

Magnus Øye  
Vegard Solheim  
Petter Drønnen

# Autonomous inspections for process industry by a quadruped robot with use of neural networks

May 2020

**NTNU**

Norwegian University of Science and Technology  
Faculty of Information Technology and Electrical Engineering  
Department of ICT and Natural Sciences

**Bachelor's thesis**

**2020**







Magnus Øye  
Vegard Solheim  
Petter Drønnen

# **Autonomous inspections for process industry by a quadruped robot with use of neural networks**

Bachelor's thesis  
May 2020

**NTNU**

Norwegian University of Science and Technology  
Faculty of Information Technology and Electrical Engineering  
Department of ICT and Natural Sciences



Norwegian University of  
Science and Technology





Norwegian University of  
Science and Technology

**Autonomous inspections for process  
industry by a quadruped robot with use of  
neural networks**

Vegard Solheim, Petter Drønne, Magnus Kvendseth Øye

**Bachelor thesis**

Faculty of Information Technology and Electronics

Norwegian University of Science and Technology

Norway

May 19, 2020

Supervisor 1: Ivar Blindheim

Supervisor 2: Aleksander Skrede

Pages / Appendix

129 / 100

## Mandatory self declaration / group declaration

Every group member is responsible for familiarizing themselves with what is legal aids, guidelines for their use and rules about the use of sources. This statement should make the group members aware of their responsibilities and what consequences that can occur if cheating takes place. Failure to accept this declaration does not exempt students from their responsibilities.

<i>You fill out the declaration by clicking in the box to the right of each section 1-6:</i>		
1.	I/We hereby declare that my/our answer is my/our own work, and that I/we have not used any other sources or received any help other than that mentioned in the answer	<input checked="" type="checkbox"/>
2.	I/We declare that this reply: <ul style="list-style-type: none"> <li>• has not been used for another exam at another department/university/university college or abroad</li> <li>• does not refer to the work of others without being stated</li> <li>• does not refer to own previous work without it being stated</li> <li>• has all the references listed in the literature list</li> <li>• is not a copy, duplicate or copy of the work or response of others</li> </ul>	<input checked="" type="checkbox"/>
3.	I/We are aware that violations of the above are regarded as cheating and may result in the cancellation of examinations and exclusion from universities and colleges in Norway, jf. <a href="#">University and college Act §§4-7 and 4-8</a> and <a href="#">Examination regulations §§14 and 15</a>	<input checked="" type="checkbox"/>
4.	I/We are aware that all submitted assignments can be plagiarized in Ephorus, see Guidelines for electronic submission and publication of credits for studio assignments	<input checked="" type="checkbox"/>
5.	I/we are aware that ntnu university will handle all cases where there is suspicion of cheating according to the <a href="#">university's study regulations §31</a>	<input checked="" type="checkbox"/>
6.	I/we have familiarized ourselves with the rules and guidelines in the use of <a href="#">source and references on the library's website</a>	<input checked="" type="checkbox"/>

# Publishing Agreement

**Number of credits: 20**

**Supervisors: Ivar Blindheim, Aleksander Skrede**

## Authorization for electronic publication of the thesis

Author(s) have copyright on the thesis. This means, among other things, the exclusive right to make the work accessible to the general public ([Åndsverkloven §2](#)). All papers that fulfill the criteria will be registered and published in Brage HiM with the author(s) approval. Tasks that are except public or tape-free will not be published.

**I/we hereby grant NTNU in Aalesund a free right  
to make the thesis available for electronic publishing:**

YES  NO

**Is the assignment pledged (confidential)?**

YES  NO

(Placement agreement must be completed)

- If Yes:

**Can the thesis be published when the bonding period  
is over?**

YES  NO

**Is the assignment except the public?**

YES  NO

(Contains confidential information. [Jfr. Offl. §13 / Fvl. §13](#))

**Date: May 19, 2020**

## Preface

What intrigued us was building a prototype of a product from the ground up. This prototype should include several elements from what we have learned in school over the course of three years such as microcontrollers, programming and 3D modelling. Also, we wanted to show that we have learned other things outside the school such as ROS and inverse kinematics. Another important aspect was that this should be relevant to the process industry. This proved to be the case and large companies like [AkerBP](#) [1] and [Tennet](#) [76] are exploring the potential of similar robots in the oil and gas industry.

This report was written on NTNU campus Ålesund for NTNU in the subject IE303612 Bacheloroppgave. The project was started in January and finished in late May 2020. The group consisted of three students from Automatiseringsteknikk at NTNU Ålesund. All members have similar background from automation, and all three have certificate of apprenticeship in automation. We would like to inform the reader, that a general understanding of engineering, computer technology and automation is required to fully understand the content of this report.

Ålesund, May 19, 2020

---

Vegard Solheim

---

Petter Drønnen

---

Magnus Kvendseth Øye



## **Acknowledgement**

We would like to thank all contributors who have helped us during this project, and especially would we like to thank:

- Our supervisor Ivar Blindheim at NTNU for guidance throughout the project.
- Our supervisor Aleksander Skrede at NTNU for guidance throughout the project.
- Anders Sætersmoen at NTNU for the help with ordering parts, and lending equipment.
- Family and friends who have supported us throughout this period.
- Fellow students for good discussions about the project.

## **Executive summary**

This project aims to develop a quadruped robot prototype for autonomous inspections in the process industry with use of neural networks. This includes modelling and building a prototype that can equip sensors for collecting data. Including to the already mentioned tasks, multiple software system has been developed to handle: navigation, image processing, inverse kinematics, artificial intelligence and graphical user interfaces to aid in the inspection missions.

The results proves that the prototype is successful in its tasks. As the main goal is to make the robot successfully go through an inspection mission autonomously, and provide real time report status back to the graphical user interface where an operator can observe.

The prototype has accomplish many of the sub goals that were stated in the preliminary report. Utilizing concurrent processes and artificial intelligence, the robot manages to detect and classify all the equipment in the given task. With use of image processing it is able to extract tags and values from gauges with a high accuracy. In addition the robot is dynamic with self stabilization, and has an inverse kinematic model of the legs to move efficiently around.

This mentioned, there are many improvements to be made on both the design and software that can increase the reliability and performance of the system.

# Contents

Preface . . . . .	iii
Acknowledgement . . . . .	iv
Executive summary . . . . .	v
<b>1 Introduction</b>	<b>1</b>
1.1 Background . . . . .	1
1.2 Project introduction . . . . .	2
1.3 Aim and objectives . . . . .	2
1.4 Report content . . . . .	3
<b>2 Theory</b>	<b>4</b>
2.1 Robot kinematics . . . . .	4
2.2 Linear motion . . . . .	5
2.2.1 Timing Belt-driven motion . . . . .	6
2.3 Electric motors . . . . .	6
2.3.1 Brushless motor . . . . .	6
2.3.2 Motor driver . . . . .	6
2.4 Encoder . . . . .	6
2.4.1 Incremental encoder . . . . .	7
2.5 Communication protocols . . . . .	7
2.5.1 Serial . . . . .	7
2.5.2 TCP/IP . . . . .	7
2.5.3 REST-API . . . . .	8
2.6 Database . . . . .	8
2.7 Robot Operating System . . . . .	9

2.7.1	Nodes	9
2.7.2	Topics	9
2.7.3	Messages	9
2.8	Cloud Computing	10
2.9	Computer vision	10
2.9.1	Camera	10
2.9.2	Image processing	10
2.9.3	Region of interest	11
2.9.4	Segmentation	11
2.9.5	Morphology	12
2.9.6	Feature extraction	13
2.10	Optical Character Recognition	14
2.11	PID Controller	15
2.11.1	Operation	15
2.12	Quaternion	16
2.13	Visual Inertial Odometry	16
2.14	Navigation	17
2.14.1	Occupancy grid	18
2.14.2	Costmap	18
2.14.3	Pointcloud	18
2.15	Trajectory	19
2.15.1	A* Global planner	19
2.15.2	DWA Local planner	20
2.16	Machine learning	21
2.16.1	Neural networks	21
2.16.2	Input and hidden layers	22
2.16.3	Neurons	22
2.16.4	Output layer	23
2.16.5	Activation function	23
2.16.6	Deep neural network	24

2.16.7	Dataset	24
2.16.8	Multi-class classification	25
2.16.9	One hot encoding	25
2.16.10	Convolutional neural network	25
2.16.11	Convolution layer	26
2.16.12	Pooling layer	26
2.16.13	Fully connected layer	27
2.16.14	Transfer learning	27
2.16.15	Training	27
2.16.16	Evaluation	28
2.16.17	Intersection over union	29
2.17	Programming Language	30
2.17.1	Python	30
2.17.2	C++	30
2.17.3	SQL	30
<b>3</b>	<b>Materials</b>	<b>31</b>
3.1	Motor driver	31
3.2	Motors	31
3.3	Incremental rotary encoders	32
3.4	Jetson Nano	32
3.5	Teensy 4.0	33
3.6	Xbox One Controller	33
3.7	Remote computer	34
3.8	Camera	34
3.9	Construction and Electrical components	34
3.9.1	Batteries	35
3.10	Software and libraries	35
3.10.1	Software	35
3.10.2	Libraries	36

<b>4</b>	<b>Method</b>	<b>38</b>
4.1	Project Approach	38
4.2	Approach due to the Covid-19	39
4.3	Leg movement	40
4.3.1	Leg trajectory	40
4.3.2	Simulation	41
4.4	Convolutional neural networks	42
4.4.1	R-CNN	43
4.4.2	Fast R-CNN	44
4.4.3	Faster R-CNN	44
4.5	Review	46
4.6	Testing	46
4.6.1	Leg movement testing	46
4.6.2	Robot testing	47
4.6.3	Electrical testing	47
4.6.4	Communication testing	47
4.6.5	Image processing testing	48
4.6.6	Convolution neural network testing	48
4.7	Physical design	49
4.7.1	Robot	49
4.7.2	Stand	55
4.7.3	Electrial layout	56
4.8	Implementation	59
4.8.1	System overview	59
4.8.2	Graphical user interface	60
4.8.3	Robot program	66
4.8.4	Robot calibration	68
4.8.5	Robot motion	69
4.8.6	VIO	72
4.8.7	PID-Controllers	72

4.8.8	Xbox controller . . . . .	74
4.8.9	SQL Database . . . . .	76
4.8.10	REST-API . . . . .	76
4.8.11	Navigation . . . . .	80
4.8.12	Image processing . . . . .	81
4.8.13	Convolution neural network . . . . .	85
<b>5</b>	<b>Results</b>	<b>87</b>
5.1	Reviews . . . . .	87
5.1.1	Electrical testing . . . . .	87
5.1.2	System testing . . . . .	88
5.1.3	Communication . . . . .	89
5.1.4	Design reviews . . . . .	89
5.1.5	Software reviews . . . . .	90
5.2	Final results . . . . .	91
5.2.1	Work-space of the robot . . . . .	91
5.2.2	Roll and Pitch regulators . . . . .	92
5.2.3	Robot motion . . . . .	93
5.2.4	REST-API . . . . .	94
5.2.5	Mission workflow . . . . .	95
5.2.6	Performing mission . . . . .	97
5.2.7	Graphical user interface . . . . .	101
5.2.8	Software solution . . . . .	103
5.2.9	Convolution neural network . . . . .	106
<b>6</b>	<b>Discussion</b>	<b>110</b>
6.1	Motion . . . . .	110
6.2	Robot . . . . .	111
6.2.1	Robot type selection . . . . .	111
6.2.2	Robot selection basis . . . . .	112
6.2.3	Legs . . . . .	112

6.3	Physical structure and design	113
6.4	Robot accuracy	113
6.4.1	Gait controller	114
6.5	Communication protocol	114
6.6	Navigation	115
6.7	Remote computing	116
6.8	Software	116
6.9	Image processing	116
6.10	Optical Character Recognition	117
6.11	Convolution neural network	117
6.11.1	Dataset	117
6.11.2	Model	117
6.11.3	Evaluation	117
6.12	Improvements for the future	119
6.13	Experiences	119
6.13.1	Division of labor	119
<b>7</b>	<b>Conclusion</b>	<b>120</b>
	<b>Appendices</b>	<b>130</b>
<b>A</b>	<b>Reports</b>	<b>131</b>
A.1	Preproject report	131
A.2	Risk analysis Covid-19	159
A.3	Gantt diagram	165
A.4	Progress reports	167
A.4.1	Progress report 19.01.20	167
A.4.2	Progress report 09.02.20	169
A.4.3	Progress report 01.03.20	171
A.4.4	Progress report 15.03.20	174
A.4.5	Progress report 29.03.20	177



A.4.6	Progress report 12.04.20 . . . . .	180
A.4.7	Progress report 19.04.20 . . . . .	183
A.4.8	Progress report 03.05.20 . . . . .	186
A.4.9	Progress report 19.05.20 . . . . .	188
A.5	Meeting reports . . . . .	190
A.5.1	Meeting report 08.01.20 . . . . .	190
A.5.2	Meeting report 14.01.20 . . . . .	193
A.5.3	Meeting report 05.02.20 . . . . .	196
A.5.4	Meeting report 19.02.20 . . . . .	199
A.5.5	Meeting report 04.03.20 . . . . .	202
A.5.6	Meeting report 15.04.20 . . . . .	205
A.5.7	Meeting report 23.04.20 . . . . .	208
A.5.8	Meeting report 07.05.20 . . . . .	212
<b>B</b>	<b>Bill of material (BOM)</b>	<b>215</b>
B.1	BOM . . . . .	215
<b>C</b>	<b>Pictures and progress videos</b>	<b>219</b>
C.1	Pictures . . . . .	219
C.2	Progress videos . . . . .	219
<b>D</b>	<b>Drawings</b>	<b>220</b>
D.1	Mechanical drawings . . . . .	220
D.2	Electrical drawings . . . . .	220
<b>E</b>	<b>Source Code</b>	<b>229</b>
E.1	Source Code . . . . .	229

## Terminology and Abbreviations

**CV** Computer Vision

**GUI** Graphical User Interface

**HDR** High Dynamic Range

**CAD** Computer Aided Design. A tool used to model virtual objects.

**AI** Artificial intelligence is the simulation of human intelligence processes by machines, especially computer systems.

**CNN** Convolutional Neural Network is a Deep Learning algorithm which can take in an input image.

**SVM** Support vector machine

**RPN** Region Proposal Network

**PCA** Principal Component Analysis

**maP** Mean average Precision

**maR** Mean average Recall

**MCU** Microcontroller is a compact integrated circuit designed to govern a specific operation in an embedded system.

**DWA** Dynamic Window Approach

**PID** Proportional integral derivative controller

**API** Application Programming Interface, activates functions from a remote software.

**TCP** Transmission Control Protocol, connection oriented transmission protocol of information.

**UDP** User Datagram Protocol, non connection based transmission protocol of information.

**IP** Internet Protocol is a "best effort" delivery protocol

**LIDAR** Light detection and ranging. A sensor used to record distance.

**Pose** A pose defines an objects orientation and position in operational space.

**OSI** Open Systems Interconnection

**IMU** Inertial measurement unit used to calculate the orientation of the body to an object

**DOF** Degrees of Freedom, number of configurations for a object

**RMS** Root-Mean-Squared

**FPV** First-Person View

**VIO** Visual-Inertial Odometry

## List of Figures

2.1	Forward kinematics	5
2.2	Inverse kinematics	5
2.3	REST-API	8
2.4	Image processing [81]	11
2.5	Morphology [82]	12
2.6	Optical Character Recognition	14
2.7	PID Controller	15
2.8	The quaternion plane [101]	16
2.9	Visual Inertial Odometry[72]	17
2.10	Trajectories from the velocities [53]	20
2.11	Different sort of activation functions [58]	23
2.12	Convolutional neural network [67]	26

2.13	Confusion matrix	28
3.1	Odrive V3.6 [83]	31
3.2	MN5212 KV340 [75]	31
3.3	Encoder [3]	32
3.4	Jetson Nano [45]	32
3.5	Teensy 4.0 [54]	33
3.6	Xbox One Controller [40]	33
3.7	Intel D435 [25]	34
3.8	Intel T265[26]	34
3.9	Battery [13]	35
4.1	Leg Trajectories	40
4.2	One point simulation	41
4.3	Simulation	42
4.4	Faster RCNN Architecture	45
4.5	Front Right View	49
4.6	All parts of the body	50
4.7	Main frame of the robot	51
4.8	The front of the robot	51
4.9	Sides of the robot	52
4.10	Actuator section view	53
4.11	Motors and pulleys	53
4.12	Nylon Legs	54
4.13	Aluminium legs	54
4.14	Robot stand	55
4.15	Robot stand	55
4.16	Electrical Assembly Plate	56
4.17	Camera Assembly Front	57
4.18	Camera Assembly Rear	57
4.19	On Board Control Panel	58

4.20 System Overview . . . . .	59
4.21 Graphical User Interface . . . . .	60
4.22 Create new mission . . . . .	62
4.23 Cloud computer overview . . . . .	63
4.24 Robot controller overview . . . . .	64
4.25 State machine overview . . . . .	65
4.26 Main operation flow . . . . .	67
4.27 Main operation . . . . .	68
4.28 Trajectory Control on motors . . . . .	69
4.29 Gait controller . . . . .	70
4.30 Inverse Kinematic . . . . .	71
4.31 PID for motors [85] . . . . .	73
4.32 PID for pitch and roll . . . . .	73
4.33 Xbox controller buttons . . . . .	74
4.34 Xbox controller mapping . . . . .	75
4.35 Database structure . . . . .	76
4.36 Database models . . . . .	76
4.37 REST-API Login . . . . .	77
4.38 REST-API Get . . . . .	77
4.39 REST-API Post . . . . .	78
4.40 REST-API Delete . . . . .	79
4.41 Navigation . . . . .	80
4.42 Gauge value detection . . . . .	81
4.43 Gauge input image . . . . .	82
4.44 Gauge image gray . . . . .	82
4.45 Gauge found circles . . . . .	82
4.46 Gauge preprocessing . . . . .	82
4.47 Gauge threshold . . . . .	83
4.48 Gauge lines found . . . . .	83
4.49 Preprocessing for OCR . . . . .	84

4.50 Training example 1 . . . . .	85
4.51 Training example 2 . . . . .	85
4.52 Training example 3 . . . . .	85
4.53 Training example 1 . . . . .	85
4.54 Training example 2 . . . . .	85
4.55 Training example 3 . . . . .	85
4.56 CNN Evaluation Example 1 . . . . .	86
4.57 CNN Evaluation Example 2 . . . . .	86
4.58 CNN Evaluation Example 3 . . . . .	86
5.1 Final robot . . . . .	91
5.2 Mission planer result . . . . .	95
5.3 Final Mission . . . . .	97
5.4 Final Mission protocol . . . . .	97
5.5 Linear position result . . . . .	98
5.6 Orientation z result . . . . .	99
5.7 Orientation w result . . . . .	99
5.8 Fire extinguisher result . . . . .	100
5.9 Gauge result . . . . .	100
5.10 Exit sign result . . . . .	100
5.11 Closed valve result . . . . .	100
5.12 GUI Manual result . . . . .	101
5.13 GUI FPV result . . . . .	101
5.14 GUI Inspection result . . . . .	102
5.15 Total training loss . . . . .	106
5.16 Confusion matrix . . . . .	107
5.17 CNN Precision . . . . .	107
5.18 CNN Sensitivity . . . . .	108
5.19 CNN Accuracy . . . . .	108
5.20 CNN Result Example 1 . . . . .	109

5.21 CNN Result Example 2 . . . . .	109
6.1 Optimized gait controller . . . . .	114

## List of Tables

2.1 Quaternion multiplication [101] . . . . .	16
4.1 Training data information . . . . .	86
5.1 System testing . . . . .	88
5.2 Pitch Regulator . . . . .	92
5.3 Roll Regulator . . . . .	92
5.4 REST API calls . . . . .	94
5.5 Navigation point coordinates . . . . .	98
5.6 Navigation angle coordinates . . . . .	98
5.7 OCR processing results . . . . .	99
5.8 Detection and classification results . . . . .	99
5.9 ROS topic publish frequency . . . . .	104
5.10 Loop times on the MCU . . . . .	105

# Chapter 1

## Introduction

### 1.1 Background

The topic of this thesis is to investigate the viability of using a quadruped robot equipped with sensors to conduct autonomous inspections missions with use of neural networks and image processing for analyzing data. Achieving this will prove that robots can be used in the process industry for routine inspections, or do autonomous mission that would otherwise risk human life.

These types of robots are in the initial phase of development, and the development of these robot is expensive. This thesis will utilize non industrial components and build an inexpensive robot that can be used for testing and research. Throughout this thesis, will we explore the viability of using such a robot in industrial applications.

Several companies like AkerBP[1] and TenneT[76] are just in the initial staring phase of testing the potential of these robots on some of their offshore platforms.



## 1.2 Project introduction

This thesis seeks to prove that an autonomous mobile robot can optimize inspection operations in a process industry. With doing so, eliminating unnecessary human interaction, as this can be dangerous, complicated and expensive. In this paper the emphasis will be on design, machine learning, and finally the usability of the robot. During consideration of different choices, the priority will be on combining robustness, speed and accuracy with minimal trade-off.

## 1.3 Aim and objectives

The aim for this thesis is to build a prototype of an autonomous mobile robot as proof of concept for future automated solutions in the process industry. The robot will operate either autonomously or manually. The main task will be to move around, detect, classify and inspect equipment as fast and accurate as possible. Whereas the core objectives for creating the robot is mentioned below. The robot should be able to:

- Stabilize when standing
- Walk and turn with input from an Xbox controller
- Walk and turn with input from the trajectory planner
- Avoid collisions
- Detect and classify equipment
- Extract a value from a gauge
- Read tag of equipment
- Run autonomous missions

## 1.4 Report content

This thesis is structured in the following way:

**Chapter 1 - Introduction** - The background and motivation for this thesis is presented together with the problem description.

**Chapter 2 - Theory** - The theory for all the technical aspects of this thesis is presented.

**Chapter 3 - Materials** - Contains the materials, components and information used for creating the prototype in this project.

**Chapter 4 - Method** - Methods used to perform tests and development of the different solutions is presented.

**Chapter 5 - Results** - All test results and solutions are presented.

**Chapter 6 - Discussion** - A discussion regarding the different solutions and test results obtained is made. Some personal thoughts regarding the different solutions are presented.

**Chapter 7 - Conclusions** - The thesis work is concluded.

# Chapter 2

## Theory

This chapter contains the theoretical basis that is needed for making decisions throughout the project.

### 2.1 Robot kinematics

Kinematics describes the motion of a robot regarding position, velocity, and acceleration of objects and whole groups of objects. Kinematics does not consider the causes of motion and thereby does not consider the force or torque required for generating motion. When calculating robot kinematics, coordinate system transformation is often used to simplify the solution. This lets several grouped objects to be represented within a common coordinate system. The objects can move within other coordinate systems without needing to calculate every object in one large coordinate system.

Figure 2.1 and 2.2 illustrates the transformation from a joint coordinate system to a coordinate system representing real-world coordinates. A joint coordinate system defines each joint with its position or angle. The real world coordinate system uses Cartesian coordinates and has its origin at the base of the robots frame. In robot kinematics, there are two main problems that needs to be solved, which is forward kinematics and inverse kinematics [103].

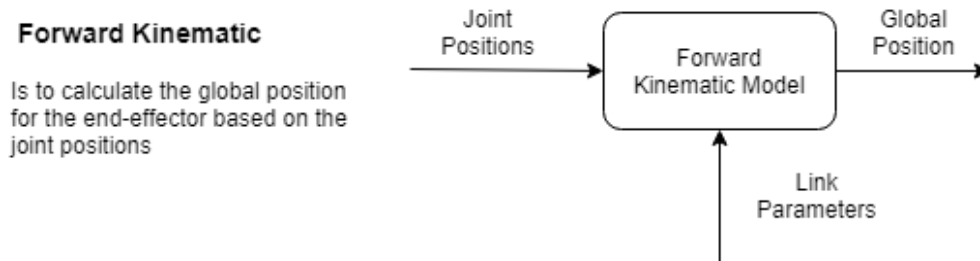


Figure 2.1: Forward kinematics

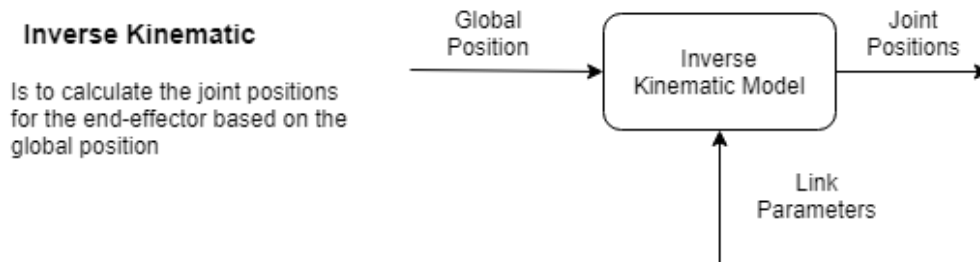


Figure 2.2: Inverse kinematics

## 2.2 Linear motion

Some electrical machines require linear movement to work efficient. Because most electrical actuators provide rotational motion, it is necessary to convert rotational motion into accurate linear motion. There are several ways of achieving this conversion. Some of the commonly used techniques are either by belt-driven actuators, ball screw driven actuators, or by rack and pinion driven actuators. The three actuator implementations, all apply linear motion by rotary motors, but they differ in strengths and limitations. Accuracy provided by the different methods depends on the actuator providing the revolution motion[37].

### **2.2.1 Timing Belt-driven motion**

Timing belts are toothed belts often used to transfer mechanical power. They can be mounted on two or more toothed wheels with a matching tooth profile, often called sprockets, letting the belt mesh with the sprocket. Use of timing belt offers synchronization between the rotation of the sprockets it is mounted around, while also allowing for gearing ratios by using different sizes of sprockets [99].

## **2.3 Electric motors**

### **2.3.1 Brushless motor**

A brushless motor is asynchronous motor powered by direct current (DC) electricity via an inverter, or switching power supply which produces an alternating current (AC) electric current to drive each phase of the motor via a closed loop controller. The controller provides pulses of current to the motor windings that control the speed and torque of the motor [92].

### **2.3.2 Motor driver**

Motor drivers controls the torque, speed, and direction of rotary and linear electric motors. They function by taking a low-current control signal and turning it into a higher-current signal that drives the motor [68].

## **2.4 Encoder**

An encoder is a sensing device that provides feedback on motion. Encoders convert motion to an electrical signal that can be read by some types of control devices in a motion control system, such as a counter or PLC. The encoder sends a feedback signal that can be used to determine position, count, speed, or direction. A control device can use this information to send a command for a particular function [17].

### 2.4.1 Incremental encoder

An incremental encoder generates a pulse for each incremental step in its rotation. Although the incremental encoder does not output absolute position, it can provide high resolution. For example, an incremental encoder with a single code track, referred to as a tachometer encoder, generates a pulse signal whose frequency indicates the velocity of displacement [51].

## 2.5 Communication protocols

### 2.5.1 Serial

What characterizes serial communication is that it transmits one bit at a time, compared to parallel communication which sends several bits at a time. The two main types of serial protocols are based on Synchronous and Asynchronous communication [27].

Synchronous Serial has at least two wires, where one of them is a clock signal wire paired with a data signal wire. This means all the devices follows the external clock and all the transfers will be based on this clock. It makes synchronizing the speeds easier and the protocol more straightforward [27].

The second type, Asynchronous Serial simply means that data is transferred without an external clock being synced. Because of this, more steps is taken in the protocol to ensure reliable transfer and receiving of data [27].

### 2.5.2 TCP/IP

TCP stands for Transmission Control Protocol and is a connection oriented protocol that belongs to the transport layer in the Open Systems Interconnection (OSI) model. TCP is a standard that defines how to establish and maintain communication between devices, and transfers data by creating one fixed connection between two host units. After the connection is established, the devices can communicate continuously while the connection is active. TCP provides reliable

communication, where TCP ensures that all data is delivered to the recipient and in properly order [66].

IP stands for Internet Protocol and belongs to the network layer in the OSI model. IP defines what, how, and in what way machines should communicate. Each host unit has its own unique IP address that identifies it from other devices in the Internet. This along with TCP defines how computers send packets of data to each other [77].

### 2.5.3 REST-API

A RESTful API is an application program interface (API) that uses HTTP requests to GET, PUT, POST and DELETE data. This implies that that a back-end utilizes a API as a middle layer from the logic and the user. The user can utilize the functionality to either request for data, update, store or delete data as shown in figure 2.3. [90].

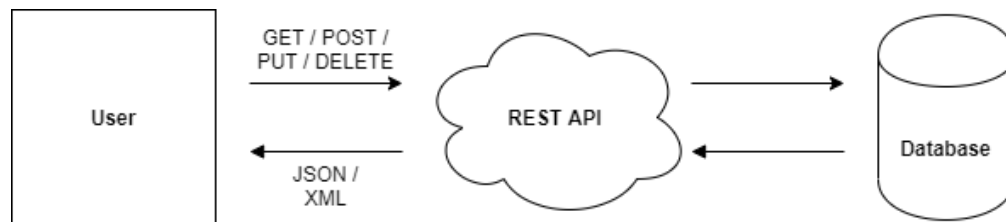


Figure 2.3: REST-API

## 2.6 Database

A database is an organized collection of data, generally stored and accessed electronically from a computer system. Where databases are more complex they are often developed using formal design and modeling techniques. Data within the most common types of databases in operation today is typically modeled in rows and columns in a series of tables to make processing and data querying efficient. The data can then easily be accessed, managed, modified, updated, controlled, and organized [48].

## 2.7 Robot Operating System

Robot Operating System (ROS), is a large, community developed framework that aims to make writing code for robots easier. ROS is a collection of tools, libraries and conventions that is put together to aid the development of robot software. The goal of the ROS project is to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms. In order to fully grasp the concept of ROS, a couple of key elements needs to be explained further [62].

### 2.7.1 Nodes

An individual program (or executable) is referred to as a node in the ROS context. ROS allows users to run many nodes simultaneously, and handles the communication between nodes internally. This means that a complex robot setup can be run by several individual nodes cooperating to achieve a wanted behavior [64].

### 2.7.2 Topics

Topics in the ROS context are comparable to data streams. Topics are methods used for communication between different applications. ROS allows for both publication and subscription of topics. The mechanics of this convention is that a node can publish data to a topic, which is automatically sent to all nodes subscribed to that particular topic. The data contained in a topic is defined in the message type of that given topic [65].

### 2.7.3 Messages

A message is a simple data structure, comprising typed fields. Messages are developer defined data types often consisting of multiple variables of different type. This means that a single message can contain many different variables of different types [63].



## 2.8 Cloud Computing

Cloud computing is the on-demand availability of computer system resources, especially data storage and computing power, without direct active management by the user. If the connection to the user is relatively close, it may be designated as an edge server [87].

## 2.9 Computer vision

Computer vision is a machines way of solving complex vision problems without human interaction. This is done by gaining visual information from digital images. All applications that use information gained from image processing in order to make decisions, uses computer vision [5].

### 2.9.1 Camera

Cameras are the sensors in computer vision, and are used for producing images for further processing. Various factors in the application affects which camera is best suited for image capturing. These factors is among other things, light, surrounding medium, available space and weight. Regarding camera types, they vary in features as resolution, image sensor, frame rate, shutting technique and the ability to capture color [86].

### 2.9.2 Image processing

The use of image processing can be separated into two application areas: human vision applications and computer vision. What these application areas have in common is that they are both used for image analysis. Human vision applications have humans as the end user, thus limiting the amount of information that can be extracted from the image to what is possible to see with the eye. With computer vision applications, the end user is a computer. Large amount of information is possible to be extracted, as a computer is capable of revealing almost the entire electromagnetic spectrum, while the human eye is limited to detecting only visible wavelengths. Another advantage computer vision has over human vision is the ability to neglect the features that are irrelevant for the task at hand [81].

A Venn diagram between computer and human vision are shown in figure 2.4.



Figure 2.4: Image processing [81]

Computer vision consists of a variety of methods and techniques used for manipulating and extracting information from digital images. Often, image processing is used for computer vision where the result of the processor determines the next action of the machine. Processing techniques as segmentation and morphology, are used for finding features in images [81].

### 2.9.3 Region of interest

A region of interest (ROI) is a portion of an image that you want to filter or perform some other operation upon. The process usually consists of the removal of some of the peripheral areas of an image to remove extraneous trash from the picture, to improve its framing, to change the aspect ratio, or to accentuate or isolate the subject matter from its background [91].

### 2.9.4 Segmentation

Segmentation is the techniques used for partitioning images into segments, finding objects or boundaries in images, the segments are then labeled making them easier to separate from the image.

Thresholding is the easiest way for segmenting or binarizing a image. It uses the gray-level or color-level for selecting what should be retained when making the image into binary values, 0 and 1. The gray-level used for deciding what to retain and what to discard is either set manually by the user or by adaptive methods using local or global features in the image. Manual thresholding is a point-operation, meaning that it checks the gray-level of each pixel. If the gray-level is above the threshold level the pixel is set to the binary value 1, otherwise it is set to the binary value 0. Adaptive thresholding techniques either calculates a thresholding level for the entire image, or for all local areas in the image. The thresholding value is calculated by a predefined feature, such as skew or variance [60].

### 2.9.5 Morphology

Morphological operations are used for changing pixel values in gray-scale and binary images relative to the value of the neighbour pixels. As the operations uses the neighbour pixels for alternating a pixel value, and not the pixel value itself, it is well suited for use on binary images. A structure element also called template is used for controlling which neighbour pixels that are being considered [15].

Erosion and dilation are two basic morphology techniques used for alternating the size of the blobs in binary images. The erosion operation has the effect of shrinking the size of binary objects, while dilation has the opposite effect and enlarges the size of binary objects. Figure 2.5 shows a small binary image before and after the morphological operations erosion and dilation has been used. The same structure element is used for both erosion and dilation [59].

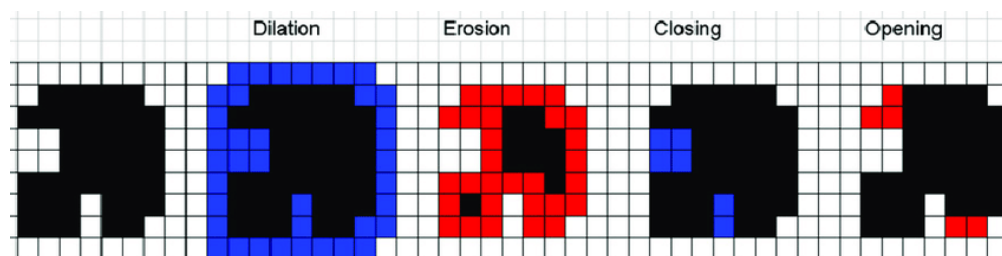


Figure 2.5: Morphology [82]

### 2.9.6 Feature extraction

Feature extraction is a process of dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for processing. A characteristic of these large data sets is a large number of variables that require a lot of computing resources to process [11]. There are multiple algorithms that can extract features from images, and some are listed here:

- **Edge detection** - identifying points at which the image brightness changes sharply or, more formally, has discontinuities [96].
- **Corner detection** - by considering the differential of the corner score with respect to direction directly, instead of using shifted patches [97].
- **Blob detection** - detecting regions that differ in properties, such as brightness or color, compared to surrounding regions. Informally, a blob is a region of an image in which some properties are constant or approximately constant; all the points in a blob can be considered in some sense to be similar to each other [98].

## 2.10 Optical Character Recognition

Optical character recognition (OCR) is the electronic or mechanical conversion of images on typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo or from subtitle text superimposed on an image. The structure of an OCR can be seen in figure 2.6.

Feature extraction decomposes glyphs into "features" like lines, closed loops, line direction, and line intersections. The extraction features reduces the dimensionality of the representation and makes the recognition process computationally efficient. These features are compared with an abstract vector-like representation of a character, which might reduce to one or more glyph prototypes. Nearest neighbour classifiers such as the k-nearest neighbors algorithm are used to compare image features with stored glyph features and choose the nearest match [43].

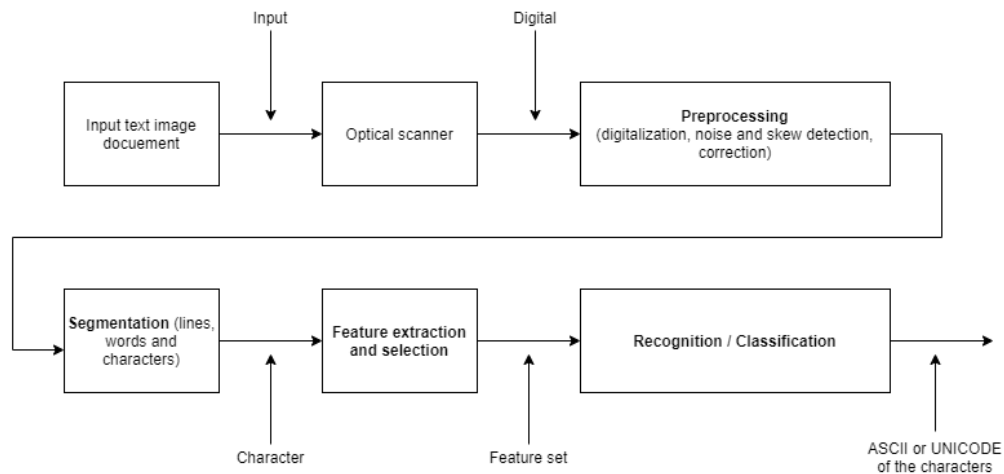


Figure 2.6: Optical Character Recognition

## 2.11 PID Controller

A proportional integral derivative controller (PID controller) is a feedback mechanism used in industrial control systems and a variety of other applications that require continuous modulated control. A PID controller continuously calculates an error value  $e(t)$  that is the difference between the desired set point ( $SP$ ) and a measured process variable ( $PV$ ) and uses a correction based on proportional, integral and derivative conditions to make the error value approach zero [93].

### 2.11.1 Operation

The feature of the PID controller is the ability to use the three control conditions proportional, integral and derivative influence on the regulator output to apply accurate and optimal control. Figure 2.7 illustrates a block diagram showing the principles for how these concepts are generated and used. It shows a PID controller that continuously calculates an error value  $e(t)$  which is the difference between a desired set point  $r(t)$  and a measured process variable  $y(t)$ , and uses a correction based on proportional, integral and derivatives. As a result, the controller will try to minimize the error over time by adjusting a control variable  $u(t)$  [93].

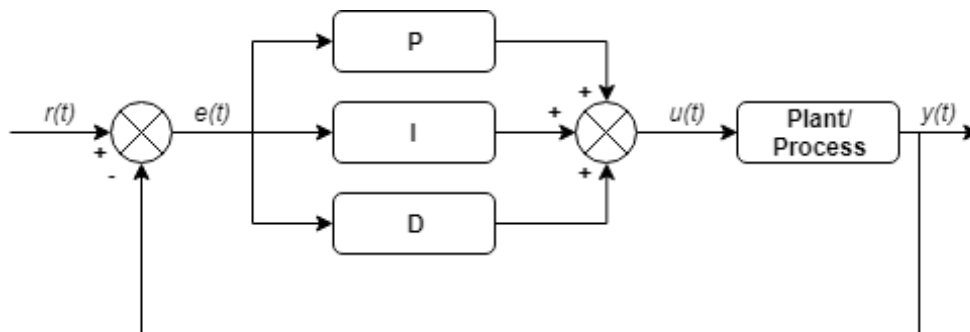
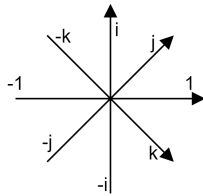


Figure 2.7: PID Controller

## 2.12 Quaternion

Quaternion is a way to extend the complex numbers, and can be seen in figure 2.8. Quaternions are often used for telling the 3D orientation of an object. The multiplication of quaternions are shown in table 2.1.



$\times$	1	i	j	k
1	1	i	j	k
i	i	-1	k	-j
j	j	-k	-1	i
k	k	j	-i	-1

Figure 2.8: The quaternion plane [101]

Table 2.1: Quaternion multiplication [101]

Quaternions are generally represented in the form shown in equation 2.1. Where  $a$ ,  $b$ ,  $c$ , and  $d$  are real numbers, and  $i$ ,  $j$ , and  $k$  are the fundamental quaternion units [101].

$$a + bi + cj + dk \quad (2.1)$$

## 2.13 Visual Inertial Odometry

Visual-Inertial Odometry (VIO) is the process of estimating the state (pose and velocity) of an agent (e.g., an aerial robot) by using only the input of one or more cameras in addition to one or more Inertial Measurement Units (IMUs) attached to it. VIO is the only viable alternative to GPS and lidar-based odometry to achieve accurate state estimation [72].

Cameras and IMUs are complementary sensor types. A camera accumulates the photons during the exposure time to get a 2D image. Therefore they are precise during slow motion and provide rich information, which is useful for other perception tasks, such as place recognition. However, they have limited output rate ( $\sim 100\text{Hz}$ ), suffer from scale ambiguity in a monocular setup, and are not robust to scenes characterized by low texture, high speed motions (due to motion blur) and High Dynamic Range (HDR) (which may cause over- or under-exposure of the image). By contrast, an IMU is a proprioceptive sensor measuring the angular velocity and the external acceleration acting upon it. An IMU is scene-independent, which renders it unaffected by the aforementioned difficulties for cameras. Thus, it is the ideal complement to cameras to achieve robustness in low texture, high speed, and HDR scenarios. Additionally, an IMU has high output rate ( $\sim 1,000\text{Hz}$ ). However, it suffers from poor signal-noise ratio at low accelerations and low angular velocities. Due to the presence of sensor biases, the motion estimated from an IMU alone tends to accumulate drift quickly. Therefore, a combination of both cameras and IMUs, such as in figure 2.9, can provide accurate and robust state estimation in different situations [72].

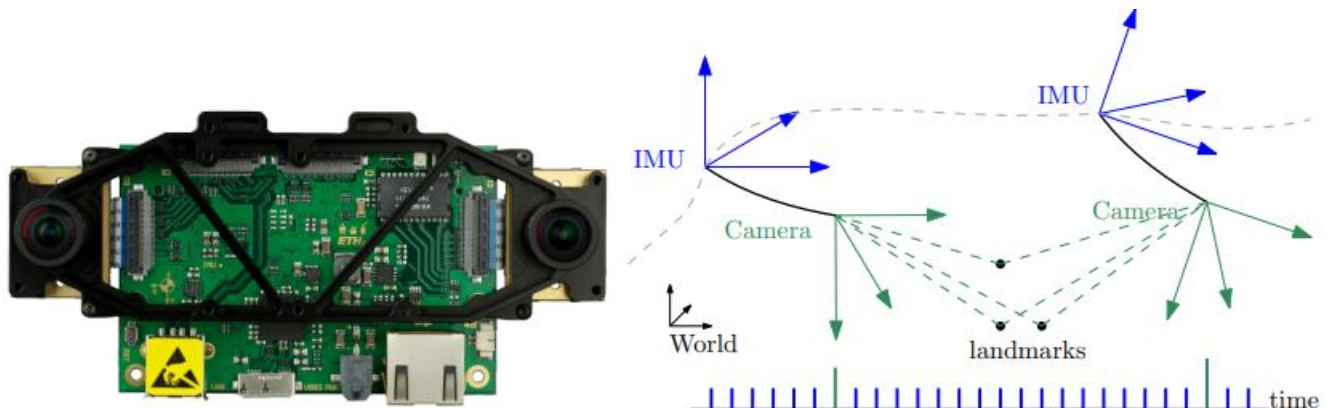


Figure 2.9: Visual Inertial Odometry[72]

## 2.14 Navigation

For any mobile device, the ability to navigate the surroundings is important. Avoiding dangerous situations such as collisions and unsafe conditions comes first, but if the robot has a purpose that relates to certain places in the robot environment, it must find those places.



Robot navigation means the robot's ability to determine its own position in the frame of reference and then plan a trajectory toward some target location. To navigate the environment, the robot or any other mobility device requires a representation, ie. a map of the environment and the ability to interpret that representation [34].

### **2.14.1 Occupancy grid**

Occupancy grids are used to represent a robot workspace as a discrete grid. Information about the environment can be collected from sensors in real time or be loaded from prior knowledge. Laser range finders, bump sensors, cameras, and depth sensors are commonly used to find obstacles in the robot's environment. Occupancy grids are used in robotics algorithms such as path planning. They are used in mapping applications for integrating sensor information in a discrete map, in path planning for finding collision-free paths, and for localizing robots in a known environment. Maps that are different in sizes and resolutions depending on application can be created [79].

### **2.14.2 Costmap**

A costmap is a probability occupancy grid that uses probabilistic values to create a more detailed map representation. This representation is the preferred method for using occupancy grids. Each cell in the occupancy grid has a value representing the probability of the occupancy of that cell. Values close to 1 represent a high certainty that the cell contains an obstacle. Values close to 0 represent certainty that the cell is not occupied and obstacle free. The probabilistic values can give better fidelity of objects and improve performance of certain algorithm applications [35].

### **2.14.3 Pointcloud**

Point clouds are datasets that represent objects or space. These points represent the X, Y, and Z geometric coordinates of a single point on an underlying sampled surface. Point clouds are a means of collating a large number of single spatial measurements into a dataset that can then represent a whole. When colour information is present, the point cloud becomes 4D [23].

## 2.15 Trajectory

Trajectory planning is moving from point A to point B while avoiding collisions over time. Trajectory planning is sometimes referred to as motion planning and erroneously as path planning. Trajectory planning is distinct from path planning in that it is parametrized by time. Essentially trajectory planning encompasses path planning in addition to planning how to move based on velocity, time, and kinematics [100].

### 2.15.1 A\* Global planner

A\* is an informed search algorithm, or a best-first search, meaning that it is formulated in terms of weighted graphs: starting from a specific starting node of a graph, it aims to find a path to the given goal node having the smallest cost (least distance travelled, shortest time, etc.). It does this by maintaining a tree of paths originating at the start node and extending those paths one edge at a time until its termination criterion is satisfied. At each iteration of its main loop, A\* needs to determine which of its paths to extend. It does so based on the cost of the path and an estimate of the cost required to extend the path all the way to the goal. Specifically, A\* selects the path that minimizes:

$$f(n) = g(n) + h(n) \quad (2.2)$$

where  $n$  is the next node on the path,  $g(n)$  is the cost of the path from the start node to  $n$ , and  $h(n)$  is a heuristic function that estimates the cost of the cheapest path from  $n$  to the goal. A\* terminates when the path it chooses to extend is a path from start to goal or if there are no paths eligible to be extended. The heuristic function is problem-specific. If the heuristic function is admissible, meaning that it never overestimates the actual cost to get to the goal, A\* is guaranteed to return a least-cost path from start to goal [95].

### 2.15.2 DWA Local planner

In order to transform the global path into suitable waypoints, the local planner creates new waypoints taking into consideration the dynamic obstacles and the robot constraints. So, to recalculate the path at a specific rate, the map is reduced to the surroundings of the vehicle and is updated as the vehicle is moving around. It is not possible to use the whole map because the sensors are unable to update the map in all regions and a large number of cells would raise the computational cost. Therefore, with the updated local map and the global waypoints, the local planning generates avoidance strategies for dynamic obstacles and tries to match the trajectory as close as possible to the provided waypoints from the global planner. A well established method for local path planning is the Dynamic Window Approach (DWA). An example of a trajectory can be seen in figure 2.10 [9].

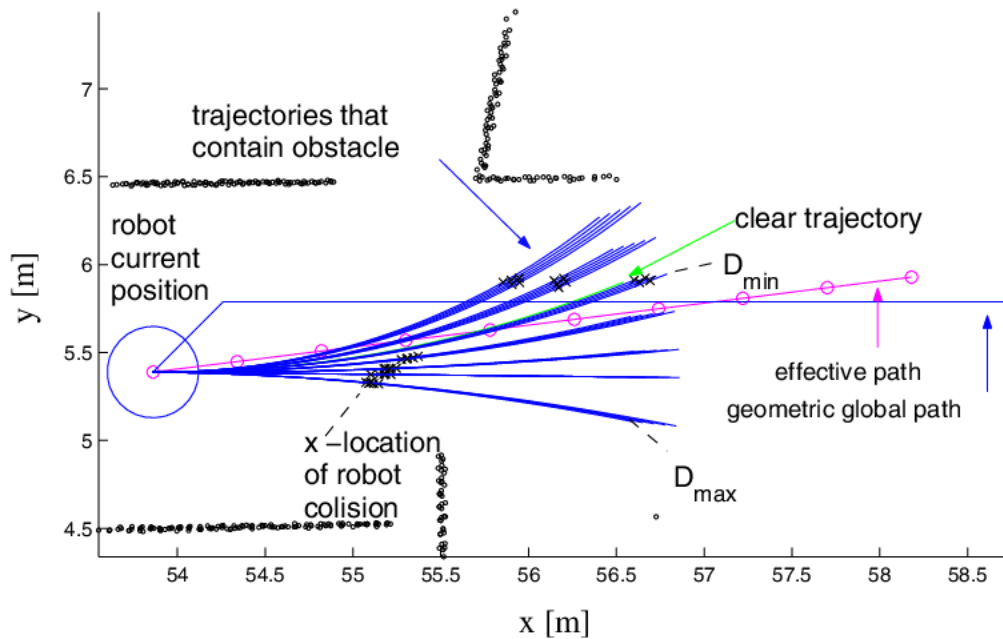


Figure 2.10: Trajectories from the velocities [53]

**The Dynamic Window Approach can be summarized in four steps [16]:**

- Discretely sample the robot's control space  $(v, w)$ .
- For each sample, perform a simulation from the current state for a fixed duration forwards in time to predict what would happen if the control variables in this sample were chosen.
- For each simulation, score the trajectory. The score can be based on multiple factors like speed, proximity to global path and clearance from obstacles. Samples that result in simulations with collisions are removed.
- Pick the  $(v, w)$  sample that results in the highest scoring trajectory and execute.
- Repeat the process.

## 2.16 Machine learning

Machine learning is all about understanding and extracting knowledge from data. It is a researching field within Artificial Intelligence (AI) and has a lot of different fields branching out of it. The use of applications assisted by machine learning algorithm can be found at a daily basis. All from picking out recommended music, videos, movies and other entertainment to unlocking our phones and making our work easier. The use of machine learning has a big influence on how data-driven research is done today. It helps scientists understand problems that is not necessarily recognised by humans, such as finding particles, analysing DNA and recognising cancer [70].

### 2.16.1 Neural networks

Neural networks is inspired by our own human brain. The human brain contains roughly 86 billion neurons and the connections in between these neurons is what makes our mind so powerful. This makes all, from controlling our body, thoughts, memories and much more possible. The idea to use this model of neural network in computing is not a new research field. It was presented for over 60 years ago, but the technology at that time had no possibilities to apply such a model. The CNN is the computers equal to human eyes and is copying how shapes, color

and shades are processed in our brain [42]. There are many kinds of neural networks and some of the easiest to imagine is a feed-forward neural network which contains the following parts:

- The input layer
- The hidden layers
- Neurons
- The output layer
- Activation function

### 2.16.2 Input and hidden layers

The input layer works as a entrance to the hidden layers. The entrance is tailored for the data to fit to the neural network. The data often needs to be processed to fit into a input layer, and when the data is in the same size and dimensions as the input layer, the data is ready to be processed in the neural network. The hidden layers is what makes up the neural network, and is often called a "black box", however this is where the "magic" happens. The hidden is often described by the number of hidden layers. This is the amount of layers with a decided amount of neurons in the depth of the network. In between the layers are connections between the neurons from the previous and to the next layer [42].

### 2.16.3 Neurons

Neurons are connected to the previous layer and receives values from the previous neurons with a weight and then pushed through an activation function. The value of the current neuron can be mathematically described like [42]:

$$\text{ActivationFunction}(N(1 - 1_l) * w_l + N(2 - 1_l)) = N \quad (2.3)$$

### 2.16.4 Output layer

The output layer is similar to the input layer. It is the exit of the neural network and this is where the results come. Therefore, the output layer has to be in the same dimensions and size as the results that are expected. This means that the labelled known training results should have the same size and dimensions as the output layer. So if the model classifying an image to be either a "gauge" or a "valve" the output should be an approximation of what the network thinks it is. E.g. [0.95, 0.02] is showing that this neural network is 95% sure it's a "gauge" [42].

### 2.16.5 Activation function

As mentioned in the section above, the activation function is a function used to calculate the new value of a neuron. This is used to normalize data and keep the neurons under control. A neuron can often make conclusions by looking at noise (i.e. data that are misleading or confusing) and keep evolving around this misunderstanding. There are therefore a lot of different types of activation functions, and the most known are listed in figure 2.11 [42].

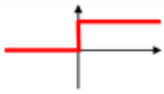
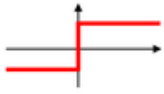
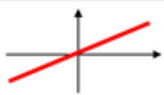
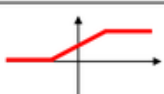


Activation function	Equation	Example	1D Graph
Unit step (Heaviside)	$\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Sign (Signum)	$\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$	Perceptron variant	
Linear	$\phi(z) = z$	Adaline, linear regression	
Piece-wise linear	$\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multi-layer NN	
Hyperbolic tangent	$\phi(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	Multi-layer NN	

Figure 2.11: Different sort of activation functions [58]

## 2.16.6 Deep neural network

Deep learning architectures such as deep neural networks, deep belief networks, recurrent neural networks and convolutions neural networks have been applied to fields including computer vision, speech recognition, natural language processing, audio recognition, machine translation, medical image analysis, material inspection and many more. Where they have produced results comparable to, and in some cases superior to human experts [42].

## 2.16.7 Dataset

### 2.16.7.1 Picking better data

To solve the problem, it is useful to pick the data that suits the problem best. This includes removing data that are not useful or generating/adding more data that are wanted and what supports the conclusion [22].

### 2.16.7.2 Pre-processing and scaling

In neural networks and convolutional neural networks, the algorithms are sensitive to the scale of the data. Therefore, it is a common practice to scale and pre-process the dataset before giving it to the network. The data should be adjusted in such a way that it suits and optimises the performance of the network. In a lot of cases the data is too detailed, the content is "irrelevant" for the problem that you are trying to solve, or just not fit for the network to function optimally. In this case, scale the data to fit the network by e.i. reducing dimensions (PCA), changing the data type, or fitting it into another shape that suits better. A very important thing to remember when you are changing the training data is that the test data should be pre-processed in the same way [32].

### 2.16.8 Multi-class classification

In a lot of cases it is enough to calculate single values like is it a gauge in this image or not. However, in cases where there are more than two classes to classify, you get a problem that is an instance of multi-class classification. When this is the case and each single point should be classified into only one of the categories, meaning that one of the categories is 1 and rest is 0, you have a instance of single-label, multi-class classification [32].

### 2.16.9 One hot encoding

When classifying multiple classes or solving categorical variables (single-label, multi-class classification) such as the one mentioned in the chapter above. The most common way to represent the data is using the one-hot encoding, also known as categorical encoding. The reason behind this is to simplify the variables into true-false, or 0-1 vales to describe what class it is representing [32].

### 2.16.10 Convolutional neural network

A Convolutional Neural Network (CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a CNN is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, CNN have the ability to learn these filters/characteristics [67].



### 2.16.11 Convolution layer

The convolution layer performs a 2D convolution that compares a squared filter over the entire images. The filter could represent an edge, line, dark spot or light spot, that scores the parts of the images to how similar that part of the image is to the filter. After applying the filters the convolution layer gives the result to the next layer [67].

The first convolution layer looks at simple filters, but the filters get more complex in the following layers and could represent entire objects instead of lines or spots in the end. In Figure 2.12 one can see how the different layers have different complexity and can recognize objects like faces and cars in the last layer [67].

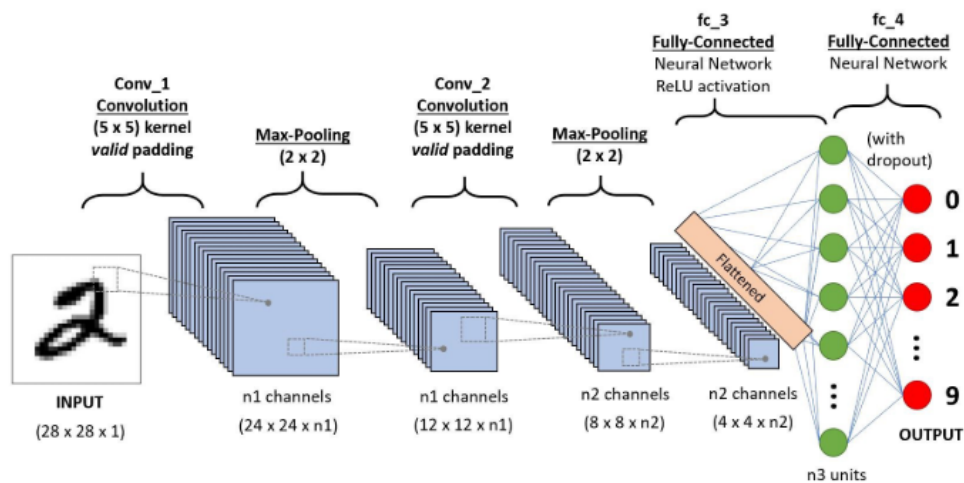


Figure 2.12: Convolutional neural network [67]

### 2.16.12 Pooling layer

The convolution networks often includes local or global pooling layers. This down-samples the output and it uses for example max pooling or average pooling to define the down-sampled value. Including this, does the pooling layer also make the filter less sensitive to position. Setting what actually is appearing in the image in focus [67].

### 2.16.13 Fully connected layer

The final layer in a CNN is named Fully Connected Layer. This is where every value that has gone through all the filters earlier gets a vote to find what the answer is going to be. All the values are listed up in a single fully connected table where it weights against all objects. The one object with the strongest average is what the CNN will return as its result [67].

### 2.16.14 Transfer learning

Transfer learning is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. For example, knowledge gained while learning to recognize cars could apply when trying to recognize trucks. This area of research bears some relation to the long history of psychological literature on transfer of learning, although formal ties between the two fields are limited [69].

### 2.16.15 Training

Training a neural network typically consists of two phases:

- A forward phase, where the input is passed completely through the network. During the forward phase, each layer will cache any data (like inputs, intermediate values, etc), as it is later used in the backward phase. This means that any backward phase must be preceded by a corresponding forward phase [50].
- Backward phase, is where gradients are back-propagated (backprop) and weights are updated. During the backward phase, each layer will receive a gradient and also return a gradient. It will receive the gradient of loss with respect to its outputs  $\frac{\partial L}{\partial \text{out}}$  and return the gradient of loss with respect to its inputs  $\frac{\partial L}{\partial \text{in}}$  [50].

### 2.16.16 Evaluation

Model evaluation metrics are required to quantify model performance. The choice of evaluation metrics depends on a given machine learning task (e.g. classification, regression, ranking, clustering, topic modeling, among others). Some metrics, such as precision-recall, are useful for multiple tasks [41].

#### 2.16.16.1 Confusion matrix

Confusion matrix, as seen in figure 2.13, is a table that is often used to describe the performance of a classification model (or “*classifier*”) on a set of test data for which the true values are known. It allows the visualization of the performance of an algorithm [73].

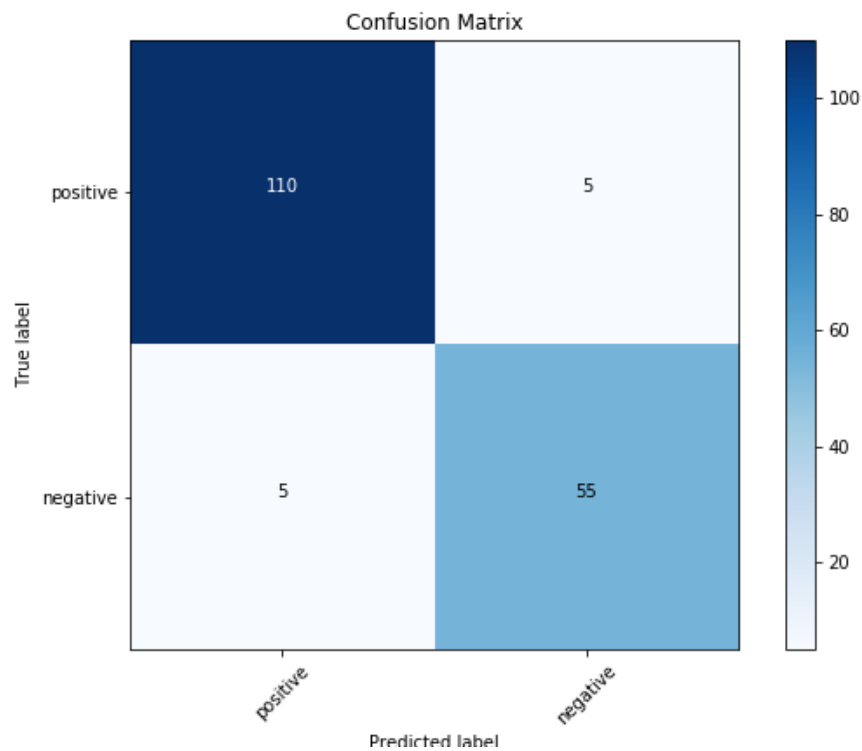


Figure 2.13: Confusion matrix

- **Positive (P):** Observation is positive (for example: is patient positive).
- **Negative (N):** Observation is not positive (for example: is not patient positive).
- **True Positive (TP):** Observation is positive, and is predicted to be positive.

- **False Negative (FN):** Observation is positive, but is predicted negative.
- **True Negative (TN):** Observation is negative, and is predicted to be negative.
- **False Positive (FP):** Observation is negative, but is predicted positive.

### Precision

$$Precision = \frac{TP}{TP + FP} \quad (2.4)$$

### Sensitivity

$$Sensitivity = \frac{TP}{TP + FN} \quad (2.5)$$

### Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.6)$$

### F1 Score

$$F1_{Score} = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (2.7)$$

#### 2.16.17 Intersection over union

IoU is a metric used to calculate the similarity between two arbitrary shapes. The IoU score is calculated using the following formula:

$$IoU = \frac{|A \cap B|}{|A \cup B|} \quad (2.8)$$

While IoU provides a good metric for overlap, it does not provide any information about the orientation and form of overlap. Multiple different overlaps can result in the same IoU score. If there is no overlap between  $A$  &  $B$ , no information about the distance between the two shapes can be provided [18].

## 2.17 Programming Language

A programming language is a formal language, which comprises a set of instructions that produce various kinds of output. Programming languages are used in computer programming to implement algorithms and structures.

### 2.17.1 Python

Python is an interpreted, high-level, general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library [89].

### 2.17.2 C++

C++ is a general-purpose programming language, and modern C++ has object-oriented, generic, and functional features in addition to facilities for low-level memory manipulation [88].

C++ was designed with a bias toward system programming and embedded, resource-constrained software and large systems, with performance, efficiency and flexibility of use as its design highlights. C++ has also been found useful in many other contexts, with key strengths being software infrastructure and resource-constrained applications, and performance-critical applications [88].

### 2.17.3 SQL

SQL is a domain-specific language used in programming and designed for managing data held in a relational database management system, or for stream processing in a relational data stream management system [94].

# Chapter 3

## Materials

### 3.1 Motor driver

The motor driver is a ODrive v3.6 56V from ODriverobotics. The driver board without connectors can be seen in figure 3.1 [61].

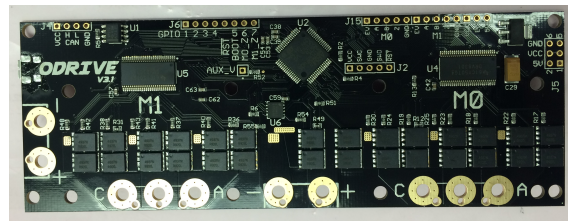


Figure 3.1: Odrive V3.6 [83]

### 3.2 Motors

The motors are the T-motor MN5212 KV340 model and can be seen in figure 3.2. It offers nearly 1 Nm of torque with nominal load speed up to 6300 rpm when driven with 24V. It weighs in at 249 gram [75].



Figure 3.2: MN5212 KV340 [75]

### 3.3 Incremental rotary encoders

The encoders are magnetic incremental rotary encoders. These encoders have a resolution of 2000 pulses per revolution and can be seen in figure 3.3 [3].

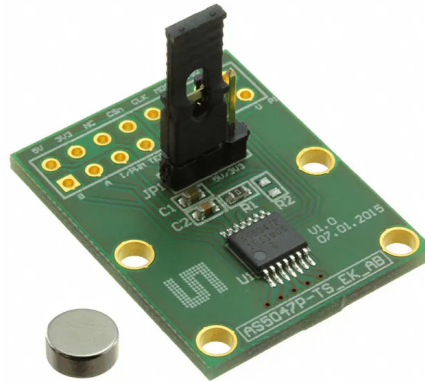


Figure 3.3: Encoder [3]

### 3.4 Jetson Nano

The Jetson Nano runs on Linux and have integrated support for USB, Ethernet and GPIO pins for connecting hardware. Figure 3.4 shows a Jetson Nano [45].



Figure 3.4: Jetson Nano [45]

### 3.5 Teensy 4.0

The microcontroller (MCU) is a Teensy 4.0. The MCU features an ARM Cortex-M7 processor at 600 MHz, and have 23 I/O pins . The MCU can be seen in figure 3.5 [54].

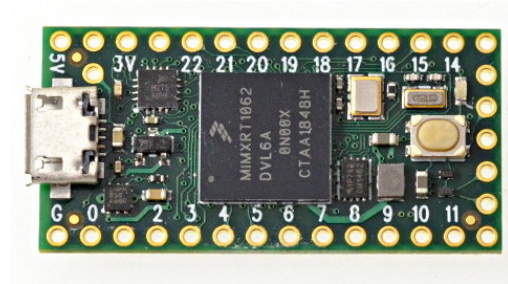


Figure 3.5: Teensy 4.0 [54]

### 3.6 Xbox One Controller

The remote controller is an Xbox one controller. An Xbox One Controller can be seen in figure 3.6.



Figure 3.6: Xbox One Controller [40]



### 3.7 Remote computer

The main components in the remote computer are:

- **CPU Intel i7-7700K** - 4.5 GHz clock speed. 4 cores and 8 Threads
- **GPU GeForce GTX 1080 Ti** - 11 GB memory and 3584 NVIDIA CUDA Cores
- **RAM Corsair Vengeance** - DDR4 3200Mhz 24GB

### 3.8 Camera

The cameras that are used is the Intel Realsense D435, see figure 3.7, and the Intel Realsense T265, see figure 3.8. The Intel D435 is a depth camera that can be used indoors and outdoors. The maximum range is approximately 10 meters[25]. The Intel Realsense T265 is a tracking camera that has precision tracking and is small and light weight[26].



Figure 3.7: Intel D435 [25]



Figure 3.8: Intel T265[26]

### 3.9 Construction and Electrical components

The construction consists of aluminium profiles, and self produced parts that was 3D printed in plastic or cut in Plexiglas. The components used in the electrical part of the project were mainly bought online. In appendix B.1, a full list of all the materials and its quantity are listed. How the components are connected, and how they are used can be seen in Appendix D.2. A deeper explanation of the overall structure of the robot can be read in chapter 4.7

### 3.9.1 Batteries

The chosen power source were three 1000mAh 6S LiPo batteries seen in figure 3.9. A voltage converter is used for connecting the Jetson to power.



Figure 3.9: Battery [13]

## 3.10 Software and libraries

The following software and libraries has been used throughout this project.

### 3.10.1 Software

- **Overleaf** - Free web-based tool for writing in LaTeX. Some of its features are spell checking and cloud-based storing. The cloud-based storage makes real-time editing possible and thereof its possible to cooperate in the same files[49].
- **Platform IO** - PlatformIO is a cross-platform, cross-architecture, multiple framework, professional tool for embedded systems engineers and for software developers who write applications for embedded products[55].
- **PC Schematic Automation** - Electrical CAD software. Has been used to draw all electrical schematics and the electrical layout diagrams[52].
- **Fusion 360** - 3D designing software[4].
- **Cura** - Slice STL files and generates G-code for 3D printing[80].
- **Draw.io** - For making simple diagrams and illustrations[12].

- **Visual Studio Code** - Visual Studio Code is a source-code editor developed by Microsoft for Windows, Linux and macOS. It includes support for debugging, embedded Git control and GitHub, syntax highlighting, intelligent code completion, snippets, and code refactoring[39].
- **Odrive Tool** - The ODrive Tool is the accompanying PC program for the ODrive. It's main purpose is to provide an interactive shell to control the device manually[46].

### 3.10.2 Libraries

The following libraries has been used in combination with Visual Studio Code:

#### 3.10.2.1 Python libraries

- **PyQT** is a Python binding of the cross-platform GUI toolkit Qt, implemented as a Python plug-in. [7].
- **OpenCV** is a library of programming functions mainly aimed at real-time computer vision [47].
- **Numpy** is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays [44].
- **Pytesseract** - is an optical character recognition (OCR) tool for python. It recognizes and “reads” the text embedded in images[24].
- **TensorFlow** - is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks [78].
- **Flask** - is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries [56].
- **Matplotlib** - is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots

into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+ [36].

- **Keras** - is an open-source neural-network library written in Python. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible [31].
- **ROS Python** - is a client API that enables Python programmers to quickly interface with ROS Topics, Services, and Parameters. The design of rospy favors implementation speed (i.e. development time) over runtime performance so that algorithms can be quickly prototyped and tested within ROS. It is also ideal for non-critical-path code, such as configuration and initialization code [8].

### 3.10.2.2 C++ libraries

- **Rviz** - is a 3D visualizer for the Robot Operating System (ROS) framework [10].
- **RTAB-Map** (Real-Time Appearance-Based Mapping) is a RGB-D, Stereo and Lidar Graph-Based SLAM approach based on an incremental appearance-based loop closure detector [33].
- **ODriveArduino** is a library used to operate the ODrive motor driver [84].
- **ROS Arduino** is a Arduino-specific extensions required to run roserial-client on an Arduino. It is meant to demonstrate how easy it is to integrate custom hardware and cheap sensors into your ROS project using an Arduino [38].

# Chapter 4

## Method

### 4.1 Project Approach

Once a week the project leader held a comprehensive check of how far the progress had come within the individual tasks. Even though there was a leader, the structure was a flat organization where decisions and agreements were taken together. Each team member was responsible for one, or more main tasks and reported to the leader. This was to relieve the leader so it was easier to keep an overview.

Every other week it was held a meeting with the supervisors. During these meetings, the progress was discussed, solutions to possible challenges and suggestions for possible improvements. In the early stages of the project, it was established a plan based on the tasks. To ensure a good result and efficient approach a detailed project plan was made. This plan contained detailed information about when subtasks should start and end. A short version of the plan is listed below.

- Make a detailed concept drawing of the total system. It contains kinematic, measurements and a basic design. This will give all the team members an overview of task needed to be done and a basic idea of what the result should be in the end.
- Early start on designing the prototype. The robot will be designed using 3D CAD. All parts will be ordered early in the project to ensure they are delivered before the building phase.

Consequently, parts that needs to be manufactured can be done while waiting for the parts ordered.

- Programming will start at once. First the program structure will be done, and then proceed to programming the communication part and at last the robot motion. Since a lot of the parts will be 3D printed which takes time, the production of these parts will be done along the programming.
- Assembling the robot will be started simultaneous with the programming since the division of labor is done to make this efficient.
- Testing will be performed when the robot is assembled and the necessary programs are completed. All objectives will be tested separately before tested combined.

The full version of the project plan can be found in appendix [A.1](#).

## 4.2 Approach due to the Covid-19

Because of the outbreak of the Covid-19 in Norway late February and further the school closing in middle of March some changes to the approach were done. First a risk analysis due to a pandemic were made, and this can be seen in appendix [A.2](#).

Secondly several things were done to minimize the spread of the disease. The first two weeks after the school closed everyone were stationed at their own homes. The tasks that needed to be done were split between members for minimal need of physical contact. An example is that one of the member took the robot home and it was his job to work on leg movement. After these two weeks the group members met and worked together because all members needed to have access to the robot. The whole time the school were closed all meetings with the supervisors were conducted online.

Overall the thesis was not to affected of the pandemic, but experienced smaller problems and delays. For instance the two week period not working together were a setback. Ordering and

printing parts were much slower. Not being able to use the schools equipment was sometimes tedious.

## 4.3 Leg movement

### 4.3.1 Leg trajectory

The leg trajectory is made up of two half elliptical curves which are made by using trigonometric functions as can be seen in figure 4.1. The amplitude, length and frequency of the curves can be changed. The amplitude decides how high the curve will be, and the length decides the step length of the robot. The frequency decides the speed of the motion. The height of the robot can also be changed to make the robot for example crawl. Changing these parameters plus the sample number in positive or negative direction, makes it possible to get the robot to do various speeds such as walking or running.

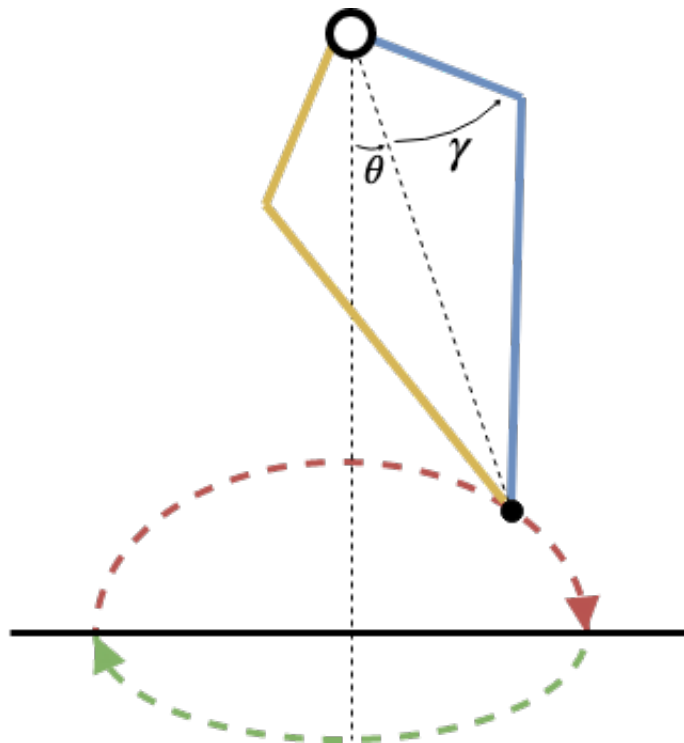


Figure 4.1: Leg Trajectories

### 4.3.2 Simulation

The simulation can calculate the position of the legs with  $x$  and  $y$  inputs, and then show the legs and print the corresponding angle for achieve this. An example of this can be seen in figure 4.2. The inputs here are  $x = 30$ ,  $y = -200$  and the angle for the right foot is  $58.5^\circ$  and the left foot is  $-41.5^\circ$ , when  $0^\circ$  is straight down.

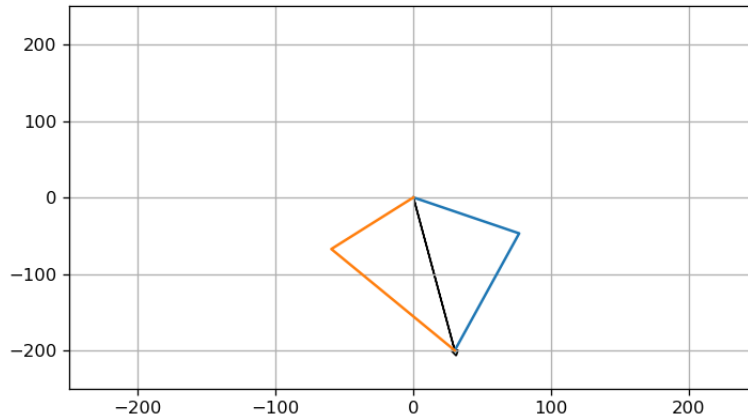


Figure 4.2: One point simulation



The simulation can also simulate the whole leg movement with given parameters.

The parameters are:

- Step length
- Height of the robot
- Over and under amplitude
- Frequency

An example of this can be seen in figure 4.3 or a demonstration here [Simulation](#)

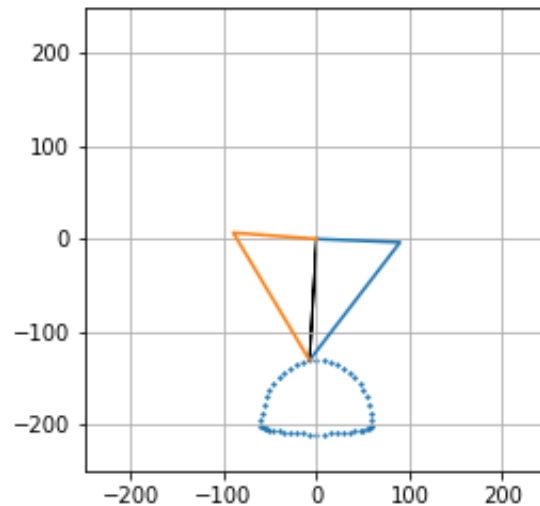


Figure 4.3: Simulation

## 4.4 Convolutional neural networks

There are several meta-architectures for object detection. Some of the most popular approaches are, SSD, R-CNN, Faster R-CNN, and R-FCN. While these networks are built with different approaches, they all need feature extractors.

### 4.4.1 R-CNN

Region Convolutional Neural Network (R-CNN) combines conventional CNN with bottom-up region proposals to localize objects. The model consists of three modules; a CNN for feature extraction, category independent region proposals, and a class specific Support-vector Machine (SVM) [2].

**Region Proposal.** Instead of generating a large amount of regions, R-CNN limits itself to approximately 2000 regions. Each region is selected using an algorithm. R-CNN is agnostic to the region proposal method. There are many different algorithms, such as objectness, constrained parametric min-cuts, and selective search. To more accurately enable comparison between difference detection models, selective search is used [28].

**Feature Extraction.** From each region in the region proposal, a 4096-dimensional feature vector is extracted using a CNN. Each region is transformed into a shape that is compatible with the CNN, where inputs are required to have a fixed pixel size. All pixels are warped in a tight bounding box around the object, regardless of aspect ratio or size. These regions are propagated through the CNN [21].

**Support vector machine.** A SVM is an algorithm which finds a separating line between data of two classes. The separating line is usually optimized with regards to margin, such that the distance from the line to any data point is as large as possible. In the final layer of the CNN, a SVM is implemented to determine whether or not the bounding box contains an object, and if it does, it tries classify what kind of object it is [19].

**Matching Strategy** To improve localization performance an simple bounding-box regression stage is implemented. All the bounding boxes below a certain IoU threshold is discarded, e.g  $IoU < 0.4$ . The bounding box with highest IoU is kept while the rest is discarded using non-maximum suppression [21].

### **4.4.2 Fast R-CNN**

Fast R-CNN is an updated version of R-CNN. The main differences is that regions of interest are pooled, and a single feature map is extracted from them. In fast R-CNN not all proposals need to be classified, instead convolution operations are done once per image. Training time is reduced from 84 hours to 9.5 hours [20].

### **4.4.3 Faster R-CNN**

Faster R-CNN further expands on the R-CNN approach. It consists of two modules; a deep fully convolutional network for region proposal, and the detector from the Fast R-CNN model. The Region Proposal Network (RPN) takes an image as input and gives a set of rectangular object proposals with corresponding objectness score as output. These outputs then gets fed through the detector network which determines class and score.

Figure 4.4 display the model architecture. Using an alternating training method, the model can be trained in four steps [57].

- Train the RPN
- Train the Faster R-CNN detector network using the region proposals from the RPN
- Use the trained detector to initialize a new RPN training session, where the shared convolutional layers becomes fixed while tuning only the layers unique to the RPN.
- Keeping the convolutional layers fixed, fine tune the unique layers of the detector network.

By sharing the convolutional layers between the RPN and the detector network the computational cost is drastically reduced, resulting in reduced processing time [57].

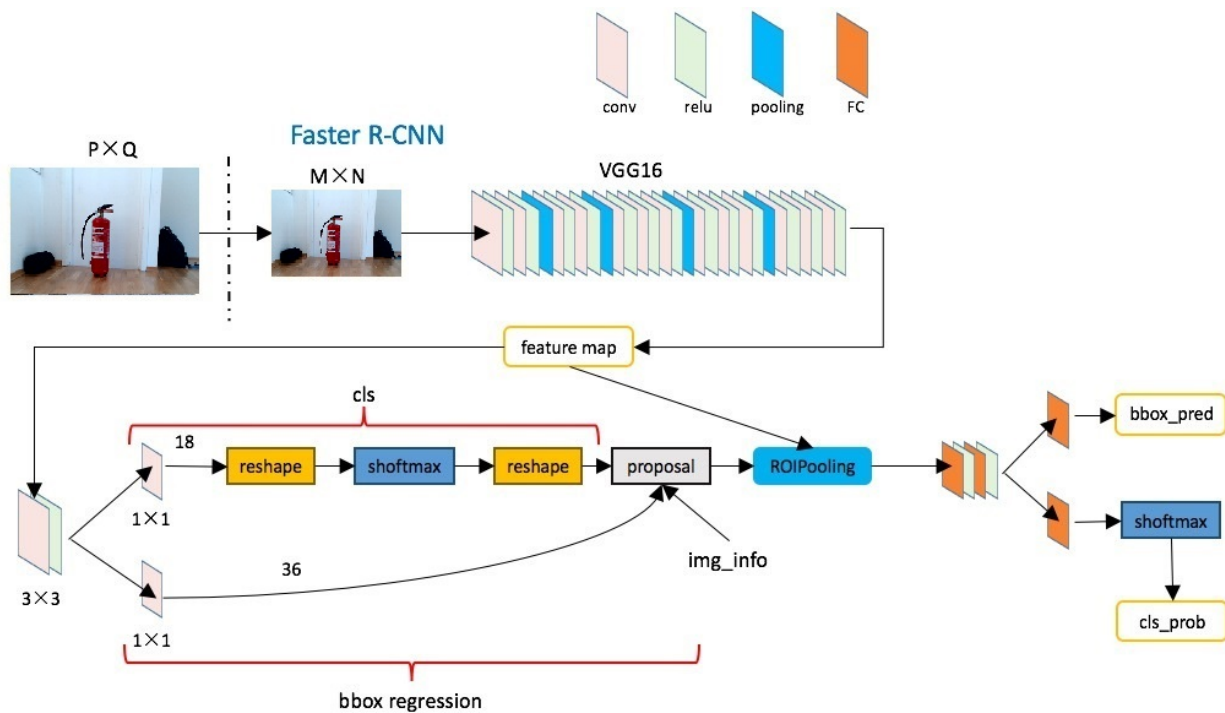


Figure 4.4: Faster RCNN Architecture

## 4.5 Review

Throughout the implementation of the project it was done reviews of hardware, software and design. The reviews were performed in order to make sure that the work performed and choices made were the best option for creating a machine that operates as desired.

- Software review will include checking that the code follows the rule of thumb with high cohesion, low coupling and contain thread safe operations. Tasks of each subsystem should be well defined.
- Hardware review was needed for controlling that the total system of hardware works well together, and with a functionality that was as fail safe as possible.
- Design review will be performed for assuring that components are well placed, and that the various solutions are as simple and effective as possible. A focus with the design was to make the prototype with as few parts as possible, leading to few sources of error and easy to service.

## 4.6 Testing

Several tests had to be performed during the project. Some test approaches are explained beneath.

### 4.6.1 Leg movement testing

There were several steps in testing the movement of the legs. First the movement of the legs were simulated in the simulation that were made, see section [4.3.2](#) for more information regarding this. After the simulated movement looked good, the movement was tested on one foot. The testing of one foot and then later all four were done with the robot on a stand so the motors could move more freely. The program did also have a lot of constrains so that the wrong input data or calculation would not make the foot go to places it could be damaged. After passing the mentioned checks, the robot would be put on the ground for the last phase of the testing.

### **4.6.2 Robot testing**

For the robot to function as specified it had to perform a set of tasks correctly. The tasks that were to be realized before it could be considered operational was the following:

- Stand still on all four legs
- Stabilize itself
- Walk forward and backwards
- Turn
- Adjust its movement with input from an Xbox controller or trajectory planner
- Stop and inspect on waypoints

### **4.6.3 Electrical testing**

The electrical installation had to be controlled before it was powered. This is necessary to avoid damaging components by applying wrong voltage levels or short circuits. Controlling the wiring was done by using a multi-meter for measuring the resistance from point to point in the circuits. Before applying power all sensitive electronics had been disconnected. Thereafter the voltage on the voltage supplied was measured, and if it was correct the electronics could be connected again.

### **4.6.4 Communication testing**

The communication used will at first be tested in small scale with only the controller nodes, to verify that the method works with the chosen controllers for the project. To ensure that the method used is robust and optimal for this application it needs to be tested in the environment that it will operate in during normal operation. This can be a harsh environment for communication as it includes power supplies and motor drivers that generates electromagnetic noise.

### **4.6.5 Image processing testing**

Several parameters were needed to be tuned in the implemented algorithms to manage to extract the most accurate amount of data from the processed image. The reason for this is the amount of data that is possible to extract is limited to the light, and thereby the algorithms have to be robust since the robot will traverse in a environment which contains dynamic light sources.

### **4.6.6 Convolution neural network testing**

The CNN had to deliver an extremely high accuracy while running fast. An evaluation had to be made to quantify the performance.

The evaluation metrics that the testing is based upon are the following:

- Confusion matrix
- Precision
- Sensitivity
- Accuracy
- Mean average Precision (mAP)
- Mean average Recall (mAR)
- Inference time

## 4.7 Physical design

This chapter explains how the most essential parts of the robot is designed and manufactured. All constructed parts are designed using Autodesk Fusion360 and Autodesk Inventor. Custom parts were either manufactured in the Tunglab at NTNU campus Aalesund, or made by suppliers in the nearby mechanical industry, such as Prodex and Stranda Stålindustri. The Tunglab has 3D-printers and a CNC laser for use on wood and plastic. Figure 4.5 shows a 3D made picture of the robot.



Figure 4.5: Front Right View

### 4.7.1 Robot

Since the main object of this thesis is not on the product design, it was chosen to look online for inspiration. The reason for this is to save time and therefore use more time on the software. The final result drew a lot of inspiration from an open source project done at Stanford university by students. All about their project can be read here [Github](#) [29] or here [Report](#) [30].



From their project several smaller and bigger parts were altered or added. The original design did not have the cameras nor the Jetson Nano so the design was changed to accommodate this. There were also many smaller changes made to fit our needs. The complete 3D-model of the robot can be found here [3D-model](#). More pictures of the design and robot can be found here [Pictures](#)

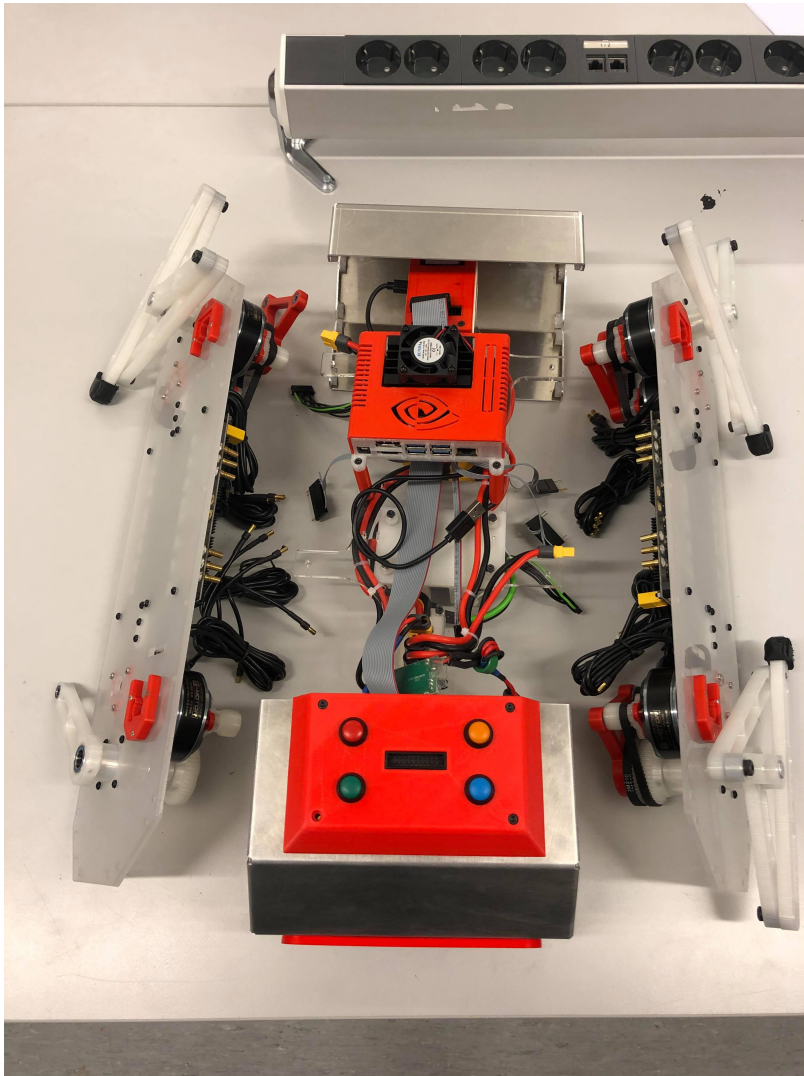


Figure 4.6: All parts of the body

### 4.7.1.1 Main frame

The main frame is made out of plexiglas and cut with the schools laser cutter. On the main frame the Jetson Nano and the MCU are located on the top side. On the bottom side the batteries are attached. On the main frame there are also several connectors for easily attaching or detaching the batteries, power switch etc. The main frame is shown in figure 4.7.

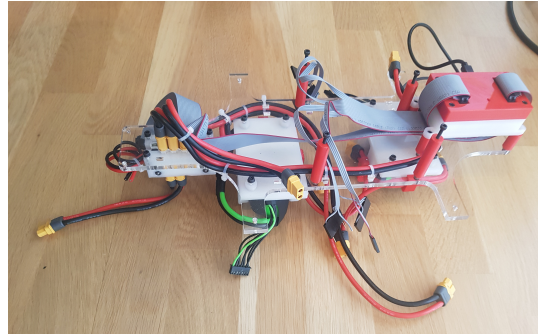


Figure 4.7: Main frame of the robot

### 4.7.1.2 Front and backside

Both the front and backside are press braked plates made out of laser cut aluminum. On the front there is a rectangular cut out to accommodate mounting of the camera assembly as can be seen in figure 4.8. The camera holder is a special 3D-printed holder for easily removal of the cameras.



Figure 4.8: The front of the robot

### 4.7.1.3 Sides

Each side contains the parts needed for moving the legs. Each side has two ODrives motor drivers, where each driver is controlling two motors. The encoders are also connected on the sides, and the parts transferring the motor movement to leg movement are also presented here. A picture of the sides can be seen in figure 4.9.

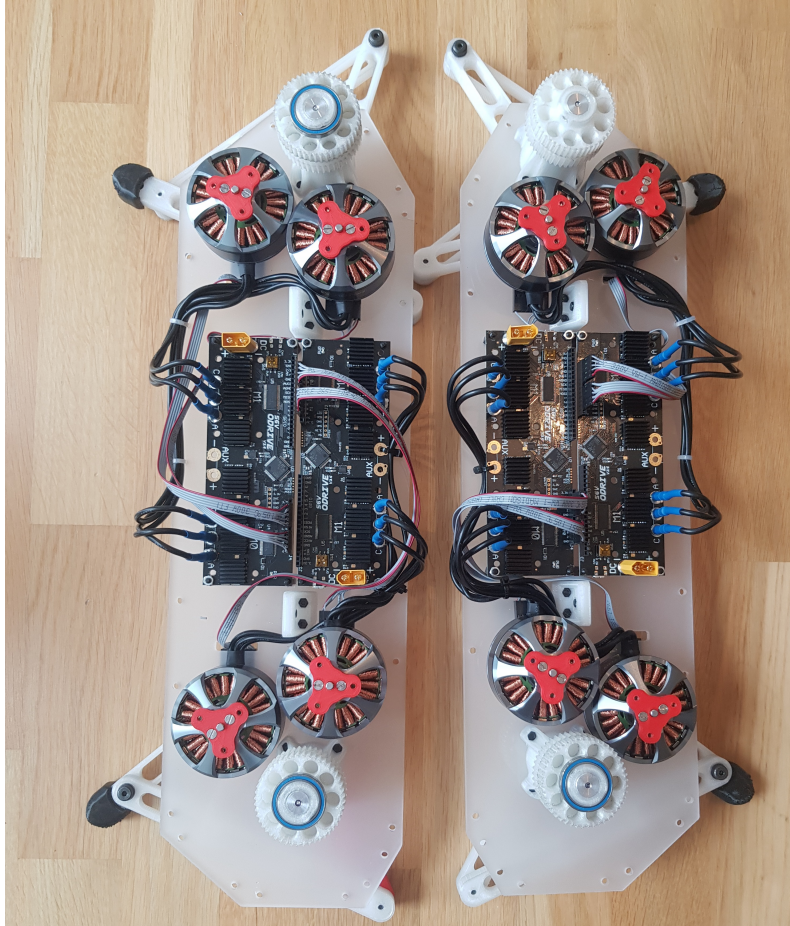


Figure 4.9: Sides of the robot

#### 4.7.1.4 Motors and pulleys

Each actuator is built by two motors and a coaxial hub and can be seen in figure 4.10. The coaxial hub allows for two concentrically mounted shafts to turn independently of each other while transferring rotational movement to the output end of the shafts. Each shaft have a pulley fixed on one side that the motors transfer rotational movement to via a timing belt. The legs are then mounted on the output side of the coaxial hub. The gear ratio between input and output is 3:1, giving one revolution on the output shaft for every three revolutions the motor provide. Figure 4.11 shows the finished construction.

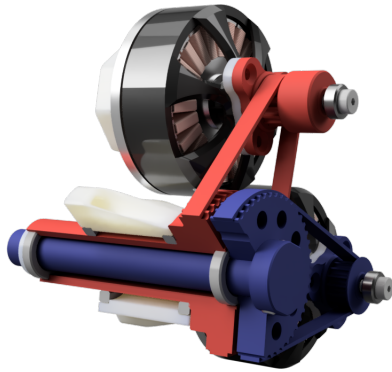


Figure 4.10: Actuator section view

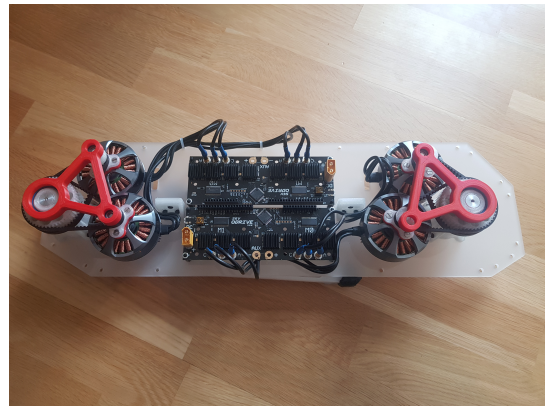


Figure 4.11: Motors and pulleys



### 4.7.1.5 Legs

Each leg is made from four links where two and two have the same length. The ends of the links are connected with each other using ball bearings which means they can rotate freely from each other. The upper set of links are connected to the coaxial shafts. So by rotating the motors a change in foot position will occur. Two different types of legs used can be seen in figure 4.12 and 4.13.



Figure 4.12: Nylon Legs

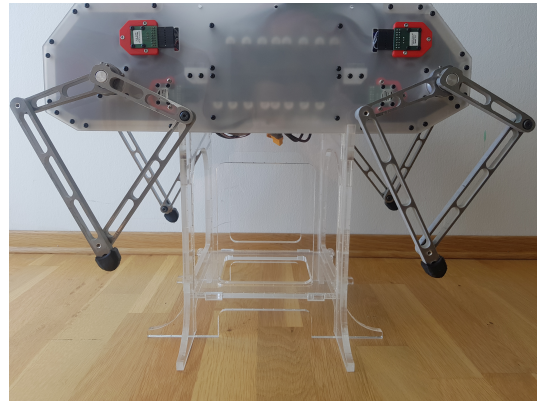


Figure 4.13: Aluminium legs

### 4.7.2 Stand

A stand was designed to make it easier for testing. The stand makes it possible for the robot to move its legs without interfering with the ground. This was crucial for testing the movement of the robot, and helped reducing the time spent on software development. The stand can be seen in figure 4.14 or as finished product in figure 4.15

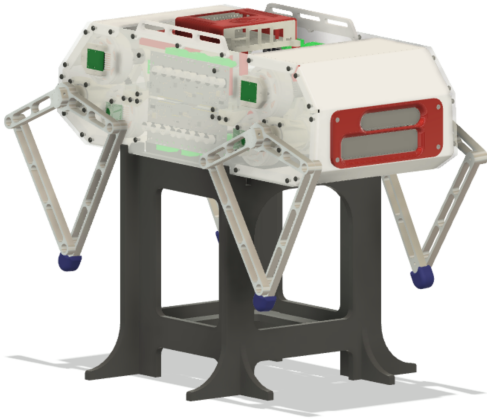


Figure 4.14: Robot stand



Figure 4.15: Robot stand

### 4.7.3 Electrical layout

Much of the electrical components is built on the skeleton plate that holds the robot chassis together. The assembly includes Jetson Nano computer, MCU, batteries and power a distribution unit, see figure 4.16.

The Jetson Nano and the MCU is mounted in their own 3D printed enclosure that protects them from being damaged and retains them in their proper placements. The enclosure for the MCU is custom made. While for the Jetson Nano a design made by ecoiras , published on thingiverse, was used. However minor changes was needed so it could be mounted on the plate. The original case for Jetson Nano can be found here [Jetson Nano case \[14\]](#) and is used under the [Attribution 4.0 International \[6\]](#) license.

The batteries are mounted on the bottom of the plate in custom made brackets that has a retainer clip that does not require any tools for opening the bracket, making the process of performing battery changes a lot easier in the tight space.



Figure 4.16: Electrical Assembly Plate

The two cameras are mounted together, one on top of the other, placing the center of the stereo camera directly under the center of tracking location on the tracking camera. This ensures that the coordinate systems of the two camera modules are identical except a slight offset in the z-axis. This makes data from the cameras easier to work with in software.

The bracket in which the cameras are mounted can be snapped into a holding bracket that is mounted on the front of the robot. The front and back of the bracket can be seen in figure 4.17 and 4.18, respectively.

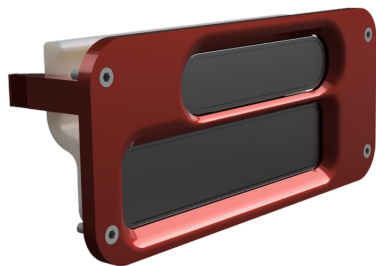


Figure 4.17: Camera Assembly Front

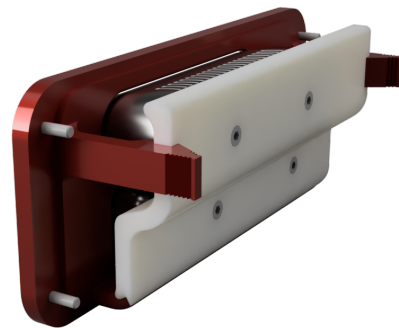


Figure 4.18: Camera Assembly Rear



On top of the front of the robot there is mounted a button control panel, as seen in figure 4.19. This control panel have four push buttons with built in LED's for control of the robot's basic actions. The LED's are connected to a PCB soldered in-house that further connects to the MCU through a flat ribbon cable with connectors on both sides. This modular designs allows for easy disconnection and connection if necessary. For example when disassembling the robot.

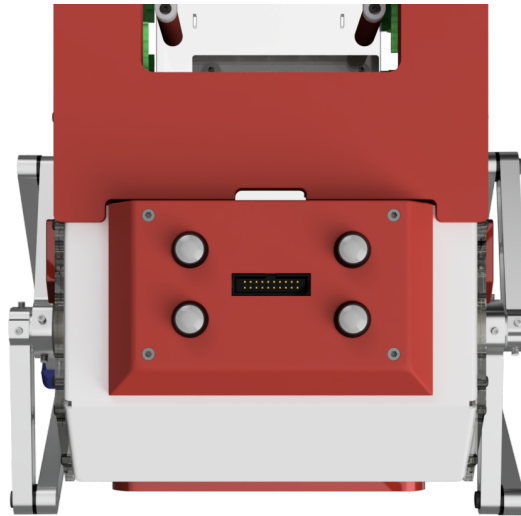


Figure 4.19: On Board Control Panel

To see how the electrical components are connected, check appendix [D.2](#).

## 4.8 Implementation

This chapter explains how the various systems of the robot works, and how these are implemented in the finished system.

### 4.8.1 System overview

Figure 4.20 introduces the system in its entirety. It consists of several systems:

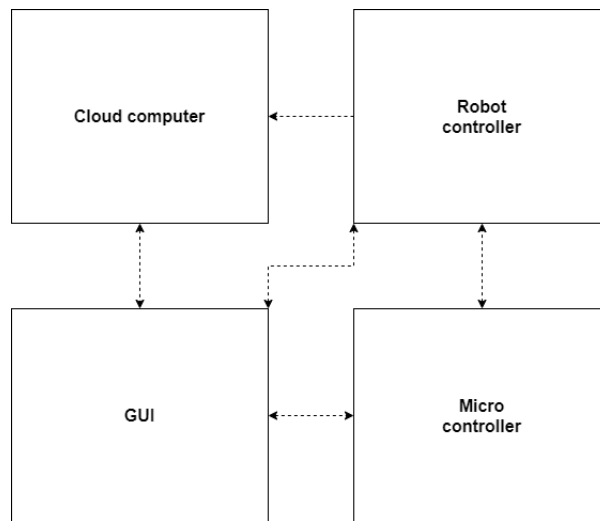


Figure 4.20: System Overview

### 4.8.2 Graphical user interface

A graphical user interface (GUI) has been developed. The interface provides the operator with option for controlling the robot, or just observe it in its autonomous mission. The GUI was created with use of PyQt in the Python programming language, and how it is implemented can be seen in figure 4.21.

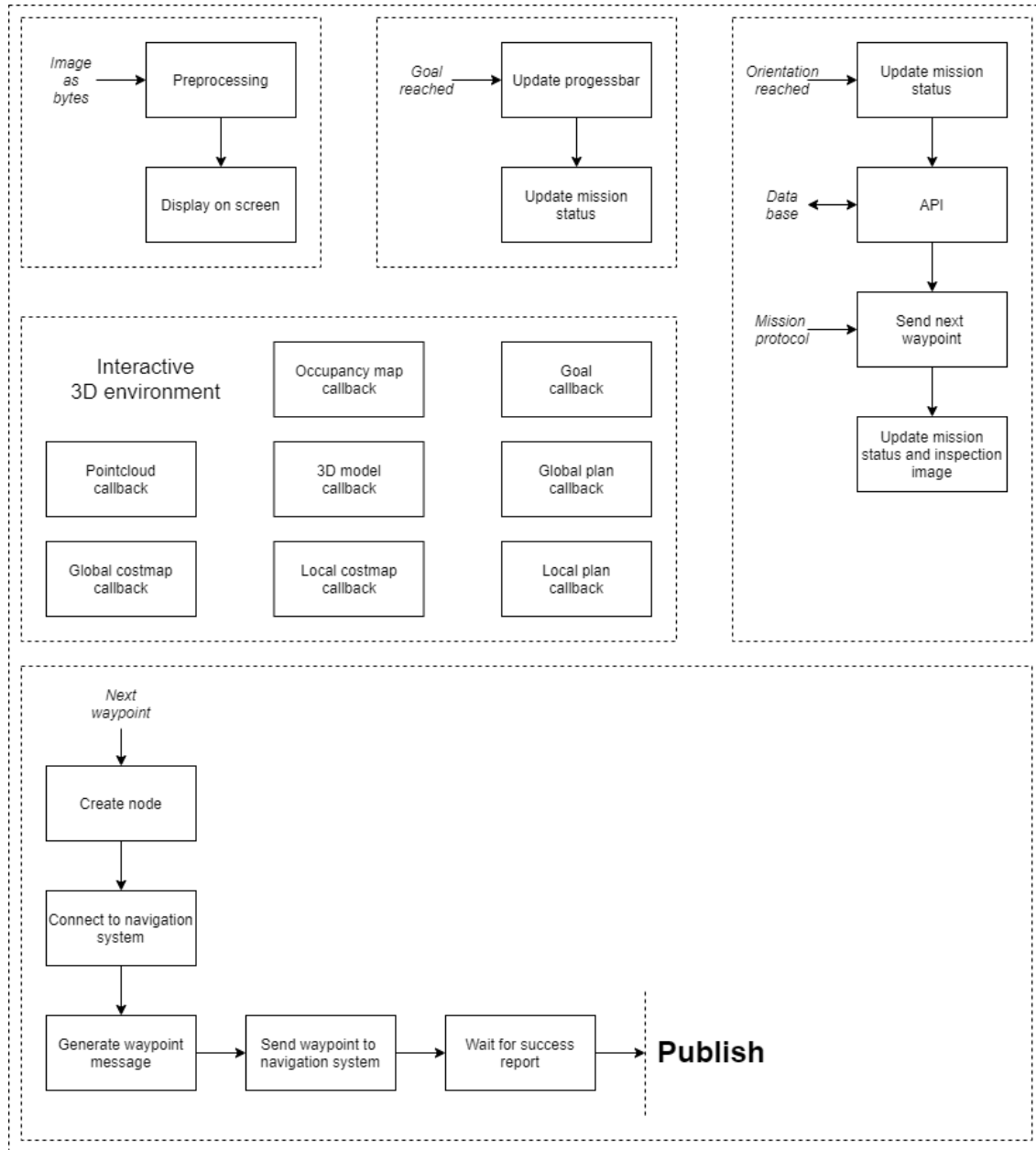


Figure 4.21: Graphical User Interface

It is implemented for three different scenarios in mind. The first one is simply made for manual control, and displays the Xbox controller with linked button to motion mapping for ease of use. The second scenario is showing one of the cameras in full screen to give the operator a first person view. This mode can be viewed in both manual and autonomous operation. The last scenario is full autonomous operation running a inspection mission. This mode contains multiple layers of logic that works concurrently. The reason for this is that data is received from different sources at different time intervals, and the operator has to be able to navigate and use the interface at any given time without input lag or freezes.

The concurrent data-handling is happening in the ROS subscriber callback functions. These callback functions is event based and runs each time new data is received. The status and mission protocol can utilize this and update in real time without interrupting the operator handling the 3D interactive environment. The 3D interactive environment is build into the GUI with use of some of the RVIZ widgets. These widgets are made for the ROS ecosystem, and supports all of the standardized messages which requires to be linked to correct topics that sends the correct message forms.

The GUI utilize the REST-API to request data from the database. As the database stores CNN output and equipment attributes, can the GUI fetch this information and display it to the operator.

#### **4.8.2.1 Create new mission**

An interface has been developed to generate new mission protocols. This is made as an cooperation with the robot itself and the operator controlling it. As shown in figure 4.22 is the start point the initial task. This reads the robots current position and orientation in space and stores it in the protocol. The operator can now move the robot to a new location and add the robot position as a waypoint. This will prompt the operator with a choice to make it a waypoint with no action, or add a inspection action linked to the waypoint. If the last is chosen the operator will have the control to adjust the robot head to the desired position that points towards an equipment. After that, the operator will add a ground truth to which equipment is to be detected and

add the corresponding attributes to the equipment. This loop can be done multiple times until a full plan has been created. This leads to the last step which is to save mission and send it over to the "Generate Mission Service" who's job is to repeatedly send API requests until the whole mission is saved in the database for later reference.

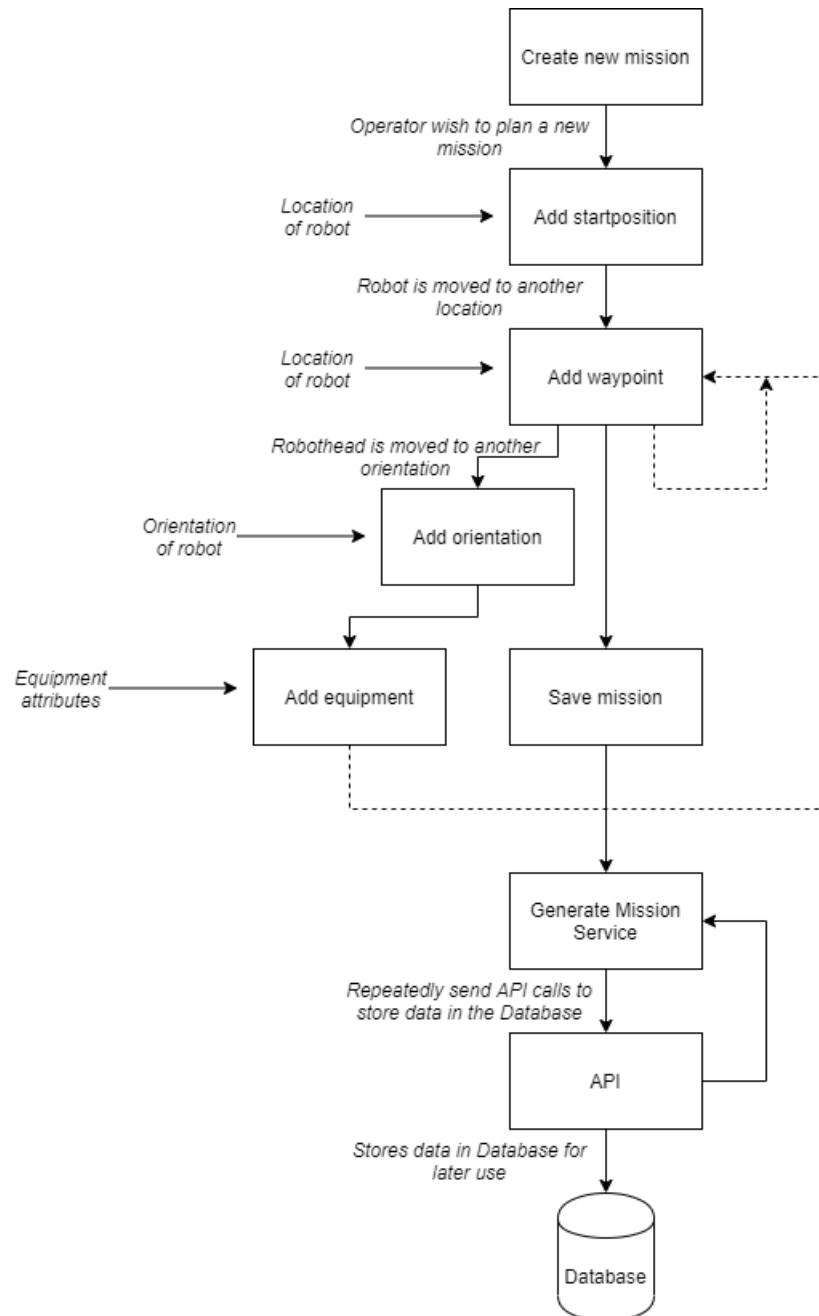


Figure 4.22: Create new mission

### 4.8.2.2 Cloud computing

The cloud server receives a new image that is sent by the TCP client on the robot controller. The image is decoded from bytes to an array again, and then served too the CNN model for further computation. The CNN will predict the equipment type and based on the prediction it's forwarded to image processing and OCR or straight trough to the API that inserts it in the database. Figure 4.23 shows the overview of this system.

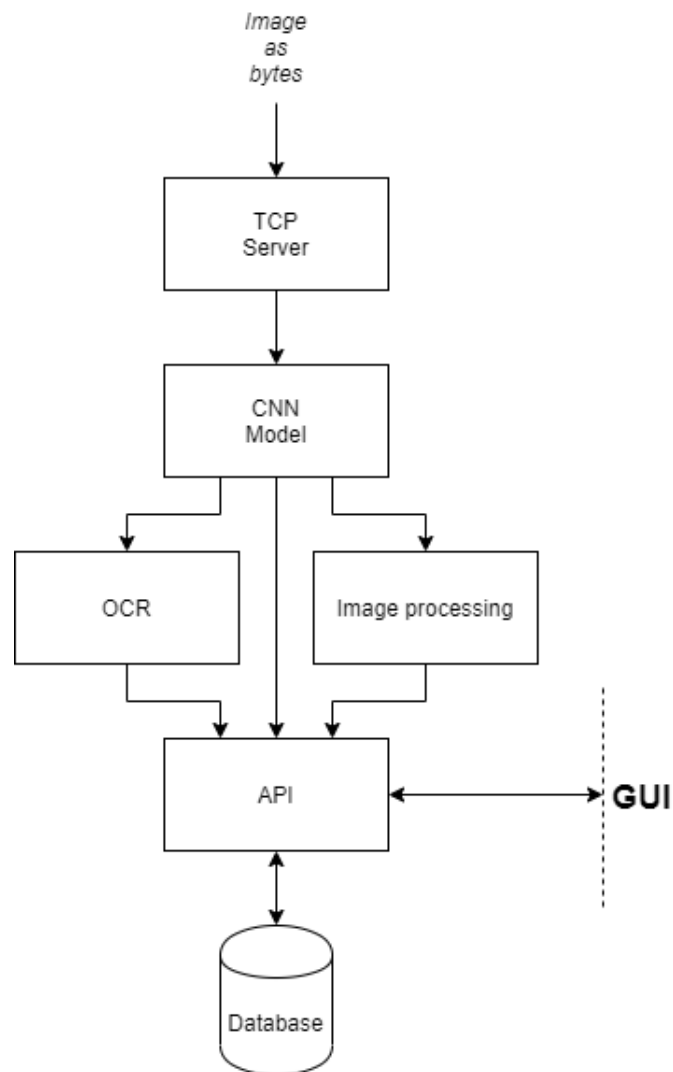


Figure 4.23: Cloud computer overview

### 4.8.2.3 Robot controller

The robot controller is the Jetson Nano running on the robot. This has been made to work as a communication hub. Publishing data acquired from the hardware and distributing it to other systems that is in need of this data. At the same time it is the only physical link to the MCU, so the system is the middle-ware for communicating with the MCU and motorcontroller. The overall system of the robot controller is shown in figure 4.24.

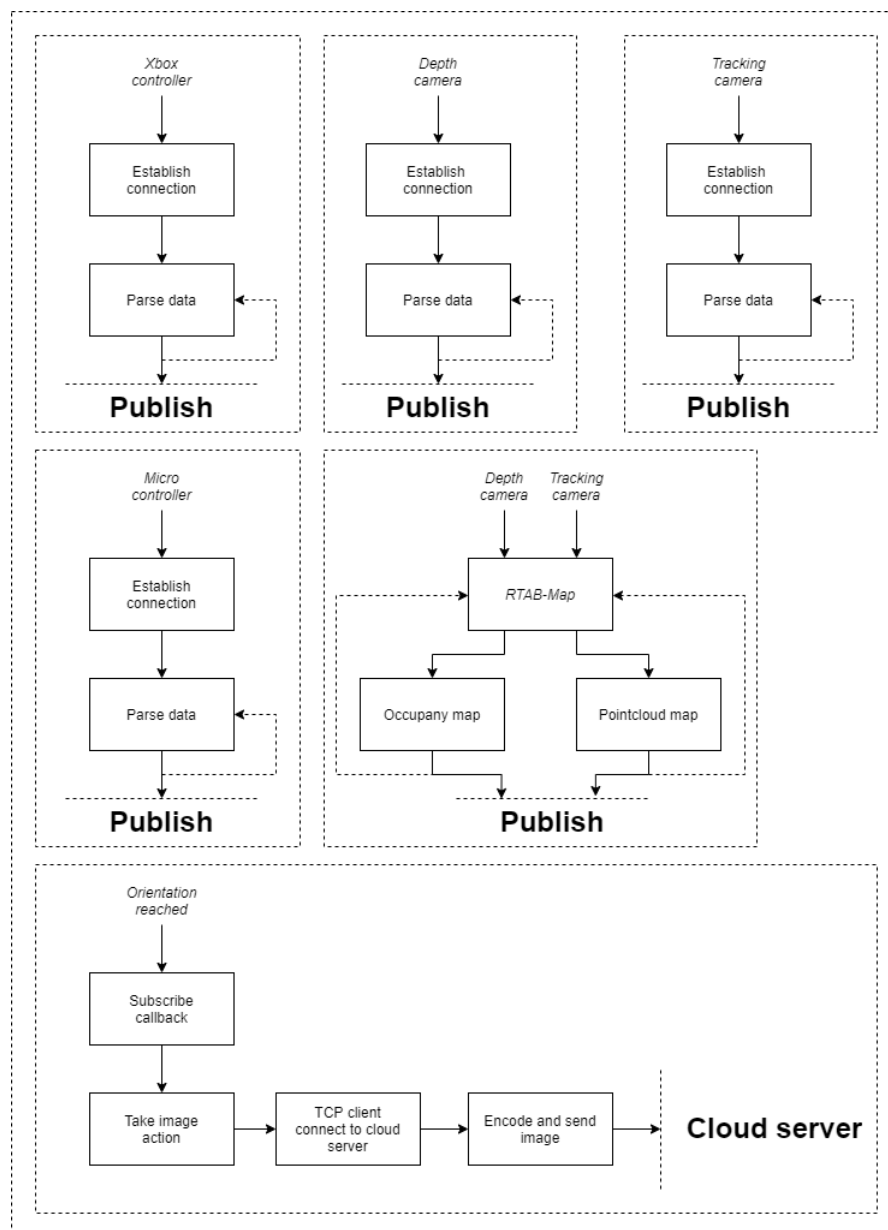


Figure 4.24: Robot controller overview

### 4.8.2.4 Embedded state machine

The main loop on the MCU contains a state machine, and can be seen in figure 4.25. As seen from the figure the state machine consists of a state for every action. This is done to simplify the structure, and minimizing processing for the MCU. The use of delay that freezes the program is also absent, and instead it's implemented timers that don't freeze the program where timed execution is necessary. This makes the loop time much more stable and reading of inputs consistent.

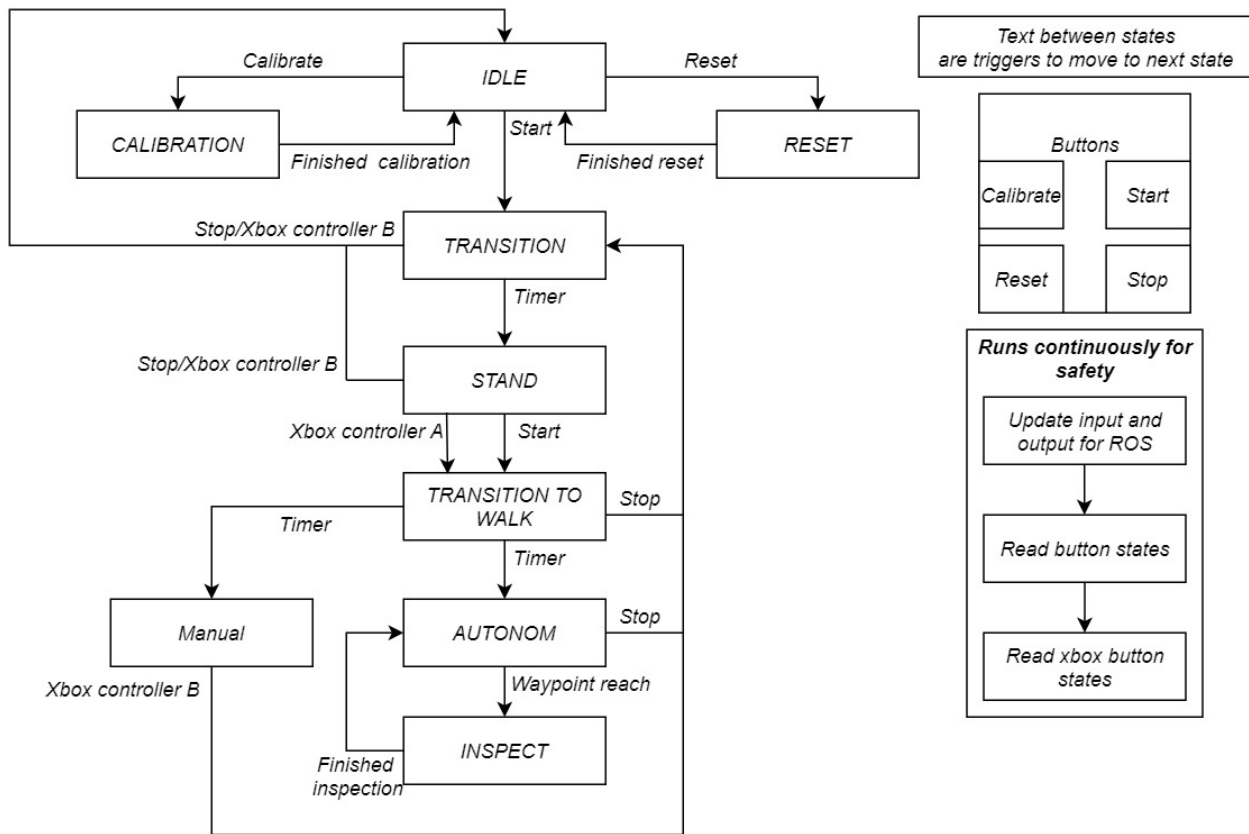


Figure 4.25: State machine overview



### 4.8.3 Robot program

This section elaborates how vital parts of the program and tasks were solved.

#### 4.8.3.1 Program structure

The program structure is split up in different smaller structures. This means the system is developed with a high abstraction level in mind. In same way, it can handle interruptions, fails and other causes that will disrupt normal function of the program. In the end, it is the graphical user interface that binds everything together, and the other services/processes working collaboration with this. Working as branches for external input to be processed and parsed to the correct format and published to other resources.

All low-level logic, such as checking sensors and driving motors, will be handled by the MCU. Interactions between the Jetson Nano and the MCU are based on commands. The MCU is responsible for utilizing the data. The data contains information like *xbox controller inputs*, *linear velocity*, *angular velocity*, *goal reached*.

### 4.8.3.2 Operation flow

The robot operates in five states, see Figure 4.26. The first state is *IDLE*, and here the system waits for further instructions after being turned on. Second state is *CALIBRATE* which has to be started with a user input. This is where the motor, encoder and mapping calibrations takes place. If any errors occurs during calibration, the system moves to *IDLE* state where actions from the user is required to proceed to *RESET* state to check and reset errors. If start is pressed the robot will move to either *MANUAL* or *AUTONOMOUS* if the robot executes the previous instructions without errors. *MANUAL* state respond to interaction with the Xbox controller. Where the robot can perform forward, backward, adjust height and turn operations. The *AUTONOMOUS* will do operations based on the input from the navigation system.

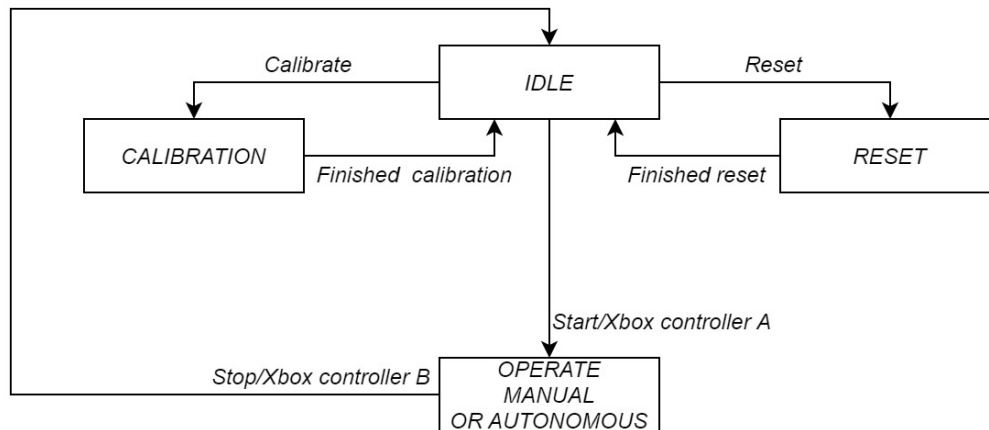


Figure 4.26: Main operation flow

### 4.8.3.3 Operation state flow

Figure 4.27 shows what happens from the moment it is powered up and goes in depth when the user has pressed start, and the main operation program enters the *AUTONOMOUS* state. Here the program reads the input and writes output to ROS every loop. The input from ROS controls the robots movements based on the navigation system. The button states are also check every loop, and if at any point the stop button is pressed the robot stops. The robot will move around until the robot has reach its goal and ROS tells it to inspect. When inspecting the robot can tilt for a better angle to take pictures. When the inspection is complete the robot will again move, until either the stop is pressed or a new goal is reach.

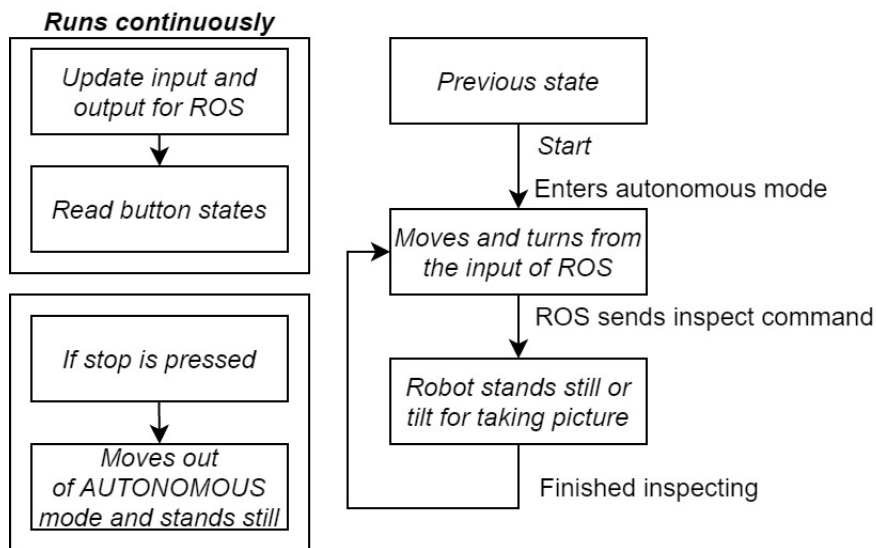


Figure 4.27: Main operation

### 4.8.4 Robot calibration

Calibrating the robot must be done before any other motion can be applied. The calibration gives the area for which the robot can move freely, but is also the foundation for the transformation from the inverse kinematics model to encoder coordinates.

Calibration is done by setting the ODrive in a calibration mode where every single motor is turned approximately one revolution in each direction to measure motor specific data (e.i. phase inductance and phase resistance).

## 4.8.5 Robot motion

This section describes how the robots moves its motors

### 4.8.5.1 Trajectory control on motors

How the motors move from their actual position to a new set point is done in trajectory control mode. The mode makes it possible to limit the acceleration, deceleration and max speed of the trajectory. With doing so it's possible to alter the movement speed of the motors and therefore the velocity of the robot. An example of the trajectory for a new position of the motors can be seen in figure 4.28

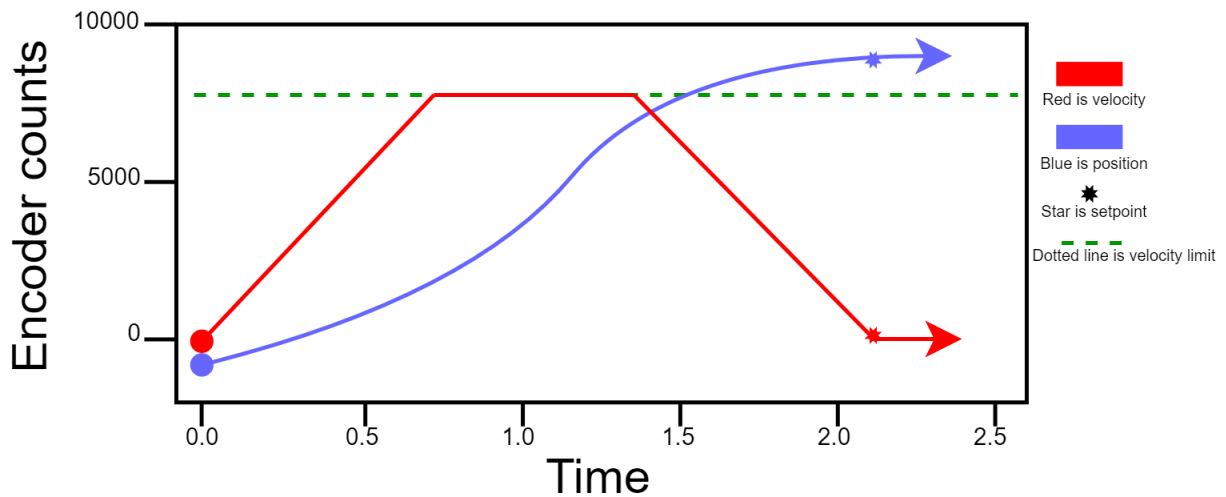


Figure 4.28: Trajectory Control on motors

### 4.8.5.2 Gait controller

The gait controller controls how the four legs work together to move the robot. The chosen gait was trot and how this works can be seen in figure 4.29. When using a trot gait two and two diagonal legs makes a pair and work together. When pair one moves backwards they have contact with the ground making the robot go forward, while the other pair is in the air moving forward. This movement alternate between the pairs and move the robot at a steady pace forward or backwards. If the step length on one of the sides is change the robot will turn, and it will always turn to the side with the smallest step length. How fast it turns depends on how different the step lengths are.

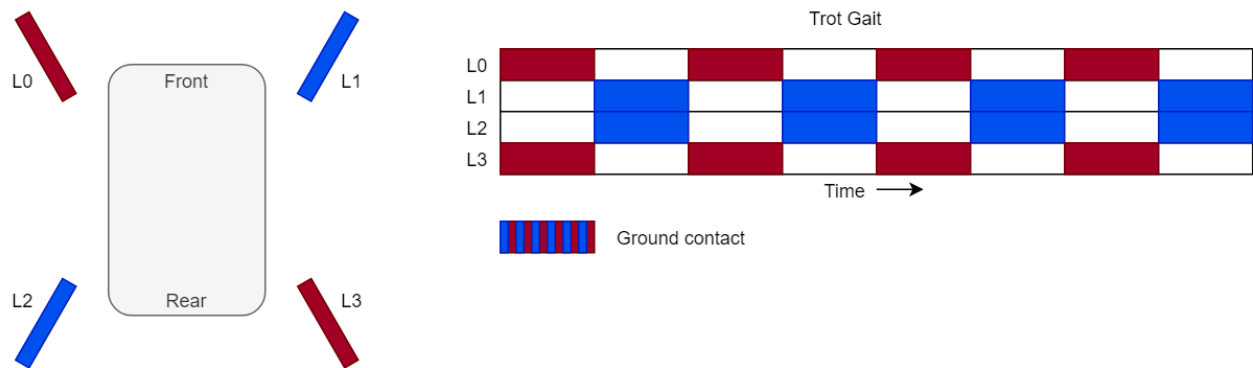


Figure 4.29: Gait controller

### 4.8.5.3 Inverse kinematic

From the sinusoidal curves that was mentioned in section 4.3.1 the x and y for moving the legs are calculated. By using this in a coordinate system it is possible to calculate what position the motors should be in.

The different angles mentioned and an example of a leg placement can be seen in figure 4.30.

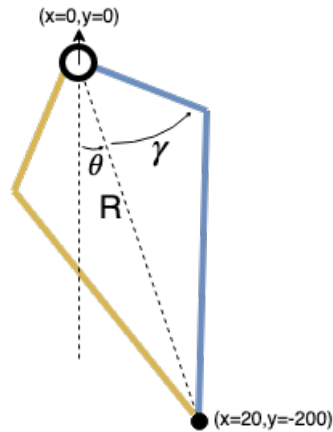


Figure 4.30: Inverse Kinematic

First the  $R$  is calculated

$$R = \sqrt{x^2 + y^2} \quad (4.1)$$

The  $R$  is used for making sure that the legs do not go out of bounds.

Then  $\theta$  and  $\gamma$  are calculated

$$\theta = \arctan\left(\frac{x}{y}\right) \quad (4.2)$$

Using the Law of Cosines

$$\gamma = \arccos\left(\frac{8100 + R^2 - 25600}{180 * R}\right) \quad (4.3)$$

From this the motor angles can be calculated. For the right foot the equation is

$$\alpha = \gamma - \theta \quad (4.4)$$

For the left foot the equation is

$$\alpha = -\gamma - \theta \quad (4.5)$$

The leg separation( $\gamma$ ) are the same for both legs, but with opposite sign. Then  $\theta$  is subtracted from  $\gamma$ . When the foot is on the right side of the center ( $+x$ ) the  $\theta$  is negative. This means the right foot will get a larger angle, and the left foot will get a smaller angle. The opposite occurs when the foot is on the left side of the center ( $-x$ ). When the finale angle ( $\alpha$ ) is calculated this is mapped from  $-360^\circ$ -  $360^\circ$  to  $-6000$  -  $6000$  motor counts.

### 4.8.6 VIO

The Jetson retrieves the orientation quaternions, that has the variables  $x, y, z$  and  $w$ , from the VIO in the Intel T265. After this the quaternions are sent to the MCU where they are change to pitch, yaw and roll angles. The equations 4.6, 4.7 and 4.8 shows how the quaternions are calculated to pitch, yaw and roll, respectively. The  $\arctan2$ , mention in equations 4.7 and 4.8, is a function in C++ called  $\text{atan2}$  that returns the principal value of the arc tangent of  $y/x$ , expressed in radians. The last part of the equations  $\frac{180}{\pi}$  is the change from radian to degrees.

$$Pitch = -\arctan(2 * (x * z - w * y)) * \frac{180}{\pi} \quad (4.6)$$

$$Yaw = \arctan2(2 * (w * z + x * y), w^2 + x^2 - y^2 - z^2) * \frac{180}{\pi} \quad (4.7)$$

$$Roll = \arctan2(2 * (w * x + y * z), w^2 - x^2 - y^2 + z^2) * \frac{180}{\pi} \quad (4.8)$$

### 4.8.7 PID-Controllers

#### 4.8.7.1 PID controller for motors

The PID controller for controlling the motors are implemented directly in the ODrives. Changing the values of the controller changes the speed, stiffness and overshoot of the motors. When tuning the PID the stiffness of the robot was important, because the robot legs need to behave like a suspension, but still be stiff enough to lift the robot.

The PID controller implemented can be seen in figure 4.31. From the figure one can see that the implemented PID are actual a P, PI and PI in cascade. This is predefined in the ODrive and there are only three values to change, position gain, velocity gain and velocity integrator gain.

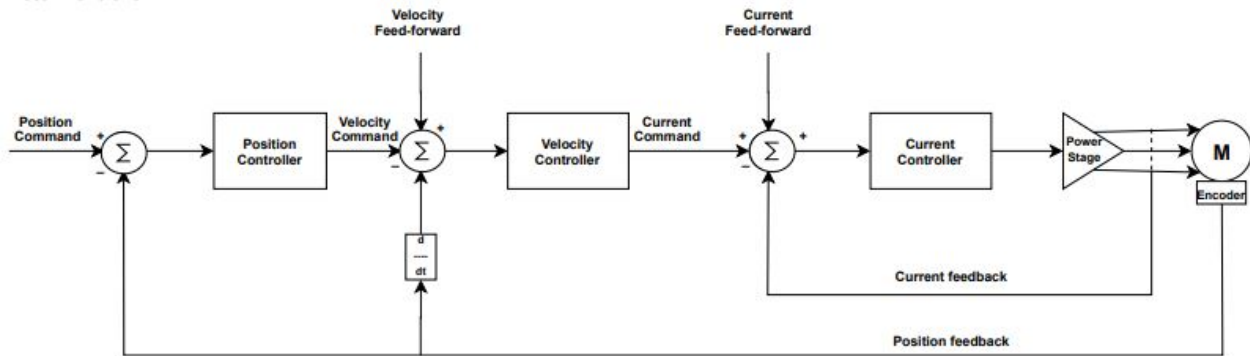


Figure 4.31: PID for motors [85]

#### 4.8.7.2 PID controller for stabilize the robot

The PID that makes the robot stabilize itself gets the current value of the orientation from the processed VIO data. After calculating the error of the pitch, yaw and roll from their setpoints the value gets added or subtracted to the height of the robot. With doing so the robot will try to keep itself completely level at all time. Figure 4.32 illustrates the implemented PID, and it's a standard PID controller.

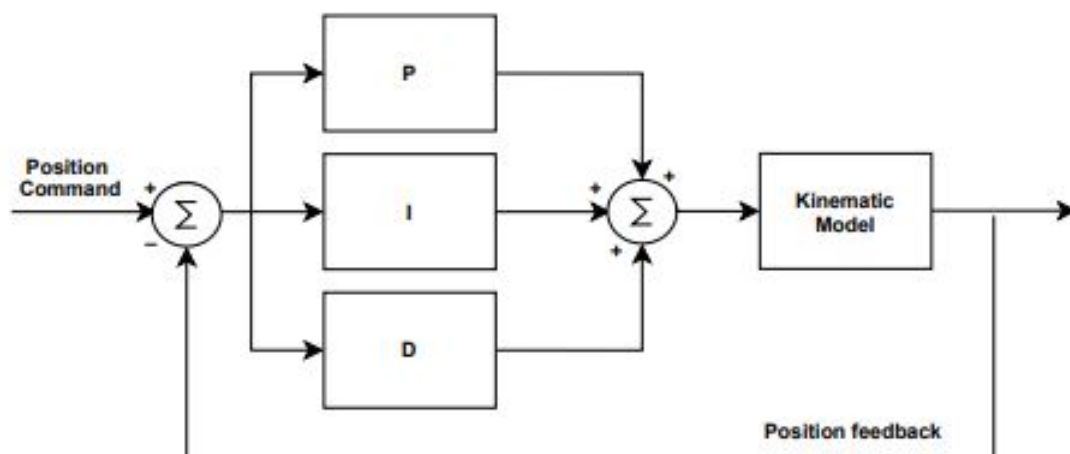


Figure 4.32: PID for pitch and roll



### 4.8.8 Xbox controller

The Xbox controller is used for controlling the robot in manual mode. The MCU receives and stores all button values, but only the buttons that have names in figure 4.33 are in use.



Figure 4.33: Xbox controller buttons

Figure 4.34 shows the value range the MCU receives, and what the default value is when the buttons/joystick are in their default state. Further it shows the value range the MCU map the input to. Last it explains what the different buttons are used for.

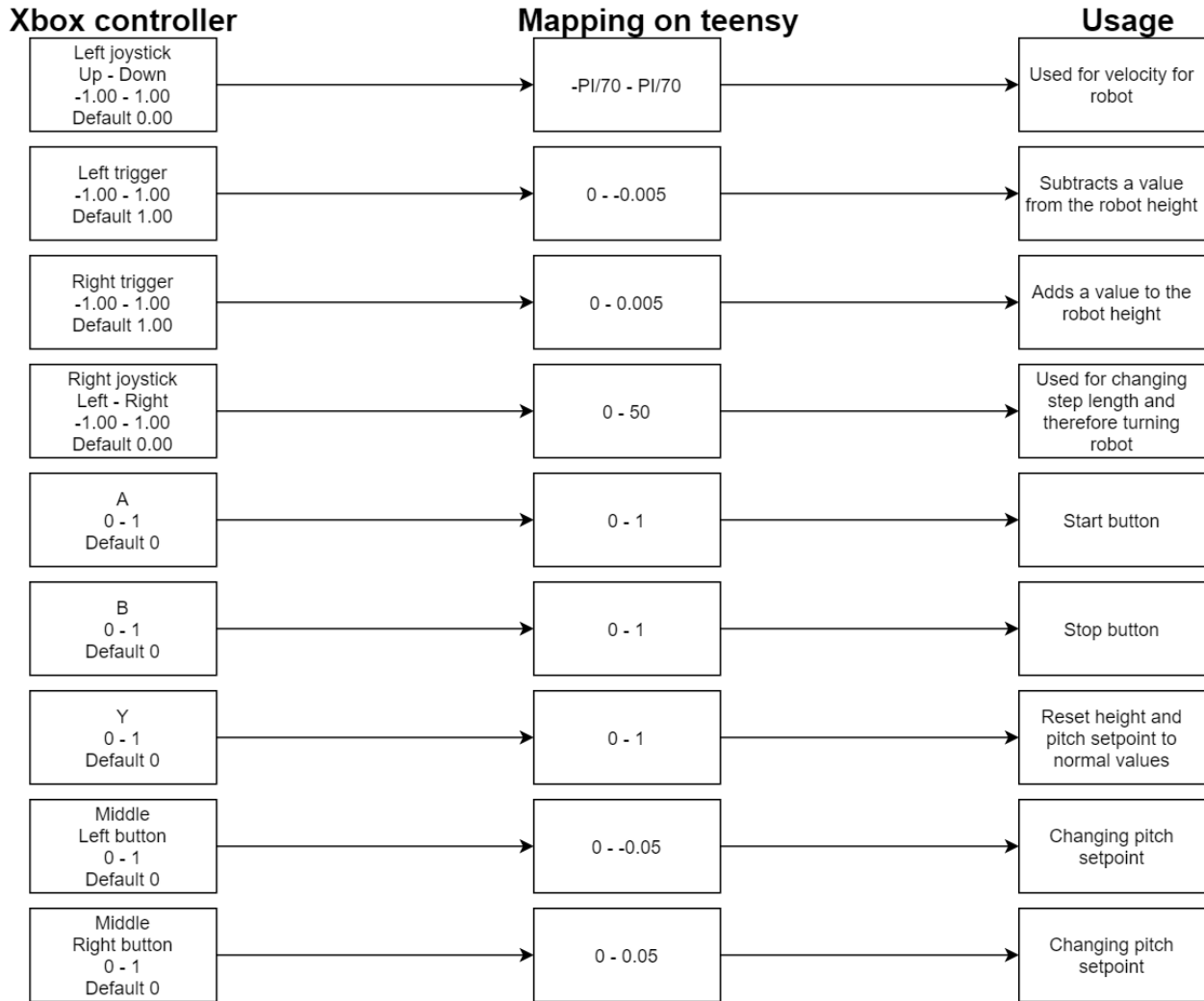


Figure 4.34: Xbox controller mapping

### 4.8.9 SQL Database

The implemented SQL database is implemented in the cloud, and are accessible for anyone to read, however the writing or updating values is strictly restricted to only the once who have access to the login credentials. As a result does the database operate as a safe and reliable source to save and access mission protocols and equipment with attributes.

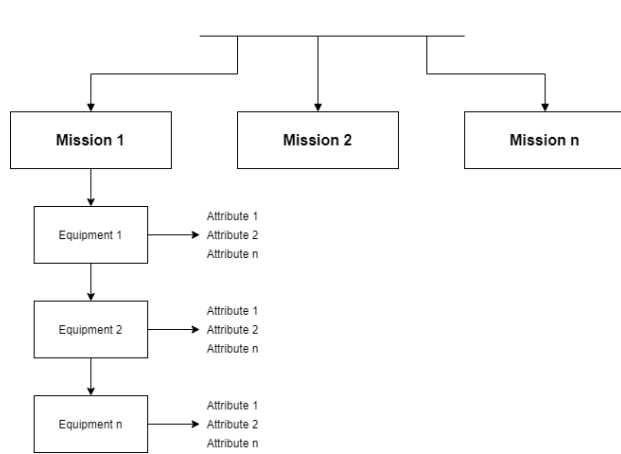


Figure 4.35: Database structure

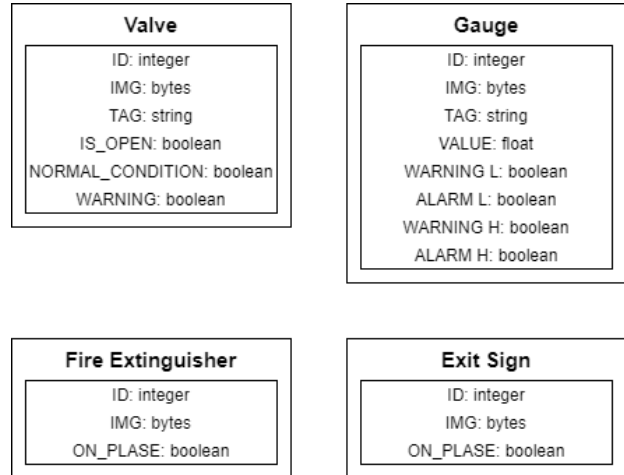


Figure 4.36: Database models

The database structure is sorted into missions. Where each mission have the given equipment that is in the mission plan. Each of the equipment have attributes, which can be seen in figure 4.35 and 4.36. A common attribute for all equipment is *ID* and *IMG*, where *ID* is used for equipment identification, and *IMG* is the image taken during mission execution. Other than that, does the equipment have several other values that is used for warning or alarm handling in the GUI.

### 4.8.10 REST-API

As mentioned above is the REST-API the middle link between the database and external devices/systems. The API runs on the cloud and awaits requests. The requests received is given in a special pattern.

### 4.8.10.1 Login

The first functionality the API brings is the *Login* functionality. As shown in figure 4.37 is the login used to acquire a valid token. The token is a JSON Web Token(JWT) and is generated to be valid for 5 days before being invalid. Operators need a valid token to *POST*, *UPDATE*, *DELETE* anything from the database. The *READ* function is open for anyone because reading can not alter any of the stored data.

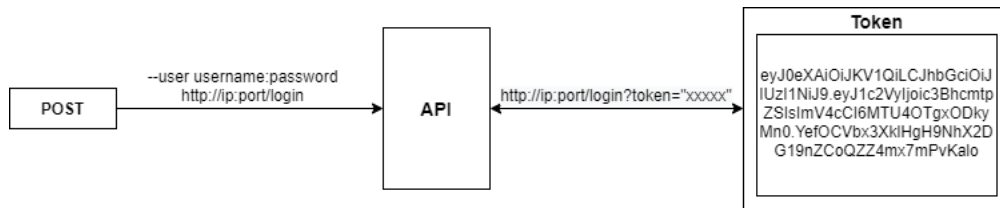


Figure 4.37: REST-API Login

### 4.8.10.2 Get

The get requests correlates to read. With the given request shown in figure 4.38 will this tell the API to send a *query call* to the database. Based on the query call the database will look on the given mission-ID and respond with the found valve at the given valve-ID. The API will redirect the result to the operator/system that requested the data.

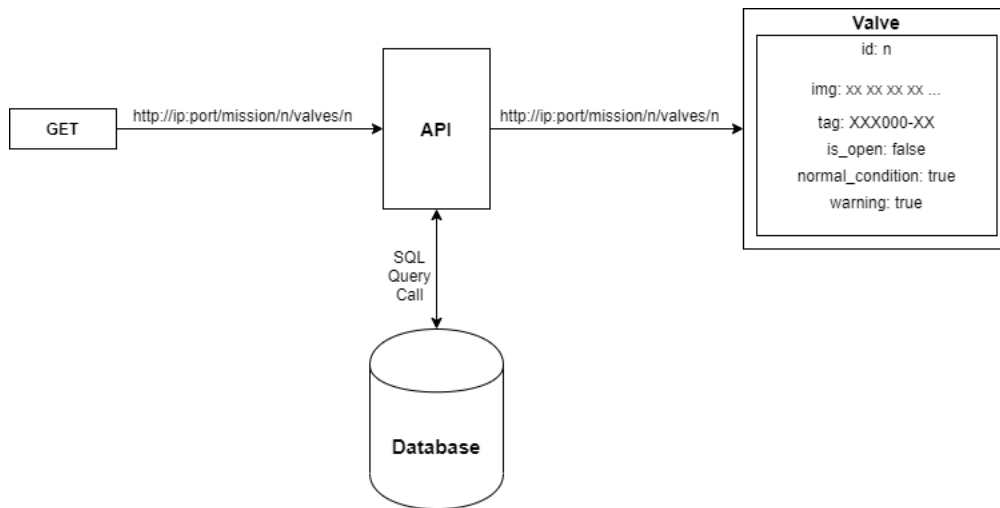


Figure 4.38: REST-API Get

### 4.8.10.3 Post and Update

The post and update goes under same section, cause of the similarities they have, and are shown in figure 4.39. Both needs to have a valid JWT token to post. Bundled in the request is content with data formatted as JSON. This content will be posted to the API, which if the token is valid, will store the data in the database. The only difference between post and update is that update does not need to have all the attributes in the content. It can update only parts of the attributes the equipment holds.

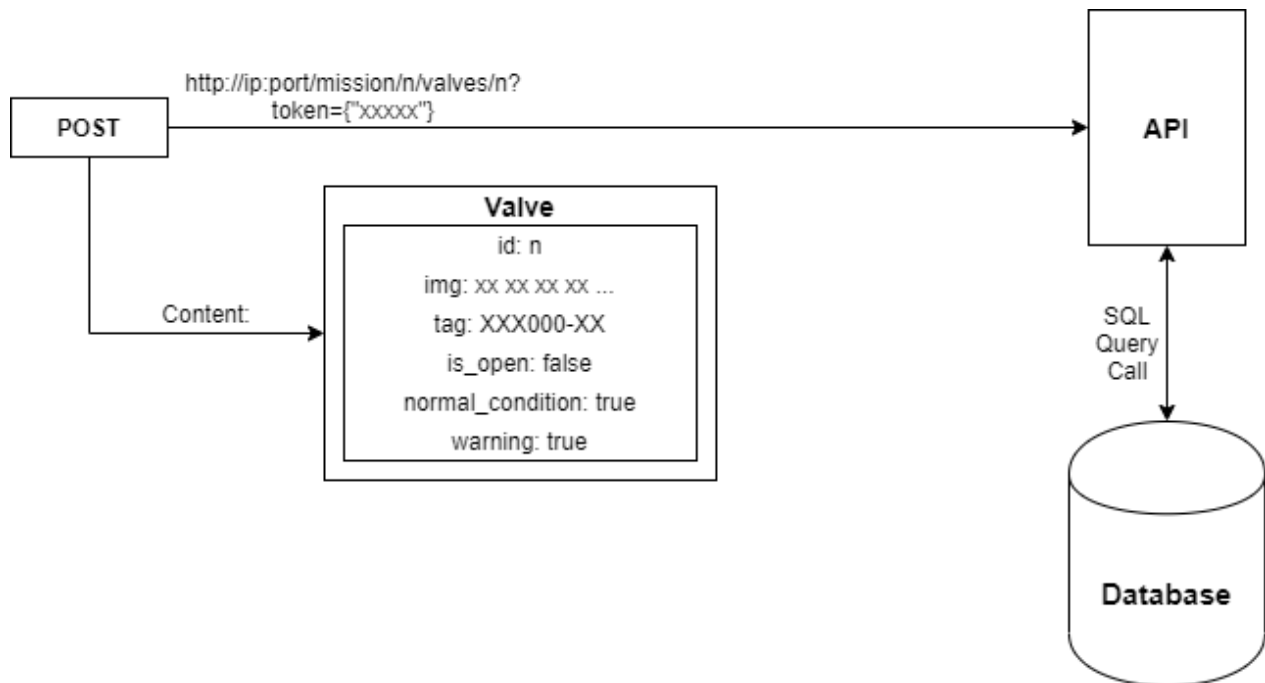


Figure 4.39: REST-API Post

#### 4.8.10.4 Delete

The last of the five functionalities the API brings is the delete functionality. This operation is used to remove a entry in the mission, or simply remove the whole mission itself. As for post and update, is a valid JWT token required to do such a operation. A delete request can be seen in figure 4.40.

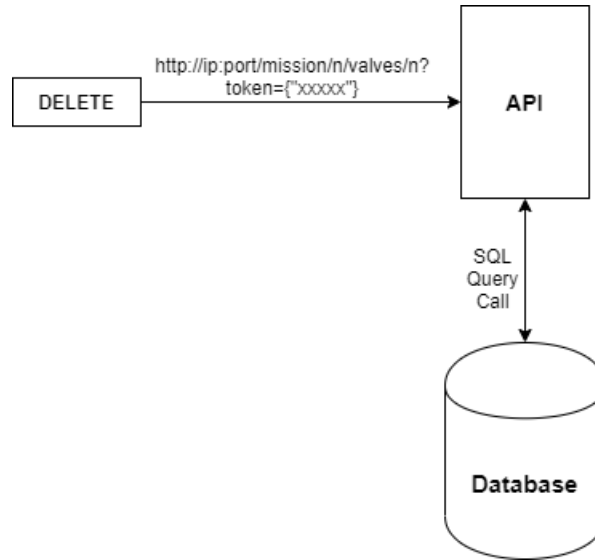


Figure 4.40: REST-API Delete

### 4.8.11 Navigation

The navigation system is built up from components in the ROS library *move\_base*. As seen in figure 4.41, the system gets inputs from three external systems. The first one is the *Pointcloud* that goes in **Global costmap** and **Local costmap** where it is used to calculate the costmap that constrains the robots motion-space. Based on the *goal* input and the constrains from the costmap will the global planner calculate the optimal path to move along. As the robot can never follow the plan exactly will the local planner plan velocity commands repeatedly to achieve the lowest cost while the local costmap is being taken in consideration.

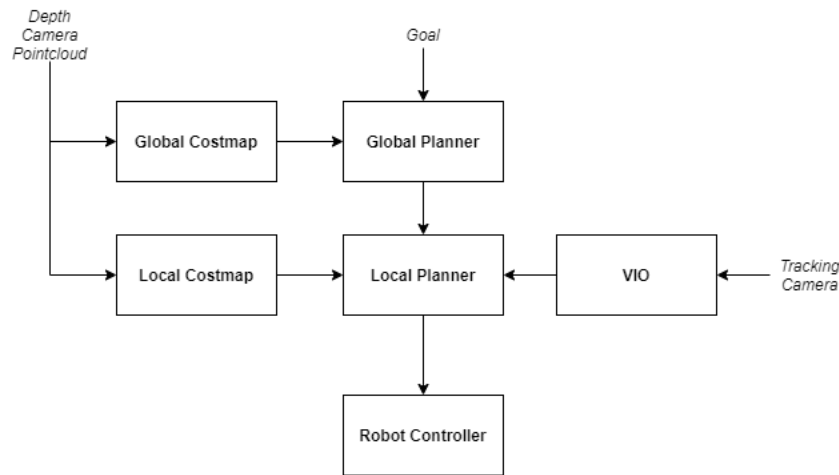


Figure 4.41: Navigation

Built in VIO from the Intel T265 camera is being used to get the robots position in space. As camera is in front, a translation of the odometry has to be done to move it to the center of the robot.

Robot configurations has also been set to the controller as constrains. Some of the parameters set is e.g. the accuracy threshold in linear position and orientation and others limit the acceleration and velocity the robot can execute. In the same way have both the global and local planner parameters that limits their search range and update frequency to be effective and computational possible within a set time frame.

### 4.8.12 Image processing

During the project there were developed two image processing procedures. The first technique was meant to extract the correct value from the gauge. This was to report value, but also compare it against the warning and alarm limits set by the operator. The second method was pre-processing on the image before it got sent to the OCR. This was supposed to make the OCR be more robust and able extract text that otherwise might not be possible.

#### 4.8.12.1 Gauge value detection

Figure 4.42 displays all the operations that takes place to extract the value the dial is pointing at. As the image is classified and the equipment is detected as a gauge by the CNN it is ready to extract the value of which the dial is pointing at. This task is done in multiple parts. The following section will go more in depth and data have been taken out and displayed to highlight each step of the process.

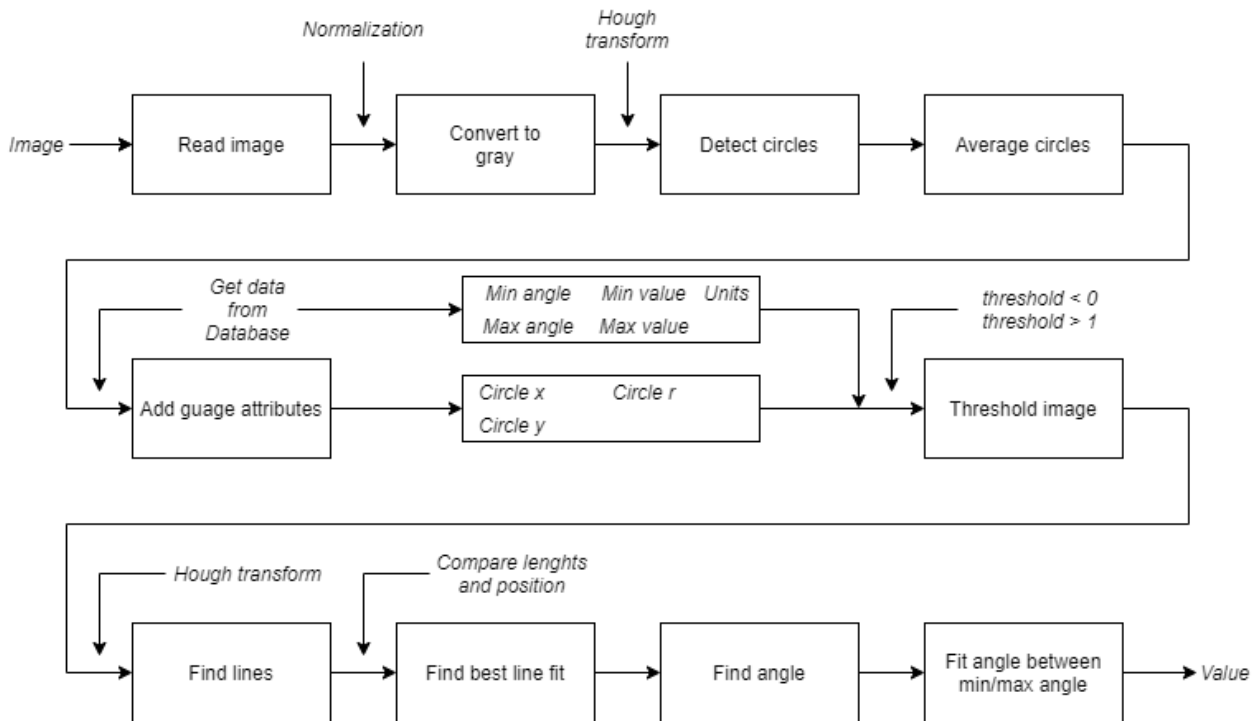


Figure 4.42: Gauge value detection



The first task was to get the gauge input image, figure 4.43, acquired from the gauge detection CNN, as a matrix with  $(600, 480, 3)$  shape. Secondly, the image figure 4.44 was converted to a gray-scale image  $(600, 480, 1)$ . Figure 4.45 and 4.46 shows circles and numbers that are drawn on top of the original image. This was just for illustration purpose, and the gray-scale image was used throughout the rest of the process.



Figure 4.43: Gauge input image



Figure 4.44: Gauge image gray

The following sequence finds the center point and the circle around the glass where the values that are placed are used as references to adjust the values in the database. The following process with storing information about the gauge is done in the mission planning interface, which is mentioned in section 4.8.9.



Figure 4.45: Gauge found circles

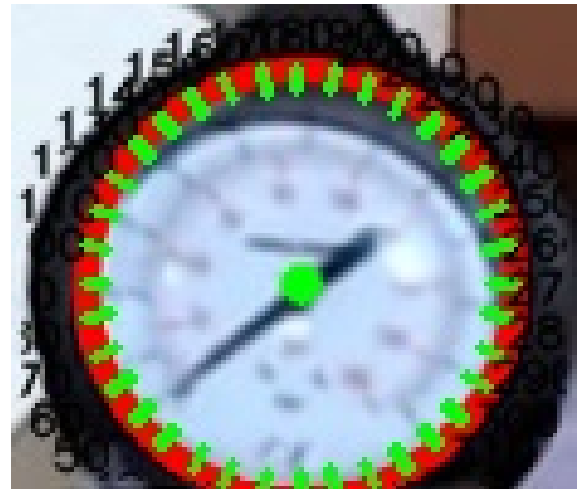


Figure 4.46: Gauge preprocessing

The angle of the dial was found by first thresholding the image to highlight the black parts and subtract the white one. This separated the dial from the background as seen in figure 4.47, and a Hough Transform was applied to find all of the lines. This resulted in a fine line in the middle, while noise around the circle was present as seen in figure 4.48. By comparing the length of the lines to a minimum length and the position against the origin, the noise was found and subtracted, which resulted in the dial line being the only line left.



Figure 4.47: Gauge threshold



Figure 4.48: Gauge lines found

When the furthest point from the origin was found, the angle could be calculated:

$$\theta = \sin^{-1}\left(\frac{(x1 - x)}{(y - y1)}\right) \quad (4.9)$$

and then pinpoint where the dial is pointing towards in comparison with the start angle, end angle, start value and end value that was found in the earlier stages.

### 4.8.12.2 Preprocessing for OCR

Like the gauge value detection, OCR also benefited from the color to gray-scale conversion. That being mentioned, more preprocessing was needed to increase the chance of extracting the written tag on the equipment. The following task is skew correction which simply applies translation and rotation to the image to get the text straight. The last preprocessing step was to erode the text which made it thicker and more visible. At this point, the necessary steps were done to increase the success rate of the OCR to extract the correct tag. The whole process can be seen in figure 4.49.

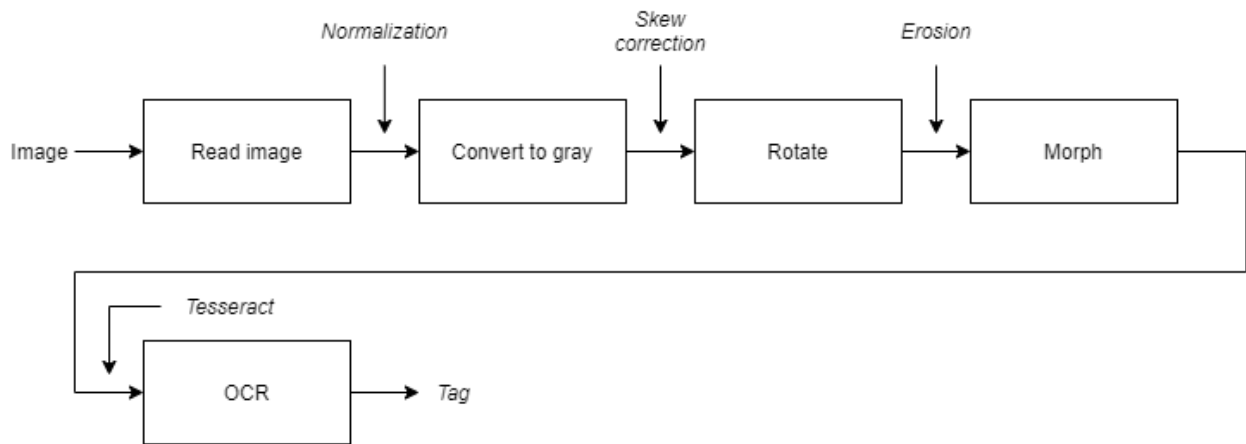


Figure 4.49: Preprocessing for OCR

## 4.8.13 Convolution neural network

### 4.8.13.1 Collecting data

It was important to gather as much training data as possible since the CNN model's accuracy scaled good with the data given. As a result of this, it was collected 356 images containing the different equipment types, aiming to capture the equipment from multiple angles and surroundings to generalize the model while also making it more robust to external noise. Three examples of pictures taken can be seen in the figures 4.50, 4.51 and 4.52



Figure 4.50: Training example 1

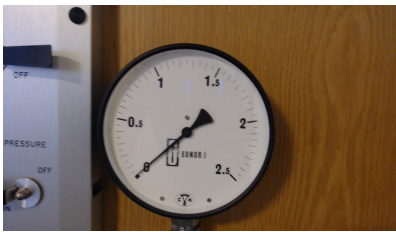


Figure 4.51: Training example 2



Figure 4.52: Training example 3

### 4.8.13.2 Labeling data

Labeled data is a designation for pieces of data that have been tagged with one or more labels identifying certain properties or characteristics, or classifications or contained objects. This was necessary to use a supervised machine learning setup. As shown in the figures 4.53, 4.54 and 4.55 there is now a bounding box around the objects that the CNN should detect and classify.

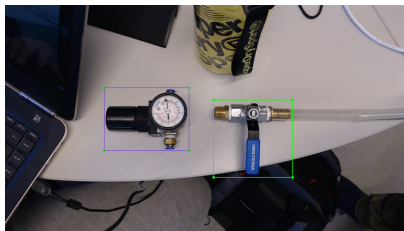


Figure 4.53: Training example 1

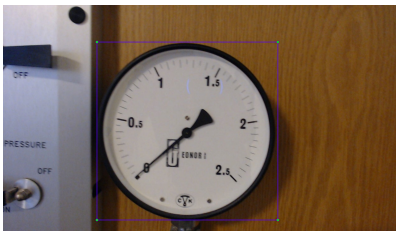


Figure 4.54: Training example 2



Figure 4.55: Training example 3

From the following images, was XML file generated containing the data e.g. *filename, size, id, labels and bounding box coordinates*. This XML file, containing data seen in table 4.1, had to be converted to CSV-format before it could be used by the model.

Filename	Width	Height	Class	Xmin	Ymin	Xmax	Ymax
filename1.jpg	1280	720	Gauge	578	282	763	447
filename2.jpg	1280	720	Exit_sign	596	379	695	486
filename3.jpg	1280	720	Closed_valve	487	230	654	400
filename4.jpg	1280	720	Open_valve	766	307	974	481
filename5.jpg	1280	720	Fire_extinguisher	398	409	518	518

Table 4.1: Training data information

### 4.8.13.3 Training

The training of the CNN model was done with Google's machine learning framework, TensorFlow. The training saved checkpoints of the model every 10000 epoch, keeping the latest top 5 if there would be any interruption during the training process. The training took approximately 9 hours to complete 200000 epochs. However, the training can be done in much less time, which will be explained more in-depth in the results chapter 5.2.9.

### 4.8.13.4 Evaluation

It was important to evaluate the model before it is being used in operation. The trained model was therefore introduced to new, never before seen images, and on these it performed the object detection and classification seen in the figures 4.56, 4.57 and 4.58.



Figure 4.56: CNN Evaluation Example 1

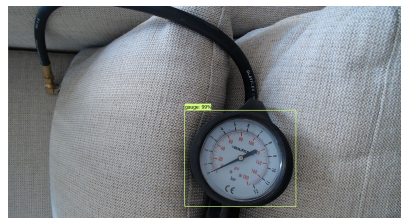


Figure 4.57: CNN Evaluation Example 2

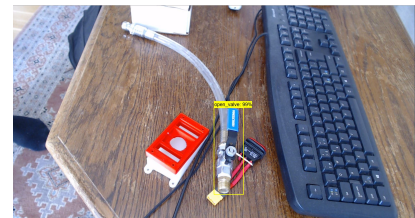


Figure 4.58: CNN Evaluation Example 3

# Chapter 5

## Results

### 5.1 Reviews

This chapter elaborates the tests and reviews performed during the project, and how they affected the final result.

#### 5.1.1 Electrical testing

Before testing the robot, all the wiring of the components on the robot needed to be controlled. This was done to make sure the wiring was done correctly and that there were no short circuits that could potentially damage the electrical components. Consequently all connection points and wires were measured with a multimeter. After the test was completed with no faults, the voltage was applied and measured that all of the different voltages were correct.

### 5.1.2 System testing

Before assembling all the parts together, a set of tests were done. These tests made sure that all the connected components were working and configured correctly in the software. The tests done and the result is shown in table 5.1.

Function	Result
Test emergency stop	Works
Test buttons	All four works
Test leds	All four works
Test motors	All eight works
Test encoders	All eight works
Test cameras	Both works

Table 5.1: System testing

#### 5.1.2.1 Leg movement testing

The leg movement of the robot went through several iterations. In the first iteration all the legs moved synchronised as can be seen in [progress video one](#) in appendix C.2. The reason for this was that the most important thing was to make the legs able to follow the sinusoidal movement. After achieving this it was quickly realized that moving all the legs synchronised would not make the robot move. Therefore the trot movement was chosen as the primary gait. This was implemented before the robot was ever tested on the ground, and therefore the synchronised leg movement was never tested in practice.

### 5.1.2.2 Robot testing

The tests that were done was heavily inspired by what the core objectives the robot should perform. This includes leveling itself, walking straight and turning without any input, get controlled by an Xbox controller and testing all the features the autonomous mode had to offer. The progress videos that can be found in [appendix C.2](#) shows the testing environment and improvements that were achieved under testing.

One problem encountered while testing worth mentioning was that some of the motors tended to become too warm, resulting with a break so the motors could cool down.

### 5.1.3 Communication

#### 5.1.3.1 RX/TX

The sending from the MCU to the Odrive and the other way around are following a set of templates and already made methods from an existing library. The templates and methods that are used can be seen in the source code [E.1](#).

### 5.1.4 Design reviews

In the process of assembling the robot it was discovered a need for modification of several components. Most of the changes that had to be made were due to unforeseen interactions with other components or the restrictions of the robots area of movement. The changes were mostly implemented with the 3D-printed parts as these are the easiest to reproduce.



### 5.1.5 Software reviews

Throughout the software progression and development of structures, a variety of different options has been considered, where some of them would work fine and as expected. By reviewing the code logic and structure and seeking guidance, different and better versions with looser coupling has been developed.

The switch to ROS made the proposed software solution change completely. The reason for this is that the first implementation was made to run each hardware component in a different process, and different components were a publisher or a subscriber, or in some cases both. Most of the communication was up and running and the implementation of the navigation system was under development. It was at that point the Jetson Nano had trouble to run everything at once, and most of the sensor data did not deliver as intended anymore. However, with use of ROS and the ecosystem it brings with it, the Jetson Nano published data at a much better rate and even led to a better systematic architecture regarding the communication between systems.

## 5.2 Final results

This chapter elaborates on results of the various systems implemented in the robot and the robot in its entirety. A demo of the robot can be found here: [demo video](#). Figure 5.1 is a picture of the final robot.



Figure 5.1: Final robot

### 5.2.1 Work-space of the robot

The robot was designed to be used within several types of work spaces, ranging from large land based process plants to smaller oil platforms. This means the robot should have a great mobility and be able to maneuver tight spaces and narrow passages and other sites with confined space.

The robot has a good mobility and in combination with its small size, with its area being only 460mm long and 280mm wide, it has good maneuverability. This is evident in the testing where

u-turns in narrow spaces and navigation through narrow passages are performed. In addition to this, with the height being only 230mm when set at its lowest level, it can move through tiny spaces inaccessible to humans. And even though its height is short it can still inspect object higher from the ground by tilting its front side (where the cameras are mounted) upwards. In this manner it can still see objects such as exit signs etc.

### 5.2.2 Roll and Pitch regulators

Regulators for the robots pitch and roll rotation were implemented to stabilize the robot. Post-processed sensor readings from the VIO were used as reference value where the respective angles were applied in the two separate pitch and roll PID-controllers. Both control values given from the regulators were scaled in millimeters to be able to directly apply them to the height setpoints for the legs. The control variable from the pitch regulator were added to the front legs height setpoints and subtracted from the rear legs setpoints, while the control variable from the roll regulator were added to left legs setpoints and subtracted from the right legs setpoints. As a result, the robot stabilized itself if the ground was not in level.

The regulator performance is tested by tilting a table the robot was standing on top of. The performance was tested with several PID settings, where the final settings and results can be seen in the tables 5.2 and 5.3.

Settings	
P	4.0
I	.001
D	-
Results	
Type	Value
Overshoot	15mm
Settling Time	1.3s

Table 5.2: Pitch Regulator

Settings	
P	2.0
I	.001
D	-
Results	
Type	Value
Overshoot	10mm
Settling Time	2.2s

Table 5.3: Roll Regulator

As can be seen from both tables, zero derivative was used in the regulators, essentially making them PI regulators.

### 5.2.3 Robot motion

The motions of the robot legs were accurate and had the rapidity needed to perform moderate to high walking speed. However, challenges will arise if the leg motion is set considerably faster, as this would cause the robot leg to lag. e.g if a robot leg should perform a circular motion where the speed is highly increased it will cause the robot leg to move in a smaller circle, as it never reaches its setpoint before a new is given.

The cooperation between the legs have proven to be synchronous and to provide a decent gait. On the other side, when testing the gait performance, it was clear that different floor surfaces yielded different results in terms of gait speed. E.g flooring like concrete would yield better results than laminate floors since the concrete floor is less slippery than the laminate one. This was revealed even stronger when the robot was turning sharp corners, as it was dependent on the legs to overcome the frictional force on the outer legs of the turn and still stick to the ground on the inner legs of the turn. This proved to be difficult and many permutations of settings were tried before succeeding.

When developing the robot motion, it was intended to continuously read the position and the current that the motors have. This proved difficult as the ODrive motor drivers froze and did not function correctly when reading its registers at a frequency higher than 20Hz. Therefore, reading the position and current was not feasible, since our refresh time for sending new setpoints for the motors was far higher.

#### 5.2.3.1 Robot movement

When performing movements with the robot it reached its set position with correct orientation. The behaviour of the robot resulted in a non holonomic system, which means the degrees of freedom the robot can move in are not entirely independent. E.g. when parallel parking is performed with a car, it has to move forward and reverse to be able to move its position to the side. The robot has the same characteristic, and therefore is not able to move to the sides independently from moving forward and reverse.

### 5.2.3.2 Robot accuracy

As shown throughout extensive testing, the accuracy of the robot is decent and it is able to perform the same task several times with the same level of precision. On the other hand, a considerable deadband filter had to be implemented for the orientation of the robot. This filter was given a tolerance of 0.8 radians which is equivalent to about 45.8°. With smart planning of routes a better orientation was still realized when the robot reached its waypoints. Furthermore, tolerances of 13cm were added in the x and y direction. Consequently, the robot was able to reach its destination without oscillating around its set position.

### 5.2.4 REST-API

The REST-API is fast and scalable and can operate multiple requests concurrently. As each request spawns a new thread there are not noticeable delays with requesting or updating data. The login functionality made it secure for external hackers, and the update frequency for the token lets it not be decrypted. Duration of each API call is listed in table 5.4.

Operation	Time [ms]
LOGIN	11
GET	65
POST	70
PUT	20 - 70
DELETE	40

Table 5.4: REST API calls

The variance in *PUT* came from the various content sizes it could have. It depended on, if boolean or float values are updated, or the whole image is.

### 5.2.5 Mission workflow

This section elaborates the finished workflow when the robot was on a mission. Figure 5.2 illustrates the main tasks working while the robot executes a given mission. However major systems like *Robot Controller and Navigation Systems* contains subsystems that have their own runtimes doing different operations. This is not presented in the main workflow to simply the flow of operations done.

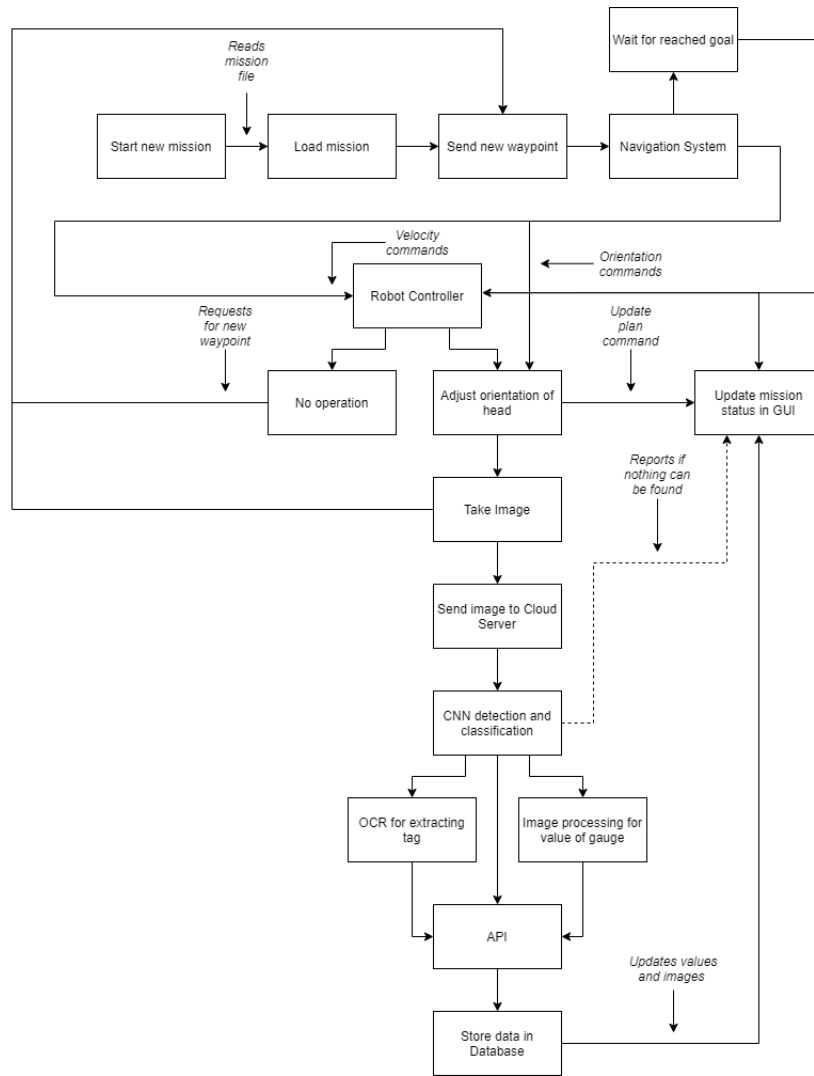


Figure 5.2: Mission planer result

The end result of the mission workflow combines all the systems to work together to execute the given mission. This operation starts with loading the mission protocol into the GUI. From there is a subsystem called to send the first waypoint to the navigation system. This subsystem will

wait until the robot has reached its position before the robot can proceed to execute the correct head adjustment. While this is happening is the navigation system running to plan the optimal global path to move, and the local planner uses DWA to publish velocity commands to the robot controller.

The robot controller sends the given velocity commands to the inverse kinematic model which calculates the correct motor positions to move the robot in the given direction. When the head orientation is in position triggers an event to notify the robot controller to take an image. This leads to a TCP client to start and sends the image to the cloud server. As this wraps up, a trigger signals to the mission planner to continue with the next waypoint.

As a result of the TCP server fetching the image, is it forwarded to the CNN model to detect and classify the equipment that is present in the image. Based on the prediction is it either sent to OCR for tag extraction and image processing for extracting gauge value, or simply forwarded to the API to store it in the database. At last the asks the GUI for the equipment attributes and display the results accordingly for the operator to see.

### 5.2.6 Performing mission

There has been a full test of the system to determine the final result. The following test is the determinant factor for how good the overall system works and the robots performance. The performance of the mission can be seen in the last clip in the [demo video](#). In figure 5.3 the robots initial position is shown. The X's on the ground symbolize waypoints that were generated during mission planning. The following equipment is marked with its initial letter; *Valve*, *Exit Sign*, *Gauge*, *Fire Extinguisher*.

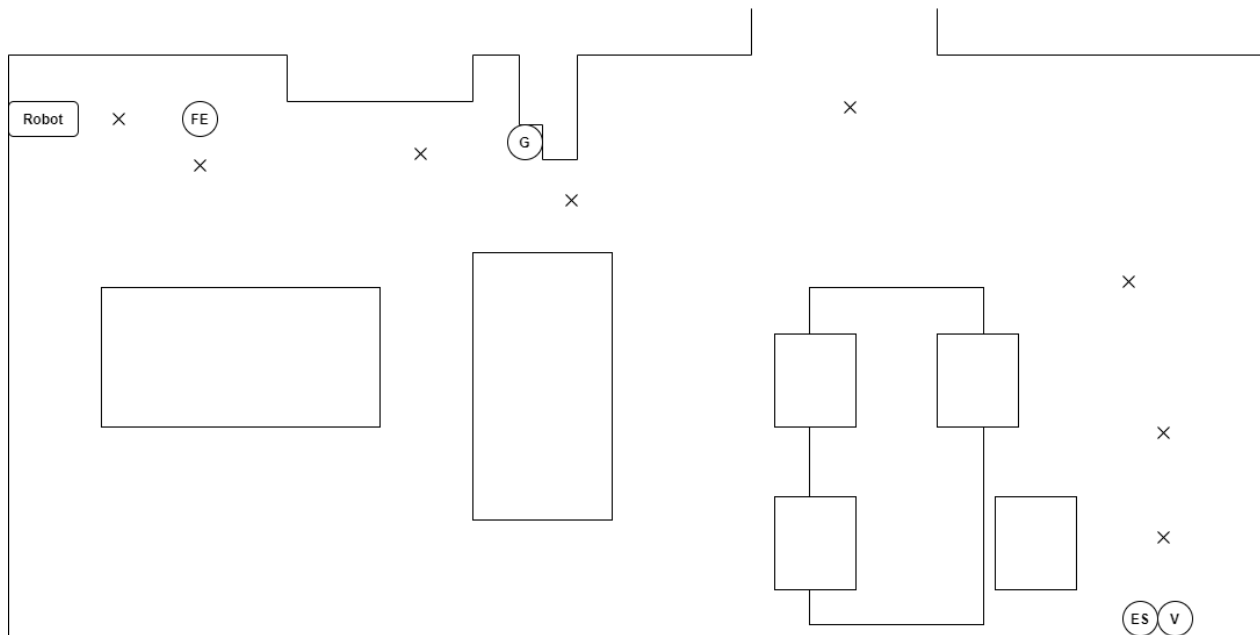


Figure 5.3: Final Mission

The operation protocol generated had the following structure seen in figure 5.4:

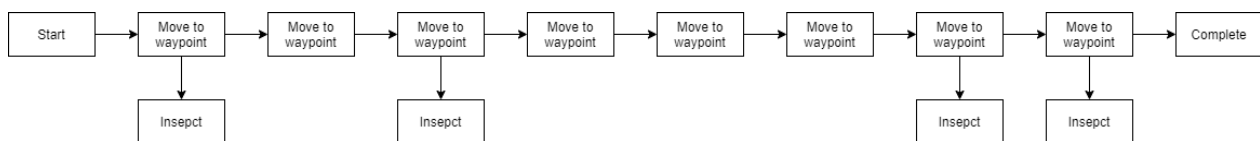


Figure 5.4: Final Mission protocol

The result is split up in different parts, as it will quantify the performance for the different systems more clearly.



### Navigation

The global planner plans each waypoint within the given update-frequency on 1Hz. Due to no other moving objects where there no replanned global plans. Table 5.5 and 5.6 was generated from the data acquired from the mission.

$x$	$y$	$\hat{x}$	$\hat{y}$
0.411	-0.029	0.415	-0.031
1.660	-0.545	1.666	-0.555
2.561	-0.513	2.566	-0.502
3.521	-0.859	3.526	-0.865
5.061	0.131	5.066	0.128
7.090	-0.321	7.100	-0.327
7.452	-1.336	7.457	-1.349
7.455	-2.309	7.457	-2.321

Table 5.5: Navigation point coordinates

$z$	$w$	$\hat{z}$	$\hat{w}$
0.007	0.992	0.000	0.999
-0.041	0.987	-0.081	0.997
-0.021	1.002	0.016	1.000
-0.108	1.068	-0.050	1.000
0.123	0.945	0.165	0.986
-0.478	0.863	-0.348	0.936
-0.694	0.689	-0.698	0.716
-0.700	0.681	-0.698	0.716

Table 5.6: Navigation angle coordinates

All of the linear and angular positions are successful at each waypoint, and the whole mission is executed without interruptions or stops. The logged linear points can be seen in figure 5.5 and it shows that both  $x$ ,  $y$  and  $\hat{x}$ ,  $\hat{y}$  match good and all points are below the set threshold of 0.18m. The dark blue points are the set waypoints to be reached, while the orange dots are the actual robot position at that waypoint.

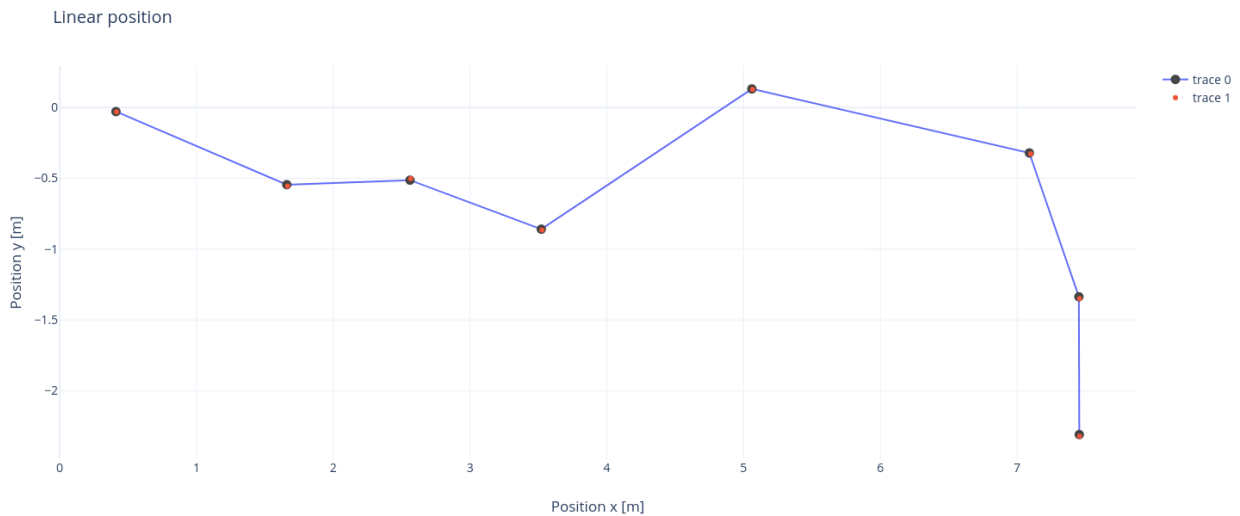


Figure 5.5: Linear position result

In comparison, figure 5.6 and 5.7 shows the correlation between  $z$ ,  $w$  and  $\hat{z}$ ,  $\hat{w}$  to also be under the threshold value of 0.8 radians. The orange lines is the set orientation to be executed. The blue lines shows what the robot's actual orientation is at each waypoint.

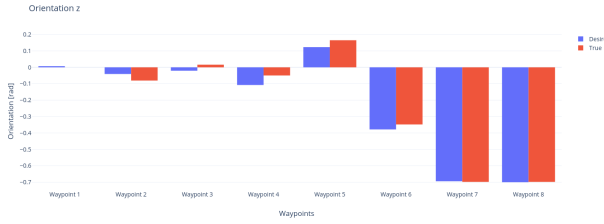


Figure 5.6: Orientation z result

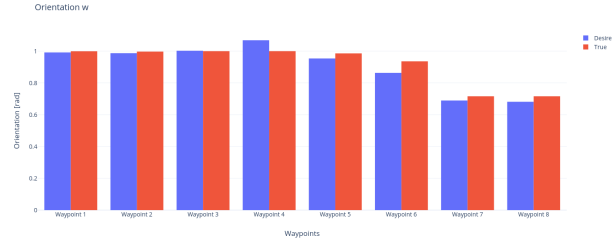


Figure 5.7: Orientation w result

### Image processing

Under the mission there were three image processing tasks conducted. The first and second tasks at waypoint three and the third task on the last waypoint. Table 5.7 shows that all operations were successful in the mission.

Predicted	True
DPG100-56	DPG100-56
0.00	0.00
PSV100-47	PSV100-47

Table 5.7: OCR processing results

### Detection and classification

All detections and classifications that were done by the CNN model came back successful. Table 5.8 shows the predicted versus the true class for all of the inspected equipment. A more detailed result describing the accuracy of the CNN can be found in section 5.2.9.

Predicted	Truth
Fire extinguisher	Fire extinguisher
Gauge	Gauge
Exit Sign	Exit sign
Closed valve	Closed valve

Table 5.8: Detection and classification results

Figures 5.8, 5.9, 5.10 and 5.11 shows the detections and classifications that were done under mission.



Figure 5.8: Fire extinguisher result

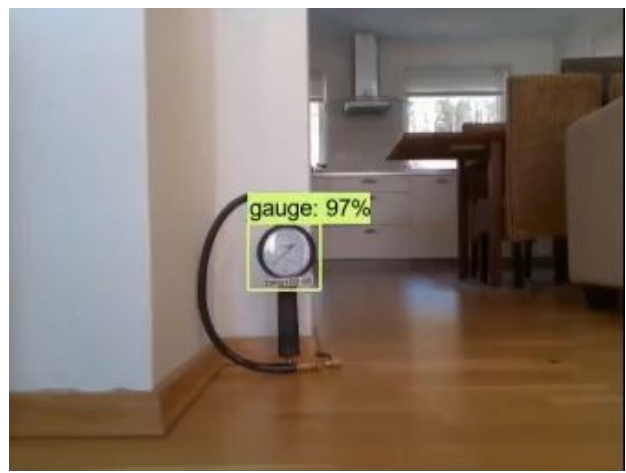


Figure 5.9: Gauge result



Figure 5.10: Exit sign result

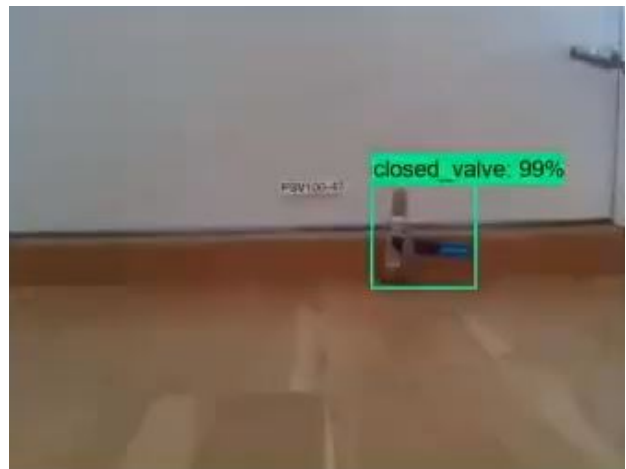


Figure 5.11: Closed valve result

## 5.2.7 Graphical user interface

As seen in the figures below, all the windows have a top status bar. This is to display connection status while the operator can easily change to another view or emergency stop the robot at any time.

### 5.2.7.1 Manual interface and FPV interface

The view in figure 5.12 and 5.13 gives the operator a simple reference to control the robot with the controller. Or simply adjust the robot behaviour by adjusting the sliders or button combinations on the interface.

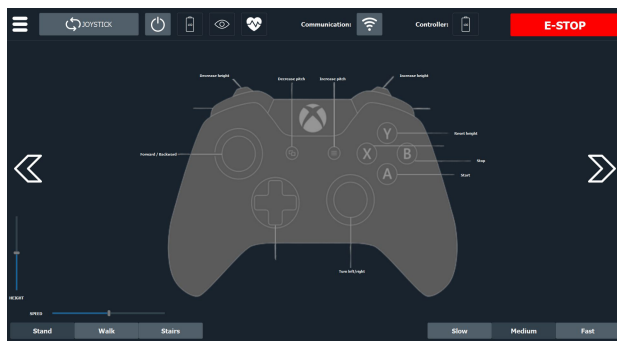


Figure 5.12: GUI Manual result

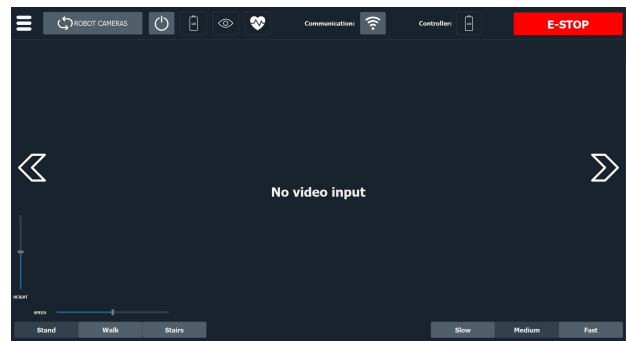


Figure 5.13: GUI FPV result

### 5.2.7.2 Inspection interface

Figure 5.14 is the main interface that have multiple widgets embedded in the main frame. In the top left corner there is a live feed from one of the cameras mounted on the robot. The operator have the ability to change to different cameras at any time. The different camera presentations are the following: *Color, Depth, Fisheye, Infrared*.

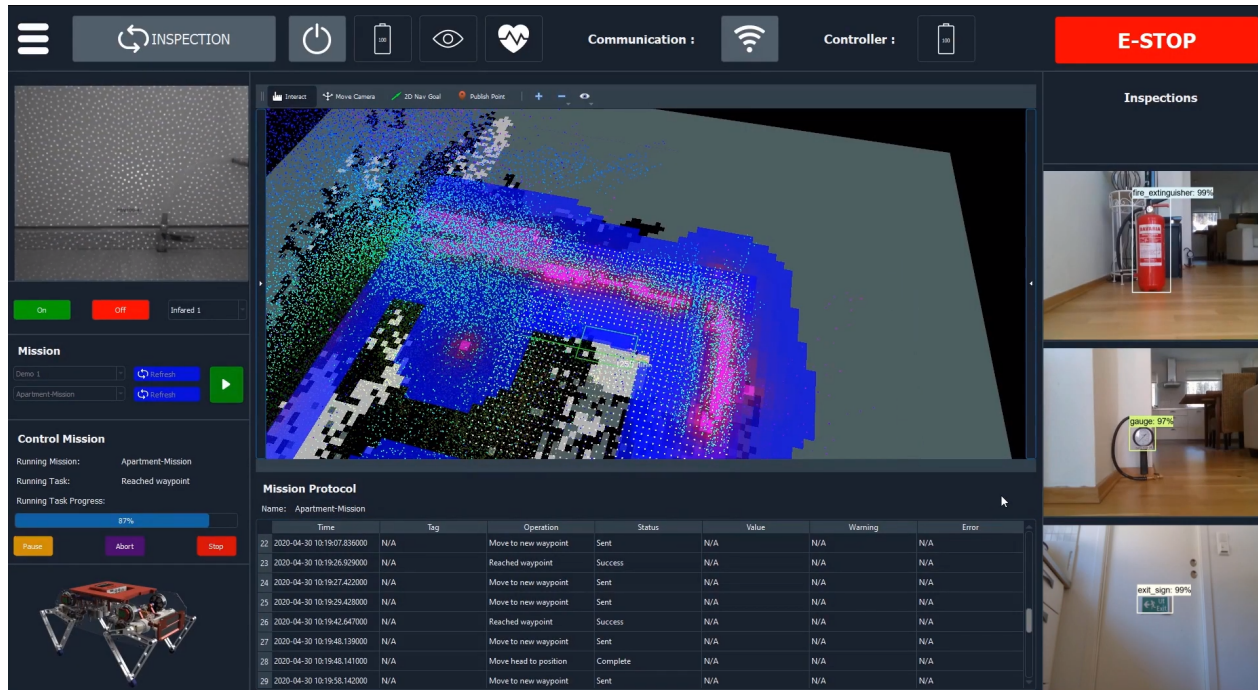


Figure 5.14: GUI Inspection result

The lower section to the left is where the operator can load missions and run them. *Control Mission* widget displays the current mission running and at what task it's currently at. A progress-bar updates in real time based on where the robot is on it's mission. At last the operator have the ability to either pause, abort or stop the mission.

The center of the view contains a live 3D interactive view of the robots exploration in the environment. The following is displayed: *Point cloud, Global costmap, Local costmap, Occupancy grid, Robot position, Global path, Local path, Goal, Robot model*. Under this view the action log is displayed. Based on the mission protocol the log is updated with the robots tasks and progress. At the same time information about the inspected equipment will appear. This can be attributes

as *tag, value, warning and alarm*.

At last the whole right section is made to display the output from the CNN model and is updated continuously as new images is fetched from the API.

## **5.2.8 Software solution**

### **5.2.8.1 Remote computation**

It takes approximately one second for the remote server to receive a full image, make the CNN detect and classify the different objects in the image and post the picture and details to the API.

### 5.2.8.2 ROS

In table 5.9 is the result of the different topics listed and their publish frequencies. They all differ in frequency, and the reason for that is either the need for data is low or computational heavy. Topics like: */joy*, */in\_position*, */move\_base\_simple/goal* and */goal\_reached* is event based and will only publish based upon request.

Topic	Frequency	Info
<i>/move_base/TrajectoryPlannerROS/local_plan</i>	20 Hz	Local path plan
<i>/move_base/TrajectoryPlannerROS/global_plan</i>	1 Hz	Global path plan
<i>/rtabmap/grid_map</i>	30 Hz	Grid map
<i>/rtabmap/scan_map</i>	10 Hz	Point cloud from depth camera
<i>/move_base_simple/goal</i>	1 Hz	Waypoint
<i>/move_base/TrajectoryPlannerROS/cost_cloud</i>	5 Hz	Cost cloud around robot
<i>/move_base/global_costmap/costmap</i>	1 Hz	Global costmap
<i>/move_base/local_costmap/footprint</i>	10 Hz	Robot occupation
<i>/move_base/global_costmap/footprint</i>	10 Hz	Robot occupation
<i>/move_base/local_costmap/costmap</i>	10 Hz	Local costmap
<i>/t265/odom/sample</i>	200 Hz	Robot pose
<i>/d435/infra1/image_rect_raw</i>	15 Hz	Infrared image
<i>/d435/rgb/image_rect_raw</i>	15 Hz	Color image
<i>/d435/depth/image_rect_raw</i>	15 Hz	Depth image
<i>/t265/fisheye1/image_rect_raw</i>	15 Hz	Fisheye image
<i>/cmd_vel</i>	20 Hz	Desired robot motion
<i>/joy</i>	150 Hz	Joystick outputs
<i>/in_position</i>	1 Hz	Robothead in position
<i>/goal_reached</i>	1 Hz	Robot on waypoint

Table 5.9: ROS topic publish frequency

### 5.2.8.3 Embedded device

In table 5.10 the loop times of the MCU can be seen. The main case is measured between the start of the state machine and at the end of it. Code that is always executed is the reading of the button states and the Xbox controller button states. The walk case is measured when the program enters the walk case and is stopped when it's exiting the case. The walk case only runs one time every one millisecond. One thing to mention is that this is without the input from ROS, but the robot behaved in the same way when it got the input so the times should be very similar. With this short loop time the program is always able to handle incoming data from serial, and detecting button presses with fast response.

Case	Time [us]
Main	1
Walk	455
Both	456

Table 5.10: Loop times on the MCU



### 5.2.9 Convolution neural network

The CNN developed was trained on a large variety of self taken images. This resulted in a total of 376 labeled objects for training, and 150 for testing. As figure 5.15 shows, the data was sufficient enough to train the model to get a very low loss on the training data within 40000 iterations. This proves the robustness of the model, and it can with very little training data be retrained to any equipment in a few hours.

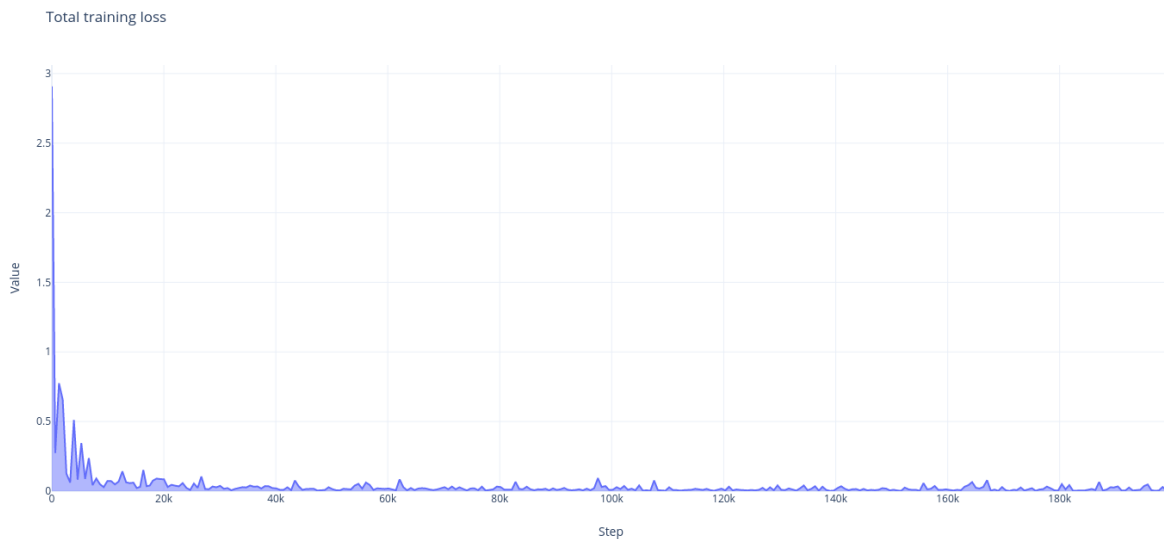


Figure 5.15: Total training loss

#### 5.2.9.1 Evaluation metrics

The CNN was tested in order to verify its metrics. The CNN model was tested with a number of never before seen sampled images. By knowing the correct labels in each image, precision, sensitivity and accuracy could be calculated and a confusion matrix generated.

### 5.2.9.2 Confusion matrix

The confusion matrix figure 5.16, reveals the accuracy of the CNN model. As the confusion matrix shows, 98.67% of the objects in the images were detected and classified correct.

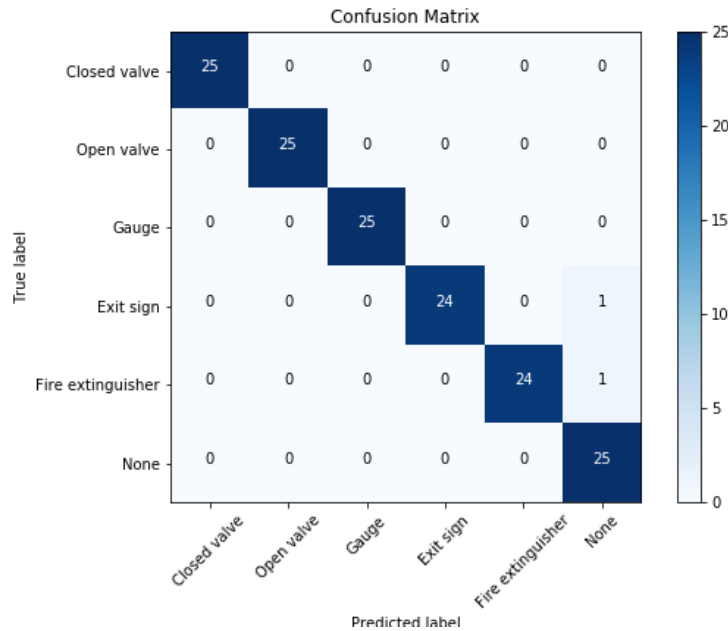


Figure 5.16: Confusion matrix

Figure 5.17 shows the precision of the CNN model. The evaluation shows that the precision is 100% on all of the equipment. The None precision is on 96%, and from figure 5.16 can it be seen that there is 2 predicted *None*, when the true label was *fire extinguisher* and *exit sign*.

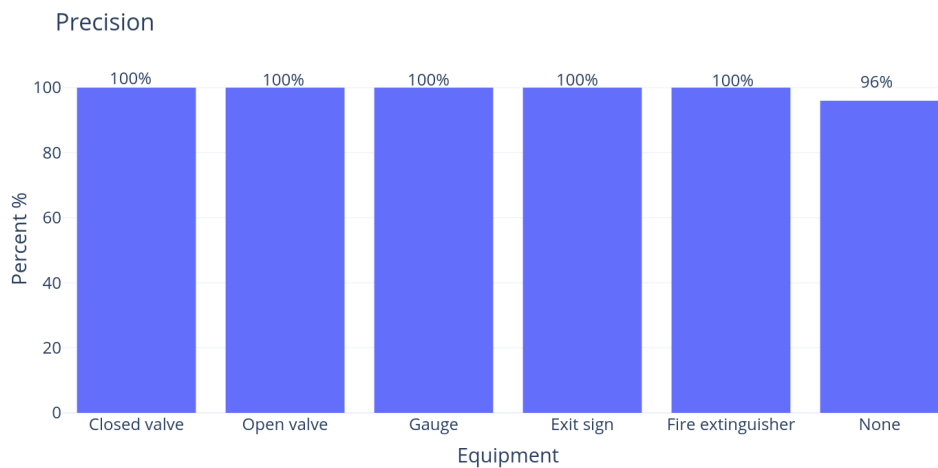


Figure 5.17: CNN Precision

Figure 5.18 shows a lower result than 5.17 did with precision. The reason for this is that sensitivity punish false negatives which results in a lower percentage on *Exit sign*, *Fire extinguisher* and *None*.

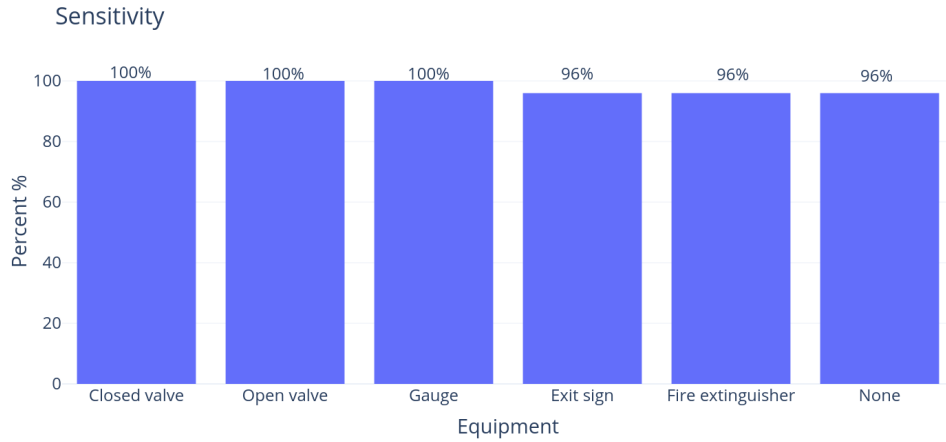


Figure 5.18: CNN Sensitivity

Figure 5.19 shows the accuracy of the model. It can be seen that the CNN model can predict each equipment with high confident.

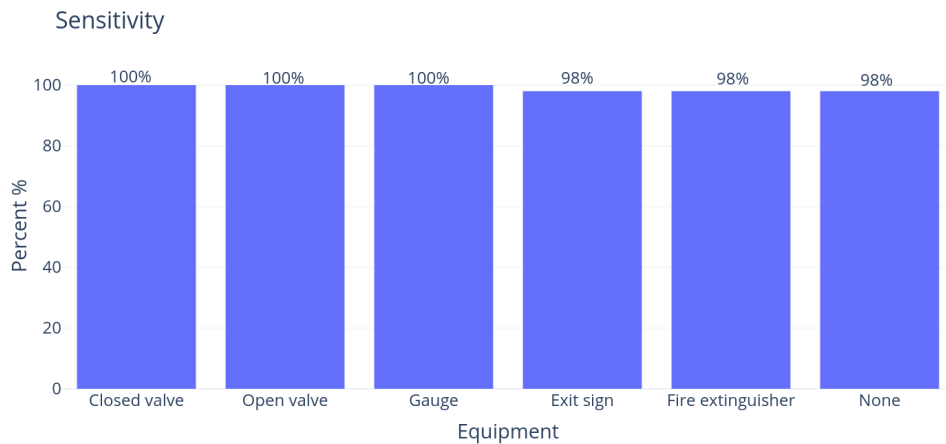


Figure 5.19: CNN Accuracy

Figure 5.20 and 5.21 is two examples from the images that were in the evaluation set.

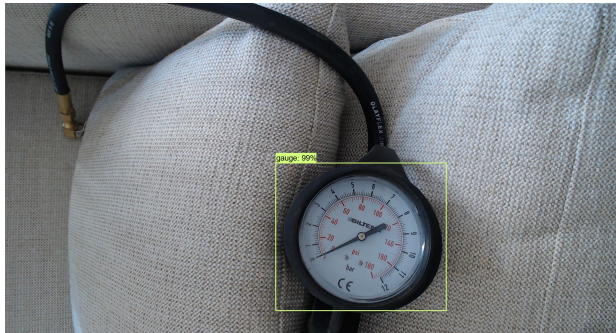


Figure 5.20: CNN Result Example 1

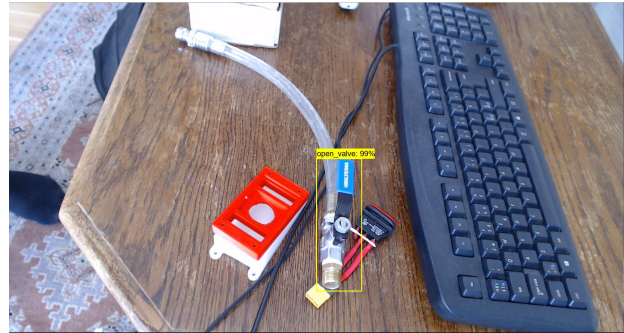


Figure 5.21: CNN Result Example 2

# Chapter 6

## Discussion

In this chapter we discuss the results obtained in the previous chapter and the reliability of the data collected. Including our experiences during the project.

### 6.1 Motion

During the development of the system responsible for motion, it was clear that locomotion with different speeds and through corners was in fact possible. The fact that the leg motion will lag on higher speeds does not impact the results since the issue will never occur with the intended usage, where slow to moderate speeds is needed. On the other hand, it is still worth mentioning if future needs or usage should change.

When it comes to the different gait result among different flooring, this would most likely not be an issue if the robot had shoulder joints, as this would result in each of the legs being able to stand in the same place while rotating around the robot's z-axis (yaw motion). This would improve its performance on different types of flooring and ground surfaces. As seen with other similar robots, many include the shoulder joint resulting in improved mobility. However, this was not deemed important enough as the project budget did not allow for it. On the other hand, the results regarding locomotion points to that locomotion is possible on the ground surfaces that was tried, while some being better than others.

Since reading the position and current of the motors driven by the ODrive motor drives was not possible, there was no chance of implementing features such as detection of leg contact with ground. By not being able to sense when leg hit the ground it became impossible to make adaptive motions. This would have been most favourable to have implemented since it would mean the robot would adapt to different terrains.

## 6.2 Robot

To design and build a robot for this applications, several elements need to be taken into account. The most general specifications are mentioned beneath.

### 6.2.1 Robot type selection

Among legged robots, the quadruped robots have good mobility and stability of locomotion, while the typical biped robots lack the locomotion stability. From a system and control view, the quadruped robot is a good choice since it naturally offers more stability than bipeds. In addition, four legs are less complicated to construct and maintain than if a robot with additional legs, like a hexaped, were chosen. Quadruped robots are more versatile than wheeled and tracked robots, and more stable than biped robots [102].

The necessities of mobile robots for complex and dangerous environments have initiated the development of dynamic quadruped machines, which exploit the potential advantages of the legged locomotion and enhanced mobility in unstructured terrains. Such a versatile system with high maneuverability should be labor efficient, cost-effective, and indispensable in industries. The shape also helps the robot to walk in a comparatively narrow region as it possesses smaller footprint [71].

Considering the stability and dynamic movement facility of the structure, the development of a quadruped robot is selected for this thesis.

Even though better quality quadruped robots are available, the costs of those are beyond reach

for general researchers. The designed robot will be very cheap compared to the similar robots available. This robot can be easily constructed from materials available locally.

Although wheeled and tracked robots can work well in even surface, most of them cannot work in uneven surface. In other words, only a portion of the earth's landmass is accessible for the existing ground robots. Compared with the wheeled and tracked robots, legged robots have the potential to walk in a much wider variety of terrains [74].

### **6.2.2 Robot selection basis**

The robot type selected for the purpose of this project is a quadruped robot initially developed by the Robotics division at Stanford. It met all the criteria set regarding using brushless DC motors, encoders and most importantly the low cost. Their actuator design offered a compact footprint, robust mechanics with little backlash. Furthermore, many of the parts needed for the design were easily accessible, or easy to manufacture.

The actuator design and parts of the robot body were the key elements taken from their robot as it would be time consuming to develop and would need extensive testing and perhaps several prototypes to get a comparable result. In addition, building a robot is not the main purpose of this project, whereas controlling and using the robot are. All about their project can be found here [Github](#) [29] or here [Report](#) [30].

### **6.2.3 Legs**

The first version of the legs were made out of nylon from 3D-printing. This worked perfect for testing the movement of the legs with little to none load. After the testing part was done and load were applied on the foot, the nylon proved not to be strong enough. Therefore a revision was done and the legs were made out of aluminium instead.

### 6.3 Physical structure and design

In the beginning of the project, an extensive amount of time was used for designing the parts of the robot in 3D and finding suited components. This paid off when assembling the robot. Most of the parts and components were installed without any problems and no major changes were needed. The finished prototype is robust with its motion being sturdy and reliable during operation.

### 6.4 Robot accuracy

Regarding the robots accuracy it is clear that it can be improved. Adding filters with thresholds at around  $45^\circ$  should not be necessary but this comes back to the holonomic behaviour described in 5.2.3.2. To overcome this problem, the locomotion could be further developed. On the other hand, if there is no way to get better performance by further software changes, it could be considered to alter the leg design. Furthermore, since all the testing of the locomotion is done physically it proved to be very time consuming to evaluate all the possible ways of walking. However, this could be done with reinforced learning running a physics engine simulating the robot reaction to a vast number of different options and then applying it to the physical model afterwards.



### 6.4.1 Gait controller

The gait controller implemented in the robot works to prove the concept of this thesis, but have improvement potential. An improved gait controller can use many different gaits that changes depending on desired velocity and angle of the robot. Figure 6.1 illustrates one optimized gait controller. This optimized controller have one additional gait called walk gait, and can be seen in the bottom right corner. The new gait works by allowing only one leg to move at the time, making the robot have three legs in the ground at all time. This makes the robot much more stable at slower speed. In the top right corner one can see the implemented trot gait where two legs moves together. This is just two of many gaits. More examples of gaits that can be implemented are a gait for running or a gait for traversing stairs.

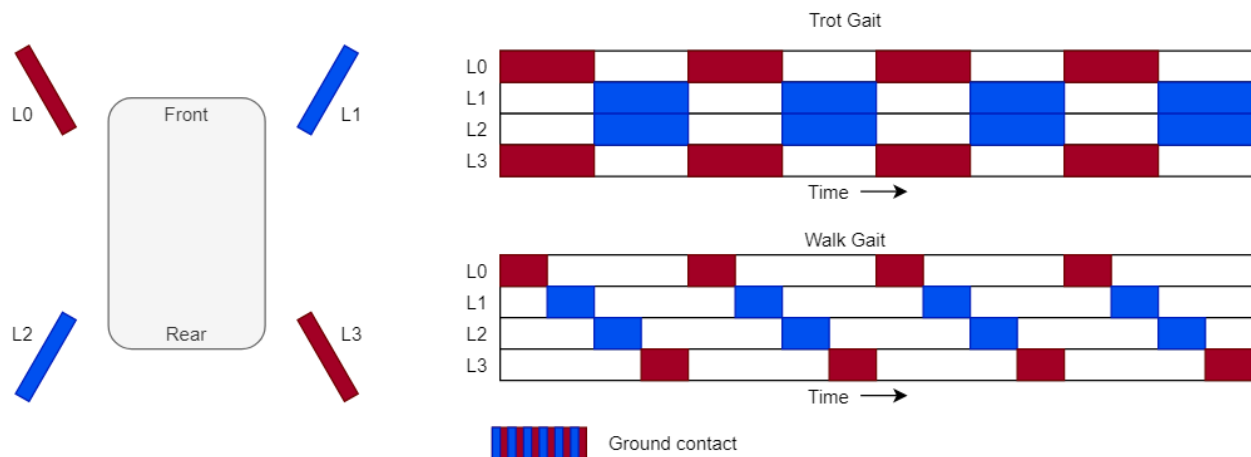


Figure 6.1: Optimized gait controller

## 6.5 Communication protocol

Serial is the protocol for communication between the controllers locally. With serial there is no need for a bus, and a point to point connection is easy to maintain and handle. By using USB connection from the Jetson Nano to the other controllers the cables are by standard, shielded from start to end. This eliminates some noise which was wanted because the Serial is not noise resistant, and the Odrive brushless motor driver emits a small amount of electromagnetic noise to the system. As the Jetson Nano communicates with each device on separate lines, it also eliminates collision and cross talk between the devices.

TCP/IP is the protocol for communication between the controllers remotely. This was chosen because it's a standardized communication protocol, allows cross-platform and is a scalable, client-server architecture. This allows networks to be added without disrupting the current services. UDP was also a option for remotely communication cause of its speed, but was not used for the lack of reliable transfers. Which is important for a safety system.

The bluetooth protocol was chosen for the communication between the Xbox controller and robot. The reason for bluetooth is the simple wireless connection witch is strong on long ranges and does not require a cable between the devices.

One major change that took place in the middle of the project phase was the change to ROS over ZMQ. This led to a major turn around in the software architecture. From the start was ZMQ the chosen communication framework for internal processes and between the different systems. As more of the systems came together ZMQ did not scale well, and Jetson Nano was not able to send out data reliably. With the change to ROS, many of the problems got resolved, and it opened up multiple other implementations that were available for the ROS ecosystem.

## 6.6 Navigation

The Jetson Nano was one of the bottle necks to the thesis. It led to changes in the overall structure of where the different system had to be run to sufficiently perform the inspections without any faults. This led the choice to make the navigation system to run on the computer running the GUI. This is by no means necessary, but the performance boost gained was tremendous. This might not be that lucrative, but as the robot itself depend to be on the same network as GUI to operate anyway is this something that is not noticeable for anyone outside. However, if the navigation system gets disconnected during mission will the robot not function properly, and the mission has to be aborted to not cause any damage to the robot or others.

## 6.7 Remote computing

The need for more computing power than the Jetson could offer was known from the beginning because of similar work done before. In a better scenario the Jetson would be upgraded to a more powerful version, but because of the limitation in the budget was the use of a remote computer for additional power the way to go. The overall solution worked without problem and as intended, but there are weaknesses to the reliability of the robot. There are several problems that can occur that makes the detection and classifying unable to work as intended. The biggest problem that can occur is the loss of the connection to the server. This was never encountered, but can be completely eliminated with upgrading the Jetson.

## 6.8 Software

As mentioned earlier in the thesis there have been a major alterations to the software side with the change from ZMQ to ROS. Half of the thesis period was conducted to get ZMQ to work both as internal and external communication protocol. A lot of work had to be deprecated due to this change. Despite that, ROS brings an ecosystem with applications that is suited for robots and with a standardized messaging system that made it easy to extend the applications to the purposes of this thesis. So in the change was a big setback at first, but it turned out to increase the performance and add robust functionalities that otherwise would not have been implemented.

## 6.9 Image processing

Extracting the value where the dial of the gauge is done with robustness in mind. This led to a combined implementation of image processing and a ground truth of each gauge. The ground truth is set during creation of new mission plans, and the work of the operator has to be trusted. This can be a source of error, as the operator can misinterpret the values and set them wrong. On the other side, this implementation does not require the robot to see any value on the gauge, and can be based on the dial and precisely tell the value indirectly.

## 6.10 Optical Character Recognition

The OCR implemented need optimal conditions to function properly. The text has to be very clear and straight to be able to read it. Much of this is done in the preprocessing steps before sending it to the OCR task. Skew-correction and erosion benefited the outcome most, and a better camera is needed to increase the success rate for extracting the tag of the equipment.

## 6.11 Convolution neural network

### 6.11.1 Dataset

It is worth mentioning that the images in the dataset is taken inside in a static environment. This means that the source of light is not dynamic as it would be outside. Including lack of external noise as rain and dust. However, for this thesis the acquired dataset is sufficient enough to prove that the model can generalize and differentiate between the given equipment classes at a high precision and accuracy.

### 6.11.2 Model

In this thesis, only one CNN model was trained. The reason for this is how good the evaluation turned out. Both in accuracy and inference time the Faster R-CNN delivered to the system requirements. However it is known that architectures like SSD can challenge Faster R-CNN inference speeds and deliver close to same accuracy. But as mentioned earlier, the required inference time for sufficient updates is high , meaning it would not benefit the system to implement a faster CNN architecture.

### 6.11.3 Evaluation

Evaluation of the model is done by a random cut in the dataset. This means that the images in the evaluation set may contain bad images that negatively affect the result. At the same time, the evaluation set is small, which means that each prediction counts for a large percentage in the evaluation metrics. This has in this instance proven to be positive for the metrics, but with a

evaluation set with a bigger range of images this might alter the results that is presented in this thesis.

## 6.12 Improvements for the future

The following points are suggestions for further development of the Robot.

- **Hardware**

- Upgrade from Jetson Nano to Jetson AXG Xavier.
- One more joint per leg for better mobility.
- Increase the battery capacity for longer operation.
- Add battery management.
- Use absolute encoders to eliminate any need of calibration at every start-up.
- Add a LIDAR for 360° object avoidance.
- Equip more sensors for more inspection tasks.

- **Software**

- Move the CNN to local, too eliminate external disruptions.
- Train the CNN model on even more equipment.
- Get current feedback from motors
- Integrate more forms of motion.
- Integrate a motion planner go up stairs.

## 6.13 Experiences

### 6.13.1 Division of labor

All group members have different job and field experience. The different backgrounds have been used in the work distribution to take advantage of each members expertise. In a multidisciplinary project, as this turned out to be, with planning, constructing, and building the robot, led to self-learning and everyone developing new skills during the project thesis.

# Chapter 7

## Conclusion

The content of this thesis concerns the development of a quadruped robot prototype for autonomous inspections in the process industry with use of neural networks. This includes modelling and building a prototype that can equip sensors for collecting data. Including to that has multiple software systems been developed to handle; navigation, image processing, inverse kinematics, artificial intelligence and graphical user interfaces to aid in the inspection missions.

The results proves that the prototype is successful in its tasks. The main goal was to make the robot successfully go trough a inspection mission autonomously and provide real time report-statuses back to the GUI where an operator could observe.

The prototype has accomplished many of the sub goals that were stated in the preliminary report. Utilizing concurrent processes and artificial intelligence, the robot manages to detect and classify all the equipment in the given task. With use of image processing was it able to extract tags and values from gauges with a high accuracy. In addition the robot is dynamic with self stabilization and have a inverse kinematic model of the legs to move efficiently around.

This mentioned, there are many improvements to be made on both the design and software that can increase the reliability and performance of the system. All the improvements are listed in section [6.12](#).

# Bibliography

- [1] AkerBP. Exploring the potential of robotics in the oil and gas industry, 02 2020. URL [www.akerbp.com/exploring-the-potential-of-robotics-in-the-oil-and-gas-industry/](http://www.akerbp.com/exploring-the-potential-of-robotics-in-the-oil-and-gas-industry/).
- [2] Anton Konushin Alexey Artemov. Region-based convolutional neural network, 05 2020. URL <https://www.coursera.org/lecture/deep-learning-in-computer-vision/region-based-convolutional-neural-network-yU6QP>.
- [3] ams. Encoder as5047p-ts ek ab, 04 2020. URL [https://www.digikey.no/product-detail/no/ams/AS5047P-TS\\_EK\\_AB/AS5047P-TS\\_EK\\_AB-ND/5452344](https://www.digikey.no/product-detail/no/ams/AS5047P-TS_EK_AB/AS5047P-TS_EK_AB-ND/5452344).
- [4] Autodesk. Fusion 360, 1 2019. URL <https://www.autodesk.com/products/fusion-360/students-teachers-educators>.
- [5] Jason Brownlee. A gentle introduction to computer vision, 07 2019. URL <https://machinelearningmastery.com/what-is-computer-vision/>.
- [6] Creative Commons. Attribution 4.0 international, 05 2020. URL <https://creativecommons.org/licenses/by/4.0/>.
- [7] Riverbank Computing. Python tesseract, 04 2020. URL <https://www.riverbankcomputing.com/software/pyqt/>.
- [8] Ken Conley. rospy, 04 2020. URL <http://wiki.ros.org/rospy>.
- [9] S. Thrun D. Fox, W. Burgard. The dynamic window approach to collision avoidance, 03 1997. URL <https://ieeexplore.ieee.org/document/580977>.



- [10] Josh Faust Dave Hershberger, David Gossow. rviz, 04 2020. URL <http://wiki.ros.org/rviz>.
- [11] DeepAI. Feature extraction, 03 2020. URL <https://deepai.org/machine-learning-glossary-and-terms/feature-extraction>.
- [12] Draw,io. Draw.io, 1 2019. URL <https://about.draw.io/>.
- [13] Ebay. Battery, 05 2020. URL <https://www.ebay.com/itm/RC-Turnigy-Graphene-Panther-1000mAh-6S-75C-Battery-Pack-w-XT60-/232988935523>.
- [14] ecorias. Jetson nano case, 05 2019. URL <https://www.thingiverse.com/thing:3603594>.
- [15] Nick Efford. Digital image processing: A practical introduction using java, 03 2020. URL <https://www.cs.auckland.ac.nz/courses/compsci773s1c/lectures/ImageProcessing-html/topic4.htm>.
- [16] Eric Perko Eitan Marder-Eppstein. base-local-planner, 05 2020. URL [http://wiki.ros.org/base\\_local\\_planner?distro=kinetic](http://wiki.ros.org/base_local_planner?distro=kinetic).
- [17] Encoder. What is an encoder?, 02 2016. URL <http://encoder.com/blog/company-news/what-is-an-encoder/>.
- [18] Hamid Rezaatofghi et al. Generalized intersection over union: A metric and a loss for bounding box regression, 04 2020. URL <https://arxiv.org/abs/1902.09630>.
- [19] Rohith Gandhi. Support vector machine — introduction to machine learning algorithms, 06 2018. URL <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>.
- [20] Ross Girshick. Fast r-cnn, 04 2020. URL <https://github.com/rbgirshick>.
- [21] Ross Girshick. Rich feature hierarchies for accurate object detection and semantic segmentation, 04 2020. URL <http://arxiv.org/abs/1311.2524>.

- [22] Alexandre Gonfalonieri. How to build a data set for your machine learning project, 03 2020. URL <https://towardsdatascience.com/how-to-build-a-data-set-for-your-machine-learning-project-5b3b871881ac>.
- [23] David Gray. What are point clouds? 5 easy facts that explain point clouds, 03 2020. URL <https://info.vercator.com/blog/what-are-point-clouds-5-easy-facts-that-explain-point-clouds>.
- [24] Samuel Hoffstaetter. Python tesseract, 04 2020. URL <https://github.com/madmaze/pytesseract>.
- [25] Intel. Depth camera d435, 02 2020. URL <https://www.intelrealsense.com/depth-camera-d435/>.
- [26] Intel. Tracking camera t265, 02 2020. URL <https://www.intelrealsense.com/tracking-camera-t265/>.
- [27] JIMBLOM. Serial communication, 11 2019. URL <https://learn.sparkfun.com/tutorials/serial-communication/all>.
- [28] Tanay Karmarkar. Region proposal network (rpn) — backbone of faster r-cnn, 08 2018. URL <https://medium.com/egen/region-proposal-network-rpn-backbone-of-faster-r-cnn-4a744a38d7f9>.
- [29] Nathan Kau, Aaron Schultz, Natalie Ferrante, and Patrick Slade. Stanford doggo: An open-source, quasi-direct-drive quadruped github, 2019. URL <https://github.com/Nate711/StanfordDoggoProject>.
- [30] Nathan Kau, Aaron Schultz, Natalie Ferrante, and Patrick Slade. Stanford doggo: An open-source, quasi-direct-drive quadruped, 2019. URL <https://arxiv.org/abs/1905.04254>.
- [31] Keras. Keras documentation, 11 2019. URL <https://keras.io/>.

- [32] Dhairya Kumar. Introduction to data preprocessing in machine learning, 12 2018. URL [www.towardsdatascience.com/introduction-to-data-preprocessing-in-machine-learning-a9fa83a5dc9d](http://www.towardsdatascience.com/introduction-to-data-preprocessing-in-machine-learning-a9fa83a5dc9d).
- [33] Mathieu Labbé. Rtab-map, 04 2020. URL <http://introlab.github.io/rtabmap/>.
- [34] Weber Perdigão Macedo Marcelo Becker, Carolina Meirelles Dantas. Obstacle avoidance procedure for mobile robots, 03 2020. URL [https://web.archive.org/web/20131126163902/http://www.abcm.org.br/pt/wp-content/symposium-series/SSM\\_Vol2/Section\\_IV\\_Mobile\\_Robots/SSM2\\_IV\\_05.pdf](https://web.archive.org/web/20131126163902/http://www.abcm.org.br/pt/wp-content/symposium-series/SSM_Vol2/Section_IV_Mobile_Robots/SSM2_IV_05.pdf).
- [35] Eitan Marder-Eppstein. costmap-2d, 05 2020. URL [http://library.isr.ist.utl.pt/docs/roswiki/costmap\\_2d.html](http://library.isr.ist.utl.pt/docs/roswiki/costmap_2d.html).
- [36] Matplotlib. Matplotlib: Python plotting, 11 2019. URL <https://matplotlib.org/>.
- [37] Steve Meyer. Linear motion, 4 2009. URL <https://www.therobotreport.com/linear-motion/>.
- [38] Adam Stambler Michael Ferguson. roserial-arduino, 04 2020. URL [http://wiki.ros.org/roserial\\_arduino](http://wiki.ros.org/roserial_arduino).
- [39] Microsoft. Visual studio code, 1 2019. URL <https://code.visualstudio.com/>.
- [40] Microsoft. Xbox controller, 04 2020. URL <https://www.xbox.com/nb-NO/accessories/controllers/xbox-wireless-controller>.
- [41] Steve Mutuvi. Introduction to machine learning model evaluation, 03 2020. URL <https://heartbeat.fritz.ai/introduction-to-machine-learning-model-evaluation-fa859e1b2d7f>.
- [42] Chris Nicholson. A beginner's guide to neural networks and deep learning, 1 1990. URL <https://skymind.ai/wiki/neural-network>.
- [43] Nicomsoft. Optical character recognition (ocr) – how it works, 03 2020. URL <https://www.nicomsoft.com/optical-character-recognition-ocr-how-it-works/>.

- [44] Numpy. Numpy, 11 2019. URL <https://numpy.org/>.
- [45] Nvidia. Jetson nano, 04 2020. URL <https://developer.nvidia.com/embedded/jetson-nano-developer-kit>.
- [46] Odrive Robotics. Odrive tool, 1 2019. URL <https://docs.odriverobotics.com/odrivetool.html>.
- [47] OpenCV. Opencv, 11 2019. URL <https://opencv.org/>.
- [48] Oracle. What is a database, 05 2020. URL <https://www.oracle.com/database/what-is-database.html>.
- [49] Overleaf. Overleaf, 1 2019. URL <https://overleaf.com>.
- [50] Ravindra Parmar. Training deep neural networks, 03 2020. URL <https://towardsdatascience.com/training-deep-neural-networks-9fdb1964b964>.
- [51] pc control. Incremental encoders, 01 2008. URL [https://www.pc-control.co.uk/incremental\\_encoders.htm](https://www.pc-control.co.uk/incremental_encoders.htm).
- [52] PCSchematic. Pcschematic automation, 1 2019. URL <https://www.pcschematic.com/en/electrical-cad-design-drawing-software/electrical-cad/electrical-cad-user-perspective.htm>.
- [53] Ivan Petrovic. An integrated approach to real-time mobile robot control in partially known indoor environments, 01 2005. URL [https://www.researchgate.net/figure/Dynamic-Window-trajectories-and-effective-path-The-maximum-velocity-case\\_fig2\\_37441306](https://www.researchgate.net/figure/Dynamic-Window-trajectories-and-effective-path-The-maximum-velocity-case_fig2_37441306).
- [54] PJRC. Teensy 4.0, 04 2020. URL <https://www.pjrc.com/store/teensy40.html>.
- [55] PlatformIO. Platformio, 1 2020. URL <https://platformio.org/>.
- [56] The Pallet Projects. The pallet projects, 11 2019. URL <https://www.palletsprojects.com/p/flask/>.

- [57] Shaoqing Ren. Faster r-cnn: Towards real-time object detection with region proposal networks, 04 2020. URL <http://image-net.org/challenges/LSVRC/2015/results>.
- [58] riptutorial. Activation function, 04 2020. URL <https://riptutorial.com/machine-learning/example/31624/activation-functions>.
- [59] Ashley Walker Robert Fisher, Simon Perkins and Erik Wolfart. Morphology, 03 2020. URL <https://homepages.inf.ed.ac.uk/rbf/HIPR2/morops.htm>.
- [60] Linda G.Shapiro Robert M.Haralick. Image segmentation techniques, 03 2020. URL <https://www.sciencedirect.com/science/article/abs/pii/S0734189X85901537>.
- [61] Odrive Robotics. Odrive v3.6 specs, 05 2020. URL <https://eu.odriverobotics.com/shop/odrive-v36>.
- [62] ROS. Ros, 03 2020. URL <http://wiki.ros.org/ROS/Introduction>.
- [63] ROS. Messages, 04 2020. URL <http://wiki.ros.org/Messages>.
- [64] ROS. Nodes, 04 2020. URL <http://wiki.ros.org/Nodes>.
- [65] ROS. Topics, 04 2020. URL <http://wiki.ros.org/Topics>.
- [66] Margaret Rouse. Tcp/ip (transmission control protocol/internet protocol), 10 2008. URL <https://searchnetworking.techtarget.com/definition/TCP-IP>.
- [67] Sumit Saha. A comprehensive guide to convolutional neural networks — the eli5 way, 12 2018. URL <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>.
- [68] Sam. Motor drivers vs. motor controllers, 08 2019. URL <https://core-electronics.com.au/tutorials/motor-drivers-vs-motor-controllers.html>.
- [69] Dipanjan (DJ) Sarkar. A comprehensive hands-on guide to transfer learning with real-world applications in deep learning, 11 2018. URL <https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a>.

- [70] sas. Machine learning what it is and why it matters, 11 2019. URL [sas.com/en\\_us/insights/analytics/machine-learning.html](https://sas.com/en_us/insights/analytics/machine-learning.html).
- [71] Atsushi Horigome & Gen Endo Satoshi Kitano, Shigeo Hirose. Titan-xiii: sprawling-type quadruped robot with ability of fast and energy-efficient walking, 03 2016. URL <https://robomechjournal.springeropen.com/articles/10.1186/s40648-016-0047-1>.
- [72] Davide Scaramuzza and Zichao Zhang. Visual-inertial odometry of aerial robots, 05 2020. URL <https://arxiv.org/pdf/1906.03289.pdf>.
- [73] Abhishek Sharma. Confusion matrix in machine learning, 04 2020. URL <https://www.geeksforgeeks.org/confusion-matrix-machine-learning/>.
- [74] JiuHong Ruan Xuewen Rong SYibin Li, Bin Li. Research of mammal bionic quadruped robots: A review, 09 2011. URL [https://ieeexplore.ieee.org/abstract/document/6070476?fbclid=IwAR19MMDVf3F3GafgcxGf1uh-cPahf\\_7A121vQEMA462R3P0Zdj1upH1sbbc](https://ieeexplore.ieee.org/abstract/document/6070476?fbclid=IwAR19MMDVf3F3GafgcxGf1uh-cPahf_7A121vQEMA462R3P0Zdj1upH1sbbc).
- [75] T-motor. Mn5212 kv340, 04 2020. URL <http://store-en.tmotor.com/goods.php?id=377>.
- [76] Power Technology. Anybotics and tennet test world's first offshore autonomous robot, 02 2020. URL <https://www.power-technology.com/news/tennet-offshore-autonomous-robot/>.
- [77] TechTerms. Ip, 02 2020. URL <https://techterms.com/definition/ip>.
- [78] Tensorflow. Tensorflow, 11 2019. URL <https://www.tensorflow.org/>.
- [79] Sebastian Thrun and Arno Bucken. Integrating grid-based and topological maps for mobile robot navigation, 05 2020. URL [https://www.ri.cmu.edu/pub\\_files/pub1/thrun\\_sebastian\\_1996\\_8/thrun\\_sebastian\\_1996\\_8.pdf](https://www.ri.cmu.edu/pub_files/pub1/thrun_sebastian_1996_8/thrun_sebastian_1996_8.pdf).
- [80] Ultimaker. Cura, 1 2019. URL <https://ultimaker.com/software/ultimaker-cura>.

- [81] Scott E Umbaugh. Digital image processing and analysis: Human and computer vision applications with cviptools, second edition, 02 2020. URL <https://www.amazon.com/Digital-Image-Processing-Analysis-Applications/dp/143980205X>.
- [82] Michael Vetter. Digital terrain models from airborne laser scanning for the automatic extraction of natural and anthropogenic linear structures in: Geomorphological mapping: a professional handbook of techniques and applications, 03 2020. URL [https://www.researchgate.net/figure/Mathematical-morphology-where-opening-is-the-combination-of-dilation-followed-by-erosion\\_fig2\\_233818899](https://www.researchgate.net/figure/Mathematical-morphology-where-opening-is-the-combination-of-dilation-followed-by-erosion_fig2_233818899).
- [83] Oskar Weigl. Odrive v3.6, 02 2017. URL <https://github.com/madcowswe/ODriveHardware>.
- [84] Oskar Weigl. Odrivearduino, 11 2019. URL <https://github.com/madcowswe/ODrive/tree/master/Arduino/ODriveArduino>.
- [85] Oskar Weigl. Control, 02 2020. URL <https://docs.odriverobotics.com/control>.
- [86] Wikipedia. Camera, 11 2019. URL <https://en.wikipedia.org/wiki/Camera>.
- [87] Wikipedia. Cloud computing, 11 2019. URL [https://en.wikipedia.org/wiki/Cloud\\_computing](https://en.wikipedia.org/wiki/Cloud_computing).
- [88] Wikipedia. C++, 11 2019. URL <https://en.wikipedia.org/wiki/C%2B%2B>.
- [89] Wikipedia. Python (programming language), 11 2019. URL [https://en.wikipedia.org/wiki/Python\\_\(programming\\_language\)](https://en.wikipedia.org/wiki/Python_(programming_language)).
- [90] Wikipedia. Representational state transfer, 11 2019. URL [https://en.wikipedia.org/wiki/Representational\\_state\\_transfer](https://en.wikipedia.org/wiki/Representational_state_transfer).
- [91] Wikipedia. Region of interest, 11 2019. URL [https://en.wikipedia.org/wiki/Region\\_of\\_interest](https://en.wikipedia.org/wiki/Region_of_interest).
- [92] Wikipedia. Brushless dc electric motor, 03 2020. URL [https://en.wikipedia.org/wiki/Brushless\\_DC\\_electric\\_motor](https://en.wikipedia.org/wiki/Brushless_DC_electric_motor).

- [93] Wikipedia. Pid controller, 01 2020. URL [https://en.wikipedia.org/wiki/PID\\_controller](https://en.wikipedia.org/wiki/PID_controller).
- [94] Wikipedia. Sql, 04 2020. URL <https://en.wikipedia.org/wiki/SQL>.
- [95] Wikipedia. A\* search algorithm, 05 2020. URL [https://en.wikipedia.org/wiki/A\\*\\_search\\_algorithm](https://en.wikipedia.org/wiki/A*_search_algorithm).
- [96] Wikipedia. Edge detection, 03 2020. URL [https://en.wikipedia.org/wiki/Edge\\_detection](https://en.wikipedia.org/wiki/Edge_detection).
- [97] Wikipedia. Corner detection, 03 2020. URL [https://en.wikipedia.org/wiki/Corner\\_detection](https://en.wikipedia.org/wiki/Corner_detection).
- [98] Wikipedia. Blob detection, 03 2020. URL [https://en.wikipedia.org/wiki/Blob\\_detection](https://en.wikipedia.org/wiki/Blob_detection).
- [99] Wikipedia. Timing belt (camshaft), 05 2020. URL [https://en.wikipedia.org/wiki/Timing\\_belt\\_\(camshaft\)](https://en.wikipedia.org/wiki/Timing_belt_(camshaft)).
- [100] Wikibooks. Robotics/navigation/trajectory planning, 03 2020. URL [https://en.wikibooks.org/wiki/Robotics/Navigation/Trajectory\\_Planning](https://en.wikibooks.org/wiki/Robotics/Navigation/Trajectory_Planning).
- [101] Wikipedia. Quaternion, 05 2020. URL <https://en.wikipedia.org/wiki/Quaternion>.
- [102] Avis H. Cohen Yasuhiro Fukuoka, Hiroshi Kimura. Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts, 03 2003. URL [https://journals.sagepub.com/doi/abs/10.1177/0278364903022003004?casa\\_token=uXbIGQgo2ZoAAAAA%3Afa7tZGONzj5Kci\\_xVWIW0m\\_1vhd2uGe2pYhCZ8z\\_WOSYc551vwmMWRv1KRL\\_hK7LHHXzPuosQOCgbw&](https://journals.sagepub.com/doi/abs/10.1177/0278364903022003004?casa_token=uXbIGQgo2ZoAAAAA%3Afa7tZGONzj5Kci_xVWIW0m_1vhd2uGe2pYhCZ8z_WOSYc551vwmMWRv1KRL_hK7LHHXzPuosQOCgbw&).
- [103] Chen Zhou. Robot motion analysis - kinematics, 11 1999. URL <https://www2.isye.gatech.edu/~czhou/MOTION.pdf>.



# **Appendices**

# **Appendix A**

## **Reports**

### **A.1 Preproject report**



NTNU

Norwegian University of  
Science and Technology

# Autonomous inspections for process industry by a quadruped robot with use of neural networks

Bachelor Thesis: IE303612

Petter Drønner, Vegard Solheim, Magnus Øye

Januar 2020

Preliminary report

Faculty of Information Technology and Electrical Engineering

Norwegian University of Science and Technology

Supervisor 1: Ivar Blindheim

Supervisor 2: Aleksander Skrede

## **Executive summary**

This preliminary project report includes formalities regarding the distribution of work, group rules, and ambitions through the bachelor's thesis in automation. The fundamental task is to develop a prototype that is able to move through, and inspect objects in a dynamic environment. Finally, the plan is to stand with a robot capable of moving in a room both manually and autonomously. This leads the group to gain a deeper understanding of different automation techniques, and apply them practically. There are several aspects of automation that will be used in the task, such as SLAM, path planning, neural networks, which will assist in inspections, navigation and an understanding of the environment. A prototype will be built to obtain proof of concept.

# Contents

Executive summary . . . . .	i
<b>1 Introduction</b>	<b>1</b>
Terminology . . . . .	2
<b>2 Project organization</b>	<b>3</b>
2.1 Project group . . . . .	3
2.1.1 Assignments for the project group - organization . . . . .	3
2.1.2 Assignments for the project manager . . . . .	3
2.1.3 Assignments for Secretary . . . . .	4
2.1.4 Assignments for other members . . . . .	4
2.2 Management-group . . . . .	4
<b>3 Agreements</b>	<b>5</b>
3.1 Agreements with employer . . . . .	5
3.2 Workplace and resources . . . . .	5
3.3 Group norms - rules of cooperation - attitudes . . . . .	5
3.3.1 Terms and Conditions . . . . .	5
3.3.2 Acceptance . . . . .	6
<b>4 Project description</b>	<b>7</b>
4.1 Thesis problem - goal - purpose . . . . .	7
4.2 Project specification . . . . .	7
4.3 Method for development . . . . .	8
4.4 Information gathering . . . . .	8
4.5 Risk analysis . . . . .	9
4.6 Main activities in further work . . . . .	10
4.7 Schedule / Timetable . . . . .	12
4.7.1 Main project plan . . . . .	12

<i>CONTENTS</i>	0
4.7.2 Project control assets . . . . .	13
4.7.3 Development assets . . . . .	14
4.7.4 Internal audit . . . . .	14
4.7.5 Decision making process . . . . .	14
4.7.6 Choice of robot . . . . .	14
<b>5 Documentation</b>	<b>16</b>
5.1 Reports and technical documents . . . . .	16
<b>6 Scheduled Meetings and Reports</b>	<b>17</b>
6.1 Meetings . . . . .	17
6.1.1 Meetings with the control group . . . . .	17
6.1.2 Project meetings . . . . .	17
6.2 Periodic reports . . . . .	17
6.2.1 Progress reports (including milestone) . . . . .	17
<b>7 Deviation management</b>	<b>18</b>
<b>8 Equipment needs / Requirements for implementation</b>	<b>19</b>
<b>Appendices</b>	<b>20</b>
<b>A First appendix</b>	<b>21</b>
A.1 Gantt Diagram . . . . .	21
<b>Bibliography</b>	<b>23</b>

# Chapter 1

## Introduction

With this thesis we want to develop a system whose task is to do autonomous inspection tasks in the process industry, with the use of a quadruped robot in a dynamic environment. The robot must be able to be controlled by a graphical user interface that support both manual and autonomous control. In addition should it support functions to detect predefined objects and shapes. This can for example be reading a gauge, or get the position of a valve.

The group has to model, construct and develop software to tackle the tasks at hand. This includes electrical equipment with actuators for motion, in support with sensors and ambient detection for space orientation. At last, the software will utilize path planning, motion control, reconstruction of scene in combination with neural networks to guarantee safe navigation and detection in a dynamic environment.

## Terminology

- **Quadruped Robot** – Robot that uses four legs for motion
- **SLAM** - Simultaneous Localization And Mapping
- **NN** - Neural Network
- **CNN** - Convolutional Neural Network
- **GUI** - Graphical User Interface
- **RRT\*** - Rapidly-Exploring Random Trees
- **SCRUM** - Agile process framework for managing complex knowledge work



# Chapter 2

## Project organization

### 2.1 Project group

Student ID number
Petter Drønnen - 479674
Vegard Solheim - 484225
Magnus Øye - 479675

Table 2.1: Student ID numbers for everyone in the group who submits the assignment for assessment in the subject ID: IE303612.

#### 2.1.1 Assignments for the project group - organization

Responsibility	Group member responsible
Project Manager	Magnus Øye
Secretary	Vegard Solheim
Worker	Petter Drønnen

#### 2.1.2 Assignments for the project manager

The project manager's area of responsibility will be to have an overview of the progresses in the project. At the same time, tasks such as:

- Make sure all group members are working with the project in a way that is productive and relevant to the desired solution.
- Coordinate with the group members to get an update of the projects progression.

### **2.1.3 Assignments for Secretary**

The secretary's area of responsibility will mainly be to maintain records associated in conjunction with group meetings.

- Arrange meetings
- Take notes under meetings
- Send info about when and where the meetings are held
- Ensure that the progress plan is updated prior to the meetings
- Ensure that all team members document their tasks in the project

### **2.1.4 Assignments for other members**

All team members are responsible for the tasks assigned to them through the project plan. Where each task will have one in charge, and one that will be available to help. Like this, each member is responsible for writing down how they solved the task, the work methodology and implementation.

At the same time, is everyone responsible for ensuring that all the activities that are assigned to each group member are worked on, and completed on time. Finally, everyone will participate in the report writing continuously during the project period.

## **2.2 Management-group**

Project advisors:

- Ivar Blindheim – Assistant Professor, NTNU
- Aleksander Skrede – PhD Candidate, NTNU

# Chapter 3

## Agreements

### 3.1 Agreements with employer

As this is a custom work assignment, no client has been assigned to it, nor has it entered into an agreement with anyone.

### 3.2 Workplace and resources

The group will do most of the work at the electronics laboratory on the NTNU campus Aalesund. Here the group will have access to various tools and the advisors will be easily accessible.

NTNU is willing to finance the project, with a budget of NOK 20 000.

### 3.3 Group norms - rules of cooperation - attitudes

#### 3.3.1 Terms and Conditions

Group members shall make every effort to:

- **Attendance**

- Attend in all meetings.
- Inform other group members prior to any absence, if possible.
- Actively seek any information missed in the event of absence.
- Aid any group member in recovering from an absence.

- **Completing of task**

- Complete all work assigned in a timely manner.
- Put sufficient effort and planning into such work.
- Seek aid in a timely manner if the work cannot be completed.
- **Use of time and resources**
  - Remain dedicated to the task at hand.
  - Be active in any problem-solving process.
- **Shared responsibility**
  - Correspond with other group members regarding any concern or problem.
  - Ensure other group members have a complete understanding of the subject matter.
  - Indicate to other group members if something is unclear.
  - Ensure that in no case will a single group member complete all necessary work or abstain completely from the project.
  - Come to a consensus regarding the solution to a problem.
    - \* Disagreements will be dealt with by reasoned argument.
    - \* If a solution cannot be agreed upon, the solution to be presented will be voted upon.

### **3.3.2 Acceptance**

Each of the group members agrees to abide by the terms and conditions outlined herein. Breach of this contract will result in a verbal warning the first and second offence. Third offence violation will result in dismissal from the group.

# Chapter 4

## Project description

### 4.1 Thesis problem - goal - purpose

The fundamental task is to develop a prototype that is able to move through, and inspect a dynamic environment. Finally, the plan is to stand with a robot capable of moving in a room both manually and autonomously. This leads the group to gain a deeper understanding of different automation techniques, and apply them practically. There are several aspects of automation that will be used in the task, such as SLAM, path planning, neural networks, which will assist in inspections, navigation and an understanding of the environment. A prototype will be built to obtain proof of concept.

### 4.2 Project specification

Our goal is to have a fully functioning robot after completing the bachelor. This involves a robot that can go into the modes manually controlled by a controller or in autonomous mode. The overall overview of the robot will be represented in a user-friendly GUI. This will include direct streaming from the camera and sensors. The operator should be able to select the mode, and other useful information from the robot that should be displayed. The quality should be sufficient in relation to the robot's use. The financial framework is followed as closely as possible, where one of the goals is to make the most of the school's existing resources.

Must have:	Should have:	Could have:
Working and user-friendly GUI	Independent navigation around or over obstacles	Follow an object
Live feed from depth camera and SLAM	Visualize data from depth camera	Self-learning in terms of movement
A robust mechanical construction	Module-based software	Digital simulation
Manual control	Safe operations	Inspect a given area
Autonomous control	Reading a gauge	Detection of gas leak and oil spill
Real Time Performance	Get the position of a valve	
	Reading analog counter	

### 4.3 Method for development

The group will use Instagantt from Asana to document progression through the project. This will control the various activities and who is responsible for them.

Furthermore will the development of the software already start from the begin, even though none of the components are available. At the same time, 3D modeling and printing will be done in the initial phase while the components are on the way.

In terms of software development, the group will follow an “agile” approach to development, SCRUM. The reason for this is that during the project period the group will build up new information that was not known at the start of the project. As a result, the software will have several iterations.

The electronics part will start when the components are in place and the mechanical is modeled and made, and finally, continuous testing will be ongoing to eliminate most errors at an early stage.

### 4.4 Information gathering

**Gathered:**

- Read reports and articles from projects that are similar to ours
- Experience from previous projects
- Found components that fit together and will work well for our project
- Investigate different technologies that can be used to solve the problems

**Planned:**

- Get a deeper understanding of the different technologies to implement them

## 4.5 Risk analysis

The time frame of the project is about 5 months. With such a time frame, the opportunity to realize the project seems achievable in terms of workload and experience from previous projects.

The group does not see the need to have a proposal for any clarification or refinement of the project as this is a custom task and the workload is adapted to challenge the students, but not overwhelming.

Risk	Severity	Reduce risk
Illness	2	Good nutrition and sleep
Delays in components	2	Plan ahead and work on other things instead of waiting
Loss of computer	1	Cloud storage and version control
Restate a solution	2	Good planning. Find a possible solution before it is implemented
Component failure	3	Quality components. Proper use and testing
Electrial shock	1	Powerless before working on the robot
Not charged battery	1	Access to multiple batteries, recharge the day before
Stress / to much work	3	Open communication in the group, where we help each other
Short circuit	3	Plan, and follow wiring diagram
Damage during transport	2	Secure the robot well
Loss of sensors or control during activity	2	Detection of loss, and performs a safe action based on the loss

Table 4.1: *Severity: 1 - 3, where 1 is LOW and 3 is HIGH. Frequency: Green - Red. Where Green is LOW and Red is HIGH.*

To succeed, a few things are especially important. First, continuity of work and good progress are important. Secondly, it is important that the group cooperates well. Some threats to success are major unforeseen problems either with construction or software that cannot be solved in an appropriate and good way.

## 4.6 Main activities in further work

Descriptions of planned main activities, and the most important sub-activities for the implementation of the project. The name on the main activities has the responsibility for the work gets done. The name on the sub-activities has the responsibility for doing the work.



<b>Nr</b>	<b>Activity</b>	<b>Responsibility</b>	<b>Cost/Time</b>
<b>A1</b>	<b>Mechanical</b>	<b>Vegard</b>	
A1.1	Frame / bodywork	<b>Vegard &amp; Petter</b>	4 days
A1.2	Leg assembly	<b>Vegard &amp; Petter</b>	1 day
A1.3	Actuators	<b>Petter &amp; Vegard</b>	2 days
<b>A2</b>	<b>Electronics</b>	<b>Petter</b>	
A2.1	Electrical diagram	<b>Petter</b>	2 days
A2.2	Batteries	<b>Vegard &amp; Petter</b>	1 day
A2.3	Motordrivers with associated sensors	<b>Magnus &amp; Vegard</b>	1 day
A2.4	Main Processor as well as other sensors	<b>Petter &amp; Magnus</b>	1 day
<b>A3</b>	<b>Software</b>	<b>Magnus</b>	
A3.1	Graphical User Interface	<b>Magnus &amp; Petter</b>	20 days
A3.2	Navigation	<b>Vegard &amp; Magnus</b>	7 days
A3.3	Position control	<b>Petter &amp; Magnus</b>	6 days
A3.4	Path planning	<b>Vegard &amp; Magnus</b>	9 days
A3.5	Object detection and classification	<b>Magnus &amp; Vegard</b>	5 days
A3.6	I/O Operations	<b>Magnus &amp; Petter</b>	3 days
A3.7	Intern communication	<b>Magnus &amp; Vegard</b>	4 days
A3.8	Manual control	<b>Vegard &amp; Petter</b>	2 days
A3.9	Autonomous control	<b>Magnus &amp; Vegard</b>	20 days
A3.10	Calibration sequence	<b>Magnus &amp; Petter</b>	1 day
A3.11	Inverse kinematics	<b>Petter &amp; Vegard</b>	7 days
A3.12	Open loop control	<b>Petter &amp; Vegard</b>	3 days
A3.13	Closed loop control	<b>Petter &amp; Vegard</b>	10 days
A3.14	External communication	<b>Magnus</b>	2 days
A3.15	Locomotion	<b>Petter &amp; Vegard</b>	12 days

<b>A4</b>	<b>Report</b>	<b>Everyone</b>	
A4.1	Preliminary	<b>Everyone</b>	14 days
A4.2	Weekly	<b>Vegard</b>	3 days
A4.3	Meetings	<b>Vegard</b>	2 days
A4.4	Main	<b>Everyone</b>	60 days
<b>A5</b>	<b>Project management</b>	<b>Vegard</b>	
A5.1	Planning	<b>Vegard &amp; Magnus</b>	5 days
A5.3	Keep track of progress	<b>Magnus &amp; Vegard</b>	2 days
<b>A6</b>	<b>Testing and calibration</b>	<b>Petter</b>	
A6.1	Electronics	<b>Petter &amp; Vegard</b>	3 days
A6.2	Software	<b>Magnus &amp; Petter</b>	6 days
<b>A</b>	<b>All activities</b>		<b>215 days</b>

## 4.7 Schedule / Timetable

### 4.7.1 Main project plan

#### A1 - Mechanical

This point will depend a great deal on the procurement of parts, which are largely made up of 3D-printed, machined and other fabricated parts, but the equipment and fabrication opportunities that the school has will be largely utilized. In addition to this, everything will be mounted, and possibly new iterations are made of whether tolerances are wrong or parts do not meet their required requirements. The CAM software used is Fusion 360 by Autodesk.

**A2 - Electronics**

Electrical equipment consists mainly of drawing, and connecting the electrical to the robot. Where drawing of the electrical will take place in PC-Schematic. The task also consists of procuring and installing electrical components such as battery, motor controls and various sensors to the robot.

**A3 - Software**

Programming will be a major part of this project, where several advanced concepts will be implemented, and communicated in a secure and optimized way. Different open source libraries and different coding languages will be used on the respective systems. Where the communication internally and between the systems will use specified and known protocols. Finally, the graphical user interface will have a central role for the system.

**A4 - Report**

Everyone in the project should write the report throughout the project to ensure that procedures and methods are documented while still fresh in memory.

**A5 - Project management**

Creating a detailed plan will reduce the errors later in the project. Concept drawings are made for all the main parts of the project. The concept drawings must contain all the equipment and measurements needed to complete each part of a project. During the planning an equipment list will be created and parts will be ordered.

**A6 - Testing and calibration**

Testing and calibration will be an important part of the end of the project. The work will include extensive hardware and software testing and optimization through the whole project.

**4.7.2 Project control assets**

- Jupyter notebook for visualization and prototyping.

- LaTeX to write the report with Master Thesis of NTNU as template.
- OneDrive for cloud storage and file sharing.
- GitHub for version control of code.

### **4.7.3 Development assets**

The group sees a need for machining certain mechanical components to keep budget within the cost framework.

- Development environment for programming: Visual Studio Code, Arduino IDE.
- Autodesk Fusion 360 for mechanical modeling.
- PC-Schematic for electrical documentation

### **4.7.4 Internal audit**

Every Monday, the group will have a meeting where there will be a joint review of progress and general follow-up of the project. Characteristic that a goal has been achieved is that the requirement for the specific goal has been met.

### **4.7.5 Decision making process**

Decisions made during the preliminary project were first discussed in plenary, then the majority within the group determined the outcome. Such a process is also how the group envisions making important decisions in the main project.

### **4.7.6 Choice of robot**

Among legged robots, the quadruped robots have good mobility and stability of locomotion, while the typical biped robots lack the locomotion stability. The quadruped robot is also a good choice from the point of view of a system and controller design. On the other hand, four legs are less complicated to construct and maintain than six ones. Quadruped robots are more versatile

than wheeled and tracked robots, and more stable than biped robots [3].

The necessities of mobile robots for complex and dangerous environments have initiated the development of dynamic quadruped machines, which exploit the potential advantages of the legged locomotion and enhanced mobility in unstructured terrains. Such a versatile system with high maneuverability should be labor efficient, cost-effective, and indispensable in industries. The shape also helps the robot to walk in a comparatively narrow region as it possesses smaller footprint [1].

Considering the stability and dynamic movement facility of the structure, the development of a Quadruped robot is selected for this thesis.

Even tho better quality quadruped robots are available, the costs of those are beyond reach for general researchers. The designed robot will be very cheap compared to the similar robots available. This robot can be easily constructed from materials available locally.

Although wheeled and tracked robots can work well in even surface, most of them cannot work in uneven surface. In other words, only a portion of the earth's landmass is accessible for the existing ground robots. Compared with the wheeled and tracked robots, legged robots have the potential to walk in a much wider variety of terrains [2].

# Chapter 5

## Documentation

### 5.1 Reports and technical documents

- Electrical documentation
- Mechanical documentation
- System source code
- Report
- User manual

All documents will be stored in a cloud solution for safety reasons.

# Chapter 6

## Scheduled Meetings and Reports

### 6.1 Meetings

#### 6.1.1 Meetings with the control group

The group has scheduled a meeting with all the group members together with the supervisor every other week. Prior to the meetings, members will update the report and project progress.

#### 6.1.2 Project meetings

A meeting is scheduled every Monday at 8.30 am to update the project plan, and discuss the problems that have arisen and possible solutions. At the same time, will the main topics be noted by the secretary for reference.

### 6.2 Periodic reports

#### 6.2.1 Progress reports (including milestone)

A report will be written at the end of each week. This report will contain progress in the project, problems that have arisen, and relevant topics. These reports will be the basis for the meeting with the supervisors.

# Chapter 7

## Deviation management

If progress is to occur behind schedule, the group must revise the planned plan. If there is not going to be a solution to the problem, a reconciliation will be carried out where less relevant activities will be removed. It will be the group leader who calls for a vote, and each member of the group will have equal contribution with a vote.



# Chapter 8

## Equipment needs / Requirements for implementation

Components and equipment needed for project execution:

<b>Component</b>	<b>Reason</b>
Jetson Nano	Processing unit, runs the software
Teensy 4.0	Microcontroller, I/O operations
LED's	Status indicator
Push buttons	Physical user interface
Switches	Power cutoff
MPU	Measuring motion in 3D space
Batteries	Power the robot
Brushless motors	Gives the robot motion
Absolute encoders	Position control of the motors
Router	External communication with the robot
Voltage regulator	Regulate voltage for lower voltage driven components
Power distribution board	Distribute the power around the robot
Motordrivers	Interface for controlling the motors
Accessories	Wires, screws, nuts, rivets, shrink sleeves ...
Construction material	Construction of robot
Depth camera	Understanding the environment
Tracking camera	Location and mapping
3D printer	3D printing of parts

# **Appendices**

# **Appendix A**

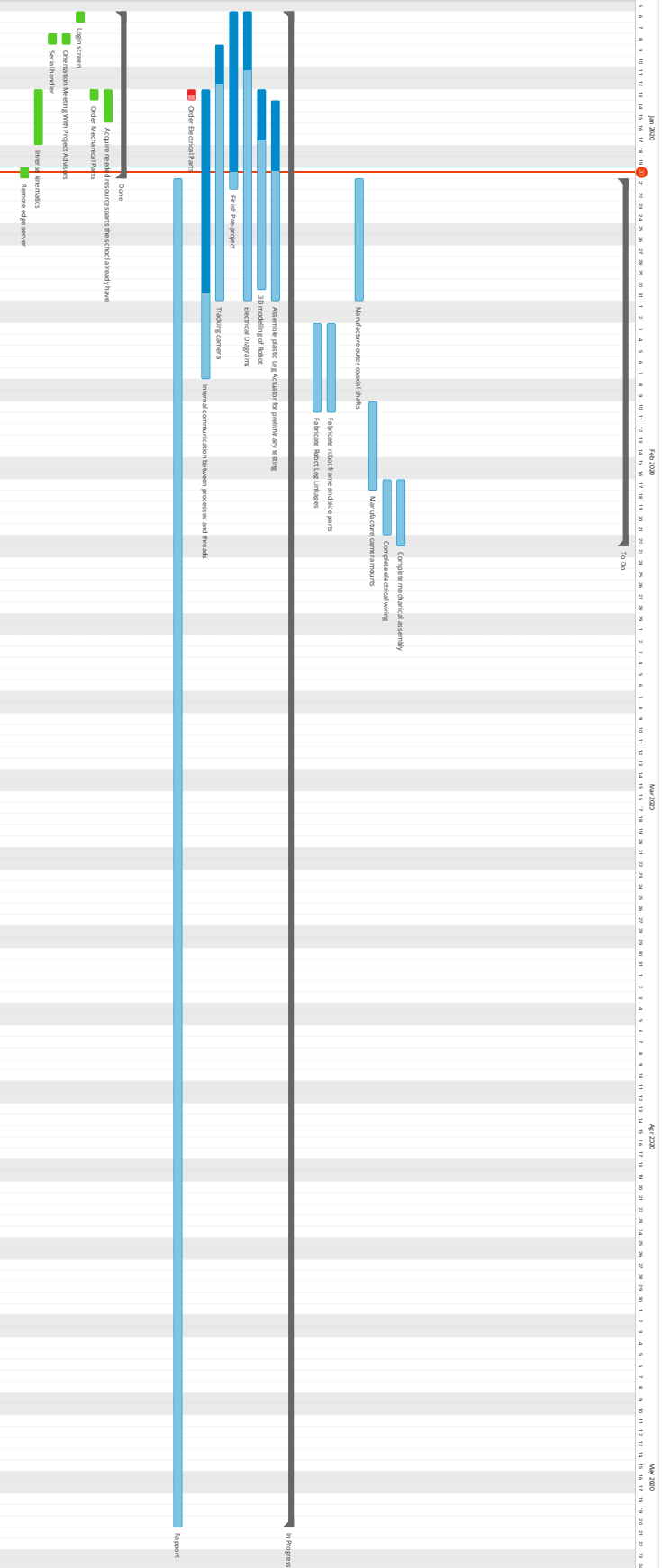
## **First appendix**

### **A.1 Gantt Diagram**

# Quadraped Robot - Navigation and Inspection

Release Date: 2020-05-20 10:30

ID	Task Name	Priority	Start	End	Status
1	Deploy Camera	High	2020-05-10	2020-05-10	Done
2	Open loop control	Low	2020-05-10	2020-05-10	Done
3	Close loop control	Low	2020-05-10	2020-05-10	Done
4	External communication	High	2020-05-10	2020-05-10	Done
5	Localization	High	2020-05-10	2020-05-10	Done
6	Q-Questions	High	2020-05-10	2020-05-10	Done
7	Path planning	High	2020-05-10	2020-05-10	Done
8	Manual control	High	2020-05-10	2020-05-10	Done
9	Autonomous control	High	2020-05-10	2020-05-10	Done
10	Calibration equipment	High	2020-05-10	2020-05-10	Done
11	Navigation	High	2020-05-10	2020-05-10	Done
12	Autonomous control	High	2020-05-10	2020-05-10	Done
13	Autonomous control	High	2020-05-10	2020-05-10	Done
14	Train and optimize CNN	High	2020-05-10	2020-05-10	Done
15	Getter training data for CNN	High	2020-05-10	2020-05-10	Done
16	Complete mechanical assembly	High	2020-05-10	2020-05-10	Done
17	Complete electrical wiring	High	2020-05-10	2020-05-10	Done
18	Manufacture outer camera shells	High	2020-05-10	2020-05-10	Done
19	Manufacture outer camera shells	High	2020-05-10	2020-05-10	Done
20	External communication server	High	2020-05-10	2020-05-10	Done
21	External communication server	High	2020-05-10	2020-05-10	Done
22	External communication server	High	2020-05-10	2020-05-10	Done



19 Progress

# Bibliography

- [1] Atsushi Horigome & Gen Endo Satoshi Kitano, Shigeo Hirose. Titan-xiii: sprawling-type quadruped robot with ability of fast and energy-efficient walking, 03 2016. URL <https://robomechjournal.springeropen.com/articles/10.1186/s40648-016-0047-1>.
- [2] JiuHong Ruan Xuewen Rong SYibin Li, Bin Li. Research of mammal bionic quadruped robots: A review, 09 2011. URL [https://ieeexplore.ieee.org/abstract/document/6070476?fbclid=IwAR19MMDVf3F3GafgcxGf1uh-cPahf\\_7A121vQEMA462R3POZdj1upH1sbbc](https://ieeexplore.ieee.org/abstract/document/6070476?fbclid=IwAR19MMDVf3F3GafgcxGf1uh-cPahf_7A121vQEMA462R3POZdj1upH1sbbc).
- [3] Avis H. Cohen Yasuhiro Fukuoka, Hiroshi Kimura. Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts, 03 2003. URL [https://journals.sagepub.com/doi/abs/10.1177/0278364903022003004?casa\\_token=uXbIGQgo2ZoAAAAA%3Afa7tZG0Nzj5Kci\\_xVWIW0m\\_1vhd2uGe2pYhCZ8z\\_WOSYc551vwmMWRv1KRL\\_hK7LHHXzPuosQOCGbw&](https://journals.sagepub.com/doi/abs/10.1177/0278364903022003004?casa_token=uXbIGQgo2ZoAAAAA%3Afa7tZG0Nzj5Kci_xVWIW0m_1vhd2uGe2pYhCZ8z_WOSYc551vwmMWRv1KRL_hK7LHHXzPuosQOCGbw&).

**A.2 Risk analysis Covid-19**



NTNU

Norwegian University of  
Science and Technology

# **Risk analysis Covid-19**

## **Autonomous inspections for process industry by a quadruped robot with use of neural networks**

Bachelor Thesis: IE303612

Petter Drønne, Vegard Solheim, Magnus Øye

March 2020

Faculty of Information Technology and Electrical Engineering

Norwegian University of Science and Technology

Supervisor 1: Ivar Blindheim

Supervisor 2: Aleksander Skrede

# Contents

- 1 Risk analysis Covid-19** **1**
- 1.1 Measures taken to reduce the spread ..... 1
- 1.2 Consequences of the virus ..... 2
- 1.3 Risk analysis ..... 3



# Chapter 1

## Risk analysis Covid-19

A risk analysis for the project was made before the project started. In that analysis the group addresses the normal risks, but because of the unforeseen spread of a global pandemic a new risk analysis were made.

In this risk analysis we will address the risk associated with a pandemic, the measures taken to reduce the spread and how it affected us and the project.

### 1.1 Measures taken to reduce the spread

The group follows all of the Norwegian Directorate of Health recommendation considering the Covid-19, and can be read here [Recommendation](#). Daily measures taken can be seen in figure 1.1. In addition to this the group does the following:

- Stays home as much as possible
- All meetings with supervisors is done by video
- Splitting work so more can be done from home



Figure 1.1: Measures to prevent infection

## 1.2 Consequences of the virus

There are several consequences of the virus. With the school closing the group loses the ability to use the schools resources such as classrooms and equipment's. Also the printing of 3D-parts is harder and takes longer time. The delivery of parts can be heavily delayed or canceled. If one of the members gets the virus one can expected one out of two outcomes. The first and mildest outcome is that the member don't get severe symptoms and can therefor work from home. The second outcome is if the member get severe symptoms and therefor can't do any work. This will put the other members under a lot of work.

### 1.3 Risk analysis

Risk	Severity	Reduce risk
Sick due to Covid-19	3	Good hygiene and reduce unnecessary contact with people
Delays in components	2	Plan ahead and work on other things instead of waiting
Missing equipment	2	Plan ahead and find other places to get equipment that is needed
School closed	2	Bring necessary components home
Communication problems	2	Have a lot of online meetings
Home quarantine	1	Reduce unnecessary contact with people Don't travel outside of home municipality

Table 1.1: Severity: 1 - 3, where 1 is LOW and 3 is HIGH. Frequency: Green - Red. Where Green is LOW and Red is HIGH.

**A.3 Gantt diagram**

# Quadruped Robot - Navigation and Inspection

Read-only view, generated on 19 May 2020

ACTIVITIES	ASSIGNEE	EH	START	DUE	%
<b>To Do</b>		-			0%
<b>In Progress</b>		-			0%
<b>Issues</b>		-	24/Feb	24/Feb	0%
✓ Motors are delayed and hav...	Vegard Solheim	-	24/Feb	24/Feb	0%
<b>Done</b>		-	06/Jan	20/May	100%
5 ✓ Electrical Diagrams	Petter Drønne	-	06/Jan	31/Jan	100%
6 ✓ Finish Pre-project	Vegard Solheim	-	06/Jan	21/Jan	100%
7 ✓ Login screen	Magnus Øye	-	06/Jan	06/Jan	100%
8 ✓ Orientation Meeting With Pr...	Magnus Øye	-	08/Jan	08/Jan	100%
9 ✓ Serial handler	Magnus Øye	-	08/Jan	08/Jan	100%
10 ✓ Tracking camera	Magnus Øye	-	09/Jan	31/Jan	100%
11 ✓ 3D modelling of Robot	Vegard Solheim	-	13/Jan	30/Jan	100%
12 ✓ Internal communication bet...	Magnus Øye	-	13/Jan	07/Feb	100%
13 ✓ Order Electrical Parts	Petter Drønne	-	13/Jan	13/Jan	100%
14 ✓ Acquire needed resourcesp...	Vegard Solheim	-	13/Jan	15/Jan	100%
15 ✓ Order Mechanical Parts	Vegard Solheim	-	13/Jan	13/Jan	100%
16 ✓ Inverse kinematics	Petter Drønne	-	13/Jan	17/Jan	100%
17 ✓ Assemble plastic Leg Actuat...	Vegard Solheim	-	14/Jan	31/Jan	100%
18 ✓ Remote edge server	Magnus Øye	-	20/Jan	20/Jan	100%
19 ✓ Report	Petter Drønne	-	21/Jan	20/May	100%
20 ✓ Manufacture outer coaxial s...	Vegard Solheim	-	21/Jan	31/Jan	100%
21 ✓ Calibration sequence	Magnus Øye	-	26/Jan	31/Jan	100%
22 ✓ Depth Camera	Magnus Øye	-	03/Feb	06/Feb	100%
23 ✓ Gather training data for CNN	Vegard Solheim	-	03/Feb	03/Feb	100%
24 ✓ Fabricate robot frame and s...	Vegard Solheim	-	03/Feb	10/Feb	100%
25 ✓ Fabricate Robot Leg Linkages	Vegard Solheim	-	03/Feb	10/Feb	100%
26 ✓ Annotate training data	Magnus Øye	-	04/Feb	04/Feb	100%
27 ✓ Train and optimize CNN	Magnus Øye	-	05/Feb	05/Feb	100%
28 ✓ External communication	Magnus Øye	-	10/Feb	10/Feb	100%
29 ✓ Manufacture camera mounts	Vegard Solheim	-	10/Feb	17/Feb	100%
30 ✓ Make a box with integrated ...	Vegard Solheim	-	12/Feb	12/Feb	100%
✓ Complete mechanical asse...	Vegard Solheim	-	17/Feb	06/Mar	100%
✓ Electrical wiring	Petter Drønne	-	17/Feb	24/Feb	100%
33 ✓ Open loop control	Petter Drønne	-	18/Feb	25/Feb	100%
✓ Manufacture Front and Rea...	Vegard Solheim	-	19/Feb	19/Feb	100%
35 ✓ Manual control	Vegard Solheim	-	01/Mar	18/Mar	100%
36 ✓ IO Operations	Magnus Øye	-	08/Mar	13/Mar	100%
37 ✓ External communication bet...	Magnus Øye	-	22/Mar	27/Mar	100%
38 ✓ Closed loop control	Petter Drønne	-	01/Apr	01/May	100%
39 ✓ Path planning	Vegard Solheim	-	01/Apr	01/May	100%
40 ✓ Autonomous control	Magnus Øye	-	01/Apr	01/May	100%
41 ✓ Navigation	Vegard Solheim	-	01/Apr	01/May	100%
42 ✓ Position control	Petter Drønne	-	01/Apr	01/May	100%
43 ✓ Locomotion	Petter Drønne	-	01/Apr	01/May	100%
44 ✓ Waypoint system	Petter Drønne	-	13/Apr	08/May	100%
45 ✓ SQL Database	Magnus Øye	-	20/Apr	08/May	100%
46 ✓ REST-API	Magnus Øye	-	20/Apr	08/May	100%
47 ✓ Optical Character Recogniti...	Magnus Øye	-	27/Apr	08/May	100%
48 ✓ Collision avoidance	Vegard Solheim	-	27/Apr	01/May	100%



## **A.4 Progress reports**

### **A.4.1 Progress report 19.01.20**

<b>Bachelor Thesis: IE303612</b>	Project Autonomous inspections for process industry by a quadruped robot with use of neural networks	Number of meeting this period 1). 2 planned	Firm - Contracting NTNU in Ålesund	Page 1 av 1
<b>Progress report</b>	Period/week(s) 2	Number of hours this period. (from log) Approx. 90	Project group (name) Petter Drønnen Vegard Solheim Magnus Øye	Dato 19.01.20

Main goal/purpose for this periods work - Finish preliminary project	
Planned activities this period - Meet supervisors - Work with preliminary project - Start on main tasks - Installation of required software on Jetson Nano	
Actually conducted activities this period - Met supervisors - Finished preliminary project - Got Jetson Nano up and running - Finished inverse kinematic - Finished ordering parts and acquired needed resources/pars from school	
Description of/ justification for potential deviation between planned and real activities -	
Description of/ justification for changes that is desired in the projects content or in the further plan of action – or progress report -	
Main experience from this period - This period was the first, therefore it felt a bit slow - When we first got supervisors and were ready to start work, we got a lot done. Like starting on the inverse kinematic and got the jetson nano running. - All in all, a good first period where the group got well underway	
Main purpose/focus next period - Start on mechanical tasks - Continuing software development	
Planned activities next period - Continuing working on main tasks from gantt diagram	
Other -	
Wish/need for counseling - Nothing in particular	
Approval/signature group leader Magnus Øye	Signature other group participants Petter Drønnen Vegard Solheim

1) Note here brief feedback on the number of meetings - by type (internal, steering group, meeting with supervisor) - during this reporting period

**A.4.2 Progress report 09.02.20**



<b>Bachelor Thesis: IE303612</b>	Project Autonomous inspections for process industry by a quadruped robot with use of neural networks	Number of meeting this period 1). 1 planned	Firm - Contracting NTNU in Ålesund	Page 1 av 1
<b>Progress report</b>	Period/week(s) 3	Number of hours this period. (from log) Approx. 120	Project group (name) Petter Drønne Vegard Solheim Magnus Øye	Dato 09.02.20

Main goal/purpose for this periods work	
<ul style="list-style-type: none"> <li>- Start on mechanical tasks</li> <li>- Continuing software development</li> </ul>	
Planned activities this period	
<ul style="list-style-type: none"> <li>- Continuing working on main tasks from gantt diagram</li> <li>- Test cameras</li> <li>- Make a simple simulation of the foot movement</li> <li>- Make 3d-printed encapsulation for the cameras</li> </ul>	
Actually conducted activities this period	
<ul style="list-style-type: none"> <li>- Most of the internal communication is finished</li> <li>- The cameras are up and running on intel's own software</li> <li>- Made a simple simulation and found a problem on are inverse kinematic. These problems are now solved</li> <li>- The encapsulation for the cameras is finished</li> <li>- Trained and early AI for detecting vales and manometers. This worked well, but still need more work</li> </ul>	
Description of/ justification for potential deviation between planned and real activities	
-	
Description of/ justification for changes that is desired in the projects content or in the further plan of action – or progress report	
-	
Main experience from this period	
<ul style="list-style-type: none"> <li>- Felt like an ok period. In week five, almost nothing was done because of the subject “Industri 4.0” had lectures and tasks that took the whole week to finish.</li> <li>- We are really starting to feel the lack of components, so we hope this arrive early next period so we can start testing.</li> </ul>	
Main purpose/focus next period	
<ul style="list-style-type: none"> <li>- Physical testing</li> </ul>	
Planned activities next period	
<ul style="list-style-type: none"> <li>- Test the communication with motors and encoders</li> <li>- Test the inverse kinematic</li> <li>- Work on software with main focus on camera</li> </ul>	
Other	
Wish/need for counseling	
<ul style="list-style-type: none"> <li>- Nothing in particular</li> </ul>	
Approval/signature group leader	Signature other group participants
Magnus Øye	Petter Drønne Vegard Solheim

1) Note here brief feedback on the number of meetings - by type (internal, steering group, meeting with supervisor) - during this reporting period

**A.4.3 Progress report 01.03.20**

<b>Bachelor Thesis: IE303612</b>	Project Autonomous inspections for process industry by a quadruped robot with use of neural networks	Number of meeting this period 1). 1 planned	Firm - Contracting NTNU in Ålesund	Page 1 av 2
<b>Progress report</b>	Period/week(s) 3	Number of hours this period. (from log) Approx. 150	Project group (name) Petter Drønnen Vegard Solheim Magnus Øye	Dato 01.03.20

<p>Main goal/purpose for this periods work</p> <ul style="list-style-type: none"> <li>- Test the communication with motors and encoders</li> <li>- Test the inverse kinematic</li> <li>- Work on software for running multiple processes concurrently</li> <li>- Implement a GUI for manual control</li> </ul>
<p>Planned activities this period</p> <ul style="list-style-type: none"> <li>- Continuing working on main tasks from gantt diagram</li> <li>- Test cameras on Jetson</li> <li>- Test IO's</li> <li>- Test one Odrive and two motors. Odrive and motors are taken from an older project.</li> <li>- Complete mechanical assembly</li> <li>- Created a working manual control graphical interface</li> <li>- Both cameras and serial communication is publishing information concurrently</li> </ul>
<p>Actually conducted activities this period</p> <ul style="list-style-type: none"> <li>- Cameras are running on Jetson</li> <li>- Buttons and LEDs are working as planned</li> <li>- Two motors are working as intended</li> <li>- All mechanical work is done, except motors and encoders.</li> <li>- All electrical work is done, except motors and encoders.</li> <li>- GUI for manual control finished</li> </ul>
<p>Description of/ justification for potential deviation between planned and real activities</p> <ul style="list-style-type: none"> <li>- The motors still have not arrived, so it is not possible to finished mechanical assembly yet</li> </ul>
<p>Description of/ justification for changes that is desired in the projects content or in the further plan of action – or progress report</p> <p>-</p>
<p>Main experience from this period</p> <ul style="list-style-type: none"> <li>- Stable progress but feels a bit to slow because the lack of the motors.</li> <li>- We are trying our best without the eight motors. Really hope we can get them by week 11. Week 10 is short due to Industry 4.0 course.</li> </ul>
<p>Main purpose/focus next period</p> <ul style="list-style-type: none"> <li>- Physical testing</li> <li>- In week 10, at least three full days goes to completing the last module of Industry 4.0</li> <li>- More work on the detection and classification model</li> </ul>
<p>Planned activities next period</p> <ul style="list-style-type: none"> <li>- Get all eight motors working</li> <li>- Start sending information back and forth between Teensy, Jetson and GUI</li> <li>- Have a working communication between all the systems</li> <li>- Have an open loop gait for locomotion</li> <li>- Report</li> <li>- Teensy threads</li> </ul>
<p>Other</p> <ul style="list-style-type: none"> <li>- Almost all of week 7 was spent working on Industry 4.0.</li> </ul>
<p>Wish/need for counseling</p>

1) Note here brief feedback on the number of meetings - by type (internal, steering group, meeting with supervisor) - during this reporting period

<b>Bachelor Thesis: IE303612</b>	Project Autonomous inspections for process industry by a quadraped robot with use of neural networks	Number of meeting this period 1). 1 planned	Firm - Contracting NTNU in Ålesund	Page 2 av 2
<b>Progress report</b>	Period/week(s) 3	Number of hours this period. (from log) Approx. 150	Project group (name) Petter Drønnen Vegard Solheim Magnus Øye	Dato 01.03.20

- Nothing in particular	
Approval/signature group leader Magnus Øye	Signature other group participants Petter Drønnen Vegard Solheim

1) Note here brief feedback on the number of meetings - by type (internal, steering group, meeting with supervisor) - during this reporting period

**A.4.4 Progress report 15.03.20**

<b>Bachelor Thesis: IE303612</b>	Project Autonomous inspections for process industry by a quadruped robot with use of neural networks	Number of meeting this period 1). 1 planned	Firm - Contracting NTNU in Ålesund	Page 1 av 2
<b>Progress report</b>	Period/week(s) 2	Number of hours this period. (from log) Approx. 150	Project group (name) Petter Drønne Vegard Solheim Magnus Øye	Dato 15.03.20

<p>Main goal/purpose for this periods work</p> <ul style="list-style-type: none"> <li>- Physical testing</li> <li>- In week 10, at least three full days goes to completing the last module of Industry 4.0</li> <li>- More work on the detection and classification model</li> </ul>
<p>Planned activities this period</p> <ul style="list-style-type: none"> <li>- Get all eight motors working</li> <li>- Start sending information back and forth between Teensy, Jetson and GUI</li> <li>- Have a working communication between all the systems</li> <li>- Have an open loop gait for locomotion</li> <li>- Report</li> <li>- Teensy threads</li> </ul>
<p>Actually conducted activities this period</p> <ul style="list-style-type: none"> <li>- All eight motors are working</li> <li>- Information is sent and received between the different systems</li> <li>- Robot can perform different movements and gaits while resting on the workbench</li> <li>- Report</li> <li>- Started working on implementing threads on the Teensy, not yet finished.</li> <li>- Conducted multiple tests with ROS</li> <li>- Localization is complete</li> <li>- Collision avoidance is under progress</li> <li>- Different mapping methods has been researched</li> <li>- Implemented a xbox-controller interface, communication and serialization</li> </ul>
<p>Description of/ justification for potential deviation between planned and real activities</p> <ul style="list-style-type: none"> <li>- Did not have time for more work on detection and classification model, and because of the school is in shutdown this work is put on hold.</li> </ul>
<p>Description of/ justification for changes that is desired in the projects content or in the further plan of action – or progress report</p> <p>-</p>
<p>Main experience from this period</p> <ul style="list-style-type: none"> <li>- Finally got the motors so felt like a lot were done this period.</li> <li>- Communication between systems are starting to take shape.</li> <li>- ROS has been tested, and the group got a better understanding how mapping can be implemented</li> </ul>
<p>Main purpose/focus next period</p> <ul style="list-style-type: none"> <li>- Physical testing</li> <li>- Use xbox-controller to perform different movements and operations</li> <li>- Mapping</li> <li>- Collision avoidance</li> </ul>
<p>Planned activities next period</p> <ul style="list-style-type: none"> <li>- Keep improving movement of the robot.</li> <li>- If possible, start testing walking the robot on the floor</li> <li>- Start implementing threads on Teensy</li> <li>- Have a working mapping and collision detection system</li> </ul>
<p>Other</p>

1) Note here brief feedback on the number of meetings - by type (internal, steering group, meeting with supervisor) - during this reporting period

<b>Bachelor Thesis: IE303612</b>	Project Autonomous inspections for process industry by a quadraped robot with use of neural networks	Number of meeting this period 1). 1 planned	Firm - Contracting NTNU in Ålesund	Page 2 av 2
<b>Progress report</b>	Period/week(s) 2	Number of hours this period. (from log) Approx. 150	Project group (name) Petter Drønnen Vegard Solheim Magnus Øye	Dato 15.03.20

<ul style="list-style-type: none"> <li>- Because of the corona virus the group is no longer on school.</li> <li>- All members work from home and have their own tasks to complete.</li> <li>- If necessary, the group can meet in one of the members house.</li> <li>- Progress video → <a href="https://youtu.be/TQ3uUlvOUME">https://youtu.be/TQ3uUlvOUME</a></li> </ul>	
Wish/need for counseling	
- Nothing in particular	
Approval/signature group leader  Magnus Øye	Signature other group participants  Petter Drønnen Vegard Solheim

1) Note here brief feedback on the number of meetings - by type (internal, steering group, meeting with supervisor) - during this reporting period

**A.4.5 Progress report 29.03.20**



<b>Bachelor Thesis: IE303612</b>	Project Autonomous inspections for process industry by a quadruped robot with use of neural networks	Number of meeting this period 1). 0 planned	Firm - Contracting NTNU in Ålesund	Page 1 av 2
<b>Progress report</b>	Period/week(s) 2	Number of hours this period. (from log) Approx. 190	Project group (name) Petter Drønne Vegard Solheim Magnus Øye	Dato 29.03.20

Main goal/purpose for this periods work	
<ul style="list-style-type: none"> <li>- Physical testing</li> <li>- Use xbox-controller to perform different movements and operations</li> <li>- Mapping</li> <li>- Collision avoidance</li> </ul>	
Planned activities this period	
<ul style="list-style-type: none"> <li>- Keep improving movement of the robot.</li> <li>- If possible, start testing walking the robot on the floor</li> <li>- Start implementing threads on Teensy</li> <li>- Have a working mapping and collision detection system</li> </ul>	
Actually conducted activities this period	
<ul style="list-style-type: none"> <li>- Using the xbox-controller to perform different movement is working</li> <li>- Occupancy grid mapping now working</li> <li>- Collision detection based on occupancy grid</li> <li>- Torque measurements from the motors</li> <li>- Waypoint system for semi-autonomous created</li> </ul>	
Description of/ justification for potential deviation between planned and real activities	
<ul style="list-style-type: none"> <li>- Need to print new gear before the robot can stand on its own. Not so easy as we are not on school anymore but will get Markus or Øyvind to fix it for us.</li> <li>- Thread implementation is postponed until further notice due to its a big job and uncertain of the speed benefits. Therefore, we focus on finishing more urgent tasks.</li> </ul>	
Description of/ justification for changes that is desired in the projects content or in the further plan of action – or progress report	
-	
Main experience from this period	
<ul style="list-style-type: none"> <li>- Working from home works ok, but the plan forward is to meet and work together.</li> </ul>	
Main purpose/focus next period	
<ul style="list-style-type: none"> <li>- Physical testing</li> <li>- Path planning</li> <li>- Utilize waypoint system</li> </ul>	
Planned activities next period	
<ul style="list-style-type: none"> <li>- Start walking the robot on the floor</li> <li>- Follow given waypoint path</li> </ul>	
Other	
<ul style="list-style-type: none"> <li>- Because of the corona virus the group is no longer on school.</li> <li>- All members work from home and have their own tasks to complete.</li> <li>- If necessary, the group can meet in one of the members house.</li> <li>- Progress video → <a href="https://youtu.be/gNAVOJ_CyQM">https://youtu.be/gNAVOJ_CyQM</a></li> </ul>	
Wish/need for counseling	
<ul style="list-style-type: none"> <li>- Nothing in particular</li> </ul>	
Approval/signature group leader	Signature other group participants
Magnus Øye	Petter Drønne

1) Note here brief feedback on the number of meetings - by type (internal, steering group, meeting with supervisor) - during this reporting period

<b>Bachelor Thesis: IE303612</b>	Project Autonomous inspections for process industry by a quadruped robot with use of neural networks	Number of meeting this period 1). 0 planned	Firm - Contracting NTNU in Ålesund	Page 2 av 2
<b>Progress report</b>	Period/week(s) 2	Number of hours this period. (from log) Approx. 190	Project group (name) Petter Drønnen Vegard Solheim Magnus Øye	Dato 29.03.20

	Vegard Solheim
--	----------------

1) Note here brief feedback on the number of meetings - by type (internal, steering group, meeting with supervisor) - during this reporting period

**A.4.6 Progress report 12.04.20**

<b>Bachelor Thesis: IE303612</b>	Project Autonomous inspections for process industry by a quadraped robot with use of neural networks	Number of meeting this period 1). 0 planned	Firm - Contracting NTNU in Ålesund	Page 1 av 2
<b>Progress report</b>	Period/week(s) 2	Number of hours this period. (from log) Approx. 200	Project group (name) Petter Drønne Vegard Solheim Magnus Øye	Dato 12.04.20

Main goal/purpose for this periods work	
<ul style="list-style-type: none"> <li>- Physical testing</li> <li>- Path planning</li> <li>- Utilize waypoint system</li> </ul>	
Planned activities this period	
<ul style="list-style-type: none"> <li>- Start walking the robot on the floor</li> <li>- Follow given waypoint path</li> </ul>	
Actually conducted activities this period	
<ul style="list-style-type: none"> <li>- Robot is standing and walking on the floor</li> <li>- Collision detection</li> <li>- Pathplanning</li> <li>- Pointcloud map construction</li> <li>- Controllers stabilization the center of gravity</li> <li>- Replaced ZMQ with ROS</li> <li>- Manual controlling over bluetooth</li> <li>- Did some testing with LIDAR</li> </ul>	
Description of/ justification for potential deviation between planned and real activities	
<ul style="list-style-type: none"> <li>- The robot needs more optimization before it can follow a given waypoint path, so this is postponed to next period.</li> </ul>	
Description of/ justification for changes that is desired in the projects content or in the further plan of action – or progress report	
-	
Main experience from this period	
<ul style="list-style-type: none"> <li>- Very good period where a lot got done.</li> <li>- Made new legs in aluminum because the nylon legs were not stiff enough sideways</li> </ul>	
Main purpose/focus next period	
<ul style="list-style-type: none"> <li>- Physical testing</li> <li>- Path planning and collision detection testing</li> <li>- Utilize waypoint system</li> <li>- Optimize CNN and add more objects</li> </ul>	
Planned activities next period	
<ul style="list-style-type: none"> <li>- Improve movement on the robot</li> <li>- Follow given waypoint path</li> <li>- Test autonomous system</li> </ul>	
Other	
<ul style="list-style-type: none"> <li>- Because of the corona virus the group is no longer on school.</li> <li>- All members meet up in one of the members home.</li> <li>- ROS has been chosen because of its performance boost over ZMQ</li> <li>- Progress video → <a href="https://youtu.be/oO2DqNTosMY">https://youtu.be/oO2DqNTosMY</a></li> </ul>	
Wish/need for counseling	
<ul style="list-style-type: none"> <li>- Clarification on how we will write our theory section, see previous mail for more info</li> </ul>	
Approval/signature group leader	Signature other group participants
Magnus Øye	Petter Drønne

1) Note here brief feedback on the number of meetings - by type (internal, steering group, meeting with supervisor) - during this reporting period

<b>Bachelor Thesis: IE303612</b>	Project Autonomous inspections for process industry by a quadraped robot with use of neural networks	Number of meeting this period 1). 0 planned	Firm - Contracting NTNU in Ålesund	Page 2 av 2
<b>Progress report</b>	Period/week(s) 2	Number of hours this period. (from log) Approx. 200	Project group (name) Petter Drønne Vegard Solheim Magnus Øye	Dato 12.04.20

	Vegard Solheim
--	----------------

1) Note here brief feedback on the number of meetings - by type (internal, steering group, meeting with supervisor) - during this reporting period

**A.4.7 Progress report 19.04.20**

<b>Bachelor Thesis: IE303612</b>	Project Autonomous inspections for process industry by a quadraped robot with use of neural networks	Number of meeting this period 1). 1 planned	Firm - Contracting NTNU in Ålesund	Page 1 av 2
<b>Progress report</b>	Period/week(s) 1	Number of hours this period. (from log) Approx. 90	Project group (name) Petter Drønne Vegard Solheim Magnus Øye	Dato 19.04.20

Main goal/purpose for this periods work	
<ul style="list-style-type: none"> <li>- Physical testing</li> <li>- Path planning and collision detection testing</li> <li>- Utilize waypoint system</li> <li>- Optimize CNN and add more objects</li> </ul>	
Planned activities this period	
<ul style="list-style-type: none"> <li>- Improve movement on the robot</li> <li>- Follow given waypoint path</li> <li>- Test autonomous system</li> </ul>	
Actually conducted activities this period	
<ul style="list-style-type: none"> <li>- The USB on the teensy broke, so one day went for fixing this</li> <li>- Robot is walking in an almost straight line and are turning left/right on his own</li> <li>- Robot is adjusting speed and turning rate with input from navigation planner</li> <li>- Making report ready for a status overview</li> <li>- Created SQL Database</li> <li>- Created REST API connecting GUI, CNN and Database</li> <li>- Added OCR for reading tag on equipment</li> </ul>	
Description of/ justification for potential deviation between planned and real activities	
-	
Description of/ justification for changes that is desired in the projects content or in the further plan of action – or progress report	
-	
Main experience from this period	
<ul style="list-style-type: none"> <li>- We are now pleased with the manual performance</li> <li>- Fire extinguishers and exit signs will be added to the inspection task</li> </ul>	
Main purpose/focus next period	
<ul style="list-style-type: none"> <li>- Focus on autonomous walking</li> <li>- Optimize CNN and train for the new objects</li> <li>- Use image processing for extracting information out of manometers</li> </ul>	
Planned activities next period	
<ul style="list-style-type: none"> <li>- Follow given waypoint path</li> <li>- Inspect a given area, updating Database and GUI</li> <li>- Walking straight and turning need optimizing</li> </ul>	
Other	
<ul style="list-style-type: none"> <li>- Because of the corona virus the group is no longer on school.</li> <li>- All group members meet in the house of one of the members</li> <li>- Supervisors wanted a meeting 22. April, therefore only one-week period. This may be the standard forward since we are reaching the end.</li> <li>- Progress video → None this week</li> </ul>	
Wish/need for counseling	
<ul style="list-style-type: none"> <li>- We will submit a draft of our report and have a discussion regarding it on the next meeting.</li> </ul>	
Approval/signature group leader	Signature other group participants
Magnus Øye	Petter Drønne

1) Note here brief feedback on the number of meetings - by type (internal, steering group, meeting with supervisor) - during this reporting period

<b>Bachelor Thesis: IE303612</b>	Project Autonomous inspections for process industry by a quadraped robot with use of neural networks	Number of meeting this period 1). 1 planned	Firm - Contracting NTNU in Ålesund	Page 2 av 2
<b>Progress report</b>	Period/week(s) 1	Number of hours this period. (from log) Approx. 90	Project group (name) Petter Drønne Vegard Solheim Magnus Øye	Dato 19.04.20

	Vegard Solheim
--	----------------

1) Note here brief feedback on the number of meetings - by type (internal, steering group, meeting with supervisor) - during this reporting period



**A.4.8 Progress report 03.05.20**

<b>Bachelor Thesis: IE303612</b>	Project Autonomous inspections for process industry by a quadraped robot with use of neural networks	Number of meeting this period 1). 1 planned	Firm - Contracting NTNU in Ålesund	Page 1 av 1
<b>Progress report</b>	Period/week(s) 1	Number of hours this period. (from log) Approx. 200	Project group (name) Petter Drønne Vegard Solheim Magnus Øye	Dato 03.05.20

Main goal/purpose for this periods work	
<ul style="list-style-type: none"> <li>- Focus on autonomous walking</li> <li>- Optimize CNN and train for the new objects</li> <li>- Use image processing for extracting information out of manometers</li> </ul>	
Planned activities this period	
<ul style="list-style-type: none"> <li>- Follow given waypoint path</li> <li>- Inspect a given area, updating Database and GUI</li> <li>- Walking straight and turning need optimizing</li> </ul>	
Actually conducted activities this period	
<ul style="list-style-type: none"> <li>- Following a given waypoint path works</li> <li>- Inspecting an area will update the database and GUI</li> <li>- After optimization CNN has a mAP and mAR close to 1.0</li> <li>- Preprocessing images for better OCR</li> <li>- Robust implementation of image processing for gauges</li> <li>- Big changes to the GUI</li> <li>- Full demo and testing were done</li> </ul>	
Description of/ justification for potential deviation between planned and real activities	
-	
Description of/ justification for changes that is desired in the projects content or in the further plan of action – or progress report	
-	
Main experience from this period	
<ul style="list-style-type: none"> <li>- We are now pleased with the overall performance, therefore the rest of the time left will be used for writing report</li> </ul>	
Main purpose/focus next period	
<ul style="list-style-type: none"> <li>- Report</li> </ul>	
Planned activities next period	
<ul style="list-style-type: none"> <li>- Write report</li> </ul>	
Other	
<ul style="list-style-type: none"> <li>- Because of the corona virus the group is no longer on school.</li> <li>- All group members meet in the house of one of the members</li> <li>- Progress video → None this week</li> <li>- Attachments: Diagrams for report</li> </ul>	
Wish/need for counseling	
-	
Approval/signature group leader	Signature other group participants
Magnus Øye	Petter Drønne Vegard Solheim

1) Note here brief feedback on the number of meetings - by type (internal, steering group, meeting with supervisor) - during this reporting period

**A.4.9 Progress report 19.05.20**

<b>Bachelor Thesis: IE303612</b>	Project Autonomous inspections for process industry by a quadruped robot with use of neural networks	Number of meeting this period 1). 0 planned	Firm - Contracting NTNU in Ålesund	Page 1 av 1
<b>Progress report</b>	Period/week(s) 2,5	Number of hours this period. (from log) Approx. 250	Project group (name) Petter Drønnen Vegard Solheim Magnus Øye	Dato 19.05.20

Main goal/purpose for this periods work - <b>Report</b>	
Planned activities this period - <b>Write report</b>	
Actually conducted activities this period - <b>Report is now finished and ready for submission</b>	
Description of/ justification for potential deviation between planned and real activities -	
Description of/ justification for changes that is desired in the projects content or in the further plan of action – or progress report -	
Main experience from this period - <b>We are now pleased with the report</b>	
Main purpose/focus next period -	
Planned activities next period -	
Other - <b>This will be the last progress report</b>	
Wish/need for counseling -	
Approval/signature group leader <b>Magnus Øye</b>	Signature other group participants <b>Petter Drønnen</b> <b>Vegard Solheim</b>

1) Note here brief feedback on the number of meetings - by type (internal, steering group, meeting with supervisor) - during this reporting period

## **A.5 Meeting reports**

### **A.5.1 Meeting report 08.01.20**

# Meeting

<i>Date:</i> 08.01.2020	<i>Place:</i> NTNU, Campus Ålesund, Rom L160
<i>Time:</i> 12:00	<i>Meeting nr.:</i> 1

<i>Invited:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Magnus Øye Petter Drønnen	<i>Attended:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Magnus Øye Petter Drønnen
---	--

This is an orientation meeting with supervisors and participants of a bachelor project spring of 2020. The supervisors have been explained what the goal of this project is and which direction the group wishes to take.

---

## 1 Bachelor thesis

### Agenda item:

Give questions and discuss how the bachelor should be conducted.

### Conclusion:

#### 1. Regarding the Requirement specification:

If parts of the bachelor changes on the way it should still be written in the report if it can be reasoned and documented why the change happened. Even though some task do not lead anywhere there is still time spent and it should be taken into account.

#### 2. Judgement of the thesis:

The supervisors have great influence on how the thesis is judged but the final grade is mainly given by the sensor.

#### 3. Involvement of the supervisors:

The supervisors will give advice and will answer question the group gives. If questions cannot be answered the supervisors will communicate these further.

**4. Title of thesis:**

The title of the thesis should be specific so that the work does not get too broad and little innovative.

**5. Reason for choosing a quadruped robot as vehicle:**

A reason for choosing a quadruped robot should be given and what areas of application it is to be used within. The reasoning should partly reflect upon research materials.

**6. Thesis funding:**

The group should directly contact program manager Webjørn Rekdalsbakken to ask for project funding as this thesis will supersede the ordinary budget limit.

---

### **3 ADDITIONAL ITEMS**

- Plan time for next meeting  
Meetings should be biweekly on Wednesdays at 12:00 pm.
- Meeting reports: The group will write meeting reports for every meeting and send these to the supervisors for approval. These reports will be important for tracking decisions.
- The group should do some research on what similar type of work have been done by other in the past and look to the industry and see what might be needed.

---

**A.5.2 Meeting report 14.01.20**



# Meeting Report

<i>Date:</i> 14.01.2020	<i>Place:</i> NTNU, Campus Ålesund, Rom L160
<i>Time:</i> 12:00	<i>Meeting nr.:</i> 2

<i>Invited:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Magnus Øye Petter Drønnen	<i>Attended:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Magnus Øye Petter Drønnen
---	--

---

## 1 APPROVAL OF NOTICE

**Agenda item:**

Approve the notice

**Conclusion:**

---

## 2 REVIEW OF PROGRESS

**Agenda item:**

Discussing and approving the progress.

**Discussion & Conclusion:**

Supervisors have given feedback regarding the pre project. The group have worked with changes tied to this and additionally specifying the thesis.

Supervisors is content with the measures taken to concretize the thesis and that process industry is the chosen area of application. The thesis will mainly be about using a quadruped robot to conduct

inspections in a process environment by using neural networks. The group have reasoned why a quadruped robot should be used and will incorporate this into the pre project where supporting literature on the field will be referred to.

The area of responsibility has been restated and further specified and is now aiming towards that a single group member do the system integration is his assigned field and the responsibility to perform sub task is divided by the members assigned to the respective tasks.

---

### **3 ADDITIONAL ITEMS**

- Plan time for next meeting  
Next meeting is set for February 5<sup>th</sup> at 12pm.
-

**A.5.3 Meeting report 05.02.20**

# Meeting Report

<i>Date:</i> 05.02.2020	<i>Place:</i> NTNU, Campus Ålesund, Rom L160
<i>Time:</i> 12:00	<i>Meeting nr.:</i> 3

<i>Invited:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Magnus Øye Petter Drønnen	<i>Attended:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Magnus Øye Petter Drønnen
---	--

---

## 1 APPROVAL OF NOTICE

For coming meetings, the agenda for the meeting and the progress report should be sent along with the notice.

---

## 2 REVIEW OF PROGRESS

**Agenda item:**

Discussing and approving the progress.

**Discussion:**

The progress was elaborated to the supervisors.

**Conclusion:**

No comments on the progress.

---

## 3 ADDITIONAL ITEMS

- The group should find companies that could benefit from the project or target the project towards a company's current need.
  - Next meeting is set for Wednesday 12:00, February 19<sup>th</sup>.
-

**A.5.4 Meeting report 19.02.20**

# Meeting Report

<i>Date:</i> 19.02.2020	<i>Place:</i> NTNU, Campus Ålesund, Rom L160
<i>Time:</i> 12:00	<i>Meeting nr.:</i> 4

<i>Invited:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Magnus Øye Petter Drønnen	<i>Attended:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Petter Drønnen
---	--

---

## 1 APPROVAL OF NOTICE

### Agenda item:

Approve the notice.

### Conclusion:

Meeting notice should be sent two days prior to the meeting.

---

## 2 REVIEW OF PROGRESS

### Agenda item:

Discussing and approving the progress.

### Discussion:

Progress to assembly and software has been done but will soon halt due to motors have not been shipped yet. Anders have been asked several times to contact the supplier to try and expedite the shipping but have not done so.

### Conclusion:

Ivar will contact Webjørn Rekdalsbakken about speeding up Anders's process with contacting the supplier. Meanwhile, we should be looking for alternative suppliers that can provide the motors at an earlier date.

---

### **3 ADDITIONAL ITEMS**

- Plan time for next meeting

**Next meeting is scheduled for March 4<sup>th</sup>.**

- Aleksander suggested using a Real Time Kernel for the Jetson Nano.
  - Suggestion about looking into threading functionality (Pthread\_schedparams) and using priority when scheduling tasks in the microcontroller.
-



**A.5.5 Meeting report 04.03.20**

# Meeting Report

<i>Date:</i> 04.03.2020	<i>Place:</i> NTNU, Campus Ålesund, Rom L160
<i>Time:</i> 12:00	<i>Meeting nr.:</i> 5

<i>Invited:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Magnus Øye Petter Drønnen	<i>Attended:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Magnus Øye Petter Drønnen
---	--

---

## 1 APPROVAL OF NOTICE

**Agenda item:**

Approve the notice.

**Conclusion:**

Meeting notice is approved.

---

## 2 REVIEW OF PROGRESS

**Agenda item:**

Discussing and approving the progress.

**Discussion:**

**Conclusion:**

Progress was reviewed.

---

### **3 ADDITIONAL ITEMS**

- Plan time for next meeting  
Next meeting is set for March 18<sup>th</sup>.
-

**A.5.6 Meeting report 15.04.20**

# Meeting Report

<i>Date:</i> 15.04.2020	<i>Place:</i> Skype video conference
<i>Time:</i> 12:00	<i>Meeting nr.:</i> 6

<i>Invited:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Magnus Øye Petter Drønnen	<i>Attended:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Magnus Øye Petter Drønnen
---	--

-----

## 1 APPROVAL OF NOTICE

### Agenda item:

Approve the notice

### Conclusion:

-----

## 2 REVIEW OF PROGRESS

### Agenda item:

Discussing and approving the progress.

Discuss how the current situation is affecting the progress.

### Discussion:

### Conclusion:

Approved.

---

### **3 Clarification on writing report**

**Agenda item:**

As discussed in the mail correspondence, we would like to get some clarifications on how to write our theory section in the report.

**Discussion:**

**Conclusion:**

We will submit a draft of our report and have a discussion regarding it on the next meeting.

---

### **4 ADDITIONAL ITEMS**

- Plan time for next meeting  
Next meeting is set for Wednesday 12 pm on April 22<sup>nd</sup>
-

**A.5.7 Meeting report 23.04.20**

# Meeting

<i>Date:</i> 23.04.2020	<i>Place:</i> Skype video conference
<i>Time:</i> 12:00	<i>Meeting nr.:</i> 7

<i>Invited:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Magnus Øye Petter Drønnen	<i>Attended:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Magnus Øye Petter Drønnen
---	--

---

## 1 APPROVAL OF NOTICE

### Agenda item:

Approve the notice

### Conclusion:

Approved.

---

## 2 REVIEW OF PROGRESS

### Agenda item:

Discussing and approving the progress.

### Discussion:

Elaboration was made on how GUI is built, what colors in maps should be interpreted, how Dijkstra's algorithm is used in our navigation,



---

### 3 Clarification on writing report

#### Agenda item:

Supervisor should have read the submitted draft for the report and should give input on the structure and content.

We have provided links below to previous reports that have gotten good critics and the supervisors should maybe also look at these.

<https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2415146>

<https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2499673>

<https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2564165>

<https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2559484>

<https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2610898>

#### Conclusion:

- Ivar suggested that we can use the structure we have but also use the official templates and divide our chapters within the template chapters.
- Approach heading can be replaced by Method
- Be more specific regarding the chapters. E.g design could be software design and hardware design.
- Formulas and theory should mainly be placed under theory.
- Have a more obvious separation between method and result.
- Look to Bloom's taxonomy – keeping the red thread
- Sort out content in the report where it logically belongs. Remove subjective writing, this should only appear in the discussion chapter. Use of subjective words like "we" and "the group" is discouraged.
- A report should be objective, formal.
- There should be no need to explain base knowledge like TCP and UDP unless it is relevant for why TCP is chosen over UDP.
- Be more concise and precise when writing. Avoid meaningless words like variables.
- Abbreviations like IMU, and words like Feature can be added to the terms
- Theory 2.17 - Machine Learning - a very nice chapter.
- "2.16" Dijkstra – Road map approach – Elaborate more what this really is.
- "Jetson Nano will operate as our main controller." Be strict with which tense is written, past present; future what will it be?

- Don't use words like professional finish. Other words can be more describing e.g. maintainable, better coupled, more robust, reliable etc.
- On every image we use that we have not made our self, we should refer to the source in the caption. E.g Wikipedia.com
- "2.13" SLAM – can be moved to method. But if we like to elaborate more it can stay in the theory section.
- "2.2" - Linear Motion – more specific on belt used in power transmission between motors and legs.
- The first paragraph beneath "Electric motor" section heading can be discarded. The same applies for "Communication Protocols".

things to clarify:

The use of word "kinematic" should be more specific.

---

## **4 ADDITIONAL ITEMS**

- Plan time for next meeting  
Meeting notice will be sent when testing and more writing on report have been done.
  - Aleksander should contact lab assistant Øyvind regarding use of school space for testing from the coming Monday.
-

**A.5.8 Meeting report 07.05.20**

# Meeting Report

<i>Date:</i> 07.05.2020	<i>Place:</i> Skype video conference
<i>Time:</i> 12:00	<i>Meeting nr.:</i> 8

<i>Invited:</i>  Ivar Blindheim Aleksander Larsen Skrede Vegard Solheim Magnus Øye Petter Drønnen	<i>Attended:</i>  Ivar Blindheim Aleksander Larsen Skrede Magnus Øye Petter Drønnen
---	--

---

## 1 APPROVAL OF NOTICE

**Agenda item:**

Approve the notice

**Conclusion:**

Agenda approved.

---

## 2 REVIEW OF PROGRESS

**Agenda item:**

Discussing and approving the progress.

**Conclusion:**

Progress approved.

---

### **3 ADDITIONAL ITEMS**

- Plan time for next meeting
  - Supervisors gave tips for the thesis presentation video which will be presented on May 20<sup>th</sup>.
  - Supervisors should receive the thesis for review May 13<sup>th</sup>.
-

# **Appendix B**

## **Bill of material (BOM)**

### **B.1 BOM**

Robot Assembly			
Item	Quantity	Comment	Additional info
Frame Assembly	1		
Electronics Plate Assembly	1		
Cameras Assembly	1		
Coaxial Hub Assembly	4		
Leg Assembly	4		
Control Panel Assembly	1		
Odrive Assembly	4		

Frame Assembly			
Item	Quantity	Comment	Additional info
Front Bumper	1	Laser Cut & Press Braked	1.5mm Aluminium
Rear Bumper	1	Laser Cut & Press Braked	1.5mm Aluminium
Side Plate	2	Laser Cut	4mm Plexi Glas
Top Cover Part 1	1	3D Printed	
Top Cover Part 2	1	3D Printed	
Dowel pin 4x18mm DIN7	4		
Quick Release Cotter Pin 4x20mm	2		

Electronics Plate Assembly			
Item	Quantity	Comment	Additional info
Electronics Plate 8mm Plexi Glas	1	Laser cut	
Electronics Plate Bracket	4	3D Printed	
Power Distribution Hub	1		
Power Distribution Hub Casing	1	3D Printed	
Jetson Nano Assembly	1		
Teensy 4.0 Assembly	1		
Battery Holder Standoffs	8	3D Printed	
Battery Holder	2	3D Printed	
Cable Harness Clip	2	3D Printed	
Cable Harness Clip 2	2	3D Printed	
Zip Ties	50	Max 3mm wide	
12AWG Red & Black Silicone wires	5m		
XT60H Female	8		
XT60H Male	9		
XT60BP Male	4	For PCB Solder	
Socket head cap M3x12 DIN912	12		
Shim Rings for M3 DIN988	12		

Cameras Assembly			
Item	Quantity	Comment	Additional info
Intel RealSense T265	1		
Intel RealSense D4435	1		
USB type micro B super speed to USB type A 3.0 cable	1		
USB type C to USB type A 3.0 cable	1		
Camera Mount	1	3D Printed	
Camera Frame	1	3D Printed	
Socket head cap M3x12 DIN912	4		

Leg Assembly			
Item	Quantity	Comment	Additional info
Upper Inner Leg	1	Water jet, Prodex	10mm Aluminium
Upper Outer leg	1	Water jet, Prodex	10mm Aluminium
Bottom Leg	1	Water jet, Prodex	10mm Aluminium
Bottom Leg Feet	1	Water jet, Prodex	10mm Aluminium
Rubber Sole	1	3D Printed	
Deep Grove Ball Bearing 5x10x4	6		
Shoulder Screw m4 5x10mm DIN7379	3		
Set Screw M4x8 DIN916	4		

Coaxial Hub Assembly			
Item	Quantity	Comment	Additional info
Inner shaft	1	Machined, NTNU POD	
Outer shaft	1	Machined, NTNU POD	
Coaxial Hub	1	3D Printed	
Outer Shaft Spacer	1	3D Printed	
Locking Nut M3 DIN985	4		
Deep Grove Ball Bearing 20x27x4 DIN620	2		
Deep Grove Ball Bearing 12x18x4 DIN620	2		
Deep Grove Ball Bearing 4x10x4 DIN620	2		
Set Screw Dog Point M4x8mm DIN915	4		
Dowel Pin 4x18mm DIN7	1		
Pulley Brace	1	3D Printed	
Pulley Inner Shaft 48T	1	3D Printed	
Pulley Outer Shaft 48T	1	3D Printed	
Pulley Short 16T	1	3D Printed	
Pulley Long 16T	1	3D Printed	
Pulley Spacer	2	3D Printed	
Countersunk Hex Screw M3x6 DIN7991	10		
210-GT3-6 Timing Belt	2		
Outer Encoder Mount	1	3D Printed	
Inner Encoder Mount	1	3D Printed	
T-motor MN4812 Brushless DC motor	2		
Banana plug 3.5mm Male	6		
Shoulder screw M3 4x6mm DIN7379	2		
Button Head Screw M3x16mm DIN7380	3		
Button Head Screw M2.5x10mm DIN7380	8		
Socket Head Cap Screw M3x18 DIN912	3		

Control Panel Assembly			
Item	Quantity	Comment	Additional info
Double sided prototype print board 4x6cm	1		
Flat ribbon cable PCB connector 2.54 2x10 pin	1		
Flat ribbon cable plug 2.54 2x10 pin	1		
Push button w/LED Green	1		



Push Button w/LED Blue	1
Push Button w/LED Red	1
Push Button w/LED Orange	1
Push Button Enclosure top	1 3D Printed
Enclosure Spacer	1 3D Printed
PCB retainer	1 3D Printed
Socket Head Cap Screw M3x25 DIN912	4

Jetson Nano Assembly			
Item	Quantity	Comment	Additional info
Jetson Nano	1		
Jetson Nano M2 WiFi	1		
WiFi Antenna	1		
Jetson Nano Enclosure Top	1	3D Printed	
Jetson Nano Enclosure Bottom	1	3D Printed	
Jetson Nano Enclosure Standoffs	4	3D Printed	
Socket Head Cap Screw M3x12 DIN912	8		
Shim Rings for M3 DIN988	4		

Teensy Assembly			
Item	Quantity	Comment	Additional info
Teensy 4.0 microcontroller	1		
Double sided prototype print board 4x6cm	1		
Flat ribbon cable PCB connector 2.54 2x10 pin	2		
Flat ribbon cable plug 2.54 2x10pin	2		
header pins female 2.54 1x20	1		
header pin male 2.54 1x20	1		
Flat ribbon cable	1m		
Teensy enclosure lid	1	3D Printed	
Teensy enclosure bottom	1	3D Printed	
Socket head cap M3x12 DIN912	4		
Shim Rings for M3 DIN988	4		

Odrive Assembly			
Item	Quantity	Comment	Additional info
Odrive V3.6 56V No Connectors	1		
TFT Female Header pins 2.54 1x20	2		
XT60PB Male	1		
Standoffs	4	Original Odrive Part	
Nut M3 DIN934	4		
Banana plugs 3.5mm Female	3		

Stand Assembly			
Item	Quantity	Comment	Additional info
Bottom Frame	1	Laser Cut	8mm Plexi Glas
Front	2	Laser Cut	8mm Plexi Glas
Side	2	Laser Cut	8mm Plexi Glas
Splint	4	Laser Cut	8mm Plexi Glas

# Appendix C

## Pictures and progress videos

### C.1 Pictures

Pictures from development can be found here [Pictures](#).

<https://drive.google.com/open?id=175Xf5eR7ISFx6bbiMSzkLT8xGgaTwL8Q>

### C.2 Progress videos

Same progress videos as in the progress reports:

1. [Progress video one](#)
2. [Progress video two](#)
3. [Progress video three](#)
4. [Progress video four](#)

# Appendix D

## Drawings

### D.1 Mechanical drawings

The mechanical drawings such as the 3D-model can be found here [Fusion 360 CAD model](https://drive.google.com/drive/folders/1SjDRSZ7dCbtEyQgJi2F-tcV4agTJ8wo_)  
[https://drive.google.com/drive/folders/1SjDRSZ7dCbtEyQgJi2F-tcV4agTJ8wo\\_](https://drive.google.com/drive/folders/1SjDRSZ7dCbtEyQgJi2F-tcV4agTJ8wo_)

### D.2 Electrical drawings

Skoleversion

**Bachelor**

# Bachelor

Component list

**1**

Overview

**2**

Control power

**3**

Jetson Nano

**4**

Teensy

**5**

Odrive

**6**

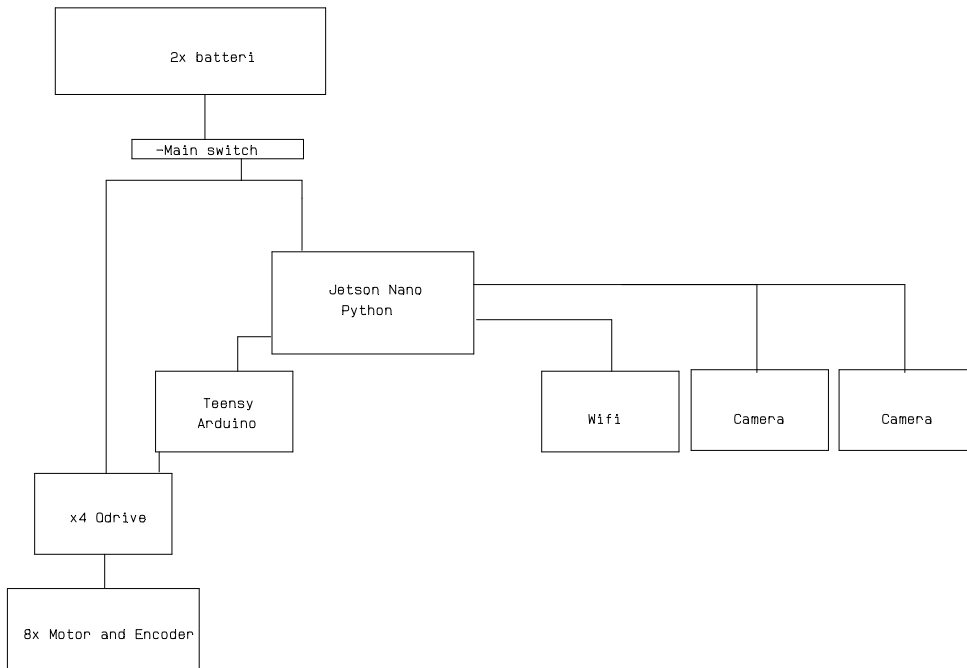
**7**

**8**

**9**

**10**

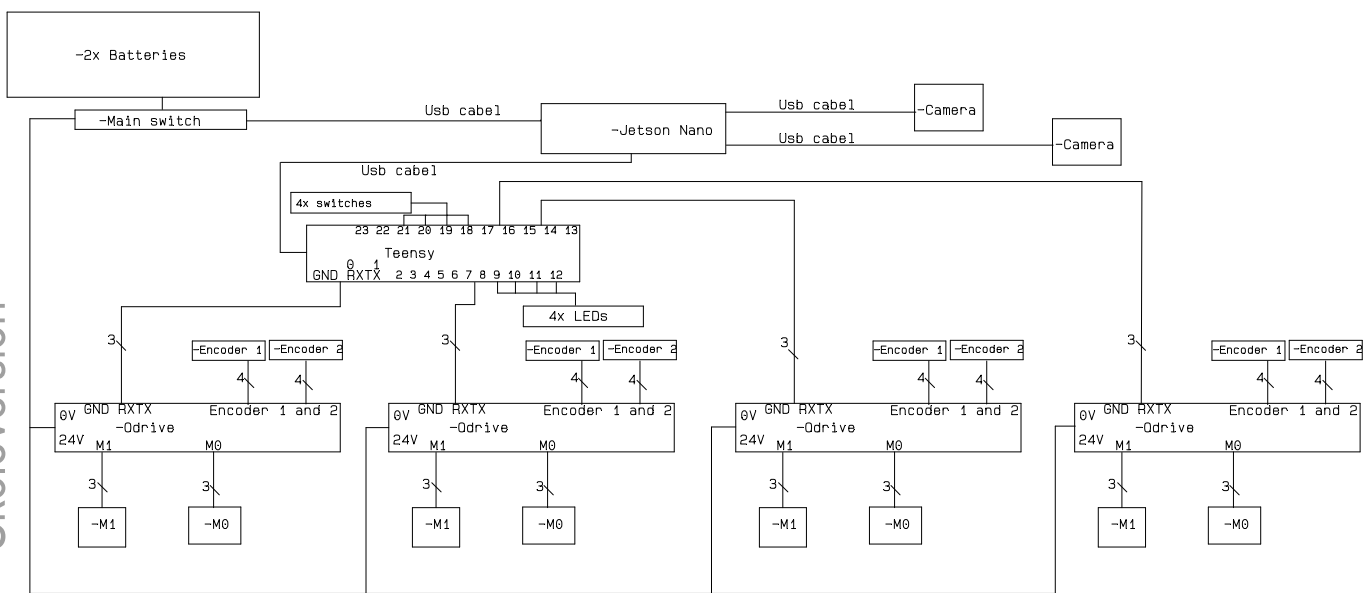




PCSHEMATIC Automation

Project title: Bachelor	Case no.:	Project rev.:	Page
Customer: NTNU	DCC:		2
Page title: Overview	Dwg. no.:	Page rev.:	Scale: 1:1
File name: Bachelor	Eng. (proj/page): Petter	Last print: 01.03.2020	Previous page: 1
Page ref.:	Appr. (date/init):	Last edit: 19.02.2020	Next page: 3
			Total no. of pages: 8

Skoleversion

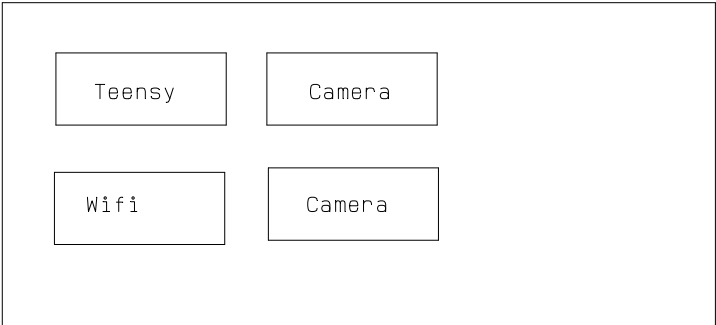


PCSHEMATIC Automation

Project title: Bachelor	Case no.:	Project rev.:	Page	3
Customer: NTNU	DCC:		Scale:	1:1
Page title: Control power	Dwg. no.:	Page rev.:	Previous page:	2
File name: Bachelor	Eng. (proj/page): Petter	Last print: 01.03.2020	Next page:	4
Page ref.:	Appr. (date/init):	Last edit: 01.03.2020	Total no. of pages:	8

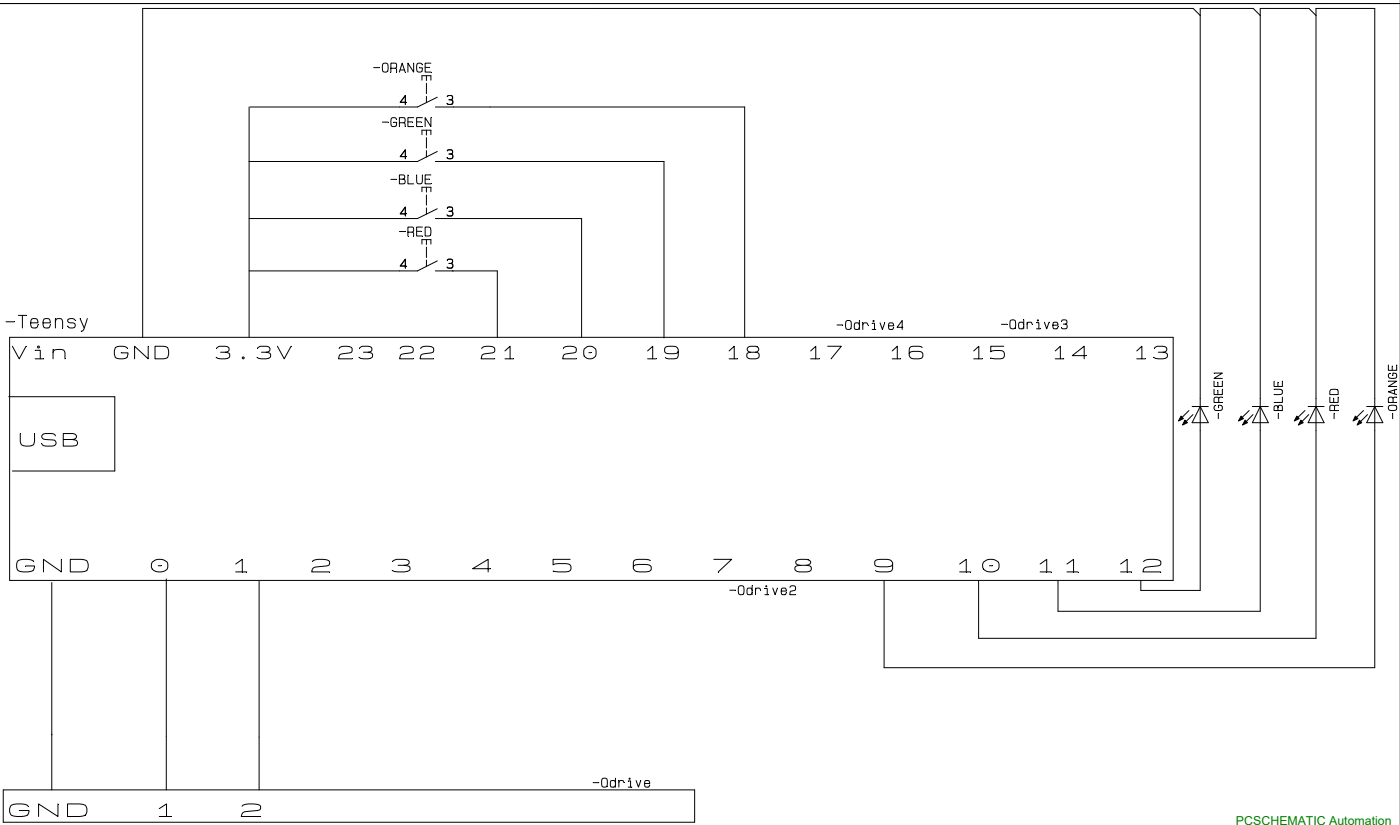


Jetson Nano



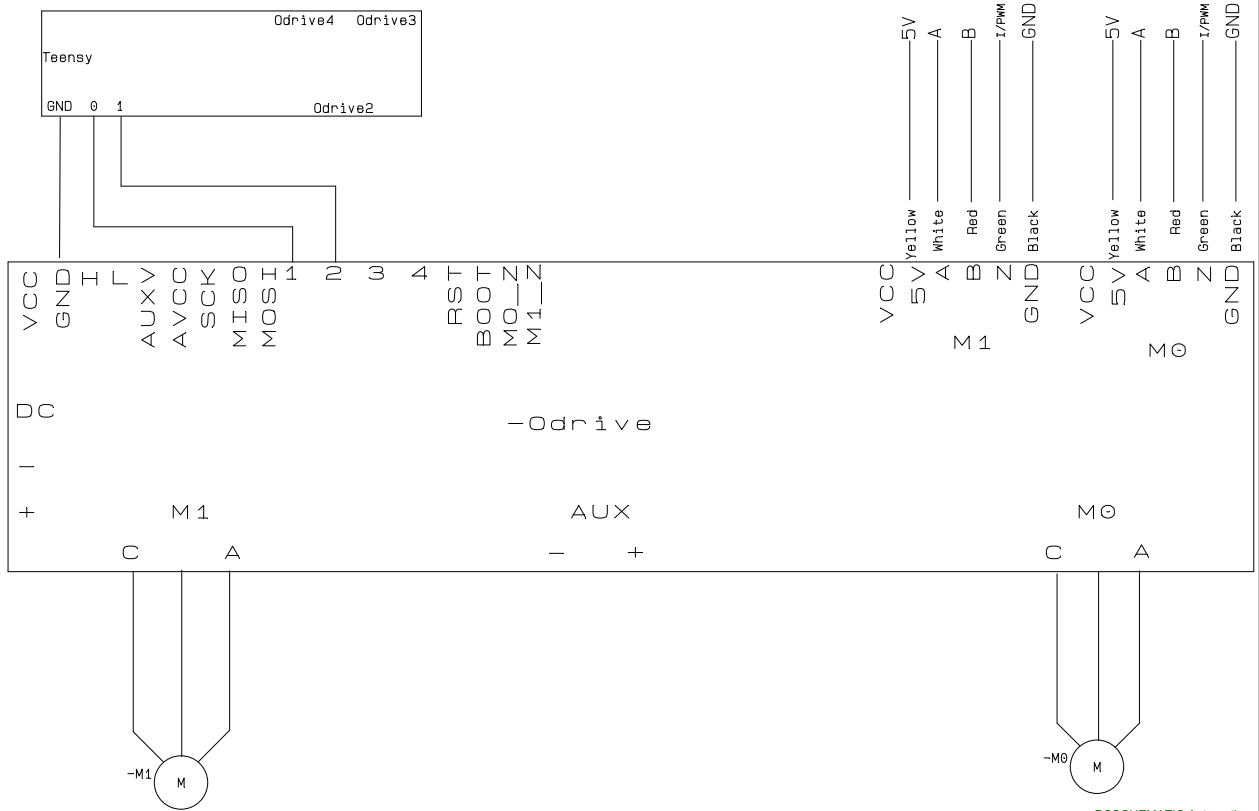
Project title: Bachelor	Case no.:	Project rev.:	Page	4
Customer: NTNU	DCC:		Scale:	1:1
Page title: Jetson Nano	Dwg. no.:	Page rev.:	Previous page:	3
File name: Bachelor	Eng. (proj/page): Petter	Last print: 01.03.2020	Next page:	5
Page ref.:	Appr. (date/init):	Last edit: 01.03.2020	Total no. of pages:	8

Skoleversion



PCSHEMATIC Automation

Project title: Bachelor	Case no.:	Project rev.:	Page
Customer: NTNU	DCC:		5
Page title: Teensy	Dwg. no.:	Page rev.:	Scale: 1:1
File name: Bachelor	Eng. (proj/page): Petter	Last print: 01.03.2020	Previous page: 4
Page ref.:	Appr. (date/init):	Last edit: 01.03.2020	Next page: 6
			Total no. of pages: 8



Project title: Bachelor	Case no.:	Project rev.:	Page
Customer: NTNU	DCC:		6
Page title: Odrive	Dwg. no.:	Page rev.:	Scale: 1:1
File name: Bachelor	Eng. (proj/page): Petter	Last print: 01.03.2020	Previous page: 5
Page ref.:	Appr. (date/init):	Last edit: 01.03.2020	Next page:
			Total no. of pages: 8

# Appendix E

## Source Code

### E.1 Source Code

The source code can be found here [Github](https://github.com/magnusoy/Sparkie)  
<https://github.com/magnusoy/Sparkie>