Automated Wheel Loader

MF2059 - KTH Mechatronics Advanced Course, 2021
Final Report

TESS ANTONSSON
LUCAS JACOBSSON FALCON
DANIEL GRÖNÅS
HOMING KONG
FREDRIK MAZUR
JOACHIM OTTOSSON
MARCUS RUDERER
JAN SIWEK
CHIEH-JU WU

In collaboration with Volvo CE
Supervisor: Tobias Vahlne

Abstract

The modern plastic recycling industry utilises heavy machinery, creating a potentially hazardous work environment. Due to the influence of human operators, the overall process can also be time inefficient and incur high labour costs. An autonomous machine is a potential solution to this dilemma.

This project is a part of the KTH Mechatronics HK capstone project, in collaboration with Volvo Construction Equipment, and focuses on investigating the feasibility of applying a fully autonomous vehicle in the recycling industry. The objective is to implement a semi-autonomous electric wheel loader concept for a plastic bale handling scenario in a scaled prototype and environment.

The report introduces the motivation of the project, state of the art analysis, and the concept design for the mechanical, electrical, and software systems. The implementation of the final design, the verification and validation of the prototype, its results, a discussion, and proposed future work are also described. The wheel loader prototype fulfils all the stakeholder requirements in a semi-autonomous testing environment. It is also able to identify multiple plastic bale models through image recognition and perform the bale transfer to a desired destination.

Acknowledgements

We would like to thank Volvo CE for this opportunity, especially Joakim Unnebäck and Bobbie Frank, for their professional inputs and feedback. We would also like to thank our team coach Tobias Vahlne for his guidance and support during the project development. Björn Möller, Vicki Derbyshire, and the rest of the teaching team have also been instrumental to the project's success.

Contents

List of Figures

List of Tables

List of Abbreviations

3D Three-Dimensional, page 17

ADC Analog-to-Digital Converter, page 17

ANN Artificial Neural Network, page 20

CAD Computer Aided Design, page 11

CE Construction Equipment, page 1

CNC Computer Numerical Control, page 87

CNN Convolutional Neural Network, page 20

CPU Central Processing Unit, page 55

DAC Digital-to-Analog Converter, page 17

DC Direct Current, page 16

DMA Direct Memory Access, page 51

DoF Degree of Freedom, page 40

FoV Field of View, page 18

GPIO General Purpose Input/Output, page 49

GPU Graphics Processing Unit, page 3

HAL Hardware Abstraction Layer, page 51

IDE Integrated Development Environment, page 59

IMU Inertial Measurement Unit, page 18

IR Infrared, page 18

LECA Light Expanding Clay Aggregate, page 39

LiDAR Light Detection and Ranging, page 18

OPCODE Operation Code, page 54

OSI Open System Interconnection, page 54

PCB Printed Circuit Board, page 50

PLA Polylactic Acid, page 86

PMSM Permanent Magnet Synchronous Motor, page 17

PWM Pulse Width Modulation, page 56

RAM Random Access Memory, page 49

RC Remote Controlled, page 27

RGB Red, Green, Blue, page 18

RGB-D Red, Green, Blue colour model with Depth, page 18

ROS Robot Operating System, page 18

SLAM Simultaneous Localisation and Mapping, page 18

SR Stakeholder Requirement, page 4

ToF Time of Flight, page 18

TR Technical Requirement, page 4

UART Universal Asynchronous Receiver-Transmitter, page 17

VO Visual Odometry, page 19

WD Wheel Drive, page 32

YOLO You Only Look Once, page 22

Chapter 1

Introduction

In this chapter, the background of the project, its scope, and the stakeholder and technical requirements are presented.

1.1 Background

In 2019, over 1.3 million tons of plastic waste was introduced to the Swedish market, by means of either importation or production. Unfortunately, only a small percentage of this is recycled [1]. When the plastic waste is reprocessed, the plastics are compressed into bales and transported in this form to unloading stations. Because of the heavy machinery that is utilised in this handling process, the work environment is potentially hazardous. Due to the influence of human operators, the overall process can also be time inefficient and incur high labour costs [2].

Today, Volvo CE (Construction Equipment) manages plastic waste handling primarily with manual, fuel-driven wheel loaders. While this is a reliable and customisable solution, a new concept has been created in collaboration with LEGO with the goal of designing the loader of the future - namely, the Volvo Zeux. This conceptual model is fully autonomous and electric, introducing an ”eye” for human interaction and a drone to increase environmental awareness. Other key features include four-wheel drive with double Ackermann steering for traction and stability, sensored wheels to maintain balance, a scissor frame for increased maneuverability, and a moving counterweight to automatically alter the vehicle's centre of gravity [3]. Because this concept (seen in Figure 1.1) is powered with electricity instead of fuel, environmental impact is reduced, and since it operates autonomously, both safety and efficiency are improved.

Figure 1.1. Volvo Zeux concept [3]

Recently, Volvo CE implemented this concept in the LX03 - a world-first prototype that will explore the possibilities of machine intelligence and decarbonisation within the industry. Similar to the Zeux concept, the LX03, seen in Figure 1.2, is both electric and autonomous. It also strives for versatility and showcases a scissor frame and moving counterweight. Currently, the prototype has been constructed at full scale, however its autonomous features are still to be fully enforced [4].

Figure 1.2. Volvo LX03 prototype [4]

This project aims to implement this concept, with support from and in collaboration with the stakeholders (Volvo CE), by developing an automated system on a scaled, prototypic loading device. This prototype requires functioning actuation and control systems to drive and perform load cycles in a controlled manner, a vision system with robust image recognition to detect bales of recycled material, and a navigation system to maneuver between loading/unloading points. Reliable machine communication is also needed to ensure all systems act in harmony. Once operational, the concept was tested on a scaled model site (containing plastic bale replicas and loading/unloading points) with the goal of successfully and performing a load and carry cycle without human intervention.

This report will further outline the project requirements, a theoretical background of the *state of the art* alternatives that exist for the prototype's mechanical, electronic, navigation, and software interfaces, the project's methodology, as well as the initial design concepts generated for the prototype. Thereafter, how the wheel loader was implemented and verified will be assessed, along with the testing results, a discussion of these results, and project conclusions. A future work section is included as well.

1.2 Scope

The main scope of this project is to build a scaled prototype that can be tested to prove the Volvo Zeux concept, with the added functionality of plastic waste handling. To achieve this, the scaled prototype was designed to detect plastic bales using image recognition and safely load, carry, and unload the plastic bales at a designated unloading spot.

The prototype is a stand-alone system, meaning that all necessary components are placed onto the prototype and function independently from remote hardware. The ratio of the model is determined by a minimum scale, limited by the space required to fit all necessary components within the prototype, and a maximum scale, defined in terms of material cost and manufacturability.

On the real operating site, as described by the stakeholder, confined spaces can occur. Therefore, the steering mechanism is designed to allow for accurate and high mobility in order to operate in these cramped areas. The lifting mechanism is constructed based on the same characteristics in order to safely load, carry, and unload the plastic bales.

For the prototype to be able to detect the material, namely the plastic bales, a vision system with robust image recognition is developed. The main limitation in this area is firstly to procure a GPU (Graphics Processing Unit) that is powerful enough to handle the computations needed to perform the image recognition, whilst remaining in accordance with the budget. Therefore, this component was selected carefully together with the stakeholders. The second constraint within this area is to create a large enough dataset and construct a robust network model to achieve a satisfactory accuracy in testing.

The site model was created to give considerably ideal conditions regarding the testing environment. That is, the surface was assumed to be flat and the surroundings were assumed as well defined.

1.3 Requirements

To define the problems that needed to be solved in order to meet the stakeholders' expectations, a set of requirements was devised. These stakeholder requirements were further evaluated to construct more extensive and detailed technical requirements. Initially, a first set of stakeholder requirements was constructed to be, in agreement with the stakeholders, classified as either a primary or secondary requirement. The secondary requirements can be seen as lower priority, meaning that these requirements would have to be satisfied if all primary requirements were fulfilled and time allowed. The primary requirements are listed in priority order, meaning that the first requirement has a higher priority than the second one, and so on. The stakeholder requirements are denoted as **SR** and technical requirements are denoted as **TR**.

Primary Requirements

1. **SR** The software shall be able to recognise plastic bales in real time.
    a) **TR** A trained neural network shall be able to perform image recognition and detection of the plastic bales.
    b) **TR** The neural network shall be able to distinguish between multiple bales stacked on top of each other or placed side by side.
    c) **TR** A dataset of sufficient size of plastic bale images shall be created to train the network model.
    d) **TR** The prototype shall have a LiDAR and/or camera embedded.
    e) **TR** The prototype shall be able to find the location of the plastic bale relative to the prototype itself.
2. **SR** The prototype shall be able to pick up plastic bales and unload them at a designated spot.
    a) **TR** The prototype shall have the mechanical strength and stability to pick up and carry the weight of the bales.
    b) **TR** The tool shall be able to lift to a height of at least 230 mm when parallel to the ground and tilt at an angle of $\pm 40°$.
    c) **TR** The prototype shall be able to align itself to the plastic bale to pick up the bale.
    d) **TR** The prototype shall be able to align itself to the funnel to drop off the bales.
3. **SR** The prototype shall be tested and verified.
    a) **TR** The prototype shall be tested on a scaled site model that includes model bales, a loading point, and an unloading point.

4. **SR** All the functionality shall be deployed on the prototype itself, or with flexibility and modularity in mind.
    a) **TR** The prototype shall have a scale of approximately 1:10.
    b) **TR** The plastic bales shall have a dimensional scale of approximately 1:10.
    c) **TR** The prototype shall carry its own rechargeable battery.
5. **SR** The prototype shall be automated.
    a) **TR** The prototype shall have the ability to know its orientation and if it is driving in a straight line.
    b) **TR** The prototype shall have the ability to know its driving distance.
6. **SR** The given budget of 50 000 SEK shall not be exceeded.

Secondary Requirements

1. **SR** The prototype shall be able to navigate itself in the environment.
    a) **TR** The prototype shall be able to navigate autonomously.
2. **SR** The prototype shall have an obstacle detection and avoidance system.

Chapter 2

Methodology

In this chapter, the team organisation, time plan, and budget necessary to execute the project goals are presented.

2.1 Organisation

The project team consists of stakeholders from Volvo CE, one team coach provided by KTH, and nine team members. The team is separated into three subteams, namely the Mechanical, Mechatronics, and Software teams, as shown in Figure 2.1. The Mechanical team was in charge of the design and manufacture of components as well as overall vehicle construction. The Mechatronics team was responsible for the electrical harness, control of motors, actuators, and sensors, as well as the vehicle control system. The Software team managed the development of the machine learning implementation for bale detection as well as the navigation. Towards the end of the project, the teams were dissolved and all team members were active in the final implementations.

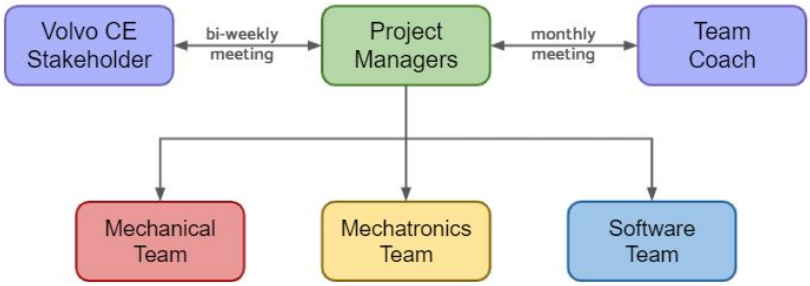

Figure 2.1. Team structure overview

2.1.1 Communication and Project Management

A bi-weekly meeting with the stakeholders was held, via the meeting platform Zoom, to discuss project progress and receive feedback. Towards the end of the project, the frequency of these stakeholder meetings was increased to once a week. Communication was held with the team coach continuously for feedback and the purchasing of components. The communication and project management platforms used within the team are listed in Table 2.1.

Table 2.1. Communication and project management overview

Purpose	Chosen Platform
Project management platform	Projectplace
Documentation storage	Google Shared Drive
Meeting scheduling platform	Google Calendar
Communication platform	Slack, Messenger
Meeting method	Zoom, physical meeting

2.2 Time Plan

The project was divided into four different phases, where each phase consists of multiple tasks for each subteam. The primary goals of the first half of the project was to have all materials and components ordered and received by the end of Phase 2, generate and evaluate the design concepts, as well as create the initial CAD design. In the beginning of Phase 3, each subteam started working on their own designated tasks. All development was to be completed by the end of Phase 3. Two weeks at the end of Phase 3 were reserved for system testing and integration. Phase 4 lasted for one month and consisted of preparing the final presentation and report, as well as doing the verification and validation testing. There was some overlap in the execution of Phase 3 and 4 since much of the work was done in parallel. An overview of the project plan is shown in Table 2.2 and an in-depth project plan is presented in a GANTT chart in Appendix A.

Table 2.2. Project phases overview

Phase	Time	Description
Phase 1	31 Mar - 31 May	Spring report, CAD model, purchase order, concept design
Phase 2	1 Jun - 29 Aug	Ordering materials and components
Phase 3	30 Aug - 30 Nov	Mechanical, Mechatronics, and Software system construction & systems integration
Phase 4	17 Nov - 19 Dec	Final report, presentation preparation, validation and verification

2.3 Budget

As stated in the stakeholder requirements, specifically SR 6, the given budget of 50 000 SEK should not be exceeded. In order to ensure that the budget was not exceeded, a cost analysis, found in Appendix B, was made. In the cost analysis, all ordered parts and materials are listed, including the quantity of each component. The estimated price for each segment, including shipping costs, is based on retailers' prices found online. The price for each segment was summed up to give a total cost estimation. A final value of the actual project spending is also provided.

Chapter 3

State of the Art

The state of the art analysis provided in this chapter will cover the mechanical, electronics, navigation, and software concepts that are relevant for this project's development.

3.1 Mechanical Structure

The mechanical structure of the vehicle has to satisfy many roles to ensure the structural integrity of the vehicle, as described by SR2. The mechanical system is responsible for fulfilling the needs of a chassis as a base for specialised equipment, a lifting mechanism, steering to navigate in narrow spaces, and a propulsion system.

3.1.1 Chassis

The chassis is a complex assembly of components that supports a vehicle's structure. It should be designed to withstand the loads that occur during usage, such as lifting the load, driving to a designated point with the load, and unloading in a safe manner. The chassis is obligated to be robust enough to withstand the forces that can appear during these actions. Some of the loads that can impact the chassis are:

- The weight of the body and cargo
- Vertical and twisting loads as a result of uneven surfaces
- Lateral forces caused by the road camber, side wind, and steering of the vehicle
- Torque transmitted from the engine and transmission

The rigidness of the chassis can be described by two major characteristics: torsion stiffness and lateral stiffness. Torsion stiffness refers to a cross section of the chassis frame. It describes how resistant the chassis is to reluctant momentum, which twists the construction. Lateral stiffness specifies the rigidity of the overall body to the lateral forces [5].

The shape and size of the vehicle chassis is correlated to the dimensions of the components mounted on it. Some of the parts that need to be taken into account when designing this structure are the sensors, steering system, propulsion system, and, especially in this project, lifting mechanism.

Scissor Frame

The scissor frame presented in the Zeux concept is a construction that allows for vertical movement of the whole chassis, whether while immobile or in motion. The choice of lifting system responsible for moving the frame depends on parameters such as lifting height, lifting weight, and the stiffness of the structure. Typically, a scissor construction is used in hydraulic lifts where one (or multiple) actuator(s) is able to lift the entire structure. An example of a scissor frame construction is presented in Figure 3.1.

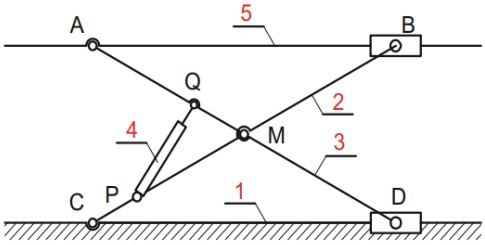

Figure 3.1. Sketch of the scissor frame mechanism: 1. Ground frame, 2. Inner frame, 3. Outer frame, 4. Cylinder, 5. Platform (hoisting terrace) [6]

In the Zeux concept, a scissor movement is accomplished by one actuator and transitional movement of the rear axle, as presented in Figure 3.2.

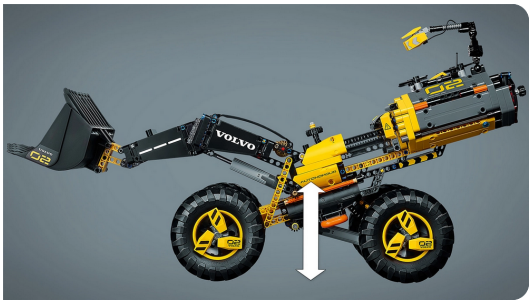

Figure 3.2. Scissor frame of LEGO Zeux concept [7]

Stiff Chassis

A stiff chassis is a concept that simplifies the construction of a vehicle. Torsion and bending stiffness calculations are simplified due to the reduced number of moving

components. Therefore, the model can easily be simplified and imported to CAD (Computer Aided Design) software or other simulation environments. However, a stiff chassis limits the options of applicable steering systems, the type of suspension, as well as method for damping the vibrations [5].

3.1.2 Lifting Mechanisms

In order to transfer, load, and unload the plastic bales, a lifting mechanism is essential, as mentioned in SR2. The movement and functions of real-scale construction vehicles are often accomplished through the usage of hydraulic fluid. In this project, hydraulic actuators were not an option, by virtue of complexity in control and mechanical design in the simplified model. The lifting mechanisms used in modern construction equipment will be investigated in this section as a reference for mechanical design.

Arm Joint System (Kinematic Linkages)

The arm joint system is a mechanism that prevails in the construction vehicle industry. It consists of a kinematic chain with linear actuators and rigid bars. The linkages of vehicles varies with their corresponding application. Figure 3.3 illustrates a sample of the kinematic design of a wheel loader. The arm joint system consists of two cylinders which enable the wheel loader to drive the bucket in a two-dimensional plane.

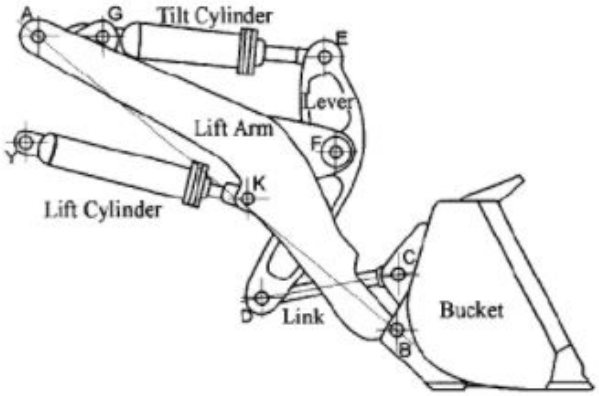

Figure 3.3. Kinematic linkage of a wheel loader [8]

Sliding System

The sliding system is primarily used in forklift trucks to lift and move materials. A carriage with forks is mounted to a vertical mast. The lifting chain is driven by the hydraulic system to elevate and lower the carriage, moving the carriage by sliding it on the rail. A pair of hydraulic cylinders are anchored to the bottom of the mast

and pivot the mast forwards and backwards to tilt the fork carriage. By tilting or lowering the forks, the target object can be transferred [9].

3.1.3 Tools

Various tools could be attached to the lifting mechanism. The configuration of the tools alter the performance and stability of the lifting motion. Choosing an appropriate tool would simplify a complicated motion path and increase the success rate of picking up the plastic bales, as described by TR2a and TR2b.

Loader Bucket

The loader bucket is one of the most common construction vehicle attachments. The general purpose of a loader bucket is to move materials, for instance soil and sand in a construction site. The bucket in Figure 3.4a is a low-profile bucket which has a long and flat bottom compared to normal buckets. A large, sealed bottom prevents the construction materials from being dropped. The bucket scoops material up via the motion of the loader vehicle.

Tynes

Tynes are the forks on the forklift which are used to make direct contact with a load for transport. Tynes are attached to the carriage of a forklift. A pair of tilt cylinders adjust the tilt movement of the carriage and the angle of the tynes relative to the ground. The forks of the machine are altered to an appropriate position and when the vehicle approaches the objective, the forks slide between the ground and the base of the objective.

Bale Grabber

A bale grabber, shown in Figure 3.4c, is an attachment on a loader that is typically used to handle bales. A pair of arms extend from the grab mast, actuated by hydraulic cylinders. The target object is held in position rigidly by the sidewise friction force. The simple mechanism of the grabber results in uncomplicated control and motion of the attachment [10].

(a) Bucket [11]

(b) Tynes [12]

(c) Bale Grabber [13]

Figure 3.4. Various lifting tools

3.1.4 Steering

The steering is a system of components that allow a vehicle to follow a desired course. The manoeuvrability of a wheel-based ground vehicle is highly dependent on its steering system. Steering can be achieved in many different ways. Most commonly, four-wheeled ground vehicles steer by applying an angle to the front wheels, which forces the vehicle to move in a curved trajectory. Another method of steering is to control the speed of the inner and outer wheels to make the vehicle turn, called differential steering [14].

Articulated Joint Steering

An articulated vehicle has a vertical pivot joint in its construction, allowing the vehicle to apply an angle between the wheel axles. Any vehicle with a towing trailer could be described as articulated. For a four-wheeled vehicle, an articulated joint can be enough to achieve a balanced steering system with tight cornering capabilities, see Figure 3.5 [14].

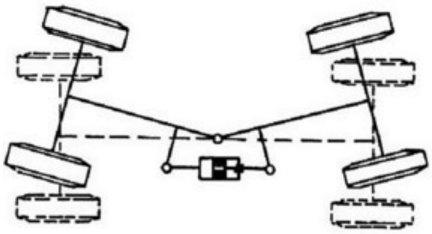

Figure 3.5. Geometry of an articulated vehicle [14]

Ackermann Steering

The Ackermann steering geometry consists of an arrangement of linkages designed to solve the problem of the inner and outer wheels of a vehicle turning with different radii. With a perfect Ackermann geometry, the inner and outer wheels turn with the ideal radius, pivoting around the centre of turning, see Figure 3.6. A good approximation of a perfect Ackermann geometry can be achieved with a four-bar linkage on the steering wheels [14][15].

The drawing in Figure 3.6 shows an example of Ackermann steering when implemented on a front-wheel steered vehicle.

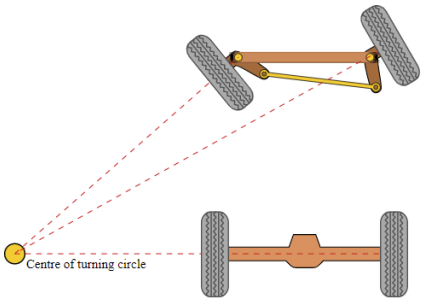

Figure 3.6. Geometry of Ackermann steering [16]

A further development of this configuration is the double Ackermann steering configuration, which is essentially a four-wheel steering system [17]. This results in a narrower turning radius as well as a pivot point parallel to the centre of the vehicle. This could also provide a more independent steering, if compromising a bit of the tire slip, by letting the vehicle crab steer [18]. This allows the vehicle to steer the different wheel pairs in-phase, and results in a diagonal path without altering the yaw direction of the vehicle. A visualisation of both the double Ackermann configuration as well as the crab steering can be seen in Figure 3.7.

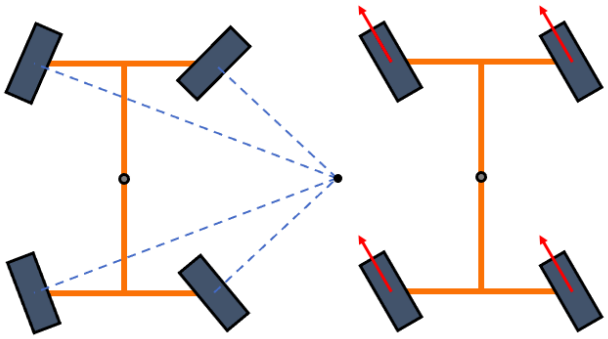

Figure 3.7. Geometry of double Ackermann steering (left) and visualisation of crab steering (right)

Mecanum Wheels

Omni-directional control of a vehicle can be achieved through the use of mecanum wheels. These wheels have rollers that are skewed with respect to the wheel axle, allowing them to roll in two dimensions. A platform with four of these wheels, individually actuated, results in a vehicle that can move in the longitudinal, lateral, and yaw directions. Controlling such a vehicle does not require excessive computations, even when including wheel slips in the calculations [19]. An example of an application using mecanum wheels is shown in Figure 3.8.

Figure 3.8. An example of an omni-directional platform using mecanum wheels [20]

3.1.5 Drive Train

The drive train of a vehicle is a set of actuators and mechanical components that together make the vehicle move. This section covers well-known solutions of drive train systems and explains their principles.

Drive Types

A wheel-based ground vehicle commonly has a set of wheels that are actuated. The choice of wheels to actuate directly affect the vehicle's performance, agility, and complexity. All-wheel drive, for instance, is beneficial for terrain vehicles since it allows for more thrust, regardless of the weight balance, and has a lower chance of getting stuck in difficult terrain. However, actuating all the wheels of a vehicle is complex and costly, especially for a vehicle with a suspension system and steering [21].

The Differential is a device that is often used for distributing the torque of a single actuator to two wheels of a single axle during cornering. As the vehicle turns, the inner and outer wheels will have different turning radii and therefore rotate at different speeds. Using a differential to solve this problem works well as long as both wheels on the axle have traction. In other cases, the differential will distribute the torque evenly to both wheels, so in the case where one of the wheels loses traction, the other wheel loses torque [22].

Front/Rear Wheel Drive is an alternative to all-wheel drive which has reduced cost and complexity. On road conditions, this solution is sufficient for most consumers, and therefore the most common design choice. During hard forward acceleration however, the load balance of the front and rear wheels change and more of the vehicle's weight pushes against the rear tyres and less against the front. This increases the maximum thrust that the rear wheels can achieve and reduces the maximum thrust of the front wheels. For a performance vehicle, rear-wheel drive is therefore beneficial for greater acceleration and all-wheel drive can increase the acceleration performance even more [21].

Wheel/Hub Motors are a less common solution for cars and transport vehicles, but more common for robots. The reason why this solution is less common for vehicles are several, but in particular there is one major problem affecting the stability and comfort. As a motor is introduced into the wheel hub or wheel itself, the unsprung mass is increased. This puts higher loads on the tyres and suspension on a bumpy road, making the ride less comfortable for a passenger [23]. This is not a problem for all applications though. Individual-wheel actuation comes with the benefit of controlability which can benefit the performance of a vehicle by dynamically balancing the torque distribution [21]. In some robotic applications, such as omni-directional and differential steered platforms, individual-wheel actuation is a requirement in order to achieve the intended functionality [19].

3.1.6 Actuators

Actuators are components that are responsible for the controlled movements of mechanical systems. An actuator requires a source of energy and a control signal and outputs a force, moment, linear movement, rotation, or a combination of these. Examples of common actuators are electric motors, servos, combustion engines, hydraulic pistons, and solenoids. Motors and engines generally output continuous rotation and torque, and are therefore the common choice for wheel actuation of ground vehicles. Pistons and linear actuators are responsible for outputting linear motion and force [24].

Electric Motors

An electric motor can be divided into two main parts: the rotor and the stator. The rotor is the part that rotates and the stator is the part which contains the rotor, attaches the motor to the machine of its implementation, and contains the electrical connections. Most electric motors generate torque and angular speed using rotating magnetic fields. Hence, an electric motor requires electromagnets that can be toggled on and off [24].

Brushed DC Motors are one of the simplest types of motors. It has permanent magnets in the stator and electromagnets in the rotor. The coil windings in the rotor are connected to a commutator which rotates with the rotor. Two brushes, that are mechanically connected to the stator, are ”brushing” against the commutator, connecting the electrical source to the coils. The commutator has two or more terminal plates that, depending on the angle of rotation, supply the coils in such a way that the magnetic field of the rotor is at an offset angle to the magnetic field of the stator. This ensures that, for a constant direct current (DC) voltage source, regardless of the rotor angle, there is a moment with a constant direction generated [24].

Brushless Motors use electrical switches (such as transistors) to control the magnetic field of the stator. The rotor on some types of brushless motors contain permanent magnets. These are called ”permanent magnet synchronous motor”, or PMSM in short. They are termed ”synchronous” because the rotor needs to be in synchronisation with the electric signal in order to work properly. This requires a properly tuned feedback control system [24].

PMSMs can have any even number of magnetic poles on the rotor and any number of phases on the stator coils. Single-phase motors are used in some applications, but three-phase motors are a more common choice when the power requirements are higher. Two-phase motors, commonly known as stepper motors, are widely used in 3D (three-dimensional) printers and industrial machines [24].

3.2 Embedded Systems

Embedded systems are necessary in the prototype to tie together the mechanical and software systems, handle communication, and manage the vehicle's electronics.

3.2.1 Microcontroller

A microcontroller is a programmable computer on a chip, designed specifically for embedded systems applications. Most microcontrollers have built-in circuits for various control applications. These circuits can for instance be analog to digital converters (ADCs, or A/D converters), digital to analog converters (DACs, or D/A converters), and parallel input/output ports. Microcontrollers can be programmed using the assembly language, or in a higher level language such as C [25]. STM32 is a family of microcontrollers based on 32-bit ARM Cortex-M processors designed to offer real-time capabilities, digital signal processing, motor control solutions, and more. The STM32 ecosystem provides hardware and software tools, as well as libraries for development of microcontroller software [26].

3.2.2 Universal Asynchronous Receiver-Transmitter (UART)

A UART is a hardware device that is used for asynchronous serial communication. UART is also a communication protocol that is associated with UART devices. This communication protocol describes the physical and data-link implementation. The UART has two signals, RX and TX. Data is sent via the TX channel, and received via RX. The data is packed into packets consisting of single word of 7 to 9 bits, start and stop bits, and an optional parity bit. The UART has individual registers for the received bytes and for the bytes to transmit, and the communication is full-duplex, meaning that communication can happen in two directions simultaneously [27].

3.3 Navigation

The vehicle is required to locate points of interest and navigate within the environment, as described by the Primary Requirements SR2 and TR5a. Thus, a navigation system is needed.

3.3.1 Operating System

The Robot Operating System (ROS) is a framework for writing robotic software. ROS contains tools, libraries, and conventions for creating complex and robust robot behaviour across robotic platforms [28]. ROS offers a variety of packages in which the ROS software is organised [29], including packages with Simultaneous Localisation and Mapping (SLAM) functionalities which can be used to construct maps for navigation with collected data [30].

3.3.2 Perception

The ability to process a 3D environment defines an autonomous vehicle's capability to position itself in space. There are several hardware solutions for robotic vision, such as light detection and ranging (LiDAR) and RGB-D (Red, Green, Blue colour model with Depth) cameras. When combined with inertial measurement units (IMU) and odometry systems, the accuracy of the estimated position is increased [31].

RGB-D sensor

RGB-D sensors commonly consist of a RGB (Red, Green, Blue) colour camera and infrared (IR) technology to measure depth and the distance to objects through time-of-flight (ToF) or stereo [32]. Depth calculations by ToF are based on the time it takes for the emitted light to reach an object and return [33], while stereo depth measures are based on the coordinate disparity of an object between two readings with different viewpoints [34]. By utilising a projected IR pattern with known variation with respect to distance, the depth can be calculated through triangulation and the surroundings mapped into point clouds, a cluster of points representing a 3D object. The use of IR projection results in dense and robust information, and improves the vision in conditions with varying and/or bad lighting [32].

LiDAR

A common option for perception and localisation is LiDAR systems. LiDARs use laser beams to scan the surroundings through laser rangefinder principles, where the laser beams' ToF is measured and the distance to the object is calculated, outputting a 3D point cloud of the scene. By rotating the base, a 360° field of view (FoV) can be achieved, using different solutions to cover the vertical scanning, such

as mechanically moving mirrors.

The LiDAR provides very accurate ranging measurements but offers poor object recognition, and is therefore typically combined with a camera [35]. Another downside of the LiDAR system is the laser projections' ability to accurately map the surroundings in challenging weather conditions, such as fog or rain [36].

IMU

An inertial measurement unit is a device that measures force, velocity, and orientation of a body through a combination of sub-units, such as gyroscopes and accelerometers. While gyroscopes and accelerometers can provide information about the angular rotation and inertial acceleration respectively, more advanced IMU systems also include magnetometers [37]. Magnetometers are mainly used for computing the direction through changes in magnetic fields and may provide a minor increase in position accuracy. IMU systems, with or without magnetometers, provide most of the information needed for navigation, but both are subject to drift errors that increase with operating time [38].

Odometry

Odometry is commonly used for positioning and estimating location with respect to an initial location, where the odometry data may be acquired from a sensor such as LiDAR, encoder, or camera. Each solution has its weaknesses, but can be combined to provide an increased accuracy of the odometry measures. A common combination consists of a LiDAR and wheel encoder or visual odometry (VO), where the secondary data source (Encoder/VO) allows a point cloud to be mapped with respect to a fixed coordinate system [39][40].

3.4 Computer Vision

Computer vision is necessary for the prototype to recognise plastic bales, and therefore fulfill SR1. In this section, the main underlying theories for computer vision are discussed.

3.4.1 Image Processing

In order to implement computer vision, the computer has to process the images taken by the camera. Image processing as a computational technique has existed for a long time and has varied applications, such as digitisation, restoration, and segmentation [41].

Image processing is the method which transforms a camera image to a data representation. Moreover, working with real-time image processing creates constraints on

image processing speed, since each frame has to be processed within a specific time frame [42].

3.4.2 Image Recognition

Image recognition is a method to achieve computer vision. It represents a set of techniques for detecting and analysing images to enable the automation of a specific task. The three main tasks within the field of image recognition are [43]:

- Classification

  Classification is the identification of the ”class”, i.e. the category to which an image or the objects inside the image belong.

- Detection

  Detection is to perform classification and localisation. It is a technique to distinguish between objects in an image or video and use bounding boxes to show their location within that image/video.

- Segmentation

  Segmentation is to perform classification on each pixel in an image. The whole image is divided into pixels which can then be classified and labeled.

3.4.3 Convolutional Neural Network

An Artificial Neural Network (ANN) is a computing system architecture that mimics the biological neural networks of the human brain and its operation. In other words, an ANN is a set of neurons organised in layers. Each neuron is a mathematical operation that takes its input, multiplies it by its weights (the strength of the connection between two neurons), plus its biases (a parameter that adjusts the output to better fit the model to the data), and then passes the sum to the other neurons. Equation 3.1 shows a simple linear transformation of a neuron. A neural network learns how to classify an input by adjusting its weights and biases based on previous computations [44].

$$Output = Weight * Input + Bias \tag{3.1}$$

An example of an ANN is the Convolutional Neural Network, or CNN, named after the convolutional layers within its architecture, and can be seen in Figure 3.9. The process of a CNN can be separated into two parts - feature extraction and classification. This section presents some basic information regarding CNN components [44].

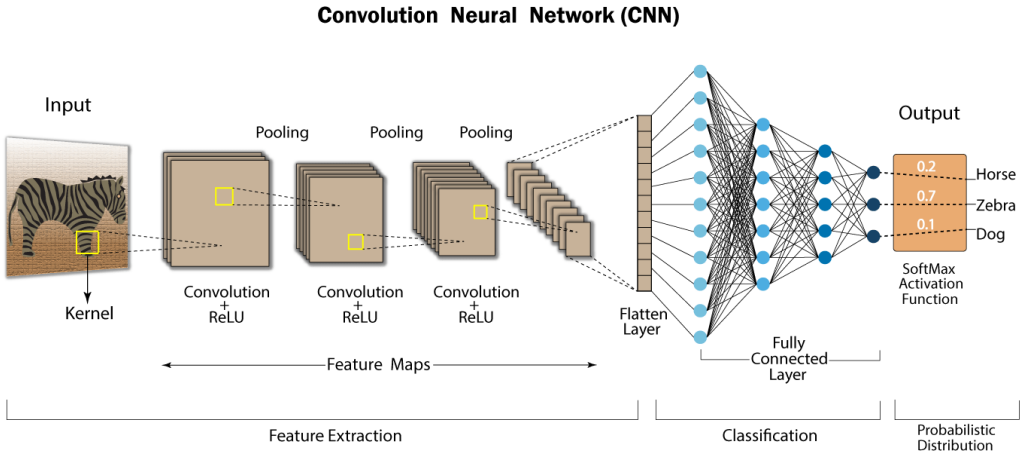

Figure 3.9. CNN structure [45]

Filters/Kernels

Filters or kernels are used to extract features such as edges from the images. They are matrices that slide over the input data, perform element-wise multiplication within the sub-regions of the input data, and generate the output as matrices, which is called a "Feature Map" [44]. An example is shown in Figure 3.10.

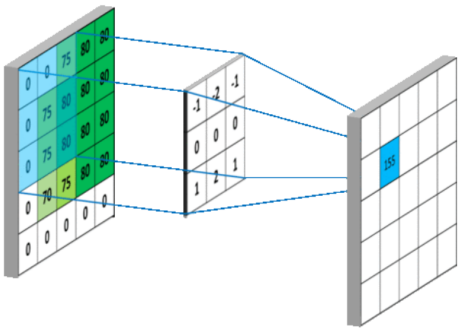

Figure 3.10. Filter and Feature Map [46]

Zero Padding

Zero padding is a kind of same-padding technique to preserve the original input size for the output of the convolutional layer. This technique adds a border of pixels all with zero value around the edges of the input images as shown in Figure 3.11 [44].

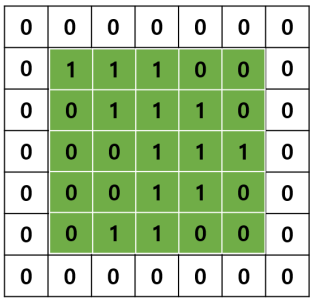

Figure 3.11. Zero padding [47]

Pooling

There are two primary types of pooling - max pooling and average pooling. Max pooling takes the maximum value from a specific region in the feature map, while average pooling takes the average as shown in Figure 3.12 [44].

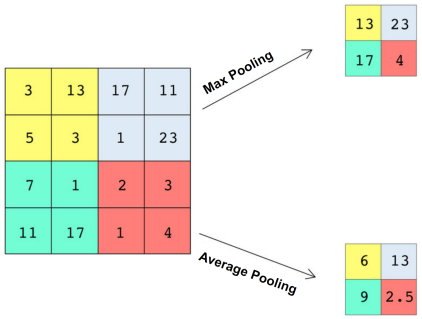

Figure 3.12. Max Pooling and Average Pooling

3.4.4 YOLO v3

A common object detection algorithm used for real-time applications is the You Only Look Once (YOLO) detector. YOLO is both fast and accurate, and changes in the model's size can be made without retraining it [48].

The YOLO detection algorithm works as a CNN. However, it only uses 1x1 convolutions for its predictions [49]. YOLO v3 uses a feature extractor called Darknet-53, which for object detection tasks provides a network of 106 convolutional layers [50]. Figure 3.13 shows the YOLO v3 network architecture.

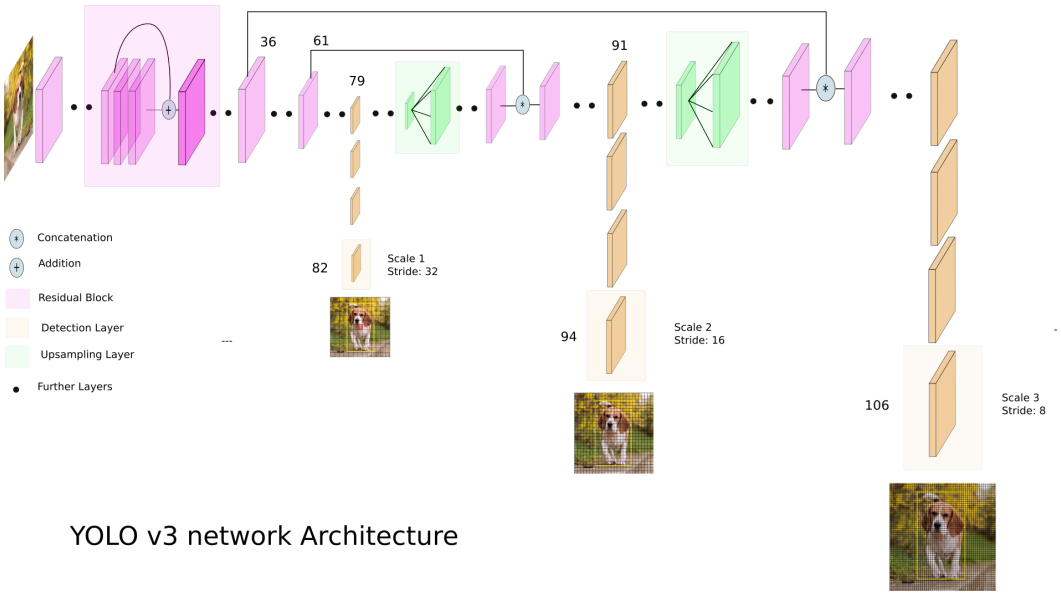

Figure 3.13. YOLO v3 architecture [50]

3.4.5 Data Augmentation

Neural networks typically require large, diversified datasets in order to efficiently train the network and accurately predict its classes. With an increase in complexity of the network (i.e. a larger number of hidden layers and therefore a larger number of hidden nodes), there is also an increase in the number of trainable parameters. These parameters are how the network maps the inputs (for example, an image of a dog) to the relevant outputs (i.e. the label "dog"). With an increase in parameters, more data is needed to train them. However, oftentimes the amount of data is limited, which can result in an unregularised and therefore inaccurate network. To solve this problem, data augmentation can be introduced with which new data is synthesised by altering the data that is already available. This not only increases the dataset, but also diversifies it [51]. Some of the techniques for data augmentation of images include [52]:

- Geometric transformations
  - Scaling
  - Cropping
  - Flipping
  - Rotation
  - Translation
  - Gaussian Noise
- Photometric (colour) transformations

- Altering brightness
- Altering contrast
- Altering saturation
- Altering hue
- Changing to grayscale

Some of these techniques are demonstrated in Figure 3.14.

Figure 3.14. Examples of image transformations used in data augmentation [51]

Utilising data augmentation techniques such as these has been proven to increase the accuracy of neural network classifications [53].

Chapter 4

Concept Design

In this chapter, the generated concepts of the prototype are discussed. The concepts are divided into two major systems - mechanical structure and software. The functional design and an introduction to how the comparison matrices were created will also be presented.

4.1 Functional Design

It is important to take the desired functionalities from the technical requirements into account when designing a concept. Some of the technical requirements have a direct influence on the hardware that is chosen, such as TR 1d and TR 4c. Other requirements, like TR 1a and TR 5a, describe functionalities of the prototype that can be achieved in many different ways and with the use of different hardware. From the theory covered in the state of the art analysis, it was carefully chosen that a prototype wheel loader (of a scale of roughly 1:10) including a chassis, lifting mechanism, propulsion system, steering mechanism and perception system, was to be developed. The technical and stakeholder requirements, outlined in Section 1.3, provide a list of functionalities for the prototype, including:

- An image recognition and image localisation system
- A navigation and localisation system
- A vehicle and actuator control system
- An energy storage and power management system

For the first two functionalities, a powerful GPU is needed due to the high demands on computational power that these require in order to function with minor latency. Although vehicle and actuation control can be handled by the same processor, it was considered to develop those functionalities using separate hardware. This benefits the prototype's modularity due to the introduced possibility to replace the hardware that is implemented for the autonomous system without affecting the

vehicle's mechanical capabilities. The choice also benefits parallel development since the mechanical structure, the mechatronic systems, and the ROS systems could be tested individually during development, without needing the presence of other systems.

To justify how the systems would interconnect and communicate with each other, meet their power demands, and how the functionalities should be implemented, a concept for the functional architecture was developed early on and used as a guide for concept generation and hardware planning. The functional architecture can be seen in Figure C.1 in Appendix C, and illustrates how the systems are implemented. The systems are assigned to four different groups to describe the subsystems that could be developed in parallel and tested individually. Some of the hardware mentioned in the functional architecture was chosen through the help of a comparison process that is explained in the following sections.

4.2 Comparison Matrix

The comparison matrix is a tool to visualise similarities and differences between simple products as well as complex and abstract concepts. The matrix helps to organise and classify the elements in which products are compared. The features and characteristics of each element are evaluated according to a set of criteria, enabling easy recognition of their advantages and disadvantages in order to bolster the decision making process.

To use this method, the following steps are necessary. Firstly, criteria are chosen for evaluation, then each concept receives scores in a specified scale for every criterion. In this project, the concepts were initially scored from 1 to 5 for each criterion, where 5 is the best. Thereafter, the criteria are compared to each other and ranked according to importance to weight their influence on the final result. The final step is to multiply the concepts' initial scores for every category by the weighted criteria to receive the final evaluated values for each concept.

4.3 Mechanical Structure

In this section, the design choices that have been made regarding the mechanical structure will be evaluated and explained. The concept design for the vehicle will also be presented.

4.3.1 Chassis

The prototype operates indoors with ideal environmental conditions, so the construction is not obligated to withstand weather conditions such as rain or sub-zero temperatures. Therefore, the construction of the chassis is focused on mechanical

properties to withstand the loads that act on it, as well as ensuring proper dimensions to fit all components. Different concepts were taken into account and evaluated using the comparison matrix method. Three designs were considered:

- Customised stiff chassis
- Zeux concept chassis (scissor frame)
- Construction based on a remote-controlled (RC) rally car's chassis available on the market

The designs were evaluated using these factors:

- Weight of the finished chassis assembly
- Reliability in the presence of unfavorable factors
- Complexity of the construction
- Cost, i.e. the expense of purchasing the necessary components and materials
- Accessibility of the assembly parts in the market

The result of this process is visible in Appendix D in Figure D.1 where, in the pentagonal graph, the value of each concept is presented. Figure D.1 shows that the best candidate is the stiff chassis concept, however this option lacks some core functionalities, such as the scissor feature for better versatility. Therefore, the Zeux frame is instead selected to better fulfill the scope of the project, since this will not induce any major influence on the cost and reliability. The final scores of the three concepts, with the full score calculations available in Table D.1 in Appendix D, were:

Table 4.1. Final score of chassis evaluation

Type of chassis	Score
Customised stiff	11.5
Scissor frame concept	9.125
RC rally car's	6.375

The main parts of the body were designed to be manufactured from steel sheet metal and connected with each other using conventional methods, such as screws or blind rivets. The initial concept design is presented in Figure 4.1.

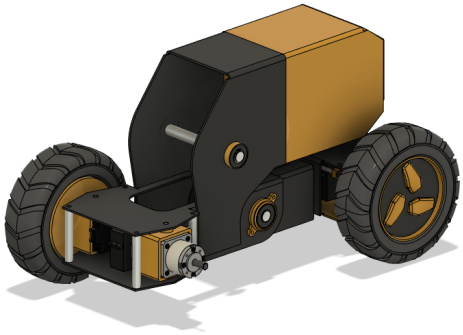

Figure 4.1. The chassis design

Scissor Frame Mechanism

The scissor frame mechanism is realised by joining two parts of the chassis to allow them to pivot. The front part is connected to the lifting mechanism and the main body. The rear part of the scissor mechanism is connected to the front of the chassis with a linear actuator responsible for lifting the construction and pivoting. This initial scissor frame concept is demonstrated in Figure 4.2.

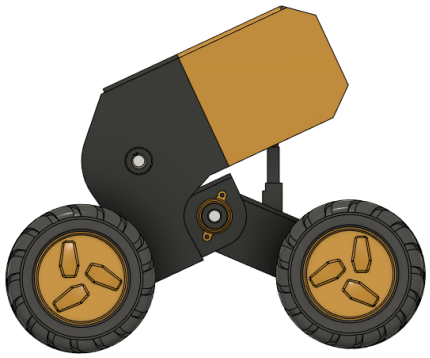

Figure 4.2. The design of the chassis with scissor frame

The scissor frame mechanism allows the vehicle to add functionalities such as:

- Adjusting the height of the bucket by moving the rear axle
- Adjusting the angle of the bucket to the ground
- Adjusting the height of the vehicle to navigate on rougher terrain

The scissor frame is a custom design for the purpose of this project, inspired by the Zeux concept.

4.3.2 Lifting Mechanism

The potential lifting mechanisms and tool attachments of the prototype model are presented in Sections 3.1.2 and 3.1.3. The two lifting systems considered were kinematic linkages and sliding systems, which were examined with regards to the following criteria:

- Cost, i.e. expense of purchasing components and materials
- Accessibility of the assembly parts in the market
- Flexibility of the lifting mechanisms and motion speed
- Complexity of controlling the lifting mechanisms
- Reliability, i.e. accuracy and precision of lifting mechanisms

The final scores of the lifting mechanisms are tabulated in Table 4.2, and how this was calculated is shown in Table D.2, Appendix D.

Table 4.2. Final score of lifting mechanisms evaluation

Type of lifting mechanisms	Score
Sliding system	8.25
Kinematic linkage	11.75

The kinematic linkage mechanism has a higher score than the sliding system. This result was due to the lack of agility of the sliding system since a tilting cylinder would be needed to perform the loading and unloading motions. Even though the sliding system had a more straightforward control principle, this advantage was not significant. Correspondingly, the kinematic linkages were preferable over the sliding system and was therefore chosen as the final concept for continued work.

Tool Attachment

After deciding the lifting system's mechanical design, an appropriate tool attachment could be selected. A bucket would be the appropriate tool for the prototype model. This decision was made after discussions with the stakeholders, where the importance of collecting plastic bale fragments was emphasised. Among the three possible solutions, only the bucket has the ability to preserve the fragments since tynes and bale grabbers are not able to achieve this purpose. Consequently, the bucket will be used in the project prototype.

A CAD drawing of the initial lifting mechanism design is shown in Figure 4.3. A customised bucket is driven by Z-bar linkages. The Z-bar linkages are composed of pins and customised metallic bars. Sidewise planer linkages are connected by cylindrical bars and pins. Two linear actuators are attached in the mid-plane of

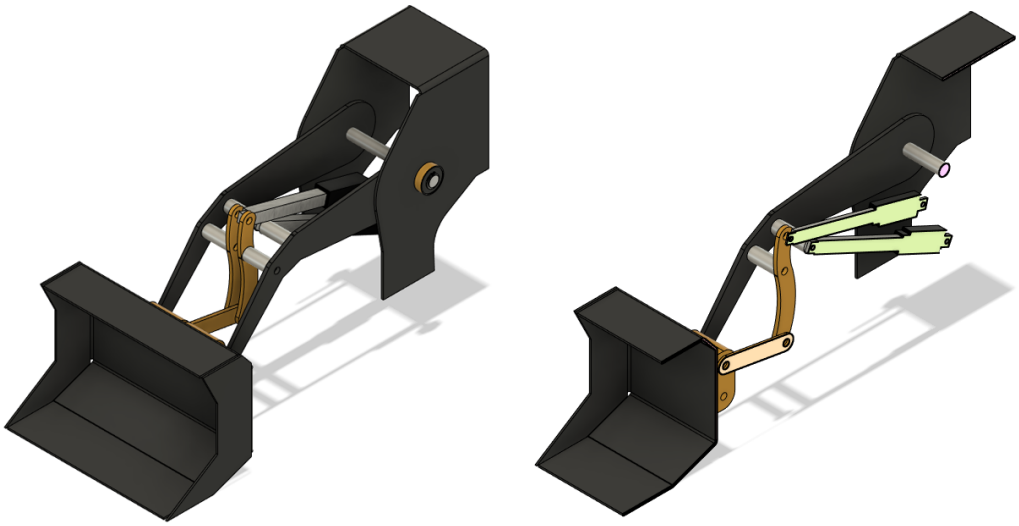

Figure 4.3. Design of the lifting mechanism

the mechanical structure to drive the lifting and unloading motion, as described in Section 3.1.2. The bucket is designed to be manufactured by sheet metal and linkage components have a simple geometry which will enable them to be easily machine-manufactured.

4.3.3 Steering

Three main steering mechanisms were featured for matrix comparison: Double Ackermann, articulated steering, and mecanum wheels. Mecanum wheels were considered for the many functionalities that they provide without requiring mounting of an actual steering mechanism. The main flaw with this solution is that if there is waste or other obstacles on the ground, that could interfere with the mechanics of the wheels, hindering movement. Therefore, mecanum wheels were ruled out.

Articulated steering can provide the turning angle necessary to achieve prototype manoeuvrability and is highly popular in wheel loader constructions. However, since the bucket is mounted to the chassis, it would move along with the chassis while turning which makes the vehicle itself difficult to steer in cramped spaces. Meanwhile, Ackermann steering limits the movement of the bucket which is preferable in the context of confined spaces.

The designs were evaluated using these factors:

- Cost, i.e. the expense of purchasing the necessary components and materials
- Accessibility of the assembly parts in the market

- Weight/Dimension ratio, influence of the system on vehicle weight
- Controlability of the vehicle with the steering solution
- Reliability in the presence of unfavorable factors

The weighting diagram of the presented systems with the above criteria is located in Appendix D in Figure D.3. The evaluated scores of the steering solutions are shown in Table 4.3, and how they were calculated in Table D.3, Appendix D.

Table 4.3. Final score of steering mechanisms evaluation

Type of steering mechanisms	Score
Mecanum wheels	8.75
Ackermann steering	9.5
Articulated steering	8

Due to the evaluated scores, the decision was made to implement Ackermann steering. To decrease the turning radius even further using this solution, the mechanism can be applied to both front and rear axles, called a double Ackermann. With one steering servomechanism for each axle and a system of rods which connect both wheels in one axle, the wheels are turned. The initial concept for the steering system is presented in Figure 4.4.

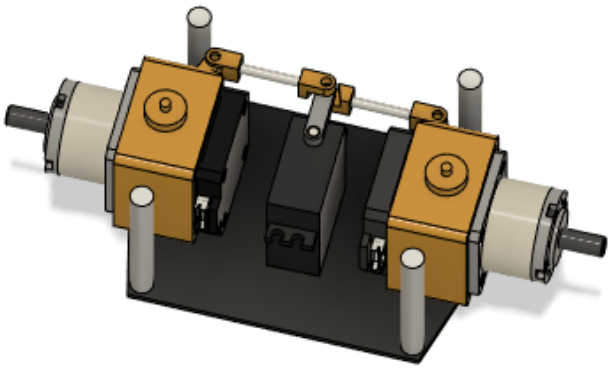

Figure 4.4. Design of the Ackermann steering mechanism

The minimum turning radius for the initial concept was designed to reach a value of 450 mm, as seen in Figure 4.5.

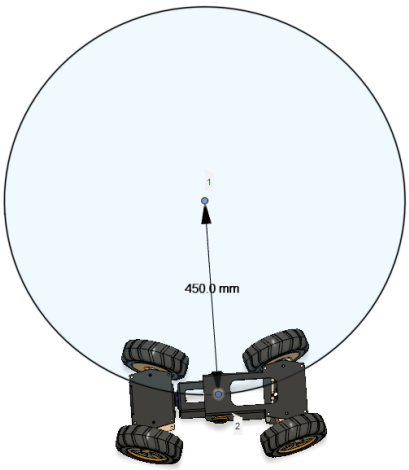

Figure 4.5. Turning radius of the double Ackermann solution, 450 mm

4.3.4 Drive Train

In this subsection, drive train components will be evaluated with explanations of the functionalities that were considered.

Wheels

The wheels are designed to have enough elasticity to maintain constant contact with the ground. Therefore, no additional suspension was needed. The choice of wheel was narrowed down to three types:

- Air-filled rubber tires
- Airless rubber tires
- Custom hollow tires

Airless rubber tyres were chosen due to their availability on the market, cost, and reliability.

Propulsion System

For the propulsion system, the options taken into consideration were two- or four-wheel drive (WD) and number of motors used. The systems were evaluated using these factors:

- Cost, i.e. the expense of purchasing the necessary components and materials
- Accessibility of the assembly parts in the market
- Weight/Dimension ratio, i.e. the influence of the system on vehicle weight

- Controlability of the vehicle with the propulsion system
- Reliability in the presence of unfavorable factors

The evaluated results are visible in Table 4.4 and Figure D.4, with the score calculations available in Table D.4, Appendix D. These results show that the best solution is 2WD with two motors, one for each front wheel. This solution provides a substantial amount of power while maintaining a good grip to the ground due to the motors' placement on the front axle, where most of the vehicle's weight is focused. Therefore, a two motor solution was chosen due to the simplicity that it provides to the powertrain. It removes the necessity of using differential mechanisms, cardan joints, or other axles that would deliver momentum to the wheels. The differential steering is solved through adjusted control of the motor speed which takes into account the differences in wheel rotary speed when turning.

Table 4.4. Final score of propulsion system evaluation

Type of propulsion system	Number of motors	Score
2 WD	1	8.625
2 WD	2	10.125
4 WD	1	6.125
4 WD	2	7.125
4 WD	4	9.375

Propulsion Motors

Propulsion motors were selected to further develop the vehicle model. The potential candidates were brushless DC motors, stepper motors, and DC motors. An evaluation graph, shown in Figure D.5 (Appendix D), was created to illustrate the decision process with the following criteria, which are based on TR4a and SR6:

- Cost, i.e. the expense of purchasing the motor and corresponding parts
- Accessibility of the motor's mechanisms
- Power to weight ratio
- Complexity of implementation and control
- Reliability of the motor's behaviour

The candidates had a very similar performance in cost, accessibility, and power to weight ratio. The stepper motor is preferable in controlability whereas the brushless DC motor is preferable in reliability. The final, evaluated scores are presented in Table 4.5 and the calculations to determine these scores are presented in Appendix D, Table D.5.

Table 4.5. Final score of propulsion motor evaluation

Type of Motor	Score
Brushless DC motor	9.25
Stepper motor	9.625
DC motor	7.875

Conclusively, the stepper motors scored highest in the evaluation graph. The overall grading of stepper motors and brushless DC motors were approximately identical. The decision was made by assessing the significance of reliability and complexity. Simpler motors would simplify the embedded system control. On the other hand, reliable motors would provide a stable vehicle performance. The stepper motors could expedite the electronic design and control programming, whilst being a reliable solution. At the same time, the excess stability provided by the brushless DC motor is redundant for this application. As a result of the above analysis, stepper motors were chosen as the propulsion motors.

4.4 Software

In this section, the design choices that have been made with regards to the prototype's software implementation (navigation and image recognition) will be motivated and explained. This is in accordance with SR1, SR2, and SR5.

4.4.1 Navigation

Since the prototype is required to localise and orient itself, as stated in TR5a, high demands are put on its navigation system. In order for the prototype to perform the task, a considered solution was to let it build its own map of the environment in which it operates. Moreover, the prototype also has to localise the loading and unloading points for delivery of the plastic bales. To perform these tasks, a SLAM algorithm was considered with the operating system ROS.

Mapping

In order for the prototype to create its own map, it would receive input from a camera and create a map of its surroundings with SLAM algorithm, see Figure 4.6. For mapping, a LiDAR is often used because it provides a 360° input of the surroundings. However, a camera with stereo vision, such as Stereolabs' ZED 2 camera, was considered sufficient in this project, since the mapping would be constrained to a small environment. For the initial navigation concept, mapping using SLAM was considered.

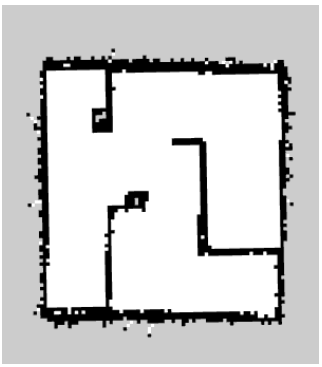

Figure 4.6. Map generated from SLAM based algorithm [54]

Localisation

The second crucial part of navigation was localisation. The prototype has to know where to pick up the plastic bales and where to drop them off, as well as its own position. The information about the robot's movement and position can be retrieved from a built-in IMU in the camera.

4.4.2 Image Recognition

The concept for the network was to develop it in the programming language Python with the YOLO v3 neural network architecture. To handle the processing power required to run image recognition with a CNN in real-time, a powerful GPU, such as the one a Nvidia Xavier AGX 32GB provides, was considered sufficient. Further information for the CNN solution is discussed below.

Convolutional Neural Network

In this project, the YOLO v3 architecture is chosen, a type of Convolutional Neural Network. There are three main reasons why a CNN is preferred over a feed-forward neural network for image recognition:

1. Using Spatial Structure

   A fully-connected feed-forward neural network learns by flattening out an image into vectors, which at the same time diminishes the spatial structure of an image. This leads to a low accuracy of prediction, and little to zero accuracy in cases of complex images with pixel dependencies. The usage of a sliding window in a CNN however, can capture the spatial and temporal dependencies in an image since each neuron only connects to its receptive field.

2. Parameter Sharing and Local Connectivity

The number of parameters in a neural network grows rapidly with an increase in the number of hidden layers. In a fully-connected feed-forward neural network, it would require a very high number of neurons even in a shallow architecture. This architecture is generally impractical for larger inputs such as high resolution images. Comparing to that, the architecture of a CNN successfully tackles this issue through parameter sharing and local connectivity.

- Parameter Sharing

  The vectors of weights and biases are called filters and represent particular features of the input. Parameter sharing is the sharing of the same filters between many neurons. This reduces the memory required during training since these filters are used across all receptive fields that share the filter, as opposed to each receptive field having its own weights and biases vector in a fully-connected feed-forward neural network.

- Local Connectivity

  Local connectivity is the concept of each convolutional neuron connecting and processing data only for its receptive field (subset of the input image), unlike a fully-connected feed-forward neural network where all the neurons are fully connected.

3. Reduce Model Overfitting

   With less parameters needed in a CNN, the possibility of overfitting the model is therefore reduced.

Dataset

Because no repository currently exists of labelled plastic bale images, the dataset for this project had to be created independently. A possible method was by collecting images online and taking images of the custom plastic bales in real life. Due to the time-consuming nature of this data collection, data augmentation, with geometric and/or photometric transformations, can be utilised to both expand and diversify the dataset.

Chapter 5

Implementation

This chapter will introduce how the generated concept designs were implemented in the actual prototype to fulfill the project requirements.

5.1 Mechanical Design

This section describes the final mechanical design of the proposed concept in detail along with its subsystems. The final design of the automated electric wheel loader can be seen in Figure 5.1.

Figure 5.1. Final CAD drawing

The fully assembled vehicle consists of about 120 manufactured parts and 350 fasteners (including washers), with a total weight of 12.8 kg. The vehicle is 245 mm wide, 725 mm long, and 265 mm high in its standard position.

5.1.1 Chassis

The scissor frame concept is implemented in the chassis subsystem, which is mainly composed of 2 mm thick sheet metal components formed into a hollow construction. In this hollow chassis, a linear actuator is incorporated, with the functionality of extending the distance between the body and the chassis similar to a scissor frame, as can be seen in Figure 5.2. This feature grants the functionality of adjusting the bucket configuration via this additional degree of freedom and offers more stable loading and unloading characteristics. The chassis can theoretically reach an angle of 35.7°. This is however a maximum which is hard to reach in reality, due to no feedback on this particular actuator, and no end-stop on the upper position.

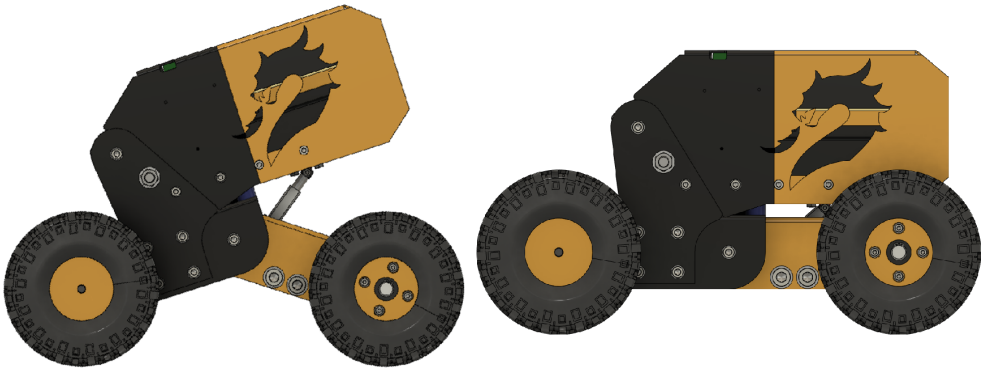

Figure 5.2. Final design of prototype chassis. Left: Scissor frame at maximum extension, Right: Standard position

5.1.2 Lifting

A kinematic linkage with a bucket attachment concept is implemented in accordance with the description in Section 4.3.2. The bucket is manufactured in 2 mm sheet metal and formed by bending, along with weldments to ensure a rigid construction. On the back wall of the bucket, holes for connecting the lifting arm assembly were made, as well as mounting holes for the bucket sensor fixture. The sensor mount is 3D-printed and offers protection for the proximity sensor used for detecting loads. Furthermore, the linkage includes two linear actuators: one lifting actuator that is mounted in parallel with the chassis and one tilting actuator which is mounted in parallel with the lifting arms, as is illustrated in Figure 5.3.

The lifting actuator controls the height of the bucket by driving the 4 mm thick metal linkage arms, while the tilting actuator controls the bucket angle. The linkage construction in this matter is called a TA-linkage, which in this case allows for reaching a height of approximately 230 mm and a bucket angle between -43° and 58°.

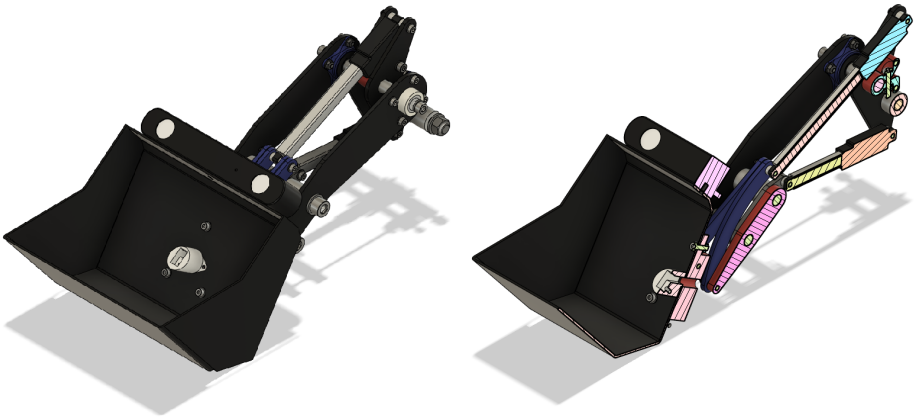

Figure 5.3. Final design of lifting subsystem, Left: Complete lifting subsystem, Right: Sectional view of lifting subsystem from mid plane

5.1.3 Steering

The previously proposed Double Ackermann steering mechanism was implemented, as illustrated in Figure 5.4. The left assembly represents the front section and the right assembly represents the rear section, where each servo motor in both cases controls the motor hubs via steering linkages. As seen in Figure 5.4, the design of the rear axle with a central pivot occupies space and therefore the rear steering does not mirror a symmetric Ackermann linkage. The steering angle is thus limited to the magnitude of 12 degrees for proper functioning and control purposes.

Figure 5.4. Final design of the steering subsystem, Left: Front steering, Right: Rear steering

5.1.4 Plastic Bale

Two different types of bales were constructed in the project. The main model, which was used throughout the project, was manufactured in plastic through the method of 3D printing. The second model was made in sheet metal, filled with Light Expanded Clay Aggregate (LECA). This filling was used to achieve the sought-for density, and resulted in a reasonable weight of the bale in relation to the wheel loader.

5.2 Analytical Model

This section describes the analytical models implemented on the prototype. These models are used in order to estimate a physical actuation through mathematical models. This allows for an easy feedback on the actuations performed by the prototype, as well as a physical meaning of the actuator values, such as degrees and distance.

5.2.1 Vehicle Position Model

The implementation of double Ackermann steering required that both the front and rear wheels could steer. A simple way of modeling a vehicle is through the use of a bicycle model. Normally, such a model would steer with one wheel and have the other wheel fixed. However, for a symmetric double Ackermann steered vehicle, both wheels have a degree of freedom. Therefore, a two Degree of Freedom (DoF) bicycle model was used to model the vehicle, see Figure 5.5. The model defines a global coordinate system (X, Y) with the origin at O, and a local reference system (x, y) fixed to the centre of the vehicle o. The local reference system is offset by an angle Ψ and a vector $\mathbf{r}_{Oo}$. The steering angles of the wheels are denoted as δ_1 and δ_2 and the vehicle turns around the centre point C at a radius R away from this point, where R refers to the radius to the point in the local reference system that moves purely in the x direction (given there is no wheel slip). The distance to this point, from the center of the rear wheel is denoted as q, and the wheel base of the vehicle as L.

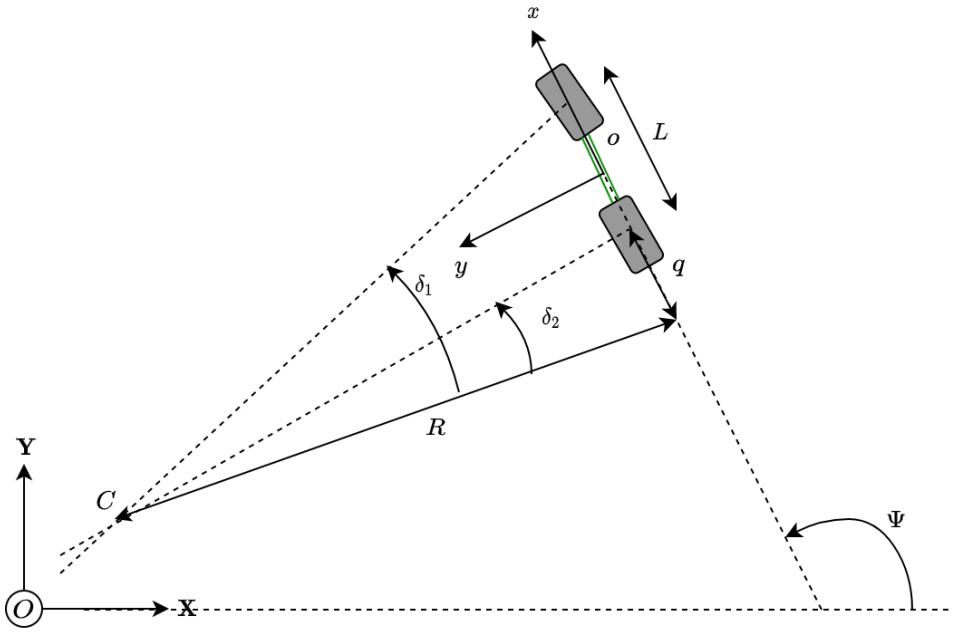

Figure 5.5. Bicycle model with 2 degrees of freedom

Using trigonometry, the following equation could be derived:

$$\dot{\Psi} = \frac{v_x}{R} = \frac{v_x(\tan\delta_1 - \tan\delta_2)}{L}.$$

The velocity of the vehicle could further be described with

$$v^2 = v_x^2 + v_y^2,$$

where v is the total speed of the vehicle, and the velocity vector being $\dot{\mathbf{r}}_{Oo} = \begin{bmatrix} v_x \\ v_y \end{bmatrix}_{x,y}$.

Each component of this vector was derived to

$$v_x = \frac{2}{\sqrt{(\tan\delta_1 + \tan\delta_2)^2 + 4}} v,$$

$$v_y = \frac{\tan\delta_1 + \tan\delta_2}{\sqrt{(\tan\delta_1 + \tan\delta_2)^2 + 4}} v,$$

with the model treated as kinematic. It is desired though, to describe the vehicle's velocity in the global coordinate system. To do that, a rotation of the vector is required. This can be done using the rotation matrix

$$\mathbf{R}(\Psi) = \begin{bmatrix} \cos\Psi & -\sin\Psi \\ \sin\Psi & \cos\Psi \end{bmatrix}$$

multiplied with the velocity in respect to the local coordinate system

$$\dot{\mathbf{r}}_{Oo} = \mathbf{R}(\Psi) \begin{bmatrix} v_x \\ v_y \end{bmatrix}.$$

Now, a system of differential equations that can be solved numerically has been obtained, given an initial condition Ψ_0, X_0, and Y_0. The simplest way of doing this is through the use of the Euler step method

$$\Psi[n] = \Psi[n-1] + \dot{\Psi}[n-1]\Delta t,$$

$$\mathbf{r}_{Oo}[n] = \mathbf{r}_{Oo}[n-1] + \dot{\mathbf{r}}_{Oo}[n-1]\Delta t,$$

where $\Delta t = t[n] - t[n-1]$ is the time step between each iteration.

5.2.2 Generalised Ackermann Steering Model

For any Ackermann steering design, it is beneficial to place a reference system local to the vehicle. The origin of this reference system was placed at the point, which is at the lateral centre of the vehicle, that can only move in the vehicle's longitudinal direction (given there is no lateral wheel slip). This point would be at the centre of the rear axle for a normal Ackermann-steered vehicle with a fix rear axle. For a symmetric double-Ackermann steered vehicle, it would instead be in the very centre of

the vehicle. Since the wheel loader has zero camber and caster, only two dimensions were considered in the model (the $X - Y$-plane). To avoid repetitive mathematical derivations, a generalised Ackermann model was made. The resulting equation of this model could then be used to express the wheel steering angle of all four wheels, given the vehicle dimensions and assuming perfect Ackermann geometry. The model is shown in Figure 5.6 and its goal was to obtain the following:

1. Wheel angle as a function of steering angle for an arbitrary wheel, given its relative coordinate

2. Wheel angular velocity as a function of steering angular velocity and the above

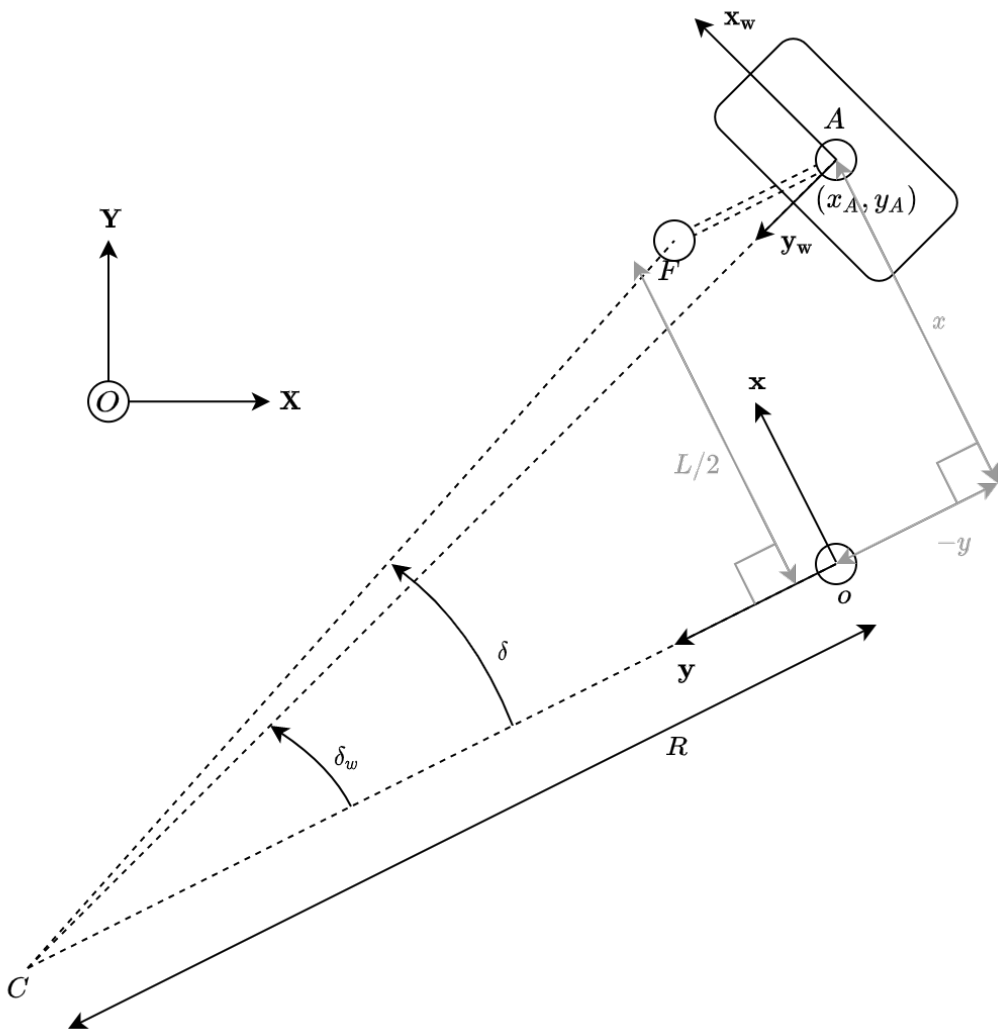

Figure 5.6. Geometry of a generalised Ackermann-steering model

The point C denotes the turning centre of the vehicle and the sought wheel steering angle is denoted as δ_w. R is the turning radius and (x_A, y_A) the wheel offset from the vehicle's centre o to the wheel centre A, which is defined such that the lateral axis y always intersecting with C. The model shows a left turn, but for a right turn, the steering angle δ (defined as the theoretical angle of a wheel at the centre of the front axle F) is negative, and for straight driving, $\delta = 0$. For ideal Ackermann, the wheel is assumed to always roll around the turning centre C. Firstly, the turn radius and wheel steering angle can be described using trigonometry:

$$\tan\delta = \frac{L}{2R} \Rightarrow R = \frac{L}{2\tan\delta},$$

$$\tan\delta_w = \frac{x_A}{R - y_A} = \frac{x_A}{\frac{L}{2\tan\delta} - y_A} = \frac{2x_A\tan\delta}{L - 2y_A\tan\delta} \Rightarrow$$

$$\delta_w = \arctan(\frac{2x_A\tan\delta}{L - 2y_A\tan\delta}).$$

Secondly, with the help of the chain rule and division rule, the time derivative could be derived to the following:

$$\dot{\delta}_w = \frac{2x_A L}{\cos^2\delta((L - 2y_A\tan\delta)^2 + (2y_A\tan\delta)^2)} \times \dot{\delta}.$$

5.2.3 Wheel Speed Model

The problem with the previously described model is that the wheels do not actually turn around their own centre points. There is an offset b, along with the wheel axle, of which the the wheel is actually turning. This however, does not affect the steering angle (for Ackermann steering) of the wheels if the point A is considered to be the turning point of the wheel, since the wheel axle still aligns with C. It does however have an effect on the speed of the wheels when driving. To obtain the true wheel speed, given any input (steering angle, angular velocity, and longitudinal velocity), the steering linkage of the vehicle was modeled, as seen in Figure 5.7. In this model, the wheel's centre point is at B, which is at a distance b away from A. Having this model was necessary for accommodating for the inner and outer wheels rotating with different speeds during cornering, when there is a change in the steering angle, or when both are happening at the same time.

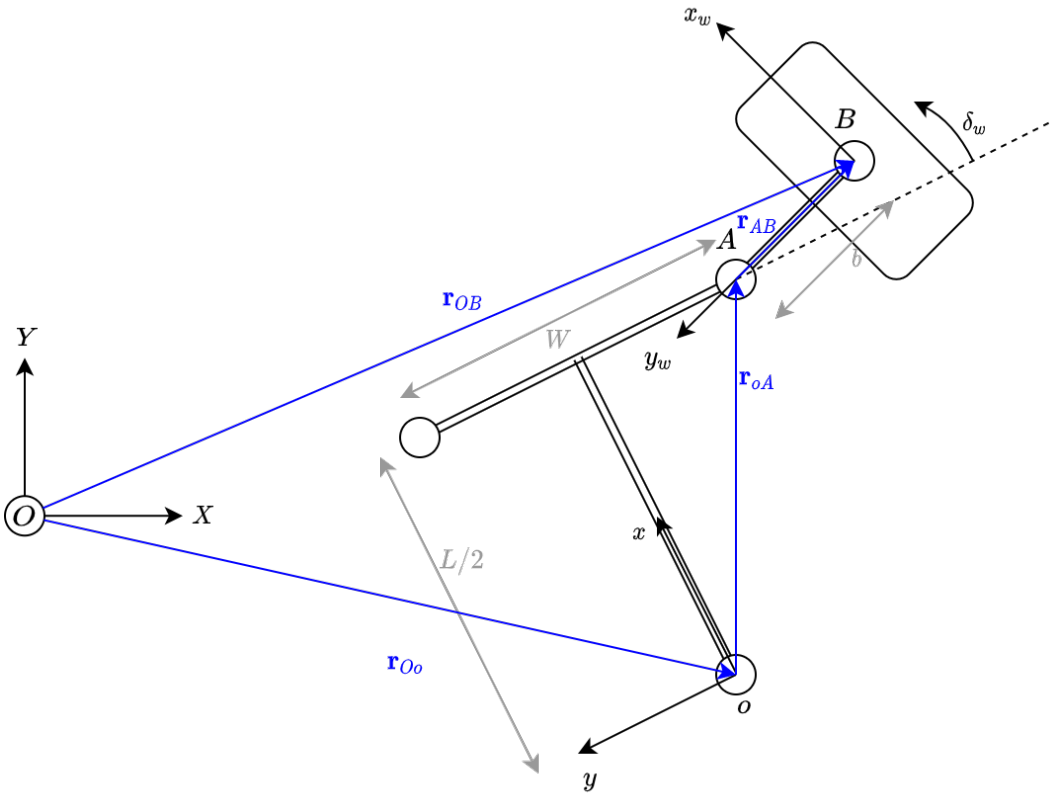

Figure 5.7. Geometry of the steering linkage of the vehicle, modeled in respect to a global coordinate system

The goals of this model was to obtain the following:

1. Position of the wheel, depending on the position and heading of the vehicle and steering angle of the wheel
2. The speeds of the wheel projected on the wheel's longitudinal axis, depending on:
   - Position and speed of the vehicle
   - Angular velocity of the vehicle, in the global reference system
   - The steering angle and angular velocity of the wheel

Using the vectors defined in the model, the global position vector of the wheel $\mathbf{r}_{OB}$ can be defined using vector algebra:

$$\mathbf{r}_{OB} = \mathbf{r}_{Oo} + \mathbf{r}_{oA} + \mathbf{r}_{AB}.$$

What is sought is the time derivative of $\mathbf{r}_{OB}$, which can be obtained after deriving the time derivative or the other vectors, using the following equation:

$$\dot{\mathbf{r}}_{OB} = \dot{\mathbf{r}}_{Oo} + \dot{\mathbf{r}}_{oA} + \dot{\mathbf{r}}_{AB}.$$

Then, this vector can be projected onto the wheel's x- and y-axis to obtain the projected speeds:

$$v_{xw} = \frac{\dot{\mathbf{r}}_{\mathbf{OB}} \cdot \mathbf{x}_w}{\|\mathbf{x}_w\|^2} = \dot{\mathbf{r}}_{OB}^T \mathbf{x}_w,$$
$$v_{yw} = \dot{\mathbf{r}}_{OB}^T \mathbf{y}_w,$$

where $\mathbf{x}_w$ and $\mathbf{y}_w$ are unit vectors that align with the wheel's local reference system. When ideal Ackermann steering is assumed, $v_{yw} = 0$ is expected at all times (this criteria was also used to verify the model). After doing all the steps mentioned above analytically, the following equation was obtained:

$$v_{xw} = v_y \sin\delta_w + v_x \cos\delta_w + (x_A \sin\delta_w + y_A \cos\delta_w + b)\dot{\Psi} + b\dot{\delta}_w. \tag{5.1}$$

With ideal Ackermann steering, $v_y = 0$ can be assumed, but this term was kept in the equation due to the implementation of crab steering.

5.2.4 Tool Actuation Model

The tool, i.e. the bucket and lifting mechanism, was modeled in the vehicle's $x-z$-plane using the illustration in Figure 5.8. Trigonometric relationships between the dimensions of the joints $(a, b, c, ..., g)$, connection points $(A - F,\ P_1, P_2, P_3)$, and variables $(L_1, L_2, \alpha, h, \beta)$ were identified, described mathematically, and then used to derive a relationship between $[\alpha, h]$ and $[L_1, L_2]$. In the pair of input variables, α is the bucket angle and h the height of the bucket tip above ground. The output variables are the lengths of the linear actuators, L_1 and L_2. The objective of this model was to find an analytic expression for L_1 and L_2, with respect to α and h. However, it was quickly realised that the mathematical problem was too complex, resulting in very long mathematical expressions. Therefore, MATLAB's symbolic solver was used to solve the equations.

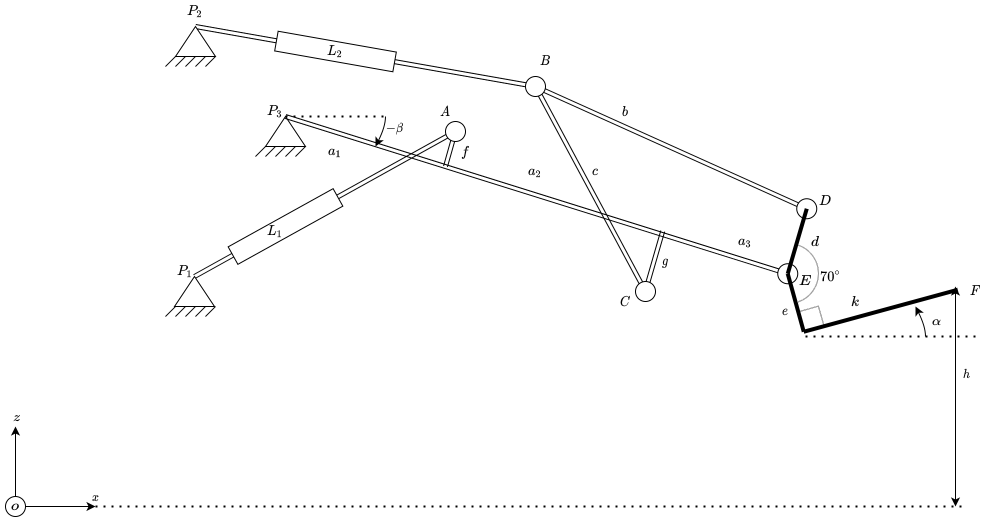

Figure 5.8. Model of the lifting mechanism and the bucket

With the analytical expression obtained, multiple values of α and h were tested to find the region of which the actuator lengths L_1 and L_2 are feasible, i.e. within their limited ranges. Given the specifications, these ranges were $L_1 \in [118, 168]$ mm and $L_2 \in [168, 268]$ mm. The found region is shown in Figure 5.9.

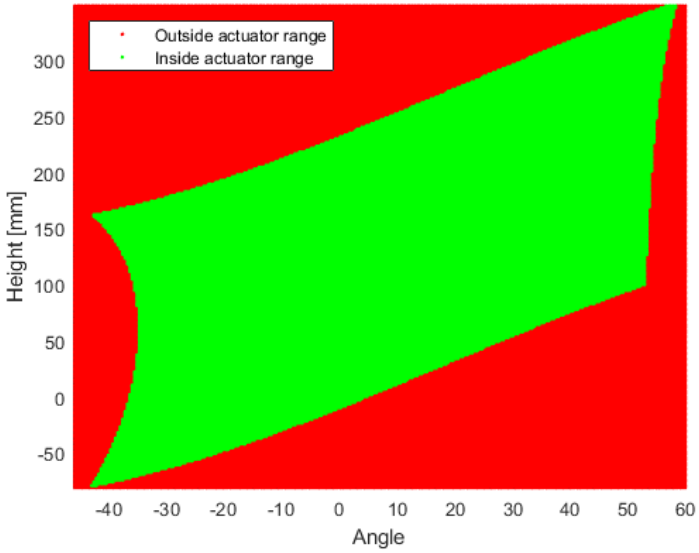

Figure 5.9. Graphical map of the feasible bucket angle and height region

With the aim of implementing the model in the embedded controller on-board with the prototype, it was realised that the length of the expression was problematic for two reasons. Firstly, it would be a tedious and time consuming task to simplify and convert the expression from one programming language to another. Secondly, such a long expression is a computationally heavy task for the limited 32-bit ARM processor used, since it needs to function in real time. Therefore, attempts were made to see if a polynomial best-fit approximation, conducted by calculating points from the mathematical model, would be a viable option.

Using the data points within the feasible region described in Figure 5.9, polynomial best-fit functions ($f_1 : \mathbb{R}^2 \to \mathbb{R}$, $f_2 : \mathbb{R}^2 \to \mathbb{R}$) of degree two and three were used. The errors of each point in the feasible region was then calculated to assess whether the degree of the best-fit polynomials were sufficient. The two histograms shown in Figure 5.10 and 5.11 show that the majority of the points have an error which is less than 1 mm which, considering the imperfections in the physical assembly, are tolerable.

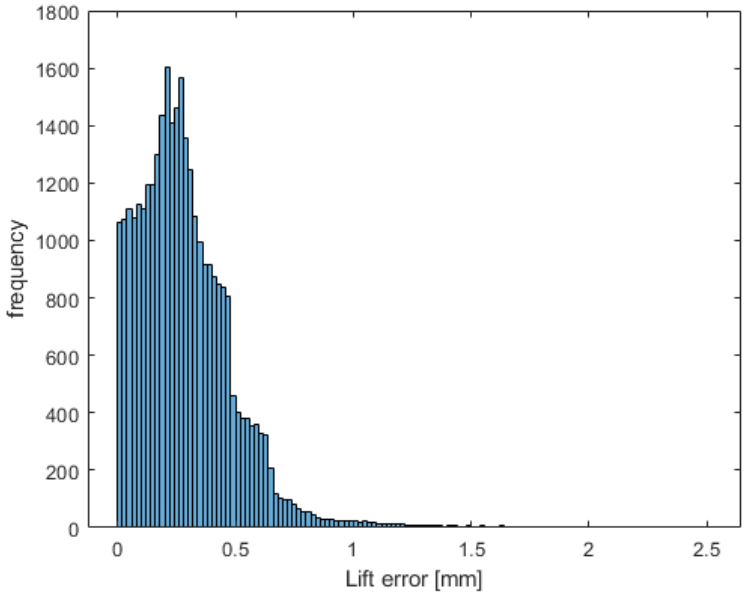

Figure 5.10. Histogram of the error distribution of the polynomial approximation of the length of the lift actuator L_1

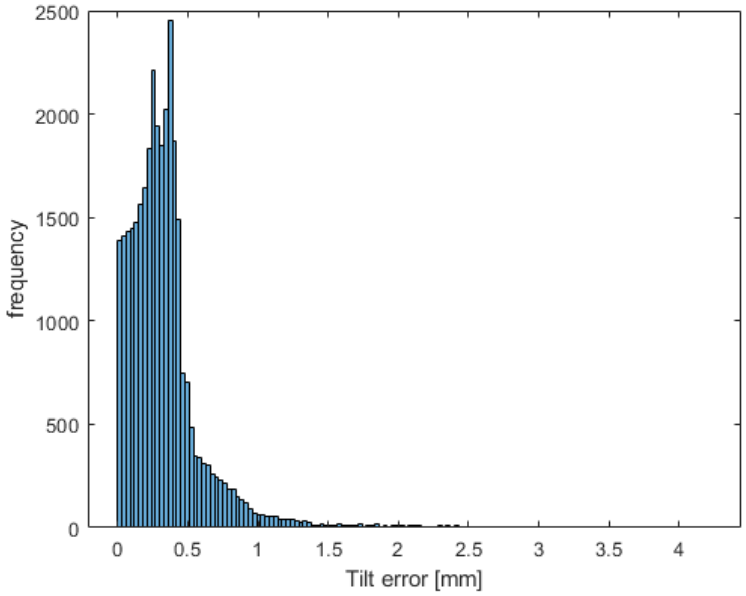

Figure 5.11. Histogram of the error distribution of the polynomial approximation of the length of the tilt actuator L_2

5.2.5 Scissor Joint Actuation Model

The scissor joint mechanism was modelled to obtain the scissor actuator length as a function of the scissor angle. This was done by collecting seven data points from the CAD assembly and using a linear best-fit approximation to find a mathematical function. Due to the lack of positional feedback of this actuator, it was not possible to accurately control its position. Therefore, it did not make sense to implement a more sophisticated model. The linear approximation function, together with the collected data points used, are shown in Figure 5.12. The linear approximation was then offset slightly to intersect with the data point at 0°. This offset linear function was the function that was implemented in the embedded controller.

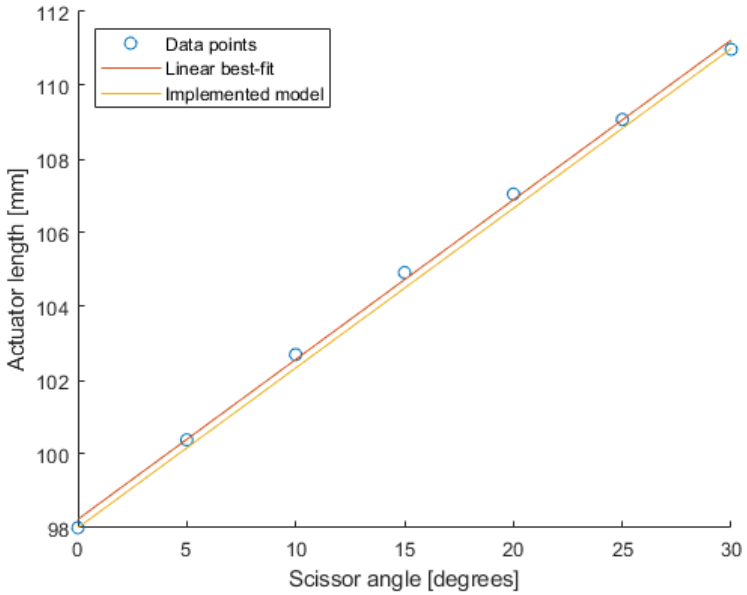

Figure 5.12. Graph showing the approximated model used for controlling the scissor joint

5.3 Hardware Implementation

This section describes the selection of hardware and electrical architecture based on the functional design.

5.3.1 Functional Architecture

The wheel loader is an integrated system of multiple collaborating subsystems, in which the inputs of interest to these systems are operator control inputs and environmental factors (distances from near objects). Outputs of interest include torque on the wheels, forces on the environment, and status updates. The functionality of the systems is to provide specific outputs depending on the inputs. In this case, the

wheel loader follows the operator's commands explicitly and informs the operator of the position of bales if it detects any.

In order to provide all these functions, the system must be designed to support them. An architecture of the developed system is shown in Figure C.1 in Appendix C. It shows the functional elements of the system and what kind of information that flows between them. The text in parentheses in the figure shows how the implementation was done physically. Foremost, it was decided to separate the *Robot Operating System (ROS)* and the *Vehicle Control Unit (VCU)* into two different subsystems. This choice was made to provide parallelism, both in the system itself and in the development. The information flow between the ROS and the VCU consists of control commands and status updates. The system can have two operators, a manual operator (i.e. a human) or an autonomous operator (i.e. an artificial neural network), the latter of which can be implemented inside the ROS subsystem. The whole system is powered by a single energy source. Since the ROS and the VCU are functionally independent, it is also possible to assign them to different hardware, of which either of them can be changed later on, without the need of changing the rest of the system. This feature makes the system more future-proof and adaptable.

5.3.2 Choice of Hardware

The most critical part of the system was considered to be the image recognition system within the ROS. It is known that image recognition in real time requires a significant amount of computational power. Therefore, a powerful embedded computer that can support the ROS, available on the commercial market and within the budget, was selected: the Nvidia Jetson AGX Xavier. For the VCU implementation, there were multiple options that were considered before settling for the STM32 Nucleo-L476RG as hardware. This was chosen because the STM32 framework was favourable and this particular variant has a suitable form factor and sufficient hardware peripherals, GPIOs (General Purpose Input/Output), flash memory, RAM (Random Access Memory), and clock frequency for the project's needs. For remote control of the wheel loader prototype, a Bluetooth transceiver module was used, namely the HC05. This module was chosen since it was easily accessible on the consumer market and because Bluetooth technology opens up the possibility to control the prototype wirelessly, using a smartphone or laptop. The choice of remaining hardware was limited by market availability in general, and therefore, the design of the wheel loader was adapted to that hardware, rather than vice versa. This method of hardware selection was both time saving and efficient, since it eliminated the need for redesigning the wheel loader to fit new parts and the time consuming task to search for scarce products.

Due to its built-in features, the choice of the camera was a ZED2 stereo camera. Such features include both depth perception as well as a built-in IMU. The depth perception was concluded to be well-needed since the depth values had to be ex-

tracted to determine bale distance from the prototype. The need of an IMU was significant since it was favourable to know the camera's orientation with respect to both movement and yaw.

5.3.3 Electrical Architecture

The interconnections between the hardware elements of the system was planned out through the use of a graphical tool. This plan was visualised through an electrical architecture shown in Figure 5.13. The core of this architecture is the electrical distribution board, which distributes all the signals between the hardware elements. This component, in the form of a PCB (Printed Circuit Board), greatly reduces the amount of wires under the hood of the wheel loader, and thus simplifies the assembly. These were the main reasons why such hardware was chosen. Furthermore, a PCB is also able to embed more than just wires, but also voltage regulators, motor drivers, filter capacitors, and protective fuses and diodes, in a space efficient manner. This board contains labelled connectors that the cables would connect to, making it easier to replace a single hardware element such as an actuator or a sensor.

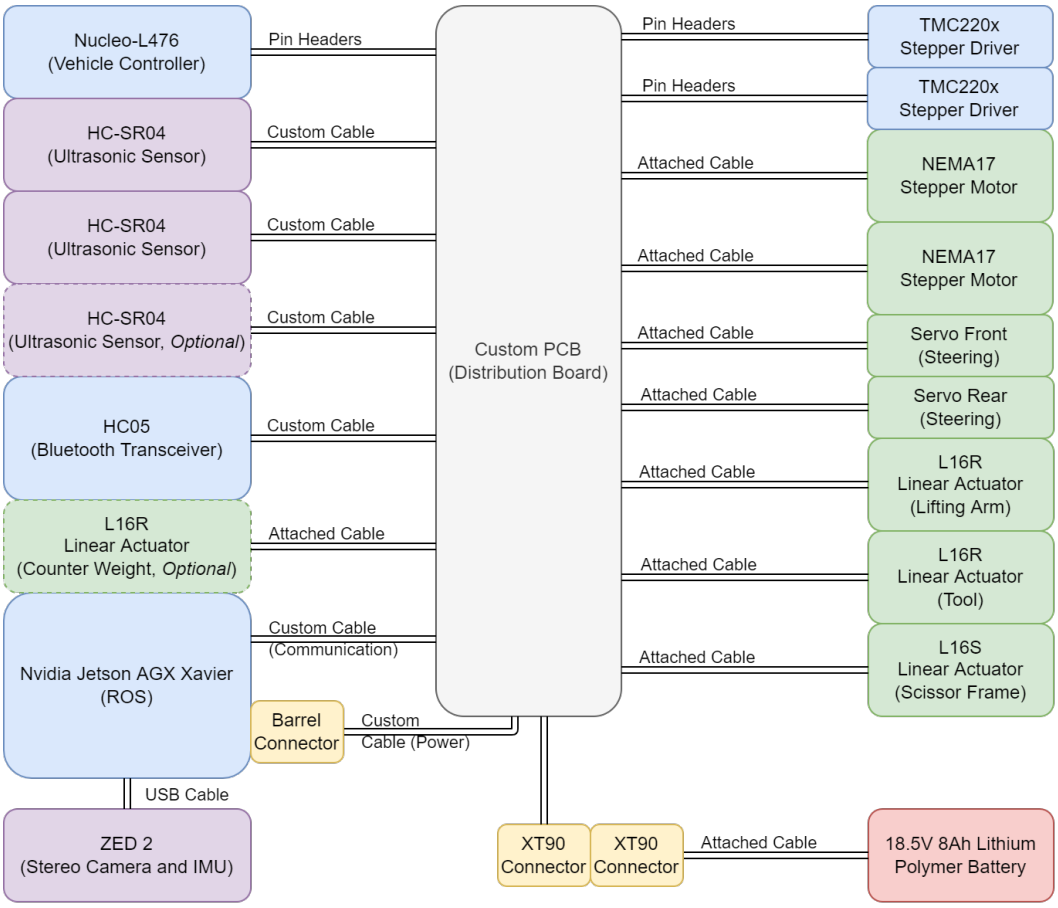

Figure 5.13. Electrical architecture

5.4 Embedded Systems

This section introduces how the functionality of the wheel loader was implemented in the embedded controller.

5.4.1 Firmware

The firmware written on the STM32L476RG microcontroller was divided into 5 abstraction layers, as shown in Figure 5.14 with some of the files that belong to each layer. The *Hardware Abstraction Layer*, or *HAL* in short, is the lowest abstraction layer. This layer makes an abstraction of the internal hardware inside the microcontroller, such as the registers, clocks, timers, GPIO ports, and all other peripherals. It also abstracts the configuration of the hardware using explanatory functions and macros. All the code for this layer was automatically generated by the STM32 CUBE MX software, after configuring an *ioc* extension file using the software's interface.

On top of the HAL, another layer, which was chosen to be referred to as the *Process Abstraction Layer*, was used to abstract the internal processes of the microcontroller. These processes refer to the control of and callbacks from the peripherals and include the timers, UART, DMA (Direct Memory Access), interrupts, ADC, and GPIO. These all happen in the background and are independent of the CPU. This layer contains interfaces for controlling these background processes with code that will run on the CPU.

On top of the process layer, there is an abstraction layer for all the actuators and sensors that the firmware interacts with. This layer was chosen to be referred to as the *Attribute Abstraction Layer*. In this layer, every actuator and sensor is abstracted as an object, using data structures in C, whose handles are passed into functions, together with other parameters. This makes it easier for the programmer to keep track of which attribute that is handled.

The next abstraction layer, that will be referred to as the *Platform Abstraction Layer*, abstracts the platform itself (the vehicle in this case), the communication to it, as well as the control of it. Again, using the same programming pattern, the wheel loader is an instance of a data structure called "vehicle", containing all the states of the vehicle, its sensors, and actuators.

The final layer is the *Application Abstraction Layer*. In this layer, application threads containing application code can run in parallel with the rest of the code. The application program decides what the platform does, depending on states, user inputs, and/or sensor inputs. Though it is possible to implement a real-time operating system, it did not become relevant since the application in this project only would require a single thread. Instead, the application function is triggered by a timer

interrupt every 100 ms.

Apart from the five abstraction layers, there is a gray zone for some codes that need to cover more than a single abstraction layer. This code is specifically the setup code where all the abstractions are defined and initiated in the main thread. The main thread is where the program starts and the configuration file contains all the tunable parameters.

Each abstraction layer uses only the interfaces from one layer below (if there are any) and/or calls back to a function at some layer above. There is an exception though for the communication code, because it uses the interfaces from the UART code and some HAL interfaces.

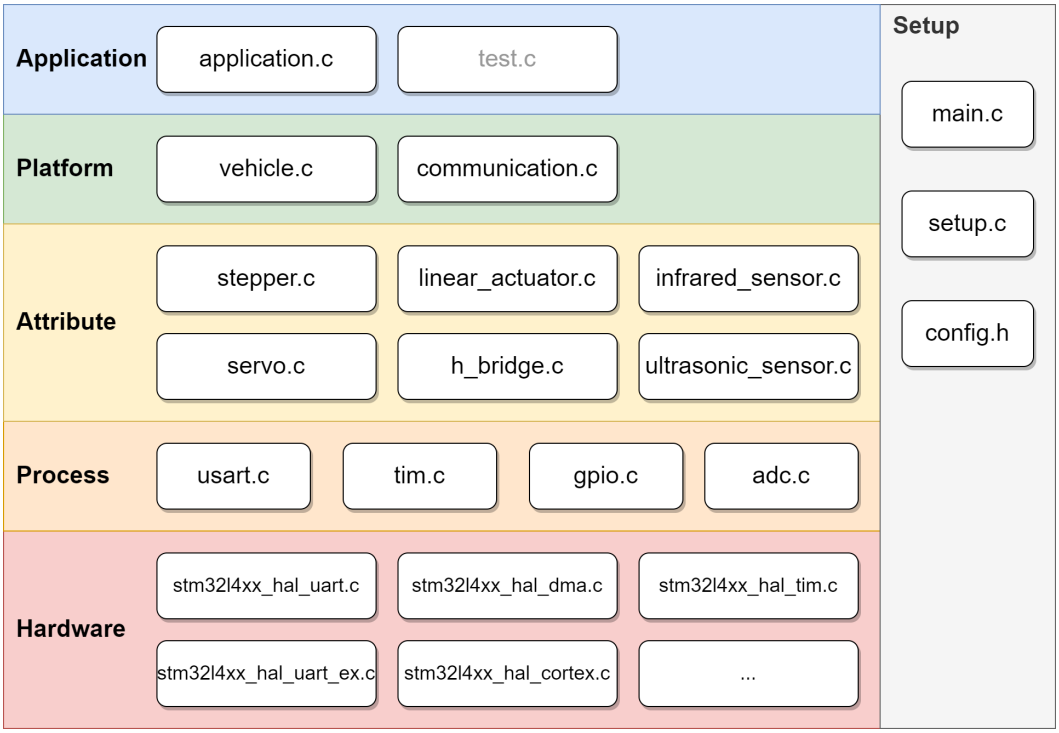

Figure 5.14. The abstraction layers for the firmware

This abstraction layer based code structure was chosen since every implementation in the firmware would require a lot code. Without any planned structure, the code could quickly become complicated. Furthermore, using a design pattern remnant to object-oriented programming, it becomes both easier to understand and easier to keep track of the abstractions. This object-like pattern was inherited from the HAL code design pattern, to keep the firmware code consistent.

5.4.2 Application

The application part of the firmware was designed with safety in mind. A state machine was firstly designed, see Figure 5.15, which was then translated into C code, using 'if' logic.

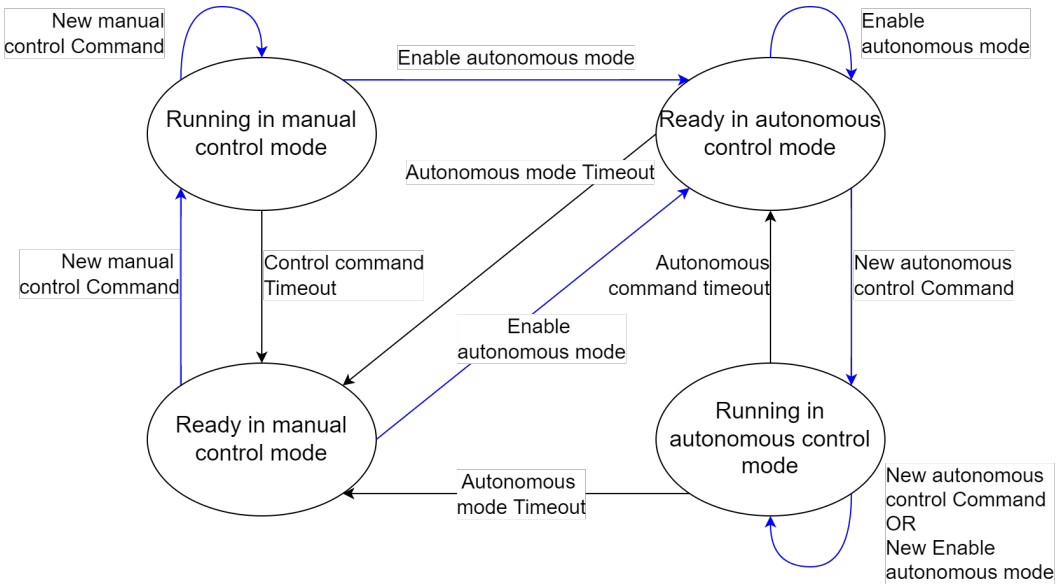

Figure 5.15. State machine for the application logic

By default, the application is in the *Ready in manual mode* state. This means that the vehicle is stopped and waiting for a command. Given a manual control command, it will enter the *Running in manual mode* state. The rest of the state machine works in the same fashion. The state is stored inside a global variable and each transition, from one state to another, is implemented using an 'if'-statement. The transitions marked as blue arrows are transitions that are controlled via operator actions. The rest, marked as black arrows, happen automatically in the firmware.

5.4.3 Communication

The communication to the wheel loader platform happens via serial communication links. These communication links were chosen due to the UARTs that were readily available on both the microcontroller and the main processing unit. The protocol is very simplistic, user friendly, and widely used. Therefore, it was easy to find a Bluetooth receiver which uses the same protocol. The limitation of this UART protocol is that it only works between two peers. Luckily, the STM32L476RG has 5 UARTs that can be configured and used individually, giving it the possibility to communicate with multiple devices. The UARTs that were used were USART1 for the HC-05 Bluetooth transceiver, USART2 for the USB communication (mainly for debugging purposes), and USART3 for the AGX Xavier.

There are several types of messages that are useful for interfacing with the wheel loader, but the serial protocol works only up to the data link layer (in the OSI, or Open System Interconnection, model), without any kind of addressing. The data is divided into a stream of bytes which are individually packaged into frames and sent one-by-one. There is no way for the firmware to distinguish one message from another, solely based on the data. Hence, it was necessary to have a protocol on top of the serial protocol. For this application, there were seven different messages that would be received by the platform, and four that would be sent from the platform. Only one message is presented in both directions, which is a communication test message that contains a string of up to 63 ASCII characters. All the messages are shown in Table 5.1.

Table 5.1. Messages that are used to interface with the firmware

Message	Type	ID	Length (bytes)	Direction
Autonomous System Control Command	Periodic	0b011	6	RX
Test Message	Event	0b11x	0...63	RX
Enable Autonomous Mode	Periodic	0b010	0	RX
Manual Control Command	Periodic	0b001	6	RX
Request Remote Execution Command	Event	0b101	1	RX
Request Vehicle Status	Event	0b100	0	RX
Set Coordinate System	Event	0b000	12	RX
Test Message Reply	Event	0b11	0...63	TX
Error Message	Event	0b00	1	TX
Remote Execution Command	Event	0b10	1	TX
Vehicle Status	Periodic	0b01	23	TX

All messages were sent over serial, where TX are the messages being sent by the firmware and RX being the ones received. The message identifier, or operation code (OPCODE) , is sent in the first byte of the frame together with additional metadata. The following bytes contain the data payload, see Table 5.2.

Table 5.2. Description of the message frames

Byte	0								1								2								...
Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	...
Test message frame	OPCODE								Data Length								Data[0]								...
Command frame (RX)	OPCODE								Sequence Numbe								Data[0]								...
Reply frame (TX)	OPCODE								Sequence Number								Data[0]								...
																	Data[1]								

On the firmware side, the UART peripherals receiving module is configured in DMA mode, meaning that they will have direct access to a buffer in the memory. Additionally, they are configured with 'idle interrupt', meaning that an interrupt routine is triggered when the communication line goes from busy to idle. This was chosen because, since each of the messages have different lengths, there is no way to know

how many bytes to wait for. Therefore, by triggering the interrupt whenever a full message is sent completely, it saves the firmware from several additional computations and the CPU from additional context switches. The full message is simply received in the background, with the help of the DMA, and the message is handled only once it has been fully received. The interrupt function is passed the length of the received message, as well as the handle to the UART that the message was delivered to. The function can then compare the message identifier and metadata with the length, validate the message, and decide what to do with the message using some simple logic.

5.4.4 Vehicle Control

As mentioned, the controlling interface of the vehicle is through serial communication, in which the control messages contain 6 bytes of data. The data bytes store the target velocity, steering angles (front and rear), bucket angle, bucket height, and the scissor joint angle respectively. These values are encoded into 8-bit values, ranging from 0 to 255, where 0 corresponds to the minimum value and 255 to the maximum. This prevents the operator from ever inserting a value which is beyond the mechanical limits of the wheel loader, since these limits are hard-coded in the firmware.

Once the message has been validated, the data from the buffer is copied and added to a message queue. The application consumes these messages from the queue and tells the platform what to do, based on their data and the current state of the vehicle. It does that by changing target values for the vehicle's velocity, steering angle, etc., through the use of a global 'wheel loader' abstraction. In the vehicle control loop, which runs in the background 50 times per second, the real values of the vehicle's states (velocity, steering angle, etc.) are ramped towards their set targets. Servo angles, motor speeds, and actuator lengths are calculated in this loop, based on mathematical models which were derived from the mechanical geometries of the prototype, see Section 5.2. The control loop also simulates how the wheel loader moves in a global coordinate system, using the kinematic bicycle model described in Subsection 5.2.1.

The real states of the vehicle (including sensor readings), together with the estimated position and heading of the wheel loader are replied to every command message received. This message is the 'Vehicle Status Message' that can be seen in Table 5.1.

5.4.5 Stepper Motor Control

The stepper motors are driven by individual stepper motor drivers of the type TMC220x, which only require a small set of signals. Most of the signals were not necessary to control in real time, and therefore were hard-wired to ground or 3.3

Volts. The signals that were chosen to be controlled were the enable signal, step signal, and direction signal. The enable signal simply enables current to run through the motor coils, when set low. This is useful to control because even when the motors are not spinning, some current is passing through the motors in order to maintain their position, as long as the motor drivers are enabled. By disabling the motors when they are not moving, the power consumption of the prototype can be reduced. The direction signal is used to set the direction of the motor and the step signal is used to perform a rotation step in that direction. As the motor has 200 steps per revolution, one step corresponds to 1.8° of rotation. Such a step is what is called a ”full” step, and these steps can be further reduced into ”micro” steps. However, because micro-stepping was not necessary for this application, it will not be investigated further. To get the best use of the stepper motors, a simple state machine was used to control them, see Figure 5.16.

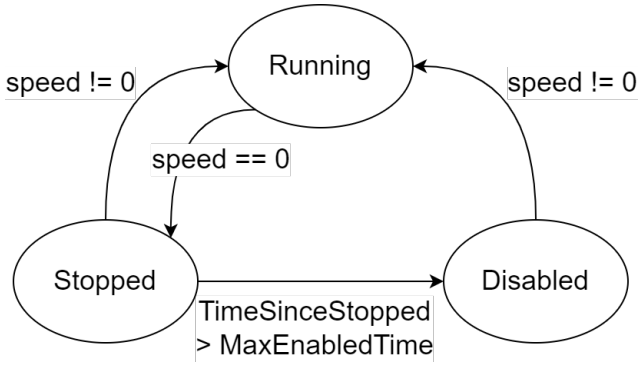

Figure 5.16. State machine for control of the stepper motors

This state machine consists of three states, namely Running, Stopped, and Disabled. The step signal is generated through timer 2 and 5 (for each motor respectively), using Pulse Width Modulation (PWM) with a duty cycle of 50% and a variable frequency, depending on the speed. The stopped state is necessary because, when the motor speed is set to zero, the frequency is set to zero and the period would be $\frac{1}{0}$, which is undefined. The way that the period is set is by writing a value to a 32 bit register which cannot take values that are too big or undefined. Thus, instead of writing such values to the register, the timer is stopped, preventing any PWM signal on the step wire. The timer is started again once the motor speed is set to a non-zero value, returning to the running state. After being in the stopped state for some time, the state machine automatically switches to the disabled state, disabling the motors to save power. Since the motors need to be able to run at different speeds, it was necessary to use individual timers for controlling them.

5.4.6 Servo Motor and Linear Actuator Control

The servos and linear actuators (except for the scissor joint actuator) have their own embedded drivers with feedback control systems. Their absolute positions are controlled by a PWM signal, which has a pulse width between 1 ms and 2 ms and a frequency of 50 Hz to 200 Hz. The position reference is signalled through the width of the pulse. A callback function, triggered by a timer interrupt, is used to ramp the pulse width from the current value to the target value (which is set by the operator through the communication link). The ramping is necessary for the linear actuators because it appeared that they would not operate if the reference position is too far from their actual position. Since their extension and retraction speed is limited, the change of the reference signal must not be any faster. The servo motors had the same limitations, and thus the same type of ramping was used. The linear actuators also needed to be fetched, by sweeping the reference signal over the whole range and then ramp back to the target position, at startup. Without doing that, they would refuse to move entirely. This problem however, was not present for the servo motors, which only required a constant PWM signal for a few seconds during startup to be initialised, and they would slowly move towards the reference position.

The scissor joint actuator was also a linear actuator but without an embedded feedback control system. This type was used here due to the lack of physical space in the construction. This actuator featured a pair of limit switches, protecting it from extending or retracting beyond its own mechanical limits. To control this actuator, an H-bridge of type ZXBM5210 was embedded on the distribution PCB. This H-bridge has two inputs, Forward and Reverse. When Forward is set high and Reverse is set low, the linear actuator would extend, and vice versa. To get more control over the speed of the actuator, the hardware timer4 was used, with one channel connected to Forward and one connected to Reverse in PWM mode. Giving the channels opposite polarity, while using the same duty cycle, resulted in alternating PWM signals which drove the H-bridge in fast-decay mode. The velocity of the actuator was then controlled by changing the duty cycle from 0% (full retraction speed) to 100% (full extension speed), where 50% duty cycle makes the actuator stand still.

5.4.7 Sensors

The vehicle has an infrared sensor in the bucket and two ultrasonic sensors on the rear. The infrared sensor, based on QRE1113GR, has an analog current signal which depends on the amount of infrared light that it senses. This signal's current passes through a pair of resistors, inducing a voltage difference that is measured using a 12 bit ADC. The full schematic can be seen in Figure 5.17. The A/D conversion is implemented in polling mode, since it was easier to set up and fast enough for this implementation. The value from the conversion is then sent directly to the operator via the communication link.

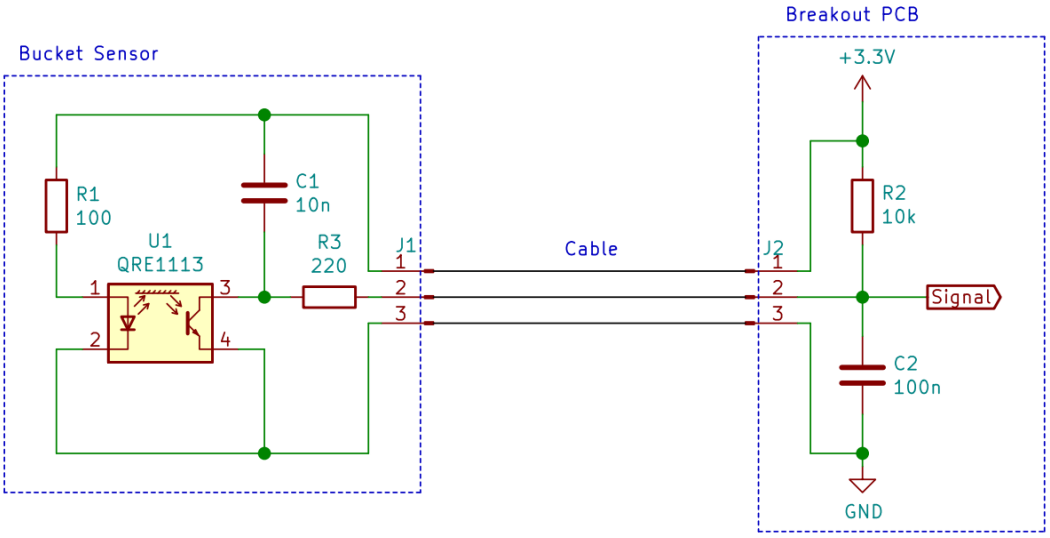

Figure 5.17. The full electrical schematic for the bucket sensor

The ultrasonic sensors are of type HC-SR04. This sensor is interacted with using a digital trigger signal which starts a new measurement given a 10 μs pulse. The measured value is then signalled via the echo signal, which sends a pulse whose length is the measured reflection time. The TRIG signal is generated using a timer in PWM mode. The same timer can be used to measure the length of the ECHO pulse using a separate timer channel in input capture compare mode. The timer channel captures the timer counter value on every rising and falling edge and triggers an interrupt function. Simple logic is then used to figure out which capture value that corresponds to which edge of the pulse, and the difference of the values is calculated, giving the echo time of the ultrasonic signal. The distance is then calculated using the assumption that the speed of sound is 340 m/s. Both ultrasonic sensors are implemented using a single hardware timer and they share the same trigger signal. Although this could cause problems with the sensors interfering with each other, it did not cause any issues on this application.

5.4.8 Safety

Running an autonomous system on a wheel loader can have dangerous outcomes whenever something goes wrong. Therefore, safety was taken into account when designing the firmware. The first safety feature was the use of periodicity of control messages, that will cause the wheel loader to actuate. These messages have a limited time-to-live, meaning that after some time, the wheel loader will automatically stop, unless a new message has been received. The second safety feature is the ability to take over the control of the wheel loader. The wheel loader would either be in autonomous mode, listening to commands from the main processing unit, or in

manual mode, listening to commands from a manual operator. This switching is done by the manual operator by sending a periodic 'enable autonomous mode' message. As soon as this signal is removed, it will automatically switch back to manual mode, giving a manual operator a chance to stop the vehicle at any time. With these safety features in mind, both the communication messages and the application had to be designed accordingly.

5.5 Bluetooth Controller Application

An Android Bluetooth application was created to maintain easy control over the vehicle. It was developed to have manual control, switch the vehicle to autonomous mode, and read the state messages from the prototype. The purpose of developing this application was to have a remote control unit that every operator can use for further platform development and testing.

5.5.1 Environment and Structure

The application development was conducted in the Android Studio IDE (Integrated Development Environment) which is one of the most used IDE:s for this purpose. Applications are created with a base of a few major modules connected to work together. The Bluetooth application implemented in this project consists mainly of:

1. Java code that is responsible for the main functionalities. The application is built from four main classes:
    - **SelectDeviceActivity** which is called to make a selection of which Bluetooth device the application device should be paired to.
    - **DeviceInfoModel** which is responsible for extracting information about the device that the application is trying to connect to.
    - **DeviceListAdapter** which is responsible for making a list of devices to connect to. Only devices that have already been paired are listed.
    - **MainActivity** which is responsible for all application functionalities that are connected to the main layout and the items placed there.
2. XML files responsible for layouts and values prescribed to them, as well as an Android manifest which holds application permissions for accessing the phone's resources.
3. Gradle files responsible for syncing and compiling the application code.

5.5.2 Functionality

The main functionality of the application is located in the MainActivity class. After receiving input from all subclasses, the handler function manages the workflow.

After turning on the application, all the control buttons as well as the manual/autonomous switch, seen in Figure 5.18, are disabled, meaning that only the connect button is available. After pressing this button, the list of previously paired devices is presented. After choosing one of them, the Bluetooth connection socket is established and all buttons and the switch are enabled. The task of the different buttons is to change the value of a prescribed parameter while it is being pressed, and the switch is responsible for changing from sending manual control messages to passing the command to run the autonomous ready state. A detailed description is provided in Table 5.1.

The timer in the application is responsible for scheduling tasks. Such tasks include sending control messages, sending actuation commands, and showing a message box to check the battery voltage. The scheduled tasks have different activation offsets and periods. The sending of control messages is managed by a separate thread. The parameters are sent periodically but pressing each button or switch changes the value of a specific parameter by a function inside the MainActivity class. Parameters that are sent are visible in a text box under the log tag, as can be seen in Figure 5.18. For a more detailed overview, the work logic of the application can be seen in Appendix E.

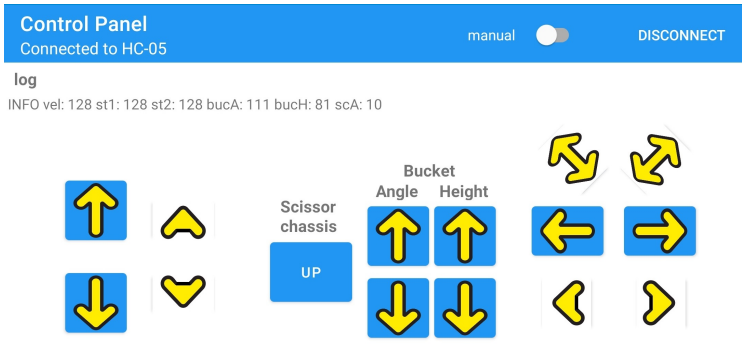

Figure 5.18. Screenshot of main Bluetooth application layout

5.6 Navigation Implementation

In order to evaluate the prototype and implement its navigation in a consistent way, a testing route and environment were designed. The test environment consists of a starting point, plastic bale model, and drop-off point, as seen in Figure 5.19. Here it can also be seen that the route is divided into both hard-coded and autonomous sections, which are described in more detail in Section 5.6.1.

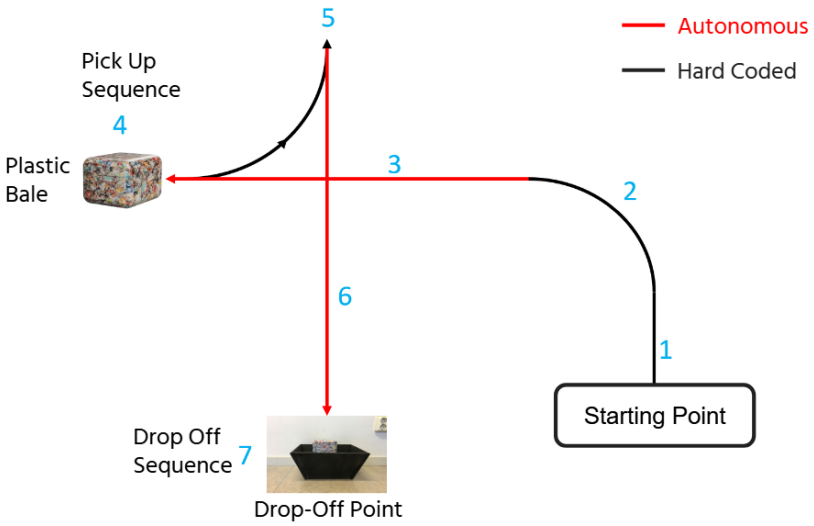

Figure 5.19. The testing environment and route

5.6.1 Test Route

The prototype begins the testing route with a hard-coded section, moving from the wheel loader's starting position until it is able to recognise a plastic bale. In Figure 5.19, it can be seen that this initial hard-coded path consists of driving straight a specified distance (Step 1) and thereafter a 90° turn to the left (Step 2). Once the prototype exits the turn and the plastic bale is identified, as described in Section 5.6.2, the navigation is switched to an autonomous mode in which the loader aligns itself to the bale (Step 3). This alignment process is outlined more thoroughly in Section 5.6.3. When the prototype recognises that it is a certain distance to the bale, it executes a predetermined pick-up sequence (Step 4). Once the proximity sensor in the bucket recognises that the bale has been loaded, the prototype reverses until it has achieved a certain orientation (Step 5). The loader then drives forward, while maintaining its orientation, until the bale at the drop-off point is recognised. Thereafter, the prototype once again becomes autonomous, aligning itself to the funnel, as described in Section 5.6.3 (Step 6). Finally, once the drop-off point is within reach, a drop-off sequence is performed to unload the plastic bale (Step 7).

During the hard-coded sections of the route, it is imperative that the prototype is aware of its orientation in space, is able to maintain that orientation when driving, and is aware of its driving distance. To achieve this, the quaternion orientation is extracted from the camera's IMU, which is relative to the camera's starting position. The correct heading for each hard-coded section can then be determined, and the prototype is programmed to maintain this orientation. This is done with a simple hysteresis controller with the IMU orientation as feedback. To calculate the driving distance of the prototype, a numerical model of the steering mechanism was used

and the command was sent from the ROS node. Note that there is no feedback from the implemented steering stepper motors.

5.6.2 Finding the Bale

From the computer vision software, YOLOv3, the program outputs a bounding box in the input image once there is a detection of a plastic bale. Since the ZED2 camera has a built-in depth sensing capability, it can measure the distance in each pixel. With this feature, combined with the information given from the bounding boxes which clarifies in which pixels the plastic bale is located, the wheel loader can calculate the distance and direction to the bale.

Since the camera only detects the distance 90° from the camera, the angle and offset have to be calculated. This is possible by measuring the distance to a reference point, as shown in Figure 5.20.

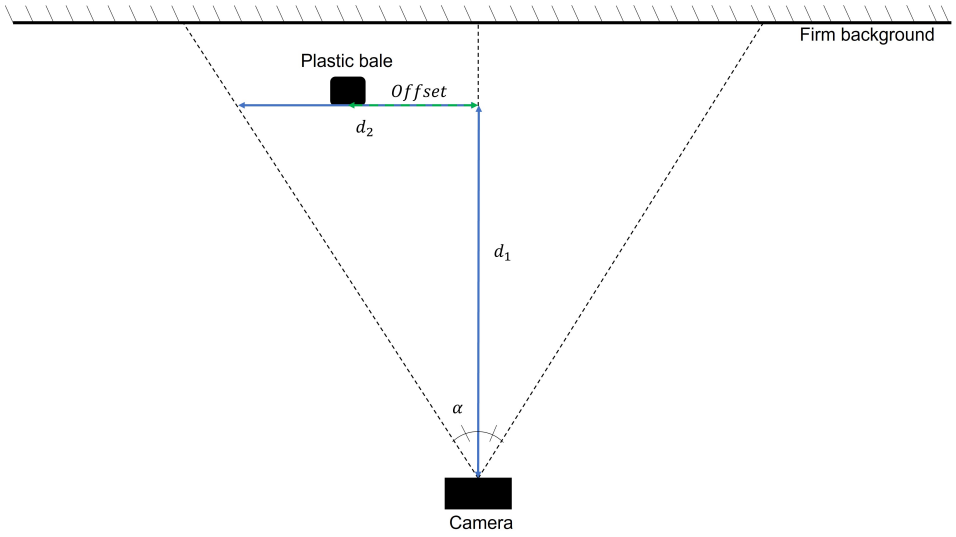

Figure 5.20. Geometry for determine the offset to the plastic bale

Moreover, the resolution is important for the calculation. There are two cases: either the bale is on the left side of the middle pixel or on the right side of the middle pixel of the camera. The resolution was set to a total of 1920 pixels in width, meaning that if the centre pixel of the bounding boxes is lower than half of the total pixel width, that is 960, the plastic bale is located to the left of the middle pixel. Conversely, if the centre pixel of the bounding box is higher than half of the pixel width, the plastic bale is located to the right of the middle pixel. In Figure 5.20, the plastic bale is located on the left side of the middle pixel.

The horizontal distance, d_2, from the middle pixel to the edge of the camera's field of view, can be calculated using trigonometry since both the distance d_1 (the depth to the bale) and the angle α (half of the camera's field of view angle) are known:

$$d_2 = \arctan(\alpha) \cdot d_1 \tag{5.2}$$

The proportion of the horizontal distance can be calculated with Equation 5.3. This is done by subtracting the location of the centre pixel of the bounding box from the pixel width of the left half of the camera's FoV, that is 960, divided by this camera pixel width.

$$Offset_{proportion} = \frac{(960 - BoundingBox_{center})}{960} \tag{5.3}$$

If the other case, i.e. that the bale is located on the right side of the middle pixel, is considered, the same calculations can be applied except for one configuration. If the bale is detected on the right side of the middle pixel, the proportional offset $Offset_{proportional}$ is calculated by subtracting the total pixel width on the right side, that is 960, from the location of the centre pixel of the bounding box and divided with the total pixel width on the same side:

$$Offset_{proportion} = \frac{(BoundingBox_{center} - 960)}{960} \tag{5.4}$$

The offset of the bale from the prototype's centre can then be calculated by multiplying the proportional offset $Offset_{proportion}$ with the horizontal distance d_2:

$$Offset = Offset_{proportional} \cdot d_2 \tag{5.5}$$

5.6.3 Vehicle Alignment

The vehicle alignment is the sequence in which the vehicle is approaching the target (either the bale in the pick-up sequence or the funnel in the drop-off sequence). During the alignment, the vehicle is driving with a constant speed of 0.1 m/s and a controller adjusts the steering angles of both individual wheel-pairs. The vehicle alignment is handled by a controller which tries to align the vehicle against the bale, both by compensating for yaw displacement ($\Delta\Psi$) and lateral displacement (ΔY) in relation to the target. The displacements are calculated from the offset to the target, the distance to the target, and the yaw offset to the target, which is graphically depicted in Figure 5.21.

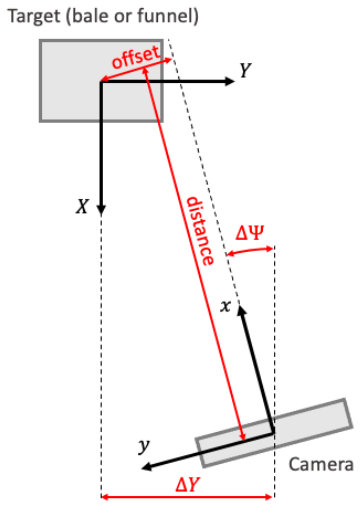

Figure 5.21. Decoupling of yaw displacement and lateral displacement

The lateral displacement is calculated through the known parameters according to

$$\Delta Y = distance \cdot \sin \Delta\Psi + offset \cdot \cos \Delta\Psi \tag{5.6}$$

while the yaw displacement, $\Delta\Psi$, is derived as the difference of actual yaw angle and targeted yaw angle. The reason for using the lateral displacement with respect to the bale instead of the actual vehicle (the offset) is to minimise the interaction effect between the inputs in the controller.

The controller is designed as a continuous dual controller with the two displacements (yaw and lateral) as inputs, and the steering angles of the two axles (rear and front) as the output. The lateral part of the controller is a PI-controller that handles both the rear and front steering angle, causing the vehicle to crab steer without altering the yaw angle. This controller is further improved with a back calculation anti-windup due to the narrow saturation limits of only 12°. The yaw controller is a P-controller which only alters the rear steering angle. This implementation in Simulink can be seen in Figure 5.22.

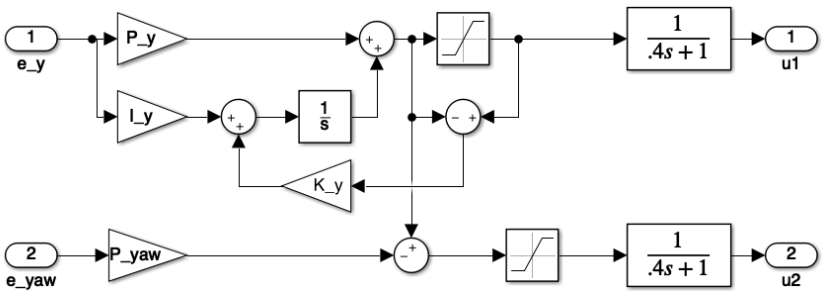

Figure 5.22. Continuous alignment controller in Simulink

This controller was discretised by applying the Euler method with a sampling frequency of 10 Hz, which was implemented to the wheel loader prototype. In order to increase the reliability of the controller, the final destination is verified according to a certain allowance of the two displacements. If this requirement is not fulfilled, the vehicle will simply reverse for a set distance and run the control-sequence again. This full procedure is implemented in the exact same way whether the target is the funnel (with a bale placed inside for image recognition) in the drop-off sequence or the bale in the pickup sequence.

5.7 Software Implementation

The implementation of the software on the prototype included the computer vision system utilised for detecting plastic bales and the ROS architecture that managed node functionalities and communication.

5.7.1 Computer Vision

In order for the wheel loader to recognise the plastic bales, and thus realise SR 1, computer vision was applied through a neural network. The machine learning algorithm used for classification was YOLO v3, which is described in detail in Section 3.4.4. The network was implemented with a customised dataset in order to recognise plastic bales.

Dataset

To customise the neural network used in this task, a custom dataset of plastic bales images was created. The dataset was constructed by both collecting images online and taking photos of the actual plastic bale prototypes. Images of plastic bales found on the internet as well as photos of the plastic bale prototype were used for creating the dataset. Once the images were prepared, bounding boxes were created via an image annotation tool. The bounding box coordinates were translated into a .txt file, which are then treated as labels during training.

Due to the limited size of the dataset, data augmentation was implemented in the form of geometric transformations to expand the size of the training dataset. In the implemented data augmentation, all the images were rotated by 90, 180, and 270 degrees. For the dataset, 550 unique images were collected. With data augmentation, these unique images could be manipulated, creating a total of 2200 images.

Training

The neural network was trained for 8000 epochs. This resulted in a loss of 0.63% and the average loss over training epochs can be seen in Figure 5.23. With a decreasing

loss, the network's predictions get better and therefore it was trained until the loss' decreasing rate more or less stagnated.

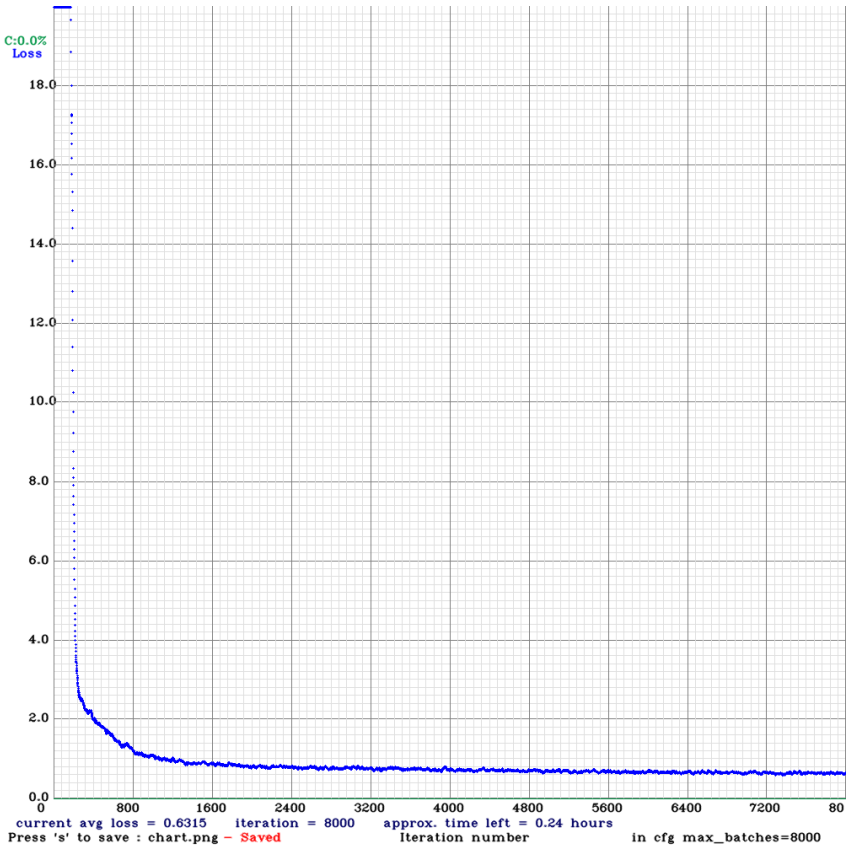

Figure 5.23. YOLO v3 training

The training was executed on a GPU available through KTH, namely a GeForce GTX 1660, on which training the network for 8000 epochs took approximately 16 hours. The batch size during training was 32 due to the limited memory on the GPU. This did not affect the final performance, however the network had to run more iterations for each training epoch.

Performance

Once the network was trained and implemented in ROS, it was integrated into the prototype. Due to the time limitation of the project, only one bale was used for running the sequence of picking up and dropping off the bale. However, the network was able to distinguish between multiple bales, even if they were stacked upon each other and/or placed next to each other.

There is a trade-off between resolution and speed when the network has to run in real-time. By lowering the resolution of the input data, the network can handle

the data faster since there are not as many pixels to compare. However, a lower resolution affects the neural network's accuracy.

5.7.2 ROS Architecture

The entire software architecture was implemented through the ROS system with Python. For easy maintainability, the codes were modulated into several ROS nodes and services. An overview of the connections between all ROS nodes, topics, and services are shown in Figure 5.24.

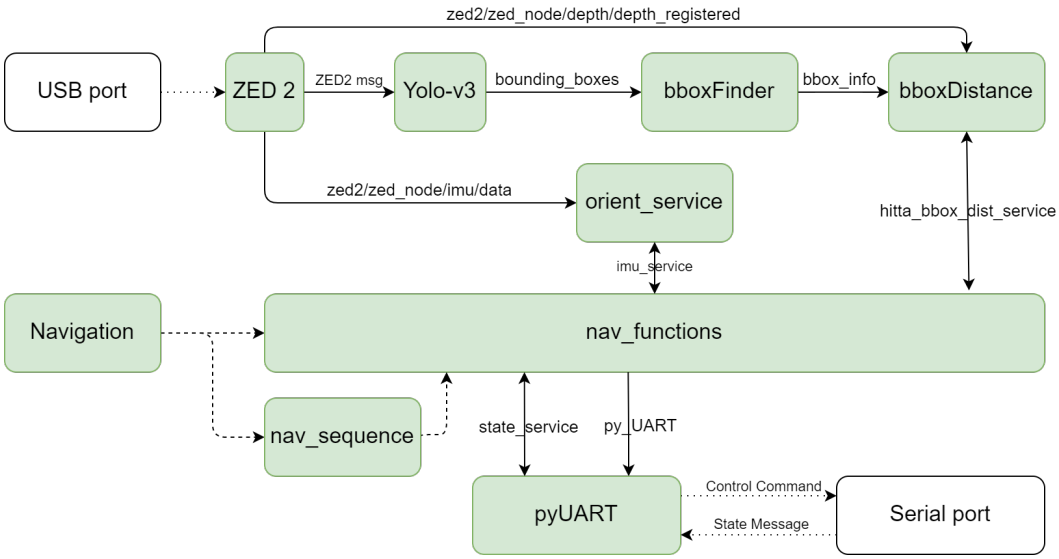

Figure 5.24. ROS system architecture

Note that in the figure, all green rectangles are the names of ROS nodes. The white rectangle represents the external communication ports (to the USB and serial port), which are not a part of the ROS system. Filled arrows going in a single direction represent the ROS topic names and the relationship between publishers and subscribers. Filled arrows going in two directions represent the service names and the relationship between services and clients.

The navigation part of the system is modulated into three separate nodes. Note that the single direction dashed lines between the nodes are used to show their relationship:

- *Navigation*: Main state machine
- *nav_sequence*: Contains all the separate navigation sequences
- *nav_functions*: Contains all the necessary navigation functions

Lastly, the dotted line between the *pyUART* node and *Serial Port* is created to show the communication between the ROS system and the microcontroller.

5.7.3 Communication

The communication between the AGX Xavier and the platform was implemented using serial, more details can be read in Section 5.4.3. Using the serial library in Python called *pySerial*, a program was written to test the communication. This program was later developed into a package with classes that could be imported and used in the ROS software. A UML class diagram of the package is shown in Figure 5.25. The package consists of three classes, namely *WheelloaderCommunication*, *Controller*, and *Observer*. The former class extends the Serial class with additional methods for establishing a connection, formatting and sending messages, and receiving and decoding messages in polling mode. The two latter classes each have their own threads. The Controller class has a thread for sending messages periodically and the Observer has a thread for receiving any incoming message. Both these classes has the same instance of *WheelloaderCommunication*, but use different methods.

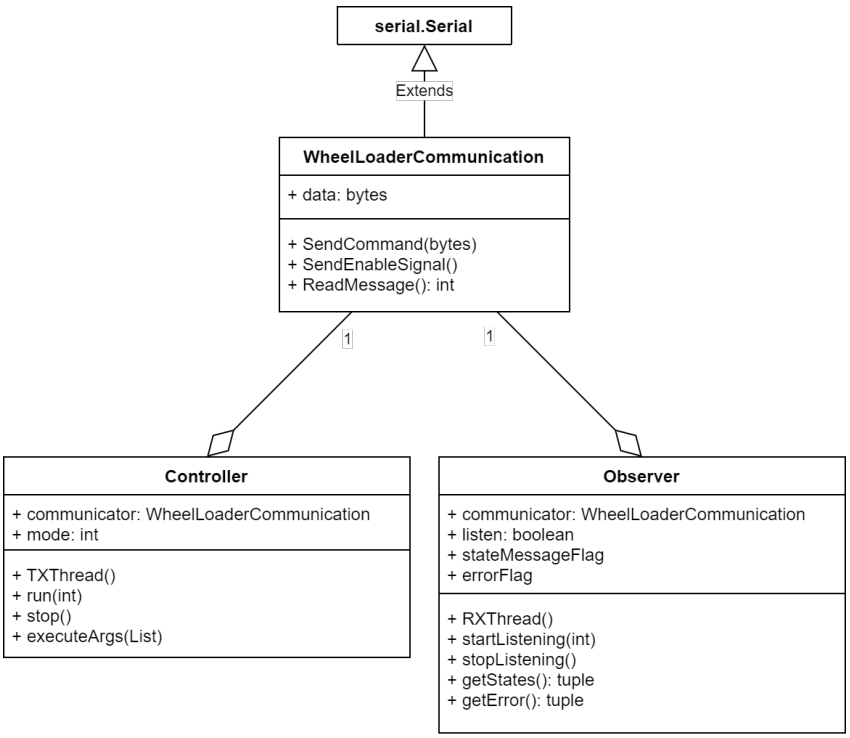

Figure 5.25. UML class diagram of the serial communication package

Chapter 6

Verification and Validation

This chapter will outline the verification and validation tests performed on the wheel loader prototype in order to evaluate whether it meets its technical requirements.

To verify and validate that all the stakeholder requirements, described in Section 1.3, are satisfied, a series of tests were designed and conducted. The verification tests for the prototype's different functions and the tests' corresponding requirements are outlined in Table 6.1. A combined test whereby the entire test routine was evaluated was also performed. The procedure for this test is outlined in Table 6.2 and Section 6.6.

Table 6.1. Verification and validation separated tests design overview

Test Category	Test Name	Requirements
Plastic Bale Recognition	Maximum distance to detect bales	TR 1a, 1e
	Accuracy of bale localisation	TR 1e
	Recognise stacked/side-by-side bales	TR 1b
Mechanical Arm	Max and min heights of the bucket	TR 2a, 2b
	Max and min tilt angles of the bucket	TR 2b
Vehicle Alignment	Alignment to the detected bale	TR 2c, 2d
Dimension and Scale	Dimension measurement of prototype	TR 4a
	Weight measurement of prototype	TR 4a
	Dimension measurement of bale	TR 3a, 4b
	Weight measurement of bale	TR 3a, 4b
Steering	Driving straight	TR 5a
	Driving distance	TR 5b

6.1 Plastic Bale Recognition

The first primary requirement for this project focuses on the software of the prototype and its ability to recognise plastic bales, see SR 1. This function is fundamental to the vehicle's autonomous abilities and must therefore be verified.

6.1.1 Maximum Distance to Detect Bales [TR 1a, 1e]

In this test, the goal is to verify what the longest distance is that the prototype's neural network can detect plastic bales at, as well as how network accuracy varies at different distances. This is in accordance with TR 1a and 1e. To evaluate this, the prototype was placed at a range of distances (from 0.2 to 1.5 meters) away from the bale. YOLO v3 was run on the prototype and the network printed the certainty of whether it detected a plastic bale or not (Accuracy %). This test was performed on three different bale models in order to increase test validity. The results of these tests can be seen in Table 7.1 in Chapter 7.

6.1.2 Accuracy of Bale Localisation [TR 1e]

In order to verify the prototype's localisation of the detected plastic bales, as outlined in TR 1e, a test was designed to evaluate the accuracy of the written depth and offset calculation algorithms. To do this, the plastic bale was placed at randomised perpendicular distances (depths) and horizontal offsets from the prototype. The localisation algorithm printed the detected distances and offsets and these results were compared to the real-life measured distances and offsets. Thereafter, the distance and offset detection accuracy was calculated, as seen in Table 7.2 in the Results chapter.

6.1.3 Recognition of Multiple Bales [TR 1b]

According to stakeholder feedback, the vehicle's ability recognise and differentiate between multiple bales, whether stacked or placed side by side, was a desired functionality. To test this and therefore verify TR 1b, 2 recognition tests were designed. In one, 6 different plastic bale models were stacked on top of each other in a single column. In the second test, the 6 bales were placed together, with 3 rows of bales stacked on top of each other and 2 bales placed next to each other in each row. Proof of the neural network's recognition of these bales can be seen in Section 7.1.3.

6.2 Mechanical Arm

The second stakeholder requirement includes that the prototype should have the abilities to transport and load between designated zones. From a mechanical point of view, this was verified through the arm linkage functionalities by conducting tests of arm reach and tilting angles.

6.2.1 Maximum and Minimum Heights of the Bucket [TR 2a, 2b]

In order to verify that TR 2a and 2b were fulfilled, the height limits of the prototype's bucket had to be measured. For the maximum bucket height, the distance from the top of the bucket to the ground was measured, when the lifting linear actuator was fully extended. For the minimum bucket height the vehicle was placed

at a known height and the distance from the bottom of the bucket to the ground was measured. Both were measured with the bucket parallel to the ground. The result of these measurements are given in Section 7.2.1.

6.2.2 Maximum and Minimum Tilt Angles of the Bucket [TR 2b]

To further evaluate the validity of TR 2b, the bucket's tilt angle range also had to be verified. This was done by tilting the bucket both upwards and downwards to the extent of the prototype's abilities. The bucket angle was then measured between the bucket edge and a horizontal line parallel to the ground. The measurements were done in the CAD model and the tilt angles can be found in Section 7.2.2.

6.3 Vehicle Alignment

For the prototype to succeed in picking up or dropping of the plastic bale, it is essential that the vehicle is able to align itself to certain points of interest. The test designed to verify this is outlined in the following section.

6.3.1 Alignment to the Detected Bale [TR 2c, 2d]

In this test, the goal is to verify the performance of the written P and PI controllers for vehicle alignment to the detected plastic bale. A well-performing vehicle alignment allows the prototype to more successfully perform the pick-up sequence, in accordance with TR 2c. To evaluate this, 25 different tests were conducted with varying bale offsets. Five tests were conducted for each offset case, namely 0 offset (bale is placed directly in front of the prototype), 25 cm and -25 cm offset, as well as 50 cm and -50 cm offset. Here a positive offset is when the bale is placed to the right of the prototype, and conversely placed to the left of the prototype for negative offsets. The bale was set at the same distance from the prototype for each test (110 cm). The number of adjustments required to attempt the pick-up sequence was recorded, as well as whether the pick-up was successful or not. An adjustment was defined as anytime the prototype had to reverse in order to attempt to better align itself to the bale. Since the drop-off procedure utilised the same algorithms and mechanisms as the pick-up alignment, these tests can also be translated to TR 2d. The results of the vehicle alignment test are outlined in Table 7.3 in Section 7.3.1.

6.4 Dimension and Scale

Since this project focuses on proving the concept of a scaled wheel loader, how the prototype's scale compares to its real-life counterpart must be analysed.

6.4.1 Dimension and Weight Measurement of Prototype [4a]

Since the vehicle is an implementation of a concept that has a real-life counterpart, the dimensions and weight of the prototype will represent a scaled version of this. As defined in the project scope and TR 4a, this scale should be approximately 1:10. To evaluate this, the implemented prototype's total length, width, height, and weight were measured. These values were compared to the dimensions of Volvo's concept implementation, the LX03, and a true representation of the prototype scale could be calculated, as seen in Table 7.4 in Section 7.4.1.

6.4.2 Dimension and Weight Measurement of Bale [TR 3a, 4b]

The project's scope, as well as TR 3a and 4b, define that the scaled environment, and therefore the bale model, should have a scale of approximately 1:10. The model bale used for testing and demonstration was therefore measured to evaluate the accuracy of its scale. The bale's length, width, height, and weight were measured and these dimensions are outlined in Table 7.5 in Section 7.4.2. The model's dimensions were then compared to an approximation of a life-size plastic bale and a ratio for the model's actual scale was provided.

6.5 Steering

The steering is an essential function of the prototype and its ability to carry out its tasks. Its ability to maintain orientation when driving straight and know its driving distance is paramount for the hard coded sections of the navigation route. Therefore, the vehicle's steering must be verified.

6.5.1 Driving straight [TR 5a]

Before testing if the vehicle is capable of driving for a specific distance, whether it is capable of driving straight should be ensured first, in accordance with TR 5a. To conduct the test, 2 meters of tape was put on the floor in a straight line. Using the Bluetooth application, the vehicle was sent the command to drive straight forward until the end of the tape. Once the prototype had stopped, the deviation of the vehicle from the taped line was measured and the side of the line the vehicle had steered to was noted. From this data, the angle the prototype deviated from the line with could be trigonometrically calculated using:

$$angle = \arctan(\frac{Deviation}{Driving\ distance}) \tag{6.1}$$

The results from 10 of these conducted tests can be found in Table 7.6 in Section 7.5.1.

6.5.2 Driving distance [TR 5b]

A test was designed to evaluate if the prototype is able to know its driven distance and thus consistently and accurately travel a specified stretch, as stated in TR 5b. To assess this, a 1 meter long strip of tape was placed on the ground to visualise a straight trajectory. The prototype was then given the command to drive forward until it registered a distance of 100 centimeters. This test was conducted 10 times and the discrepancies logged in Table 7.7 in Section 7.5.2.

6.6 Combined Test

In this test, a physical environment was set up and a simulation of the entire handling routine of the prototype was created, as outlined in Table 6.2 with corresponding technical requirements. This testing environment and route are described more in-depth in Figure 5.19 and Section 5.6.1.

Table 6.2. Verification and validation combined tests design overview

Test Sequence	Test Name	Requirements
Nav 1.1	Driving straight to a predetermined distance	TR 3b
Nav 1.2	Turn left 90°	
Nav 1.3	Driving forward until bale is detected	
Nav 1.4	Self-alignment to the bale	
Nav 2	Pick up the bale	TR 3b
Nav 3.1	Reverse while turning right 90°	TR 3b
Nav 3.2	Driving forward until funnel is detected	
Nav 3.3	Self-alignment to the funnel	
Nav 4	Drop the bale in the funnel	TR 3b

A total of 25 tests were conducted in which the following were tracked:

- Initial offset of the plastic bale
- If the bale was successfully picked up
- Number of adjustments of the vehicle required to align to the bale
- If the vehicle succeeded in dropping off the bale into the funnel
- Number of adjustments needed to align the funnel
- The entire duration of the test

The offset was defined by placing the plastic bale a specified lateral distance from the prototype's centre, after the prototype had finished navigation sequence 1.3. Five different offset cases (namely 0, ±25, and ±50) were tested 5 times each. A

positive offset represents the bale being placed to the right of the vehicle's centre, and conversely a negative offset represents the bale placed to the left. An adjustment is defined by the number of times the prototype had to reverse when aligning to the pick-up/drop-off points in order to reattempt the alignment process. A pick-up was considered successful if the vehicle could get the plastic bale into the bucket without assistance. A drop-off was successful if the vehicle could offload the bale into the designated funnel. To consistently measure the time, the timer was started when the prototype started moving during navigation sequence 1.1 and stopped after the plastic bale was dropped and reached the funnel or ground (depending on if drop-off was successful or not) during navigation sequence 4.

The entire logged data of these 25 tests and their results are shown in Table 7.8 and the summarised results can be seen in Table 7.9, both in Section 7.6.

Chapter 7

Results

This chapter presents the data collected regarding the prototype's performance in the verification and validation tests described in Chapter 6.

A set of requirements was devised in the initial phases of this project, outlined in Chapter 1.3. These specifications were used as guidelines for how to conduct the state of the art analysis, create the concept design, and implement the concepts on the actual prototype. In order to evaluate whether the requirements were maintained and upheld throughout this process, a series of verification and validation tests were designed in Chapter 6. The results of these tests are outlined in the sections below. The final, fully assembled and functioning prototype used for these tests and final demonstration can be seen in Figure 7.1.

Figure 7.1. Picture of the fully assembled prototype

7.1 Plastic Bale Recognition Tests

The results from the tests for plastic bale recognition are outlined in the following section. The logged data will evaluate whether requirements TR 1a, 1b, and 1e are fulfilled.

7.1.1 Maximum Distance to Detect Bales [TR 1a, 1e]

In this test, described in more detail in Section 6.1.1, the accuracy of the network's ability to detect bales at different distances were evaluated. The results are outlined in Table 7.1.

Table 7.1. Logged data of bale recognition test

Distance to bale [m]	**Bale 1 Accuracy** [%]	**Bale 2 Accuracy** [%]	**Bale 3 Accuracy** [%]
1.5	Not Detected	10-20	Not Detected
1.4	Not Detected	30-40	35-50
1.3	15-20	25-30	15-20
1.2	70	60	85-90
1.1	85-90	85-90	80-85
1.0	85-90	93	75-80
0.9	85	85	35-40
0.8	85-90	75-80	70-75
0.7	80-85	85	70
0.6	60	70	65-70
0.5	70-75	55	45-50
0.4	70-75	35-40	70
0.3	50	44	45-47
0.2	30	36	30-35

The best accuracy reported is 93% for bale 2, at a 1 meter distance to the bale, as seen in Table 7.1.

7.1.2 Accuracy of Bale Localisation [TR 1e]

In order to evaluate the network's accuracy in detecting a plastic bale at varying distances and lateral offsets, the test described in Section 6.1.2 was designed. The results from this test are shown in Table 7.2.

Table 7.2. Logged data of bale localisation test

Real Distance [cm]	**Detected Distance [cm]**	**Accuracy** [%]	**Real Offset [cm]**	**Detected Offset [cm]**	**Accuracy** [%]
66	65.2	98.70	5	5.5	90.00
51	51.5	99.02	6	7.1	81.67
40.5	40.8	99.26	5.5	6.5	81.82
98.5	96.1	97.56	5.8	5.4	93.10
137.5	134.8	98.04	9	14.8	35.56
109	106.4	97.61	5.5	7.9	56.36
96.5	94.2	97.62	4.5	8.3	15.56
78	78.8	98.97	6	10.1	31.67
62.8	63.8	98.41	3.5	4.5	71.43

From Table 7.2, it can be seen that the accuracy of detected distances is higher than 95%. The offset accuracy has a range from 30% to 93%.

7.1.3 Recognition of Multiple Bales [TR 1b]

Proof of the prototype's ability to recognise multiple plastic bale models, whether stacked or side by side, is shown in Figure 7.2. The test is outlined in Section 6.1.3.

(a) Six plastic bales stacked in a column **(b)** Six plastic bales divided in two columns

Figure 7.2. Neural network recognising multiple bales

From the bounding boxes the network has placed around the edges of the stacked bales, it can be seen that the prototype is able to differentiate between bales placed within close proximity of each other.

7.2 Mechanical Arm Tests

To evaluate the reach and flexibility of the prototype's mechanical arm, tests were designed to verify the reach height and tilt angle of the bucket.

7.2.1 Maximum and Minimum Heights of the Bucket [TR 2a, 2b]

When verifying the heights that the prototype's bucket can reach, as outlined in Section 6.2.1, it was established that the bucket has a maximum reach of 230 mm and a minimum of -10 mm, when the bucket is parallel to the ground. The minimum value is negative because this refers to the bucket's height in relation to the ground level of the vehicle.

7.2.2 Maximum and Minimum Tilt Angles of the Bucket [TR 2b]

According to tests performed on the prototype, described in Section 6.2.2, the bucket has a maximum tilt limit of 58° and a minimum of -43°.

7.3 Vehicle Alignment Test

This section shows the results of the test designed to verify the vehicle's alignment ability.

7.3.1 Alignment to the Detected Bale [TR 2c, 2d]

Several tests were performed to ascertain the performance of the vehicle's bale alignment ability, as described in Section 6.3.1. The logged data of these tests, namely the success rate of the pick-up sequence and the number of adjustments required to attempt the sequence, is shown in Table 7.3.

Table 7.3. Logged data of vehicle alignment to bale

No. of Test	Lateral Offset to Bale [cm]	Longitudinal Distance to Bale [cm]	Able to Pick Up [Y/N]	No. of Adjustments
1	0	110	Y	0
2	0	110	Y	0
3	0	110	Y	0
4	0	110	Y	1
5	0	110	Y	0
6	25	110	N	2
7	25	110	Y	1
8	25	110	Y	1
9	25	110	Y	1
10	25	110	Y	1
11	-25	110	Y	1
12	-25	110	Y	1
13	-25	110	Y	0
14	-25	110	Y	1
15	50	110	Y	0
16	50	110	N	4
17	50	110	Y	5
18	50	110	Y	3
19	50	110	N	4
20	50	110	Y	3
21	-50	110	Y	2
22	-50	110	Y	2
23	-50	110	Y	2
24	-50	110	Y	2
25	-50	110	Y	2

In Table 7.3, the results of the adjustments are reported for a lateral offset ranging from 0 to 50 cm in both directions. For three out of 25 tests, the vehicle was not successful in picking up the bale. The largest amount of adjustments necessary to successfully finish the sequence was five, which was reported for the lateral offset of 50 cm.

7.4 Dimension and Scale Verification

The prototype and its environment were designed to have an approximate scale of 1:10. Whether the design fulfilled this goal is revealed in the following sections.

7.4.1 Dimension and Weight Measurement of Prototype [4a]

The prototype's dimensions and weight were measured and compared to its real-life counterpart, the LX03, as described in Section 6.4.1. These measurements are shown in Table 7.4.

Table 7.4. Dimensions of the wheel loader

Measurement	Scaled Dimension	Real Dimension	Scaled Ratio
Length	72.5 cm	590 cm	1:8
Width	24.5 cm	210 cm	1:8.5
Height	26.5 cm	190 cm	1:7
Weight	12.8 kg	5000 kg	1:390

The scaled ratio of the wheel loader can be found in Table 7.4. The dimension scale ranges from 1:7 to 1:8.5, which is a close approximation to 1:10.

7.4.2 Dimension and Weight Measurement of Bale [TR 3a, 4b]

The scale of the modelled plastic bale also had to be evaluated and compared to a life-size bale, as described in Section 6.4.2. These dimensions and the bale's actual scale is shown in Table 7.5.

Table 7.5. Dimensions of the plastic bale

Measurement	Scaled Dimension	Real Dimension	Scaled Ratio
Length	12 cm	120 cm	1:10
Width	10 cm	100 cm	1:10
Height	10 cm	100 cm	1:10

Two different bale models were developed, one weighing 247 g (manufactured in plastic) and one weighing 1.4 kg (manufactured in metal).

7.5 Steering Tests

The vehicle's steering ability, both in terms of maintaining orientation and driving distance, had to be assessed. The data of the tests designed to evaluate this are revealed in the following sections.

7.5.1 Driving straight [TR 5a]

A test was designed in Section 6.5.1 to ascertain what the vehicle's deviation was when driving in a straight line for a set distance. The results from this test are shown in Table 7.6. The orientation refers to the side of the taped line the prototype deviated towards.

Table 7.6. Logged data of driving straight test

No. of test	Deviation [cm]	Set Driving Distance [cm]	Angle [deg]	Orientation
1	6.0	200	1.72	Left
2	5.7	200	1.63	Left
3	1.6	200	0.46	Right
4	5.0	200	1.43	Left
5	2.3	200	0.66	Left
6	8.4	200	2.41	Left
7	8.6	200	2.46	Left
8	2.0	200	0.57	Left
9	3.5	200	1.00	Left
10	10.5	200	3.01	Left
Average	5.36		1.53	

7.5.2 Driving distance [TR 5b]

Whether the vehicle was able to correctly and consistently drive a set distance also had to be evaluated, as outlined in Section 6.5.2. A test was designed to assess this functionality, the results of which are shown in Table 7.7.

Table 7.7. Logged data of driving distance test

No. of Test	Actual Driving Distance [cm]	Set Driving Distance [cm]	Difference [cm]
1	98.3	100	-1.7
2	100.5	100	0.5
3	100.5	100	0.5
4	101	100	1.0
5	99.3	100	-0.7
6	99.5	100	-0.5
7	100	100	0
8	99.7	100	-0.3
9	100.5	100	0.5
10	99.8	100	-0.2
Average	99.91	100	-0.09

7.6 Combined Test

After all of the previous tests were completed to investigate the vehicle's functionalities independently, a combined test sequence was performed to evaluate these functionalities together, described in more detail in Section 6.6. The logged data of this combined test is shown in Table 7.8.

Table 7.8. Logged data of the combined test

No. of Test	Bale Lateral Offset [cm]	Able to Pick Up [Y/N]	No. of Adjustments	Able to Drop off [Y/N]	No. of Adjustments	Time [m:s]
1	0	Y	0	Y	2	02:02
2	0	Y	0	Y	1	01:48
3	0	Y	4	N	3	03:08
4	0	Y	0	Y	0	01:39
5	0	Y	2	N	1	02:07
6	25	Y	1	Y	2	02:44
7	25	Y	2	N	3	02:51
8	25	N	2	Y	0	02:14
9	25	Y	2	Y	3	02:40
10	25	Y	2	Y	6	03:07
11	-25	Y	1	Y	2	02:07
12	-25	Y	1	N	3	02:12
13	-25	Y	1	N	7	03:04
14	-25	Y	1	Y	1	01:52
15	50	Y	1	Y	0	01:53
16	50	Y	3	Y	0	02:24
17	50	Y	3	Y	0	02:31
18	50	Y	4	N	2	02:55
19	50	Y	5	Y	2	03:08
20	50	N	5	N	8	04:59
21	-50	Y	5	N	3	02:44
22	-50	Y	6	Y	0	03:01
23	-50	Y	4	N	1	02:25
24	-50	Y	4	Y	0	02:22
25	-50	N	5	N	2	02:33

The results were further divided, and the average results of each offset can be seen in Table 7.9.

Table 7.9. Results of the combined test

±Bale Offset [cm]	Bale Pickup Success Rate [%]	Average No. of adjustments	Bale Drop off Success Rate [%]	Average No. of adjustments	Average Time Duration [m:s]
±0	100	1.2	60	1.4	1:49
±25	90	1.4	70	2.7	2:24
±50	100	1.2	60	1.4	1:49

7.7 Fulfilment of Requirements

The tests and their results outlined in the previous sections were designed to prove whether the project's requirements had been fulfilled or not. A summary of these

results, and especially if the corresponding requirements were met, is shown in Table 7.10.

Table 7.10. Summary of Result Section

Technical Requirements	Fulfilled?	Comment
1.a A trained neural network shall be able to perform image recognition and detection of the plastic bales	Yes	· The prototype is able to detect a plastic bale from 0.2m to 1.3m, shown in Table 7.1
1.b The neural network shall be able to distinguish between multiple bales stacked on top of each other or placed side by side.	Yes	· Figure 7.2 shows a successful detection of 6 bales
1.c A dataset of sufficient size of plastic bale images shall be created to train the network model	Yes	· A dataset of 2200 images was created and the prototype was able to recognise plastic bales
1.d The prototype shall have a LiDAR and/or camera embedded.	Yes	· ZED2 camera was implemented and installed on the bucket
1.e The prototype shall be able to find the location of the plastic bale relative to the prototype itself.	Yes	· The detected bale distance has an accuracy of 95%, Table 7.2
2.a The prototype shall have the mechanical strength and stability to pick up and carry the weight of the bales	Partially	· Structural stability was accomplished · Lighter bale could be picked up and carried · Linear actuator limited the ability to pick up heavier bale
2.b The tool shall be able to lift to a height of at least 230 mm when parallel to the ground and tilt at an angle of ±40	Yes	· Maximum height of bucket is 230 mm · Maximum tilt angle is 58° · Minimum tilt angle is -43°
2.c The prototype shall be able to align itself to the plastic bale to pick up the bale	Yes	· For 25 tests, 22 were successful · Maximum number of adjustments was 5 · Data in Table 7.3
2.d The prototype shall be able to align itself to the funnel to drop off the bale	Yes	· In Table 7.3, the vehicle is proven able to align to a bale · These results can be translated to the drop-off alignment
3.a The prototype shall be tested on a scaled site model that includes model bales, a loading point, and an unloading point	Yes	· Figure 5.19 shows the testing route
4.a The prototype shall have a scale of approximately 1:10	Partially	· The prototype has a dimension scale of 1:8 in length, 1:8.5 in width, and 1:7 in height · Dimensions in Table 7.4
4.b The plastic bales shall have a dimensional scale of approximately 1:10	Yes	· The plastic bales have a 1:10 scale · Dimensions in Table 7.5
4.c The prototype shall carry its own rechargeable battery	Yes	· 21 V rechargeable battery is used as the power source of the vehicle
5.a The prototype shall have the ability to know its orientation and if it is driving in a straight line	Yes	· Data in Table 7.6 indicates that the vehicle can drive straight
5.b The prototype shall have the ability to know its driving distance	Yes	· Proven to detect the distance with error no more than 1 cm · Table 7.7
6 The given budget of 50 000 SEK shall not be exceeded	Yes	· Spendings amounted to 34 922 SEK · Appendix B

Chapter 8

Discussion and Conclusions

This chapter will present discussions of the report's content and what conclusions can be drawn from it. This includes a reflection on the test results and an analysis of the project's ethics and sustainability. Deliberations about the prototype's mechanical, navigation, and computer vision systems are also presented.

8.1 Reflections on Results

The tests performed were chosen in order to verify the requirements stated by the stakeholders. One of the most important requirements in this project was the recognition of the plastic bale, Section 7.1, since this is the foundation for if this autonomous loading and unloading would work in a real-life implementation. It can be seen in Table 7.2 that the prototype is able to detect bales at a decent range, up to about 1.3 meters. In the context of a life-size machine, this would correspond to 13 meters, meaning that if it can be assumed that these results would be linearly scalable, it would give the vehicle some distance to align to the target. Tests were also performed on the accuracy of the detections as well as recognising multiple bales at once, which proved robustness in the recognition, and suggests that this method has potential to work in a larger extent.

One of the subsystems that has to work well in order to investigate this concept is the mechanical arm system, Section 7.2. Its manoeuvrability was investigated, and concluded that it can reach a height of 230 mm with a horizontal bucket. This value might not be sufficient for all applications, but is enough to prove this particular concept, since no emphasis was put on reaching high objects. The same goes for the tilt angle, which can tilt -43° from its upmost position. The concept can be proven with this system due to the fact that the prototype can successfully drop the bale from this height and with this tilt angle.

The vehicle alignment tests, Section 7.3, were the actual tests that verified if the recognition of the bale could be useful in real-time. It showed how well the vehicle

can align towards the bale and initiate the pick-up sequence. Table 7.3 shows the results of these tests, and displays promising results for the ability to pick up bales. The functionality of the navigation and the controller is further discussed in Section 8.4. A noticeable issue from the results regarding the alignment is that the pick-up success rate is heavily dependant of the relative position of the bale in relation to the vehicle. When the offset to the bale is negative (meaning that it is located to the left) the success rate is much higher than when the offset is positive (it is located to the right). The reason for this is partly due to the location of the camera lens for object detection, which is the left lens with an offset from centre of the vehicle with about 60 mm. Some mechanical constraints with uneven Ackermann steering on the rear wheel pair, as well as the calibration of the steering angles, could also contribute to this.

The dimension and scale tests, Section 7.4, were designed in order to investigate the relationship between the chosen scale for this project (1:10) and the actual counterpart of the prototype, the LX03. As can be seen in Table 7.4, the prototype resulted in a close approximation to 1:10, but in general a bit larger. The reason for this was that there were many components that needed to fit on the prototype which were not scalable, such as controllers, the AGX Xavier, and the camera. These parts all contributed to compromising the chosen scale resulting in a close to, but not entirely correct, scale. The weight scale is assumed to be scaled in cube (10^3), meaning a scale of 1:1000 was targeted, representing a weight of about 5 kg. That is, in comparison to the actual weight of 12.8 kg, quite far off. However, if the resulting scale of the vehicle is also included (1:8, 1:8.5, and 1:7), the weight ratio would result in 1:476 meaning a weight of 10.5 kg, which is not far away from the actual results. The weight and scale are only chosen in order to follow a scaled concept, and therefore not too much emphasis was put on the targeted scale. The same is valid for the scale of the bale, which is designed according to the previously stated approximation. The weight of the actual bale can only be estimated, and was initially set to 1.4 kg. That resulted in a too heavy block for the higher positions of the linkage, resulting in a bale with a reduced weight of 247 g being implemented instead. This meant that the requirement is only partially fulfilled, but it is not considered to be a big problem due to that this does not affect the proofability as long as the correct bale is being used.

In order to verify the ability of maintaining orientation as well as distance driven according to the analytical model, steering tests, described in Section 7.5, were performed. These tests verified how well the vehicle is able to maintain driving in a straight line. These results in Table 7.6 show that the vehicle has a tendency to deviate to the left due to unsymmetrical mechanical structures and mechanical backlash. The average deviation of the vehicle is around $\pm$ 5.36 cm, with orientation deviating $\pm$ 1.53 degrees. These deviations are small enough to be ignored, especially considering the short distances the prototype travels. It can thus be concluded that the vehicle is capable of driving straight. Regarding the verification of

driving a specified distance, it can be concluded from Table 7.7 that the vehicle is capable of driving for a specific distance with negligible discrepancies. It can also be noted that the vehicle had a tendency to drive less than the specified distance, and not more. This might be due to the calibration of the tyre-diameter, or due to longitudinal wheel slip due to uneven and dusty tyres.

The combined test, Section 7.6, with results seen in Table 7.8, comprised of the most extensive and realistic test cycle. These tests were evaluated on all different sequences, counting both the number of adjustments in the pick-up sequence and the number of adjustments in the drop-off sequence. The amount of failed combined tests were unfortunately quite high, with some tests requiring quite a few adjustments (i.e. reversing and re-aligning). From the 25 tests conducted, the success rate of picking up the bale, regardless of the initial offset, was 93% with an average of 1.33 adjustments to align the vehicle itself to the plastic bale. The success rate of dropping off the bale into the funnel was 67% with an average of 2.27 adjustments needed to align the prototype to the funnel. The average time duration of the success case is 1 minute 20 seconds. The definition of a success case is when the vehicle succeeds in both picking up and dropping off the plastic bale. One of the reasons for this was that the prototype was not able to detect and calculate the position of the bale placed in the funnel entirely correctly. This is further discussed in Section 8.5.2. The built-in magnetometer in the camera's IMU also had a strange resolution, with the angles spanning from $-177°$ to $+177°$ with zero being the direction that the camera is initiated in. This gave the vehicle a blind-spot at 180° of $\pm 3°$. Unfortunately, this was also the direction that the drop-off sequence was performed in, which is one of the reasons why the controller did not work as intended. In the end, the biggest reason for not being able to succeed in more combined tests was due to a limited amount of tests before the official tests were started. This meant that the controllers were not completely fine-tuned, as well as that no ideal setup of the funnel had been found.

8.2 Ethics and Sustainability

Since the goal is to implement this concept on an actual product, it is essential that ethics and sustainability are carefully considered.

8.2.1 Ethics

Ethical issues are possible when an automated wheel loader is introduced to the market. This concept aims to reduce the need for human intervention in the plastic bale handling process. Such a process utilises heavy machinery and thus creates a potentially hazardous work environment for the human operators. Removing the human operators from such work environments can be seen as a safety precaution. However, it may also challenge people's livelihoods by reducing their job opportunities. In this case, such a trade-off should be carefully considered.

The conceptual wheel loader is fully autonomous which may lead to safety concerns. It is important to examine the possibility of collisions, as it may cause injuries and/or damage to the surrounding environment.

Even though the conceptual wheel loader is fully autonomous, people will still have to attend to it in terms of maintenance. It is therefore important that the wheel loader presents and includes safety features that contradict risky and hazardous operations.

8.2.2 Sustainability

The conceptual wheel loader is fully electric, meaning that the implementation of such a wheel loader will reduce emissions and resource usage since the current plastic waste handling process primarily makes use of fossil-fueled wheel loaders.

Since this prototype is applicable to waste handling processes, plastic recycling methods could be made more efficient due to the autonomous attributes of the concept. By improving the efficiency of such processes, recycling costs can be reduced, thus reducing the cost of recycled material. The positive effect of this could be an expansion of the recycling market, which could increase the overall percentage of plastics being recycled, and thus decrease the pollution of the atmosphere and the oceans.

Sustainability was not considered when designing the electronics. However, modularity was emphasised meaning that most parts can be accessed, replaced, and recycled. Besides the electronics, most of the materials chosen in the overall prototype design are material of common usage, such as aluminum sheet metal and PLA (Polylactic Acid), which is a more environmentally friendly plastic. This concludes that the conceptual wheel loader assuredly can be designed with emphasis on sustainability.

8.3 Mechanical

The mechanical design that was briefly described in Section 5.1 was implemented during the course of this project, and is subject to manufacturer errors and limitations.

8.3.1 Manufacturing

One of limitations of the mechanical implementation was the access to the machines as well as correct materials on school premises. Access to the mechanical workshop is restricted by educational hours held in the machine workshop, in which this project is not included. The admission to tools and machines was therefore limited to

biweekly meetings outside educational hours. Such a restriction is a big challenge for manufacturing as the available time for production is further reduced.

Furthermore, milling machines, lathe machines, and bending machines used when manufacturing components require precise manual control. As access to more precise machines, such as Computer Numerical Control (CNC) machines, is mostly restricted to research purposes and by operational knowledge, the manual machines were the only viable option. The challenge was therefore the fact that it was common to fabricate parts that did not fulfill the requirements due to operator errors, and time consuming to redo the parts.

The manufacturing tolerance issue is another big concern related to the available machines. As the process of bending was performed with manual bending equipment, the parts resulted in insufficient precision. To counter this error, some play was introduced in the screw holes of the bent parts. Furthermore, tolerance errors exist in both the 3D-printed and metallic parts. This results in an observable backlash both for the steering system and the linkage system.

8.3.2 Plastic Bale

Due to the high density of the material used in the construction of the arm linkage and the bucket, the weight of the bucket and arm assembly became rather heavy. Since the actuator type, suitable for the implemented prototype scale, is small, they are also limited in force. These factors therefore resulted in a challenge of handling bales heavier than that of the plastic model.

8.4 Navigation

As described in Section 5.6, the wheel loader runs with both hard coded and autonomous sections in the testing environment. Below are discussions on how the implemented navigation performed.

8.4.1 Depth Calculations

The depth calculation operated satisfactorily for the project's scope and the prototype's basic functionality, but there are some flaws with the approach. First of all, only the middle pixel of the bounding box is used to determine the depth. This was deemed sufficient, since this provides a depth measure for the bale. The issue is that it does not account for the orientation of the plastic bale. By using multiple pixels to extract the depth, angles could be calculated and provide bale orientation. However, since the edge pixels of the bounding box often are outside of the actual plastic bale, they would give a misleading distance. Due to this, a single point to calculate the depth seemed sufficient for the scope of the project.

8.4.2 Offset Calculations

The offset calculations depend on the depth. An issue with this is that if the wheel loader is too close to the bale with a large offset, it might lose the bale from the camera view.

8.4.3 PI and P-Controllers

In this project, the autonomous driving is controlled by a PI-controller in combination with a P-controller. The controllers were tuned so that the prototype would be able to align its position to the target (the bale). The controller uses the input from the depth calculations as well as the built-in IMU in the ZED2 camera. It then supplies the vehicle with steering angles in order to move towards the requested destination. Due to a limited amount of extensive combined tests, the controllers are only satisfactorily tuned, and not fully optimised for all scenarios. The timing and rates of the different applications (i.e. camera, image recognition, and controller) also cause problems with the control cycle, with the largest problem being the frame rate and timing of the image recognition. This could potentially be solved by a theoretical model which updates at a higher rate, but due to time constraints this was not implemented.

8.5 Computer Vision

As shown in Section 7.1.3, the implemented neural network was able to recognise plastic bales in real time. The following sections will outline some discussions regarding how the prototype's computer vision functionality was implemented.

8.5.1 YOLO v3

The YOLO network, after training the architecture with the created dataset, was able to recognise the plastic bales in a functioning way. However, the Darknet framework which it runs on is not intended for an AGX Xavier and this could act as a bottleneck for speed. Because of this, other object classification architectures might work similarly, or even better in combination with this computer.

8.5.2 Recognising the Funnel

Due to time limitations, the network was only trained to recognise plastic bales. Therefore, a plastic bale was placed in the dedicated drop-off funnel in order for the wheel loader to recognise it and adjust towards it. With YOLO, it is possible to recognise multiple classes and therefore the network could have been trained to recognise both plastic bales and the funnel. Furthermore, this would introduce some issues with the training, and the dataset would have to be expanded with images of similar funnels.

8.5.3 Object Detection

The network detected plastic bales from the live feed and output the coordinates surrounding them. This seemed sufficient for the scope, however it does not provide any information about the orientation of the plastic bale relative to the prototype.

A possible solution to this is to combine the detection with depth sensing in multiple pixels in order to create a point cloud representing the 3D object. This was briefly implemented but it brought up two major challenges. Firstly, the camera's depth was not accurate enough to create a functioning 3D object. Secondly, the computing power required is increased when the system checks the depth for every pixel, which in turn might slow down the whole process. Because of these challenges, the depth sensing was only used in the final prototype to find the distance to two objects at a time - namely the pixel in the middle of the bale's bounding box and the reference background.

Chapter 9

Future Work

In this chapter, possible future improvements and tasks are presented.

9.1 Full Autonomous Mode

The prototype is currently semi-autonomous, since it is autonomous in two sections of the test route with hard coded sections implemented only to ensure the bale is within the range of the computer vision. Because the computer vision relies on the camera's resolution, the wheel loader is only able to classify the plastic bales in a relatively small range. Enhancement could therefore be done by enlarging the detection zone and camera quality of the vehicle and thereby allow for more autonomous functionality.

Furthermore, the prototype wheel loader is not programmed for a scenario with multiple bales. It is able to detect multiple bales but not programmed to move the bales one by one. In a fully autonomous mode, multiple bale transfer could be implemented. The machine would then need to be able to distinguish the correct sequence of bales to handle and how to pick one up without disturbing another.

To achieve a fully autonomous prototype, improved mapping and navigation functionalities would also be necessary in order to autonomously navigate in the environment and locate the plastic bale without assistance. This proposal for future work is discussed further in Section 9.6.

9.2 Obstacle Detection and Avoidance System

Currently, the prototype is designed for plastic bale detection only. Obstacle detection is not implemented in the vision system of the prototype machine or through the sensors at the moment. The ultrasonic sensors could be used to detect obstacles behind the wheel loader. Furthermore, a mapping function in combination with a

path planning algorithm could be designed in order to avoid obstacles and navigate around them.

9.3 Increasing Computing Speed

The prototype has a functioning computer vision algorithm, which recognises plastic bales. However, the running software is not optimised for computing speed, and therefore the system is only able to handle around 4 frames per second with the chosen image resolution. Since the scope of this project was to prove a concept, optimisation was not the main focus. Therefore, it seemed sufficient to run on 4 frames per second.

Furthermore, for future research, it could be a good idea to look into increasing the computing speed. This would be helpful with the alignment for the bales, since the frequency of the positional feedback would also be increased.

9.4 Tyre

The original fillers of the tyres, provided by the supplier, were not strong enough to withstand the weight of the vehicle. A solution was 3D printing in combination with foam fillers. This filler reduced the deformation of the tyres and gave a more stable performance. However, deformation still occurred and this influenced the levelness of the bucket in terms of the whole prototype's balance. An improvement would therefore be to find a better filling material or redesigning the fillers to reduce the bucket's inclination.

9.5 Backlash

The front and rear Ackermann steering have different structures and the backlash in the current steering system is noticeable. This was solved by using PI-controller and P-controller to tune the input steering angle. The steering angle accuracy could be improved by redesigning it as a more reliable steering system and by manufacturing the components with better tolerances.

9.6 Mapping and Localisation

SLAM allows the prototype to create a 2D or 3D map of its surroundings. This function allows the prototype to know its relative position in the environment. An additional feature which could be enabled by SLAM is the localisation of the plastic bales recognised by the camera by marking them on the map with a unique ID. The relative locations of the recognised plastic bales, its unique ID, and the total number

of entities could then be linked to an online logistic platform to track and control the storage of the recycling environment.

9.7 Embedded Systems and Communication

The microcontroller firmware was developed to a level at which it was working sufficiently. However, the program was not fully reliable, as the communication sometimes stopped working and every so often, one of the stepper motors would stop rotating. These issues required the microcontroller to be reset manually. The causes of these issues are yet to be investigated.

Moreover, it was realised at a later stage in the project that a framework for serial communication within ROS already existed, called rosserial, and it has support for the STM32 ecosystem. Implementing this framework could make the communication more reliable and easier to implement with ROS. Therefore, it may have some practical benefits for the platform.

9.8 Mechatronics

Two of the linear actuators used introduced some limitations on the prototype. Firstly, the tool lift actuator struggled to lift the bucket higher than approximately 10 cm, when loaded with the heavier plastic bale model. To solve this problem, another lift actuator could be added, working in parallel with the original one. This would essentially double the amount of lifting force. Secondly, the scissor joint actuator did not have any positional feedback other than limit switches. The problem was that there was not enough room for the actuator and there was a risk that it would break if full extension was attempted. To prevent the actuator from breaking, the time of extension was limited by the firmware. This however, resulted in unpredictable scissor angles which were dependent on the load in the bucket. Implementing an external limit switch, or potentiometer, on the scissor joint could solve this inaccuracy problem.

Bibliography

[1] Svenska MiljöEmissionsData (SMED). *Kartläggning av plastflöden i Sverige.* Naturvårdsverket, 2019. URL: http://naturvardsverket.diva-portal.org/smash/get/diva2:1327285/FULLTEXT02.pdf.

[2] *What is Waste Incineration?* URL: https://www.conserve-energy-future.com/advantages-and-disadvantages-incineration.php. (Accessed: 19.05.2021).

[3] *Volvo CE and LEGO Technic Introduce Zeux.* URL: https://www.volvoce.com/global/en/this-is-volvo-ce/what-we-believe-in/innovation/zeux/. (Accessed: 19.05.2021).

[4] *Introducing The Concept LX03.* URL: https://www.volvoce.com/global/en/this-is-volvo-ce/what-we-believe-in/innovation/lx03/. (Accessed: 15.12.2021).

[5] Shravan H. Gawande, A. A. Muley, Rahul N. Yerrawar. 'Optimization of Torsional Stiffness for Heavy Commercial Vehicle Chassis Frame'. In: (2018). DOI: 10.1007/s42154-018-0044-6.

[6] Anh-Tuan Dang, Dinh-Ngoc Nguyen, Dang-Hao Nguyen. 'A Study of Scissor Lifts Using Parameter Design'. In: (2020).

[7] *Lego Zeux concept.* URL: https://www.lego.com/pl-pl/kids/sets/technic/volvo-concept-wheel-loader-zeux-88a0d74a84374016a6b37363d0c41cc4. (Accessed: 10.05.2021).

[8] Sameer Prabhu, Jeffrey Wendlandt. 'Multi-Domain Modeling and Simulation of an Electro-Hydraulic Implement System'. In: (2006). DOI: 10.4271/2006-01-3490. URL: https://www.researchgate.net/publication/268378859_Multi-Domain_Modeling_and_Simulation_of_an_Electro-Hydraulic_Implement_System.

[9] Torcan life equipment. *What is forklift? Working mechanism where is it used?* 2020. URL: https://torcanlift.com/what-is-forklift-working-mechanism-where-it-is-used/. (Accessed: 19.05.2021).

[10] Prodig. *Bale Grab.* 2021. URL: http://www.prodigattachments.com/en/products/agricultural/6-bale-grab. (Accessed: 19.05.2021).

[11] *Grading Bucket*. URL: https://www.volvoce.com/europe/en/attachments/wheel-loader-attachments/loader-buckets/grading-lwl/. (Accessed: 19.05.2021).

[12] *Forklift Forks*. URL: https://www.liftruck.co.uk/shop/forklift-forks/forklift-forks.html. (Accessed: 19.05.2021).

[13] *1803 mm (71 In) Bale grab*. URL: https://www.warrencat.com/new-equipment/attachments/bale-grabs/1803-mm-71-in-bale-grab/. (Accessed: 19.05.2021).

[14] Manfred Harrer, Peter Pfeffer. *Steering Handbook*. Springer International Publishing AG Switzerland. ISBN: ISBN 978-3-319-05448-3. DOI: 10.1007/978-3-319-05449-0.

[15] W. A. Wolfe. 'Analytical Design of an Ackermann Steering Linkage'. In: (1959).

[16] Wikipedia. *Ackermann Steering Geometry*. URL: https://en.wikipedia.org/wiki/Ackermann_steering_geometry. (Accessed: 19.05.2021).

[17] Carlos Cândido. 'Auxiliary mechanisms for telerobotics'. In: (2009). URL: https://www.researchgate.net/publication/247162034_Auxiliary_mechanisms_for_telerobotics.

[18] Arun Singh. 'Study of 4 Wheel Steering Systems to Reduce Turning Radius and Increase Stability'. In: (2014). URL: https://www.researchgate.net/profile/Rajiv-Chaudhary/publication/281450446_Study_of_4_Wheel_Steering_Systems_to_Reduce_Turning_Radius_and_Increase_Stability/links/55e8419008ae3e12184229f0/Study-of-4-Wheel-Steering-Systems-to-Reduce-Turning-Radius-and-Increase-Stability.pdf.

[19] Stephen L. Dickerson, Brett D. Lapin. 'Control of an Omni-Directional Robotic Vehicle with Mecanum Wheels'. In: (1991).

[20] Patent Yogi. *Reinventing wheels - Mecanum wheels that can move a vehicle in any direction*. URL: https://patentyogi.com/american-inventor/reinventing-wheels-mecanum-wheels-that-can-move-a-vehicle-in-any-direction/. (Accessed: 19.05.2021).

[21] Matthias Felden, Patrick Bütterling, Peter Jeck, Lutz Eckstein, Kay Hameyer. 'Electric vehicle drive trains: From the specification sheet to the drive-train concept'. In: (2010). DOI: 10.1109/EPEPEMC.2010.5606531.

[22] Claudio Annicchiarico , Mirko Rinchi , Stefano Pellari , Renzo Capitani. 'Design of a Semi Active Differential to Improve the Vehicle Dynamics'. In: (2014).

[23] David J Purdy, Dave Simner. 'Brief investigation into the effect on suspension motions of high unsprung mass'. In: *Journal of Battlefield Technology* (2004). ISSN: 1440-5113.

[24] Professor Dr.-Ing. habil. Hartmut Janocha (Ed.) *Actuators, Basics and Applications*. Universtat des Saarlandes. DOI: 10.1007/978-3-662-05587-8.

[25] Dogan Ibrahim. *Microcontroller Based Applied Digital Control*. John Wiley Sons Ltd. ISBN: 978-0-470-86335-0.

[26] STMicroelectronics. *STM32 32-bit Arm Cortex MCUs*. 2021. URL: hhttps://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html#overview. (Accessed: 19.12.2021).

[27] Yi-yuan Fang, Xue-jun Chen. 'Design and Simulation of UART Serial Communication Module Based on VHDL'. In: (2011). DOI: 10.1109/ISA.2011.5873448.

[28] ROS. *About ROS*. URL: https://www.ros.org/about-ros/. (Accessed: 19.05.2021).

[29] ROS. *Packages*. URL: https://wiki.ros.org/Packages. (Accessed: 19.05.2021).

[30] NVIDIA. *What Is Simultaneous Localization and Mapping?* URL: https://blogs.nvidia.com/blog/2019/07/25/what-is-simultaneous-localization-and-mapping-nvidia-jetson-isaac-sdk/. (Accessed: 19.05.2021).

[31] Veli Ilci Andand, Charles Toth. 'High Definition 3D Map Creation Using GNSS/IMU/LiDAR sensor Integration to Support Autonomous Vehicle Navigation'. In: (2019). DOI: 10.3390/s20030899.

[32] Xiaofeng Ren, Dieter Fox, Kurt Konolige. 'Change Their Perception'. In: (2013). DOI: 10.1109/MRA.2013.2253409.

[33] A. Kolb, R. Koch, R. Larsen. 'Time-of-Flight Cameras in Computer Graphics'. In: (2010). DOI: https://doi.org/10.1111/j.1467-8659.2009.01583.x.

[34] Manaf A. Mahammed, Amera I. Melhum, Faris A. Kochery. 'Object Distance Measurement by Stereo VISION'. In: *International Journal of Science and Applied Information Technology (IJSAIT)* 2.2 (2013). ISSN: 2278-3083.

[35] You li, Javier Ibanez-Guzman. 'Lidar for Autonomous Driving: The Principles, Challenges, and Trends for Automotive Lidar and Perception Systems'. In: (2020). DOI: 10.1109/MSP.2020.2973615.

[36] M. Kutila, P. Pyykönen, et al. 'Automotive LiDAR performance verification in fog and rain'. In: (2018). DOI: 10.1109/ITSC.2018.8569624.

[37] Norhafizan Ahmad, Raja Ariffin Raja Ghazilla, et al. 'Reviews on Various Inertial Measurement Unit (IMU) Sensor Application'. In: *International Journal of Signal Processing Systems* 1.2 (2013). DOI: 10.12720/ijsps.1.2.256-262.

[38] Daehee Won, Jongsun Ahn, et al. 'Performance Improvement of Inertial Navigation System by Using Magnetometer with vehicle Dynamic Constraints'. In: (2015). DOI: 10.1155/2015/435062.

[39] Mohammad O. A. Aqel, Mohammmad H. Marhaban, et al. 'Review of visual odometry: types, approaches, challenges, and applications'. In: (2016). DOI: 10.1186/s40064-016-3573-7.

[40] Ji Zhang, Sanjiv Singh. 'LOAM: Lidar Odometry and Mapping in Real-time'. In: (2014). DOI: 10.15607/RSS.2014.X.007.

[41] AvMaria M. P. Petrou, Costas Petrou. *Image Processing: The Fundamentals*. John Wiley Sons Ltd. ISBN: 978-0-470-74586-1.

[42] Xavier P., Burgos-Artizzu, et al. 'Real-time image processing for crop/weed discrimination in maize fields'. In: 75 (2011).

[43] Pulkit Sharma. *Image Classification vs. Object Detection vs. Image Segmentation*. 2019. URL: https://medium.com/analytics-vidhya/image-classification-vs-object-detection-vs-image-segmentation-f36db85fe81. (Accessed: 19.05.2021).

[44] Ian Goodfellow, Yoshua Bengio and Aaron Courvile. *Deep Learning*. MIT Press. ISBN: 978-0-262-03561-3.

[45] Patent Yogi. *Convolutional Neural Network — Deep Learning*. URL: https://developersbreach.com/convolution-neural-network-deep-learning/. (Accessed: 20.05.2021).

[46] MLNotebook. *Machine Learning Deep Learning Fundamentals*. URL: https://mlnotebook.github.io/post/CNN1/. (Accessed: 20.05.2021).

[47] 호롤리. *호다닥 공부해보는, CNN (Convolutional Neural Networks)*. URL: https://gruuuuu.github.io/machine-learning/cnn-doc/. (Accessed: 20.05.2021).

[48] Joseph Redmon and Ali Farhadi. 'YOLOv3: An Incremental Improvement'. In: *arXiv* (2018).

[49] Vidushi Meel. *YOLOv3: Real-Time Object Detection Algorithm (What's New?)* URL: https://viso.ai/deep-learning/yolov3-overview/. (Accessed: 9.12.2021).

[50] Ayoosh Kathuria. *What's new in YOLO v3*. URL: https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b. (Accessed: 9.12.2021).

[51] Great Learning Team, Arun K. *Understanding Data Augmentation — What is Data Augmentation and how it works?* 2020. URL: https://www.mygreatlearning.com/blog/understanding-data-augmentation/. (Accessed: 19.05.2021).

[52] Shorten C., Khoshgoftaar T.M. 'A survey on Image Data Augmentation for Deep Learning'. In: 6.60 (2019). DOI: https://doi.org/10.1186/s40537-019-0197-0.

[53] J. Wang, L. Perez. 'The Effectiveness of Data Augmentation in Image Classification using Deep Learning'. In: (2017). URL: https://arxiv.org/pdf/1712.04621.pdf.

[54] *Turtlebot3: Slam*. URL: https://emanual.robotis.com/docs/en/platform/turtlebot3/slam/. (Accessed: 21.05.2021).

Appendix A

GANTT Chart

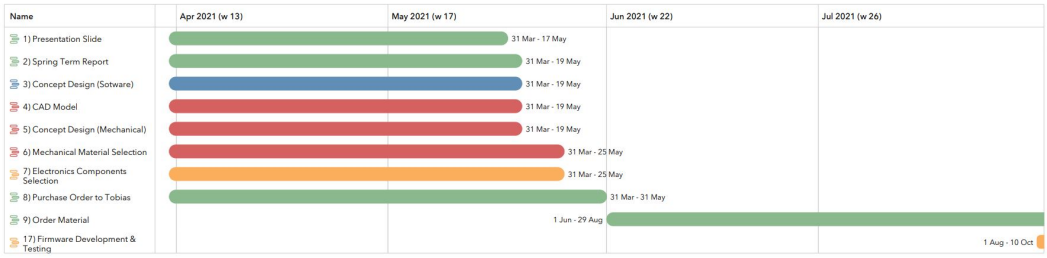

Figure A.1. GANTT chart of the spring time plan

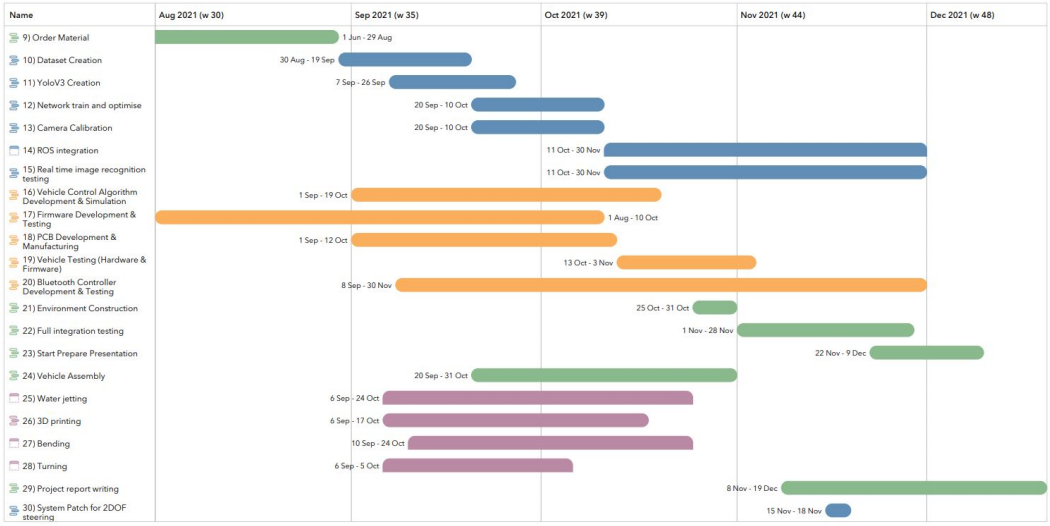

Figure A.2. GANTT chart of the fall time plan

Appendix B

Budget

Part No.	Part name	Subgroup	Notes	Comment	Quantity	Price/unit (SEK)	Price total (SEK)	Shipping (SEK)	Ordered (y/n)	Recieved (y/n)	Available (in sto	Link	Article No.	Store
1	L16-R Miniature Linear Servo	Mechatronics	Actuator	581,01	3	1200	3600	183,24	SANT	SANT	SANT	link	L16-R	Actuonix
2	L16-S Miniature Linear Actua	Mechatronics	1 or 2 actuators	Wrong length 5	2	1200	2400		SANT	SANT	SANT	link	L16-S	Actuonix
3	4pcs RC 2.2 Crawler Mud Tir	Mechanical	Wheels		1	482,93	482,93	366,44	SANT	SANT	SANT	link		Amazon
4	Nvidia Jetson AGX Xavier	Software	GPU	5828,74	1	14000	14000	423,79	SANT	SANT	SANT	link	945-82972-004	Arrow
5	KULLAGER 6001 2 RS 41-41	Mechanical	Bearing		8	49,9	399,2		SANT	SANT	SANT	link	6001 2RS 41-41	Biltema
6	NUCLEO-L476RG	Mechatronics	STM Microcontroller		2	181	362		SANT	SANT	SANT	link	497-15881-ND	Elfa
7	ZXBM5210-S-13	Mechatronics	H-Bridge		5	9,58	47,9		SANT	SANT	SANT	link	ZXBM5210-S-13	Digikey
8	ROB-09454	Mechatronics	Proximity sensor for bucket		3	24,52	73,56		SANT	SANT	SANT	link	1568-1554-ND	Digikey
9	MC7805CTG	Mechatronics	DC regulator 5V		5	4	20		SANT	SANT	FALSKT	link	MC7805CTGOS	Digikey
10	PRT-16764	Mechatronics	Pin Sockets 40 pin, Extended		10	9,24	92,4		SANT	SANT	SANT	link	1568-PRT-1676	Digikey
11	Breadboard Jumper Wire Kit	Mechatronics	Breadboard kit		1	117	117		SANT	SANT	SANT	link	301-15-119	Elfa
12	F7010 - Fuse 10A 32V Red, if	Mechatronics	Fuse, 10A		25	0,8982	22,455		SANT	SANT	SANT	link	301-71-841	Elfa
13	H7810 - Fuse Holder MiniOT	Mechatronics	Fuse holder, MiniOTO		3	15,6	46,8		SANT	SANT	SANT	link	110-29-328	Elfa
14	TMC2208 STEPPER MOTOR	Mechatronics	Stepper driver TMC2208		2	277,22	554,44		SANT	SANT	SANT	link	25040	3D Prima
15	Digital Servo 20kg / 0.16sec	Mechatronics	Digital servo		3	285	855		SANT	SANT	SANT	link	DL5020	KULLAGER
16	Nylon XT90 Connectors (1 pc	Mechatronics	XT90 connectors (pair)		3	25	75		SANT	SANT	SANT	link	C24523	KULLAGER
17	Turnigy Accucel-6 80W 10A 1	Mechatronics	Lipo balance charger, 1-6S		1	300,35	300,35		SANT	SANT	SANT	link	9052000155-0	Hobbyking
18	Luxorparts Delbar kopplings	Mechatronics	Jump Wires, male-female		1	89,9	89,9		SANT	SANT	SANT	link	87900	Kjell
19	Luxorparts Delbar kopplings	Mechatronics	Jump Wires, female-female		1	89,9	89,9		SANT	SANT	SANT	link	87906	Kjell
20	Bluetooth-modul för Arduino	Mechatronics	HC05 Bluetooth module	Received an Ard	1	199	199		SANT	SANT	SANT	link	87061	Kjell
21	Sheet metal (steel)	Mechanics	Received from KTH	2,0 x 1000 x 500	0	255	0		FALSKT	SANT	FALSKT	link	503105	Montano
22	Oisolerade DC/DC-omvandla	Mechatronics	DC regulator, 6A		1	132,27	132,27		SANT	SANT	SANT	link	894-APXW012A	Mouser
23	Nema 17 Stepper Bipolar L=	Mechatronics	Stepper motor		2	216,51	433,02		SANT	SANT	SANT	link	17HS13-0404S	Stepperonline
24	Zed2	Software	Camera		1	3740	3740		SANT	SANT	SANT	link		Stereolabs
25	Aluminium rod D12	Mechanical	Received from KTH		0	200	0		FALSKT	SANT	FALSKT			
26	Teflon rod	Mechanical	L=500mm Ø =12mm		1	480	480		SANT	SANT	FALSKT	link	##########	Maskindelen
27	Teflon rod	Mechanical	L=500mm Ø =30mm		1	80	80		SANT	SANT	FALSKT	link	##########	Maskindelen
28	Gens ace 6000mAh 18.5V 25	Mechatronics	Battery		2	1423,37	2846,74		SANT	SANT	SANT	link	GA00013	GensAce
29	Custom ordered PCBs	Mechatronics	PCBs		10	9,994	99,94		SANT	SANT	SANT	link		Seeed Studio
30	HC-SR04 Ultrasonic Distanc	Mechatronics	Ultrasonic Sensor		2	39,88	79,76		SANT	SANT	SANT	link	301-39-186	Elfa
31	Plexgear Multiadapter för USE	Mechatronics	USB-C docking station		1	599	599		SANT	SANT	SANT	link	62099	Kjell & CO
32	USB 3.0 Extenstion Cord	Software	USB Extension Cord		1	179,9	179,9		SANT	SANT	FALSKT			Kjell & CO
33	Kompostgaller	Environment	90x90x70		4	150	600		FALSKT	FALSKT	SANT	link	##########	K-rauta
34	Golvskyddpapp	Environment			1	100	100		FALSKT	FALSKT	SANT	link	##########	K-rauta
35	Dual band antenna	Mechatronics			4	25	100		SANT	FALSKT	SANT	link	##########	RS components
36	Intel 8265 m2 wifi module	Mechatronics			1	100	100		SANT	FALSKT	SANT	link	##########	RS components
37	Spray paint	General	1 yellow, 1 matte black		2	190	380							Järna

Figure B.1. The components and costs in the budget calculation

Budget:	50000	SEK
Parts cost:	33398,465	SEK
Shipping cost:	973,47	SEK
Total estimated cost:	34371,935	SEK
Actual spending:	34922	SEK
Remaining Budget:	15078	SEK

Figure B.2. The summary in the budget calculation

Appendix C

Functional Architecture

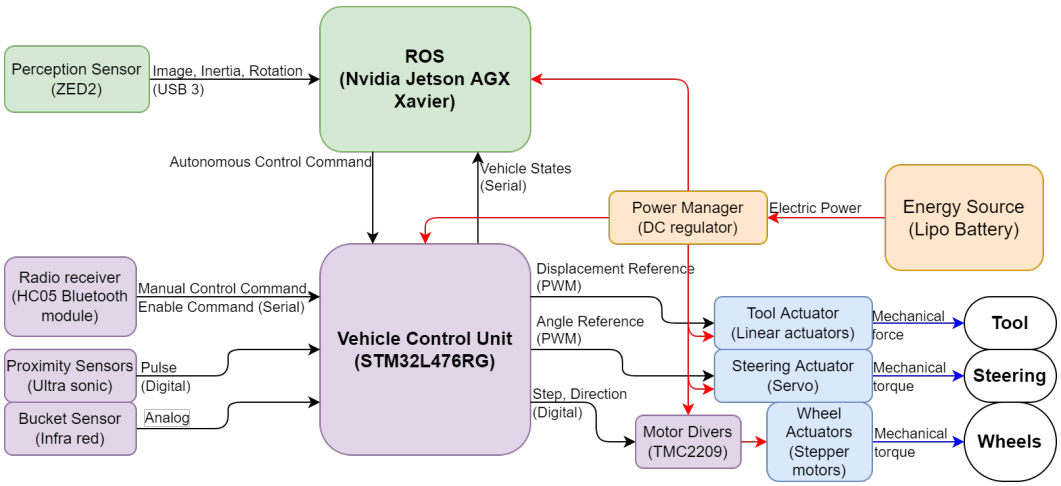

Figure C.1. Functional architecture

Appendix D

Evaluation of the Design Concepts

Table D.1. Weighted evaluation of the chassis concepts

		Zeux chassis		Stiff custom chassis		RC car chassis	
Criteria	Criteria Weight	Original Score	Weighted Score	Original Score	Weighted Score	Original Score	Weighted Score
Weight	25%	3	0,75	4	1	5	1,25
Reliability	87.5%	4	3.5	5	4.375	3	2.625
Complexity	75%	4	3	5	3.75	2	1.5
Cost	50%	3	1.5	4	2	1	0.5
Accessibility	12.5%	3	0.375	3	0.375	4	0.5
Weighted score for each concept			**9.125**		**11.5**		**6.375**

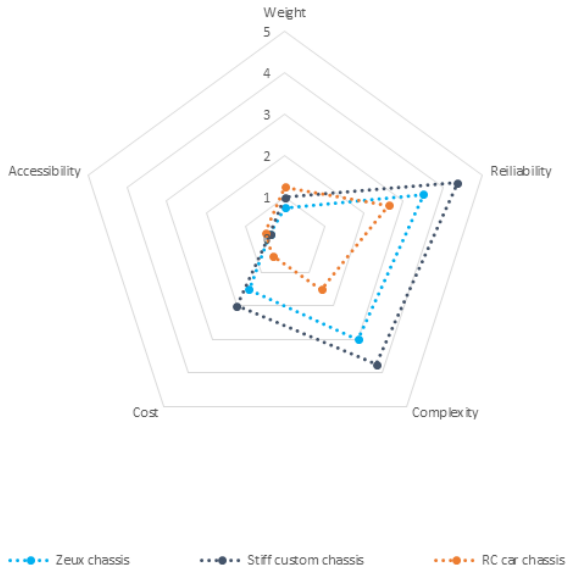

Figure D.1. Chassis concept evaluation diagram

Table D.2. Weighted evaluation of the lifting mechanism concepts

		Sliding system		Kinematic linkage	
Criteria	Criteria Weight	Original Score	Weighted Score	Original Score	Weighted Score
Cost	37.5%	3	1.125	4	1.5
Accessibility	12.5%	5	0.625	4	0.5
Flexibility	87.5%	3	2.625	5	4.375
Complexity	25%	5	1.25	4	1
Reliability	87.5%	3	2.625	5	4.375
Weighted score for each concept			**8.25**		**11.75**

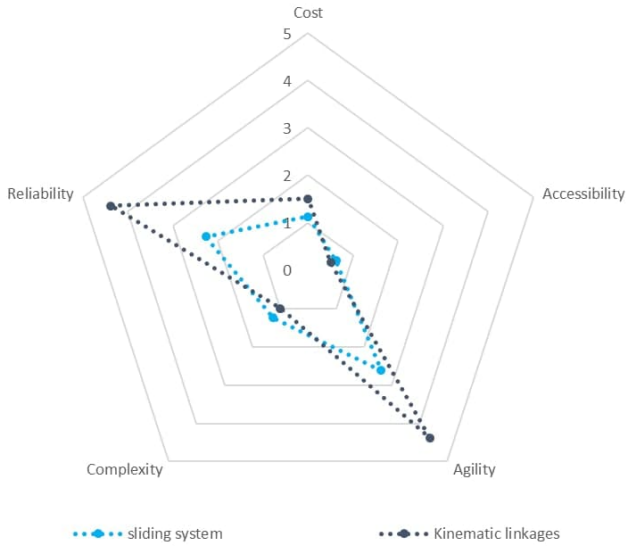

Figure D.2. Weighted lifting system evaluation diagram

Table D.3. Weighted evaluation of the steering concepts

		Mecanum wheels		Ackermann steering		Articulated steering	
Criteria	Criteria Weight	Original Score	Weighted Score	Original Score	Weighted Score	Original Score	Weighted Score
Cost	25%	2	0.5	4	1	4	1
Accessibility	50%	5	2.5	3	1.5	3	1.5
Weight/Dimension	0%	3	0	3	0	3	0
Controlability	75%	5	3.75	4	3	2	1.5
Reliability	100%	2	2	4	4	4	4
Weighted score for each concept			**8.75**		**9.5**		**8**

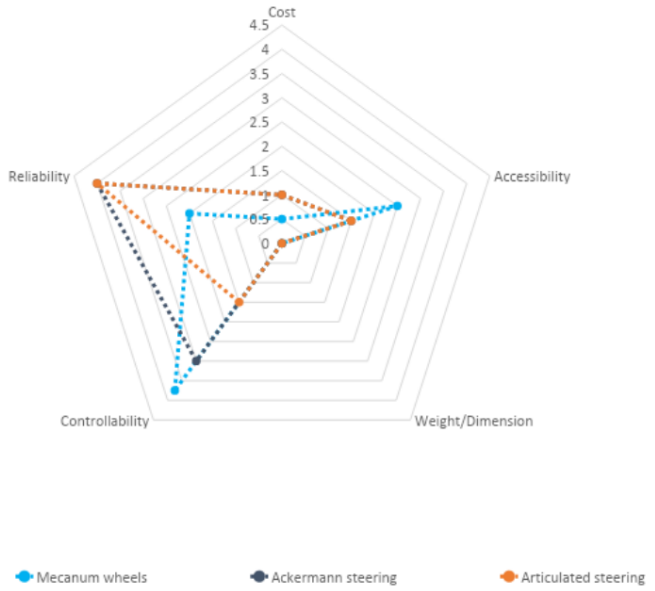

Figure D.3. Steering concept evaluation diagram

Table D.4. Weighted evaluation of the motor drive type concepts

		2WD,1M		2WD,2M		4WD,1M		4WD,2M		4WD,4M	
Criteria	Criteria Weight	Original Score	Weighted Score	Original Score	Weighted Score	Original Score	Weighted Score	Original Score	Weighted Score	Original Score	Weighted Score
Cost	37.5%	4	1.5	3	1.125	3	1.125	2	0.75	2	0.75
Accessibility	0%	3	0	5	0	1	0	2	0	4	0
Weight/Dimension	37.5%	5	1,875	3	1,125	4	1,5	3	1,125	2	0,75
Controlability	87.5%	3	2,625	5	4,375	2	1,75	3	2,625	4	3,5
Reliability	87.5%	3	2,625	4	3,5	2	1,75	3	2,625	5	4,375
Weighted score for each concept			**8.625**		**10.125**		**6.125**		**7.125**		**9.375**

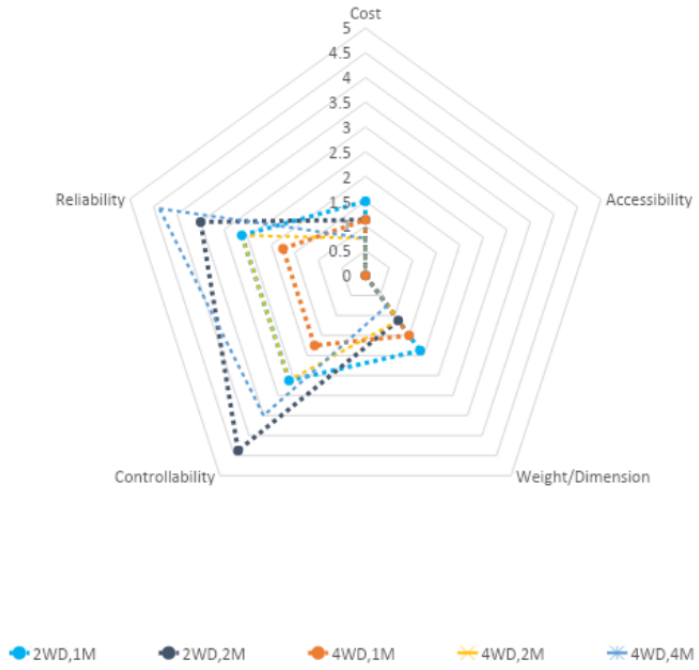

Figure D.4. Motor drive type concept evaluation diagram

Table D.5. Weighted evaluation of the propulsion motor types

		Brushless DC motor		Stepper motor		DC motor	
Criteria	Criteria Weight	Original Score	Weighted Score	Original Score	Weighted Score	Original Score	Weighted Score
Cost	37.5%	2	0.75	4	1.5	3	1.125
Accessibility	12.5%	2	0.25	5	0.625	4	0.5
Power/Weight	25%	5	1.25	2	0.5	4	1
Complexity	87.5%	3	2.625	5	4.375	3	2.625
Reliability	87,5%	5	4.375	3	2.625	3	2.625
Weighted score for each concept			**9.25**		**9.625**		**7.875**

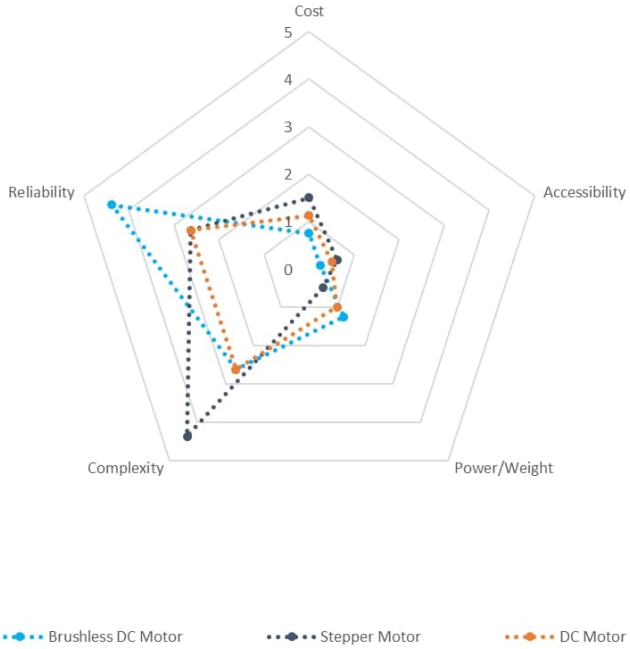

Figure D.5. Weighted propulsion motor evaluation diagram

Appendix E

Bluetooth

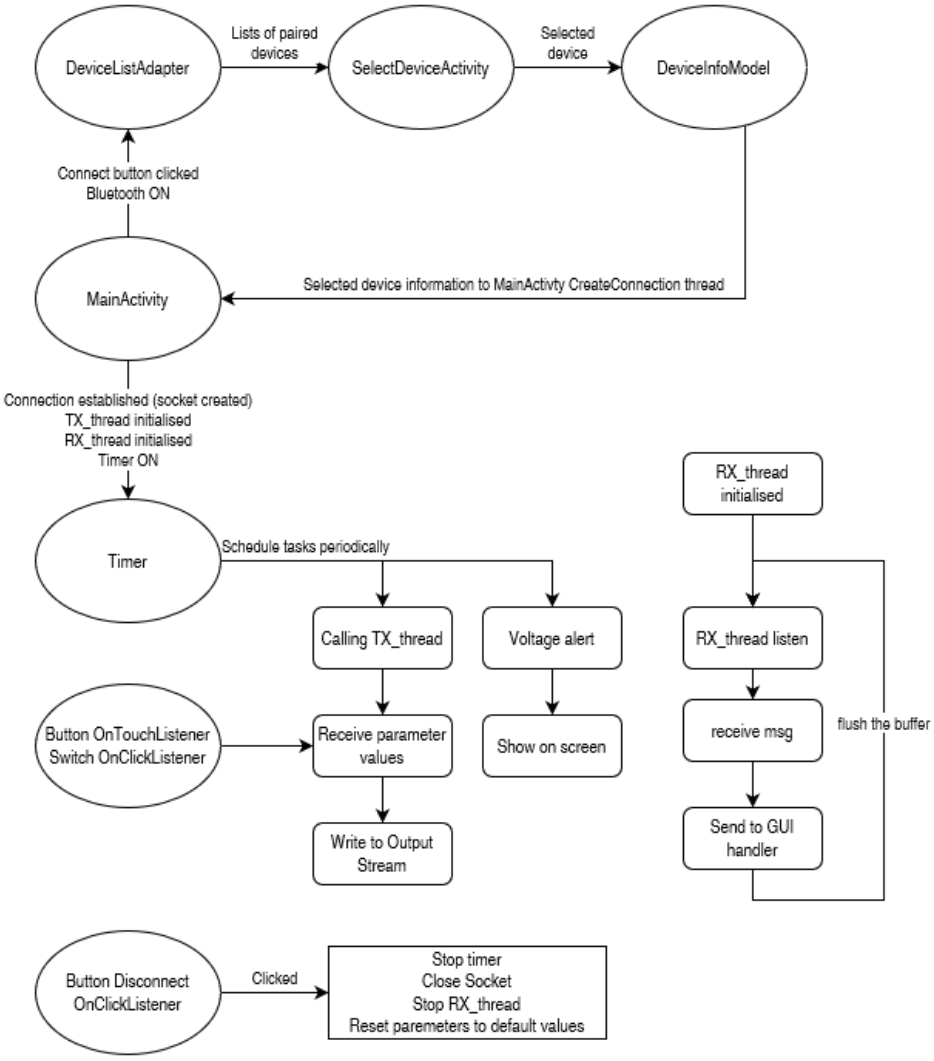

Figure E.1. Diagram of the work logic for the Bluetooth application