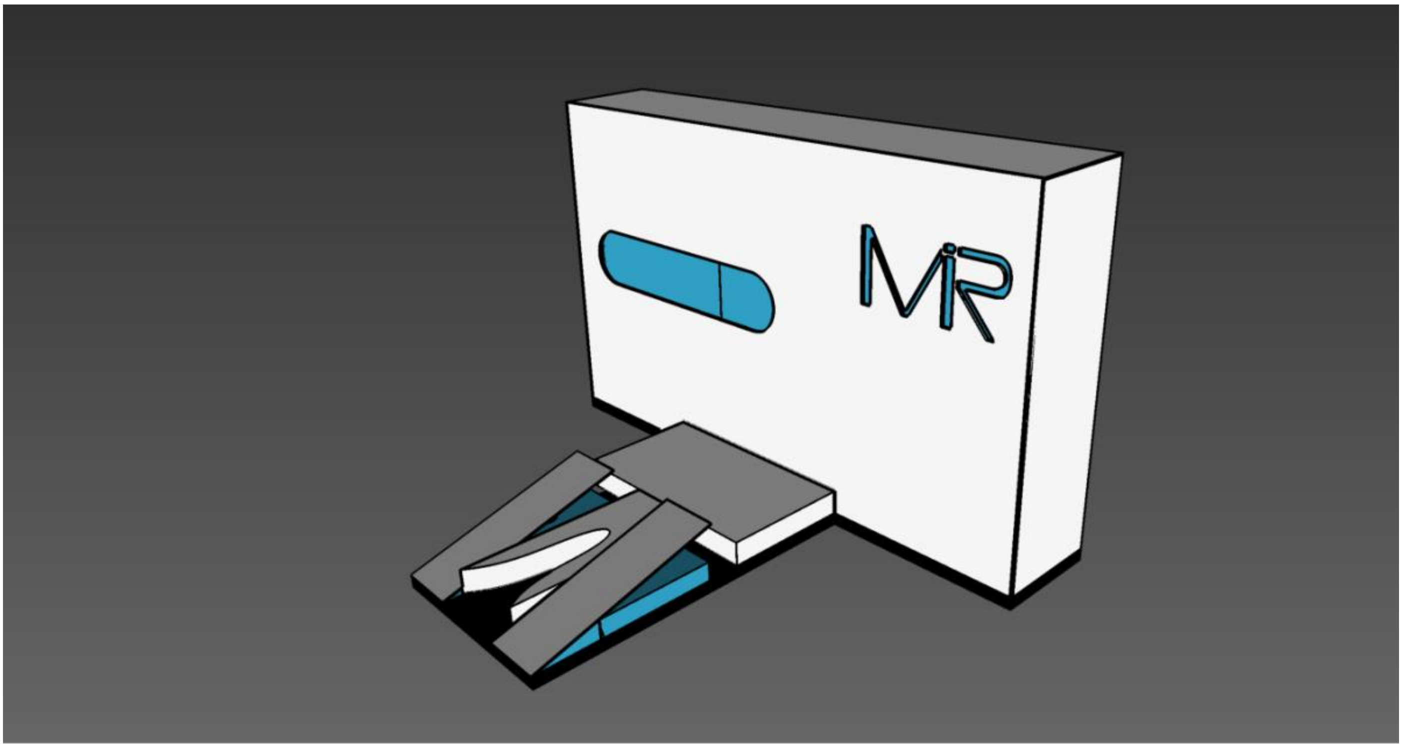
R&D of a loading and charging dock for an Autonomous Indoor Vehicle



Carlo Schalley & Manuel Serrao

The Maersk Mc-Kinney Moller Institute, University of Southern Denmark

Limburg Catholic University College, Belgium

Manuel.Serrao@outlook.com & CarloSchalley@gmail.com

Date of delivery: May 28th, 2013

External Promoter: Bruno Hanssen

Internal Promoter: Nico Bartholomevis

"It is the obvious which is so difficult to see most of the time. People say 'It's as plain as the nose on your face.' But how much of the nose on your face can you see, unless someone holds a mirror up to you?"

Isaac Asimov

Preface

Four months ago we started this project as a final project work, one last achievement to unlock in order for us to graduate, but it turned out to be a battle against time. This paper is the report of this long process. It cannot express the long days spent in the lab or the long nights spent at home, battling shoulder to shoulder with fellow students and friends, with a can of Redbull keeping us awake, the joy for a new idea, the hope for good results and the sadness and tiredness with each failed attempt.

First of all thanks to our families who are always encouraging and supporting us in all the things we do. They are following us in the adventurous idea of doing our internship somewhere else in the world.

A big thanks to our supervisor Mr. Bruno Hansen, in these four months we learned more than we expected. We have learned great lessons not only in robotics but also in "real-life". It will be hard to find another supporting supervisor like him.

A sincere thanks to the people working in the Robolab. It was a nice environment to fulfil our internship.

Another big thanks to the students of Dalum Landbrugsskole, the amount of enthousiasm and support received from them is extraordinary.

We would also like to thank our promoter, Mr. Nico Bartholomevis and Ms. Greet Raymaekers, international Departmental Coordinator, for giving us the opportunity to do our final project work abroad.

A big thanks to Ms. Petra Van Gansen and Ms. Evangelia Tzika for assisting us with the language corrections of the thesis and giving tips to improve.

We would also like to thank Ms. Inger Marie Jessen, Exchange Coordinator at the International Office of SDU for helping us a lot of the necessary paperwork.

And to all the people we got to know during our stay in Denmark, thank you!

In the end, thanks to you, reader. If you are reading this line after the others, you at least read one page of our thesis. Thank you.

ABSTRACT

R&D of a loading and charging dock for an Autonomous Indoor Vehicle (AIV).

By: Carlo SCHALLEY
Manuel SERRAO

Internal Promoter: Mr. N. BARTHOLOMEVIS – Limburg Catholic University College (LCUL)
External Promoter: Mr. B. HANSEN – University of Southern Denmark (SDU)

In general it has become economically more interesting to replace human labour by robots in the industry. Robot replacement for internal logistics in a hospital is just a matter of time. For example the AIV can bring dirty laundry, food, drinks, medical tools and supplies (modules) to the desired destination. This will make the entire supply chain more efficient while you need less employees.

Our job during this internship was to develop a dock from scratch where the robot could charge, load, unload and transport different kind of modules. Firstly we had to do a lot of brainstorming in order to sketch different 3D models with Google SketchUp and Inventor for the dock. Since this was not our field of study we also made a lot of Lego models to get a better understanding of the mechanics. Once we had the sketches and ideas we narrowed them down until we only had one or two good models.

After the mechanical part we started with the electronics and software behind the dock. We tested Raspberry and Arduino and chose Arduino, an easy to learn language based on C over Raspberry which is Linux based. Since this was more in our field of studies it felt more easy and less time-consuming than the mechanical part.

The dock in its current state will be able to charge, remove and store modules from the robot. The total package compared with the competitors, is cheaper than the average AIV in the current size range.

The future of AIV's is looking bright and we had a chance to learn a lot about the mechanical and electrical part of these during our internship.

Table of content

List of figures

List of Tables

Introduction

This internship is a cooperation through Erasmus between the Limburg Catholic University College (LCUL) and University of Southern Denmark (SDU). Within SDU a company named MiR gave us the opportunity to work on Mobile Industrial Robots. When we first got introduced to this project we had never worked with mobile robots, challenge accepted.

A mobile robot is an automatic machine that is able to move in any given environment, move around in the environment and is not fixed to one physical location. This particular mobile robot will be able to carry a load which is determined by the customer. The robot itself is a vehicle and the load can exist out of shelves, drawers, refrigerated units or others which make the robot suitable to transport the required load.

The robot navigates between predefined locations in its working environment by using the two laser scanners and a 3D camera. The laser scanners are also used to ensure that the robot does not collide with people or obstacles.

The goal of this internship was to develop a loading and charging dock for the MiR100 from scratch so the mobile robot could load, unload and charge. Through this paper we will try to guide you through the entire research and development process.

Objective

As we stated earlier in our abstract and introduction the objective of this thesis is to research and develop a dock from scratch where the robot could charge itself, load modules and unload modules. This has to be possible without any human interactions.

Chapter I : The product

This chapter will mainly describe the basics you need to know in order to understand what this thesis is all about and includes more general knowledge whereas the next chapters are more specific.

Project description

The entire project consists of three parts. We will work out one part in depth as this is our thesis. The part that we will be doing is the loading dock and the other two parts are the removable modules and the robot.

Removable Modules

These modules will be manufactured and designed by other companies and can be used in different branches e.g. : automotive industry, medical industry, space industry, nuclear industry, security, ...

The modules can be refrigerated units that can cool and or transport food, drinks, medical supplies. But there is also a possibility to store dirty laundry and trash, or to be used in hazardous situations such as taking care of nuclear products e.g. nuclear waste, nuclear powered machine dismantling. The modules can also serve for security, pick and place or even as cleaning units.

Figure 1: Trash module

Figure 2: All round module

Mobile robot

The alpha version of the MiR100 was already under construction when we got introduced to this project. We immediately had a working robot to our disposal which helped us get a good understanding of what the expectations were.

The mobile robot (MiR100) will transport the modules to their desired destination. For example: A laundry module needs to be emptied, the operator makes use of the Human Machine Interface (HMI) in order to tell the robot that the module is ready for transportation. The robot then transports the module to where the dirty laundry is stored.

The mobile robot is equipped with two mapping sensors, one in the front and one in the back. These will map the entire room in combination with a Kinect sensor which is placed on the front side of the robot to establish a 3D view. More about this from page 55 to 58.

The robot is equipped with a wireless router in order to have WIFI access for communication. This gives the opportunity to monitor the robot through HTML in order to check the status of the robot with any computer or even mobile phone.

Figure 3: MiR100 Alpha

Loading and charging dock

A mobile robot needs to be charged on a daily base in order to be able to perform every day. This is where we come in, starting with the research of all the different opportunities and requirements of the robot, work them out and use this information to implement it in the development of the loading and charging dock.

We will have to remove the modules and charge the robot when required. Removing these modules can be done in various ways. We have to look into all the possible ways of removing a module from the robot e.g. Forklift principle, hook, sideways forklift principle, lifting from bottom sides, magnets, vacuum.

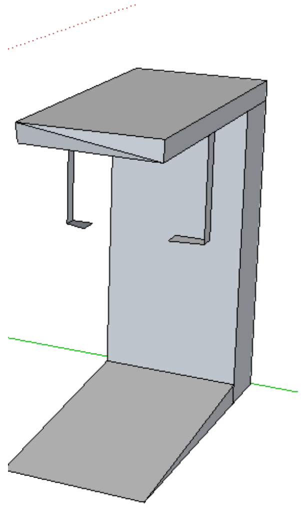


Figure 4: Loading dock with top grab

The robot needs to be charged as well. In order to get a better understanding of how a mobile robot can get charged we will have to do more brainstorming. As we have the internet to our disposal we quickly found our inspiration online and were combining ideas like it was nothing.

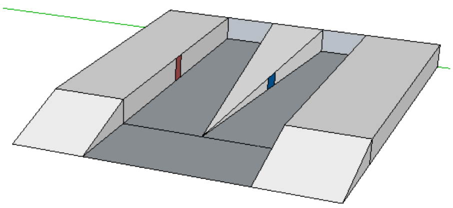


Figure 5: Charging dock with guidance

Identification of the opportunities and requirements

Opportunities

Firstly we had to look into all the opportunities which the Mobile Industrial Robots or better MiR100 had to offer, since this topic was quite new to us. So we started to question where and how Mobile Industrial Robots can make life more easy and efficient for people ? Our first customer was a hospital, so we started to look for different opportunities in hospitals.

Figure 6: Hospital of Copenhagen

Mobile industrial robots (MIR) show a lot of opportunities. They will ensure a better ergonomic workplace since the jobs no one would like to do are being done by the MiR100. Because these robots stay on the job 24/7 they are more productive than their human counterpart. They might not be able to do the most complex jobs yet, but in time they will be able to do most of the hard labour.

The robots are not going to steal jobs, they will only take the more intensive and monotonous jobs away. In turn they bring more jobs which require higher education as the robots need to be monitored and maintained. Summarised, robots have the following advantages:

- hard labour replaced by robots;
- boring labour replaced by robots;
- continuity of the robots;
- Higher educated jobs.

Requirements

The most important requirement is safety. MiR100 is not allowed to harm people or destroy the environment. Therefor there are three visual safety barriers and also electrical safeties.

- The first safety that is built into the robot is the emergency stop button which is located at the side of the robot, so anyone can push the button when they feel something is wrong or too dangerous.
- The second safety are the SICK S300 sensors which are used to map the environment, these sensors have a 2D view of the room and can see obstacles moving closer. This information will be used to either stop moving or avoid the obstacle.
- A third safety is the Kinect sensor which provides a 3D view from the front in order to detect the approaching obstacles even better.
- The electrical safety relay will make sure to prohibit any power given to the motor when the sensors detect any dangers, as well as the SICK S300 sensors do.

After the security requirements it is time to take a look at the requirements of the robot itself before we start brainstorming further about the loading dock. If we follow these requirements we will ensure an optimised structural integrity i.e. the structure's uncompromised ability to safely resist the required loads.

- The robot should be able to carry a minimum of 100 kg as the modules have a maximum weight of 100 kg.
- The robot should have a maximum height of 190 cm when combined with a module .

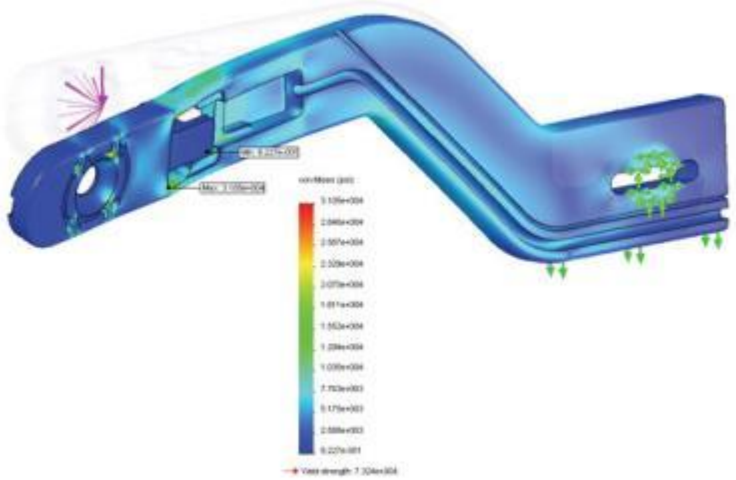


Figure 7: Strength analysis example

Identification of the market and competitors

Market

Although the first mobile robot was built in 1953, until now the market has been slowly growing and adapting to the needs of the customers, now that the technology is there and the mobile robots are capable of more for less the market might be on the verge of exploding.

For example in 2011 there were 2,141 mobile robots sold for logistics at an average price of €110,000 per mobile robot. The IFR (International Federation of Robotics) expects that from 2012-2015 there will be an increase of sales up to 10,340 mobile robots per year.

(source: World Robotics Service 2012/IFR Statistical Department)

According to Eurostat there are 2,1 million companies in Europe and when we exclude the microenterprises that are not interested in mobile robots we become the following numbers:

- Manufacturing: 218,000 companies
- Food: 180,000 companies
- Transportation: 115,000 companies

Which brings us to a total of: 513,000 companies excluding hospitals. And with an estimated market share of up to 10%, equivalent to almost 50,000 companies and a lifetime of up to 10 years for the robot, gives us a total annual demand of more than 5,000 mobile robots.

If we include the number of hospitals in the EU we get another 4974 potential customers located in our target group. Each hospital on average will be able to buy 8 mobile robots with a replacement every 5 years, this means an annual market of an additional 800 mobile robots.

(source: Designer Gruppe 406, Bilagsrapport, MiR100, Semesterprojekt Spring 2013)

Competitors

There are three primary competitors active in the European market. Two companies with a general mobile robot similar to the MiR100; Adept (U.S. company), Neobotix (German company) and one company with a dedicated mobile robot for hospitals: Aethon (U.S. company)

Adept Technology, Inc., 5960 Inglewood Drive, Pleasanton, CA 94588, USA, www.adept.com.

Adept is the largest U.S. manufacturer of all sorts of robots for all kinds of industries. In 2012, their total turnover reached around 60 million. USD. Adept has come out of 2012 with a significant deficit and in March 2013 replaced the management.

Adept was acquired in 2010 by Mobile Robots Inc., with their family of Pioneer robots for Universities they have a good brand. Adept level was in the next 2-3 years to mature market for mobile robots (ifl. chat with CEO 2011). They have two robots MT400 Courier and from the middle Lynx 2013, which is comparable to MiR100.

Their mobile unit is an equivalent to less than 30 people.

Neobotix GmbH, Weipertstr. 8-10, 74076 Heilbronn, Deutschland, www.neobotix.de.

Neobotics is started from the Fraunhofer Institute for Production and Automation (IPA), Stuttgart, in 1999, in order to sell ideas from IPA. They have delivered robots to museums, universities and for Audi. In 2010, Neobotix GmbH started. They are less than 10 people and had a turnover in 2011 of 250,000 EURO

Aethon Inc., 100 Business Center Drive, Pittsburgh, PA 15205, www.aethon.com.

Aethon produce the successful TUG, which is dedicated to hospitals. Since 2011, they have been the driving force behind the introduction of mobile robots in hospitals, primarily in the United States. They delivered 77 TUG in 2012. In 2012, they opened up the market in Denmark with their robots, currently being tested in the Aabenraa Hospital.

Aethon has 75 people employed, and has raised more than 80 million. USD in capital. In 2011, revenue Aethon to 7.1 million. USD.

(scource: Designer Gruppe 406, Bilagsrapport, MiR100, Semesterprojekt)

Description of the ideas

Brainstorming

Since we are doing our internship with two, our brainstorming group consisted of two members, sometimes three if we count our supervisor. We managed to come up with a lot of ideas which in turn led to the completion of the charging dock.

Firstly we had to come up with ideas of what the purpose of the robot could be in order to get a better understanding on how to develop the dock. During this process we were getting a more extensive image of the product; how to shape, position the dock and most important: Why? Why do we do it like this ? Can we do it in another way ? Will the pros be bigger than the cons ? We had to figure this out during our brainstorming phase in the first couple of weeks.

Development of the product

The development process of the dock can be described by a lot of brainstorming, combining the ideas we have found during the brainstorm process and putting them into practice. At first we had to come up with a lot of ideas in order to narrow down the ideas towards the end. After the first couple of brainstorming sessions we got a better view of the entire process as well.

We then started with sketching on a drawing board and soon found out that we are not the artists we thought we were. So we were advised to draw with a free to use 3D sketching program named GoogleSketchUp. With the help of this program we managed to get the ideas from our head into a basic 3D sketch. Once we had these sketches we could narrow down the ideas based on the different sketches we had, until in the end only a couple of sketches were left to choose from.

Figure 8: google SketchUp logo

At this point we decided it was best to make a couple of basic Lego designs to get a better idea of the mechanics and if we would find any flaws in the design we could easily alter the sketches afterwards.

Figure 9: Lego logo

Once this process was finished we started to convert our improved 3D sketches thanks to the Lego models into wooden prototypes. These wooden prototypes were on a real life scale so we could actually see the results right in front of us. We could spot the errors from all the different kind of prototypes we made and tried to correct them. It really felt like a trial and error procedure.

After we applied this procedure to the wooden prototypes we improved our sketches again and moved on to metal prototyping, and we did the same procedures over again.

After these trial and error procedures we wanted to refine our drawings but something was holding us back, GoogleSketchUp was not the program we were looking for.

Another student suggested us to make these drawings with Autodesk Inventor which really helped us a lot in terms of getting a good 3D drawing ready and then build a prototype.

Figure 10: Inventor logo

Completion of the product

The completion of the dock could only be achieved by working together and keeping a comfortable working environment where there is time for a break, joke and of course serious discussions. We had to share all our ideas and sometimes we had a totally different view of things but we managed to work everything out just fine by distributing the workload evenly.

Chapter 2: Idea phase

This chapter mainly consists of a description of the brainstorming before any practical actions were taken. The brainstorming process later on changed a lot since we were testing our ideas out in practice.

Two different packages

The customer has two options which will either greatly increase the efficiency of the entire system and therefore increase the price or the customer can also opt for a less efficient but cheaper system. It is advised that small companies better start with the cheaper package and if there is need for more production and therefore efficiency, it will always be possible to upgrade afterwards.

Explanation of these different packages;

Option 1: The company has bought the basic package which consists of:

- charging station;
- robot;
- module.

This package is cheaper since you only have a charging station and the module is permanently attached to the robot for the daily tasks. This means that the robot will have downtime when employees are unloading the module, which in turn is slower and costs money for being slower.

Option 2: The company has bought the advanced package which consists out

- charging station & docking station;
- robot & lifting module;
- module.

The big difference between these 2 is that with option two you can have more modules available since the robot will drop a used module in the docking station and takes an unused one. This way the robot can already get to its next job instead of waiting for someone to empty/refill the module. There will be almost no downtime which means more productivity!

Possible modules

As we have mentioned before these modules will be manufactured by other companies, but in order for these modules to fit on the robot we have to declare the basic specifications a module has. These basic specifications are to determine the dimension of the modules and the placement of the electrical contacts on the robot and the dock. The Human Machine Interface (HMI) is also determined through these specifications.

Currently the following company is interested in working together with us for the design and production of the modules:

Handy

Handy has been providing customized solutions since 1949. They are active in medical, food and industrial logistic industries.

Website: www.beka.dk

Figure 11: Handy logo

Automazion

Automazion has been created with a desire to help Nordic companies improve competitiveness through the use of robotics. Their mission is to provide the right robotics for customers who want to automate or assist customers.

Figure 12: AUTOMAZION logo

Website: www.automazion.com

Existing modules

The following are examples of already existing modules which will have to be redesigned in order to fit the robot and the robot's design.

Laundry module

MiR100 mounted with a laundry module waits near the room where the dirty laundry is located. The employee notices MiR100 and fills the module with the dirty laundry and with one simple interaction of the Human Machine Interface sends away the robot. The robot navigates to the docking station or laundry facilities to unload the module or wait until it has permission to leave.

Figure 13: Laundry module

MiR100 arrives at the laundry room where all the dirty laundry is gathered, it will pick up a module which has been cleared for pick-up by an employee. It will then move this module all the way to the laundry facilities in order to drop-off the module and/or pick up a new module if required or navigate to a module in the building that has been cleared for pick-up.

Trash-bin module

The same principle is used here, the robot is already equipped with the trash-bin module or has to pick one up at a docking station. In turn it will then bring the module to the desired destination.

Figure 14: Trash module

Tool module

Nowadays there are tools for everything, so why not have the MiR100 carry your tools around for you ? Your tools are now one button away.

Figure 15: Tool module

All round modules

These modules can transport basic tools, phones, chargers, televisions, computers, any hardware you can think of..

Figure 17: All round module 1

Figure 16: All round module 2

Figure 18: All round module 3

Loading dock: Different drives

If we were to use a drive in our loading dock to lift a 100 kg module, which should it be ? Pneumatic, hydraulic or electrical ?

After some brainstorming we have ruled out pneumatics and hydraulics by looking at the following pros, cons and combined these with the demands of the location where this project is going to be implemented.

Drives	Pros	Cons
Pneumatic drives	Cheap Lightweight Safe for high explosive areas Fast movement	Air has to be filtered Noise Limited pressure Acceleration & deceleration, not easy to control
Hydraulic drives	Compact Sturdy Acceleration & deceleration easy to control Great inherent damping	Expensive to build up pressure More pressure = more noise Pressure= higher chance leaks High energy usage
Electrical drives	Cheap Reliable High stiffness Great accuracy High power	Expensive components

Table 1: Drives pros & cons

Even though electrical drives have expensive components, in the long run they will make up for it.

Loading dock: Mounting

We both agreed on that of all the different options of mounting the dock, the 4th option was the most appealing to us. Mounting it on the floor gives a lot of freedom to the customer. The last option was also a consideration but this system would use more room and you also would need another ramp which in turn would make it more expensive. We wanted to keep it as cheap as possible so this was not an option for us.

Mounting the lift	Pros	Cons
On the wall	No extra additions to the robot No maximum weight of lift	Wall needs to be able to support Needs more room Alignment needs to be perfect Harder to install
On the ceilling	No extra additions to the robot No maximum weight of lift	Extra additions to the ceiling Height of the ceiling
On the floor	No extra additions to the robot No maximum weight of lift You can mount it where you want No extra additions to the wall No extra additions to the ceiling	Sturdiness of the floor Electrical wiring below floor Water resistant for cleaning
On the floor (2nd) (robot drives through)	Faster in & out	More material needed (ramp)

Table 2: Mounting pros & cons

Loading dock: Lift methods

As for lifting the modules we decided that if we apply two forks in the bottom powered by an electrical drive we will have the best results because the forks are less likely to be hit by the robot in standby and less materials are used this way. If this would not work out we can always let the robot do the lifting on its own as this is also a very good option.

Lifting the modules	Pros	Cons
Forklift front	Easy to create Easy Program Big surface	Robot can get hit by forks Module needs opening for forks Can't drive through
Forklift sides	Less likely to hit robot Implemented in the bottom	Smaller surface to rest on Wider bottom required Can't drive through
Coming down then sides	Less space required	More complex
Robot lifts own module	No separate lifting Easy to maintain	More complex Battery might drain faster
Two forks in bottom from side	Less likely to hit robot	Module needs to be wider More space required

Table 3: Lifting pros & cons

Charging station: Charge methods

As for charging the robot we will most likely implement contacts in the bottom of the loading dock as this ensures optimal connectivity between the robot and the connector in the bottom. The bottom connectors will be held in place with a strong spring system. The robot just has to drive in and once it is past the spring system the robot is on a fixed position and ready to charge if necessary.

Charging the robot	Pros	Cons
Under an angle	More easy for robot to get in/out	Robot has to constant brake Lifting more complex (angle) Connector has to be reached
Drive in under angle, then straight	Lifting more easy No constant braking	More bumps for the robot Length of charger increased
Charge from bottom (force)	Good contact Easy to make	Damage to bottom robot Force required for optimal connection
Charge from sides	Good contact Easy to make	Force required for optimal connection
Electromagnetic (allu. Plates)	No contacts	Powerfull EM field required
Charge through wheels	Compact	Moving components, brush req.
Wireless	No connection needed	A lot of power needed (losses)
Charge from the top	Connection closer to battery	More complex system

Table 4: Charging pros & cons

Chapter 3: Concept

After the brainstorming in the idea phase we were ready to make the sketches from our remaining ideas followed by the realisation of wooden prototypes.

Sketches

The sketches are all made with Google SketchUp. As we had no experience whatsoever with 3D design this was an interesting experience. The first few images are downloaded from the library while the rest of the images are all made by us.

Library sketches

As these sketches are found in a library and were merely for inspiration we will integrate the good ideas in our next sketches.

- The first sketch represents a forklift, if we were to use the forklift principle it could look like this.
- If we were to use vacuum we could use the same shapes as the second sketch.
- The third sketch represents lifting the module sideways.

As we found lifting the module sideways was one of the best ideas we could come up with, besides that the robot lifts the module, these are good sketches to get an idea of how the robot will drive in and the module gets lifted.

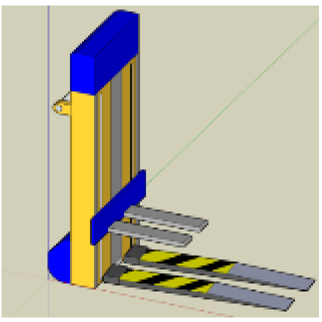


Figure 20: Forklift sketch

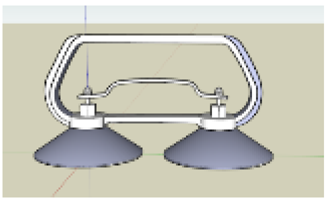


Figure 21: Vacuum lift sketch

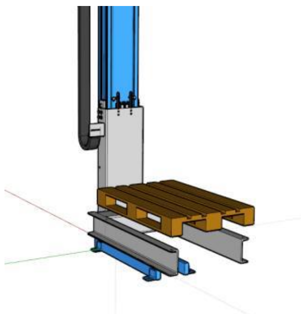


Figure 19: Side lift sketch

Lifting sketches

These are our very first sketches, really basic and kept simple. The first one will lift from above, the second one will be using the forklift principle while the third one will use the same forklift principle but will approach from the side.

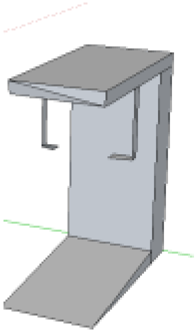


Figure 24: Loading dock with top grab

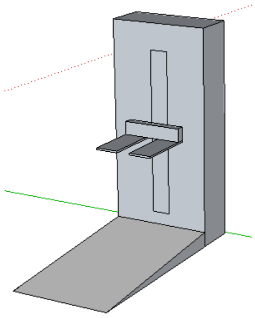


Figure 23: Loading dock with bottom grab

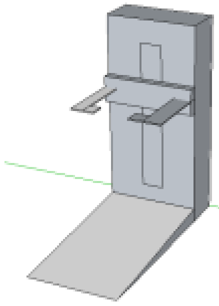


Figure 22: Loading dock with side grab

While we were drawing these sketches we came up with a totally different idea: what if we would use a scissor lift to lift the module ?

The first one shows the platform when the lifting system is not being used and in the second sketch we can see a scissor like lifting. The robot drives in, the scissor lift goes up and the robot drives out.

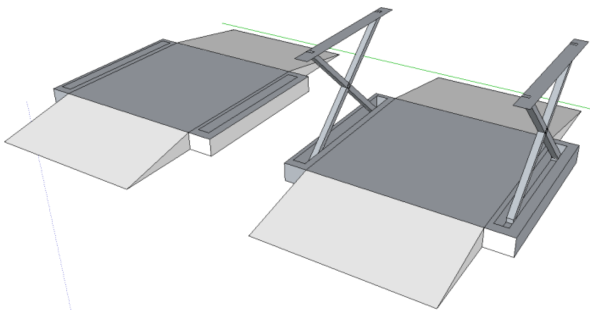


Figure 25: Lifting from sides

The remaining ideas were now ready to be refined and any errors to be worked out.

Charging sketches

The following sketches are our very first versions from the charging module, in meanwhile we changed them a bit but the general idea is the same. The robot has to drive in with some force in order to get to its fixed position. The amount of force required depends on the spring system from the two charging connections.

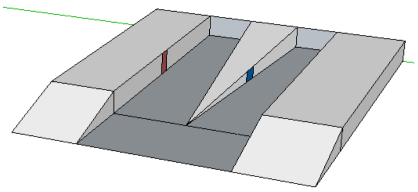


Figure 27: Charging and guidance 1

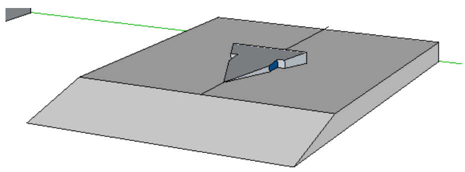


Figure 26: charging and guidance 2

On the next sketch you can see the two charging mechanisms. These will charge the robot from the front. This will require constant force from the robot in able to get a good contact so we have discarded this idea but in turn refined it which you can clearly see in our next sketches.

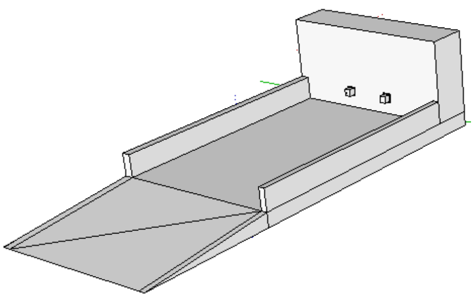


Figure 28: Charging front

The sketches down below show a better approach of how to charge the robot. The first sketch is lacking some kind of guidance system but you can clearly see what we were trying to do. The robot drives in and through a spring return connection the robot has to drive in with some force. This will ensure optimal charging.

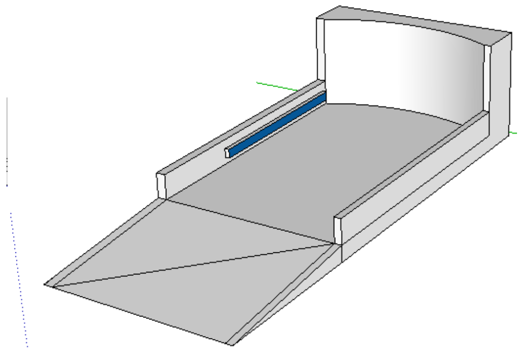


Figure 29: Charging sides

The second sketch shows a guidance system so the robot will always be in the same place when charging. Again: this is really crucial to ensure optimal charging. This is a drive through version but combined with the first one this could actually be a good concept to start off. Simply because there is a lot of space to make a contact to charge the robot, this is visible on the last picture.

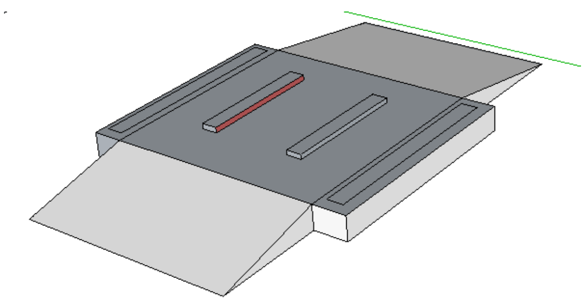


Figure 31: Charging near battery and guidance

Figure 30: Charging connectors

These are all good concepts to work with and improve wherever improvement is needed.

Wooden prototypes

We used the best sketch from the previous conclusions to show which one we are trying to prototype. This sketch is actually a combination of the previous two sketches. You can clearly see the charging contacts it has but without the drive through version.

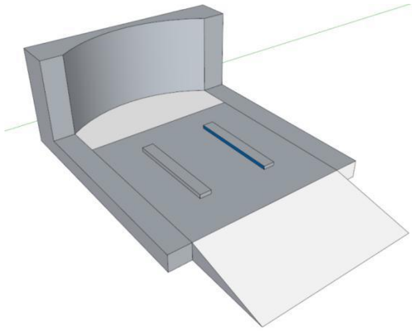


Figure 32: Concept 1

Wooden prototype I

We first tried to use a wheel guidance in order to guide the robot towards the charging contacts. This ended up damaging the wheels from the sides.

When the robot would move towards the guidance there was a chance that it would directly hit it, so we had to make the edges smooth to solve that problem.

When the robot had to drive out, the wheels would turn around and damage themselves. So we had to smoothen these edges as well.

Figure 33: Prototype 1

After the first prototype we mounted the two charging contacts to check if the battery would fit and look for any flaws in the idea since the first prototype already had quite some flaws.

Wooden prototype 2

Now that the wheels are guided we added a guidance for the battery holder in order to get the charging contacts as close as possible to the battery. But this prototype had two ways of guiding the robot into the correct position; firstly by guiding the wheels and secondly by guiding the battery holder. We were making it too complicated, we wanted to keep it simple. If we where to make it more complex, more things could go wrong. So in order to keep it simple we made a third prototype.


Figure 34: Prototype 2

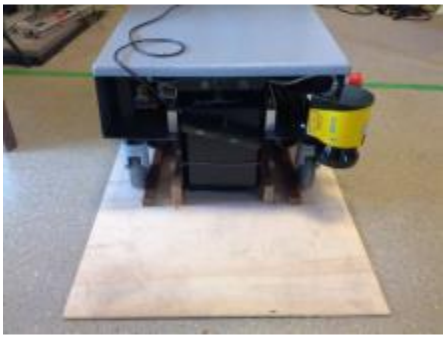


Figure 35: Prototype 3

Wooden prototype 3

This prototype will have the wheel guidance removed in order to keep it more simple. Instead only the battery holder will be guided. This prototype only has the last bit of the battery guidance. We will most probably add a V shape to the prototype in order to guide the robot more. This will be the 4th prototype.


Figure 37: Prototype 4


Figure 36: Prototype 5

Wooden prototype 4

The last prototype we describe in this chapter includes space for two charging contacts on the floor to charge the robot and a guidance which we mounted at the end, a triangle that fits in the bottom of the robot. This way the robot will guide itself into the correct position this way. No more damaging the wheels, no complex algorithms for the robot to drive in. Just a triangle form on the plate (circle) and a carved in triangle under the robot (circle). The charging contacts can be mounted on the rectangular pieces to ensure optimal contact.

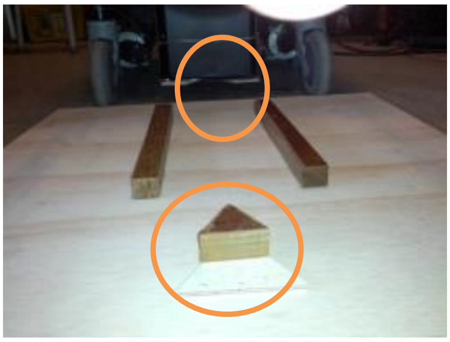


Figure 38: Guidance prototype

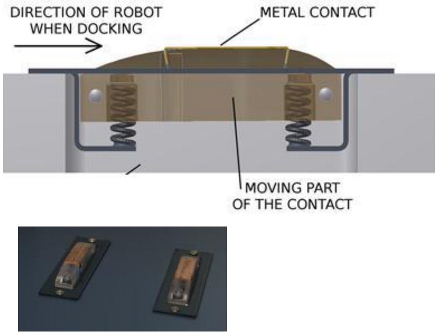


Figure 39: Example contacts

Conclusions

After all the prototyping we have learned valuable lessons.

Firstly we learned that we were making it too hard for ourselves regarding the lifting system but we came up with some nice ideas. We are going to use the idea where we lift with a scissor lift mechanism, but instead of lifting it externally with the module we are going to try and implement a scissor lift in the robot itself. Of course now that we were using the robot to lift the modules we had to do more research about more different lifting possibilities from the robot.

Secondly we observed that our first three prototypes were not up to standard but after a lot of tweaking and again having to deal with trial and error we managed to come up with a fourth prototype which worked quite well.

Chapter 4: Refining

In this chapter we describe the ideas of our brainstorming at first and afterwards the wooden prototyping.

Brainstorming: Lifting

In last chapter we concluded that it would be a great addition if the robot could lift the modules on its own. We had to find actuators that could actually fit inside the robot and lift the desired height. Therefore we had to refine our previous ideas.

Required dimensions

The following images gave us a better idea of the required dimensions, the first picture shows us 22 cm (back) and the second picture shows us 14 cm (front).

This means that the actuators we are looking for need to be really compact and at the same time being able to lift 100kg, preferably more in order to have some margin.


Figure 40: measurements

Linear Actuators

We have been looking at other different solutions for lifting the module with the robot. Our first idea was to do it with actuators, four linear actuators to be exact. One placed in each corner. They should have a minimum capacity of 250N in order to lift a maximum load of 100kg. In order for this to work we need another frame on top of the robot to ensure stability.

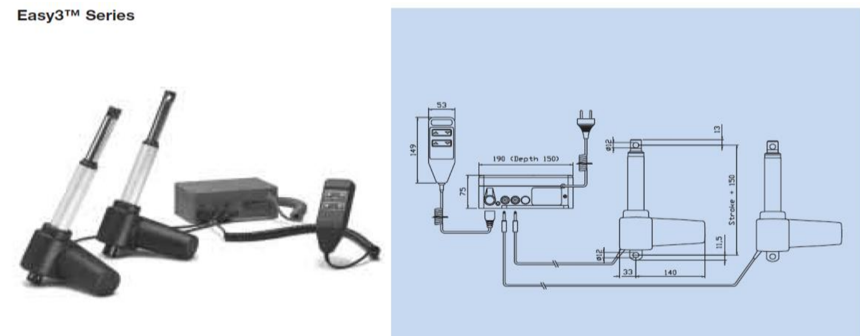


Figure 41: : Liniar actuators (www.rs-online.com)

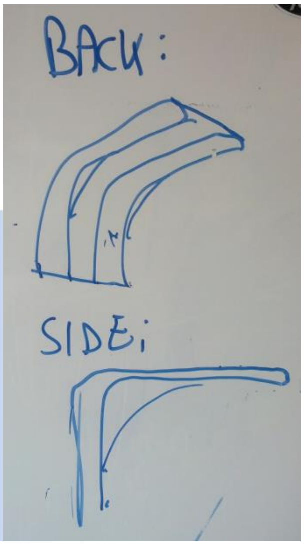


Figure 42: Lift from back

After looking for these actuators on the internet we had to rule this idea out because we don't have the space to mount these inside the robot. We could only mount them inside the back of the robot but not in the front. If we would mount them inside the back of the robot we are even lacking a couple of centimeters of space but we might be able to work our way around that.

So instead of having 4 actuators we can use 2 and mount them in the back. We still have to find the actuators with the correct dimensions and power. The image on the right can give you an idea of how it can be implemented. You first see a sketch from the back and one from side view. In order to use this we will have to make sure it is mechanically possible and do simulations first.

Scissor lift mechanism

Another idea was to lift the module by using a scissor lift mechanism. This scissor lift can be made to extend in the negative z direction to reach the module and then lift. This will require two linear cylinders mounted on/in the robot where there is room. You can clearly see in the image below that this might also require some extra space compared to the previous solution. We also have some simple Lego designs underneath the first sketch.

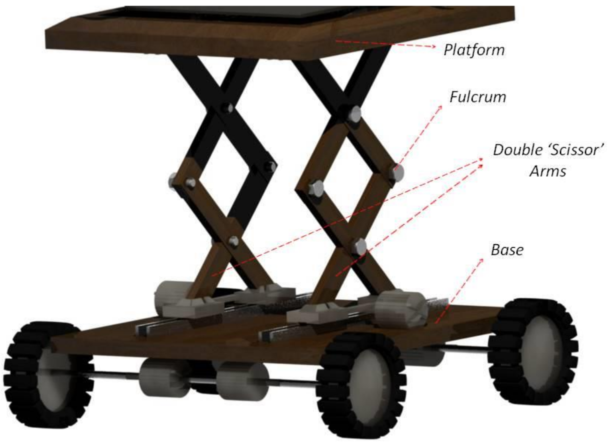


Figure 43 Scissor lift (www.Robotix.in)

Some of our basic Lego designs we used to get started for some idea generations:

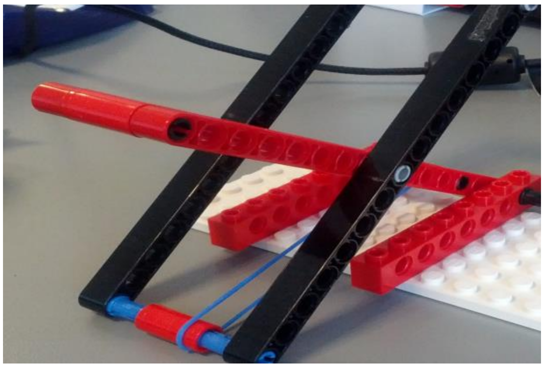


Figure 44: Lego test 1

Figure 45: Lego test 2

Note that the scissor lift mechanism will probably use a lot of space and during transportation of the modules the mechanism will be under constant stress. The amount of stress for the limited surface we have might be too much so let's move on to our next option.

Scissor lift revamped

After looking on the internet for a viable scissor lift that is able to lift what we want and is also light enough, we have found something interesting. A company named Southworth located in the United States has a product called Lift-Tool. It is a lightweight portable lifting tool that is supposed to be powered by a cordless drill can be inserted in the front and then turned left or right whether you want to go up or down. If we build in a 24V motor to turn it left or right we might have found ourselves a cheap way to lift external modules. More information about the Lift-tool:

Specifications:

- capacity of 136kg;
- lowered height of 8.69cm;
- raised height of 45cm;
- platform size is 58.42cm by 55.88cm;
- weight of 14.51kg.

Figure 47: Southworth scissor lift 1

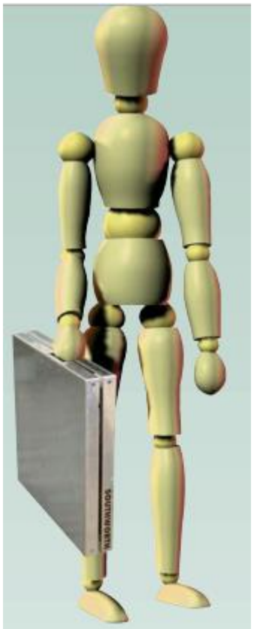


Figure 46: carrying example


Figure 49: Southworth scissor lift 3

Figure 48: Southworth scissor lift 2

Spindle/Power screw powered

After some research we found another solution which is a power screw. This one can be mounted in the back of the robot and lift a platform. See the images down below to have a better idea

.

Figure 50: rotating Power screw (www.Powerjacks.com)

Figure 51: Translating Power screw (www.Powerjacks.com)

The first figure shows a close up of the power screw while the second figure shows the motor mount on the bottom and a fitting that is mounted on the power screw, as the screw rotates the fitting will go up or down depending on the direction of the motor. (Rotating)

The third figure shows the motor mounted near the top and we can clearly see there is no more fitting and instead the power screw will now go up or down. (Translating)

If we can find power screws with the desired dimensions this could actually be a good choice. This is a really stable and reliable form of lifting. The examples from Powerjacks are actually too strong for our needs, they vary from 10kN to 400kN whereas we only need a maximum of 1kN.

Electric cylinder

The electric cylinder EPCO (from Festo) is a mechanical linear drive with piston rod and permanently attached motor. The driving component consists of an electrically actuated spindle that converts the rotary motion of the motor into linear motion of the piston rod.

This electric cylinder is able to lift the desired height and force, it would be a good option for us to use. We will have to study the datasheets more thoroughly in order to get a better understanding of all the characteristics from these electric cylinders. But currently they are our best option to go with.

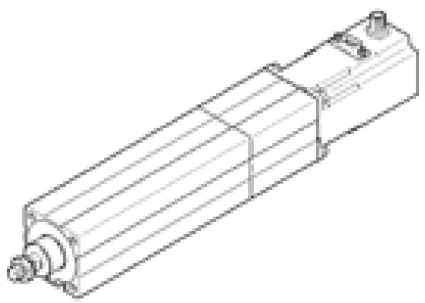


Figure 53: electric cylinder (www.Festo.com)

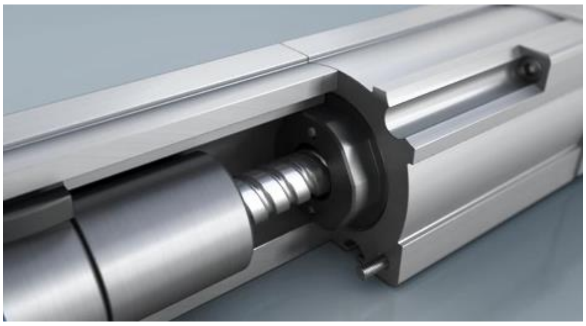


Figure 52: Inside view of an electric cylinder (www.Festo.com)

Brainstorming: Competitors

Until now we have been generating ideas without looking at the competition, but in order to achieve a good design it might be a good choice to take a look at the competition. This way we can analyse how they solved some of the issues we might be having.

About Aethon

Aethon improves healthcare efficiency and patient care by providing innovative logistics solutions. Its industry leading mobile robotic platform (TUG) enables hospitals to automate and improve the delivery and retrieval process across all major functions – including medications, supplies, meals, linen, and waste removal.

Its product tracking and security software (MedEx) gives hospitals the ability to track, monitor and electronically document the delivery process, particularly for high-value medications and other items requiring consistent chain-of-custody documentation.

Figure 54: AETHON 1

(scource: http://www.aethon.com/)

Aethon: TUG

Like its tugboat namesake, the TUG safely navigates through hospital corridors, elevators and departments to get items from point A to point B. It rolls 24 hours, 7 days a week to make both scheduled and on-demand deliveries, and never gets sidetracked from its mission. Whether it is a single IV or an entire unit's medication orders … a roll of gauze or enough supplies to replenish a supply closet, there is nothing too small or too large for the TUG to transport. The versatile TUG system attaches to and transports a wide variety of hospital carts and can be employed for any application and be attached to any of a series of carts (linens, dietary, waste, meds, lab).

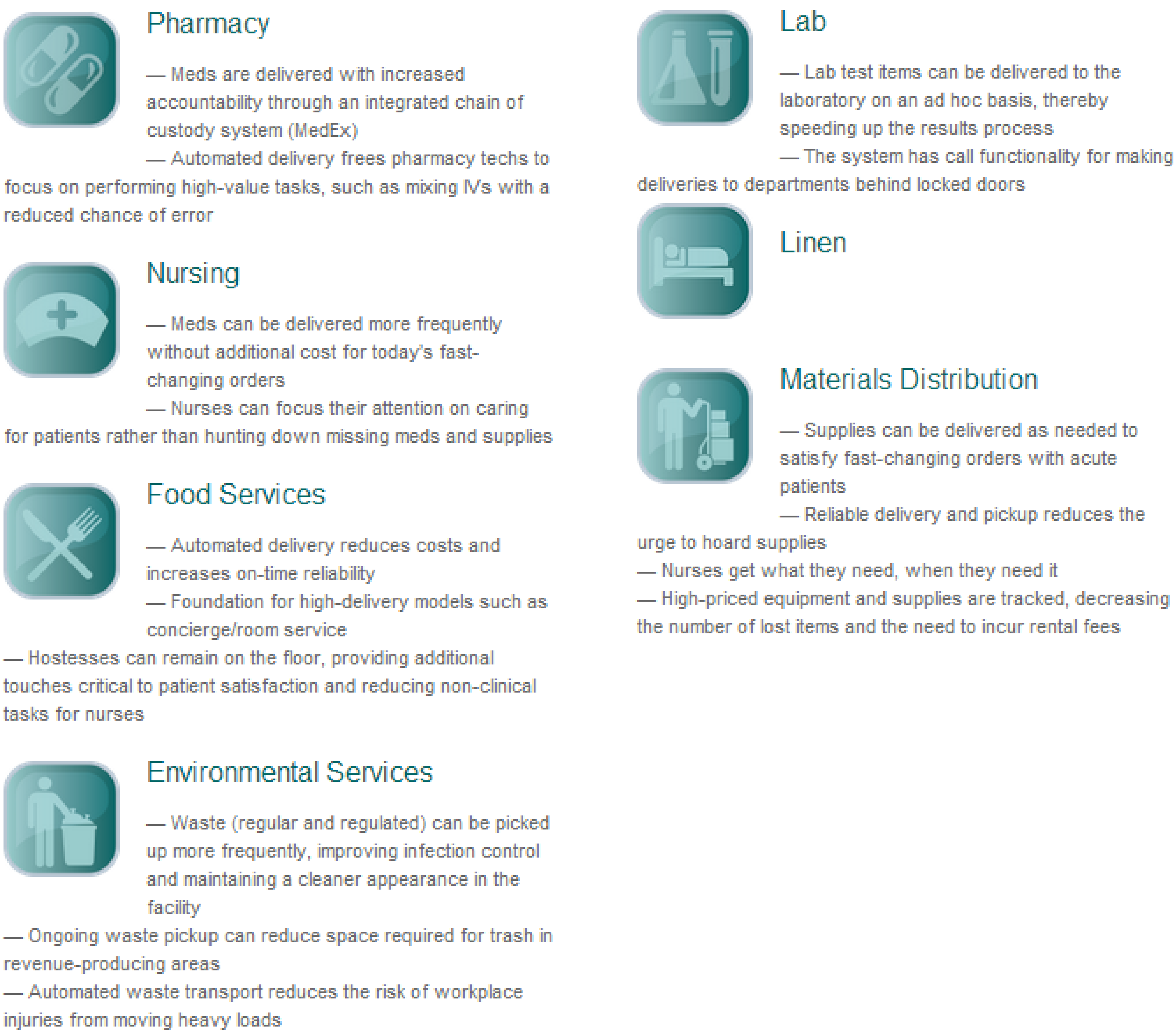


Figure 55: AETHON 2

Analysis of the TUG

We paid a visit to a research center where the TUG is located. We had the opportunity to look at the TUG and to analyse it.

The first thing we noticed is that TUG is quite small compared to the MiR100. This is because the robot itself will not be carrying the cargo but merely "tugging" the attachment where the cargo can be placed upon.

We have also been looking at the charging dock of the TUG and we might use some solutions they are using.

The following images show the TUG with and without its cover. We can see the circuit board, batteries, sensors and wiring.

Figure 56: TUG 1

Figure 57: TUG 2

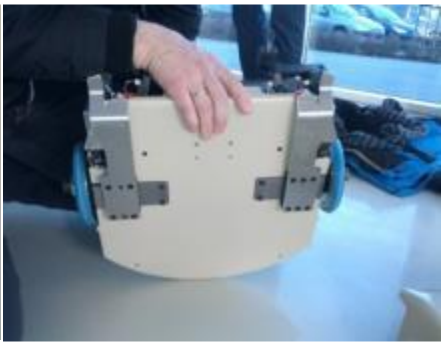


Figure 58: TUG 3

Figure 59: TUG 4

The following images show the attachment the TUG will be tugging, this attachment serves as a Human Machine Interface (HMI) and has a lot of the safety features like an emergency stop, laser sensor, infra-red sensors. The attachment stores the cargo while it gets tugged around.

The attachment also has a speaker combined with the interface buttons. You can see that this is quite a big construction while the robot is really small. We will have everything mounted in/on the robot which makes the MiR100 bigger.

The second image shows a view from the inside of the attachment, we can see the electrical wiring which connects the buttons to the circuitboard.

Figure 61: TUG 5

Figure 60: TUG 6

The bottom of this attachment shows us the locking and lifting system works, we can see the spindle from the back in the middle.

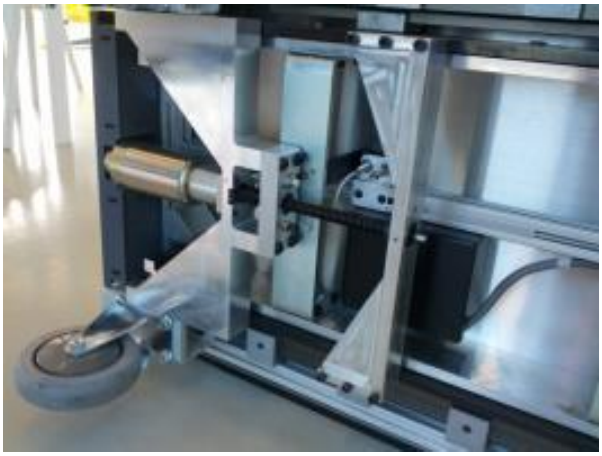


Figure 62: TUG bottom

The last image shows the charging dock of the TUG. As we had to develop the charging dock of the MiR100 this could give us the inspiration we needed.

The charging dock gave us a few new ideas. For example you can see that the corners are 45°, on first sight it might look like an unnecessary design but then we found out these corners where actually used to guide the robot correctly to the charging station. So afterwards we were looking for a solution like this to guide the robot properly.

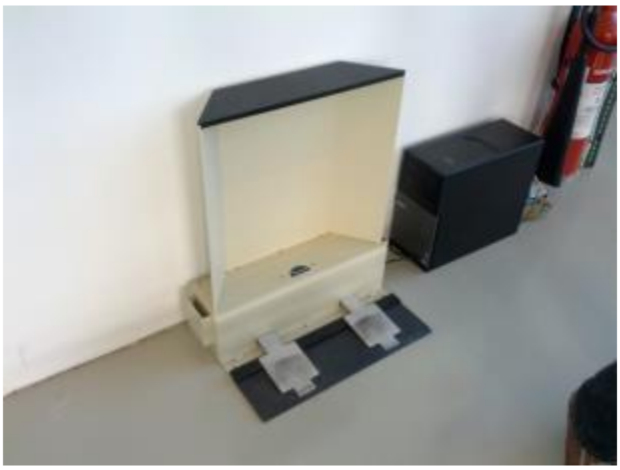


Figure 63: TUG dock

About Adept

Adept Technology, Inc. is a global, leading provider of intelligent vision-guided robotics systems and services. Adept systems provide unmatched performance and economic value throughout the production lifecycle, enabling customers to achieve precision, quality and productivity in their assembly, handling and packaging processes.

Founded in 1983, Adept Technology is the largest U.S.-based manufacturer of industrial robots. Adept intelligent automation product lines include industrial robots, configurable linear modules, machine controllers for robot mechanisms and other flexible automation equipment, machine vision, and systems and applications software. Adept provides specialized, cost-effective robotics systems and services to high-growth markets including Packaged Goods, Life Sciences, Disk Drive/Electronics and Semiconductor/Solar; as well as to traditional industrial markets including machine tool automation and automotive components.

Manufacturing its products in Pleasanton, California, Adept markets, distributes, and supports its products worldwide both directly and through channel partners. Adept's robotics products and services are the solutions of choice for many industry-leading corporations. More than 25,000 non-captive robots, built specifically with Adept's customers in mind, and more than 30,000 Adept controlled robots are installed worldwide. Adept is ISO 9001:2000 certified.

Adept believes intelligent automation is key to the success of any medium or high volume discrete manufacturing enterprise. Adept's strategy is to provide a broad range of high reliability, configurable, intelligent robot, controller, vision and software products along with world-class service to allow manufacturer's to maximize the productivity, flexibility and quality of the products they manufacture.

(Source: http://www.adept.com/)

Figure 64: Adept logo

Adept: Lynx

The Adept Lynx is self navigating Autonomous Indoor Vehicle (AIV) designed for dynamically moving material in challenging environments that may include confined passageways as well as dynamic and peopled locations.

Figure 65: Adept Lynx

Unlike traditional autonomously guided vehicles (AGVs), Lynx requires no facility modifications, such as floor magnets or navigational beacons, saving users up to 15% in deployment costs. Lynx includes Adept's proprietary software and controls allowing it to intelligently navigate around people and unplanned obstacles, that render traditional AGVs incapacitated and it can be programmed and functional within a day.

Designed for developers, integrators, and end-users the system can be customized for a variety applications and payloads. Manufacturing, warehousing, clean tech, and laboratories are just a few environments ideal for Lynx.

About Neobotix GmbH

Neobotix started originally at the Fraunhofer Institute for Production and Automation in Stuttgart. Since more than ten years, they design, develop and manufacture mobile robot systems of all kinds and for customers all over the world.
(Source: http://www.neobotix-roboter.de/)

NEOBOTIX

Neobotix: MPO-700

The omnidirectional MPO-700 is the ideal base for high-end service robots. Its four Omni-Drive-Modules enable it to move extremely smoothly into any direction. This robot is even capable of rotating freely while driving to its destination. The Omni-Drive-Modules of the MPO-700 feature important benefits compared to other omnidirectional drive kinematics, like for example the MPO-500's Mecanum wheels.

- Fully omnidirectional manoeuvrability
- Very steady movements
- High stability and payload
- Compact, easily integrated drive units

This makes the MPO-700 a premium alternative for applications that require omnidirectional movements without the limitations of traditional kinematics.

Figure 66: Neobotix: MPO700

Brainstorming: Lego

After we took a look at the competitors we went back to the drawing board and decided to make a simple Lego model that would explain the process of the MiR100, modules and loading dock. So we could simply explain it through a Lego model.

Loading dock:

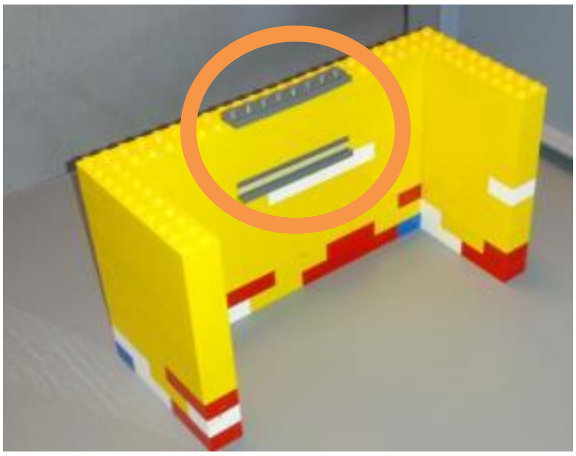


Figure 67: Lego test 1

Module:

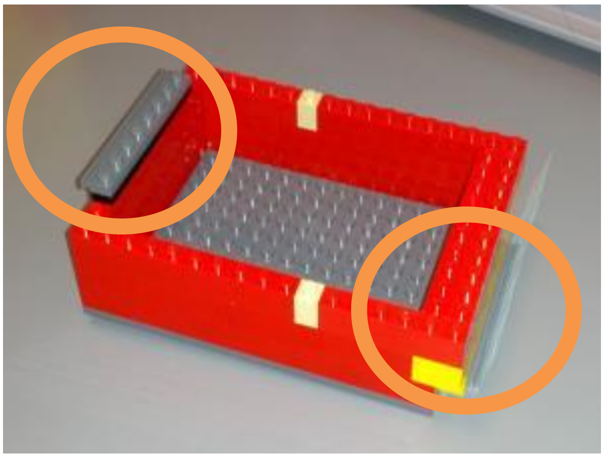


Figure 68: Lego test 2

Robot:

Figure 69: Lego test 3

Module on top of the robot:

Figure 70: Lego test 4

The most important part of the first three images are the orange circles which are placed around the locking system. We observe that the loading dock, module and robot have these grey pieces attached which can lock the module in place.

Now that we have explained the basic locking mechanism you can now observe how it works on the following images.

Module on top of the robot (side):

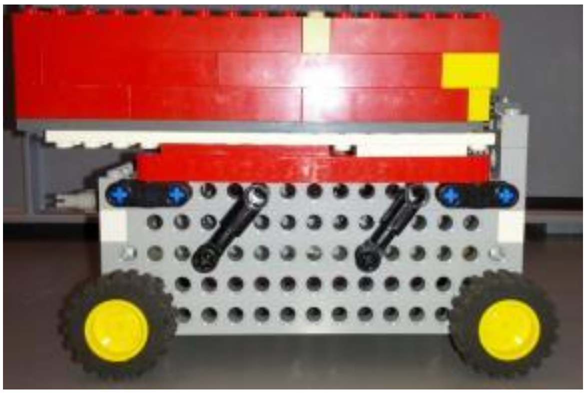


Figure 72: Lego test 5

Zoom of the lock (OPEN):

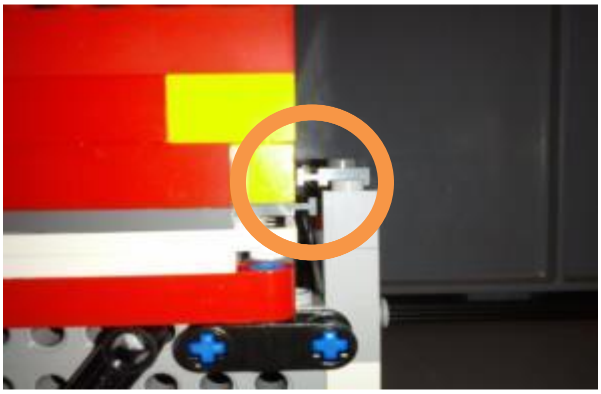


Figure 71: Lego test 6

Turn motor right, actuators come up:

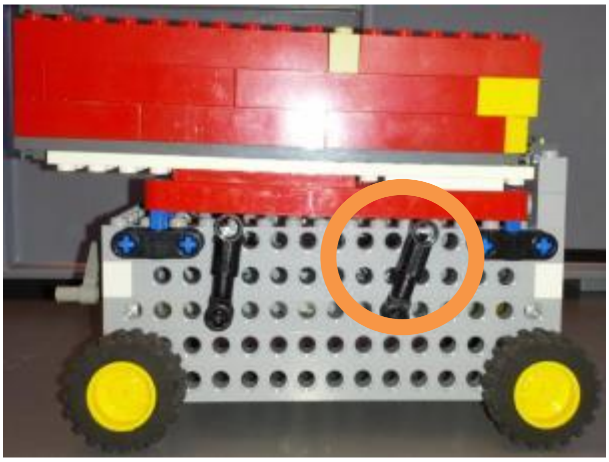


Figure 73: Lego test 7

Zoom of the lock (CLOSED):

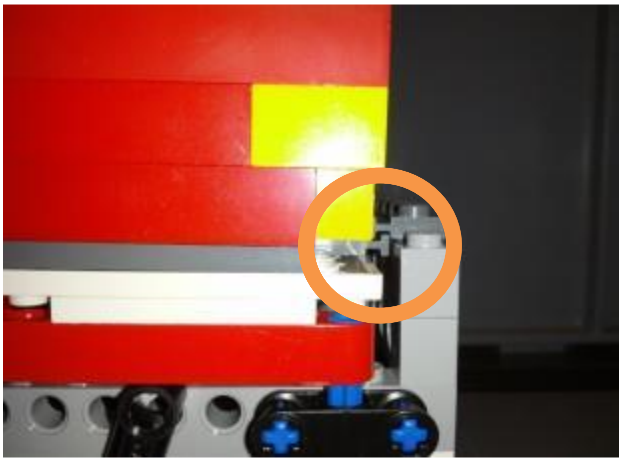


Figure 74: Lego test 8

We observe that the module gets locked once the robot lifts. This ensures that no one can steal the modules when the robot is driving around so that no cargo will be lost.

The last part has to do with locking the module on the loading dock. We were looking for an efficient way of locking the module without making it too hard. The Lego designs came in really handy for this as you will see on the following images.

Robot drives in the dock:

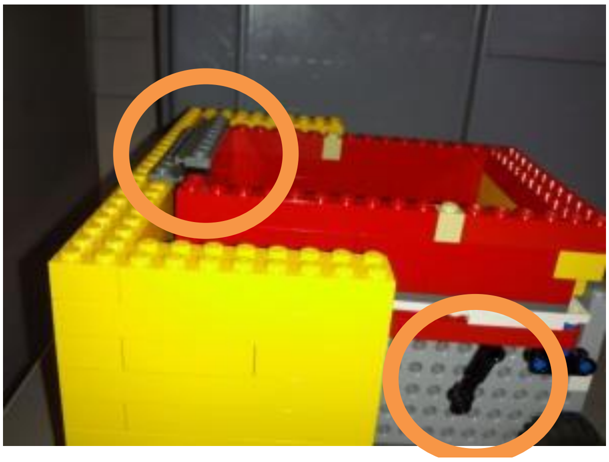


Figure 76: Lego test 9

Actuators go down & lock (motor):

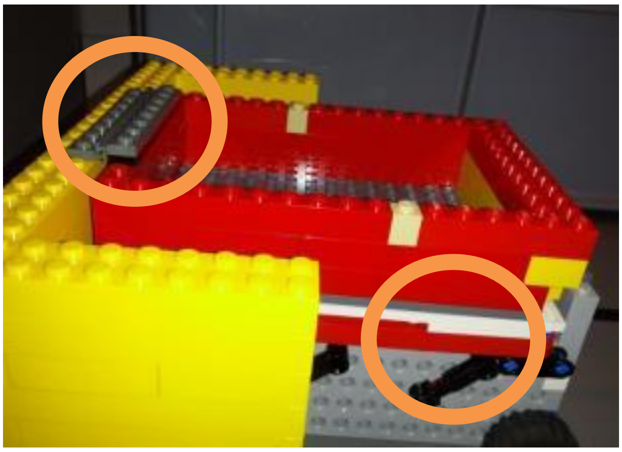


Figure 75: Lego test 10

Figure 77: Lego test 11Robot drives

These images explain the entire process. After all the brainstorming it was time to put all these ideas into practice and refine our wooden prototype.

Prototyping: Wood

After the brainstorming sessions we decided to try and make something similar like the TUG charging station.

We added a back plate to our current wooden prototype in order for the laser to detect the two 45° corners so the robot could recognize this pattern and navigate towards the loading dock. We left the guiding system intact, as we can remove it really fast if it will not be needed.

The two large pieces of wood in the middle represent an area where we can mount the charging contacts. And last but not least, the rectangular shaped wood with a triangle carved out represents the bottom plate of the battery.

Figure 78: Wood prototype 1

Picture by picture you can see it reach the destination being guided by the triangle piece. This might not be necessary as we said before.

Figure 80: Wood prototype 2

Figure 79: Wood prototype 3

Conclusions

Because our robot could not have a clear view of the 45° corners, we had to change these. After thinking about what the robot could distinguish best, we came to the conclusion that we had to keep it simple. Therefore just a square form with a corner of 90° might be better to generate easier algorithms.

If we would use our previous prototype with the 45° corners we would receive the following result from the 2D scanner which is the primary mapping sensor of the MiR100.

Figure 81: Concept Font panel 1

When we would take these corners away we might get a clear view and the robot could recognize the following patterns more easily. Therefore we made two different types and we tested them both to see which one would perform best.

Figure 82: Concept front panel 2

When we were monitoring the robot's vision we noticed that a squared edge performed better. So we had to get rid of the 45° shaped corners. On the next page you can see how the robot sees/reads the world.

Chapter 5: Vision and Prototyping

In this short chapter we will observe how the robot "sees" the world and then adapt our prototype in a way to improve the shape in order for better detection and easier algorithms.

Vision

The following image represents how the 2D sensors map the environment. There is a lot of 'noise' on this image and we only need the pattern that allows the robot to navigate towards the loading dock. We can also distinguish the vision from the Kinect sensor as it is showing us a 3D view from the front of the robot in color whereas the 2D sensors show white or red dots. The image can be divided in six parts. On the next pages we will explain what each part represents and which part is most crucial.

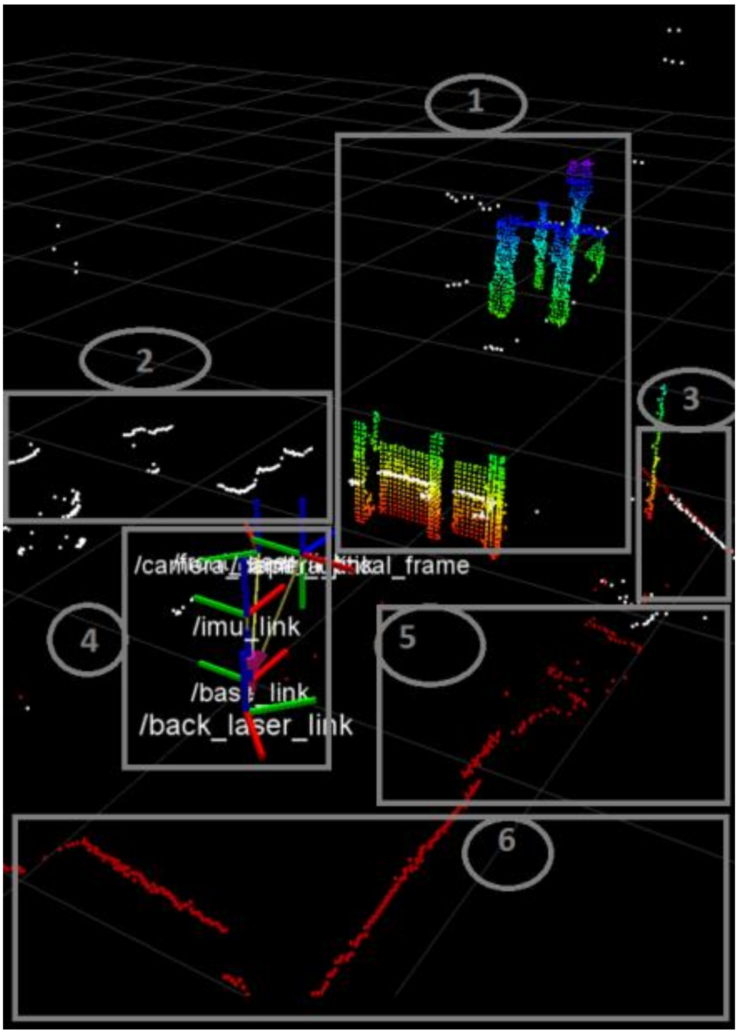


Figure 83: Robot vision 1

Part I :3D vision

The Kinect sensor provides the MiR100 with 3D vision, in the bottom left of the vision image we observe our new prototype dock which has the 90° angles we discussed earlier. The Kinect sensor is mainly used for an extra safety and to detect obstacles that approach the robot from the front more detailed. The structure on the background of the image is irrelevant to our thesis and merely on the picture to show how far the sensor can see the 3D environment while being encased in the robot as the encasement limits the view of the sensor.

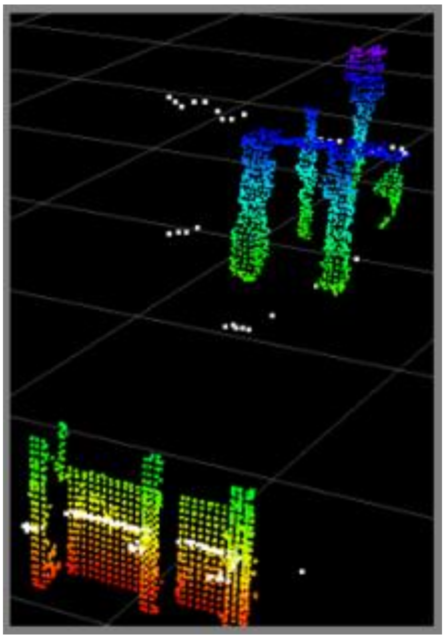


Figure 85: Robot Vision 2

Figure 84: Kinect sensor 1

Kinect sensor

As we stated earlier, the Kinect sensor is mounted on the front of the robot and gives us a 3D view for in case we need to distinguish some specific objects at a certain height or to detect dangerous situations. Although we have 3D vision this sensor comes with a big disadvantage: the sunlight (infrared) can literally blind the robot. So the Kinect sensor will not be used as a primary sensor.

Figure 86: Kinect sensor 2

Part 2: 2D vision

The two S300 safety laser scanners provide the MiR100 with 2D vision of the environment 360° around the robot. In order to get the full 360° coverage it is required to use two of these sensors, one mounted on the front of the robot on the left side and the other one mounted on the back of the robot on the right side.

Figure 87: Sensor placement

If we then take a look at the vision image we can clearly distinguish the round shapes of legs in a 2D environment.

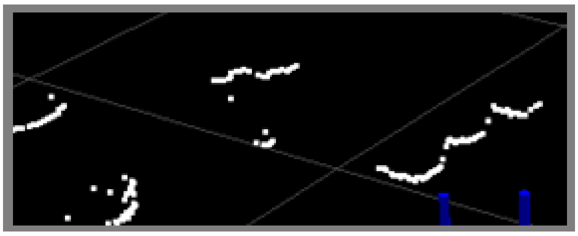


Figure 88: Sensor vision 2D

S300 Safety laser scanners

As we stated earlier the two S300 safety laser scanners are mounted in opposite position from each other to ensure 360° 2D coverage. These safety laser scanners map the environment and allow the robot to navigate.

Figure 90: Sick 300

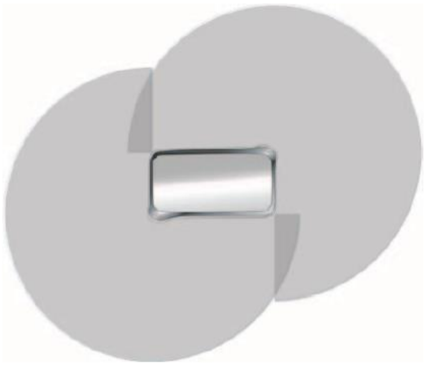


Figure 89: Sick area scan

Part 3: Combined 2D vision

The following image does not only have the white 2D dots but also the red dots. These dots represent the area that both sensors could detect at the same time.

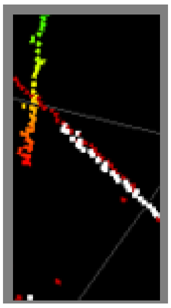


Figure 91: 2D vision 1

Parts 4,5,6: Base of the robot

The fourth part represents the base of the robot and all the axes of the different sensors combined. While parts 5 and 6 show the wall in the back of the robot .

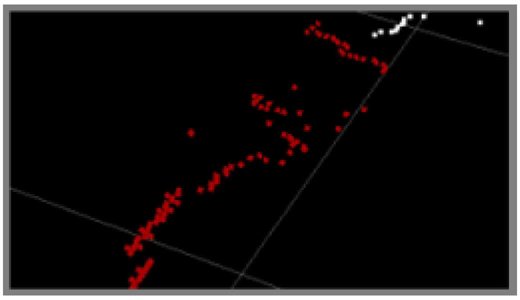


Figure 93: 2D vision 2

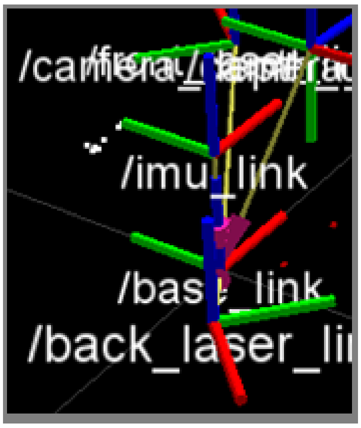


Figure 92: 2D vision 3

Prototyping

On the previous pages we looked at the vision of the following prototype models, it made more sense to show these prior to the prototypes.

Prototype 90° edges 1

We made some changes to the prototype. This way the robot will hopefully see the edges better which might result into an easier algorithm for the docking procedure. This is what the Kinect sensor spotted three pages back.

Figure 94: Wood Prototype 90° 1

Prototype 90° edges 2 (gray)

This prototype will only have a slightly different form in comparison to the previous, in order to determine which prototype works best and test at least two different options. It all depends on how clear the robot "sees" the prototype, we might have to choose between one of these prototypes in the end.

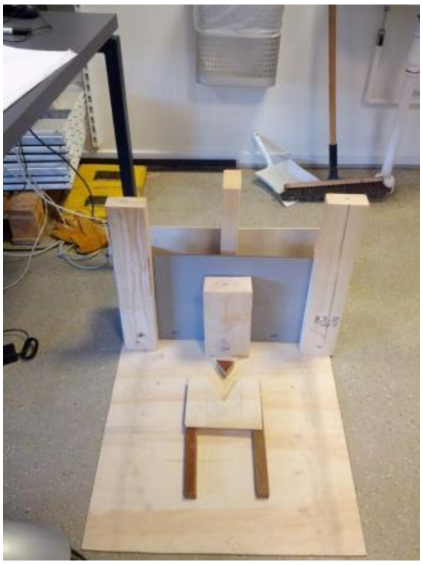


Figure 95: Wood prototype 90° 2

Conclusion

After looking at the images of the vision we realised there were line of sight issues, the laser would not detect anything right behind the corner if positioned right in front of the loading dock. This might cause trouble and actually make the algorithm more complex instead of easy.

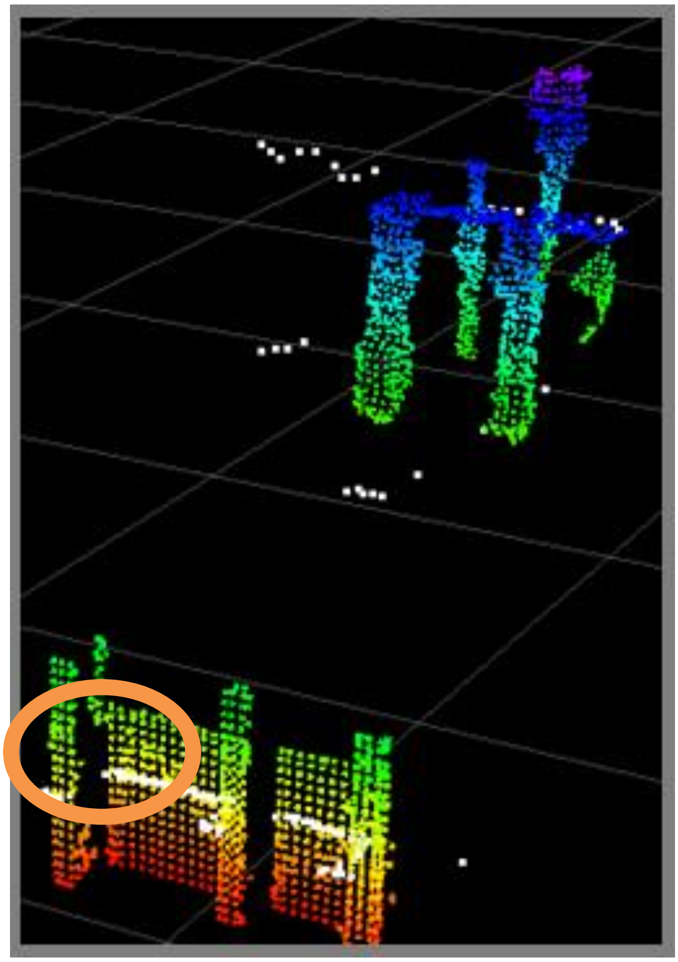


Figure 96: 3D vision

Chapter 6: Improved sketching and prototyping

This chapter will mainly focus on the fact that the laser scanners work better when the scanned surface has a corner, a big surface is required since our previous ideas taught us that the small surfaces are not easy to detect with the laser scanners. This was an error we made that we could have avoided if the surface of our corners would be bigger.

Sketches

So we had to go back to the drawing board, and this time we were looking for big surfaces. And so big surfaces it has.

Middle bend

Down below you see two different sketches. The first sketch shows a triangle where the sensor can pinpoint the position of the robot, more of that on the next pages. The second one shows the same principle but then reversed, we have to find out which one performs better and why.

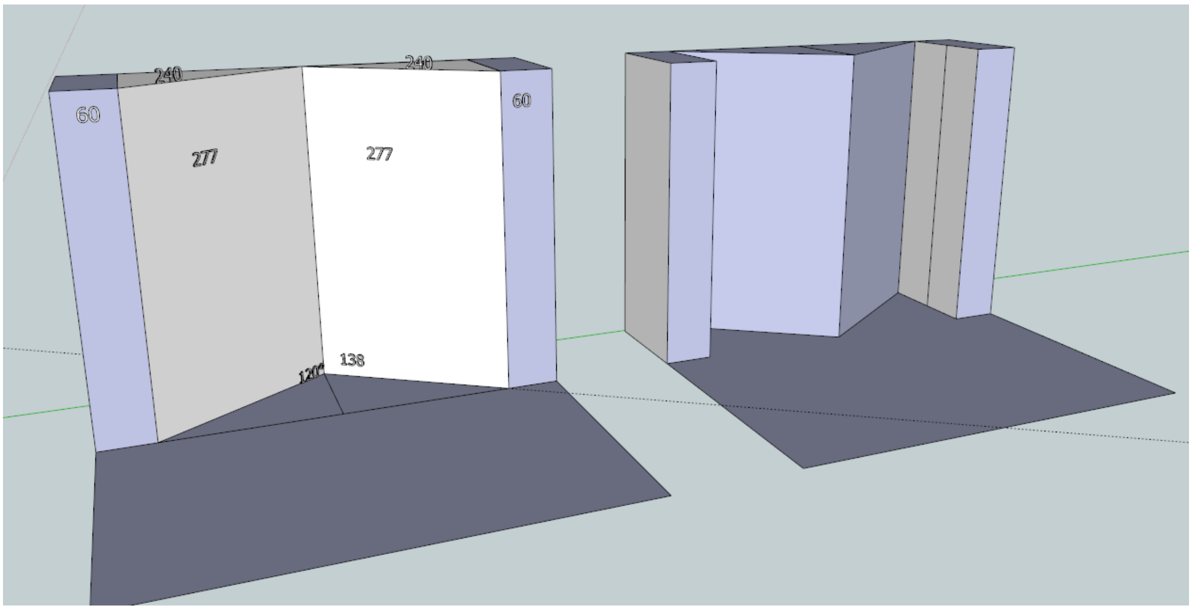


Figure 97: Middle bend front plate

Side bend

During our time in the metal workshop we came up with a new, different idea. The sensors of our robot are not mounted in the middle but on the sides. This means it would make more sense to change the position of the triangle more towards the sensor. Afterward we would have to compare all four designs with each other and hopefully one will show decent results.

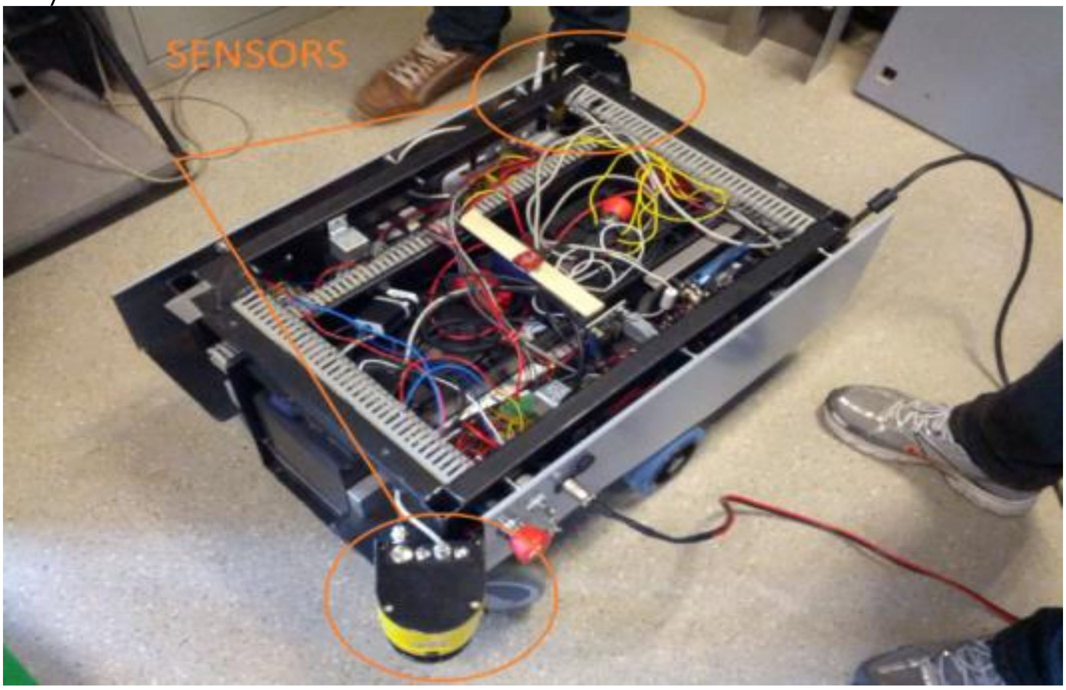


Figure 98: sensor placement

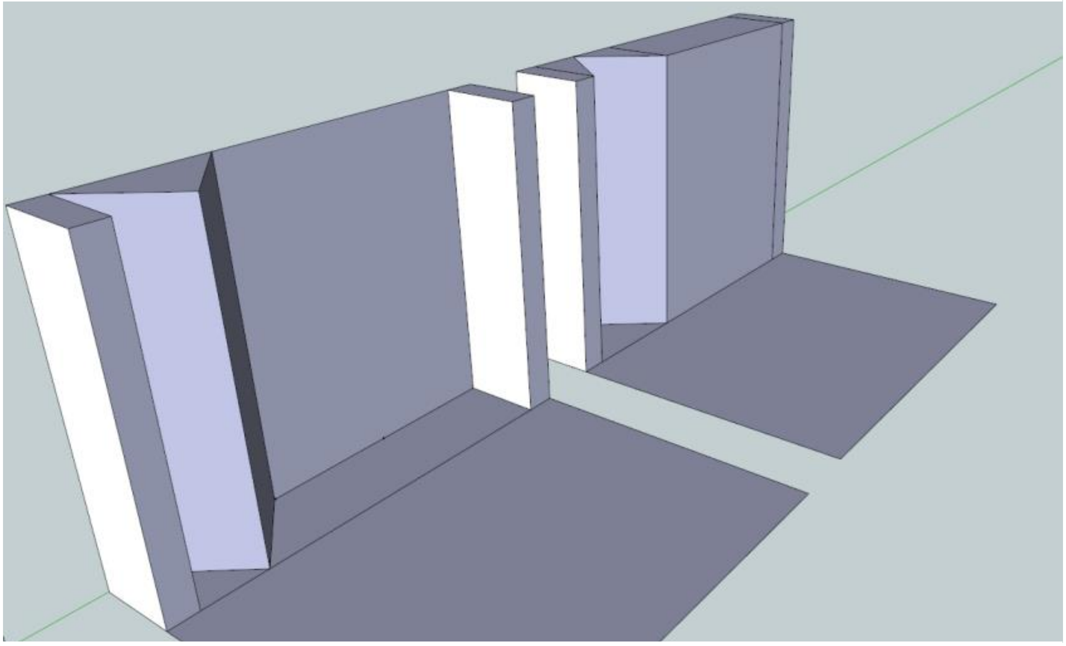


Figure 99: Side bend

Metal Prototypes

Metal prototype middle bend

The robot drives in while the sensor pinpoints the exact position. Hopefully the sensor will recognize this form better. The second image shows a very simple approach of the vision from the sensor.

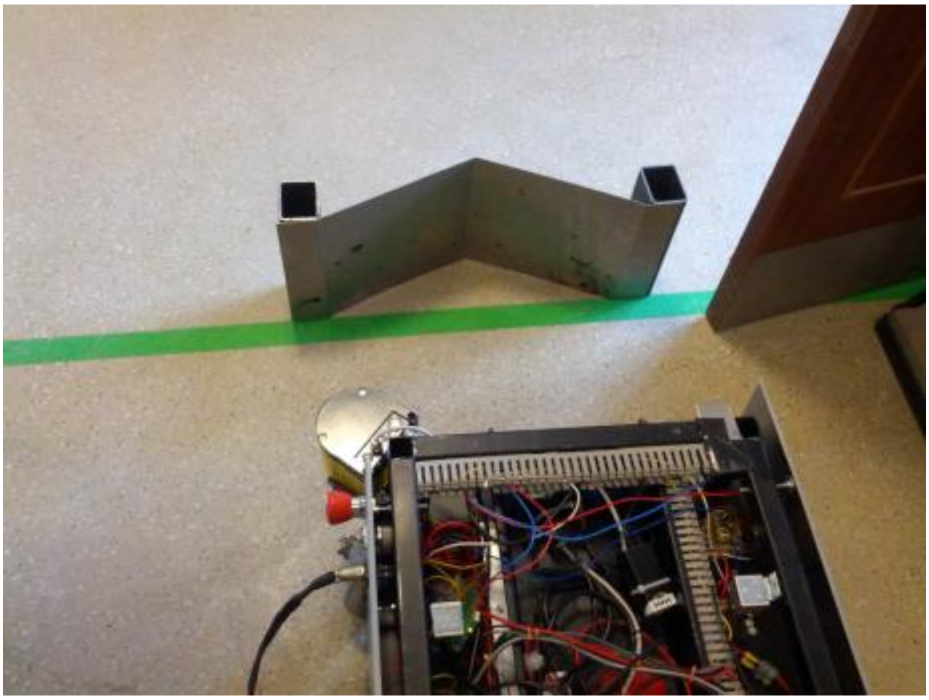


Figure 100: Metal prototype 1

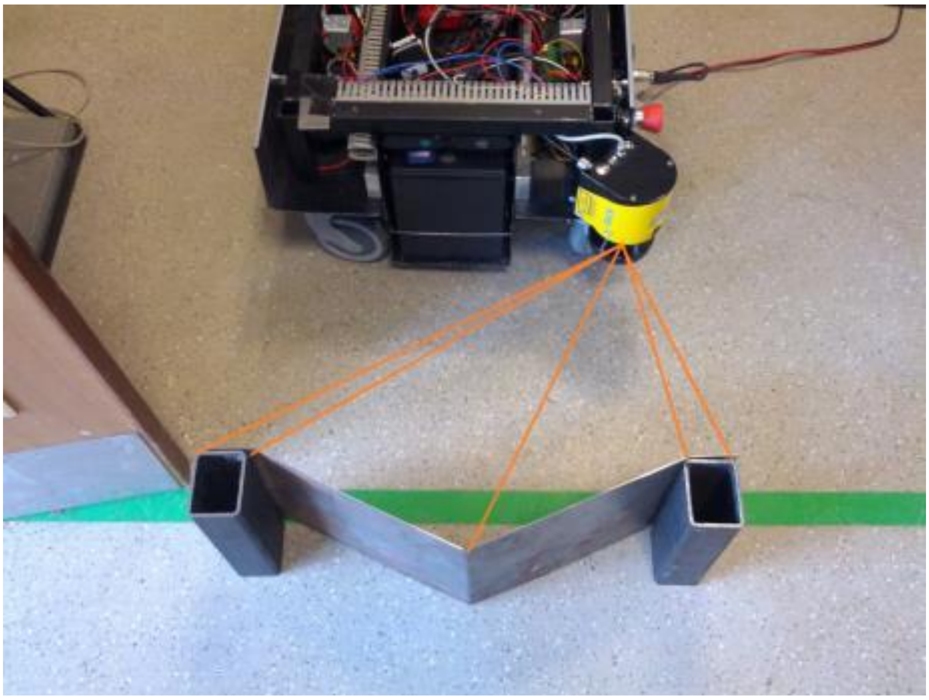


Figure 101: Metal prototype 2

Metal prototype middle bend reversed

The robot drives in while the sensor pinpoints the exact position. Hopefully the sensor will recognize this form better. The second image shows a very simple approach of the vision from the sensor.

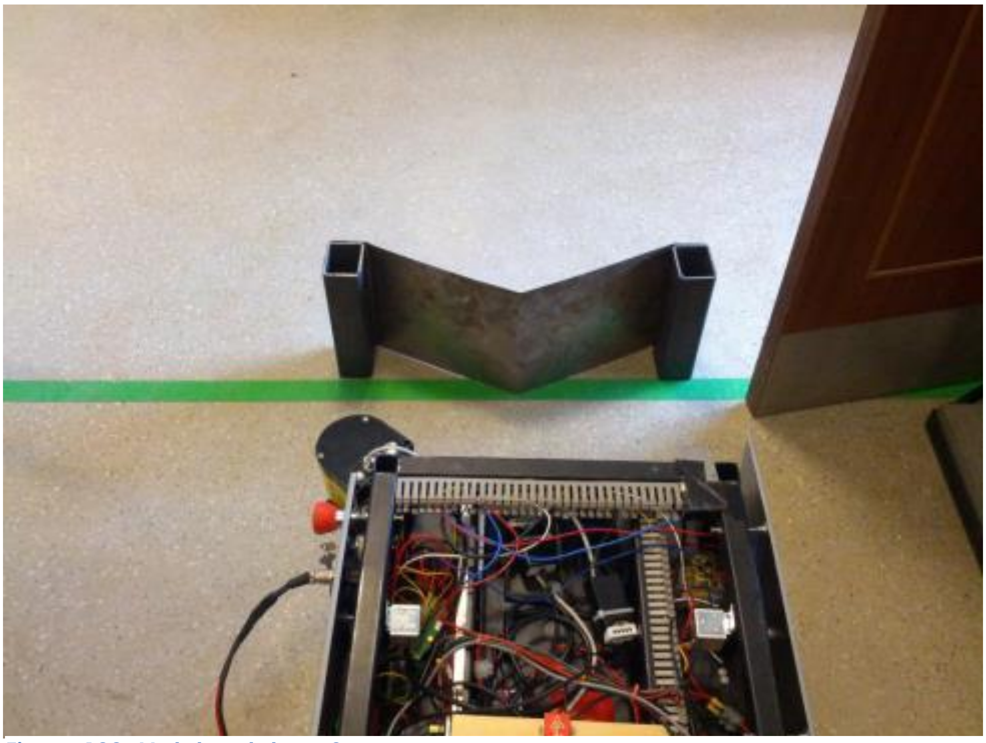


Figure 102: Metal prototype 3

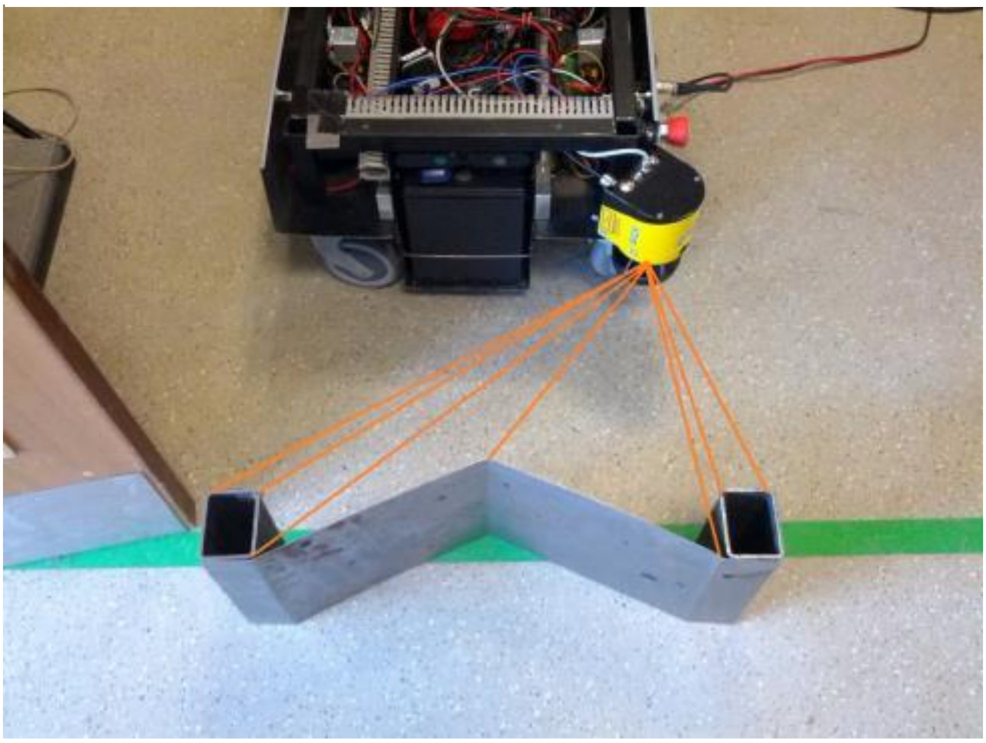


Figure 103: Metal prototype 4

Metal prototype side bend

This prototype is the same principle as the previous but we only moved the triangle on the same side as the sensor. Hopefully we will get better results as it makes more sense to place the laser as close as possible to the bend. It is now more easy for the sensor to pinpoint the exact location. The second image again shows a very simple approach of the vision from the sensor.

Figure 104: Metal prototype 5

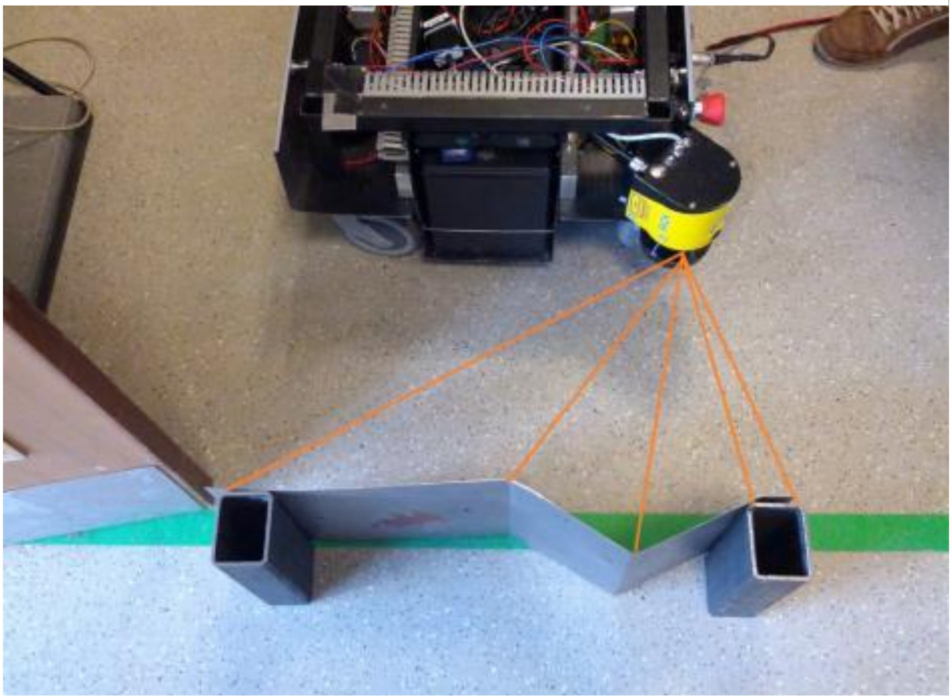


Figure 105: Metal prototype 6

Metal prototype side bend reversed

Same as previous but then reversed, we might get better or worse results. We will have to wait for the results to come in now. The second image again shows a very simple approach of the vision from the sensor.

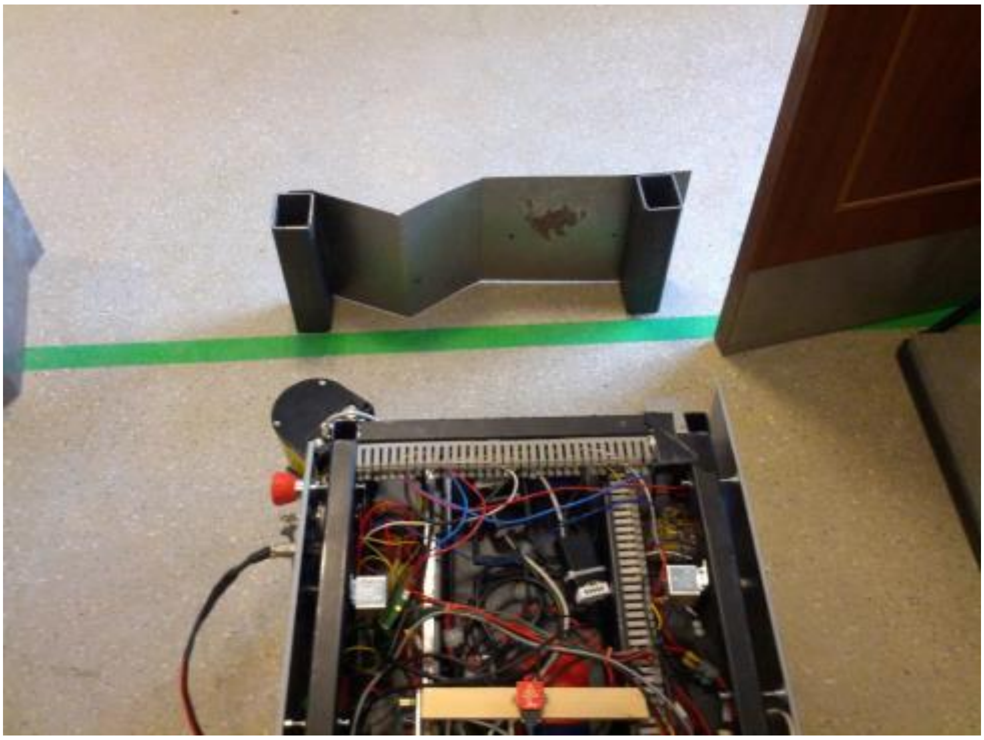


Figure 106: Metal prototype 7

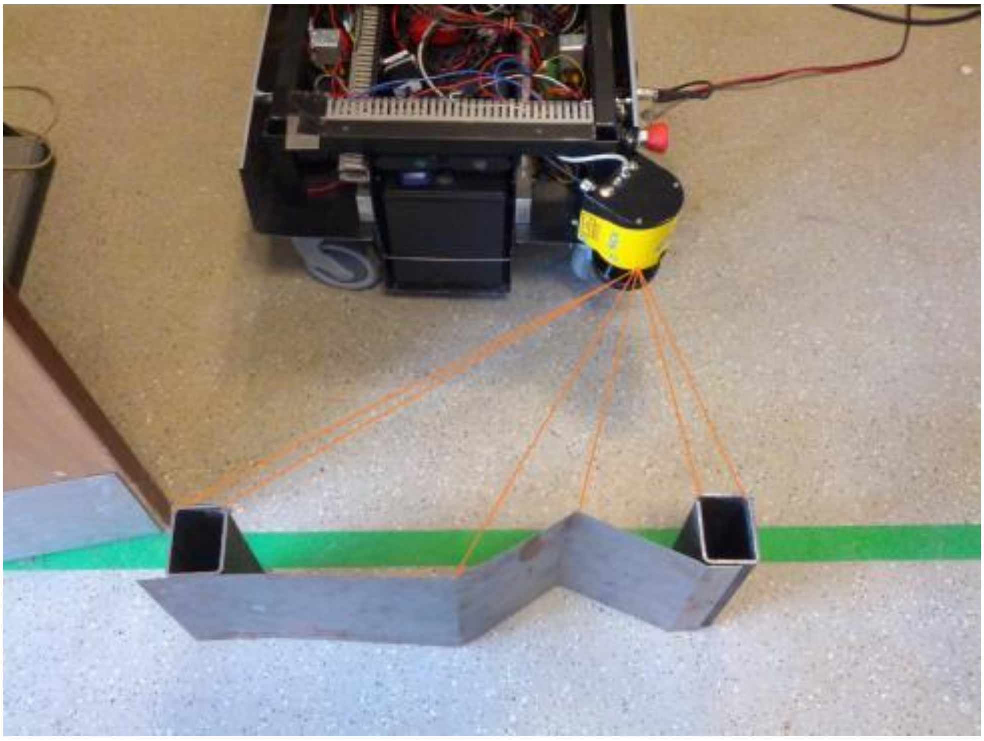


Figure 107: Metal prototype 8

Conclusion

The prototypes where the bend is located on the side gave the best results. And this was because the sensor is aligned closer to the bend which means there will be no line of sight issues when the robot moves closer to charge..

We can take the two vectors from the triangle and where they come together we draw a line. From this line we have the perfect angle of approach, using the principle of minimizing the offset as fast as possible. With a PID controller principle we can get in a straight line for the docking. With the Constant Track Error principle we implement the PID controller to get rid of the error.

Chapter 7: Used Logic and coding

The communications with the modules/robot/loading dock need to run on external logic. The robots' main CPU usage will be on an average of 90% when using the Kinect sensor. In order not to put too much strain on the main CPU we will add compact, cheap logic in order to take care of extra inputs, outputs and communication about the charging. This logic will either be the Raspberry Pi or an Arduino board. They are both compact and cheap, perfect for our external logic modules.

Raspberry Pi

The Raspberry Pi is an ARM based computer running under GNU/Linux. It is a very compact and cheap solution while still being versatile. We will use the Raspberry Pi for the communication with the loading dock to the robot and server, and to control some I/O's.

source: Wikipedia

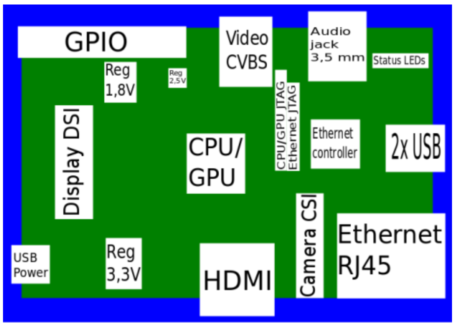


Figure 109: PI-layout

Figure 108: Raspberry PI

Figure 110: Raspberry PI logo

Source: http://en.wikipedia.org/wiki/Raspberry_Pi

Test coding

If you are not used of working in a Linux environment it is actually quite challenging to code on a Raspberry Pi. But as we had the internet to our disposal we found plenty of guides.

ARDUINO

Arduino is an open-source physical computing platform based on a simple i/o board and a user friendly development environment that implements the Processing/Wiring language. Arduino can be used to develop stand-alone interactive objects or can be connected to software on your computer.

(Scource: arduino.cc)

ARDUINO™

Figure 111: Arduino logo

ARDUINO DUEMILANOVE

Duemilanove (see picture down below) is the enhanced release of the Diecimila Arduino. In addition to the previous new features (auto-reset function, extended power connector, built in led, USB overcurrent protection) there is automatic power selection. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

(Scource: arduino.cc)

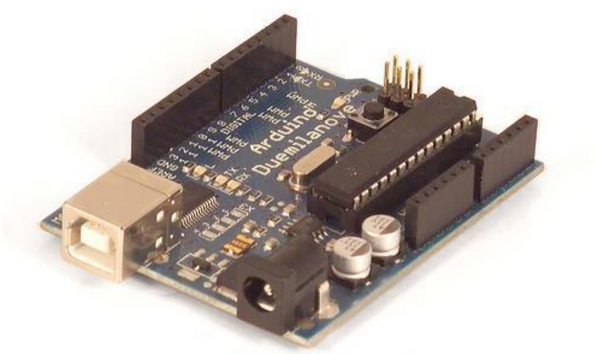


Figure 112: Arduino board

Test coding

The first few tests we made to get familiar with the Arduino.

Blink light

```
int pushButton = 0;
void setup()
{
 Serial.begin(9600);
pinMode(13,OUTPUT);
pinMode(12,OUTPUT);
pinMode(pushButton,INPUT);
}

void loop()
{
 digitalWrite(13,HIGH);
 digitalWrite(12,LOW);
 delay(1000);

 digitalWrite(13,LOW);
 digitalWrite(12,HIGH);
 delay (1000);

 int buttonState =digitalRead(pushButton);
 Serial.println(buttonState);
 delay(1);
}
```

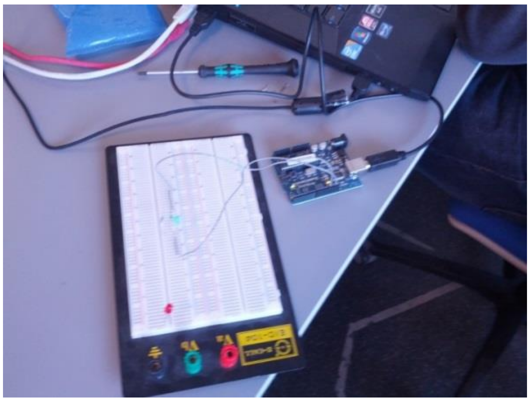


Figure 113: Coding Test 1

PWM

```
int led = 3;//pwm pin nr3
int brightness = 0;//start brightness 0
int fade =5;//brightness ++points out of 255


void setup()
{
 pinMode(led,OUTPUT);//set led_pin to output
}
void loop()
{
 analogWrite(led,brightness);//write brightness value to led
delay(30); //wait 0.03s
brightness = brightness + fade;//recalculate brightness

if (brightness==0||brightness==255)//if brightness = 0 or brightness = 255
{fade =-fade;}                    //change sign of fade
}
```

Comparison Raspberry Pi and Arduino Duemilanove

When we compare the Raspberry Pi with the Arduino Duemilanove we can see some big differences.

- There is a significant price difference, the Duemilanove costs 1/3 or less of a Raspberry Pi.
- The Duemilanove can be configuared more easily.
- The Duemilanove can be programmed more easily.
- The Raspberry Pi has more computing power for the Raspberry Pi.
  (But do we need this ?)
- The Raspberry Pi dissipates more heat.

If we don't need the computing power from the Raspberry Pi we should better use the Arduino Duemilanove, and so we did.

Conclusion: Arduino

Raspberry Pi & Arduino comparison	Pros	Cons
Raspberry Pi	Faster processor More RAM More flexibility of I/O	Expensive Hard to configurate Hard to program Takes more time to learn More heat dissipation
Arduino	Cheap Easy to configurate Easy to program Takes little time to learn Less heat dissipation	Slower processor but fast enough Less RAM but enough Less felixibility of I/O but enough

Table 5: comparison Arduino - Raspberry pi

The Arduino is cheaper and easier to program which are the two main reasons why we choose it.

Coding

We will now clarify how the programming is done and how it works.

Working program

The actual program can be found in the appendix.

Hearth beat

During a meeting we came to the conclusion that we needed to add a "hearth beat" in our code. This means the Arduino will constantly send data so the main program of the robot can recognize changes really simple and can react to these changes. This way we will also know the Arduino is online and doing its job while providing the necessary data. How we applied the hearth beat code:

```
unsigned long Current_Time = millis();

 if (Current_Time - Previous_Time > HB_delay)
{

  Serial.println(heartbeatmessage);

  Previous_Time = Current_Time;
}
```

Figure 114: Heartbeat code

Serial messages

We edited the program so it is possible to change all the messages at the top of the program. This makes it easier to find or disable these messages. How we applied the serial messages code:

```
//serial messages

char heartbeatmessage[]                          = "heartbeat"                                          ;
char Start_Message[]                             = "start"                                              ;


char Lift_Going_Up_Message[]                     = "Lift Going Up"                                      ;
char Lift_Already_Up_Message[]                   = "Lift Already Up"                                    ;
char Lift_Going_Down_Message[]                   = "Lift Going Down"                                    ;
char Lift_Already_Down_Message[]                 = "Lift Already Down"                                  ;
char Charging_Stop_Button_Pressed_Message[]      = "for charging please release the -Stop charging button-"  ;
char Charging_Started_Message[]                  = "Charging Started"                                   ;
char Charging_Stopped_Message[]                  = "Charging stopped"                                   ;
char Lift_Up_Message[]                           = "Lift is up"                                         ;
char Lift_Down_Message[]                         = "Lift is Down"                                       ;
char Charge_Stopped_By_Button_Message[]          = "charging stopped by button"                         ;
char Robot_Stopped_Message[]                     = "Robot Stopped"                                      ;
char Stop_Robot_Command[]                        = "Stop Button Pressed Stop The Robot"                 ;
char Robot_Started_Message[]                     = "Robot Started"                                      ;
char Start_Robot_Command[]                       = "Start Button Pressed Start The Robot"
```

Figure 115: Serial message code

Character protocol

When the robot is in position a character has to be send in order to know that the robot is in the actual position, therefor we had to make a protocol for this.

First we will send a Char to the Arduino. After receiving this Char the Arduino knows that the MiR100 is in position. After this the next Char will be sent with the different tasks.

Examples of these tasks:

- Charging: -c-
- Stop charging: -s-
- Lift up: -u-
- Lift down: -d-

The MiR100 also has two pushbuttons , these are used to manually stop and start the robot. If one of these buttons are being pushed the Arduino sends a text through serial communication. The text sent through the serial communication can always be changed.

Emergency stop monitoring

For the emergency stop we use an optocoupler. This makes us able to monitor the safety circuit with the Arduino since it will be a 5V signal and the optocoupler also isolates both circuits from eachother which means another built in safety.

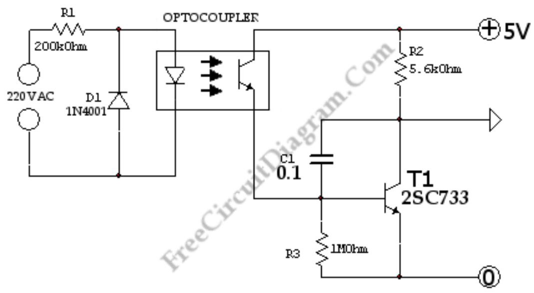


Figure 116: Optocoupler schematic

Testing board

This testing board includes four LEDs and five normal pushbuttons. We could simulate the lift going up, down, stop and charging. This was a nice way to test out our code before implementing it into the robot. This way we could still make some changes without losing too much time.

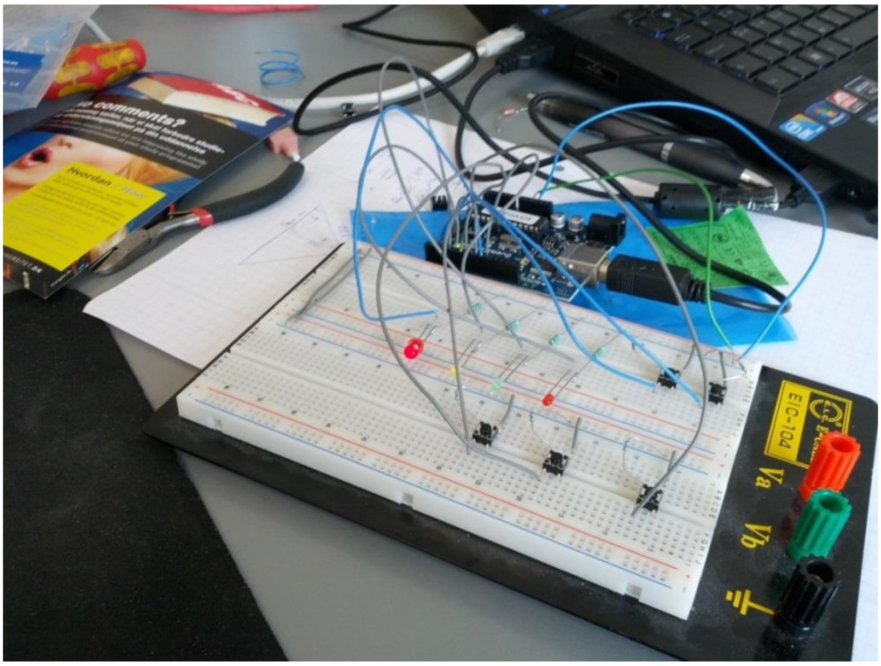


Figure 117: Test code 2

Coding flowcharts

We realised that if we would use the plain code it can get really messy to explain how it works. So that is why we use the flowcharts so the code is more easy to understand. The actual code can be found in the appendix.

Main program

When the Arduino first starts it will initialize all settings for the inputs and outputs, it will also enable the serial communications. After this is done it will start the main programmed loop. In this loop all inputs will be read and the microcontroller will check for serial messages. If the Arduino receives a serial message it will read this message and go to Casing.

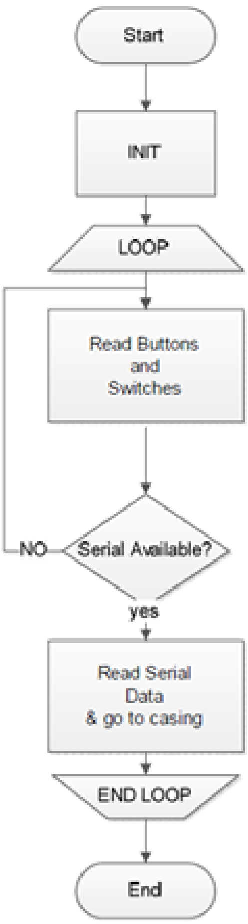


Figure 118: Flowchart 1

Charging case "c"

In this case the microcontroller will first check if the robot is in position. After that it will check if the stop charging button is pressed. If the robot is not in position or the button is pressed the Arduino will send a message back. If the robot is in position and the button is not pressed the Arduino will activate the pins in order to charge the battery.

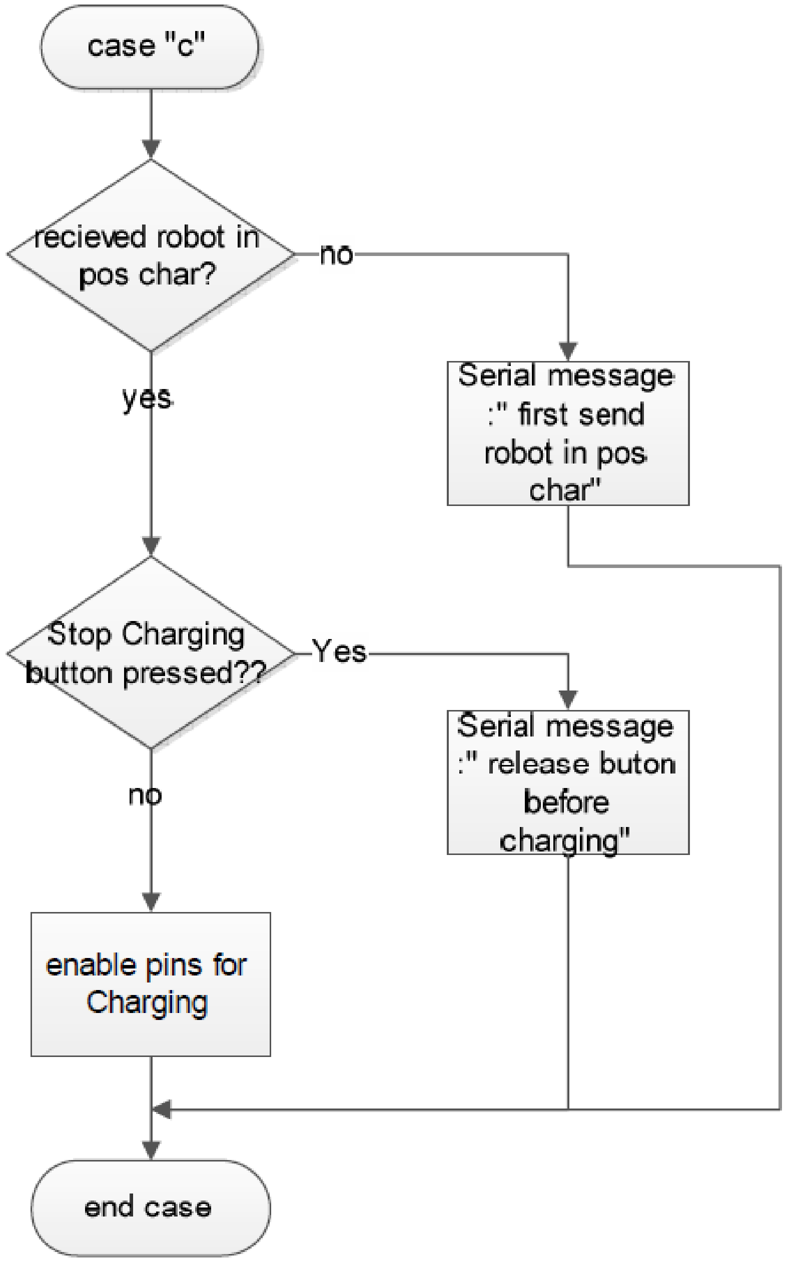


Figure 119: Flowchart 2

Robot in position case "x"

This case shows if the robot is in position or not. When the "x" is received a variable will be set to high.

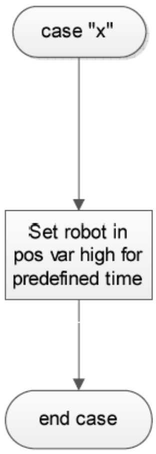


Figure 120: Flowchart 3

Start charging case "s"

In this case the robot will start charging. But the robot in position char must be received first. If this char is received the Arduino will activate the pins for the relay.

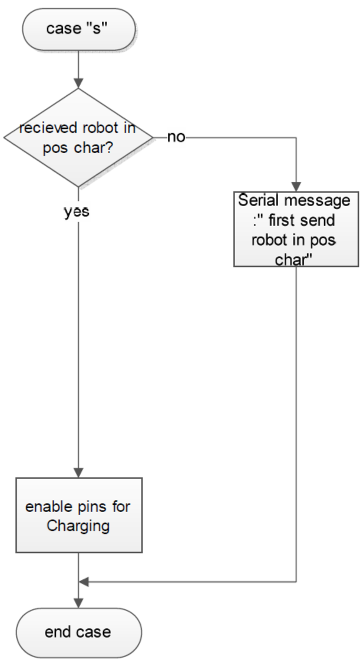


Figure 121: Flowchart 4

Lift up case "u"

when this case is executed the Arduino will first check if the In position char was send. If this didn't happen a message will be send back. If the robot was in position and the lift was not already up, then the Arduino will activate the according pins so the lift can go up. Otherwise it will send an error message back.

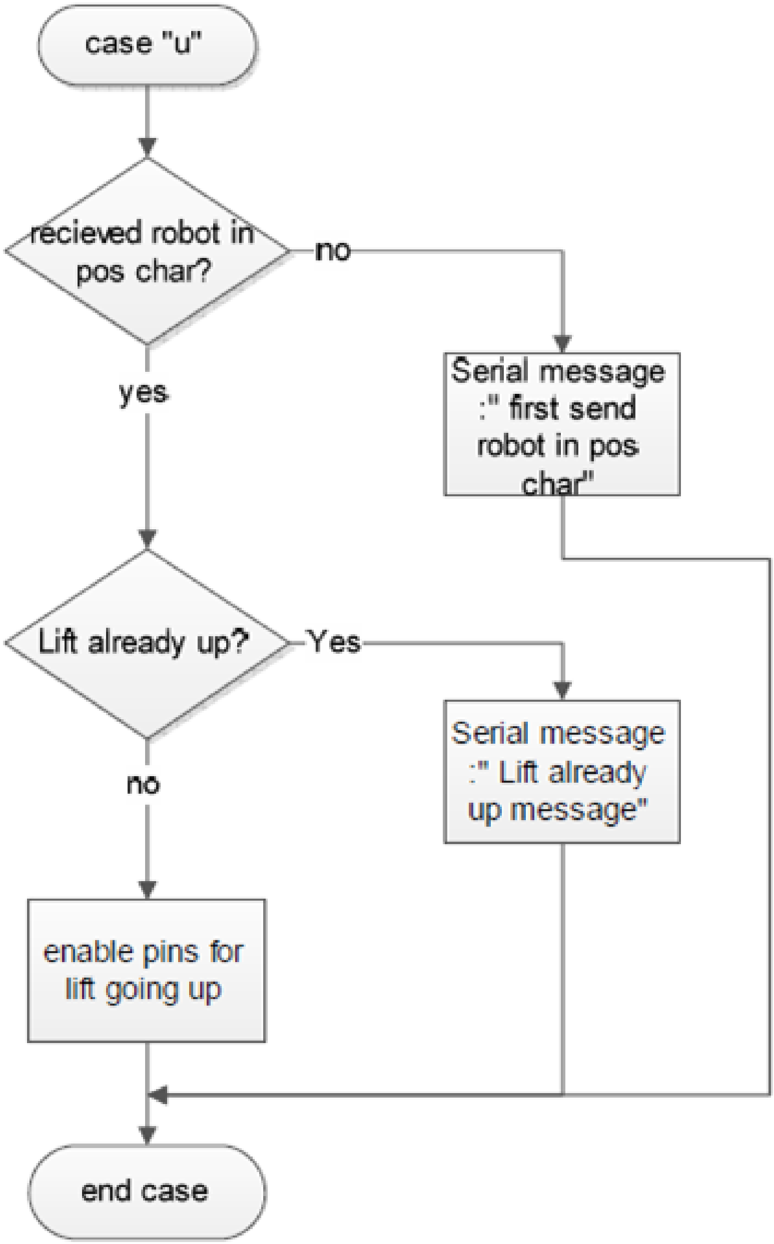


Figure 122: Flowchart 5

Lift down case "d"

This is exactly the same flowchart as the one from the "u" case, the only difference is that this one is for the downwards motion of the lift.

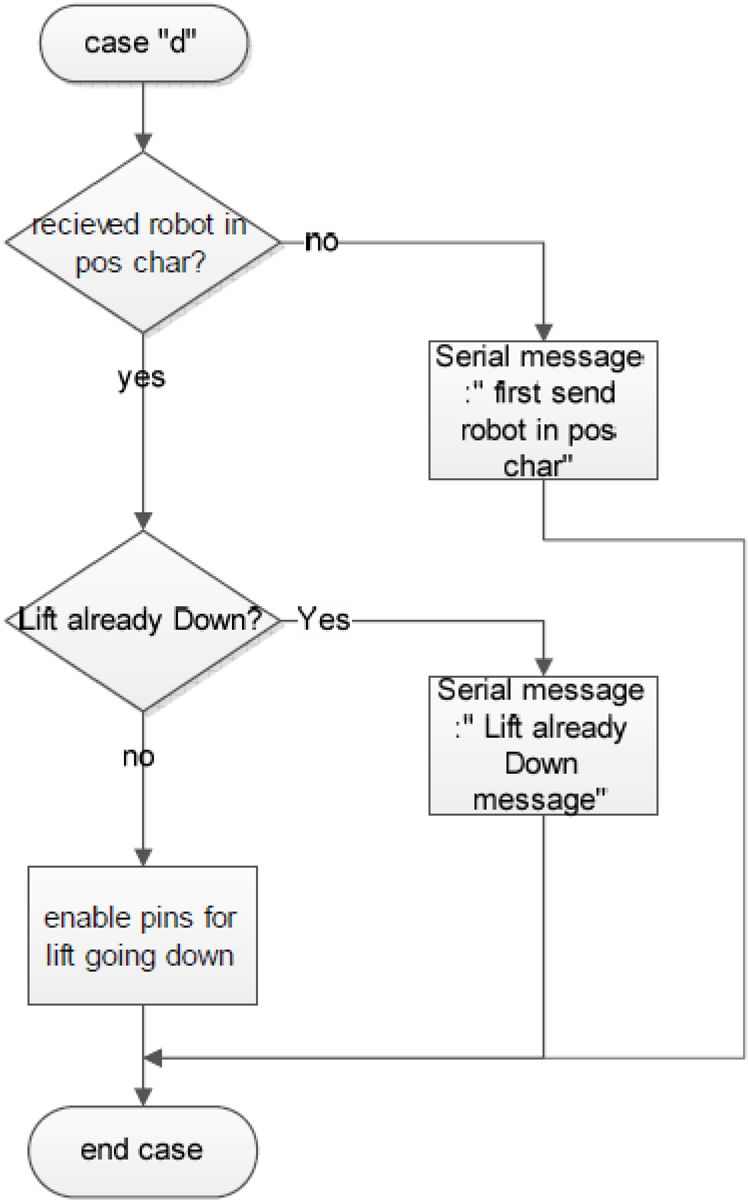


Figure 123: Flowchart 6

Integration of cases

This is a small flowchart to show that after the casing, the Arduino continues the main program.

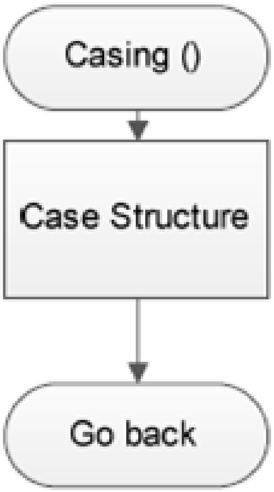


Figure 124: Flowchart 7

Chapter 8: Production

In this chapter we will show the entire production process of the alpha version from the charging dock.

Charging station

The charging station which is our main subject for this thesis will now be built after all the brainstorming and sketching.

List of electrical components

These are the used components of the charging station.

Power supply

The SMC-HF 1600 power supply which will be provided by GNB.

Figure 125: SMC-HF 1600 power supply

Charging contacts

The charging contacts are made in the mechanical workshop of the school and will be mounted on the bottom of the robot in order to make a good contact with the aluminium contacts.

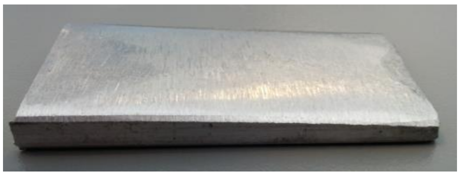


Figure 126: Prototype contacts

Aluminium contacts

The aluminium contacts are required to make a good contact with the robot, these are also handmade in the workshop. They are mounted left and right from the guidance. The first picture shows the aluminium contacts combined with the guidance and the second picture shows the actual guiding piece which will be mounted under the robot.

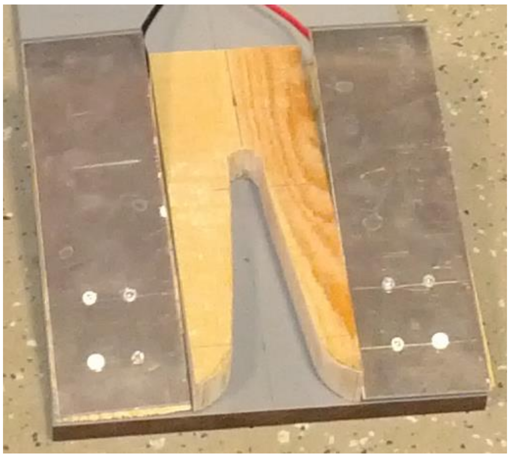


Figure 128: Prototype contacts 2

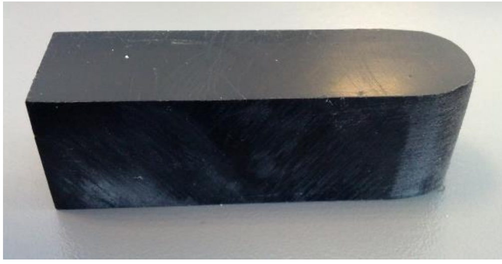


Figure 127: Guidance prototype

Wiring

For charging we will use 2.5mm² souple core with red and black insulation.

Figure 129: H07V-K

Arduino

We have discussed the Arduino before, we will be using the Duemilanove.

H-bridge

A H-bridge is needed when the lifting system is in action in order for the motor to work.

		
	Without Fan	**With Fan**
Voltage Range	6V – 24V (28V absolute max)	Same
Current (H-bridge)[1]	20A cont. at 100% duty cycle 17A cont. at 70% 45A 5 second peak	25A cont. at 100% 20A at 70% 45A 5 second peak
Current (each half-bridge)*	Same as above	Same as above
Current (ganged half-bridge)*	40A cont. at 100% 35A cont. at 70% 70A 5 second peak	48A cont. at 100% 38A cont. at 70% 70A 5 second peak
PWM frequency	DC – 20kHz	DC-20kHz
Current Sense Output	Vc = I * 0.075 Vc = .0.75 at 10A Vc = 2.99V at 40A	Same
Input voltage levels PA,PB,EA,EB	2.5V – 5.5V = logic high 4.5V – 28V for HV version <1.7V = logic low	Same
Size	2.5" x 2.25" x 0.5"	2.5" x 2.25" x 0.75"
Weight	37g	61g
Mounting	4x - 4-40 or M2.5 bolts	Same
Fan	None	50mm x 10mm – 12V

1. Your actual current capacity will vary based on the type of load, the length and size of wires, power supply capability and other factors.

Table 6: H-bridge

(source: http://www.robotpower.com)

Charger sketches

The amount of detail required is increasing, that is why the following sketches have colour in them to distinguish the different kind of objects. These sketches will mainly give information about the charger and the guidance.

Charger and guidance sketches part I

The following image is an overview of two different kinds of charging and guidance.

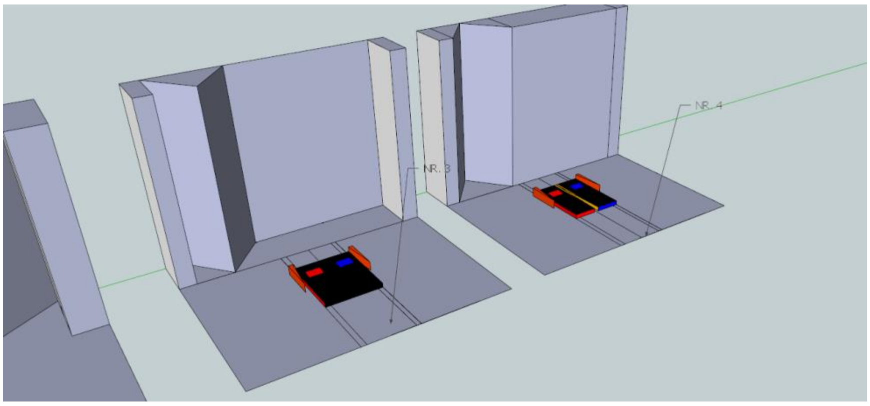


Figure 130: Sketch guidance 1

On the first image we observe the two charging contacts which are red and blue, and the guidance which is orange. The two charging contacts will be held up by a spring so if the robot drives over them, optimal contact for charging will be maintained.

The second image also shows a small guidance addition in yellow, the robot drives in and gets guided by the yellow guidance and then fixed by the orange guidance. Please keep in mind that these are still concepts and at this point we were still looking for a good guiding system..

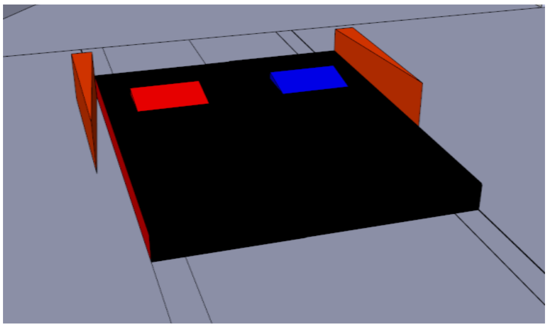


Figure 132: Sketch guidance 2

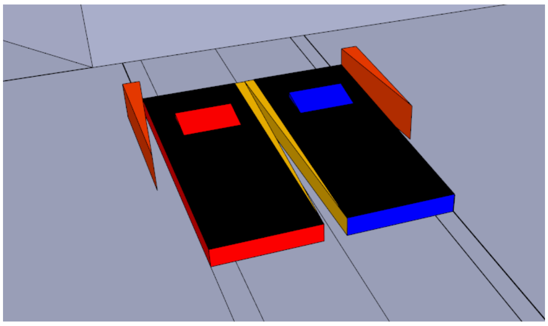


Figure 131: Sketch Guidance 3

Charger and guidance sketches part 2

The following image is an improved version of the previous sketches. The guidance has more curve in order for the robot to have less friction while being guided.

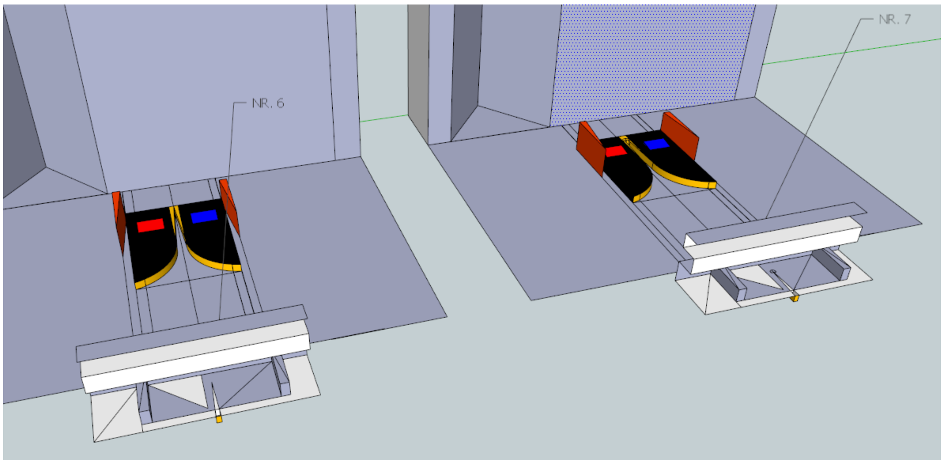


Figure 133: Sketch guidance 4

The first image shows the back of the robot where the guidance piece is placed in order to be guided by the guidance system in the second image. The orange holes leave a second option for guiding the robot but the friction of these might not be optimal for the structural integrity of the robot. The yellow guide is placed on the battery cage which is the most robust part of the robot, that is why the yellow guidance is the most viable option.

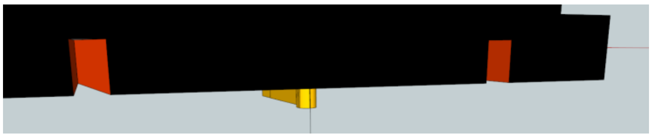


Figure 134: Sketch guidance 5

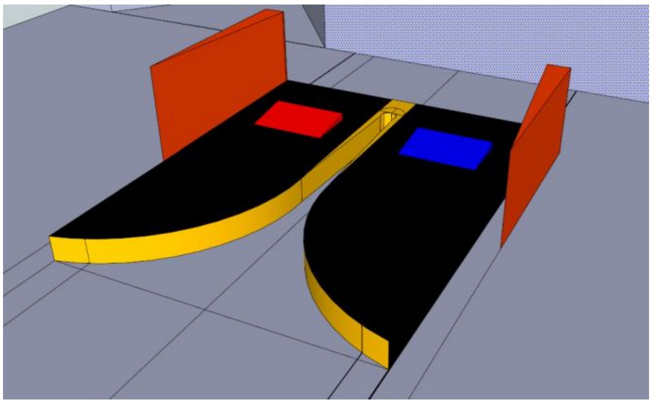


Figure 135: Sketch guidance 6

3D CAD Drawings

We will first show you all the different drawings separately and then show the assembled 3D CAD Drawing.

Bottom plate

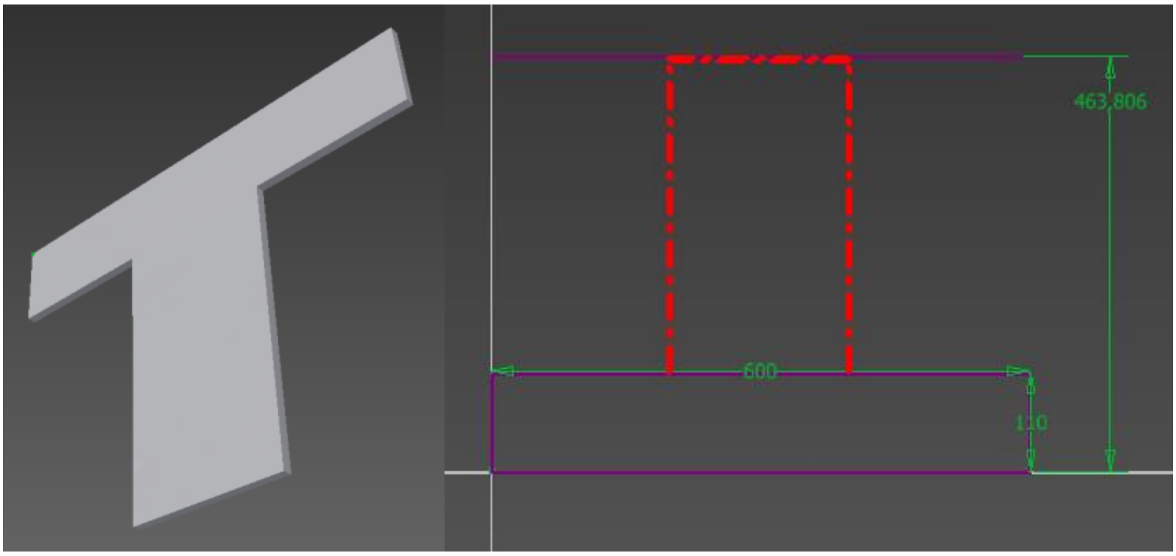


Figure 136: CAD drawing bottom

Side plates

Figure 137: CAD drawing side

Front plate

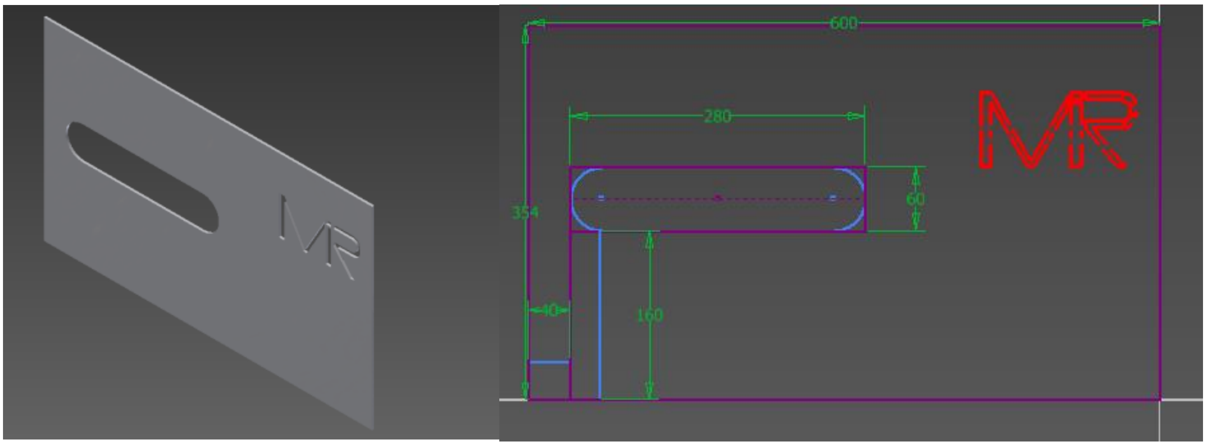


Figure 138: CAD drawing front

Top plate

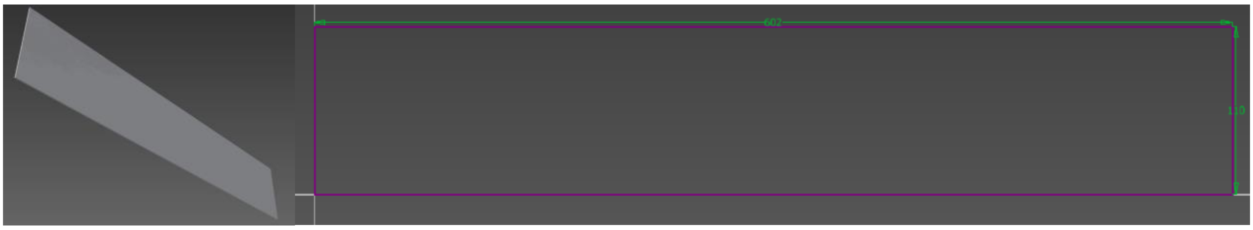


Figure 139: CAD drawing top

Ramp with hole for spring

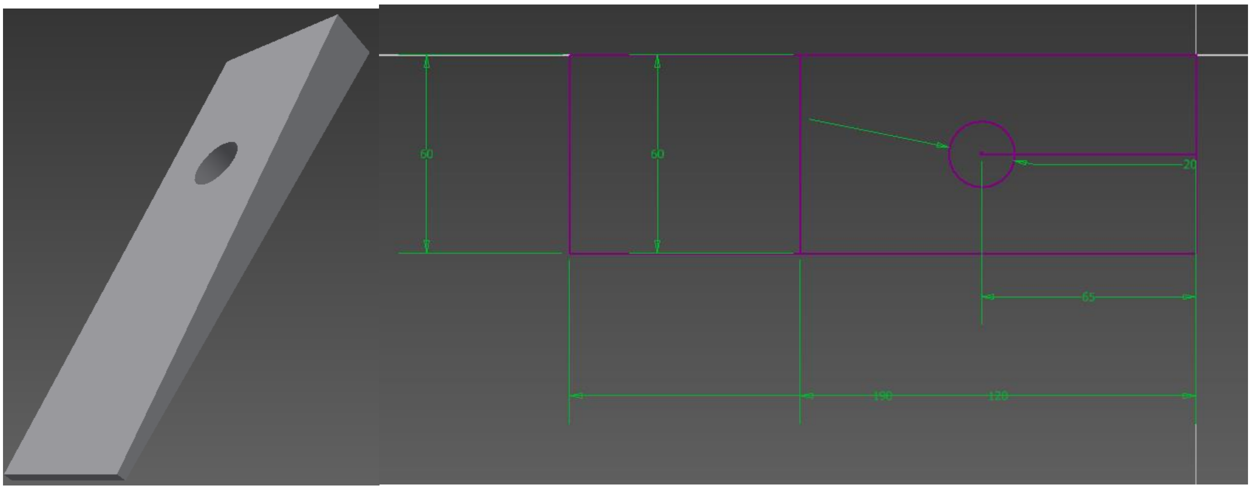


Figure 140: CAD drawing ramp

Springs

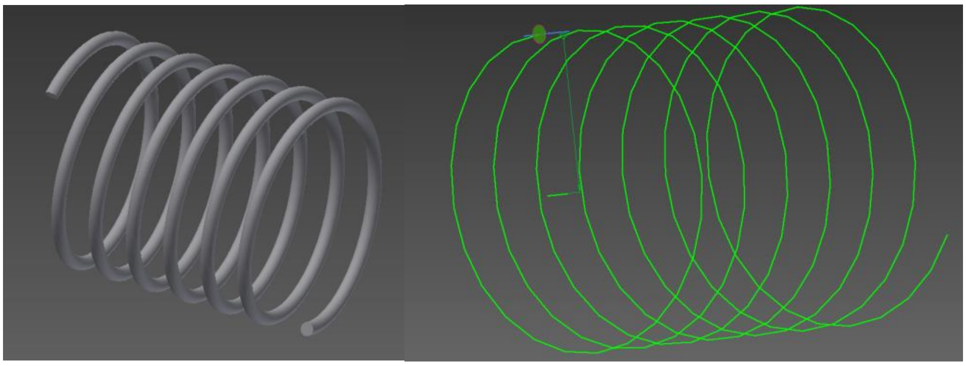


Figure 141: CAD drawing spring

Aluminium contacts

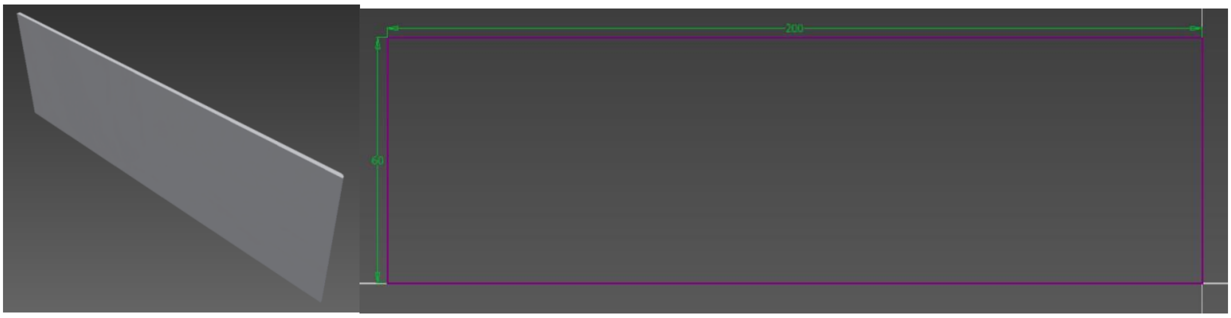


Figure 142: CAD drawing alu contact

Guidance

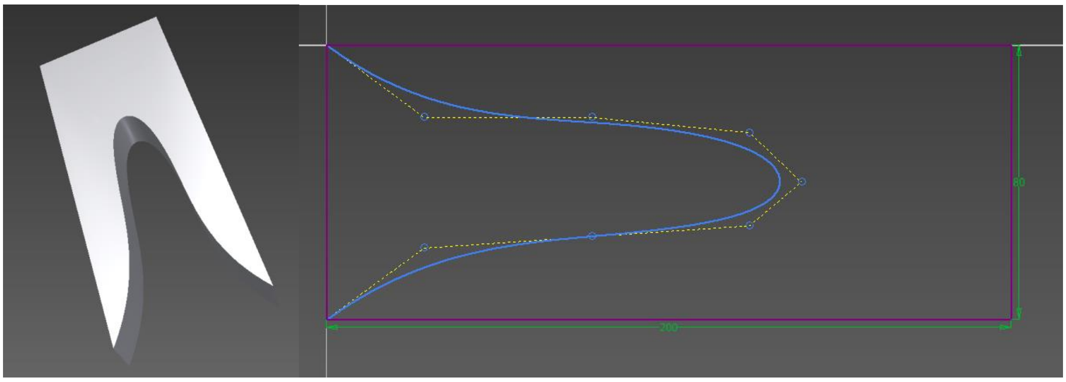


Figure 143: CAD drawing guidance

Assembly of parts

On the next two pages you will be able to see all the parts assembled into the charging station. The first image represents a more simple drawing while the second one shows a more 3D rendered image.

When we put all the parts together we can see our alpha version of the charging dock come alive. All what is left to do now is to actually build it.

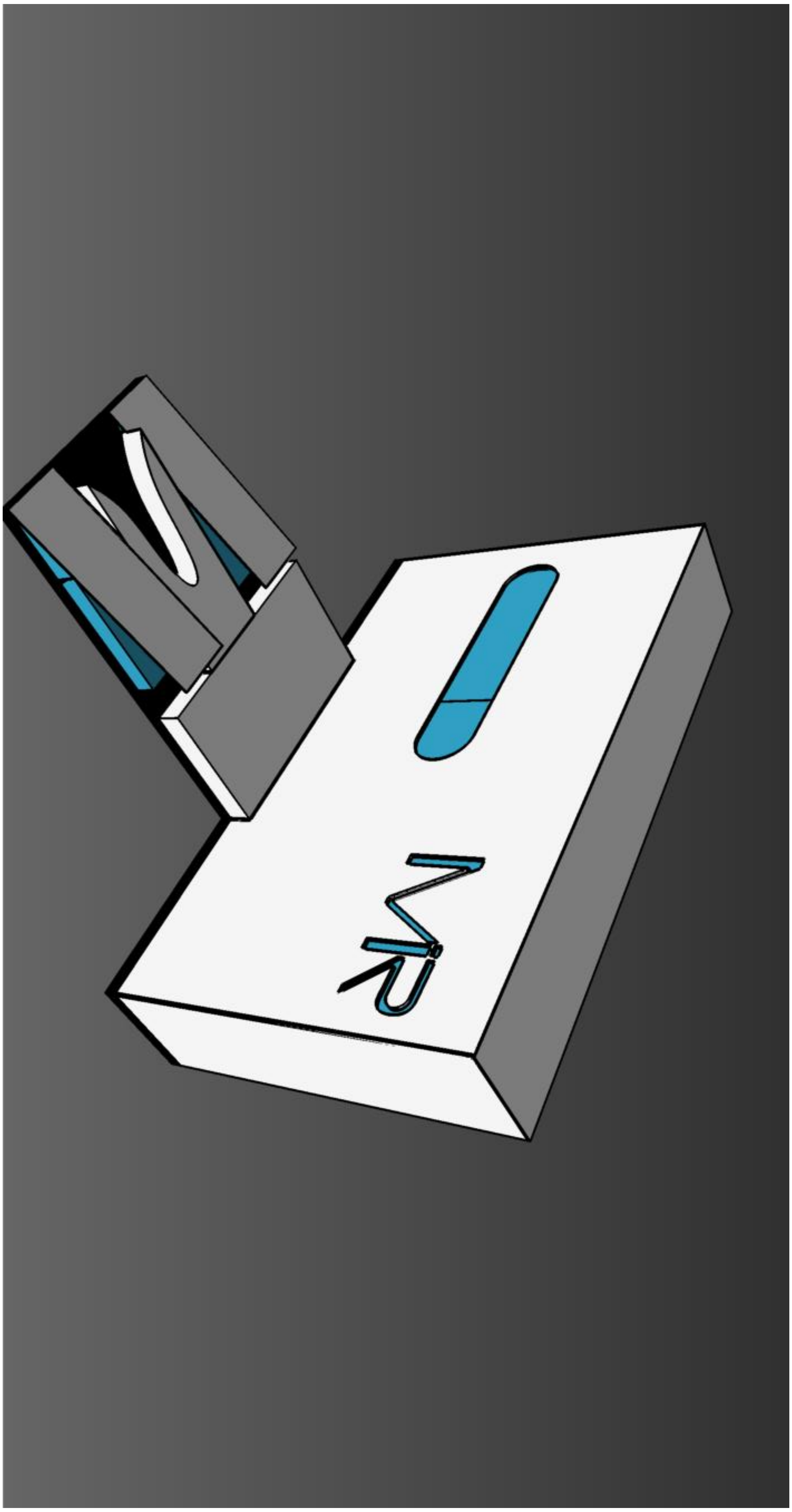


Figure 144: example charging dock 1

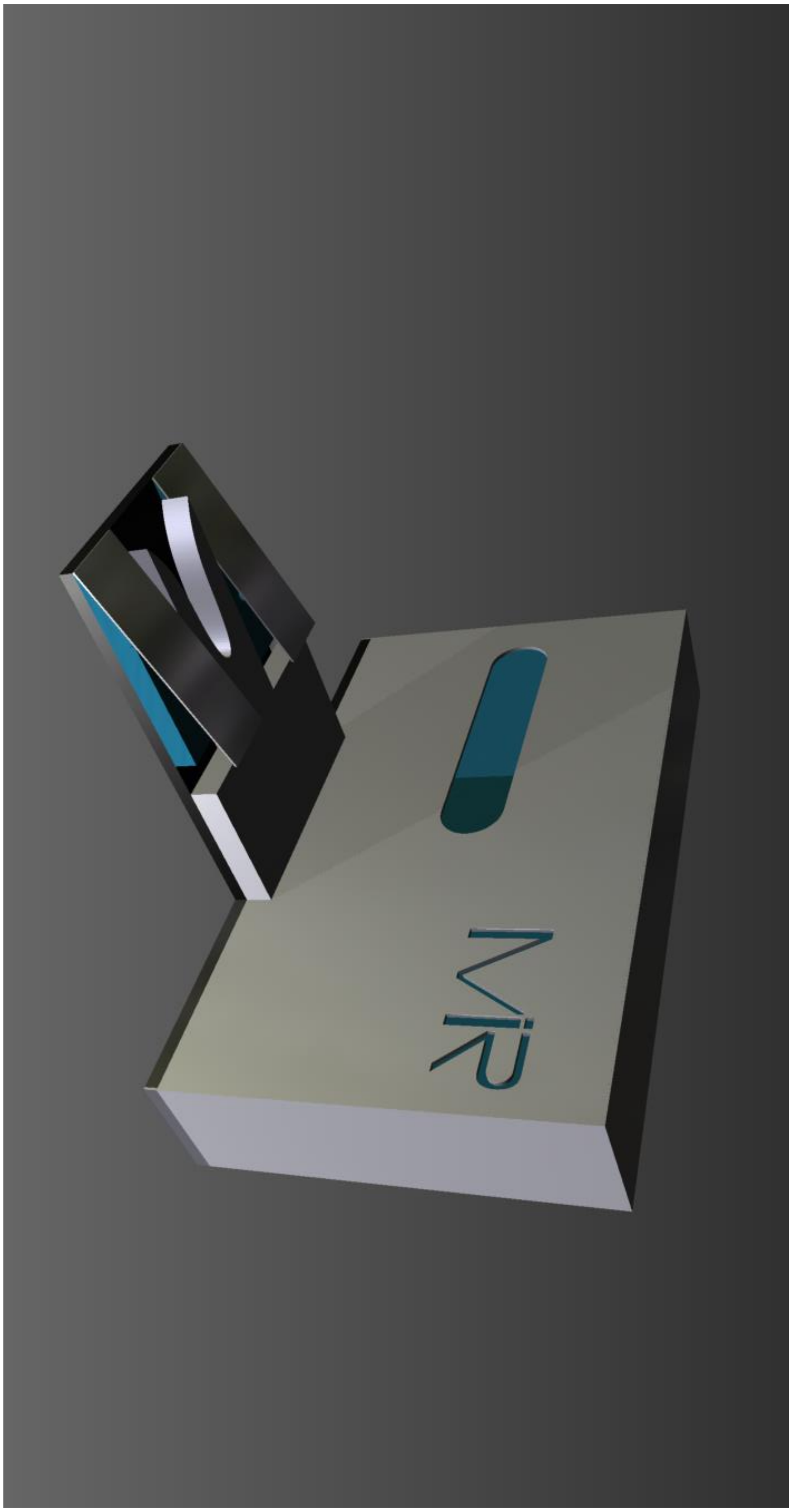


Figure 145: Example charging dock 2

Alpha version

After all the sketches and 3D rendering it is time to build the alpha version of the MiR100 charging station.

Assembly of parts

The first image is the front plate, the second image is the front plate being cut and the third image shows the cutting of the side and top panels. The fourth image shows how the rectangular shaped hole from the front plate was grinded. And the last two images show the final result.

Figure 146: Construction of last prototype

Final result

A charging dock can't charge without any charging contacts, and it is no dock if there is no guidance. In the image you can see the finished product, the wooden guidance and the aluminium contacts with springs underneath.

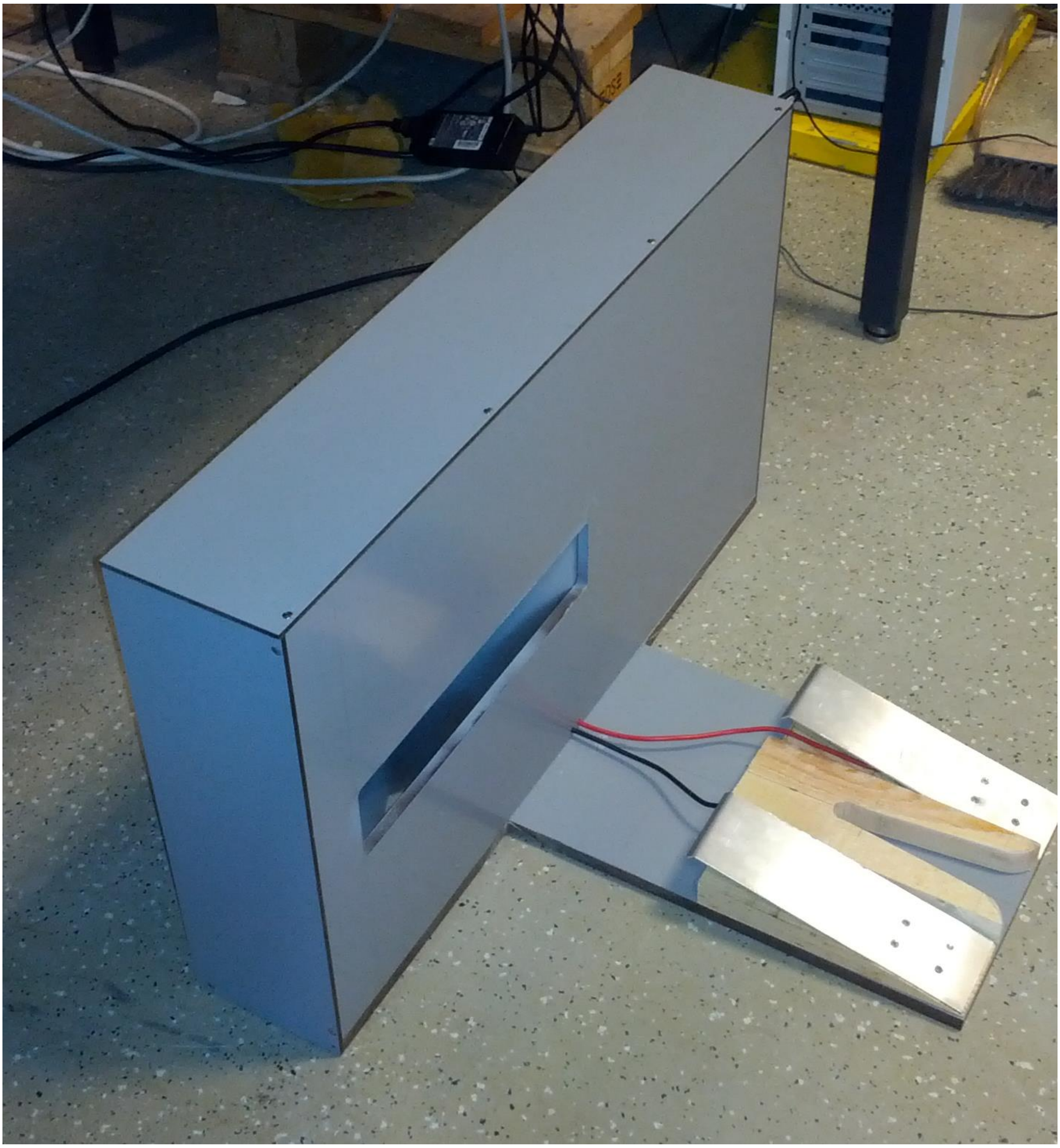


Figure 147: Charging dock prototype 1

Figure 148: Charging dock prototype 2

Loading station

The loading station would have been added on top of the charging dock in order to hang the modules. To get a better understanding of the height of these modules the following pictures might help. These sketches are made by the designers Anders and Frank, they have been working on the design part of the robot.

Robot:

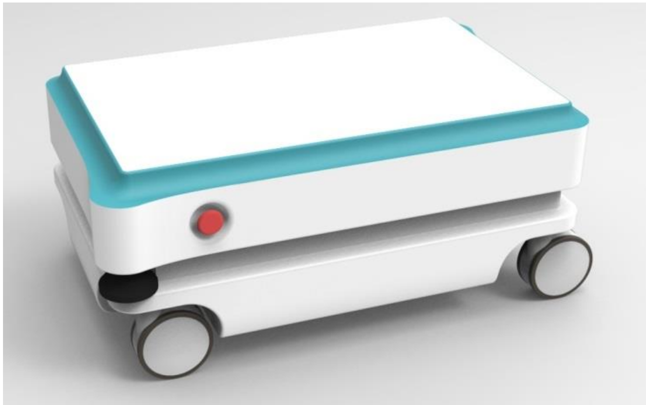


Figure 150: MiR concept 2

Robot & module:

Figure 149: MiR concept 1

Robot & module:

Figure 151: MiR concept 3

Both (Loading dock)

If we had more time we could definitely finish the loading station and we would be able to combine it with the charging station to become the loading dock. The research has been done, the prototype only needs to be designed and created followed by some basic testing.

Final Conclusion

Our objective was to research and develop a dock from scratch where the robot could charge itself, load modules and unload modules. We successfully completed the research and development process for the charging station. The robot is able to recognize the charging station in order to maneuver towards it and make contact with the charging contacts.

The process started by generating ideas. Ideas of what the modules could contain and for what they were being used for. Ideas of the different drives, of the lifting methods and of the charging methods. Followed by sketching the first concepts and building the first prototypes which made the process easier because we had something to work with and improve. After building the first prototypes we had to refine them by improving the concept. We went back to the drawing board, looked at the competitors, made Lego models and made new prototypes with this information. At this point we were progressing really fast and determined the triangular shape of the laser guidance. Afterwards we started with the sketches of the alpha version and also concluded that Arduino was the best choice for our needs when it came to programming. Followed by the production of the alpha version of the charging station.

The results of our research ended up producing the alpha version of the charging station. The main findings we had where that the trial and error principle, sketching – building – testing, rinse and repeat. Proved to be a very efficient method in terms of finding the errors of the different kind of prototypes.

During our research we learned that: we had to be open minded in the idea phase; we had to keep it simple; patience is a virtue.

If we had to start over from scratch knowing what we do now, we would definitely start sooner with building the prototypes because we got most of our results from testing and improving them.

If we had more time we would be able to complete the loading station, where the modules could be mounted on. The basic idea of the loading station is to combine, mount it on top of the charging station to get the loading dock as a result. This is now future work, somebody else can continue with our work.

The project ended with basic testing of the charging station. The robot had to navigate towards the charging station and get a good contact in order to charge. After all it was a very interesting experience to work with mobile robots. We think that there is a great future in these mobile robots because nowadays more and more tasks are being executed by robots. Ranging from the easiest to the most complex actions, these robots will find their way to the industry and will hopefully not disappoint.

Appendix

Power supply datasheet

The size of the battery charger with regard to the size of the battery and available charging time.

Type description				Recommended battery size standard charger		Recommended VRLA battery size standard charger		Number of phases / mains current (A)		
				Charging time (IUIU)						
Type	Batt. V.	Amp	Cabinet type	8 hours cap. Ah/5h	12 hours cap. Ah/5h	12 hours cap. Ah/5h	14 hours cap. Ah/5h	1-phase 220-230 V	3-phase 380-400 V	Weight kg
SMC-HF	12	20	P4	125	237	117	140	3,5		1,5
		30	P4	185	350	175	205	4,5		1,5
		50	P4	310	590	292	350	4,5		1,5
		60	P6	371	710	350	420	9		3,1
	24	20	P4	125	237	117	140	3,5		1,5
		30	P4	185	350	175	205	4,5		1,5
		45	P6	281	530	265	315	9		3,1
		60	P6	371	710	350	420	9		3,1
		80	E3	495	947	460	560	10,8		10
		100	E3	619	1 184	585	700	13,6		10
		120	E3	743	1 421	700	840	16		10
		150	T2	929	1 776	880	1 050		7,3	15
	36	20	P4	125	237	117	140	4,5		4,5
		40	P6	248	474	233	280	10		10
		60	E3	371	710	350	420	12		10
		80	E3	495	947	460	560	16		10
		100	T2	619	1 184	585	700		7,2	15
		130	T2	774	1 480	765	910		9,4	15
		150	T2	929	1 776	880	1 050		10,8	15
	48	15	P4	93	175	88	105	4,5		1,5
		30	P6	185	350	175	205	9		3,1
		40	E3	248	474	233	280	10,4		10
		60	E3	371	710	350	420	15,7		10
		80	T2	495	947	460	560		7,7	15
		100	T2	619	1 184	585	700		9,6	15
		130	T2	774	1 480	765	910		12,5	15
	72 - 80	40	T2	248	474	233	280		6,4	15
		60	T2	371	710	350	420		9,7	15
		80	T2	495	947	460	560		16	15

Dimensions (mm) and data:

Cabinet	Height	Width	Depth
P4	112	230	75
P6	135	229	87
E3	425	255	90
T2	417	255	229

All cabinets are classified as IP20. Other classifications can be obtained as an option.

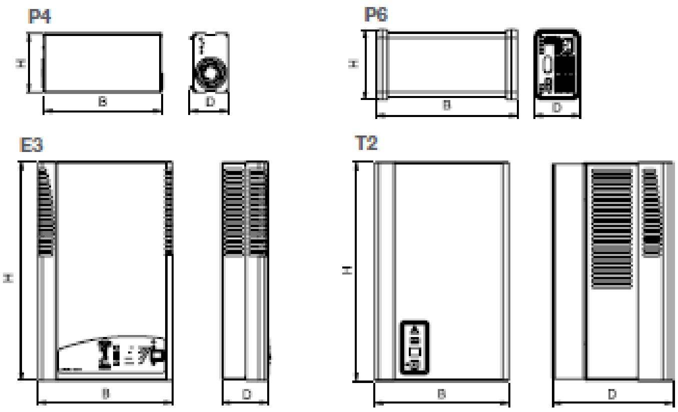
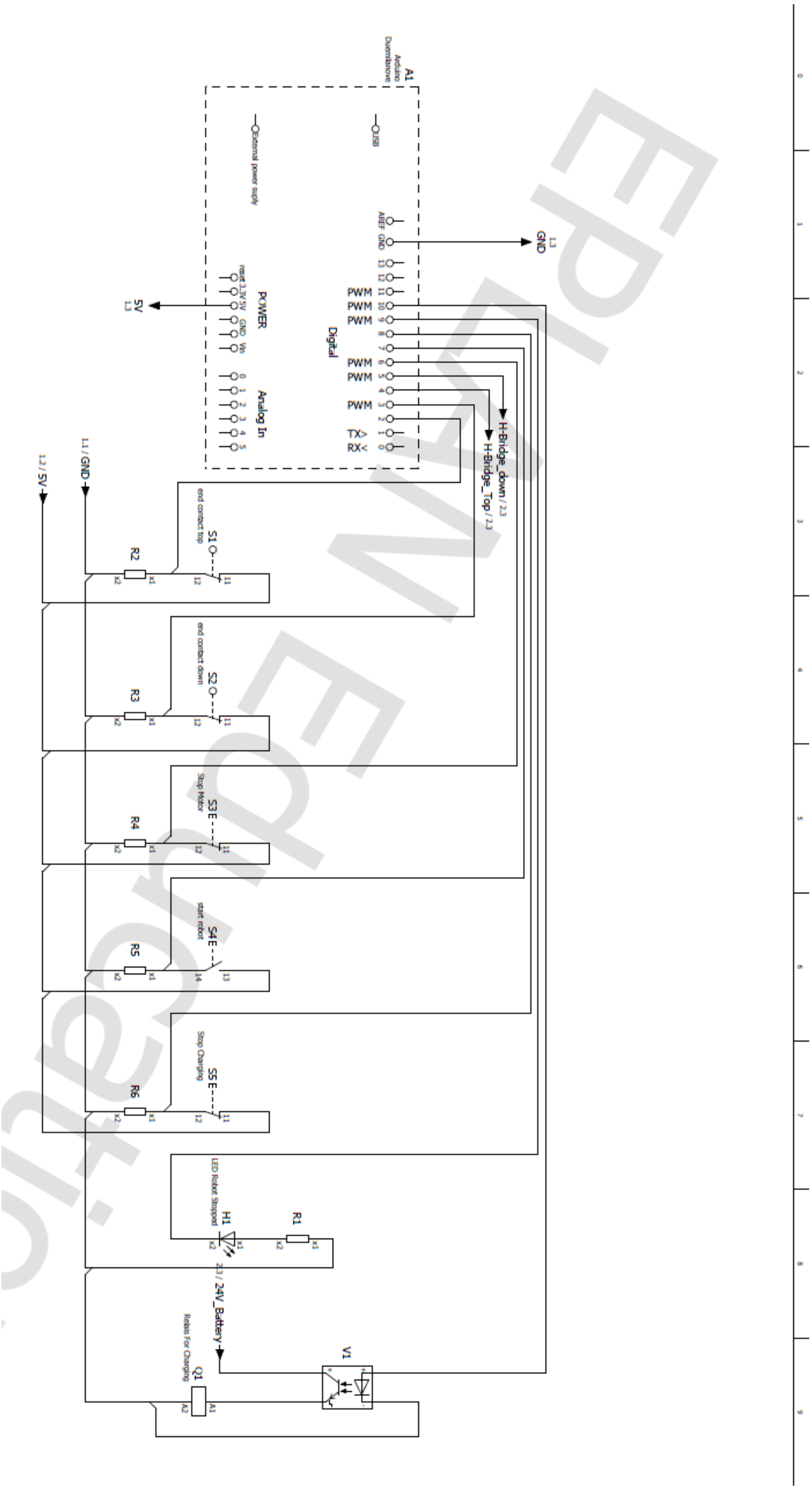
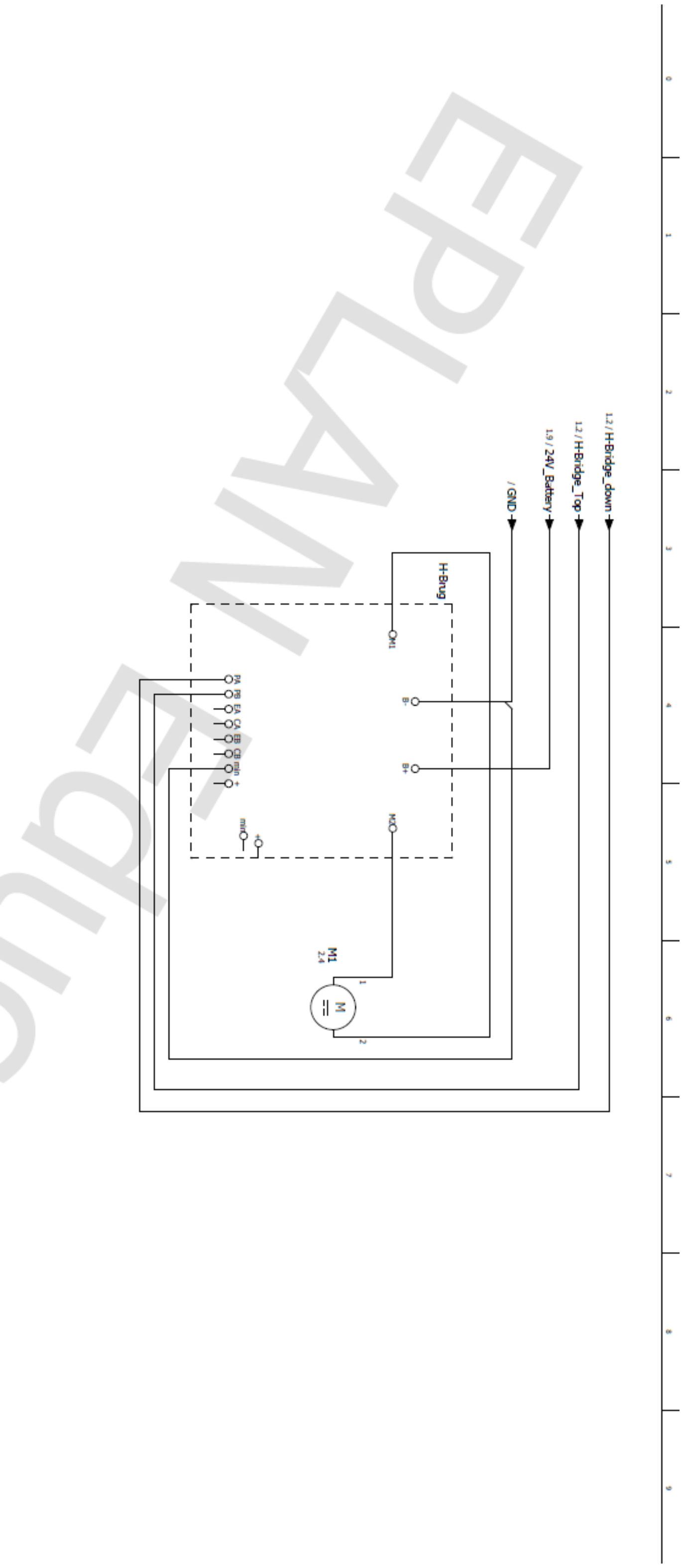


Figure 152: Datasheet Charger

ELECTRICAL SCHEMATICS




H-Brug
/ GND
1.9 / 24V_Battery
1.2 / H-Bridge_Top
1.2 / H-Bridge_down
B-
B+
M1
M1
2.4
EPLAN Edu

Main program code

```
int Lift_Up_Sens   =         2          ;      //
int Lift_Down_Sens =         3          ;      //
int Lift_Up_Act    =         4          ;      //here you enter which sensor or accumulator is
int Lift_Down_Act  =         5          ;      //attached to which pin
int Start_Robot    =         6          ;      //
int Stop_Robot     =         7          ;      //
int Stop_Charge    =         8          ;      //
int Light_Stop     =         9          ;      //
int Relais_Charge  =         10         ;      //
int H_Bridge_Enable=         11         ;      //

// variables for waiting without having to use the delay function
unsigned long Previous_Time  =        0          ;      //
int HB_delay                 =        10000      ;      // in ms (1s = 1000ms)


//serial messages

char heartbeatmessage[]                    = "h\n"                                  ;
char Start_Message[]                       = "init\n"                                      ;


char Lift_Going_Up_Message[]               = "+u\n"                            ;//
char Lift_Already_Up_Message[]             = "-u\n"                            ;//
char Lift_Going_Down_Message[]             = "+d\n"                            ;//
char Lift_Already_Down_Message[]           = "-d\n"                            ;//
char Charging_Stop_Button_Pressed_Message[] = "-c\n"                           ;//
char Charging_Started_Message[]            = "+c\n"                            ;//to disable messages put  two backslaches ( // )  in front of the message line
char Charging_Stopped_Message[]            = "+s\n"                            ;//
char Lift_Up_Message[]                     = "Lift is up"                      ;//
char Lift_Down_Message[]                   = "Lift is Down"                    ;//
char Charge_Stopped_By_Button_Message[]    = "charging stopped by button"      ;//
char Robot_Stopped_Message[]               = "Robot Stopped"                   ;//
char Stop_Robot_Command[]                  = "Stop Button Pressed Stop The Robot"  ;//
char Robot_Started_Message[]               = "Robot Started"                   ;//
char Start_Robot_Command[]                 = "Start Button Pressed Start The Robot" ;//
char Lift_is_Not_Moving_Message[]          = "lift not moving"                 ;//
char Lift_is_Moving_Message[]              = "lift moving "                    ;//
char Robot_Is_Charging_Message[]           = "robot charging"                  ;//
char Robot_Is_Not_Charging_Message[]       = "robot not charging"              ;//
char Error_Message[]                       = "-\n";
```

```
// variables needed for buttons

int lastbuttonstate_Up          =  1 ;
int lastbuttonstate_Down        =  1 ;
int lastbuttonstate_Charge      =  1 ;
int lastbuttonstate_start       =  0 ;
int lastbuttonstate_stop        =  1 ;

// variables for status of lift and charger

int Lift_Moving                 =  0 ;
int Charging                    =  0 ;

// variables for robot in position char  ( -x- )

int           Robot_In_Pos_Var          =  0 ;
unsigned long Robot_In_Pos_Time         =  0 ;
int           Robot_In_Pos_Interval     =  10000 ;                    // value in Milliseconds

// variables for status of charging and lift.

int Q_Mark_Var= 0;
unsigned long Q_Mark_Time = 0;
int Q_Mark_Interval = 10000; // value in millisec

void setup()
{
                                                                 // initialize serial communication:
   Serial.begin  ( 9600 )          ;
   Serial.println( Start_Message )  ;
                                                                // initialize the pinmodes:
   pinMode( Lift_Up_Sens    ,INPUT)  ;
   pinMode( Lift_Down_Sens  ,INPUT)  ;
   pinMode( Start_Robot     ,INPUT)  ;
   pinMode( Stop_Robot      ,INPUT)  ;
   pinMode( Stop_Charge     ,INPUT)  ;

   pinMode( Lift_Up_Act     ,OUTPUT) ;
   pinMode( Lift_Down_Act   ,OUTPUT) ;
   pinMode( Light_Stop      ,OUTPUT) ;
   pinMode( Relais_Charge   ,OUTPUT) ;
   pinMode( H_Bridge_Enable ,OUTPUT) ;
}
```

```
void loop()
{

  unsigned long Current_Time = millis();

   if (Current_Time - Previous_Time > HB_delay)
   {

    Serial.println(heartbeatmessage);

    Previous_Time = Current_Time;
   }

  // see if data is available in input buffer
   if (Serial.available() > 0)
   {
    int inByte = Serial.read();
       //execute case depending on what char was send
       //via serial communications
       // possible commands
       //*** lift up      : u
       //*** lift down    : d
       //*** start charge : c
       //*** stop charge  : s

    switch (inByte)
    {


    case 'u':                              //case for lift up
      if (Robot_In_Pos_Var==1)
      {
        if (digitalRead(Lift_Up_Sens)!=0)      // NC contact for lift_up_sens
        {
           digitalWrite(H_Bridge_Enable, HIGH);    // enable the H-bridge pin
           digitalWrite(Lift_Down_Act, LOW);       // disable the pin for lift Down
           digitalWrite(Lift_Up_Act, HIGH);        // enable tje pin for lift up
           Lift_Moving = 1;                        // set variable so show the lift is moving
           Serial.println(Lift_Going_Up_Message);  //  send a message back that the lift is going up
        }
        else
        {
          Serial.println(Lift_Already_Up_Message); // error message
        }
      }
      else
```

```
      {
        Serial.println("first send - Robot In Position Char"); // error if robot is not in position
        while(Serial.available()>0) Serial.read();             // clear the serial buffer
      }
    break;


    case 'd':                                       //case for lift down
      if (Robot_In_Pos_Var==1)
      {
        if (digitalRead(Lift_Down_Sens)!=0)
        {
            digitalWrite(H_Bridge_Enable, HIGH);    // enable the H-bridge pin
            digitalWrite(Lift_Up_Act, LOW);         // disable the pin for lift up
            digitalWrite(Lift_Down_Act, HIGH);      // set the pin for lift down
            Lift_Moving = 1;                        // set variable so show the lift is moving
            Serial.println(Lift_Going_Down_Message);  //  send a message back that the lift is going down
        }
        else
        {
        Serial.println(Lift_Already_Down_Message);   // error message
        }
      }
      else
      {
        Serial.println("first send - Robot In Position Char"); // error if robot is not in position
        while(Serial.available()>0) Serial.read();             // clear the serial buffer
      }
    break;


    case 'c':                                     // case for charge
      if (Robot_In_Pos_Var==1)
      {
        if (digitalRead(Stop_Charge)==1)                 // if stop charge button is pressed send error message
        {
        Serial.println(Charging_Stop_Button_Pressed_Message);
        }
        else if(digitalRead(Stop_Charge)==0)
        {
        digitalWrite(Relais_Charge,HIGH);                //set charging pin to high
        Serial.println(Charging_Started_Message);        // send message back that charging has started
        }
        Charging = 1;                                    // set variable that charging started
      }
      else if (Q_Mark_Var ==1)                           // this if loop is used that if a question mark was sended before the c
      {                                                  // the status of the charging relay wil be send back as a message
```

```
    {   |                                          // the status of the charging relay wil be send back as a message
      if (Charging == 1)
      {
        Serial.println(Robot_Is_Charging_Message);          // message that the robot is charging
      }
      else
      {
        Serial.println(Robot_Is_Not_Charging_Message);      // message that the robot is not charging
      }
    }
    else
    {
      Serial.println("first send - control Char");          // error if robot is not in position
      while(Serial.available()>0) Serial.read();            // clear the serial buffer
    }
  break;


  case 's':                                  // case for stop Charging
    if (Robot_In_Pos_Var==1)
    {
        digitalWrite(Relais_Charge,LOW);             // set the charging relay pin to low
        Serial.println(Charging_Stopped_Message);    // send a message back that the robot has stopped charging
        Charging = 0;                                // set charging variable to 0
    }
    else
    {
      Serial.println("first send - Robot In Position Char"); //send the robot in position char first
      while(Serial.available()>0) Serial.read();             // clear the serial buffer
    }
  break;


  case 'x':
    Robot_In_Pos_Var=1;                                     // set the robot in position variable to high
    Robot_In_Pos_Time=millis();                             // set the time for the timer
  break;

  case '?':
    Q_Mark_Var= 1;                                          //set the question mark variable to high
    Serial.println("qmark recieved");                       //send serial message back
    Q_Mark_Time = millis();                                 // set time for timer
    break;
  case 'l':                                                 // case for asking the status of the lift.
    if (Q_Mark_Var == 1)
    {
      if (Lift Moving == 1)
```

```
      {
        Serial.println(Lift_is_Moving_Message);
      }
      else
      {
        Serial.println(Lift_is_Not_Moving_Message);
      }
    }
   break;

  default:                                    // if an unidentified code is sent write error and flush rx buffer
   while(Serial.available()>0) Serial.read();
   Serial.println(Error_Message);
   break;

  }
}
  unsigned long Robot_In_Pos_Ct = millis();                              //if you send the programm the in position Char.
  if ((Robot_In_Pos_Ct - Robot_In_Pos_Time) > Robot_In_Pos_Interval)     // this code resets the Robot_in_position_Bit after a specific time
  {
    Robot_In_Pos_Var=0;
  }

  unsigned long Q_Mark_Ct = millis();                                    // if you send a "questionmark" to the programm
  if ((Q_Mark_Ct - Q_Mark_Time) > Q_Mark_Interval)                       // this code will reset the Q_Mark_Bit after a specific time
  {
    Q_Mark_Var=0;
  }

  int buttonstate_Up =digitalRead(Lift_Up_Sens);          // read top sensor from lift
  delay(20);

  if (buttonstate_Up!=lastbuttonstate_Up)                 //if sensor state changed execute
  {
     if ((buttonstate_Up !=1)&&(Lift_Moving==1))          // if sensor state is not equal to 1 and lift is going up or down execute
     {
        digitalWrite( Lift_Up_Act,LOW);                   // stop lift from going up
        digitalWrite(H_Bridge_Enable, LOW);
        Serial.println(Lift_Up_Message);
        Lift_Moving =0;                                   // set lift_moving_bit to "0"
     }
     lastbuttonstate_Up=buttonstate_Up;                   // put previous state of button in buttonstate
  }


  int buttonstate_Down =digitalRead(Lift_Down_Sens);      // same as top sensor for lift exept this is for the bottom one
```

```
int buttonstate_Down =digitalRead(Lift_Down_Sens);       // same as top sensor for lift exept this is for the bottom one
delay(20);

if (buttonstate_Down!=lastbuttonstate_Down)              // code for stopping the lift when an endswitch is reached
{
   if ((buttonstate_Down !=1)&&(Lift_Moving==1))
   {
      digitalWrite( Lift_Down_Act,LOW);
      digitalWrite(H_Bridge_Enable, LOW);
      Serial.println(Lift_Down_Message);
      Lift_Moving = 0;
   }
lastbuttonstate_Down=buttonstate_Down;
}


int buttonstate_Charge =digitalRead(Stop_Charge);         // same as top sensor for lift exept this is for charging
delay(20);

if (buttonstate_Charge!=lastbuttonstate_Charge)           // code for stopping charging when a button is pressed
{
   if ((buttonstate_Charge ==1)&&(Charging==1))
   {
       digitalWrite( Relais_Charge,LOW);
       Serial.println(Charge_Stopped_By_Button_Message);
   }
lastbuttonstate_Charge=buttonstate_Charge;
}


int buttonstate_stop =digitalRead(Stop_Robot);            // code for stopping the robot manualy
delay(20);

if (buttonstate_stop!=lastbuttonstate_stop)
{
   if (buttonstate_stop ==0)
   {
      digitalWrite( Light_Stop,HIGH);
      Serial.println(Robot_Stopped_Message);
      Serial.println (Stop_Robot_Command);
   }
   lastbuttonstate_stop=buttonstate_stop;
}




int buttonstate_start =digitalRead(Start_Robot);          // code for starting the robot manualy
delay(20);

if (buttonstate_start!=lastbuttonstate_start)
{
   if (buttonstate_start ==1)
   {
   digitalWrite( Light_Stop,LOW);
   Serial.println(Robot_Started_Message);
   Serial.println (Start_Robot_Command);
   }
   lastbuttonstate_start=buttonstate_start;
}
```