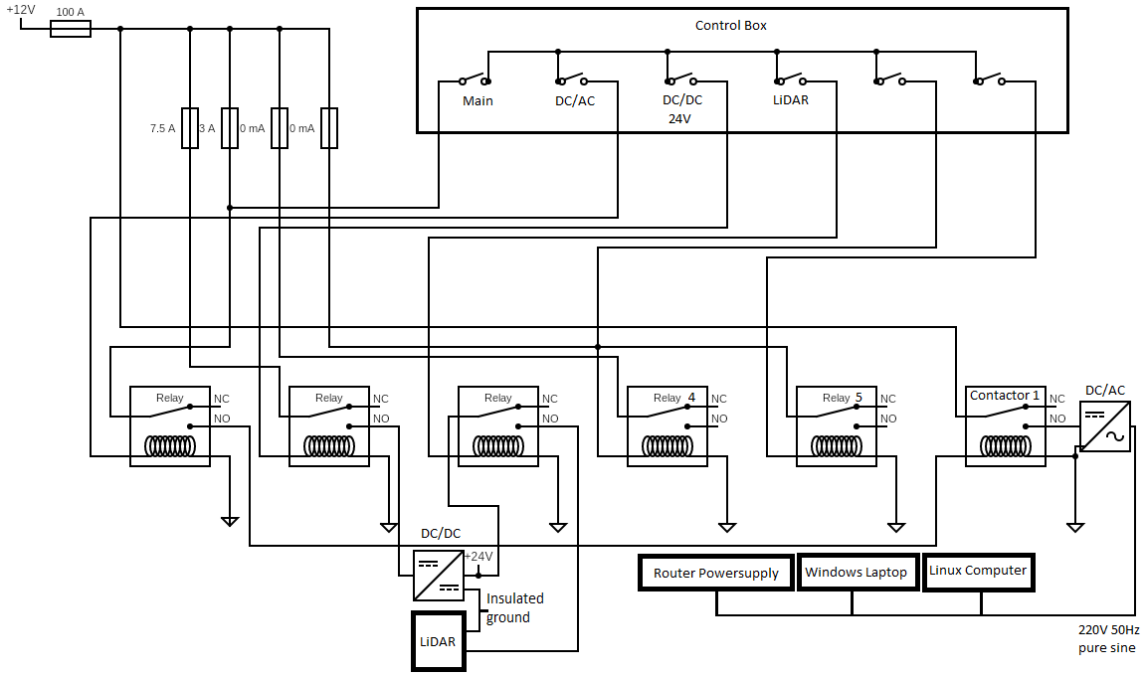


Figure 22: Electrical schematic of the power electronics.

4.7.1 Cable sizing

In order to decide how big cables are needed to support the system the total amount of current running through them must be decided. According to the assumptions made above, the total amount of power that the system will draw during full load is 1300W. Further, assuming that there will be losses in power supply converters, cables, etc, an additional 10% is added to the total

amount of power. The current running through the 12V side of the system can then be calculated via $P = UI$ and equals 120A. According to [86] a 50 mm^2 copper cable can support 119A if the cable is in a thermally enclosed environment, which is the worst-case scenario. If the cable is in free air, it is rated for 219A. It was therefore safe to assume that the cable will be enough for powering the electronics added to the car.

The complete BOM for the electrical system can be found in Appendix G.

5 Verification and Validation

5.1 On Board Unit and AD-EYE Software

A stress test was run on the OBU which it handled without problems. The AD-EYE software was validated to work with the sensors, but due to the time limit the system was never validated to work with CAN. Hence, HIL communication was not verified nor validated.

5.2 Camera

To ensure that the camera was properly integrated into AD-EYE, the camera and the simulation were run together. The ZED camera showed in the rqt graph and published its information on the intended topic. Moreover, the integration could be validated by looking at the image view from the camera of the simulated vehicle in RViz. The result was the view from the real camera, and the object detection worked properly, see Figure 15 (a). As the car never drove autonomously, HIL validation was never done. This validation should be done in further work with this project.

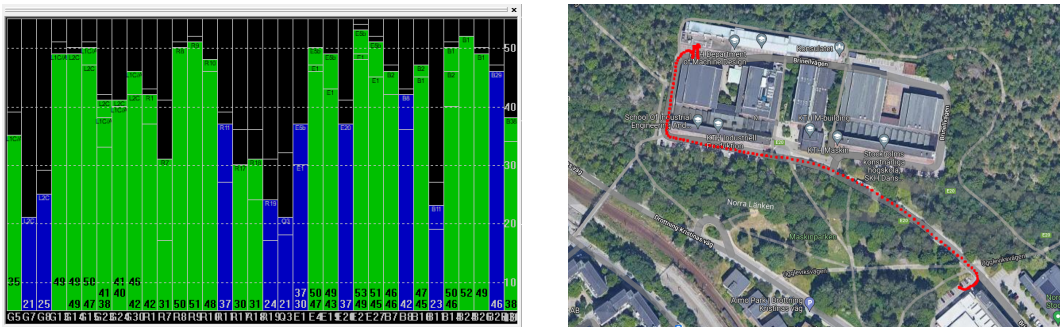
5.3 LiDAR

To validate the integration of the LiDAR towards the AD-EYE software, first, the Docker container for the LiDAR driver was started. Then the rqt graph was checked to ensure that the expected topics were published. After that, the container was stopped and the simulation of AD-EYE was started, but without the LiDAR. Then the container was started again. This resulted in the point cloud from the LiDAR showing up in RViz. Obstacle detection also started, and obstacles based on the point cloud were marked as can be seen in Figure 15 (b). If the container then was stopped, the point cloud stopped updating and the obstacle detection stopped until the container was started again.

5.4 GNSS-RTK

To ensure the proper operation of the GNSS receiver, first, the hardware was tested. This was done by mounting the receiver on the car and connecting it to a Windows laptop running u-center, an evaluation software for u-blox receivers [87]. When starting u-center, it was first ensured that a good GNSS connection existed, in this case, $\sim$ 20 satellites over 40 dB, see Figure 23 (a). After that, the reported longitude, latitude, and altitude were checked against the actual position. This gave an error of less than 1 meter. Then the status of the RTK service was verified, by checking the *Fix Mode* in u-center. When the RTK service has a connection, the *Fix Mode* should include either "FLOAT" or "FIXED" which it did when the car was outside. When driving the car inside, the RTK connection disappeared within a few seconds. This was also verified by a status light on the PCB of the receiver labeled "NO RTK", which turned red when the car was taken inside. Before

the car was taken inside, it was manually driven on a small test drive. Then the recorded positions were superimposed on a map to verify that the position of the car was correctly reported. The map can be seen in Figure 23 (b).



(a) Connected satellites

(b) Map of the test drive

Figure 23: The connected satellites and the test drive.

When the receiver was proven to work correctly, the ROS nodes used had to be verified. First, with the GNSS receiver running and connected to the OBU, the node for reading the NMEA sentences was started. Then the published topic was checked that it was correct and contained the correct longitude, latitude, and altitude. Then the Autoware node that converts the coordinates to X, Y, and Z was started and its output was also confirmed.

5.5 CAN Communication

To confirm that the CAN system communication was working properly, each step of the system was validated separately before testing the whole communication flow. First, a validation of the connection between CANoe software and the vehicle was done, and CAN frames were sent and received successfully after using the databases and the communication protocol provided by VCC. This phase of the communication was sufficiently robust as no disturbances were noticed.

After that, the connection between the OBU and the CANoe software was validated. For that, a description file was needed to define the content of the messages and the database's variables that are being exchanged between CANoe and the OBU. Moreover, a program was established on the OBU to package the bytes into the right CAN frame data field in order for the messages to be readable by the CANoe. This phase was validated by sending a message that changes the initial values of some variables on the database. The message was sent from the C++ script on the OBU to the CANoe and as an answer to that, a message with the new values of the variables was received on the OBU side. During this phase, there were some noticeable delays when receiving information about the updated variables. This could be due to issues on the code programmed on the OBU side. Furthermore, in order to change the right variable value, it is essential to ensure that the message size sent from the OBU side is identical to the message size expected by the CANoe.

The last phase of the validation process was to ensure robust communication throughout the whole system. This means that a message published by a specific ROS topic should be successfully transferred through the OBU and translated to a CAN frame that was readable by the CANoe software and could be used to control the vehicle. For that, a ROS topic that publishes messages from a gamepad was created. The gamepad was then used to control some of the vehicle's functions, such as lights and steering wheel.

An important aspect of the communication system is safety and robustness, which was an essential part of the stakeholder's requirements. This was taken into consideration during the validation process, and tests were done to make sure that the driver can shut down the communication system and manually take control of the vehicle. As some information regarding the CAN system are not allowed to be published, the emergency shutdown for the CAN system is described in more detail in a separate Appendix.

5.6 Mechanical Mounting

The structure on the roof is fastened to the car with four 8.8-quality M10 screws. Each screw is rated for 47 kN which should be sufficient. While validating the GNSS the car was run outside for a short distance. No issues with the mechanical installation were observed.

5.7 Power Electronics

To validate that the electrical system works as intended a stress test was conducted where all components were powered via the car battery. During this test it was also verified that each component could be turned on and off from the control box. The total current through the system was measured with a clamp meter in order to verify that the correct assumptions had been made when designing the system. The maximum current drawn from the battery when the OBU, one LiDAR, one camera, one GNSS, one router, one Vector VN1630A and one laptop was connected was 31A which is well below the maximum rating of the system. This leaves room for installing more components to the system.

To check for bad connections, crimps, or other anomalies, a thermal camera was used. In Figure 24, thermal imaging of the system while running the stress test can be seen. During the stress test all of the components were powered via the car's 12V system. To strain the system the AD-EYE software was run together with a stress test program called S-TUI [88]. The hot spot in the bottom left of the figure is the contactor which is used to connect/disconnect the DC/AC converter. A lot of current goes through this contactor which is why it appears hot. The temperature for the contactor stabilised at 38 degrees Celsius which is well within the rating. The hotspots to the right in the picture are the air vents from the computers.

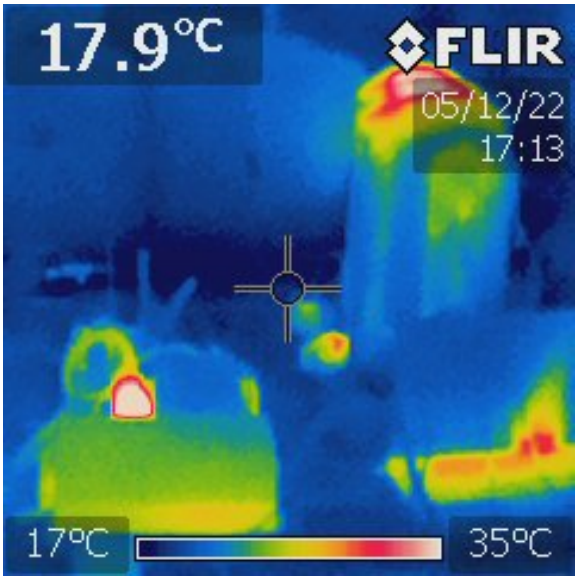


Figure 24: Thermal imaging of the running system.

6 Results

This section presents the results achieved in the project.

6.1 On Board Unit and AD-EYE Software

The OBU was able to run the code necessary for running the sensors and the AD-EYE software. Hence, the AD-EYE software was successfully run on the OBU and worked with the integrated sensors, which fulfills stakeholder requirement 1.3. Additionally, it was able to communicate with the CANoe software through an Ethernet connection, hence fulfilling stakeholder requirement 1.1. However, the OBU did not manage to run the code necessary for running the sensors, the AD-EYE software and establish communication with the CANoe software simultaneously.

6.2 Camera

In this project, one camera was implemented on the front of the car. The camera is mounted on the vehicle and powered by the OBU, which fulfills stakeholder requirements 1.2.1 and 1.2.2 regarding the camera. The integration worked well with the AD-EYE software, which fulfills stakeholder requirement 1.3 regarding the camera. However, further work has to be done to evaluate the integration to the vehicle and the overall function of the camera during autonomous driving in HIL mode before stakeholder requirement 1.4 can be fully fulfilled. Moreover, the camera was successfully calibrated, both intrinsically and extrinsically with the LiDAR, and the process was documented, which fulfills requirement 1.2.4.

6.3 LiDAR

The Ouster LiDAR used in this project is securely mounted on the roof of the car. It is powered by an isolated DC/DC power supply from the car's 12 V system. The communication between the LiDAR and the OBU was established through Local Area Network (LAN), and thus, the stakeholder requirement 1.2 is fulfilled for the LiDAR. It has also been shown that the LiDAR correctly transmits the required data to the AD-EYE software where it is processed, and thus stakeholder requirement 1.3 is fulfilled in terms of the LiDAR.

6.4 GNSS-RTK

Like the other sensors, the GNSS receiver is mounted to the structure on the roof of the car. The receiver is connected with USB to the OBU which handles both power and data, and therefore stakeholder requirement 1.2 is fulfilled for the GNSS-RTK receiver. However, the integration towards the AD-EYE software has not been verified and validated and therefore stakeholder requirement 1.3 is not fulfilled regarding GNSS.

6.5 CAN Communication

While testing the communication system, three different signals were sent and received between the vehicle and the ROS node that was created in the OBU for testing purposes. These signals are: controlling lights, shifting gears, and controlling the steering wheel direction. When sending signals at a high rate, it was noticed that the CAN system froze and did not respond to further signals. Furthermore, the system worked well when only the test ROS node was published. However, when several nodes were started for the other components of the system, there were some delays noticed in the CAN system. Such problems were solved by restarting the system. Moreover, even when encountering such problems the driver was still able to shut down the external communication system and gain full control of the vehicle when needed.

The emergency shutdown process is included in a separate Appendix, together with detailed information about the CANoe setup, and data exchange process. This Appendix was handed to stakeholders fulfilling requirement 1.1.1.

6.6 Mechanical Mounting

The finished sensor structure can be seen in Figure 25. Figure 26 shows all the hardware fastened in the boot of the car.

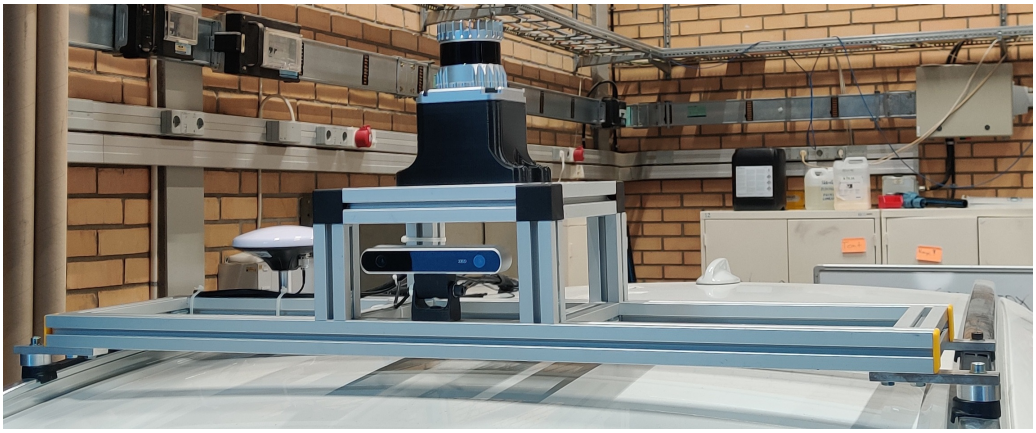


Figure 25: Sensor mounting structure on top of the car.

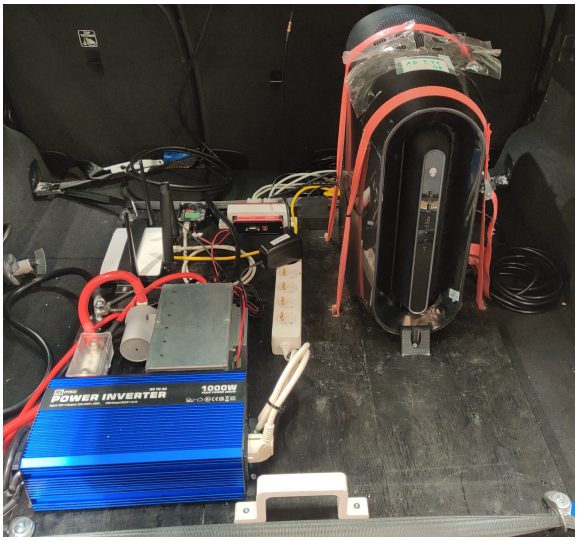
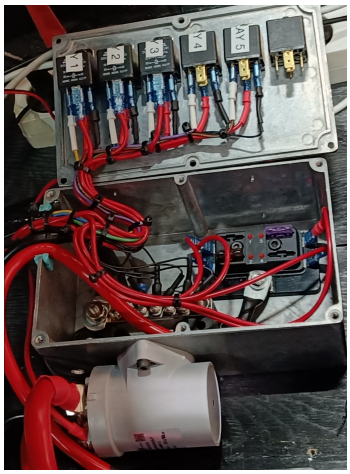


Figure 26: Hardware mounting in the boot of the car.

6.7 Power Electronics

Figure 27 shows the final result of the relay box and the control box that connects and disconnects the different components from the car's 12V system. It is possible to turn on / off every individual component and the main switch can be used to turn off all of the components.



(a) Relay box

(b) Control box

Figure 27: Finished result of the control box and relay box.

6.8 Stakeholder Requirements

Table 3 lists all stakeholder requirements and whether they were fulfilled, partly fulfilled or not fulfilled.

Table 3: Fulfilment status of the Stakeholder Requirements.

Requirement	**Fulfilled**	**Partly fulfilled**	**Not fulfilled**
1.1.1	X		
1.1.2		X	
1.2.1	X		
1.2.2	X		
1.2.3	X		
1.2.4	X		
1.2.5	X		
1.3		X	
1.4			X
2.1	X		
3.1			X
3.2			X
3.3			X

7 Discussion and Conclusion

The following section presents a discussion and conclusions drawn based on this project.

7.1 On Board Unit and AD-EYE Software

The solution with the Alienware computer as OBU worked well. The implementation of the AD-EYE software also worked well when the subsystems of the car were run separately, but further work would have to be done to integrate the CAN communication to the AD-EYE software. The whole system was never run in HIL mode, an more time would have to be spent on the OBU and AD-EYE software to further improve the implementation.

7.2 Camera

The final solution included only one forward facing camera on the car. The SOTA showed that more sensors will be needed in order to achieve a perception system of the car that makes it adequate for autonomous driving. The main reason as to why only one camera was used was because of the time constraint. Hence, the camera implementation can be improved in further work with this project. Furthermore, the installed camera is more advanced than what is needed for its intention in this project. The camera is used as a mono-camera even though it is a stereo camera, which means that either a simpler camera can be used or the current camera can perform additional tasks. However, the older version of Autoware that is used on the OBU does not support stereo-vision, so in order to utilise this the version of Autoware would have to be upgraded.

7.3 LiDAR

The workaround with the version incompatibilities was achieved with a Docker container. While this adds some overhead, it was decided as the best solution. It will also give some future proofing, as long as the expected LiDAR data is in the same format, the Docker container will work and future incompatibilities can be avoided.

As explained in subsubsection 2.7.2, mounting the LiDAR on the roof will create blind areas around the car. This was mitigated by mounting the LiDAR higher up on the mounting structure. It was also decided to shift the LiDAR forward to decrease the blind area in front of the car at the cost of increasing the blind area behind it as was shown in Figure 19. This decision was based upon the assumption that in normal driving, it is most important to have good coverage in front of the vehicle.

7.4 GNSS-RTK

As stated in the results, the GNSS receiver and the RTK service works, and the position is published to ROS. However, the interfacing towards AD-EYE is yet to be proven. This was mainly due to the time constraint since it would have required either editing of the existing maps used by Autoware, or new ones would have to be created.

7.5 CAN Communication

Robustness, flexibility, and reliability were the main focus while designing and implementing the communication protocol that will be used to connect the vehicle to the AD-EYE software. A great amount of time and resources were allocated to develop the solution to translate CAN messages while being transferred between different softwares and operating systems. This resulted in a flexible and adjustable communication system where new groups of messages that control other functions in the vehicle can be easily added to the system. However, when overloading the system, some signals were lost meaning that there are some limitations to be considered when further developing the system. Furthermore, the system is based on LAN communication and wired connection, which showed sufficient results in terms of latency and coverage. Moreover, the results show that the driver can gain full control over the vehicle at any time, meaning that the safety requirement is satisfied.

However, due to time constraints, the tests were mainly done on a ROS node that replicates the functionalities of the AD-EYE software. This means that more tests are needed to ensure the reliability of the communication system while connected to the AD-EYE software. Another drawback with the current communication system is that priorities are not introduced, which needs to be considered in further development.

7.6 Mechanical Mounting

For the current sensor setup, the mechanical mounting fulfills requirement 1.2. However, if more sensors are to be added in the future they are now limited to being mounted in the front part of the roof. An alternative could be to expand the mounting system with roof racks that could be mounted on the rails of the car. This could make it possible to mount sensors towards the back of the car, for example, a back facing LiDAR or camera. For adding more sensors, the current CAD model could be used for evaluating the FoV in order to optimise the placement. It is easy to add new sensors to the existing structure with the use of t-nuts in the aluminum frame, sensors can be arbitrarily placed on the frame. By using aluminum as material in the frame, the structure is resistant to oxidation, making it suitable for use in the potentially harsh environment on the road. Aluminum also offers a good rigidity-to-weight ratio. The GNSS antenna came with a magnetic mount, however, it was decided to mount the GNSS antenna on the aluminum frame as that is more rigid than the magnet and it is possible to keep the antenna away from the FoV of the LiDAR.

7.7 Power Electronics

The concept of the control box and relay setup came about with the need for a safety switch. The intention for this safety switch was to disconnect the components added to the car for this project in case of failure. The components added can theoretically run indefinitely as long as the car's engine is running, since the alternator of the car is continually charging the battery. The alternator of the car is specified to deliver a maximum of 180A, which is well above the maximum of 31A that the components added in this project draw. However, it is unknown how much current the car itself draws during normal circumstances as that varies depending on what features are used such as electrical motors, windscreen wipers and seat heaters. It is therefore hard to determine how much current can be drawn as a maximum from the car's electrical system.

All cables in the relay and control box follow the common practice of cable colouring. Red cables are used for the positive terminal and black is the ground. Signal wires are colour coded to ease the troubleshooting of the cable harness that connects the relay box and the control box.

The concept of the control box and relay setup came about with the need for a safety switch. The intention for this safety switch was to disconnect the components added to the car for this project in case of failure.

8 Future Work

This section considers suggestions for further improvements that could be made to the testing platform.

8.1 On Board Unit and AD-EYE Software

Suggestively, an OBU intended for autonomous driving should be used, e.g., the NVIDIA DRIVE Hyperion Autonomous Vehicle Development Platform [89]. Choosing a faster and purpose built OBU could reduce the time required for starting the simulation environment and be more sufficient in terms of computational power. Additionally, by choosing a purpose built solution, the amount of EMI could be reduced in the testing platform since it can be powered directly by the 12V system of the vehicle. As mentioned in subsection 4.1, not all of the sensors available in the simulation environment were used on the testing platform. Therefore, each node related to a sensor that exists in the simulation environment, but not on the testing platform, should be inactivated in the AD-EYE software. Hence, a future improvement of the system would be to inactivate these nodes successfully or add the same amount of sensors used in the simulation environment to the vehicle.

8.2 Camera

Currently, the ZED camera is used as a mono-camera although it supports stereo vision. In further work, the full capacity of the camera could be utilised and the stereo vision could be used for achieving depth vision. However, the version of Autoware that is used in this project does not support stereo vision, and would hence also have to be upgraded. Furthermore, additional cameras should be integrated to the system facing sideways and backwards in order to accomplish a complete FoV. In future work the camera should also be validated to work in HIL mode.

8.3 LiDAR

As previously mentioned, the LiDAR has some blind spots around the vehicle. While they still exist, this project has tried to minimise them with the limitation of one LiDAR. If more LiDARs were to be used, they can be placed to cover a larger area around the vehicle and reduce the blind spots. One interesting feature of the Ouster LiDAR not explored in this project is its embedded near-infrared camera which outputs a 360° grayscale image. The implementation of this would however require support in AD-EYE and Autoware.

8.4 GNSS-RTK

The next step for the GNSS receiver is to integrate it towards AD-EYE. To do this, first, the point cloud map and vector map used for localisation and navigation have to use the true coordinate

system i.e., the same as the GNSS uses. This can either be achieved by updating the existing maps in AD-EYE or creating new ones as suggested by optional stakeholder requirement 4.3.

8.5 CAN Communication

During tests, the implemented solution for the communication system provided sufficient signal transmitting, but there are some improvements that can be added to the solution in order to provide a faster, more reliable, and more extensive communication system.

As mentioned previously, three signals were used to prove the reliability of the introduced solution. Unfortunately, that will not be enough to implement the solution on the AD-EYE software, meaning that more signals need to be added both on the CANoe side and on the Linux side. Furthermore, priorities and threading should be implemented in order for the communication system to manage more critical situations where several signals are transmitted at the same time. After implementing that, the communication system can be connected to the ROS topic that is published by AD-EYE where signals can be sent and received directly from AD-EYE to the vehicle.

Furthermore, the CANoe software is a good tool to visualise and transmit data, but using it introduces extra costs because of the need for the Windows operating system. To solve that, it is recommended to implement an End to End communication protocol on the OBU which will enable direct communication with the vehicle. However, for that to be implemented, more information about the communication protocol used by VCC is needed.

8.6 Mechanical Mounting

The mounting brackets for the LiDAR and the camera are now 3D printed with PolyLactic Acid (PLA) plastics. This plastic has a glass transition temperature of around 50-80 degrees Celsius [90] which could pose a problem on sunny days. An alternative could be to print the same models in another material such as Acrylonitrile Butadiene Styrene (ABS), which has a glass transition temperature of 105 degrees Celsius, or mill them from aluminum.

8.7 Power Electronics

The stakeholder requirements state that one additional embedded computer along with a camera and a LiDAR should be able to be added to the system. As the stress test showed, one additional, similar computer should be able to run on the same DC/AC converter. As for the camera and LiDAR, it is not specified which camera or LiDAR will be used. Therefore, it is unknown how much current and which voltage levels these components require. However, the relay box is prepared with two unused relays that are controllable from the control box. It is possible to connect new components to these relays directly if they run off 12V, and the only thing required is to insert a fuse of a suitable size for the new component in the fusebox. If the new component needs a different

voltage level than 12V a converter needs to be added in between, similar to how the LiDAR is connected, see the schematic in Figure 22.

8.8 Overall system

The initial stakeholder requirements were more extensive and included additional areas, e.g., V2X communication, 5G, and implementation of the RSU as presented in section 2. However, as the stakeholder requirements were condensed, these areas were excluded from the final implementation and not explored further in this project. Nonetheless, due to the contribution of this project, several initial requirements which are important for the concept of enabling autonomous driving features properly are now more feasible and can be considered in the future development of this platform.

9 Bibliography

References

[1] S. of Automotive Engineers. "Sae levels of driving automation™ refined for clarity and international audience." (2021-05-03), [Online]. Available: https://www.sae.org/blog/sae-j3016-update (visited on 2022-05-22).

[2] J. K. Thomas Knoll, *Photoshop*, version 22.1.0, 2021. [Online]. Available: https://www.adobe.com/es/products/photoshop.html.

[3] J. Cusack. "Self-driving vehicles are steadily becoming a reality despite the many hurdles still to be overcome – and they could change our world in some unexpected ways." (2021-10-30), [Online]. Available: https://www.bbc.com/future/article/20211126-how-driverless-cars-will-change-our-world (visited on 2022-05-23).

[4] F. M. Company. "Volvo pilot assist." (2022), [Online]. Available: https://www.volvocars.com/se/support/topics/anvanda-din-bil/bra-att-veta/pilot-assist-och-korfaltsassistans (visited on 2022-05-06).

[5] B. AG. "Overview of driver assistance systems." (2021-09-16), [Online]. Available: https://www.bmw.com/en/innovation/the-main-driver-assistance-systems.html (visited on 2022-05-12).

[6] "Ford co-pilot360tm technology." (), [Online]. Available: https://www.ford.com/technology/driver-assist-technology/.

[7] Tesla. "Tesla autopilot." (2022), [Online]. Available: https://www.tesla.com/sv_SE/autopilot (visited on 2022-05-19).

[8] D. Coldewey. "Tesla vaunts creation of 'the best chip in the world' for self-driving." (2019-04-22), [Online]. Available: https://techcrunch.com/2019/04/22/tesla-vaunts-creation-of-the-best-chip-in-the-world-for-self-driving/ (visited on 2022-05-18).

[9] M. McFarland. "Tesla owners say they are wowed – and alarmed – by 'full self-driving'." (2022-03-16), [Online]. Available: https://edition.cnn.com/2021/11/03/cars/tesla-full-self-driving-fsd/index.html (visited on 2022-05-17).

[10] T. Reid. "Example of ford research vehicle." (2019-09), [Online]. Available: https://www.researchgate.net/figure/Example-of-Fords-research-autonomous-vehicle-platform-sensors-used-in-localization-and_fig5_336515529 (visited on 2022-05-15).

[11] K. Kirkpatrick, "Still waiting for self-driving cars; why is it taking so long for autonomous vehicles to hit the road?" *Communications of the ACM*, vol. 65, 4 2022. DOI: DOI:10.1145/3516517.

[12] R. Hussain and S. Zeadally, "Autonomous cars: Research results, issues, and future challenges," *IEEE Communications Surveys Tutorials*, vol. 21, no. 2, pp. 1275–1313, 2019. DOI: 10.1109/COMST.2018.2869360.

[13] D. Ghosh. "What are robotaxis?" (2021-12-17), [Online]. Available: https://htschool.hindustantimes.com/editorsdesk/news-trends/robotaxis-are-coming-up-but-what-are-they/ (visited on 2022-05-21).

[14] M. Herger. "Autox opens driverless robotaxis for the public." (2021-01-27), [Online]. Available: https://thelastdriverlicenseholder.com/2021/01/27/autox-opens-driverless-robotaxis-for-the-public/ (visited on 2022-05-23).

[15] R. Bellan. "Waymo opens driverless robotaxi service to san franscisco employees." (2022-03-20), [Online]. Available: https://techcrunch.com/2022/03/30/waymo-opens-driverless-robotaxi-service-to-san-francisco-employees/ (visited on 2022-05-05).

[16] D. Michael, M. Kleanthous, M. Savva, S. Christodoulou, M. Pampaka, and A. Gregoriades, "Impact of immersion and realism in driving simulator studies," *Int. J. Interdiscip. Telecommun. Netw.*, vol. 6, no. 1, pp. 10–25, 2014-01, ISSN: 1941-8663.

[17] AstaZero. "Test site." (2022), [Online]. Available: https://www.astazero.com/en/tracks-facilities/test-site/ (visited on 2022-05-21).

[18] Millbrook. "Test bed for connected and autonomous vehicles — cav — millbrook." (2022), [Online]. Available: https://www.millbrook.co.uk/services/connected-and-autonomous-vehicle-testing/ (visited on 2022-05-21).

[19] G. H. Ruffo. "Musk says giga berlin will be testbed for new tesla technologies." (2020-10-07), [Online]. Available: https://insideevs.com/news/447771/musk-giga-belin-testbed-new-tesla-technologies/ (visited on 2022-05-07).

[20] P. Biswas. "Iit-h sets up testbed on autonomous navigation (tihan) for unmanned aerial vehicles." (2020-12-29), [Online]. Available: https://timesofindia.indiatimes.com/home/education/news/iit-h-sets-up-testbed-on-autonomous-navigation-tihan-for-unmanned-aerial-vehicles/articleshow/80011899.cms (visited on 2022-05-02).

[21] J. He-Rim. "Hyundai motor to launch self-driving tech test bed." (2021-10-12), [Online]. Available: http://www.koreaherald.com/view.php?ud=20211012000799 (visited on 2022-05-09).

[22] A. Salter. "7 universities that are pushing the boundaries of autonomous driving." (2021-10-18), [Online]. Available: https://www.2025ad.com/7-universities-that-are-pushing-the-boundaries-of-autonomous-driving-2021 (visited on 2022-05-16).

[23] C. in Automation - CiA. "Classical controller area network (can)." (2022), [Online]. Available: https://www.can-cia.org/can-knowledge/can/classical-can/ (visited on 2022-04-10).

[24] C. in Automation - CiA. "Can: From physical layer to application layer and beyond." (2022), [Online]. Available: https://www.can-cia.org/can-knowledge/ (visited on 2022-04-10).

[25] M. S. U. Alam, "Securing vehicle electronic control unit (ecu) communications and stored data," pp. 9–10, 2019.

[26] C. Electronics. "Can bus explained - a simple intro [2021]." (2021-11-01), [Online]. Available: https://www.csselectronics.com/pages/can-bus-simple-intro-tutorial (visited on 2022-04-10).

[27] K. H. Johansson, M. Törngren, and L. Nielsen, "Vehicle applications of controller area network," *Springer*, vol. 6, no. 6, pp. 741–765, 2005. DOI: 10.1007/0-8176-4404-0_32.

[28] C. Electronics. "Obd2 explained - a simple intro [2021]." (2021-11-01), [Online]. Available: https://www.csselectronics.com/pages/obd2-explained-simple-intro (visited on 2022-04-10).

[29] R. De Andrade, K. Hodel, J. Fransisco Justo, A. Laganá, and M. Mauro Santos, "Analytical and experimental performance evaluations of can-fd bus," *IEEE Access*, vol. 6, no. 3, pp. 21 287–21 289, 2018. DOI: 10.1109/ACCESS.2018.2826522.

[30] C. in Automation - CiA. "Can fd - the basic idea." (2022), [Online]. Available: https://www.can-cia.org/can-knowledge/can/can-fd/ (visited on 2022-04-04).

[31] N. Corporation. "Nvidia drive sdk." (2022), [Online]. Available: https://developer.nvidia.com/drive/drive-sdk/ (visited on 2022-04-09).

[32] N. Corporation. "Nvidia drive px, Scalable ai supercomputer for autonomous driving." (2022), [Online]. Available: https://www.nvidia.com/content/nvidiaGDC/sg/en_SG/self-driving-cars/drive-px/ (visited on 2022-04-08).

[33] R. Smith. "Nvidia announces drive px 2 - pascal power for self-driving cars." (2016-01-05), [Online]. Available: https://www.anandtech.com/show/9903/nvidia-announces-drive-px-2-pascal-power-for-selfdriving-cars/ (visited on 2022-04-08).

[34] N. Automotive, *Drive px 2 hands-on, Drive px 2 sdk 4.1.8.0*, Rev. 20170725, 2017-07-25.

[35] N. Corporation. "Introducing the new nvidia drive px 2, For autocruise driving and hd mapping." (2016-10-13), [Online]. Available: http://web.archive.org/web/20161013023854/http://www.nvidia.com/object/drive-px.html/ (visited on 2022-04-09).

[36] N. Corporation. "Solutions for self-driving cars & autonomous vehicles." (2022), [Online]. Available: https://www.nvidia.com/en-us/self-driving-cars/ (visited on 2022-04-08).

[37] N. Corporation. "Software for self-driving cars." (2022), [Online]. Available: https://www.nvidia.com/en-us/self-driving-cars/drive-platform/software/ (visited on 2022-04-08).

[38] H. Ullah, N. G. Nair, A. Moore, C. Nugent, P. Muschamp, and M. Cuevas, "5g communication: An overview of vehicle-to-everything, drones, and healthcare use-cases," *IEEE Access*, vol. 7, pp. 37 251–37 268, 2019.

[39] A. I. Al-Alawi, "Wifi technology: Future market challenges and opportunities," *Journal of computer science*, vol. 2, no. 1, pp. 13–18, 2006.

[40] E. J. Oughton, W. Lehr, K. Katsaros, I. Selinis, D. Bubley, and J. Kusuma, "Revisiting wireless internet connectivity: 5g vs wi-fi 6," *Telecommunications Policy*, vol. 45, no. 5, p. 102 127, 2021, ISSN: 0308-5961. DOI: https://doi.org/10.1016/j.telpol.2021.102127. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S030859612100032X.

[41] I. Corporation. "What is wi-fi?" (), [Online]. Available: https://www.intel.com/content/www/us/en/gaming/resources/wifi-6.html (visited on 2022-04-07).

[42] E. Notes. "What is cellular communications: Mobile technology?" (), [Online]. Available: https://www.electronics-notes.com/articles/connectivity/cellular-mobile-phone/what-is-cellular-communications.php (visited on 2022-04-07).

[43] R.-s. Ltd. "What's the difference between high-band, mid-band and low-band 5g?" (), [Online]. Available: https://www.router-switch.com/faq/what-s-the-difference-between-high-band-mid-band-and-low-band-5g.html (visited on 2022-04-07).

[44] A. K. Gulia, *A simulation study on the performance comparison of the v2x communication systems: Its-g5 and c-v2x*, 2020. [Online]. Available: http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1422828&dswid=-2617 (visited on 2022-05-21).

[45] RENISHAW. "Optical encoders and lidar scanning." (), [Online]. Available: https://www.renishaw.fr/fr/optical-encoders-and-lidar-scanning--39244 (visited on 2022-05-02).

[46] "Point clouds for beginners." (2022), [Online]. Available: https://geoslam.com/point-clouds/ (visited on 2022-05-03).

[47] A. Dubey, "Stereo vision—facing the challenges and seeing the opportunities for adas applications," Texas Instruments, 2020. [Online]. Available: https://www.ti.com/lit/wp/spry300a/spry300a.pdf?ts=1650270194572&ref_url=https%253A%252F%252Fwww.google.com%252F#:~:text=If%20we%20have%20a%20single,called%20a%20stereo%20vision%20system (visited on 2021-05-01).

[48] C. Wang, X. Wang, H. Hu, Y. Liang, and G. Shen1, "On the application of cameras used in autonomous vehicles," 2022-04-12. [Online]. Available: https://link.springer.com/article/10.1007/s11831-022-09741-8 (visited on 2022-05-21).

[49] M. M. Vázquez. "Radar transceivers: A key component for adas & autonomous driving." (2021), [Online]. Available: https://www.renesas.com/us/en/blogs/radar-transceivers-key-component-adas-autonomous-driving-blog-1-why-do-we-need-radar (visited on 2022-05-21).

[50] J. J. Asiag. "How ultrasonic sensor data is powering automotive iot." (2021-04-17), [Online]. Available: https://otonomo.io/blog/ultrasonic-data-automotive-iot/#:~:text=Ultrasonic%20sensors%20mimic%20echolocation%20used,immediate%%20of%20a%20vehicle. (visited on 2022-05-21).

[51] X. Lu, X. Xin, L. Yishi, *et al.*, "Imu-based automated vehicle slip angle and attitude estimation aided by vehicle dynamics," *Sensors*, vol. 19, no. 8, 2019, ISSN: 1424-8220. DOI: 10.3390/s19081930. [Online]. Available: https://www.mdpi.com/1424-8220/19/8/1930.

[52] S. Sukkarieh, E. Nebot, and H. Durrant-Whyte, "A high integrity imu/gps navigation loop for autonomous land vehicle applications," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 3, pp. 572–578, 1999. DOI: 10.1109/70.768189.

[53] A. D. Team. "Sensor position offset compensation." (2021), [Online]. Available: https://ardupilot.org/plane/docs/common-sensor-offset-compensation.html (visited on 2022-04-04).

[54] Lantmäteriet. "Gps och andra gnss." (2022), [Online]. Available: https://www.lantmateriet.se/sv/Kartor-och-geografisk-information/gps-geodesi-och-swepos/GPS-och-satellitpositionering/GPS-och-andra-GNSS/ (visited on 2022-04-04).

[55] EUSPA. "What is gnss?" (2021-12-04), [Online]. Available: https://www.euspa.europa.eu/european-space/eu-space-programme/what-gnss (visited on 2022-04-04).

[56] Lantmäteriet. "Gps och satellitpositionering." (2022), [Online]. Available: https://www.lantmateriet.se/sv/Kartor-och-geografisk-information/gps-geodesi-och-swepos/GPS-och-satellitpositionering/ (visited on 2022-04-04).

[57] T. Lombardo. "Waymo rolls out driverless taxis in phoenix." (2020-10-27), [Online]. Available: https://www.engineering.com/story/waymo-rolls-out-driverless-taxis-in-phoenix (visited on 2022-05-20).

[58] S. Jeyachandran. "Introducing the 5th-generation waymo driver: Informed by experience, designed for scale, engineered to tackle more environments." (2020-03-04), [Online]. Available: https://blog.waymo.com/2020/03/introducing-5th-generation-waymo-driver.html (visited on 2022-05-20).

[59] "Autoware." (2021), [Online]. Available: https://www.autoware.org (visited on 2022-04-25).

[60] "Simcenter prescan." (2022), [Online]. Available: https://www.plm.automation.siemens.com/global/en/products/simcenter/prescan.html (visited on 2022-04-25).

[61] "Simulink." (2022), [Online]. Available: https://se.mathworks.com/products/simulink.html (visited on 2022-04-25).

[62] "Why ros?" (2021), [Online]. Available: https://www.ros.org/blog/why-ros/ (visited on 2022-04-25).

[63] N. Mohan. "Ad-eye." (2022), [Online]. Available: https://www.adeye.se/home (visited on 2022-05-02).

[64] V. Hanefors, C. Jensen, J. Jungåker, *et al.*, "Ad-eye, Implementation of an autonomous driving platform," Kungliga Tekniska Högskolan, 2020. [Online]. Available: https://www.kth.se/social/files/605e01487fb52d433333764d/ad-eye-final-report.pdf (visited on 2022-04-29).

[65] "Kth research concept vehicle." (2021-01-07), [Online]. Available: https://www.itrl.kth.se/about-us/labs/rcv-1.476469 (visited on 2022-04-30).

[66] "Research concept vehicle (rcv) – research platform for future vehicle concepts." (2014-06-10), [Online]. Available: https://www.itrl.kth.se/polopoly_fs/1.487591.1600688751!/RCV_Mikael_Nybacka_v3_red.pdf (visited on 2022-04-30).

[67] S. Ackels, P. Benavidez, and M. Jamshidi, "A survey of modern roadside unit deployment research," eng, in *World Automation Congress Proceedings*, vol. 2021-, 2021, pp. 137–141, ISBN: 1685241115.

[68] U. D. of Transportation. "Standards training." (2021), [Online]. Available: https://www.pcb.its.dot.gov/StandardsTraining/mod57/ppt/m57ppt.pdf (visited on 2022-05-21).

[69] M. Akkila, E. Haltorp, N. Määttä, *et al.*, "Ad-eye, Automated driving & its testbed," Kungliga Tekniska Högskolan, 2021. [Online]. Available: https://www.kth.se/social/files/62430685fa6b7fbe26328fb2/ad-eye-finalreport.pdf (visited on 2021-05-01).

[70] Felix, "Sensor set design patterns for autonomous vehicles," 2019-01-25. [Online]. Available: https://autonomous-driving.org/2019/01/25/positioning-sensors-for-autonomous-vehicles/ (visited on 2022-05-21).

[71] "Inertial measurement units for localisation and navigation of autonomous vehicles," 2021-07-21. [Online]. Available: https://sensible4.fi/technology/articles/imu-and-autonomous-vehicles/ (visited on 2022-05-21).

[72] L. F. S. Ltd. "Magnetic lidar mount." (2022), [Online]. Available: https://levelfivesupplies.com/product/magnetic-lidar-mount/ (visited on 2022-05-21).

[73] V. Hanefors, C. Jensen, J. Jungåker, *et al.*, "Ad-eye, State of the art report," Kungliga Tekniska Högskolan, 2020. [Online]. Available: https://www.kth.se/social/files/605cbdd0206fc2b8f16daa2f/ad-eye-springterm-report.pdf (visited on 2022-05-16).

[74] J. Ltd, *Diagrams.net/draw.io*, version 18.1.1. [Online]. Available: https://app.diagrams.net.

[75] N. Corporation. "Nvidia drive px 2 autochauffeur mechanical and installation guide." (2018-04-27), [Online]. Available: https://developer.nvidia.com/driveworks/files/drive-px2-mechanical-installation (visited on 2022-05-16).

[76] S. T. Help. "What is stlc v-model?" (), [Online]. Available: https://www.softwaretestinghelp.com/what-is-stlc-v-model/ (visited on 2022-12-16).

[77] V. Stock. "V model software development." (), [Online]. Available: https://www.vectorstock.com/royalty-free-vector/v-model-software-development-methodology-scheme-vector-34705748 (visited on 2022-12-16).

[78] S. Inc. "The camera that senses space and motion." (), [Online]. Available: http://sudude.com/index-5.html (visited on 2022-12-06).

[79] Ouster. "Github - ouster-lidar/ouster-ros: Official ros drivers for ouster sensors." (2022-11-18), [Online]. Available: https://github.com/ouster-lidar/ouster-ros (visited on 2022-11-20).

[80] I. Ouster. "Os 1 mid-range digital lidar sensor." (2022), [Online]. Available: https://ouster.com/products/scanning-lidar/os1-sensor/ (visited on 2022-12-16).

[81] MathWorks. "What is lidar-camera calibration?" (), [Online]. Available: https://se.mathworks.com/help/lidar/ug/lidar-camera-calibration.html (visited on 2022-12-06).

[82] Ardusimple. "Rtk-ssr receiver with pointperfect (internet or l-band)." (2022), [Online]. Available: https://www.ardusimple.com/product/rtk-ssr-receiver-with-pointperfect-internet-or-l-band/ (visited on 2022-11-20).

[83] E. Perko. "Nmea navsat driver." (2020-06-13), [Online]. Available: http://wiki.ros.org/nmea_navsat_driver (visited on 2022-11-20).

[84] H. Alexander. "Fast data exchange with canoe." (2015-08-07), [Online]. Available: https://www.vector.com/se/en/download/fast-data-exchange-with-canoe/ (visited on 2022-12-17).

[85] Siemens. "Solid edge." (), [Online]. Available: https://solidedge.siemens.com/en/ (visited on 2022-12-07).

[86] L. S.G.H. Electric Wire & Cable Co. "Appendix 2 - current carrying and voltage drop tables for pvc insulated and xlpe insulated cables." (), [Online]. Available: https://www.sghcable.com/pdf/cable_usage/App-2E%20V2.pdf (visited on 2022-12-08).

[87] u-blox. "U-center gnss evaluation software for windows." (2022), [Online]. Available: https://www.u-blox.com/en/product/u-center (visited on 2022-11-24).

[88] amanusk. "The stress terminal ui: S-tui." (), [Online]. Available: https://github.com/amanusk/s-tui (visited on 2022-12-17).

[89] N. Corporation. "Nvidia drive hyperion autonomous vehicle development platform." (), [Online]. Available: https://developer.nvidia.com/drive/hyperion (visited on 2022-12-17).

[90] S. Vouyiouka and C. Papaspyrides, "4.34 - mechanistic aspects of solid-state polycondensation," in *Polymer Science: A Comprehensive Reference*, K. Matyjaszewski and M. Möller, Eds., Amsterdam: Elsevier, 2012, pp. 857–874, ISBN: 978-0-08-087862-1. DOI: https://doi.org/10.1016/B978-0-444-53349-4.00126-6. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780444533494001266.

A Commands for running AD-EYE without the simulation

To run AD-EYE without the simulation, enter the following commands in separate terminals:

1. `roslaunch zed_wrapper zed.launch`
2. `docker run --rm -it --network host ouster-ros:hk`
3. `rosrun tf static_transform_publisher 1.25 0 2.22 0 0 0 /base_link /os_sensor 30`
4. `roslaunch adeye manager_simulation.launch`
5. `./play_rosbag`

Comments

1. Start the ZED camera node.
2. Start the docker for the Ouster LiDAR.
3. Create a static transform between the Ouster LiDAR (os_sensor) and the rear axle of the car (base_link). “1.25 0 2.22 0 0 0” are the relative coordinates and orientations between the Ouster LiDAR and the car in metres.
4. Start the AD-EYE manager.
5. Start playing the rosbag. The “play_rosbag” file contains the topics that should be included as well as some flags for speeding up the process. When the initialization process is finished the rosbag can be stopped.

B Commands and code related to the implementation of the ZED camera on the OBU

Installing the ZED SDK

1. Check the Ubuntu version and CUDA version installed on the OBU. At the time of writing the Ubuntu version is 16.04 and the CUDA version is 10.0.
2. Download the appropriate SDK version from Stereo Labs website, see https://www.stereolabs.com/developers/release/. Look in their legacy folders for older versions compatible with the Ubuntu and CUDA versions. For this project, ZED SDK 3.5 was used, see https://www.stereolabs.com/developers/release/3.5/.
3. Once the download is complete, you can follow the installation process of the ZED SDK available here: https://www.stereolabs.com/docs/installation/linux/.

Installing the ZED wrapper

1. Download the appropriate ZED wrapper version from GitHub. To be compatible with the SDK version, an older version of the ZED wrapper must be installed. The versions can be found on https://github.com/stereolabs/zed-ros-wrapper/releases?page=1. For this project, ZED wrapper version 3.5 was downloaded, see https://github.com/stereolabs/zed-ros-wrapper/tree/v3.5.
2. Once the download is complete, you can follow the installation process of the ZED wrapper available here: https://www.stereolabs.com/docs/ros/. For this project, the workspace created was called "catkin_ws_ZED".

Source workspace

Edit the bachrc file to source the ZED workspace each time a new terminal is opened:

1. `$ gedit ~/.bashrc`
2. Add: `source /home/adeye/catkin_ws_ZED/devel/setup.bash --extend`

Remapping

The left camera of the ZED camera was remapped to publish its data to the appropriate topic in the AD-EYE software with the following steps:

1. Navigate to the launch file located in:
   `catkin_ws_ZED/src/zed-ros-wrapper-3.5/zed_wrapper/launch`
2. Add the following to the launch file `zed.launch`:

```
<remap from="/zed/zed_node/left_raw/image_raw_color" to="/camera_1/image_raw"/>
```

Coordinate transform

For the static transform between the centre of the ZED camera and the base frame (base_link, located in the rear axle of the vehicle), the highlighted values were modified in `zed.launch`:

```
<!-- EDIT: default value "0.0" of cam_pos_x and cam_pos_z has been changed to correspond
to the position of the camera relative the Volvo XC90 -->
    <arg name="cam_pos_x" default="1.38" /> <!-- Position respect to base_link -->
    <arg name="cam_pos_y" default="0.0"  /> <!-- Position respect to base_link -->
    <arg name="cam_pos_z" default="1.86" /> <!-- Position respect to base_link -->
    <arg name="cam_roll"  default="0.0"  /> <!-- Orientation respect to base_link -->
    <arg name="cam_pitch" default="0.0"  /> <!-- Orientation respect to base_link -->
    <arg name="cam_yaw"   default="0.0"  /> <!-- Orientation respect to base_link -->
```

Starting the ZED camera node

Connect the camera to a USB3.0 port on the computer and type the following in a terminal:

```
$ roslaunch zed_wrapper zed.launch
```

If the camera is not detected, pull out the cable and reconnect it to the computer.

C Creating a Docker image for the LiDAR and running it

Connect the Ouster LiDAR and verify that it works:

1. Connect the LiDAR to the Interface Box.
2. Connect the Interface Box with an ethernet cable to the same router as the computer.
3. Connect power to the LiDAR.
4. Enter `os-xxxxxxxxxxx.local`, where `xxxxxxxxxxx` is the serial number of the LiDAR, into the URL field of a web browser, if a webpage shows up it works.
5. Since the Docker container we will create can have difficulties to resolve the hostname, a static IP is preferably reserved for it in the router. As an example, `192.168.1.10` can be chosen.

Install docker:

```
$ sudo apt-get install docker.io
```

Clone the Ouster-ros repository in an empty directory:

```
$ git clone --recurse-submodules https://github.com/ouster-lidar/ouster-ros.git
```

Build the Docker image:

Since we will use some launch parameters, and do a remapping, an additional launch file can be created and included in the Docker image. This launch file is available at the end of this document.

1. Place the launch file in the root of the directory the repository was cloned into.
2. In the directory, open the file named "Dockerfile" with a text editor.
3. Go to the end of the file and on the line
   `&& roslaunch ouster_ros sensor.launch \"$@\" \`
   change `sensor.launch` to the name of the new launch file.
4. Build the Docker image with the following command:
   `$ docker build . -t ouster-ros:hk`

The Docker container can now be started with two optional launch parameters:

- `sensor_hostname` – The hostname of the LiDAR, preferably dotted IP address.
- `lidar_mode` – The resolution and rate of the LiDAR. Possible values are:
  - 512x10
  - 512x20
  - 1024x10
  - 1024x20
  - 2048x10
  - 4096x5

To start the Docker container with default parameters:

```
$ docker run --rm -it --network host ouster-ros:hk
```

D Launch file to be included in the Docker image

Launchfile:
Below is the launch file that should be placed in the directory before the Docker image is built. The default value of the argument **sensor_hostname** should be the IP reserved for the LiDAR, in this case **192.168.1.10**.

```
<launch>
    <arg name="sensor_hostname" default="192.168.1.10" doc="hostname or IP
in dotted decimal form of the sensor"/>

    <arg name="lidar_mode" default="1024x10" doc="resolution and rate;
possible values: {
        512x10,
        512x20,
        1024x10,
        1024x20,
        2048x10,
        4096x5
        }"/>

    <remap from="/ouster/points" to="/points_raw"/>

    <include file="$(find ouster_ros)/launch/sensor.launch">
        <arg name="sensor_hostname" value="$(arg sensor_hostname)"/>
        <arg name="lidar_mode" value="$(arg lidar_mode)"/>
        <arg name="timestamp_mode" value="TIME_FROM_ROS_TIME"/>
        <arg name="viz" value="false"/>
    </include>
</launch>
```

E Calibration of camera and LiDAR

This document includes the steps for the intrinsic calibration of the camera and the extrinsic calibration of the camera and LiDAR.

Autoware includes packages for camera calibration and camera-lidar-calibration, located at:
`~/AD-EYE_Core/autoware.ai/src/autoware/utilities/autoware_camera_lidar_calibrator`.

For the calibrations, the "*A3 - 40mm squares - 9x6 verticies, 10x7 squares*" chequerboard was printed from:
https://markhedleyjones.com/projects/calibration-checkerboard-collection.

Startup
The positions of the sensors relative to the rear axle of the car was measured (in metres):

- Camera centre: 1.38 0 1.86 0 0 0
- LiDAR centre: 1.25 0 2.22 0 0 0

To start the sensors and add their relative positions as static transforms, write the following in separate terminals:

1. `roslaunch zed_wrapper zed.launch`
2. `rosrun tf static_transform_publisher 1.38 0 1.86 0 0 0 /base_link /zed_camera_center 30`
3. `docker run --rm -it --network host ouster-ros:hk`
4. `rosrun tf static_transform_publisher 1.25 0 2.22 0 0 0 /base_link /os_sensor 30`

Intrinsic calibration
The intrinsic calibration of the camera was started with the following command:

```
rosrun autoware_camera_lidar_calibrator cameracalibrator.py --square 0.039 --size 9x6
image:=/camera_1/image_raw
```

The checkerboard was moved around in the field of view of the camera until enough samples had been taken (when all the bars in the camera view had become green).

This generated a .yaml file with the intrinsic camera parameters:

- `20221104_1435_autoware_camera_calibration.yaml`

Extrinsic calibration
First, the workspace had to be sourced:

```
source AD-EYE_Core/autoware.ai/build/autoware_camera_lidar_calibrator/devel/setup.bash
```

Then the extrinsic calibration of the camera and LiDAR was started with the following command:

```
roslaunch autoware_camera_lidar_calibrator camera_lidar_calibration.launch
intrinsics_file:=/home/adeye/20221104_1435_autoware_camera_calibration.yaml
image_src:=/camera_1/image_raw
```

This opened an image viewer. Rviz was configured by following these steps:
Open Rviz and add the pointclud from the Ouster as well as the TF tree:

- Add → By topic → /points_raw (below "/ouster") → PointCloud2
- Add → By display type → rviz → TF

The camera view and point cloud from the LiDAR was then visible in the image view and in Rviz.

Calibration steps:

- Move the checkerboard around in the field of view of both the camera and LiDAR.
- Click on a point in the image, and then click on the same point in the LiDAR point cloud by using the Publish Point tool in Rviz.
- Repeat with a minimum of 9 different points.

This generated a .yaml file with the extrinsic camera-LiDAR parameters:

- `20221118_123552_autoware_lidar_camera_calibration.yaml`

F BOM for sensor mounting on top of the vehicle

BOM for the sensormounting on top of the vehicle

Alucon	Profil 40 x 40, Lätt. T-Spår 8		001-204-3	2	SEK	675,60	SEK	1 351,20
Alucon	Hörnfäste 40x40		006-200	4	SEK	110,95	SEK	443,80
Alucon	Spårmutter M8, T-spår 8		004-203	40	SEK	11,41	SEK	456,40
Alucon	Universalfäste 40, T-spår 8		008-202	25	SEK	91,81	SEK	2 295,25
Custom	LiDAR Mount			1	-			
Custom	ZED Camera mount			1	-		-	
					Total		SEK	4 546,65

G BOM for Electrical system

Shop	Item Name	Item Number	Qty	Unit price	Total price
Control Electronics					
RS Components	Durakool Plug In Automotive Relay, 12V dc Coil Voltage, 40A Switching Current, SPDT	915-6638	3 SEK	25.38 SEK	76.14
RS Components	TE Connectivity Panel Mount Automotive Relay, 12V dc Coil Voltage, 30A Switching Current, SPDT	782-6950	3 SEK	39.46 SEK	118.38
Elfa	631NH - Toggle Switch ON-OFF SPST, Apem	135-22-340	6 SEK	54.57 SEK	327.42
RS Components	RS PRO Pure Sine Wave 1000W Power Inverter, 12V Input, 230V Output	179-3330	1 SEK	2,982.10 SEK	2,982.10
Elfa	TMDC 40-2415 - DC/DC Converter 9 ... 36V 24V 1.67A 40W, Traco Power	300-08-558	1 SEK	738.00 SEK	738.00
			Cost		SEK 4,242.04
Cabling & Accessories					
Farnell	Wire, Tri Rated, Per Metre, PVC, Red, 16 AWG, 1.5 mm^2	2528177	10 SEK	9.68 SEK	96.80
Farnell	Wire, Tri Rated, Per Metre, PVC, Black, 16 AWG, 1.5 mm^2	2528084	10 SEK	9.68 SEK	96.80
Elfa	3630A - Ring Terminal Blue 4.3 mm PVC Pack of 100 pieces, Vogt	148-20-898	1 SEK	114.00 SEK	114.00
RS Components	RS PRO Blue Insulated Female Spade Connector, Receptacle, 0.8 x 6.35mm Tab Size, 1.5mm^2 to 2.5mm^2	178-8296	100 SEK	1.37 SEK	136.70
Elfa	10-4 - Non-Insulated Ring Terminal 4.3mm, M4, 10mm^2, Pack of 100 pieces, NEMIQ	301-80-519	1 SEK	359.72 SEK	359.72
Farnell	Fuseholder, Panel Mount, 32V, 30A, Blade Fuse, Quick Connect, 4 Positions	2901452	1 SEK	180.88 SEK	180.88
RS Components	RS PRO Black Nylon Releasable Cable Tie, 250mm x 4.5 mm	811-1802	1 SEK	154.53 SEK	154.53
RS Components	RS PRO Black Nylon Cable Tie, 155mm x 3.6 mm	209-8131	100 SEK	1.14 SEK	114.00
Biltema	Battery Cable, red, 50 mm^2	24-564	1 SEK	749.00 SEK	749.00
Biltema	Battery Cable, black, 50 mm^2	24-565	1 SEK	749.00 SEK	749.00
Biltema	Battery Cable, red, 16 mm^2	24-560	1 SEK	259.00 SEK	259.00
Biltema	Battery Cable, black, 16 mm^2	24-561	1 SEK	259.00 SEK	259.00
Biltema	Terminal Block, Universal	25-233	1 SEK	189.00 SEK	189.00
Biltema	Round cable shoes, 16 mm^2, M10, 10 pcs.	24-581	2 SEK	79.90 SEK	159.80
Biltema	Round cable shoes, 50 mm^2, M10, 10 pcs.	24-584	2 SEK	125.00 SEK	250.00
Biltema	ANL Fuse, 100A	25-2310	1 SEK	84.90 SEK	84.90
Biltema	Fuse Holder	25-2319	1 SEK	169.00 SEK	169.00
Biltema	Battery Cable Lugs, 70 mm^2, 2-pack	35-1659	1 SEK	79.90 SEK	79.90
Biltema	120 flat-pin fuses Midi	35-463	1 SEK	84.90 SEK	84.90
			Cost		SEK 4,286.93
Mounting Accessories					
RS Components	RS PRO 4m Clawhook Webbing Strap, 45mm Wide, 600kg Breaking Strain	729-3117	2 SEK	136.50 SEK	273.00
RS Components	Protecta Stainless Steel Anchor Point, 15kN	161-8391	4 SEK	97.35 SEK	389.40
Custom	CANalyzer Mounting Bracket	3D printed, CAD file available in CAD-folder	1 SEK	- SEK	-
Custom	LiDAR Interface Box Mounting Bracket	3D printed, CAD file available in CAD-folder	1 SEK	- SEK	-
Custom	GNSS Interface Box Mounting Bracket	3D printed, CAD file available in CAD-folder	1 SEK	- SEK	-
Custom	Router Mounting Bracket	3D printed, CAD file available in CAD-folder	1 SEK	- SEK	-
Custom	Handle	3D printed, CAD file available in CAD-folder	1 SEK	- SEK	-
Custom	Alienware Computer Mounting Support	3D printed, CAD file available in CAD-folder	2 SEK	- SEK	-
			Cost		SEK 662.40