University of Southern Denmark

MDB2 - Autonomous Buoy

Report Author:

Anders Kristensen
Jegvan Jon Hansen
Jens Holger Haastrup Sheeran
Julius Ferdinand Andreas Born
Jón Geir Sveinsson
Kun Qian
Markus Hofmann
Martin Hviid Kristensen
Michael Hviid Aarestrup
Rinson Davis
Ying Gao

Course:
Mechatronics Design and Build 2
06.01.2018

Abstract

The scope of this project was to make a autonomous buoy. The development was divided into four main aspects: mechanical, control, hardware and software. The project was concluded, due to time constraints, before the complete buoy could be tested at sea. This meant that some aspects where not fully tested. That said the project outcome provides a solid platform for further development and testing. All project development is documented and can be found on `https://github.com/neophytedk/Autonomous-Buoy`.

Contents

1 Introduction

The task given for this project was to improve on the design of the previously made sailbuoy and to add an autonomous mode of operation. The previously made sailbuoy is shown in figure 1. This sailbuoy was constructed by a previous Mechatronics Design and Build Course and was further developed as a master thesis project [5].

Figure 1: Previously Developed Sailbuoy [5]

The following improvements were suggested as improvements:

- Mechanical Improvements
  - Increase length of hull
  - New keel design
  - Improvement in the costing
  - Decrease in the weight
- Autonomous Mode
  - Adjust PID-controller values for water environment

Use existing algorithm for internal weight

- Operation time

  Solar panels/ wind turbine attached to hull

In order to achieve these tasks the overall group split into sub-groups for mechanical, control, hardware and software. The work done and results achieved are therefore presented in the following chapters for each sub-group.

2 Mechanical Parts

2.1 Sailbuoy Design

Design of the hull The design of the hull for autonomous sail buoy is based on the earlier build sail buoy from L. Frommhold and J. Hagemann [5]. Several interviews with both authors provides profound insights about weaknesses and strengths of their sail buoy. The two main weaknesses are first of all that the ration between bow to stern and port to starboard is too little and second the weight is too high for a boat this size. Moreover, they had issues with the water-tightness at the keel of the boat. The missing length between bow and stern cause tremendous instability, when waves hit the buoy. Furthermore, the missing length makes it almost impossible to steer on one course. The too heavy buoy also causes steering problems, as the mass-moving mechanism does not have such a big effect. All these problems should be solved by the new hull for the autonomous sail buoy. In doing so, the new hull was planned with a new construction concept that facilitates a lighter hull body. With this concept, the body is mainly built out of fiber reinforced plastics, whereby glass fiber is used to reinforce the material. For shaping a hull with liquid polyester and chopped strand map a mold is mandatory. Since the form of the previous sail buoy has shown no drawbacks in its shape except its length, it was used as mold for the bow and middle part of the built sail buoy. The remaining part, including stern, was built with a special designed mold. The design of the constructed mold is a combination of an extension of the previous used hull shape and a hull shape for oversea sailing boats that can deal with big waves, since the initial sailing boat had problems with waves. The mold was build according to worth wile hull development with ribs. All ribs were mounted on a wooden rod, which was fixed and aligned to the old hull.The built mold can be seen in figure 2.

Figure 2: Built mold with previous sailing buoy and extension ribs

Afterwards, the spaces between the ribs were filled by covering the constructed mold with fabric. The old boat, which serves for a mold, was covered with plastic film to protect it and to make sure it will not stick to the new hull. After the first run, the old boat hull was carefully taken out from the printout. The back part of the mold made out of ribs was kept in, as it is used as inner frame for force distribution. The area where the keel from the old buoy was, was filled with fiber reinforced plastics afterwards. The next step for finishing the hull was building an inner wooden frame for the bow. The frame was built with several ribs. The inner frame was finished by connecting all ribs to a wooden plate, which serves as deck. After the latter process, the hull was ready to get more layers of polyester and glass fiber. Overall, the new designed hull reach a length of around 200 cm.

A further idea for the hull was to provide a sailing buoy which can be customized for different test set-ups and environmental conditions. That implies variable positions of the keel and the sail. The idea to move the sail to different positions was solved by two rails on the deck in which the mast can be slided or fixed. This feature requires a tremendous solid deck in order to ensure a save mast fixation at any position. In doing so, an additional frame was built inside the boat. The frame is made out of 20x20 Item aluminum profiles and it is fixed at the inner side of the port, starboard and deck. The two outer rails are fixed with inner frame in a sandwich-constructions. This means both frames laying perfectly align above and they are screwed together, in-between the wooden plate from the deck is pressed together from both aluminum frames. This kind of construction provides an ideally force distribution from the mast over the frame and deck into the hull of the boat. All connections within the frame are screwed. The back part of the inner frame is also utilized as opening of the deck and the lid is fixed on the frame, as well. The lid is also built out of 20x20 Item aluminum profiles with two hinges. For maximizing the customization, the keel is also movable. The attachment is similar to the construction of the mast, as the keel is also mounted on an aluminum profile. The keel is screwed on the profile and by loosening the screws the keel can be slide forth and back. The attachment of the aluminum profile at the hull was more critical, since it has to be watertight and the hull has no straight surface. Furthermore, the connection has to hold the force from the water plus the force from the 15kg keel bulb. The problem was solved by attaching a bent aluminum plate on the inside of the hull. The plate is fixed with screws and four threaded holes to the outer profile. The hull is again pressed between both components. The water tightness is ensured with triple protection. The first protection is made with several layers of fiber reinforced plastics around the outer profile, which avoids that water can reach the threads. The second protection is silicon inside the threads and around all components nearby. Both sealing methods do not allow to disassemble the outer profile from the hull, since otherwise water tightness cannot be ensured. The third protection is that the keel is no water tight part of the hull. This means a leakage of the keel will not affect the water tightness of the hull. The inner aluminum plate is bent according to the shape of the inside of the hull. The tight fit of components facilitates a good force distribution. The hull got some extra layers of fiber reinforced plastics at this part of the keel to make sure the hull can forward the applied force to the rest of the hull.

The aluminum plate also serves as inside and outside point of intersection of forces at the boat. The outside forces are mainly transferred through the keel. The inner forces are mainly created by the mass-moving mechanism. For this reason, the mass-moving mechanism is directly mounted on the bent aluminum plate. Thereby, the generated moment from the mass-moving can directly transferred into the hull and into the keel. The mass-moving mechanism is at-

tached on the aluminum plate with screws and 4 angles. In any case, it always can be quickly disassembled. The fixed aluminum plate with assembled mass-moving mechanism can be seen in figure 3.

Figure 3: Installed aluminum plate with assembled mass-moving mechanism

The control box is mounted in the bow of the boat under the sail. It is horizontal screwed on the ribs of the front part of the boat. The height of the boat intentionally decreases to the stern of the boat. The idea behind it is to keep the wind attacked surface as low as possible in order to decrease the wind drift.

2.2 Hull, Keel and Sail Calculations

In order to complete the design of the sailbuoy a number of calculations needed to be done. These calculations include:

- Centre of Effort of the Sail
- Center of lateral resistance of the Hull
- Calculation of the Lead
- Calculation of Keel Weight and Length

These calculations would allow for the correct positioning of the mast and keel, and also the calculation of the keel weight and keel length. The overall length of the hull of 2 metres had

already been decided early in the project after consulting with the previous years sailbuoy designers.

Calculation of the centre of effort of the sail (CE)

The calculation of the CE is done by calculating the geometric centre of effort. This is achieved by taking the intersection of the lines between the mid point of each side and the opposing angle as seen in figure 4 below [1]

Figure 4: Calculation of Centre of Effort

This gave the result that the CE was 302 mm from the mast.

Center of lateral resistance of the Hull (CLR)

The CLR was also calculated geometrical using a 2-dimensional cut-out of the hull and keel. The keel CLR should be placed in the front third of the keel and therefore several cut-outs where constructed until a CLR in the correct place was achieved. One cut-out example is shown in figure 5 below.

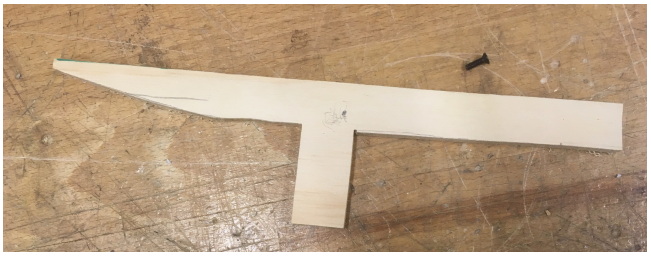

Figure 5: Calculation of Centre of Lateral Resistance

The calculation led to the keel centre being placed at 1 metre from the stern of the sailbuoy.

Calculation of the Lead

The lead is the distance from the centre of effort of the sail (CE) to the centre of lateral resistance and is shown in 6 below. Our chosen design incorporated both the possibility of moving the mast position and the keel position. However, it was still required to have a position upon where to mount the profiles where the sail and keel could be adjusted around.

Figure 6: Calculation of Lead [1]

There is no exact recommended position for the length of the lead. However, a distance of 5%

to 9% of the hull length is recommended [1]. This gives a lead distance of between 100mm and 180mm. The average of this (140mm) added to the already calculated distance from the CE of the sail to the mast (302mm) gives a total distance of 442mm from the mast to the CLR.

As the sailbuoy allows the sail and keel to be adjusted, the centre of the profiles for mounting the keel and mast were placed at approximately 450mm apart. This would allow for changes in the lead from 5% to 9%.

Calculation of Keel Weight and Length

In order to save on costs it was planned to reuse one of the keel weights that was left over from the previous sailbuoy project. These keels, weighing 8kg and 15kg respectively, were then calculated using the formula:

$$b = \frac{F_{4bft}\, S \cos(\alpha) \cos(\beta)\; a}{G \sin(\alpha)}$$

[2]

where:

b = length from the keel weight to the waterline m^2
F_{4bft} = $16N/M^2$ (4 Bft or moderate breeze)
S = Sail area m^2
α = righting angle 30^o (commonly used for sailboats)
β = sail angle beam reach 90^o (sail at maximum)
a = distance CE to water line m (calculated as 252 mm
G = Keel Weight in kg

Using this formula with either the 8kg or 15kg weight, two respective keel lengths were found.

They were for

- 8kg - 1.05 m
- 15kg - 0.55 m

This length is from the waterline to the keel weight and therefore 20 cm must be subtracted leaving keel lengths of

- 0.95 m
- 0.35 m

The length of 0.95 m was judged to be impractical. Therefore the 15kg weight with a keel length of 0.35 m was chosen.

2.3 Manufacturing Process

After the general design of the sail buoy and the calculations of hull, keel and sail, we move into the next step of hull manufacturing. As mentioned before, aiming at transferring the design into a functioning prototype of the sail buoy, the boat should be built with the following requirements:

A. The boat should be in larger size while lighter than before, so that two people can easily

move it;

B. It should also be waterproof enough, to secure the safety and well-running of the electronic and controlling components;

C. The surface should be polished, to have an elegant outlook.

In order to fulfil the needs above, we divide the manufacturing process to be the following procedures: raw material choosing, work flows and safety control.

2.3.1 Main Material Choosing

There are various kinds of raw materials for boat building, such as wood, rubber, steel, aluminum, and so on. Considering the weight-saving parameter, we decided to use Fiber Reinforced Plastic (FRP), rather than the others.

The Fiberglass accounts for most of the fibers used in FRP due to its inexpensive production and relatively good strength to weight characteristics.Aramid fiber is usually used in reinforcing strengthens key areas of sailboats(e.g. bows and keel sections) and provides improved shock absorption. While carbon fiber masts offer major performance and vessel-stability benefits.

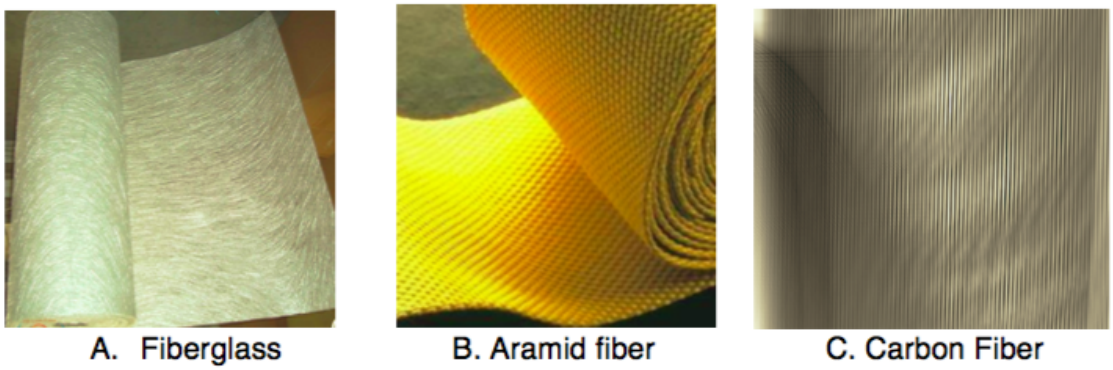

Figure 7: Main Material Choosing

Considering both the cost and strength, we dicide to use fiberglass again as the main material.

2.3.2 Work Flows

Based on the designing idea of the hull, we implemented the following four work flows for manufacturing:

A. Mold Construction

As mentioned in the design of hull, a wooden skeleton was mounted behind the old hull to extend the length of the original hull to 2 meters.

B. Shapping the Hull with Fiberglass and Hardener

After the mold was built, we covered the original hull with some plastic film and fixed fabrics on the outside of the wooden skeleton as a protection. Later, we start shapping the hull with fiberglass and hardener to reinforce to FRP. The weather was OK in Early November and the first layer of FRP dried quickly. The effect turned out to be ideal and we took out the original hull to continue with the second layer, to make it much strengthened. Moreover, we put more materials to fulfill the hole for the fin. At the same time, we have to grind the surface with sandpaper, so as to have a smooth surface for later coloring.

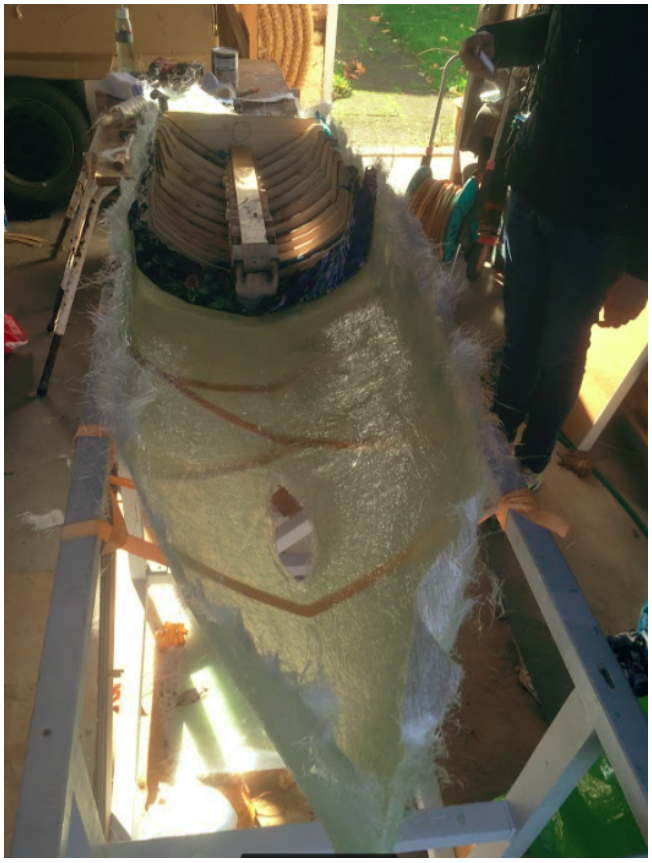

Figure 8: Shapping the Hull with Fiberglass and Hardener

C. Add Putty and Grinding

It was already late November, the weather in Denmark was cold, windy and rainy, when we found that the surface of the FRP was not so smooth. There were some air holes and after grinding, through which water can come inside the boat. Therefore, we began to add another putty on the outside of the FRP.

Figure 9: Putty

This putty is generally used in refinishing small damages of cars, which make the paint work fast and effective. The putty also performs quite well in case of waterproofness. Moreover,

the putty's color looked nice and saved our work for painting. However, we had to grind a lot before it turned out its smooth and perfect color. This work was rather tough due to the bad weather and shabby working environment. Anyway, we managed to finish it in December.

Figure 10: Grinding the Boat

D. Set up the Keel, Sail and the Electronic Components

After the hull was built, we managed to transport it to the workshop at SDU, there we mounted the keel, sail and other electronic components with assistance of the aluminum profiles (See the design of hull).

E. Sealing and Test.

Before the overall test of the autonomous buoy, we decided to have a first water test without the electronic components, just in case of jeopardize the control system. First, we sealed the lid with adhesive thermoplastic gasket, which is super waterproof. Then we made an easy lock for the lid, in case the huge waves blow and open the lid.

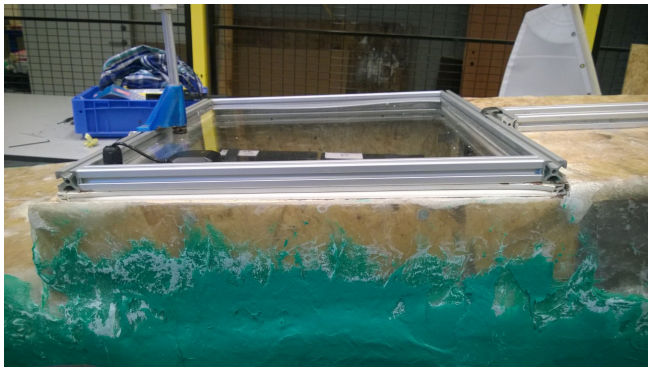

Figure 11: Sealed lid with Lock

After all these preparation, we tested the boat in the sea. Finally, the boat floated well without even a drop of water coming inside. Which means, the mechanical part was fully finished.

Figure 12: Test the Water Resistance of the Boat

2.3.3 Safety Control during Manufacturing and Test

During the boat manufacturing process, there are some substances which if not properly controlled can have severe effects on both human health and surround environment. Here are some safety measures in our project:

A. Safety in Manufacturing

During the building work, we use some hazardous Chemicals, such as "Spartelmasse med glastråde", a kind of polyester glue. We have to wear some facial tasks and gloves to avoid inhaling harmful air and direct skin contact. However, with considering the limited buget and long-time work, these are the best options we can have.

On the other hand, the proper use of tools is of much importance. We have various kinds of tools, some of them must have big power supply and together with safety glasses on. There was one issue during the grinding, one of our group members was injured by the grinding wheel due to the lack of safety baffler. Fortunately, he stopped working right away and took first-aided tools immediately. He was not badly injured, but the issue was still an alarm for us to always remember: Safety is the most important thing during work.

Moreover, we should pay attention to the environment during work, and we did. As we could not work in the workshop at SDU in the beginning, we moved the boat to a private garage. During work, we used a lot of chopped fiberglass mats because the shreds are easy to drop from cutting. We had to clean it right after to keep the environment neat and clean, as the fiberglass is hard to degrade.

B. Safety in Test

In the end of this semester, cold winter in Denmark, we were ready to test the water resistance of the boat. We were extremely careful during the test, especially that it was already dark outside after the day working hours. Well, on the other hand, this also showed our tight time plan. For further tests, we need to carry out a more thorough plan and ensure more facilities for safety sake.

2.4 Limitations and Challenges

The body of the buoy was, as described in the previous chapters, redesigned totally and manufactured by the group of mechanics. Nevertheless, this process was more tedious and complicated as anticipated. This was caused by a number of reasons. This chapter will discuss the

limitations of the building process and what challenges occurred along the progress of the hull.

Workshop
The workshop, which could be used for the building of the hull, was provided by a teacher involved in the buoy projects. The group was allowed to use his garage to work on the hull. It was not possible to laminate, filler and paint the body of the buoy in the workshop of the university, where all necessary tools would have been provided. This was caused by the fumes of the mentioned materials used for building. Since the fumes are toxic and flammable, it is necessary to work in a space, which either has a suction plant or is well ventilated. Even though the university workshop is connected to a suction system, it was not allowed the use the localities, since the suction system is not designed to handle that kind of fumes, which bound the group to work in the garage located at Midtkobbel 10 in Sønderborg, Denmark. Utilizing the garage as a workshop was cause for several problems, which elongated the process of building the hull drastically. The materials that were used by the group need a room temperature of above 15 degrees Celsius or more to harden properly and in an acceptable timeframe. This applies for the resin, which was used to form the glasfiber-hull, the filler, which was used to even out the body and the paint, which should prevent the hull from oxidation due to the elements and especially the salt-water the buoy is designed to operate in.

Polyester and glass fiber
The garage the group was told to use was not heated, due to the time of the year, the temperatures while building were constantly below 15 degrees Celsius and towards the end of the building phase even below zero degrees. This caused that from the moment the first layer of polyester and glass fiber, it showed that the drying-times are a major problem, since new layers just can be added after the first layer was partly hardened. Instead of doing several layers in a day, the group was forced to wait at least several hours to continue the building process. The application of single layers was chosen for the first layers, since the group needed to build a structure and modify it, before the finishing layers could be applied. Due to the in-company periods and talent programs the group members were attending, it was manly possible to work in the afternoon, which meant, that the waiting-times for one layer could easily mean an interruption of 24 hours, before the hull could be modified further. Taking the size of the new design in consideration, it showed that the group had to wait a lot and meet oftentimes to continue the build of the hull.

Filler
The same problem also caused, that the application of the filler was as tedious and complicated as the application of the polyester. In this case, the group was held to order from a shop (www.glasfiber-center.dk) to acquire material for the built. This was caused due to regulations of the university and also was recommended of Kasper Paasch. The filler ("Spartelmasse m/ glastråde tp. Oldopal 740-0588) bought for the hull showed to be not applicable as planned, since it did not dry and remained gooey. Since it was not possible to work on that surface, the group was forced to clean up all the filled-up areas and look for alternatives. Due to the hobby and the relations to other mechanics and painters, one of the group members found an alternative in 2-component glass fiber putty, which is widely used in the car-repair sector. The properties of the material ware sufficient and also it non-hygroscopic, which made the group

decide to buy several tins of that material. The material turned out to work great, but also needed long-drying times because of the cold weather. This made the sanding-process more complicated, because the filler can be easily sanded when it is not hardened out entirely. Due to the lack of experience in building buoys out of glass fiber, the group ended up with a very uneven surface in the back of the hull. Using a lot of filler to even out that surface actually had a good side-effect to the balance of the boat, since it made the back heavier.

Paint
The group already bought anti-fouling paint to protect the hull from the elements. Since the build of the hull took much longer than expected, it needed to get transported to the university workshop to get equipped with the electronics. Once the boat arrives there it means for the group that the option for using paint was over, since it was not allowed to be painted or dried inside of the building. Conversations with the workshop-leader Reiner Hübel and the contacts that recommended the car-filler, made the group decide to not apply paint to the hull yet. Applying paint at that temperatures and high air-humidity, means that the paint will not dry properly and could cause that the paint is not sticking to the hull and will basically remain in a gooey consistency which makes the hull unable to be handled and carried manually. Reiner Hübel and the contacts were convinced, even though some group members had concerns of loss glass fibers transferring water to the inside of the hull, that the glass fiber hull and the putty are watertight and that testing should be possible. To make sure that the electronics did not get damaged, the group decided to test the hull for water-leaks (see previous chapters). The paint could be returned and removed from the budget.

Health and safety
The group was aware of the toxicity and the danger that comes with using glass-fiber, polyester and filler. The group members tried to protect themselves by wearing safety-glasses, gloves and dust-masks. As written in the previous parts of the chapter, it is necessary to work in a place, which either is connected to a suction plant or is well ventilated. Since the budget for the hull was very limited, the group decided on buying dust-masks, which costs just a little fraction of the respirators, which actually would have been needed in a project of that size and with that amount of material applied. It occurred that group members needed to take breaks during the process, since the fumes caused nausea and a light-headed feeling. It cannot be acceptable that low budgets cause to save money on safety.

Transport of the buoy
As mentioned before, the location of the hull was giving what the group members were able to do to it. Transporting the buoy in the beginning meant that a station-car was enough to load the hull, transport it and unload it. In the end of the project it got more complicated to work on the heavy hull and transport it from one place to another. Giving the fact that most of the processes for building the hull could just be executed while the hull was placed in the garage, meant that the transport of the hull grew to a key factor for the progress made. Most of the time the hull was located in the garage and was transported to university by the end of the time-frame when testing was needed. Also sanding the excess filler was not allowed in university because of the dust, which spreads into every little corner of the localities, since the the tools provided were not able to the job properly and suck away the dust. Transporting the

hull back to the garage and to university again in the beginning of January was not an option, since it was not clear who can be assigned for the transport. This is mainly caused why no member of the group either has a car, which is big enough, or has a trailer-hitch on his car.

In the end it showed that the group was able to overcome most of the challenges that occurred during the process, even though it meant that the current prototype also looks like a prototype and is not painted and sanded finally yet. Nevertheless, the hull lives up to the expectations made, since it can be broken down the essential requirements:

- Able to swim
- Balanced in the water
- Water-tight and rigid
- Hold all electronics necessary (including battery-sled)
- Mounting points for sensors
- Work with old sail
- Equipped with new designed keel
- Assembled for testing

For a future project it will be highly recommended to provide the building-group of the project with better equipment and tools and most important with a place, where the group can work on the project – without having temperature-, ventilation- and transport-problems.

3 Modeling and Control

This section covers the components of the controller and presses of developing it. Starting from the basic principals and working up to a finalized controller. The goal is to develop a general controller for any system based on the same principals as the buoy in this project.

3.1 Basics of a sailing vehicle

The behavior of a sailing buoy is quite complex while compared to a conventional ship, since one has to take into account of the wind direction in order to reach a particular way point.This is this simple fundamental fact that makes considerably more difficult to steer than a conventional ship. A simple illustration of this is the so-called no-go zone or dead-zone, which is the interval of headings closer to the wind direction, for which a sailing buoy looses propulsion completely[4]. Any vehicle can be considered to have 6 degrees of freedom, its position along each spacial axis (x, y and z) and the rotating around each axis (pitch, roll and yaw), however some systems can be simplified to fewer degrees of freedom for control purposes.
The buoy system can be reduced to 4 degrees of freedom as both height (z axis) and pitch (rotation around the y axis of the buoy) can be safely ignored under normal operating conditions as their effect on the behavior of the system is minimal. The system has a single input, mass position, where most other marine vehicles have 2 or more inputs, rudder, sail positions or propeller speeds.
The 4 degree of freedom model includes the surge,sway,yaw and roll motions. In order to design a control system for roll damping, it is necessary to add the roll equation to the model. Inclusion of roll means that the restoring moment due to buoyancy and gravity must be included. The resulting model is a 4 degree of freedom maneuvering model that includes roll (surge, sway, roll and yaw).

3.2 4 DOF Model

The Fossen's notation is widely used in the industry for conventional ships, and which allows for a simple and compact notation useful in modelling and control purposes. Hence, a sailing buoy is represented by the following expression

$$M\dot{\nu} + N(u_0)\nu + G\eta = \tau \tag{1}$$

where u_o =constant, ν =$[u, v, p, r]^T$ and η =$[x, y, \phi, \psi]^T$ are the states while τ is the control vector[3].

$$M = \begin{bmatrix} m - Y_v & -mz_g - Y_p & mx_g - Y_r \\ -mz_g - K_v & I_x - K_p & -I_{xz} - K_r \\ mx_g - N_v & -I_{xz} - N_p & I_z - N_r \end{bmatrix}$$

Here M=M_{RB}+M_A, where M_{RB}=Rigid body inertia matrix, M_A=Hydrodynamic system inertia matrix.
Here X = Force in x-direction(surge)

Y=Force in y-direction(sway)
K=Moment about x-axis(roll)
N=Moment about z-axis(yaw)
u=Linear velocity motion in x-direction(surge)
v=Linear velocity motion in y-direction(sway)
p=Angular velocity about x-axis(roll)
r=Angular velocity about z-axis(yaw)
x=Position in x-direction(surge)
y=Position in y-direction(sway)
ϕ=Euler angle rotation about x-axis(roll)
ψ=Euler angle rotation about z-axis(yaw)
The expression for $N(u_o)$ is obtained by linearization of C(ν) and D(ν) about u = u_0 which gives

$$N(u_0) = \begin{bmatrix} -Y_v & -Y_p & mu_0 - Y_r \\ -K_v & -K_p & -mz_g u_0 - K_r \\ -N_v & -N_p & mx_g u_0 - N_r \end{bmatrix}$$

where C(ν) is the system Coriolis-centripetal matrix and D(ν) is the damping matrix.The linear restoring forces and moments for a surface vessel (buoy) can be written as,

$$G = diag\left\{0, \quad WG\bar{M}_T, \quad 0\right\} \tag{2}$$

where W = mg is the weight and $G\bar{M}_T$ is the transverse metacenter height.
In addition to these equations, the kinematic equations (assuming q = θ =0) are added to system model.

$$\dot{\phi} = p \tag{3}$$

$$\dot{\psi} = cos(\phi)r \approx r \tag{4}$$

The linearized model(1) together with (3) and (4) can be written in state-space form by defining the state vector as x =$[v, p, r, \phi, \psi]^T$. The elements associated with the matrices A and B are given by

$$\dot{x} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & 0 \\ a_{21} & a_{22} & a_{23} & a_{24} & 0 \\ a_{31} & a_{32} & a_{33} & a_{34} & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} b_{11} & b_{12}..... & b_{1r} \\ b_{21} & b_{22}..... & b_{2r} \\ b_{31} & b_{32}..... & b_{3r} \\ 0 & 0..... & 0 \\ 0 & 0..... & 0 \end{bmatrix} u \tag{5}$$

where the elements a_{ij} are found from

$$-M^{-1}N(u_0) = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{6}$$

$$-M^{-1}G = \begin{bmatrix} a_{14} \\ a_{24} \\ a_{34} \end{bmatrix} \tag{7}$$

while the elements bij depend on what type of actuators are in use.

3.3 Control basis

The controller will be made to get the system to set spacial coordinates, the chosen approach is to use control allocation that split the system in to its individual modules working from the first module that takes in way points and down to the module that sets the position of the mass.
The modules got split up as follows.

- Navigation
- Course/Heading selection
- Course/Heading control
- Tilt control

Not all of the modules were implemented in the project but they will all be explained here. The reason for splitting the controller in to these modules is so that each module is only affecting a singe portion of the system at a time.
The reason for this approach is that there does not exist a mathematical model of the buoy and constructing such a model is complicated as it would be primarily based in fluid dynamics, which is a field where none of the group members have any experience with. However a mathematical model of a sailing boat made by Jerome Jouffroy whitch is described in his paper on Autonomous Sailing Buoys[4] and has been published in MathWorks community [8] which will be used for verification.

3.3.1 Tilt control

This module sets the position of the control mass, for that it uses the measured tilt (pitch) of the buoy and its desired tilt, and the error between measured and desired tilts is then feed into a gain controller. The gain controller implied here is a simple P-gain controller, just a simple P controller turned out to be sufficient as simulations comparing use of P, PI and PID controllers showed minimal improvements as the steady state error that is characteristic of the P-gain controller is compensated for in the course/heading control module
The effect gotten from controlling the tilt is that the tilt of a sail buoy with no external moving parts on the hull is change its heading.

3.3.2 Course/Heading control

This module outputs the desired tilt to the tilt control, for that it takes the proceed course/heading and desired course and as before calculating the error and feeding it into a gain controller. This

gain controller is a PI-gain controller, it turned out to be the only realistic choice between the P, PI and PID. P-gain controller could be tuned to have minimal steady state error but that would only remain true as long as conditions on the water stay the same as when the tuning took place. On the other hand a PID-gain controller is difficult to stabilize as the fast reaction gotten from the D-gain seams to cause instability.

3.3.3 Course/Heading selection

This module outputs the proceed course/heading to the course/heading controller, it takes in the measured heading, the measured course and the measures speed. A weighted average is then taken of the course and heading with the speed of the buoy determining the the weight. The two weights are set up so there sum is always 1, when the speed is 0 the heading weight is 1 and then ramps down to 0 as speed increases, the slope of that ramp need some test data to be determined as it needs to some portion of the top speed.
The reason for including this module is sensor noise and small movements caused by the water, these cause the measured course to become inaccurate.

3.3.4 Navigation

This is the module whit the most potential as it could contain everything from course planing to obstacle or hazard zone avoidance. In this case the focus is simply point to point navigation, needed for that is the destination coordinates, current coordinates, wind direction in reference to the boat and a table for wind direction/speed compared to the boat speed (angle of attack, wind speed and buoy speed). The table is needed to find the dead zone of the buoy, where the does not catch any wind, this has no effect while sailing down wind. However when sailing up wind the boat needs a certain angle of attack (angle between heading and wind direction) to retain any forward motion, the specific minimum angle of attack for the buoy depends an many factors with the keel and sail being the two biggest factors. The buoy will needs to sail back and forth to gain an upwind way-point and this has two significant points that need to be taken into account, what are the most efficient angles of attack and how far from the straight line course do we want the buoy to go. The most effective course can be determined with parameter minimization and the turning can be done by adding wights into the parameter minimization but this is all dependent having the aforementioned table of test data.

3.4 Tuning process

Tuning of the controller made done a single module at a time and since it is designed as a generic controller it is not practical to tune done with anything other than trial and error on the physical system. The specific method can be any PID-tuning method but some care has to be taken as these methods assume a linear system and might therefore not have the expected results.

3.4.1 Tuning tilt controller

A simple way of tuning this P-gain controller is to start with the P-gain low and increase it until the system starts oscillating and then reducing the gain by half as by Ziegler–Nichols method of tuning P-gain controller. To make sure about the quality of the tuning this should be repeated for several different desired tilts as the non-linearity might still lead the system to not settle for a given tilt.

3.4.2 Tuning course/heading controller

With the tilt controller tuned it is now possible to tune the PI-gain controller in the course/heading controller, it has to be tuned while sailing down wind and else using the same approach as the for the tilt controller. The raw heading and a manually applied desired course/heading are used as inputs.

3.5 Conclusion

The general goals for designing a generic controller for a buoy with no externally actuated components was successful even though the lack of test data on the physical system halted further progress.
The first improvements would be to see how best to implement the course/heading selection, as using the course is preferred so to find the limits of the system for course accuracy are concerned.
The next and bigger task would be to implement the parameter minimization efficiently on the system hardware so that it can run along with everything else. The main task in this is to collect test data to make this possible.

4 Hardware

This chapter covers the details of the design and implementation for the hardware, which can all be found on: `https://github.com/neophytedk/Autonomous-Buoy`.

4.1 Overall design

As shown in figure 13, a PC application is to be running while controlling the buoy or getting sensor data from it, so a pair of RF modules are used, one is connected to PC while another one is to microprocessor. The power supply module is to power all the sensors and the microprocessor. Wind sensor is to get wind direction and speed information, the communication type is RS422, leading to the need of a UART-RS422 module so that microprocessor can communicate with it using UART. The IMU is to get the inertial information of the buoy, and the GPS is to get the position. There is a motor in the center of the buoy, with shifting the weight on it to left or right, the buoy can be tipped. In the motor controller module, the H-bridge is to drive the motor, the limit switch is to detect if the weight has reached the left side or right, and the encoder is to get the position of the weight.

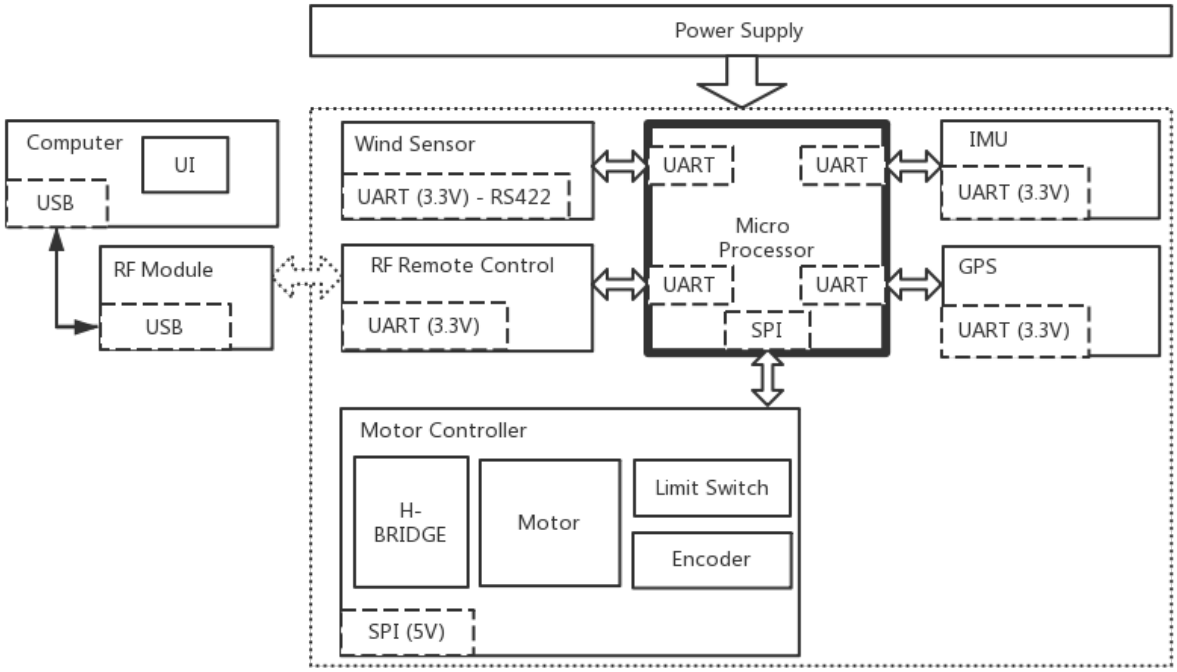

Figure 13: Structure diagram

4.2 Detailed design and implementation

4.2.1 Microprocessor

The reason that we choose Nucleo-144 instead of others is mainly due to the consideration of software development, which will be addressed later in the software part.

It has on-board ST-LINK/V2-1 debugger/programmer with USB re-enumeration capability, which makes it way easier to develop the software and also debug it. For final design, we supply it with 5V since it has a LDO on board converting the 5V to 3.3V.

4.2.2 Remote control

For remote communication, we choose a X-Bee pair from the Digi International, Inc. These modules are ideal for applications requiring low latency and predictable communication timing. Providing quick and robust communication in point-to-point applications. With a breakout board for this module, we only need to supply with 5V since it has a LDO on it converting the 5V to 3.3V, which is the voltage level needed for X-Bee modules.

4.2.3 Sensors

To get the absolute position of the buoy, a GPS module is needed. The one used here is the NEO-7P from the company U-BLOX. The breakout board for this module is also readily available, and it is supplied with 3.3V.

To get the motion of the buoy, an IMU sensor is used. It is a student version of the module TransducerM from the company Syd Dynamics with output rate 300Hz. One thing to be clear, the supply voltage of this module is 5V while the voltage of the UART port is 3.3V.

To get the dynamics of the environment, a wind sensor is used. We are using an ultrasonic wind sensor from the company LCJ Capteurs. The supply voltage of this sensor ranges from 8V to 30V DC. With a sampling rate of 30Hz, it has an output rate of 2Hz, which gives more accurate results. Since the communication type of this sensor is RS422, which cannot be connected directly to microprocessor. An RS422 to UART module is to be used as figure 14.

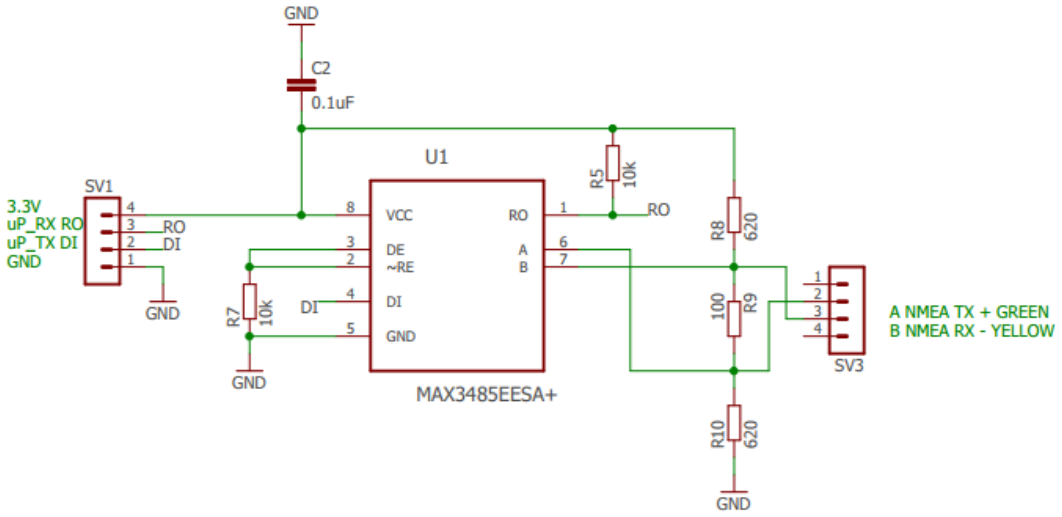

Figure 14: RS422 to UART

This chip is not fully duplex. We need to set or reset the enabling pins DE and RE to select either to receive data or send. Since we only need to read from the wind sensor, instead of using a GPIO pin to control the enabling pins, we pull them down to GND so that it is always receiving data.

4.2.4 Motor controller

To drive the motor, an H-Bridge chip is needed. The one chosen here is LMD18200, which is a 3A H-Bridge designed for motion control applications. It is ideal for driving DC motors and it accommodates peak output currents up to 6A. To get a relatively accurate position of the weight, we mount an encoder on the shaft of the motor. It is a Wisamic 600p/r incremental rotary encoder with a 6mm shaft, whose power supply ranges from 5V to 24V DC. To lower the coupling of each module, an Atmega328 chip is used to control the hardware so that only commands are needed from the central microprocessor. Figure 15 shows the schematic drawing of this module.

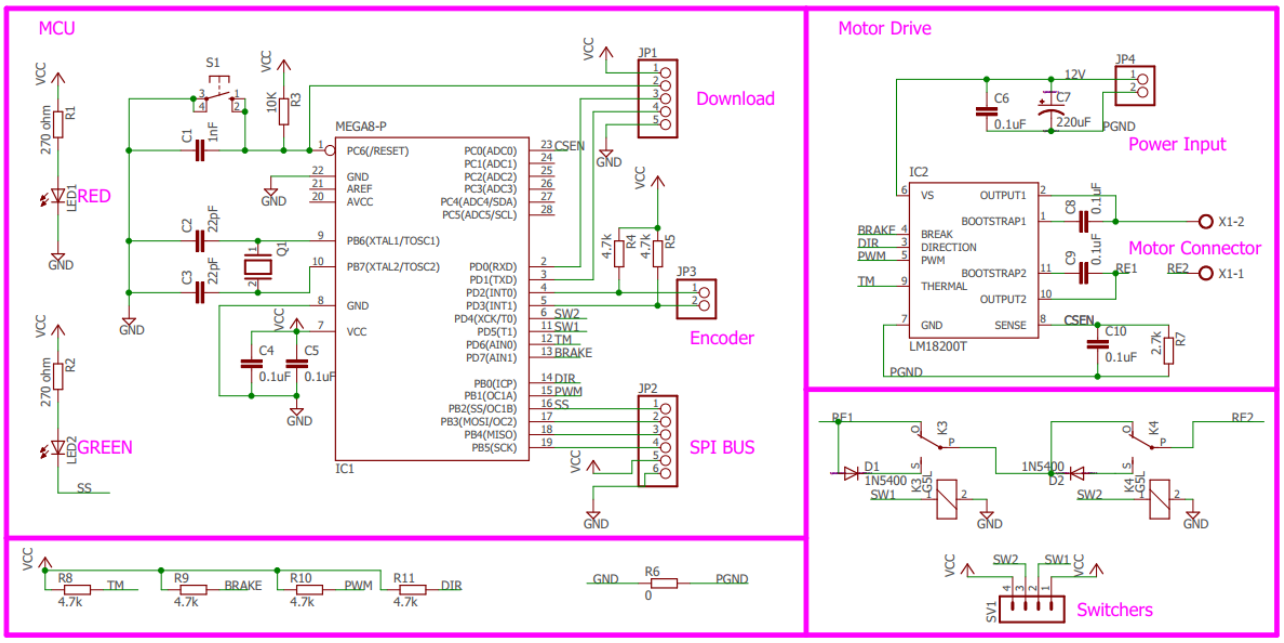

Figure 15: Motor controller

In the MCU part, an LED is used for indicating that it is powered on while another one is for showing the SPI bus is working. The Download header is for updating the program inside this chip, the SPI BUS header is to communicate with the central microprocessor, and the Encoder header is to be connected with the A/B wires of the aforementioned encoder. The outputs of the H-Bridge are not directly connected with motor. With a combination of relays, diodes and the limit switches, the motor can be stopped by hardware. The idea is, normally, the weight does not hit any of the limit switches, so, both relays are in off state so that the current can pass both ways. Once the weight hits a switch, for example SW1, then the relay 1 is on so that the current can only go in opposite direction because of the diode. Additionally, pull-up resistors are needed for driving the H-Bridge.

When in practice, we met a problem with the noise from the motor, which occasionally led to the reset of this Atmega328 chip. So, we found a motor filter consisting of 3 capacitors and 2 inductors, as shown in figure 16.

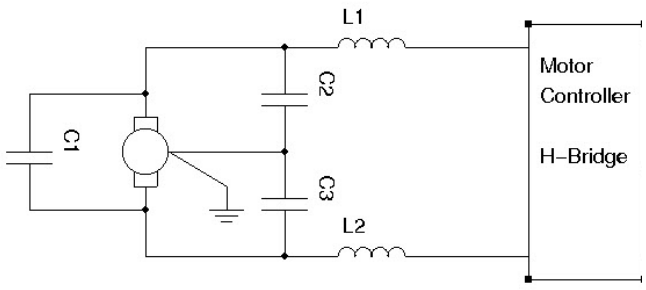

Figure 16: Filter

The capacitors are of ceramic kind, with the value 0.1uF and rated voltage 100V. And the inductors are self-made in one choke with the value of about 70mH. And as recommended, this filter module should be as close as it can be to the motor.

4.2.5 Power supply

The input DC voltages are 12V and 24V. 24V is only used to drive the H-Bridge. Then we need some DC-DC converters and LDOs so that we have 8V for wind sensor and encoder, 5V for microprocessor, motor controller, IMU and RF remote control, and 3.3V for GPS and RS422-UART module. The whole design is shown in figure 17.

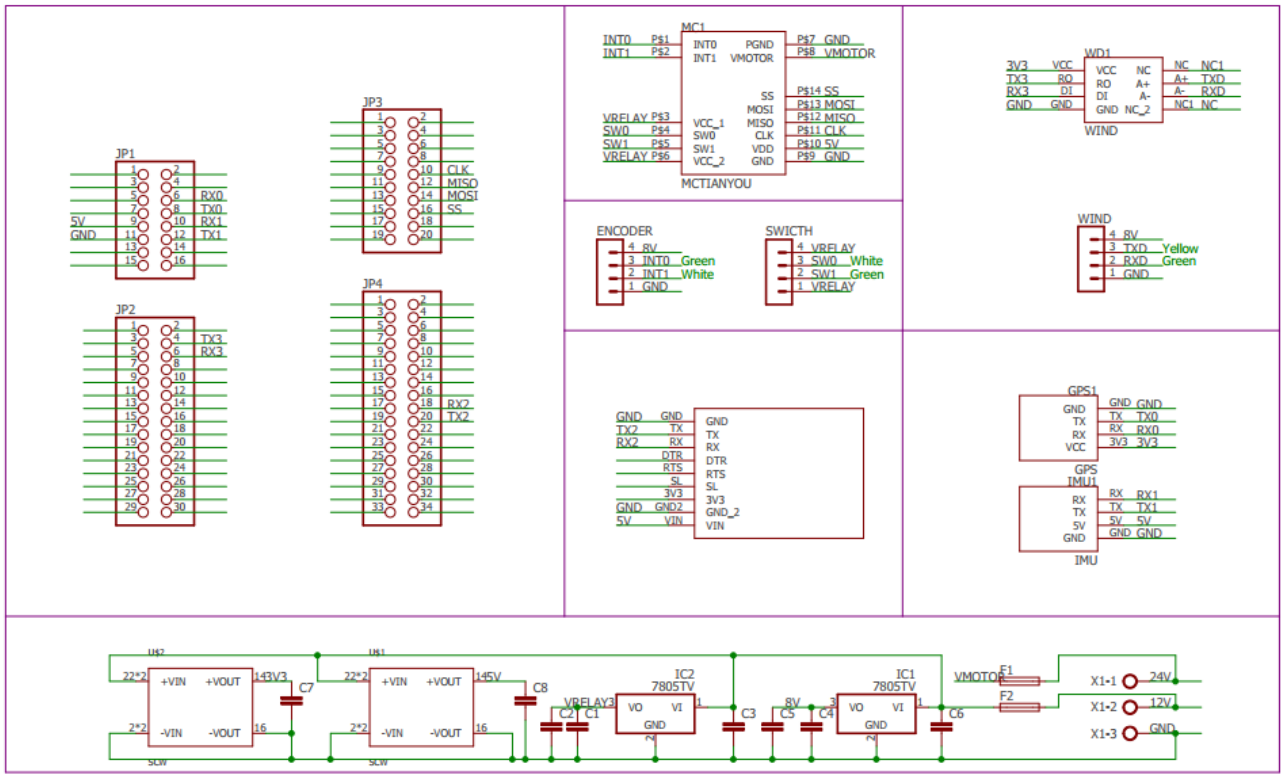

Figure 17: Power supply

After the fuses, which are highly recommended for high current protection, the 24V goes to the H-Bridge, and the 12V goes to 2 DC-DC converters (outputs are 5V and 3.3V) and 2 LDOs (outputs are 8V and 5V). The 5V from the LDO is only for the relays since there is some potential noise from these relays.

4.3 Final implementation

The PCB boards for the final implementation were milled at SDU Sønderborg. To protect the hardware the system where implemented in a IP67 resistant cabinet. This can be seen on Figure 18

Figure 18: Final hardware implementation

4.4 Conclusion

The final hardware implementation realize the functionality wanted for the project. The initial problems inferred by excessive noise in the system where neutralized by adding components to handle the voltage spikes from the relays and the motor.

5 Software

This chapter covers all the details about the embedded software for the Nucleo board and the GUI software for the PC. All the code is included in the github under software:
`https://github.com/neophytedk/Autonomous-Buoy`

Due to the limited time constraints in this project, it was decided early in the project phase, that UART where to be used instead of the I2C bus to communicate with the sensors. I2C was tried, but proved to be more complex than expected. UART was rather simple to implement, and the numerous UART ports of the Nucleo allowed us to utilize it for all of the sensors.

5.1 Board choices

In the start of the project the correct hardware platform had to be chosen. In order to chose the one which meet the requirements the best, a table with pros and cons was made. Table 1 and 2 contains candidates which have been assessed as being the most suitable ones.

	Rpi 3	**BBBlack**	**BBBlue**
I2C	yes	yes	yes
PCB Plugin	40-pin header	2x46-pin header	4-pin ribbon connectors
Supply	5V, 2,5A	5V, 500mA,	
Community	high	medium	small
Ethernet	yes	yes	yes
Processor	1.2GHz, 4 core ARM	1GHz AM335x, ARM Cortex-A8	1Ghz ARM Cortex-A8
Memory	1Gb Ram. No onboard mem	512MB Ram,4Gb eMMC flash	512MB Ram,4Gb eMMC flash
Cost	34,99 € (260 kr)	49,99$ (320 kr)	79,99$ (519 kr)

Table 1: Pros/cons for different boards

	Nucleo-144 STM32FXXXYY	**Intel Galileo (gen.2)**
I2C	yes	yes
PCB Plugin	2x 36-pin header	6-pin, 9-pin, 8-pin, 10-pin headers
Power usage		5v, micro-USB
Community	Libraries provided by STM	small
Ethernet	yes	yes
Processor	100 Mhz ARMCortex®-M4	Intel Quark X1000, 400 Mhz
Memory	2MB flash, 256+4 KB of SRAM	256MB Ram MB int. storageMicroSD ext. storage
Cost		55,95 $(350 kr)

Table 2: Pros/cons for different boards

Initially, the Raspberry Pi 3 was the favorite. However, the RPi3 would be an overkill for the application it was to be used for. To get some experience in embedded development, the Nucleo board was the best choice. It has pin-headers which makes it possible to mount on another pcb. STM has a comprehensive set of libraries and also the CubeMX software. It has an ethernet port, which makes the system expandable with web functionality. It has several I2C busses,

UART ports, SPI busses and so on. Lastly, it has a FPU, which is an advantage, for controller calculations.

The Nucleo board stmf429zi has been chosen. It offers the most capability in that series. The datasheet, features, software libraries and so on can be seen here [7].

5.2 Architecture

The architecture for the embedded system running on the Nucleo board is seen in figure 19 below.

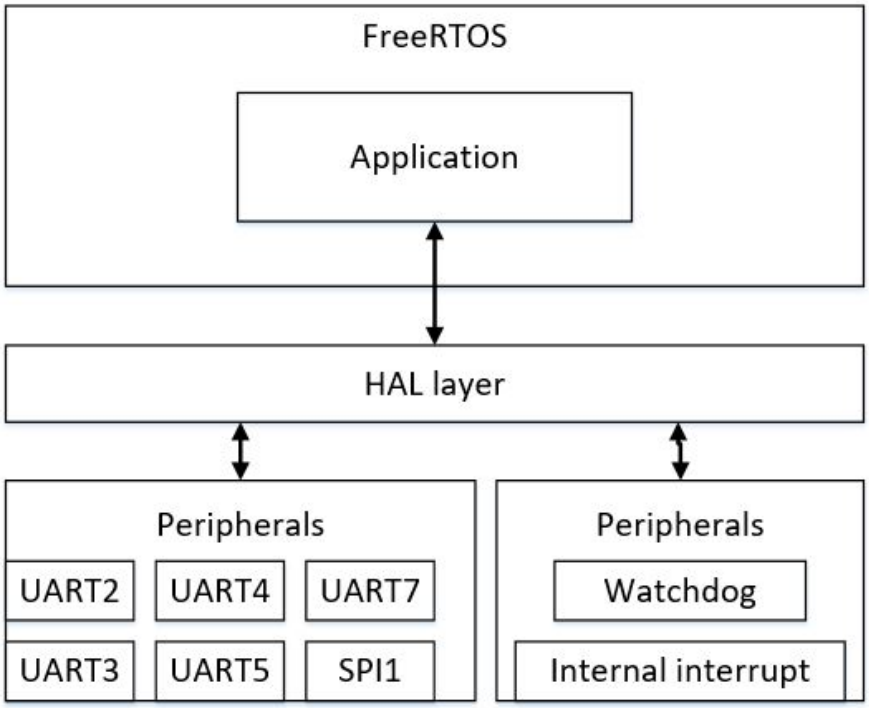

Figure 19: Architecture in the system

This is a classical layered architecture for an embedded software system. The application software, is the one that is made to control the boat. FreeRTOS is used as the operating system where the application is running on top of. The HAL layer provides an interface for communicating with the peripherals on the Nucleo board. The structure of the application is described in section 5.3.

5.2.1 Middlewares

The first thing to notice about the architecture is that is uses middleware libraries. These are FreeRTOS and the Hardware Abstraction Layer (HAL). These middlewares are generated automatically from a program called CubeMX. This is a free program provided by ST electronics. CubeMX is a graphical program that help developers by generating finished code ready to be included in any of the ST developments boards. It shortens the time used for developing significantly.

The basic elements chosen in CubeMX are:

- FreeRTOS
- SPI1
- USART2
- USART3
- UART4
- UART5
- UART7
- IWDG
- SYS: TIM1
- GPIO's

The reason for this specific selection of UART ports is that the pin placement on the Nucleo board made it easier to create the underlying PCB.
The GPIO pins to be configured are pin assignments for UARTs and pins for LEDs on the Nucleo development board. There are three LEDs on the board, and access to these helps debugging greatly.
Setting timer1 as the source for the system is strongly recommend. This will make the system more reliable. See user manual for CubeMX, UM1718, sec. 4.11.6 [6] for more details.
With these settings, CubeMX can new generate a full project for a variety of IDEs. In this project, the 'System workbench for Stm32' have been used. This is an IDE based on Eclipse.

Only FreeRTOS is available as the operating system in CubeMX. Also, the light weight tcp/ip stack (lwip) and the FAT file system can be included. The two latter ones are not used in this project though. They do however make the system ready for adding extra functionality without bigger hassles.

5.3 Application

The structure of the application running on top of FreeRTOS, looks like the one in figure 20.

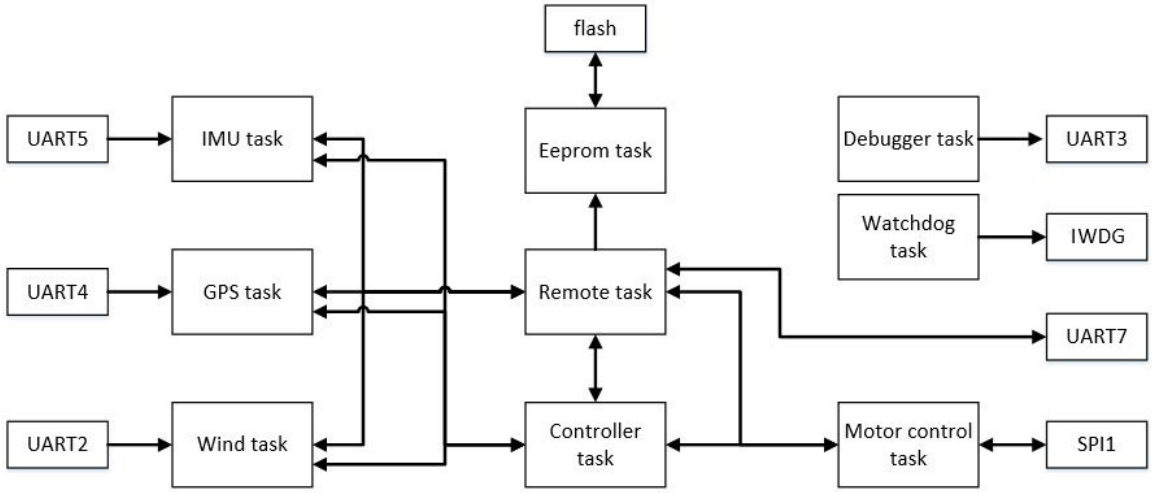

Figure 20: Communication in tasks between and to the hardware

Above is showed how the different tasks communicate with each other internally and with the corresponding hardware. Each task in the system has its own job to take care of and distributing data through the system and/or provide interfaces to other tasks.

In order to maintain data integrity, the task communicates with each other using synchronization mechanisms. These are provided by the FreeRTOS API and are e.g. queues, mutexes or semaphores and so on. Most of the tasks implements simple state machines, to implement their corresponding protocol.

Each of the tasks are described in details in the following sections.

5.3.1 Remote control

The remote control task is one the central tasks for configuring and handling operations on the boat. The structure of the remote task is seen on figure 21 below.

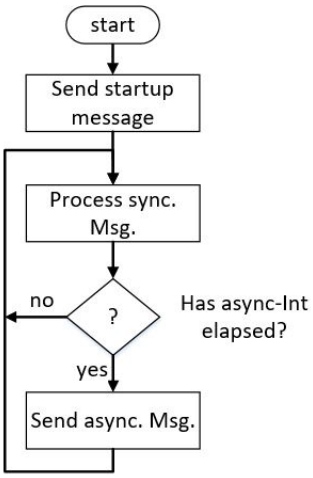

Figure 21: Flow within the remote task

When the whole system for the boat starts up, a message is sent to host, as an inform that it is now online and ready. After this, a loop is entered which handles requests from the host, and sends asynchronous messages to the host with a specific interval. This is done so the host does not have to poll it every time, hence it lowers the data link preoccupation.

For the purpose of communicate between the boat and the host, a protocol has been made, to be used when communicating between the boat and the host PC.
The protocol has a fixed package length of 8 bytes, as shown in table 3.

Byte	0	1	2	3	4	5	6	7
Type	SOT	Cmd	Id	d0	d1	d2	d3	crc

Table 3: Protocol for remote communication

Whenever a request is sent from the host, the boat always replies with a package of 8 bytes back.
SOT is the start of transmission. It is hex 96, and is used for synchronization.
Command is a bitmask, describing the type of message as seen in table 4.

Bit	7	6	5	4	3	2	1	0
Type	set/get	async msg	sync reply	-	-	-	-	-

Table 4: Bitmask for command byte

The async msg means that it is a message sent asynchronously from the boat to the host (without a prior request from the host).
The sync reply means it is a reply from the boat to a request that the host made.

The ID specifies which 'register' to get or set on the boat. It can be one of those listed in table 5.

Data byte 0-3 is the payload of the message.
CRC is the checksum of the message. It helps keeping the data integrity of the package. It is calculated from XOR'ing bytes 0 to 6.

Id	Description	Data type
1	Error	byte[4]
2	Status	byte[4]
3	Configuration	bool[32]
10	Wind speed	float
11	Wind direction	float
12	Wind temperature	float
20	GPS longitude	float
21	GPS latitude	float
22	GPS speed over ground	float
23	GPS course over ground	float
30	IMU roll	float
31	IMU pitch	float
32	IMU yaw	float
40	Motor control P	float
41	Motor control I	float
42	Motor control D	float
43	Motor position	float
50	Heading control, gain1	float
51	Heading control, gain2	float
52	Heading control, gain3	float
53	Setpoint longitude	float
54	Setpoint latitude	float
55	calibration value	float

Table 5: Command id's and their datatypes

The config id is a boolean array of 32. It is used to put the boat in different operational modes, as shown in table 6. The bits not included, are not used but makes it possible for expanding with extra functionality.

Bit	Description
24	Manual operation
25	Automatic operation
16	Controller calibration on
17	Controller calibration off
18	Start motor calibration
19	Calibrate, set north

Table 6: Bits in config id

5.3.2 Controller

The controller is getting the values from the sensor task that it needs. It also has the setpoint values, it has gotten from the remote task. With those values, it calculates the setpoint for the mass, which is then send to the motor control task. For details on the controller math, refer to section 3.
A simple state machine, which runs in a loop wraps the calculations so it can be set in either manual or automatic mode.

5.3.3 Debugger

In the start of the phase of the software developing, a debug task was made. This helps significantly in debugging. It provides a structured way for printing debug message to the console, and disable/enable message for each task.

The debugger task writes to the uart3 port, as the only task. This is also the reason for its necessity. If multiple tasks tries to access the uart3 registers by them self and at the same time, it will result in unpredicted behaviour.

The debug task works by providing a queue for other tasks. This actually works as a producer-consumer relation. Every time a task puts an item in the queue, the strings are copied to an internal statically allocated array. It is then popped by the debug task to be sent over the uart3.

5.3.4 Eeprom

User/configuration values sent from the host remotely, must be stored on the Nucleo controller. Otherwise, they are lost after poweroff or after reboot.

Unfortunately, The Nucleo-144 stm32f429zi does not have an on-chip eeprom. Alternatives considered are shown in table 7.

	Cons	**Pros**
Eeprom chip	time requiring implementation	correct/prof. way to do it
SD card	time requiring implementation	lots of storage. Sensor data can be logged
SRAM	Backup-battery required	With battery, RTC can also be used
Flash	Has limited amount of re-writes	Simple implementation

Table 7: Pros and cons for different methods of storing data onboard

The flash storage solution has been chosen. The primary reason is for the simple implementation and given the limited time constraint of the project. CubeMX already created libraries for writing the flash. Values are set relatively few times on the boat, why it has been assessed to be acceptable.

The flash on the stm32f429zi is a 2 MB flash divided into two banks, 1 and 2 which are each divided into sectors of 4x16 KB, 1x6 4KB and 7x128 KB. The program are stored in bank 1 from sector 0. Therefor, the user values are stored in the the last sector in bank 2, sector 23.

To address the problem of limited re-writes, and the fact that a section must be erased before it is re-written, a routine where made which is showed in figure 22 below.

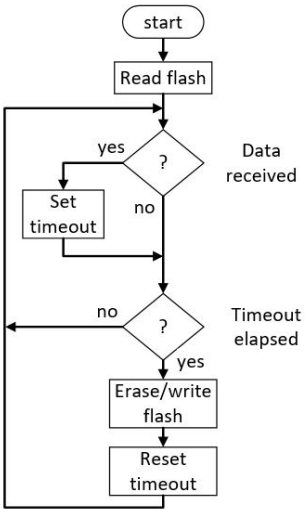

Figure 22: Flow within the eeprom task

5.3.5 Watchdog

As the hardware inside the buoy cannot easily be accessed, a Watchdog service will be implemented to recover from potential software malfunctions. If the down-counting timer is "timed out", the Watchdog will invoke a reboot of the processor. During normal operation a Watchdog task will refresh the Watchdog timer to prevent it from elapsing.

CubeMX offers two Watchdog services, the Independent Watchdog (IWDG) and the Window Watchdog (WWDG). The IWDG, which is independently clocked must be reset before it times out. The WWDG timer must be reset within a time window, if you reset it too early or too late it will cause the processor to reset.

The independent watchdog is clocked separately from the system clock, meaning it will be active even if the main clock drifts or malfunctions. The Windowed Watchdog however can generate a early wakeup interrupt (EWI) just before restarting, where one can have functionality for debugging or having the system put in a safe state.

We choose to implement the Independent Watchdog in our project, as it was the simplest one. Having a Windowed Watchdog with an early interupt wakeup, would be useful, however it was discarded because of time pressure. Choosing one of the watchdog services will resolve malfunctions, however one should use both if you can.

Independent Watchdog
The Nucleo's independent watchdog uses a 12-bit free-running down counter, which is independently clocked by the Nucleo's 32 kHz low-speed internal RC oscillator (LSI).

During initialization one must specify the 12-bit down counter reload value, and the pre-scaler.

The IWDG reload value and the pre-scaler decides how quickly the Watchdog timer will "time out". The relation is described using:

$$\frac{Reload\ Value}{freq\ [khz]\ /\ Prescaler} = timeout\ [ms] \tag{8}$$

Using equation 8, one can make a table before deciding the desired value:

	4	**8**	**16**	**32**	**64**	**128**	**256**	
4095	511,88	1.023,75	2.047,50	4.095,00	8.190,00	16.380,00	32.760,00	**milli sec**
2047	255,88	511,75	1.023,50	2.047,00	4.094,00	8.188,00	16.376,00	**milli sec**
1023	127,88	255,75	511,50	1.023,00	2.046,00	4.092,00	8.184,00	**milli sec**

Table 8: Timeout as a function of Reload value and IWDG-Prescaler

For our implementation we used a reload value of 4095 and the IWDG pre-scaler of 64, meaning the Watchdog timer will timeout every 8.1 seconds, unless it's being refreshed by the Watchdog Task.

5.3.6 Motor control

The motor control task gets the encoder value, and sets the speed and direction of the motor over the SPI1 bus. A standard PID controller is implemented in the code to control the position of the mass. Also a state machine is used, which serves as the interface for other tasks. This s showed in the figure23 below.

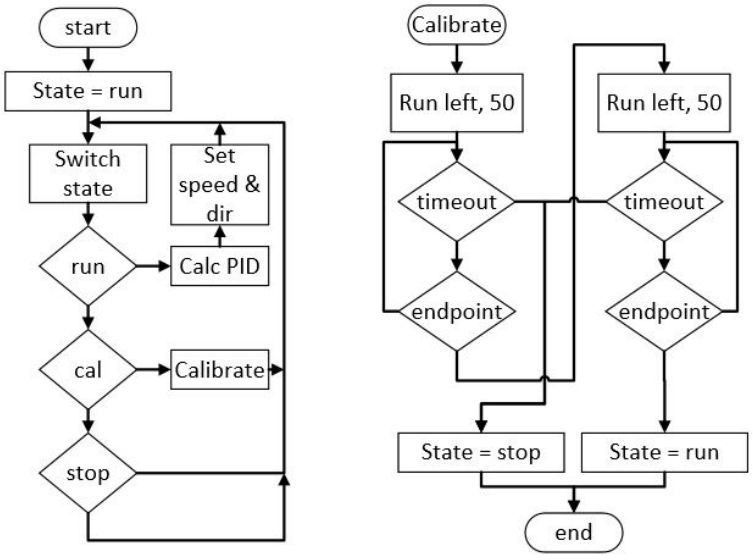

Figure 23: left: flow within motor control task. right: calibration routine

For communication on the SPI1 bus with the motor controller, a simple protocol has been implemented. The protocol is a fixed package length of 4 bytes requests sent from the Nucleo and 5 bytes reply from the motor controller. The motor always acknowledges with this reply,

regardless of data being set or get. The request and reply messages are specified in table 9 and 10.

Byte	0	1	2	3
Type	SOT	Id	dat	crc

Table 9: Request message from Nucleo

Byte	0	1	2	3	4
Type	SOT	Id	dat0	dat1	crc

Table 10: Reply message from motor controller

The start of transmission, SOT is hex 96, and is used for synchronization. The checksum, CRC is an XOR checksum of byte 0-2 for the request and byte 0-3 for the reply. The Id's are specified in table 11. The speed/dir id is specified in table 12 and the status payload is specified in 13.

Id	**Description**
0	Set speed/direction
1	Get status
2	Get position
3	Reset position

Table 11: Id's in motor control protocol

Bit	0 - 6	7
Description	data	0: left / 1: right

Table 12: Speed/dir Id. Bit 7 indicates direction

Bit	0 - 13	14	15
Description	-	1: left switch active	1: right switch active

Table 13: Status reply bits. Bits 14 - 15 indicates left and right switch

5.3.7 GPS sensor

The GPS task uses uart4, to communicate with the GPS chip, u-blox NEO-7P. The GPS transmits data with specific intervals through its uart interface with the NMEA protocol. It is an ascii protocol used for marine electronics. An example of a message, received from the GPS is showed below.

```
$GPRMC,083559.00,A,4717.11437,N,00833.91522,E,0.004,77.52,091202,,,A*57
```

The message consist of the start field, '$'. 'GPRMC' indicates which and how many fields (values) to follow, which are each separated by a comma. The asterix, '*' indicates the checksum. Lastly, the message is ended with a carriage return and line feed.

The '$' in the start and CR and LF in the end are used to synchronizing the reception of the message. This is implemented with a simple state machine in the code.

When a message has been received, its checksum is validated. If it is correct, the message is parsed with string functions from the c std lib, and the values are then made available to other tasks.

5.3.8 IMU sensor

The IMU sensor is connected to uart5 port. It is a Syd Dynamics TM352. The company provides a communication library implemented in c++. A beta-version in c is also provided. The latter one is included in the project.

In order to use the library, it must be supplied with functions for byte-wise writing and reading on the uart5 port. Besides this, the library uses functions for dynamic memory allocation from the c std lib. This proved to cause issues with the memory management. Instead functions from the FreeRTOS API where used. These are vPortFree and pvPortMalloc. They are thread-safe and lets FreeRTOS handle the memory allocation.

At the start-up of this task, the IMU is set to send data over the uart5 port with a specific interval. This way no polling is required. When data is received, the updated values are made available to other tasks.

5.3.9 Wind sensor

The windsensor, a LCJ captures CV7 is connected to uart2 port. For communication it uses the NMEA protocol, the same as for the GPS. Therefor, the structure of this task is same, as the one for the GPS. Refer to section 5.3.7 for more details.

5.4 PC GUI

To control the buoy it was decided to create a GUI with Qt QML on the ground station. The GUI had to show the position of the buoy and one should be able to monitor sensors and get and set parameters for the control. The GUI that was made is seen on figure 24. Further more it is possible to see the up-time of the system and if a message is received by the controller. Status in the top left corner will show updated if a message is correct received by the controller.

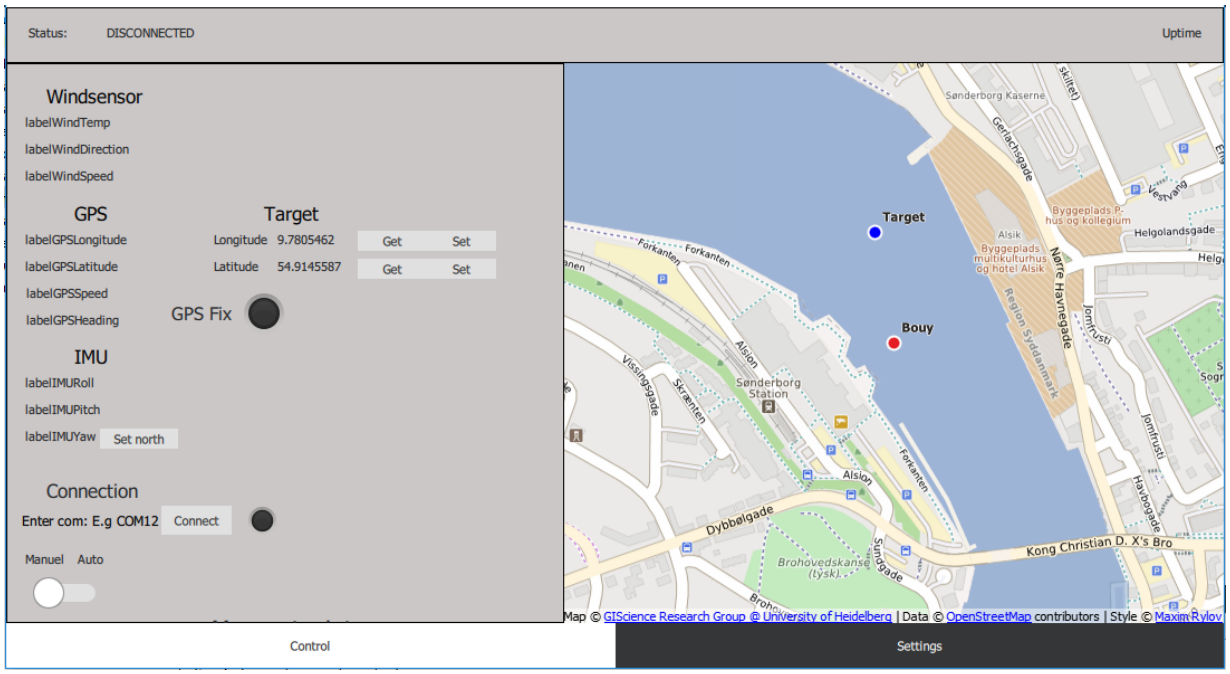

Figure 24: Control view of the GUI

On the settings page it is possible to adjust the controller parameters. For positioning the mass a PID controller is used. For the header controller 4 control parameters are used. 2 for the heading controller. 1 for the tilt stabilizing controller and 1 for calibrating the til controller. Further more it is possible to calibrate the motor controller. By switching on calibration in the heading controller menu only the tilt controller will be enabled. All this can be seen on figure 25

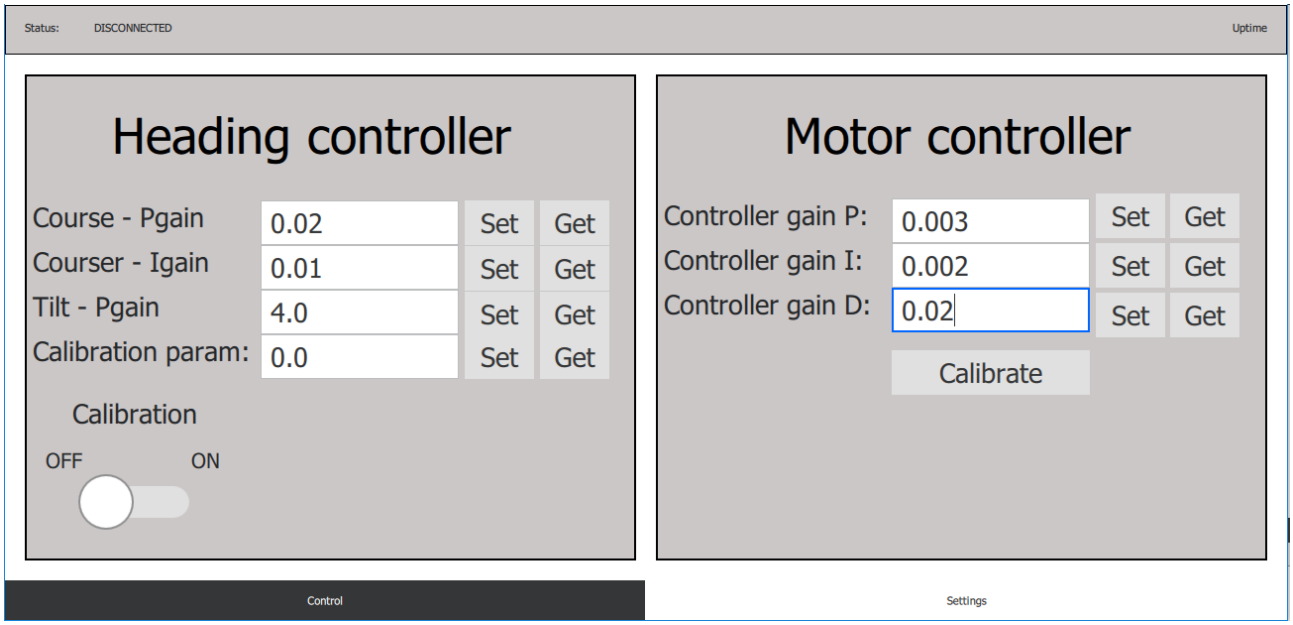

Figure 25: Settings view of the GUI

5.5 Software conclusion

Using the Nucleo STM32F429 board with CubeMX it was possible to develop a sturdy software platform for the buoy. The software platform is an architecture, which is divided into different layers. This structure makes the platform extensible, as additional functionality can be added to each of the layers. The hardware platform may be changed, by also changing to a suitable HAL layer, keeping the top layer, the user application.
With CubeMX, both a FAT file system or the lwip stack can be implemented for the specific board, which would be obvious improvements for the software of this project.
With the PC GUI an intuitive interface for controlling and operating the buoy is developed. Qt proved to be a very suitable for developing graphical applications for Windows, as well as other operating systems. It also provides functionality like serial port communication. In addition to that, it is extendable with C++ classes.
The system has been tested on the desk, where it proved to fulfill the basic requirements. All sensor values are successfully read. The controlling works in conjunction with the latitude- and longitude setpoint when the boat is turned to either left or right. The stabilizing control also works, when the buoy is tilted.
Early in the project phase, some of the sensors that where supposed to use I2C, where switched to use UART instead. This did not cause any problems, as the board used has enough UART ports. However, if several new sensors where to be added, the I2C would be an obvious improvement. Also, if the Nucleo board where to be changed to one with fewer UART ports, this would cause a problem.
The source code for this project can be found on:
`https://github.com/neophytedk/Autonomous-Buoy`

6 Conclusion

The project was concluded, due to time constraints, before the complete buoy could be tested at sea. This means that elements such as the buoy sailing capabilities, controller tuning, real-operation hardware testing and long term mechanical reliability where not fully tested. That said the project outcome provides a solid platform for further development and testing. All project development is documented and can be found on `https://github.com/neophytedk/Autonomous-Buoy`.

References

[1] R. E. E. Lars Larsson, *Principles of Yacht Design*. Adlard Coles Nautical, London, 2000.

[2] J. Kimball, *Principles of Yacht Design*. CRC Press, 2009.

[3] T. I. Fossen, *Handbook Of Marine Craft Hydrodynamics And Motion Control*. Wiley, 2011.

[4] J. Jouffroy, "Autonomous sailing buoys – final report –", 2015.

[5] F. L. S and H. J. C., "Sailbuoy: Development and evaluation of a wind-driven scientific sensor platform", 2017.

[6] *Cubemx usermanual, um1718.pdf*, `http://www.st.com/content/ccc/resource/technical/document/user_manual/10/c5/1a/43/3a/70/43/7d/DM00104712.pdf/files/DM00104712.pdf/jcr:content/translations/en.DM00104712.pdf`, (Accessed on 21/012/2017),

[7] *Official stm32 website about stm32f429zi*, `http://www.st.com/en/microcontrollers/stm32f429zi.html`, (Accessed on 21/012/2017),

[8] *Sailingyachtmodel.zip - file exchange - matlab central*, `https://se.mathworks.com/matlabcentral/fileexchange/47981-sailingyachtmodel-zip?s_tid=srchtitle`, (Accessed on 12/31/2017),