UiO : Department of Informatics
University of Oslo

# A Low-Cost Autonomous Mobile Robot for the Eurobot Competition

Eivind Wikheim

Master's Thesis Autumn 2015

# A Low-Cost Autonomous Mobile Robot for the Eurobot Competition

Eivind Wikheim

August 3, 2015

# Abstract

The Eurobot Open challenge provides an interesting and challenging arena for the research, development and testing of autonomous mobile robots. It is aimed towards young people and students, and seek to, in their words "encourage the practice of sciences in a friendly atmosphere".

Even though anyone can participate in the competition, it is difficult to compete for the top spots without large budgets and backing from commercial sponsors. Teams from leading universities are routinely spending as much as 100,000 NOK on their projects. In this thesis, it is investigated whether it is possible to design and implement an autonomous mobile robot for the Eurobot 2015 challenge on a limited budget, and still be able to compete against these teams.

An autonomous mobile robot capable of solving the tasks presented by the Eurobot challenge has been successfully designed and implemented for less than 10,000 NOK, which is a fraction of the amount spent on the robot by the leading teams. The robot proved its worth in the international competition, held in *Yverdon-les-Bains* in Switzerland in May of 2015, by successfully completing its matches, scoring an acceptable amount of points, and winning two of its matches.

The ambitious goal of competing with the top teams was not achieved, and the results of this thesis suggest that it is difficult to do so without a higher budget and more sophisticated components.

# Acknowledgements

I would like to thank my supervisor Kyrre Glette and co-supervisor Kim Mathiassen for their support and guidance throughout the project.

I would also like to extend a thanks to the other faculty members at ROBIN for their help and input during the project, and in particular to Head Engineer Yngve Hafting, who has assisted us with suggestions for practical solutions, and helped with the acquisition of many of the components used in the project.

A thanks also to my fellow master students, that has made working on this thesis a little brighter, and helped with discussions and suggestions, and of course the daily chess game. And finally I thank my family, that has supported me throughout my education and the writing of the thesis.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Introduction

Since robots were first introduced into the manufacturing industry , robots have been an important part of ... (something furthering the work-force or something), by taking on tasks that are difficult, dangerous or simply too menial to be performed effectively by humans.

Mobile robotics have been important for expanding the areas in which robots can be used to solve tasks, and mobile robots are now used in fields outside manufacturing, such as farming[19], storage facilities[21] and nuclear waste cleanup[32]. Over the last decade, mobile robots have also become commonplace in private homes, where robots are used to solve menial tasks like cleaning and vacuuming[1] and lawn-mowing[36]. This frees up the time of the users, allowing them to spend their time on more important tasks, like watching TV and/or browsing the web.

There are multiple factors that have allowed for this expansion to happen, and the arguably most important of these factors have been that the technology and hardware used in mobile robots have become cheaper and more readily available, allowing companies to build robots on budgets that the average person can afford. Components like small, accurate and efficient electric motors, light high-performance batteries, and sophisticated sensors for obstacle detection that only a few years ago were reserved for high-budget commercial and military applications are now cheap enough to be used in low-budget robots for private use.

## 1.2 Eurobot

Eurobot is an annual, international amateur robotics with a vision of "encouraging the practice of sciences in a friendly atmosphere". It was founded in 1998 by the French voluntary science organization Planète Sciences, and has taken part every year since. The goal of the competition is to create autonomous, mobile robots that are able to solve a series of challenges. An example of two such robots can be seen in Figure 1.1.

Every year hundreds of students and members of robot-interested clubs and organizations take part in the event, challenging their skill in the areas of programming, electronics, project management and more.

The competition itself sees teams face each other in matches, where the winners are the team whose robots score the highest amount of points. Points are awarded for completing tasks, which are designed to challenge the robots navigation, precision and spatial awareness. The tasks are loosely related to a theme chosen by a committee, and serves to give each years competition an unique flavor. The theme, tasks and rules of this years competition is described in detail in Chapter 3. Contrary to robot fighting competitions where the goal is to destroy or disable the opponent, the Eurobot rules focus heavily on fair play, and avoiding to collide with or damage opposing robots and other obstacles is an important part of the challenge.

*Figure 1.1: Two robots from Eurobot 2015.*[1]

The only team that have been representing Norway in earlier years competition have been teams of master students from the Norwegian University of Science and Technology (NTNU), who have competed every year from 2000 to 2012. They will not, however, be competing in this years competition.

## 1.3 Motivation

Something that became apparent during the initial research phase was that even though the vision of Eurobot focuses on fair play and scientific achievement, the competition for the top spots is a fierce competition between the teams with large amounts of resources, manpower and funding. The most dedicated teams have as many as 15 to 20 contributing

members, and administer large budgets with backing from multiple commercial sponsors.

Acquring data about net spending of Eurobot teams is difficult, as teams rarely discuss their budgets and funding openly. One of few teams that do, occasionally, reveal their project spending are the teams from NTNU. According to their papers they have spend between 65,000 NOK and 100,000 NOK each year[37, 53]. Discussions with members from other teams suggest that many teams regularly spend even more than this, and have robots with total part costs of between 100,000 NOK and 150,000 NOK.

This results in the division of participants into two separate classes; the teams with large amounts of resources and funding that can realistically compete for the top spots, and the less resourceful teams that are left to compete amongst themselves for placements on the bottom half of the table. Another consequence of these kinds of budgets is that the resulting robots, in our opinion, are not as interresting as they potentially could be, if the various components were closer to a price-range where they could realistically be used in mobile robots for everyday applications.

### 1.3.1 Goals

The main goal of this thesis is to investigate whether it is possible to successfully build robot for competing in Eurobot with a small team and a limited funding, and still manage to stay competitive with the teams with more resources. This will be extra challengeing as it is the first time a team from the University of Oslo (UiO) is participating in the competition, so all parts of the project has to be started from scratch.

The work of the thesis will include:

- Organization of the project.
- Selection of components.
- Design and construction of the mechanical and electrical parts for the robot.
- Implementation of communication and control systems.
- Participation in the international Eurobot competition.

## 1.4 Project Organization

As mentioned in the the previous section, this will be the first time a team from UiO participates in the Eurobot competition. This makes the project more demanding for a couple of reasons. No members of the team with the project has any previous experience from Eurobot, nor has any of the other personnel affiliated with the project.

This lack of experience means that extra work has to be put into the project management and planning stages, to ensure that a good timeline for the completion of the project is developed and executed properly. Additionally the workload will be significantly higher as there is no preexisting components, code or materials to use as a starting point. Every part of the project must be built from the ground up, meaning that a large amount of components and tools must be either purchased or manufactured manually.

#### Workload distribution

The project is set up as a group task in an effort to distribute the workload. As it was decided that each student would write a separate thesis, each of the members were given an independent focus area. The team worked together on many aspects of the project, but this focus area is what each of the members have written their thesis about. Dividing the project into three areas of responsibility is no trivial task. It was critically important to not only ensure that all vital parts of the projects would be handled without any team member in particular being overwhelmed, but also that each member would have enough material on which to write their thesis.

| **Member** | **Part** | **Content** |
|---|---|---|
| André Kramer Orten | NAVSYS | Navigation, localization and obstacle avoidance |
| Bendik Kvamstad | AI | Planning and task execution |
| Eivind Wikheim | CTRLSYS | Design and construction of the physical robot and its control systems |

*Table 1.1: The final project division.*

The final distribution of the project was as shown in Table 1.1. This thesis will describe CTRLSYS, while NAVSYS and AI are described in separate theses by André Kramer Orten[46] and Bendik Kvamstad[40] respectively.

Even though the project is split into parts, the team decided that we would work as closely together as possible to solve the tasks, both to improve progression and to avoid problems with integrating the various parts of the projects in the later stages of the process. This has been described as a problem for teams from NTNU when they have split the project up in multiple parts, and is something we hope to avoid[23, 37].

**NAVSYS** will be tasked with keeping track of the robot's position in the global coordinate system, as well as tracking the position of other robots and important game pieces. The detection and avoidance of the

opposing robots is especially important, as colliding with them may cause disqualifications.

**AI** performs strategic planning at a high level, deciding where to move and what tasks to perform. This information is relayed as commands to CTRLSYS, which attempts to execute the commands.

**CTRLSYS** involves acquiring components for and building the physical robot, including a motor and power system, tools for solving various tasks. In addition this part will include setup of the main computational unit that all parts of the project will use, as well as the creation of the control systems for all these parts. CTRLSYS does no intelligent planning, in order not to interfere with AI, and simply attempts to execute the commands to the best of its ability.

## 1.5 Thesis Outline

**Background** describes concepts and techniques of components and tools used throughout the project.

**Eurobot 2015** first discusses the rules of the Eurobot competition in general, and then explains the tasks and point system of this years competition.

**Implementation** documents the entire process of building the robot, from the implementation of the game area, robot chassis and motor system to the control software.

**Experiments and Results** contains tests and experiments performed to investigate the performance and stability of the various parts of the robot. Each section contains a description of how the experiments were performed, a presentation of the data itself, and an analysis section discussing the results in detail.

**Discussion and Conclusion** expands on the analysis performed in the previous chapter, and discusses how well each part of the robot functions in practice. A detailed account of how the robot performed in the international competition is provided, and what was learned from this experience. Finally the chapter discusses how well the goals of the thesis were satisfied overall, and what parts of the project could have been solved differently. Lastly there is a section about future work, and how the robot potentially can be reused and improved for later competitions.

# Chapter 2

# Background

## 2.1 DC Motors

The most common way to power the locomotion of mobile robots are with direct current (DC) motors. This type of motors is a class of electrical machines that convert direct current electrical power into mechanical motion[41].

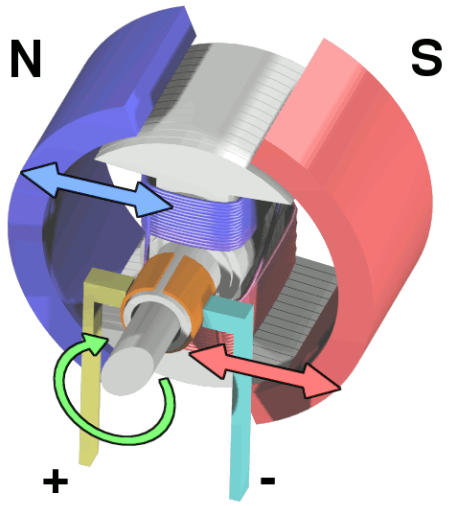



*Figure 2.1: Illustration of the basic principles of a brushed DC motor.[11]*

The basic working principle of a dc motor are illustrated in Figure 2.1. The output shaft of the motor is connected to a rotor, which contains armature windings capable of creating a magnetic field. The magnetic field is perpendicular to the magnetic field created by stator windings located around the rotor. When current is introduced into the windings of the rotor the magnetic field is created, and the attraction/repulsion between this fields magnetic poles and the poles of the stator poles causes the rotor to turn. As the poles approach each other the polarity of the current trough the motor is reversed, reversing the polarity of the rotor and continuing its

rotational movement. This is accomplished by a commutator, which acts as a mechanical switch for reversing the current. Reversing the polarity of the circuit connected to the commutator reverses the direction of the rotor, and in turn the direction of the motor shaft.

#### Drawbacks

The mechanical construction of the regular, brushed DC motors have multiple drawbacks. The electrical circuit is connected to the commutator via components known as brushes, that keep the connection to the rotating commutator and shaft by gliding along it as it rotates. This causes a slight amount of power-loss due to friction, and the parts in contact degrade over time. This also means that the brushes have to be replaced regularly for the motors to function optimally.

#### Brushless DC motors

Brushless direct current motors solve these problems by removing the need for a direct electrical connection to the moving rotors armature windings. Instead, the magnetic field in the stator windings are rotated around the motor using an electrical controller, which produces an ac waveform based on the dc input[41]. The rotor consists of permanent magnets, and as the magnetic field in the stator windings is rotated around the motor, the rotor moves to keep up. This design removes the need for contact between parts, which reduces friction and the increases the performance of the motor significantly. Reliability is also increased as the parts do not wear down over time, and drastically reduces the need for regular maintenance. The only drawback of these motors compared to the regular brushed type is the added complexity of the electrical controller, which increases the price. Despite the added cost, the advantages have led to brushless motors replacing brushed in almost all applications.

#### Gears

DC motors usually produce motion with high speed and low torque, and must be geared down in order to scale the speed/torque characteristic of the output to what is required for the desired movement velocities. The power output by a dc motor can be described as in Equation (2.1), the rotational power of the motor $P_{rot}$ is a product of the torque $\tau$ and the angular velocity of the motor shaft, $\omega$. Increasing the gear ratio will simultaneously increase the amount of torque and decrease the rotational speed of the motor shaft, while the rotational power stays constant.

$$P_{rot} = \tau * \omega \tag{2.1}$$

### 2.1.1 Gear Backlash

A common problem with gears and geared motors, and especially low-cost variants, are high amounts of backlash. Backlash is an effect leading to loss

of motion during motor movement, and is a result of excessive clearance between the components inside the gear[6]. This concept is illustrated in Figure 2.2.

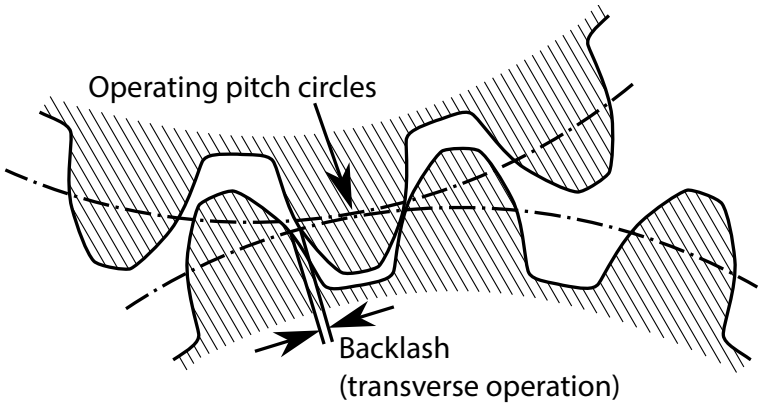



*Figure 2.2: Illustration of the backlash concept[12].*

Clearance is necessary in order to compensate for manufacturing tolerances, and to allow for proper lubrication between parts. Not allowing for sufficient clearance between gear components can result in excessive wear on gear teeth, high heat generation and interference between gear teeth.

**Effect on Movement**

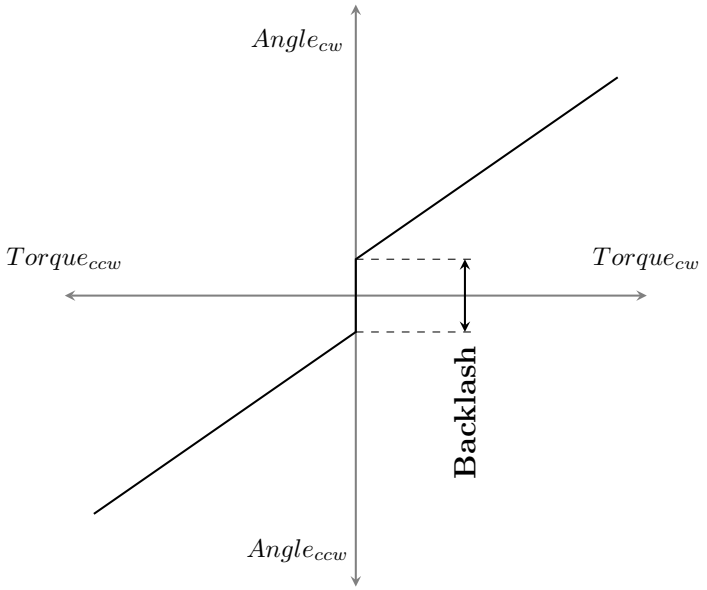



*Figure 2.3: Illustration of the effect backlash has on movement. While changing rotational direction, there is a gap where the motor produces no torque in either direction.*

Figure Figure 2.3 illustrates the effect of backlash on rotational movement. The measured backlash value is how far the motor has to travel after a load reversal before the motor shaft is engaged and the wheel starts turning. The effect on the robots transitional movement is easily calculated as shown in Equation (2.2), where $b_d$ is the distance error caused by backlash, $W_r$ is wheel radius and $b$ is the backlash in degrees.

$$b_d = \frac{(2 * \pi * W_r)}{360} * b \tag{2.2}$$

## 2.2 H-bridge Motor Controller

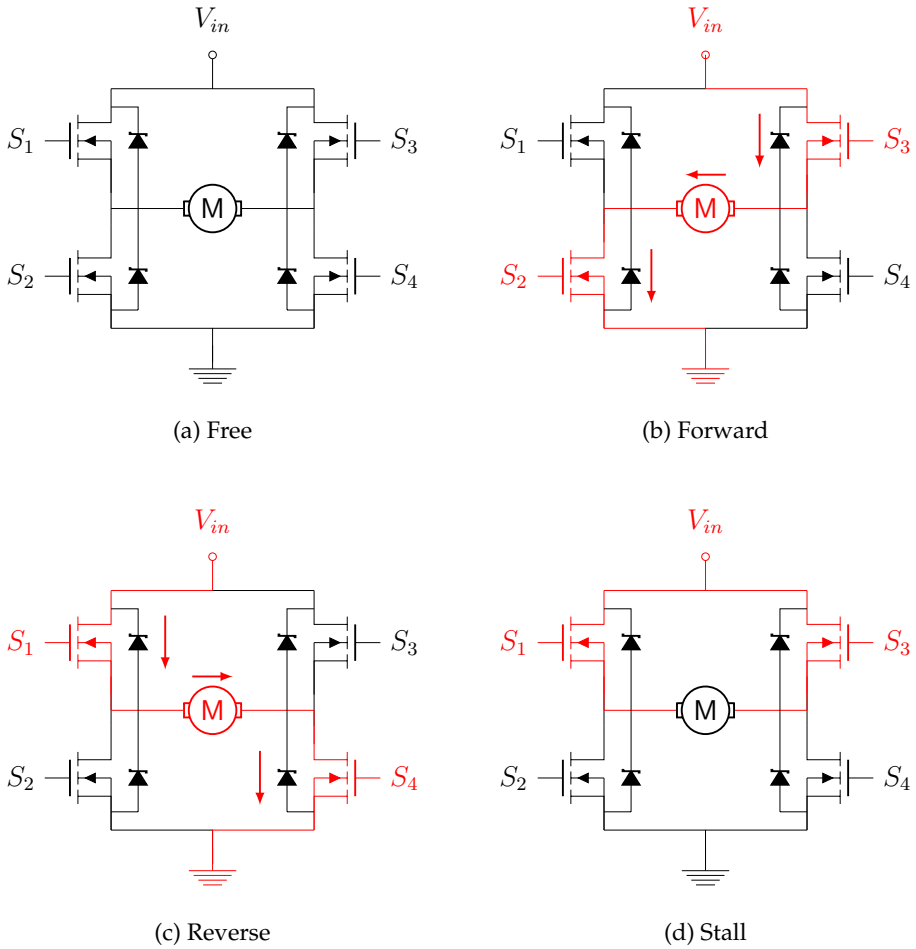
As digital circuits like microcontrollers usually can not provide enough power to drive a motor, additional driver circuits are required. A common circuit to use for this purpose is know as a H-bridge circuit. This type of circuits the most common switch configuration for power electronics. It is commonly used in components like voltage convertors, induction heating and motion control. The reason to use a H-bridge for motor control is that it is a good way to provide high current and high voltage to a load such as a motor[28]. The H-bridge circuit consists of two separate sides, called half-briges, each with two transistors acting as switches. The state of the motor is decided by which transitors are open at the time.

### 2.2.1 States

Figure 2.4 illustrates all the states of a H-bridge controlling a motor, $M$, where open transitors and wires with current flowing through are colored red.In Figure 2.4a all transistor switches are closed, and no current passes across through the circuit. This means that none of the magnetic fields in the motors windings are active, and the motor shaft is allowed to rotate freely. In Figures 2.4b and 2.4c, one transistor connected to $V_in$ and one connected to ground is opened on opposite sides of the motor, allowing current to flow across the motor. This actuates it and causes the motor shaft turn. The rotational direction of the shaft depends on the direction of the current. The final state, illustrated in Figure 2.4d, is called the stall state or brake state. In this state both supply lines to the motor are connected, effectively short circuiting the motor. In this state the motor pulls its maximum amount of current, called *stall current*, and continues to provide a high amount of torque while keeping the motor shaft stationary. If the circuit is left in a stalled state for long periods of time, the high amount of power may lead to overheating and possibly damage to components.

### 2.2.2 Over Voltage Protection

H-bridge controllers are usually outfitted with diodes in order to prevent damage to components during voltage spikes. The reason this is necessary is that DC motors during deceleration act as generators, transferring kinetic energy from motion to electric energy in the form of increased voltage in




(a) Free (b) Forward

(c) Reverse (d) Stall

*Figure 2.4: H-bridge motor controller states. Components in red mark open connections.*

the circuit. The diodes open paths directly to ground and $V_{in}$ to prevent the spike from potentially harming components on the controller, as shown in Figure 2.5. If the direction of travel of the motor is reversed, the polarity of the voltage is reversed, and the other pair of diodes will be opened instead.

## 2.3 Servo Motors

In some cases there is necessary to control the exact rotational position of the motor. A common and inexpensive way to do this is to use servo motors, which are a type of rotary actuators that utilize closed loop feedback to allow for precise control of angular position and rotational speed[35]. Complex P.I.D. controllers (explained in Section 2.4) are used to control the actuation of the motors. Servo motors have become popular for use in remote control, robotic and manufacturing applications, and are used for both hobbyists and commercial applications. The main advantage

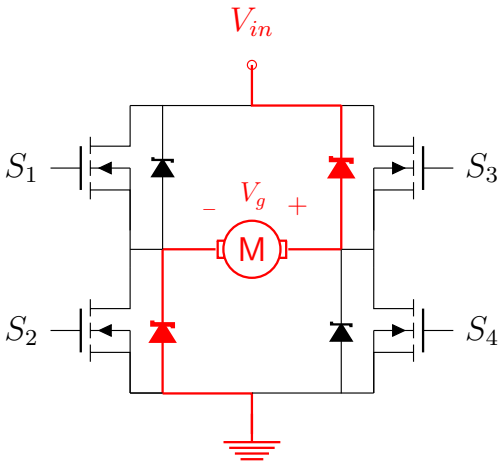



*Figure 2.5: Open path trough diodes during overvoltage. Reversing motor direction will reverse polarity of the voltage generated, $V_g$, and the two other diodes will be opened instead.*

of the servo motor is that it provides a commercial off-the-shelves (COTS) solution to the problem of motor position control, and removes the need for designing, building and testing complicated control systems.

## 2.4 P.I.D Control

P.I.D. is an abbreviation for *"proportional, integral, derivative"*, and these terms are descriptive of how the system works[5]. A typical PID controller differentiates system input (the desired output, e.g. the goal position of a motor), with feedback from the controlled object to get an *error signal*. The integral and proportional elements are calculated from this error signal, while the derivative is calculated directly from the systems feedback output. These factors are used to tweak the system to steer the output faster and more precisely towards the desired output.

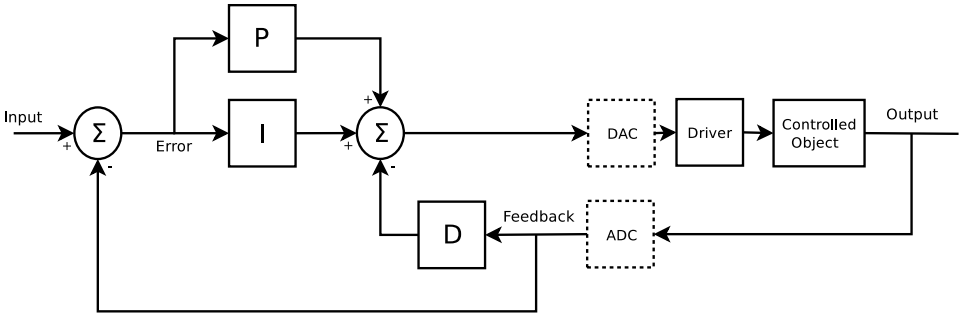



*Figure 2.6: Example of a basic P.I.D. controller.*

### Gain Factors

Each of the control terms are multiplied by a associated *gain value*, that can be tweaked to control how much influence each element has on the output of the system. Achieving a deep understanding how these different gain factors for the proportional, integral and derivate terms affect the output of the system requires a solid understanding of complex control theory. This is however not required in order to simply tune a PID circuit to give a desired response. A basic understanding of each component is enough to use a trial-and-error process to tune the controller, and there is also programs that can perform this operation[33, 55].

### Proportional Gain

The proportional term is the simplest to understand and implement, as the calculation simply is the current error-value multiplied by the gain constant $K_p$[55]. Raising the $K_p$ value increases the rate at which the output accelerates towards the target, while simultaneously increasing the risk over overshooting the desired out, and may lead to oscillation of the output.

### Integral Gain

Integral control is used to correct for any steady offsets from the desired value[33]. Integral control by itself does not increase system stability, but it is used in tandem with proportional control. As the integral is based on previous changing of the signal, it means the output will move faster back towards the goal after an overshoot, and result in a smoother response curve. The integral term does not necessarily decrease the time until the system settles, in fact it may increase it, but a good integral gain, $G_i$, ensures that the output will settle at the correct value. Unlike the proportional component, integral control depends both on previous values and the timespan over which these values were samples, which means that timing of the sampling actions are important in order to get correct values[55].

### Differential Gain

The differential control term is the last value of the position minus the current value of the position. This acts as a rough estimate of the movement velocity, and can be used to predict where the next position will be[55, 33]. The differential component is the most difficult to fully comprehend, and to tune correctly. The differential gain $G_d$ is usually much higher than the other gains, so noise on the signal is amplified to a much greater effect. As the differential term is directly tied to the positional change, high frequency signals, which often is/contains noise, is amplified. For these reasons the sampling rate must be kept stable in order to avoid creating extra noise on the signal. Filtering of the signal is possible in order to reduce noise, but it will decrease the usefulness of the differential term[55].

## 2.5 Rotary Encoders

When using electrical motors it is often important to know the current rotation and motion of the motor shaft. The most common way this is done is with encoders, which are electro-mechanical devices that transfer the position or motion of a motor to a readable output. This information can in turn be used to calculate the distance traveled, speed and acceleration of the motors. There are multiple types of encoders, but the two main types are incremental and absolute encoders. Absolute encoders note the current position of the shaft, while the incremental encoders count encoder *ticks* to gather information about the motion of the shaft. Equation (2.3) shows how to calculate distance traveled $D_T$ by a wheel connected to a motor, based on the number of ticks from an incremental encoder. The circumference of the wheel with radius $r_W$ is simply divided by the resolution of the encoder $R_E$, and multiplied by the number of ticks $T_E$ .

$$D_T = \frac{(2 * \pi * r_W)}{R_E} * T_E \tag{2.3}$$

#### Hall Effect Sensors

One type of encoders typically used in DC motors are magnetic Hall Effect sensors. These sensor use the Hall Effect to determine the angle of the motor shaft, by measuring a voltage across the sensor [30]. This voltage is called the Hall Voltage, denominated $V_H$, and is caused by forces exerted on the electrons as they traverse the magnetic field of a semiconductor inside the sensor, which is called the Hall element[41]. It can be calculated by the equation shown in Equation (2.4). As the conductor carrying current $I$ changes its rotation relative to the magnetic field $B$, $V_H$ changes proportionally. When $V_H$ is mapped to a 360°rotation, reading $V_H$ is enough to determine the angular position of the conductor, which can then be used as encoder feedback.

$$V_H \propto I \times B \tag{2.4}$$

## 2.6 Lithium Polymer Batteries

All forms of mobile electronic component needs a mobile power source in order to function. The most common way is to use batteries. Lithium Polymer (LiPo) batteries are the among the most common types of batteries to use for these types of mobile applications, and is commonly used in equipment like laptops, radio controlled cars and planes, electric and hybrid vehicles and more[51]. They have become popular because of their excellent power-to-weight ratio, large capacity, high discharge rates and no memory effect[58].

### Drawbacks

Lipo batteries have few major drawbacks, as they are relatively expensive, have a low amount of lifetime charge cycles, and requires extensive care in order to keep functioning properly[58]. Lipo cells can be damaged or ruined both by overcharging and undercharging, and require protectional circuits to prevent overcharge or undercharge of its cells. Additionally cells may catch fire or explode if punctured or overcharged, and because of this lipo batteries should be kept in fire-safe bags, especially while charging. Even though these drawbacks are quite severe, they are still often outweighed by the advantages. Lipo batteries are commonly used in mobile robotics, mainly because of their high energy density, which helps the robots maintain a long lifecycle, as the weight of the battery often is a significant portion of the robot's total weight[57].

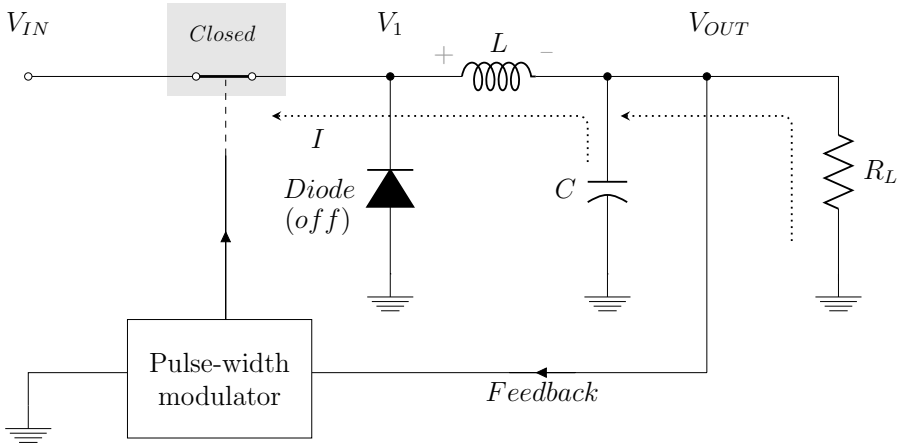
### Lipo Cells

Lipo batteris are built up of individual lipo cells. The nominal voltage of lipo cells is commonly referred to as 3.7V, but in reality it can vary from 3.7V to 3.3V vary depending on factors like cell structure, exact chemical composition and temperature[47]. For the purposes of this thesis the nominal voltage will be considered to be 3.7V. Due to the nature of Li-Po batteries and their natural reduction in voltage levels during discharge, a voltage regulator is necessary if a stable output voltage is desired.

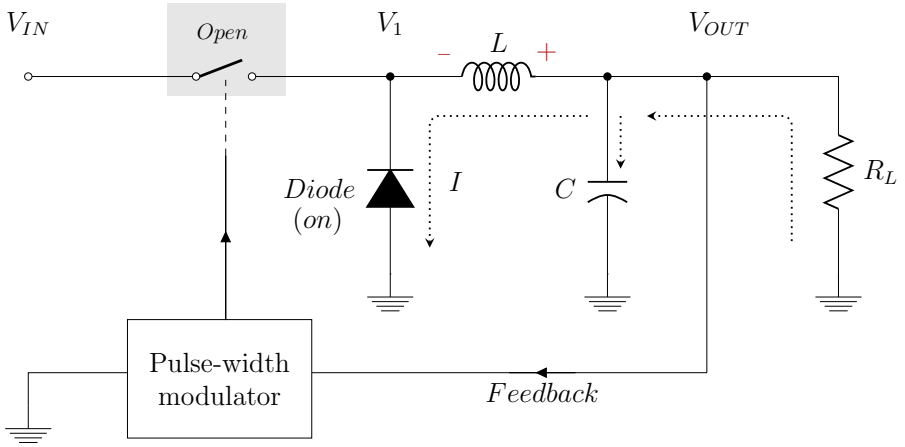
## 2.7 Switching DC-DC Regulators

Switching regulators are based on the concept of opening and closing a transistor in order to transform unregulated DC voltage into a series of pulses[41]. The output dc voltage is the average value of the pulses, and can be controlled by regulating switching frequency (i.e. pulse width) in relation to changes in input voltage levels. The adjustments to the switching frequency are accomplished by sending feedback from the output to a pulse-width modulator. The most important feature of switching regulators compared to linear regulators are their efficiency, which typically range from 70%-90% across a range of input voltages[29]. Linear regulators on the other hand only have a high degree of efficiency when the difference between input and output voltage is low, and will otherwise dissipate a lot of power trough heat generation, as that is their primary way of reducing voltage. A high grade of efficiency during voltage conversion reduces the amount of energy wasted, thus improveing battery life and reducing heat generation.

**Switching**




(a) Switch closed, pulse high




(b) Switch open, pulse low

*Figure 2.7: States of a switching voltage converter.*

s

Figure 2.7 illustrates the switching performed by the regulator. While the transistor switch is closed as in Figure 2.7a, an induced voltage is developed by across inductor $L$ and the load connected to $V_{OUT}$, that opposes a change in current. When the transistor then opens as illustrated in Figure 2.7b, the inductor develops a voltage in the opposite direction. At the same time the diode acts as a closed switch, providing a path for the current. With the help of the capacitor discharging, this keeps the load current stable[41].

## 2.8 Optical Switch

A common way to check for the presence of an object is optical switches. The optical sensors works by sending out light, usually infrared, from a light transmitter. As long as the path of the light is not obstructed, by for instance putting materials that either reflect or absorb a high amount of the ir-light in the slot, the light is detected by the photoelectric receiver on the opposite side of the slot. As long as the sensor senses light the output is pulled either high or low depending on the configuration of the switch, and when light is obstructed the output is inverted[45]. These types of sensor can for example be used in smoke detector alarms, that enable the alarm if smoke covers the gap between the emitter and receiver. They can also be used for implementing enabling switches, where a start or stop signal is sent to a system when a key, pin or similar object is removed or pulled out from its slot.

## 2.9 Locomotion

Selection of a locomotion configuration has a significant impact on the design of an autonomous mobile robot, and this decision is often not given the appropriate weight when planning the setup of a project[4]. This configuration affects the performance of the robot in several key areas, including speed and acceleration, platform stability, motion smoothness and weight limitations. There is no specific rule in the Eurobot rules that states that the robots *have* to use wheels for propulsion. However, the layout of the board and setup of the challenges strongly favor a wheel-based configuration, as it requires extremely accurate positioning over short distances on flat, smooth surfaces, as well as frequent direction-changes.

### 2.9.1 Wheel types

There are several different wheel types a robot can use, the most common ones are displayed in Figure 2.8. An example of a normal, flat wheel commonly used as driving wheels on robots is shown in Figure 2.8a.

#### Caster Wheels

Figure 2.8b shows a type of caster wheels. These types of wheels are often use on things like mobile chairs and shopping carts, and they allow the object to travel in all directions. The movement of a caster wheel can be split into two parts; the translational movement of the wheel itself, and the rotational movement of the joint holding it. As long as the angle between wheel and the direction of motion is low and the rotational movement is small, it functions like a regular wheel and there are no undesired effects movement. Problems arise when the angle between the wheel and robot heading is high, and the rotational movement required by the wheel is high. As the wheel itself is off-center to the attachment of the joint, the

(a) Normal side wheel

(b) Caster wheel[14]

(c) Ball wheel

(d) Mecanum wheel[13]

*Figure 2.8: Common robot wheel types.*

friction between the wheel and surface is opposing the desired motion. As the joint starts rotating, the resistance from the wheel starts applying a lateral force to the back end of the vehicle, which often lead to inaccuracies and erratic movements. For these reasons caster wheel are rarely use on high-performance robots.

#### Ball Wheel

Ball wheels are a type of spherical, omni-directional wheels. They consist of a spherical ball mounted inside an enclosure, with wheel bearings separating them from the enclosure. The advantage of ball wheels over casters is that they do not have the extra joint, that may affect movement. The drawbacks is that ball wheels often are less robust and can handle less weight, as the weight of the vehicle will be pressing directly down on the wheel bearings.

#### Mecanum Wheels

Figure 2.8d show a Mecanum wheel, which is a type of omni-directional wheels. It was developed to enable a vehicle to move in any direction without changing its heading[48]. The wheels allow for flexible movement, as the robot does not need to turn to go sideways. The drawbacks of Mecanum and other types of similar wheels is that they are expensive, and encoder measurements are problematic, especially when moving sideways.

## 2.10 Wheel Configurations

### 2.10.1 Skid Steering

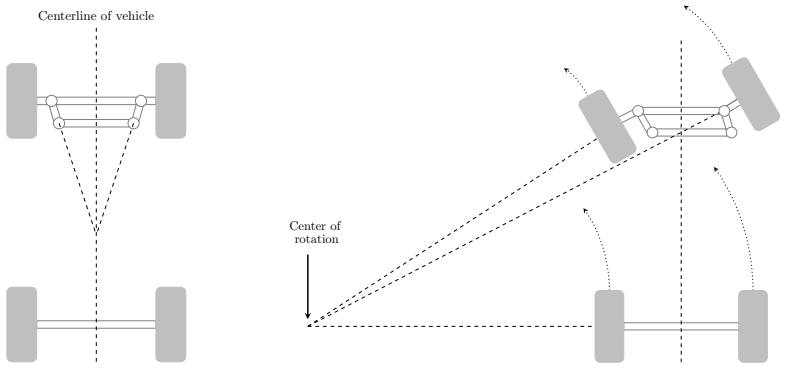
Skid steering is one of the simplest steering mechanisms to implement, as it involves no steering of the wheel axles. It consists of two (or more) pairs of wheels connected on opposing sides of the robot. Heading changes are performed by spinning the wheels on each side of the robot in opposite directions relative to each other, at which point the robot will skid while rotating. The resulting motion is controlled by lateral displacement of the chassis rather than explicitly steering the wheels, and the turning angle/radius is decided by the difference in velocities and the distance between the wheel sides. Skid steering is often used for mobile robots that need to operate in rugged terrain, because it easily can make point-turns and has symmetrical maneuverability forwards and backwards. The main drawback of skid steering is that excessive torque is required to get the wheels to spin (skidding), which in turn consumes excessive amounts of power [4]. Other drawbacks include problematic odometry, as encoder readouts are mostly useless when the wheels are skidding, maneuverability performance being heavily dependent on surface traction, and the complex gear systems required to power all wheels.

#### Tank Tracks

This configuration can also be implemented with tracks instead of wheels, which increases traction because of the increased contact surface, which in turn improves maneuverability in rough terrain. This will however also enhance all of the previously mentioned drawbacks.

### 2.10.2 Ackermann Steering

Ackermann steering [2] is the most common steering scheme for modern personal and commercial vehicles such as cars. It is a form of coordinated steering, and utilizes a mechanical coupling of the front wheels that compensates for the fact that the outer wheel(s) in a turn will have a larger turn radius than the inner wheel. In this configuration a direction change is handled by two (or all) wheels turning at the same time, but not at exactly the same angles. Ackermann steering is designed to improve maneuverability and controllability for vehicles traveling at high speeds, and specifically for vehicles that has to do cornering maneuvers while maintaining a high speed. One drawback is poor maneuverability at low speeds. In an Ackermann configuration the front wheels has a limit of the angle they can be turned, which is less than 90 degrees. This means that a robot based on Ackermann steering will be unable to do a point turn, and to move directly sideways it would need to make several movements forwards and backwards (like cars when parallel parking).




(a) Driving straight.

(b) While turning the back wheels follow an arc set by the front wheels.

*Figure 2.9: Illustration of the Ackermann steering principle.*

### 2.10.3 Tricycle Drive

Tricycle drive is common configuration for three-wheeled robots. It consists of two rear drive wheels, which can be powered either by a common motor trough a gear system, or independently. The front wheel provides the steering by turning of the wheel axle. The tricycle configuration has multiple inherent drawbacks, which include not being able to do an effective point-turn and being inherently unstable. As an

example the Reliant Robin is one of very few cars to use this configuration, and is widely acknowledged as the single most unstable car of all time. An alternative three-wheeled configuration is to have the two drive-wheels at the front, and let the back wheel control the heading, similarly to a rudder.

### 2.10.4 Differential Drive

Differential drive is very similar to skid steering, the difference is that instead of two pairs of driving wheels, there is only one. These two wheels are fixed on a common axis, and heading changes is performed by, as the name suggests, moving the wheels with different speeds and/or directions. To maintain the robots balance a number of support wheels are added, usually either caster or ball wheels. These can be positioned in a variety of configurations, where the most common is to have centralized ball wheels in the front and back. If the balance of the robot allows it, a configuration with only one support wheel can be used.

#### Advantages

The advantage of differential drive compared to skid steering is that it increases the efficiency of the propulsion by minimizing lateral skidding [8]. This also makes odometry easier and more precise as the wheels are not spinning (as much) on the surface. One caveat with differential drive is that rotational movement will be around the center of the common drive wheel axis instead of around the center of the robot, which may be slightly inconvenient for path planning.

## 2.11 Rapid Prototyping

Rapid prototyping is the process of using techniques like computer aided design (CAD) and 3D printing to create prototypes, and eventually finished products. The name "rapid prototyping" refers to the fact that the process of adapting existing designs for new iterations is a fairly simple and fast process. This has multiple advantages, chief of which is the ability to model components to fit the exact needs of the project, instead of designing a project around the limitations of available components. Using traditional manufacturing, changing a part can set projects back for long periods of time while waiting for the new parts to be manufactured and shipped. An added benefit of rapid prototyping is that the material costs for creating new parts is relatively low. Once an, admittedly significant, one-time investment in machinery is done, it is possible to produce a virtually unlimited amount of parts relatively cheaply. It also allows for easy replacement of parts if something gets damaged. For these reasons rapid prototyping, and 3D-printing in particular, have been use liberally throughout the project.

[0] An example of the challenges of this configuration: https://youtu.be/QQh56geU0X8

### 2.11.1 Fused Deposition Modeling

Fused Deposition Modeling (FDM) is a type of additive manufacturing where plastic heated to a semi-molten state, and fed trough an extrusion head with high pressure[25]. As illustrated in Figure 2.10, this produces a continuous string of plastic, which is used to form layers of plastic. The layers can be as thin as 0.13 mm. As the plastic cools, the layers fuse together to form a cohesive part. Support material is used to allow the printers to create overhanging geometry.

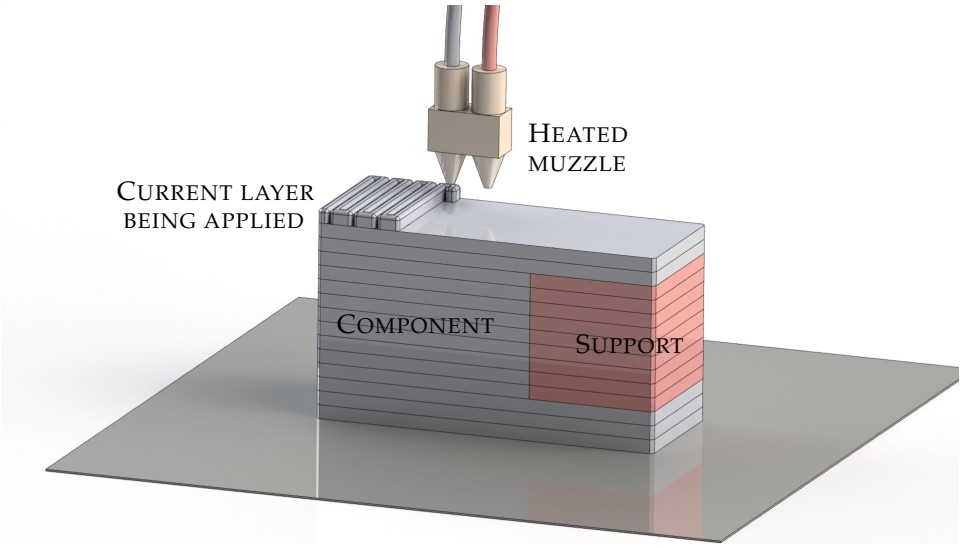



*Figure 2.10: Illustration showing the basics of Fused Deposition Modeling. The thickness of layers are exaggerated greatly to showcase the process.*

The parts produced by FDM printing have a relatively poor surface finish as the fusing of the plastic layers leave rough edges. They are however robust and durable, and can be manufactured with a high degree of precision[26]. These traits are optimal for use in robotics appliances, and is the reasons why this technique have been selected for use in this project.

#### Fortus 250mc

The 3d printer used to produce all the custom parts on this project is the Fortus 250mc from Stratasys, which is based on the aforementioned FDM technology[31]. The Fortus 250mc has a z-axis resolution between 0.178 and 0.33 mm, depending on what type of material is used.

### 2.11.2 Solidworks

Solidworks have been used for all the CAD processes on the project, as it is one of the most widely used and recognized software for these tasks, and is also licensed by the University of Oslo. The workflow in Solidworks is to extrude 2d sketches in order to create 3D models. A number of pictures exported from Solidworks will also be shown during this thesis, some of


*Figure 2.11: The Fortus 250mc FDM 3D printer*

which are directly exported while others are post-processed by rendering software[16].

## 2.12 IR-based Triangulation System

The global navigation system used in this project is designed by Andre Kramer Orten, and is described in his thesis[46]. The concept is to use three active beacons located at beacon support platforms around the area the robot will move, as shown in Figure 2.12 (these platforms will hbe discussed further in Section 3.4). These beacons emit coded IR-signals, so they can be differentiated easily. A rotating beacon tower positioned on top of the robot reads these IR-signals with a IR-receiver. By noting at what angles the tower can read the different signals, the angular distance between the towers can be approximated.

## 2.13 PC/104

PC/104 is a standard for a small form factor embedded system with detachable modules[15]. A system like this was evaluated for use as the main processing unit for the robot, as it would provide a small form factor pc with most of the features of regular computers. Another advantages is that modules can be added and removed as necessary, which means that only the modules required for this project have to be acquired. The drawbacks is that all parts of the system would need to be purchased.

## 2.14 Operating Systems

The selection of an operating system (OS) is an important choice because it both limits the selection of tools and software for the project, and directly

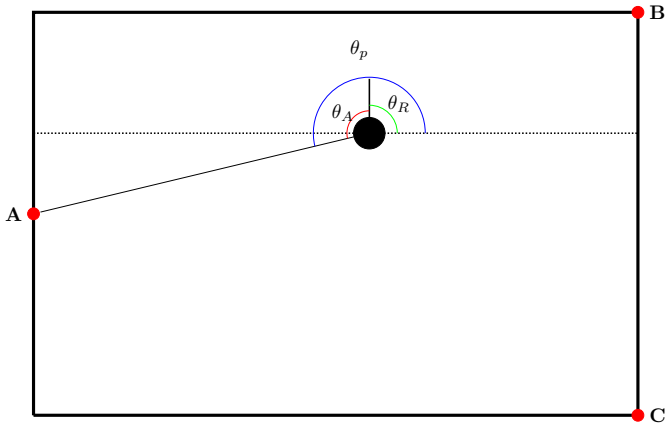



*Figure 2.12: Figure from NAVSYS[46], illustrating the concept of the beacon triangulation system.*

affects the performance and responsiveness of the entire system.

### 2.14.1 GNU/Linux

Linux is a free and open source Unix-compatible kernel, that together with the GNU system forms the basis for the operating system group known as *GNU/Linux*[52, 54]. Operating systems built on GNU/Linux are known as *distributions*. For the purposes of this thesis, the distinctions between the different distributions are negligible, and the name *Linux* will be used to described both the kernel and the entire OS for simplicity.

#### Advantages

An advantage of using Linux compared to a closed-source system is that as it is possible to modify the entire system, and even remove, replace or customize components of the kernel itself. Other advantages include being able to run on almost any piece of hardware, from desktop pcs and supercomputers to embedded systems and mobile phones. Additionally Linux has a enormous selection of programming languages, tools and software libraries.

#### Completely Fair Scheduler

The default task scheduler of the Linux kernel is the Completely Fair Scheduler (CFS), and was introduced in kernel version 2.6.23 in 2007. The main idea behind CFS is to maintain fairness in the CPU-time between processes by giving each process a fair amount of run-time on the CPU[34]. This is accomplished by assigning a *virtual runtime* to each process, and then prioritizing giving CPU-time to the processes with lower virtual runtime. Instead of using hard priority levels like most other schedulers,

processes are instead given factors of decay. The factor is higher for low priority tasks, which means that their CPU-usage will decline faster. This scheduler is optimized for providing a good desktop experience, and the lack of hard priority makes the scheduler unfit for use in real-time systems.

#### Kernels

One of the advantages of using a completely open OS such as Linux is that all parts of the systems can be tweaked according to personal needs and preferences. Even the kernel, the core of the system, can be customized or even completely replaced with a different version. It is usually beneficial to use the newest stable kernel, as newer kernel versions have additional features, bug-fixes and possibly higher performances, however older kernels may be preferred for stability reasons, as they have undergone more thorough testing.

#### The PREEMPT_RT Patch

A number of projects have attempted to increase Linux s viability as a platform for real-time computation. The main effort is currently focused around a patch known as `PREEMPT_RT`[49]. The goal of this patch is to convert Linux into a completely preemptible kernel, which means that the execution of a process always can be interrupted if a higher priority process is introduced into the system. This is necessary to create a deterministic, real-time environment, in which higher priority tasks reliably will finish first. The `PREEMPT_RT` patch attempts to solve this by implementing a custom scheduler that avoids unnecessary delaying steps in the sequence of operations required for task switching [10].

### 2.14.2 VxWorks

Another alternative for the OS could be VxWorks. Unlike Linux, VxWorks is built specifically to be a real-time operating system (RTOS). It is mainly designed for embedded systems but can be run on regular computers, although not all hardware and components are supported. It has been used successfully in large commercial applications, most notably by NASA on their Mars exploration rovers[39]. The OS and related tools are proprietary licensed by the American company Wind River, and must be purchased in order to use them.

#### Drawbacks

Even though it is a great fit for a mobile robot due to its real-time nature, it may be difficult to use due to limited software and toolsets, as well as being unfamiliar to everyone in the project. The licensing and cost of purchasing is also an issue.

### 2.14.3 Xenomai

Xenomai is a project using attemtpting to ease migration from traditional RTOS to Linux, by using the Linux kernel as a base for a real-time operating system[22]. Real-time performance is achieved in one of two ways; the first alternative is to run a co-kernel called *Cobalt* besides the Linux kernel, extending it to better solve time-critical requests, such as interrupt handling and real-time thread scheduling. This core has elevated privileges over the native kernels activities. The other alternative is to rely on the real-time capabilities of the native Linux kernel, and use it to form the *Mercury* core. It uses some features implemented in the PREEMPT_RT patch mentioned in Section 2.14.1. The advantage of selecting Xenomai is that it can deliver real-time capabilities outside of what the Linux kernel can deliver natively, while not requiring expensive licenses and tools like VxWorks, as all of Xenomais code is licensed under the GPL or LGPL licenses.

#### Drawbacks

The drawback is that for applications that do not have hard real-time requirements, Xenomai is not significantly more performant than Linux with a stock kernel, and the added extensions can cause bugs not present in the stock kernels[7]. The community surrounding Xenomai is also small compared to that of Linux, so getting assistance if problems are encountered may prove difficult.

## 2.15 Interprocess Communication

To enable multiple processes or programs to work together it is necessary to implement some form of communication between them. This is referred to as interprocess communication (IPC). There are a multitude of different ways to implement this, depending on the type of programming language, operating system and hardware one is using. As our project is organized in order to allow all members to use any type of technology they wish to, only the types of communication available for most modern platforms and programming languages are of interest. Additionally, a form of communication can be used across a network as well as locally is preferable, as it would make collaborating on development easier.

#### Direct Memory Communication

With the last point in mind, techniques like shared memory or memory mapping are not good choices as they are mainly intended to work locally, and are not well suited for working with multiple processes across networks. Distributed shared memory systems do exist[44], but they require manual setup of segment allocation, synchronization and protection against simultaneous data access, such as semaphores, as processes otherwise could cause errors by altering the wrong data. Finally,

it could cause issues when communicating across different platforms and languages as they may handle pointers and memory differently.

### 2.15.1 Message Queues

Message queues are a form of asynchronous communication protocols, and solve many of the problems with distributed memory systems by functioning as an abstraction layer across a variety of communication technologies[43]. The protocol abstracts away the need for senders and receivers of data to interact with one another, and simultaneously removes the need for synchronization or data access protection. Another advantage of message queues is that they can be implemented on top of different underlying communication technologies, and messages can easily be sent to other threads or processes both locally and across networks.

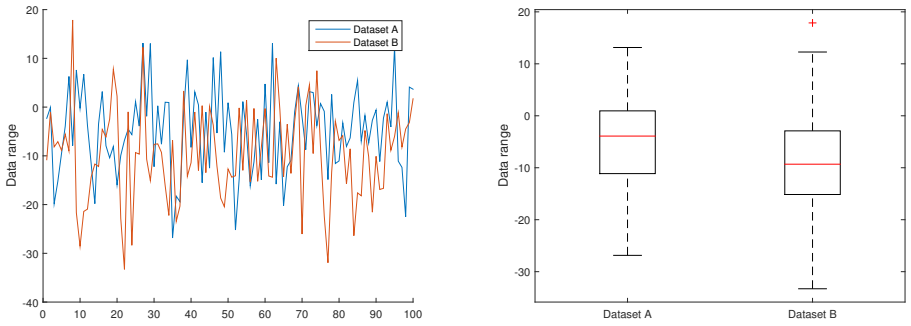
There are a vast number of message queue implementations available for most conventional operating systems. They differ from each by the nature of their implementations, and what features they provide. For instance some messaging queues restrict communication only to other directly related processes, others allow for communications to unrelated processes on the same file system, and some also allow for communication across networks. Other distinguishing features are whether or not multiple processes are permitted to use the same connection, and if the communication is synchronized by the protocol.

### 2.15.2 ZeroMQ

As mentioned there are numerous message queue implementations, both proprietary and open source, and selecting the one best suited to an application is no simple task. The most important criteria for our communication system were that it should have implementations in all conventionally used languages and operating systems, have a simple and easy to use application programming interface (API), be highly performant in regards to latency, and most importantly be reliable and robust. Based on these specifications, ZeroMQ seemed to be one of the implementation best suited for our needs[27]. In addition to having implementations on most modern architectures (including a port to VxWorks) and for almost all conventionally used programming languages, it is highly performant due to its brokerless, asynchronous architecture, shown among others in a comparison test performed between message queue implementations by CERN[17]. ZeroMQ allows for seamless transitions between communications on local system between processes and network communication, which simplifies the process of testing the relation between the various programs used in the project, as they do not need to run on the same hardware.

## 2.16 Box Plots

Some types of data are difficult to represent effectively with traditional plots or graphs. One tool for dealing with this problem is a type of graphs known as box plots[42]. In box plots each set of data is represented as a box, where a central line marks the median of the data, and the top and bottom edges of the box marks a fixed percentile, usually 25 and 75. Whiskers on the top and bottom of the box mark areas with more extreme data points that is still not considered outliers, and outliers are plotted individually. This is useful when each dataset contains a large amount of data, for instance when each dataset include multiple runs of the same experiment. The greatest advantage over more traditional plots is that box plots allows for a better understand of how the values are spaced out in the different datasets. Figure 2.13 illustrates the concept for differentiating between two arbitrary datasets; the line plot is hard to read and gather information from, while the box plot clearly displays median, total range of values and the spread of values, and makes the task of comparing the datasets easier.




(a) Line plot

(b) Box plot

*Figure 2.13: Comparison between line plot and box plot representation for two arbitrarily generated datasets.*

# Chapter 3

# Eurobot 2015

This chapter explains the setting, tasks and rules of this years Eurobot competition[20]. The first part of the chapter will be spent discussing the general setup of the game board and constraints of the robot, before exploring the points system and various tasks.

## 3.1 Location and Theme

The 2015 Eurobot competition is held in *Yverdon-les-Bains* in Switzerland, and the theme for this years competition is "Robomovies". All of the objectives are inspired by real tasks vaguely related to movies, cinemas or movie production, such as putting up spotlights on a movie set, rolling out a red carpet or gathering popcorn baskets. This chapter will discuss the specifics of the rules, tasks, and everything else regarding the competition.

## 3.2 Robot Constraints

The Eurobot rules specify detailed constraints and obligatory features that teams have to adhere to in order to qualify for the competition. Table 3.1 shows the most important ones. There are additional rules that focus on fair play, which state that robots may not, for instance, use colors and shapes reminiscent of the game board in attempts to confuse the opponents.

#### Stop system

The rules specify that all robots have a easily reachable emergency stop button, to allow referees to stop a robot at any time. The button must render the robot completely motionless, meaning that no motors should be either actively breaking or energized.

#### Starting device

Additionally the robots have to be outfitted with a starting device. This device must be activated by the pulling of a cord, which then must be completely detached from the robot. The secondary robot can either have

| Constraint | Main Robot | Secondary Robot |
|---|---|---|
| Start configuration circumference | $\leq 1200mm$ | $\leq 700mm$ |
| Deployed circumference | $\leq 1500mm$ | $\leq 900mm$ |
| Height | $\leq 350mm$ | $\leq 350mm$ |
| Max. internal voltage | 48V[1] | |
| Min. battery capacity | 3 matches | |
| Starting cord | ✓ | |
| Emergency stop button | ✓ | |
| Automatic shutdown | ✓ | |
| Obstacle avoidance system | ✓ | |
| Beacon Support | ✓ [2] | |

*Table 3.1: Overview of obligatory features. These components must be included on all robots, even the smaller secondary ones, in order to pass the homologation phase.*

its own starting device, or be started indirectly by the main robot when its starting device is activated.

## 3.3 Game Area

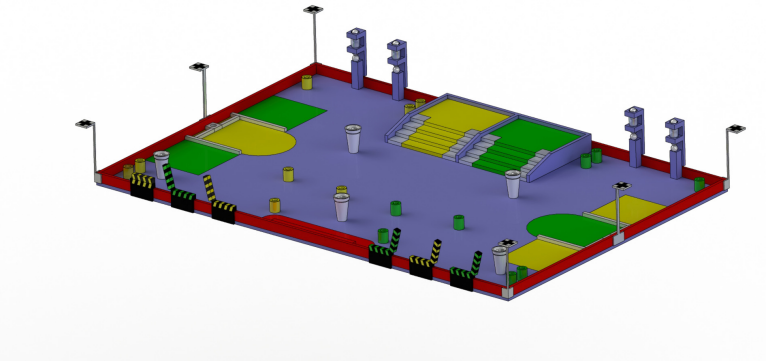


*Figure 3.1: The Eurobot 2015 game area. The image is rendered from the Solidworks model used in Chapter 4.*

[1]Higher voltages are allowed inside unaltered commercial devices.

[2]Not strictly required, but not providing a beacon support to a team requesting it results in an automatic forfeit.

The game area is a three by two meter rectangle with a wall along the sides. The surface is painted with different colors to distinguish various areas. Some of these areas are used as zones in which certain tasks have to be execute in order to score points, while others are painted purely as an aid for navigational purposes. Located along the opposing short ends of the board are the teams starting areas, which in this competition is called "cinemas". The cinemas are used both as starting areas for the robots and as a storage area used for two of the tasks. Black lines are painted on the board, leading from the starting areas to the clapper-boards and platform areas. These lines can be used for navigation.

## 3.4 Beacon Support Platforms

A total of 6 support platforms for laser/ir beacons are placed above the short sides of the board, one in each corner and one in the middle. Each team has access to a triangle consisting of 3 of these, and can use them for triangulating a position. An additional beacon can be place on the opponent teams robot. These are used for the IR-based positioning system implemented in NAVSYS[46]. Each robot is required to provide a platform for the opponent team to potentially place a beacon on.

## 3.5 Point System

Table 3.2 contains an overview of all the tasks and the points they reward. The *seprate* tasks include team-specific playing elements, meaning that both teams could potentially get full scores in these categories. The only task that has shared playing elements are the *Popcorn* task, which means this is the only task where a team can directly devoid the opposing team points. The amount of points scored in a match will determine the winner of the match, and also their standing on the result table. The remainder of this chapter will discuss each of the tasks, and the game elements related to them, in detail.

## 3.6 Spotlight Task

Spotlights are the most complex of the tasks, as it involves handling multiple game pieces as well as pinpoint navigation. A total of sixteen cylindrically shaped objects called *stands*, shown in Figure 3.2), are positioned at predesignated positions on the game board. There are eight stands for each team, painted in their respective team colors. To score points the stands must be placed in one of three *building areas*; each team has a *protected* building area in their starting areas, and there is an unprotected shared building area. Protected areas means that points are remove by the opposing teams robots, compensation and penalty points are added at the end of the match.

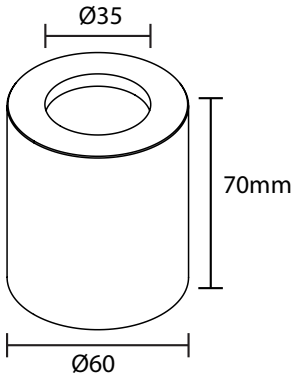



*Figure 3.2: Size of the Stands*

| Task | Type | Number | Points Each | Bonus | Total |
|---|---|---|---|---|---|
| Spotlights | Separate | 8 | 2 | 3[1] | 40 |
| Clapperboards | Separate | 3 | 5 | - | 15 |
| Stair climbing | Separate | 1 | 15 | - | 15 |
| Red carpet | Separate | 8 | 4 | 2 | 24 |
| Popcorn | Shared | 40 | 1 | - | 40 |
| Total | - | - | - | - | 114 |

*Table 3.2: Total number of points achievable for a team.*

### Light bulbs

In addition to the stands there are four tennis balls, called *light bulbs*, two for each team. One of them are located in each teams starting area, and can be pre-loaded into one of the robots. The other two are located in the shared building area, and can be picked up.

### Stacks

Stacks are created by putting stands on top of each other, and adding a light bulb on top. The point system for stacks are slightly convoluted, especially in the English version of the rules which is translated from French. Any number of stands can be place in each of the building areas, in stacks or otherwise, but only one stack in each of the areas will count as a *spotlight*, and provide bonus points. Each stand provides two points, and stands in valid spotlights provide an extra three bonus points. The setup of this

[1] Valid spotlights provide 3 bounus points *per stand*.

task allow for different types of tactics; either attempting to score a high amount of bonus points by stacking the spotlights, or the safer approach of simply collecting all the stands and not focus on the bonus points. The first tactic allows for a higher amount of points, but stacking the towers require extreme precision and caution, and is riskier as the slightest misstep may lead to scoring no points from this task at all.

## 3.7 Popcorn Task

A total of 5 popcorn cups are positioned on the board, each of them approximately halfway full, containing 4 popcorns. To fill them completely, additional popcorns can be gathered from dispensers along one of the long sides of the board. There is a total of 4 dispensers, each containing 5 popcorns. Popcorns in cups located in the correct areas at the end of the match award 1 point each. One peculiar thing about the popcorns and popcorn cups is that the are the only playing elements that is common for both teams, which means points can be "stolen" from the opposing team at any time. The only thing that matters when counting points regarding the popcorns is the position of the cups at the end of the match.

#### Initial Popcorn Placement

The second task consists of collecting *Popcorn Cups* filled with *Popcorns*. The playing elements shown in Figure 3.3. A total of five cups are positioned at predefined positions on the game board, and unlike the stands these are shared between the teams. Each cup initially contains four popcorns, and more can be added by retrieving them at *Popcorn Dispensers* located along the top side of the board. There are four dispensers in total, each containing five popcorns.

#### Valid areas

Each popcorn in a cup inside a *valid area* at the end of the match provides one point. Valid areas for storing popcorn cups are inside the starting area, and in the squares on each side of the opponent teams starting area, painted in our (change this) team color. Unlike stands (and stacks), popcorn cups are not protected even if it is located inside a valid area. This means that it is allowed to topple, move, or even "steal" cups. Spending the last parts of the match attempting to seek out and remove the opponents popcorn cups might be a valid strategy, although it would have to be done with surgical precision in order to avoid penalties for interfering with stands (as explained in Section 3.6).

## 3.8 Clapper-boards Task

The third task is inspired by movie sets, and involves closing the top part, the *clapper* of the movie *clapper-boards* located along the bottom side of the

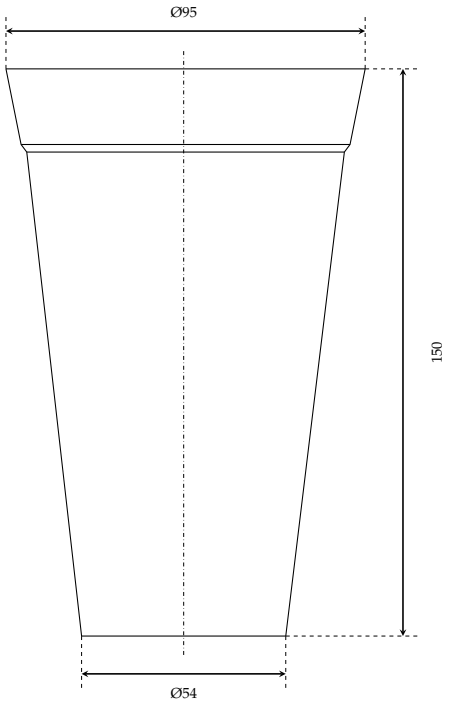



*Figure 3.3: Size of popcorn cups*

board. Illustrations of the clapper-boards are presented in Figure 3.4. There are a total of six clappers, three for each team, painted in the respective team colors. Two of the clappers belonging to each team are located on same side of the map as their starting area, while the last on is on the opposite side, in between the opponents clappers.

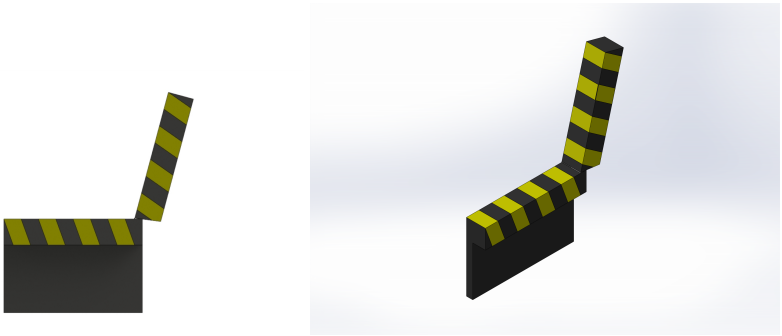


*Figure 3.4: Clapper-boards*

**Points**

If a clapper is closed at the end of the match, it gives five points to the team it belongs to. This means that closing a clapper belonging to the other team provides them with five points. There are however no further punishment distributed for closing the wrong clappers.

## 3.9 Staircase Task

The most distinguishing features of this years game board compared to earlier competitions are the staircases, displayed in Figure 3.5. This competition is the first time a stair of this type is included, and it poses an interesting challenge. Arguably the easiest way for robots to climb stairs, besides legs or other peripherals, is tank-style tracks. Tracks have a number of drawbacks, as mentioned in Section 2.10.1. These drawbacks must be weighed against the advantage of having an easy way of climbing the stairs. An alternative solution may be to let the second, smaller robot perform the stair-climbing.

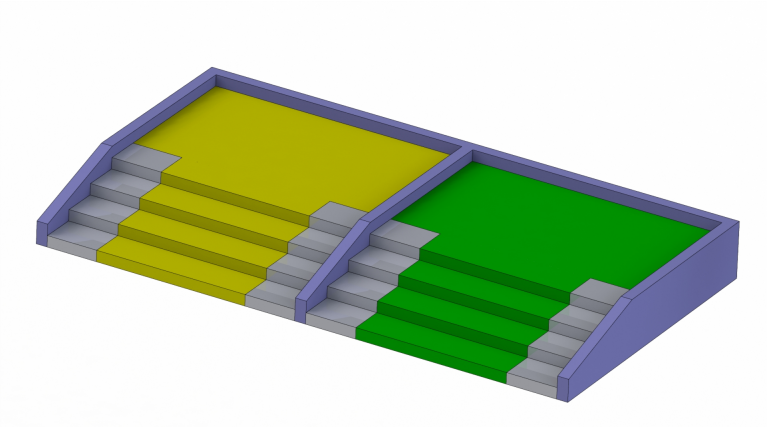


*Figure 3.5: The staircases, one for each team. The uncolored areas are called walkways, used for the Red Carpet task.*

### Points

If a robot is located on the top of the correct staircase at the end of the match, the team is awarded fifteen points. A robot being *located* at the top of the stairs is defined as the robot not being in contact with anything else than the top step of the staircase, and the walls encasing it.

## 3.10 Red Carpet Task

The last task is also related to the staircases in Section 3.9. Each of the stairs have unpainted areas along both sides, called *walkways* (see Figure 3.5). The goal of the task is to cover these walkways with the *red carpet*, which is symbolized two red pieces of cloth that can be loaded onto a robot during the match preparation period, in either the main or secondary robot. There are two obvious strategies for solving this task; one is to let the robot drop the carpets while climbing the stairs, and the other is to throw or lay them from a position at the end of the stairs.

**Points**

Two points are awarded for each step completely or partially covered by a carpet, with an additional four bonus points awarded for each walkway where all the steps are at least partially covered. There is no definition of what a constitutes a *partially* covered step in the rules, but it is assumed that the carpet touching a step will be enough to qualify that step for points.

# Chapter 4

# Implementation

This chapter will discuss how every part of the robot has been implemented, as well as what components have been considered and ultimately selected. As this thesis is heavily focused on the design and implementation of the physical and electrical parts the robot, this section will contain most of the work performed throughout the thesis.

## 4.1 General Strategy

The budget restrictions and team philosophy discussed in Section 1.3 has had a significant impact on the approach taken in the implementation phase, and on many of the decisions that have been made. Due to the short time frame and low number of team members on the project, ease of implementation and time-efficiency have been an essential aspect of selecting the right solutions. This has been carefully balanced against the price of each component and its impact on the project spending.

#### Custom-made vs Off-the-shelves

Using off-the-shelves components can save time and work, and it often trivializes implementation as the components can be put straight to use in the project. The downside is that they can be expensive, and it might be difficult to find components that exactly match the specific criteria the project requires. This must be weighed against the budget impact of the parts, as well as the suitability of the pre-fabricated components compared to custom solutions. A huge advantage for this project is access to an advanced 3D printer, the Fortus 250mc mentioned in Section 2.11.1. This allows for printing of custom-made plastic parts, which is extremely helpful for creating specialized structural parts. With access to a printer these parts can be created quickly and relatively cheap. 3d-modeling and printing of parts can however be time-consuming.

## 4.2 Game Board

To be able to perform realistic testing of the various tasks, a replica of the Eurobot game board, illustrated in Figure 3.1, had to be acquired. To keep costs to a minimum the board was built manually from the ground up with basic materials. The Eurobot regulations[20] do not specify what type of material the base of the board is supposed to be made of, and administrators were unable to specify this when inquired directly. This seems like a potentially huge oversight, as it may affect the traction of wheels and game pieces, and could have a significant impact on robots movement and general operation.

### Our Board

We decided to use chipboard plates as the material for the board, as they are cheap, light, easy to modify, and reasonably smooth when painted. The plates were manually cut into the correct sizes and spray-painted to resemble the actual Eurobot game board to the best of our ability. Leftover pieces of the chipboard were used to make the staircases. The surface of the board ended up being reasonably close to the official boards, and although the traction was slightly lower than on the boards used in the actual competition, it did not have a significant effect on our performance. The resulting game board is shown in Figure 4.1.

(a) Everything set up (b) In use during IR beacon testing

*Figure 4.1: The finished Eurobot game board.*

### Game Elements

The additional components, each described in detail in Chapter 3, were built as they became necessary. Stands were first 3D printed in order to have perfectly sized elements to test and calibrate the robots gripping functionality. Additional stands were later made out of wood to more accurately depict the weight of the official stands[20]. Simple plastic cups were used as popcorn cups as they had almost exactly the same shape as the cups used in the competition. Popcorn dispensers were never built, as that task ended up not being performed (see Section 3.7). Only a simple prototype were made of the movie clapperboards, and this task were

considered so simple that no further time were spent creating additional copies.

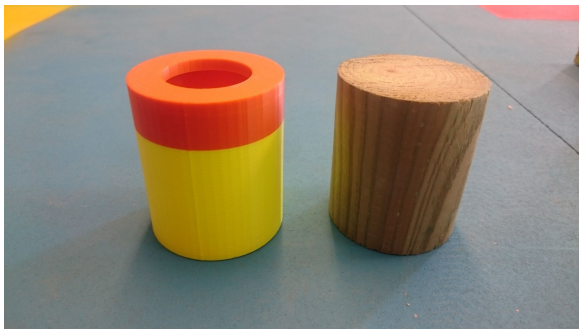


*Figure 4.2: Stands in plastic and wood.*

The task of building the beacon support tower for the robots has been given to NAVSYS, because this platform will be positioned directly above the location assigned to the robots beacon tower, and could potentially limit its design-space or interfere with its operation[46].

## 4.3 Chassis

One of the first and most defining choices to make while planning a robot is what type of chassis to base it on. This choice defines the shape of the robot, and may put limitations on the size and shape of the components that can be added later, including motors, wheel setup and the location of sensors.

### 4.3.1 CNC Milling & Laser-cutting

Research from previous competitions[1] show that many of the more serious teams build their robots a with custom-made chassis made by computer numerical control (CNC) machining[24]. Two examples of these kinds of robots is displayed in Figure 4.3, which is a picture from this years competition. The advantage of this approach is that these robots can be tailored for this exact competition, which may be advantageous in order to ensure perfect positioning of components and tools for this competitions tasks. As an example, for tasks that include gathering of game elements robots may benefit from having internal room for storage and handling of the elements. While this level of customization can be a benefit, it can also be a drawback, as it makes the robot and its components harder to reuse in future competitions.

[1] Examples can be seen at https://www.youtu.be/LjXhaaQoPHw and https://www.youtu.be/IB_bd7uPNsM?t=183.

*Figure 4.3: Two examples of robots with custom-built chassis at Eurobot 2015.*[2]

#### Alternative Materials

The optimal materials for the base are metal like steel or aluminium, as this results in the most robust and rigid base. If CNC-milling of a metal chassis is too expensive, alternative materials like wood or plastics can be cut with a laser-cutter. These materials can make for a cheaper substitute, but will result in a less rigid and robust base.

### 4.3.2 Alternative Solutions

An alternative to building everything from the ground up is to start with off-the-shelves parts, or even a full robot kit. An example of such a kit is the HCR Mobile Robot Kit from DFRobot, shown in Figure 4.4. A unit of this kit was available for use in our project, as it had been used in a previous project at the university, and was now discarded. The total price of this kit when new was relatively steep at $561 (4605.81NOK), but as neither the motors, wheels or any of the sensors or controllers would be used, the actual value of the kit as used in this project would be significantly lower, as it was now basically just three metal plates. The advantage of this kit in particular is that all parts are detachable and replaceable, making it easy to modify. Theres also plenty of screw holes for attaching additional components. Our team decided to go with this option, as it meant that we could speed up the startup process and get a working prototype finished quickly. This in turn allowed development and testing of the various subsystems, like the control software for the motors, to be started earlier.

#### Status of the Kit

The kit had as mentioned been used for a previous project, and was not in great shape, but the chassis was complete with all the structural parts

[2]Source: http://www.robots15.ch/photos_videos, Accessed on 29-06-2015

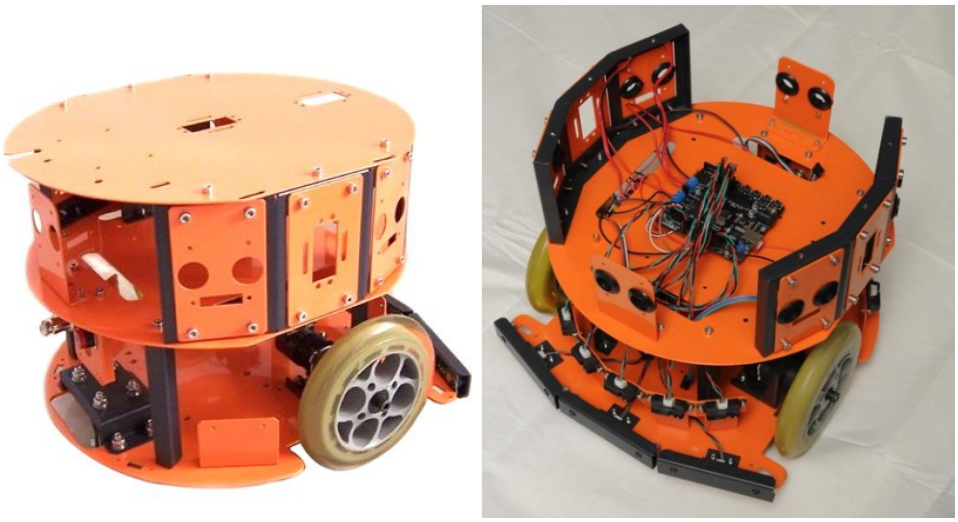


*Figure 4.4: DFRobot HCR Mobile Robot Kit.*[3]*Only the structural parts and level plates were used in this project.*

intact. The rest of the kit were not in as good of a condition as the chassis. All the sensors and other electronics had been stripped away for use in various other projects. The wheels are slightly crooked, leading to wobbly movement, and the material of the tires is a slippery plastic with almost no grip. Combined with a thin contact patch this makes them exert almost no traction. The motors were in poor shape and just barely had enough power to move the robot at walking pace.

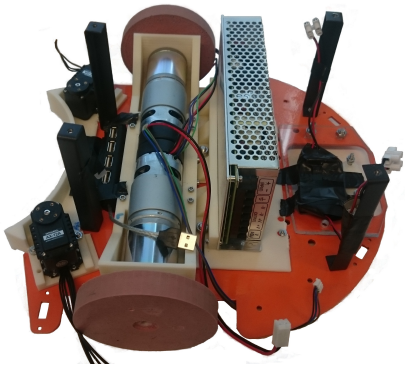
The support wheel included in the kit is a caster wheel, which is problematic for accurate movement as described in Section 2.9.1. In addition to exhibiting the discussed symptoms particular to caster wheels, this wheel specifically was not well constructed. Rotational movement was impeded by a wobbly joint, and the bearings in the wheel itself were worn and did not spin well, even when properly lubricated. For these reasons the wheels were not a good fit for the robot, and had to be replaced.

### 4.3.3 Level Structure

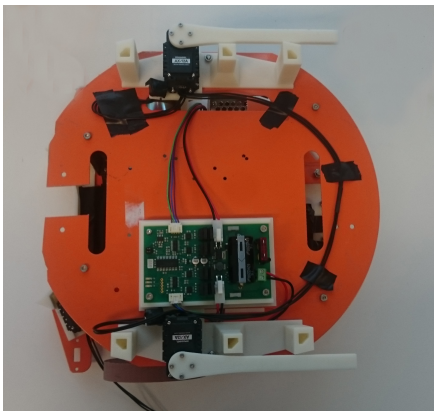
The DFR robot chassis includes three metal plates that form the levels, with metal beams to connect the plates. The beams are solid enough to provide good structural integrity, and the size is perfect for use as the base of our robot. In addition to the level plates from the set, an additional level was laser-printed in clear PMMA plastic.

An overview of the layout of the four levels on the robot, with all the components fitted, is described in Figure 4.5. The original structural beams included in the kit is used to separate the bottom and second levels, and can be seen in Figure 4.5a. The original beams for connecting the second and third row go along the entire sides of the robot (see Figure 4.4), and was replaced with 3d-printed connectors separated in two parts. The bottom

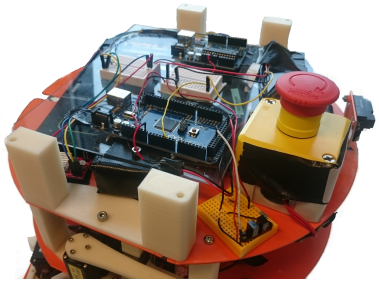
[3] http://www.robotshop.com/en/dfrobot-hcr-mobile-robot-kit-sensors-microcontroller.html, Accessed on 05-02-2015



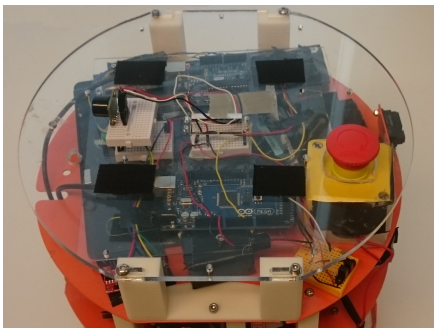
(a) The base level includes motor mounts, servo motors for gripper arms, two voltage regulators and an USB-hub for connecting various components to the laptop.



(b) The second level contains the motor controller and servos for side arms. Missing from the picture is the battery container, which is placed parallel to the motor controller.



(c) The third level contains the laptop, start/stop systems and sensors, and on top of a clear acryllic plate two Arduino boards with related circuits.



(d) The laser-cut top level. The sensor on the front is an ultrasonic proximity sensors used by NAVSYS. The velcro strips are for attaching the opponents beacon support tower

*Figure 4.5: Robot level structure. Each of the components will be discussed in detail in the course of this chapter.*

parts, shown in Figure 4.5b, double as a frame for two servo motors, which will be described in Section 4.7.2. The beams connecting the third and fourth levels are also 3d-printed.

## 4.4 Drive System

As mentioned in Section 2.9, the selection of the locomotion setup is important for the performance of the robot. To narrow the search for the optimal motors and wheel configuration it is necessary to specify the most important criteria. The two criteria given the most weight while selecting the wheel configuration were the configurations effect on movement accuracy, and the complexity of development. Based on these requirements, all configuration where position cannot be tracked by odometry alone, such as skid steering, were immediately eliminated as they would negatively impact the robots accuracy. Configurations requiring more than two motors were also discarded, as it would put an unnecessary strain on the budget without providing any significant advantages.

### 4.4.1 Motor Requirements

The motors from the DFR robot kit were immediately discarded as too weak, as they were struggling to move robot even at crawling speed. Additionally the speed of the motors were not equal even with exactly the same input voltage, which lead to curved trajectories and inaccuracies on even the simplest movements, and one of the encoders did not function properly. To be able to select appropriate motors and gears for the locomotion system, the required speed and torque figures must be calculated.

To find the approximated specifications for the motor systems, requirements were set in the form of minimum speed and torque the system would need to provide. The maximal weight of the robot was set to $15kg$, the minimum top-speed to $0.5m/s$ and the minimal acceleration to $1m/s$. These numbers are rather arbitrarily chosen after what our team decided our minimum speed should be, but should provide an indicator of the required motor specifications. Calculations of the minimal requirements for the new drive system is shown in Equations (4.1) and (4.3).

$$RPM_r = \frac{V_r}{2 * \pi * r} \tag{4.1}$$

$$F_r = \frac{1}{motors} * W * a \tag{4.2}$$

$$\tau_r = F_r * r \tag{4.3}$$

#### RPM

The amount of speed required from the motors, measured in the revolutions per minuted (RPM), is calculated as shown in Equation (4.1), where $V_r$ is the minimum robot velocity in meter per second and $r$ is the wheel diameter in meters. The radius of the wheels were subject to change, but were set to the radius of the wheels we had access to at the time, which was $3.81mm$. This gives a minimal RPM of 125.3.

#### Torque

The amount of force the system should provide is calculated in Equation (4.2), where $F_r$ is the minimum system force in Newton, $W$ is the weight of the robot, and $a$ is the minimum acceleration constant. To calculate the torque from the force Equation (4.3) is used, where $r$ is the wheel radius. Based on the requirements mentioned earlier the minimal torque the motor system should provide is 0.3 Newton meters.

### 4.4.2 Selection Process

As the parameters in the previous section were chosen rather arbitrarily based on our observations, the results of these calculations can be considered to be guidelines more than actual hard requirements. In addition to the calculated requirements of speed and force the new system must either include position encoders, or be easy to attach encoders to. Lots of effort were put into research optimal motors and controllers that meet all the specified requirements, without straining the budget too much.

### 4.4.3 Maxon Motors

The first type of motors that were looked into were advanced brushless EC motors, such as the EC45 from Maxon shown in Figure 4.6. These have many advantages; they are lightweight and compact in size, but still produces more than enough torque and speed. The complex motor controllers have excellent position/speed control features, including automatic breaking functionality, which means the robots movements will be smooth and fast. It also is worth noting that NTNU replaced their old system with motors similar to these in 2010[53], and have been pleased with their performance.

#### Drawbacks

Despite all the advantages, these motors have one significant drawback; the price tag. The cost of the motors themselves are not prohibitively high, for instance the mentioned EC45 has a price of between 120CHF (1020.00NOK) and 250CHF (2125.00NOK) per motor, depending on exact specifications. However, this is only a part of the total cost. The cost of matching gears range from 149CHF (1266.50NOK) for spur gears to above 250CHF (2125.00NOK) for planetary gears.

*Figure 4.6: Maxon EC45 brushless DC motor*

Sophisticated motor controllers are required to fully utilize the capabilities of the EC motors. The recommended system for the EC45 (and similar motors) is the EPOS2, which at at price point of 522CHF (4437.00NOK) is quite expensive.

### 4.4.4 Devantech motors

As the type of motors discussed in the previous chapter ended up being too expensive for our project, the requirements were slightly adjusted, and the Devantech RD03 Robot Drive system were ordered instead. The system includes two 24V DC motors with gears, as well as a motor driver, that can be seen in Figures 4.7 and 4.8. The reasons to select this system in particular was that the specs of the gear and motor fulfilled the requirements for speed and torque, and has a fair price. It is also an important point that the motor controller included in the system is specifically designed to control these motors, which meant that it should have no issues controlling the motors.

(a)

| Specifications | |
|---|---|
| Rated Voltage | 24V |
| Rated Current | 2.1A |
| No Load Current | 0.5A |
| Rated Torque | 16 kg/cm |
| Rated Speed | 122rpm |
| Gearbox Reduction | 49:1 |
| Encoder Resolution | 980 |

(b)

*Figure 4.7: Devantech EMG49 24V motors*

### 4.4.5 Motor controller

The Devantech MD49 motor controller is a serial-controlled dual motor driver manufactured specifically for use with the Devantech EMG49 motors. The controller itself is a dual H-bridge motor driver, capable of delivering up to 5A continuous current to two motors at 24V. It has multiple useful features, summarized in Figure 4.8.

#### Key Features

One of the key attributes of the MD49 is the built-in serial communication interface, which makes it easy to control directly from a PC via a universal serial bus (USB) connection, and eliminates the need for additional electronic circuits. Other useful features are protection against short-circuits, under/over voltage and excess current. Output to the motors can be controlled automatically by a power regulator, and a constant can be set to determine the amount of acceleration. The impact of these settings will be explored in further detail in Section 5.2.3.

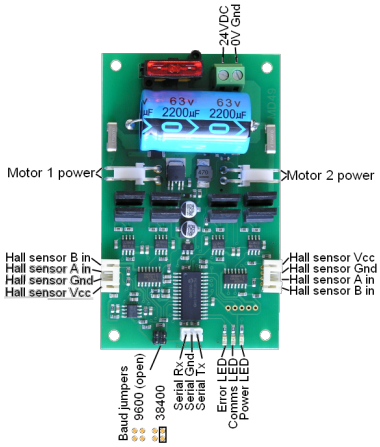
#### Encoder Odometry

Odometry is the technique of using encoder feedback to calculate approximate position and velocity. For motors fitted with encoders this is a fairly simple process. The Devantech EMG49 are fitted with Hall sensors (described in Section 2.5) with a resolution of 980 ticks per revolution, and the controller automatically reads these encoder ticks and provides counts that can be read through the serial interface. This simplifies the process of measuring distance traveled by each motor, which can be used to calculate the robots direction, heading and velocity.

### 4.4.6 Odometry

As the drive system utilizes a differential drive configuration with two driving wheels sharing a wheel axis, the center of rotation for the robot will be the middle of this shared axis. This reference point is the center of rotation for the center of the robot, and is where position, movement and heading will be calculated from.

The current state trajectory of the robot can be integrated as shown in Equation (4.4), where $x_0$, $y_0$ and $\theta_0$ are the starting position for the movement, and $v$ and $\omega$ are translational and rotational velocity. These velocities can be further decomposed into the angular movement of each drive wheel, $\omega_1$ and $\omega_2$, as shown in Equations (4.5) and (4.6), which can be directly integrated from encoder feedback.




(a)

| Specifications | |
|---|---|
| Rated Voltage | 24V |
| Rated Current | 5A (for each motor) |
| Communication | Serial (38400 baud) |
| | Voltage protection |
| Other Features | Short-circuit protection |
| | Variable regulator |
| | Encoder processing |

(b) Devantech MD49 specifications.[4]

*Figure 4.8: The Devantech MD49 motor driver card.*

$$q(t) = \begin{bmatrix} x(t) \\ y(t) \\ \theta(t) \end{bmatrix} = \begin{bmatrix} x_0 + \int_0^t v\cos(\theta) \\ y_0 + \int_0^t v\sin(\theta) \\ \theta_0 + \int_0^t \omega \end{bmatrix} \tag{4.4}$$

$$v = \frac{\omega_1 + \omega_2}{2} \tag{4.5}$$

$$\omega = \frac{\omega_1 - \omega_2}{2} \tag{4.6}$$

### 4.4.7 Motor Mounts

Mounts to attach the Devantech motors to the main chassis were printed. The first iteration (Figure 4.9) were designed to be as simple as possible in order to get the system up and running quickly, so that testing and implementation could be started. The design turned out to be poor; the mounts were not stable enough as they did not lock the back end of the motors in place, causing significant problems with undesirable horizontal movement of the mounts relative to the robot base. The exact impact on navigation accuracy was difficult to measure as it was an effect that increased over time, depending on the types of movement, and the force exerted. For instance fast reversals of direction exaggerates the problem. But the effect had a clearly noticeable negative impact on movements.

#### Second Iteration

For the next iteration the focus was on eliminating the horizontal backlash of the first version. This was solved by creating mounts that lock into each

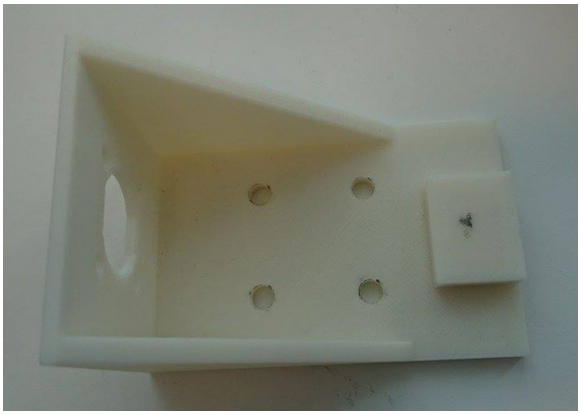


*Figure 4.9: First iteration motor mounts for Devantech*

other to form a solid unit, and simultaneously lock the motors firmly in place. The result can be seen in Figure 4.10. The improvements worked perfectly, and the second iteration of the mounts are able to hold the motors firmly in place, eliminating any horizontal movements.

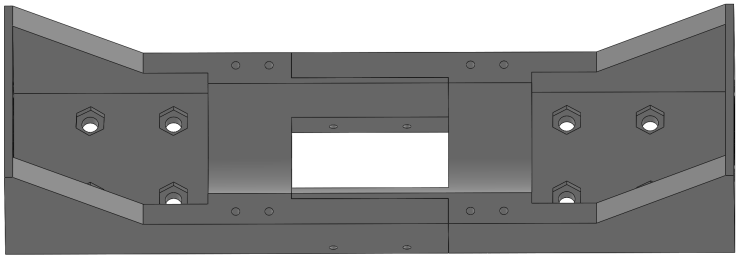
## 4.5 Wheels and Tires

Wheels, and potentially tires or tracks, are integral parts of a robots locomotion-system as they are the only points of impact between a robot and the ground. All the force generated by the robots locomotion system is exerted on the environment trough the contact patches of the tires, which means that the planning and setup of this system is critical to the robots overall performance[4]. A good locomotion configuration ensures that enough force can be put down to enable the robot to perform all its tasks smoothly, quickly and with a high degree of precision.

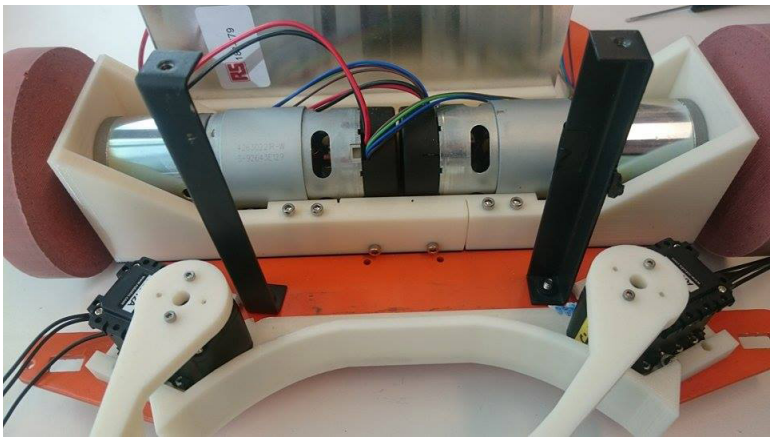
For more general purpose mobile robots the locomotion system have to be able to negotiate different types of terrain, sometimes with unknown types of surfaces and obstacles. This problem is non-existent in this case, as the Eurobot board only features flat, smooth surfaces (with the exception of the staircase in Section 3.9).

### 4.5.1 Setup

As mentioned in the motor sections, the criteria for the locomotion system was that it should not be based on more than two motors, be easy to implement and work with, and have a high degree of accuracy. The natural choice based on the information provided in Section 2.9 is the differential drive setup described in Section 2.10.4. This configuration is simple to implement; two motors driving a wheel on opposing sides of the robot, and one or more support wheels keeping the robot stable.



(a) The 3D models in Solidworks



(b) The finished parts mounted on the robot.

*Figure 4.10: Second iteration motor mounts for Devantech*

### DFRRobot Wheels

The wheels originally fit on the DFRrobot are not well suited for precision robotics. The contact patches on the tires are incredibly thin, leading to underwhelming traction. Additionally the build quality of the wheels are poor, and they have both become deformed during use, leading to wobbly movement.

### Devantech Wheels

A pair of wheels, displayed in Figure 4.11 were included in the Devantech motor set. Unfortunately the tires on these wheels are made of a hard, slippery plastic and produce almost no traction (see figure). The contact patch looks wide, but by closer inspection of the images one can see that the contact patch is in fact only a thin line along the center of the wheel.

The size is also an issue, with a diameter of 125mm they are far larger than optimal, as a larger wheel decreases the resolution of the robots positional accuracy.

*Figure 4.11: Wheels included with the Devantech system.*

### Molded Silicone Wheels

Finding wheels that fit the size criteria with a reasonable price point and delivery time turned out to be difficult, so it was instead decided to make a new pair of wheels from scratch by casting them in silicone rubber.

The first step was to print core, structural part of the wheel, shown in Figure 4.12a. The hubs are designed to fit the motor shafts of the Devantech motors in Section 4.4.1, and the outside edges are fitted with gear teeth to better hold the silicone tires in place on the inner parts and prevent them from sliding.

(a) Wheel core (b) The wheel core inside the casting mold

*Figure 4.12: Solidworks models of the parts used in the wheel casting process.*

A mold for the wheel was modeled and printed, and the wheel core fitted inside it as illustrated in Figure 4.12b. The molds were filled with a mixed two-component silicone rubber, and left to set.

### Result

The final result can be seen in Figures 4.13 and 4.14. After some small adjustments the hubs fit the motor shafts of the motors perfectly. The screw tightening system, shown in Figure 4.14 turned out to not work very

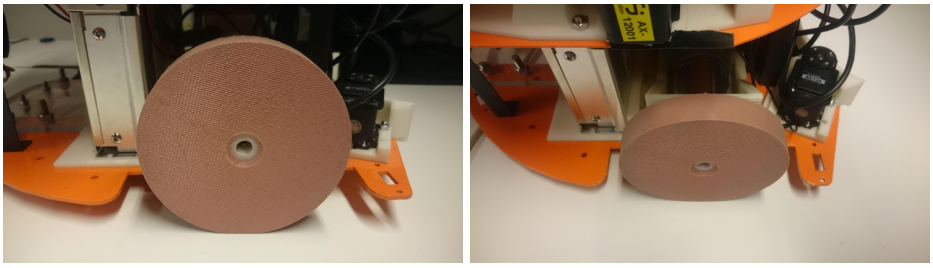


*Figure 4.13: The finished silicone wheel.*

well as the screws ripped up the threads in the plastic when pulled too tightly. However, this did not turn out to be a problem as the wheel hubs themselves are tight solid enough to hold the wheels firmly in place during movement.

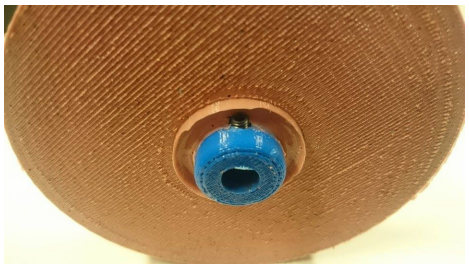


*Figure 4.14: Wheel hub tightening screw.*

Both tires have a couple of small imperfections due to air-bubbles, which suggest that even more work could have been put into removing the last traces of air during the molding process. These imperfections are however purely cosmetic, and are not significant enough to affect the structural integrity of the tire. Overall the quality of the tires exceeded the expectations, they are solid and provide high amounts of traction due to the slightly sticky texture of the silicone rubber. An inconvenient effect of this stickyness is that it gathers significant amount of dust and dirt while driving, and they must be maintained and cleaned regularly. A comparison between the wheels traction can be found in Section 5.2.1.

## 4.6 Power System

The Eurobot regulations allow for several kinds of power sources[20], but a battery-based system is the only realistic option. A system to deliver power to the motors and other electrical components had to be designed. Production of a specialized power-card is outside the scope of this thesis, as it would take too much time and resources, so off-the-shelves components will be used instead.

Electrical motors pulls the highest amount of current, often called stall current, when starting movement from a dead stop, and when doing a sudden stop such as hitting a solid object. This higher than usual pull

of current may last for several seconds, and is called a power spike. If a spike is significant enough, the voltage of the power circuit may be pulled down far enough that other connected components quit working or are reset. If the components are fragile they may also be damaged, as the power spikes may expose them to power they are not rated for.These spikes may also damage components in the circuit as they may be exposed to power they are not rated for. Measures can be taken to reduce the effect of the power spikes on other components, but the simplest step is to place fragile components in separate circuits from motors.

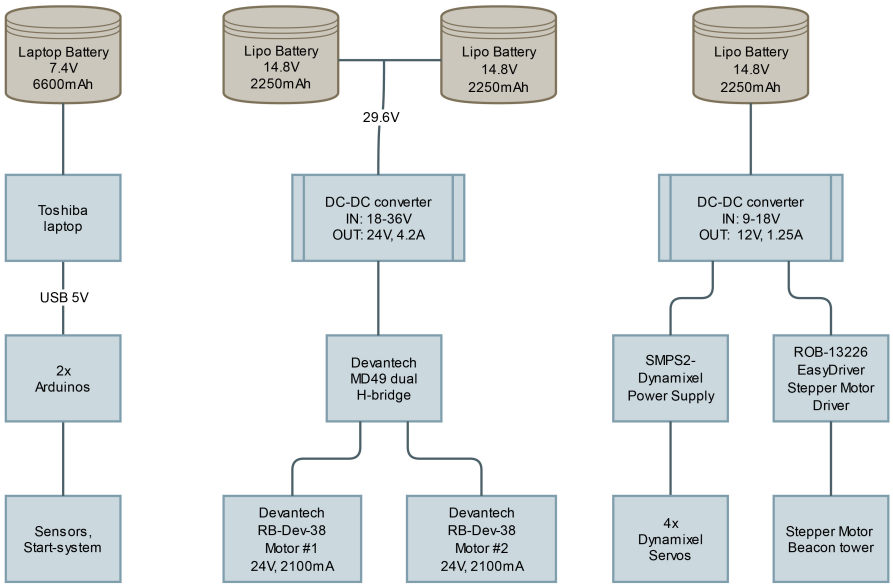
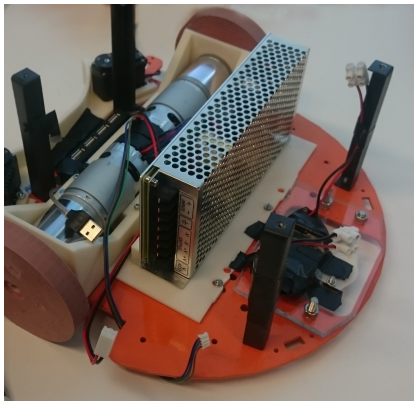



*Figure 4.15: Overview of the battery system and its connected components.*

An overview of the power system is provided in Figure 4.15, which will be discussed further throughout this section. There are two main power systems; the motors are powered by two batteries, and their output voltage is converted to 24V by a dc-dc converter. The servo power system is similar, and consists of one battery, and a converter regulates the voltage down to 12V. The EasyDriver Stepper Motor Driver and the Stepper Motor for the beacon tower is a part of the NAVSYS thesis, but is powered from this system. In addition to these systems, the laptop has a built-in battery that is used to power itself, as well powering two Arduino cards, and their connected sensors, via two 5V USB connections.

### 4.6.1 DC-DC Converter

The dc-dc voltage regulator chosen is the SD-100 from RS-components. The reason for using a switching regulator is discussed in Section 2.8, and this one in particular was selected as it has all the necessary specifications, and is reasonably priced compared to similar products. The dimensions of the



(a) Converter fitted on the robot.

| Specifications | |
|---|---|
| Voltage input range | 19-36V |
| Output Voltage | 24V DC |
| Rated Current | 4.2A |
| Switching Frequency | 83kHz |
| Dimensions | 199x98x38mm |
| Weight | 650g |
| Other Features | Short-circuit protection |
| | Overload protection |
| | Over Voltage protection |

(b) Specifications

*Figure 4.16: The RS-Components SD100 DC-DC Converter. The main drawback is its large size, but it fits on the robot without issues.*

box are rather large, but as Figure 4.16a shows, positioning it on the robot is not an issue. The reason the converter is not positioned further towards the back of the robot in order to improve weight distribution, is that the batteries were originally intended to be placed behind it. As the battery safety container ordered later did not fit well there, it would in retrospect have been better to position the converter further back, over the support wheel.

### 4.6.2 Servo Power System

The power system for the servos is implemented in the same way as the motor power system, but at a lower voltage. It consists of one 14.8V lipo battery stepped down to a stable 12V by the converter shown in Figure 4.17.


(a) The converter

| Specifications | |
|---|---|
| Voltage input range | 9-18V |
| Output Voltage | 12V DC |
| Rated Current | 1.25A |
| Switching Frequency | 330kHz |
| Dimensions | 50.8x40.6x9.5mm |
| Weight | 48g |
| Other Features | Short-circuit protection |

(b) Specifications

*Figure 4.17: Tracopower TEN 25-1222 DC-DC Converter.*

### 4.6.3 Batteries

The battery system must be able to supply enough current, while maintaining a stable voltage, Consideration must also be taken to keep the weight of the batteries low. According to the regulations the robot must be able to complete three consecutive matches (of 90 seconds), as well as the match preparation stages, without recharging or replacing batteries.

The preparation stages take between 10 and 15 minutes, but during this time the motors should not have to spend any amount of energy. The minimum amount of the batteries need to be able to keep the robot powered were arbitrarily chosen as 30 minutes, as this should be more than enough safety margin. The minimum capacity can then simply be calculated as shown in Equation (4.7).

$$C_{min} = I * t_{min} \tag{4.7}$$

Equation (4.7) shows a worst-case calculation of the amount of capacity necessary to run the robot for a total of 30 minutes.

This amount of time, $t_{min}$, was arbitrarily chosen, to make sure the batteries can supply enough power to last at least three matches of 90 seconds, and the match preparation phase of 10 minutes. This allows for a solid safety margin, as the motors will not be expending any significant amounts of power during the preparation phase. The calculation assumes a worst-case scenario, where the motors are drawing their maximum continuous current. $C_{min}$ is the minimum total battery capacity necessary for the chosen time, and $I$ is the total amperage of the two motors. There will be additional losses of power due to factors external to the batteries like losses in the circuits, but with the worst-case situation draw being used in the calculation, this amount should still be sufficient.

#### LiPo Batteries

In addition to the power requirements, considerations should be made to maximize energy density and minimize charge-time. These are the main reasons lipo batteries were selected, as these fulfill all these requirements, as mentioned in Section 2.6. As explained in that section lipo batteries consists of cells of 3.7V, which makes it impossible to produce lipo batteries with exactly 24V. The closest would be a 7-cell battery at 25.9V, but that is not commonly manufactured and usually have a high cost. Additionally running lipo batteries with different number of cells in series is not recommended, as it may cause problems during charging and discharging.

It was instead decided to purchase more standard 4-cell lipo batteries at 14.8V, displayed in Figure 4.18, and run two of these in series to create a combined 29.6V, then step this voltage down to 24V with a voltage converter. As explained in Section 2.6, lipo cells deliver different voltages during their discharge cycle, meaning that a voltage regulator would be a necessary component anyway. An advantage of using these batteries is that a single one can be used to power components running at 12V, with a smaller converter.

(a) Battery with added connector

| Specifications | |
|---|---|
| LiPo cells | 4 |
| Nominal Voltage | 14.8V |
| Max continuos discharge | 4A |
| Nominal Capacity | 2250mAh |
| | Over/under voltage protection |
| Other features | Short circuit protection |
| | Charge balancer |

(b) Specifications

*Figure 4.18: The lithium polymer batteries.*


*Figure 4.19: Lipo batteries inside the fireproof safebag, with connectors sticking out.*

According to the Eurobot regulations all lithium-based batteries must be inside safe containers at all times, and we chose to use fireproof lipo-safe bags as seen in Figure 4.19, as they are flexible and easy to position on the robot.

### 4.6.4 Stop system

As mentioned in Section 3.2, all robots must be outfitted with specific equipment to allow them to be started and stopped easily. The stop system has been solved by routing the positive wires from batteries to the respective power systems through a stop button enclosure. The enclosure is modified with an additional contact, in order to stop both the power to the motors and the servo-system simultaneously. Pushing the button down cuts the power to both systems, and the motors, motor controller and servos are de-energized. The enclosure is mounted so that the button is positioned at the top level on the back of the robot, as shown in Figure 4.20, to be as easily reachable as possible.

### 4.6.5 Start system

The start system, shown in Figure 4.21 has been implemented using a slotted optical switch, described inSection 2.8. A piece of IR-absorbent

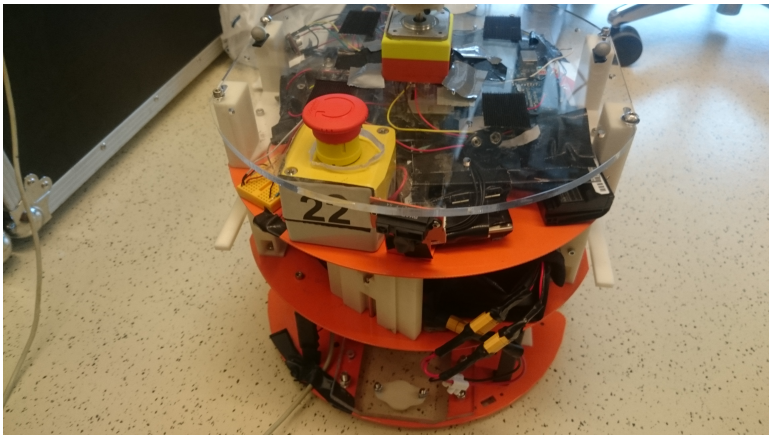
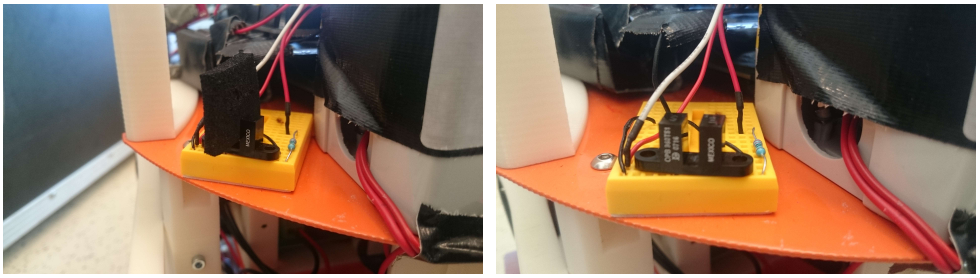


*Figure 4.20: Stop button mounted on the robot.*

material attached to a cord, the *start pin*, is placed in the slot, ´´arming" the system and causing the switch to pull its output high. When the start-pin is removed the output of the switch is pulled low. If the system was in an armed state, this is interpreted as a start signal. The output of the switch is read at a set frequency by an Arduino run by NAVSYS, who sends the start signal to the waiting AI. This is described in further detail in the NAVSYS part of the project[46].



(a) Sensor blocked; system is in armed state.

(b) Sensor open; if system was in an armed state robot will now start.

*Figure 4.21: The start system.*

## 4.7 Part Construction

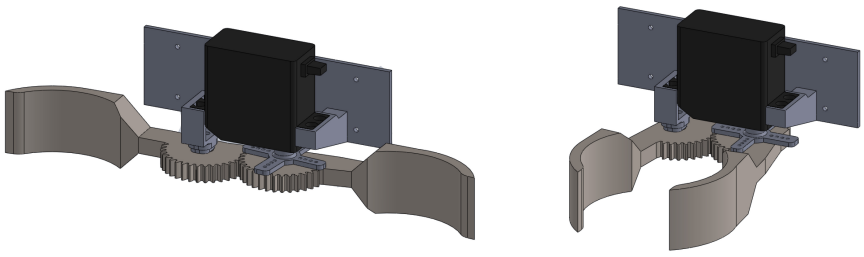
This section describes the construction of structural parts of the robot and the layout of the different levels, as well as the creation of tools used to solve the various Eurobot tasks. Most of the parts in this section is modeled in Solidworks and printed on the Fortus 250mc fused deposition modeling printer. As mentioned in Section 2.11, the reason for using 3D-printed parts is that it allows for creating of the exact parts needed, and the resulting parts are robust and relatively cheap to make.

### 4.7.1 Gathering Tasks

Both the spotlight and popcorn tasks specified in Section 3.5 require some form of gripping mechanism for gathering game objects. Using the same tools for both tasks will reduce the time and components (such as servo motors) used on implementation, but may be disadvantageous for solving the task in the least amount of time, as fewer total game elements can be held at the same time. The spotlight task also requires a mechanism for lifting the stands in order to build the stacks and potentially spotlight towers.

#### First Iteration

The first version of the gripping mechanism, displayed in Figures 4.22 and 4.23, was designed to be as mechanically simple as possible and fit a simple servo motor that we had available. The construction consists of two gripper arms that were designed to perfectly grip a stand. One of them is attached directly to the servo, and its movements are mirrored to the opposite gripper arm by a row of gear teeth, thus only requiring one servo.



(a) Open position (b) Closed position

*Figure 4.22: 3d models of the first generation grippers in Solidworks.*

The mechanical configuration worked well, but the servo struggled to exert enough force to hold objects firmly. The lack of a position controller was a huge problem, as the servo would lose track of the position of the arm when it encountered too high resistance on a movement. This will be discussed in more detail in Section 5.5.

#### Second Iteration

Because of the problems with the servo used in the first iteration, it was switched out with a more robust Dynamixel AX-12A servo, displayed in Figure 4.24. Contrary to the cheaper servo the Dynamixel has a sophisticated position control system with a PID regulator, and have a large range of configuration settings. The settings allow for tweaking of parameters like maximum speed and torque, and also make it possible to set a define a range of valid positions. According to the AX-12A's specifications, seen in Section 4.7.1, it is slightly faster than the SpringRC servo (59 RPM as opposed to 54), but more importantly it is capable of

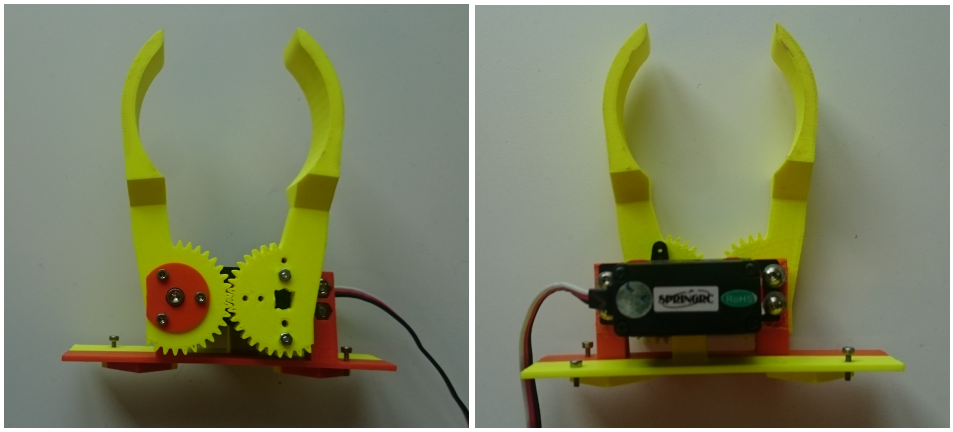


*Figure 4.23: Gripper arms mounted on servo, connected by gear teeth.*


(a) Servo

| Specifications | |
|---|---|
| Operating Voltage | 12V |
| Max Current | 900mA |
| Stall Torque | 1.5 Nm |
| No-load Speed | 59 RPM |
| Angular Resolution | 0.29° |
| Dimensions | 32x50x40 mm |
| Weight | 55g |
| Other Features | Sophisticted position controller |

(b) Specifications

*Figure 4.24: Dynamixel AX-12A servo motor.*

producing almost three times as much torque. This, combined with an advanced PID-controller, should be enough to solve the problems which the previous servo had of not being able to keep hold of objects.

Figure 4.25 shows the new generation grippers, redesigned to fit the new servo. The centerpoint of the arms is moved slightly forward compared to the previous generation, but this did not affect the arms ability to grip stands, as the arms still hit perfectly. This new generation were able to successfully grip stands and hold them in place during lifts, as demonstrated in Figure 4.26.

### Third iteration

A month before the competition the point gathering strategy was changed somewhat. This was a result of the current state of the various parts of the projects, most significant of them was the problem with movement accuracy, and the relation between AI and CTRLSYS. The navigation-system was also less accurate and reliable than anticipated, which meant

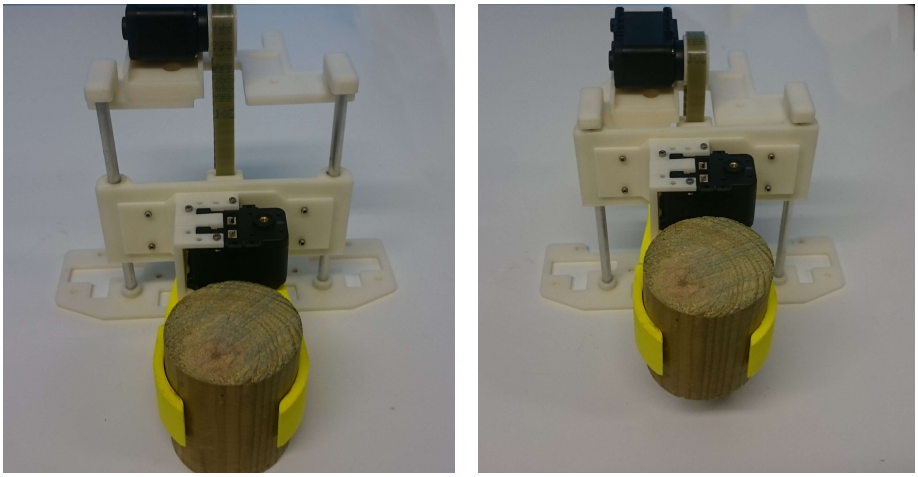
*Figure 4.25: Second gripper iteration with Dynamixel servo.*

that only encoder was used for positioning. These problems meant that we could not reliably build the spotlights, as exact navigation is required in order to prevent missing a stand, which may result in the spotlight falling over and the loss of all points. The slightest errors would lead to the stacks not being built correctly. This meant that either the mechanical design had to be drastically improved to ensure correct positioning of stands before attempting to build stacks, or a simpler strategy had to be chosen. The theory was that by simply gathering the stands inside the areas without stacking them, the task still awards a decent amount of points, and the chance of failing the task, as well as the time spent, is reduced drastically.

To accommodate the revised strategy some modifications were applied to the base and front of the robot. As shown in Figure 4.27, the base plate was cut slightly inward, in order to allow for the stands and popcorn cups to be contained inside the front of the robot, to increase the amount of objects that could be gathered simultaneously. The new strategy worked well, and sped up the gathering of game elements considerably, but also reduced the maximum amount of points our robot could score, by eliminating the ability to get bonus points from the spotlight task (see Section 3.6).

#### Lifting mechanism

Figure 4.27 also shows the lifting mechanism implemented to . The design is mechanically simple, and consists of a stepper motor turning a belt via a gear, which in turn raises and lowers a trolley connected to the belt. The diamond-shaped holes are designed so that another part with a gripper mechanism can easily be attached. The lift was however cut from the robot soon after it was implemented, as explained in Section 4.7.1.



(a) Lift bottom position (b) Lift top position

*Figure 4.26: Gripping system holding a stand in place while lifting.*

### 4.7.2 Clapper-board Shutter

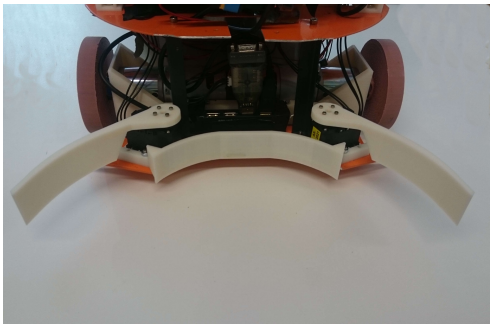
Closing the clapperboards is the easiest task to perform, and simply requires the design of an arm that can extended while the robot drives past the clapperboards. Multiple possible designs were considered, but sticking with the design philosophy, the implementation with the simplest mechanical design was chosen. A servo is mounted on each side on the second level, and a simple arm mounted on each one. While the robot is in its start configuration the arms are held along the side, and are then rotated out 90 degrees in order to hit the clappers.

## 4.8 Main Processing Unit

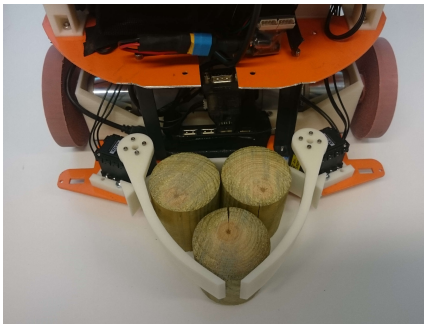
The main processing unit (MPU) is the piece of hardware that can be considered the core of the robot, responsible for running the artificial intelligence and handling its communication with the various subsystems. The choice of hardware platform for the MPU severely impacts the selection of software, programming languages, libraries and communication systems. Different hardware platforms may also have varying performance in relation to the amount of computation possible, which may impact the general responsiveness of the system.

There is numerous ways to implement a MPU. Conventional approaches include using laptops [3, 38], custom-built PCs based on mini-atx motherboards[37] , and PC/104 stacks[9, 24]. The PC/104 system was considered, but the team ultimately decided against it as it would be too expensive, and in addition none of the team members had any experience with the system.

(a)



(b) Lift bottom position



(c) Lift top position

*Figure 4.27: Third and final gripper iteration.*

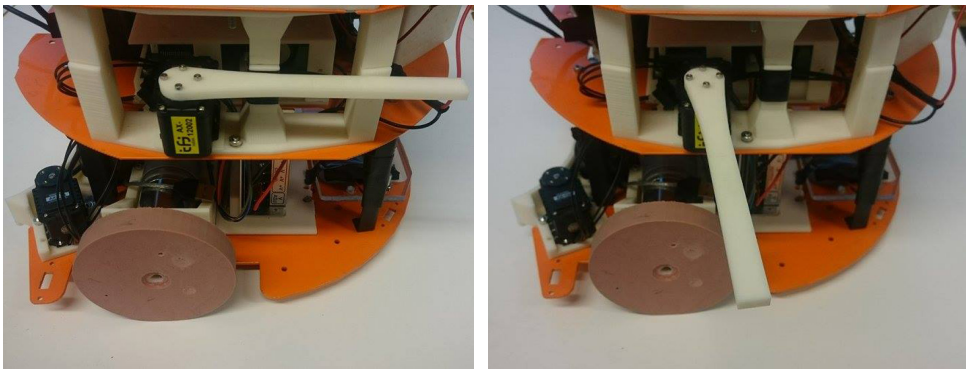
### 4.8.1 x86 Laptop

The most important reason for the decision to base the system on a laptop was that it allows us to use every conceivable combination of programming languages, libraries and software tools, and gives every member of the team as much freedom as possible. Using a laptop is also in many ways more convenient than using for instance a development board. Ethernet and wireless network cards and multiple USB connections makes them easy to connect to, and a screen means easier interfacing. One of the most important conveniences however is the built-in batteries, which can run a laptop for multiple hours and removes the need for a dedicated power system.

A laptop is often a more expensive piece of equipment than some of the other alternatives, but finding someone with a discarded, slightly older laptop is usually not a difficult task, so getting one cheap or even for free is often possible. In this project the laptop used were a old mini-laptop one of the team-members had replaced and stopped using.

#### Drawbacks

The challenge of using a x86-based laptop running Linux as the MPU is that neither the hardware nor software are, by default, optimized for real-time operations. The Linux scheduler (ref CFS) is optimized for desktop use, and focuses on giving each process their fair share of processing time, and attempts to have short response-time on interactive operations initiated by



(a) Closed state (b) Open state

*Figure 4.28: Arms to shut clapperboards.*

(a) The modified laptop (b) Laptop mounted on the robot

*Figure 4.29: The Toshiba NB550D-105 Laptop used as the main processing unit of the project. It had to be modified in order to fit on the robot.*

users. This is done to avoid having programs wait too long, as it provides a poor user experience. CFS has no hard, direct prioritization of processes, and is not fit for use in a real-time system.

To attempt to mitigate this problem, various custom kernels with different schedulers and settings were tested. The results from these tests show the impact on latency, and are summarized in Section 5.3.

### The laptop and Modifications

The laptop used on the final version of the robot is a Toshiba NB550D-105 laptop, that was gifted to the project by one of the team members. It is a basic small laptop with a AMD Dual-Core 1.0 gigahertz CPU and 1GB of memory. These specifications are obviously outdated compared to more modern laptops, but they are fine for the requirements of this project.

To remove some weight and make the laptop easier to mount, the screen, along with the internal antenna for the wireless network card were removed, and the result can be seen inFigure 4.29. To communicate with the laptop directly a wireless USB dongle was used, and during the competition we used a twisted network cable to communicate with it

directly trough ethernet, to avoid having to bring a router. The backup solution in case networks fail is to connect the laptop to a screen via HDMI. To make room for the stop-button enclosure on the same level as the as the pc, some unnecessary parts of the laptop casing were cut off, leaving only the necessary parts of the laptop chassis.

## 4.9 Operating System

With a general purpose pc as the MPU, a range of different operating systems are available. As mentioned in Section 2.14, a major reason for selecting a pc as the MPU was the freedom to use any programming language, technology or framework. With that in mind, the obvious choice for the OS was Linux, as it has a virtually unlimited selection of software. As discussed in Section 2.14 Linux is not, and cannot be made into, a hard real-time operating system. Its performance in that respect is not comparable to for instance VxWorks. It does however have good performance on almost any hardware, and optimized for low latency it seems to be respondent enough to function as the core of the system. Compiling a kernel with a customized versions of the PREEMPT_RT patch mentioned in Section 2.14.1, to turn it into a soft real time system in order to reduce latency. The result of this can be seen in Section 5.3.

## 4.10 Control Software

The control software is an entire system that controls all physical parts of the robot. It communicates with the AI as described in Section 4.11, and executes the actions requested by the AI. CTRLSYS sends velocity information to the motor-controller which controls the direction and speed of the motors. Additionally encoder feedback from the motors is used to estimate the robots current position, direction and velocity by odometric calculations. This information is relayed to NAVSYS, and is sent as an input to a filter calculating the robot's global position. A significant part of the development time was spent writing the control software, but as most of the code and details are generally not that interesting, this section will just contain a brief explanation of how they works.

### 4.10.1 Strategy During Implementation

As the project have split the implementation of the AI and the control system into two parts, the planning aspect of the movement control have been kept to a minimum in CTRLSYS, in order to not interfere with the planning performed by the AI. Trying to improve the plan sent by AI might generate additional, hard to debug results if the robot does not behave as the AI expects it to. The movement pattern used by the robot is a simple ´´point and shoot" pattern, where the robot in order to move between positions first will rotate towards the goal position, then drive in a straight line to reach the gaol. The distance between the waypoints is kept short

enough that the AI can use this method to plan its paths effectively. This type of movement pattern is used both in order to keep the movements predictable for the AI, but also to increase the reliability of the encoder feedback. This method of driving is slower and less direct than a system which allows arc movements, but is safer and more reliable when driving on encoders[24].

## 4.11 ZeroMQ Message Queue Implementation

Because all members of the project write their own separate programs, as explained in Section 1.4, it is necessary to implement some form of communication between them. To allow each of the programs to be treated as modules, and remote testing of applications, it was decided to use a messaging queue library with a networking capability. The reasons for selecting ZMQ in particular is described in Section 2.15.

Our implementation of the communication protocol in ZMQ is very simple, but effective. In CTRLSYS a ZMQ-server is run in its own thread, accepting connections from AI, NAVSYS and other ZMQ-clients used for testing. A communication protocol consisting of messages with a size of 6 bytes are use to send the information, which can be either a command, or a request for information about the current state of the robot, for instance a position vector.

A useful feature is that ZMQ handles all connections in the background, meaning that there is, to the user, no discernible difference between a local and a network connection (aside from a slightly high latency). This proved to be a very useful feature while testing the cooperation of the modules during rapid development periods, as it allowed us to work separately without having to push/pull files and recompile, or alternatively develop remotely. Processes like this is made much simpler by revision control systems like Git, which were used throughout the project, but it still severely slows down the workflow compared to simply developing locally. The implementation of the ZMQ server, as well as a token ZMQ-client, can be found in Sections A.1 and A.2.

## 4.12 The Secondary Robot

The main robot is based on a two-wheel differential drive configuration, described in Section 2.9. This configuration is not suited for climbing the stairs of the staircase task (see Section 3.9), because the robot do not have the traction or ground clearance necessary to ascend the steps on the staircase. The Eurobot rules (Chapter 3) allows each team to field an additional robot. This robot must follow the same rules regulations as the main robot, and has additional size constraints. In addition to climbing the stair-case, this robot should also solve the red-carpet task (see Section 3.10).

### 4.12.1 Rover 5 chassis

The Rover 5 chassis was considered as a base for the secondary robot, because of its simplicity and availability. Another important point about the Rover 5 is that it uses a skid steering (see Section 2.10.1) configuration with tracks instead of wheels. This greatly increases traction, and would make it perfect for the task of climbing the stairs.

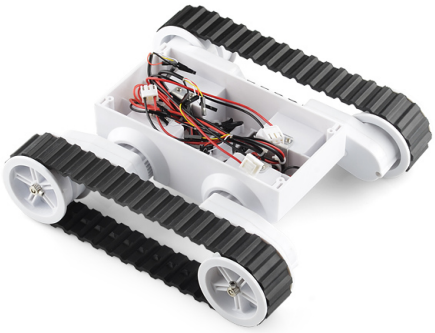


*Figure 4.30: Rover 5 chassis*

### 4.12.2 Explorer PCB Motor Controller

The Explorer PCB is a custom-made motor controller for the Rover 5 chassis, and contains a dual FET H-bridge, similar to the main robots controller implemented in Section 4.4.5. Additionally it has many other useful features such as built-in voltage regulators, servo controllers, a trickle charger and four IR proximity sensors. There is no battery system included in the Rover5 kit, but Explorer PDB contains a built-in power-system optimized for NiMh/NiCd batteris, as well as a charging circuit. This power-system distributes power to the motors and motor controller, various sensors and encoders, as well as power-rails for connection of servo-motors.

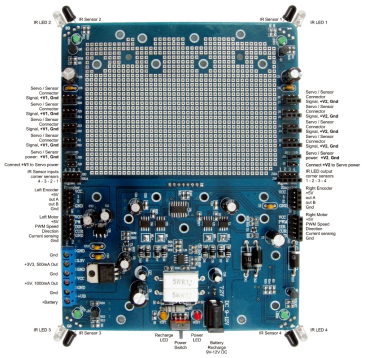


*Figure 4.31: Explorer PCB Motor Controller*

### 4.12.3 End of Development

Due to limited time, it was decided to cut the development of the secondary robot to instead focus on getting the main robot ready for the competition. The reasoning behind this decision was that the secondary robots must adhere to all the constraints as the main robot, and must contain all the same features. This would include development of a start and stop system, proximity detection systems, beacon support platform and more. This was based on the impression our team had from research and videos from previous competition, that the secondary robots would be subject to the same level of scrutiny during the homologation phase as the main robot. This turned out to not be the case, and building the secondary robot would take far less time and resources than we thought. This will be discussed further in Section 6.2.5.

# Chapter 5

# Results

This chapter contains results from various tests and experiments performed in order to determine whether each part of the complete system is working satisfactory. Each section will include an explanation of the tests performed, graphs or tables showing the results themselves, and an analysis section discussing the results. Some sections also include modifications made in response to the findings in the tests, and how those modifications affected the performance of the related systems.

## 5.1 Power System

This section will inspect the performance of the two power systems, and a number of tests are performed in order to ensure that the system is stable and able to deliver the amount of power necessary to drive their respective motors at full capacity.

### 5.1.1 Measuring Equipment


(a)

| Specification | Range | Resolution |
|---|---|---|
| Current | 0-130A | 0.01A |
| Voltage | 0-60V | 0.01V |
| Power | 0-6554W | 0.1W |
| Charge | 0-65Ah | 0.001Ah |
| Energy | 0-6554Wh | 0.1Wh |

(b)

*Figure 5.1: Turnigy 130A Watt Meter and specifications.*[1]

[1] http://www.hobbyking.com/hobbycity/forum/uploads/7563/Wattmeter.zip, Accessed at 02-06-2015

The measuring unit used for gathering statistical data is the Turnigy 130A Watt Meter/Power Analyzer. This type of measuring units have a lower resolution than more sophisticated equipment like oscilloscopes, but the specification, shown in Figure 5.1, are accurate enough for use in these experiments.

**Oscilloscope**

To perform a more in depth analysis of the behavior of voltage in the circuits a Tektronic TDS 1001B oscilloscope[2] was used.

### 5.1.2 Statistics

| Power source | Measurement | Idle | Driving | |
|---|---|---|---|---|
| | | | Free | Load |
| Converter Input | $A_{CONT}$ | 0.10-0.11A | 0.86-0.89A | 0.94-0.96A |
| | $A_{PEAK}$ | 0.11A | 1.12A | 1.73A |
| | $W_{PEAK}$ | 3.4W | 34.9W | 52.5W |
| | $V_{MIN}$ | 31.43V | 31.2V | 31V |
| Converter Output | $A_{CONT}$ | 0.01-0.03A | 0.75A | 1.0-1.1A |
| | $A_{PEAK}$ | 0.03A | 1.01A | 1.88A |
| | $W_{PEAK}$ | 0.7W | 24.3W | 45.2W |
| | $V_{MIN}$ | 24.1V | **12.21V** | **7.81V** |

*Figure 5.2: Measurements performed on the voltage converter in different states. Unexpected results are marked in gray.*

The initial test was performed by measuring various parameters while the robots in three different states; idle, driving with the wheels spinning freely, and driving with a load (i.e. driving on the ground). The driving tests included driving straight distances at maximum speed, sudden stops and fast turns, to mimic an actual usecase. The results of the measurements were fairly consistent, and the worst-case results found from these measurements are shown in Figure 5.2.

**Analysis**

The performance of the regulator is generally on par for what is expected, and enough current is delivered to supply the motor controller and motors. There is a slight loss in power during the conversion, which matches up with the converters efficiency rating of 78-83%, and is to be expected.

[2] http://www.tek.com/oscilloscope/tds1000

The most interesting results in Figure 5.2 are the minimum voltage $V_{MIN}$ on the converters output while driving. There is a significant drop in voltage during the tests, and in the worst-case it reached 7.8V, almost a 68% decrease in voltage from the desired 24V. This could be a result of the regulator struggling to deliver enough current to meet the draw from the motors, however the data does not indicate any excessive spikes in current drawn. This may suggest that there is a more fundamental problem in the power system, and will be examined further in Section 5.1.3.

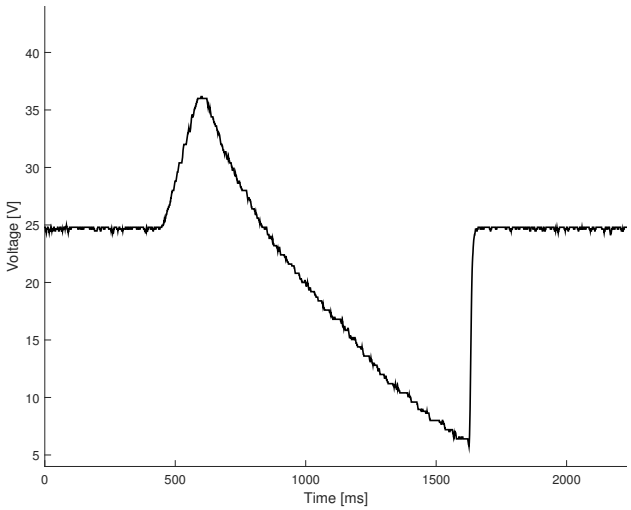
### 5.1.3 Motor Power System Performance

To further investigate the drop in voltage during the conducted tests, measurements were performed with the oscilloscope mentioned in Section 5.1.1. The graphs in Figures 5.3a to 5.3c shows voltage measured at the converters output (directly connected to the motor controllers input) during rapid deceleration from top speed, for high, medium and low acceleration settings, respectively. The tests were performed with the same setup as in Section 5.1.2, but only the free driving tests were performed as the robot needed to stay stationary during the measurements. The data in Figure 5.3a show a large spike in voltage just as the robot starts to decelerate, which continues rising for about 200ms, before dropping slowly over a period of 1000ms until it reaches a low point of close to 5V, and quickly rises back up to 24V. The results for the same test with a medium acceleration setting in Figure 5.3b has the same result, but the voltage spikes is significantly smaller, and during the test with a low acceleration setting in Figure 5.3c no significant change in voltage was observed.

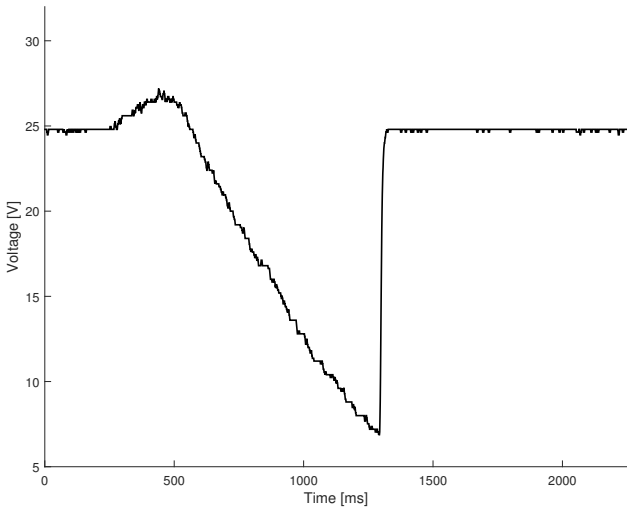
To attempt to isolate the source of the error additional tests were performed while using an external power supply instead of batteries. In the test presented in Figure 5.4a the external power source was connected trough the regulator, and in this case the result is similar to the tests with the batteries connected. Figure 5.4b shows the test performed with the external power supply connected directly to the motor controllers input, without the regulator. In this there is still a spike in voltage over 4-500ms, but it is not followed by a slow drop in voltage over 1000ms as in the previous test.

#### Analysis

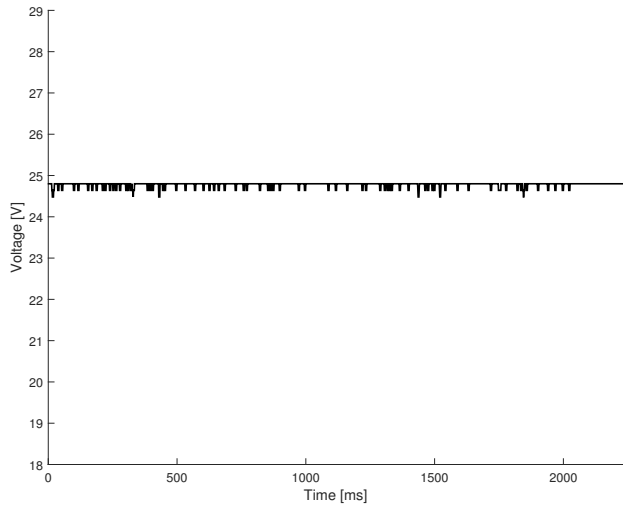
During decelerating the motors acts like generators, generating a spike in voltage as shown by the graphs. The protective diodes in the motor controller create a direct path from ground to voltage source trough the motor in order to protect the internal components from the voltage spike, as explained in Section 2.2.2. The spike is thus vented out into the circuit connected to the controller, which in this case is the regulator. This causes the voltage on the output of the regulator to exceed its overvoltage threshold. The regulator is turned off by its protective circuits, and remains disabled until the voltage has dropped back below the threshold. According to its manual the converter has a startup time of 2 seconds, however as evident by the graphs in Figures 5.3a and 5.3b this time is




(a) Max acceleration setting

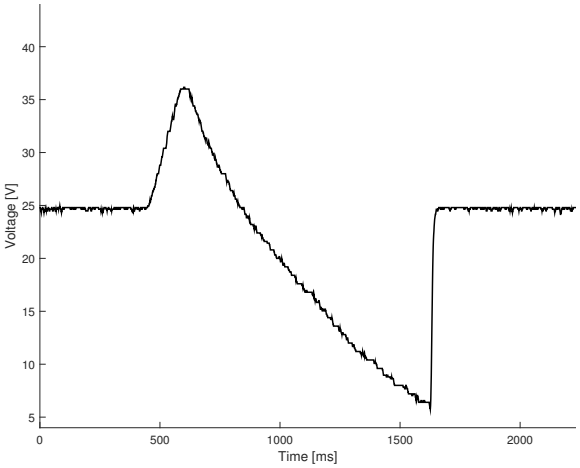



(b) Medium acceleration setting

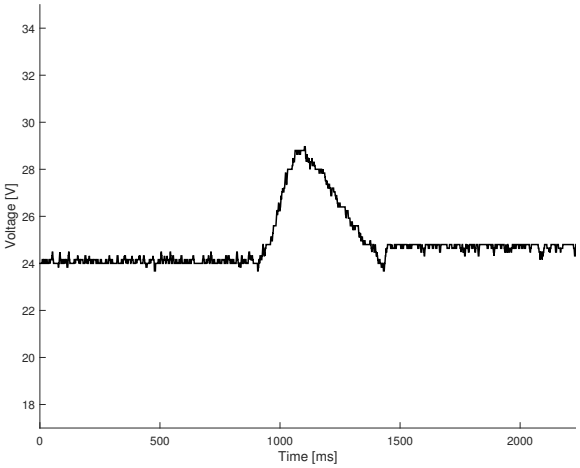



(c) Lowest acceleration setting

*Figure 5.3: Converter output voltage during deceleration from top speed to halt.*




(a) PS connected trough converter




(b) PS connected directly to motor controller

*Figure 5.4: Same test as in Figure 5.3, but with the motor controller power by an external power supply (PS).*

reduced to between 700-900ms during a hot-start. The voltage can reach a point as low as 5V before the converter has returned to an operational state and pulls it the voltage back up to 24V. During this low point there is barely enough power in the circuit to power the motor controller.

**Fix**

To handle this problem a simple protective circuit was implemented, illustrated in Figure 5.5. The converter output $V_i$ is protected by the unidirectional Schottky diode $D1$, that ensures that spikes in voltage are unable to reach the regulator. A Schottky was chosen as they have a low forward voltage drop and fast switching compared to regular diodes, improving the efficiency of the system. The transient-voltage-suppression (TVS) diode $D2$ gets rid of most of the overvoltage by attempting to pull any voltage above its cutoff threshold to ground. With optimal components this cutoff can be set right above 24V, to allow for some natural fluctuation in voltage.

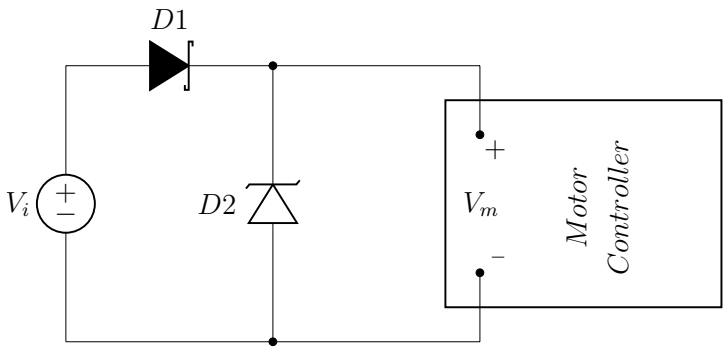
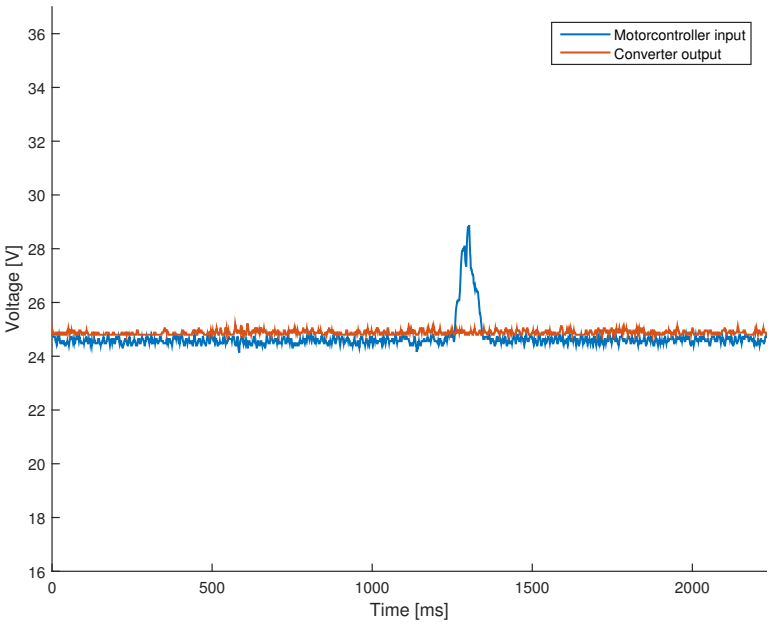



*Figure 5.5: The protective circuit for the voltage converter.*

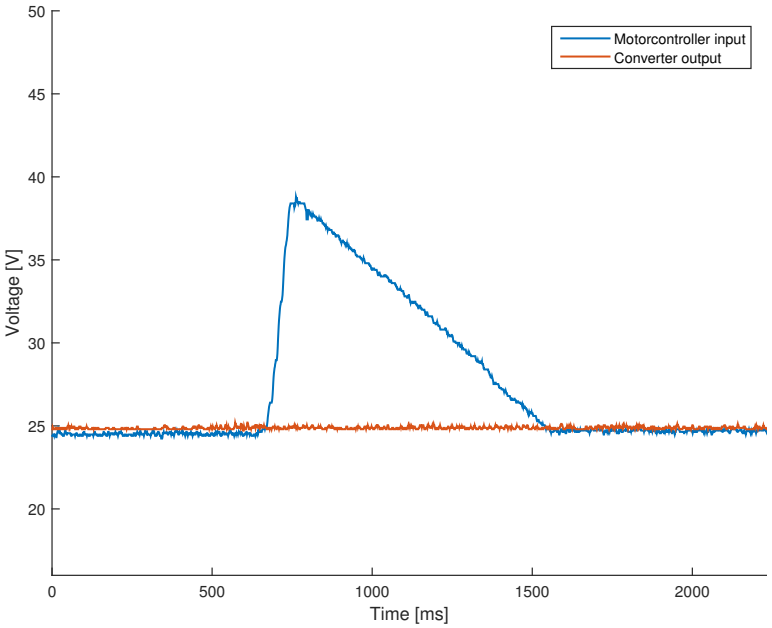
Figure 5.6b is the resulting voltage between the regulator and motor controller when the regulator is protected by a diode, and clearly shows why the rest of the protective circuit is necessary;

With the regulator blocked off by the diode, the voltage on the motor controller's input peaks at approximately 40V, which is over 166% of the controllers preferred voltage of 24V. These kinds of voltages can potentially blow the fuse on the motor controller, or even do damage to its components.

The TSV diode solves this by reacting to any sudden increases in voltage that surpasses its cutoff threshold, and allows the excess voltage trough to ground. Figure 5.6a displays the end result of the protective circuit; the output of the regulator is stable because it is protected by a diode, and the voltage spikes are significantly decreased. The voltages now peak at approximately 28V, which the motor controller can handle.




(a) Output voltage with only a Schottky diode protecting the converter




(b) Output voltage with the full protective circuit

*Figure 5.6: Regulator output after adding protective circuits.*

### 5.1.4 Servo Power System Performance

The servo power system was measured with the oscilloscope during testing in the same way as the motor power system. To measure the actual worst-case result, the measurements were performed while all servos were working. The result was consistent across all tests, and the result is displayed in Figure 5.7.

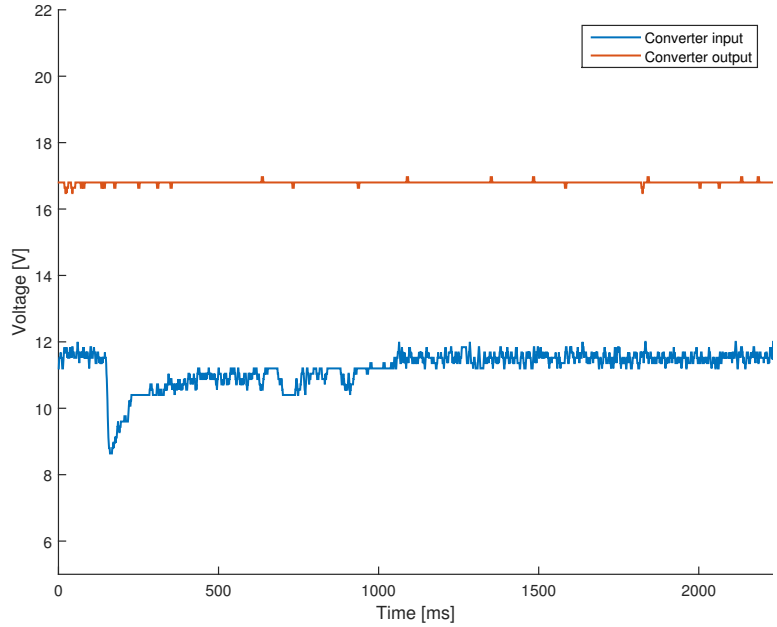



*Figure 5.7: Performance of the 9->18V to 12V converter during maximal load (all 4 servos working simultaneously)*

#### Analysis

The power delivery system for the servos worked without problems. Even during maximal load, with all four Dynamixel servos working and the beacon tower stepper motor standing still, the system was able to reliably deliver enough power without any significant spikes or dropds in voltage.

## 5.2 Drive System

This section contains the tests used to measure the performance of the drive system and related components, and the results of these tests.

### 5.2.1 Wheel Traction

Table 5.1 shows the results from a wheel comparison tests. The tests were done in cooperation with NAVSYS, and was performed by measuring the amount of weight each wheel could hold.

| Wheels | Devantech | Silicone |
|---|---|---|
| Width | 16$mm$ | 11$mm$ |
| Diameter | 124.8$mm$ | 101.6 |
| Static friction | 44.4$N$ | 20.44$N$ |

*Table 5.1: Wheel traction comparison.*

**Analysis**

These results show that the molding of new wheels have been successful, and the resulting wheels have over twice the traction of the previous wheels, even though they are significantly thinner. The lower diameter of the wheels is also an advantage, increasing the amount of torque produced by the motor and improving the positional resolution of the robot.

### 5.2.2 Backlash in Devantech gears

The gears included in the Devantech RD03 drive system has a problem with excessive amounts of backlash. The backlash value was measured using a digital precision angle gauge to be approximately 1.35°, which is a significant amount. The backlash is the same on both motors, which suggests that this is simply due to the tolerances of the gear parts, rather than faults during manufacturing. This is fortunate because as the same amount of backlash on both motors have less impact on movement accuracy. On straight motions the only impact is a slight increase or decrease in the distance traveled, and on rotational movement this will be canceled out, *if* the previous motion left each motor in the same position (i.e were a straight motion). This is usually the case while using a point-and-shoot type of movement scheme. The diameter of the final generation of wheels is approximately 102$mm$, and using Equation (2.2) this results in a backlash distance error of approximately 2.40 mm. An attempt was made to compensate for this backlash in the distance calculations in the control software, but the effects of the backlash turned out to be too unpredictable to compensate for. The attempts did more harm than good, as the worst-case accuracy was made worse when the compensation was added and the gears were not in the expected position relative to the center.

### 5.2.3 Speed and Acceleration

The MD49 motor controller have multiple settings for acceleration, as well as a regulator that can be enabled or disabled. The regulator uses feedback from the encoders to dynamically regulate the power output as required. There is no information available on how this regulator is implemented, and there is no way to tweak the output of the regulator. Tests were performed to test the impact of the regulator at various acceleration settings, and the results are shown in Figure 5.8.

### Tests

Simple experiments were performed to test of acceleration and deceleration of the robot while traveling in a straight line, and were performed by allowing the robot to reach maximum speed before being ordered to halt immediately, and the velocity of the robot is calculated by encoder readouts taken with a sample-rate of 250 Hz. The graphs displays average results over ten runs each, with standard deviation represented by the shaded area. The spike in velocity consistently appearing around the 2 second mark is an effect of a slightly increased delay between two samples during the sending of the stop signal, giving the impression of a sudden raise in velocity. This is hard to avoid as the stop signal has to be sent across the same serial bus as the encoder reading, but could possibly have been compensated for by time-stamping the samples.

### Result Analysis

The graphs are a clear display of the effect of the regulator, as well as the difference between the levels of acceleration. The regulator improves the response-time from a stop-signal is received until breaking starts, and results in the motors reaching a standstill quicker. The acceleration is also slightly smoother, resulting in better controlled movements. Figure 5.9 shows the relation between acceleration and stopping distance, and the effect of the regulator on this. It is interesting to note that on the highest acceleration setting the regulator increases the stopping distance, even though the deceleration starts at a lower velocity.

A unfortunate effect of the regulator is that it does not allow the motors to reach their top speed. This could possibly have been avoided if the gains of the controller could be tweaked, but it does not have a large impact on the robots performance as the top speed is not necessary anyway. However, these graphs are derived from encoder readouts, and does not account for physical effects like skidding and instabilities caused by too sudden movements. These effects were measured to have negligible effect on the distances traveled, as the high amount of traction of the wheels allows the robot to break effectively without any measurable skidding.

### Lack of Breaking Capabilites

The lack of a proper breaking mechanism, as seen in Figure 5.9, leads to problems with the accuracy of the system. Even with the highest acceleration setting, the stopping distance both with and without the regulator enabled is close to $2cm$. This is simply not good enough for high-precision movements. Lots of effort was put into tweaking the control-system to start the breaking exactly early enough to counteract the long stopping distance. This countermeasure was difficult to perfect as it depends on the types of movements performed, and even though this helped, it still was not as consistent as a motor system with breaking capabilities and a more sophisticated controller would have been.

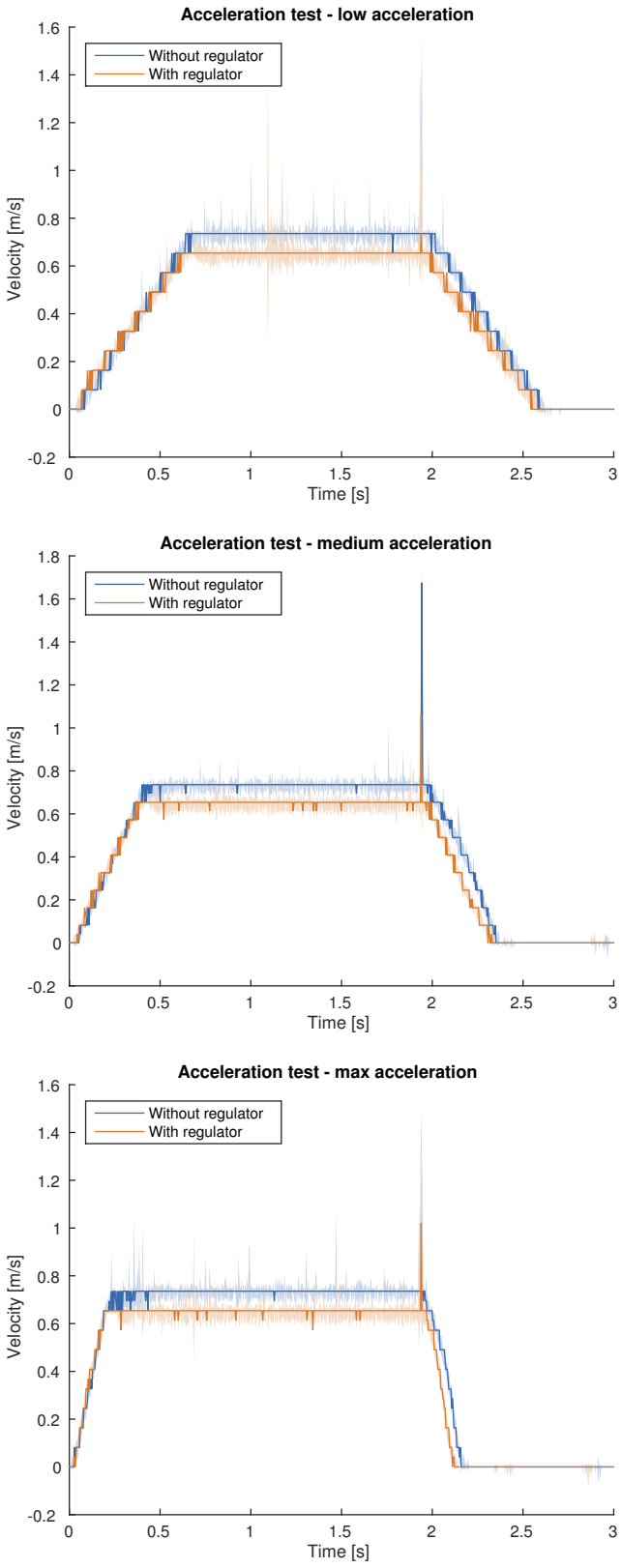



*Figure 5.8: Test results; acceleration and deceleration. The shaded area shows the standard deviation.*

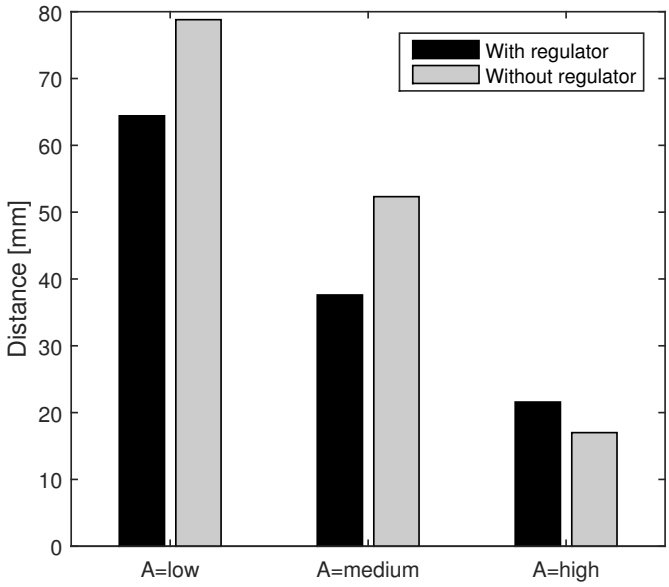



*Figure 5.9: Average stopping distance of motors from top speed, for each acceleration setting.*

## 5.3 Laptop Performance

This section contains performance tests of the laptop as a soft real-time system, which has been tested by comparing a version of the stock Linux kernel against a kernel custom-compiled with the PREEMPT_RT real-time patch.

### 5.3.1 Latency Tests

The test-suite used is the scheduling latency benchmarking tool Cyclictest, which is considered the de-facto test for measuring real-time performance in Linux[10]. Cyclictest is a fairly simple test which measures the time from a event occurring, for instance a program wanting to execute an instruction, to the start of actual work. This is averaged over millions of tries, and outputs the best-case, worst-case and average-case response times. It does not reveal any information about what exactly causes the latencies, but in this case we are simply concerned about the average- and best-case performance of the different kernels.

What Cyclictest actually measures is the latency of the kernel and CPU scheduler in response to a stimulus (external interrupts)[50]. It works by measuring the amount of time passed between the point when a timer is set by a thread expires (i.e. the thread "wakes up"), $t_1$, to the point $t_2$ when that thread is actually run by the CPU[56]. The result of this calculation is the latency of that wakeup delay $t_w$, and can be calculated as shown in Equation (5.1).

$$t_w = t2 - (t1 + sleep_time) \tag{5.1}$$

**Custom-Compiling linux-3.1 Kernels with PREEMPT_RT**

While it is theoretically possible to custom compile a current Linux kernel with the *PREEMPT_RT* patch, it is not officially supported. Attempts were made to compile a both a current 3.16.0-44 kernel and long-term-support (LTS) 3.14.36 kernels with patch, but were both times face with insurmountable problems during compilation. The available versions of the current kernels that can support this patch have had source-files modified by people outside the Linux project, and are not considered stable or even usable. The latest release of Ubuntu Linux that has a officially supported real-time kernel is Ubuntu 10.0f4 LTS. By default it ships with the 2.6.32-21-generic kernel, but installation of the alternative real-time kernel is simple and straight-forward. The latest real-time patched kernel available for Ubuntu 10.04 is linux-2.6.31-11-rt. This is the last kernel that officially supports the real-time, and has because of this been the kernel used as the real-time kernel in these experiments, while the stock Linux kernel is the more recent 3.14.36-LT kernel.

### 5.3.2 Test Results

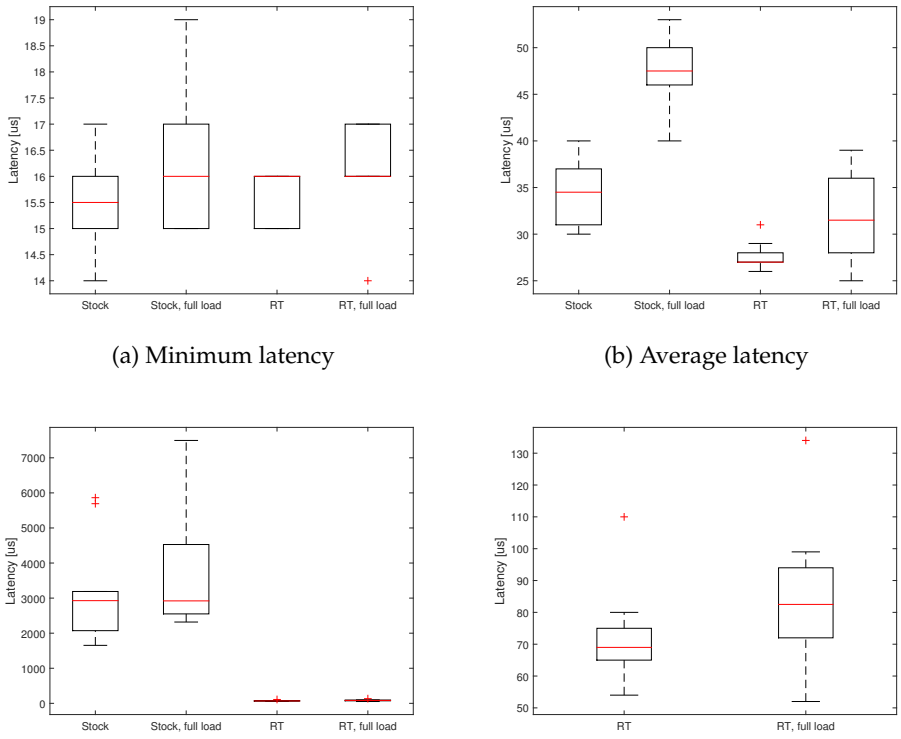
The tests were performed by running cyclictest with 5 threads, and collecting data from a large number of runs. This was done first with no load on the system except cyclictest itself, and then with a system load of close to 100%. This was done to test whether the amount of work the system had to perform affected the performance in regards to latency.

**Analysis**

| Test | Min | Avg | Max |
|---|---|---|---|
| Stock - no load | 15.5 | 34.6 | 3149.5 |
| Stock - load | 16.3 | 47.6 | 3712.0 |
| PREEMPT_RT - no load | 15.6 | 27.6 | 71.4 |
| PREEMPT_RT - load | 16.1 | 31.5 | 85.3 |

*Table 5.2: Mean result of each latency category across all tests. All results are in $\mu$.*

The results in Figure 5.10 are as expected. There is almost no difference in the test of minimum latency in Figure 5.10a, and both of the kernels have a low minimum latency, even during load. Minimum latency is also largely irrelevant for measuring the performance of the robot. The test of average latency is more interesting, and here the results vary significantly.




(a) Minimum latency

(b) Average latency

(c) Maximum latency

(d) Maximum latency, RT only

*Figure 5.10: Performance of stock kernel vs RT kernel*

As seen in Table 5.2, without load the mean average latency of the stock kernel is only 14% higher then for the RT-kernel, but during heavy load the difference rises to 51%.

But the most significant results are in the worst-case latency test, displayed in Table 5.2. Both with and without system load the worst-case performance of the RT kernel is significantly better than the stock kernel's. During load the mean latency of the stock kernel is 4267% higher than the RT. In order to properly see the RT plots of maximum latency, they are repeated in Figure 5.10d. The worst-case latency of the stock kernel is over $7ms$, which is large enough that it might affect the operation of the robot due to latencies, and would make the systems responsiveness slow.

The application of the Linux RT-patch has successfully turned the laptop running Linux into a low-latency system, that for the purpose of this project can be defined as soft realtime, as it minimizes the amount of deadlines missed. In this case the deadline will be the maximum amount of time to response, before the delay interferes with the operation of the control-system, potentially causing it to perform less than optimally in regards to accuracy of movement[**?**].

## 5.4 Communication

Fast and reliable communication between components is a key element of a distributed system, to make sure its behavior is deterministic. This is even more critical in a system where operations are expected to be performed in close to real-time. Especially the communication between the AI and control-system must be stable and reliable, as missed instructions may lead to problems completing tasks, or in the worst case allow the robot to crash into objects or opposing robots. To measure the performance and stability of the chosen implementation a number of tests have been performed, which will be presented in this section.

### 5.4.1 ZeroMQ Performance Tests

To make sure that a ZeroMQ-based communication implementation would robust and responsive enough for use for interprocess communication, tests were performed with the built-in ZMQ test-suite. The tests were performed on the Toshiba laptop while in use on the robot, with all regular programs and tools running to create a testcase as close to a competition as possible. The latency and troughput performance of ZeroMQ was tested both in inprocess and interprocess communication. Remote communication performance was not tested as is not a relevant usecase, because the robot will not be connected to a network during the competition. Additionally the performance of remote network communication depend on a too large number of factors, like the configuration of the network, to give useful data. The results are presented as boxplots in Figures 5.11 and 5.12.

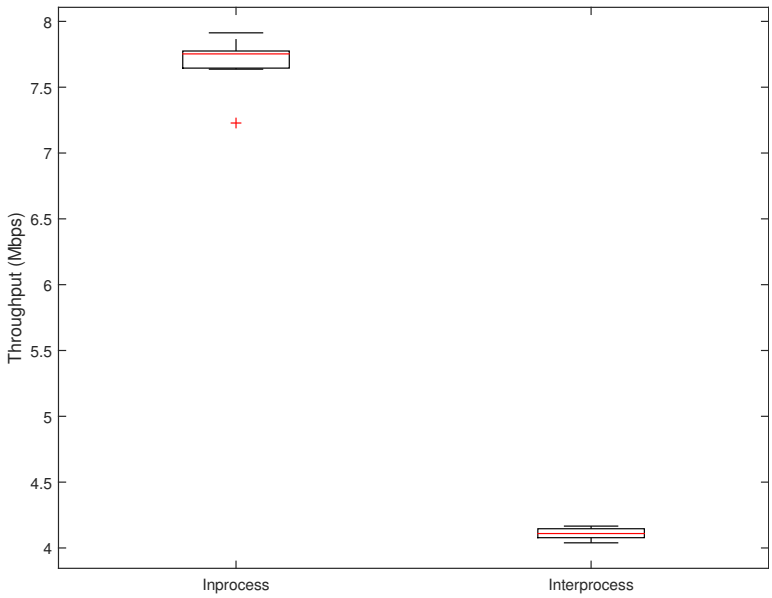



*Figure 5.11: Average ZeroMQ throughput. Message size: 1B, message count: 100000*

In the troughput test 100000 messages with a size of 1 byte were sent

trough the connection, and the troughput performance is the average amount of data sent trough the connection. Figure 5.11 is averaged over ten of these tests.

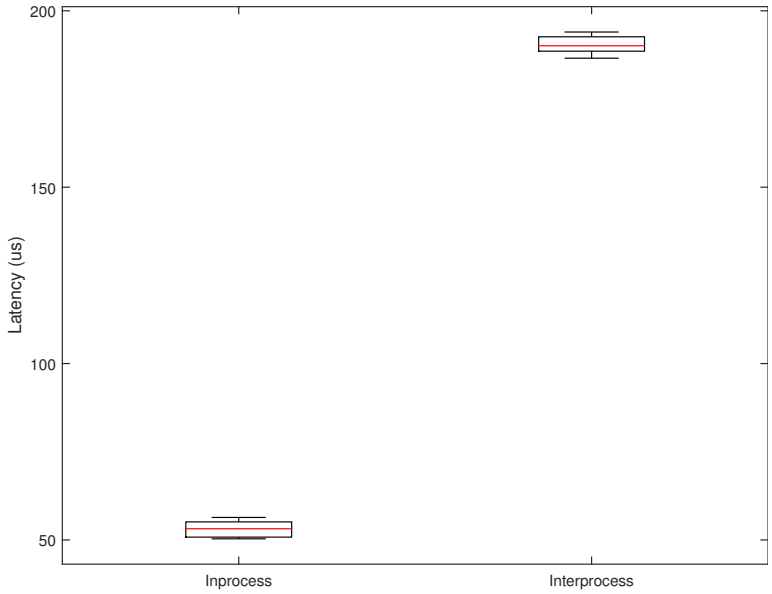
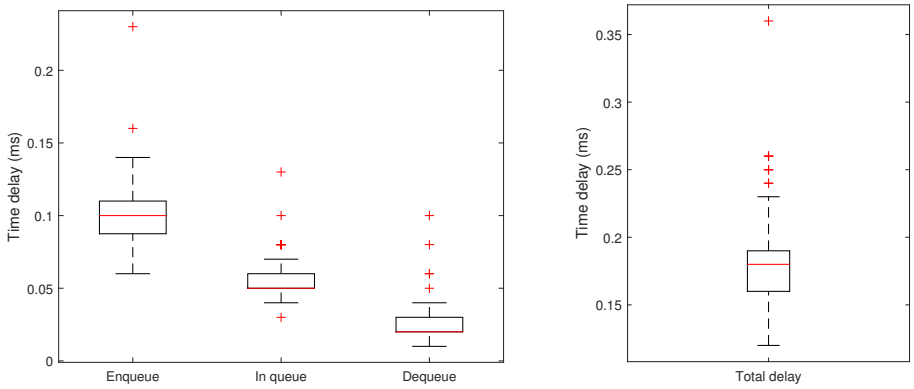



*Figure 5.12: Average ZeroMQ latency. Message size: 1B, round-trip count: 10000*

The latency benchmark is measure by sending a message with a size of 1 byte on a so-called *round-trip*, where the packet is sent back and forth trough the connection 10000 times. The latency performance is the worst-case time it takes for the packet to reach the other end. This test does not however cover the total cost of using this type of communication compared to having every part of the project integrated in one program.




(a) Factors split up

(b) Combined

*Figure 5.13: The total time spent from a command enters the control software through the ZMQ-server, until the system starts performing the action.*

To investigate this cost, another test were performed with the full control program, to assess the total delay from an command is sent by AI until the robot starts performing the action specified by that command.

Figure 5.13 present the results of these tests, Figure 5.13a shows the time expended from a command is received by the ZMQ server until it is stored in the internal command-queue by the communication-thread, the time expended while the command is waiting in the queue (as the first and only element), and the time from the command is dequeued untill the robot starts performing the action. Figure 5.13b combines these delays into the total amount of delay from a command enters the program trough the ZMQ server until the requested action is performed.

The results are reasonably consistent, though a few outliers exist, likely stemming from

**Analysis**

Throughput is not the most interresting performance benchmark for this project, as the amount of data sent will be relatively low. However, it is important to ensure that it infact is high enough to suffice. As the data illustrates, there is, obviously, a significant performance drop when going from inprocess to interprocess communication, both in respect to data troughput and latency. The total data rate of all the project's communication connections can easily be estimated as shown in Equation (5.2). The required bytes per second of throughput $B_{ps}$ is equal to the number of bytes per message $B_pm$, times the number of messages per seconds $M_{ps}$.

$$B_{ps} = B_{pm} * M_{ps} \tag{5.2}$$

The protocol used in this project is relatively simplistic, and the largest commmands sent are 6 bytes, with a 6 byte return message. AI sends commands to the control system at an absolute maximum rate of 200 commands per second, while NAVSYS requests position updates at most 10 times each second. This results in data rates of 2400 and 120 bytes per second for AI and NAVSYS respectively, and a total data rate for all communication in the project of 2.52 kB/s. This means that the troughput of a ZMQ interprocess communication implementation is almost 2000 times higher than what is required in this project, and even if the amount of communication would be massively increased this would likely never be an issue.

The latency benchmarks are more interesting, as the latency defines the responsiveness of the robot to commands from the AI and requests from NAVSYS.

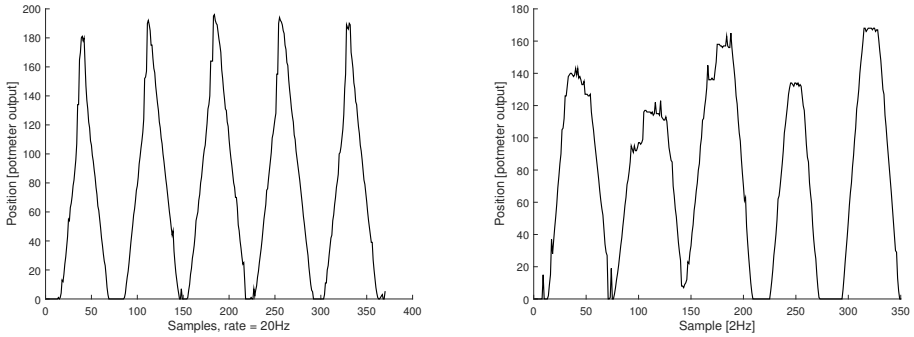
## 5.5 Mechanical Parts

The performance of the mechanical parts of the robots must be tested both to determine how well suited they are to solving their respective Eurobot

tasks.

### 5.5.1 First Gripper Iteration

The first iteration of grippers had problems with accuracy, and with developing enough force to hold the game elements in place during lifting operations. The accuracy problems are caused by the lack of a positioning controller. When faced with too much resistance, such as attempting to hold an object in a tight grip, the motor gives and the motor shaft shifts its position relative to the servo's internal position measurement. This can be observed in Figures 5.14a and 5.14b, where the motorshaft's position over time is measured using a potentiometer. The goal positions given to the servo is 0 and 190.




(a) Servo no resistance

(b) Servo with resistance

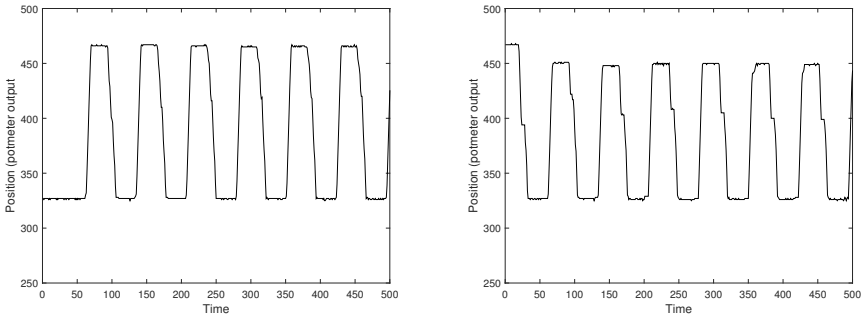
*Figure 5.14: Error measurements of gripper movement.*

### 5.5.2 Second Gripper Iteration

After replacing the servo with the more advanced Dynamixel servo and redesigning the mounting bracket, the same tests were performed. The results of these are displayed in Figure 5.15. The position range is changed because a different potentiometer was used (for practical reasons), otherwise the tests are completely identical.

### 5.5.3 Analysis

The first graph in Figure 5.14a shows the servo's movement with no obstacles or resistance. It is reasonable accurate, although the acceleration is neither as smooth, consistent or rapid as desired. In Figure 5.14b resistance to the gripper arms movement is introduced in the form of a solid object, in this case a eurobot stand. The goal position of the grippers is set slightly tigher than past the solid object in order to exert enough force to hold it tight. The motor gives under the pressure, and does not consistently hit the goal position. The torque output by the servo is also not high enough to hold objects tightly while being lifted. These problems makes the servo unfit for use for this application. As seen in the graph, the motor gives, but




(a) a (b) b

*Figure 5.15: Error measurements of gripper movement.*

the position measuring still assumes the motor reached its goal. This leads to a shift of the entire range of motion, which makes the grippers unable to actually grip anything, and may also lead to damage of the arms or other equipment. The servo is simply not strong enough, and combined with the lack of a position controller this makes it unfit for use in these kinds of robotic applications.

Figure 5.15 clearly demonstrated the advantage of using a servo with a advanced position controller. When these servo's are unable to reach the goal position, the motor does not give, but instead keeps exerting its maximum allowed amount of force towards the goal position. When the grip is relaxed the arms are able to consistently reach their starting position. Additionally the movements are smooth and swift, and the servo is able to develop enough forms to keep a tight hold of the game elements, even while lifting them.

# Chapter 6

# Discussion

## 6.1 General Discussion

This section will review the parts of the project that has not already been reviewed in the Results chapter, and discuss how well they worked as well as what could have been done differently.

### 6.1.1 Chassis

A while into the project we started to realize that opting to use the off-the-selves robot kit as a starting point for our robot was a mistake (Section 4.3). This descision did what it was supposed to do, and got the project started quicker than it would have with a more customized solution. However, the overall time spent modeling and creating custom parts and tweaking them to fit inside the shape of the curved and slightly irregular shapes of the robot probably led to a higher implementation time than what would have been the case for a custom chassis. Starting with a rectangular base with flat sides, potentially laser-cut from PMMA plastic like the top level mentioned in Section 4.3.3, would have made developing parts easier, and probably have led to a better designed robot. That being said, the parts from DFR robot kit themselves worked well. The plates were light, and easy to modify, while still being solid enough to resist bending or flexing during movement. The robot had good structural integrity, and there were no issues directly related to this during the competition.

### 6.1.2 Support Wheel

The instability and inaccuracy of the various support wheels tested have been a constant source of errors and frustration throughout the project. Figure 6.1 shows the ball wheel that was originally supposed to be used as the support wheel the robot. It was ordered from *RobotShop Inc.*, and would under normal circumstances have arrived in good time before the competition. Unfortunately the order was first delayed almost a month due to a miscommunication with the vendor, and when it was finally shipped it was lost during transport. As a backup a smaller version of the same wheel was acquired from another project. This wheel had a specified

load capacity of 15 kg [1], and should theoretically have worked fine as a replacement.


*Figure 6.1: The originally ordered support wheel.*

**Performance**

The replacement wheel performed well during the initial testing, and the robots movements seemed to be smooth and accurate. However, immediately before the team were about to leave for the competition, the wheel started showing signs of fatigue. The ball would sometimes get stuck on the bearings leading to the robot dragging the wheel along the floor instead of it rolling. This led to significant offsets in both rotational and translational movements.

**Alternatives**

With no alternative wheels available on short notice, an effort was made to improve the current wheel. It was cleaned for dust and dirt which had gathered in the lubricating oil around the ball bearings, and re-lubricated. This seemed to improve the performance drastically, and although it still did not perform perfectly, it was significantly better than before the maintenance was performed. Unfortunately this improvement lasted only temporary, and the problems with the wheel locking up started to reappear during the Eurobot qualification phase. At that stage it was impossible to get a replacement, so the only option was to lubricate it as much as possible, and reduce the speed slightly to reduce the severity of the errors.

[1] http://www.robotshop.com/en/dagu-ball-caster-16mm-20mm.html, Accessed on 2015-06-15

| Item | Units | Unit Price | Total Price |
|---|---|---|---|
| Devantech 24V Motor System (two motors and motor driver) | 1 | 2616 NOK | 2616 NOK |
| Dynamixel AX-12A | 4 | 356 NOK | 1424 NOK |
| Ansmann 14.8V 2250 mAh LiPo | 6 | 456 NOK | 2736 NOK |
| Isolated DC-DC Converter 24V | 1 | 525 NOK | 525 NOK |
| Isolated DC-DC Converter 12V | 1 | 488 NOK | 488 NOK |
| Various electrical components | - | - | 300 NOK |
| 3D printer plastic (Fortus 250mc) | 610g | 3 NOK/g | 1830 NOK |
| Laptop | 1 | 0 NOK | 0 NOK |
| Total | - | - | 9919 NOK |

*Table 6.1: Total amount spent on the robot.*

### 6.1.3 Spending

Table 6.1 shows the total amount of money spent on components for the robot. The most expensive components are the motors, servo motors and batteries, which make up most of the expenses. Some parts could have been acquired even cheaper, for instance some of the 3d-printed components could easily have been produced cheaper by creating them manually in other materials like plastic or wood. Still, with the total amount spent on components being less than 10,000 NOK, the goal of keeping the expenses low when compared to the most resourceful teams must be considered accomplished.

#### Additional Costs

Additional costs during the project that are less relevant to determine the cost of the robot has been included in its own table, in Table 6.2. These costs rely too heavily on things like team size, the teams origin country, the current Eurobot host country and other factors to be directly relevant to determine the actual cost of the project. These costs can however be interesting for other teams considering to apply, which is why they have been included.

## 6.2 The Competition

### 6.2.1 Preparation

In the time leading up to the international competition the focus was on making the tweaks necessary to allow the robot to at least pass the qualification round. The control-software was optimized for the tasks that

| Item | Units | Unit Price | Total Price |
|---|---|---|---|
| Eurobot Membership | 1 | 450 NOK | 450 NOK |
| Customs clearance & shipping Devantech | 1 | 784 NOK | 784 NOK |
| Flight tickets | 3 | 1100 NOK | 6600 NOK |
| Shipping Various Components[2] | - | 0 NOK | 0 NOK |

*Table 6.2: Additional costs related to the project.*

would be performed, and the acceleration and speed constants configured to allow for as precise and smooth movements as possible.

### 6.2.2 NAVSYS

Improvements and optimization of the triangulation-based navigation system was also attempted leading up to the competition, but when it still were not accurate or reliable enough to be useful in the last days before the team was leaving for Switzerland, it was decided that the best course of action was to cut our losses and not use the positioning system. The robot could navigate fine with only odometry based encoder feedback, but this meant that if the robot somehow got off course by hitting anything, it would likely continue to stray off course and not be able to complete the planned tasks. This also meant that we had no tracking of the opposing robots, and would not be able to see them until they were detected by our proximity sensors. This is discussed in further detail in the thesis on NAVSYS[46].

### 6.2.3 AI and General Strategy

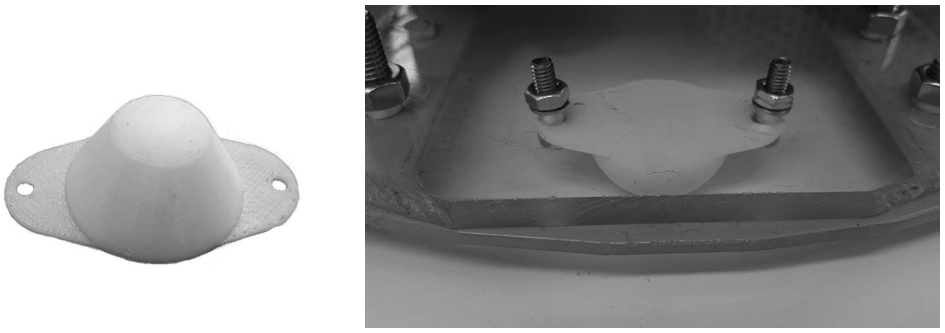
As NAVSYS would not be used except for close-range obstacle detection, some of the more sophisticated functions of the AI became largely irrelevant, such as planning actions based on where the opponents robots are located on the board. It also led to the adoption of a more conservative strategy, where the robot would focus most on the game elements on its side of the board, to reduce to probability of the robot hitting something and possibly losing its bearings. Some problems with the goal oriented action planning algorithm also led to that being scrapped in favor of a simpler, more direct planning system. This is discussed further in the thesis on AI[40].

#### Support Wheel Problems

Late during the last day of testing the support wheel started showing signs of fatigue, which was leading to slight positional inaccuracies.

[2]Most of the ordering was coordinated with the departements usual part purchases, so no additional shipping had to be paid.

This wheel had been used in previous projects, and the wheel bearings sometimes got stuck while moving in the same direction for too long. As NAVSYS had been considered too unreliable to be used during the competition[46], having accurate movements were critical for the robot to be able to navigate effectively on odometry alone. No alternative wheels were available on such short notice, so the wheel was attempted salvaged by adding lubrication, which seemed to work well, as the inaccuracies were drastically reduced. However, as dust and dirt started to gather in the lubrication, it became obvious that the attempt at fixing the problem had not been successful. As we had no replacement the only option was to attempt to mediate the effect the support wheels rotation had on movement as much as possible, by further reducing acceleration and top speed constants to avoid rapid movements that seemed to exaggerate the problem. This was somewhat successful, but meant that we had to lower our ambitions for scoring points, as the robot was too slow to perform its entire planned run.



(a) Support pillar (b) Mounted on the robot in place of the support wheel

*Figure 6.2: A solid pillar printed as a replacement of the support wheel.*

When the team got back from the competition an experiment was performed by printing a solid pillar, shown in Figure 6.2, as a replacement for the dysfunctional support wheel. This turned out to work perfectly on smooth surfaces, and would have been a better solution for the competition than the ball wheel that was used.

### 6.2.4 Matches and Results

The rules specify that each team shall be granted fifteen minutes of preparation. However, before the first match on the first day of the competition, which our team unfortunately were selected to participate in, no information were given to any teams until 2 minutes before the matches started. The competition was a part of a highly produced live-stream with a tight schedule, and no delays were accepted from the teams, so not further preparation time was granted. This led to a incredibly hectic preparation phase. A consequence of this was that we did not inspect the game board well enough before the match started, and one of the game pieces was placed slightly out of position. That error potentially cost us four points

from a popcorn cup with four popcorns inside, that fell as we attempted to grab it from its slightly offset position.

#### Points

| Task | Number | Points Each | Bonus | Total |
|---|---|---|---|---|
| Spotlights | 5 | 3 | 0 | 15 |
| Clapperboards | 2 | 5 | 0 | 10 |
| Stair climbing | 0 | 15 | 0 | 0 |
| Red carpet | 0 | 4 | 0 | 0 |
| Popcorn | 0 | 4 | 5 | 0 |
| Total | - | - | - | 25 |

*Table 6.3: Points scored in the first match.*

Despite all of the mentioned problems, the robot managed to do a large part of its planned route, and even though we lost the match to a superior opponent, we ended up with an acceptable score of 25 points, distributed as shown in Table 6.3.

#### Proximity Sensor Miscalibration

Another problem detected at the start of the second match was that the proximity sensors were not properly calibrated for the official track, and the sensor on the back of the robot was detecting the beacon support tower directly behind the starting area, preventing the robot from moving at the start of the game[46]. After a few seconds it did however start, by some lucky coincidence, perhaps a single reading of the sensor reported a just high enough distance for the robot to be allowed to start. This should not have been a problem at all, as the robot should only check the rear proximity sensor while reversing or decelerating. The robot still managed to finish the same tasks as in the first match, however our team received an official warning after the match because our robot tools continued moving for a second after the time was up. This was due to a slight error in the way time was handled in the AI combined with the slow start due to the sensor miscalibration, and was fixed before the next match.

#### Subsequent Matches

The other matches were largely uneventful, and even though improvements were attempted they all ended with the same points as the first match.

Table 6.4 shows the result of each of the matches. As evident by our scores, our robot completed the same tasks in all runs, but in the second

| Our Score | Opponents score | Opponent | Country |
|---|---|---|---|
| 25 | 33 | Druinobot | Canada |
| 25 | 97 | ESEO-ANGERS | France |
| 25 | 11 | Natural Born Killers | United Kingdom |
| 25 | 19 | OUCHIE-TIME | Tunisia |
| - | - | Forfeit | - |
| - | - | Forfeit | - |

*Table 6.4: Result of all matches.*

and third matches the robot was also very close to scoring additional points from the popcorn cups, but they fell over while being deposited. With some luck, or a slightly better design of the gripper system, these cups could potentially have provided some extra points. There was also a slight problem with the range sensors detecting the popcorn-dispensers along the side of the game board as opponents, shutting down movement and cutting our runs short, potentially costing us up to 6 points from 3 stands.

#### Final Ranking

Our position after four matches was 26th of the 36 teams that passed the qualification phase, with a 100 points scored total. We unfortunately had to forfeit our last two matches, supposed to take part in the middle of the day on Sunday, as our travel back to Norway was booked rather early on Sunday. This caused us to drop from place 26 to 31, which in the bigger picture is largely insignificant. For the remainder of this chapter we will used 26th as the final ranking, as that better reflects the actual performance of the robot. All in all our team deemed the result as acceptable, considering it is the first year UiO takes part in the competition.

### 6.2.5 Things Learned During the Competition

As mentioned in Section 6.2.2 we decided not to use the global positioning system based on beacon triangulation in the competition, and after inspection of other robots and discussion with the other teams, we realized that attempting to implement such a system in the first place had been a mistake. Even though the Eurobot rules allocate an entire chapter to the description of the beacon support platforms and how the triangulation system can be built, not a single team out of the 36 participating actually used systems like this at all. Discussions with the other teams revealed that they all had chosen not to implement it for the same reasons we had come to realized; it is not easy to make systems accurate and reliable enough to actually be useful, as encoder odometry already is incredibly accurate with good locomotion systems, especially if some form of calibration actions are

implemented. Calibration of positioning can for instance be performed by measuring distance to objects with a known location, such as the walls. This is not a foolproof measure, but it reduces the likelihood of the robot losing track of its position.

A global navigation system is helpful if the robot hits obstacles and completely loses its bearings, but the amount of resources that has to be put into such a system to make it good enough is far too much, and the resources are better spent improving the systems that *prevent* the robot from actually hitting said obstacles. These problems are discussed in detail in the thesis on NAVSYS[46].

#### Secondary Robots

One misconception we had after our research phase was that the secondary robots would be tested as strictly as the main robots during qualification, as they have all the same constraints and obligatory features[20]. The development of the secondary robot was halted as a result of this, as mentioned in Section 4.12.3. This assumption turned out wrong, and the referees demands for the secondary robots turned out to be much more lenient than the main robots, both in relation to following building standards and in relation to crash-avoidance testing. With this taken into consideration, the secondary robot could have been finished with significantly less effort and resources than originally anticipated. This could have made it possible to complete the stair climbing and red carpet task, potentially doubling our point collection. Figure 6.4 contains two examples of robots that partook in the competition, to illustrate how simple and uncomplicated some of the robots that passed the qualification phase were.

### 6.2.6 Tournament Ranking

NTNU's results in Eurobot is an obvious unit of measurement for the success of our project, as it is their budgets that were used as an example in Section 1.3. Figure 6.3 shows an overview of their results from they first were a part of the competition in 2000 up to the last time they participated in 2012. As the graph shows they have generally performed reasonably well, with 2005 and 2009 standing out with 11th place out of 50 teams and 4th place out of 43 teams, respectively. But they have also struggled in some competitions, such as the 2010 and 2011 competitions where they did not pass the qualification round.

As mentioned in Section 6.2.4, the team deemed our 26th place out of 36 teams as acceptable. Even though it is below the average result of the NTNU teams, we feel that the results must be considered reasonable, both due to the vast difference between the size of the budgets, and the fact that all their projects have had the benefit of NTNU's previous experience in the Eurobot competition.

Our robot did finish all of its matches without notable incidents, and scored an acceptable amount of points according to the expectations.

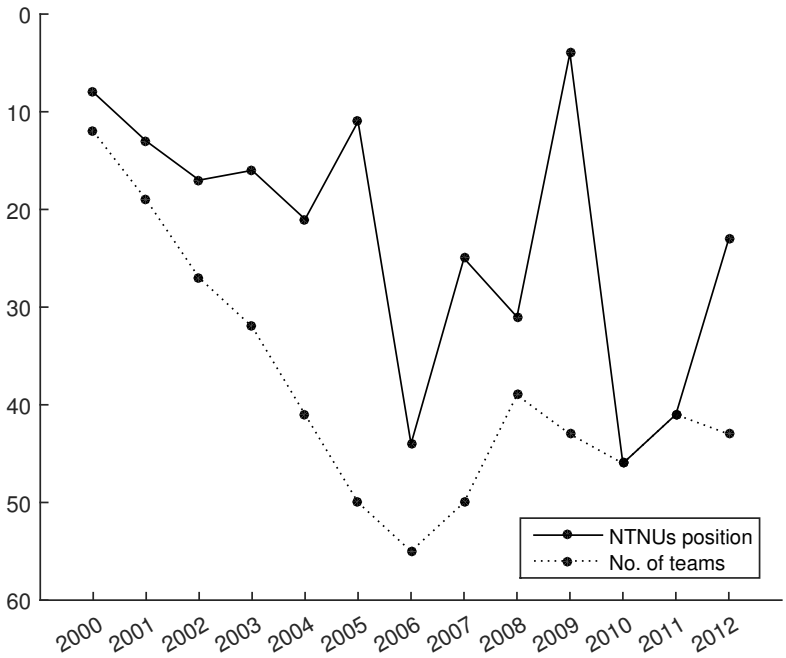



*Figure 6.3: Overview of NTNU's results, compared to the number of teams participating[24].*

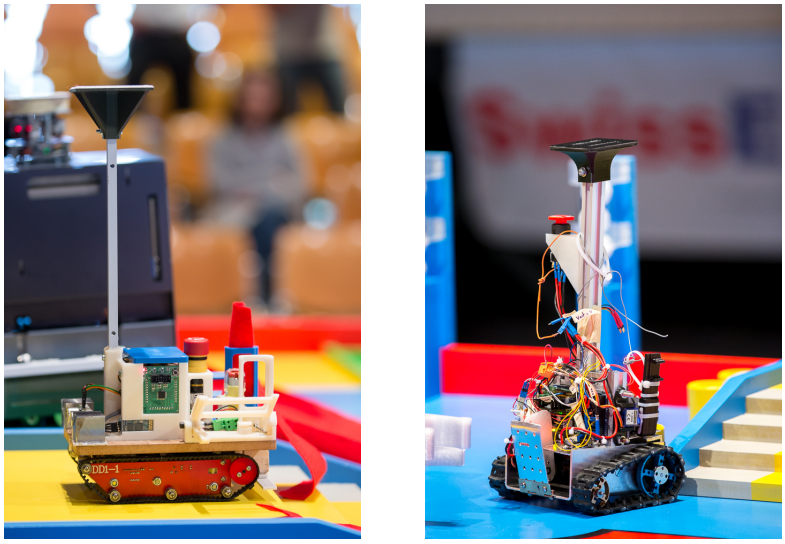


*Figure 6.4: Examples of mechanically simple secondary robots, from this years competition.*[3]

[3]Source: http://www.robots15.ch/en/photos_videos, Accessed on 29-06-2015

## 6.3 Conclusion

The main goal of the project was to create a low-cost robot, that is able to perform well in the Eurobot competition. This goal has been reached, as a functional autonomous mobile robot has been designed, that is capable of solving the tasks it was designed to solve. This has been shown during the international competition in Switzerland, where the robot passed the qualification round, and successfully completed all of its matches. In all matches an acceptable amount of points were scored, and two of the four matches were won. As discussed in Section 6.2, these results are on par for what can be expected by team in their organization's first participation in the Eurobot competition.

The other part of this goal, that was not accomplished, was the more ambitious goal of being able to compete effectively against the teams with larger amounts of resources. Even though the robot and the team performed well for a first time contestant, and also relative to our own expectations, our robot was nowhere near challenging the teams at the top of the table, as evident by our final ranking (Section 6.2.4).

The organization of the project worked well, and the split of the project into three tasks was considered successful. Even though not all parts of the systems ended up being used during the international competition, every member of the team contributed in roughly equal amounts to the project, and got a large enough area of focus to base their master these on.

The conclusion is that the robot has successfully been designed and constructed on a low budget of under 10,000 NOK that it is fully functional, and can compete, score points and even win matches in the Eurobot competition. However, in order to effectively compete shoulder to shoulder with the best teams and contest a spot at the top of the result table, a larger budget and potentially a larger team is required.

## 6.4 Future Work

This section contains things that could have been improved or added on the robot, as well as things that could be done differently for future Eurobot projects on UiO.

### 6.4.1 Better Mechanical Solution for Stacking

Seeing all the other solutions at the competition provided inspiration for how the gripper system could have been improved. Many of the teams, especially the teams with a large number of contributors, had fast, well-functioning mechanical systems based on pneumatic actuators, which allowed them to move and stands with extreme quickness and precision. There were also multiple better solutions than ours for collecting popcorn cups, like storing them internally in the robot, grabbing hold of the edge

of the cup and dragging it, or simply using a gripper system separate from the stand-collecting system.

### 6.4.2 Better Motors

To be able to compete against the best teams, the motors and motor controller used on this project (discussed in Section 4.4.4), are simply not good enough. As reviewed in Section 5.2.3, the acceleration and top speed are fine, but the lack of proper breaking capabilities severly impacts the accuracy and reliability of the drive system.

### 6.4.3 Finishing the Secondary Robot

Having a secondary robot, even a less sophisticated one, would have significantly increased the maximum amount of points our team could have aqcuired. As mentioned in Section 6.2.5, they could have been built far easier than we had assumed based on the Eurobot regulations, and with this information in hand it would probably have been focused more on finishing that robot.

### 6.4.4 Adding a HMI Board

Some teams had a Human-Machine Interface (HMI) board for adjusting their robot's settings on the fly[18]. This can be a large advantage, as no pc connection means that it is easier to debug the robots quickly in between matches, and also means that some parameters (such as task prioritization) can be optimized for each opponent. One example of this was a team that had one plan for teams with two robots, and a separate plan for teams with only one robot, that allowed their robots to be more aggressive in their movements over the board.

### 6.4.5 Recommendations for Future Projects

Working with the Eurobot has been a great experience, and in the end we were able to create a functional robot. However, if UiO is interested in doing more Eurobot projects in the future, it should be considered adding more members to the team, in order to improve the amount of resources and time possible to spend on each part of the project. If the teams wants to contests for a spot on the the of the result table, more work has to be put into aquiring sponsors and funding, as a higher quality of components is required for performing well enough to compete against the best teams. Especially the motors and

# Bibliography


[1] iRobot company information - history. http://www.irobot.com/About-iRobot/Company-Information/History.aspx, 2015. Accessed on 2015-07-29.

[2] R. Ackermann. *Observations on Ackermann's Patent Moveable Axle, for Four Wheeled Carriages: Containing an Engraved Elevation of a Carriage, with Plans and Sections, Conveying Accurate Ideas of this Superior Improvement ...* R. Ackermann, 1818.

[3] Silas FR Alves, Humberto Ferasoli Filho, Renê Pegoraro, Marco AC Caldeira, Joao M Rosário, and Wilson M Yonezawa. Educational environment for robotic applications in engineering. In *Research and Education in Robotics-EUROBOT 2011*, pages 17–28. Springer, 2011.

[4] Dimitrios Apostolopoulos. Analytic configuration of wheeled robotic locomotion. Technical report, Carnegie Mellon University, 2001.

[5] Karl Johan Åström and Tore Hägglund. *Advanced PID control*. ISA-The Instrumentation, Systems, and Automation Society; Research Triangle Park, NC 27709, 2006.

[6] V.S. Bagad. *Mechatronics*. Technical Publications, 2009.

[7] Jeremy H Brown and Brad Martin. How fast is fast enough? choosing between xenomai and linux for real-time applications. In *12th Real-Time Linux Workshop (RTLWS'12)*, pages 1–17, 2010.

[8] Luca Caracciolo, Alessandro De Luca, and Stefano Iannitti. Trajectory tracking control of a four-wheel differentially driven mobile robot. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 4, pages 2632–2638. IEEE, 1999.

[9] Petr Čermák and Jozef Kelemen. An attempt to teaching programming with robots. In *Research and Education in Robotics-EUROBOT 2011*, pages 78–87. Springer, 2011.

[10] Felipe Cerqueira and Bjorn Brandenburg. A comparison of scheduling latency in linux, PREEMPT RT and LITMUSrt. Technical report, Max Planck Institute for Software Systems, 2013. https://www.mpi-sws.org/ bbb/papers/pdf/ospert13.pdf, Accessed on 2015-03-21.

[11] Wikimedia Commons. Electric motor cycle 2. https://commons.wikimedia.org/wiki/File:Electric%5Fmotor%5Fcycle%5F2.png, 2006. Accessed on 2015-01-23.

[12] Wikimedia Commons. Backlash. https://commons.wikimedia.org/wiki/File:Backlash.svg, 2010. Accessed on 2014-11-03.

[13] Wikimedia Commons. Meacnum rad. https://commons.wikimedia.org/wiki/File:Meacnum_Red.png, 2013. Accessed on 2015-07-17.

[14] Wikimedia Commons. Swivel caster with poly brake. https://commons.wikimedia.org/wiki/File:Swivel_Caster_with_Poly_Brake.jpg, 2013. Accessed on 2015-07-19.

[15] PC/104 Embedded Consortium et al. Pc/104 specification. *http://www.pc104.org/pc104_specs.php*, 2003.

[16] Solidworks Corp. Solidworks. http://www.solidworks.com/, 2015. Accessed on 2015-08-02.

[17] Andrzej Dworak, M Sobczak, Felix Ehm, Wojciech Sliwinski, and P Charrue. Middleware trends and market leaders 2011. In *Conf. Proc.*, volume 111010, 2011.

[18] Raimund Edlinger, Michael Zauner, and Walter Rokitansky. Wheeled mobile autonomous robot for eurobot 2011. In *2nd International Conference on Robotics in Education, Vienna, Austria*, pages 147–150, 2011.

[19] Adam Clark Estes. 13 fascinating farming robots that will feed humans of the future. http://gizmodo.com/13-fascinating-farming-robots-that-will-feed-our-future-1683489468, March 2015. Accessed on 2015-07-29.

[20] Eurobot. Robomovies. http://www.eurobot.org/attachments/article/31/E2015_Rules_EU_EN_final.pdf, 2015. Accessed on 2015-1-23.

[21] Ivar Fjeldheim. Autostore vds concept according to vds cea 4001. http://autostoresystem.com/vds-concept-for-the-protection-of-hatteland-autostore-warehouse-systems-with-a-sprinkler-system-according-to-vds-cea-4001, June 2015. Accessed on 2015-07-29.

[22] Philippe Gerum. Xenomai - implementing a rtos emulation framework on gnu/linux. *White Paper, Xenomai*, 2004.

[23] Kristin Holst Haaland. Eurobot 2010: Navigasjonssystem. Master's thesis, Norwegian University of Science and Technology, 2010.

[24] Are Halvorsen, Sindre Røkenes Myren, and Andreas Hopland Sperre. Eurobot NTNU 2012: Treasure island. Master's thesis, Norwegian University of Science and Technology, 2012.

[25] Joe Hiemenz. 3D printing with FDM: How it works. *Stratasys Inc*, 1:1–5, 2011.

[26] Joe Hiemenz. FDM and PolyJet 3D printing: Determining which technology is right for your application. *Stratasys Inc*, 1, 2014.

[27] Pieter Hintjens. *ZeroMQ: Messaging for Many Applications*. O'Reilly Media, Inc., 2013.

[28] Mats Høvin. INF4500 lecture 6 - h-bridge. http://heim.ifi.uio.no/matsh/inf4500/lec/0.php?s=17\&l=6\&d=0, 2015. Accessed on 2015-07-21.

[29] Brian Huffman. Efficiency and power characteristics of switching regulator circuits. Technical report, Linear Technology, 1991.

[30] Honeywell Inc. Hall effect sensing and application. *Honeywell Micro Switch Sensing and Control*, 2002.

[31] Stratasys Inc. Fortus 250mc. http://www.stratasys.com/3d-printers/design-series/fortus-250mc, 2015. Accessed on 2015-1-23.

[32] Mari Iwata. Robots star in cleanup of japanese nuclear plant. http://www.wsj.com/articles/SB10001424052702304732804579421003952552062, March 2014. Accessed on 2015-07-29.

[33] Michael A Johnson and Mohammad H Moradi. *PID control*. Springer, 2005.

[34] M. Tim Jones. Inside the linux 2.6 completely fair scheduler. Technical report, IBM Corporation, 2009. http://www.ibm.com/developerworks/library/l-completely-fair-scheduler/l-completely-fair-scheduler-pdf.pdf, Accessed on 2015-02-10.

[35] David Kaiser. Fundamentals of servo motion control. *Parker Compumotor*, 11, 2001.

[36] Ola Kinnander. Rise of the lawn-cutting machines. http://www.bloomberg.com/bw/articles/2012-10-25/rise-of-the-lawn-cutting-machines, October 2012. Accessed on 2015-07-29.

[37] Gunnar Kjemphol. Eurobot 2007. Master's thesis, Norwegian University of Science and Technology, 2007.

[38] Christian Wegger Kjølseth and Øystein Wergeland. Eurobot 2009. Master's thesis, Norwegian University of Science and Technology, 2009.

[39] Roger Klemm and Glenn Reeves. Vxworks on the mars exploration rovers. Technical report, Pasadena CA: Jet Propulsion Laboratory and National Aeronautics and Space Administration, 2005.

[40] Bendik H. Kvamstad. Reinforcement learning as a decision making strategy for a mobile robot. Master's thesis, University of Oslo, 2015.

[41] Thomas L. Floyd and David M. Buchla. *Electronics Fundamentals*. Pearson, 2010.

[42] Robert McGill, John W Tukey, and Wayne A Larsen. Variations of box plots. *The American Statistician*, 32(1):12–16, 1978.

[43] Mark Mitchell, Jeffrey Oldham, and Alex Samuel. *Advanced linux programming*. New Riders, 2001.

[44] Bill Nitzberg and Virginia Lo. Distributed shared memory: A survey of issues and algorithms. *Distributed Shared Memory-Concepts and Systems*, pages 42–50, 1991.

[45] Optek Technology Inc. *Photologic® Slotted Optical Switch OPB960-990 Series*. http://optekinc.com/datasheets/opb960-990_series.pdf, Accessed on 2015-04-05.

[46] Andre Kramer Orten. A low-cost indoor positioning system - eurobot (2015). Master's thesis, University of Oslo, 2015.

[47] Carolyn Renee Pals and John Newman. Thermal modeling of the lithium/polymer battery. Technical report, Lawrence Berkeley Lab., CA (United States), 1994.

[48] Gregg W. Podnar. The uranus mobile robot. *Autonomous Mobile Robots: Ann. Rep*, 1985.

[49] Steven Rostedt and Darren V. Hart. Internals of the RT patch. In *Proceedings of the Linux Symposium, Volume 2*, pages 161–172. Linux Symposium, 2007.

[50] Frank Rowand. Using and understanding the real-time cyclictest benchmark. *Embedded Linux Conference*, 2013. Sony Mobile Communications.

[51] Bruno Scrosati and Jürgen Garche. Lithium batteries: Status, prospects and future. *Journal of Power Sources*, 195(9):2419–2430, 2010.

[52] Richard M. Stallman. Linux and the GNU system, 2014. Accessed on 2015-06-02.

[53] Stian J. Søvik. Eurobot 2010 fremdriftssystem. Master's thesis, Norwegian University of Science and Technology, 2010.

[54] Linus Torvalds et al. The linux operating system. *See www. linux. org*, 1999.

[55] Tim Wescott. Pid without a phd. *Embedded Systems Programming*, 13(11), 2000.

[56] Clark Williams. Finding real-time linux kernel latencies, 2010. Accessed on 2015-06-02.

[57] Xuesu Xiao and William (Red) L. Whittaker. Energy considerations for wheeled mobile robots operating on a single battery discharge. Technical Report CMU-RI-TR-14-16, Carnegie Mellon University Robotics Institute, 2014.

[58] Masaki Yoshia, Ralph J Brodd, and Akiya Kazawa. *Lithium-Ion Batteries*. Springer, 2009.

# Appendices

# Appendix A

# Code

All the code used in the project can be found at https://github.com/eivinwi/eurobotuio, the control program is located in the *Program* folder. Most of the code will not be included here, as even though a lot of work was put in to writing and optimizing it, the resulting code is not that interesting to reads.

## A.1 ZeroMQ Server

```
void zmqServer(std::string addr) {
    com_running = true;
    // Prepare context and socket
    zmq::context_t context (1);
    zmq::socket_t socket (context, ZMQ_REP);

    // Bind to to address on the form: <protocol:ip:port>
    // Example usage:   <tcp://localhost:5555
    socket.bind (addr.c_str());

    while (true) {
        zmq::message_t request;
        // Wait for next request from client
        socket.recv (&request);

        // Translate from message to string
        std::string recv_str = std::string(static_cast<char*>(request.data())
        std::string reply_str;

        std::vector<int> args = extractInts(recv_str);
        int num_args = args.size();

        switch(args[0]) {
            /*
             * SWITCH-CASE finding the action to perform,
             * and the correct response to put in reply_str
             */
```

```
        }

        //create and send reply message
        zmq::message_t reply(reply_str.length());
        memcpy ((void *) reply.data (), reply_str.c_str(), reply_str.
        socket.send (reply);
        usleep(1);
    }
    com_running = false;
}
```

## A.2 ZeroMQ Test Client

```
#include <zmq.hpp>
#include <string>
#include <sstream>
#include <iostream>

int main (int argc, char *argv[])
{
        // Prepare the context and socket
        zmq::context_t context (1);
        zmq::socket_t socket (context, ZMQ_REQ);

        // Connection to ZMQ server running on port 5555 locally
        socket.connect("tcp://localhost:5555");

        std::string mystring = "Hello, server!";

        // Do 10 requests, waiting each time for a response
        for (int request_nbr = 0; request_nbr < 1; request_nbr++) {
                zmq::message_t request( mystring.length()+1 );
                memcpy ((void *) request.data (), mystring.c_str(), m
                socket.send (request);

                // Wait for reply
                zmq::message_t reply;
                socket.recv (&reply);
                std::string recv_str = std::string(static_cast<char*>
                std::cout << "Received reply: " << recv_str << std::e
        }
        return 0;
}
```

# Appendix B

# Poster

Each team had to create a poster to hang in their team's designated areas during the competition. The following page show ours. The poster had a deadline long before the actual competition, so the robot was still in an early stage of development.

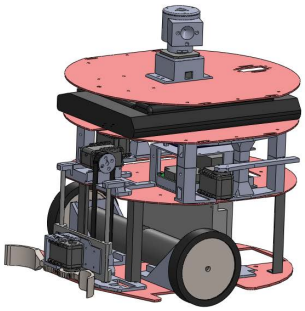
UiO : Department of Informatics
University of Oslo

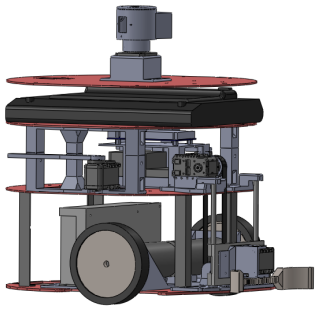
# Balotelli's Support Group

## Robot Design

Most of the parts for our robot are custom-made by 3D-printing.

The parts are modelled in Solidworks and printed on a Fortus 250mc. This printer uses a technique called Fused Deposition Modelling, and the parts are created by printing layers of molten plastic that fuses together.





## Motor Systems

**1. Devantech 24V 49:1 Motors**

| | |
|---|---|
| Voltage | 24 V |
| Torque | 1.6 N.m |
| Speed | 122 rpm |
| Rated Current | 2. 1A |
| Encoders | Hall Sensors |

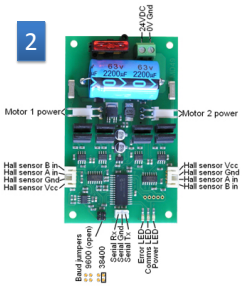
**2. Devantech MD49 Motor Driver**

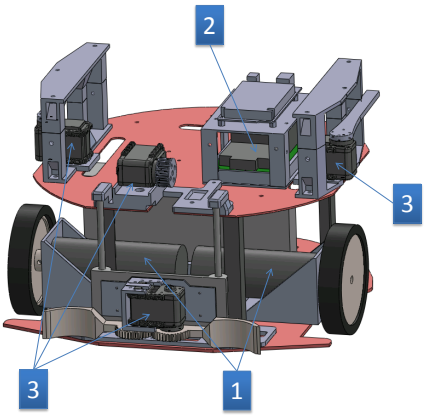
| | |
|---|---|
| Communication | Serial TTL |
| Voltage | 24 V |
| Rated Current | 5 A |

**3. Dynamixel AX-12A Servo**

| | |
|---|---|
| Voltage | 12V |
| Torque | 1.5 N.m |
| Speed | 59 RPM |



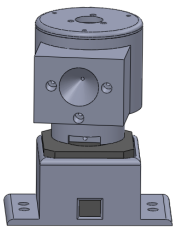


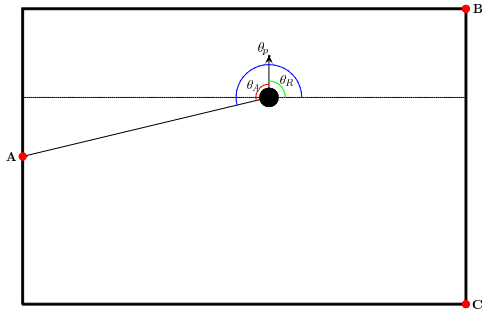




## Navigation

The navigation is based on two separate systems; odometry based on encoder readings, and an infrared beacon system.

The robot uses odometry to find an approximate local position, while the beacon system is used to find the global position of the robot with respect to the playing field. An extended Kalman filter is used for the sensor fusion.

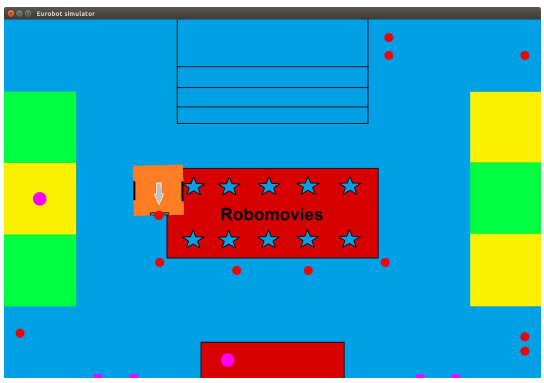





## AI

The AI uses a dynamic Goal Oriented Action Planning system.

Goals and actions have feature weights calculated from position and world state. Using simulation and reinforcement learning the robot has been trained to evaluate the feature weights. The planner will dynamically choose the highest ranked plan rated by the trained evaluation of the features.




**Team members:**
Andre Kramer Orten
Bendik Kvamstad
Eivind Wikheim