Panacea – UUV for Aquaponic systems

Final Report

By Ole Oevermann, Quim Paüls and Valérian Charriere

European **P**roject **S**emester

Autumn 2022

YRKESHÖGSKOLAN NOVIA

Abstract

Aquaponics consists in simultaneously raising fish and growing plants in water, in a way that the plants are fed with the feces of fish. However, sick fishes can spread diseases, so they have to be removed. There needs to be the possibility to remove these fishes remotely, since it may not be possible to do it manually if the aquaponics system is big. The solution is using a remotely controllable unmanned underwater vehicle

The goal of this project is to design and build a prototype of an Unmanned Underwater Vehicle that is able to submerge, move up, move forwards, backwards, sidewards and tilt in order to get a better view and being able to grip the fish. The prototype should be able to submerge 20 meters deep into the water and reach a speed of 1 m/s.

First some research was done. This consisted on searching for what kind of ROV's are already on the market and what their purpose is. Investiagtion on what kind of sensors, cameras, and motors that could be used was also done, as well as investigating which driving technology to use.

Simulatenously, the CAD design of the prototype was done, using the software Solidworks. The design of the body of the submarine was done, although pipes were finally used as the body of the prototype. A part from the body, other components, like attachments for the pipes, gears and racks for the buoyancy control or cases and bases for the propellers were designed and printed.

Finally the programming and wiring in order to control the propellers and the movement of the syringes was done. The software used was Arduino.

Some setbacks were encountetered during the elaboration of this project. They were mainly due to the lack of experience in some of the fields, principally in the electronics and programming. Moreover, some techincal problems when 3D printing, and the slow deliveriy of material ordered online necessary for the prototype are also obstacles that were found.

In general the goals of the project have been met, since a succesful prototype to present to the stakeholders has been built, that can be improved and more ellaborate in upcoming EPS projects.

Acknowledgments

Finishing our EPS project wouldn't have been possible without the help and collaboration of several persons, institutions and involved professors.

First, we would like to thank the Novia University of Applied Sciences for giving us the chance to be a part of the European Project Semester. Being a part of the EPS gave us a chance to acquire new skills and knowledge, as well as discovering a new country and working with people from different countries and environments.

We would also like to thank our home universities, École nationale d'ingénieurs de Tarbes, Universitat de Lleida and Osnabrück University of Applied Sciences for giving us the chance to participate in the EPS.

Next, we thank several people that were involved in our project:

- A special appreciation goes to our supervisor Tobias Ekfors for continuously being interested in the process of the project and for guiding and helping us throughout the semester in overcoming the obstacles and problems that we found.
- Hans Lindén for giving us advice on electronics and for providing us hardware and electronic components that we needed for the prototype.
- Roger Nylund for guiding us and helping us to get to know each other, not only with the team members, but also with the other EPS classmates and professors.
- Osku Hirvonen for giving us a basis on 3D printing and allowing us to use the material anytime we wanted.
- Philip Hollins and Hanna Kuusisto for giving us a good basis and tips on how to properly write and complete the written report.
- Chrysi Dresnali for guiding and helping us, specially at the beginning of our stay, to adapt to the new country and way of life.
- All the EPS and Erasmus friends for making the semester much more fun and enjoyable

Table of Contents

List of abbreviations

ROV/ROUV: remotely operated underwater vehicle

AUV: automated underwater vehicle

RC: remote controlled

PID-Controller: proportional–integral–derivative controller

UBEC: universal battery eliminator unit

ESC: electronic speed controller

EPS: European Project Semester

CAD: computer-aided design

LED: light-emitting diode

HP: horsepower

DC: direct current

PID: proportional–integral–derivative

IDE: integrated development environment

RPM: revolutions per minute

List of Figures

List of Tables

List of Equations

List of Code

1. Overview and Background

In the following section, an overview of the project background and objectives is presented as well as a short introduction of the team members, stakeholders and risks than can be met during the project.

1.1. Novia University of Applied Sciences

Novia University of Applied Sciences is placed along the Swedish-speaking coast of Finland and acts in Vaasa, Turku, Raasepori, and Pietarsaari. With about 320 employees and 4 800 students, the university is the largest Swedish-speaking UAS in Finland. Novia UAS has a close working relationship with the local business community and an active alumni network. (novia.fi, 2022)

1.2. European Project Semester

The European Project Semester (EPS) is a program by nineteen European universities offered to students who have completed at least two years of study. EPS is created with engineering students in mind, but other students who want to participate in an engineering project are also welcome. It is crafted to address the design requirements of the degree and teaches engineering students all the necessary skills they need to succeed in today's working force (European Project Semester, 2022).

Traditionally, there are courses offered that support the teams in their project as well as life in Finland. These courses are Team Building, Project Management, Swedish survival course, academic writing and intercultural English.

Seven projects were offered in Autumn 2022:

- Panacea – UUV for Aquaponic systems
- Sustain4Future
- Affordable Energy for Zambia
- Logistic module
- Module exchanger
- IoT weather station

- Escape Room game

The project team has been assigned to "Panacea – UUV for Aquaponic systems" with the supervisor Tobias Ekfors.

1.3. Project Background

The Project is about developing and building an unmanned underwater drone (UUV) to catch sick fish. The project's name is Panacea. Panacea was the Greek goddess of universal health that used a potion to heal the sick. (Grimal, 2008) For the team to get a general overview, a Hi-level planner for the project was made.

1. What is the (working) TITLE of the project?

Panacea – UUV

2. What is the AIM of the project?

The aim of the project is to catch sick fish out of an aquaponic system or a fish farm with an unmanned underwater vehicle.

3. Project description/background

Short backstory text:

- Identifying the 'problem' or 'need'
- Indicating that this project will supply the 'solution'

The problem is that sick fishes can spread diseases in Aquaponic systems or fish farms. We build a remote-controlled submarine that catches the sick fish and prevents the spread of diseases.

4. What are the PROJECT OBJECTIVES?

Bullet list 3-4 SMART objectives (use active verbs- see list)

To identify the needs to catch sick fish by doing an analysis in the first month.
To evaluate the possibilities to build a UUV until the 14th of October.
To create a 3D model in CAD of the UUV until the 29th of October.

5. What do you envisage to be the possible RISKS to the project?

Identify and bullet list first draft ideas of these hurdles or 'bumps in the road' that may derail the project

- We fail to build the submarine.
- The submarine isn't waterproof.
- Sickness of team members
- Breaking any component
- Failing the programming part

6. Who are the key STAKEHOLDERS in the project?

FYI. Stakeholders are anyone that has some influence on the project
List all those individuals, groups, organisations, companies that you will need to communicate with during the project lifecycle

- Our team
- Tobias Ekfors
- Suppliers
- Teachers
- Novia UAS
- Home Universities

7. What DELIVERABLES are out of SCOPE and not to be included in the project.

When a project is written a clear understand of what the boundaries are needs to be clarified from the start.

For example: a contractor is installing a new bathroom and discovers a rotten floor needs to be removed. Is this IN or OUT of SCOPE of the original contact specification.

List those **DELIVERABLES** out of **SCOPE** and excluded from your project.........

- The final product
- The gripper
- The full programming part

1.4. Aquaponic systems

In an aquaponic system fish cooperate with plants and this consists of the two words aquaculture (the growing of fish in a closed environment) and hydroponics (the growing of plants usually in a soil-less environment) (youmatter, 2020).

The problem with aquaponic systems is that sick fish can disrupt the ecosystem and spread diseases.

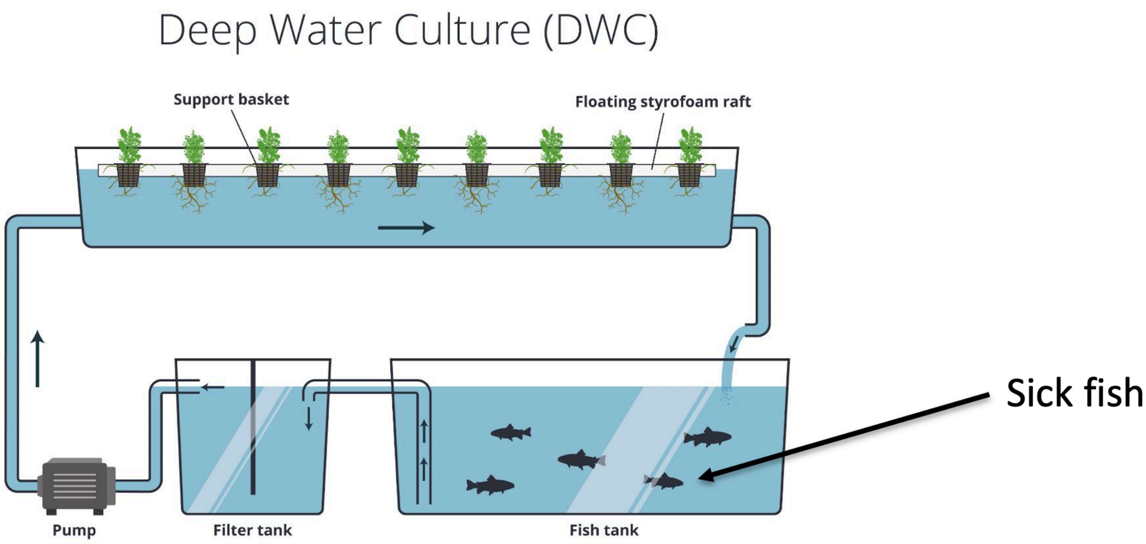


Figure 1: Aquaponic Systems

It is difficult to manually collect sick fish in larger systems and this is the reason why the project was created.

1.5. The Team and the Roles

The team consists of three members, that are introduced below.

Ole Oevermann

Country: Germany

Age: 24

Studies: Automotive engineering

My name is Ole, I am studying mechanical engineering in Osnabrück, Lower Saxony, Germany at the University of Applied Sciences for my bachelor's degree. Osnabrück is in Lower Saxony, Northern Germany. I'm in my fifth semester and chose the EPS at the Novia UAS because I think it will be interesting to work on a project in a multi-cultural team with lots of different cultures. I wanted to do my semester abroad in one of the Nordic countries because I like the landscapes and the calm people.

Quim Paüls

Country: Spain

Age: 22

Studies: Mechanical engineering

My name is Quim, I am a mechanical engineering student at Universitat de Lleida, in Lleida, Catalunya. Lleida is in the northeast of Spain, 150 km away from Barcelona. I choose to do the EPS at NOVIA University of Applied Sciences because I thought it would be very interesting to see a university in a different country and a different way of educating, and because it would be nice to work on a project with people from different nationalities.

Valérian Charriere

Country: France

Age: 21

Studies: General / Mechanical engineering

My name is Valérian, I am an engineering student at the National School of Engineering in Tarbes (ENIT), in the southwest of France. Tarbes is 150 km away from Toulouse. I am in my seventh semester and have chosen to do my semester abroad at the Novia UAS doing the EPS because, for me, it is better to work and learn around a project with a multi-cultural team than just following the courses in my home university. Also, I wanted to go to the Nordic countries because I never have been to this part of the world.

2. Scope Management

This part is based on organizing the different tasks that will have to be completed during the execution of the project, including all the subtasks.

2.1. Project Aims and Objectives

The aim of this project is to design and build a prototype of an ROV. But the prototype needs to fill some objectives for its success. These objectives are to pick up sick fish without killing them, to control the ROV underwater (up and down, forward, and backward, left, and right), and to film/take pictures underwater. In addition, a report needs to be done for two important dates. On the 25th of October, the midterm report, and on the 13th of December, the final report.

2.2. Work Breakdown Structure – Deliverables and Tasks

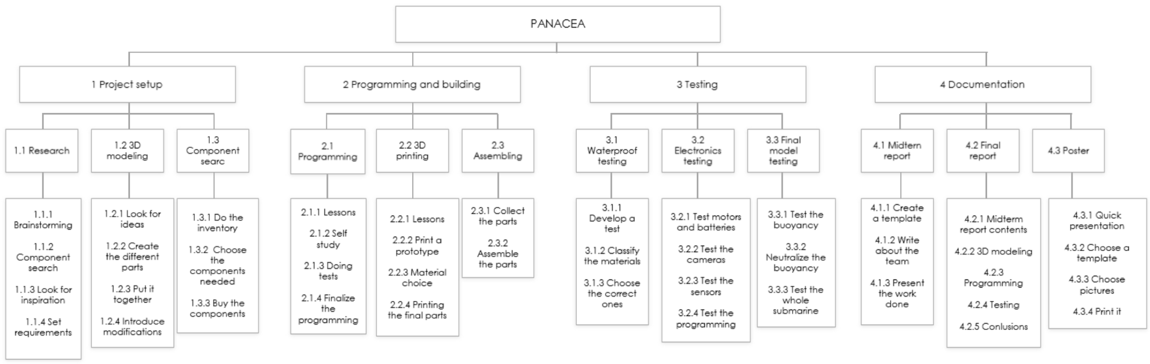


Figure 2: Work Breakdown Structure

Figure 2 shows that the project is divided into 4 parts.

The first part is about all the research and the first ideas of what the group wants for the design and the components needed for that.

The second part is probably the most difficult task: programming and testing. As the name indicates, on this part the group needs to program all the functionalities of the robot and for that, the members of the team need to learn how to program. As no member has a lot of experience in this field, this will probably be the most challenging part of this project.

The third part is about all the tests that need to be done to finish the robot like waterproofing or testing if the components work well. That is a very important part because if the tests fail, the problems detected will have to be fixed, and this could take a lot of time.

The last part is about all the documentation. This has been started since the beginning because there needs to be a follow-up of the project but above all, a midterm report must be written as well as a final report.

3. Time Management, Communication, and Resources

In the next section, the organizing and assigning of the different tasks is shown with the Gantt scheduler. The section also contains an analysis of the people that will intervene in the project with the stakeholder's chart, and finally the way to communicate with all the stakeholders, with the communication register plan.

3.1. Project Scheduler and Responsibility

Gantt Chart was used as a scheduler. This gives a very good overview of the individual tasks from the WBS and visually shows very well how much time you have and how the individual tasks depend on each other. In addition, the people responsible are shown, so that you can contact them directly if something is not going well or there are other problems.

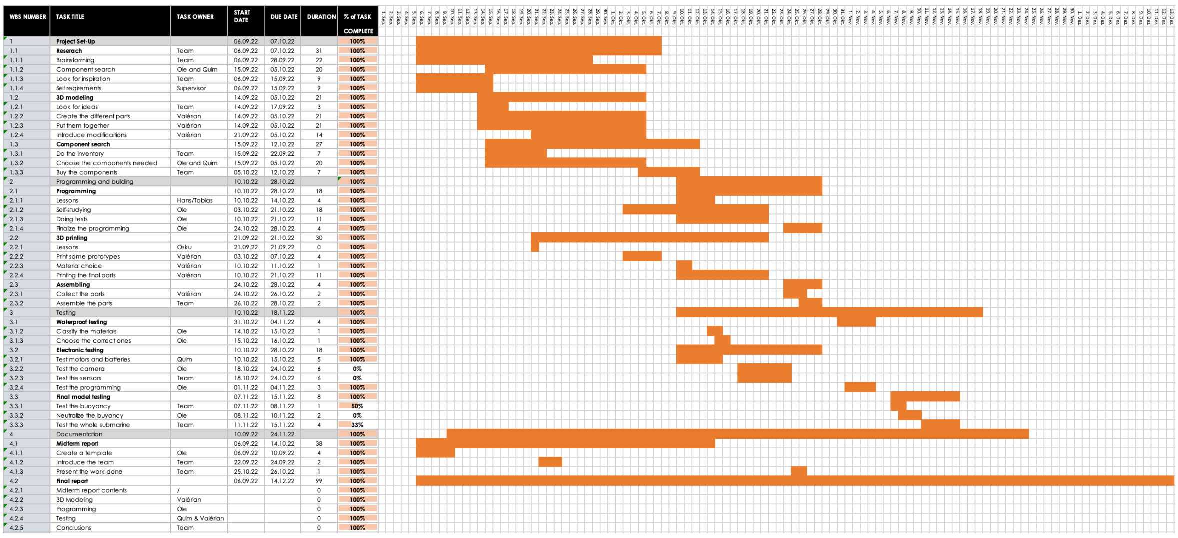


Figure 3. Gantt Scheduler

A full-sized Version can be found in Appendix 2.

3.2. Stakeholder Identification and Analysis

A Stakeholder Analysis is made to identify the parties that are directly and indirectly involved in the project. A detailed stakeholder analysis was made and presented visually.

No.	Stakeholder	Extent of interests	Degree of influence	Interests, expectations	Size of group	Chances (+) and dangers (-)	Possible measures	Status
s1	Quim Paüls	10	10	get the bachelor thesis done, learn new skills		+Having interest in learning new skills, and desgning and building a proper prototype. --Not having enough skills/knowledge and not having enough time.	- Having lessons - Self learning - Dedication	
s2	Ole Oevermann	10	10	learn new skills, work in a cool team		+Having interest in learning new skills, and desgning and building a proper prototype. --Not having enough skills/knowledge and not having enough time.	- Having lessons - Self learning - Dedication	
s3	Valérian Charriere	10	10	meet new coultures, leran new skills		+Having interest in learning new skills, and desgning and building a proper prototype. --Not having enough skills/knowledge and not having enough time.	- Having lessons - Self learning - Dedication	
s4	Tobias Ekfors	10	8	get the project done, be part of a nice team		+Wanting to help and teach his students. -Not having enough time to dedicate to the project.	-Organizing the tasks properly	
s5	Roger Nylund	6	2	help the team, make the teams enjoy the projects		+Wanting the group to work together succesfuklly and with a good team spirit.		
s6	Hans	1	8	help us in programming and hardware decisions		+Teaching us electronics skills. -Not having enough sessions	-Trying to find tme to teach us lessons	
s7	Novia UAS	6	3	get their projects completed,		+Wanting their projects to be completed, giving the proper resources.		
s8	Home Universitys	5	3	Proof they have given good education		+Hoping theur students give a good image of the univeristy		
s9	Material suplliers	1	7	to supply good materials and stuff		+Supplying the necessary materials for the project. -Being to expensive and not providing on time	-For us, trying to fins good but cheap enough material	
s10	Friends	1	3	No interest		+Helping us with different skills. -Distracting us from work	-For us, trying not to get too distracted from work..	
s11	Other teachers	3	4	to teach helpful things		-Taking us time with their subjects	-Not giving too much work after class	

Figure 4: Stakeholder Chart

As seen, there are many different stakeholders, with different levels of influence.

The diagram shows which stakeholders need more attention and which stakeholders only need to be informed frequently.

3.3. Communication Register/Plan

Stakeholder Communication Plan			
Stakeholder	Communication type	Frequency	Contact
Tobias Ekfors	MS Teams	weekly	tobias.ekfors@novia.fi
Ole Oevermann	MS Teams Whatsapp	daily	ole.oevermann@edu.novia.fi +49 152 0881 3872
Quim Paüls	MS Teams Whatsapp	daily	quim.pauls@edu.novia.fi +34 618 699 332
Valérian Charriere	MS Teams Whatsapp	daily	valerian.charriere@edu.novia.fi +33 6 19 73 26 60
Roger Nylund	E-Mail	low	roger.nylund@novia.fi
Novia UAS	E-Mail	low	studentservices@novia.fi
Home Universitys	E-Mail	low	m.kiebert@hs-osnabrueck.de ri@udl.cat eu-outgoing@enit.fr

Table 1: Communication Chart

The stakeholder communication plan gives a quick and easy overview of all parties involved in the project and outlines the communication possibilities. In addition, the communication frequency is indicated.

4. Risk Management

The following part consists in identifying the possible risks that may prevent the project from succeeding, looking into the possible parties, accidents, events or situations that could make the project fail and also evaluating their probability of becoming a problem and their importance if they did.

4.1. Risk Identification and Analysis

This graph shows the people or organizations who will intervene with importance on our project. It demonstrates that the three members of the team will manage the project closely (on the top right). On the opposite, on the bottom left, there are the stakeholders who will just monitor the project and won't have any influence on it.

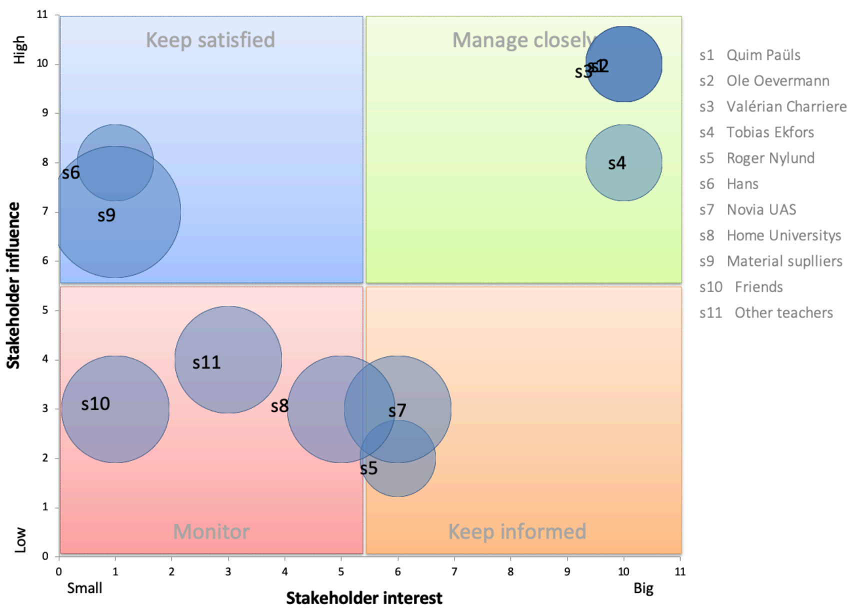


Figure 5: Stakeholder Risk analysis

4.2. Risk Register and Mitigation/Contingency

RISK DESCRIPTION	IMPACT DESCRIPTION	IMPACT LEVEL	PROBABILITY LEVEL	PRIORITY LEVEL	MITIGATION NOTES	OWNER
Give a brief summary of the risk.	What will happen if the risk is not mitigated or eliminated?	Rate 1 (LOW) to 5 (HIGH)	Rate 1 (LOW) to 5 (HIGH)	(IMPACT X PROBABILITY) Address the highest first.	What can be done to lower or eliminate the impact or probability?	Who's responsible?
The buoyancy of the submarine isn't neutral	we have to in/decrease the weight	3	2	6	Doing the calculations properly	Team members
Some components aren't waterproof	They will brake	5	1	5	Making sure we choose and buy the right components	Quim
Technical problems	Submarine doesen't work	5	2	10	Designing well, taking good lessons and self-learning	Team members
Not having enough time	Protoype isn't finished on time	4	2	8	Putting enough dedication to the project.	Team members
Not having the necessary material	Not being able to build the prototype	5	1	5	Buying things on time and to reliable suppliers.	Valérian
Not having enough knowledge of electronics/programming	The design is not good	3	3	9	Self leraning, having good lessons, recieving help	Valérian
Some components (ex: propellers) break when used	Submarine doesen't work	5	1	5	Choosing good material and good suppliers	Quim
Submarine doesen't catch fish	Not completing its purpose	3	3	9	Having a good design and gripper	Ole
Cameras aren't good enough	Not having a good vision	3	2	6	Buying accurate cameras	Ole
Losing the signal when we put the submarine in to the water	We loose control of it	5	1	5	Taking good care of the electronic connections	Team members

KEY
LEVEL
1
2
3
4
5

Figure 6: Risk Register

The excel table shown in figure 6 shows that the project could fail in many ways, as there are many potential risks. It is also necessary to focus on the level of importance and probability combined of the risks, shown in the numbers and colours. As seen, there are no very high risks in red, but there are a lot of many medium risks that will have to be taken care of.

5. Research

The next part contains the research needed for having a good basis when starting to design and build the ROV. It contains some information of the different categories of UUV, and how this divide into subcategories, with some characteristics and functions of each. It also consists on research on which driving technologies are available as well as the decision of which one will be used. Moreover, there is some research on what type of sensors, motors, cameras, sealings and lightning will be used. Finally, a schematic of the electronic system that will be used is shown.

5.1. Introduction to Unmanned Underwater Vehicles (UUV)

5.1.1. ROV and AUV

UUVs are divided into two categories:

- Remotely operated underwater vehicles (ROUV or ROV): ROVs are used for underwater tasks to replace divers in difficult water conditions. (He, Wang, & Ali, 2020)
- Autonomous underwater vehicles (AUV): It is completely autonomous and does not need any action from the surface (Fossen, 2022)

Only the ROUVs are of interest to the project.

5.1.2. Remotely Operated Underwater Vehicle

In recent years, the focus of researchers has increasingly been on UUVs to explore the underwater world. Especially ROVs are very well suited for such applications. There are many industrial applications such as oil reservoir discovery, pipeline maintenance, but also fish farm monitoring (He, Wang, & Ali, 2020). Especially with fish farms, it is difficult to get out sick or dead fish as a diver (Doornekamp, 2021). There, ROVs are very well-suited because they are small and manoeuvrable.

5.1.3. What the market has to offer.

There are several ROV categories on the market, and the classification is usually based on their size.

- Micro: They weigh less than 3 kg and there are used as an alternative to a diver, for entering places in which a diver could never, like a small pipe.

- Mini: Weighs until 15kg, and it is used also for substituting a diver, but for completing tasks that require more strength or power.

- General: The power is less than 5 HP, usually carrying a sonar unit, used for light survey applications, with a maximum working depth of 1000 m.

- Inspection class: Used for industrial or commercial use, with the purpose of data gathering and observation. Typically equipped with live-feed video, still photography, sonar, and other data collection sensors.

- Light work class: Usually less than 50 HP, they are often able to carry some manipulators and the maximum working depth is less than 2000 m.

- Heavy work class: Less than 220 Hp, they can carry two manipulators and have a maximum working depth of 3500m.

- Trenching & Burial, with a range of 200 to 500 HP, they can have a maximum working depth of 6000 m.

(Committee, 2016)

For the project, the type of ROV that will be used is the mini.

5.2. Requirements

The final product will have to meet with the following requirements:

- Diving a minimum of 20 meters deep

- Reach up to 1 $m*s^{-1}$ forwards
- Control of the ROV with a ground module via cable connection over the water
- Camera attachment and video transfer to the ground module
- Gripper mount and gripper
- 3D printing as much as possible

5.3. The decision of the driving technology

5.3.1. Driving technology

To go forward, backward, or to the side propellers are needed. To dive deeper screw propeller thrusters or automatic buoyancy control can be used. Before almost all submarines used screws, but engineers are looking for alternatives nowadays because they aren't very effective and are prohibited in some areas. Propellers are easy to control and use, and they don't need any PID controller. However, they make a lot of noise, consume a lot of energy, and it is difficult to set a maximum depth without sensors. (Unnamed Underwater Vehicles, 2021)

Figure 7: Screw Propeller **(Vevor.ca, 2022)**

A possible alternative is automatic buoyancy feedback control. This type of control uses a syringe or a special piston or cylinder with an attached servomotor. With buoyancy control, it is easy to set a maximum depth, the depth is controlled with sensors by a PID, and consumes less energy. The PID control, however, must be very good otherwise this kind of technology is useless. (Salomaa, 2017)

For the project, a combination of both propellers and buoyancy will be used, because buoyancy control would move the ROV up or down too slowly.

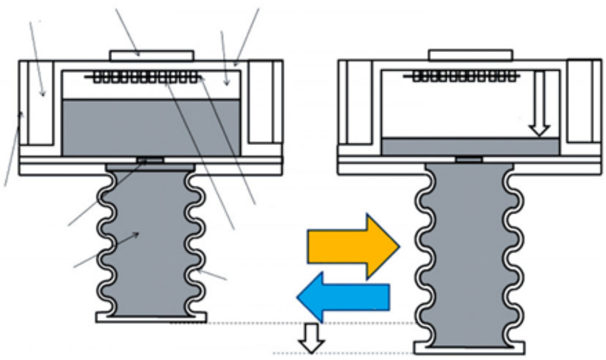


Figure 8: Buoyancy demonstration **(researchgate, 2013)**

5.4. Buoyancy Control

Buoyancy control is needed to keep the buoyancy neutral even with a load of e.g., a fish or other things and, as mentioned, to go up and down. To guarantee neutral buoyancy, the volume of the UUV must be calculated, and from this the buoyancy force. Then the UUV must be loaded with a mass so that the following equation is true:

$$\boldsymbol{F_{buoyancy} = F_{gravity}}$$

Equation 1: Neutral Buoyancy

A rough calculation with a simplification of the cylinders used in the prototype was calculated. The calculation can be found in appendix number 6.

With the rough calculation, we get a necessary mass of about 13 kilograms.

Another possibility is an experimental approach, with which one can make the fine-tuning after a rough calculation. By pulling the finished UUV underwater with a newton meter and loading the UUV with lead according to the measured force. Since F = m * a applies, a mass of m = F / a is needed.

Since components such as motors, side plates, and other attachments are not considered in the rough calculation, the ROV will be loaded experimentally after the prototype is built until neutral buoyancy is assured.

5.5. Sensors

5.5.1. Distance sensors

Distance sensors will have to be used to be aware of the distance between the ROV and the bottom of the sea. The camera that will be used will show a good perspective, but not good enough to have eyes on the bottom of the sea.

The distance sensor that will be used is a waterproof ultrasonic sensor, the DYP-A02YY PMW.

Figure 9: Distance Sensor **(Amazon, 2022)**

5.5.2. Pressure Sensors

A pressure sensor will also be used, to keep control of the depth. With the sensor, information on the pressure level at every moment is given when generating an electrical sign in proportion to the water pressure they measure. Knowing that pressure underwater increases 1 atmosphere for every 10 meters of depth, it will be possible to always know the depth of the ROV. (Lapresa, 2011)

The chosen pressure sensor is from the manufacturer RS PRO and can read from 0 to 6 bar of pressure.

Figure 10: Pressure sensor **(RS components, 2022)**

5.5.3. Gyroscope

Another sensor used will be a gyroscope. This one will be able to do the functions of giving information on the acceleration of the ROV and its inclination. The sensor used for this is a GY-86 MPU-6050 Accelerometer.

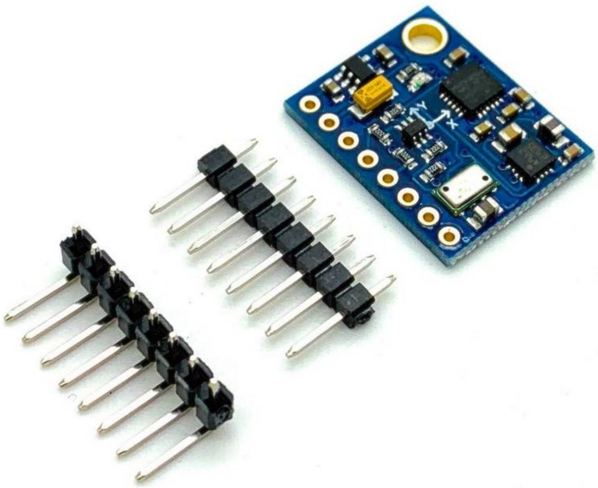


Figure 11: Accelerometer **(Berrybase, 2022)**

5.6. Cameras

There are many options available in the market that would suit the purpose of the ROV, some of these are:

- **GoPro Hero9 Black**
  - Lots of mounting options
  - Excellent video quality (5K)
  - Water resistant to 60 m (IP8X)
  - Ultra-Wide View
  - Video transfer difficult
  - Cheapest in the comparison (380 €)

Figure 12: GoPro Hero9 black **(Gigganti, 2022)**

- **GoPro Hero10 black**
  - Many attachments possibilities
  - Excellent video quality (5,3K60FPS)
  - Water resistant to 60 m
  - Ultra-Wide View
  - Video transfer difficult
  - Expensive (500 €)

Figure 13: GoPro Hero10 black **(Kameraliike, 2022)**

- **Insta360 ONE RS**
    - Many attachments possibilities
    - Excellent video quality (6K)
    - Expandable to 360° Camera
    - Water resistant to 60 m (IP8X)
    - Ultra-Wide View
    - Video transfer difficult
    - Expensive (450 €)

Figure 14: Insta360 **(Fotonordic, 2022)**

As seen, these are very good quality cameras, which would for sure give a very good view of under the sea, but they may not be very easy to place, and they are a bit expensive.

There are different options in the market, and one that appears to be a very good choice is a Raspberry Pi camera. In this case, the price has a range of 15-70€, and it can be a fisheye camera to get a view from a very wide angle.

The camera chosen is the Raspberry Pi Camera V2.1.

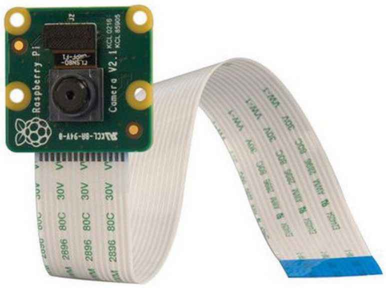


Figure 15: Raspberry Pi camera **(Dustinhome, 2022)**

5.7. Remote control technology

For the remote control of the ROV, there are 2 options. The first one is using a wireless connection, and the second one is using a wire.

The option of using a wireless connection appears to be much more comfortable, as there is no need for a physical wire that has a maximum length. Water shields very well, so that a Bluetooth or Wi-Fi connection has a range of a few centimeters. It would be possible to do it with Sonar or light. However, it is very difficult to establish this kind of connection underwater, so this is not the option used. (Mohammad Harun-Or-Rashid, 2019)

Using a cable, limits the range of movement of the ROV, but it is not expensive, there will be no interferences with the connection, and a basic cable will work well. However, an IPX8-certified connector will be needed (Source IEx., 2022).

5.8. Motors

There are several waterproof motors on the market, and there are two basic types of them:

- **With sensor:** They have a great acceleration right down from zero RPM, but are susceptible to being contaminated when used in muddy water
- **Sensorless:** They are quite the opposite of the motors with sensors, as they work well in polluted water, but strutter at low RPM. (Ryan, 2019)

Since the ROV will operate in an environment with sick fishes and nature, that can tend to be quite muddy, and it doesn't need to have such a strong acceleration, the chosen motor will be a sensorless brushless motor. For the first version of the prototype a brushed waterproof DC motor was chosen.

5.9. Seals

To make sure the ROV is waterproof, and the electronic components don't get wasted, all the parts that compose its structure will have to be properly sealed. There are 2 possibilities for this: liquid sealing and universal gaskets.

- **Liquid sealing**

    - Cheap
    - Very easy application
    - Difficult when the parts must be disassembled
    - Easy to seal joints that are not perfectly flat
    - Gluing gives more stability

Figure 16: Liquid sealant **(Puuilo, 2022)**

- **Universal gaskets**

    - Cheap
    - Difficult to cut the right size of the gasket
    - Easy to disassemble and assemble
    - Joints must be perfectly flat

Figure 17: Universal Gaskets **(Swift supplies, 2022)**

Again, a combination of both elements will be used for sealing the different parts of the ROV. Since using a liquid sealant is a very good idea but then it would be difficult or impossible to access the interior of the ROV if some electronics/cameras must be modified, gaskets will be used in at least two sealings of the main pipes, so it is possible to open them and access the inside, and liquid will be used for the rest of sealings.

5.10. LEDs

LEDs will be placed in the two smaller cylinders on each side of the ROV. They don't need to be waterproof if they're placed inside a spheric plexiglass. Another option is simply to get waterproof LEDs to be placed on the outside of the ROV.

Figure 18: SMD LED 5050 **(Valokas, 2022)**

5.11. Electronic Schematics

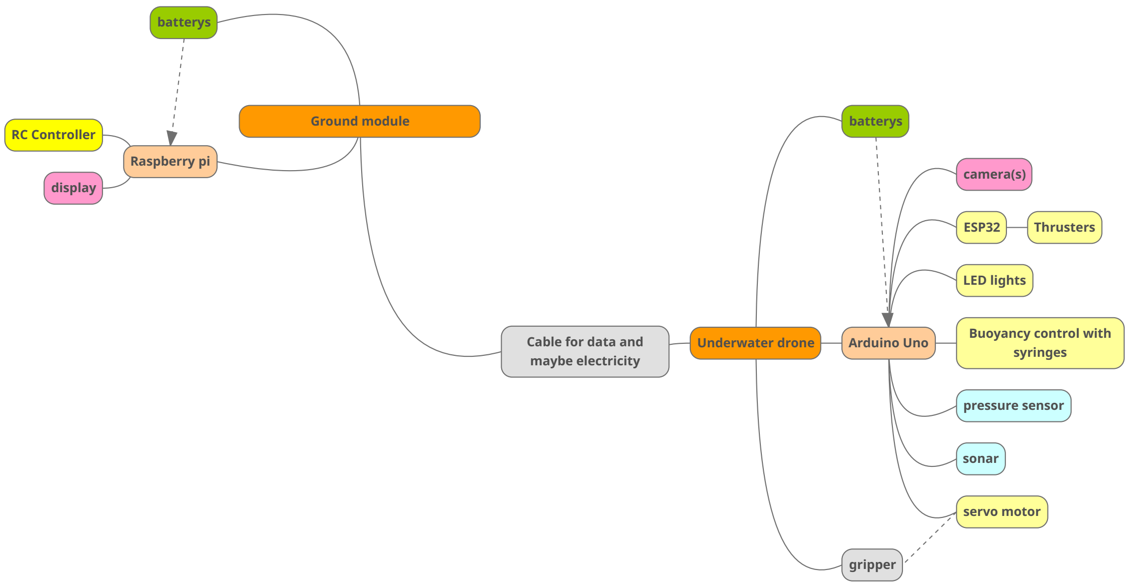


Figure 19: Electronic schematics

To control the ROV and thus all electrical components the platform Arduino and Raspberry Pi were chosen. An Arduino is a physical computing platform that consists of software and hardware (Arduino.cc, 2018). An Arduino contains a microcontroller and several inputs and outputs. The microcontroller is not used for the camera and the video stream to the ground module, because the Arduino has no video output. A single board computer is used for this. Since Raspberry Pi is very popular here, it was chosen. The Raspberry Pi has different inputs and outputs and works like a standalone computer.

In a second prototype or the final product, another system could be chosen. The ArduSub platform is specially developed for ROVs and has several functions. It consists of a Pixhawk and a Raspberry Pi as a so-called companion in the ROV and a laptop as a ground module to control the ROV (ArduSub, 2022).

Since the first prototype will not be finished, another electronic schema is chosen as seen in figure 20.

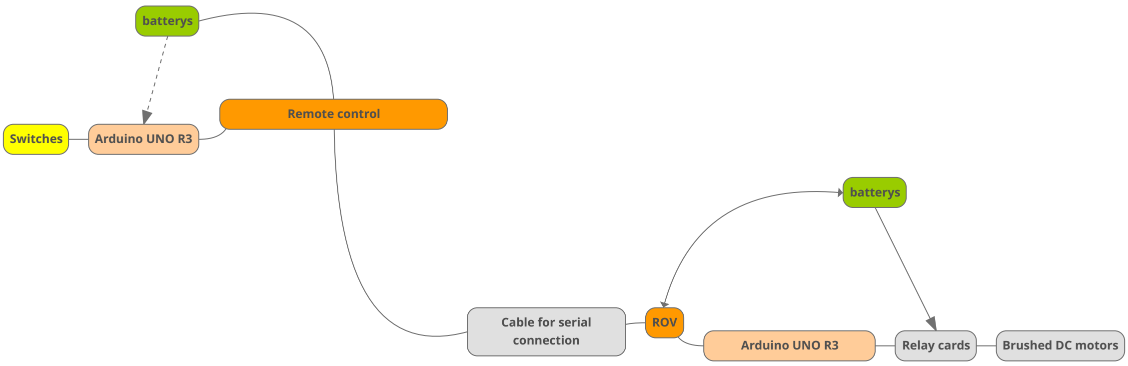


Figure 20: Used electronic schematic

6. Ballasts system

The system for controlling the buoyancy of the submarine will consist of two plastic syringes, placed on each side of the submarine, inside the smaller tubes. Both syringes have a capacity of 200 ml. A flexible plastic tube connects the end of the syringes to the exterior of the submarine, ending in the water.

For each syringe, a gear rack is 3D printed, which has a tooth in both sides. This rack is attached to the barrel of the syringe, and this barrel is attached to the pipe, establishing the barrel as the fixed part of the system. The servo motor is fixed at the flat end of the syringe plunger, and its rotation is transferred to both sides of the rack with a basic system of gears. As a result, when the motor turns, its rotational movement is transferred into linear movement in the racks, and the plunger of the syringe moves forwards and backwards as needed.

When the syringes are filled with water, the buoyancy of the submarine decreases, helping it to move down, and when this water is propelled to the outside, the buoyancy will increase, creating a force that will help the submarine move up.

7. Final model's CAD design

The following section shows the CAD design of the final model. However, this model wasn't the one used for the prototype, since the structure of it was complex to print with the available printers and there were some issues, so it was decided to save the design for upcoming projects and use a simpler structure for the first prototype with pipes.

7.1. Global view of 3D model

For starting to the present the 3D model view of the ROV, a full view of it is shown to have an idea of its look.

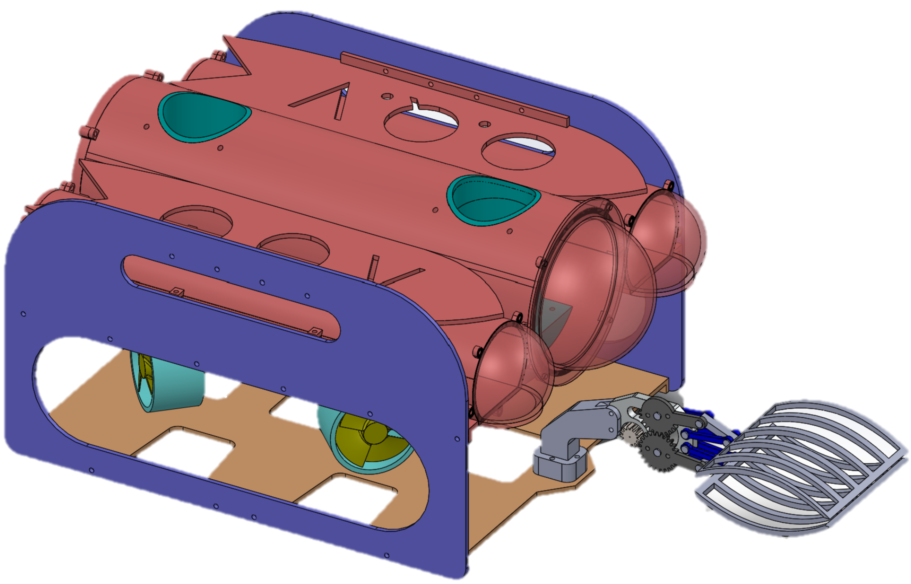


Figure 21: Isometric View of the ROV

As seen in the picture, the ROV doesn't have the shape of an RC submarine. It doesn't need to have a form of RC submarine but a form of discovery and research ROV. So, for this, it needs to have protection for all the major components except for the gripper.

7.2. Various views of the ROV

The submarine will measure approximately 500 mm in length for 400 mm of wide for 300 mm in high without the gripper, with the gripper the submarine's length increases to 800 mm.

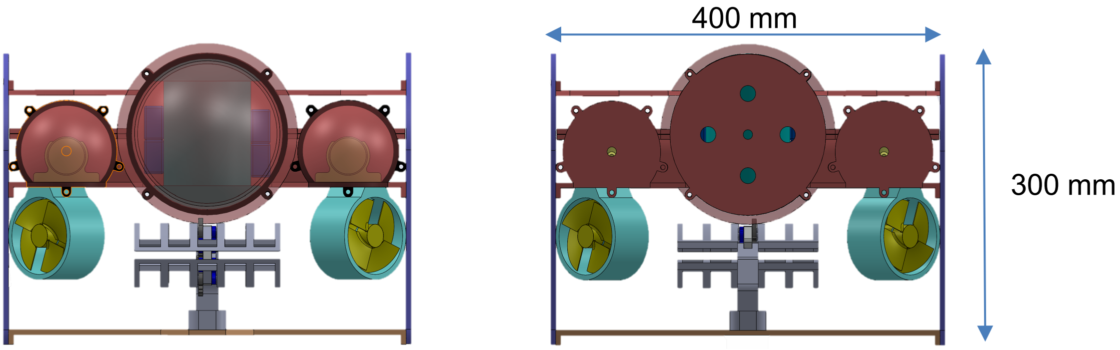


Figure 22: Front and back view of the ROV

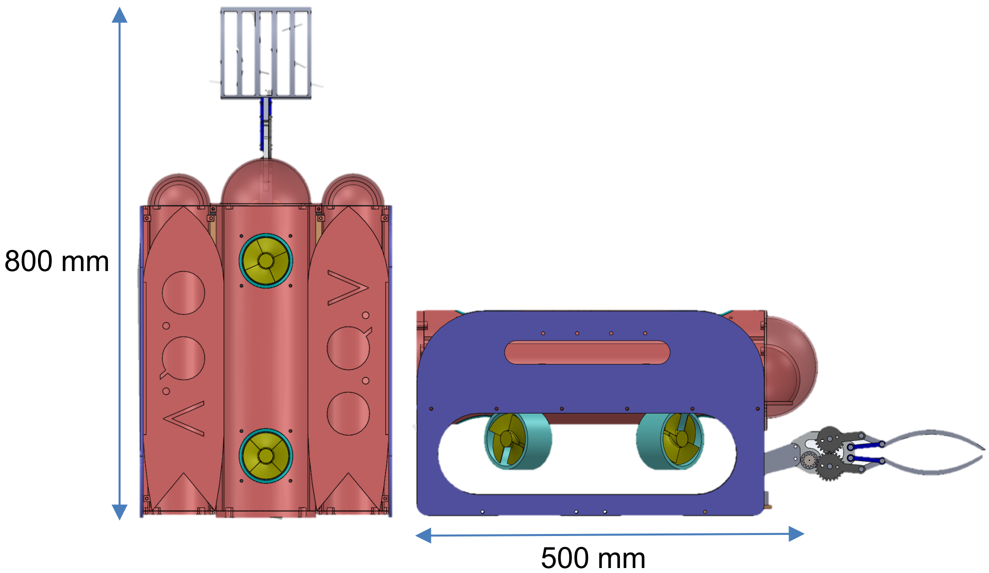


Figure 23: Top and side view of the ROV

As seen in the figures, the submarine has a total of 6 propellers. The ones located inside the inner tube, are for making the ROV go up and down, and are placed inside a case that allows them to have more contact with water and reduces the resistance to it by the structure of the submarine. The other 4 propellers make the submarine move in any direction within the horizontal plane, and the shape of the structure surrounding them is also made in order to reduce the resistance to water and help the propellers to pump water as needed.

7.3. Exploded view of the ROV

Understanding the way, the different parts of the ROV are put is a bit confusing, but this "exploded" view makes it possible to visualize, before seeing each important part one by one, all the parts that are assembled in the 3D model.

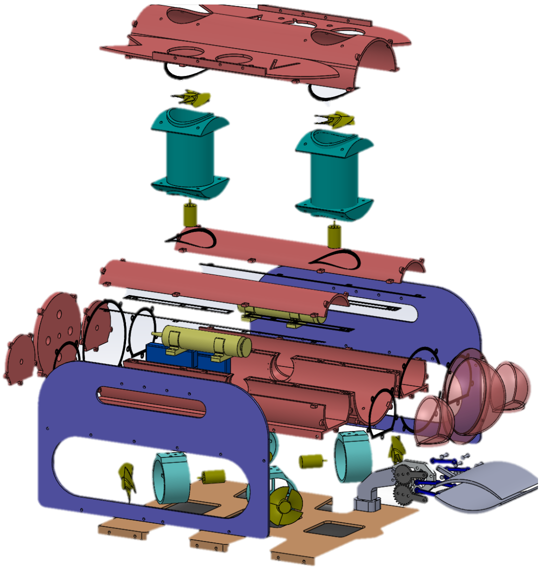


Figure 24: Exploded view of the ROV

In figure 23, some details that were unspotted in the previous pictures can be seen. There is a clear view of the cases that contain the inner propellers, and the ballast system used for the buoyancy control placed inside the smaller side tubes can also be appreciated.

7.4. The main components

In this section, pictures of the main components that shape the submarine are show. This gives a more detailed view of how the ROV is formed, and the function and composition of each of the parts.

7.4.1. The central component

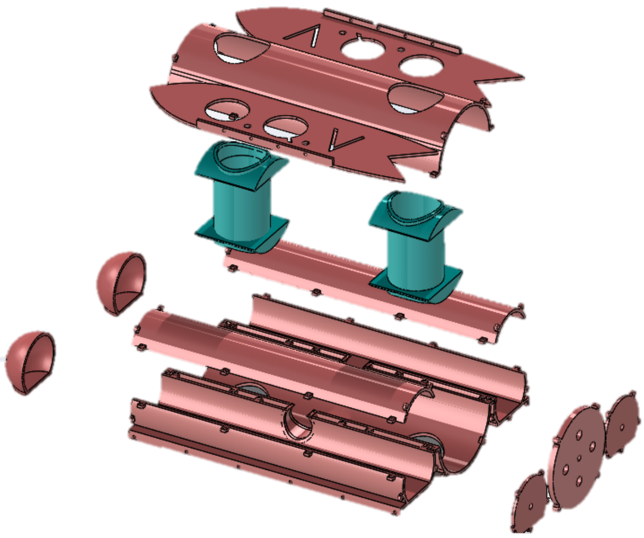


Figure 25: Exploded view of the central component

This is the central component with an exploded view that allows to see all the parts needed for just 1 component. A full-sized view can be found in the appendix 3.

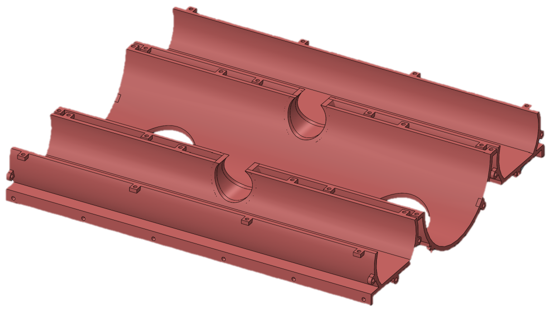


Figure 26: Lower part of the 3 tubes

Here is the most important part. There are 3" half-tube" which will be used to store all the electronic components for the main tube and to store both ballasts (syringes with their mechanism). This part will be storing all the electronic components, that why it will have to be totally waterproof.

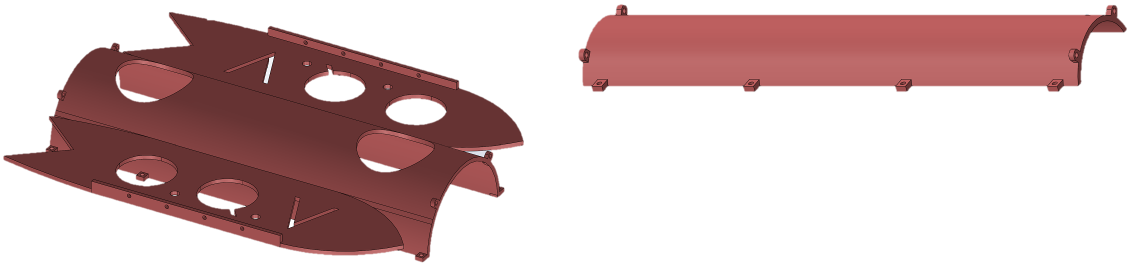


Figure 27: Top part of the central cylinder and side cylinders

Then, for completing the 3 half-tubes, it is necessary to add these parts. On the first picture, there is the ROV’s name inlaid. There is nothing special about these parts except that the first part will help to rigidify the ROV.

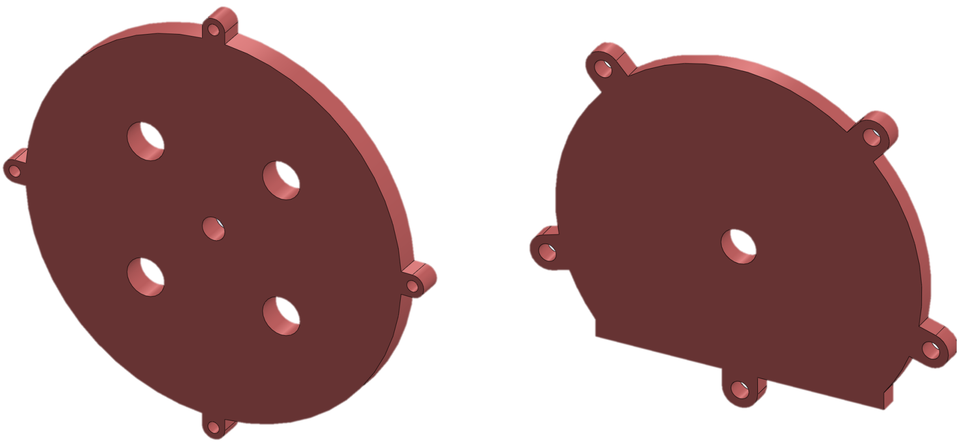


Figure 28: End pieces of the back and front

For closing the backside of the central component, a simple closure is added. But the second piece has a hole in the centre, made for the pipes of the ballasts which pump water outside for reducing the weight of the ROV. The first piece has many holes for all the cables that connect the outside to the inside, for the remote controller, batteries, sensors.

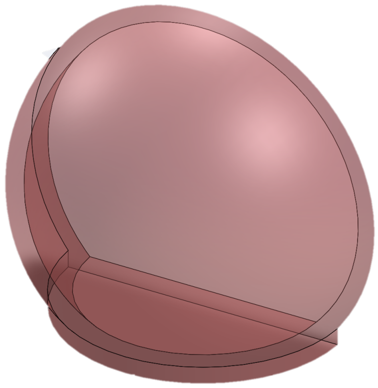


Figure 29: Spherical closing of the front

For closing the frontside of the central component and more specifically the two little tubes, it's necessary to print a piece with spherical form for better penetration in the water. The front closure of the large tube is explained later.

Figure 30: Top view of the inside tubes

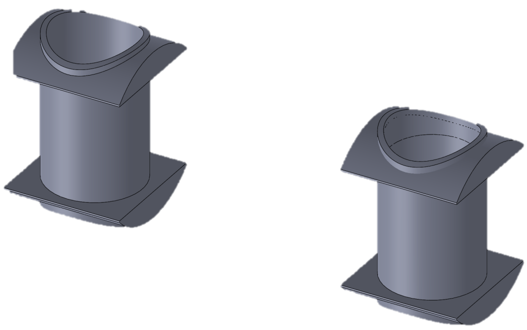


Figure 31: View of the inside tubes

To finish with the parts which compose the principal component of the ROV, it is necessary to have 2 tubes inside the big tube for keeping the two motors with their propellers, that allow the submarine to go up and down with good stability.

7.4.2. The side components

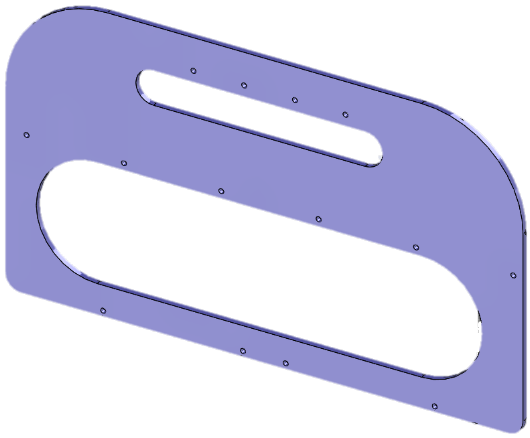


Figure 32: Side plate

The main characteristic of the shape of this plate is the two holes that it has. For the one on top, the smallest, the shape allows to grab the submarine with the hands. The second one is bigger because the four motors and the propeller are at its level, and they need clear access to water to propel and therefore need space for evacuating the water.

7.4.3. The floor inferior

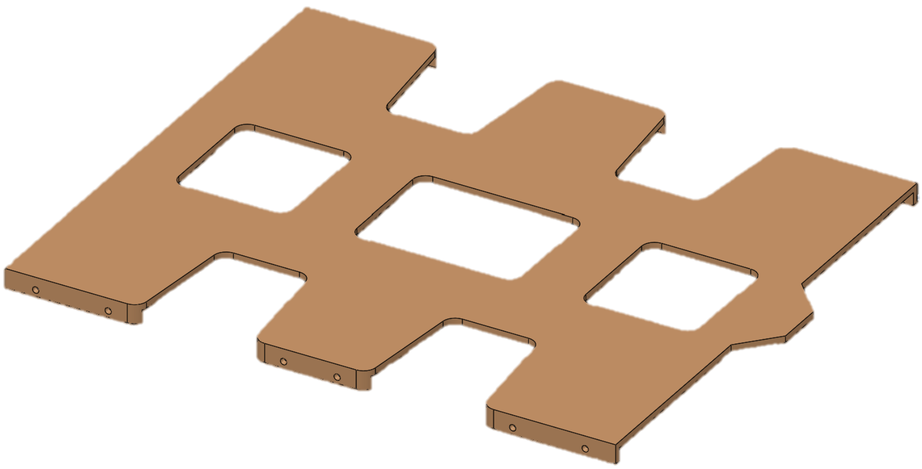


Figure 33: Inferior plate

The floor inferior has three utilities. The first one is to rigidify the ROV from the bottom, the second one is to protect the four motors and the third one is to receive and to support the gripper. But with these three purposes, it must have the least surface in contact with water to have the least resistance when it goes down in water. That's why it has this shape with these holes.

7.4.4. The motor's protection with propeller

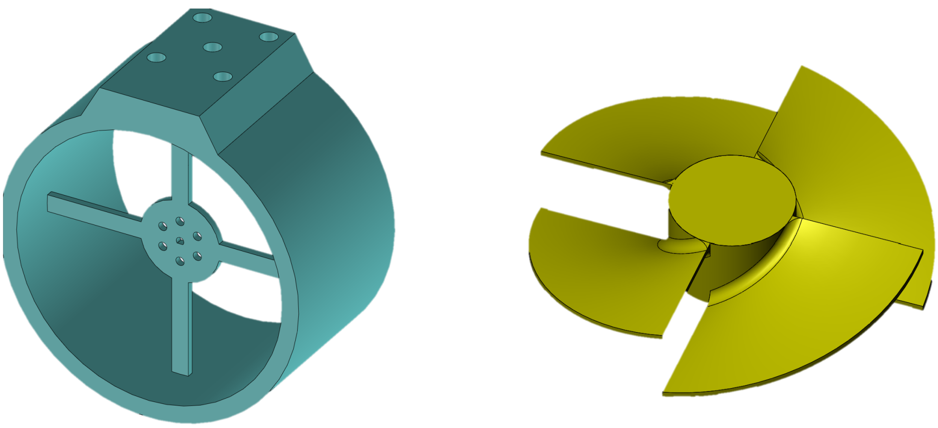


Figure 34: Protection and propeller

The first picture is a 3D model for protecting the motor and the propeller. It enables to fix both sides of the complete engine (motor + propeller).

The second picture is a propeller.

7.4.5. The gripper

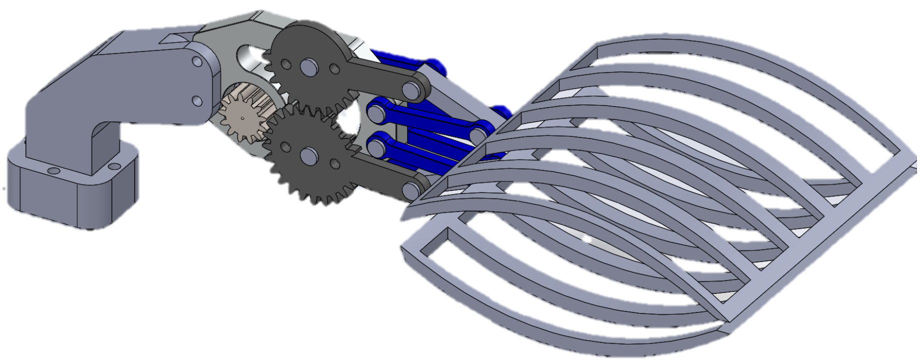


Figure 35: Gripper

The gripper is the second most important component of the ROV. It allows to catch the fishes. For its composition, there is the piece which fixes the gripper to the ROV, the main body and the gripper. To control the mechanism, its necessary to print three gears to open and close the gripper. The first gear (left) is driven by a servo motor. The two parts of the gripper are perforated to get the least opposition against the water force but with enough material for not letting the fish escape. A exploded view of the gripper can be found in appendix number 4.

7.4.6. The others

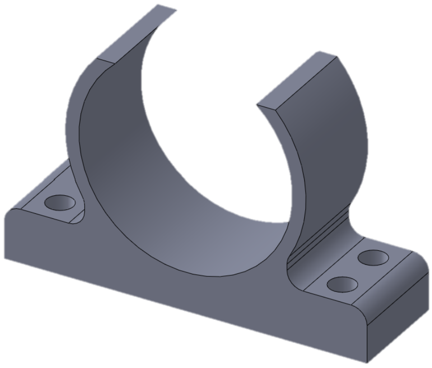


Figure 36: Syringe fixer

This part will receive the ballasts, that will basically be syringes, so they must be fixed on the structure. It takes two parts like this to correctly fix the ballasts.

8. First prototype’s 3D design

The next section shows the CAD design of the structure used for the first prototype. Pipes were used for this model since it was an easier and faster solution. However, some parts were also designed with CAD and 3D printed, like attachments, cases, bases…

8.1. Global view of 3D model

The following picture is an overview of the full shape of the submarine, which gives an idea of how the first model looks like.

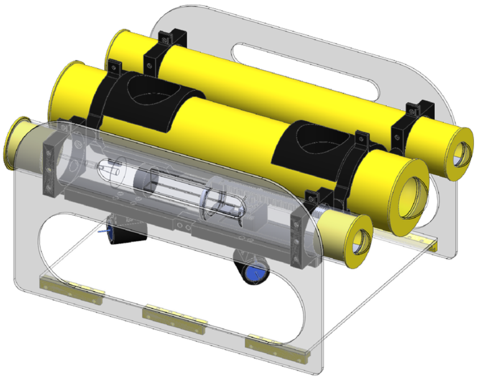


Figure 37: Global 3D view of the first prototype

As seen in the picture, the main components of the structure for the first prototype, are the 3 pipes. The two smaller ones on each side contain the ballasts, and the central pipe contains the electronics. The 3 of them have a spheric plexiglass in the front. In black, the pipe attachments are seen, as well as the bases and cases for the propellers, both the inner propellers, for going up and down, and the propellers on the bottom, for moving forwards, backwards and sidewards.

8.2. Various views of the ROV

This model is also formed by 6 propellers. 2 of them go across the central tube and are used for making the submarine go up and down. The other 4 are placed on the bottom of the submarine, in a way that they allow it to move in any direction. It is also appreciated how the 3 pipes are fixed together with the black 3D printed attachments. Moreover, the cases that contain and fix the propellers are also seen.

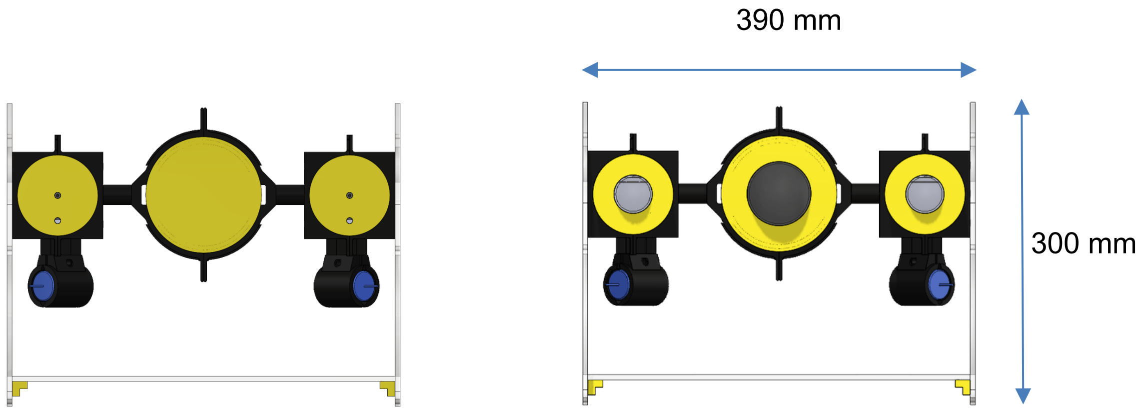


Figure 38: Back and front view of the submarine

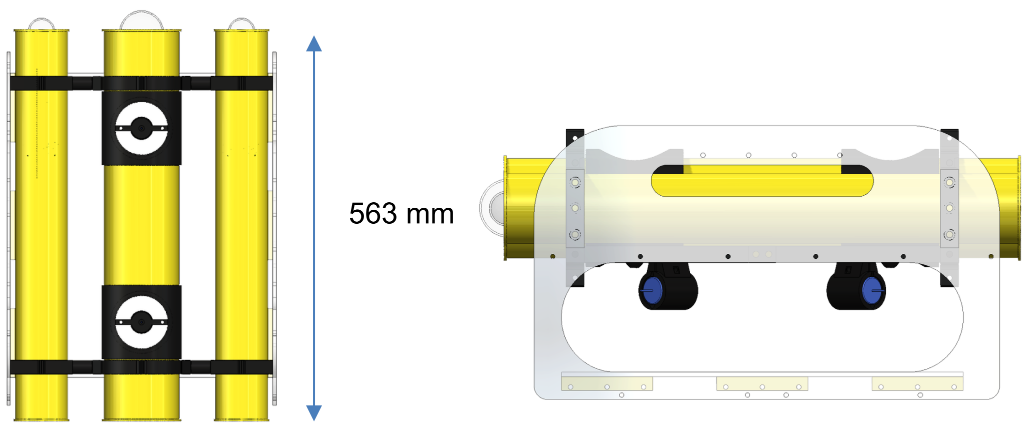


Figure 39: Top and sideview of the submarine

The submarine has a length of 563 mm a width of 390 mm and a height of 300 mm.

8.3. Exploded view

The next view helps to see all the items that form the submarine.

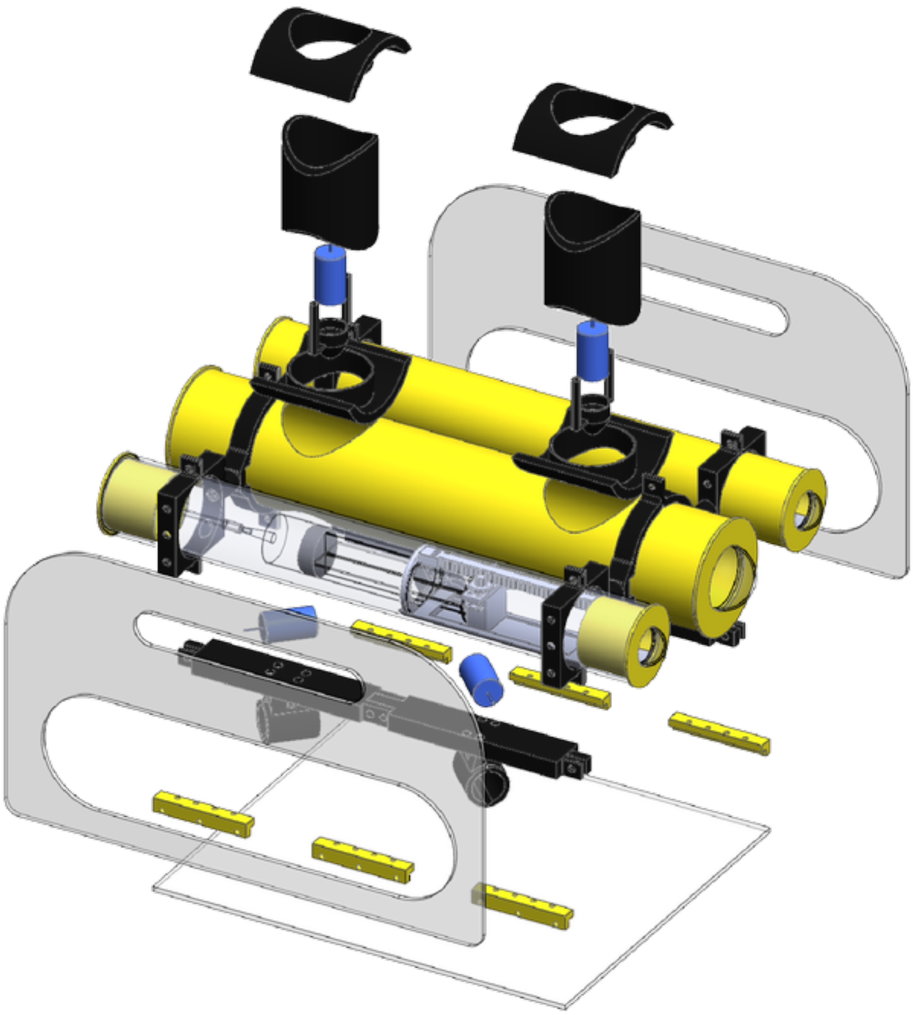


Figure 40: Exploded view of the first prototype

This exploded view allows to see with more details the way the different components that form the submarine are placed. It also shows the ballasts inside the side tubes, and the inner propellers that go through the central tube.

8.4. The main components

In the following section, the main components that format the submarine are shown with a more detailed view and an explanation o what their function is.

8.4.1. The pipes

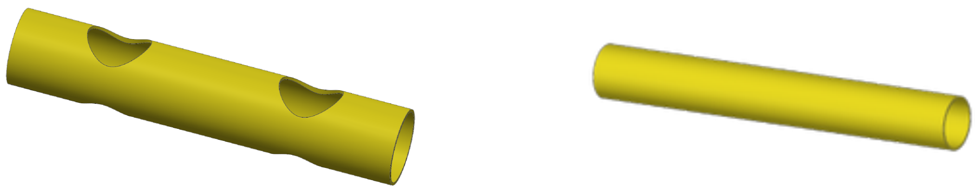


Figure 41: Big central pipe and smaller side pipe

This figure shows first the big central pipe, with the two holes in it for the inner propellers, and then the smaller side pipe.

8.4.2. Bottom propellers support and case

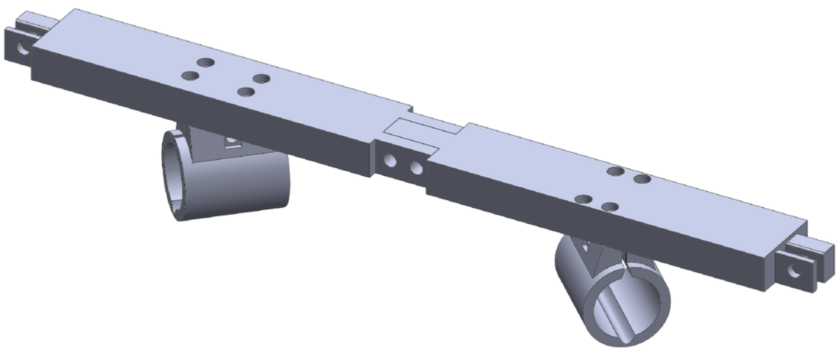


Figure 42: Bottom propellers support and case

The picture shows the base for the bottom propellers, and the cases that sustain the propellers attached to it. This part goes fixed at the bottom of the side tube, and there is one on each side.

8.4.3. Inner propellers protection

Figure 43: Inner propellers fixer and cap

The last figure shows on the left the tube that contains and fixes the inner propellers that go across the central tubes. Attached on top of this tube there is the cap, that adapts to the shape of the central tube.

8.5. Pipe closures

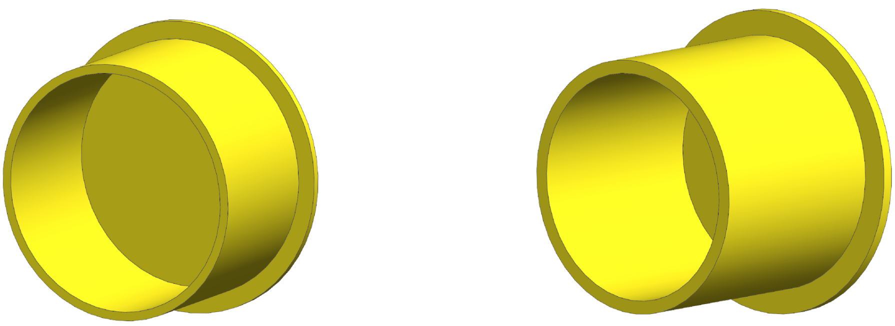


Figure 44: Pipe closures for the big and small tubes

These are the closures that go on the end of the 3 tubes. The one in the right is the closure for the bigger pipe and on the left is the closure for the small tubes. These will have to be properly sealed, since no water can enter in them because it would damage the electronic components.

8.6. Spheric plexiglass

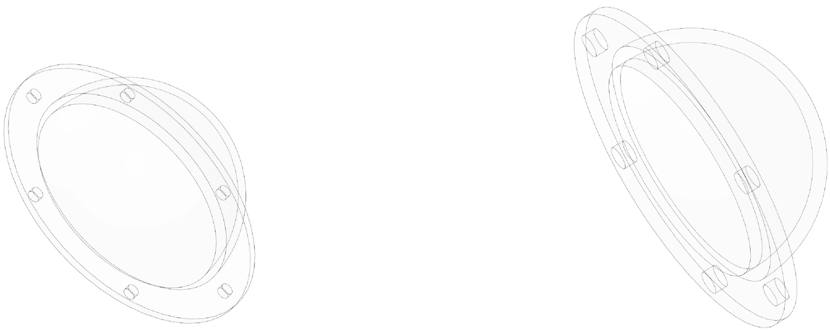


Figure 45: Spheric plexiglass for the big and small pipe

This figure shows the plexiglasses that will be placed in front of the three tubes. On the left is the plexiglass for the big pipe and on the right the one for the two small tubes. Inside the plexiglass of the central pipe there will be the camera for the final modes, and inside the small tubes' plexiglass there will be the LEDs.

8.7. Ballast system

The next part gives a detailed view of the 3D design of the ballast system. It gives a more proper vision on which are the components of the system, how they are placed inside the tubes and what their function is.

8.7.1. Exploded view of the inside of the side tubes

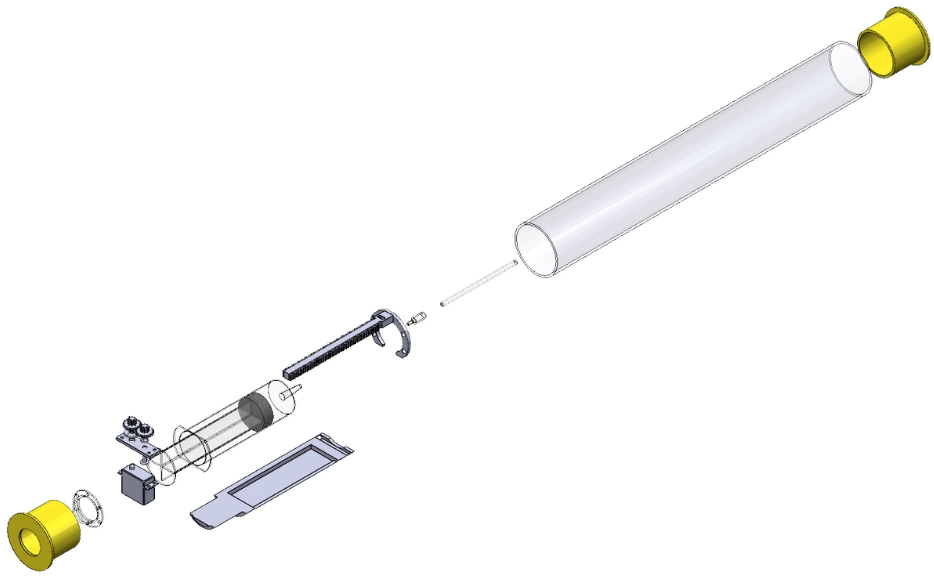


Figure 46: Exploded view of the ballast system

The last figure shows an overview of the inside of one of the side tubes, that helps to understand and visualize how the different components are placed inside the tube.

8.7.2. Global exploded view of the ballast system

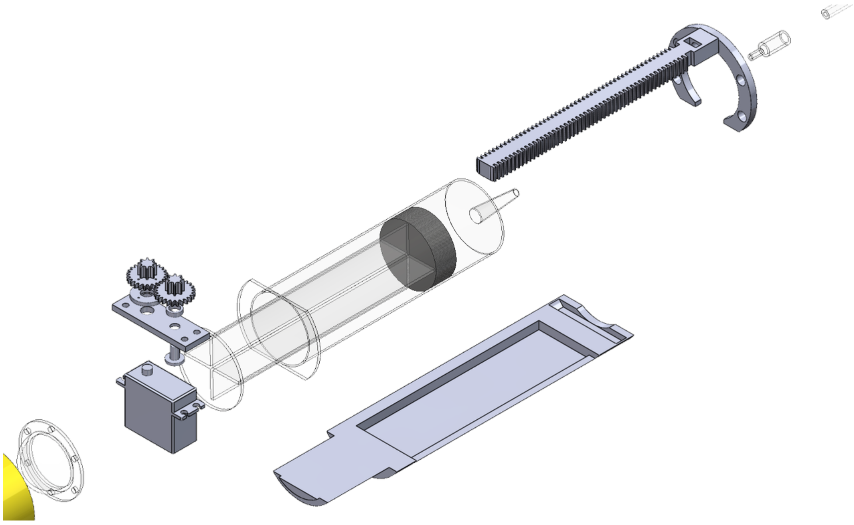


Figure 47: Exploded view of the ballast system

This picture shows a global view of all the items that compose the ballast system, the motor is attached to the plunger flange of the syringe, and the rack is attached to the barrel of the syringe.

8.7.3. Exploded view of the motor and gear system

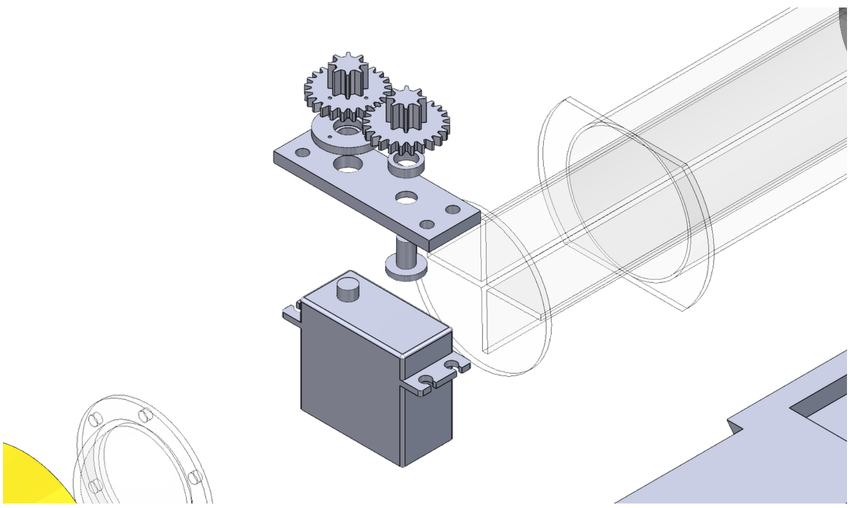


Figure 48: Exploded view of the motor and gears

In this last figure, a more detailed view of the gear system is shown, with a detail of all the small pieces that form it.

8.7.4. Base for the syringe

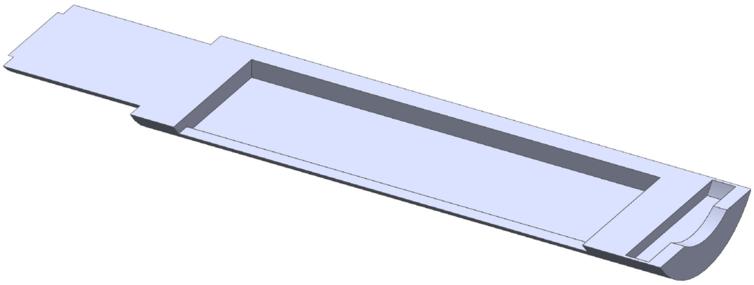


Figure 49: Syringe base

This part is a base where the syringes used for the buoyancy control are placed. Two of them are printed, one for each syringe, and they are placed and fixed inside the smaller tubes for correctly attaching the syringes.

8.7.5. Syringe fixer

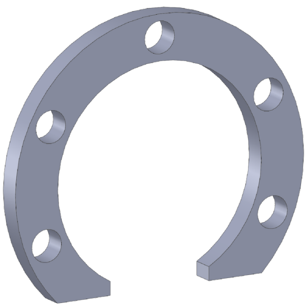


Figure 50: Syringe fixer

This is the part that attaches the barrel of the syringe to the tube that contains it, and that makes the barrel the fixed part of the ballast system.

8.7.6. Racks

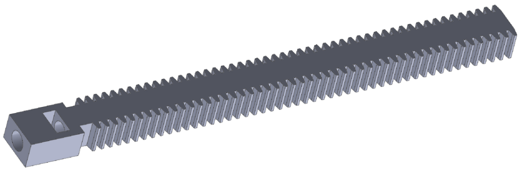


Figure 51: Rack

The last figure shows the design of the rack that is attached to the barrel of the syringe. As seen, it has teeth in both sides.

8.7.7. Gears

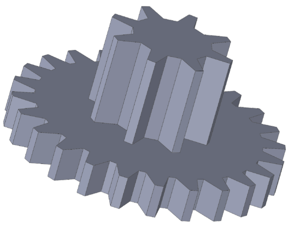


Figure 52: Gears

Two of these gears are 3D printed, and they are attached to the servo motor, transferring its rotation to both sides of the syringe rack.

8.7.8. Syringe representation

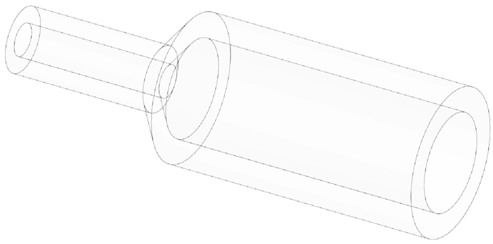


Figure 53: Representation of the syringe

The last picture represents the syringe where the previous system of gears, racks and attachment will be applied.

9. Inventory

Although many things will have to be ordered, some stuff like batteries, motors, chargers, and other electronic components were provided to the team for its use, as seen in the following inventory.

Part No.	Name	Manufacturer	type	Description	Quantity
1	Brushless Motor	Surpass Hobby	2435/4Y	4800 KV (non waterproof) 25A >Controller	6
2	Waterproof Screw propeller Motor	Unit 3D North Point Ho	N	N	8
3	Connector Brushless Motor	Surpass Hobby	3,175 mm 32DP	Gear to connect on the drive shaft	15
4	Servo Motor	N	MG996R	Motor to connect screws	9
5	Multifunction Balance Charger	Haisito	/	To load multiple Batterys	1
6	Float Switch	CJBIN	D-FQKG-CD0693	To detect de level of liquid	8
7	electronical RPM Control with brake	Drfeify	20A	to control the RPM of the Motor	6
8	Waterproof RPM Control		35A ESC	to control the RPM of the Motor	2
9	Brushless Motor	GoolRC	S2838	4500KV (non-waterproof)	2
10	Li-Po Guard	/	/	To protect Li-Po Batterys	2
11	Battery Pack	VPFlight	Li-Po Battery	2200mAh 11,1V 30C 24 Wh	4

Table 2: Inventory Chart

10. Material used

In the following part the material used for the making of the prototype is shown. There is a part for the hardware used, for the electronic components, and for the structural material, which includes apart from some material that was bought, the 3D printed parts that had been previously designed with solidworks.

10.1. Hardware for the first prototype

The hardware used is shown and explained in the next section. Some of it was provided by the university.

10.1.1. Arduino board

The Arduino board is used for monitoring the electronics. It controls the movement of the propellers, being able to activate each one for separate and make them turn in both directions. The Arduino also controls the movement of the two syringes, by implementing the PID control system on it, to modify the buoyancy of the submarine. The LED, as well as the pressure, sonar, distance, and temperature sensor will also be guided by the Arduino. The model used for this project is the Arduino Uno board.

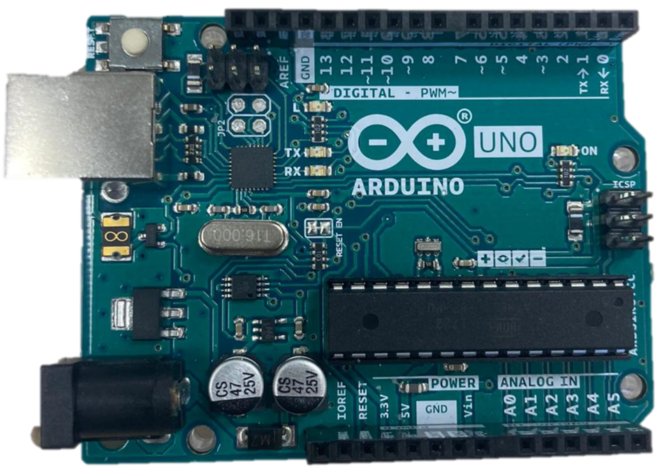


Figure 54: Arduino Uno Board

10.1.2. Breadboard

A breadboard is used for making all the necessary electronic connections when using the Arduino or the Raspberry for controlling the different devices.

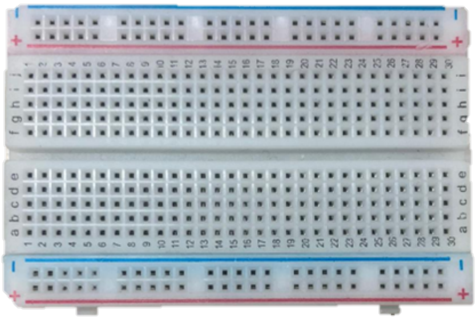


Figure 55: Breadboard

10.2. Electronics

Next, the electronics used are presented. Just like in the previous part, some of these were provided by the university while some had to be bought.

10.2.1. Li-Po Batteries

The batteries used for providing electricity to the motors, propellers and boards are Li-Po batteries. These provide a voltage of 11.1V, composed of 3 cells of 3.7V. (Beginners Guide to LiPo Batteries for FPV Drones, 2020)

Figure 56: Li-Po batteries

10.2.2. Brushless motors

Brushless motors are used for testing the correct functioning of the Arduino programming and circuits, but not for the final product, since these are too powerful to use for running the propellers. An electronic controller is used to switch DC currents to the motor windings, and this produces a magnetic field that rotates in space and makes the rotor of the motor rotate. The phase and amplitude of the DC are adjusted by the controller, which makes it able to control the speed and torque of the motor. (Control differences between ac induction motor and brushless dc motor?, 2012) The motor used is the BL MOTOR S2838.

Figure 57: Brushless motor

10.2.3. Servo motors

A servo motor is a rotary actuator through which it is possible to control the angle, speed, and acceleration. They are closed-loops servomechanisms that control their movement and final position through feedback. The black and the red cable are for the power supply and the orange cable is connected to the Arduino, which controls the angle of the servo motor. A digital signal is used as input to represent the position ordered for the output shaft. The servo motors are used to control the movement of the pistons of both syringes. They are connected to the pistons by using a system of gears and racks, making the piston move forward and backward as desired. The model used is the MG996R.

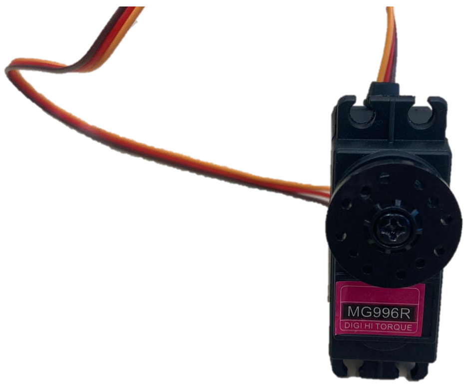


Figure 58: Servomotor

10.2.4. Brushed motors with propellers

Brushed motors are used for controlling the rotation of the propellers; thus each propeller will be attached to the brushed motor. This kind of motor have a similar way of working to the brushless motors, but the brushed ones have coils in their center rotating around permanent magnets while a brushless motor has permanent magnets in the center that rotate around the coil. (Bong, 2022).

Figure 59: Brushed DC Motor

10.2.5. Potentiometer

Potentiometers are resistance dividers consisting in three-terminal resistors that allow to adjust the voltage output by rotating a contact. The potentiometer is used when running the tests to control the rotating speed of the motors, as well as for simulating a pressure sensor when testing the PI controller for buoyancy control.

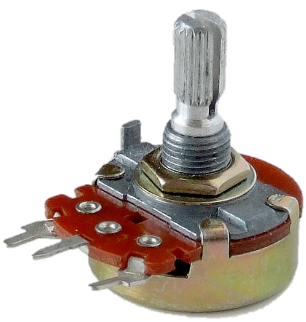


Figure 60: Potentiometer **(Wikipedia, 2022)**

10.2.6. 5V Power Supply

The Arduino is needing a stable 5V Power Supply. To provide this a UBEC is needed. Since there are already some ESCs with 5V output in the inventory these are used. It is also valuable to use them because the second prototype will run with brushless motors and needs ESCs.

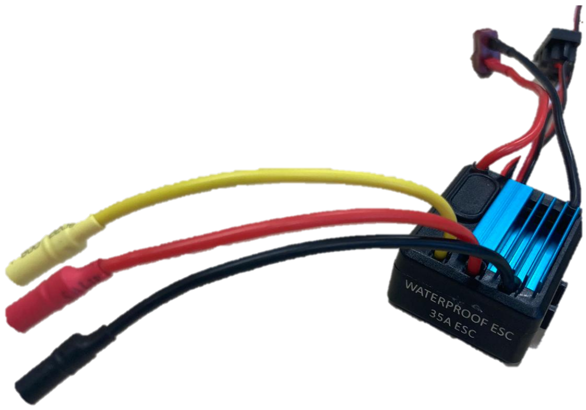


Figure 61: waterproof ESC

10.2.7. 8 relay module

Figure 62: 8 relay module

The relays are needed to control the motors. Two relays are required for each motor, except for the two horizontal motors, as these do not have to be controlled individually. The relays are controlled with 5 V and are separated from the 12 V system by optocouplers. The single relay card works the same way as the 8-relay card. The relays are connected with the secondary Arduino.

10.2.8. 1 relay module

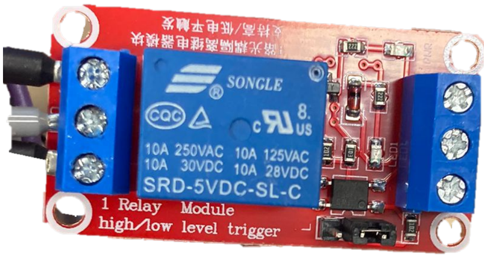


Figure 63: 1 relay module

10.2.9. Cable connectors

The cable connectors are used to supply the power from two batteries to the motors.

Figure 64: Cable connectors

10.2.10. Switches

To control the ROV the switches are used in the remote control.

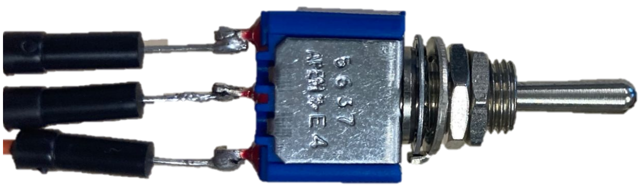


Figure 65: Switch

10.3. Structural material

In the following section, the material that forms the structure is presented. This includes parts that were bought and other items that were 3D printed. It includes also, apart from the parts that compose the structure the items that form the ballast system.

10.3.1. Pipes

Although the final product will have a different structure based on 3D-printed plates and tubes, 3 plastic tubes were bought for building and testing the first prototype. The tubes have a length of 575 mm. The big one, which is placed in the center has a diameter of 100 mm, and the two small tubes, placed on each side, have a diameter of 75mm.

Figure 66: One of the side tubes

10.3.2. Tube attachments

To attach the tubes 4 metallic attachments were bought, to be placed at each end of the two side plates, and 4 plastic attachments were 3D printed as a support to these. For the big central types, the attachments were directly 3D printed. The tubes are put together by using screws to join the attachments from the side pipes to the big central pipe.

Figure 67: Plastic and metallic attachment

10.3.3. Syringes

To control the buoyancy two syringes are placed on each side of the submarine. These have a total volume of 200 ml, and some parts have been cut and modified for adapting the syringes to the racks that will be attached to them for moving the syringe piston forwards and backward.

Figure 68: Syringe

10.3.4. Syringe base

For properly fixing the syringes to the side tubes a base has been 3D printed, with a flat top where the syringe is attached and a round bottom that fits the tube.

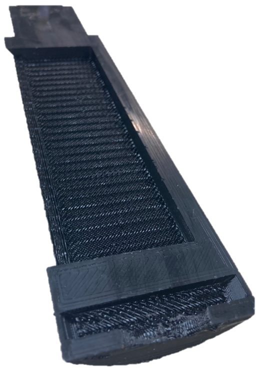


Figure 69: Syringe base

10.3.5. 3D-printed racks, gears, and syringe attachment

The mechanical system for converting the rotational movement of the servo motor into linear movement of the syringe plunger is based on 3D-printed gears and racks. Two racks are necessary, one for each piston, as well as two transmission systems. Two attachments are also required, for attaching each syringe barrel to the pipes.

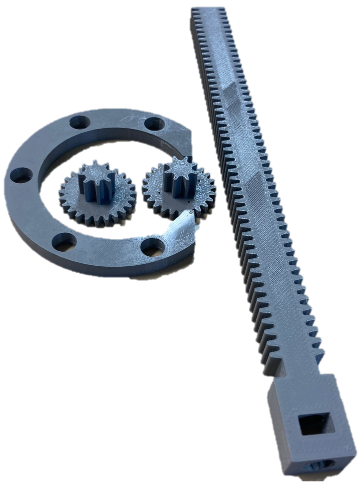


Figure 70: 3D printed rack gears and attachments

10.3.6. 3D printed propellers case and base

Two rectangular bases where the propellers are fixed have been printed. These are placed at the bottom of the two smaller tubes and screwed to the attachments of the pipes. To fix the propellers to their bases, a case has been 3D printed for each of them, and they are attached to the base with a screw.

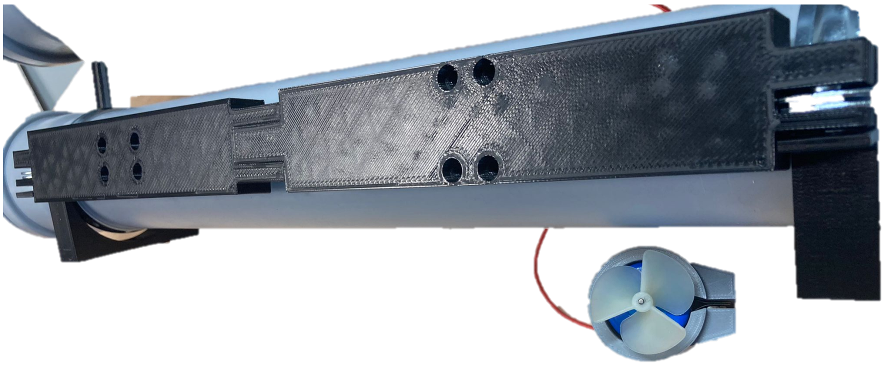


Figure 71: Rectangular base and propeller case

10.3.7. Inner propellers case

The propellers that make the submarine move up and down are placed in the center of the submarine and go through the pipes. For a proper attachment of these, and for being able to propel enough water and not having too much resistance with the body of the submarine, two cases have been printed, one for each inner propeller. However, these will finally not be used for the first prototype but can be used for the next version.

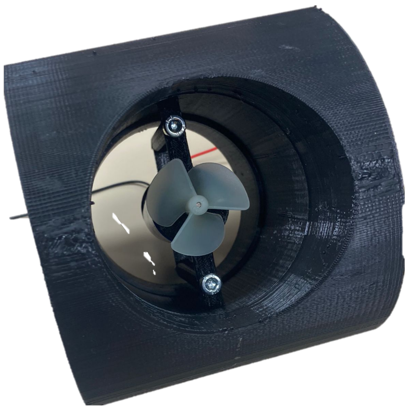


Figure 72: Inner propeller case

10.3.8. Screws

Different types and measures of screws are used in various parts of the prototype for securing, fixing, and attaching the items.

Figure 73: Screws

11. Software

11.1. PI Controller for buoyancy control

To enable buoyancy control, a PI Controller is required. The PI controller consists of a proportional part and an integral part. Due to the proportional part, the controller reacts quickly, but without an integral part a permanent control deviation would result since the setpoint/actual value adjustment is not possible due to the proportional part alone (StudyFlix, n.d.)

The following block diagram in figure 72 shows how the PI Control is structured.

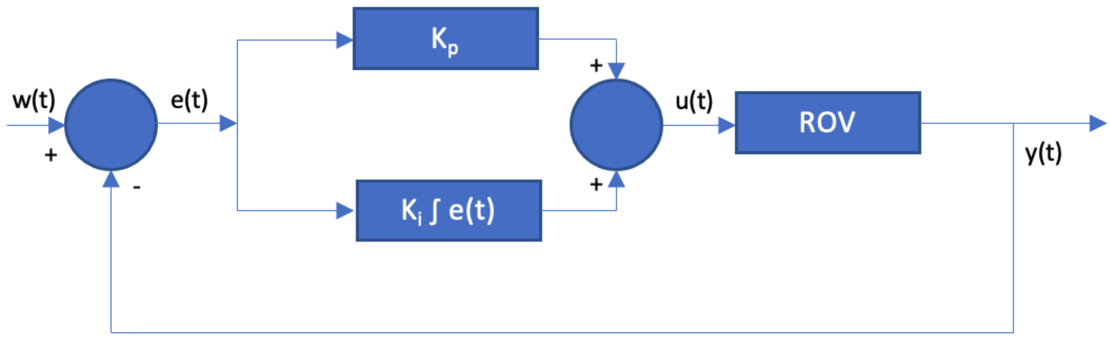


Figure 74: PI Controller

The w(t) part represents the setpoint, and the e(t) value is the error after subtracting the actual value y(t) from the setpoint w(t). The K_p value represents the proportional gain and the K_i value the integral gain. The u(t) value embodies the manipulated variable.

A big advantage is that there are already complete feedback control libraries for the Arduino. The PID_v2 library from Brett Beauregard was chosen. This library provides a ready-to-use PID controller which can be implemented in the source code. Only the K_p and K_i values must be adjusted and additionally the K_d value has to be zero to get an PI controller. If the values are set too high, the system may oscillate so that it does not reach the set point.

There are various rule of thumb methods for determining the values. For this purpose, the completed ROV must be allowed to descend, and the buoyancy control must be set to fully submerge. Then you can record the step response of the system. The step response is the dive rate over time. From this you can determine the K_p and K_i values with the appropriate procedure.

11.2. Control of the six thruster motors

A relay card and the Arduino are utilized to control the six drive motors. Since DC brush motors are used, no upstream controller is implemented. The motors are controlled individually via the relays, and the motors can only be switched on and off, and the direction of rotation can be changed. This can be made possible with two changeover relays, the so-called H bridge. In the following figure you can see how the system looks like on one motor.

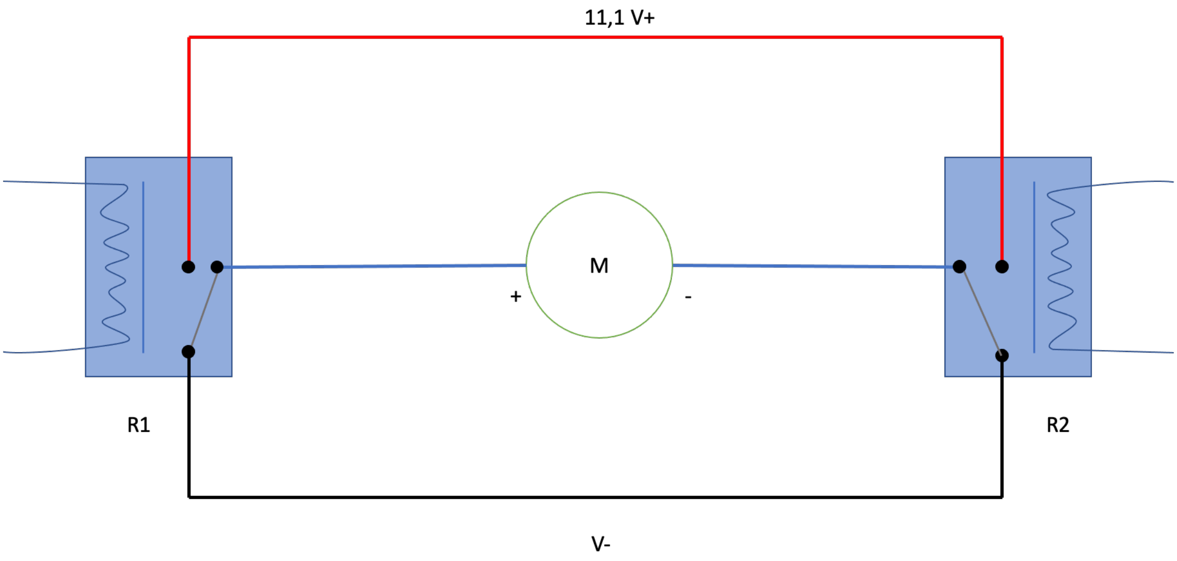


Figure 75: H-Bridge with DC motor

In the rest position of relay 1 and relay 2, the motor is at negative on both poles. As soon as R1 is activated, the motor receives a voltage of 11,1 V at the plus pole and rotates clockwise in this example. When R2 is activated and R1 is in rest position, the motor gets voltage at the minus pole and therefore rotates counterclockwise.

11.3. Remote control implementation

There are several ways to implement remote control. One possibility is to connect the switches via a long cable to the Arduino in the ROV. But you would need one cable per switch. In addition, a ground cable and a power supply. This possibility was not chosen, because with too many switches the cable becomes too big.

It was decided to use a system consisting of two Arduinos. One Arduino is in the ROV, the secondary Arduino, and one in the remote control, the primary Arduino. To ensure the communication between the two Arduinos, the Software Serial library is used. With this you can establish a serial connection between two devices. Two communication lines and a ground connection are needed.

A baud rate of 9600 baud was chosen, because it is fast enough, and a high range is guaranteed. In this case only one character, which consists of 8 bits, is transmitted per user request, for example "drive forward".

The use of this system is additionally advantageous because the primary Arduino in the remote control can be replaced by a Raspberry Pi and thus with the help of a graphical user interface and display the control can be facilitated, as well as a video transmission is possible.

The following table shows the pinout of the two Arduinos.

Table 3: Pin assignment primary Arduino

Arduino Pin Nummer	function	I/O
0 RX		
1 TX		
2	Switch forward	INPUT
3~	Switch backward	INPUT
4	Switch left	INPUT
5~	Switch right	INPUT
6~	Switch up	INPUT
7	Switch down	INPUT
8		
9~		
10~		
11~		
12	Software Rx	
13	Software Tx	
A0/14		
A1/15		
A2/16		
A3/17		
A4/18		
A5/19		

The primary Arduino gets the user's input through the switches and processes it. There are only inputs and the serial connection available. An extension to the buoyancy control is possible because many pins are free.

Table 4: Pin assignment secondary Arduino

Arduino Pin Nummer	function	I/O
0 RX		
1 TX		
2	RelaisIN1 left front motor +	OUTPUT
3~	RelaisIN2 left front motor -	OUTPUT
4	RelaisIN3 right front motor +	OUTPUT
5~	RelaisIN4 right front motor -	OUTPUT
6~	RelaisIN5 left back motor +	OUTPUT
7	RelaisIN6 left back motor -	OUTPUT
8	RelaisIN7 right back motor +	OUTPUT
9~	RelaisIN8 right back motor -	OUTPUT
10~	RelaisIN9 vertical motors +	OUTPUT
11~	RelaisIN10 vertical motors -	OUTPUT
12	Software Rx	
13	Software Tx	
A0/14		
A1/15		
A2/16		
A3/17		
A4/18		
A5/19		

The secondary Arduino receives the user request via the software serial connection to the primary Arduino and thus controls the relays. The secondary Arduino contains only outputs and the serial connection.

11.4. Arduino IDE

To be able to program a program and upload it to the Arduino an IDE (integrated development environment) is needed. Here the Arduino IDE is used. The program has the programming interface, downloadable libraries, a serial connection to the Arduino and a debug function.

For the buoyancy control the Servo library is used to control the servo motors and the ArduPID library for the feedback control.

To control the motors only the SoftwareSerial library is used.

The entire code can be found in appendix number 5 and 6.

11.4.1. Primary Arduino code explanation

In the following the code of the primary Arduino will be explained.

Code 1: Including of the SoftwareSerial library

```
#include <SoftwareSerial.h>
SoftwareSerial ArduinoSlave = SoftwareSerial(12, 13);  // Rx, Tx

```

As shown in Figure 55, the SoftwareSerial library is linked and the transmitter and receiver pins are set. The receiver pin Rx is 12 and the transmitter pin Tx is 13.

Code 2: Declaration of the pins and the variables

```
6   // Pins of the switches
7   const int switchforwards = 2;
8   const int switchbackwards = 3;
9   const int switchleft = 4;
10  const int switchright = 5;
11  const int switchup = 6;
12  const int switchdown = 7;
13
14  //Status of the switches
15  int statusswitchforwards = 0;
16  int statusswitchbackwards = 0;
17  int statusswitchleft = 0;
18  int statusswitchright = 0;
19  int statusswitchup = 0;
20  int statusswitchdown = 0;
```

To define all pin numbers, a constant integer is applied as shown in Figure 56, lines 7 to 12. Each value of the variable stands for one pin on the Arduino.

In line 15 to 20 variables are created to store the different switch positions. The value 0 is stored directly so that they are off at the beginning.

Code 3: Creation of the sender variable

```
//Variable to transmit to Slave
int mode = 0;

int modebefore = 0;
/*
0 is resting system
1 is forwards
2 is backwards
3 is left
4 is right
5 is up
6 is down
*/
```

As Figure 57 shows, the variables `mode` and `modebefore` are created. The variable `mode` is later sent to the secondary Arduino. In order not to fill up the buffer unnecessarily, the variable `modebefore` is used. Later in the program it can be seen how this works. As seen in the green annotation in the code, numbers from 0 to 6 represent the different driving modes of the ROV. These values are sent to the secondary Arduino depending on the user request.

Code 4: Void setup()

```
37  void setup() {
38
39    // declare the pins of I/O
40    pinMode(switchforwards, INPUT);
41    pinMode(switchbackwards, INPUT);
42    pinMode(switchleft, INPUT);
43    pinMode(switchright, INPUT);
44    pinMode(switchup, INPUT);
45    pinMode(switchdown, INPUT);
46
47    // Start the Rx Tx
48    ArduinoSlave.begin(9600);
49    Serial.begin(9600);
50  }
```

The `setup()` function runs when the Arduino is started. In this part mostly configurations are made. The previously created variables for the pins are now declared as output or input pins on the Arduino, as shown in Figure 58 lines 40 to 45. Since the primary Arduino only reads switches, all pins are defined as input.

Lines 48 and 49 start the serial connections. `ArduinoSlave.begin(9600);` starts a serial connection to the secondary Arduino with 9600 baud. `Serial.begin(9600);` initializes the connection to the USB connected computer. This is used to trace the data stream and debug the code.

Code 5: Readout of the switch positions

```
52  void loop() {
53    // read status of the switches
54    statusswitchforwards = digitalRead(switchforwards);
55    statusswitchbackwards = digitalRead(switchbackwards);
56    statusswitchleft = digitalRead(switchleft);
57    statusswitchright = digitalRead(switchright);
58    /* statusswitchup = digitalRead(switchup);
59    statusswitchdown = digitalRead(switchdown); */
```

After the `setup()` function the `loop()` function is executed. As the name suggests, it runs in a loop. In figure 59 line 54 to 59 the switch positions are read. While the switch is in rest position, which means no voltage is applied to the respective pin, a 0 is read in. As soon as the switch is activated, a number greater than 0 is read in. The number 0 corresponds with the state `LOW` and a number greater than 0 with the state `HIGH`.

Code 6: if loop to determine the mode

```
62  // drive forwards
63  if (statusswitchforwards == HIGH) {
64    mode = 1;
65  }
66
67  // drive backwards
68  else if (statusswitchbackwards == HIGH) {
69    mode = 2;
70  }
71
72  // drive left
73  else if (statusswitchleft == HIGH) {
74    mode = 3;
75  }
76
77  // drive right
78  else if (statusswitchright == HIGH) {
79    mode = 4;
80  }
81
82  // drive up
83  else if (statusswitchup == HIGH) {
84    mode = 5;
85  }
86
87  // drive down
88  else if (statusswitchdown == HIGH) {
89    mode = 6;
90  }
91
92  //rest
93  else {
94    mode = 0;
95  }
96  if (mode != modebefore) {
97    ArduinoSlave.print(mode);
98    // ArduinoSlave.print('x');
99    Serial.println(mode);
100   modebefore = mode;
101 } else {
102 }
```

The mode is determined in the if loop shown in Figure 60. Depending on the switch position, the mode is stored in the variable mode.

After the first `if` loop another one runs as shown in lines 96 to 102. This loop compares the last sent mode with the `modebefore`. If they are equal, the program jumps to `else` and starts again at the top of line 53. If `mode` is not equal to `modebefore`, `mode` is sent to the secondary Arduino via the bus connection. Likewise, for testing purposes, `mode` is sent to the computer and the value of `modebefore` is set equal to `mode`. After that the program runs again from line 53.

11.4.2. Secondary Arduino code explanation

In the following the code of the secondary Arduino will be explained.

Since the creation of variables and the `setup()` function was already discussed in the primary Arduino, it will not be discussed further here and started directly with the `loop()` function.

Code 7: Serial bus query

```
64 void loop() {
65
66   while (ArduinoMaster.available() == 0) {
67   }
68   Serial.println("receiving...");
69
70
71
72     char modechar = ArduinoMaster.read();
73
74     if (modechar == '0') {
75       mode = 0;
76     } else if (modechar == '1') {
77       mode = 1;
78     } else if (modechar == '2') {
79       mode = 2;
80     } else if (modechar == '3') {
81       mode = 3;
82     } else if (modechar == '4') {
83       mode = 4;
84     } else if (modechar == '5') {
85       mode = 5;
86     } else if (modechar == '6') {
87       mode = 6;
88     }
89       else {
90     }
```

Figure 61 shows the code for receiving data from the primary Arduino. In line 66, the `available()` function queries whether there is data in the buffer. If no data is received, the empty `while` loop runs. After a value is sent from the primary Arduino the requested value of `available()` changes to 1. Then the value is read from the buffer in line 72 and stored in the variable `modechar`. The following `if` loop converts the value stored in the `modechar` variable into the integer variable `mode`.

Code 8: if loop for selecting the relays to be controlled

```
94   //drive forwards
95   if (mode == 1) {
96     r1 = false;  //relais 1 on (left front motor clockwise)
97     r3 = false;  //relais 3 on (right front motor clockwise)
98     r5 = false;  //relais 6 on (left back motor counterclockwise)
99     r7 = false;  //relais 8 on (right back motor counterclockwise)
100  }
101  //drive backwards
102  else if (mode == 2) {
103
104    r2 = false;  //relais 2 on (left front motor counterclockwise)
105    r4 = false;  //relais 4 on (right front motor counterclockwise)
106    r6 = false;  //relais 5 on (left back motor clockwise)
107    r8 = false;  //relais 7 on (right back motor clockwise)
108  }
109  //drive to the left
110  else if (mode == 3) {
111    r1 = false;  //relais 1 on (left front motor clockwise)
112    r4 = false;  //relais 4 on (right front motor counterclockwise)
113  }
114  //drive to the right
115  else if (mode == 4) {
116    r2 = false;  //relais 2 on (left front motor counterclockwise)
117    r3 = false;  //relais 3 on (right front motor clockwise)
118  }
119  else if (mode == 5) {
120    r9 = false;
121  }
122  else if (mode == 6) {
123    r10 = false;
124  }
125  //Resting system
126  else if (mode == 0) {
127    //reset all Relais values
128    r1 = true;
129    r2 = true;
130    r3 = true;
131    r4 = true;
132    r5 = true;
133    r6 = true;
134    r7 = true;
135    r8 = true;
136    r9 = true;
137    r10 = true;
138  }
139  else {
140  }
```

In line 94 to 150 the mode to be executed is filtered by another `if` loop and the boolean variables of the individual relays are set to false. This means the specific relay gets controlled. For example, in mode 1, line 95 to 100, all motors are controlled to provide forward drive.

Code 9: Controlling the relays

```
142    //switch relais with the read booleans
143    digitalWrite(relaisIN1, r1);
144    digitalWrite(relaisIN2, r2);
145    digitalWrite(relaisIN3, r3);
146    digitalWrite(relaisIN4, r4);
147    digitalWrite(relaisIN5, r5);
148    digitalWrite(relaisIN6, r6);
149    digitalWrite(relaisIN7, r7);
150    digitalWrite(relaisIN8, r8);
151
152    delay(20);
```

The last code section controls the individual relays via the output pins, depending on the drive modes. After that the code starts again at the `while` loop.

11.4.3. Buoyancy control code explanation

In the following the code of the buoyancy control will be explained. The code is separated because it is not implemented, since the mechanical system is not finished.

The complete code can be found in appendix 7.

Code 10: Declaration of variables

```
#include <ArduPID.h>
#include <Servo.h>

ArduPID myController;
Servo bcontrol;

double setpoint = 25;
double input;
double output;
double p = 5;
double i = 2;
double d = 0;
double min = 0;
double max = 50;
int servooutput;
```

First the libraries must be added, and the variables must be declared. For the ArduPID a controller has to be created and also a servo for the servo library. The PID library needs some variables like the `setpoint`, `input`, `output` and the variables for the controller values. The d variable is set to zero because only a PI controller is needed. Additionally, the variable `servooutput` is created for the servo motor.

Code 11: `void setup()` of the buoyancy control

```
void setup() {
  bcontrol.attach(10, 500, 2300);  //Attach the Servo at Pin 9 with PWM from 500..2300
  Serial.begin(9600);
  myController.begin(&input, &output, &setpoint, p, i, d);
  Serial.println("chosen depth:");
  Serial.println(setpoint);
}

void loop() {
```

In `void setup()` the servo is connected to pin 10 and the pulse width control is set. Additionally, the serial connection to the computer is started and the PID controller is started. Here in the code the `setpoint` is already set to 25 meters. in the finished code the `setpoint` should be set by a user query. In addition, the `setpoint` is sent to the computer.

Code 12: `void loop()` of the buoyancy control

```
25 void loop() {
26
27   input = analogRead(A0);  //Pressure Sensor simulated by potentiometer
28   input = input / 20;      //convert 0...1023 to 0...50 m
29
30   myController.compute();
31   myController.debug(&Serial, "myController", PRINT_INPUT |     // Can include or comment out any of these terms to print
32                                                 PRINT_OUTPUT |  // in the Serial plotter
33                                                 PRINT_SETPOINT | PRINT_BIAS | PRINT_P | PRINT_I | PRINT_D);
34   servooutput = map(output, 0, 255, 0, 180);
35
36   Serial.print("Variable_1:");
37   Serial.print(servooutput);
38   Serial.print(",");
39   Serial.print("Variable_2:");
40   Serial.println(input);
41
42   bcontrol.write(servooutput);
43 }
```

In the `void loop()`, the `input` value is first read in lines 27 to 28. Here a potentiometer simulates the pressure sensor. The read value between 0 and 1023 is converted to a depth of 0 to 50 meters.

Then the pi controller starts and gives an `output` between 0 and 255. This value is mapped to 0 to 180 degrees in the variable `servooutput`. The servomotor is then controlled with the value stored in `servooutput`.

12. Outlook

Since the first prototype has not yet reached the project goal, a small outlook on the continuation is given here.

The existing system consisting of the two Arduinos can be extended by one or more Raspberries. One Raspberry is needed directly in the ROV to process and forward the video signal of the camera. This can also be used to control the secondary Arduino. By using brushless motors, the battery life and performance can be improved. Outrunners should be used as they are easier to mount. Also, the Plexiglas spheres and the LED light should be installed. A patch cable can be used to connect the Raspberry in the ROV to another Raspberry in the ground station. By using patch cables, you can achieve long cable lengths. The primary Arduino is omitted when using a Raspberry in the ground station. The Raspberry in the ground station or remote control can be equipped with a display which allows the user to see the camera and all values and simplifies the control via a graphical user interface. An appropriate program must be written for this purpose. A battery controller must also be installed in the ROV, so that the current battery status is available. For input a controller can also be used which is connected to the Raspberry via USB.

An existing system can also be used. The ArduSub system offers many functions, as well as a complete software that is already well equipped. The frame of the ROV is supported. Everything that is needed is presented in the ArduSub wiki.

The use of wastewater pipes can be kept, as they are very stable. The cable entry into the ROV needs to be revised for deeper dives.

The mechanics of the buoyancy control must be improved because it still has too high tolerances in the gear drive.

13. Conclusions

In general, it can be said that the goals that the team had at the beginning of this project have been achieved, since a successful prototype has been designed and built on schedule. However, there were some requirements that the final product should have that couldn't be met and will be left for upcoming EPS projects.

From the beginning it was clear that this was a challenging project and that it would be difficult, regarding the lack of experience of the team members in a task like this and the limited time for completing the project.

However, the team did the research that was necessary to acquire the needed knowledge for completing several parts of the project and put hard work in the fields in which they already had more expertise.

Many parts and components were designed with solidworks, as well as a structure for the prototype. This structure, however, was finally not used for the prototype, since there were some problems with the 3D printing, because the prototype was too big for the available printers, and it was also difficult to divide it by parts and print them separately. For this reason, it was finally decided to use pipes as the structure of the submarine. The CAD design, however, is saved and could be available to use for upcoming projects.

As for the electronics, remote control and video camera, the initial idea was that the submarine would be controlled from the ground module with a remote control, and the submarine should have a camera through which a live stream of the ROV's view would be played. However, this could also not be implemented, due to the complexity of it and the lack of expertise of the team in this field and the first prototype doesn't have a camera, and the propellers of the submarine are controlled by on/off switches.

To sum up, the team worked hard to do the research, designing, building, and programming of a decent prototype, and the result has been quite convincing despite not being able to meet with all the requirements that the final submarine will have.

14. Resources

novia.fi. (2022). Retrieved 09 19, 2022, from https://www.novia.fi/about-us

European Project Semester. (2022, 09 20). Retrieved from European Project Semester: europeanprojectsemester.eu/concept

Mind Tools Team. (2022, 10 06). *Mind Tools*. Retrieved from https://www.mindtools.com/pages/article/newLDR_83.htm

Hollins, P. (2022). Project Management Course. Vaasa.

Fossen, T. I. (2022, 10 17). *ResearchGate*. Retrieved from https://www.researchgate.net/profile/Thor-Fossen/publication/221787728_Guidance_Laws_for_Autonomous_Underwater_Vehicles/links/0912f50a28572b4264000000/Guidance-Laws-for-Autonomous-Underwater-Vehicles.pdf?origin=publication_detail

He, Y., Wang, D., & Ali, Z. (2020). A review of different designs and control models of remotely operated underwater vehicle. *Measurement and Control*, 1561-1570.

Doornekamp, R. (2021, 06 24). *DeepTrekker*. Retrieved from https://www.deeptrekker.com/news/top-3-uses-aquaculture-rov

(2021). *Unnamed Underwater Vehicles.* Thematic Research.

Salomaa, T. (2017). *DEPTH CONTROL SYSTEM ON AN AUTONOMOUS MINIATURE.*

Lapresa, M. (2011).

Mohammad Harun-Or-Rashid, D. B. (2019). *Fabrication and Study of an Unmanned Under Water Vehicle.*

Ryan. (2019). *Sensored vs Sensorless Brushless Motor Applications.*

youmatter. (2020, 05 16). Retrieved from https://youmatter.world/en/definition/aquaponics-sustainable-benefits-system/

Committee, R. O. (2016). *ROV Cattegories-Summary.*

Source IEx. (2022, 10 22). Retrieved from https://www.sourceiex.com/Catalogs/IP%20Degress%20Testing%20Details.pdf

(2018, 02 05). Retrieved from Arduino.cc: https://www.arduino.cc/en/Guide/Introduction

(2013). *What is Electronic Speed Control (ESC) & Its Working.* elprocus.

(2022). Retrieved from ArduSub: https://www.ardusub.com/introduction/required-software.html

(2020). *Beginners Guide to LiPo Batteries for FPV Drones.* fpvfc.

(2012). *Control differences between ac induction motor and brushless dc motor?* Stack Exchange.

Bong, N. (2022). *Using Brushed vs Brushless DC Motors.*

StudyFlix. (n.d.). *PI Regler*. Retrieved 11 17, 2022, from StudyFlix: https://studyflix.de/informatik/pi-regler-1430

Grimal, P. (2008). *Diccionario de la miotlogía grieca y romana.*

Vevor.ca. (2022). Retrieved from https://www.vevor.ca/outboard-propeller-c_11821/vevor-48-8m0084495-outboard-propeller-boat-propeller-4-blade-14-nbsp-x-19-aluminum-p_010747149560

researchgate. (2013). Retrieved from https://www.researchgate.net/figure/Buoyancy-control-device-mechanism_fig2_257880809

Amazon. (2022). Retrieved from https://www.amazon.com/T-king-Waterproof-Ultrasonic-Distance-Sensor/dp/B08SQSGKNF

RS components. (2022). Retrieved from https://au.rs-online.com/web/p/pressure-sensors/7975046

Berrybase. (2022). Retrieved from https://www.berrybase.de/es/gy-87-mpu-6050-beschleunigungssensor/gyroskop-hmc5883l-magnetfeldmesser-bmp180-barometer

Gigganti. (2022). Retrieved from https://www.gigantti.fi/product/urheilu-ja-vapaa-aika/kamerat-ja-valokuvaus/kamerat/actionkamerat/gopro-hero-9-black-actionkamera/214889

Kameraliike. (2022). Retrieved from https://www.kameraliike.fi/fi/product/gopro-hero-10-black-actionkamera/18832

Fotonordic. (2022). Retrieved from https://www.fotonordic.fi/product/117339/insta360-one-rs-twin-edition

Dustinhome. (2022). Retrieved from https://www.dustinhome.fi/product/5010946006/camera-v2-video-module

Puuilo. (2022). Retrieved from https://www.puuilo.fi/wurth-tiivisteliima-silikon-special-250-70ml?gclid=Cj0KCQiA4uCcBhDdARIsAH5jyUknzB3TZ0UtES0XTi-tG_W-NAos8HqBJZLA1zugjKPMPwKxLkcRptgaAi2OEALw_wcB

Swift supplies. (2022). Retrieved from https://www.swiftsupplies.com.au/c4400-gasket-material-1000mm-x-1500mm-sheets

Valokas. (2022). Retrieved from https://valokas.fi/tuote/g4-sivu-12-smd-led-5050-10-30v-24/

Wikipedia. (2022). Retrieved from https://simple.wikipedia.org/wiki/Potentiometer#/media/File:Potentiometer.jpg

Appendix

Appendix 1: Rough calculation of the buoyancy

$$V_{Tubes} = r^2 * \pi * l$$

Equation 2: Volume of a cylinder

with:

The radius of the tubes:

$$r_1 = 65\ mm$$

$$r_2 = 37{,}5\ mm$$

Length of the tubes:

$$l = 580\ mm$$

Calculation of tube one:

$$V_{Tube,1} = (65\ mm)^2 * \pi * 580\ mm$$

$$V_{Tube,1} = 7{,}6985 * 10^6\ mm^3$$

$$V_{Tube,1} = 0{,}0076985\ m^3$$

Calculation of tube two:

$$V_{Tube,2} = (37{,}5\ mm)^2 * \pi * 500\ mm$$

$$V_{Tube,2} = 2{,}5624 * 10^6\ mm^3$$

$$V_{Tube,2} = 0{,}0025624\ m^3$$

Calculation of the total volume:

$$V_{Tube,total} = V_{Tube,1} + 2 * V_{Tube,2}$$

There are two $V_{Tube,2}$ and one $V_{Tube,1}$

$$V_{Tube,total} = 0{,}0025624\ m^3 + 0{,}0076985\ m^3$$

$$V_{Tube,total} = 0{,}012823\ m^3$$

Calculation of the buoyancy force:

$$F_{Buoyancy} = \rho_{water} * V_{Tube,total} * g_{earth}$$

Equation 3: Hydrostatic Buoyancy Forces

With:

Density of water:

$$\rho_{water} = 998\frac{kg}{m^3}$$

Gravitational acceleration:

$$g_{earth} = 9{,}81\ \frac{m}{s^2}$$

$$F_{Buoyancy} = 998\frac{kg}{m^3} * 0{,}012823\ m^3 * 9{,}81\ \frac{m}{s^2}$$

$$F_{Buoyancy} = 125{,}54\ \mathrm{N}$$

To calculate the necessary weight of the UUV we use Newton's Second Law:

with:

$$F = m * a$$

Equation 4: Newton's Second Law

$$\Leftrightarrow m = \frac{F}{a}$$

$$m_{Submarine} \approx \frac{F_{Buoyancy}}{g}$$

$$m_{Submarine} \approx \frac{125,54}{9,81}$$

$$m_{Submarine} \approx \underline{12.8\ kg}$$

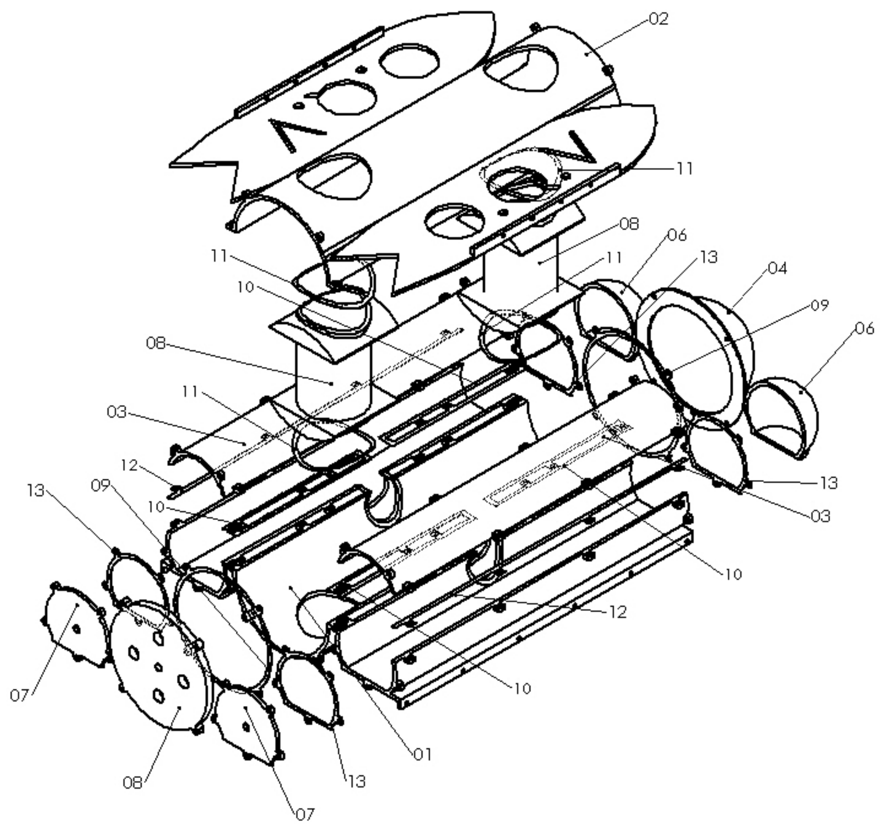
YRKESHÖGSKOLAN NOVIA

Appendix 2: Gantt Chart

WBS NUMBER	TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION	% of TASK COMPLETE
1	**Project Set-Up**		06.09.22	07.10.22		100%
1.1	**Reserach**	Team	06.09.22	07.10.22	31	100%
1.1.1	Brainstorming	Team	06.09.22	28.09.22	22	100%
1.1.2	Component search	Ole and Quim	15.09.22	05.10.22	20	100%
1.1.3	Look for inspiration	Team	06.09.22	15.09.22	9	100%
1.1.4	Set reqirements	Supervisor	06.09.22	15.09.22	9	100%
1.2	**3D modeling**		14.09.22	05.10.22	21	100%
1.2.1	Look for ideas	Team	14.09.22	17.09.22	3	100%
1.2.2	Create the different parts	Valérian	14.09.22	05.10.22	21	100%
1.2.3	Put them together	Valérian	14.09.22	05.10.22	21	100%
1.2.4	Introduce modificaitions	Valérian	21.09.22	05.10.22	14	100%
1.3	**Component search**		15.09.22	12.10.22	27	100%
1.3.1	Do the inventory	Team	15.09.22	22.09.22	7	100%
1.3.2	Choose the components needed	Ole and Quim	15.09.22	05.10.22	20	100%
1.3.3	Buy the components	Team	05.10.22	12.10.22	7	100%
2	Programming and building		10.10.22	28.10.22		100%
2.1	**Programming**		10.10.22	28.10.22	18	100%
2.1.1	Lessons	Hans/Tobias	10.10.22	14.10.22	4	100%
2.1.2	Self-studying	Ole	03.10.22	21.10.22	18	100%
2.1.3	Doing tests	Ole	10.10.22	21.10.22	11	100%
2.1.4	Finalize the programming	Ole	24.10.22	28.10.22	4	100%
2.2	**3D printing**		21.09.22	21.10.22	30	100%
2.2.1	Lessons	Osku	21.09.22	21.09.22	0	100%
2.2.2	Print some prototypes	Valérian	03.10.22	07.10.22	4	100%
2.2.3	Material choice	Valérian	10.10.22	11.10.22	1	100%
2.2.4	Printing the final parts	Valérian	10.10.22	21.10.22	11	100%
2.3	**Assembling**		24.10.22	28.10.22	4	100%
2.3.1	Collect the parts	Valérian	24.10.22	26.10.22	2	100%
2.3.2	Assemble the parts	Team	26.10.22	28.10.22	2	100%
3	Testing		10.10.22	18.11.22		100%
3.1	**Waterproof testing**		31.10.22	04.11.22	4	100%
3.1.2	Classify the materials	Ole	14.10.22	15.10.22	1	100%
3.1.3	Choose the correct ones	Ole	15.10.22	16.10.22	1	100%
3.2	**Electronic testing**		10.10.22	28.10.22	18	100%
3.2.1	Test motors and batteries	Quim	10.10.22	15.10.22	5	100%
3.2.2	Test the camera	Ole	18.10.22	24.10.22	6	0%
3.2.3	Test the sensors	Team	18.10.22	24.10.22	6	0%
3.2.4	Test the programming	Ole	01.11.22	04.11.22	3	100%
3.3	**Final model testing**		07.11.22	15.11.22	8	100%
3.3.1	Test the buoyancy	Team	07.11.22	08.11.22	1	50%
3.3.2	Neutralize the buyancy	Ole	08.11.22	10.11.22	2	0%
3.3.3	Test the whole submarine	Team	11.11.22	15.11.22	4	33%
4	Documentation		10.09.22	24.11.22		100%
4.1	**Midterm report**		06.09.22	14.10.22	38	100%
4.1.1	Create a template	Ole	06.09.22	10.09.22	4	100%
4.1.2	Introduce the team	Team	22.09.22	24.09.22	2	100%
4.1.3	Present the work done	Team	25.10.22	26.10.22	1	100%
4.2	**Final report**		06.09.22	14.12.22	99	100%
4.2.1	Midterm report contents	/			0	100%
4.2.2	3D Modeling	Valérian			0	100%
4.2.3	Programming	Ole			0	100%
4.2.4	Testing	Quim & Valérian			0	100%
4.2.5	Conclusions	Team			0	100%

NOVIA

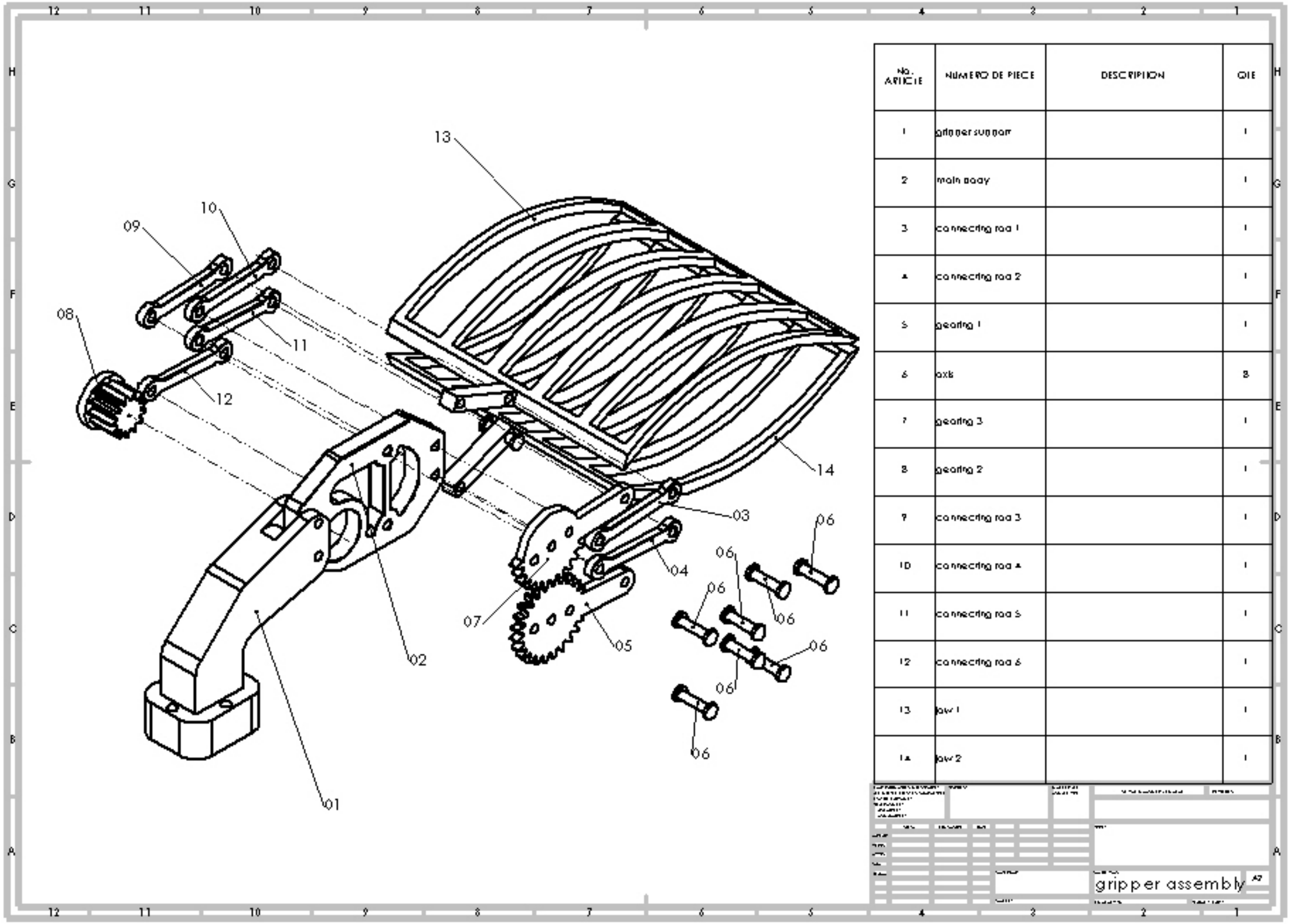
Appendix 3: Exploded Tank assembly



No. ARTICLE	NUMERO DE PIECE	DESCRIPTION	QTE
1	upper inner lower part		1
2	upper inner upper part		1
3	high closing ballasts		2
4	camera part		1
5	ballast front closure		2
6	tank back closure		1
7	ballast rear closure		2
8	inner tube		1
9	large tank round seal		2
10	central section joint - 1		4
11	inner motor seal		4
12	central section joint - 2		2
13	seal for ballasts		4

tank assembly A2

Appendix 4: Exploded Gripper assembly



No. ARTICLE	NUMERO DE PIECE	DESCRIPTION	QTE
1	gripper support		1
2	main body		1
3	connecting rod 1		1
4	connecting rod 2		1
5	gearing 1		1
6	axis		8
7	gearing 3		1
8	gearing 2		1
9	connecting rod 3		1
10	connecting rod 4		1
11	connecting rod 5		1
12	connecting rod 6		1
13	jaw 1		1
14	jaw 2		1

Appendix 5: Code of primary Arduino

```
//Master code

#include <SoftwareSerial.h>
SoftwareSerial ArduinoSlave = SoftwareSerial(12, 13);  // Rx, Tx

// Pins of the switches
const int switchforwards = 2;
const int switchbackwards = 3;
const int switchleft = 4;
const int switchright = 5;
const int switchup = 6;
const int switchdown = 7;

//Status of the switches
int statusswitchforwards = 0;
int statusswitchbackwards = 0;
int statusswitchleft = 0;
int statusswitchright = 0;
int statusswitchup = 0;
int statusswitchdown = 0;

//Variable to transmit to Slave
int mode = 0;

int modebefore = 0;
/*
0 is resting system
1 is forwards
2 is backwards
3 is left
4 is right
5 is up
6 is down
*/


void setup() {

  // declare the pins of I/O
  pinMode(switchforwards, INPUT);
  pinMode(switchbackwards, INPUT);
  pinMode(switchleft, INPUT);
  pinMode(switchright, INPUT);
  pinMode(switchup, INPUT);
  pinMode(switchdown, INPUT);

  // Start the Rx Tx
  ArduinoSlave.begin(9600);
  Serial.begin(9600);
}

void loop() {
  // read status of the switches
  statusswitchforwards = digitalRead(switchforwards);
  statusswitchbackwards = digitalRead(switchbackwards);
  statusswitchleft = digitalRead(switchleft);
  statusswitchright = digitalRead(switchright);
  statusswitchup = digitalRead(switchup);
  statusswitchdown = digitalRead(switchdown);


  // drive forwards
  if (statusswitchforwards == HIGH) {
    mode = 1;
  }

  // drive backwards
  else if (statusswitchbackwards == HIGH) {
    mode = 2;
  }

  // drive left
  else if (statusswitchleft == HIGH) {
    mode = 3;
```

```
  }

  // drive right
  else if (statusswitchright == HIGH) {
    mode = 4;
  }

  // drive up
  else if (statusswitchup == HIGH) {
    mode = 5;
  }

  // drive down
  else if (statusswitchdown == HIGH) {
    mode = 6;
  }

  //rest
  else {
    mode = 0;
  }
  //compare mode with sent mode
  if (mode != modebefore) {
    //send new mode
    ArduinoSlave.print(mode);
    // ArduinoSlave.print('x');
    Serial.println(mode);
    modebefore = mode;
  }
  else {
//repeat the loop
  }
  delay(50);
}
```

Appendix 6: Code of secondary Arduino

```
//Slave code

#include <SoftwareSerial.h>
SoftwareSerial ArduinoMaster = SoftwareSerial(12, 13);

int mode = 0;

//Pins of the Relais
const int relaisIN1 = 2;  //left front clockwise
const int relaisIN2 = 3;  //left front counterclockwise
const int relaisIN3 = 4;  //right front clockwise
const int relaisIN4 = 5;  //right front counterclockwise
const int relaisIN5 = 6;  //left back clockwise
const int relaisIN6 = 7;  //left back counterclockwise
const int relaisIN7 = 8;  //right back clockwise
const int relaisIN8 = 9;  //right back counterclockwise

//Relais to go up and down
const int relaisIN9 = 10;   // Motor up
const int relaisIN10 = 11;  //Moter down

//boolean variables to switch the relais
bool r1 = true;
bool r2 = true;
bool r3 = true;
bool r4 = true;
bool r5 = true;
bool r6 = true;
bool r7 = true;
bool r8 = true;
bool r9 = true;
bool r10 = true;

void setup() {
  // Start Serial communication
  Serial.begin(9600);
  ArduinoMaster.begin(9600);

  //declare the pins of the IN/OUTPUTS
  pinMode(relaisIN1, OUTPUT);
  pinMode(relaisIN2, OUTPUT);
  pinMode(relaisIN3, OUTPUT);
  pinMode(relaisIN4, OUTPUT);
  pinMode(relaisIN5, OUTPUT);
  pinMode(relaisIN6, OUTPUT);
  pinMode(relaisIN7, OUTPUT);
  pinMode(relaisIN8, OUTPUT);
  pinMode(relaisIN9, OUTPUT);
  pinMode(relaisIN10, OUTPUT);

  //turn off every relais
  digitalWrite(relaisIN1, r1);
  digitalWrite(relaisIN2, r2);
  digitalWrite(relaisIN3, r3);
  digitalWrite(relaisIN4, r4);
  digitalWrite(relaisIN5, r5);
  digitalWrite(relaisIN6, r6);
  digitalWrite(relaisIN7, r7);
  digitalWrite(relaisIN8, r8);
  digitalWrite(relaisIN9, r9);
  digitalWrite(relaisIN10, r10);
}

void loop() {

  while (ArduinoMaster.available() == 0) {
  }
  Serial.println("receiving...");

  if (ArduinoMaster.available() > 0) {

    char modechar = ArduinoMaster.read();

    if (modechar == '0') {
      mode = 0;
```

```
  } else if (modechar == '1') {
    mode = 1;
  } else if (modechar == '2') {
    mode = 2;
  } else if (modechar == '3') {
    mode = 3;
  } else if (modechar == '4') {
    mode = 4;
  } else if (modechar == '5') {
    mode = 5;
  } else if (modechar == '6') {
    mode = 6;
  }
    else {
  }

Serial.println(mode);

  //drive forwards
  if (mode == 1) {
    r1 = false;  //relais 1 on (left front motor clockwise)
    r3 = false;  //relais 3 on (right front motor clockwise)
    r5 = false;  //relais 6 on (left back motor counterclockwise)
    r7 = false;  //relais 8 on (right back motor counterclockwise)
  }

  //drive backwards
  else if (mode == 2) {

    r2 = false;  //relais 2 on (left front motor counterclockwise)
    r4 = false;  //relais 4 on (right front motor counterclockwise)
    r6 = false;  //relais 5 on (left back motor clockwise)
    r8 = false;  //relais 7 on (right back motor clockwise)
  }

  //drive to the left
  else if (mode == 3) {
    r1 = false;  //relais 1 on (left front motor clockwise)
    r4 = false;  //relais 4 on (right front motor counterclockwise)
  }

  //drive to the right
  else if (mode == 4) {
    r2 = false;  //relais 2 on (left front motor counterclockwise)
    r3 = false;  //relais 3 on (right front motor clockwise)
  }


  else if (mode == 5) {
    r9 = false;
  }


  else if (mode == 6) {
    r10 = false;
  }

  //Resting system
  else if (mode == 0) {
    //reset all Relais values
    r1 = true;
    r2 = true;
    r3 = true;
    r4 = true;
    r5 = true;
    r6 = true;
    r7 = true;
    r8 = true;
    r9 = true;
    r10 = true;

  }

  else {
  }
```

```
      //switch relais with the read booleans
      digitalWrite(relaisIN1, r1);
      digitalWrite(relaisIN2, r2);
      digitalWrite(relaisIN3, r3);
      digitalWrite(relaisIN4, r4);
      digitalWrite(relaisIN5, r5);
      digitalWrite(relaisIN6, r6);
      digitalWrite(relaisIN7, r7);
      digitalWrite(relaisIN8, r8);

      delay(20);
    }
    else {

    }
  }
```

Appendix 7: Code of the buoyancy controller

```
#include <ArduPID.h>
#include <Servo.h>

ArduPID myController;
Servo bcontrol;

double setpoint = 25;
double input;
double output;
double p = 5;
double i = 2;
double d = 0;
double min = 0;
double max = 50;
int servooutput;

void setup() {
  bcontrol.attach(10, 500, 2300);  //Attach the Servo at Pin 9 with PWM from 500..2300
  Serial.begin(9600);
  myController.begin(&input, &output, &setpoint, p, i, d);
  Serial.println("chosen depth:");
  Serial.println(setpoint);
}

void loop() {

  input = analogRead(A0);  //Pressure Sensor simulated by potentiometer
  input = input / 20;      //convert 0...1023 to 0...50 m

  myController.compute();
  myController.debug(&Serial, "myController", PRINT_INPUT |     // Can include or comment out any of these terms to print
                                                PRINT_OUTPUT |  // in the Serial plotter
                                                PRINT_SETPOINT | PRINT_BIAS | PRINT_P | PRINT_I | PRINT_D);
  servooutput = map(output, 0, 255, 0, 180);

  Serial.print("Variable_1:");
  Serial.print(servooutput);
  Serial.print(",");
  Serial.print("Variable_2:");
  Serial.println(input);

  bcontrol.write(servooutput);
}
```