

Johan-Henrik Fylling & Kevin Mentzoni
Halvarsson

What is needed to make an indoor autonomous robot system valuable?

Master's thesis in Computer Science
Supervisor: George Adrian Stoica
Co-supervisor: Terje Røsand
June 2022

Johan-Henrik Fylling & Kevin Mentzoni Halvarsson

What is needed to make an indoor autonomous robot system valuable?

Master's thesis in Computer Science
Supervisor: George Adrian Stoica
Co-supervisor: Terje Røsand
June 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Abstract

In this project and a preliminary project, we have built an autonomous rover-like service robot capable of mapping and navigating indoors, a task management system, and a human-robot interface. The hardware used is cheap off-the-shelf parts, and any external software used is open source. The main control components of the robot are a Raspberry Pi 4, two Arduino Nano Every, and an RPLIDAR-A1. The software stack is mainly based on Robotics Operating System 2 Galactic and React.

The goal was to contribute to robotics by learning how robots can be useful by finding tasks they can help with in building operations and discovering what the requirements of such a system would be. We are not experienced with robotics, and a large part of the project has been about learning about state of the art in robotics with a learning-by-doing methodology. To robotics experts, it might be interesting to see how software engineers approach the development of robotic systems.

We have a background in computer science and are therefore experienced with user-oriented development, which has been the main research method. Useful tasks were elicited through interviews and the prototype was evaluated using usability tests. The general feedback was that the system would be useful and that it had great potential. The results of this project are a report containing information about how one can build autonomous robots with Robotics Operating System 2, some tasks for robots that can potentially be useful to janitors, requirements for indoor robot systems, technical solutions, and suggestions for further research.

Sammen drag

I dette og et innledende prosjekt har vi bygd en autonom servicerobot som kan kartlegge omgivelsene sine og navigere innendørs, et oppgavestyringssystem, og et menneske-robot grensesnitt. Maskinvaren som er brukt er billig hyllevare, og all ekstern programvare er åpen kildekode. Hovedkontrolldelene består av en Raspberry Pi 4, to Arduino Nano Every, og en RPLIDAR-A1. Software-stacken er hovedsaklig basert på ROS 2 Galactic og React.

Målet med dette prosjektet var å bidra til fagfeltet robotikk ved å finne ut hvordan roboter kan være nyttig ved å finne oppgaver de kan gjøre innen drift og vedlikehold av bygg, og finne ut hva kravene til et slikt system ville vært. Vi har ingen erfaring med robotikk fra tidligere, og en stor del av prosjektet har derfor vært å bli kjent med fagfeltet som det er idag ved en "learning-by-doing"-metodikk. For eksperter innen robotikk kan det være interessant å lese om to dataingeniørers tilnærming til utviklingen av et robot-basert system.

Vi har en bakgrunn innen brukerorientert systemutvikling, som har vært vår metodikk for å gjøre forskning. Nyttige oppgaver ble funnet gjennom brukerintervju og prototypen av systemet ble evaluert gjennom brukertester. Den generelle tilbakemeldingen var at systemet var nyttig og at det hadde potensiale. Resultatet av prosjektet er en rapport som inneholder informasjon om hvordan man bygge en autonom robot basert på ROS 2, en rekke oppgaver som kunne ha vært nyttig for en robot å hjelpe vaktmestre med, krav til et innendørs robotsystem, tekniske løsninger, og forslag til hvordan veien kan se ut om man skal fortsette med prosjektet.

Acknowledgements

The project was made possible by IDI and the ISSE Group, providing the required hardware and gift cards as a symbolic thank you to the test subjects. We would like to thank our supervisor and co-supervisor George Adrian Stoica and Terje Røsand for their advice and guidance in this project. We would also like to thank all our participants for taking part in interviews and usability tests.

A special thanks goes to Tiril Pettersen and Fredrik Monsen who partook in practice-runs of the interviews and the usability tests.

We would also like to thank our friends, family, and girlfriends for motivation and support during the course of the project.

Contents

Abstract	iii
Sammendrag	iv
Acknowledgements	v
Contents	vi
Figures	x
Tables	xiii
Acronyms	xiv
Glossary	xv
1 Introduction	1
1.1 Research questions	1
2 Background and theory	3
2.1 System development as a research method	3
2.2 Robot technology	4
2.2.1 Arduino	4
2.2.2 Motor Control	5
2.2.3 PID controller	5
2.2.4 Classification of robots	5
2.2.5 Autonomy in Robots	6
2.2.6 Raspberry Pi	7
2.2.7 LIDAR	7
2.3 Situational Awareness	8
2.4 ROS	8
2.4.1 ROS Computation Graph	8
2.4.2 ROS Interfaces	9
2.4.3 Nodes	9
2.4.4 ROS Topics	9
2.4.5 ROS Services	10
2.4.6 ROS Actions	10
2.4.7 Security	10
2.4.8 ROS and frontend	12
2.5 Indoors Navigation	12
2.5.1 Nav2	12
2.5.2 Pose	13

- 2.5.3 Point clouds 13
- 2.5.4 SLAM 13
- 2.5.5 Path planning 14
- 2.5.6 Costmaps 14
- 2.6 Interview & Observation 14
- 2.7 Analyzing Qualitative Data 15
- 2.8 Human Centered Design 15
 - 2.8.1 Usability tests 16
- 2.9 Navigating indoors 16
- 2.10 Interfacing interactions between robots and humans 16
 - 2.10.1 Designing a graphic user interface for interactions with robots 17
 - 2.10.2 Manually steering a robot 18
- 2.11 TAM 20
- 3 Research Method 22**
 - 3.1 Process 22
 - 3.2 Literature Review 27
 - 3.2.1 Searching 27
 - 3.2.2 Obtaining 28
 - 3.2.3 Assessing 28
 - 3.2.4 Reading 28
 - 3.2.5 Critically Evaluating 28
 - 3.3 Interviews and observation 28
 - 3.3.1 Interviews 28
 - 3.3.2 Group Interview 29
 - 3.3.3 Observation 29
 - 3.4 Usability testing 30
 - 3.5 Data Analysis 30
 - 3.5.1 Interviews 30
 - 3.5.2 User tests 31
 - 3.5.3 Observation 32
 - 3.6 Participants 32
 - 3.6.1 Interviews 32
 - 3.6.2 User test 32
 - 3.7 Ethics, Participants Rights, & the collection and storing of data . . . 33
 - 3.7.1 Ethics 33
 - 3.7.2 Participants Rights 33
 - 3.7.3 Collecting and Storing Data 33
- 4 Related work 35**
 - 4.1 Automated Guided Vehicles (Service Robots) 35
 - 4.1.1 Swisslog 36
 - 4.1.2 HelpMate 36
 - 4.1.3 Pathfinder 36
 - 4.2 Boston Dynamics 37
 - 4.3 Common service robots 37

4.4	Multi robot fleet management system	38
5	Prototype	39
5.1	System Architecture	40
5.2	Hardware and software stacks	41
5.3	Building a Robot prototype	42
5.3.1	Hardware	43
5.3.2	Navigation	48
5.3.3	Robot control	50
5.3.4	Live video	51
5.4	Control system and Human Robot Interface	52
5.4.1	Figma Prototype	55
5.4.2	Functionality	62
5.4.3	Running the Control Center Prototype	63
5.4.4	Controlling a robot	64
5.4.5	Map	64
5.4.6	Issue detection	67
6	Results	69
6.1	Interviews and observation	69
6.2	Robot prototype	69
6.3	Control center prototype	70
6.4	Interviews	70
6.4.1	Results of interviews	70
6.5	Observation	72
6.6	Usability tests	74
6.6.1	Pilot study	74
6.6.2	Observations during usability tests	74
6.6.3	Overview page	75
6.6.4	Report page	76
6.6.5	New patrol page	78
6.6.6	Transport page	80
6.6.7	Inspection page	80
7	Discussion	82
7.1	Prototype	82
7.1.1	Alfred	82
7.1.2	Camera	83
7.1.3	Hardware changes	83
7.1.4	IMU	84
7.1.5	VSLAM	84
7.1.6	Perception in 2D and 3D	85
7.1.7	Steering and control	85
7.1.8	Live video	86
7.1.9	Control system	87
7.1.10	Maps	87
7.1.11	Expanding the fleet	88

- 7.2 Research Method 88
 - 7.2.1 Ethics 88
 - 7.2.2 Design process 89
 - 7.2.3 Observation 89
 - 7.2.4 Participants 89
 - 7.2.5 Interviews 90
 - 7.2.6 Transcription 90
 - 7.2.7 Moderation of the usability test 91
 - 7.2.8 Interview after usability test 91
 - 7.2.9 The Setting of the Usability Test 92
- 7.3 Technology acceptance 93
 - 7.3.1 Business value 93
- 7.4 Results of Usability tests 94
 - 7.4.1 Overview page 95
 - 7.4.2 Report page 95
 - 7.4.3 New patrol page 95
 - 7.4.4 Transport page 95
 - 7.4.5 Inspection page 95
- 8 Conclusion and Further Work 97**
 - 8.1 Research questions 97
 - 8.2 Technical solutions 98
 - 8.2.1 Maps 98
 - 8.2.2 Frontend 98
 - 8.3 Further work 99
 - 8.3.1 Method 99
 - 8.3.2 Alfred 99
 - 8.3.3 Expanding the fleet 99
 - 8.3.4 Control System 100
- Bibliography 102**
- A Pictures of control system web app 108**
- B Consent agreement 116**
- C Interview Guide 120**
- D SUS Form 123**
- E Project Report 126**
- F Wiring Diagram Alfred 145**
- G Evaluation Guide for Participant Observation 147**
- H Moderator Script for Usability Test 149**

Figures

2.1	SAEs Autonomy Scale	7
2.2	RPLIDAR A1 mounted on top of Alfred	8
2.3	ROS nodes, topics, and services	9
2.4	ROS actions	10
2.5	Secure DDS architecture overview	11
2.6	An example of a behaviour tree	13
2.7	A screen capture of Johan playing the game F1 2021, which features a view that shows a forward scene, a view that is common in driving games	17
2.8	Different controllers used for teleoperation	20
3.1	A high level timeline of the project	23
3.2	Redrawn version of Oates' research model. The methods and strategies we used are the boxes with a grey background	23
3.3	Redrawn version of Davis' Technology Acceptance Model (TAM)	25
3.4	Technology Acceptance Model 3 (TAM3)	26
3.5	The table that sums up our findings from the interviews	30
3.6	An example of how we analyzed a quote to deduct a value from it, and the connected quote	31
4.1	Automated Guided Vehicle transporting a cart at St. Olavs Hospital	35
4.2	Atlas from Boston Dynamics	35
4.3	Pathfinder, an AGV developed for hospital logistics	36
4.4	Pathfinder carrying payload	36
4.5	Spot from Boston Dynamics	37
4.6	Stretch from Boston Dynamics	37
5.1	Alfred, the robot	40
5.2	System architecture	41
5.3	High level diagram of the system architecture	43
5.4	Motor control architecture	44
5.5	Wiring diagram for motor controls	45
5.6	The Arduino and motor control compartment of Alfred	46
5.7	The battery and power supply compartment of Alfred	47

5.8	Gazebo on the left, RVIZ on the right. Simulated robot performing SLAM in a simulated environment	48
5.9	A simulated robot with map, pose, local, and global costmaps visualised in RVIZ	49
5.10	Debugging control values	50
5.11	Outliers in angular velocity	51
5.12	Live stream architecture	52
5.13	Web application: New patrol page.	54
5.14	A part of the design system that shows what colors we were to use, how all buttons should look like, what fonts we were to use, and what input fields should look like	56
5.15	A part of the design system that shows how notifications with icons, dropdown, tooltips, and the navbar should look.	57
5.16	The proposed designs for the report list and a report	58
5.17	The overview-page design is meant to quickly give the user an overview and an understanding of what needs their attention.	59
5.18	The suggested design of the list displaying all robots and a robot's status page when you click on it in the list	60
5.19	The suggested design for a page that shows all stored tasks and the page of a single task that can be accessed when you click on it. Here you can see all reports that have been made on all the previous executions of the task and how you can select a robot to execute it	61
5.20	A screenshot that shows how scenes are connected using flows to create interactions in the prototype	63
5.21	Docker compose file	63
5.22	A screenshot of the GUI when the user manually controls the robot. The map and the robots position is on the left, and the video stream from the robot is on the right	64
5.23	The robot pose is transformed from the map origin frame to GCS. The world frame are the largest axes, the original frame are the stippled axes, while the transformed frame is the smaller, solid axes	65
5.24	Mazemaps with transformed map layered on top	65
5.25	Mazemaps with robot position transformed onto it	66
5.26	The Control system has detected a chair	68
5.27	Control system sent notification in MS Teams	68
6.1	Perceived value of robot functions	71
6.2	Web app: overview page	75
6.3	Web app: Report example	76
6.4	Web app: Report example, map	77
6.5	Web app: New patrol view	78
6.6	Web app: Example of patrol	79
6.7	Web app: Transport view	80
6.8	Web app: Inspection view	81

7.1 A frame of the internal voxel grid being projected over the robot's environment in a real-world retail environment with a red laser scanner for ground truth 85

Tables

3.1	Search example	27
3.2	SUS-Score interpretation	32
6.1	Perceived usefulness of tasks after interviews	69
6.2	SUS-Score and perceived usefulness after usability tests	74

Acronyms

BI Behavioral Intention. 20

ESC Electronic Speed Controller. 5, 99

HRI Human-Robot Interface. 2

IC Integrated Circuit. 5

IDI Institutt for datateknologi og informatikk. v

ISSE Information Systems and Software Engineering. v

NSD Norsk senter for Forskningsdata. 29

PEU Perceived Ease of Use. 20

PU Perceived Usefulness. 20

PWM Pulse-width Modulation. 4

ROS Robotics Operating System. xv

SMF Sealed Maintenance Free. 42, 43

Glossary

Data Distribution Service A specification that describes a publish and subscribe model for distributed communication. 10

Dynamic Adaptive Streaming A HTTP based protocol for streaming video, commonly supported. Quite similar to HLS. 86

FFmpeg A complete, cross-platform solution to record, convert and stream audio and video. It is able to decode, encode, transcode, mux, demux, stream, filter and play pretty much anything that humans and machines have created [1]. 42

HTTP Live Streaming A HTTP based protocol for streaming video, commonly supported. Developed by Apple and released in 2009. 86

Navigation 2 A ROS software for robots performing navigation. 48

React A JavaScript library for building user interfaces [2]. 42

Real-Time Messaging Protocol A communication protocol for streaming video, audio and other data. Originally developed as a proprietary protocol for Flash Player. 86

Real-time Transport Protocol A network protocol for transceiving video and audio data through IP networks, typically over UDP. 51

Robotics Operating System 2 The Robot Operating System 2 (ROS 2) is version two of Robotics Operating System (ROS). iii

System-of-systems synthesizer a tool which provides protocol translation between different subsystems[3]. 87

ticks A tick is a way to measure the distance travelled by a wheel. In the same way a clock is split up into seconds or minutes, a wheel revolution is split up into ticks. By counting the number of ticks a wheel has rotated over a time

period, one can calculate the distance this number of ticks corresponds to.
50

TypeScript JavaScript with syntax for types [4] . 42

Video4Linux 2 Video For Linux 2 aka Video4Linux 2 or V4l2 is version two of an API for capturing video on Linux systems [5]. 52

Web Real-Time Communication A peer to peer technology for the web that can transport video, audio and data in real time. Free to use and open source.
86

Webpack At its core, webpack is a static module bundler for modern JavaScript applications. When webpack processes your application, it internally builds a dependency graph from one or more entry points and then combines every module your project needs into one or more bundles, which are static assets to serve your content from [6] . 42

Chapter 1

Introduction

1.1 Research questions

The field of robotics has been reserved for academics and large companies for a long time due to the costs of the hardware and software. But in recent times this has changed, as AI has become more accessible to non-experts, and cheap, easy-to-use single-board computers and micro-controllers like Raspberry Pi and Arduino have become available. This has led to a large number of hobbyists being introduced to electronics and robotics. These projects often end there, as hobby projects, and robotics in professional use remain dominated by large companies such as Boston Dynamics and Swisslog. But does it have to be like this? Do you need to be backed by a large company to build a robot that can be used in a professional setting? These were the questions that sparked our interest in this project.

When starting a project to create an autonomous indoor rover, a lot of research and tutorials can be found on the subject. Given the right hardware, one can get started with autonomous navigation quickly. Then begs the question, how is this useful? How can rovers like this provide any benefit or business value to humans? These are the questions we wanted to answer in this project, and have been formulated as the research question:

RQ1: What is needed to make an indoor autonomous robot system valuable?

which we believe can be answered by answering the following subquestions:

RQ1.1: How can indoor rovers assist professionals in building operations?

RQ1.2: How can a robot-fleet control system be designed to be considered useful to professionals in building operations?

In addition to discovering and building technical solutions, and requirements, which has taken the majority of our time, we have analyzed building operations from the perspective of janitors. The hypothesis is that if we can create a prototype that proves useful to janitors, we will learn about how autonomous systems can be useful in general, what is required for usability and usefulness, and about state-of-the-art-technology.

The prototype has been further developed from the work done in a preliminary project to this one. See Appendix E for the project report done in the preliminary project. Background & theory (chapter 2) contains theory applied in the thesis. Research method (chapter 3) describes the process and methods we used for our research. We have explored related works (chapter 4) which has similar robots, that operates in a similar environment, or involves more advanced robots. The prototype chapter (chapter 5) describes Alfred, the prototype robot, and the prototype control system with task management and HRI. The results chapter (chapter 6) details all the results from this project. This includes the robot prototype Alfred, the control system, and data from the interviews & the usability tests. The discussion chapter (chapter 7) contains discussions regarding results & methods. The conclusion chapter (chapter 8) contains our conclusions in regards to our research questions, and our recommendations for further work.

The link to a YouTube video demoing the prototype can be found in the ReadMe file of the alfred-task repository at:

<https://gitlab.stud.idi.ntnu.no/indoor-robots/alfred-task> (Shortened: **<https://bit.ly/3zzigZu>**)

Chapter 2

Background and theory

This chapter contains information about theories applied in the thesis. Some of the topics are known to us from our professional and academic background, while some are new. section 2.1 is about system development as a research method, and why and how it works as research. section 2.2 explains information and standardized definitions in robotics, and important technology used to build the prototype. See sections section 2.3, section 2.4, and section 2.5 for situational awareness, ROS, and indoor navigation respectively. The section 2.6 is about learning about the domain and the users. section 2.10 is about creating good interfaces between robots and humans.

2.1 System development as a research method

Oates et al. explains that creating any kind of computer-based product is a form of research. In the process, one must perform design and development in a systematic way, find, generate, and analyse appropriate data in order to draw conclusions. [7] Oates et al. describes a research model (See figure 3.2). If this model is applied, research must have a motivation, literature review, a research question, conceptual framework, a strategy, data generation method, and data analysis [7].

For a system development project to be considered proper research, it must demonstrate academic qualities in addition to technical ones, and it needs to contain novel research. The design and creation strategy focuses on developing new IT products. The resulting contribution can for example be a construct, model, a method or a combination of them [7].

With the design and creation strategy, the produced technical solutions will be the main contributions. It is also possible to combine it with other strategies like experiments to provide other contributions by testing how systems are accepted by users for example [7].

The data generation methods interview and observations can be used as quantitative or qualitative methods. If used as qualitative, one collects words, images, sounds, and more. In an interview, the researchers ask questions and control much of the conversation. In observation, the researchers get to observe what the subjects are actually doing. This can be different from what they report they are doing [7].

Nunamaker et al. concludes in their research that systems development can be a credible research methodology [8].

2.2 Robot technology

A robot is a machine that can be programmed to perform tasks. These can be simple tasks such as moving an object from one place to another and can be as complex as driving a car without crashing. ISO defines a robot as a "programmed actuated mechanism with a degree of autonomy (3.2) to perform locomotion, manipulation or positioning" [9].

2.2.1 Arduino

Arduino produces single-board microcontrollers. The microcontrollers are popular because of their cheap price and ease of use. To program Arduino boards one needs a computer with the Arduino IDE and a USB cable to upload it. Most Arduino boards have a set of both digital and analog input/output pins that may be used to control the behaviour of things such as motors or lights, and measure things using different sensors such as temperature sensors or light sensors.

PWM is an acronym for Pulse-width Modulation. It is a technique used by digital devices to reduce the average voltage output by switching a stable voltage on and off at a high rate. An example could be a PWM signal with a 1kHz rate. 100% output, which is called a duty cycle of 100%, means the voltage is on 100% of the time. With a 70% duty cycle, the signal is on 70% of the time. Since the rate is 1kHz, the period is $1/1000\text{s} = 1\text{ms}$. The 70% duty cycle then means that for each period, the voltage is on for 0.7ms and off for 0.3ms. With an Arduino, the PWM-enabled output pins can be controlled using the `analogWrite(pin: int, dutyCycle: int)` command. The arguments are the output pin number and an integer from 0-255 where 255 is 100% duty cycle [10].

Arduino and Raspberry Pi boards use an asynchronous serial protocol for serial communication. RX and TX pins are available both on Arduino and Raspberry Pi and could be used to establish a serial line. This has drawbacks, as the voltage varies. Some Arduino's are 3.3V, while some are 5V. The Raspberry Pi uses 3.3V. Arduino boards have USB-UART converters that allow serial communication over USB.

The microcontrollers on Arduino boards are single-core and single-threaded. There

is no support for multi-threading. All tasks are blocking the only thread available. Some use a technique called protothreading to perform multitasking, which involves using the same thread to keep checking if the parallel tasks need processing at the highest rate possible. This can make writing multitasking programs on an Arduino clean and simple. Any process in a protothread that takes a long time will still block the only thread available and potentially cause issues with the multitasking [11]. Arduino boards can use hardware interrupts to execute code when a pulse occurs on an interrupt-enabled pin. This is set up by calling the `attachInterrupt(digitalPinToInterrupt(pin), ISR, mode)` command in the Arduino program. ISR is an Interrupt Service Routine, which is a function attached to the interrupt with some limitations [12].

2.2.2 Motor Control

Two common types of motor controllers are motor controller IC's (Integrated Circuit) and ESC's (Electronic Speed Controller), where the latter is more advanced. The motor controller IC can be used to control the speed and direction by amplifying the control signals from a controller into high power outputs that can be used by loads like a DC motor. ESC's can control the speed and direction of the motor. The L298N and DRV8871 motor control IC's are based on an H-bridge. An H-bridge is a circuit with four switches that can switch power and polarity to a load. This can be created with switches, relays or solid-state with transistors [10].

2.2.3 PID controller

A PID controller is a proportional, integral, derivative controller. It is an algorithm for regulating a process with widespread use in industry. It consists of a control loop calculating the error between the setpoint and the measured process value, and then producing a control value to correct the process with the goal of reducing the error. The correction is calculated based on the three terms of the controller, proportional, integral, and derivative. The equation for PID in its theoretical form is:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt}$$

Where K_p , K_i , and K_d are non-negative coefficients. These coefficients must be tuned to the specific application of the PID controller. [13]

2.2.4 Classification of robots

International Federation of Robotics (IFR) is a non-profit organization aiming to promote research, development, use, and international co-operation. They refer to international ISO standards for robotics related to safety, performance criteria, modularity, and vocabulary. Standards concerning robots is prepared by the "Robotics" committee, ISO Technical Committee 299. ISO defines two major categories of robots used by IFR, industrial robots and service robots [14]. ISO 8373:2021

also defines medical robots as a third category not regarded as industrial nor service robots [9].

The official ISO definitions are:

- **Industrial Robot:** "Automatically controlled, reprogrammable multipurpose manipulator (4.14), programmable in three or more axes, which can be either fixed in place or fixed to a mobile platform (4.16) for use in automation applications in an industrial environment" [9]
- **Service robot:** "Robot (3.1) in personal use or professional use that performs useful tasks for humans or equipment" [9]
- **Medical robot:** "Robot (3.1) intended to be used as medical electrical equipment or medical electrical systems" [9]

In the robotics field rovers are a subclass of robots that is formed like a land vehicle such as a car. Rovers are usually tasked with the exploration of places that would be infeasible for a human to explore. A known example of a rover is NASA's Mars exploration rover, Curiosity. Depending on the equipment installed and the tasks it performs, this could be classified either as an industrial or service robot.

2.2.5 Autonomy in Robots

In relation to robots, autonomy describes the level of independence a robot has from human control. A fully autonomous robot is completely independent of human control and uses artificial intelligence to process and make decisions, while a semi-autonomous robot is operated by both automatic and human control. A common setup for semi-autonomous robots is to have the robot automatically operate on its own, and whenever a decision has to be made, a human operator makes the decision, while in a fully autonomous setup artificial intelligence would make the decision.

The level of autonomy is not strictly non-autonomous, semi-autonomous, or fully autonomous, but usually on a scale that has several values in between. The scale varies based on the autonomous system, but an example of a known scale is SAEs scale for autonomous cars seen in figure 2.1. This scale is specifically for autonomous cars, but the scales for other types of vehicles or machines are more or less the same.

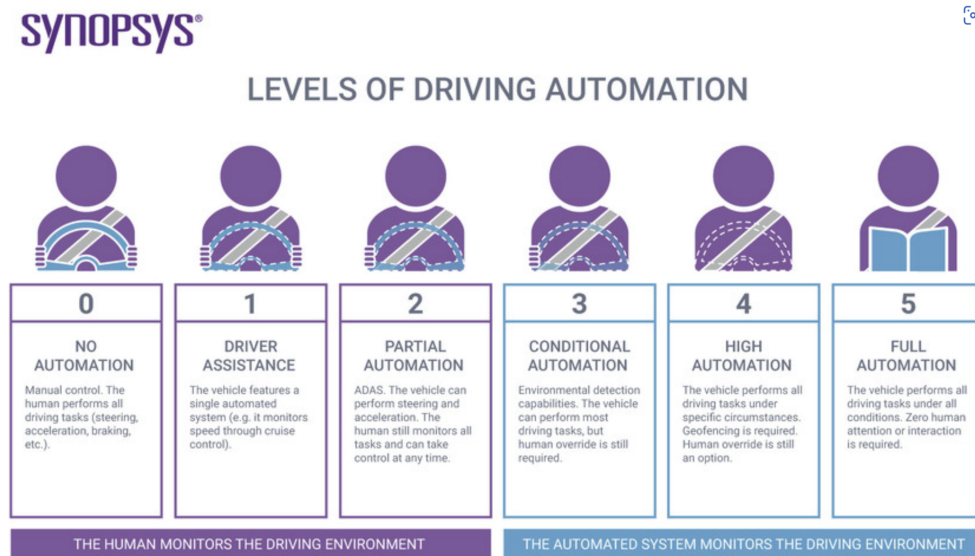


Figure 2.1: SAEs Autonomy Scale

Source: [15]

2.2.6 Raspberry Pi

Raspberry Pi Foundation is a UK-based charity that produces single-board computers. Their goal is to empower people everywhere with computing and digital making [16]. Raspberry Pi units are small computers with their own operating system and are therefore able to do anything a normal computer can do, only limited by its hardware. The advantages are its small form factor, low power consumption, low cost, and how easy it is to interface with the physical world through I/O ports.

2.2.7 LIDAR

LIDAR is a method for determining ranges. To measure the range between the LIDAR and an object, light in the form of a laser is sent towards the object and measuring the time it takes for the laser to reflect to the LIDAR. LIDAR is an acronym for light detection and ranging or alternatively laser imaging, detection, and ranging.

LIDAR sensors are divided into scanning and non-scanning sensors. Scanning sensors can be used in 2D or 3D. A 360° 2D LIDAR scanner that rotates around the vertical axis can be mounted on an additional stepper motor to rotate it around the horizontal axis to perform scans in several layers, resulting in a 3D scan [17].

The RPLIDAR A1 from SLAMTEC is a 2D 360° LIDAR with a range of 12 meters with an adjustable scanning rate between 2 and 10hz and a sample rate of 8000 samples per second [18]. Figure 2.2 shows an RPLIDAR A1 mounted on the pro-

prototype robot.



Figure 2.2: RPLIDAR A1 mounted on top of Alfred

2.3 Situational Awareness

Situational awareness (SA) is to be aware of your surroundings and the situation that you are in, understanding what it means to you now and in the future [19]. Usually, SA is oriented around a goal or task like driving a vehicle. Endsley et al. describe three levels of situation awareness [19]:

- Level 1 - Perception of elements in the environment
- Level 2 - Comprehension of the current situation
- Level 3 - Projection of future status

Proper situational awareness is considered a key factor for good decision-making by humans [19]. A study by Endsley [20] shows that 88% of accidents involving human error in major airlines were attributed to problems with situation awareness.

2.4 ROS

ROS (Robot Operating System) is an open-source software development kit for robotics applications. ROS offers a standard software platform to developers across industries that will carry them from research and prototyping all the way through to deployment and production [21].

2.4.1 ROS Computation Graph

When controlling a robot, ROS structures the process into a so-called *computation graph*. The Computation Graph is the peer-to-peer network of ROS processes that are processing data together. The basic Computation Graph concepts of ROS are nodes, Master, Parameter Server, messages, services, topics, and bags, all of which provide data to the Graph in different ways [22].

2.4.2 ROS Interfaces

ROS applications typically communicate through interfaces of one of three types: messages, services, and actions [23].

2.4.3 Nodes

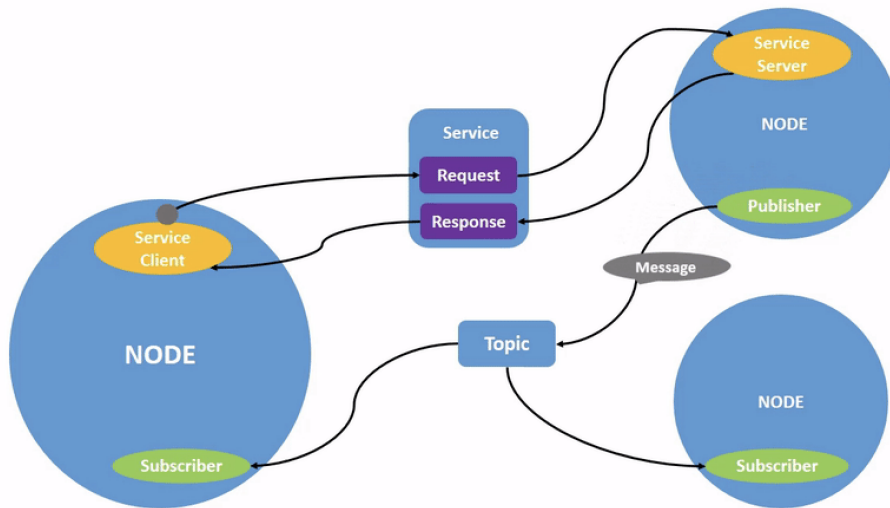


Figure 2.3: ROS nodes, topics, and services

Source: [24]

Nodes are processes that perform computation. ROS is designed to be modular at a fine-grained scale; a robot control system usually comprises many nodes [22].

2.4.4 ROS Topics

Messages are routed via a transport system with publish/subscribe semantics. A node sends out a message by publishing it to a given topic. The topic is a name that is used to identify the content of the message. A node that is interested in a certain kind of data will subscribe to the appropriate topic. There may be multiple concurrent publishers and subscribers for a single topic, and a single node may publish and/or subscribe to multiple topics [22].

2.4.4.1 ROS Messages

Nodes communicate with each other by passing messages. A message is simply a data structure, comprising typed fields. Standard primitive types (integer, floating-point, boolean, etc.) are supported, as are arrays of primitive types. Messages can include arbitrarily nested structures and arrays (much like C structs) [22].

2.4.5 ROS Services

A ROS service is similar to a ROS node but follows a different and much more rigid communication paradigm. While nodes communicate using a one-way, many-to-many paradigm, services follows a one-to-many, request-and-response paradigm. Consisting of one server and one or several clients [22].

2.4.6 ROS Actions

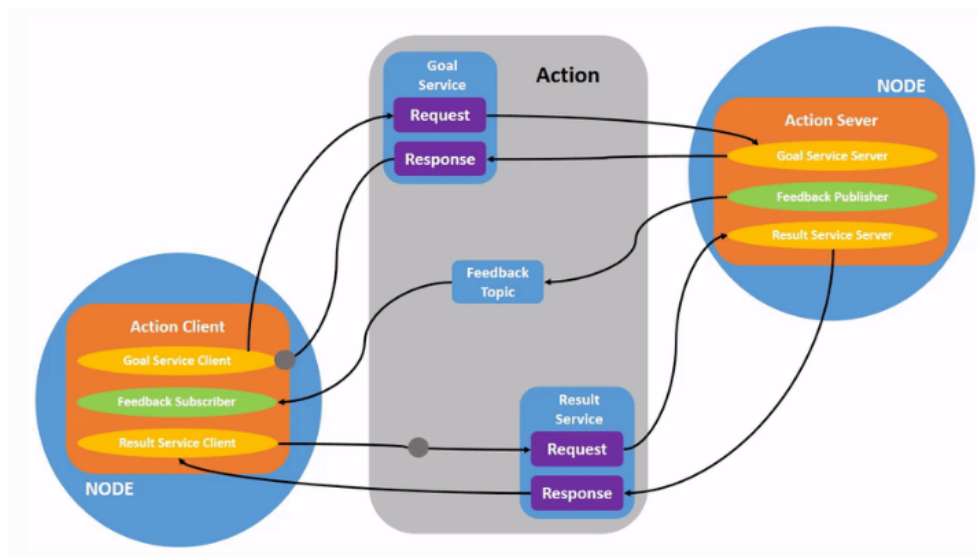


Figure 2.4: ROS actions

Source: [25]

ROS actions use an identical communication paradigm, but when an action client sends a request to an action server to fulfill some goal, it receives feedback on the progress of the action while it is being performed [22].

2.4.7 Security

ROS 2 uses Data Distribution Service (DDS) as middleware to handle communication, and utilizes the security features of DDS which comes with Secure DDS, an extension to the DDS specification. DDS is purely a specification for a Publisher-Subscriber pattern of distributed communication of which ROS 2 supports a few different implementations. The default one for ROS 2 Galactic which is the ROS 2 version we are using in this project, is Eclipse Cyclone DDS. Figure 2.5 shows an architectural overview of secure DDS.

Secure DDS defines five Service Plugin Interfaces which adds security features to DDS. ROS 2 utilizes the three features that are required in the DDS security specification: authentication, access control and cryptographic. Logging and data

tagging is not used in ROS 2 since it is not required by the specification, and is thus not supported by all DDS implementations.

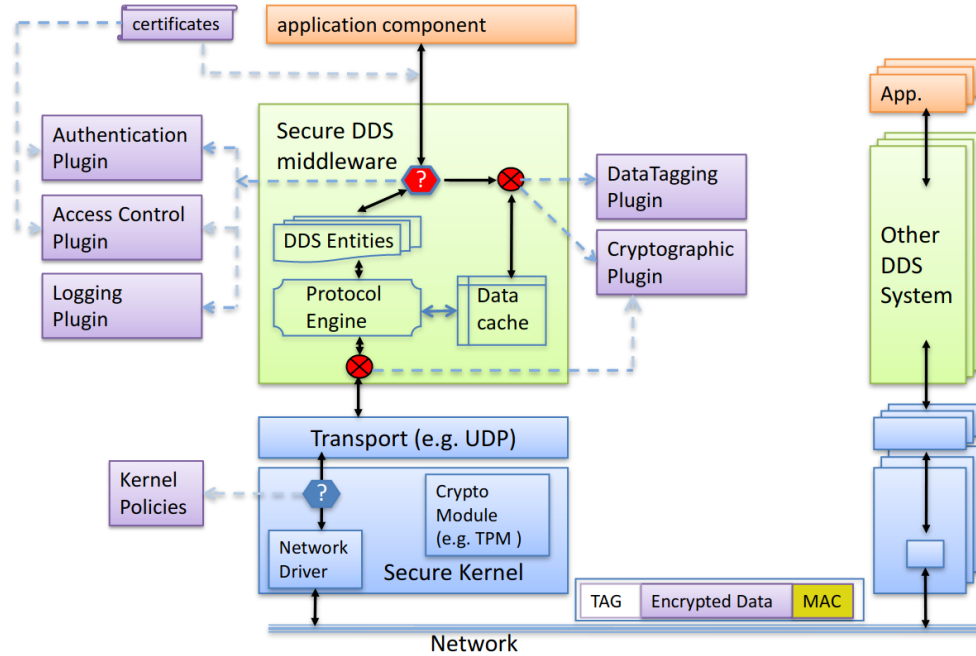


Figure 2.5: Secure DDS architecture overview

Source: [26]

2.4.7.1 Authentication

Authentication is based on a public key infrastructure with a Diffie-Hellman key exchange, x.509 certificates and Certificate Authorities (CA). This allows ROS to restrict access to topics, services, actions and entire DDS domains.

2.4.7.2 Access control

ROS 2 uses a plugin called DDS:Access:Permission to perform access control. It requires two signed XML documents per domain participant. They specify how the domain should be secured and the permissions of specific domain participants, which are authenticated using the methods above. The files are signed by a CA trusted by the DDS:Access:Permission plugin.

2.4.7.3 Encryption

The cryptographic plugin of the DDS implementation specified by the DDS-Security spec handles all cryptographic needs of Secure DDS. Multiple cryptographic suites

are available for encryption in the DDS Security spec, but ROS 2 has selected a plugin with AES-GCM-GMAC, a form of symmetric authenticated encryption.

2.4.7.4 Security summary

The alternatives to securing a ROS 2 system are either using isolated physical networks not connected to the internet, VPN, or make use of the security features included with ROS 2 and DDS.

2.4.8 ROS and frontend

Roslib is a JavaScript library that contains classes and utilities for describing ROS resources and handles communication with `rosbridge_suite` through websockets. Rosbridge Websocket Server from `rosbridge_suite` transceives and translates messages back and forth between ROS and roslib. This can allow apps and web apps communicate with ROS.

2.5 Indoors Navigation

Indoor mapping and localization is a hard problem, which can be solved in several ways. Some of the reasons that makes the problem hard are lack of, or poor GPS signals, high accuracy requirements, a lack of maps, dynamic environments and obstacles like stairs, elevators and doors [27, p. 5].

Indoor localization, or tracking, is often done with a real-time locating system (RTLS) which includes multiple different methods and technology depending on the application. RTLS can be based on GPS for global tracking, Ultra Wide-Band (UWB) antennas and receivers, Bluetooth Low Energy (BLE) beacons and devices, WiFi, choking points, etc. A different approach is Simultaneous Localization and Mapping (SLAM) [27, p. 5].

2.5.1 Nav2

Nav2 or Navigation 2 is a project that seeks to find a safe way to have a mobile robot move from point A to point B. This includes dynamic path planning, computing velocities for motors, avoiding obstacles, and structure recovery behaviours [28].

2.5.1.1 Behaviour trees

Behaviour trees are Nav2's way of structuring the behaviour of a robot using ROS action servers, where each action server represents a node in the tree. A complete behaviour tree usually consists of two sub-trees concerned with recovery and navigation behaviour. The navigation sub tree consists of nodes that take care of path planning and sending velocity commands based on the path planning, while

the recovery sub-tree consists of nodes that try to perform recovery actions when needed.

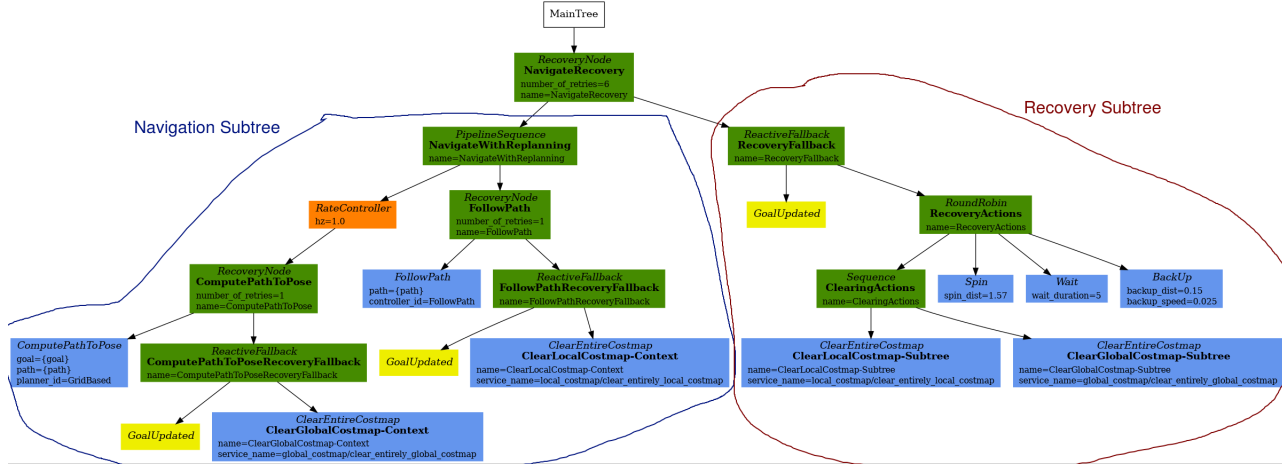


Figure 2.6: An example of a behaviour tree

Source: [29]

2.5.2 Pose

In robotics, pose describes the position and orientation of a robots coordinate system or frame relative to another coordinate system or frame [30, p. 8]. In Computer Vision, poses are used to help localize a robot.

2.5.3 Point clouds

Point clouds are a way of representing the world as clouds of data points, where each point is given a coordinate in a given coordinate system.

2.5.4 SLAM

SLAM is an acronym for simultaneous localization and mapping which is a set of algorithms that tries to map the environment to an agent and at the same time tries to localize this agent in this mapped environment.

All SLAM algorithms need to be able to sense the environment and this is done by using sensors. Commonly used sensor types are LIDAR sensors, sonar sensors, and cameras. By using the latter sensor you would perform something called V-SLAM or visual SLAM.

A common setup, due to its cheap and simplicity is to perform SLAM using a LIDAR. The LIDAR creates point clouds by scanning the environment, and the SLAM algorithm estimates the pose of a robot by comparing the point cloud it sees with other stored point clouds and pose pairs.

2.5.5 Path planning

In Robotics, path planning is a field concerned with determining how a robot should move to achieve its goals, which usually is getting from point A to point B. The path planning problem involves computing a collision-free path between a start and a goal position. Very often, besides obstacle avoidance, path planning algorithms also try to find the fastest way [31]. Known path planning algorithms are A* and Dijkstra's algorithm.

2.5.6 Costmaps

Costmaps are a fundamental concept widely used in path planning. A costmap describes the environment around a robot, in a grid-like format. Each cell in the grid is assigned a cost, which signifies how much effort it would cost a robot to move through this cell. If an obstacle is in a cell, this cell would get a large cost, while an empty area would have a low cost. By representing the world in this way a path planning algorithm can find the optimal path without the robot crashing. Figure 5.9 in chapter 5 shows a visualised costmap.

NAV2 keeps both a local and global costmap.

2.5.6.1 Local costmap

A local costmap is a small costmap used for dynamic object detection. The map is kept small so it can be computed and updated fast and often, to avoid the robot crashing. If an object is detected, a new safe path is computed and merged into the planned path.

2.5.6.2 Global costmap

The global costmap is used for planning the path the robot is to take to its intended location. The size of the global costmap depends on how much of an area a robot has discovered, and is only limited by the hardware of the robot. Due to its size, the global costmap is updated less often.

2.6 Interview & Observation

An interview is a form of structured conversation widely used in research. There are many forms of interviews, but it is normal to split interviews into three categories: structured-, semi-structured-, and unstructured interviews. In a structured interview, all questions and the order of the questions are planned beforehand. In an unstructured interview, no questions are planned beforehand. A semi-structured interview is somewhere in between the aforementioned types of interviews, in that there usually is a few planned questions that may or not be asked, and in which

it is possible that other questions may arise and be asked should the interviewer want to [7].

An interview is considered a qualitative data source, which means it allows for a deeper understanding of a topic, but will at the same time usually have much fewer respondents due to taking more time to conduct [7].

Observations are another qualitative data generation method often used in research. During an observation one or several observers observe something happen. Observation is often used for observing people when they work, people that test things, and how things are done [7].

Observations can be an alternative to performing an interview, as a person explaining how they are doing something is not necessarily how a neutral observer would say they are doing the same thing [7].

2.7 Analyzing Qualitative Data

Qualitative data includes all non-numeric data, which can be words, images, sounds, and so forth [7, p. 266]. Due to the nature of the data, it often needs to be prepared in some type of way before it can be used. This often involves abstracting from the data themes and patterns that can relate to your research topics. There are few established procedures for analyzing qualitative data, and the quality of the data is often dependent on the skill of the researcher [7, pp. 266–267].

Analyzing qualitative data is done using one of two approaches: an inductive or an deductive approach. An inductive approach creates a theory based on the data, while in an deductive approach the researcher has a theory beforehand and tries to see if this theory is correct or not based on the data [7, p. 274].

2.8 Human Centered Design

Human centered design is a design philosophy that revolves around four principles [32]:

1. **People-centered:** Focus on people and their context in order to create things that are appropriate for them.
2. **Understand and solve the right problems, the root problems:** Understand and solve the right problem, the root causes, the underlying fundamental issues. Otherwise, the symptoms will just keep returning.
3. **Everything is a system:** Think of everything as a system of interconnected parts.
4. **Small and simple interventions:** Do iterative work and don't rush to a solution. Try small, simple interventions and learn from them one by one, and slowly your results will get bigger and better. Continually prototype,

test and refine your proposals to make sure that your small solutions truly meet the needs of the people you focus on.

The belief is that by adhering to these principles, one will create products that actually solve a problem and is usable for the intended end-users. The last principle states that one should perform iterative work, doing small changes between each iteration, and evaluating each iteration. A common way of evaluating a prototype is to perform usability tests.

2.8.1 Usability tests

A usability test is a type of test that tests the usability of a product. A usability test usually consists of a user test where a test subject is to interact with the product. This could be done by giving the subject a couple of tasks to perform in the system. Afterward the test subject is set to evaluate the system. This can be done in the form of an interview or by using a form such as the commonly used SUS form.

2.9 Navigating indoors

In [27] we found that there are several alternatives for navigating indoors. Real-time locating systems that used GPS, ultra-wide band senders/receivers, Bluetooth Low Energy senders/receivers, WiFi are all alternatives, as well as the method we used, which was simultaneous localization and mapping or the abbreviation SLAM which it is more known as. We chose this method as the literature proved it performed well and the only extra equipment we would need was a LIDARsensor, which is relatively cheap.

We also discussed different SLAM algorithms such as ORB-SLAM3, Cartographer, Hector, GMapping, and the SLAM algorithm used in the ROS package SLAM TOOLBOX which is based on the Open Karto-library, which was the algorithm we ended up using. The project report can be read in its entirety in Appendix E, and the chapters discussed above are chapters 3.4 and 3.5.

2.10 Interfacing interactions between robots and humans

When one designs an interface between robots and humans, there are many things to consider. What level of autonomy should the robot have? Should the robot be fully autonomous, completely steered by its operator, or something in between? Should the user be a factor when designing the interface? Without any previous experiences and knowledge about subjects such as these, it was natural to consult the literature.

2.10.1 Designing a graphic user interface for interactions with robots

[33] is a paper that summarizes the interviews of a number of experts from the Robotics Institute at Carnegie Mellon University in Pittsburgh, Pennsylvania in the United States of America. The paper summarizes the experiences of the experts with a focus on the interface between fully- and/or semiautonomous robots and their operators. The findings are summarized under 7 themes, but there are 4 themes we found especially relevant: Interface Design, Command Inputs, Control, and Remote Awareness.

2.10.1.1 Remote Awareness

The paper states that if the user is to control a robot, the interface should display information that helps the user understand the remote environment and at the same time maintain situational awareness. One of the suggestions was that both map and video views should be available in the interface. They also suggest that the camera should provide a forward scene, as this is a common way for automotive representation, an example of this can be seen in figure 2.7.



Figure 2.7: A screen capture of Johan playing the game F1 2021, which features a view that shows a forward scene, a view that is common in driving games

It was also suggested that one can aid the user in maintaining situational awareness by displaying a dashboard that shows information such as speed, position, tilt angle, and even more advanced data.

2.10.1.2 Control

The summary brought up that there was a discussion regarding the level of control of the robot. Some suggested that the level of control should be so low that the user could control wheels independently of each other while others recommended a higher level so steering the robot would be more akin to driving an RC-car. The general suggestion was to operate at a high level, but give the user the option to drop down in level in case of unusual events.

The level of autonomy was discussed as well, but the general suggestion was to keep the human in the loop, and perhaps implement a sliding autonomy, which means it could be fully autonomous at times but also go down in the level of autonomy.

2.10.1.3 Command Inputs

The experts in the paper recommended that the control of a robot should be flexible in the way it can receive input from the user. Suggestions in the paper were a joystick-type of steering combined with video and waypoint selection on a map. The general guideline was to spend time when considering how one can enhance human-robot communication.

The experts further advised against navigating in 3D interfaces.

2.10.1.4 Interface Design

While most of the experts had varying opinions of how the other themes should be solved, the experts widely agreed that most interfaces are designed by and for themselves, making it difficult for an untrained user to use the system without training. Lots of "arcane" technical information displayed and features designed explicitly for debugging purposes were common and would make the systems hard to use without training. The appearance of the systems was also described as designed badly. As robotics are becoming more and more widely used, it means that more and more of the operators of the robots are likely to be non-robotics experts. This means that there had to be made changes in the process of designing interfaces between robots and humans. The experts recommended the usage of HCI methods for interface development and careful testing of the interfaces.

2.10.2 Manually steering a robot

Teleoperation of a robot has been the subject of much research, and the different methods of doing so are not an exception. There have been numerous experiments where people have tried alternative hardware interfaces for steering a robot such as the Wiimote (the Nintendo Wii controller held like a classic controller and motion input enabled)[34], the Joypad (using both of the Nintendo Wii controllers without motion input enabled)[34], a Playstation 3 controller [35][36], the

touch screen of a mobile phone [35] and a keyboard which seems to be the standard, as all the mentioned experiments compared the alternate input device with a keyboard. The previously mentioned methods of inputs all have in common that they use a physical interface for providing input to the robot, but there has been performed experiments where gestures [37] and the human voice [38] has been tested. Although interesting and original, this form of input seems unsuitable for an actual work environment and will therefore not be further discussed.

[35] performed an experiment using the touch screen on a mobile phone and a PS3 controller. The result of this experiment suggests that using the touch screen of a mobile phone for teleoperation was the most efficient, that the keyboard was the second most efficient. and that a PS3 controller was adequate for teleoperation but was the least efficient of the three. [34] performed a similar experiment using different controllers and showed that the Nintendo Wii-controller as a Wiimote(the controller a) in 2.8) was a viable option for using the keyboard for input as long as the situational awareness was upheld. This statement is shared by [36] who performed an experiment where they tried to see if an increased situational awareness would help a user in performing a task better. In the experiment, the user was to remotely operate a robot in a vineyard and spray grapes. The more grapes sprayed and the least amount of collisions would give the highest scores. They compared the performance of all combinations of three different variables: a head-mounted display/a PC screen, one camera angle/multiple camera angles, and using a keyboard/PS3 controller to steer the robot. The results of this experiment showed that all runs were significantly better both in spraying and navigating the vineyard when they had multiple views or had an opportunity to get a better situational awareness.

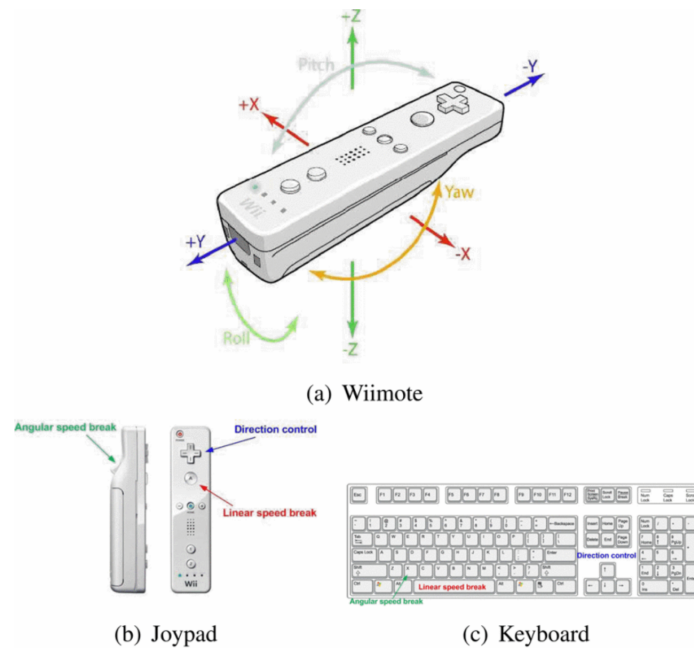


Figure 2.8: Different controllers used for teleoperation

Source: [34]

2.11 TAM

Technology Acceptance Model (TAM) was developed by Fred Davis in 1985. It was based on the "Theory of planned behaviour" by Icek Ajzen and Martin Fishbein, and consists of five elements. Design features, perceived ease of use, perceived usefulness, attitude towards using and actual system use. The goal is to predict the actual system use by measuring the other variables.

The Technology Acceptance Model has been subject to criticism, research and improvements. It has significant limitations to keep in mind. For instance, it does not measure the benefit of using a technology. Perceived Usefulness (PU) and the predicted use outputted from TAM cannot replace other measures of impact [39]. Turner et al. found that Perceived Usefulness (PU), Perceived Ease of Use (PEU), and Behavioral Intention (BI) of TAM correlate with actual usage where BI was the best predictor, PU second best, and PEU the least good predictor of actual use. They also found a significant difference in the variables' success in the prediction of actual use, if actual use was measured subjectively through reported use or objectively by automatically recorded use [39].

Much of the research on its effectiveness has been on technology the users have already been using for a while, while we in this project have attempted to apply it to evaluate a prototype system. Keung et al. [40] even claims that Davis designed the original model for use in cases where the users are already using or have used

the technology in question. While the original research was indeed performed in a situation where the users had already been using the technology for a while, Davis also suggests that the model has promise in predicting user acceptance early in the design process [41]. Davis and Venkatesh [42] suggest using TAM to evaluate pre-prototype systems. According to them, perceived usefulness can be predicted by simple non-interactive prototypes, and does not vary much from perceived usability after hands-on experience over several months. Perceived ease of use however is more difficult to predict and requires more interactive prototypes and hands-on experience. Using early testing and TAM, decision-makers can decide to continue as planned, modify the design or abandon the project.

Chapter 3

Research Method

This chapter describes the research process and methods. While continuing the development of the robot prototype, we planned the user interviews. Around the time of the interviews, Alfred was beginning to become functional enough for the planned user tests, and work on the control system began. The control system was designed in a user-oriented way and with the capabilities to support multiple robots with different capabilities. Due to time limits, some of the features are not fully implemented, but are only good enough to perform functional usability tests. We analysed the results from usability testing to evaluate the value of the tasks the system can perform, to discover requirements for the system, and needed design changes.

3.1 Process

As stated in the introduction (chapter 1), the aim of the project is to discover new knowledge by:

- Finding tasks in which autonomous rovers can assist professionals in building operations.
- Finding the best way to design a robot-fleet control system that is useful to professionals in building operations.

As the intended products of the project was going to be a robot prototype and a prototype of the robot-fleet control system, it was natural to follow a design and creation-strategy. Using Oates' diagram of the research process, one can get an overview of the strategies, data generation methods and the data analysis methods used in the research. Ours can be seen in figure 3.2.

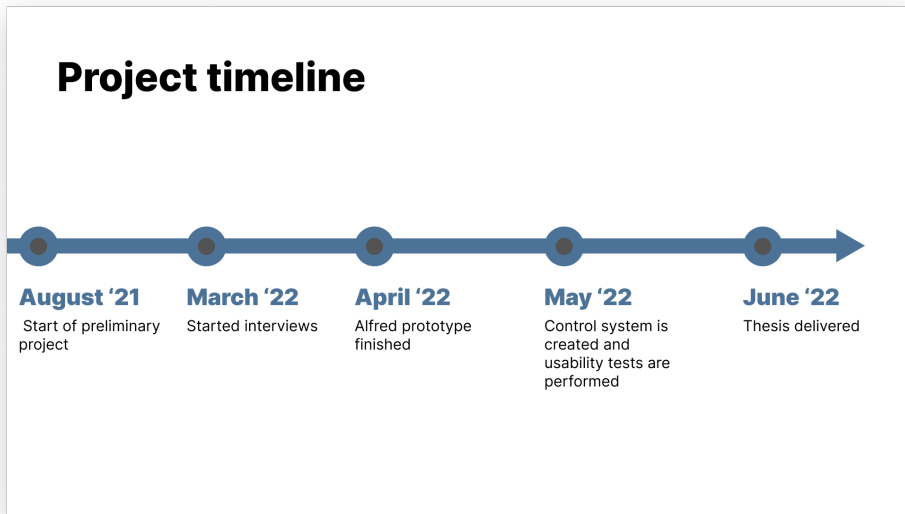


Figure 3.1: A high level timeline of the project

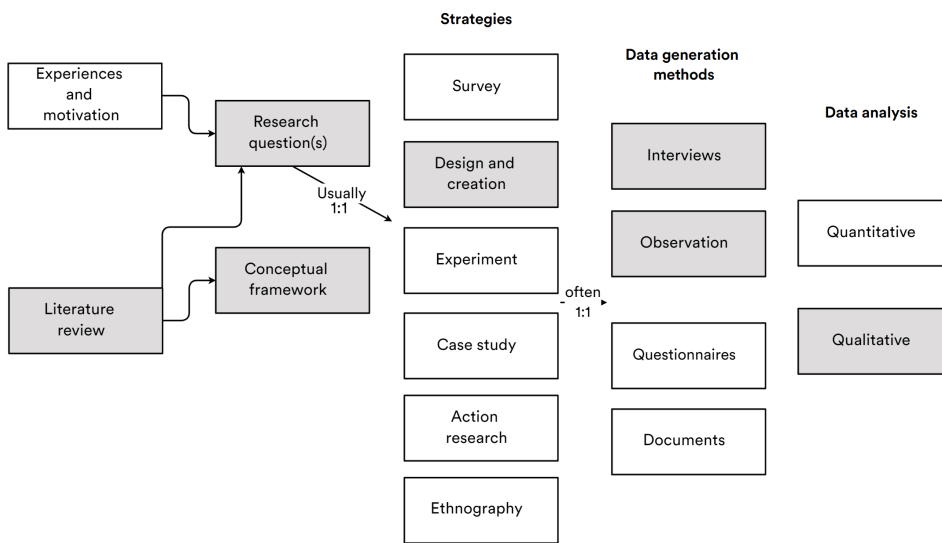


Figure 3.2: Redrawn version of Oates' research model. The methods and strategies we used are the boxes with a grey background

Source: [7]

Since the control system prototype needed robots to control, it was quite natural to start with building the robot prototype first. Due to us having no experience with building robots, it became quite natural to see what others had done before us. This was done by scouring several forums, videos, and such. Some design choices were made because it seemed like most of the robot building community

was doing it in that way, while there were other times the community was more divided. In such situations, we chose to see what research had been done on the available choices and chose the method that had the research results that seemed the most solid. An example of such a situation was when we chose how the robot was to autonomously navigate.

We viewed the robot prototype as a tool to be used in the research for creating the control system prototype, and therefore chose to not involve any end-users in the process of building the robot as the intended end-users would not have any interaction with the robot outside of the control system, and it therefore seemed pointless to involve them in this.

Having built the robot prototype, we were ready to start the work with creating the control fleet prototype. The first step in this process was to see what others had done before us, so we performed a literature review. This provided guidelines for how the interaction between the robots and humans should be done.

With that in mind, we set out to find tasks that both the robot could do and that would actually be helpful for our intended end-users. To do so, we needed to learn about how the janitorial staff worked, what tasks they performed, and how they did them. We therefore chose to interview members of the janitorial staff at relatively large buildings and observe the staff as they worked.

After analyzing the data gathered in the interviews and the observation, we found that there were a few tasks that several subjects thought could be useful in aiding them in their work. We selected the tasks that were most popular, turned them into user stories, and started creating a prototype for the robot fleet control system.

To design and create the prototype we worked after HCD/HCI principles, which meant some iterations of the prototype were created before actually programming the prototype. We started by sketching the prototype in Figma, discussing and changing the design based on discussions between ourselves. This allowed us to experiment with the appearance of the UI as well as plan how the user stories would be implemented in the prototype. After getting feedback on the Figma prototype from our advisors, we made some changes and created the prototype.

In order to evaluate the design of the system and the tasks we had gathered, we decided to perform a user test and an interview afterwards. The results of the user tests can be found in chapter 6. We are using TAM3 as a model to systematically evaluate the results from the project. The goal is not to successfully deploy a new system in a team, nor learn how to do it, but to learn more about robot technology and how it can be created to be successfully deployed. While normal use of the model is user and problem-oriented, this project is driven by technology. The main focus is not to help janitors or others in property management, but to learn more about how robots can be useful. To find good use cases and test whether a system could work, TAM3 is used. Not all of the variables of TAM3 are measured, mainly because of the technology focus of the project, and the limited time frame.

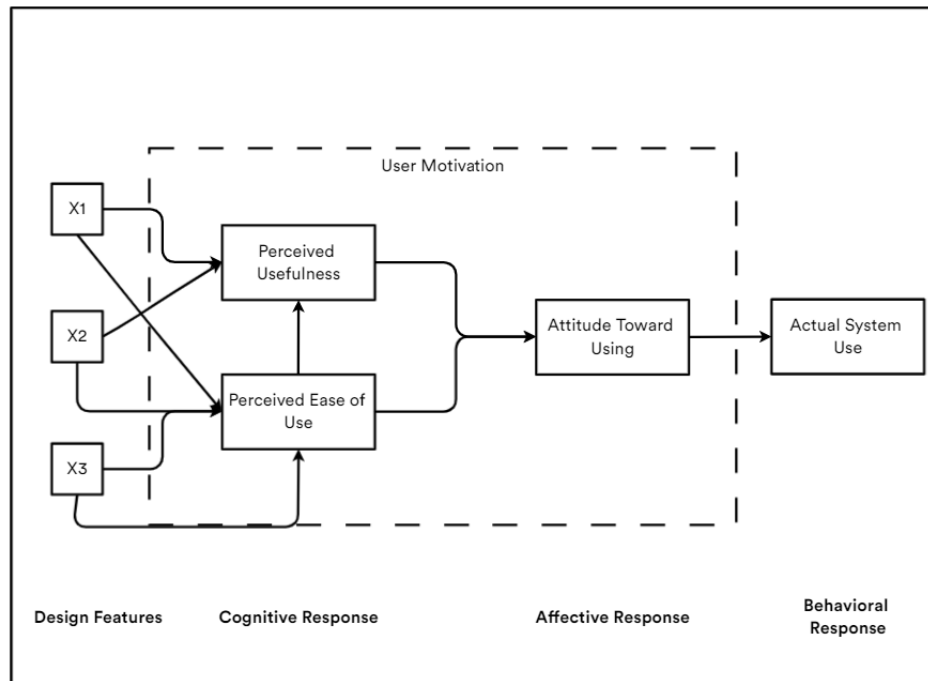


Figure 3.3: Redrawn version of Davis' Technology Acceptance Model (TAM)

Source: [43]

The design features have been defined through interviews and observation, while perceived usefulness and perceived ease of use have been measured through usability testing with the prototype.

Technology Acceptance Model 3 (TAM3) was published by Venkatesh, V. and Bala, H. in 2008 [44]. Technology Acceptance Model 2 (TAM2) was published earlier with determinants of perceived usefulness added to the original TAM. In TAM3, Venkatesh and Bala adds determinants of perceived ease of use, suggested by [45], to further develop TAM2 into TAM3, shown in figure 3.4

TAM is here used as a tool to evaluate whether it can be worth continuing research on robot systems such as this one pertaining to janitorial staff or other professionals in property management. If TAM predicts a high chance of adoption through testing with the prototype, it suggests that this technology might have a future in regards to whether the users would want it. It does not predict the production value or if it could be worth an investment. If TAM predicts high usage, it also means future work can take inspiration from our design choices, or learn from our mistakes if the scores are low.

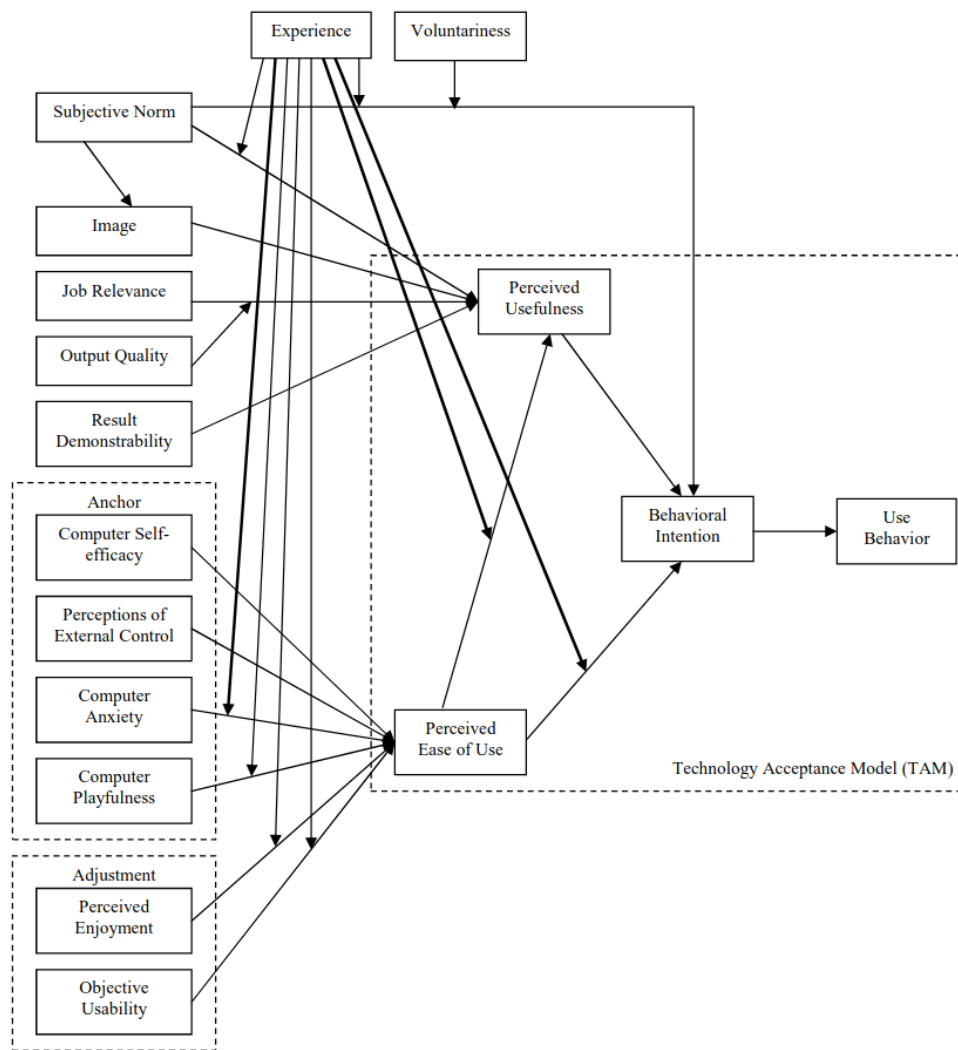


Figure 3.4: Technology Acceptance Model 3 (TAM3)

Source: [44]

3.2 Literature Review

To perform the literature review, we followed the method described in [7, pp. 80–85]. This meant that the literature review was split into 5 stages: searching, obtaining, assessing, reading, and critically evaluating. To ensure that we would not miss out on any newly released papers that could be relevant to our project, we performed the searching phase continuously during the project. If some new literature was added to our literature list, the rest of the phases was performed as well.

To create our literature list, we created a table in Notion with the following fields:

- Place searched: How we found the literature
- Search terms: What terms we used for searching when we found the literature
- Name: The name of the book, article, journal, etc.
- Link: A link to to the literature
- Keywords: A list of keywords we’ve written down ourselves, so it would be easy to find the relevant literature later on
- Summary: A short summary of the literature, written by ourselves.
- Comments: A box for writing comments and thoughts we may have had when reading the article

3.2.1 Searching

When searching we had to experiment, using the methods proposed in [7, pp. 80–85] to find the best results. This was done by finding searches that captured the essence of the literature we were interested in reading. If a term provided few results, we tried to find a synonym or an alternative way of phrasing our search to see if it provided more results. If a search provided too many irrelevant results, we tried to limit the number of results by adding more terms to the search. An example of how we worked when we wanted to find literature for designing a human-robot interface can be seen in table 3.1.

To find literature we search engines that are considered reliable and good, namely Google Scholar, IEEE Xplore, and Science Direct.

Search	Results
Design robot interface humans	Few results
HRI	Too many irrelevant results
HRI HCI	Many interesting results

Table 3.1: Search example

3.2.2 Obtaining

After finding potentially relevant literature, we had to obtain it. Luckily, we were able to find all the digital copies of literature we found relevant, so we did not have to track down any physical copies in a library or anything like this.

3.2.3 Assessing

When collecting literature, it is of utmost importance to ensure that the literature is credible. This is done in several ways but can be summarized as putting the origin of the literature under scrutiny. This includes, but is not limited to looking at the author, the publisher, where it was published, and when it was published.

The majority of the literature we used was papers published in journals and at conferences. So our assessment process mainly consisted of looking at the history of the conferences and the journals, as well as when the conferences were held/-journals were published. If they were moderately old, it was likely that they had a respectable reputation and therefore likely to be credible. And if the literature itself was too old, it was likely that the information was outdated, especially when one considers the rapid development of our field.

We also used the number of times a piece of literature had been cited as a measurement of its credibility. If the literature had been cited many times, the implication is that the academic community finds its findings credible.

3.2.4 Reading

When we read the literature, we created a short list of keywords that we thought summarized what we had read and wrote a summary of the text, with comments and thoughts we had had when we read.

3.2.5 Critically Evaluating

If a text passes through all the previous hoops, it is not a given that it is useful for our work. It is also not a given that the conclusions drawn in the text are valid based on the evidence provided in the text. We therefore had these things in mind after we had read the literature, and scrapped all texts we did not want to include based on these reasons.

3.3 Interviews and observation

3.3.1 Interviews

We planned semi-structured interviews to allow the interview subjects to speak freely. The purpose of the study was to try to discover useful tasks for robots by making interviewees think creatively and by learning more about their profession

so we could do the same, which makes this type of data gathering appropriate according to Oates [7]. We created an interview guide to start the conversation with the interviewees and help us if it stalled. The guide was prepared early in order to be reviewed by our supervisor and revised before submitting it to Norsk senter for Forskningsdata (NSD) for approval. We also tested the interview guide on a third person to discover issues with it and to get a time estimate.

We chose to record audio from the interview to make sure we do not miss anything, so we could concentrate on the interview itself. We considered recording video, but we agree with Oates's claims that it is even more intrusive and inhibiting than audio recording [7].

Once NSD had approved the application we started recruiting interviewees. We wanted people working in building operations of large semi-public buildings and started with janitors. We also attempted to recruit people from security and other departments of property management, but without luck. We did however find 12 janitors from various schools and universities in Trondheim.

All interview subjects read and signed an agreement we had approved by NSD before we conducted the interviews. Before beginning each interview, we also verbally informed them of their rights to data protection, that they could stop the interview at any time, told them about anonymization, and kindly asked them if we could record audio so we could concentrate on the conversation. One of us had the lead in the interview, with the other listening and aiding with additional questions or follow-up questions when possible or needed.

3.3.2 Group Interview

Two of the interviews were conducted in a group setting, as in these cases, the contact person preferred it. We found this acceptable since the groups would consist of two team members and a leader and this was in accordance with the recommendations of [7, p. 195] when conducting group interviews, as all the participants would be of similar status in the workplace.

3.3.3 Observation

We conducted observations with a team of janitors in a large building for about half a workday. The observation method was overt participating-observer, a type of participating observation, where we would shadow janitors and help them where we could. As Oates explains is necessary, we took field notes, writing down what we observed along with our thoughts and reflections along the way [7]. During the observation, we did not have any particular area of focus and just tried to see if we randomly discover something relevant. We were introduced to the team during a morning meeting and the team was informed about why we were there.

3.4 Usability testing

To plan and perform the usability tests, we used [46]. The usability tests consisted of a user test with 8 different tasks. These tasks were open-ended, and we took great care with the wording of the tasks, in order to not give any hints to the test subject on how to perform a task. The test was moderated and had one observer.

After the user test, we performed a short interview. The interviews consisted of a semi-structured interview, the filling out of a SUS form and us giving the interviewees statements and them saying how much they agreed with the sentiment of the statement.

3.5 Data Analysis

3.5.1 Interviews

After transcribing the interviews, they had to be further processed in order to provide useful data. To do this, we put focus on every statement made with regards to the robot and the tasks it would be useful for it to do, extracted the task we interpreted in the statement, and how much value we perceived that the user felt the robot would bring by doing/assisting with the task. We put the value on a discrete scale with "No value" being the bottom score and "Much value" being the top score. We put the data in several tables: one table that gave an overview of the value score each person gave to each robot type, and a separate table for each person, where each score has the quote or question and answer the scoring is based upon.

Person	Robot function / value	Automatic deviation & damage reports	pictures of wrongly placed things & obstructions in hallways and emergency routes	Air quality information with regards to health and economy	Cleaning	Help with recycling	Delivery of items
1			1		3		
2		3	3				
3		2	1		4		
4		4			3	3	2
5		4			3	3	2
6		4			3	3	2
7		3	3			4	
8		4	4	4	4		4
9		4	3	4		3	
10		4	3	4	3		
11		4	3	4	3		
12		4	3	4	3		
AVG		3.6	2.7	4.0	3.2	3.2	2.5
		No value	Little value	Neutral	Some value	Much value	Did not answer
		0	1	2	3	4	

Figure 3.5: The table that sums up our findings from the interviews

Many of the suggested tasks were very specific e.g a robot that could collect gravel,

Robot that uses sensors to gather information about rooms		Det hadde ikke vært så dumt å hatt en som målte luftkvaliteten i bygget. Det savner jeg egentlig å kunne sjekke litt og hatt litt målinger på.
---	--	--

Figure 3.6: An example of how we analyzed a quote to deduct a value from it, and the connected quote

and we chose to generalize tasks such as these in order to try to see if there were any types of tasks that were suggested several times. An example of this was that the tasks "a robot that could collect gravel" and "a robot that could hose down the windows" were generalized to "a robot that could help with cleaning".

3.5.2 User tests

We chose to gather several types of data from the user tests, both qualitative and quantitative data. The quantitative data was in the form of a standard SUS (System Usability Scale)-form. This meant that the users were presented with statements about their experience with using the system, and they had to express how much they agreed or disagreed with the sentiment of the statement. Their answers were on a Likert scale where each answer corresponded to a score and looked like this:

- Strongly disagree: 1 point
- Disagree: 2 points
- Neutral: 3
- Agree: 4
- Strongly agree: 5

To analyze the data we defined the variables X and Y as such:

- $X = (\text{Sum of scores for all odd-numbered questions}) - 5$
- $Y = 25 - (\text{Sum of scores for all even-numbered questions})$

And the final SUS-score was calculated by the formula:

$$SUS - score = (X + Y) * 2.5$$

This results in a SUS-score that ranges from 0 to 100, and the SUS-score is graded as can be seen in table 3.2. This gave us a clear-cut number that told us how each test subject had experienced the system

In a similar vein, we wanted to measure if the interviewees thought the system would bring any value or usefulness to their work. To gauge this we did a similar exercise, where we asked the interviewees to rate the usefulness of the tasks we had the robot perform on the following scale (??):

By grading each answer as such, it was quite easy to find how valuable each task was perceived as by finding the average value.

SUS Score	Grade	Adjective Rating
>80.3	A	Excellent
68-80.3	B	Good
68	C	Okay
51-68	D	Poor
<51	F	Awful

Table 3.2: SUS-Score interpretation

- No value: 0 points
- Little value: 1 point
- Neutral: 2 points
- Some value: 3 points
- Much value: 4 points

3.5.3 Observation

When we observed the janitors' work, we took note of all tasks the janitors did and if relevant, we elicited new tasks for the rover to do that had not come up during the interviews. As we were not a part of our intended user group we would not rate any new tasks' usefulness, but if especially relevant we would include them in the usability test as a user story.

3.6 Participants

3.6.1 Interviews

The interviewees were different members of the janitorial staff that were responsible for large areas. This included janitors, team leaders, and department leaders. They all worked for NTNU or for one of the larger high schools in the Trondheim area. The reasoning for this was that the intended end-users were this exact group, and by interviewing them we could be sure our findings would be genuine and relevant. We interviewed a total of twelve people.

3.6.2 User test

The participants in the user test were all people that had participated in the interviews. We wanted and had confirmed user tests with a total of 5 people, but ended up performing user tests with 4 people as 1 person fell ill right before the tests were to take place.

3.7 Ethics, Participants Rights, & the collection and storing of data

3.7.1 Ethics

We as researchers have a responsibility to perform research ethically. As stated in [7, pp. 54–67], there are several rights a person has when participating as a subject in research, and we as ethical researchers are responsible for making sure the subjects are aware of them. As the project performs research that includes the use of participants on behalf of a Norwegian university we were required by law to submit and had an application to Norsk senter for forskningsdata(NSD) accepted. In this application, we had to detail what types of data we were to store, how we were going to collect them, for how long we were going to store them, how the subjects were to be anonymized, and how we were going to inform the subjects about their rights as participants in the project.

3.7.2 Participants Rights

In research, there are many rights any participants have. These are summarized as follows:

- The right not to participate
- The right to withdraw
- The right to give informed consent
- The right to anonymity
- The right to confidentiality

We as researchers were therefore responsible for making the participants aware of these rights. We did this by sending the consent agreement via e-mail once a person agreed to participate in the project. This allowed the potential participant to read the document in a relaxed environment, without potentially feeling pressured to agree to participate.

The consent agreement was approved by NSD beforehand, in order to ensure that the document would sufficiently inform the participants about their rights. The consent agreement can be seen in its entirety in Appendix B.

3.7.3 Collecting and Storing Data

When we performed the interviews, we decided to store some data about the interviewees. We elected to only store the absolute minimum data necessary. This data consisted of their gender, their age, their contact information, and their connected anonymized identity. The gender and age data were to be used for the prototyping technique Personas, and the contact information was stored in case we needed to contact the people afterwards. This data and audio was stored in a Excel-document that was stored on SharePoint-server hosted on NTNUs servers.

This was in compliance with the guidelines set by NSD. In the application, we also stated that we were to store their place of work, but we elected not to do this as this data was not necessary for us, and could aid in identifying the interviewees as well.

Every interview and user test was recorded on a dictation machine lent to us by NTNU, but was deleted immediately after being transcribed.

Chapter 4

Related work

This chapter contains information about related work. The first part is about other robotics systems like the AGVs in St. Olavs Hospital and other AGVs for hospital environments. We also mention Boston Dynamics as an inspiration and as the benchmark within robotics. We also discuss Robotics Middleware Framework, which is an interesting project regarding operation of multiple fleets of robots with shared physical resources.

4.1 Automated Guided Vehicles (Service Robots)



Figure 4.1: Automated Guided Vehicle transporting a cart at St. Olavs Hospital

Source: [47]



Figure 4.2: Atlas from Boston Dynamics

Source: [48]

4.1.1 Swisslog

Swisslog, a robotic logistics company with roots in Switzerland is the supplier of automated logistics systems in St. Olavs hospital, Trondheim, Norway. They have supplied automated drug management systems, pneumatic tube transfer systems, and automated guided vehicles (AGV) with automatic door opening and elevator operation [49]. The AGVs at St. Olav can take the elevators and will verbally announce their actions to anyone around it. The AGVs have priority when calling the elevators and the elevators cannot be used by people at the same time as the AGVs. Although some claim they can be tricked to allow it [50].

Swisslog AGVs can interface with elevators and doors via I/O signals according to their web page [51]. While we could not find the exact methods, from what we know about them it is probably wireless communication. Elevator suppliers are now providing interfaces to be used with robots. TK Elevator, formerly Thyssenkrupp Elevator, have created an interface to their elevator systems that allow robots to connect and call elevators via WiFi or LTE [52].

4.1.2 HelpMate

Transitions Research Corporation (TRC) has developed HelpMate and published the paper "HelpMate: A Robotic Courier for Hospital Use" [53]. The robot does not transport trolleys, but is tall and has storage compartments built-in. It performs similar tasks to the ones in St. Olav's hospital and operates in a similar way. Krishnamurthy and Evans describe how HelpMate uses elevators. It communicates with the elevators with a radio link interface developed by TRC to allow remote control of the elevators. They also explain how the priority works and that the robot cannot ride the elevators with people onboard at the same time [53].

4.1.3 Pathfinder

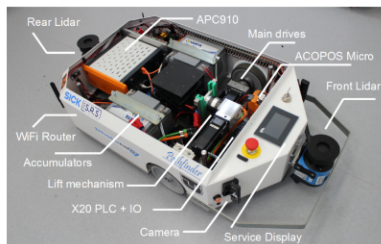


Figure 4.3: Pathfinder, an AGV developed for hospital logistics

Source: [54]



Figure 4.4: Pathfinder carrying payload

Source: [54]

Pathfinder, an AGV developed for hospital logistics, has been built for a similar environment and with similar technology stacks as Alfred. It uses two LIDAR sensors and ROS 1 to navigate. This project was completed before the release of the first ROS 2 distribution. Pathfinder can map the floors by being manually guided around and later perform localization and navigation using AMCL and Gmapping. Because the building they tested Pathfinder in has a human operating the elevators, the robot can also take the elevators by sending an SMS to the operator. When reaching the selected floor, it scans a synchronization tag on the floor next to the elevator which tells Pathfinder which map to load [54].

4.2 Boston Dynamics



Figure 4.5: Spot from Boston Dynamics

Source: [55]



Figure 4.6: Stretch from Boston Dynamics

Source: [56]

Boston dynamics are an inspiration to robotics engineers and enthusiasts around the world. Their humanoid robot Atlas (figure 4.2) is the most advanced robot of it's kind in the world [48]. On YouTube two of them can be seen completing a full parkour course: <https://www.youtube.com/watch?v=tF4DML7FIWk>. While Atlas is currently a research project, they also have commercially available robots like Spot and Stretch (figure 4.5 and 4.6) [48].

4.3 Common service robots

In many larger buildings, there is a widespread use of large cleaning robots with different levels of autonomy. The first autonomous service robots to be widely ac-

cepted for home use are vacuum robots like the ones from Roomba with many other brands available today. A study from 2013 gave Roombas to 9 households to observe the usage, acceptance, and adoption of vacuum cleaning robots. These required virtual walls and the models from 2013 did not use SLAM, but other algorithms for covering the cleaning area such as spiraling, room crossing, wall following, and random walk. The users answered that it was quite easy to use and that it had some perceived usefulness. They did not however think it was intelligent [57]. Since 2013, the robot vacuums have received more features, become far more intelligent, cheaper, and have rapidly grown in popularity.

4.4 Multi robot fleet management system

Managing a fleet of robots generates a new set of challenges and can be considered the next frontier for software like ROS. One attempt at solving the multi-fleet problem is Robotics Middleware Framework (RMF). At the core of RMF are modules for traffic monitoring, dispatch planning, scheduling and additional utilities. The goal is to allow multiple robots from multiple fleets from any vendor to operate in environments with shared resources. RMF incorporates adapters for third party vendors, infrastructure like elevators and doors, workcells with actuators and sensors.

RMF comes in a toolbox (RMF Toolbox) that in addition to its core, includes a traffic editor, simulation tools and a web-based user interface. The traffic editor can annotate floor plans with walls, doors and elevators, and manage traffic lanes. RMF Core is written in C++, however the C++ libraries have ROS2 wrappers, which makes their use language independent.

The RMF project also provides a free to use open source fleet management system, Free Fleet. Free Fleet can be used to control and manage the state of a set of robots. It does not provide planning capabilities like selecting the best suited robot for a task, or managing the use of shared resources.

RMF does not operate in unknown environments. The operating environment must be mapped out on beforehand [27].

Chapter 5

Prototype

In this chapter, you can read about the robot prototype Alfred and the control center prototype. section 5.1 & section 5.2 gives an overview of the hardware- and software stacks used in the prototypes, while the following sections go further into the details of the respective stacks.

The link to a YouTube video demoing the prototype can be found in the ReadMe file of the alfred-task repository at:

<https://gitlab.stud.idi.ntnu.no/indoor-robots/alfred-task> (Shortened: **bit.ly/3zzigZu**)

At the time of writing, the code is available at the NTNU internal GitLab. The code is split in four repositories organized in the GitLab group "indoor-robots" at <https://gitlab.stud.idi.ntnu.no/indoor-robots>

- alfred-remote repository: <https://gitlab.stud.idi.ntnu.no/indoor-robots/alfred-remote>
- alfred-robot repository: <https://gitlab.stud.idi.ntnu.no/indoor-robots/alfred-robot>
- alfred-task repository: <https://gitlab.stud.idi.ntnu.no/indoor-robots/alfred-task>
- arduino repository: <https://gitlab.stud.idi.ntnu.no/indoor-robots/arduino>

alfred-remote contains a package with launch files to start RVIZ in a pre configured setup to control Alfred. Alfred-robot contains packages and a launch file to start all services required on Alfred. Alfred-task contains the task server, frontend, database, rest-api, and video streamer packages. A more detailed description is in the Readme of each repository.



Figure 5.1: Alfred, the robot

5.1 System Architecture

Figure 5.2 illustrates the system architecture with the most important components and the flow of data between them. Ffmpeg, video, nav2, alfred_skid_controller, pose_transformer, slam_toolbox, rplidar are all processes running on the Raspberry Pi on the robot. The robot also has two Arduinos controlling the wheels. An additional server with ROS is required to run the task_server, and rosbridge. Another server is required for the database, API, video_streamer, and hosting the frontend. The GUI communicates with ROS through rosbridge. During usability testing and development, the frontend, rosbridge, task_server, and web services have all been hosted on our development machines.

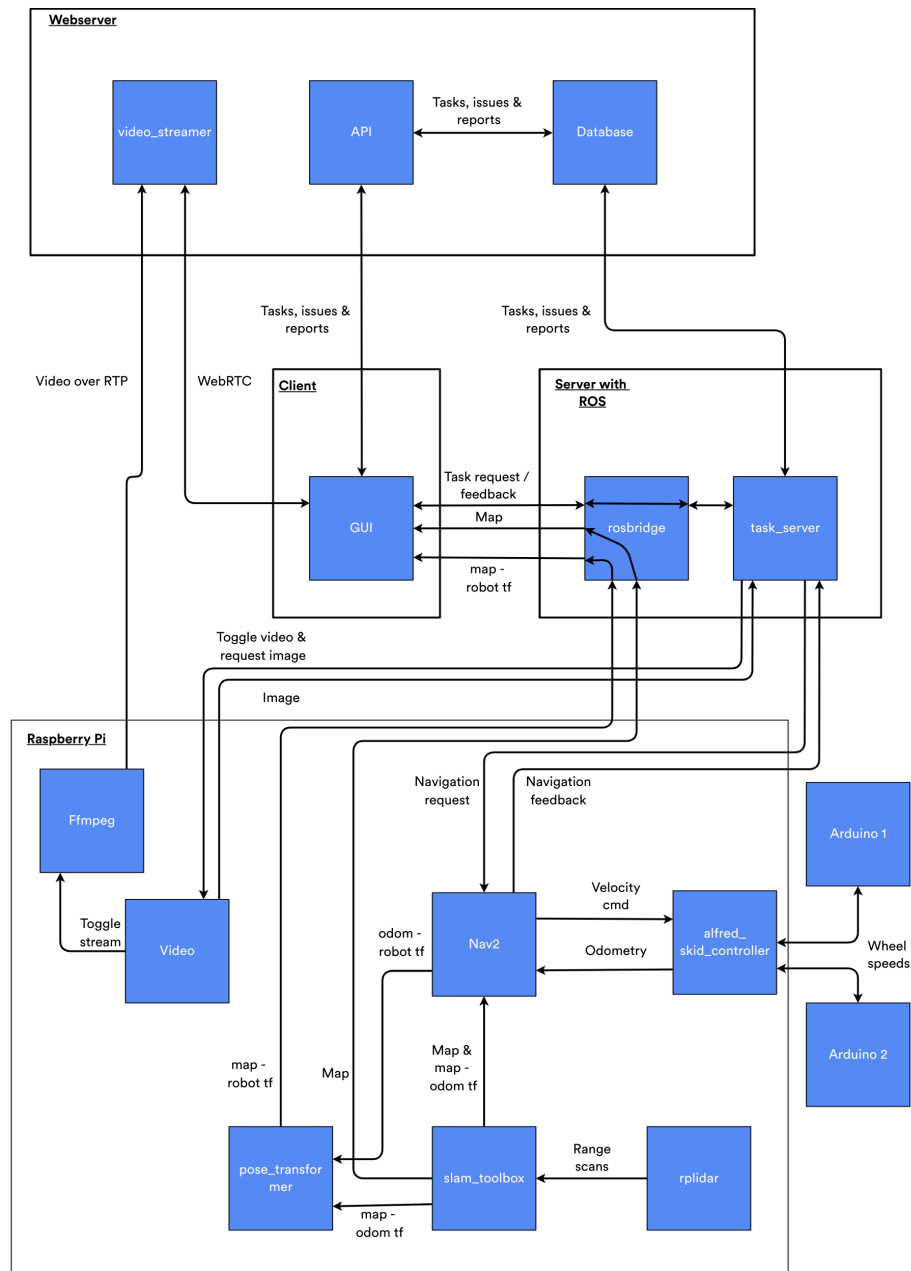


Figure 5.2: System architecture

5.2 Hardware and software stacks

The hardware used is:

- 1x Robot chassis (DF Robot 4WD Outdoor Mobile Platform for Arduino)

- 4x Geared DC motors with rotary encoders from DF robots
- 4x Adafruit DRV8871 motor drivers
- 2x Arduino Nano Every
- 1x Raspberry Pi 4 8GB
- 1x Adjustable buck converter for 12v-5v
- 1x Cut USB-C cable to power Raspberry Pi
- 1x 12V SMF battery
- 1x 12V Car battery charger

The software stack:

- FFmpeg
- Golang
- React 17
- Webpack 5
- TypeScript
- Arduino
- Python 3.8.10
- ROS 2 Galactic
- OpenVPN server and clients
- Ubuntu 20.04 on developer machines
- Ubuntu 20.04 for ARM on Raspberry Pi

5.3 Building a Robot prototype

The prototype, Alfred, has been further developed from the original prototype from the previous project report [27]. The following sections describe the prototype and how it was built. Changes made in both hardware and software is discussed in chapter 7. Figure 5.3 shows a high-level overview of the system architecture.

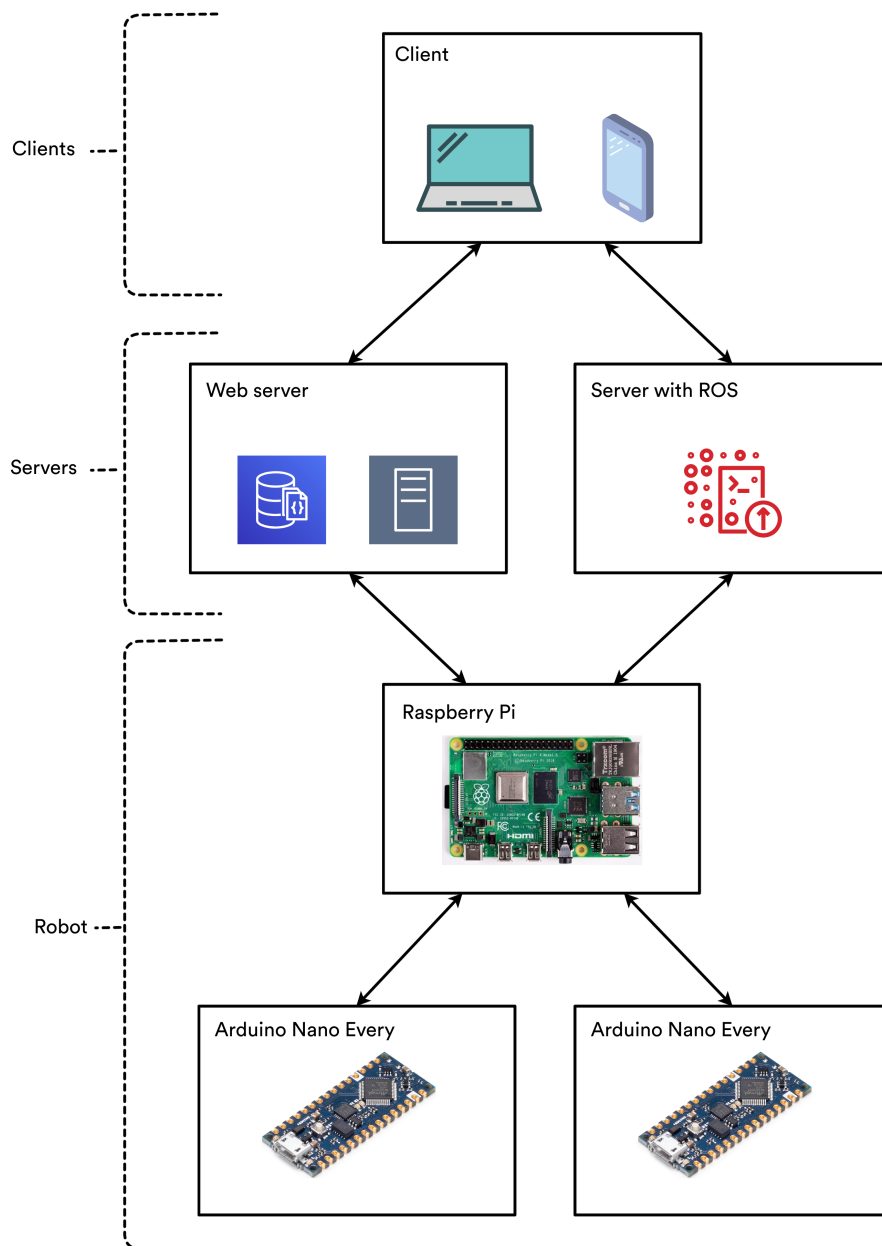


Figure 5.3: High level diagram of the system architecture

5.3.1 Hardware

Alfred is a robot with four wheels with one motor each in a skid steer setup. It is powered by a 12V Sealed Maintenance Free (SMF) battery. A type of lead-acid battery. A buck converter is placed in the power supply compartment along with

the battery to supply the Raspberry Pi with 5V power (Figure 5.7). 4 motor drivers of the type DRV8871 power one motor each, with two Arduino Nano Every boards, controlling two motor drivers each. See figure 5.6 for a picture of the motors and control hardware. Figure 5.4 illustrates how the motor control hardware is connected, with figure 5.5 as a more detailed version.

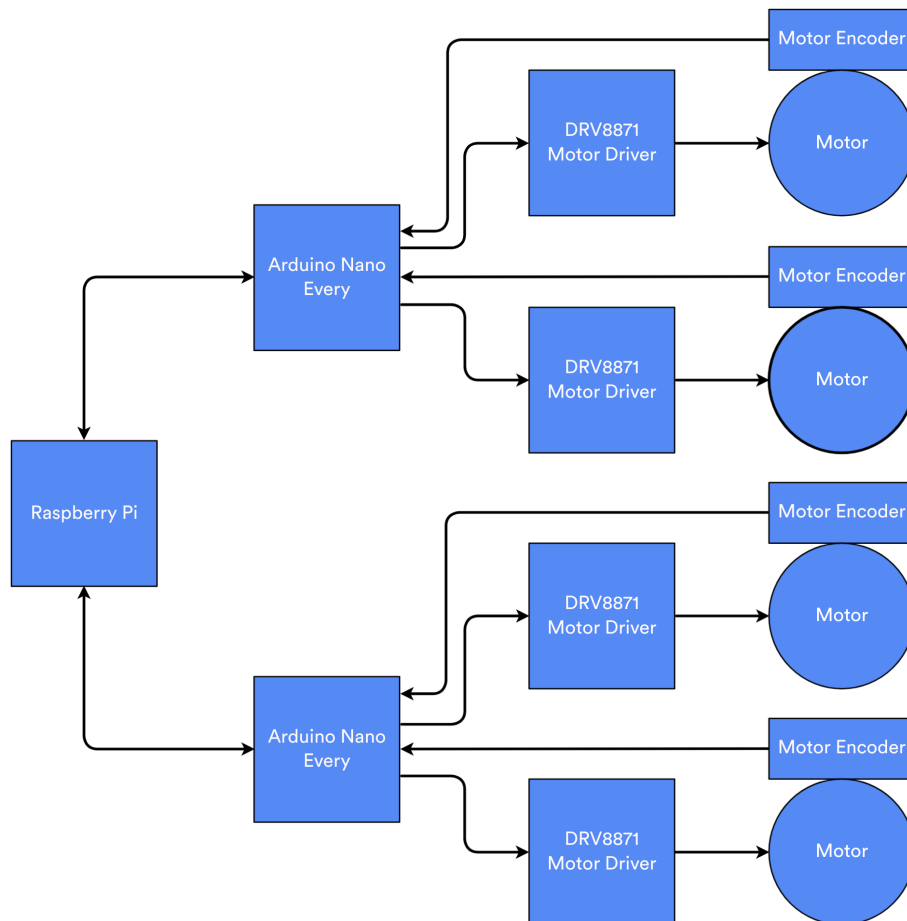


Figure 5.4: Motor control architecture

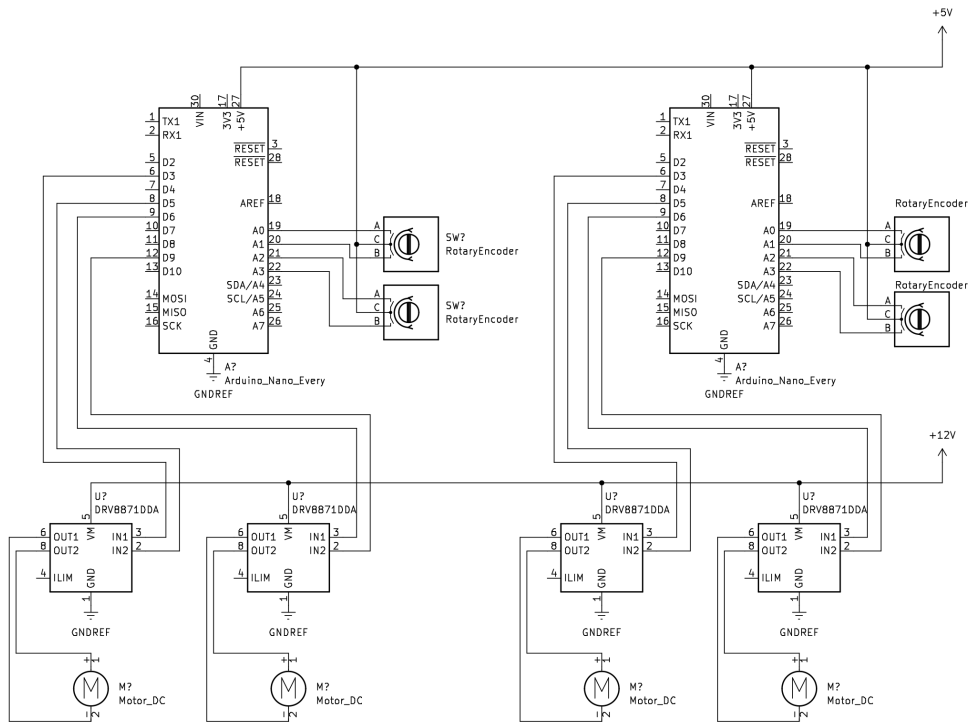


Figure 5.5: Wiring diagram for motor controls

Source: Appendix F

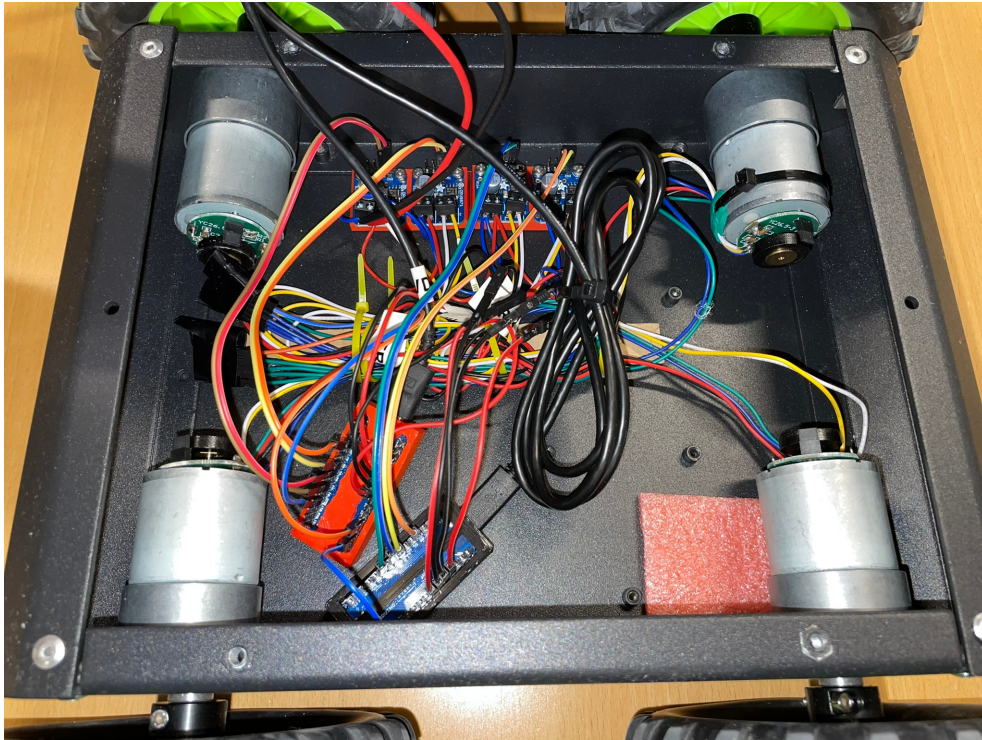


Figure 5.6: The Arduino and motor control compartment of Alfred

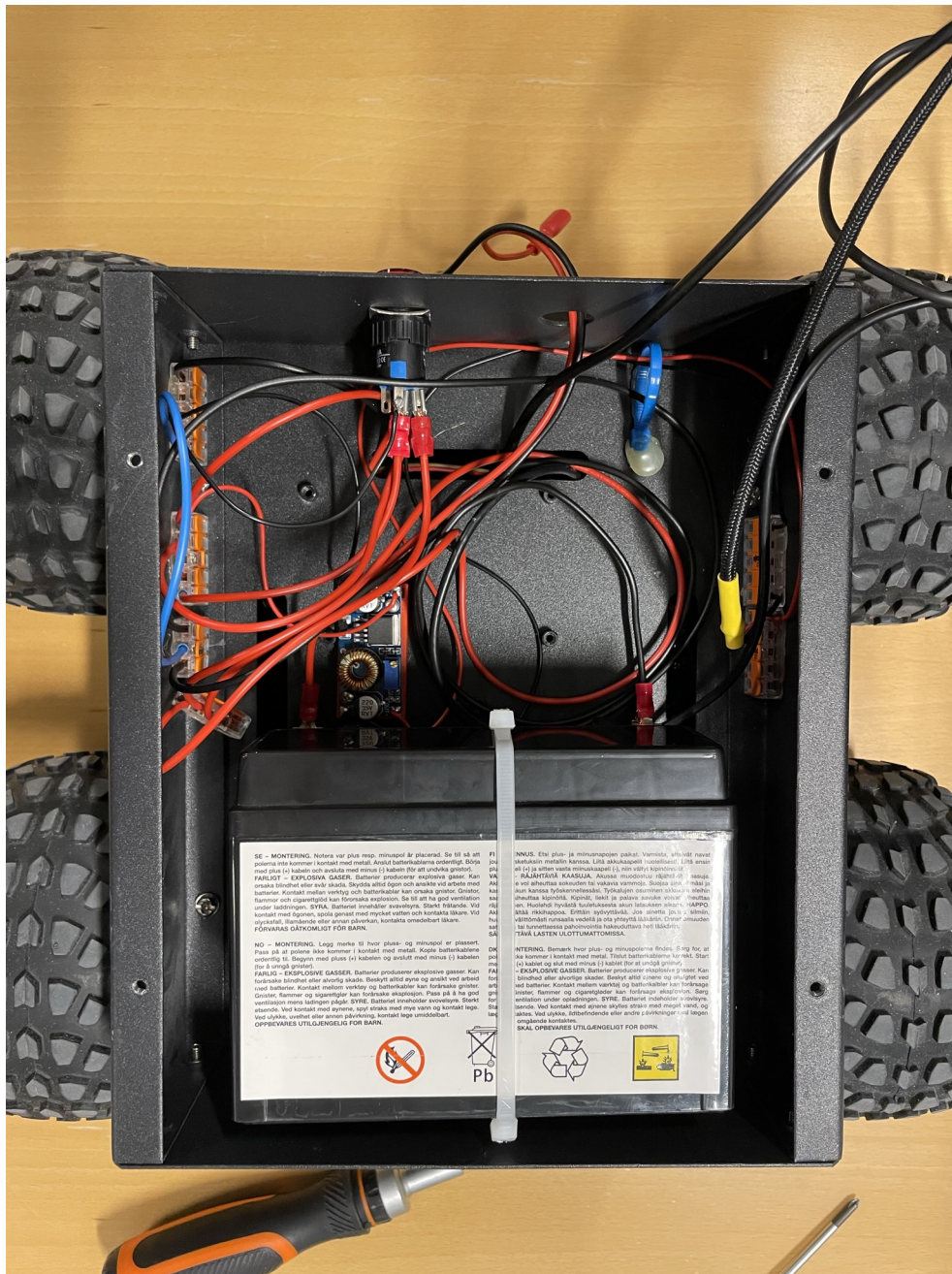


Figure 5.7: The battery and power supply compartment of Alfred

5.3.2 Navigation

Alfred uses Navigation 2 (NAV 2) with LIDAR, wheel-odometry, and SLAM-Toolbox for mapping, localization, and path planning. Figure 5.8 and 5.9 visualise Alfred's perception and comprehension. The figures are from a simulated world which 5.8 shows on the left, with the map and localization of a simulated robot similar to Alfred has created on the right. 5.9 also shows a global and local costmap which is important to path planning.

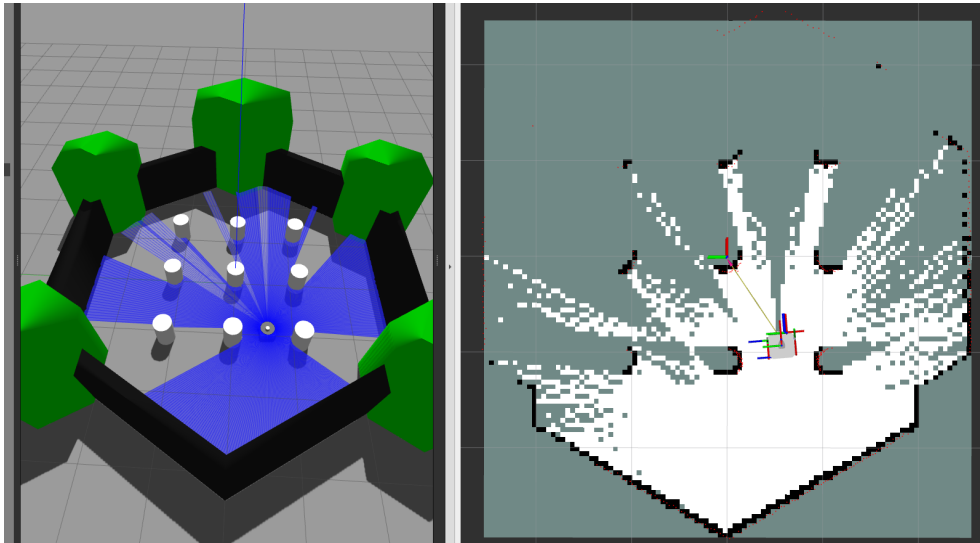


Figure 5.8: Gazebo on the left, RVIZ on the right. Simulated robot performing SLAM in a simulated environment

5.3.2.1 Costmaps

There is a global costmap for calculating the full path to a destination, and a local costmap for refined movement closer to the robot's location. Areas close to obstacles have higher costs, and obstacles will be considered non-traversable (See figure 5.9).

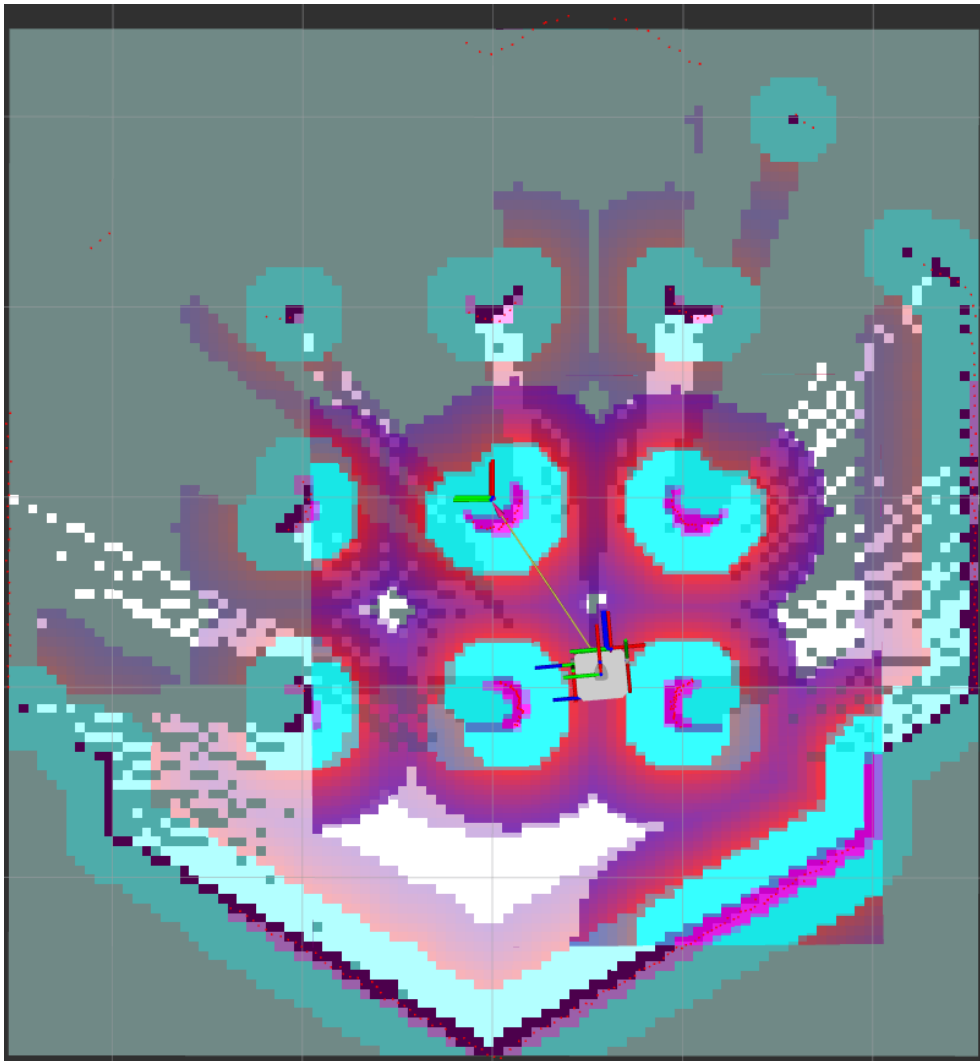


Figure 5.9: A simulated robot with map, pose, local, and global costmaps visualised in RVIZ

5.3.3 Robot control

Alfred is a four-wheel drive skid steer robot. It is controlled by adjusting the left and right wheel speeds to turn either on the spot or with a larger radius during a forward movement. It is similar to a differential drive robot with two driving wheels and a omnidirectional caster wheel but is different and more difficult to model because the wheels must skid.

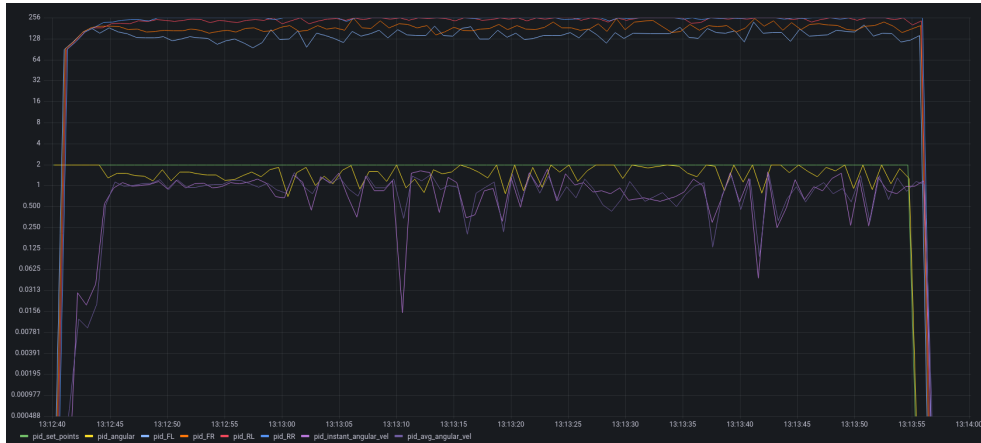


Figure 5.10: Debugging control values

NAV 2 publishes velocity commands with linear and angular velocity components to a topic to which `alfred_skid_controller` subscribes. `alfred_skid_controller` then transforms the commands into left and right wheel speed commands. The speed of each wheel is regulated by a dedicated PID controller for each wheel. The PID controllers have a linear velocity for the specific wheel as a setpoint and as feedback, calculated from the ticks counted by wheel encoders. It outputs a control signal ranging from -255 to 255. 8-bit PWM voltage output pins on the Arduino boards control the voltage input at IN1 and IN2 on the motor drivers to control speed and direction. (See figure 5.5).

Figure 5.10 shows data we collected while tuning the controller and debugging problems we experienced. The top four graphs show the absolute PID-control values for each wheel, and they show us a problem with the skid-steer setup. The two rear motors are operating at near maximum torque as the controller is sending a power request of 255 and just below. It is difficult to regulate a process with actuators operating near their limit. We believe the two front wheels receive a lower control value because the battery sits above the rear wheels, meaning less weight is on the front wheels resulting in less traction and less torque required to spin them.

The green line shows the requested angular velocity in the test, which was 2 radians per second (rad/s), and as one can see from the purple and pink lines, the measured value was roughly one rad/s. This is due to the difficulty of turning



Figure 5.11: Outliers in angular velocity

a heavy skid steer robot with low-powered motors. We experimented with using another PID controller to control the angular velocity, but it makes the controller more complex, requires more processing power, and creates more delay. The robot also tends to jerk rather than turn smoothly, which is difficult to regulate with a PID controller.

Even though the robot uses LIDAR-based SLAM to localize itself, it also depends on odometry data, which we create from the wheel encoder data. `Alfred_skid_controller` calculates a transform from the origin pose to the robot's current pose. We experienced strange anomalies, with the robot suddenly jumping outside the map. Initially, we believed it to be due to the IMU calibration, but our debugging showed the issue stemmed from the wheel odometry. Figure 5.11 shows outliers in the measured angular velocity the controller uses to calculate odometry. We further tracked down the issue to one wheel encoder intermittently reporting an unrealistically high amount of ticks. We fixed this problem by writing code that removes outliers.

5.3.4 Live video

Alfred streams live video using RTP to a server that relays RTP packages to connected clients using WebRTC. The stream can be stopped and started using ROS services. With all components connected via VPN, Alfred can stream video to the open socket on the WebRTC server. This also ensures the peer-to-peer connection for WebRTC works even on more advanced networks like Eduroam.

Sending a live video feed from a low-powered device such as the Raspberry Pi through WiFi and a VPN service is not a trivial task. Video streams can require both significant amounts of bandwidth and CPU power and may suffer from high latency. Our chosen solution is shown in figure 5.12. The Raspberry Pi on the rover uses a program called FFmpeg to send a video stream over Real-time Transport

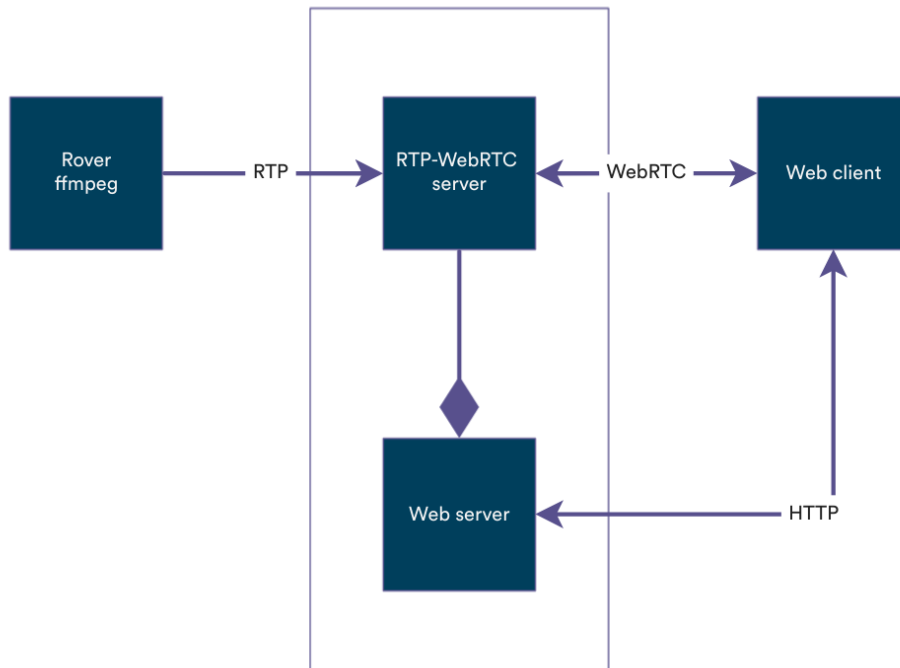


Figure 5.12: Live stream architecture

Protocol (RTP) to a server. Compiling FFmpeg from source on the Pi with the correct options selected allows the program to utilize the Pi's built-in hardware-accelerated encoding, using the h264_4l2m2 encoder, which in turn uses Video4Linux 2 (V4l2) to access the Pi's hardware codecs. We could also have used a Raspberry Pi camera, which would likely be easier to use with hardware-accelerated H.264 encoding, but we had already mounted a Logitech C930e web camera on the rover. We initially intended to use it for VSLAM, but later scrapped VSLAM for the prototype. The C930e is a wide-angle camera that can capture pictures and video in 1080p, with better quality than a Raspberry Pi camera, so we decided to stick with the C930e once we got hardware accelerated encoding to work.

5.4 Control system and Human Robot Interface

The control system consists of multiple packages found in the `alfred-task` repository (<https://gitlab.stud.idi.ntnu.no/indoor-robots/alfred-task>). The frontend is a single-page web application built with React, Typescript, Webpack and SCSS. See figure 5.13 for a picture of one of the pages in the web app. All the pages in the web app can be viewed in Appendix A. The backend consists of the Task Server, Video Streamer, REST-API and a noSQL database. See figure 5.3 and 5.2 for system architecture.

The control system is task-oriented, with robots being the resources that can be used to complete tasks. The Task Server is a ROS package we created to act as the main backend part of the control system. It is written in Python and is based on ROS. It is meant to manage multiple robots and tasks at the same time. Tasks can be requested by clients to be executed right away or on a timer, either with a specific robot or automatically selected robots. In the current version, timers are not implemented, and it has not been tested with more than one robot even though it was built to handle multiple. The timer functions were added to the frontend and timer information for tasks was stored in the database, so it was possible to perform usability testing on that function. The task server can control robot navigation tasks through NAV2's Simple Commander API. This is a wrapper around topics, services, and actions related to navigation tasks, for simple integration into other applications.

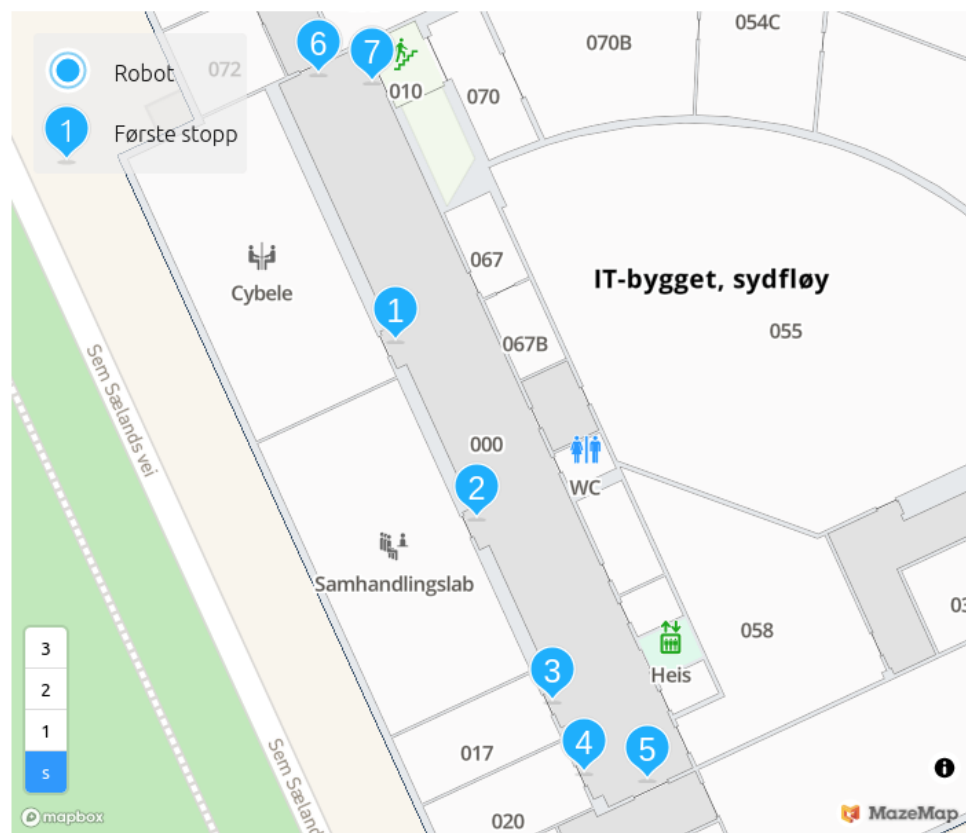
Client communication with the task server is performed through ROS topics and services, and the frontend can use ROS topics and services through a Rosbridge Websocket server using the JavaScript library `roslib`. We created a special service message type for the task server to send and receive JSON messages. Allowing many different types of messages to be sent without creating new ROS message interfaces for each type. A Python Flask REST-api is used to access the MongoDB database. The task server uses `pymongo` to access the database directly. The `rosbridge_suite` documentation does not have information about security, and it remains unclear how security would work with a frontend communicating with ROS nodes through Rosbridge. It is possible that Rosbridge cannot be used securely without using a VPN.

Ny patrulje

Tittel:

Beskrivelse:

Trykk på de stedene i kartet roboten skal kjøre innom i løpet av patruljen.
Roboten vil ta bilde på hvert av stedene.
Ønsker du å slette et punkt du har plassert, kan du trykke på det.



i Markørene viser i rekkefølge hvor roboten stopper for målinger

Lagre patrulje

Figure 5.13: Web application: New patrol page.

5.4.1 Figma Prototype

The literature review gave guidelines for how the Human-Robot Interface should be designed and can be summarized as such:

- The interface should aid in maintaining remote awareness/situational awareness
- If there is a camera view from a robot, it should be a forward-facing scene
- Map and video should be available when manually controlling a robot, to aid in maintaining situational awareness
- Manual control should be of a high level
- If there is autonomy in the picture, the human should still be kept in the loop
- Flexible input for steering the robot
- Use HCI techniques for finding a design
- Only show information that means something to a non-robotic expert
- Using a keyboard, a Nintendo Wii, a PS3 controller, and the touch screen of a mobile phone are all suitable for steering a robot

From the interviews, we had gathered some information that was helpful when designing the rest of the control center. Age-wise, we had interviewed a pretty homogeneous group as all except one person was in the age range 53-62 years old. We had also gathered from the interviews that they were efficient in using computers or cell phones, as all the interviewees reported that they used an operation management system like Lydia or Microsoft Teams to see what work that needed to be done. Based on these two things, we know we had a user group that was at the least somewhat efficient with using technology, but at the same time would be more likely to like a simple and intuitive design. With these things in mind, we first created a design system in Figma which can be seen in the figures 5.14 and 5.15.



Figure 5.14: A part of the design system that shows what colors we were to use, how all buttons should look like, what fonts we were to use, and what input fields should look like

With this as a base for the design, we made a few iterations of the design in Figma, and the final iteration of the Figma prototype can be seen in the figures 5.16, 5.17, 5.18, & 5.19.

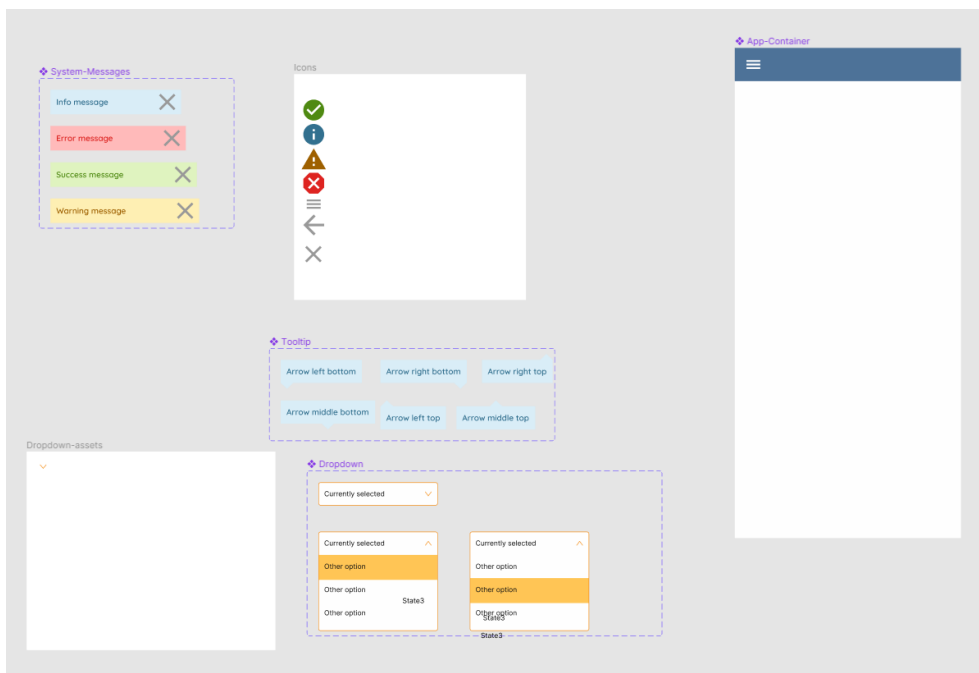


Figure 5.15: A part of the design system that shows how notifications with icons, dropdown, tooltips, and the navbar should look.

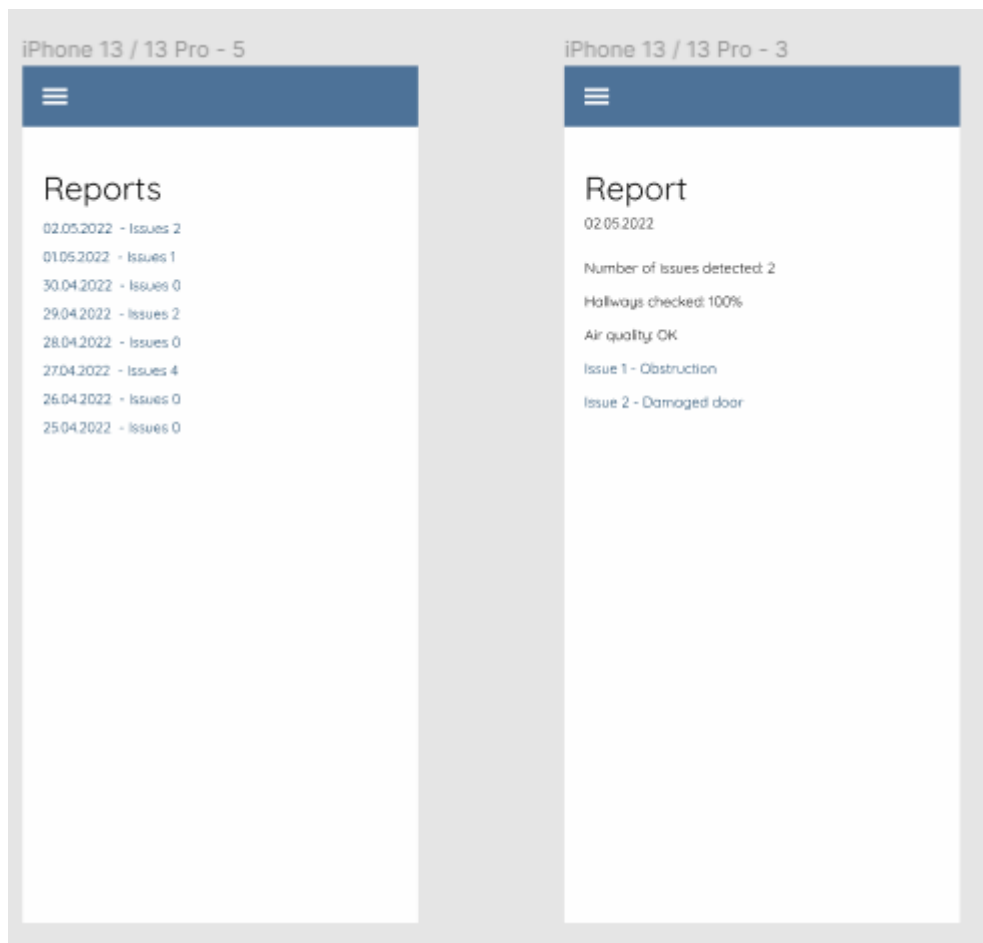


Figure 5.16: The proposed designs for the report list and a report

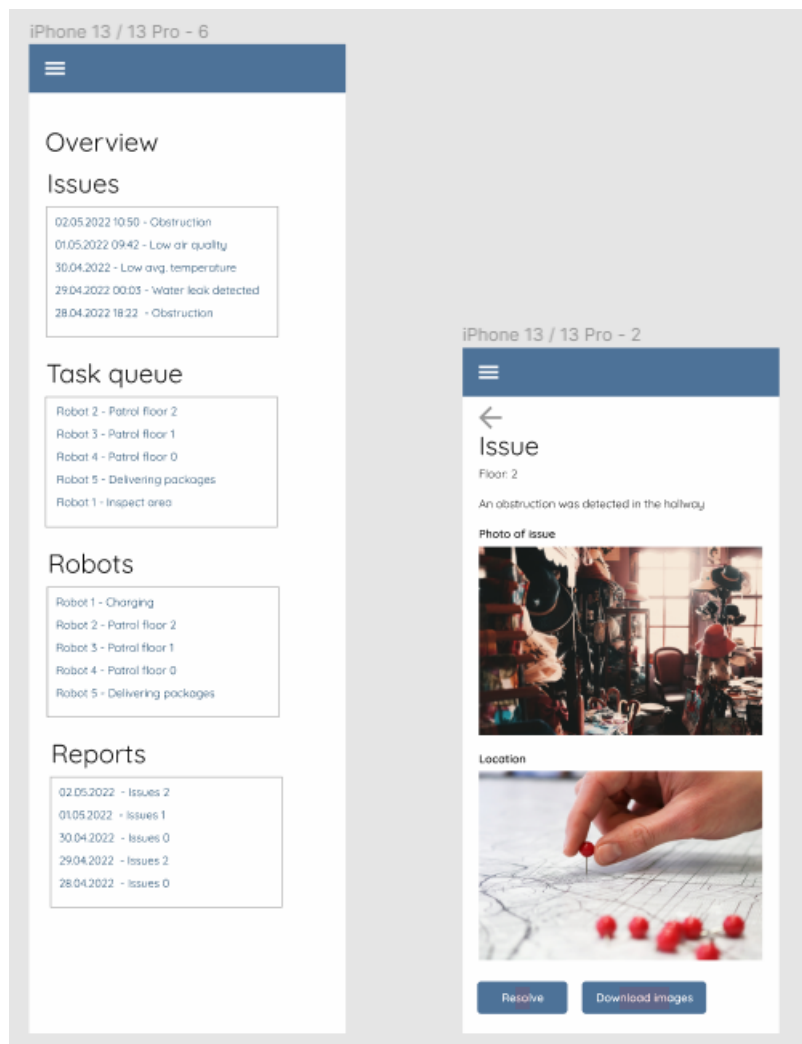


Figure 5.17: The overview-page design is meant to quickly give the user an overview and an understanding of what needs their attention.

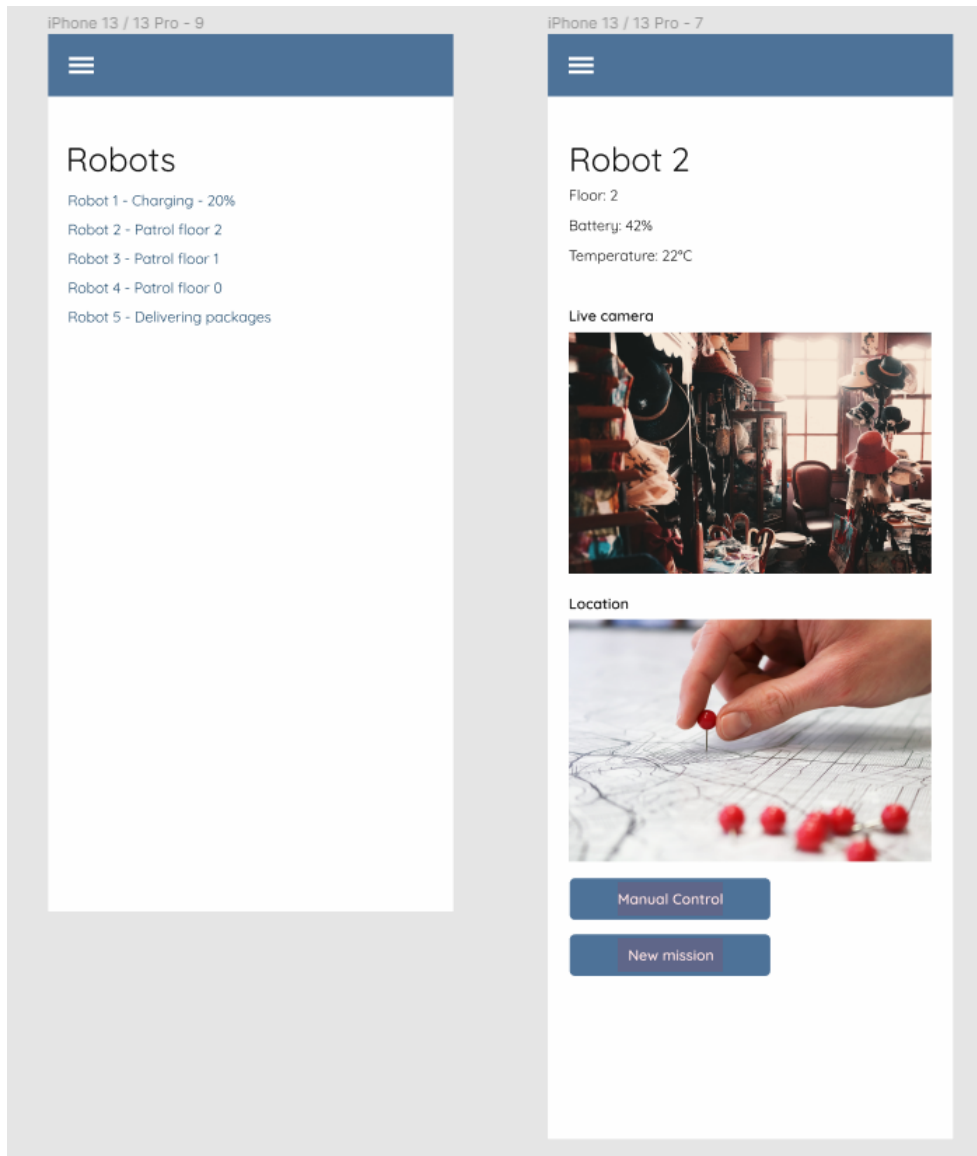


Figure 5.18: The suggested design of the list displaying all robots and a robot's status page when you click on it in the list

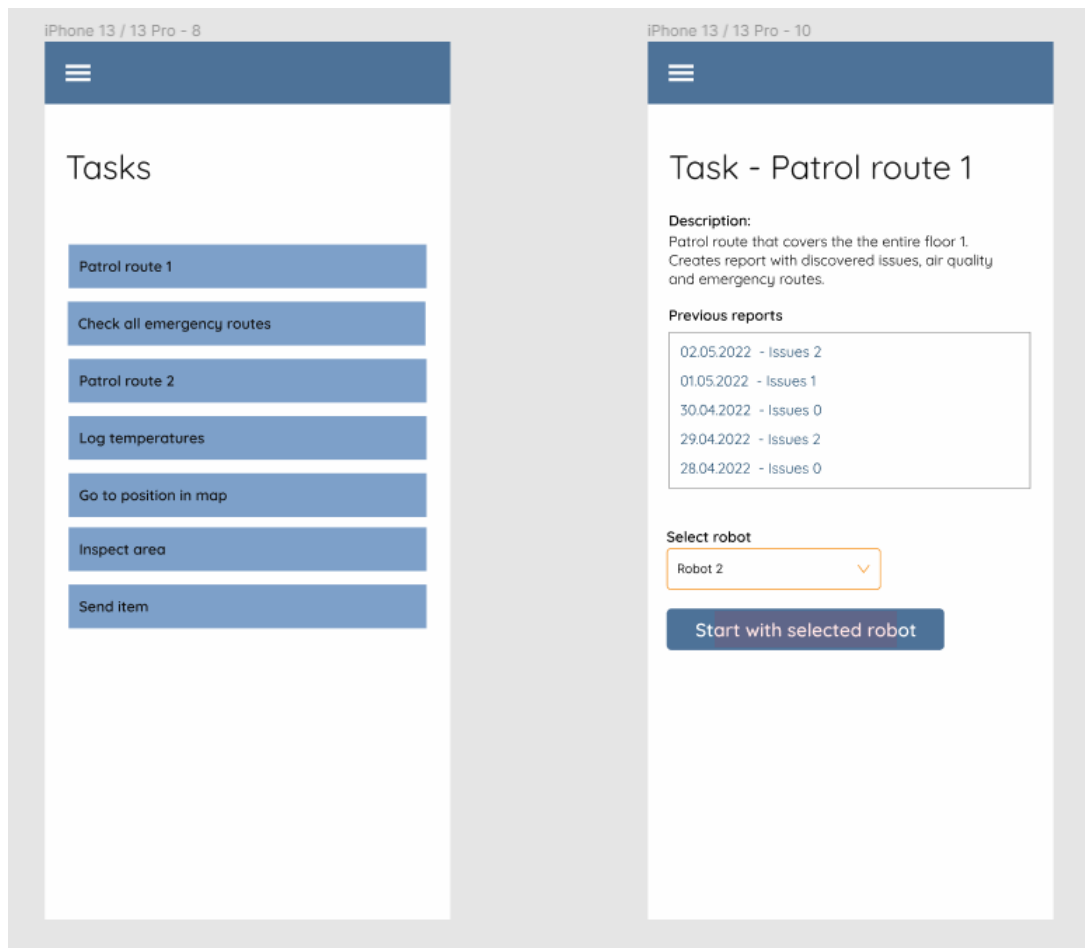


Figure 5.19: The suggested design for a page that shows all stored tasks and the page of a single task that can be accessed when you click on it. Here you can see all reports that have been made on all the previous executions of the task and how you can select a robot to execute it

5.4.2 Functionality

From the interviews, we had gathered three types of tasks that could be helpful for the janitors in their work:

- **Transport missions:** Missions where the operator of the mission places a payload on the robot and sends it to a location where a recipient receives the package.
- **Patrol missions:** Missions where the operator selects one or more places the robot is to go to and perform various measurements and take photos of.
- **Inspection missions:** Missions where the operator wants to inspect something visually

The difference between the patrol missions and the inspection mission is the level of autonomy involved in the mission. The patrol mission would be a sort of start-and-forget mission, where the operator would not think about the mission until they decided they wanted to read the report. The inspection mission on the other hand would require more attention from the operator and would involve the operator inspecting a location using the robot. The difference between these missions can also be seen as a fully autonomous mission and a semi-autonomous mission.

In the previous section you can read about how we used Figma to prototype the design of the prototype, but Figma is a powerful tool for prototyping functionality as well. In 5.20 you can see how we created interaction logic in the Figma prototype using flows. You can see a demonstration of how it looks in this video: <https://www.youtube.com/watch?v=ojaGM7q6yeY>

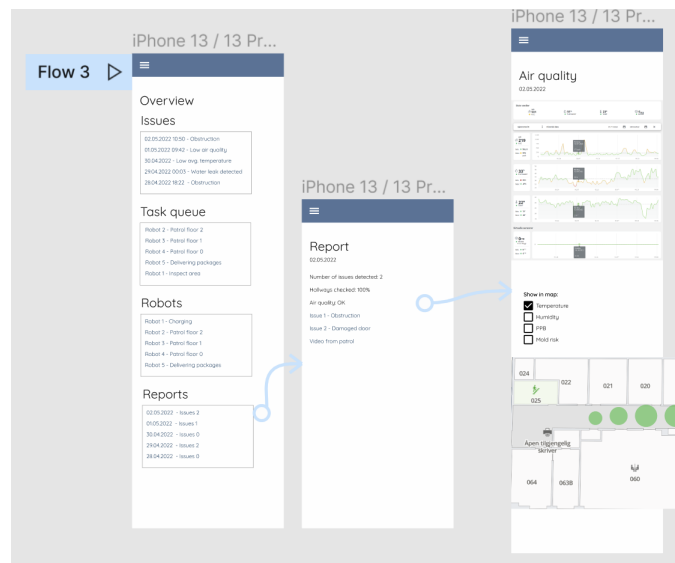


Figure 5.20: A screenshot that shows how scenes are connected using flows to create interactions in the prototype

5.4.3 Running the Control Center Prototype

The API, database and video service has docker files and docker compose files to be started in docker environments with a single command. Hosting the frontend can be done in the same way, but has only been running in development mode with a webpack development server for the usability testing. The docker compose file can be seen in figure 5.21.

```

1  version: "3.9"
2  services:
3    api:
4      build: ./web_api
5      ports:
6        - "5000:5000"
7    database:
8      image: mongo:latest
9      ports:
10     - "27017:27017"
11     volumes:
12     - ./mongodb/data:/data/db
13    video:
14     build: ./video_streamer
15     ports:
16     - target: 5004
17     published: 5004
18     protocol: udp
19     mode: host
20     - "8000:8000"

```

Figure 5.21: Docker compose file

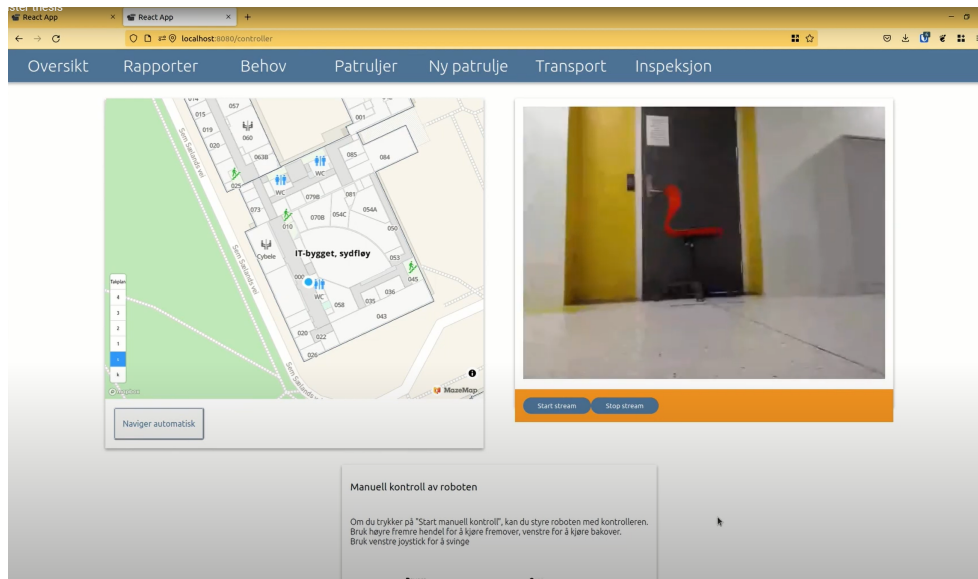


Figure 5.22: A screenshot of the GUI when the user manually controls the robot. The map and the robots position is on the left, and the video stream from the robot is on the right

5.4.4 Controlling a robot

As we discovered in the literature review, when a user is performing a task, the user needs to maintain situational awareness. The higher level of situational awareness, the more likely it is that the decisions the user takes are the best ones. We therefore chose to include both a video stream that displayed the forward scene of the robot, so the user would be able to see what was happening locally and to include a map that showed the robots position so the user would be able to see what was happening globally. This was also in accordance with the guidelines.

The literature review showed that both a keyboard and video game controllers are useful for steering a robot, so we chose to include both a keyboard and a Xbox One controller. We also wanted to include the option of using a Xbox controller as we thought it was likely that some of the users had played driving games before, and that the combination of controller and forward-scene would make the steering very intuitive. The keyboard was included as a backup option, in case anyone had problems with using the controller. In addition to teleoperation, we included the possibility to send a robot to a location by clicking on a or several locations in the map.

5.4.5 Map

Because the robots create their own maps using SLAM, they contain no geospatial metadata which is necessary for humans to experience good usability. The maps must also be more complete and put in a context, such as the world, a country, or a

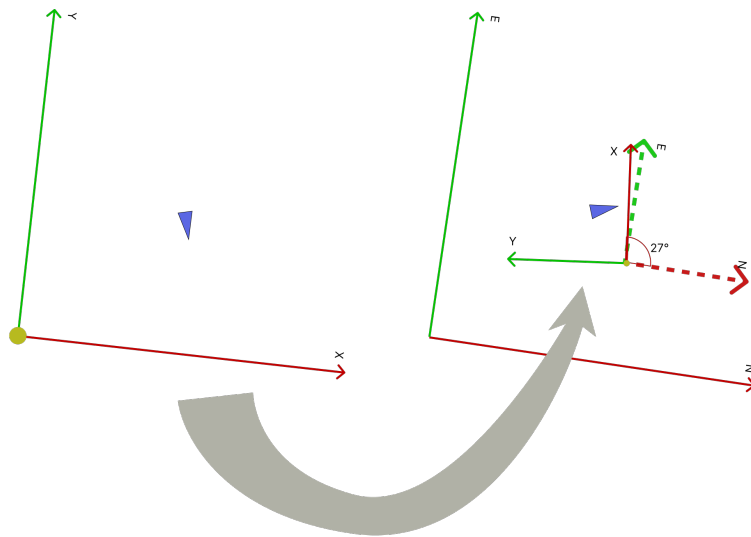
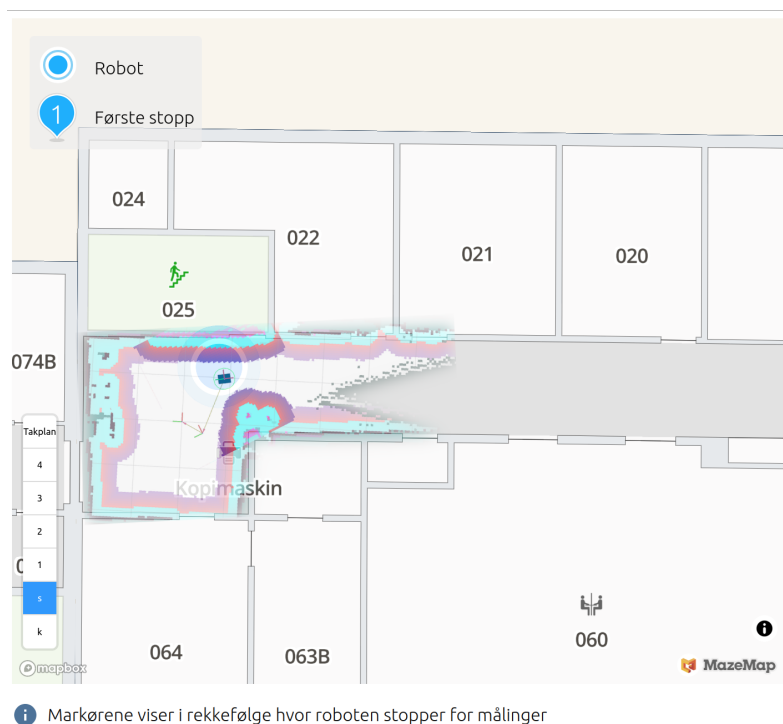


Figure 5.23: The robot pose is transformed from the map origin frame to GCS. The world frame are the largest axes, the original frame are the stippled axes, while the transformed frame is the smaller, solid axes



Markørene viser i rekkefølge hvor roboten stopper for målinger

Figure 5.24: Mazemaps with transformed map layered on top

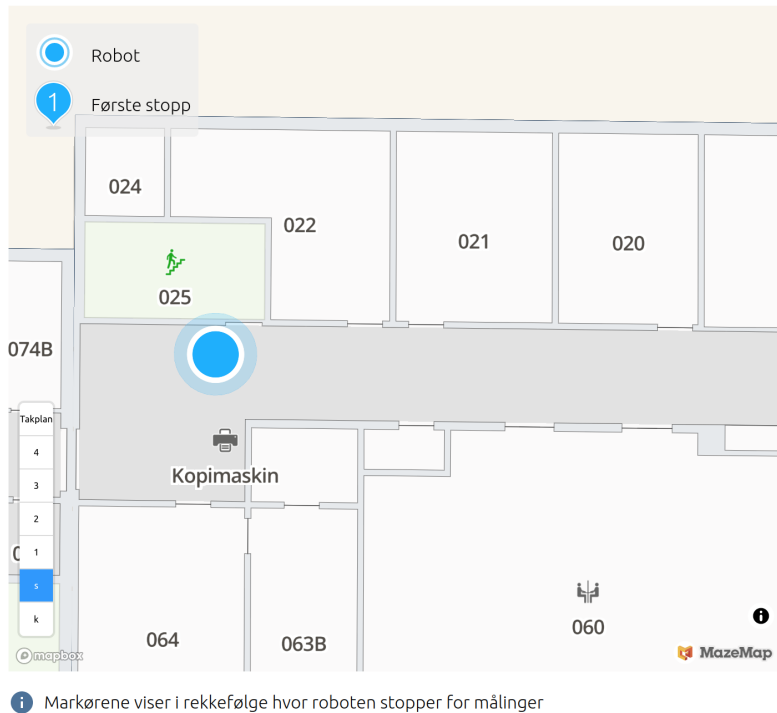


Figure 5.25: Mazemaps with robot position transformed onto it

campus. To do this, we used Mazemaps API to integrate Mazemaps into the control center application. Mazemaps uses the geographic coordinate system (GCS). We created functions to transform poses back and forth between GCS and the map-origin frame from the robots' maps created with SLAM. Figure 5.23 illustrates the transformation from the cartesian coordinate system of the robot into GCS, and figure 5.24 illustrates a robot's map transformed into GCS. This allows displaying the position of robots on a Mazemaps map and transforming marker positions and clicks on the map into coordinates used by the robots. See figure 5.25.

In the prototype, the cartesian map origin frame is given a GCS coordinate and orientation. When the map origin frame is placed in GCS, it is possible to transform any poses relative to the map origin into GCS with a rotation and a translation. The opposite operation is used to transform GCS coordinates into the cartesian map origin frame.

Using SLAM and map APIs like Mazemaps allows deploying the robot system without manually creating any maps. Mazemaps offers opportunities to create public or private maps for your buildings, with lots of customization options possible. Integrating SLAM with existing map solutions such as this can be a powerful combination. Another approach can be to create static maps for the robots from CAD drawings, which is what Mazemaps says they do to create their maps [58]. Robots can navigate with static maps using Adaptive Monte Carlo Localization

(AMCL), included in Nav2 [59].

5.4.6 Issue detection

There are several issues relevant for the system to detect and notify about. They could be water leaks, warm electric installations, blocked emergency exits, damages, air quality issues, and more. For the prototype, we tested the implementation of a temperature and humidity sensor, and automatic object detection and classification. In figure 5.26, the control system has detected a chair blocking a door. For the prototype, any detected object of certain classes seen from a given pose in a patrol will trigger an issue. This is done using OpenCV 2 and a pre-trained neural network, MobileNet SSD. The object detection code is based on a tutorial by Sears-Collins aka Automatic Addison [60].

This issue can be sent as a Microsoft Teams notification, to Lydia, or any other ticketing system integrated with the robot system. It is important to not create another ticketing system that can make users feel like they have been given the burden of yet another software system to worry about. An example of an MS Teams notification can be seen in figure 5.27.



Figure 5.26: The Control system has detected a chair

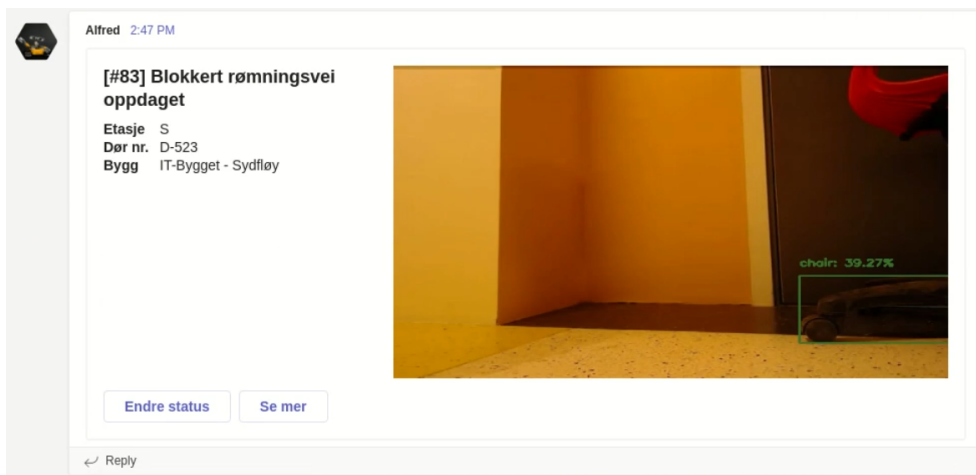


Figure 5.27: Control system sent notification in MS Teams

Chapter 6

Results

In this chapter, you can find the analyzed data from the usability tests & the interviews, descriptions of the final iteration of the robot Alfred and the command center prototype, and a description of how the command center and Alfred communicate together.

6.1 Interviews and observation

Task	Score
Air quality information	4.0
Automatically generated status reports	3.6
Cleaning	3.2
Recycling	3.2
Notifications of obstructions	2.7
Transportation	2.5

Table 6.1: Perceived usefulness of tasks after interviews

Table 6.1 shows the most useful task categories in order with a usefulness score on a scale from 0-5, where 5 is most useful.

6.2 Robot prototype

Alfred is the name of the current prototype. It has four-wheel drive in skid steer configuration. Two Arduino Nano Every boards control four Adafruit DRV8871 DC Motor Drivers, where each power one brushed DC motor with a rotary encoder installed. The Arduino boards transform speed commands from a Raspberry Pi 4 8GB into variable PWM voltage signals, count encoder ticks, and relay them to the Raspberry Pi via USB serial. An RPLIDAR-A1, connected to the Raspberry

Pi via USB, is mounted on top of the chassis scanning 360° around the robot in 2D. A Logitech C930 webcam provides live video and images. We tested using an Adafruit BNO055 IMU, but removed it due to problems with calibration. A DHT11 temperature and humidity sensor is used for air quality readings.

Alfred manages to map and navigate the entire hallway of IT-bygget, Southwing. When navigating that space, the CPU usage is sometimes nearing 100%. When that happens, the navigation and control performance is impacted.

6.3 Control center prototype

The control center consists of multiple packages found in the alfred-task repository (<https://gitlab.com/kevjohandevs/alfred-task>). The frontend is a single-page web application built with React, Typescript, Webpack and SCSS. The backend consists of the Task Server, Video Streamer, REST-API, and a noSQL database.

6.4 Interviews

6.4.1 Results of interviews

The interviews gave good insight into the work of janitors, making it easier to understand what they might need. There were also some specific ideas about what robots can do that came to light. We performed 8 interviews, 2 of which were with 3 janitors at the same time. That makes a total of 12 janitors interviewed. They worked at various schools, which we will not list due to privacy. Figure 6.1 shows the janitors' perceived value of several potential robot system functions. The scores are extracted from interpretations of answers to interview questions and various quotes from the interviews. The following types of functionality were considered to be potentially useful:

1. Automatically generated status reports
2. Notifications of obstructions
3. Notifications of damages or problems
4. Air quality information
5. Air temperature information with regards to energy efficiency
6. Cleaning
7. Recycling
8. Delivery of small and large items

The automatic status reports would be a summary of all information gathered by the system from a certain period of time, such as a weekend. It would include any notifications of problems, damages, water leaks, obstructions, air quality, and temperature deviations.

Notifications of obstructions can be anything from small items and trash to tables,

Person	Robot function / value	Automatic deviation & damage reports	pictures of wrongly placed things & obstructions in hallways and emergency routes	Air quality information with regards to health and economy	Cleaning	Help with recycling	Delivery of items
1			1		3		
2		3	3				
3		2	1		4		
4		4			3	3	2
5		4			3	3	2
6		4			3	3	2
7		3	3			4	
8		4	4	4	4		4
9		4	3	4		3	
10		4	3	4	3		
11		4	3	4	3		
12		4	3	4	3		
AVG		3.6	2.7	4.0	3.2	3.2	2.5
		No value	Little value	Neutral	Some value	Much value	Did not answer
		0	1	2	3	4	

Figure 6.1: Perceived value of robot functions

chairs, pallets, and anything that could be incorrectly placed anywhere. The main focus would be to keep emergency exits and routes clear of obstructions. All janitors have stated that they have routine checks of fire safety equipment and escape routes. Most of them reported that they do this often enough already and that obstructions are usually discovered and cleared quick enough. One janitor however told us about much more frequent routines than the others, and that his team cannot always do it as often as they want to. He also told us that the guide lights for emergency exits need to be checked regularly, and is supposed to have automatic self-checks, but that this system no longer works. This means they need to perform the checks manually. During our user tests he also admitted that while they think that they discover problems with these lights quite quickly, they recently found one that seemed like it could have been broken for quite some time. He continued to say that the problems may exist for much longer than they think.

Notifications of damages or problems involve automatically detecting deviations outside air quality and temperature readings, damages to any surfaces, doors & electrical system, water leaks, and gas leaks. The robots can be equipped with air quality sensors, computer vision, thermal cameras, and water & gas leak sensors.

Air quality and temperature information can involve temperature, humidity, CO2, Volatile Organic Compounds (VOC), Radon, and Particulate Matter (PM). Most buildings have automation systems for controlling ventilation and heat with regard to safety, comfort, and energy efficiency. The amount of distributed sensors is limited and the systems could have faults that may be detected with an external system such as this. The collected data can be used for overviews, insight, and comparison with the installed automation systems.

The janitors interviewed all work in buildings with well-established routines for

cleaning with cleaning personnel and machines. Some have automated cleaning machines as well. They are all positive to autonomous cleaning. Two of them also envisioned a robot that could efficiently pick up gravel. One of them wanted a better robot vacuum that could handle large amounts of sand and fine gravel indoors with automatic self emptying. They had tested one robot that was supposed to handle it, but it did not have high enough production quality. It had to use pre-programmed paths, quickly filled up, and had a high cost. The other janitor wanted a robot to pick up gravel from the roads and lawns in the outside areas of the school. A large amount of gravel is spread around the roads and lawns when snow is cleared and gravel is used to improve road grip on icy roads.

There are challenges with regard to recycling. Plastic currently needs to go into residual waste because it would likely not be cleaned enough, and users would likely throw incorrect types of plastics in the plastic bins. Cans and bottles often have a lot of remaining liquid. Users also throw residual waste in paper & cardboard, and in bottles & can bins. There are also special goods that need to be handled. The janitors tell us that the bins are emptied before they become overfilled and that they do not need notification of full bins. Some of them would however like to improve the recycling process.

Delivery of items came up in one of the interviews as a janitor told us they often receive small packages to the janitor's office, which are then supposed to be delivered to the front desk, or to some teacher. We then had the idea that robots can do delivery tasks. Other janitors also told us they deliver paper and toner to printers around campus. During interviews after user tests and during observation we also learned that janitors also often transport pallets and large bins. There is some positive, but slightly varied response to transportation functionality.

6.5 Observation

In addition to the interviews we conducted, we shadowed a team of janitors for half a day to observe them. We were present in their daily morning meeting where they discussed any relevant matters and planned their current day. Some team members needed access to certain areas for some assignments. As some areas were restricted with a minimum amount of people that had access, they sometimes needed to bring someone with access with them. They had worked overtime during the weekend due to a party event happening over two days with 600-700 people. There were also reports about damages and they discussed what needed to be done about them. The leader mentioned that there would be an exhibition with several thousand extra people in addition to the regular operation without extra staff. Without extra staff, and with thousands of extra people on campus, issues like full garbage bins, lack of paper towels & toilet paper, spills on the floor, and more would be more frequent and occur in different areas than they were used to. The rover could potentially be useful for observing these things. Also, fire escape routes could be blocked even more easily due to the increased amount of

people and distracted janitors and cleaners working harder in other areas. He said they really have to plan and divide their human resources well to continue smooth operation during events like this, and that is exactly something rovers could assist with.

After the meeting, we followed some of them on their assignments. We joined a small group tasked with throwing away four large recycling containers full of old books. We helped move the containers and helped with throwing away the books. We moved the containers quite a distance and thought that a much bigger and more powerful version of the rover could help do this instead of having to do this manually.

We also helped move an ice machine, and during this task, one of the janitors saw a small garbage bin that was full and stopped to empty it right away as we were passing by. This both shows they will eventually find and empty full bins, and that they move around the building so much that they will often find them, but also that using robots to observe and notify could be useful, given the random nature of finding this one. The team leader told us that they would soon receive a new model of small paper recycling bins with a compactor installed, which was supposed to reduce the number of times they had to empty them. They had tested a previous version of the prototype earlier and were going to test a more finished version this time around. The bins were developed by NTNU students which made it into a business. If the paper is compacted into a suitable container, it could integrate with rovers that pick it up and transport it to a recycling station. Alternatively, rovers could observe an indicator on the bins and notify janitors about a full bin. It might be cheaper to have an offline indicator on each bin rather than network-enabled indicators.

The leader recommended that we spoke to the campus security to discover use cases for the robot, and stressed how relevant the rovers perhaps could be due to the small number of people working in this department and the large area they had to cover. He also mentioned some issues which showed how necessary campus security can be. There had been cases of people living on campus, people wandering around without belonging on campus, or even in restricted areas.

After spending only a few hours with them we realized they had to walk a lot during the day, and asked them about it. The leader told us that he did not have any numbers on this, but confirmed that the team walks a lot through a work day. He also spoke about how sore and stiff they would get the first days after a vacation, but would become used to it after working for some time. The team leaders' area of responsibility was a large building and they would often have to walk back and forth a lot, and this would be done with steel-toe shoes on hard floors. Therefore it seems like if rovers could reduce unnecessary steps, there could be health benefits to it.

6.6 Usability tests

Person	SUS Score	SUS Score	Perceived usefulness
2	47.5	Awful	1.75
3	92.5	Excellent	3
7	90	Excellent	3
9	77.5	Good	3.75
AVG	76.9	Good	2.88

Table 6.2: SUS-Score and perceived usefulness after usability tests

6.6.1 Pilot study

The pilot study revealed a few usability issues and some problems with the tasks and interview questions. A few of the questions were almost duplicates and some had a very leading wording, so we removed some of them. The pilot test user has much experience with technology that one cannot assume for the actual test users. Because of this, he could provide constructive feedback about the interface. It also meant whenever he was unsure other users would definitively have trouble. We found that the lists in the interface did not look clickable enough. To improve it, we made sure to make the style of all lists consistent throughout the application and added an "open" button that was clearly clickable. This was clearly a smart choice, as all subjects clicked this button instead of clicking on the item itself. The meaning of the markers on the map (rover position, goals, checkpoints, and user position) were not clear enough. We added a legend on the map that explains all the relevant markers to fix this. There were also cases where the user did not receive feedback when submitting forms or starting tasks, so we made sure to add that. The pilot test user was very familiar with Xbox controllers and gaming but still needed a quick test to learn how it worked for the rover. Knowing it would be more difficult for other users, we added graphics to show them how to use it with the rover. Additionally, there were a few missing translations and a need for some improved wording.

6.6.2 Observations during usability tests

Notable observations from the user tests were:

- Patrol
 - Unopened issues on the overview page were confusing
 - Users expect to use a map for an overview
 - Some did not discover the top navigation bar right away

- Some users tried to use the "open" button on issues in the overview as a way to start using the application
- It is unclear which locations in reports have discovered issues
- Users expect to be able to click on images and map markers to get more information
- The term "patrol" (patrulje in Norwegian) was difficult to understand at first
- Most users used the timer function to start the patrol in one minute, when asked to start it now
- Transport
 - When using the transport feature, users expect to have to first send a robot to them, load it, and then choose where to send it
- Inspection
 - Some users expected to get started with manual control right away and did not understand that they could send the robot to a location first before controlling it manually
 - Two users tried to adjust the camera using the right stick on the controller
 - Most users first tried to use the patrol feature for inspection.

6.6.3 Overview page

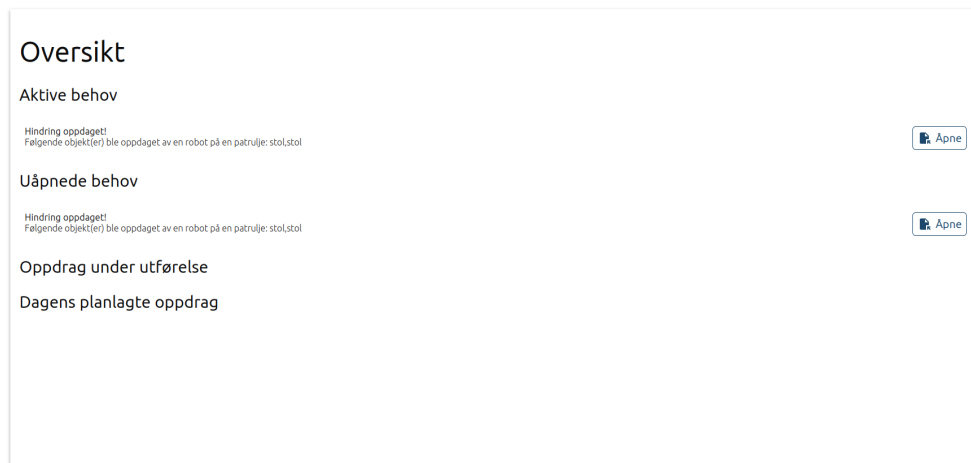


Figure 6.2: Web app: overview page

The tests revealed several opportunities to improve the usability in the prototype web application. Figure 6.2 shows the front page of the web app, meant to quickly grant an overview. It is meant to only contain important information so that the user could quickly see if any actions must be taken or not. The users clicked on open almost right away without taking the time to understand the content and

without discovering the top navigation bar of the web app. Some users were confused about opened and unopened issues. Unopened issues were listed twice, so they thought they were two different issues. Some users expected to use a map for an overview.

6.6.4 Report page

Figure 6.3 shows an example of a report that the users were asked to understand. This report stems from a patrol in which the assigned robots have taken pictures at each location in the patrol, which was two locations in this case. Each visited location is shown in the map with a marker, which can be seen in figure 6.4. Some users had trouble understanding which locations contained issues and which did not. They also expected to be able to click on pictures or on the markers on the map to get more information, which would have improved information flow.

Opprettede behov:

Hindring oppdaget!
Følgende objekt(er) ble oppdaget av en robot på en patrulje: stol,stol

[Åpne](#)

Data samlet inn ved lokasjoner

Lokasjon 1

Temperatur: 21°C
Luftfuktighet: 40%



Lokasjon 2

Temperatur: 21°C
Luftfuktighet: 40%



Figure 6.3: Web app: Report example



Figure 6.4: Web app: Report example, map

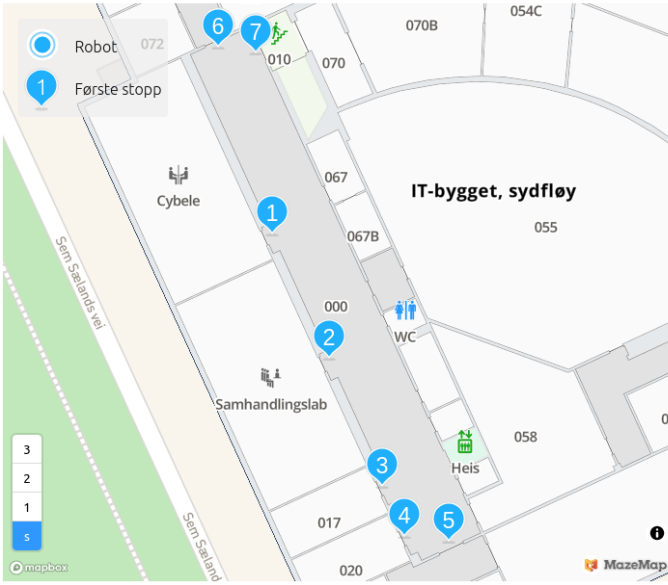
6.6.5 New patrol page

Ny patrulje

Tittel:

Beskrivelse:

Trykk på de stedene i kartet roboten skal kjøre innom i løpet av patruljen.
 Roboten vil ta bilde på hvert av stedene.
 Ønsker du å slette et punkt du har plassert, kan du trykke på det.



3
2
1
s

Mapbox MazeMap

Markørene viser i rekkefølge hvor roboten stopper for målinger

Lagre patrulje

Figure 6.5: Web app: New patrol view

Users can create new patrols in the new patrol view, as can be seen in figure 6.5. They can add a name, description, and multiple locations for a robot to visit. The robot takes pictures and other measurements in each location, and can also continuously take measurements along the way. A patrol can then be started immediately or be set on a timer to be repeated. The users used some time to understand the term "patrol", which is to be expected, but they quickly understood the concept once they got to try to create one and see it in action. See figure 6.6.

Patrulje: IT-bygget, sydfløy, etasje S

Helgepatrulje. Undersøker alle dører.

i Markørene viser i rekkefølge hvor roboten stopper for målinger

Gjenta oppdrag?

Patruljeplan

Dager aktiv:

ma ti on to fr lø sø

Hvor ofte i døgnet? ▼

Fra kl:

Velg robot: ▼

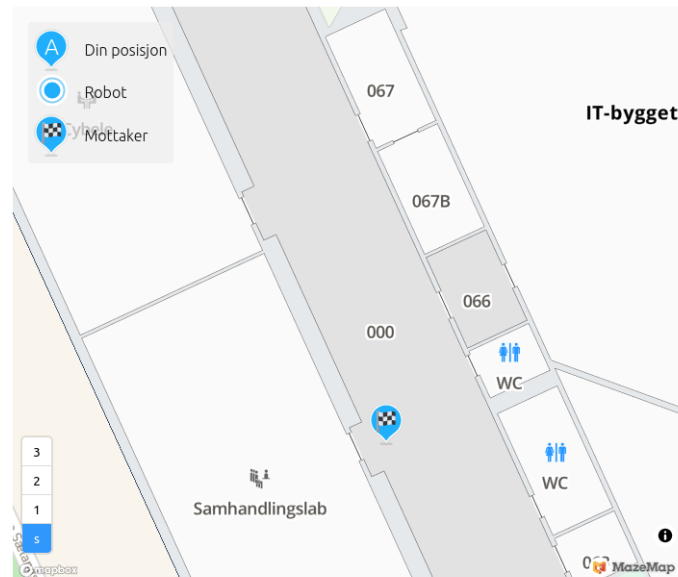
Figure 6.6: Web app: Example of patrol

6.6.6 Transport page

The transport function allows a user to send an item from them to any location. See figure 6.7. The transport view uses the users' current position retrieved from the web browser and allows the user to select the destination either by clicking on the map or searching for a room. Status feedback is then shown in the transport view, including whether the robot is on its way to pick up or deliver, waiting for users to load or unload and the current location. The robot itself is equipped with a touch screen, currently emulated by an iPhone. Once the robot arrives, the user can click continue as soon as it's ready. When users tried this function, they thought they had to get the robot to their location first, and then choose where to send it.

Transport

Velg hvor du vil sende noe. En robot kommer til din posisjon og plukker opp forsendelsen. Last opp og trykk fortsatt på selve roboten når den kommer.



Trykk i kartet eller søk etter stedet du vil sende noe her.

Send en robot til deg

Figure 6.7: Web app: Transport view

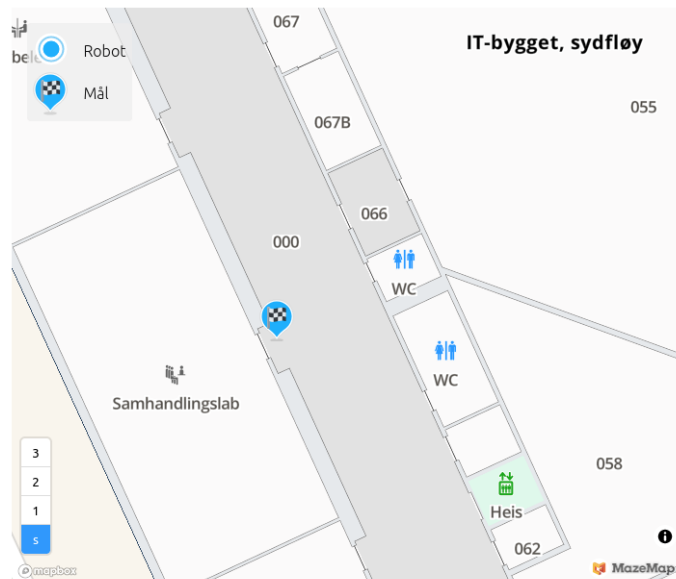
6.6.7 Inspection page

The inspection function allows users to send a robot anywhere and then assume manual control once it arrives to use the camera for inspection. The older users

had some difficulty with the Xbox controller we used to control the robot, but using the controller illustration with instructions they learned how to use it fast, and were all able to get a view of the door handle we asked them to inspect. Two users tried to adjust the camera using the right stick on the controller. Some did not understand they could automatically send the robot to its destination before assuming manual control. This could be because the distance was short, the robot was in their line of sight, and they had the controller in front of them. For both the transport and the inspection tasks, several users first tried to use the new patrol function to do these missions.

Inspeksjon

Velg et sted du ønsker å inspisere. En robot med kamera drar dit og varslar deg når den er fremme. Du kan velge å automatisk ta bilder, eller bruke manuell kontroll når den er fremme.



Trykk i kartet eller søk etter sted her

Start

Gå til manuell kontroll

Figure 6.8: Web app: Inspection view

Chapter 7

Discussion

In this chapter, we will put our research methods under scrutiny, analyze and discuss our results using TAM, and discuss the results found in the usability tests. We also discuss the prototype and the choices made to build it.

7.1 Prototype

7.1.1 Alfred

During interviews, it was well known that the current prototype had its limitations. The users knew this, but they were testing a prototype that to them had as much realism as possible and seemed as close as possible to a system that would be ready for production. So far building the prototype has been cheap. Using cheap motors, chassis, motor drivers, batteries, cameras, LIDAR, and more. To make it production-ready, we see from testing with the prototype, that it needs a perception of the environment in 3D, as discussed in 7.1.6. That means the robot at the very least, requires a more expensive 3D-LIDAR. It could alternatively, add one or more RGB-D cameras such as a Intel RealSense 435. It needs more accurate control of its driving and steering, which can be achieved with a 2-wheel drive + caster wheel setup. That does not necessarily need to increase costs. One can also consider using ESCs instead of Motor control ICs as they will provide better control of the motors.

The greatest challenge are allowing the robots to open and move through doors and elevators. If the robots cannot enter any rooms or cross hallway sections blocked by doors, their use is very limited. If they cannot use elevators, transport tasks are very limited, and in most cases, one will then need one robot per floor. These problems are great challenges technically, bureaucratically, and financially. Many areas have restricted access, and operating doors and elevators come with safety concerns. Operating key-card readers, door handles and elevator buttons are still problems for the most advanced robots like Boston Dynamic's Spot

or Atlas. With the minor business value the system adds, those features will be too expensive, and are not an option today. Several logistics vendors interface robots with doors and elevators today. In the related works chapter 4 we mentioned AGVs at St. Olav from Swisslog who claims their AGVs can interface with elevators and doors and TK Elevators who has created an interface for robots to use the elevators they supply. We encourage further research in interfacing with infrastructure using network communication. The current SLAM-generated maps are 2D cartesian maps restricted to one floor level. To allow operation on multiple floor levels, a multi-map system with map switching is required. [54] describes one way of switching maps based on the floor level.

7.1.2 Camera

At the time of the usability tests, the camera was statically mounted on the robot, but as mentioned in the previous paragraph some users tried to adjust the camera angle using the right stick, but there was one other subject that asked the moderator how one could adjust the camera. This indicates that the users found it intuitive that the camera could be adjusted. In the future, it would therefore be beneficiary to use a camera that is mounted on a sort of dynamic arm that can be controlled by the user. This could either be in the form of an additional camera mounted in such a fashion, or continue to use a single camera mounted in this fashion. Our suggestion would be the former, so the user always would have the forward scene and the possibility to gain additional awareness by looking around.

7.1.3 Hardware changes

Several hardware changes have been made throughout building the prototype. We have also attempted designs that have failed. The two L298N motor drivers have been replaced with four DRV8871. The Arduino Nano Every + Arduino Uno setup has been changed to two Arduino Nano Every boards. We tried using a BNO055 IMU, but did not get better accuracy, and we tried using monocular VSLAM which provided less accuracy.

Previously we used the L298N motor drivers, but after experiencing burnouts on a number of the motor drivers, requiring us to replace them each time, we elected to try a different motor driver. Wanting to continue in the spirit of the project, which is to keep the cost of the robot as inexpensive as possible, we chose the relatively cheap motor driver DRV8871. The main difference between the motor drivers is that one of the old motor drivers was sending control signals to two wheels, while the new motor driver only sends control signals to one wheel. This means that the robot now has one motor driver per wheel. We suspected that the cause for the aforementioned burnouts was the combined control signals sent through the driver. This was all but confirmed as none of the the motor drivers has demonstrated similar behaviour. Other reasons may be that the old drivers were low quality.

We previously used one Arduino Nano Every and one Arduino Uno to control motor drivers and read feedback from the wheel encoders. There were inconsistencies in the wheel encoder data and communication problems with the Raspberry Pi. We found that the Arduino Nano Every had fewer communication failures. To test the wheel encoder data, we conducted a test where we manually pushed the robot 2 meters and observed the distances reported by each encoder, assuming each reading should be as similar to the others as possible. We found that the most consistent data was produced while using two Arduino Nano Every boards instead of one Uno and one Nano Every. This test was also used to set calibration parameters for each wheel encoder, adjusting for any differences. On the Arduino Nano Every, all digital pins can be used as interrupt pins, while the Uno only has two pins usable for interrupts. Each encoder requires 1 interrupt pin, meaning one Every can theoretically be used for all encoders.

7.1.4 IMU

In [27] we expressed a wish to include an IMU-sensor in our SLAM setup. We spoke about using an IMU-sensor integrated in an Arduino board, but the relevant Arduino board was not available in any shops, so we bought and installed the Bosch BNO055 sensor instead. This is an IMU that has a magnetometer, accelerometer, and a gyroscope. This means that the sensor can measure 9 degrees of freedom. The board has an onboard processor that performs sensor fusion on the measurements. A ROS node reads the fused measurements and publishes data in quaternion format.

7.1.5 VSLAM

In [27] we also expressed a wish to perform VSLAM (SLAM using visual sensors such as a monocular camera) instead of SLAM. This led to us purchasing and mounting the monocular camera Logitech 930C on the robot. This camera was chosen as it is relatively cheap, staying in line with the philosophy of the project, and because of its relatively wide field of view. In [27] we discussed replacing SLAM-Toolbox with ORB-SLAM 3 as ORB-SLAM 3 is considered state of the art. When we tried to use ORB-SLAM 3 we discovered that in order to use this algorithm with NAV 2, we had to extract the data generated by ORB-SLAM 3 and convert it into the grid map format NAV 2 requires (similar to the work done in this paper [61]). This was not an option for us, as neither of us was very familiar with the data outputted by the algorithm, so doing this conversion would be too costly time-wise to create ourselves. We therefore chose to look at the older versions of ORB-SLAM and other SLAM algorithms that did this for us, but to no avail. Although we weren't able to find a VSLAM-algorithm that was able to be painlessly interfaced with NAV 2 and ROS 2, we were able to find OpenVSLAM that had comparable results to the original ORB-SLAM algorithm [62] but were only able to produce odometry data. We compared the odometry data generated by the algorithm and wheel-encoder odometry data, and the latter data was much

more accurate. We therefore scrapped the idea of using a camera in the SLAMming altogether.

7.1.6 Perception in 2D and 3D

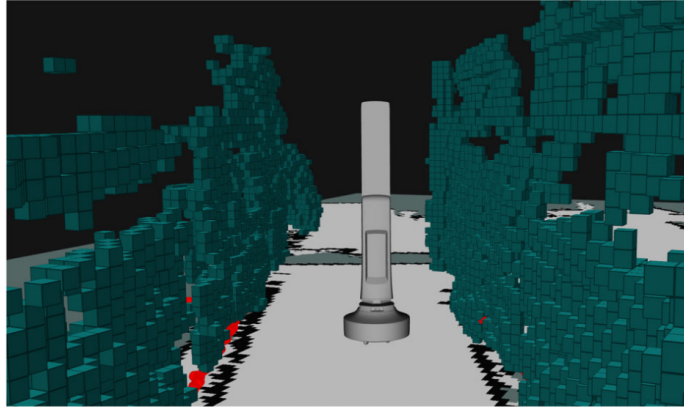


Figure 7.1: A frame of the internal voxel grid being projected over the robot's environment in a real-world retail environment with a red laser scanner for ground truth

Source: [63]

Alfred uses only a 2-dimensional LIDAR to create costmaps. A problem with this are obstacles that sit lower than the LIDAR, or hanging obstacles that are above the LIDAR sensor will not be registered. Some robots use LIDARs with multiple layers or stereo depth cameras like Intel RealSense cameras. NAV 2 allows external costmap plugins such as Spatio-Temporal Voxel Layer (STVL) to create 2D costmaps based on 3D sensor data from LIDAR or cameras [63]. According to Macenski and Simbe Robotics, in a presentation at ROSCon 2018 Madrid, from their experience in real-world applications, it is very important that the robot is aware of its surroundings in the 3D world, particularly because of people. An example they used was that children sometimes hug their robots and test the robot's reactions [64]. Figure 7.1 shows an example of a robot using STVL.

7.1.7 Steering and control

The most common form of steering and control for robots in projects like this is a differential drive robot with two driving wheels and one omnidirectional caster wheel. This form of robot setup is simple to control accurately by adjusting the left and right wheel speeds to turn either on the spot or with a larger radius during a forward movement. We chose a robot with four driving wheels in a skid steer setup, which is similar to a differential drive robot in many ways. With four-wheel drive, it is theoretically more capable for carrying loads or overcoming obstacles like door sills. It does however have a major disadvantage when it comes to control

because in order to turn, the wheels must slide. This is much more difficult to model, as it requires significant amounts of torque to overcome friction, which when overcome can result in unexpected movements. We initially marginalized this issue, thinking it would be less accurate, but still manageable when modelled as a differential drive robot. While we successfully created a prototype this way, it did make the process much more challenging and time-consuming.

7.1.8 Live video

The optimal solution would be to provide a video feed through ROS2, but it looks like the most common way of doing this are through sending individual images in series instead of an actual video stream, requiring much more CPU power and bandwidth. We cannot spare any more CPU power, as the Pi needs as much as possible available to the navigation software, and bandwidth can not be expected to be high with the robot in motion, using Eduroam WiFi and a VPN service.

There is a GitHub repository that adds H.264 features to ROS2 Image transport, called `h264_image_transport` [65], demonstrating that it is possible to send H.264 encoded video streams through ROS topics. Using ROS2 Image transport and this library or something similar could be a potential solution alternative to ours, with the advantages of ROS topics. We believe we have found an adequate solution for the prototype and attempting to use this comes with the risk of spending too much time optimizing the video feature of our prototype.

The control center software we are developing is web-based, and the rover streams video to a specified IP address (which could be a multicast address) with RTP. HTML5 `<video>` does not use pure RTP as a stream, which is why we need an intermediate transforming the RTP stream to something HTML5 supports. Several technologies are commonly used for streaming to web clients. The most common options are HTTP Live Streaming (HLS), Dynamic Adaptive Streaming (DASH), Real-Time Messaging Protocol (RTMP) and Web Real-Time Communication (WebRTC).

HLS and DASH work similarly by storing chunks of the video stream on a web server and serving them to the web browser. While they are considered low-latency protocols in their contexts, the latency is around 5-20 seconds. RTMP, while it has widespread support, it is aging technology. It can provide lower latency than HLS and DASH, but it is still in the 1-5 second range. WebRTC is modern technology, available on most platforms, and has less than 500ms latency, making it a good choice for the prototype.

WebRTC is free, open-source technology. The description from webrtc.org states that WebRTC can be used for real-time communication with support for video, voice, and generic data. The technology is available on all modern browsers and on native clients for all major platforms. In browsers, WebRTC can be used through a JavaScript API and in native clients through libraries. WebRTC is open-source

and supported by Apple, Google, Microsoft, Mozilla, and others [66].

ROS normally works in LAN (Local Area Networks) and video streaming and web apps require either a LAN connection or a public IP address. Eduroam and many other potential networks are more complex than a regular LAN, so we faced challenges with these services even within Eduroam. The solution for our prototype was to use a VPN server hosted in a cloud service with a public IP address.

7.1.9 Control system

The frontend is built with React. In order to interface ROS 2 with React, we have used the libraries `roslib` and `rosbridge`. These libraries are missing a lot of functionalities and is not feasible to use in the future. A better option would be to interface the frontend to a server using websockets and a REST-API or GraphQL, or by using a system-of-systems synthesizer (System-of-systems synthesizer) like the solutions in `open-rmf`. `Open-rmf`'s web `rmf-web` package can potentially be a good solution for a frontend or a starting point for it.

The backend we have called Task Server is built with support for multiple robots of different types in mind, but is so far only tested with one robot. It is quite similar to Robot Middleware Framework, `open-rmf`.

In the preliminary project we performed, we considered using the `open-rmf` toolbox but decided against it as the tools we considered using either were listed as experimental or outright were not made at the time. This applied to `rmf-web` and path-planning respectively. The toolbox also does not operate in unknown environments, which means one has to manually model the environment the fleet is to operate in, which can be very time-consuming. The combination of these factors led to us choosing not to use the `open-rmf` toolbox. But over the course of our project, the project has seen great developments, as `rmf-web` has changed status from experimental to stable, and a module for path-planning has been made but has the experimental status. This indicates that the work is underway and that if one is to continue with this project it should at least be something worth considering. By using software created by developers that has more experience in the domain, one could save time, and the time saved could possibly be used on modelling the robot's environment instead.

7.1.10 Maps

We spent some time using RVIZ, which displays the map generated by a robot. The maps generated are not human-friendly and contain no geospatial metadata. It is, however, possible to add such data to it by publishing poses of interest. This would be very difficult and time-consuming for developers, and it would be difficult to create this map data. We experimented with solutions for integrating the same type of map in web applications. We did not find any available mature and maintained open-source solutions.

7.1.11 Expanding the fleet

In this project, the control system was only tested with one rover-like robot. The interviews and user tests have revealed that in order for this system to be useful, it needs more robots, positioned in different places. The task server has been built to handle several robots, so the next step would be to build more robots and add them to the system. Not only would more robots increase the usefulness, but different types of robots would be useful as well. If a land-based robot that performs inspections is considered useful, it is very likely that an air-based robot like a drone would be useful as well as it could fly to places that are less accessible for janitors. This was suggested by the janitors several times both during the interviews and during the usability tests. The transport feature was scoffed at by several janitors, as Alfred is not capable of carrying a very large payload. But all the testers saw potential in this feature if the robot had been capable of carrying a larger load.

7.2 Research Method

Over the course of the project, we have tried to perform the research from a neutral standpoint and tried to use the most suitable methods. It is however close to impossible to remain completely neutral and in hindsight, methods may appear to not be as suitable as initially thought. In this section, we will discuss the methods and our process for performing the research.

7.2.1 Ethics

The project was conducted in accordance with the rules given by NSD. All subjects were made aware of their rights. Before coming for the interview, all subjects were given a copy of the consent agreement so they could read it carefully in their own time. When the subjects came to the interviews, their rights were repeated to them, after they had signed the consent agreement.

When we performed the observation, all the janitors were told during the morning meeting that we would be observing them and that we would write down anything we found useful for our project.

As the usability tests were conducted some time after the interviews, we chose to repeat the users' rights to them before we started the user tests.

We also recorded the audio of the interviews and the usability tests and all subjects were made aware of this before we started recording. The audio of the recordings was uploaded to the project Sharepoint, where they were eventually transcribed and deleted afterward.

All subjects were made aware of their right to see the data stored about them, edit this data, and delete this data. However, none of the subjects exercised this right.

7.2.2 Design process

When deciding on the general design for the web app, we made assumptions regarding the users' technological competence based on the fact that they daily used apps such as Lydia and Teams. This assumption proved to be faulty in the case of one subject and indicates that we had not made enough of an effort when we tried to get to know our users and their needs. It could therefore be beneficial to perform user interviews with a focus on design and usability with a larger amount of people from the user group, and with a much more varied group of people both gender-wise and age-wise.

7.2.3 Observation

Oates suggests an evaluation guide for evaluating participant observation (see Appendix G), used here to discuss the observation conducted with a team of janitors. They quickly adjusted to having two people observing and sometimes giving them a helping hand and seemed eager to show us the pride they take in their work.

Janitors' work is varied, which the observation confirmed. It also gave us a sense of how much they walk, which is a lot. It was difficult to keep up with them. It was also interesting to see some of the areas they work in that are not normally used by the public or by users. Because of the varied nature of their work, a lot more time would be needed to get the full picture. Observations were, however, a supplement to other research methods, which is why we consider half a work-day with one team enough. We do not think we disrupted their work much.

We both partook in the observation and the documented observations are a combination of notes both of us made. This was our attempt at triangulating the data, in order to keep the data as neutral as possible. However, it is a non-negligible fact that both of us come from a very similar academic background and the observations made by us is not necessarily the same as people of different academic and professional backgrounds than us would make.

7.2.4 Participants

The group we interviewed was a very homogeneous group of people. Out of the 12 people we interviewed, only one person identified as female, while the rest identified as male. Of the 12 people we interviewed, only one person was outside of the age group 53-62 years old. This could be problematic as the participants may not be as representative of their group as is desirable. One of the subjects told us that the janitor profession mainly consisted of men in the age group aforementioned, but that he believed this was likely to change. He had seen a larger number of female applicants for jobs in his department. Several of the janitors told us about how the profession was becoming more technical and that it is a new requirement to perform an apprenticeship before becoming a janitor. Several subjects suspected that this formal requirement could make the profession more

attractive, instead of being a fallback profession for those who had worked in other professions such as painting or carpentering. This was the case for several of the janitors we interviewed.

7.2.5 Interviews

Two groups of people were interviewed in a group setting. According to [7, p. 195] this was fine, as long as all participants are of similar status in the workplace. This was the case in both group interviews, as each group consisted of a team leader and two people from their teams. But it is worth noting that being interviewed in a group setting does not benefit all personality types. Some people have a more withdrawn and silent personality type, which can make it hard for them to make their voices heard in a group setting. This was the case in one of the group interviews, as one of the participants rarely said anything. We noticed this during the interviews and tried to direct questions to this person. This would either lead to the person giving lengthy answers or someone else would answer the question before the person had a chance to answer. Therefore, an individual interview with this person would likely bring much more value to us. Group interviews can be prone to the groupthink phenomenon, which we got to see firsthand. There were several times that the group would discuss amongst themselves what their answer to a question would be before one person provided the answer the group had agreed on. It is therefore not unreasonable to assume that someone agreed on having an opinion they only partly agreed with or opposed, but agreed on to keep the cohesion in the group intact [67]. A suggestion if anyone wanted to perform a similar type of interview would therefore be to strictly conduct individual interviews, as this would prevent the two aforementioned things entirely.

When we performed the semi-structured interviews to find tasks for the rovers, we followed an interview guide that went through multiple iterations, where we had a focus and keeping the questions strictly relevant and keeping the wording of the questions free of any leading language. This was done in order to ensure that the answers given would not be colored by what the interviewees thought we as interviewers wanted to hear. Although the interview guide went through several iterations, we still conducted semi-structured interviews. Any follow-up questions were asked off the cuff, which means they were not guaranteed to be free of any leading language, but this was still something we tried to keep in mind when asking.

7.2.6 Transcription

None of the interviews were transcribed ad-verbatim, instead we transformed statements into more concise versions of the statements taking great care to keep the meaning of the statements intact. Even though we tried to interpret the sentences from a neutral standpoint, it is a non-negligible fact that this could have introduced bias into the data. Confirmation bias is the tendency to interpret results

in a way that is favorable for oneself. By interpreting statements, it is easy to see how confirmation bias may have affected the data. An example is when we performed interviews to find tasks for the robot, we could have inflated how valuable a task would be for the interviewee when we performed the data analysis.

Similarly to observations, the interpretations we did were not necessarily the same a person of a different background would make.

7.2.7 Moderation of the usability test

In Appendix H you can read the script the moderator used in the usability tests. Usability tests are vulnerable to a number of biases. The Hawthorne effect is when a subject acts in a different effect than they naturally would, because they are aware that they are being observed. In order to try to minimize the probability of this effect happening, we positioned the observer outside of the eyesight of the subject. We also used an iPad for taking notes, as it would make less noise when anything was written down. We also tried to create a relaxed atmosphere by offering coffee and sweets.

We tried to establish an open atmosphere and stressed that we would be equally happy with negative feedback as with positive. We also stressed that the usability test was a test of our prototype, and not the subject. This was done in the attempt of getting as much feedback possible, even negative feedback.

The tasks were quite open-ended, and the result was not necessarily obvious to the test subjects as this was the first time they tried the system. Therefore, we tried to have a moderator that was quite active. What this meant was that the moderator would observe the user as they performed tasks, and if they were stuck the moderator asked if they wanted to continue to the next task. This was done in order to prevent the test subjects from feeling frustrated, which could have affected the rest of the experience with using the system. The moderator tried to give each subject what he felt was an appropriate amount of time, but it is highly unlikely that everyone has the same opinion of what an appropriate amount of time is. This could therefore have led to the test subjects feeling stressed or rushed.

Even though the moderator tried to keep an active role, he only did so when the situation called for it. Most of the time he tried to be invisible and non-helpful. By helping the subjects with performing tasks, the tasks would not have been completed by the subject, but by the subject and the moderator. The moderator therefore tried to keep as silent as possible, trying to refrain from giving away how the subject was doing, and only spoke when necessary.

7.2.8 Interview after usability test

In the interviews directly after the usability tests, the users were presented with a couple of statements they were to rate how much they agreed with or not. These statements were:

- By using robots to continuously patrol and notifying of obstacles in front fire exits, the fire safety could be increased
- It could be useful to have a status report available each morning that contained issues, obstacles, water leaks and lacking air quality
- I know, or know about someone that would transport packages using this or similar robots
- It could be useful to send a nearby robot to a reported issue, in order to inspect the issue so I can get a better overview of the problem
- I believe this system would be useful in my work day

The wording in these statements is quite opinionated, this was by design as we wanted to make it clear to the interviewees what the sentiment of the statement was. In hindsight, it is obvious that all statements were of the same opinion, namely that this system is useful. This is unlucky as it could have an effect on the answers. The subjects could experience that the statements represent our opinion, and by wanting to keep us pleased they state that they agree with a statement more than they really do or that they do not. This could perhaps have been prevented by having the statements have an alternating opinion. By this, we mean that for example having the odd-numbered statements being turned to their negation. This could have made it harder to gauge our opinion, and could have lowered the possibility of the subject being influenced. It should be noted that this did not seem to be the case, as the scores seemed to reflect the statements the subjects had made during the test and in the conversation afterward.

As mentioned in 6.6.1, some of the wording in the tasks was perceived by our pilot subject as leading. This led to us fixing the relevant tasks and revising the questions, in order to ensure that there was as little leading language as possible.

7.2.9 The Setting of the Usability Test

The location of the tests was performed in the hallway of one of the basement floors in the IT-building, Southwing at Gløshaugen. This location was chosen because of its many doors which the robot could inspect during the tests. However, there was a caveat to choosing a hallway: people walking by. We chose one of the basement floors, in order to minimize the probability of people walking by and being a distraction during the tests, but there were still several people that walked by.

All usability tests were performed during the working hours of the subjects, and several of the subjects got phone calls during the tests. One subject even had to pause the test to fix some work-related issues before resuming the test.

Both previously mentioned issues are highly distracting and could have a negative impact on the users' perceived usability of the test. Future usability tests could therefore benefit from being conducted outside working hours in a place guaranteed to have no people walking by.

7.3 Technology acceptance

According to TAM, if the users perceive the system as useful and easy to use, they are more likely to use it. If TAM predicts high usage of the system, we can argue that it could be successfully created and implemented in the respective user segment. Many software development projects have resulted in expensive failure, and TAM can be a useful model to predict success or failure and help with making decisions regarding design changes, how and whether the project should continue or not. As previously mentioned, Davis and Venkatesh [42] suggests using TAM on pre-prototype testing. In their research, they state that perceived ease of use requires hands-on experience to be evaluated. In the conducted usability tests, the users have gotten hands-on experience with a realistic prototype. The results are good perceived usefulness and ease of use. Additionally, the test subjects states that they would use the system in their work (Behavioral intention). Which also points to high actual use.

More importantly than decisions regarding whether one should create a robot system like this for janitors or other professionals in property management, this research creates knowledge regarding how such systems should be designed and what is required to construct them for a successful introduction and actual use. Some important requirements are:

1. Robots must be able to have access through doors (Open them or otherwise enter through doorways)
2. Robots must operate on several different floor levels. Preferably they should be able to move between them
3. The system should use human friendly maps in interactions with users
4. The system must understand and incorporate geospatial metadata such as room numbers and names
5. The system must have low maintenance requirements
6. The system should not be another operation management system. Any notifications or issues should be integrated with existing systems

7.3.1 Business value

TAM does not model business value. Further research regarding business value should be conducted. Other than perceived usefulness and behavioral intention, the potential improvement in performance has not been modeled.

If universities, schools, or other enterprises were to invest in a system like this they would need to either improve safety or cut costs. If it can reduce driving, walking, alert about fire hazards faster than they are caught now, improve energy efficiency and alert about water leaks, it improves safety and reduces costs, depending on the cost of the system.

There are many relevant Key Performance Indicators (KPI) to track performance

in property management. Added business value can be defined as improving one or more KPIs. If the robot system can improve one or more KPIs, it can be said to have a business value. The following KPIs were found to be relevant to the robot system.

1. Inspection completion rate
2. Work order completion rate
3. Average time to respond to maintenance requests
4. Incidents resulting in financial loss
5. Average maintenance cost per property

The first three KPIs listed pertains to tenant or user satisfaction which in turn can affect revenue long term. It is easiest to see the connection here with commercial property management, but organisations and universities such as NTNU can also rely on satisfied users. NTNU needs good employees and students to function and to receive funding. The latter two KPIs are obviously important to any property management for cutting costs.

The robot system can improve inspection completion rate, the average time to respond to maintenance requests, and work order completion rates in several ways. Inspection rates can be directly and indirectly improved by using robots to automatically perform routine inspections, and by freeing time for janitorial and other professional capacities. Average time to respond to maintenance requests can be reduced both by the freed-up time and by having robots respond first, gathering information for employees to be able to perform their tasks more accurately and quickly. This in turn improves work order completion rates. Water leak detection is in the idea stage for the robots, which could reduce the risk of costly damages resulting in financial loss. With thermal cameras, the risk of fire could be reduced, and energy efficiency could be improved. The three first KPIs can result in a lower maintenance cost per property. An example of a valuable situation is one that one of the janitors mentioned during usability testing. He states that he is responsible for multiple buildings, and a robot could be used to inspect issues before he has to travel for 15 minutes and use a car to get there, only to learn that he did not bring what he needs to correct the issue.

7.4 Results of Usability tests

The main type of functionality the janitors would use is observation. The robots will observe, inspect, log data, and report back to them. This can reduce the time and effort spent on walking and even driving, as some janitors are responsible for multiple buildings. Property management might be more interested in improved energy efficiency by having more data, or improved safety.

7.4.1 Overview page

An issue with the overview itself was that unopened issues were listed twice. Both under active and unopened issues. In hindsight, they could have been on the same list with a different form of indication of their status. The users also expected to be able to use the map for an overview, and that they would see any activity currently going on in a map.

7.4.2 Report page

Even locations without issues are included in the report, and no graphics or icons are used to indicate which locations contained issues and which did not, except for the rectangle around obstructions. Because of this, the users had some trouble understanding which locations contained problems. In the future, we would want to include graphics to indicate which locations contained issues, in addition to the information further up in the report.

7.4.3 New patrol page

Planning new patrols went fine, but we believe the GUI would benefit from clearer step-by-step instructions and less text. Since the users initially had problems understanding the term "patrol" ("Patrulje" in Norwegian), the concept could potentially be redefined or explained with an interactive tutorial for new users, as one user suggested.

7.4.4 Transport page

In the future, it would also include an ETA for pickup and delivery. When users tried this function, they thought they had to get the robot to their location first, and then choose where to send it. That was their own expectation, as the GUI states that they are to only select the destination and then the robot will come to their location. In the future, we would let the users select where an item should be picked up and let them use their own location if they wish, and allow them selecting the destination at any time during the task. The view would additionally have a clearer step-by-step instructions.

7.4.5 Inspection page

In the future, a more automatic inspection could be possible, but it was not implemented in the prototype. For both the transport and the inspection tasks, several users first tried to use the new patrol function to do these missions. It seemed like once they understood the patrol function, they thought that would be used for every task. We would solve this by providing a quick integrated tutorial in the interface for first-time users, and additional helpful information that always are available when they want it. Once they had discovered all the functionality of the

system, they found it quite easy to use and logical. It seemed like some users were familiar with the controller and forward scene setup from driving games, as two users tried to adjust the camera using the right stick on the controller.

Chapter 8

Conclusion and Further Work

In the Research questions section, you can read about our conclusions with regards to our research questions. A summary of technical solutions we consider contributions to knowledge can be found in the Technical solutions section. In the Further work-sub section, you can read our recommendations for future research.

8.1 Research questions

RQ1: What is needed to make an indoor autonomous robot system valuable?

We have discovered that in order to make an indoor autonomous robot system valuable, the system need to be able to perform tasks that are considered useful and needs to be designed in a usable way. We have discovered multiple usability requirements for a robotic system, and technical solutions to implement some of them.

We have discovered useful tasks that relatively cheap robots can perform. The question still remains if this can provide value to the business with regards to cost and impact. We briefly discussed the business value of the system with regards to KPIs in property management, and it seems like this system if further developed, can have a minor positive impact if introduced to teams of janitors with responsibility for one or more large buildings. Modelling the business value further is beyond the scope of this thesis.

Our conclusion is based on data gathered from a few people in building operations and is unlikely to apply to the whole field of property management. Therefore we cannot conclude whether a future version of the prototype system will be valuable to them. However, the knowledge gained regarding useful tasks, technical & usability requirements, and technical solutions is likely to apply in any large building.

RQ1.1: How can indoor rovers assist professionals in building operations?

Rovers like Alfred can assist janitors with inspections, routine patrols, temperature & air quality statistics, transportation & issue detection with sensors like cameras, thermal cameras, and gas sensors & water leak detectors. We suggest including other departments of property management and operations in future research.

RQ1.2: How can a robot-fleet control system be designed to be considered useful to professionals in building operations?

The positive response from usability testing confirms the design choices and process in the project. We created a user and task-oriented design rather than a robot and technology-oriented one.

Some important requirements to be useful and usable are:

1. Robots must be able to have access through doors (Open them or otherwise enter through doorways)
2. Robots must operate on several different floor levels. Preferably they should be able to move between them
3. The system should use human friendly maps in interactions with users
4. The system must understand and incorporate geospatial metadata such as room numbers and names
5. The system must have low maintenance requirements
6. The system should not be another operation management system. Any notifications or issues should be integrated with existing systems

8.2 Technical solutions

8.2.1 Maps

The maps used by the system must be designed for people. This is the biggest difference we have found that separates our prototype from other projects. To make usable maps, we integrated an existing indoor & outdoor map solution with the maps created by the robot's SLAM algorithm. By selecting a pose for the map origins, which could be the robots' docks in the future, we transformed the cartesian 2D maps of the robots into the geographic coordinate system. With the use of Mazemaps and its developer API, we could then display robot positions in a human-friendly map, and transform selected positions in Mazemaps to coordinates in the robots' maps.

8.2.2 Frontend

We have created a frontend in React integrated with ROS 2. It communicates with ROS using roslib and rosbriidge. These libraries are not complete and not a

good option for future use. Better options include interfacing the frontend to a server using websockets and REST-api or graphql, or using a system-of-systems synthesizer (SOSS) like the solutions in open-rmf.

8.3 Further work

8.3.1 Method

Seeing how the workforce for the janitor profession is likely to change in the future, performing a similar process as we have with a more varied group would therefore be beneficiary. Talking with many more people would also be necessary. The profession will likely receive more female, and younger workers with a formal education within the field. We would also advise interviewing other departments of property management. Security departments are especially interesting in that regard.

8.3.2 Alfred

Necessary improvements to Alfred and the control system:

- Add 3D perception of environment
- Improved control of driving and steering
- Access to doors and elevators
- Multiple maps

The users were aware that the prototype has limitations, but they tested one with as much perceived realism as possible. To them, the prototype probably seemed close to production-ready with regards to the features they tested.

So far the hardware used was relatively cheap. To make a production-ready version it needs 3D perception and better control of driving and steering. These things can increase costs, but not necessarily by too much. This was discussed in sections 5.3.2.1 and 7.1.1. We suggest looking into RGB-D cameras like Intel RealSense, ESC's for motor control and a diff drive setup instead of skid steer.

The robots must traverse doors and elevators (See section 7.1.1). This is the greatest limitation of the prototype today. Several projects discussed in chapter 4 have solved this. We suggest researching the use of network communication to interface with them like several AGVs are doing. Operating on multiple floors also means the robots must switch maps between floors. [54] describes one way to do this.

8.3.3 Expanding the fleet

Alfred has limited potential for transport with regard to size and weight. There can be potential in the transport feature with a higher capacity robot. Drones can

be useful by performing inspections high up, to save time, effort, and money on lifts.

8.3.4 Control System

The user tests revealed several things that could be improved in the future. Below are specific usability issues and suggested improvements. This provides a good foundation for future iterative work that involves users.

8.3.4.1 The system

The control system we have started developing has similarities to other robot middleware frameworks (RMF). Instead of continuing the development of the control system in this project, future researchers should consider using open-rmf. It might also be possible to use open-rmf with a Mazemaps integration.

8.3.4.2 Help & Tutorial

Several users were confused by the terms used in the system, and when they learned what the different things meant they were usually able to perform tasks quickly. Creating a tutorial for each page and a glossary for the terms used in the system would therefore be recommended. Having easy access to this tutorial at all times would also be smart, as it is highly unlikely that users read more than they feel they have to. This was something that was requested by one of the testers.

Using icons more frequently would be highly beneficiary. During the usability tests, we discovered that most of the testers rarely read the help text and instead just tried clicking around. By using icons that represent the different functions, one could perhaps help with this. This was something that was requested by one of the testers and also something that was suggested by the user in the pilot study.

8.3.4.3 Overview page

Communicate more clearly what "active" and "unopened" means in the system, as people got confused when the same issue was listed twice, due to the issue having both statuses. "active"/"unopened" status should probably be indicated by icons, color, and text, rather than two separate lists. Some users also expected to see a map showing all robots on the overview page, this is also something that could be implemented.

8.3.4.4 Report page

It should be communicated more clearly which issue, image, and location are connected, and which locations and images contain issues. This can be done by adding icons and colors that indicate issues or no issues. Image, location, and

issue affiliation can be communicated better by graphics, short explanatory text, expandable information, and links.

8.3.4.5 Transport page

It is possible to extract an ETA or established time of arrival from the mission feedback, and showing this would be beneficiary for all mission types, but especially the transport missions as the user have to wait for the robot to come them before the package is sent.

It would also be smart to allow the users themselves to select where the pickup point of a package should be.

Last but not least, users should be able to select the delivery point during any part of the process, and not just before starting the task.

Bibliography

- [1] 'Ffmpeg.' (2022), [Online]. Available: <https://ffmpeg.org/> (visited on 09/06/2022).
- [2] 'React.' (2022), [Online]. Available: <https://reactjs.org/> (visited on 09/06/2022).
- [3] O. Robotics. 'Soss.' (2022), [Online]. Available: <https://osrf.github.io/ros2multirobotbook/soss.html>.
- [4] 'Typescript.' (2022), [Online]. Available: <https://www.typescriptlang.org/> (visited on 09/06/2022).
- [5] M. H. Schimek, B. Dirks, H. Verkuil and M. Rubli, 'Video for linux two api specification,' *History*, vol. 6, p. 11, 1999.
- [6] 'Webpack - concepts.' (2022), [Online]. Available: <https://webpack.js.org/concepts/> (visited on 09/06/2022).
- [7] B. J. Oates, *Researching Information Systems and Computing*. SAGE Publications, 2006.
- [8] J. F. Nunamaker Jr, M. Chen and T. D. Purdin, 'Systems development in information systems research,' *Journal of management information systems*, vol. 7, no. 3, pp. 89–106, 1990.
- [9] 'Robotics — vocabulary,' International Organization for Standardization, Geneva, CH, Standard, Nov. 2021.
- [10] J.-D. Warren, J. Adams and H. Molle, 'Arduino for robotics,' in *Arduino robotics*, Springer, 2011, pp. 51–82.
- [11] E. Renard. 'Arduino protothreads [tutorial].' (2022), [Online]. Available: <https://roboticsbackend.com/arduino-protothreads-tutorial/> (visited on 11/06/2022).
- [12] Arduino. 'Attachinterrupt().' (2022), [Online]. Available: <https://www.arduino.cc/reference/en/language/functions/external-interrupts/attachinterrupt/> (visited on 11/06/2022).
- [13] R. A. Paz, 'The design of the pid controller,' *Klipsch school of Electrical and Computer engineering*, vol. 8, pp. 1–23, 2001.

- [14] 'Ifra international federation of robotics,' International Federation of Robotics. (2022), [Online]. Available: <https://ifr.org/> (visited on 08/06/2022).
- [15] Synopsys. 'The 6 levels of vehicle autonomy explained.' (2022), [Online]. Available: <https://www.synopsys.com/automotive/autonomous-driving-levels.html>.
- [16] 'Raspberry pi - about us.' (2022), [Online]. Available: <https://www.raspberrypi.org/about/> (visited on 10/06/2022).
- [17] X. Xu, M. Luo, Z. Tan, M. Zhang and H. Yang, 'Plane segmentation and fitting method of point clouds based on improved density clustering algorithm for laser radar,' *Infrared Physics & Technology*, vol. 96, pp. 133–140, 2019, ISSN: 1350-4495. DOI: <https://doi.org/10.1016/j.infrared.2018.11.019>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S135044951830639X>.
- [18] 'Rplidar a1.' (2022), [Online]. Available: <https://www.slamtec.com/en/Lidar/A1> (visited on 10/06/2022).
- [19] M. R. Endsley, B. Bolté and D. G. Jones, *Designing for situation awareness: An approach to user-centered design*. CRC press, 2003.
- [20] M. R. Endsley, 'Situation awareness and human error: Designing to support human performance,' in *Proceedings of the high consequence systems surety conference*, Albuquerque, NM, 1999, pp. 2–9.
- [21] ROS. 'Why ros?' (2022), [Online]. Available: <https://www.ros.org/blog/why-ros/>.
- [22] ROS. 'Concepts.' (2022), [Online]. Available: <http://wiki.ros.org/ROS/Concepts>.
- [23] ROS. 'About ros 2 interfaces.' (2022), [Online]. Available: <https://docs.ros.org/en/foxy/Concepts/About-R0S-Interfaces.html>.
- [24] 'Understanding ros 2 nodes.' (2022), [Online]. Available: <https://docs.ros.org/en/galactic/Tutorials/Understanding-R0S2-Nodes.html> (visited on 13/06/2022).
- [25] 'Understanding ros 2 actions.' (2022), [Online]. Available: <https://docs.ros.org/en/galactic/Tutorials/Understanding-R0S2-Actions.html> (visited on 13/06/2022).
- [26] 'Dds security.' (2018), [Online]. Available: <https://www.omg.org/spec/DDS-SECURITY/1.1>.
- [27] J.-H. Fylling and K. M. Halvarsson, 'Project report,' 2021.
- [28] 'Overview.' (2022), [Online]. Available: <https://navigation.ros.org/> (visited on 11/06/2022).
- [29] 'Detailed behavior tree walkthrough.' (2021), [Online]. Available: https://navigation.ros.org/behavior_trees/overview/detailed_behavior_tree_walkthrough.html (visited on 06/12/2021).

- [30] T. V. Haavardsholm. 'A handbook in visual slam.' (2021), [Online]. Available: https://github.com/tussedrotten/vslam-handbook/releases/download/v0.1.0/vslam-handbook_v0.1.0_2021-09-12.pdf.
- [31] S. G. Tzafestas, *Introduction to Mobile Robot Control*, 1st ed. Elsevier, 2014, ISBN: 978-0-12-417049-0.
- [32] I. D. Foundation. 'What is human-centered design.' (2022), [Online]. Available: <https://www.interaction-design.org/literature/topics/human-centered-design>.
- [33] A. Steinfield, 'Interface lessons for fully and semi-autonomous mobile robots,' *Proceedings of the 2004 IEEE International Conference on Robotics & Automation*, 2004, ISSN: 1050-4729. DOI: <https://doi.org/10.1109/ROBOT.2004.1307477>.
- [34] G. Randelli, M. Venanzi and D. Nardi, 'Evaluating tangible paradigms for ground robot teleoperation,' in *2011 RO-MAN*, 2011, pp. 389–394. DOI: [10.1109/ROMAN.2011.6005240](https://doi.org/10.1109/ROMAN.2011.6005240).
- [35] N. Velasco, D. Mendoza and A. Barrientos, 'User interfaces applied to teleoperate mobile robots with keyboard command, ps3 controller and mobile phone,' May 2015. DOI: [10.15849/icit.2015.0108](https://doi.org/10.15849/icit.2015.0108).
- [36] G. Adamides, C. Katsanos, Y. Parmet, G. Christou, M. Xenos, T. Hadzilacos and Y. Edan, 'Hri usability evaluation of interaction modes for a teleoperated agricultural robotic sprayer,' *Applied Ergonomics*, vol. 62, pp. 237–246, 2017, ISSN: 0003-6870. DOI: <https://doi.org/10.1016/j.apergo.2017.03.008>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0003687017300674>.
- [37] K. de Carvalho, D. Villa, M. Sarcinelli-Filho and A. Brandão, 'Gestures-teleoperation of a heterogeneous multi-robot system,' *The International Journal of Advanced Manufacturing Technology*, vol. 118, pp. 1–17, Jan. 2022. DOI: [10.1007/s00170-021-07659-2](https://doi.org/10.1007/s00170-021-07659-2).
- [38] Y. Lu, L. Liu, S. Chen and Q. Huang, 'Voice based control for humanoid teleoperation,' in *2010 International Conference on Intelligent System Design and Engineering Application*, vol. 2, 2010, pp. 814–818. DOI: [10.1109/ISDEA.2010.430](https://doi.org/10.1109/ISDEA.2010.430).
- [39] M. Turner, B. Kitchenham, P. Brereton, S. Charters and D. Budgen, 'Does the technology acceptance model predict actual use? a systematic literature review,' *Information and Software Technology*, vol. 52, no. 5, pp. 463–479, 2010, TAIC-PART 2008, ISSN: 0950-5849. DOI: <https://doi.org/10.1016/j.infsof.2009.11.005>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0950584909002055>.

- [40] J. Keung, R. Jeffery and B. Kitchenham, 'The challenge of introducing a new software cost estimation technology into a small software organisation,' in *2004 Australian Software Engineering Conference. Proceedings.*, 2004, pp. 52–59. DOI: 10.1109/ASWEC.2004.1290457.
- [41] F. D. Davis, 'User acceptance of information technology: System characteristics, user perceptions and behavioral impacts,' *International Journal of Man-Machine Studies*, vol. 38, no. 3, pp. 475–487, 1993, ISSN: 0020-7373. DOI: <https://doi.org/10.1006/imms.1993.1022>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0020737383710229>.
- [42] F. Davis and V. Venkatesh, 'Toward preprototype user acceptance testing of new information systems: Implications for software project management,' *IEEE Transactions on Engineering Management*, vol. 51, no. 1, pp. 31–46, 2004. DOI: 10.1109/TEM.2003.822468.
- [43] G. Fjermedal. 'Bruk av technology acceptance model 3 for å optimalisere innføring av it-systemer i små og mellomstore bedrifter.' (2018), [Online]. Available: https://ntnuopen.ntnu.no/ntnu-xmlui/bitstream/handle/11250/2561042/18243_FULLTEXT.pdf?sequence=1%5C&isAllowed=y%5C&fbclid=IwAR13_ANVh3KwzNUKjDdNQTeEcpzVTZ-C-c_rILsJme4s5u0rhHQrLGRZjbE (visited on 27/05/2022).
- [44] V. Venkatesh and H. Bala, 'Technology acceptance model 3 and a research agenda on interventions,' *Decision Sciences*, vol. 39, no. 2, pp. 273–315, 2008. DOI: <https://doi.org/10.1111/j.1540-5915.2008.00192.x>. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1540-5915.2008.00192.x>. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-5915.2008.00192.x>.
- [45] V. Venkatesh, 'Determinants of perceived ease of use: Integrating control, intrinsic motivation, and emotion into the technology acceptance model,' *Inf. Syst. Res.*, vol. 11, pp. 342–365, 2000.
- [46] H. Loranger. 'Checklist for planning usability studies.' (2016), [Online]. Available: <https://www.nngroup.com/articles/usability-test-checklist/> (visited on 11/04/2022).
- [47] '«hei, kanj du pass dæ litt!?!»' E24. (2013).
- [48] 'Boston dynamics,' Boston Dynamics. (2022), [Online]. Available: <https://www.bostondynamics.com/>.
- [49] S. H. -. P +. T. Solutions. 'Swisslog automation systems at st olavs hospital,' Swisslog. (2010), [Online]. Available: <https://www.youtube.com/watch?v=rutHGN4IMB8>.

- [50] R. A. Søråa and M. E. Fostervold, 'Social domestication of service robots: The secret lives of automated guided vehicles (agvs) at a norwegian hospital,' *International Journal of Human-Computer Studies*, vol. 152, p. 102 627, 2021, ISSN: 1071-5819. DOI: <https://doi.org/10.1016/j.ijhcs.2021.102627>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1071581921000458>.
- [51] S. H. AG. 'Swisslog,' Swisslog Holding AG. (2022), [Online]. Available: <https://www.swisslog.com/>.
- [52] O. Menzel. 'A big hit everywhere: The room service of the future is robotic,' thyssenkrupp AG. (2019), [Online]. Available: <https://engineered.thyssenkrupp.com/en/a-big-hit-everywhere-the-room-service-of-the-future-is-robotic/>.
- [53] B. Krishnamurthy and J. Evans, 'Helpmate: A robotic courier for hospital use,' in *[Proceedings] 1992 IEEE International Conference on Systems, Man, and Cybernetics*, IEEE, 1992, pp. 1630–1634.
- [54] J. Bačík, F. Ďurovsk, M. Biroš, K. Kyslan, D. Perdukova and S. Padmanaban, 'Pathfinder–development of automated guided vehicle for hospital logistics,' *Ieee Access*, vol. 5, pp. 26 892–26 900, 2017.
- [55] 'Spot,' Boston Dynamics. (2022), [Online]. Available: https://www.bostondynamics.com/sites/default/files/styles/advanced_technology/public/2020-10/spot-core-new2x.png?itok=jyCEl2-I.
- [56] 'Stretch,' Boston Dynamics. (2022), [Online]. Available: https://www.bostondynamics.com/sites/default/files/2022-03/product-hero-6_0.jpg.
- [57] J. Fink, V. Bauwens, F. Kaplan and P. Dillenbourg, 'Living with a vacuum cleaning robot,' *International Journal of Social Robotics*, vol. 5, no. 3, pp. 389–408, 2013.
- [58] Mazemaps. 'Mazemaps.' (2022), [Online]. Available: <https://www.mazemap.com/our-maps/how-to-get-started> (visited on 01/06/2022).
- [59] N. 2. 'Amcl.' (2022), [Online]. Available: <https://navigation.ros.org/configuration/packages/configuring-amcl.html> (visited on 01/06/2022).
- [60] A. Sears-Collins. 'How to detect objects in video using mobilenet ssd in opencv.' (2021), [Online]. Available: <https://automaticaddison.com/how-to-detect-objects-in-video-using-mobilenet-ssd-in-opencv> (visited on 02/06/2022).
- [61] Y. Zhou, B. Li, D. Wang and J. Mu, '2d grid map for navigation based on lcsd-slam,' *11th International Conference on Information Science and Technology*, pp. 499–504, 2021. [Online]. Available: <https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9440650%5C&tag=1>.

- [62] S. Sumikura, M. Shibuya and K. Sakurada, 'Openslam: A versatile visual slam framework,' *Proceedings of the 27th ACM International Conference on Multimedia*, pp. 2292–2295, 2019. [Online]. Available: <https://arxiv.org/pdf/1910.01122.pdf>.
- [63] S. Macenski, D. Tsai and M. Feinberg, 'Spatio-temporal voxel layer: A view on robot perception for the dynamic world,' *International Journal of Advanced Robotic Systems*, vol. 17, no. 2, 2020. DOI: 10.1177/1729881420910530. [Online]. Available: <https://doi.org/10.1177/1729881420910530>.
- [64] S. M. (Robotics), 'Use of the spatio-temporal voxel layer,' in *ROSCon Madrid 2018*, Open Robotics, Sep. 2018. DOI: 10.36288/ROSCon2018-900842. [Online]. Available: <https://doi.org/10.36288/ROSCon2018-900842>.
- [65] C. McQueen, *H264_timage_transport*, https://github.com/clydemcqueen/h264_image_transport, 2021.
- [66] 'WebRTC.' (2022), [Online]. Available: <https://webrtc.org/>.
- [67] S. A. Wheelan, *Creating Effective Teams: A Guide for Members and Leaders*, 5th ed. SAGE, 2016, ISBN: 978-1483346120.

Appendix A

Pictures of control system web app

Overview

Oversikt Rapporter Behov Patruljer Ny patrulje Transport Inspeksjon

Oversikt

Aktive behov

Hindring oppdaget!
Følgende objekt(er) ble oppdaget av en robot på en patrulje: stolstol

Apne

Uåpnede behov

Hindring oppdaget!
Følgende objekt(er) ble oppdaget av en robot på en patrulje: stolstol

Apne

Oppdrag under utførelse

Dagens planlagte oppdrag

Reports

Oversikt Rapporter Behov Patruljer Ny patrulje Transport Inspeksjon

Rapporter

Fotopatrulje - Lørdag, 21.4
1 behov ble funnet

Apne

Report

Oversikt Rapporter Behov Patruljer Ny patrulje Transport Inspeksjon

←

Sammenstillingsrapport

Slutført: 5/21/2022, 5:34:00 PM

Opprettede behov:


Hindring oppdaget
Følgende objekt(er) ble oppdaget av en robot på en patrulje: stol, stol

[Abne](#)

Data samlet Inn ved lokasjoner

Lokasjon 1

Temperatur: 21°C
Luftfuktighet: 40%



Lokasjon 2

Temperatur: 21°C
Luftfuktighet: 40%



Besøkte lokasjoner



Issues

Oversikt Rapporter **Behov** Patruljer Ny patrulje Transport Inspeksjon

Behov

Hindring oppdaget!
Følgende objekt(er) ble oppdaget av en robot på en patrulje: stol, stol

Åpne

Issue


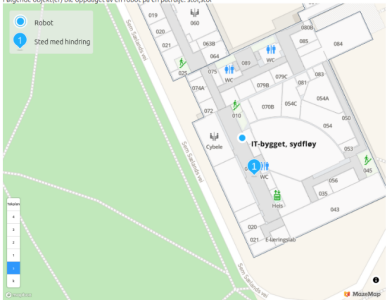
Oversikt Rapporter **Behov** Patruljer Ny patrulje Transport Inspeksjon

←

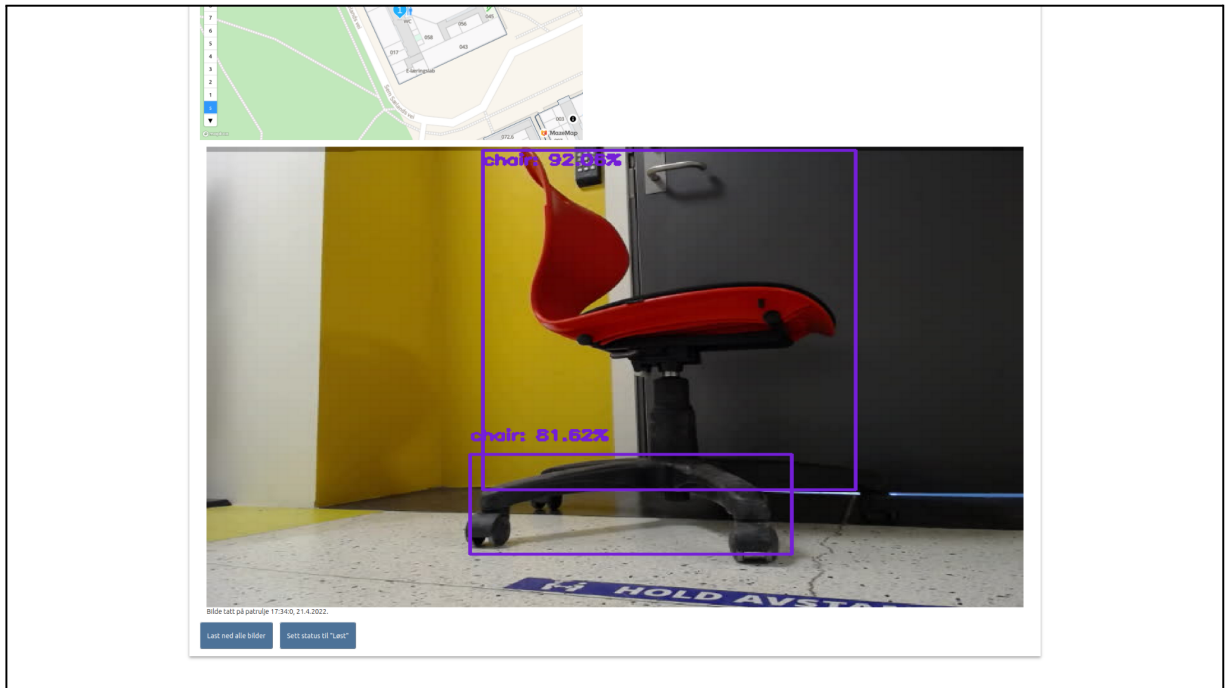
Behov

Hindring oppdaget!

Følgende objekt(er) ble oppdaget av en robot på en patrulje: stol, stol



chair 92856%



Patrols

Oversikt Rapporter Behov Patrulljer Ny patrulje Transport Inspeksjon

Patrulljer

Planlagte Patrulljer

Andre oppdrag

IT-byggjet, syvte etasje 5
Høgskoleveien. Undermåler alle dører.

Åpne

New Patrol

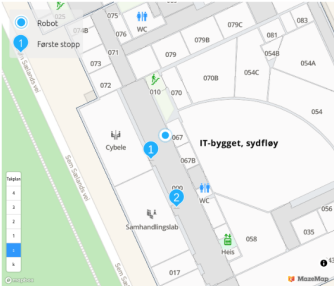
Oversikt Rapporter Behov Patruljer Ny patrulje Transport Inspeksjon

Ny patrulje

Tittel:

Beskrivelse:

Trykk på de stedene i kartet roboten skal kjøre innom i løpet av patruljen.
Roboten vil ta bilde på hvert av stedene.
Ønsker du å slette et punkt du har plassert, kan du trykke på det.



● Markørene viser i rekkefølge hvor roboten stopper for målinger

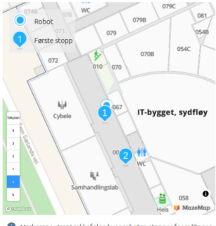
Start / Schedule Patrol

Oversikt Rapporter Behov Patruljer Ny patrulje Transport Inspeksjon

←

Patrulje: IT-bygget, sydflyet, etasje 5

Hjelpepatrulje. Underseker alle dører:



● Markørene viser i rekkefølge hvor roboten stopper for målinger

Gjenta oppdrag?

Patruljeplan

Dager aktiv:

so la fr to on ti on

Hvor ofte i døgnet? Fra kl:

En gang

Velg robot:

Manual control

Oversikt Rapporter Behov Patruljer Ny patrulje Transport Inspeksjon

Manuell kontroll av roboten

Om du trykker på "Start manuell kontroll", kan du styre roboten med kontrolleren. Bruk høyre hendel for å kjøre fremover, venstre for å kjøre bakover. Bruk venstre joystick for å svinge

Appendix B

Consent agreement

Do you wanna participate in the research project

Using autonomous rover-like robots to do indoor tasks?

This is us asking you to participate in a research project where the goal is to make the working life of the janitorial staff who oversee universities/schools easier by using robots to do tasks for them. In this document we will communicate the goals for this project and what your participation will entail.

Purpose

In the recent years we have seen an explosion of availability of cheap and easy to use electronics that interfaces hardware with software, where products like Raspberry PI and Arduino are especially known. This has opened the field of robotics to other branches of engineers, where the project group which consists of computer engineers is an example. During the fall semester of 2021 the project group has built a robot that is able to receive a command that tells it where to go, and navigate to the given goal on its own. It can also be fitted with various sensors such as cameras, temperature sensors, barometers and more, so it is really only the limits of your imagination that limits what kind of tasks the robot can perform.

The goal of the research project is therefore:

- Investigate what kind of tasks the janitorial staff who oversees large semi-public buildings has
- Find out which of the tasks that are fitting for the janitorial staff to delegate to the robot
- Design and create a command center for the janitorial staff, where they can delegate tasks to the robot and keep an eye on it while it is doing tasks

As of now moment the research questions are defined as:

- 1 How can indoor rovers assist the janitorial staff in day-to-day operations in semi-public buildings?
- 2 How can a robot-fleet command center be designed in order to be usable for janitorial staff/professionals?

Who is responsible for the research project?

This project is a part of a master thesis written at NTNU(Norwegian University of Science and Technology). The project is done by the master students Johan-Henrik Fylling and Kevin Mentzoni Halvarsson, who are supervised by Terje Røsand and George Adrian Stoica.

Why are you asked to participate?

The sample consists of janitorial staff who oversees relatively large buildings with areas that are out of bounds unless one has clearance(semi-public buildings). To ensure that the sample is representative for the population it represents we wish to interview as many people as feasible, and we therefore estimate that we will ask around 50 people who fulfills the previously mentioned criteria. Everyone that is contacted is contacted with contact information either procured from the university/school that employ them or contact information given to the project group by the supervisors.

What does your participation entail?

- If you agree to take part in the personal interview, you will participate in a semi-structured interview. This means that the interviewers will have a list of questions they wish to ask, but can deviate from if relevant. Your statements may lead to follow-up questions. The information

stored will consist of contact information, your age, and gender. This information will be anonymized in the published thesis, which means it will be impossible for you to be identified by reading the thesis alone. We estimate that the duration of the interview will be about 20-30 minutes. In addition to the stored information, we will record the audio of the interview which will be transcribed and stored electronically.

- If you agree to participate in the user test, you will be a part of a participatory observation. This means that you will be made aware of you being observed while you perform some tasks in our prototype. We will also ask you to think out loud when you are performing the tasks. During the user test notes will be taken by the observers, a screen recording of the device you are using to perform the tasks on, and an audio recording of the user test which will be transcribed and stored electronically. We estimate that the user test will be about 15-60 minutes.

Participation is voluntary

It is entirely voluntary to participate in the project. If you choose to participate, you can at any time withdraw your consent without stating why. Declining to participate or withdrawing your consent will not have any negative consequences for you.

Your privacy – how we store and use your information

We will only use your information for the purposes mentioned in this document. We will process your information confidentially and in compliance with the laws of privacy (personvernregelverket). The only people with access to your information will be the project group consisting of Johan-Henrik Fylling and Kevin Mentzoni Halvarsson. **[INSERT PART ABOUT ANONYMIZATION]** All data will be stored on the project groups OneDrive distributed by NTNU. All devices with access to the aforementioned OneDrive is code-/password protected.

When the master thesis is published, the transcribed interviews and user tests will also be published, but all participants will be anonymized to ensure that they cannot be identified. As previously mentioned participants age and gender will be stored, but will not be included in the published thesis.

What happens to your information after the finalization of the research project?

Your information will be deleted after the project is finished/the thesis is approved, which is estimated to be 13th of June.

Your rights

As long as you are identifiable in the data, you have the right to:

- gain access to the personal data stored about you, and receive a copy of the data
- correct your personal data
- delete your personal data
- send a complaint to Datatilsynet regarding the processing of your data

What gives us the right to process your personal information?

We process your personal information based on your given consent.

Tasked by NTNU, NSD – Norsk senter for forskningsdata AS has appraised that the processing of personal data in this project is in compliance with the laws of privacy (personvernregelverket).

How can I find out more?

If you have any questions regarding the project, or wish to use your aforementioned rights, you can contact the group:

- NTNU – Norwegian University of Science and Technology w/ Terje Røsand

- Epost: terjero@ntnu.no
- NTNU – Norwegian University of Science and Technology w/ George Adrian Stoica
 - Telefon: 73412088
 - Epost: stoica@ntnu.no
- NTNU – Norwegian University of Science and Technology w/ Johan-Henrik Fylling
 - Epost: johanhef@ntnu.no
- NTNU – Norwegian University of Science and Technology w/ Kevin Mentzoni Halvarsson
 - Epost: kevinmh@ntnu.no
- Safety representative at NTNU: Thomas Helgesen
 - Telefon: 93079038
 - Epost: thomas.helgesen@ntnu.no

If you have any questions regarding the appraisal of the project done by NSD, you can contact us at

- NSD – Norsk senter for forskningsdata AS via e-mail (personverntjenester@nsd.no) or via telephone: 55 58 21 17.

With kind regards

George Adrian Stoica
(Supervisor)

Terje Røsand
(Supervisor)

Johan-Henrik Fylling
(Student)

Kevin M. Halvarsson
(Student)

Declaration of consent

I have received and understood the information about *Using autonomous rover-like robots to do indoor tasks*, and has had the opportunity to ask questions. I agree to:

- participate in a personal interview
- participate in a user test

I give my consent to my personal data being processed until the project is finalized

(Signed by participant, date)

Appendix C

Interview Guide

Interview guide (EN)

For interviewer

- Try to start an open conversation with the interview subject
- Make sure not to feed ideas to the interview subjects so they tell you what they think you want to hear
- Be grateful and remember to tell them about the purpose of the interview and about anonymity
- Try to get the interview subject to summarize at the end of the interview and tell us what the most important things we talked about were, and if anything should be added
- Remember that the guide is meant as a guide to start conversation and reflection
- Make sure not to ask any questions previously asked or answered.

Understanding of profession

1. Can you tell me about your job and what your responsibilities and tasks are?
2. Could you describe a typical workday? (Could be answered already)
3. Are there any tasks that are repeated often?
4. Do you think there are any tasks that are less motivating or frustrating?

How robots can help

1. Is there anything that should be done more often, but isn't prioritized?
2. Are there any issues that are sometimes discovered a bit late?
3. If robots could do anything, what would be useful?

Questions about specific tasks

1. How are smoke detectors tested, and who does it?
2. Could you tell us about the routines for checking fire escape routes?
3. Do people ever report issues to janitors?
 - a. If so, could you walk us through how this works and how it is usually handled?

Suggestions for tasks for robots

1. Could it be useful if robots were looking for trash or other items in the hallways every morning and creates a report for you?
2. Could it be useful if robots checked fire escape routes regularly?
3. If a problem is reported to you, would it sometimes be useful to get more pictures or video of it, as well as an accurate location?

Summary

1. To summarize, what do you feel is the most important things we have talked about?
2. Is there anything you would like to add?
3. Is it fine with you if we contact you again if needed?
4. Thank you so much for participating and helping us!

Appendix D

SUS Form

1. Jeg tror jeg ville brukt dette systemet ofte

Sterk uenig	Uenig	Nøytral	Enig	Sterkt enig

2. Jeg synes systemet var unødvendig komplekst

Sterk uenig	Uenig	Nøytral	Enig	Sterkt enig

3. Jeg synes systemet var lett å bruke

Sterk uenig	Uenig	Nøytral	Enig	Sterkt enig

4. Jeg tror jeg hadde trengt hjelp av en mer teknisk anrettet person for å bruke systemet

Sterk uenig	Uenig	Nøytral	Enig	Sterkt enig

5. Jeg synes de forskjellige funksjonene i systemet var godt integrert

Sterk uenig	Uenig	Nøytral	Enig	Sterkt enig

6. Jeg synes det var vanskelig å se sammenhengen mellom de forskjellige funksjonene i systemet

Sterk uenig	Uenig	Nøytral	Enig	Sterkt enig

7. Jeg ser for meg at folk flest ville lært seg å bruke dette systemet ganske kjapt

Sterk uenig	Uenig	Nøytral	Enig	Sterkt enig

8. Jeg synes det var tungt å bruke dette systemet

Sterk uenig	Uenig	Nøytral	Enig	Sterkt enig

9. Jeg følte meg selvsikker da jeg brukte dette systemet

Sterk uenig	Uenig	Nøytral	Enig	Sterkt enig

10. Jeg trengte å lære meg mye før jeg kunne bruke dette systemet

Sterk uenig	Uenig	Nøytral	Enig	Sterkt enig

Appendix E

Project Report



DEPARTMENT OF COMPUTER SCIENCE

TDT4501 - COMPUTER SCIENCE, SPECIALIZATION PROJECT

Project report

Authors:

Johan-Henrik Fylling & Kevin Mentzoni Halvarsson

December, 2021

Table of Contents

List of Figures	ii
1 Introduction	1
2 Research Questions	1
2.1 Which tasks in a building are suitable for automation?	1
2.2 What is the best way to deploy an autonomous robotic fleet in a building?	2
2.3 How can an autonomous robotic fleet system best be developed for every part of the life-cycle?	2
2.4 How can users interact with an autonomous robotics fleet?	2
2.5 What is the most suitable way for autonomous robots to navigate inside a building?	2
2.6 Which are the most important tasks when managing a fleet of autonomous robots?	2
3 Related work	2
3.1 Mobile robots	2
3.2 Robotics Operating System (ROS)	3
3.3 Fleet and multi-fleet management	3
3.3.1 Robotics Middleware Framework	5
3.4 Indoor navigation	5
3.4.1 SLAM	9
4 What we have done	9
4.1 Hardware	9
4.2 Software	11
4.2.1 ROS	11
4.2.2 SLAM_toolbox	11
4.2.3 Navigation2	11
4.2.4 System architecture	12
4.3 Theory modules	12
4.3.1 TTK30 - Human-machine/autonomy interaction in cyber-physical systems	14
4.3.2 TTK21 - Introduction to Visual Simultaneous Localization and Mapping - VSLAM	14
5 Conclusion	14
Bibliography	15

List of Figures

1	ROS nodes, topics and services	4
2	ROS actions	4
3	Localisation of a robot in a room using UWB. Red dots are estimated robot positions.	6
4	Localisation of a robot in a room using SLAM. Small green dots are estimated robot positions.	7
5	Comparisons of UWB localization and SLAM algorithm	8
6	Summary of the most representative visual and visual-inertial systems	10
7	An example of a behaviour tree	12
8	Prototype system architecture	13

1 Introduction

One might say we have had several industrial revolutions, driven by technologies like steam power, electricity and computers. For every revolution, the time until the next one is shorter. Artificial Intelligence (AI) is among the technologies predicted to significantly impact the ongoing or coming industrial revolution. Not only has the field of AI advanced a great deal, it has also become easily available to any developer, scientist or hobbyist. In addition to AI becoming more available, electronics that interface software and hardware has become small, cheap and easy to use, with devices like Arduino and Raspberry Pi being well known.

The increased availability has led to opening up the field of robotics to a much broader spectrum of engineers, and we want to explore this ourselves from the perspective of computer scientists. We are particularly interested in the problem of managing and operating singular and multiple robots in an indoor environment, and how this technology can be used to improve people's lives.

2 Research Questions

In this report we explore existing technology for constructing autonomous robotics systems and test them out in practise to gain both qualitative and quantitative insights in such systems. This is preliminary work to our master thesis with the research question currently formulated as: "How to construct a useful autonomous fleet of robots with a web based management system for general purpose use inside buildings?".

The tentative research question of our master thesis is a rather large question, and can be split up into smaller research questions. By combining the solutions to the sub-research questions, we believe we should be able to find a solution for the main research question. The sub research questions can be found in the following subsections.

2.1 Which tasks in a building are suitable for automation?

Some potential tasks robots can do:

- Take photos of issues reported to the janitor/caretaker so he/she can come prepared to the location
- Do various sensor readings (indoor climate data, etc.) and therefore eliminate the necessity of putting up sensors everywhere
- Check if there are obstacles on the floor that do not belong there (could be limited to emergency exits)
- Add text recognition to see if somebody intends to pick up the obstacle
- Detect and track pest with cameras and check traps
- Test fire alarms
- Collect trash
- deliveries or mobile café

In order to find suitable use cases and requirements for a robotics fleet, we can involve property managers and maintenance personnel through interviews, workshops, or user tests.

2.2 What is the best way to deploy an autonomous robotic fleet in a building?

Many autonomous systems require installation of sensing equipment, mapping and annotation of the environment. The cost of deploying a robotics fleet could perhaps be reduced by simplifying the deployment process. Robots using SLAM could potentially be part of the solution because they map the environment automatically.

2.3 How can an autonomous robotic fleet system best be developed for every part of the life-cycle?

This is highly related to question 2.2, the deployment of a fleet, but focuses on how the system can best be developed with deployment, and all other life-cycle goals in mind.

2.4 How can users interact with an autonomous robotics fleet?

In order to be useful, and usable, many systems require a manual mapping and annotation process. One common system use case is then to click around on a map to get a task completed. Are there alternative ways of interacting with the fleet? Does this provide a good user experience? What if the environment is not manually mapped, but maps are only generated automatically by the robots?

2.5 What is the most suitable way for autonomous robots to navigate inside a building?

If a robot is to complete a task it needs to have a sense of its own localization at every given time. We as operators need to know this as well in order to know that every robot is operational and not stuck somewhere, and the robots need to know this so they can perform the tasks we want them to perform. Without localization the robots would basically be fumbling around in the dark. So in order to answer this research question, we need to find a way for each robot to keep track of its environment and localize itself in it.

2.6 Which are the most important tasks when managing a fleet of autonomous robots?

When managing several robots at once, there are several things to be wary of. We want to support a fleet of robots that could have varying sensors and capabilities. This requires modular, loosely coupled system that can support parts of it being exchanged with new software and hardware modules.

3 Related work

3.1 Mobile robots

A mobile robot is a robot that is capable of performing robot locomotion, meaning it is able to transport itself from place to place. In this project we will focus on ground robot locomotion. We usually split ground robot locomotion into two categories: legged motion and wheeled motion.

Legged locomotion is a type of locomotion inspired by how different animals with legs performs locomotion. Legged locomotion is further split into two categories: bipedal- and many-legged

locomotion. Legged locomotion is considered suitable when operating in complex terrain, but is due to the need for constantly achieving stability, lest the risk of the robot toppling over, less energy efficient compared to a wheeled robot.

Wheeled robots is robots that uses wheels to perform locomotion. There are a plethora of ways to perform wheeled locomotion, and the wheel type used usually depends on the the robots chosen drive type. A robots drive type describes how the wheels of the robot rotates in order to make the robot move. Common drive types are Ackermann-steering, differential drive steering and skid steer, which uses "normal wheels" i.e. wheels with or without grooves in them. Ackermann-steering is characterized by having two sets of wheels, where one set is fixed and the other can rotate around the z-axis of the point where the wheel is connected to the axle. An example of Ackermann-steering is how most cars steer. Differential drive is characterized by having two wheels that can apply different levels of torque independent of the other wheel. A well known example of differential drive steering is how a tank steers. Although a tank uses tracks, the principle for steering is the same. Mobile robots that uses differential drive often have a passive or a powered castor wheel to balance the robot. Skid-steering is closely related to differential drive steering, and the only difference is that there are at least 4 wheels on each side, where each wheel can operate independently of each other. These drive types are considered relatively easy to implement, as the steering mechanics are relatively intuitive.

A less common drive type which requires special types of wheels are omnidirectional-steering, which requires Mecanum wheels(a sort of wheel that has got rollers with rubber on them, and which is mounted at 45 degree angle compared to the direction that is considered forward for the robot) or ball-wheels. Due to the requirement of special wheels and the complex steering mechanics, this drive type is not as common the aforementioned ones.

3.2 Robotics Operating System (ROS)

Unlike what the name implies, this is not an operating system. Robotics Operating System (ROS) is an open source, modular, language independent framework, and development kit for all kinds of robots. Today there are two versions, ROS1 and ROS2, which can be interfaced for use in the same system. A system based on ROS consists of nodes with one or more responsibilities like controlling a lidar. Nodes communicate through TCP/IP with either UDP or TCP, so they can be distributed across multiple computers. Nodes can communicate in three standardised ways, through topics, services or actions. Any number of nodes can publish to or subscribe to the same topic. This can be used for continuous high rate communication and is therefore suitable, amongst other things, for sensor feedback and control signals. Services are much like traditional HTTP requests where a client makes a request and receives a response from the server. See figure 1.

In figure 2, actions are illustrated. They are very similar to services, but are more suitable for tasks that can take a long time, because the client can subscribe to feedback on a feedback topic where the server can publish progress information.

The core framework consisting of nodes, topics, services and actions are good enough on its own to be seriously considered in any robotics project, but another key advantage comes from the large number of packages available. For example, any software or robotics developer can implement a package for computer vision (CV) with very little knowledge about CV.

3.3 Fleet and multi-fleet management

A goal with this project is to manage multiple robots as a fleet through a web-interface. The management system should automatically dispatch the currently best suited robot to complete a given task. The best suited robot can be selected based on multiple parameters like energy level, sensors and equipment and current location. In addition to this assignment problem, multiple robots operating in the same environment also means they will share resources. Resources in this context can be floor space, elevators, doors, dispensers and more. Humans performing tasks in an environment with shared resources, will very easily communicate and compromise in order to

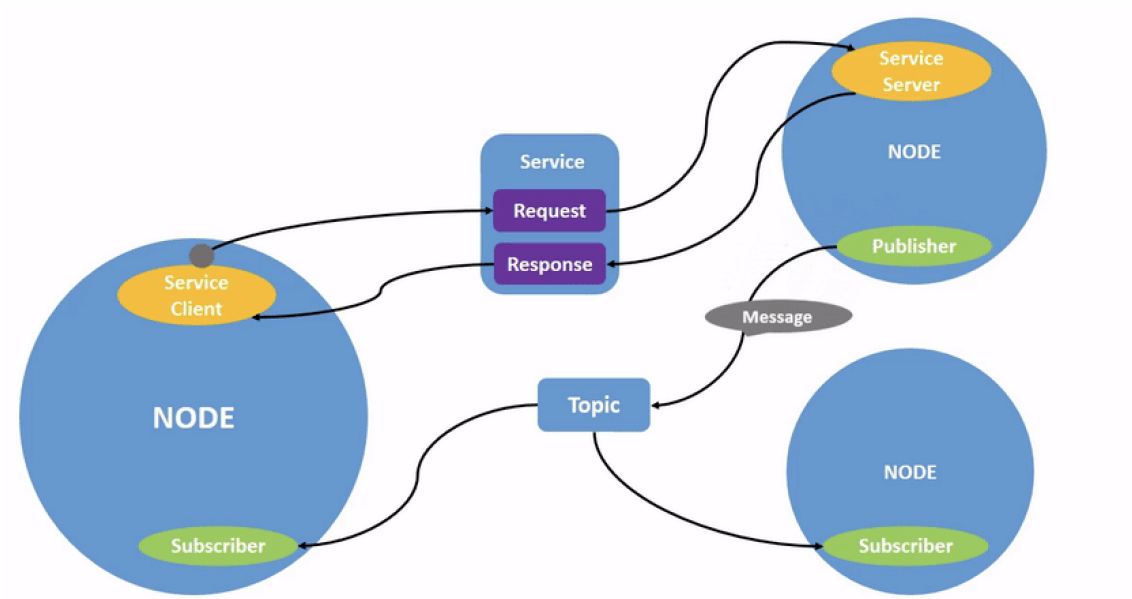


Figure 1: ROS nodes, topics and services

Source: *Understanding ROS 2 nodes* 2021

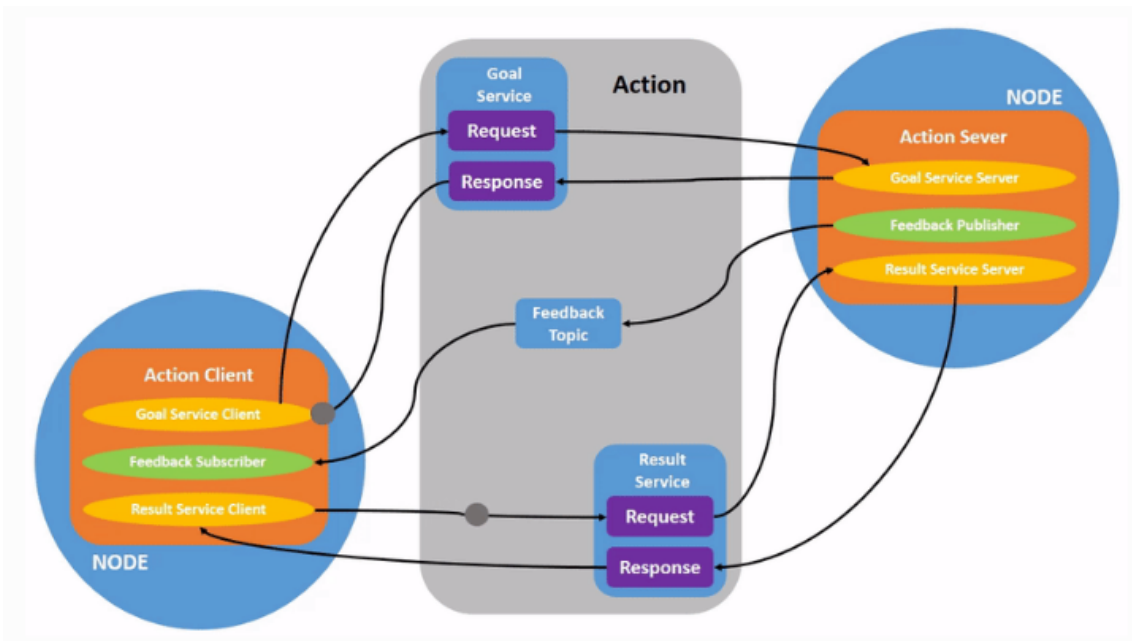


Figure 2: ROS actions

Source: *Understanding ROS 2 actions* 2021

allow everyone to complete their tasks. Informal and implicit communication is an easy task for humans, but very difficult for robots. One approach to solving this issue is using a central system with resource management, to plan the paths and actions for all robots in the environment. This issue becomes increasingly difficult once one considers the problem of multiple vendors. Another idea is to allow robots to handle conflicts peer to peer.

3.3.1 Robotics Middleware Framework

Managing a fleet of robots generates a new set of challenges and can be considered the next frontier for software like ROS. One attempt at solving the multi-fleet problem is Robotics Middleware Framework (RMF). At the core of RMF are modules for traffic monitoring, dispatch planning, scheduling and additional utilities. The goal is to allow multiple robots from multiple fleets from any vendor to operate in environments with shared resources. RMF incorporates adapters for third party vendors, infrastructure like elevators and doors, workcells with actuators and sensors.

RMF comes in a toolbox (RMF Toolbox) that in addition to its core, includes a traffic editor, simulation tools and a web-based user interface. The traffic editor can annotate floor plans with walls, doors and elevators, and manage traffic lanes. RMF Core is written in C++, however the C++ libraries have ROS2 wrappers, which makes their use language independent.

The RMF project also provides a free to use open source fleet management system, Free Fleet. Free Fleet can be used to control and manage the state of a set of robots. It does not provide planning capabilities like selecting the best suited robot for a task, or managing the use of shared resources.

RMF does not operate in unknown environments. The operating environment must be mapped out on beforehand.

3.4 Indoor navigation

Indoor mapping and localization is a hard problem, which can be solved in several ways. Some of the reasons that makes the problem hard are lack of, or poor GPS signals, high accuracy requirements, a lack of maps, dynamic environments and obstacles like stairs, elevators and doors.

Indoor localisation, or tracking, is often done with a real-time locating system (RTLS) which includes multiple different methods and technology depending on the application. RTLS can be based on GPS for global tracking, Ultra Wide-Band (UWB) antennas and receivers, Bluetooth Low Energy (BLE) beacons and devices, WiFi, choking points, etc. A different approach is Simultaneous Localization and Mapping (SLAM), which is what we have pursued in this project.

Segura et al. 2011 compares UWB-based localization to SLAM. They discuss the differences and compare the performance of the chosen UWB localization and lidar-based SLAM algorithm by running them in parallel. Their conclusion is that both systems can be used with high accuracy in an indoor environment. They found similar localization performance, visualized by figure 4 (SLAM) and 3 (UWB). Figure 5 aims to highlight the major differences of the systems. An obvious and important disadvantage to UWB localization is that robots are restricted to a sensed environment that depends on the distribution UWB antennas. Figure 3 shows one UWB antenna (red triangle) in every corner of the room they used as the test environment. This is generally the recommended setup, which requires a substantial investment in hardware and installation if it is to be widely adopted in larger areas like a university campus. The UWB localization system is only concerned with the localization problem and not mapping, which means the environment must be mapped out on beforehand, while SLAM can recursively build a map during operation. While the paper mostly focuses on localization performance, another advantage to a lidar based SLAM system is the ability to detect both stationary and dynamic obstacles, which a UWB-based system cannot.

We believe that SLAM algorithms in general have significant advantages to UWB localization and similar technologies, such as BLE, WiFi, UHF, that outweigh the disadvantages. There is also

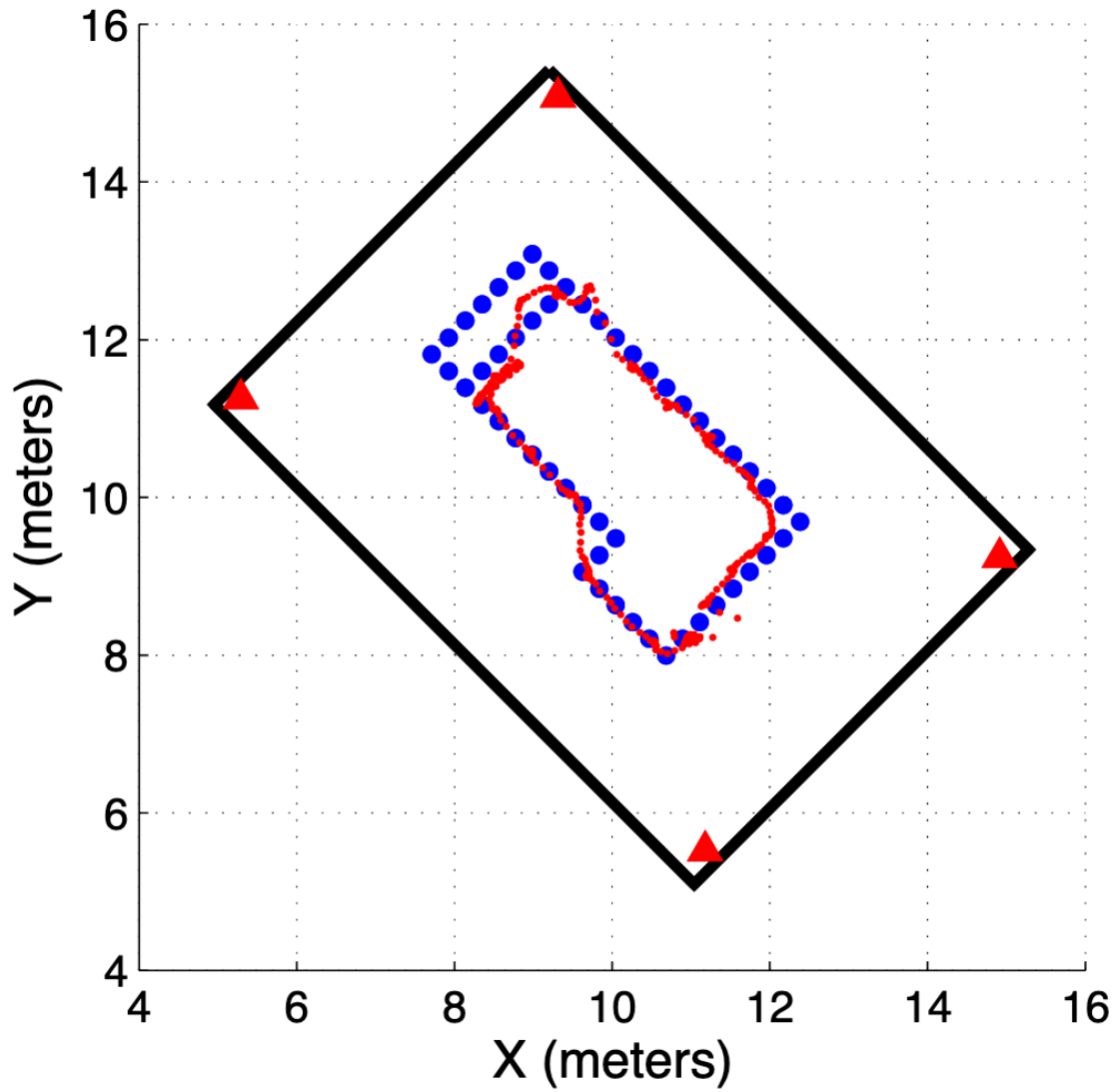


Figure 3: Localisation of a robot in a room using UWB. Red dots are estimated robot positions.

Source: Segura et al. 2011

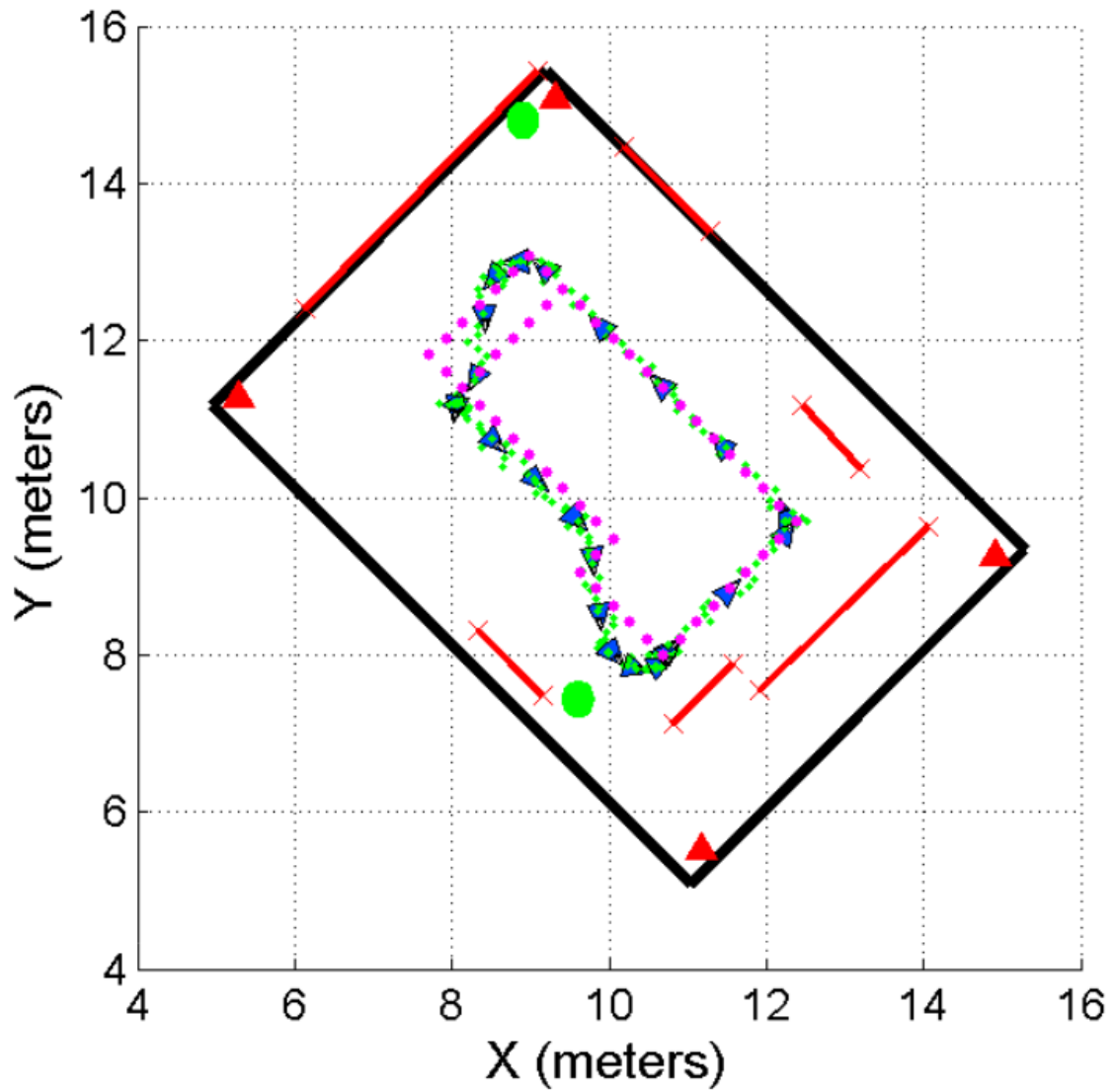


Figure 4: Localisation of a robot in a room using SLAM. Small green dots are estimated robot positions.

Source: Segura et al. 2011

UWB filtered localization system	SLAM algorithm
Depends on the UWB antennas disposition on the environment	Can map and localize in several maps of different sizes
An extra robot requires an extra UWB receiver	An extra robot requires a new range sensor appropriate for the SLAM algorithm
The localization error is not accumulative	The SLAM system state estimation error is accumulative until the SLAM closes the loop within the navigated environment (the mobile robot re-sees past features)
Concerns only localization problems	Acquires a map based on the features detection procedure incorporated on the SLAM algorithm
Features independent. The UWB localization system is not dependent on the nature of the features present within the environment	Depends on the detection and extraction of the determined features from the environment. If the SLAM algorithm does not extract any feature from the environment, then the localization errors are not minimized
Estimation does not turn into inconsistency	Due to bad features matching, the SLAM algorithm could turn into inconsistency
High dispersion of unfiltered data	Consistent estimation of the mobile robot navigated path
Covariance error remains bounded	Covariance error is minimized
Mobile robot autonomy is restricted to the sensed environment	Mobile robot autonomy is potentiated by SLAM [26]

Figure 5: Comparisons of UWB localization and SLAM algorithm

Source: Segura et al. 2011

significant improvement in SLAM algorithms since 2011, with one good Visual SLAM and Visual Inertial SLAM algorithm discussed in 3.4.1. In theory, a system using SLAM can be deployed with minimal effort with the only installation required being charging stations in addition to any task related installation. The robot(s) can then explore the environment and automatically build a map. The dynamic nature of the any environment with humans and the potentially simpler deployment is why we choose to experiment with SLAM instead of other RTL-Systems.

3.4.1 SLAM

Today's SLAM systems used in robotics are usually either based on laser scans or computer vision. Laser scans can either be in 2D with a sweeping laser range finder or in 3D with various solutions. Many projects use 2D laser scans for it's simplicity, affordability and decent performance.

Some popular 2D lidar-based SLAM-algorithms found in ROS are GMapping, Cartographer, Karto, Cartographer, Hector, and SLAM Toolbox. The most promising of these is SLAM Toolbox, which we have implemented in our prototype. This is discussed further in 4.2.2.

Figure 6 shows a summary of today's most relevant Visual SLAM, Visual Inertial SLAM, and Visual Odometry algorithms (Campos et al. 2021). ORB_SLAM was initially developed for monocular visual SLAM, and has later been released in version 2 and 3 with support for stereo and RGB-D cameras and inertial measurement units (IMU) in the latter. ORB_SLAM is an indirect visual SLAM system with local bundle adjustment and pose graph optimization. It also uses a bag of words database for place recognition and loop closing. The creators have tested each version of ORB_SLAM rigorously and compared it's performance to many other visual SLAM and visual odometry systems. Version one through three all have good performance, with version three at the top. See a summary in figure 6 (Campos et al. 2021).

Some of the qualities in ORB_SLAM that we consider to be particularly important for our use case are low computing power requirements, CPU-only algorithms, and robustness over time without unbounded growth in problem size. It avoids unbounded growth by only keeping keyframes and mappoints that are of high relevance and avoids too much redundancy. This means that one can likely implement ORB_SLAM3 on affordable low power device such as a Raspberry Pi 4 and continue operation over time within the same building.

4 What we have done

4.1 Hardware

To develop a platform for multiple autonomous robots we believe it is useful to build and test with a real robot. As expected, throughout the process of constructing the prototype robot, we met a number of challenges with both software and hardware. Some of these challenges are key to understanding the problem domain.

The prototype is a 4-wheel skid-steer robot with one motor and rotary encoder per wheel. Currently we have two motor drivers with two motors per driver. These drivers are controlled by one or more Arduino boards, acting as an interface between the physical equipment and our software. We created modular software to be able to quickly configure the software to use any number of Arduino boards to control the robot. This was because of unexpected electrical problems occurring with a single Arduino controlling all four wheels. While we managed to control two wheels with one Arduino UNO and expected the solution to scale up, the system did not behave in a deterministic manner with more than two wheels controlled by the Arduino. Under preliminary testing, the system works with two Arduino UNO's controlling two wheels at once, or with a single Arduino NANO Every controlling all four.

Arduino boards are single threaded and their support for asynchronous tasks are very poor, which is why we use them only as an interface. The Arduino boards sole purpose is to interface electrical

	SLAM or VO	Pixels used	Data association	Estimation	Relocalization	Loop closing	Multi Maps	Mono	Stereo	Mono IMU	Stereo IMU	Fisheye	Accuracy	Robustness	Open source
Mono-SLAM [13], [14]	SLAM	Shi Tomasi	Correlation	EKF	-	-	-	✓	-	-	-	-	Fair	Fair	[15] ¹
PTAM [16]–[18]	SLAM	FAST	Pyramid SSD	BA	Thumbnail	-	-	✓	-	-	-	-	Very Good	Fair	[19]
LSD-SLAM [20], [21]	SLAM	Edgelets	Direct	PG	-	FABMAP PG	-	✓	✓	-	-	-	Good	Fair	[22]
SVO [23], [24]	VO	FAST+Hi.grad.	Direct	Local BA	-	-	-	✓	✓	-	-	✓	Very Good	Very Good	[25] ²
ORB-SLAM2 [2], [3]	SLAM	ORB	Descriptor	Local BA	DBoW2	DBoW2 PG+BA	-	✓	✓	-	-	-	Exc.	Very Good	[26]
DSO [27]–[29]	VO	High grad.	Direct	Local BA	-	-	-	✓	✓	-	-	✓	Fair	Very Good	[30]
DSM [31]	SLAM	High grad.	Direct	Local BA	-	-	-	✓	-	-	-	-	Very Good	Very Good	[32]
MSCKF [33]–[36]	VO	Shi Tomasi	Cross correlation	EKF	-	-	-	✓	-	✓	✓	-	Fair	Very Good	[37] ³
OKVIS [38], [39]	VO	BRISK	Descriptor	Local BA	-	-	-	-	-	✓	✓	✓	Good	Very Good	[40]
ROVIO [41], [42]	VO	Shi Tomasi	Direct	EKF	-	-	-	-	-	✓	✓	✓	Good	Very Good	[43]
ORB-SLAM-VI [4]	SLAM	ORB	Descriptor	Local BA	DBoW2	DBoW2 PG+BA	-	✓	-	✓	-	-	Very Good	Very Good	-
VINS-Fusion [7], [44]	VO	Shi Tomasi	KLT	Local BA	DBoW2	DBoW2 PG	✓	-	✓	✓	✓	✓	Good	Exc.	[45]
VI-DSO [46]	VO	High grad.	Direct	Local BA	-	-	-	-	-	✓	-	-	Very Good	Exc.	-
BASALT [47]	VO	FAST	KLT (LSSD)	Local BA	-	ORB BA	-	-	-	-	✓	✓	Very Good	Exc.	[48]
Kimera [8]	VO	Shi Tomasi	KLT	Local BA	-	DBoW2 PG	-	-	-	-	✓	-	Good	Exc.	[49]
ORB-SLAM3 (ours)	SLAM	ORB	Descriptor	Local BA	DBoW2	DBoW2 PG+BA	✓	✓	✓	✓	✓	✓	Exc.	Exc.	[5]

Figure 6: Summary of the most representative visual and visual-inertial systems

Source: Campos et al. 2021

and digital control- and feedback-signals and communicating this with a Raspberry Pi running a Python program that controls the individual wheel speeds with a PID-controller. The idea is then to attempt to provide an actual wheel speed as close to the control signal as possible, which makes the job of navigation software easier. We setup a development environment that allows us to upload software to Arduino boards from the Raspberry PI, which means we can update the boards while the system is connected. We can also modify the Python software on the Raspberry Pi directly by using Visual Studio Code Server for faster small iterations.

While we initially expected to face some challenges while constructing the base robot platform, we met even more issues and quirks than expected, resulting in a time consuming job. The motor drivers are of low quality and are easily be destroyed resulting in unexpected electrical behaviour. The Arduino boards can be unstable and have connection issues with the Raspberry Pi or with our development laptops. For this reason we have ordered a new type of motor drivers and Arduino boards. The new Arduino boards also have an integrated IMU which will be useful for the sensor fusion in our SLAM system. Our progress during the project has been significantly hampered by devices not working as intended, and we have learned that quality components is important to enable our research. Preferably more complete and integrated components should be used in the future.

When we started building the prototype, there was already several types of robot chassis available to us. Initially, we were to experiment with the ones available to us, and if none of them were suited to our needs we were to buy new ones. There was three robots available to us, two track-based robots and one four wheeled skid-steer robot. We selected the four wheeled robot due to its size, as the the track-based robots were quite small. The decision to base our selection on size was due to several factors. Battery size is a limiting factor for how long a robot can operate before needing to be recharged, so in order to make the robots helpful rather than cumbersome we wanted to be able to fit a relatively large battery in the robot. We also need to fit a lot of electronics in the robot, so this requires space as well. One final factor is that one of the tasks we see as likely that the robot will be performing is delivering things, and the bigger the robot, the heavier and bigger items the robot can deliver.

4.2 Software

4.2.1 ROS

As mentioned in 3.4, one of the biggest common problems with controlling a robot is mapping and navigation, and there are multiple approaches to the problem. Recent SLAM algorithms have gained a great deal of improvement in robustness, speed and accuracy. Additionally, the robot can easily be deployed in a new appropriate environment.

With the motivation mentioned above, we have equipped the prototype with a 360 degree 2d-lidar scanner that feeds our chosen SLAM navigation system with data. The challenge then is to implement SLAM for the prototype. We discovered Robot Operating System (ROS) which is a modular software system with multiple frameworks and tools. It is widely used in the robotics community and many open source packages come with ROS integration. So far we have implemented Nav2 and ros2_control. Nav2 is a suite of navigation tools for SLAM-based navigation. Ros2_control contains packages for controlling robots of several different setups, such as our skid steer setup.

Ros2_control translates the wheel speeds of the robot into velocity information published on an odometry topic Nav2 listens to. Ros2_control also subscribes to a velocity command topic that Nav2 publishes commands to. Ros2_control then translates these to required wheel speeds which are sent to the aforementioned Python program running PID regulators for wheel speeds through a hardware interface that we wrote for our specific robot, as a plugin to Ros2_control.

4.2.2 SLAM_toolbox

SLAM_toolbox is a set of tools that easily allows 2d-SLAM and lifelong mapping. This means maps of the environment can be saved and continuously refined in order to create the most accurate maps possible. SLAM_toolbox subscribes to the /laserScan-topic, the topic our Lidar-node publishes its raw readings to. SLAM_toolbox then converts these readings into features and estimates the robots pose by looking up a pose in the pose graph that observes similar features. SLAM_toolbox then publishes the estimated map and the robots location in said map on the /map-topic.

When we were going to select a SLAM-algorithm, we had to select one that fit our needs and one that was possible to implement with the sensors we had at hand. As the goal for the project was to gather information in order to help us try to answer the defined research questions, we wanted to start simple and chose to only use a combination of Lidar and wheel encoder data, in order to try to get our selected SLAM algorithm to work with our robot. Upon success, we wanted to add other sensors if needed. In addition to being Lidar-based, we wanted a SLAM-algorithm that would work in real time in large areas. Some popular Lidar-based SLAM-algorithms found in ROS are GMapping, Cartographer, Karto, Cartographer and Hector. But something these algorithms have in common is that struggle to perform SLAM in large areas in real time (Macenski and Jambrecic 2021). In the end we found ROS package SLAM_toolbox, which suited our needs. SLAM_toolbox is based on the Open Karto-library, which means it uses correlative scan matching on the front end and Sparse Pose Adjustment on the backend (Kaijaluoto et al. 2015). As of now, we are only using the SLAM tools in the package, but we are planning to use the tools for lifelong mapping in the future.

As of now we have only implemented SLAM with Lidar. Only using one source of odometry makes SLAM prone to drift, which in turn may cause inaccurate position estimations in the future. We therefore want to incorporate IMU- and wheel encoder data when we perform SLAM in the future, in order to cancel out any potential drift. We are also interested in implementing a camera, so we can perform visual SLAM as well, in order to ensure the most accurate SLAM possible.

4.2.3 Navigation2

Navigation2 or Nav2 is as aforementioned a suite of tools that aids in the task of helping a robot navigate. The backbone of Nav2 is the behaviour tree structure, where each node of the tree

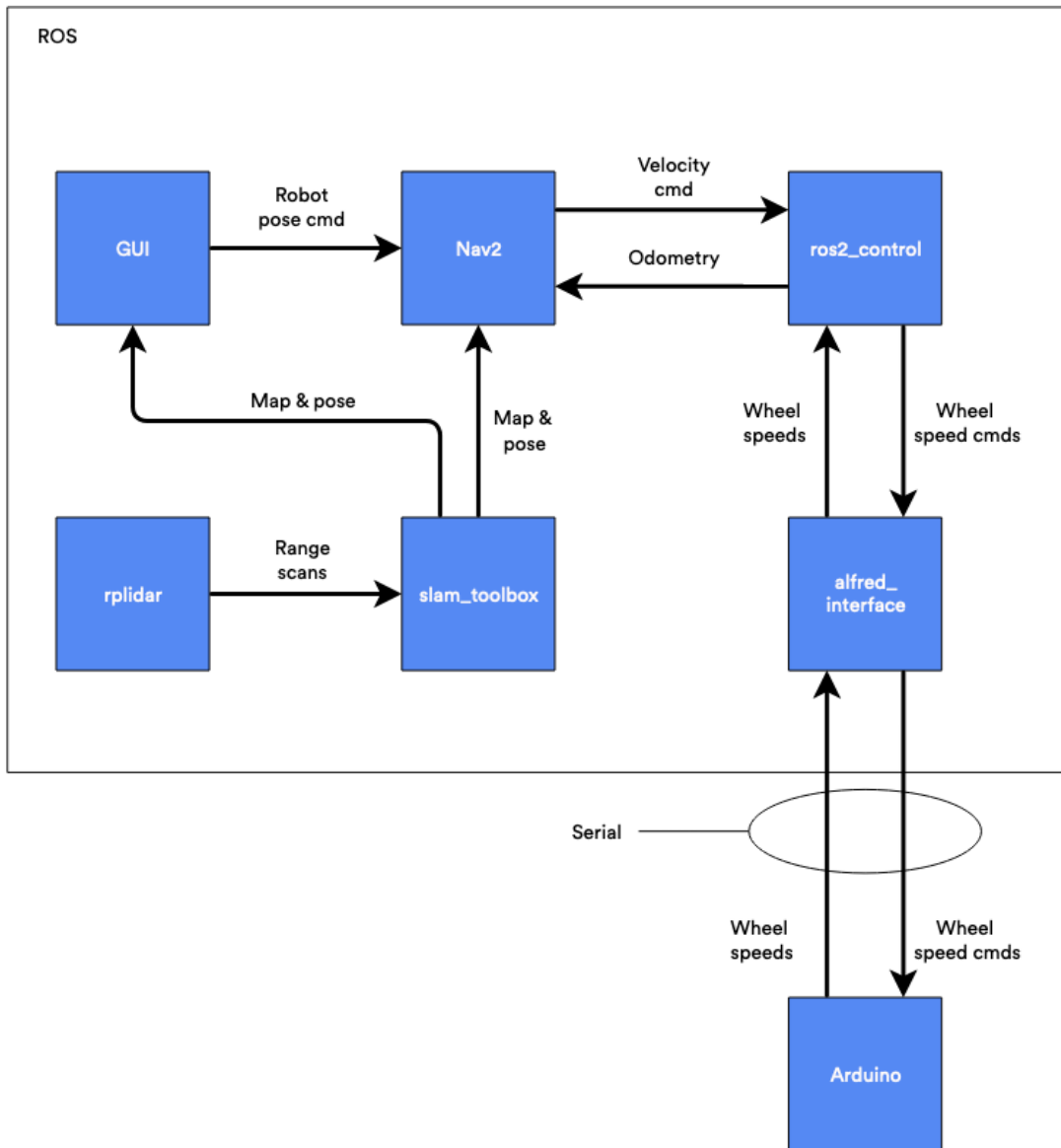


Figure 8: Prototype system architecture

related to our project. However, our institute did not offer any modules that were relevant, we therefore had to take modules provided by the Department of Engineering Cybernetics(ITK). The following subsections will give a brief description of the module, and what we learned.

4.3.1 TTK30 - Human-machine/autonomy interaction in cyber-physical systems

This module gave insight into how systems involving some sort of autonomy is designed, with a focus on keeping the humans that interact with the system "in the loop". The module also gave a short overview of the paradigm shift the precursors of the systems of today went through, as the design went from being tech-centric to being human-centric.

The module turned out to be less relevant to our project than we initially thought, as the systems that were discussed in the course were the systems found in air planes and cargo ships and similar systems, but we still think there were some ideas that are relevant for when we are going to be designing the planned web interface.

4.3.2 TTK21 - Introduction to Visual Simultaneous Localization and Mapping - VSLAM

This module was a introductory course for Visual Simultaneous Localization and Mapping, or trying to solve the SLAM-problem using some sort of camera. The module gave an introduction to the math used in SLAM. This included Lie-theory, stereo- and epipolar geometry, factor graphs and linearization of non-linear functions on the manifold. The module also gave a short review of the VSLAM-algorithms that are considered state-of-the-art.

We found this module quite difficult as there were a lot of theory packed in the course, and a lot of that theory were built on mathematics and theory that the cybernetics-students already had covered in their studies, so this was not elaborated upon in this module. This caused the module consuming a lot more of our time than the credits would suggest. Regardless of the time consumption, it was a very interesting and useful module as it provided us with the ORB-SLAM 3 algorithm, which we want to replace our current SLAM-algorithm with next semester.

5 Conclusion

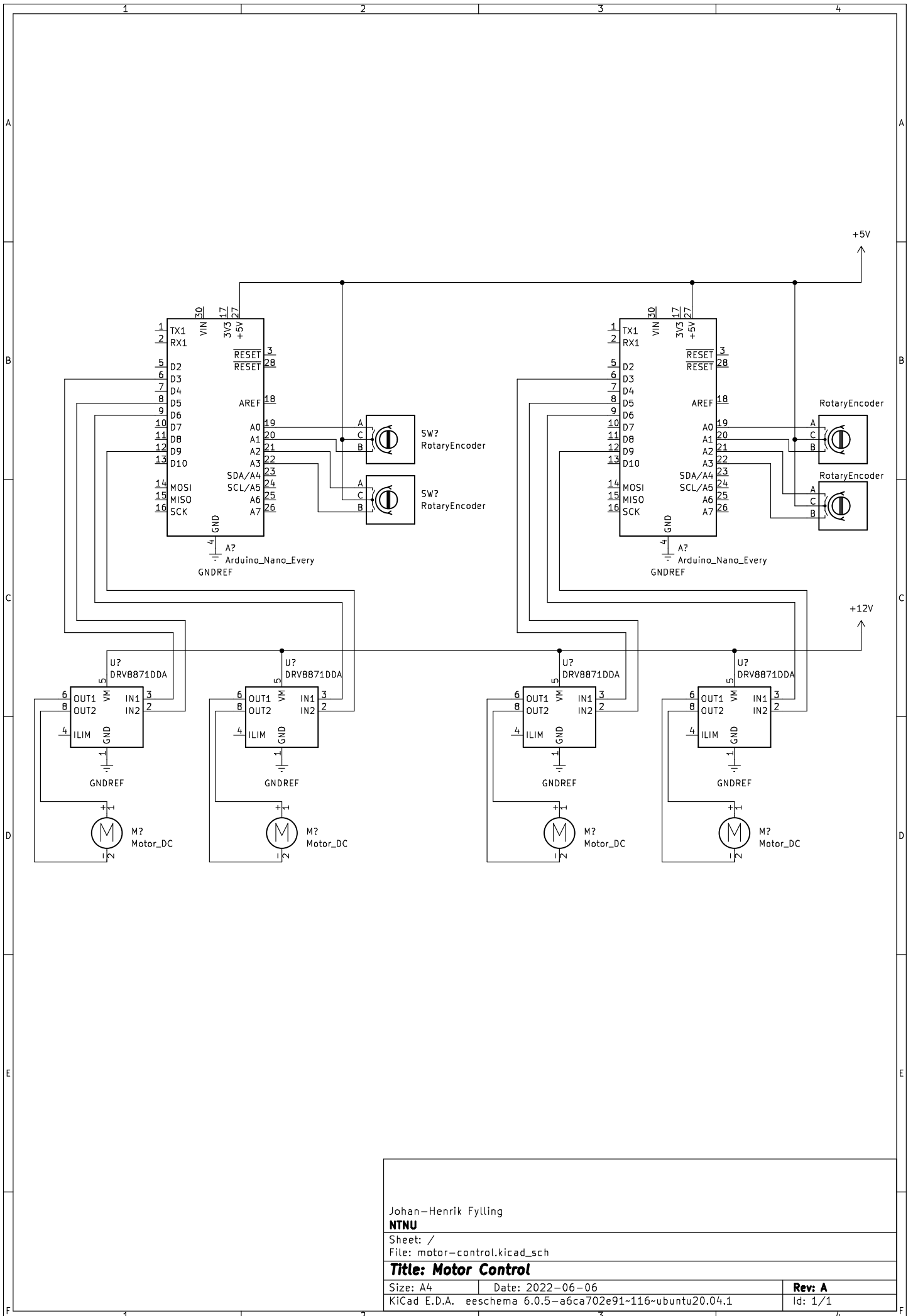
We believe we have gained enough theoretical knowledge about autonomous robotics to understand the problem and other related research, and enough knowledge about related work to form opinions on how to proceed in the next semester. In addition, the practical experience from building our own prototype will help speed up any prototyping processes. We initially had a goal to produce a usable prototype, enabling further research and we believe we are close to having such a prototype platform although it remains untested in it's latest form.

Bibliography

- Campos, C. et al. (2021). ‘ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM’. In: *IEEE Transactions and Robotics* 37, pp. 1874–1890.
- Detailed Behavior Tree Walkthrough* (2021). URL: https://navigation.ros.org/behavior_trees/overview/detailed_behavior_tree_walkthrough.html (visited on 6th Dec. 2021).
- Kaijaluoto, R., A. Kukko and J. Hyypä (2015). ‘Precise Indoor Localization For Mobile Laser Scanner’. In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XL-4/W5*, pp. 1–6. URL: <https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XL-4-W5/1/2015/>.
- Macenski, Steve and Ivona Jambrecic (2021). ‘SLAM Toolbox: SLAM for the dynamic world’. In: *Journal of Open Source Software* 6.61, p. 2783. DOI: 10.21105/joss.02783. URL: <https://doi.org/10.21105/joss.02783>.
- Segura, Marcelo J. et al. (2011). ‘Ultra Wide-Band Localization and SLAM: A Comparative Study for Mobile Robot Navigation’. In: *Sensors* 11, pp. 2035–2055. URL: <https://doi.org/10.3390/s110202035>.
- Understanding ROS 2 actions* (2021). URL: <https://docs.ros.org/en/galactic/Tutorials/Understanding-ROS2-Actions.html> (visited on 7th Dec. 2021).
- Understanding ROS 2 nodes* (2021). URL: <https://docs.ros.org/en/galactic/Tutorials/Understanding-ROS2-Nodes.html> (visited on 7th Dec. 2021).

Appendix F

Wiring Diagram Alfred



Johan-Henrik Fylling		
NTNU		
Sheet: /		
File: motor-control.kicad_sch		
Title: Motor Control		
Size: A4	Date: 2022-06-06	Rev: A
KiCad E.D.A. eeschema 6.0.5-a6ca702e91-116-ubuntu20.04.1		Id: 1/1

Appendix G

Evaluation Guide for Participant Observation

Evaluation guide for participant observation:

1. What kind of participant observation was used?
2. How long did the researcher spend in the field? Do you think this was long enough?
3. Did the researcher avoid disrupting the naturalness of the setting?
4. Has the researcher reflected on self-identity and how it affected access, perception of events and the reactions of others?
5. Has the researcher discussed the ethics of the fieldwork and any personal difficulties encountered?
6. What methods has the researcher used to try to convince you of the validity of the observations?
7. Has the participant observation led to insights that would not be possible using other methods?
8. What limitations in the use of participant observation does the researcher recognize!
9. Can you identify other flaws or omissions in the researchers' reporting of the use of participant observation?
10. Overall, how effectively do you think the use of observations for data generation has been reported and used? This has been a supplement to other research methods, and has therefore not been used to its fullest. There is much more potential in this research method.

Evaluation guide found in *Researching Information Systems and Computing* [7]

Appendix H

Moderator Script for Usability Test

Hei, til å starte med vil vi takke deg for at du deltar, det er til enorm hjelp for oppgaven vår. Som du sikkert husker er vi Johan og Kevin, vi er to masterstudenter som har laget en prototype av en robot og en prototype av systemet vi har planlagt skal styre en flåte med lignende og på sikt veldig forskjellige roboter. Idag skal du teste disse prototypene.

Dagens agenda består av en brukertest og et kort intervju. I brukertesten skal du gjennomføre et slags rollespill der du later som du er deg selv på jobb som skal bruke systemet vårt, og du vil få et par oppgaver du skal gjennomføre. Intervjuet vil omhandle din opplevelse av å bruke systemet og tanker du gjør deg underveis.

Før vi starter brukertesten er det par ting vi vil nevne først:

- **Vi vil gjøre opptak av testen, samtidig som Johan vil notere ting han observerer underveis. Jeg har rollen som testleder, men jeg vil ikke kunne hjelpe deg med å gjennomføre oppgavene. Du må gjerne spørre om ting, så skal jeg prøve å svare så mye jeg kan**
 - **Dersom du stiller spørsmål kan det hende du får spørsmål tilbake i ånden av “hvordan tror du at du skal fortsette?”**
 - **Dette kan nok oppleves som irriterende, og det beklager jeg på forhånd**
 - **Målet vårt er som sagt bare å finne ut hva som funker og hva som ikke funker**
 - **Om du ikke kommer deg videre går vi bare videre til neste oppgave**
- **Om du vil stoppe testen eller ta en pause underveis, er det bare å si ifra.**
- **Det viktigste for oss som du gjør under testen er at du tenker høyt**
 - **Hva vi mener med dette er at vi ønsker du skal si høyt hvordan du tenker når du gjør ting, og hvorfor du gjør de tingene du gjør**
 - **F.eks, om oppgaven hadde vært “Vennligst ta en bit av sjokoladen inni det papiret”, så kunne tankeprosessen vært slik**

