NTNU – Trondheim
Norwegian University of
Science and Technology

Construction and Control of an Autonomous Sail Boat

Henning Seeberg Stenersen

Master of Science in Cybernetics and Robotics
Submission date: June 2015
Supervisor: Morten Breivik, ITK
Co-supervisor: Anastasios Lekkas, IMT

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Problem Description

The problem at hand is to develop a small scale sail boat and achieve basic autonomous sailing. Where basic autonomous sailing is defined by *Capability to sail any feasible track or path within reasonable distances and weather conditions, predefined by waypoints, without further input from the user, using only the wind as propulsion.* The following elements must be considered:

1. Develop and/or combine necessary mechanical hardware to form a scale sail boat capable of acting as a test platform for autonomous sailing in relevant conditions.
2. Design and make a computer framework capable of:
    - Gather, log and make available a variety of measurements required for the application.
    - Executing high level control algorithms and provide a readable and modular implementation environment for such algorithms.
3. Integrate this framework into the sail boat and expand it with the necessary actuators and sensors required.
4. Make it possible to conduct tests in a practical way by integrating a suitable user interface.
5. Implement and test the necessary guidance, navigation and control to achieve basic autonomous sailing. This should include:
    - Algorithms and schemes to calibrate, transform and/or filter sensor readings into useful measurements.
    - Suggesting a control allocation scheme that ensures progress and controllability in all feasible points of sail, using only the sails as propulsion.
    - Suggesting control algorithms for autonomous sailing.

If there is time, the candidate may suggest a path planner for sail boats and try to achieve another level of autonomous sailing, where the sail boat sails itself towards a user defined location, taking into account geographical constraints given by a map, without further input from the user.

Preface

This master thesis marks the end of five good years as a student at the Norwegian University of Science and Technology. The life as a student has brought friendship, experience, adventure and happiness. I am left with the sense of belonging to the city of Trondheim, a town clearly characterized by the many students living within. It is with mixed feelings I forward this document to printing.

Ever since I was little, I have been drawn to creation and been curious to find the inner workings of things. This has been important for me to make my way into engineering. I want to thank my grandfather, Arne Stenersen, for spending countless hours building Lego^{TM} with me and helping me learn the woodworking tools in his garage. I would also like to thank my mother, Marguerethe Stenersen, for letting me play out my creative side, providing the resources for tools and equipment used in these activities, and put up with me when I crack her kitchen counter, accidentally melt aluminium on her stove and occupy her garage for months at the time. I also thank my entire family, Håvard Seeberg and Ragnvald Halset included, for being there for me through thick and thin.

Some mechanical parts of this project were made at the workshop at the institute of cybernetics engineering and I want to thank Glenn Angell for being helpful in this work and for stepping it up a little extra to help me complete the project on time. A thanks also goes to my supervisor, Morten Breivik.

Henning Stenersen
Trondheim, June 18, 2015

Abstract

A small scale autonomous sail boat has successfully been built and tested. It reached the goal of sailing a predefined track with no further input from the user, using only the sails as propulsion.

This intriguing project has many aspects. The boat has been built from scratch and the mechanical components and their functions are described.

In order to make the boat sail autonomously, electrical currents flow trough the computerized control circuits embedded into the boat. The design and implementation of embedded computerized control is an important aspect of this thesis.

The physics of sailing are tamed by the means of navigation, control and guidance. A control allocation scheme, in conjunction with sensor measurements, calibration and filtering, paves the way for path tracking and high level controls. A complete and optimal path planner has been suggested. This takes the constraints of sailing and the geographical constraints into account. The path planner algorithm also accommodate ocean currents and moving obstacles.

A powerful smart phone user interface is made to manage the boat while it sails through field tests and thereby generates experiences required for improvements.

Assignment given: January 12 2015 Supervisor: Morten Breivik, ITK

Sammendrag

En autonom seilbåt i liten skala har blitt bygget og testet. Denne har autonomt, eller selvstyrt,klart å seile en forhåndsdefinert bane under testing i feltet.

Dette spennede prosjektet har mange sider. Båten er bygget fra bunnen av og mekaniske deler og komponenter blir beskrevet i oppgaven. For å få båten til å seile seg selv har integrering av datastyring vært svært essensielt for prosjektet. Kalibrering og filtrering er implementert for å nyttiggjøre sensormålinger. Dette sammen med foreslått kontrollallokering for seilbåt, har gjort det mulig å lage automatiske moduser og kontrollere. Det har også blitt laget et smarttelefonbrukergrensesnitt for båten som har gjort det mulig utføre tester i feltet på en praktisk måte. Det er også foreslått en komplett og optimal baneplanlegger for seilbåt som tar hensyn til begrensningene ved seiling og geografiske avgrensinger. Denne kan utvides til å ta hensyn til havstrømmer og til å ungå objekter som flytter på seg.

Contents

Abbreviations

AUV	Autonomous Underwater Vehicle
CAN	Controller Area Network
ESC	Electronic Speed Controller
GPIO	General Purpose Input Output
IO	Input Output
PCB	Printed Circuit Board
PWM	Pulse Width Modulated
ROV	Remotely Operated Vehicle
SPI	Serial Peripheral Interface bus
UART	Universal Asynchronous Receiver/Trans.
UAV	Unmanned Aerial Vehicle
CG	Center of gravity
CB	Center of buoyancy
3D	Three dimensional
GPS	Global positioning system
IMU	Inertial measurement unit
I/O	Input and output
RAM	Random access memory

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 The field of robotic sailing

Compared to field robots, like UAVs, ROVs or AUVs, little research have been done in the field of robotic sailing [23].

According to the *World Robotic Sailing Championship* (WRSC) [23] nobody yet has managed to cross the Atlantic ocean with a sailing robot. Currently, the longest stretch achieved for a controlled truly autonomous sailing is a few hundred miles. WRSC states that a truly autonomous sailing robot is a robot that initially is programmed to reach a certain position and does not receive any corrections to it's actions until the destination is reached. Furthermore that many of the challenges building truly autonomous sailing robots remain unsolved.

1.1.1 Robotic sailing competitions and events

Currently there are three main competitions within the field, one of them is arranged together with a scientific conference.

1.1.1.1 The Microtransat Challenge

The Microtransat Challenge was established in 2005 [11]. Annually, teams are invited to cross the Atlantic ocean. They do not necessarily start simultaneously, but the fastest boat wins the challenge. Originally, only a westbound route was defined, but later attempts may also be made in the opposite direction. Figure 1.1 shows the start and finish lines. The first eastbound attempt was made in 2014.

Altogether, four teams have made six attempts over the years, but none have succeeded. However, there are teams attempting the challenge as this thesis is written. There exists position logging on most of these attempts. Four of the westbound attempts are shown in Figure 1.2 and the single eastbound attempt is shown in Figure 1.5.

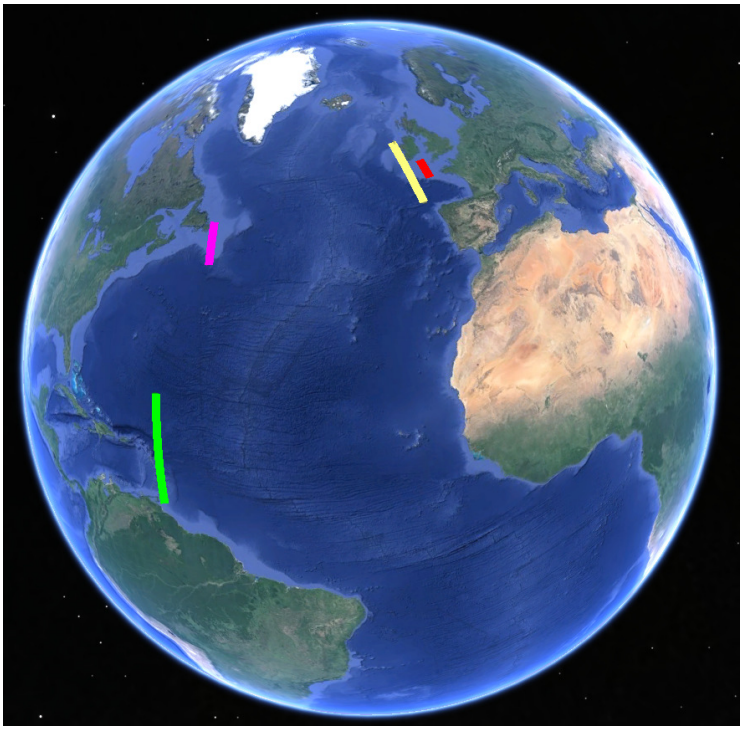


Figure 1.1: The Microtransat start and finish lines. Westbound: Red to green. Eastbound: Pink to yellow. Data from [11].

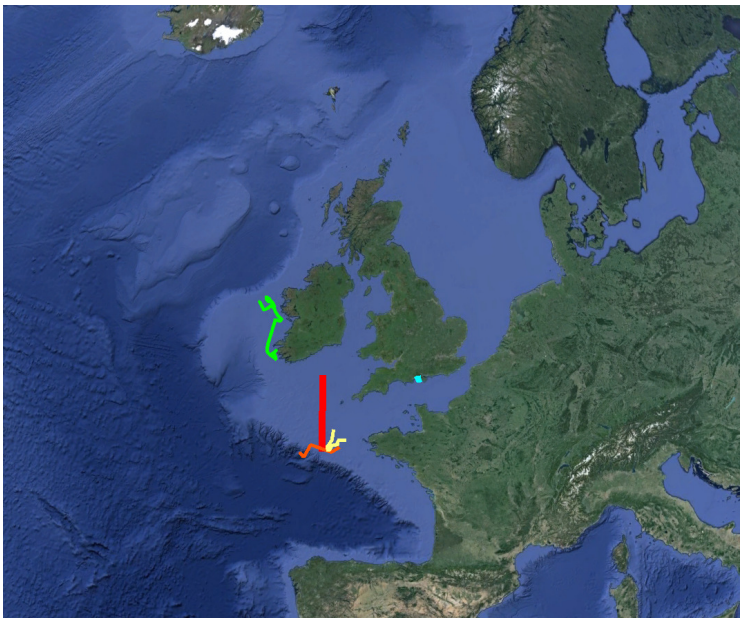


Figure 1.2: The Microtransat position logged attempts. Green=Pinta in 2010, Orange=Breizh Spirit in 2012, Yellow=Breizh Spirit in 2011, Blue=Snoopy Sloop. The Red line is the start line. Data from [11].

Aberystwyth University with their boat *Pinta* was the first team to attempt. Their plan was to sail east, cross the start line and start the westbound route. Due to a westerly wind, they did not make it to the start line before their main computer failed for an unknown reason. They had an independent tracking-device that recorded the data in Figure 1.2. After some time the tracking-device also failed. Figure 1.3 shows an image of Pinta which is a 2.95 m long, 150 kg boat.

Figure 1.3: Pinta by Aberystwyth University [11].

Team *ENSTA Bretagne* with the boat *Breizh Spirit* attempted both in 2011 and 2012, see 1.2. In 2011 they were not able to cross the start line. In 2012 their boat covered a distance of 229 km until the weather got the best of it. The boat washed ashore in Ireland nearly a month after the start date. In Figure 1.4 we can see the boat before and after the attempt. The Breizh Spirit is 1.4 meters in length, 55cm in beam and weighs 13kg.

Figure 1.4: Breizh Spirit before and after the attempt [11].

Team *Ecole Navale* with their 300kg boat made good progress in 2013, but their boat

stopped transmitting after four days. It was picked up by a passing boat.

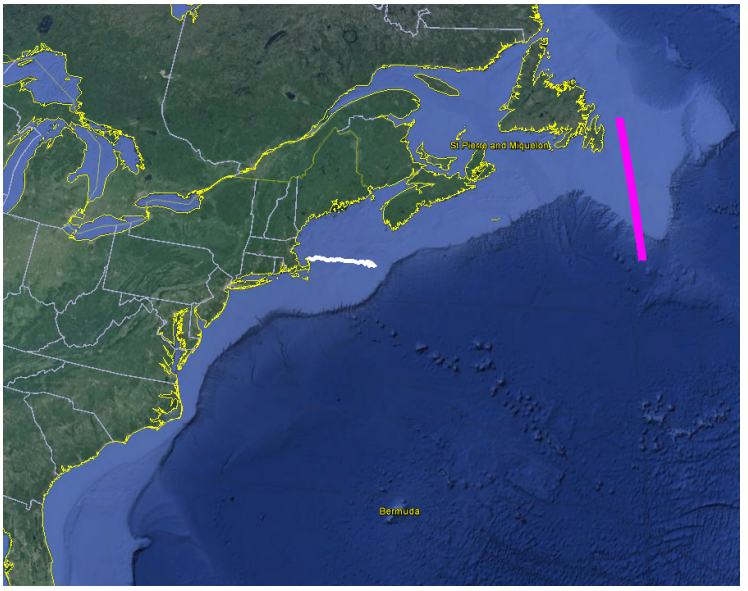


Figure 1.5: ABoutTime's eastbound attempt. The pink line is the start line. Data from [11].

As mentioned, in 2014 the first eastbound attempt was made by the *United States Naval Academy* with their boat *ABoatTime*. This run looked promising, but after nearly 400km it was caught in a scallop dragger's net [23]. Interestingly, ABoatTime is a small boat. It is 1.2 meters in length, 35cm in beam (width) and weighs 18kg. It only has a single sail that looks robustly mounted. See Figure 1.6. Its voyage can be seen in Figure 1.5.

1.1.1.2 The International Robotic Sailing Regatta

The International Robotic Sailing Regatta, also known as *sailbot*, is a student competition in North America. Its focus is to make students develop skills in different disciplines [7]. Some participants from Sailbot compete in The Microtransat Challenge and/or in World Robotic Sailing Championship.

1.1.1.3 World Robotic Sailing Championship and Conference

The International Robotic Sailing Regatta, or WRSC, was held for the 7th time in 2014. It's a regatta where autonomous sail boats, up to 4 meters in length, compete. Several competitions are held within a few days timespan. The competitions are short distance regattas as following[17][23]:

- Navigation contest: The navigation contest evaluates a boat's ability to autonomously navigate a given course. The course is shown i Figure 1.7. After starting, the boat must pass between the yellow markings, round the purple marking and cross the finish line between the red markings.

Figure 1.6: ABoatTime [11].

- Station keeping contest: A square is defined in which the boat must stay out of, see Figure 1.7. At an unknown time the boat is told to enter the square, similarly the boat is told to exit the box after an unknown time. The quicker the boat is able to enter and exit the box, the better the score. The objective is a measure of how the boat cope with time constraints while sailing autonomously.

- Collision avoidance test: Collision avoidance is one of the most important challenges within the field and the objective of the collision avoidance test is to encourage research on this topic. The course is typically set up so that boats sail in beam reach (see section 2.1.4) when an object is placed in the course. Either boats have a systems to detect the object, or at the cost of a penalty, the object size and location is downloaded from the internet. Scores are given by the boats ability to detect and avoid a collision with the object.

- Fleet race contest: The fleet race contest promotes speed and efficiency of autonomous sailing. The course is given by a triangle and is several hundred meters in length, depending on weather conditions. Points are earned by finishing first, but finishing times are adjusted with respect to boat size. In this way different boats compete fairly.

- Endurance race: The endurance contest promotes endurance and robustness. Points are earned by sailing the longest distance around a course. After a given time the race is ended and the winner is the boat who has travelled furthest - again after compensation with respect to the boat size.

- Autonomous sensing contest: In order to promote application of autonomous sailing, boats can earn extra points by performing sensing tasks at certain waypoints during the endurance race. Examples of sensing tasks are water quality readings, depth, weather parameters, photographs and more.

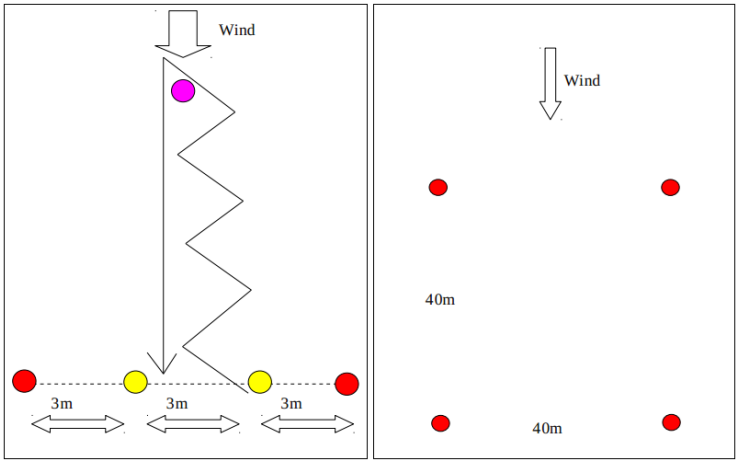


Figure 1.7: To the left: Navigation contest course [17]. To the right: Station keeping contest course [17].

WRSC defines three different classes:

- MicroMagic class: The boat is a modification to a kit called MicroMagic and restrictions apply. The boats must fit inside a sphere of 1m in diameter. The advantage of this limited class is that boats are directly comparable.

- SailBot class: The same class rules as in The International Robotic Sailing Regatta, thus attendees from The International Robotic Sailing Regatta may participate without modifying their boats. Thus the boats may not exceed 2 meters in length and the maximum draft is 1.5 meters.

- Microtransat class: The same class rules as in the Microtransat Challenge, thus at most 4 meters in length. This is the most open and free class.

Boats may win a single contest, or all contests overall, by the class. Boats may also win a single contest, or all contests overall, overall classes.

The overall winner(all contest all classes) from 2014 was *Team Seaquester* from United States Naval Academy. United States Naval Academy hosted The International Robotic Sailing Regatta on 2011. Also recall that they where the first to attempt the eastbound route in The Microtransat Challenge. Their boat, also named Seaquester, was 2 meters in length and weighed 25kg. It had a Bermuda rig and a low beam to length ratio (slim profile).

Figure 1.8: The winner of WRSC 2014: Seaquester [23].

The overall winner(all contest all classes) from 2012 was *University of Porto* with their *FASt* boat. They also won the endurance race in 2014 with the same boat. FASt is a larger boat; 2.5 meter in length, 67 cm in beam and weighs 60kg. Figure 1.9 shows this boat.

WRSC is always held together with the *International Robotic Sailing Conference*. In this conference cutting edge results and ideas are presented. Furthermore, papers are published in collections ([15], [14], [13]). The papers are describing subjects such as hardware design, energy management, control and modelling, collision avoidance, path planning, sensor design and rig design.

1.1.2 Remarks on selected literature

In the paper *VAIMOS: Realization of an Autonomous Robotic Sailboat* [15], the developtment of the autonomous sail boat *VAIMOS* is described. The project's intention for the boat developed is data acquisition for scientific use. The boat developed is a much larger boat than the boat of this master thesis. What is interesting about the *VAIMOS* is that it has successfully conduced long term test of more than 100 km.

Energy and power management strategies is an important subject within the field of autonomous/robotic sailing. In order to conduct long term missions, power must be generated. Many autonomous sail boats use solar panels to generate electricity and some have wind turbines. It seems that no one has attempted to generate power from a water turbine as the boat is sailing along. However, hybrid autonomous sailing boats exits. These can switch between sailing and propulsion with motors. An example is this is found in the paper *Developtment of ARRTOO: A Long-Endurance, Hybrid-Powered Oceanographic Research Vessel* found in [15].

The paper *Modeling and Control Design of a Robotic Sailboat* found in [15] is concerned about 6 DOF and 3 DOF models for sail boats.

Figure 1.9: The winner of WRSC 2012: FASt [22].

In [14] there is a paper called *Optimization-Based Weather Routing for Sailboats* where some experiments on bringing weather forecasts into path planning for sail boats are conducted.

A recent paper called *Toward an Autonomous Sailing Boat*[27] are concerned with modelling and simulations of a sail boat. A *detection and avoidance* system is built and tested. A prototype autonomous sail boat is also made, but results are based on simulation and tests conducted under controlled circumstances.

1.2 Why autonomous sail boats?

An autonomous sailing boat is above all viewed as an interesting, challenging and relevant problem to solve with respect to engineering cybernetics. The autonomous sail boat has some distinct properties; it does not need fuel, crew or supplies. In fact, it may generate energy during sailing and thus supply power to drones, life support systems and other equipement. Thus it may operate "infinitely" at very low cost. There are not many vehicles capable of this and thus some initial ideas of applications may be:

- Carry equipment and sensors to gather scientific data.
- Inspect offshore equipment.
- Iceberg detection and monitoring.
- Oil spill clean up operations.
- Cheaper freight and shipping to remote places.
- Surveillance and aiding coast guard; i.e. detect illegal fishing, smuggling etc.
- Several boats may act in teams or patterns.

It is also worth mentioning that development of autonomous sailing systems may lead to development of assisted sailing for ordinary sailing boats. These systems may help the sailor to sail more efficiently.

1.3 Vision: Autonomous sailing

This project is founded on curiosity and joy. The curiosity about how to construct a small scale sailing boat capable of autonomous sailing. The joy of making the components from scratch, puzzling them together and bring the system to life by means of embedded computerized control, navigation, control and guidance.

1.4 Main contributions

The main contributions are listed below:

- Design and building of a small scale sail boat with necessary hardware to test and achieve autonomous sailing in relevant conditions.
- Design and making a computer framework capable of:
  - Gather, log and make available a variety of measurements required for the application.
  - Executing high level control algorithms and provide a readable and modular implementation environment for such algorithms.
- Development of a powerful smart phone user interface which enables efficient testing and running of the system. This included integrating a data link and necessary message passing to ensure reliable and stable system management.
- Development of the necessary software and drivers to enable all the hardware features and ensure proper information flow in the system.
- Suggestion of a control allocation scheme that ensures progress and controllability in all feasible points of sail, using only the sails as propulsion.
- Suggestion of simple control algorithms for autonomous sailing.

- Execution of several field tests where basic autonomous sailing was achieved.
- Suggestion of a complete and optimal path planner that takes the constraints of sailing and the geographical constraints into account. The path planner algorithm also accommodate ocean currents and moving obstacles.

Chapter 2

Components and Physics of a Sail Boat

Without sail boats, civilization as we know it would not have existed. Sail boats greatly improved human mobility and have been a foundation for trade, transport and fishing [18]. What exactly is a sail boat? That is; which components are needed in a sailing vessel? What are the physics behind sailing?

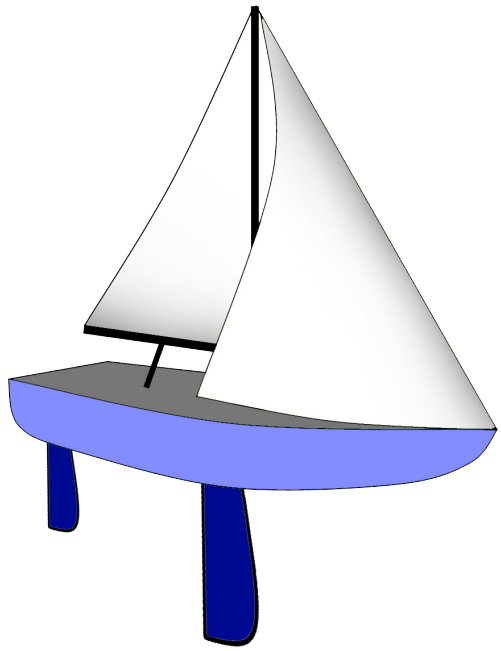


Figure 2.1: Sail boat with the common Bermuda rig.

2.1 Components of a sail boat

A sail boat works by utilizing the different velocities between the water and the surrounding air. In order to accomplish this task, some main components are needed:

- Hull
- Keel (may be integrated in the hull)
- Rudder
- Rig with sails

Following is a brief introduction to these components.

2.1.1 Hull

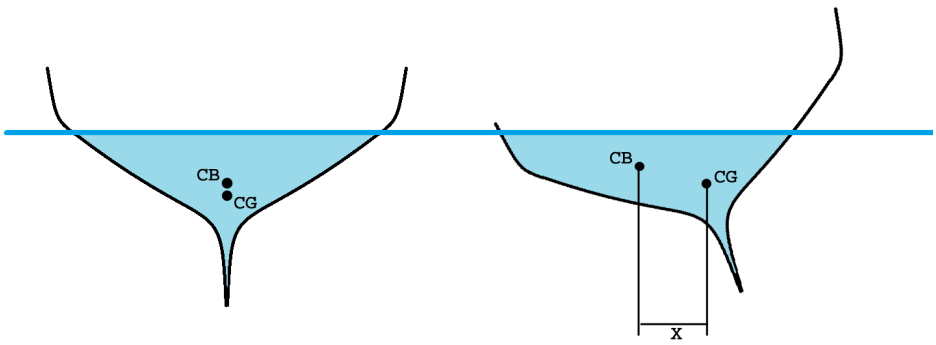


Figure 2.2: Centre of Buoyancy and rightening momentum.

The hull fulfils several tasks:

- Holding the boat and its payload afloat.
- Cutting through the water with as low drag as possible.
- Impose a rightening momentum when the hull is rolling.
- Resist movement in the pitch axis direction.

Each hull type combines these tasks in its own way which defines its characteristics.

The rightening momentum is proportional to the horizontal distance between *centre of gravity* (CG) and the *centre of buoyancy* (CB), X in Fig. 2.2. CB is the volume centre of the hull's footprint in the water. At zero roll and pitch, CG and CB is on the same Z-axis. While the CG usually stays the same, the CB is mainly dependent on the roll angle. Large pitch angles also affects CB.

2.1.2 Keel

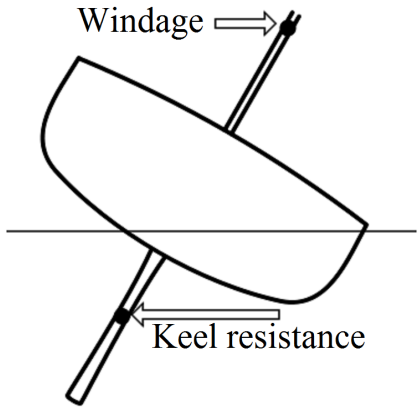


Figure 2.3: Keel cancelling horizontal forces perpendicular to heading [9].

As mentioned, the keel is either a separate structure or an integrated part of the hull [9]. It has two functions:

- Lowering the centre of gravity in order to strengthen the rightening momentum
- Acting as a hydrodynamic element, a fin, that balances out the horizontal forces along the pitch axis imposed by the wind pushing the sails. See Figure 2.3

Some sail boats have an external keel that clearly reveals its functions. For example, in Figure 2.4, the Hanse 575 clearly has a keel that makes up a large fin with a heavy ballast on the tip. On the other hand we have the Oseberg Viking ship. On this ship the keel is integrated as an edge running all along the ships bottom. The ballast is placed internally at the very bottom along the ship.

Figure 2.4: Hanse 575, [5], and the Oseberg Viking ship, [21].

2.1.3 Rudder

The rudder, also known as a control surface, is a hydrodynamic element that redirects the surrounding fluid [16]. This creates a controllable moment in the yaw direction.

On a sail boat, the rudder is only for fine tuning. The sails are set in such a way that forces are balanced about the keel. Actually, it's possible to sail a ship without the rudder, given that it has enough degrees of freedom in its sail rig. In comparison, a windsurfing board does not have a rudder.

Many rudder designs provide feedback to the sailor. This helps the sailor tell if the sails are set in such a way that the boat is balanced. This is achieved by mounting the rudder on an axis mounted in front of the rudder's area centre. In this way, the sailor feels a moment on the rudder axis proportional to the force generated by the rudder.

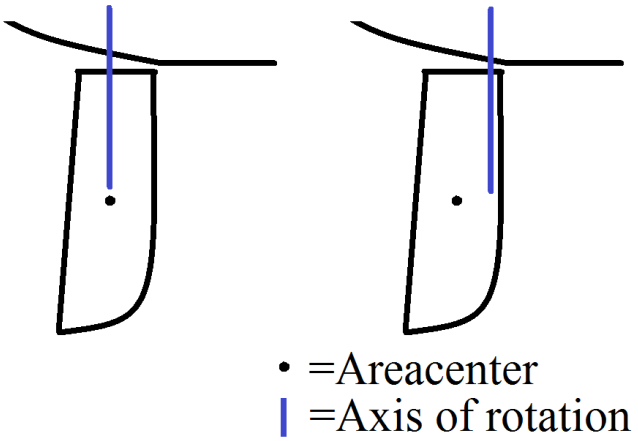


Figure 2.5: Left: Balanced rudder. Right: Unbalanced rudder, the sailor feels forces acting on the rudder.

2.1.4 Rig and sails

There are two ways to utilize a sail:

- When sailing in nearly the same direction as the wind; The wind simply pushes the boat along because the sail drags. The sail is set to achieve the highest possible drag.
- When sailing across or upwards the wind; The sail acts as an air foil and can be modelled as a wing generating lift and drag.

These two modes allows for sailing in a sector spanning typically 270°, see Figure 2.6. Over the years, one has found it convenient to define 5 main points of sail. With respect to Figure 2.6 these are:

- In Irons/into the wind (A)
- Close Hauled (B)
- Close Reach (not shown in Figure: between Close Haul and Beam Reach)
- Beam Reach (C)
- Broad Reach (D)

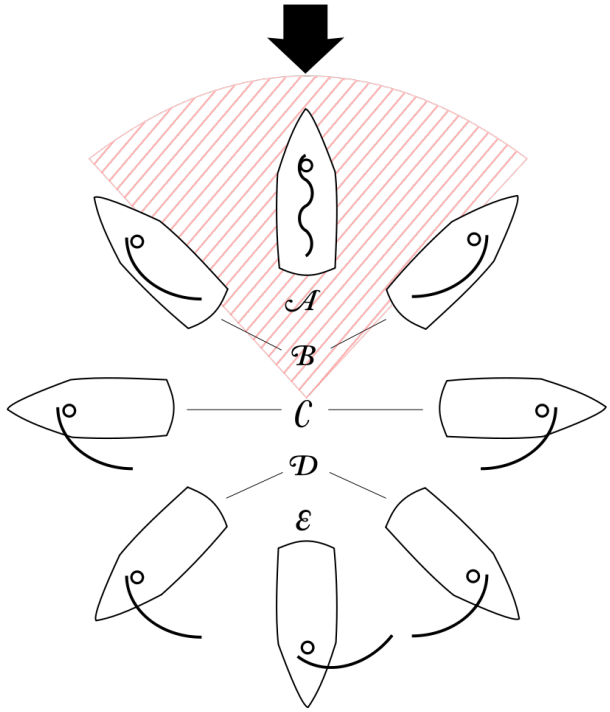


Figure 2.6: Points of sail[18].

- Running (E)

A good sail rig must:

- Achieve high freedom in the sails. Allowing for trim that optimizes the lift in the sails when reaching/hauling. Thus also allowing to sail very close hauled.
- Span a large drag area for running.

2.2 The physics behind sailing

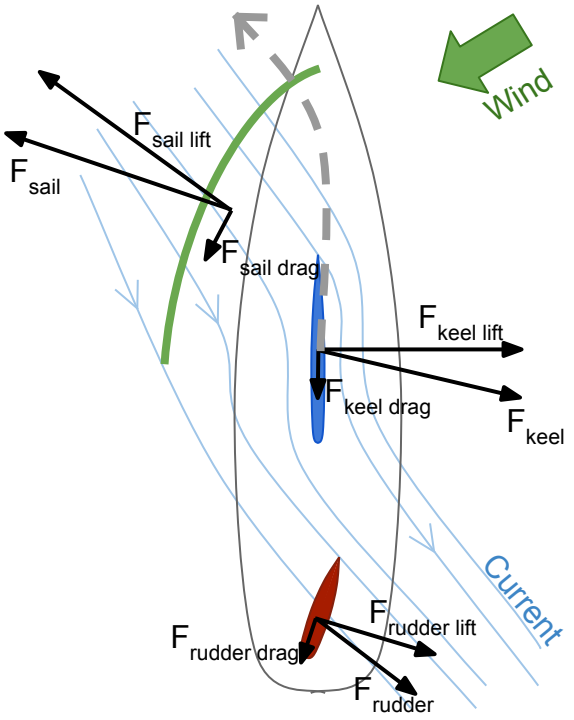


Figure 2.7: Forces acting on a sailboat.

If we are to look at the sail boat from a physical point of view, we can see that the sails, the keel and the rudder do all act as wings/foils. Hence they can all be modelled as foils [29]. The forces they create are illustrated in Figure 2.7.

From aerodynamics theory we know that lift is created when air is moving faster on the top side of the wing compared to the bottom side. Sailors have taken advantage of this knowledge and many sails carry pairs of *tell-tails* to indicate the difference in airspeed on coincident sides.

Analysis of wings are extensively researched, especially for use on aircraft. Thus good approximations and models exists and it should be possible to take advantage of this in the applications of the boat. The challenge resides in the uncertainty in which shape the sail currently holds in given conditions. This is because the sail is made of cloth and is not rigid like an air plane wing.

The sail is shaped by its geometrical composition of cloth, the points where it's attached and the current trim and how much the cloth stretches in the current wind conditions. Thus approximations must be made.

A common approximation for modelling of foils, hence sails, rudder and keel/hull are

[29]

$$F_{lift} = L = \frac{1}{2}\rho A v_a^2 C_L(\alpha) \tag{2.1}$$

$$F_{drag} = D = \frac{1}{2}\rho A v_a^2 C_D(\alpha), \tag{2.2}$$

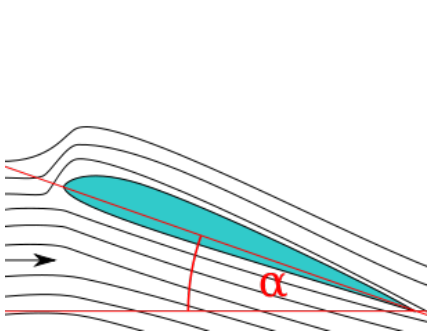
where $A > 0$ is the plan area of the foil, $\rho > 0$ is the density of the surrounding fluid, $v_a > 0$ is the speed of the surrounding fluid and $\alpha > 0$ is the angle of attack. The angle of attack is the angle the foil makes with the fluid velocity, see Figure 2.8a. $C_L > 0$ and $C_D > 0$ are the lift and drag coefficients as a function of α.

To get detailed expressions of C_L and C_D a wind tunnel testing is required [24]. An approximation for C_L is [24]

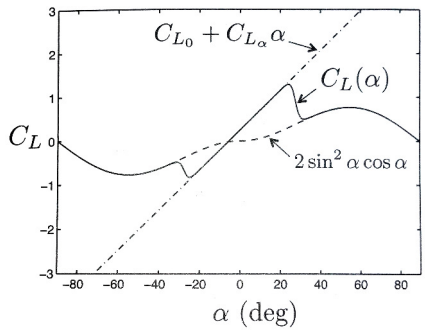
$$C_L(\alpha) = (1 - \sigma(\alpha))[C_{L_0} + C_{L_\alpha}\alpha] + \sigma(\alpha)[2sign(\alpha)sin^2(\alpha)cos(\alpha)] \tag{2.3}$$

$$\sigma(\alpha) = \frac{1 + e^{-M(\alpha-\alpha_0)} + e^{M(\alpha-\alpha_0)}}{(1 + e^{-M(\alpha-\alpha_0)})(1 + e^{M(\alpha-\alpha_0)})}, \tag{2.4}$$

where M, α_0, C_{L_0} and C_{L_α} are positive constants. This approximation incorporates the effect of stall, see Figure 2.8b. Stall is when the angle of attack so large, that the fluid can not follow the curvature of the foil. Thus turbulent separated flow occurs on the low pressure side of the foil.



(a) The angle of attack, α [2].



(b) $C_L(\alpha)$ [24].

The drag coefficient C_D is also a non-linear function of α [24]. There are two contributions to the drag:

- Parasitic drag: Caused by the shear stress of air dragging over the surface of the foil. It's roughly constant.
- Induced drag: Caused by the foil redirecting the air. It is roughly proportional to the square of the lift force.

Thus, a common approximation for the combined drag is

$$C_D = C_{D_{parasitic}} + \frac{(C_{L_0} + C_{L_\alpha}\alpha)^2}{\pi e AR}, \tag{2.5}$$

where $C_{D_{parasitic}} > 0$ is the parasitic drag constant, $e \in [0.8, 1.0]$ is the Oswald efficiency factor and $AR > 0$ is the wing aspect ratio.

2.3 The sail boat as a control system

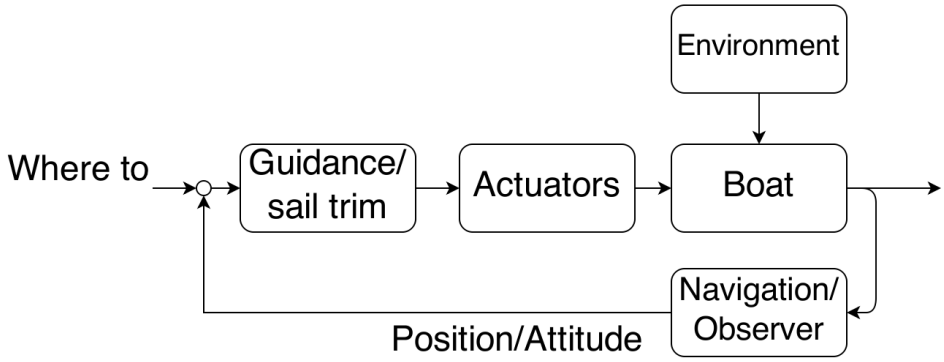


Figure 2.9: The sail boat as a control system.

In Figure 2.9 we see the sail boat viewed as a block diagram. Conventionally the loop is closed by humans doing navigation and sail trim. Furthermore the sails are usually trimmed by humans powering winches. Removing the human from the loop imposes some immediate challenges:

- Human navigation using visual references, such as lighthouses, pillars and landmarks, must be replaced by a positioning system.
- Sail trim must be automated. Actuators must replace the human work and sail trim must be calculated. Sensors must give the foundation to make trim decisions.
- Either the boat must follow a predefined feasible path or automated path planning is required for path generation. Paths must accommodate the limitations introduced by sailing.
- If the boat is sailing in areas where other vessels are present, a collision avoidance system is desirable.

Chapter 3

Boat, Rig and Mechanics

Figure 3.1: *Sigrun*

Figure 3.2: *Marguerethe*

In order to conduct the activities at hand, a boat with certain requirements are needed. The requirements are:

- The boat must sail sufficiently in order to test control theory and algorithms.
- The design must be simple and robust.
- The boat must be large enough to carry the payload (actuators, sensors and electronics) to complete the task.
- It has to be portable, thus fit inside a car.
- It must be easy to set up the boat at test cite.

As mentioned this project is founded on a hobby project. In 2010, the boat *Sigrun* was launched into water for the first time, see Figure 3.1. This was a small, 60 cm in length, remotely controlled sail boat. As it proved to be a successful and fun project, it became desirable to build a larger and more complex boat capable of sailing itself. This went hand-in-hand with studying cybernetics engineering. A new hull was made and this boat would later be known as *Marguerethe*. The new project went on hold for a while, but was later picked up with the intention of being used in this master thesis. The rig and the rudder were also completed before work on this thesis started, but the actuator system was to complex and not robust enough to be used within the scope of this thesis. A simpler actuator system needed to be made along with the deck and the keel. Thus, the option of modifying a pre-made R/C model sail boat was considered. However, it was considered best to continue the work on *Marguerethe*, keeping the above requirements in mind. Plans were made to minimize practical work with making the actuation system, deck and keel. *Marguerethe* is from here on refereed to as *the boat*.

Following are some basic facts about the finished boat;

- Length over all: 145.5cm
- Hull length: 141.5cm
- Length at waterline: 116cm
- Beam: 21cm
- Displacement: 5kg
- Mast height above waterline: 135.5cm
- Main sail area: $0.34m^2$
- Genoa sail area: $0.38m^2$
- Total sail area: $0.72m^2$
- Draft: 73cm

3.1 Hull

The hull was cast in glass fibre, Figure 3.3 shows this process. First drawings were made, then bulkheads were cut out with a CNC mill in wood. Then, thin sheets of wood were glued onto the bulkheads. This made up a wooden replica of the hull which was used to crate a female mold. Glass fibre sheets were then put in layers with epoxy inside the female mold. The end en results is the glass fibre hull. Figure 3.4 shows the glass fibre hull compared to the 3D model.

Figure 3.3: The hull casting process.

Figure 3.4: The 3D drawing compared to the finished hull.

3.1.1 Waterproofing and electronics compartments

One of the main mechanical challenges has been to ensure waterproof compartments for the sensitive electronics. Another challenge has been to ensure that the boat is unsinkable. The unsinkable property is important as the completion of the this master thesis has been dependent on a working sail boat.

A simple way of ensuring the unsinkable property is to fill the boat with Styrofoam or a similar material. At first, this seemed inconvenient as the Styrofoam would be in the way of the hardware going into the boat. Hence, a more conventional technique was used; adding bulkheads. The hull was fitted with two bulkheads, making up three compartments sealed from each other. As indicated, this is a common way of reducing the risk of sinking in case of a leak, as the boat will remain afloat with one compartment flooded.

To further improve safety and waterproofing, the electronics were placed in a waterproof compartment inside the center compartment. Thus the electronics were doubly isolated. See Figure 3.5.

This was a very neat solution as the compartments were to be accessible from the outside through hatches in the deck. This made it easy to manage and reprogram the electronics

Figure 3.5: Redundant waterproof compartments.

while it remained installed. Hence, actuators and sail trim could easily be tested on the desktop without any extra work of installing or uninstalling hardware.

After the first field test, it was clear that the compartments did not perform satisfactory, see Section 7.2. This was a setback as many readjustments had to be made.

To properly ensure a waterproof compartment, the electronics were fitted into a plastic pipe. The plastic pipe where permanently sealed of in one end. A cap to seal of the other end was turned out at the cybernetics engineering workshop at NTNU. The cap was fitted with an O-ring groove and O-ring that ensured a waterproof seal against the tube when inserted. Some wires/connectors needed to be made available outside the tube for connecting with servos and external sensors. These were pulled through the cap and sealed. The assembly can be seen in Figure 3.6 and 3.7.

The new tube housing for electronics was tested before it was put into use. After being submerged for 1 hour at 1 meter depth, there was no sign of leakage. The observant reader may have noticed that the connector pins on the outside of the cap lacks isolation. This is not a problem as the tube is well protected within the hull of the boat. If water still should make contact with the connector pins, this should only temporary restrict functionality. Inside the hull (and outside of the pipe) there is often a moist environment during sailing/field tests, but normally no water laying around. For peace of mind, Styrofoam was also added within the boat together with the new electronic housing.

The housing has proved to be a robust solution in field test. However, the solution is less neat as the electronics are now less accessible, see Section 8.2.2.

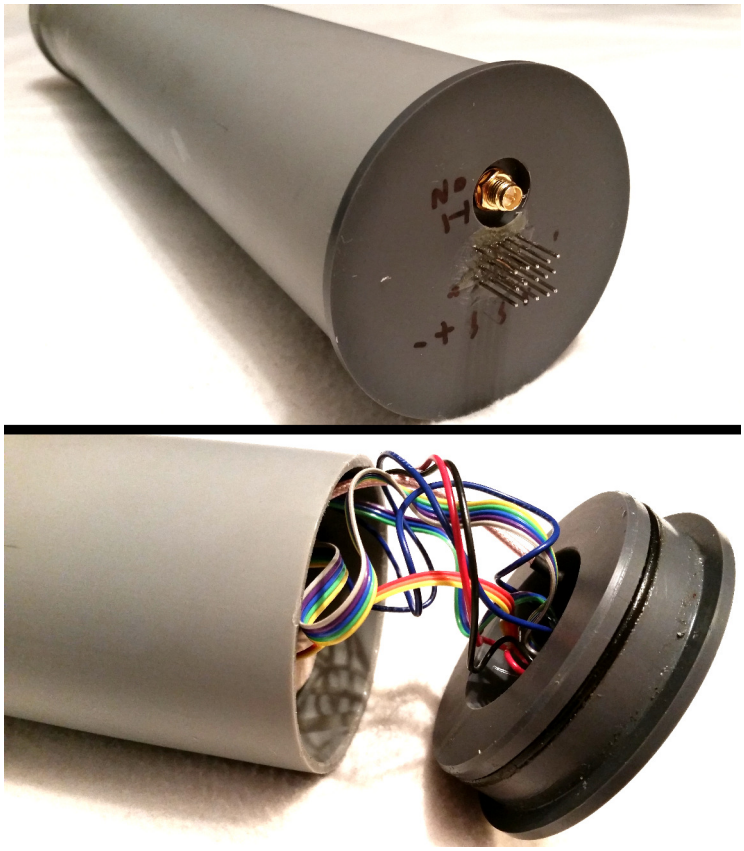


Figure 3.6: The new waterproof tube housing for electronics.

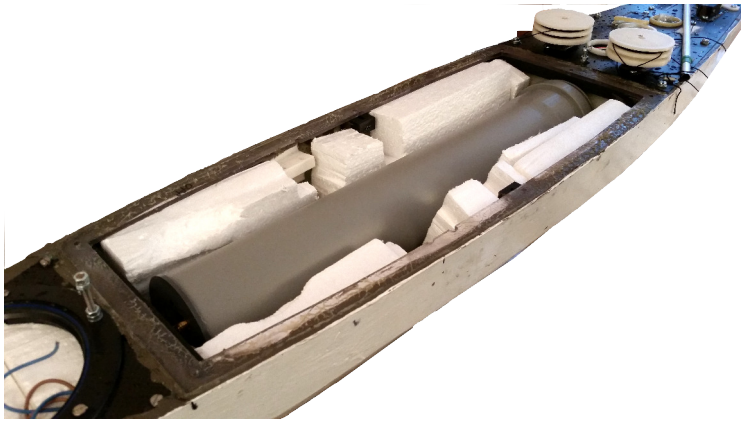


Figure 3.7: The housing placement within the hull.

3.2 Rudder

The rudder assembly,see Figure 3.8, consists of a simple servo mounted together with a tube going through the hull. The rudder axis fits inside the tube. In this way, water will not flow into the boat. This is a simple and robust solution.

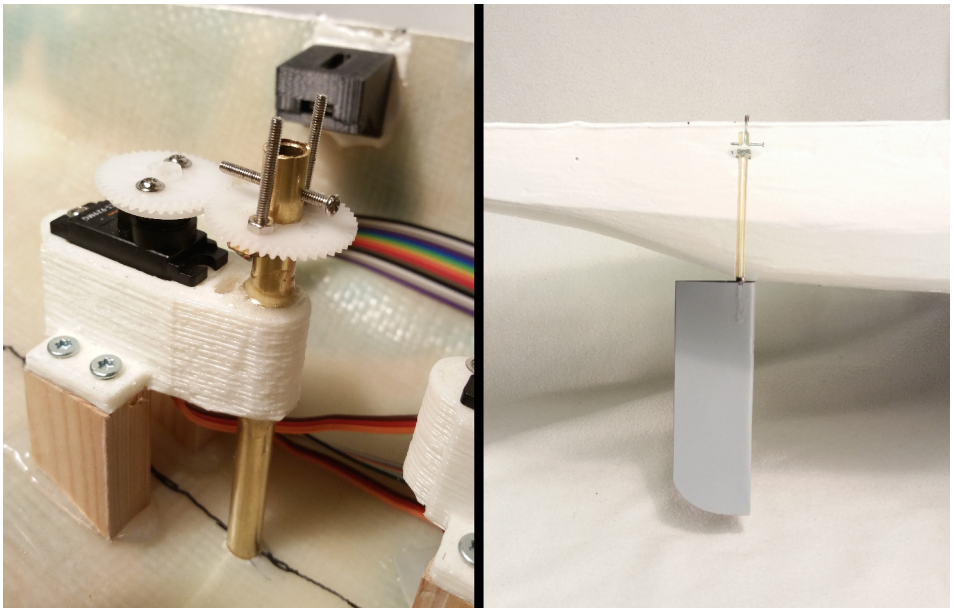


Figure 3.8: Rudder assembly

3.3 Deck

The deck is cut out of carbon fibre sheets using a professional CNC mill at the cybernetics engineering workshop. A sealing edge was also made. The sealing edge runs around the edge of the deck and are attached to the hull with silicone, see Figure 3.7 and Figure 3.10. The edge allows the deck to be detachable. The edge is lubricated with grease to form a good seal towards the deck.

Screw hulls, servo slots, hatch spaces and the mast footprint are also cut out in the same process. Thus accurate drawings were required to make these parts. Brackets with nut inserts were 3D printed and glued into the hull. In this way, screws are inserted from the top down and the nuts are self holding. Figure 3.10 shows the conceptual drawing of this assembly.

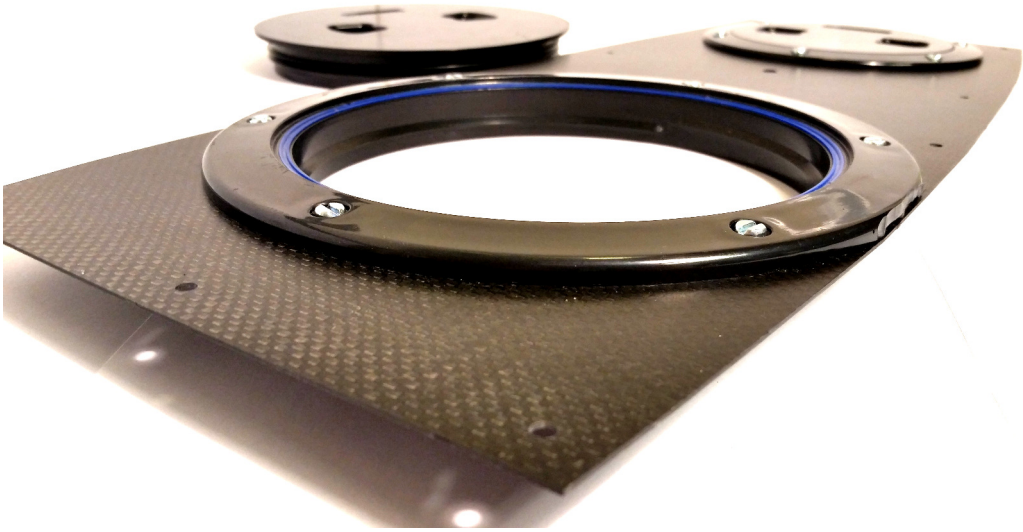


Figure 3.9: A section of the deck, made in carbon fibre, with waterproof hatches installed.

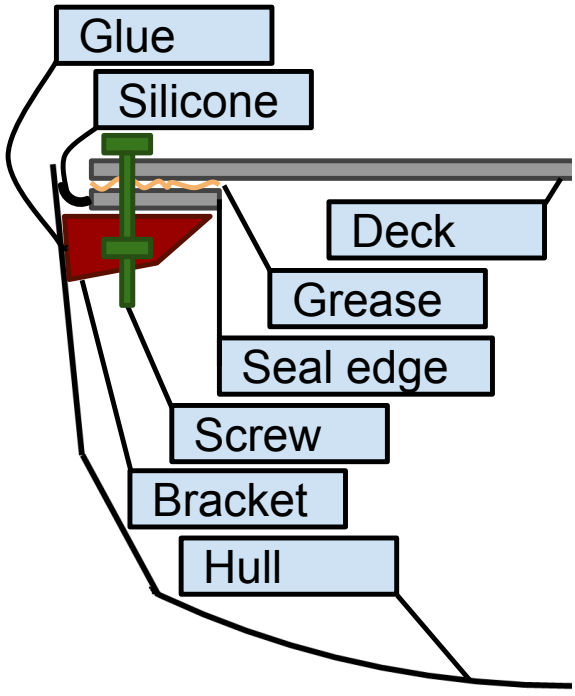


Figure 3.10: Conceptual deck assembly drawing.

3.4 Rig

It can be argued that a fixed wing rig is a better choice for autonomous sail boats. However, these are more complicated to implement and the boat is equipped with the common Bermuda rig.

Figure 3.11: Rig and sails.

The rig is made out of carbon fibre tubes and stainless steel wires. The sails are sawn from lightweight nylon cloth. The same cloth is used for kite making which has similar requirements to strength, airtightness and low weight. The rig with sails are shown in Figure 3.11.

3.4.1 Sheet drive types

On regular sail boats, the user manages the sheets of the different trim functions manually. Here, three different drive types are suggested to replace the user's manual work (see Figure 3.14), each suitable to different functions:

- Directly from drum: The most simple way of achieving the translatoric movement of the sheet is too wound the sheet, directly upon a rotating drum. The problem is that

sheets are prone to getting stuck or not wound up properly. This typically occurs when there is no tension in the string and the string wounds outside the drum or upon unwounded parts of itself. Thus the direct drum is only used, in this case, for sheets that are always tensed.

- Tensioned drum: Two strings are wounded around a drum, one clockwise and one anti-clockwise. Thus when the drum turns one string feeds, while the other retract. The two strings are tensioned to each other around a bearing using a spring. Thus there are always tension when the string is wounded up on the drum, and proper windings are assured. The sheet is attached to the string and through an exit pulley. This more robust design, is widely used in model sail boats.

- Coupled drums: The main reef and Genoa reef does not require translational movement. Instead of using bulky drive chains, two drums can be coupled with strings. In the same fashion as with the tensioned drum, one string is wounded clockwise and one anti-clockwise. Thus the coupled drums wont slip.

Common for the three drive types, is that the drums can be made in different sizes to achieve different ratios of movement. The servos driving the drums may wander up to six turns. To utilize the full resolution and accuracy of these servos, drums should be sized to utilize all the six turns.

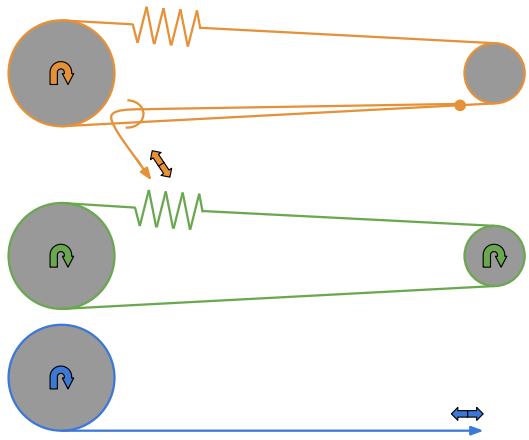


Figure 3.12: Sheet types. Blue: Direct drum. Orange: Tensioned drum. Green: Coupled drums.

3.4.2 Rig functions

The rig's function is to allow the system to shape and hold the sails depending on the current conditions. The most important trim functions are the main sail sheet and the Genoa sail sheet, which controls the angle of the main sail and Genoa sail.

Additionally the mast is rollable, thus full or partial reefing of the main sail is supported, see *main sail clew out haul* and *main sail reef* in Table 3.1. This feature has proved vital in field tests, see sections 7.5 and 7.6. The boom can also be pulled up and down too tension the main sails falling edge, see *kick/boom vang* in Table 3.1.

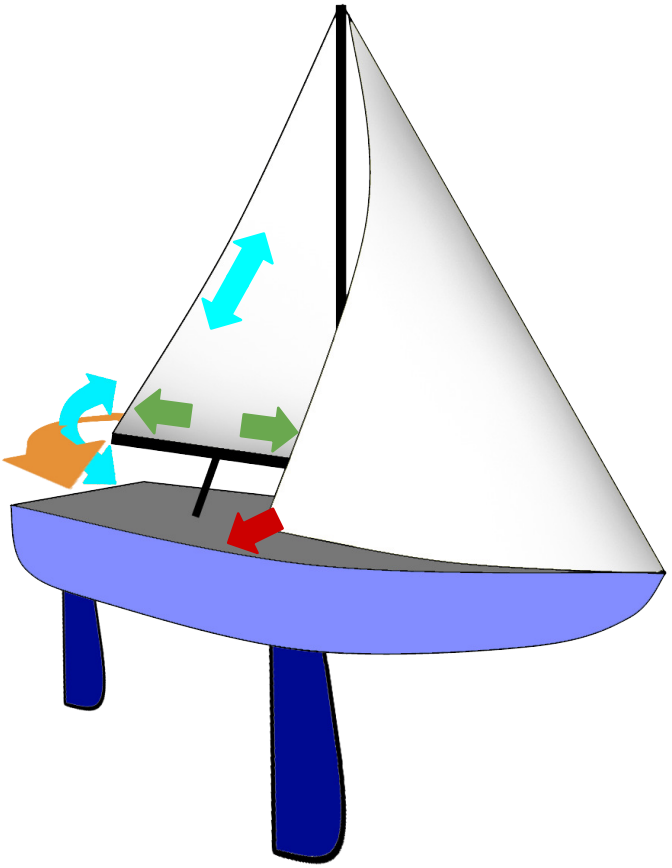


Figure 3.13: Sail trim. Orange: main sheet angle, Cyan: tack, Green: main sheet tension/shape and reef, Red: Genoa angle.

In the beginning of this project, it was hoped that it would be time to automate several trim functions. Hence, plans were made for automating the them. Also extra servo slots are cut in the aft deck to support extra winches if more automation becomes desirable. Table 3.1 lists all the specific rig functions that are available for automation. It also links the different functions to the illustration in Figure 3.13. However, in the end, only the main sheet and the Genoa sheet are automated.

Name	Function	Drive type	Drive installed	Appearance in Figure 3.13
Main sheet	Sets the boom/main-sheet angle and hence the main sheet angle of attack. In this particular rig, the main sheet does not tension the main sail in the upward direction because the sheet outlet is mounted at the same height as the boom. Thus the main sheet can be considered orthogonal to the kick or boom vang.	Tensioned drum	Yes	Orange
Genoa sheet	The Genoa sheet is the equivalent of the main sheet for the Genoa sail, but opposed to the main sheet the Genoa sheet also gives tension in both the horizontal and upward direction of the sail. Thus the Genoa shape is not adjustable, but given by the geometry of the rig and the sail itself.	Tensioned drum	Yes	Red
Kick/ boom vang	The kick or boom vang sheet pulls the boom downwards, thus increases the tension in the main sail's falling edge. With high tension, the main sail tends to not twist. Thus the angle or angle of attach is the same high up in the sail as at the very bottom. Visa versa, the sail tends to twist with low tension. Low tension results in a lower angle of attack in the top part of the sail. The twist is set to correlate with the wind gradient as the wind usually blows faster along the top part of the sail. In this way the lift can be maximized everywhere in the sail without any part of the sail staling. As the drive for this feature is not installed, the boom vang is manually set before the boat is launched into water.	Direct from drum.	No	Light blue
Main sail clew out haul	The main sail clew pulls the main sail out towards the tip of the boom. This tensions the main sail in the horizontal direction and thus changes the shape of the sail. The higher the tension the more slim wing profile. In high winds it's desirable with a slim profile in order to avoid turbulence occurring along the shape. Visa versa, in low winds it's desirable with a fat profile to maximize lift. As the drive for this feature is not installed, the main sail clew is manually set before the boat is launched into water.	Tensioned drum	No	Green
Main sail reef	In standard Bermuda rigs it is common that the main sail reef pulls the main sail upwards when setting the sail, this is not the case here. The boat is equipped with a rolling mast, thus the sail may roll inn on the mast when the main sail clew out haul is released. This allows for full or partial reefing. It is desirable that the sail is reefed in very high winds, otherwise the boat experiences very high roll angles, making sailing very inefficient. As the drive for this feature is not installed, the main sail reef is manually set before the boat is launched into water.	Coupled drums	No	Green
Genoa reef	In the same fashion as the main sail reef, the Genoa may fully or partially roll in on the rod at the leading edge. This feature is not used.	Coupled drums	No	

Table 3.1: Rig functions available for automation.

3.4.3 Drive layout

The layout of the sheets drives are shown in Figure 3.14 for the fully rigged boat. As mentioned, only the main sheet and Genoa sheet is used(2 and 3 in Figure 3.14).

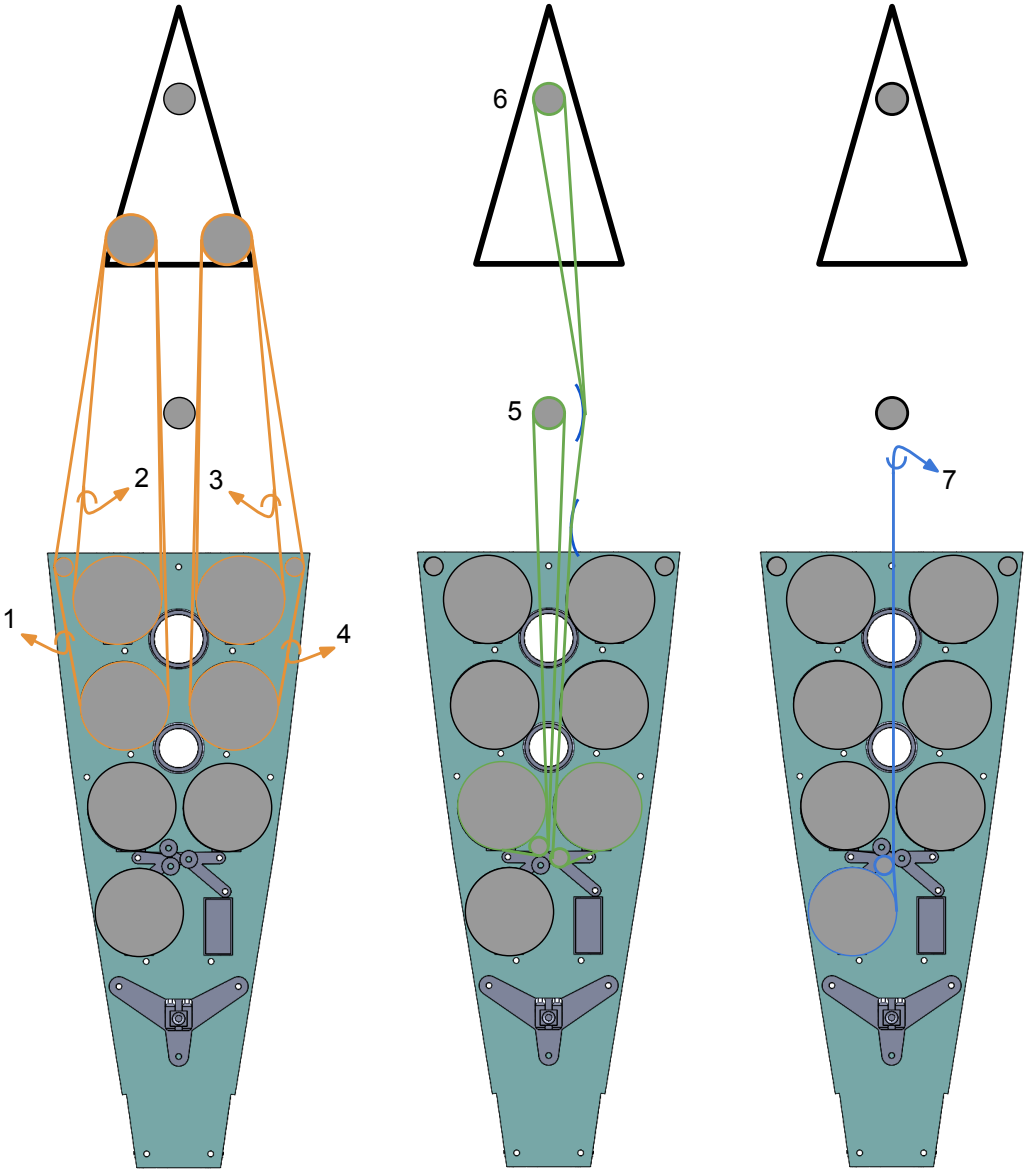


Figure 3.14: Sheet layout. See table 3.2.

Function	Number in fig
Main sheet	3
Genoa sheet	2
Secondary Genoa sheet	4
Kick/boom vang	7
Main sail clew out haul	1
Main sail reef	5
Genoa reef	6

Table 3.2: Sheet layout from Figure 3.14

3.4.4 Keel

The keel is made up of 3D printed bracket and an aluminium flat bar. The flat bar makes up the hydrodynamic element, see Section 2.1.2. A weight is attached to the tip of the flat bar to strengthen the rightening moment, see Figure 3.17.

The bracket has several attachment wholes which allows the flat bar to be positioned in 13 different positions, see Figure 3.15. The flat bar has several wholes drilled into it, and hence it can be tilted to the following angles; $-15°$, $-10°$, $-5°$, $0°$, $5°$, $10°$, $15°$, see Figure 3.16. In this way the user has some room to balance out the the boat, with respect to center of gravity and moment of force.

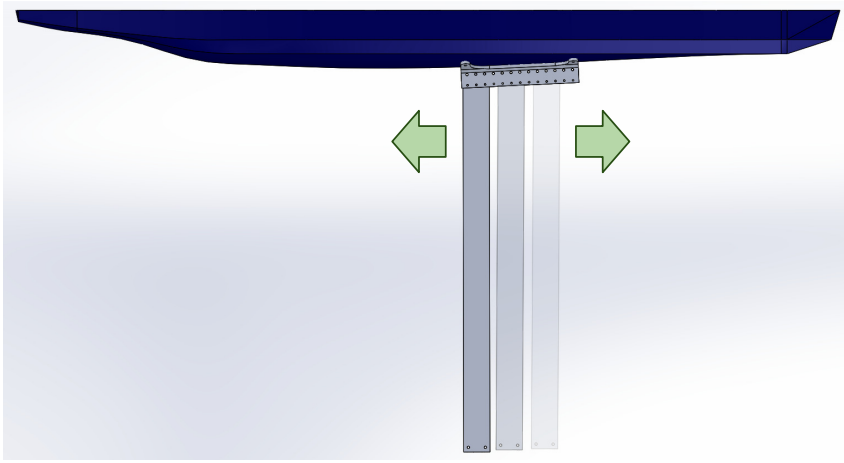


Figure 3.15: The keel can be repositioned.

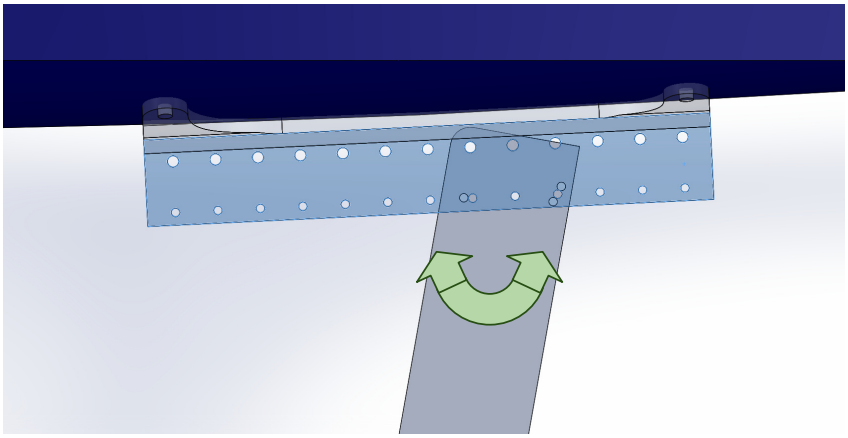


Figure 3.16: The keel can be tilted in 7 different angles.

Figure 3.17: The keel.

Chapter 4

Embedded Computerized Control

An important part of control engineering is to embed computerized control and hence make a foundation for control algorithms to reside. The embedded computers and other surrounding hardware makes up a hardware framework. The next section will address the development of such a hardware framework.

For the rest of this chapter, the hardware expansions made to this framework is described. Also, how the final user-interface is integrated into the system, and how information flows in the system is addressed. Figure 4.1 shows the final system overview and the next sections relates to it.

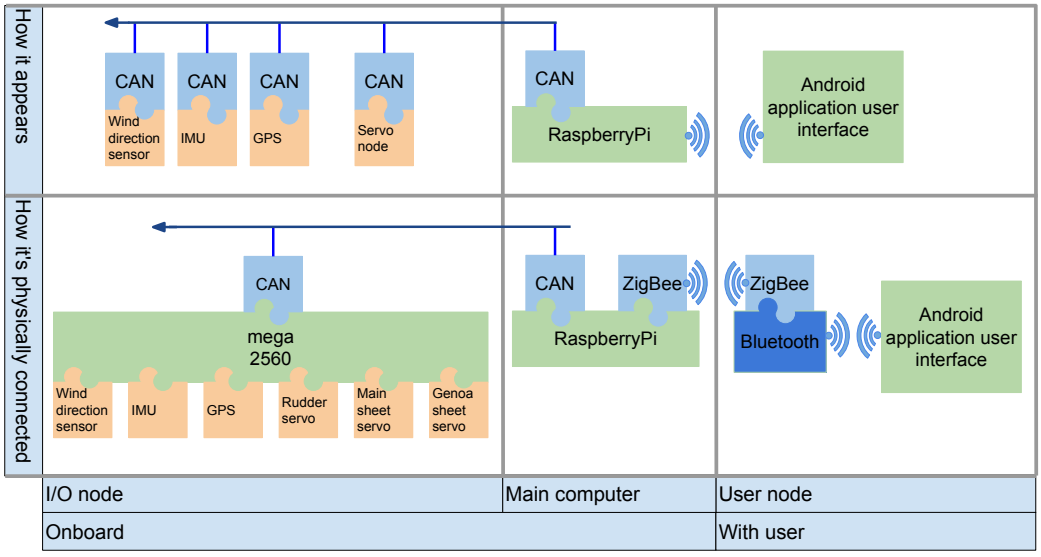


Figure 4.1: Final system overview

4.1 Developing a computerized hardware framework

In this section, the requirements and realization of the hardware framework are addressed.

4.1.1 Requirements

The computerized hardware was developed with the following criteria in mind:

- Enable running a variety of high level control algorithms for sailing/navigation/etc. that requires a higher amount of computational power.
- Enable running a variety of low level control laws prone to real time requirements.
- Manage a variety of sensors and actuators.
- Provide performance results to the user.
- Remain operational and operate safely.

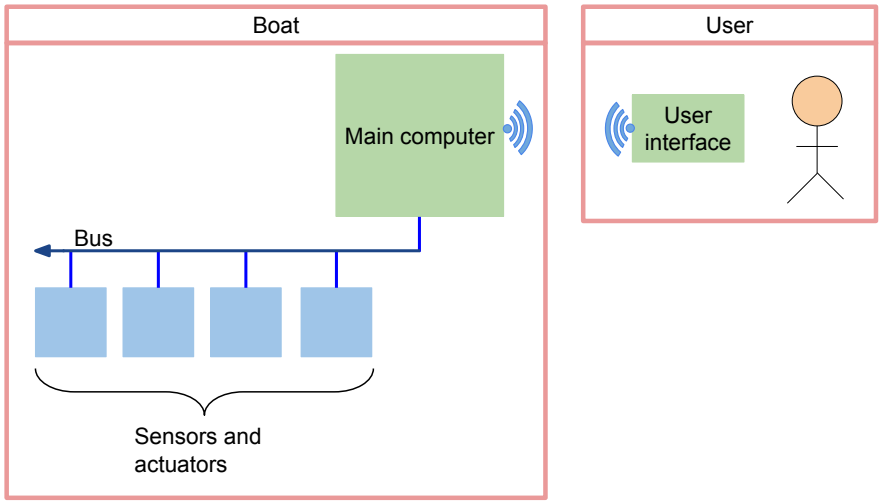


Figure 4.2: Design overview

At that time these criteria were further specified into further requirements:

- The user had to be able to achieve manual control of the system as control algorithms may fail.
- Running the system in semi automatic mode was seen as a key feature to support testing/development of the system. In example it may be desirable that the user controls the sails while the system controls the rudder in early stages of heading control.
- As there are many variables, both internal (algorithm specific) and external (whether/wind etc.), it was seen as important that the user could change variables while the boat was under operation. For example the user may want to tune a regulator during operation.

- At the time, exactly what sensors and actuators that was to become needed were unclear. Hence, it was seen as important that the system was flexible and expandable.
- It was also desirable that the system provided results both in real time and in the form of a log that could be read post testing.
- In order to keep the system maintainable, high modularity was desirable.

This is further discussed in section 8.1.2.

4.1.2 Realization

In order to meet the system requirements, it was decided to use a high level computer running a common operating system. Hence, high level software are available to speed up the implementation process of control algorithms. To achieve a modular, maintainable and expandable system, actuators and sensors were connected as nodes on an internal bus. The main computer would act as a master controlling the other nodes. Figure 4.2 shows an overview of this design. The user interacts with the system through a wireless user interface.

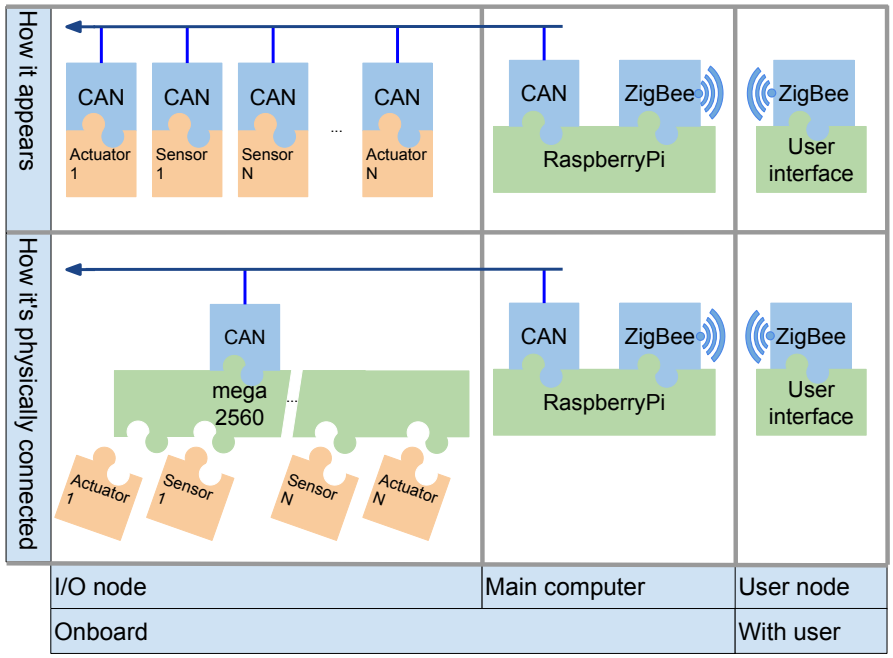


Figure 4.3: System overview

The system in Figure 4.2 was realized using a *Raspberry-Pi* computer running a common *Linux* distribution called *Rasbian*. The Raspberry-Pi computer was expanded with CAN bus. This was done by milling out a circuit board containing a *MCP2515* CAN-driver chip, connecting it to the Raspberry-Pi and writing drivers for CAN-bus functionality. The drivers were also installed as a module within *Python*. Python is a high level programming language taught suitable for the task, this is further discussed in Section 4.1.3.

When implementing the system, there was not enough time to make one node for each sensor or actuator. Hence, it was thought better to make one external node containing a powerful micro-controller with sufficient input/output/external peripherals. This would become the *ATmega2560* micro controller and the node would be referenced as *the I/O node*. Still, it is possible to connect more nodes, but it was taught that the I/O node could handle all the basic sensors and actuators needed.

It was still important to keep the modular concept to achieve maintainability. Hence, basic scheduling was achieved by modifying a port of *freeRTOS* into the ATmega2560. This required more data memory and external RAM was fitted to the I/O node. With basic scheduling it was possible to mimic the behaviour of several nodes without actually having them physically separated.

A proper user interface was not established during the depth study, but the wireless data link was established and the system was proven operational. The data link consists of two radio modules communicating using the *ZigBee* standard. Hence, it is possible for other boats or devices to join the network. However, there is not enough time in this thesis to include any other devices. The radio modules are named *xBee* and manufactured by *Digi*.

Figure 4.3 shows the system overview and Figure 4.4 shows the physical circuits/system.

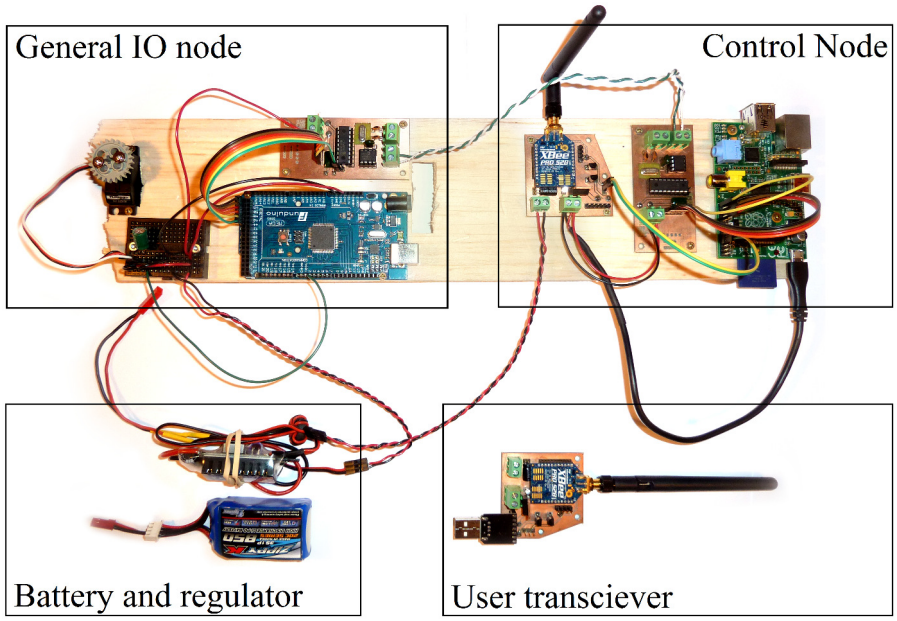


Figure 4.4: Physical system

4.1.3 Software environment

In the previous section it was mentioned that Python was used as the main programming language for software in the main computer. This language provides prototyping of different concepts in a very readable fashion. The language comes with a large standard library. Additionally, other common libraries provide functionality used with control algorithms, such as matrix operations or numerical solvers. Hence, using python shortened the

implementation time.

However, the python language is somewhat ambiguous. The program is interpreted as it runs and basic faults that usually are discovered at compilation time are sometimes not revealed. In example these could be conflicting data type faults or misspelled object names. Hence, as always it is important to test software properly to avid problems later on.

In the I/O node software are written in the C and C++ programming languages. As usual, these languages have been efficient for the hardware near programming.

4.2 Hardware expansions

During the project execution, it became desirable to keep the system simpler in order to save time. Hence, the hardware expansions stated in the next two sections, are consider necessary to achieve autonomous sailing.

4.2.1 Measurements

Three measurements are required:

- Heading, from a magnetometer
- Position, from a GPS module
- Relative wind direction

All these measurements will appear as their own node on the CAN bus.

4.2.1.1 Heading

Heading is estimated from a 3 axis magnetometer. Details on how the heading is estimated can be found in Section 6. An inertial measurement unit (IMU) chip is used to provide the measurement. This chip is called *MPU9250* and includes the 3 axis magnetometer used, the *AK8963*. The magnetometer can not be used by itself as it is strapped down to the boat frame. Hence the 3 axis accelerometer and 3 axis gyroscope can be used for tilt compensation, again see Section 6. These are micro-electromechanical systems (MEMS) and not as accurate as traditional instruments, but sufficient to estimate attitude. These measurements are in 16-bit resolution.

The *MPU9250* chip is connected to the I/O node through the I2C-bus peripheral on the Atmega2560. Driver functions were written to accommodate read and write operations on the MPU9250 registers. After the MPU9250 chip is initiated at startup, measurements are polled by the Atmega2560.

4.2.1.2 Position

Position is measured using a the *LOCOSYS LS20031* GPS module. It is interfaced using standard NMEA sentences. The NMEA standard is a common specification for communication between marine electronics [12]. Hence, the GPS module could be replaced by any other module following the standard, without any changes to the software.

The GPS module is connected to the I/O node through one of several UART peripherals on the Atmega2560. The NMEA standard is very simple. After sending an initializing sentence to the GPS module it returns estimates for position, speed and course at the specified rate.

4.2.1.3 Wind direction

Wind direction is obtained by measuring the angle of a paramagnet connected to a fin exposed to the wind. The measurement is done by the *AS5130* chip. This chip returns the angle of a paramagnet held on top of it through a non-standard serial interface. The measurement is of 8-bit resolution and pulled by the I/O node.

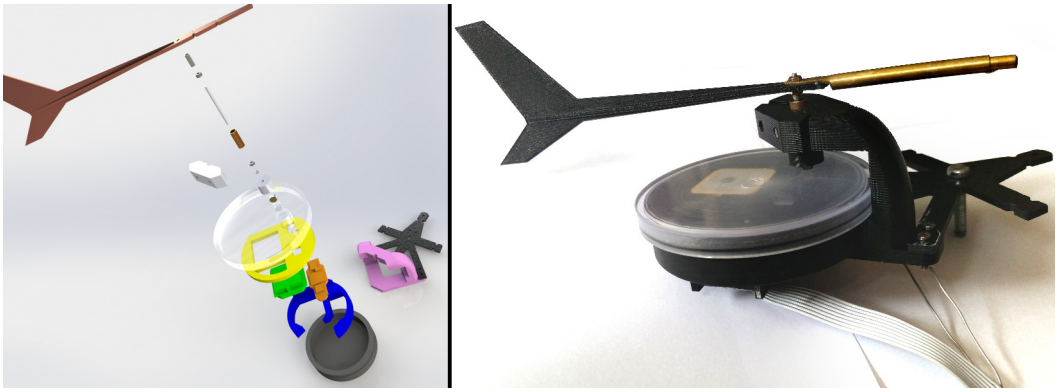


Figure 4.5: Wind direction sensor and GPS housing assembly.

The reason why a AS5130 and a paramagnet is used, is to allow waterproofing of the electronics. Waterproofing is achieved by encapsulating the chip and electronics inside a compartment, while the paramagnet remains on the outside. Figure 4.5 shows the complete wind direction sensor and GPS housing assembly. This assembly is mounted at the top of the mast.

The waterproof compartment containing the AS5130 and the GPS module is located at the bottom of the housing assembly. Within are brackets that holds the electric circuits in place. The compartment consists of a lid and a box. The box has a groove where an o-ring ensures a good seal between the box and the lid. Electrical signal pins were are inserted at the bottom of the box and sealed of. On the outside wires are solder onto these pins. The end of the wire and the pins were further cast into epoxy to ensure waterproofing, see Figure 4.6.

The waterproof compartment was turned out of plastics by the workshop at the Cybernetics engineering department. Other components were 3D printed. In order for the fin to run with low resistance, an axis assembly was manually made out of brass tubes and small

bearings.

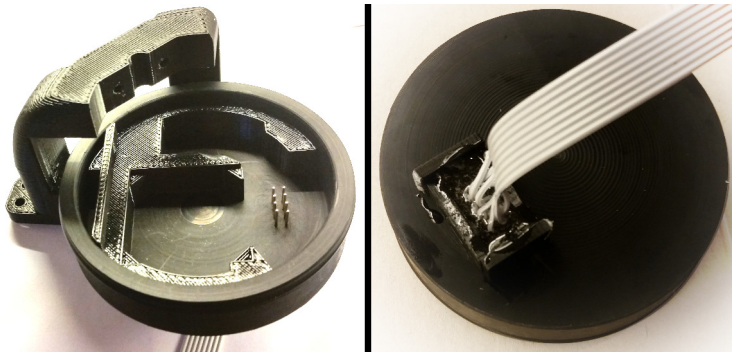


Figure 4.6: Wire seal and brackets for the housing assembly.

4.2.2 Actuators

Given the current boat and set-up, the actuators are as follows:

- A rudder servo
- A main sheet servo
- A Genoa sheet servo

All the servos are simply controlled by PWM signals generated by hardware timers that are a part of the Atmega2560 peripherals. Management of these servos appear as a single node on the CAN bus.

4.2.3 Information flow between hardware nodes and main computer

As indicated by the two previous subsections, the following nodes exists on the CAN bus in addition to the main computer, see Figure 4.1:

- The IMU(inertial measurement) node, that reads the magnetometer and provides the heading estimate.
- The GPS node which provides the position estimate.
- The wind direction sensor node which provides the relative wind direction.
- The servo node that executes set points stated by the main computer.

As mentioned in, the different nodes appear as they are physically separated on the CAN bus. This keeps the system modular and maintainable. Thus the nodes communicate through message passing, not shared memory. The trade-off is that this creates more overhead.

Figure 4.7 illustrates how a typical node on the can bus communicates with a typical worker thread on the main computer. However, this is equivalent for any two nodes communicating, but typically the nodes only talks to an entity within the main computer.

The address space on the CAN bus is 11 bits, thus quite large. Hence addresses are also assigned to functions or commands. In other words, nodes may have multiple addresses and multiple subscriptions.

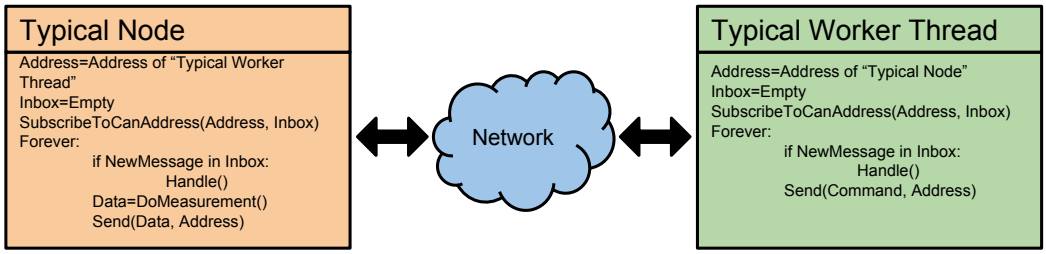


Figure 4.7: Typical communication between nodes on the CAN bus and entity in the main computer

4.3 Logging and benefits of having a log

Data logging is extremely important when building a prototype. Some of the greatest benefits of having the data log are the following:

- Lets the user analyse measurements and sometimes determine their quality.
- Lets the user evaluate the performance of control algorithms.
- Helps determining errors and bugs in the system.
- Makes it possible to simulate functions and algorithms with recorded data as input.

4.3.1 Implemented log types

The current implementation has three log types:

- *The standard error log*: The standard error stream is the output stream where a computer program outputs error messages or diagnostics. [19] In this case that would be hard errors like assignments from non-existing variables, segmentation faults and so on. These errors usually forces the program to an end. Thus, the information in the standard error stream is very useful as it would point to the line of code where the fault occurred while giving a hint to why. The standard error is logged by redirecting it to a new file when the boat is out for testing.
- *The data log*: The data log is defined specifically for the current project. From anywhere in the code, data may be queued to the data log. The data is eventually stored from the queue in a specific format with a specific time stamp. All measurements,

incoming/outgoing messages and other vital data is logged here. Each time the program starts a new data log file is created. Data from the data log are easily read post runtime by computer scripts to generate graphs or other visualizations.

- *The standard out log*: The standard output is the stream where a program writes its output data. [19] Often the standard output is directed to the console and the user can read it as plain text. This is a quick and dirty way of outputting data and errors in a less structured fashion when building a program. Thus in the same fashion as with the standard error, the standard out is redirected to a new file when the boat is out for testing.

Use of the log can be seen in Chapter 7, where graphs and illustrations of data have been generated to emphasize results and problems.

Chapter 5

User Interface

In this chapter, the user interface is discussed. This includes selecting an appropriate form of user interface and how the user interacts with it. How the user interface is integrated in the system and how information flows from the user interface is already covered in Chapter 4.

5.1 Choosing an appropriate form of user interface

In order to conduct test runs in an orderly and efficient fashion it is necessary to have some kind of user interface. Before setting up the user interface, three different designs were considered:

- A standard R/C radio, with a "go to manual" switch
- A laptop computer with a command line or graphical interface
- A smart phone with a graphical interface

R/C radio: The R/C radio was the simplest, but least sophisticated user interface considered. By putting the receiver end of the R/C radio system inside the boat, the outputted PWM signals could be read by the computer in the boat. The advantages of such a solution are:

- Simplicity and robustness as there are few likely faults that may occur.
- Low latency/good responsiveness as the R/C radio is designed this type of operation.

The disadvantages of such a solution are:

- The only information carried are the "go to manual"-signal and the servo positions for manual mode. Thus all user inputs to algorithms and tuning parameters must be set prior to launch into water. Thus whenever changes are necessary, the boat must

be brought on shore and reprogrammed with a computer. This can be problematic as hatches and compartments must be opened to get to the electronics, thus increasing the risk of getting water onto the circuits.

Laptop computer: A data link could be installed in the boat allowing for communication with a laptop computer at test cite. The advantages of such a solution are:

- Live parameter tuning and continuous user input for algorithms
- Feedback to user and the user can monitor the boat's subsystems

The disadvantages of such a solution are:

- The solution is more complex and impose greater risk of faults.
- The data link imposes more latency and less responsiveness.
- A computer is bulky and not suited to be played around with in the nearby environment of water.

Smart phone: In the same fashion as with the laptop, the data link could be connected with a smart phone. The advantages of such a solution are:

- Live parameter tuning and continuous user input for algorithms
- Feedback to user and the user can monitor the boat's subsystems
- The smart phone is well suited for testing as it is easily brought and handled in the nearby environment of water as it is small and waterproof.
- The operating system in such devices offers an application interface that makes it easy to utilize graphics, touch inputs, wireless connectivity and many more. Thus it is a fast and powerful way of realizing a proper graphical user interface.

The disadvantages of such a solution are:

- The solution is even more complex as it requires the data link to be connected to the phone wirelessly in the lack of wired connectivity.
- In the smart phone case, the data link imposes even more latency and less responsiveness.

It was decided that the smart phone solution was more desirable than the other options. This is further discussed in Chapter 8.

5.2 Tab-based user interface running on smart phone

Hence, the boat now has a tab-based application running on the Android^{TM}-platform in a Samsung S5^{TM} device. New tabs are easily integrated into the application as they are needed. Thus the application are modular and expandable.

The following tabs are integrated:

- Map and measurements
- Manual controls
- Signal strength and battery status
- Debugging

The user selects tabs by clicking the respective icon on the icon bar. If the iconbar is full, clicking the righmost icon will list tabs not shown. This is illustrated in Figure 5.1. In the following the different tabs are described.

Figure 5.1: List menu.

5.2.1 Map and measurements

Purpose The purpose of the map tab is to help the user evaluate the quality of control algorithms and measurements. It does so by displaying latest and/or historical measurements in an easy-to-understand fashion.

Map In the background of the *"Map and measurements"* tab, the current map is displayed. The user may select an appropriate map by clicking the globe symbol in the down right corner. When doing so, the application displays a list of available maps. The background map is usually chosen such that it corresponds to the map used by the path planner within the boat itself. See Section 6.4.3.2. It is possible to scroll and zoom on the map.

Location The circular location symbol is centred around the location measured by the on board GPS module relative to the current map displayed. In its center the boat symbol is drawn.

Figure 5.2: Map tab.

Heading The boat symbol within the location symbol is rotated to match the estimated heading. Furthermore a red line extends in the heading direction to further clarify this estimate.

Course The estimated course is illustrated by a green line extended from the location in the direction of the course.

Roll In the top left corner the estimated roll is drawn by a gauge mimicking the boat seen from behind.

Pitch In the top right corner the estimated pitch is drawn by a gauge mimicking the boat seen from starboard side.

Wind direction and "in-irons" zone A green arrow indicates the estimated wind direction. It points parallel to- and from- the wind towards the boat location. Behind the green arrow the "in-irons" zone is indicated.

5.2.2 Manual controls

Purpose The purpose of the "Manual controls"-tab is to enable the user to safely control the boat manually, thus overriding any control algorithms running.

Go to manual buttons The "Rudder manual"- and "Sails manual"- buttons activates manual control for respective items. Control algorithms continue running in the background in the boat's main computer regardless. When a button is unchecked, control is passed back.

When a button is clicked, its background color turns red indicating that a "request control" message is sent. Only when a respective acknowledgement is received from the

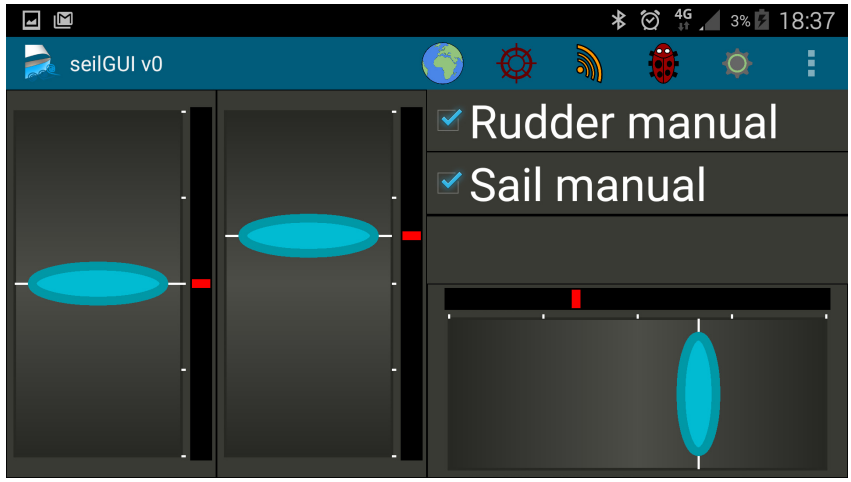


Figure 5.3: Manual mode tab.

boat, the background color is changed back. Thus the user will in all likely-hood know if an error is preventing manual control.

Set-point sliders The set-point sliders lets the user set the manual control input. Set-points are transferred to the boat when the slider is touched, untouched and continuously while being held down/moved. The sliders cover a large screen/touch area, thus it is easy for the user to use the sliders while not looking into the screen.

The left-most vertical slider corresponds to the main sail sheet. Setting it downwards correspond to pulling in the main sheet resulting in a low main sail angle. The midle slder, which is also vertical, corresponds to the Genoa sheet in a similar fashion. The right horizontal slider corresponds to the rudder.

Set-point indicators Each slider integrates a set-point indicator, seen as a red dot on a black strip next to it. This indicator displays the latest set point acknowledged by the boat. This indicator helps unveil errors.

Port side tack button The *Port side tack button* selects which side of the boat the Genoa slider corresponds to. If selected, the right or port side Genoa sheet is controlled by the slider. The starboard or left Genoa sheet is then placed at the outermost position, thus not affecting the Genoa sail.

5.2.3 Signal strength and battery status

Purpose The purpose of the *Signal strength and battery status*-tab is to enable the user to observe the signal strength on the data link. Thus, the user can avoid running the boat outside the range of the data link and get an impression of how well the data link performs.

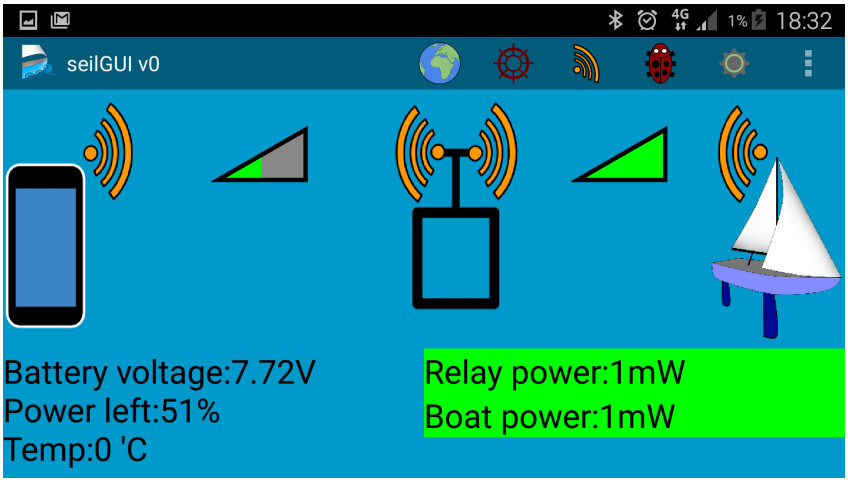


Figure 5.4: Signal strength and battery status tab.

Left signal strength indicator The left signal strength indicator indicates whether the smart phone is connected to the data link relay station, as described in Section 4. A red cross indicates that no connection is established. Otherwise signal strength is shown as in Figure 5.4.

Right signal strength indicator The right signal strength indicator indicates whether there is a connection between the relay station and the boat, as described in Section 4. This is the most relevant indicator of the two, as the distance between the relay station and the smart phone usually is invariant.

Battery voltage The battery-voltage of the boat battery is stated as text in the bottom left corner. The estimated remaining energy is also stated as a percentage value. This value is only correct when using a 2 cell lithium-polymer battery.

Transmission power As the radio modules may suffer from saturation when they are close to each other, it was necessary to be able to adjust the transmission power. In this way, the radio modules can be used on the desktop next to each other as well as at test cite with large distances. When clicking the *Relay power* or *Boat power* text, a list appears with possible options. The user interface will automatically retransmit the commands necessary to ensure that the correct power settings are achieved. At this moment, the text background turns green to indicate success, see Figure 5.5

5.2.4 Debugging

Purpose The purpose of the *Debugging* tab, as the name indicates, is used for debugging and testing. Hence it includes usful functions that does not need to be spread around on different tabs.

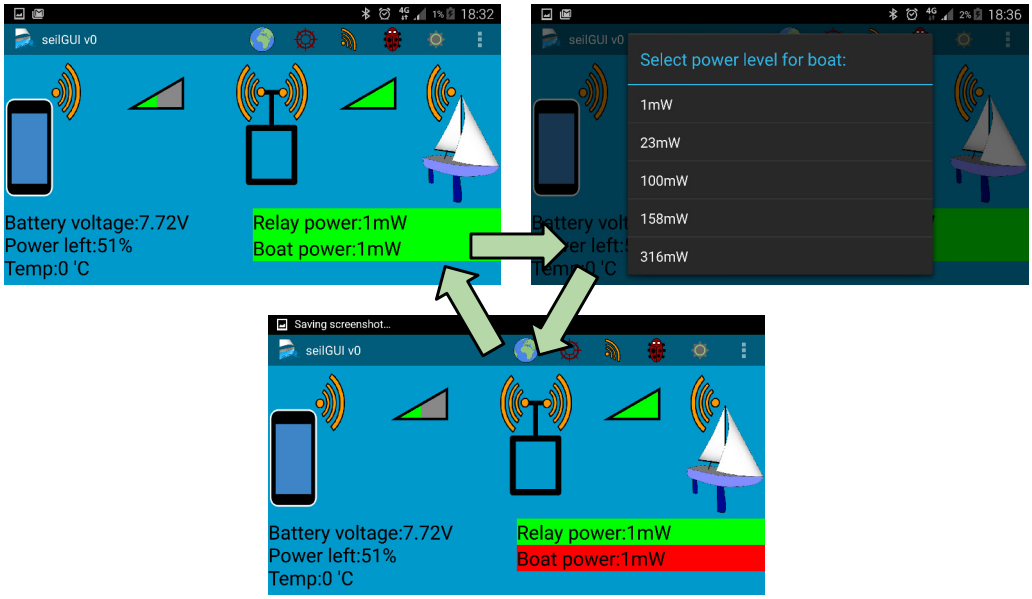


Figure 5.5: Setting the transmission power.

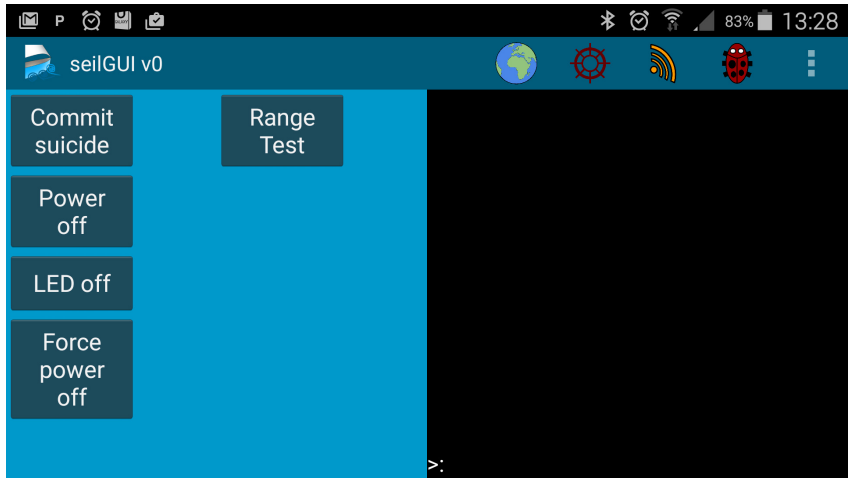


Figure 5.6: Debug tab.

Commit suicide button When pushing the *Commit suicide*-button, a command is sent to the main computer telling git to exit the running program. On success, this leaves the boat useless as nothing useful is running on the main computer, but the program exits normally.

Power off button When pushing the *Power off*-button, a command is sent to the main computer telling git to exit the running program and shut down the computer normally. This command is useful when a test is complete and the user wants to turn off the boat at test cite.

Force power on/off button When pushing the *Force power off*-button, a command is sent to the radio module in the boat, which cuts power to all logical circuits including the main computer. Unless the main computer is already turned off normally, there is a risk that files can become corrupt during such a power loss. However, this is a useful feature if there is an error in the program causing it to hang. In such an event the power can be cycled, thus restarting the main computer. The program start automatically at start up.

Print window On the right side there is a print window. The main computer in the boat may send short messages to be displayed in this window.

5.2.5 Settings

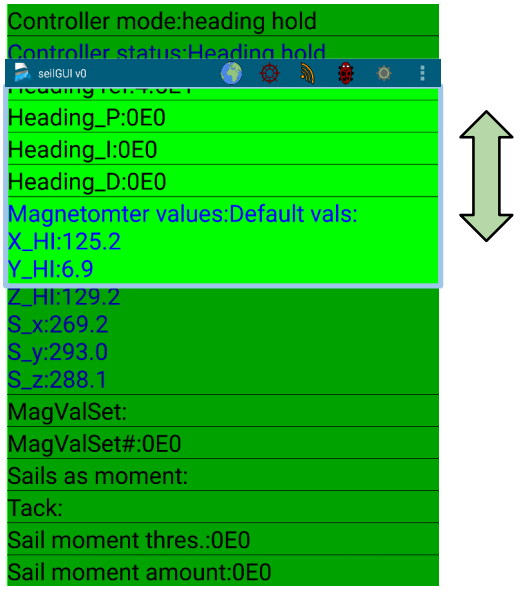


Figure 5.7: Settings tab with example variables

Purpose The purpose of the settings tab is to give the user the ability to turn on/off or modify features in real time while testing the boat. An example of this is when the user wants to test and obtain tuning variables. The setting tabs consist of a scrollable list of synchronized variables, see section 5.3.1.1. An example is given in Figure 5.7.

There are three types of variables in the list:

- Display: A display variable is set by the main computer and viewed in the settings list. These variables have blue text, see Figure 5.8.
- Number: A number variable is set by the user when it's clicked, see Figure 5.9 The main computer uses the number variable as any other variable and it may be changed at any time. The variable may be an integer or a floating point and is displayed with scientific notation.

- Switch: When the user clicks a switch variable, a list is presented on which the user can select one of the items. See Figure 5.10. Changing the item triggers an event in the main computer. If the variable is created with no selectable items, it act as a button.

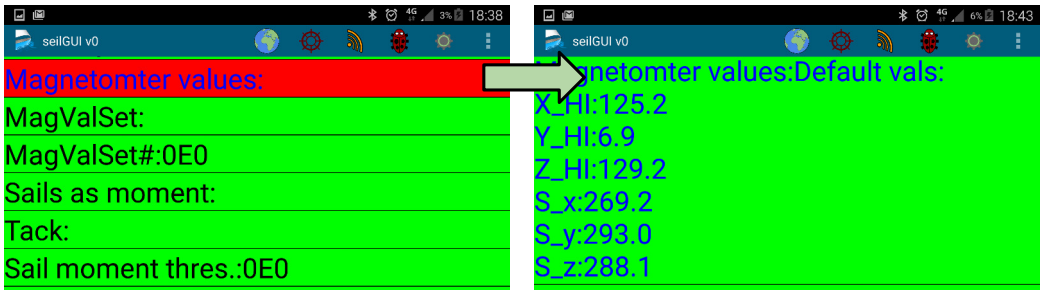


Figure 5.8: Synchronized display variable in settings tab. These have blue text and the background remains red until they are set for the first time by the main computer.

Figure 5.9: Synchronized number variable in settings tab. When clicked a number can be typed in and the background remains red until the variable becomes synchronized.

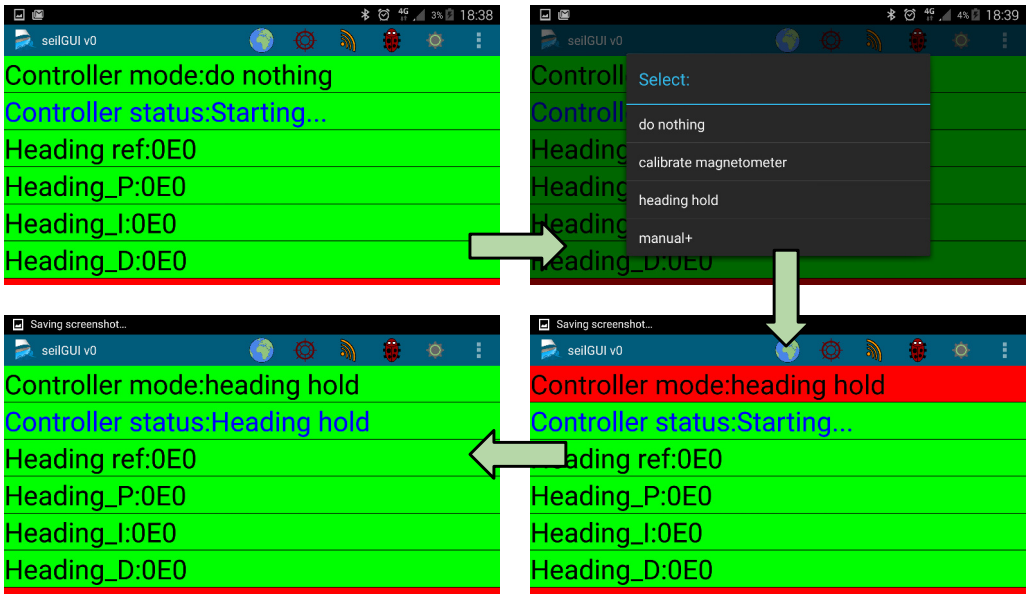


Figure 5.10: Synchronized switch variable in settings tab. When clicked a selection can be made and the background remains red until the variable becomes synchronized.

5.3 Integrating the smart phone application user interface

From Section 4 recall that a data link with long range already is established. This data link connects to a computer through a serial connection. On a the smart phone, there is no such wired connection available. On the up side the smart phone has several possible wireless interfaces. Except from the cellular network, neither of these connections offers the range required to connect to the boat. Thus two options remained, either equipped the boat with cellular network or relay the existing data link to a smart phone compatible wireless connection.

Generation	Latency
2G	300-1000 ms
3G	100-500 ms
4G	<100 ms

Table 5.1: Cellular network latency [28].

The following arguments weighted the decision between the two options:

- Latency on celluar networks is usually high, depending on coverage, see table 5.1.
- Cellular network may not be available at all test cites.
- A Bluetooth development kit from Nordic Semiconductor was already available, which included example code to forward a wired serial connection onto an android application via Bluetooth.

The decision fell on making a relay station. This will be further discussed in Section 8. Figure 5.11 illustrates the concept of the relay station.

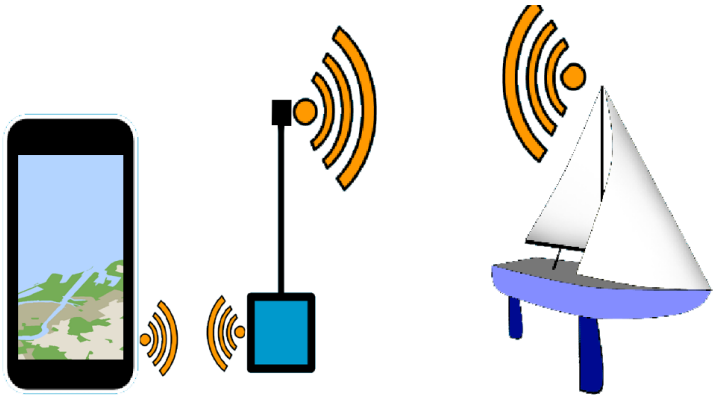


Figure 5.11: Connectivity between device and boat.

5.3.1 Information flow between the boat and user interface

The example code from the bluetooth developtment kit was modified such that the smart phone automatically would connect to the Bluetooth card. Furthermore the Bluetooth software was hidden away from the rest of the program, hence the rest of the program only see the xBee/ZigBee/radio module interface.

To further simplify the system, the radio interface is also hidden away from the rest of the program at the next level, with two exceptions:

- Tabs may request the signal strength
- Tabs may control output lines on the radio module located in the boat. In example this is used to cycle power for circuits in the boat.

Apart from these exceptions tabs only have the option to send or receive strings of data. Figure 5.12 illustrates this concept and how messages actually propagates between the user interface and the main computer. The data strings are colon separated into words, the first word indicates the address or keyword. Tabs in the user interface and worker threads in the main computer subscribes to such keywords in a similar fashion as CAN nodes subscribed to addresses, recall Figure 4.7.

5.3.1.1 Synchronized variables

In order to further simplify implementation of control algorithms, classes to implement shared synchronized variables between the boat and user interface were made. Hence, a simple *"one-liner"* creates a variable that automatically synchronizes by the means of acknowledgements, retransmissions and congestion control. On the user interface end, these variables have graphical views that automatically update. Some of these are click-able

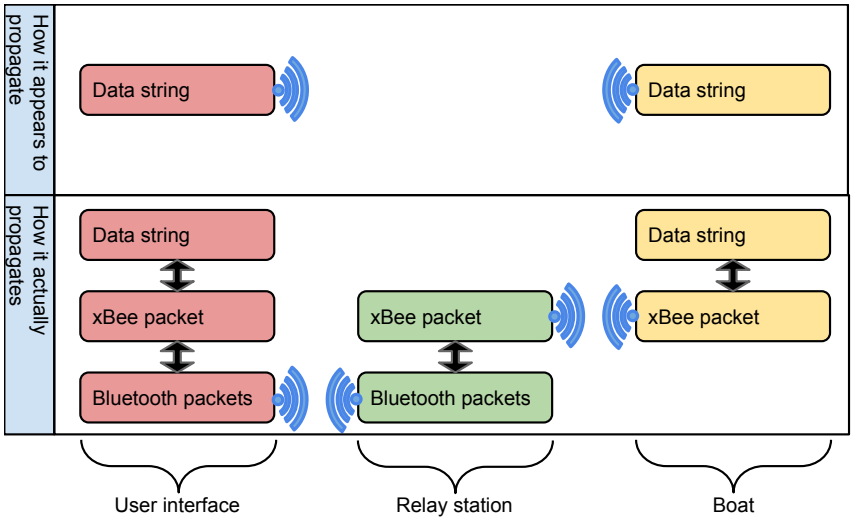


Figure 5.12: Message propagation between user interface and main computer

which allows the user to set the variable during sailing. Others are only viewable in order to provide information to the user, see Section 5.2.5.

In Chapter 5 different user interfaces are discussed. In this section, integration of the chosen user interface is discussed. The chosen user interface is a tab-based Android application running on a smart phone, see Chapter 5 and 8.

Chapter 6

Guidance, Navigation and Control

6.1 Introduction to guidance, navigation and control

Guidance, navigation and control(GNC) deals with the design of systems that automatically control or remotely control devices or vehicles that are moving under water, on the surface or in space. [26] Such a system is called a GNC system. Usually, guidance, navigation and control makes up three independent subsystems. These subsystems interact through data and signal transmissions. This typical interaction is illustrated in Figure 6.1.

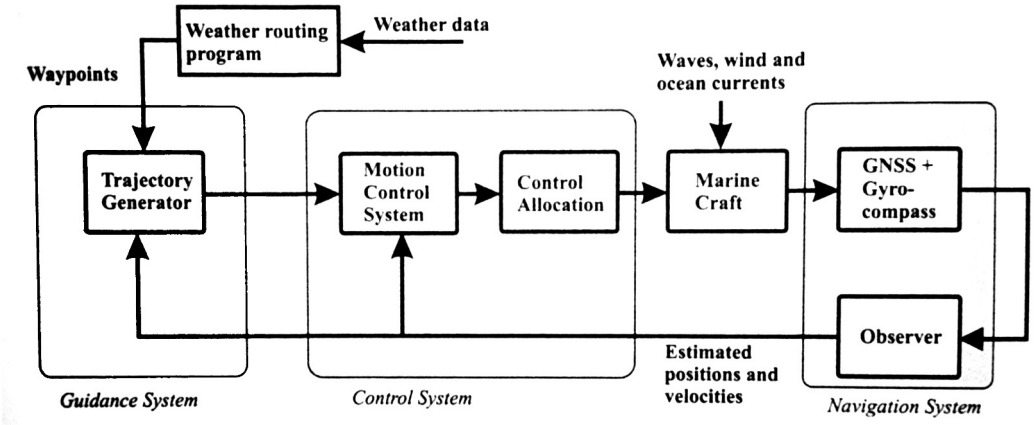


Figure 6.1: Interaction between guidance, control and navigation blocks. Figure courtesy to [26].

Guidance is the action of computing the desired position, velocity and acceleration.[26] In example,this is the equivalent of using a map to determining whether to go left or right in the next junction when driving a car.

Navigation is the action of determining the position/attitude, course, distance travelled and so on. Often measurements are inexact and prone to noise, thus estimates are made.

Control is the action of determining the necessary control forces and moments to be provided by the craft in order to satisfy the control objective.[26] The control objective is usually seen in conjunction with the guidance system such as trajectory tracking or path following. In example, this is equivalent to determining how to move the steering while biking, in order to avoid falling over while following the curvature of the road.

6.1.1 Reference frames

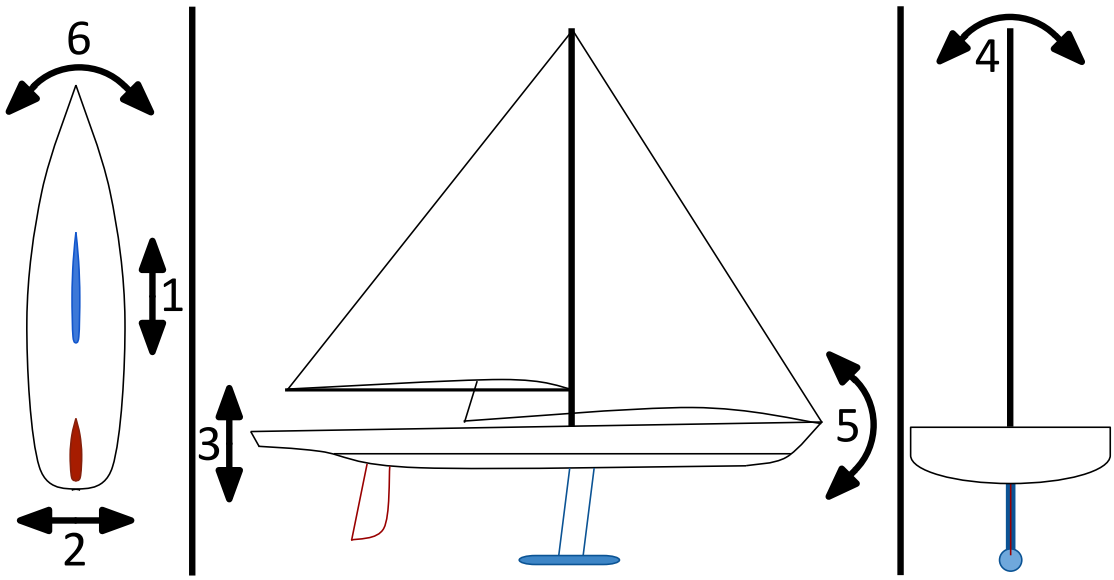


Figure 6.2: Degrees of freedom, numbers correspond to Table 6.1

Table 6.1: The notation for marine vessels, following [26].

DOF		Forces and moments	Linear and angular velocities	Positions and Euler angles
1	Motions in the x direction (surge)	X	u	x
2	Motions in the y direction (sway)	Y	v	y
3	Motions in the z direction (heave)	Z	w	z
4	Rotation about x axis (roll)	K	p	ϕ
5	Rotation about y axis (pitch)	M	q	θ
6	Rotation about z axis (yaw)	N	r	ψ

For the discussion in the following sections to make sense, some notations and definitions regarding reference frames are needed. The notations and definitions are from [26].

ECEF The earth-centred-earth-fixed reference frame has it origins fixed in the center of the earth. The frame rotates with the earth, but for this discussion it can still be considered

inertial as the effects of the rotation are neglectable. The positions measured by the GPS module are given in the ECEF frame as longitude and latitude:

$$\Theta_{en} = \begin{bmatrix} l \\ u \end{bmatrix} \tag{6.1}$$

NED The second reference frame we need to be concerned of, are the North-East-Down frame. As the name indicates, the first axis points north, the second east and the third down. Hence, the north axis, the east axis the origin defines a plane tangent to the surface of the earth. When the boats position are compared to positions of waypoints this is usually with respect to the NED frame. This is indicated by the superscript n:

$$p^n = \begin{bmatrix} N \\ E \\ D \end{bmatrix} \tag{6.2}$$

BODY The body frame is the final frame to consider. The origin of the body frame is defined somewhere within the boat, or relative to the boat, as CO. The x-axis is aligned from aft to fore, the y-axis are directed towards starboard and the z are defined from top to bottom. Hence, positive rotations follows accordingly by the right hand rule, see Figure 6.3.

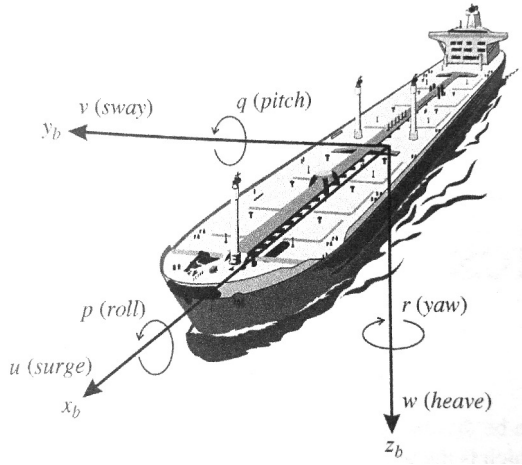


Figure 6.3: The body frame. Figure courtesy of [26].

6.2 Navigation

In order for the control algorithms to function, it is neccesarry to obtain the following information during sailing:

- Attitude
  - Roll
  - Pitch
  - Heading
- Position
- Speed
- Course
- Wind direction

This section describes how this is done.

6.2.1 Roll and pitch

If the gravitational vector is known in the boat coordinate frame, the exact roll and pitch can be obtained in the following way:

$$\phi = atan2(g_y, g_z) \tag{6.3}$$

$$\theta = atan\left(\frac{-g_x}{g_z}\right), \tag{6.4}$$

where $\phi \in (-180°, 180°)$ is the roll angle, $\theta \in (-90°, 90°)$ is the pitch angle and g is the gravitational vector. $atan2(a, b)$ is the same as $atan\left(\frac{a}{b}\right)$, except the angle is returned in all four quadrants. When $\theta \approx 90°$, the ϕ expression becomes singular and is not valid. However, this will never happen during normal operation.

Obtaining a measurement of the gravitation vector is not trivial. The accelerometer does not distinguish between the gravitation vector and forces due to acceleration. As long as the boat is not subject to acceleration the accelerometer measurement can be used directly as the gravitational vector estimate. When the boat sails in calm waters the acceleration is typically not significant in comparison to the gravitation. Still, the following complimentary filter has been used to make the measurement more robust to higher accelerations over short periods of time:

$$\phi_f = (1 - \alpha)\phi_f + \alpha\phi \tag{6.5}$$

$$\theta_f = (1 - \alpha)\theta_f + \alpha\theta \tag{6.6}$$

$$\alpha = e^{-\frac{||g_m| - |g||\delta}{\tau}}, \tag{6.7}$$

where ϕ_f and θ_f are the filtered versions of ϕ and θ, $|g_m|$ is the magnitude of the accelerometer measurement, $|g|$ is the known magnitude of the gravitational vector, δ is the time between samples and τ is a filter tuning parameter.

6.2.2 Heading

A heading estimate is critical for achieving autonomous sailing. This Section covers how this heading estimate is made, by measuring the Earth's magnetic field relative to the boat's attitude.

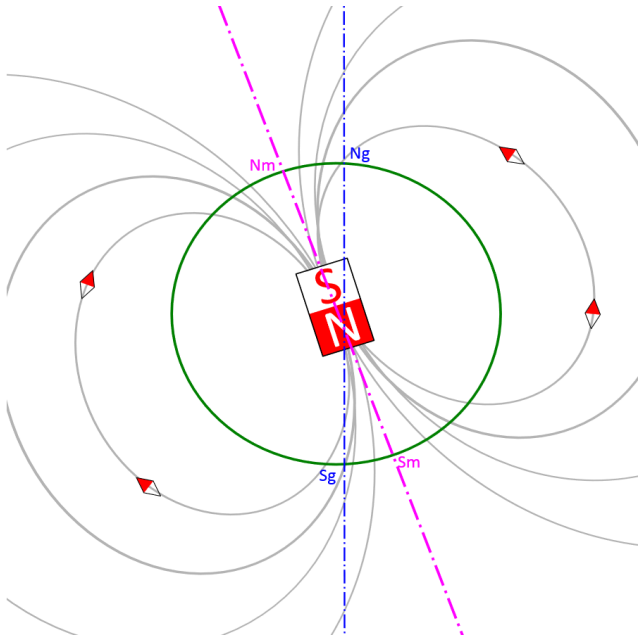


Figure 6.4: Approximation of Earth's magnetic field. Figure courtesy of [3].

Near the surface of the Earth, the Earth's magnetic field can be closely approximated by the field of a magnetic dipole positioned at the center of the Earth [3]. This magnetic dipole is offset by 10 °with respect to the rotational axis of the Earth. Figure 6.4 illustrates this approximation.

When not being located at the magnetic poles, we can see from figure 6.4 that the Earth's magnetic field has a component parallel to the surface pointing towards the north pole. The following describes how this component can be measured and used to achieve the heading estimate.

From Chapter 4, recall that a 3 axis magnetometer is fitted to the system. Ideally, this could be used to measure the Earth's magnetic field perfectly. However, there are several problems[1] [6]:

- Hard iron interference
- Soft iron interference
- Time varying magnetic fields.

- Attitude interference
- Misalignment
- Scale factor

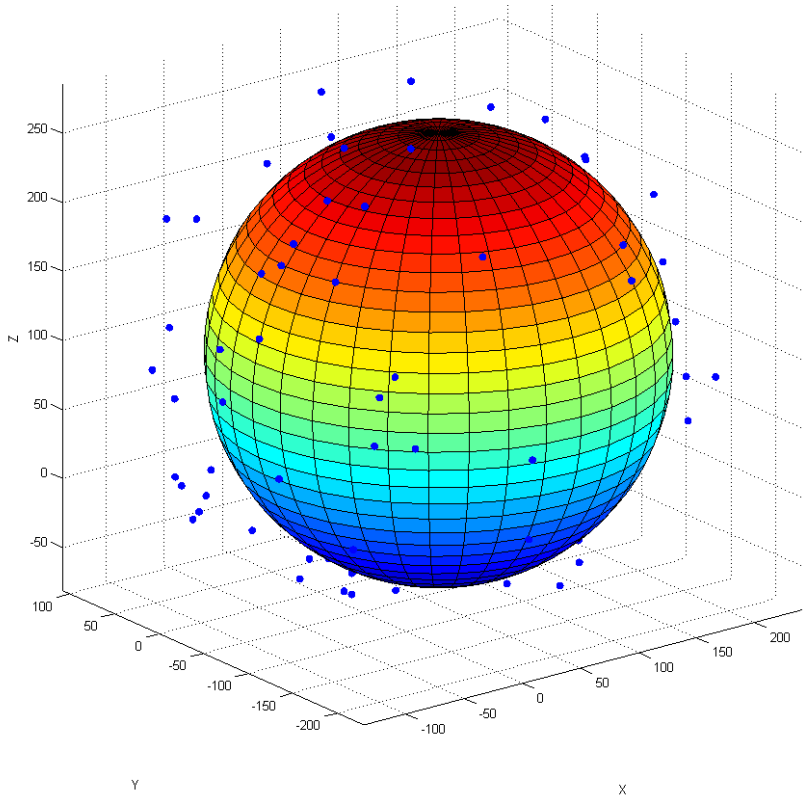


Figure 6.5: Raw magnetometer data when the magnetometer is rotated to random attitudes.

Hard iron interference The hard iron interference is the presence of other permanent magnetic fields. These are superimposed to the measurement and can be removed by a simple bias.

Soft iron interference The soft iron interference is the presence of materials that affect the magnetic field. These materials loose their magnetic property when the surrounding magnetic field is removed.

Time varying hard iron interference Time varying hard iron interference is the presence of other time varying magnetic fields. In this application this will typically by magnetic fields set up by electric motors in the boat's servos or high current flowing through power cables. These effects are hard to remove. The best precaution is to place the magnetometer as far away from such fields as possible.

Attitude interference Attitude interference are caused by uncertainties in the attitude and hence not correctly decomposing the measurement onto the parallel-to-surface plane.

Misalignment The magnetometer is made up of three independent sensors. These sensors measures the surrounding magnetic field in their respective direction. Ideally these three sensors are perfectly orthogonal, but in reality in-orthogonality causes inaccuracies. These inaccuracies are negligible compared to hard and soft iron interference.

Scale factor As mentioned, the magnetometer is made up of three independent sensors. These sensors are ideally equal in every way, but in reality they may have different sensitivity or scale. These inaccuracies are negligible compared to hard and soft iron interference.

Figure 6.5 shows raw magnetometer readings as the magnetometer is rotated in random directions. Ideally, the measurements make up an ellipsoid with radius equal to the field strength. The coloured ellipsoid in Figure 6.5 illustrates that the raw data actually makes up an ellipsoid. The hard iron interference is clearly visible as the center of the ellipsoid is not located at the origin. If the soft iron interference is significant, the result would be that the ellipsoid would become tilted.

6.2.2.1 Magnetometer/compass calibration

It is critical that the magnetometer is calibrated to compensate for hard iron interference. Furthermore, it is important that this is done after the magnetometer resides at its final location within the boat and all possible hard/soft irons are mounted in their final position.

In the further, it is assumed that soft iron interference won't tilt the ellipsoid significantly. This may lead to lee accurate heading estimates. Misalignment is also neglected. In the following, a least-squares method is presented to obtain hard iron offsets and scale factors(hence some correction to soft iron interference) [1]. This allows semi-automated compass calibration immediately upon launch.

Given the assumptions, measurements satisfies the following ellipsoid equation:

$$R^2 = \frac{(X - X_{HI})^2}{A} + \frac{(Y - Y_{HI})^2}{B} + \frac{(Z - Z_{HI})^2}{C}, \tag{6.8}$$

where $R > 0$ is the field strength, $A > 0$, $B > 0$ and $C > 0$ are the semi-principal axes of the ellipsoid, X, Y, and Z are the coordinates of a measurement, and X_{HI}, Y_{HI} and Z_{HI} are the hard iron offsets.

Eq. 6.8 can be rewritten in the following matrix form:

$$\mathbf{Ax} = \mathbf{B}, \tag{6.9}$$

as:

$$\mathbf{A} = \begin{bmatrix} X & Y & Z & -Y^2 & -Z^2 & 1 \end{bmatrix} \tag{6.10}$$

$$\mathbf{x} = \begin{bmatrix} 2X_{HI} \\ \frac{A^2}{B^2} 2Y_{HI} \\ \frac{A^2}{C^2} 2Z_{HI} \\ \frac{A^2}{B^2} \\ \frac{A^2}{C^2} \\ A^2R^2 - X_{HI}^2 - \frac{A^2}{B^2} Y_{HI} - \frac{A^2}{C^2} Z_{HI} \end{bmatrix} \tag{6.11}$$

$$\mathbf{B} = \begin{bmatrix} X^2 \end{bmatrix}. \tag{6.12}$$

In order to solve for $\mathbf{x}$, several measurements are needed. More measurements can be integrated into the matrix equation in the following way:

$$\mathbf{A} = \begin{bmatrix} X_1 & Y_1 & Z_1 & -Y_1^2 & -Z_1^2 & 1 \\ \vdots & & & & & \\ X_n & Y_n & Z_n & -Y_n^2 & -Z_n^2 & 1 \end{bmatrix} \tag{6.13}$$

$$\mathbf{B} = \begin{bmatrix} X_1^2 \\ \vdots \\ X_n^2 \end{bmatrix} \tag{6.14}$$

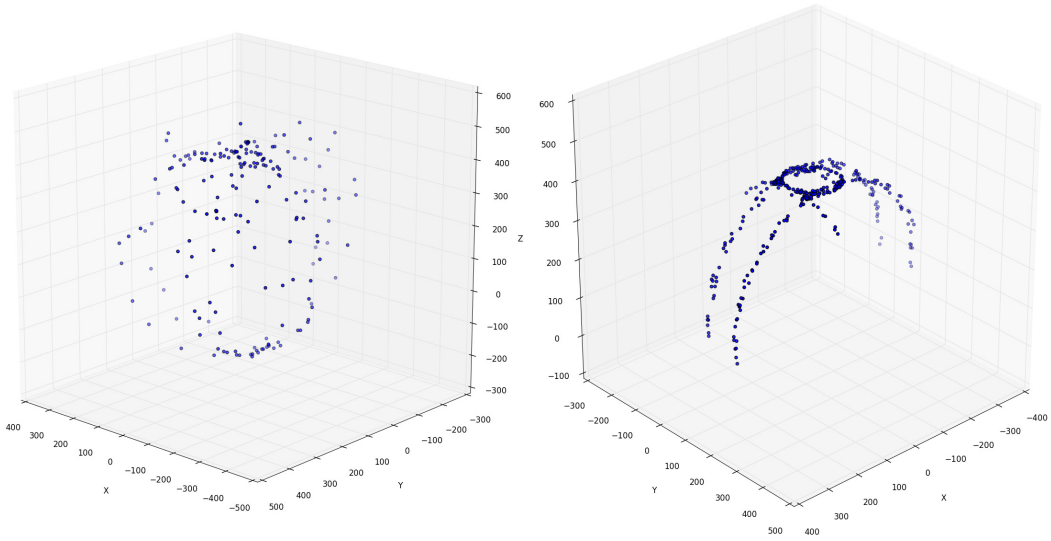


Figure 6.6: Left: Near uniformly distributed magnetometer data set. Right: An easily obtainable magnetometer data set. A full circle in yaw can be seen, and partial turns in roll and pitch.

Solving the equation exact with 6 measurements does not yield satisfactory accuracy as measurements are prone to noise. To achieve the best possible results, the magnetometer should be turned uniformly in all directions while sampling a large amount of measurements. As it is difficult to turn the boat upside down, the sample set used in the final calibrations

will not be perfect. However, by doing full rotations in the yaw direction, and half rotations in the roll and pitch directions, a satisfactory set could still be achieved. Figure 6.6 shows the difference of such data sets.

The equation is solved using the least squares method:

$$\mathbf{x} = [\mathbf{H}^T\mathbf{H}]^{-1}\mathbf{H}^T\mathbf{B}. \tag{6.15}$$

After $\mathbf{x}$ is known, the hard iron offsets are given by:

$$X_{HI} = x_1/2 \tag{6.16}$$
$$Y_{HI} = \frac{x_2}{2 * x_4} \tag{6.17}$$
$$Z_{HI} = \frac{x_3}{2 * x_5}, \tag{6.18}$$

where x_i is element i in $\mathbf{x}$.

The scale factors can be obtained from:

$$scale_x = \sqrt{A} = \sqrt{x_6 + X_{HI}^2 + x_4 Y_{HI}^2 + x_5 Z_{HI}^2} \tag{6.19}$$
$$scale_y = \sqrt{B} = \sqrt{\frac{A}{x_4}} \tag{6.20}$$
$$scale_z = \sqrt{C} = \sqrt{\frac{A}{x_5}}. \tag{6.21}$$

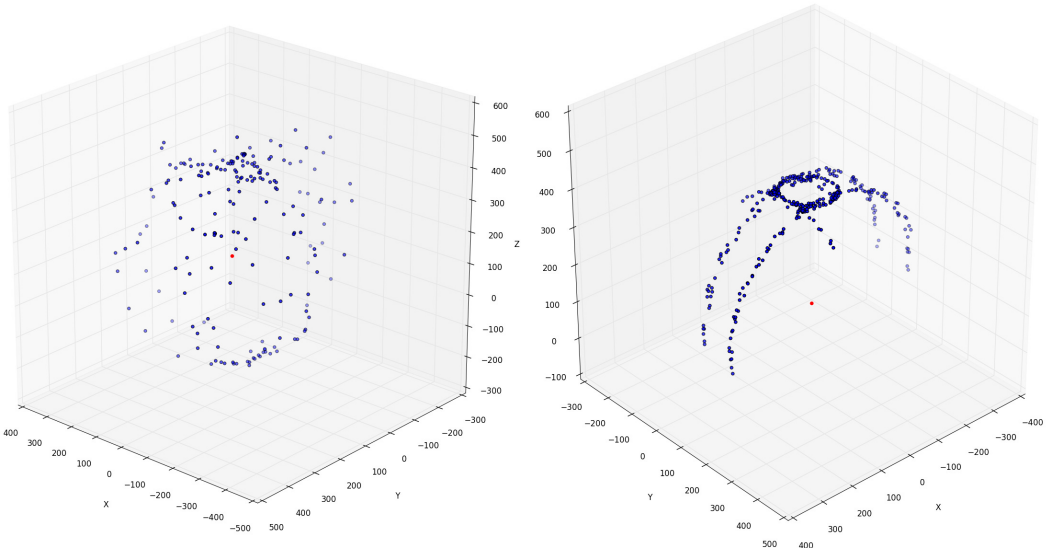


Figure 6.7: Indicated in red is the approximate ellipsoid center. The data sets are the same as in Figure 6.6.

Figure 6.7 illustrates that the algorithm obtain the ellipsoid offset for both uniformly and partially distributed sets.

After calibration is successfully performed, scale factors and offsets can be used to transform measurements,hence removing hard iron interference and some soft iron interference, in the following way:

$$\begin{bmatrix} X_{calibrated} \\ Y_{calibrated} \\ Z_{calibrated} \end{bmatrix} = \begin{bmatrix} \frac{1}{scale_x} & 0 & 0 \\ 0 & \frac{1}{scale_y} & 0 \\ 0 & 0 & \frac{1}{scale_z} \end{bmatrix} \begin{bmatrix} X - X_{HI} \\ Y - Y_{HI} \\ Z - Z_{HI} \end{bmatrix}. \tag{6.22}$$

6.2.2.2 Obtaining heading from calibrated magnetometer values

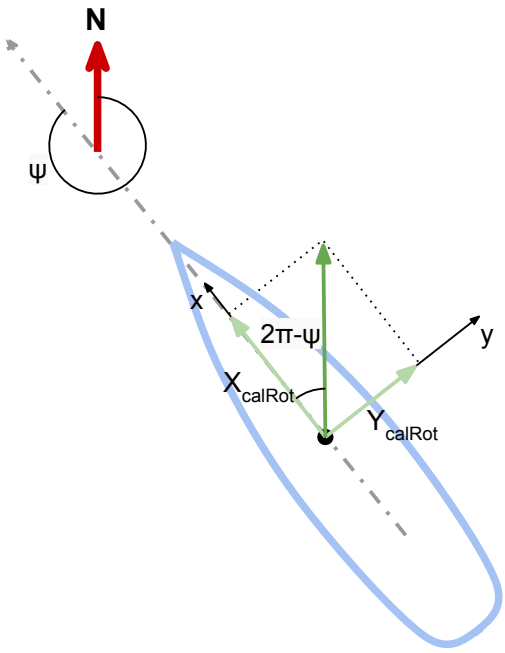


Figure 6.8: Obtaining heading from calibrated rotated magnetometer values.

To obtain the heading estimate we are only interested in the Earth's magnetic field components parallel to the Earth's surface. The magnetometer readings are given in the body frame, hence the measurement is rotated onto the surface plane:

$$\begin{bmatrix} X_{calRot} \\ Y_{calRot} \\ Z_{calRot} \end{bmatrix} = R_{y,\theta} R_{x,\phi} \begin{bmatrix} X_{calibrated} \\ Y_{calibrated} \\ Z_{calibrated} \end{bmatrix} \tag{6.23}$$

, where

$$R_{x,\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & -s\phi \\ 0 & s\phi & c\phi \end{bmatrix} \tag{6.24}$$

$$R_{y,\theta} = \begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ -s\theta & 0 & c\theta \end{bmatrix}, \tag{6.25}$$

where $s\cdot = sin(\cdot)$ and $c\cdot = cos(\cdot)$.

Following Figure 6.8, the heading can be obtained by:

$$\psi = 2\pi - atan2(X_{calRot}, Y_{calRot}) \tag{6.26}$$

6.2.2.3 Filtering the heading measurement

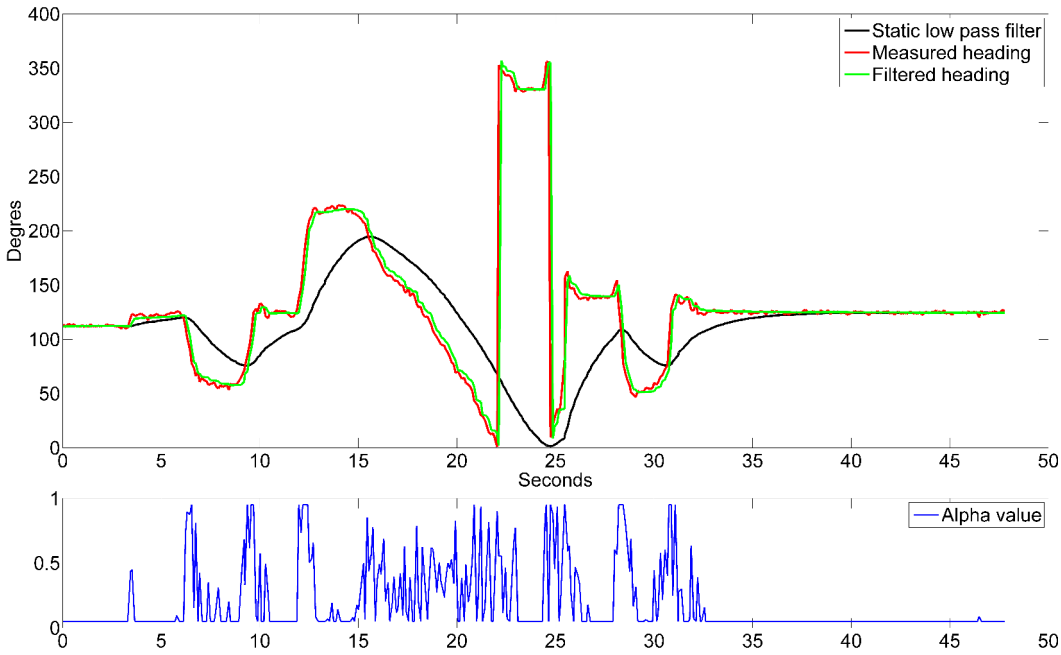


Figure 6.9: The heading filter(green) applied to a heading sample(red). The filters *alpha* value is shown below in blue. The black line is a static low pass filter with $alpha = 0.05$, which slowly converges to the heading filter.

The measured heading has a noise component with a magnitude of around ± 2-3 °. It is desirable to filter out this noise.

A first attempt was a low pass filter. As the measurement is in the range $[0°, 360°)$, there is a discontinuity when the the heading passes the 0°mark(this is clearly visible in Figure 6.9). Hence, the filter was designed in the following iterative way:

$$\Delta[k] = circularDist(\psi[k-1], \psi) \tag{6.27}$$
$$\gamma = \alpha_\psi(\psi[k-1] + \Delta[k]) + (1 - \alpha_\psi)\psi[k-1] \tag{6.28}$$
$$\psi[k] = circularMap(\gamma), \tag{6.29}$$

, where $circularDist(a, b)$ returns the least amount of degrees from b to a.ψ is the heading measurement. $\psi[k] \in [0°, 360°)$ is the new filtered heading value at iteration k and $\psi[k-1] \in [0°, 360°)$ is the previous. $circularMap(a)$ maps a into $[0°, 360°)$ by respectively adding or subtracting a whole round(360°) to a. α_ψ is the *smoothing factor*. The filters cutoff

frequency, f_c, can by calculated as:

$$f_c = \frac{\alpha_\psi}{(1-\alpha_\psi)2\pi\Delta_t}, \tag{6.30}$$

where Δ_t is the sample time.

To achieve the desirable smoothing of the input signal an α value of 0.05 was required. The sample time is 0.1 seconds (10 Hz). Hence, the phase lag became to large. See the black line in Figure 6.9.

As the amplitude of the noise is limited, it is possible to use this knowledge to design a better performing filter with a variable smoothing factor. The following iterative filter is proposed:

$$\Delta[k] = circularDist(\psi[k-1], \psi) \tag{6.31}$$
$$\alpha_{\psi,var}[k] = sat(\beta(|\Delta[k]| - \Delta_{thres}), [0, \alpha_{\psi,max} - \alpha_{\psi,min}]) \tag{6.32}$$
$$\alpha_\psi[k] = \alpha_{\psi,min} + \alpha_{\psi,var}[k] \tag{6.33}$$
$$\gamma = \alpha_\psi[k](\psi[k-1] + \Delta[k]) + (1-\alpha_\psi[k])\psi[k-1] \tag{6.34}$$
$$\psi[k] = circularMap(\gamma), \tag{6.35}$$

where $\alpha_{\psi,min} = 0.05$ and $\alpha_{\psi,max} = 0.9$ are the minimum and maximum smoothing factor, respectively. $\Delta_{thres} = 4$ is a threshold for increasing the smoothing factor, with respect to the difference in the measurement and the filtered value. $|\Delta[k]|$ is the absolute value of $\Delta[k]$. $\beta = 0.1$ decides the slope of the smoothing factor increase. Note that a high smoothing factor implies less smoothing. $sat(a, b)$ returns a saturated to the range b.

In Figure 6.9 and Figure 6.10 the filter is tested on a sample set. It is clear that the phase lag is low during large and rapid changes in heading, but still the smoothness remains high when the heading variation is small. The sample set was generated by moving the hardware by hand while sampling data points. The filter is also implemented and used in the boat which significantly has decreased actuator jitter.

6.2.3 Position, speed and course

Position, speed and course are simply obtained directly from the GPS module. As expected the course is inaccurate when the speed is very low.

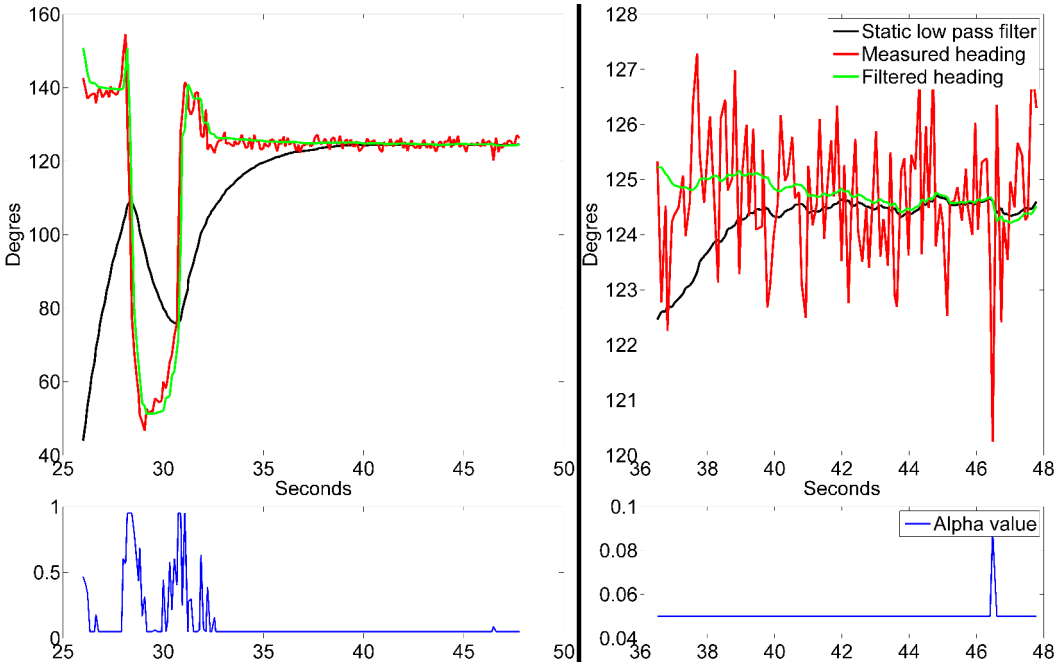


Figure 6.10: The same heading sample and filters as in Figure 6.9.

6.3 Control

6.3.1 Control allocation

Control allocation is the action of transforming, or map, control forces or moments onto set-points for actuators. Hence the controllers at the lowest level runs the actuators towards the set-points and in conjunction with the craft the actuators produces the forces and moments ordered.

The controllers at the lowest level for the boat are located inside the servos and ensure that the servo shafts are set to match the input signal. However, high frequency signals are physically low passed out as the servos can't move infinitely fast. Higher level controllers must take this into account if it is important that the set-points are properly fulfilled.

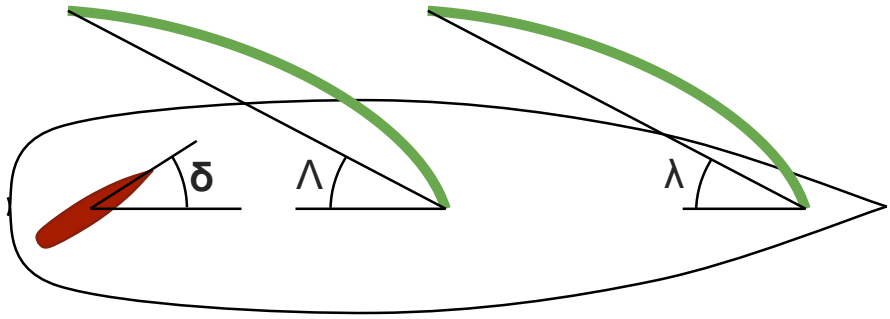


Figure 6.11: Actuator angles.

At the I/O-node end points are hard coded in. In other words, it's not possible for the main computer to command infeasible set-points for the servos. The main computer passes on set-point values in the following way:

$$ref_{ruddder} = \mathbb{Z} \in [rudder_{min}, rudder_{max}] \tag{6.36}$$
$$ref_{mainSheet} = \mathbb{Z} \in [mainSheet_{min}, mainSheet_{max}] \tag{6.37}$$
$$ref_{genoa} = \mathbb{Z} \in [genoa_{min}, genoa_{max}], \tag{6.38}$$

where $rudder_{min} \geq 0$, $mainSheet_{min} \geq 0$ and $genoa_{min} \geq 0$ are the low set point bounds for the rudder, main sail and Genoa sail and corresponds to minimal rudder angle $\delta_{min} < 0°$, the minimal main sheet angle $\Lambda_{min} \approx 0°$ and the minimal Genoa sail angle $\lambda_{min} \approx 0°$, respectively. $rudder_{max} >> 0$, $mainSheet_{max} >> 0$ and $genoa_{max} >> 0$ are the high set point bounds for the rudder, main sail and Genoa sail and corresponds to maximal rudder angle $\delta_{max} > 0$, the maximal main sheet angle $\Lambda_{max} \approx 90°$ and the maximal Genoa sail angle $\lambda_{max} \approx 85°$, respectively. $\mathbb{Z}$ are all real integer numbers, this implies that set points are rounded off before they are passed on to the actuators. The round off error is negligible as the ranges are large.

6.3.1.1 Sail angles

It is desirable for the higher level controllers to be able to set the angle of the sails. The set-points for the Genoa and main sail can be obtained from the desirable angles by basic

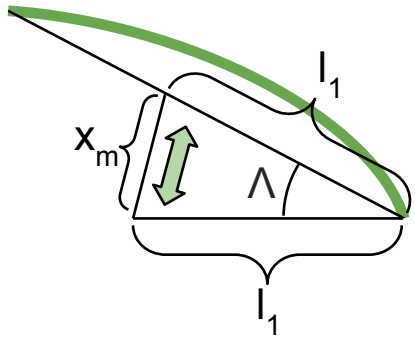


Figure 6.12: Symbols used in sail angle calculation.

trigonometry, under the assumption that the wind pulls the sails as far out as the sheets allows:

$$x_m = l_1\sqrt{2(1-\cos(\Lambda))} \tag{6.39}$$

$$x_g = l_2\sqrt{2(1-\cos(\lambda))} \tag{6.40}$$

$$ref_{mainSheet} = \frac{x_m}{x_{m,max}}(mainSheet_{max} - mainSheet_{min}) + mainSheet_{min} \tag{6.41}$$

$$ref_{genoa} = \frac{x_g}{x_{g,max}}(genoa_{max} - genoa_{min}) + genoa_{min} \tag{6.42}$$

$$\downarrow mainSheet_{min} = genoaSheet_{min} = 0 \tag{6.43}$$

$$ref_{mainSheet} = \frac{x_m}{x_{m,max}} mainSheet_{max} = \frac{l_1\sqrt{2(1-\cos(\Lambda))}}{l_1\sqrt{2(1-\cos(\Lambda_{Max}))}} mainSheet_{max} \tag{6.44}$$

$$= mainSheet_{max}\sqrt{\frac{1-\cos(\Lambda)}{1-\cos(\Lambda_{Max})}} \tag{6.45}$$

$$ref_{genoa} = genoaSheet_{max}\sqrt{\frac{1-\cos(\lambda)}{1-\cos(\lambda_{Max})}}, \tag{6.46}$$

where Λ and λ are the desired main sail and Genoa sail angles respectively. See Figure 6.12 for further explanation of the symbols.

6.3.1.2 Including moment

Higher level controllers will ask for a certain moment which in turn makes the boat go where the controller wants. There are two controllable components that generate such a moment:

- Rudder angle
- Difference in generated force by the main sail and the Genoa

In traditional sailing, the rudder is only used for fine tuning the heading. The greatest moment is set by adjusting the sails. In fact, the rudder may be omitted completely as on a wind surfing board. Hence, mapping a desirable moment should involve alterations to the main and Genoa sail angle. Thus a moment mapping function should include a main sail and Genoa sail angle contribution; Λ_{moment} and λ_{moment} respectively.

The new sail set-points are altered in the following way to accommodate this:

$$\Lambda_{tot} = \Lambda + \Lambda_{moment} \in [0°, 90°] \tag{6.47}$$

$$\lambda_{tot} = \lambda + \lambda_{moment} \in [0°, 90°] \tag{6.48}$$

$$ref_{mainSheet} = mainSheet_{max}\sqrt{\frac{1-\cos(\Lambda_{tot})}{1-\cos(\Lambda_{Max})}} \tag{6.49}$$

$$ref_{genoa} = genoaSheet_{max}\sqrt{\frac{1-\cos(\lambda_{tot})}{1-\cos(\lambda_{Max})}} \tag{6.50}$$

A simplified moment map function has been suggested:

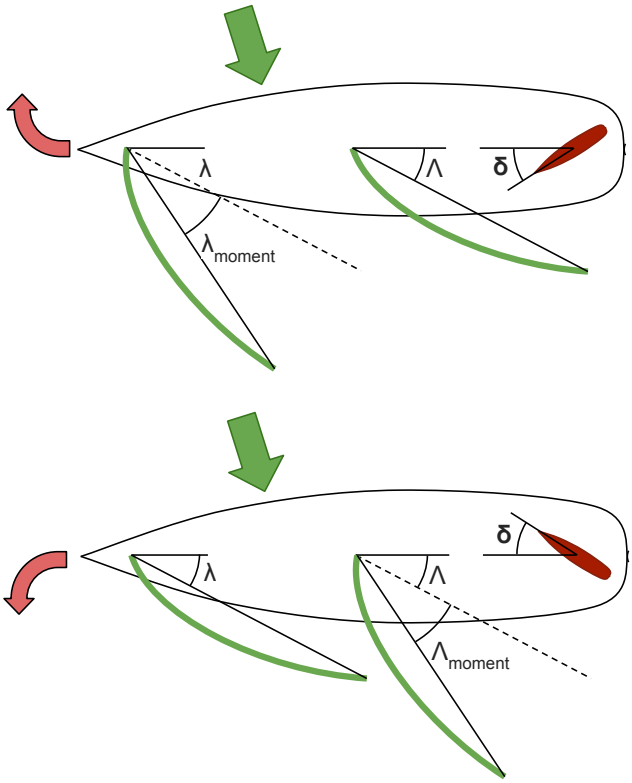


Figure 6.13: Sail moment map.

Moment map: Assuming the desirable moment is given as:

$$N_{des} \in [-1, 1], \tag{6.51}$$

where N_{des} is a virtual moment with no unit, as this is sufficient for now. The suggested moment map follows as:

$$ref_{ruddder} = \frac{N_{des}}{2}(rudder_{max} - rudder_{min}) + rudder_{min} \tag{6.52}$$

$$\epsilon = abs(N_{des}) - abs(N_{sail\ threshold}) \in [0, 1] \tag{6.53}$$

$$starboardTack = \begin{cases} true = 1 & \text{if } \omega_{rel} \in [0°, 180°) \\ false = 0 & \text{if } \omega_{rel} \in [180°, 360°) \end{cases} \tag{6.54}$$

$$\Lambda_{moment} = xor(starboardTack, isPositive(N_{des}))\epsilon\Lambda_{max\ contribution} \tag{6.55}$$

$$\lambda_{moment} = xnor(starboardTack, isPositive(N_{des}))\epsilon\lambda_{max\ contribution}, \tag{6.56}$$

where $\Lambda_{max\ contribution} \geq 0$, $\lambda_{max\ contribution} \geq 0$ and $N_{sailthreshold} \geq 0$ are tuning parameters. These are implemented as so called *synchronized variables*, see Section 5.3.1.1 and 5.2.5. $\omega_{rel} \in [0°, 360°)$ is the relative wind direction. $isPositive(a)$ returns $true = 1$ when $a > 0$. $xor(a, b)$ returns 1 when $a \neq b$, otherwise 0. $xnor(a, b)$ returns 1 when $a = b$, otherwise 0.

This moment map can more easily be described in words. The rudder is moved proportionally with the desired moment. After the certain threshold, $N_{sailthreshold}$, the respective sail eases out away from the wind proportionally to the desired moment by a scale of $\Lambda_{max\ contribution}$ for the main sail or $\lambda_{max\ contribution}$ for the Genoa sail. Figure 6.13 shows an example scenario.

During testing of the boat, $\Lambda_{max\ contribution}$ and $\lambda_{max\ contribution}$ were both typically set to 45°and $N_{sailthreshold}$ to 0, but different configurations were tested. As the boat has a main sail that is approximately the same area as the Genoa sail, it make sense for $\Lambda_{max\ contribution}$ to be equal to $\lambda_{max\ contribution}$.

Another way of doing this would be to tighten and shorten the two sails equally. However, it is not desirable to make a sail angle smaller as this may cause turbulence around that sail, which in turn lessens the sail efficiency.

As the rudder is proportional to the desired torque and the rudder dynamics are very fast, a torque is almost immediately produced when the desired torque changes.

As stated, the rudder is proportional to the desired torque and the sails remains unaffected below the $N_{sailthreshold}$, but it can be argued that this should have been the other way around. That is, the sails contributes with a moment proportional to the the desired torque and the rudder is only used after a certain threshold. This would have been an advantage as a steady state condition with no rudder drag can be achieved. This, as it is rarely the case that the desired torque is zero at steady state.

On the other hand, an advantage of the current moment map is that it utilizes the fast rudder response. The response of the rudder is of several magnitudes faster than the sails. Hence, it is believed that the rudder should act proportionally to the desired moment.

6.3.1.3 Setting the desired sail angles

This section describes how the desired sail angles are set in automatic modes of operation.

Up until this point, it is clear that the control allocator will set the sail angles as the

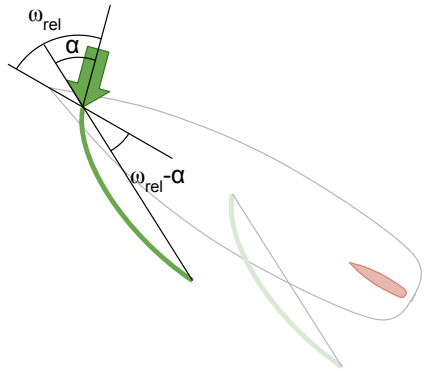


Figure 6.14: Using the angle of attack to set desired sail angles.

desired ones (Λ and λ) unless it is required to increase one of them to generate a moment. We also know form Section 2.2 that the sails can be modelled as wings which creates an amount of the desired lift dependent on the angle of attack. Hence the wind direction measurement is used to set the sail angles in conjunction with an angle of attack variable, α, see Figure 6.14:

$$\Lambda = \lambda = \omega_{rel} - \alpha \in [0°, 90°]. \tag{6.57}$$

It is no obvious choice for α. The optimal α value is dependent on the wind speed, rig configuration and other variables. For now, the α value is set by the user as a *synchronized variable*. Testing shows that the α value only needs to be updated if there is a significant change in the wind speed.

Running downwind: As the sail angles are limited to $\in [0°, 90°]$, the sails will remain at 90° when running downwind, which is desirable.

6.3.1.4 Increasing running stability

Running downwind is the least stable point of sail. The current configuration does not allow for automatic reefing of the sails and hence most of the sailing force is generated by the main sail when running downwind. This is because the Genoa sail is in the wind shadow of the main sail and has no function in this configuration, see Figure 6.15. This is problematic as the main sail is not centred, but protrude sideways. Hence, a large moment is generated by the main sail in this case. If the winds are to strong, the rudder may not suffice to counteract this moment.

To mitigate this unwanted effect, it has been experimented with the following feature: When the wind is blowing in from the stern, which indicates that the boat is running downwind, the Genoa sail is brought al the way in($\lambda_{tot} = 0$). This will generate a moment counteracting the main sail moment when running broad reach, but unfortunately has no effect when running exactly downwind. Figure 6.15 illustrates this concept.

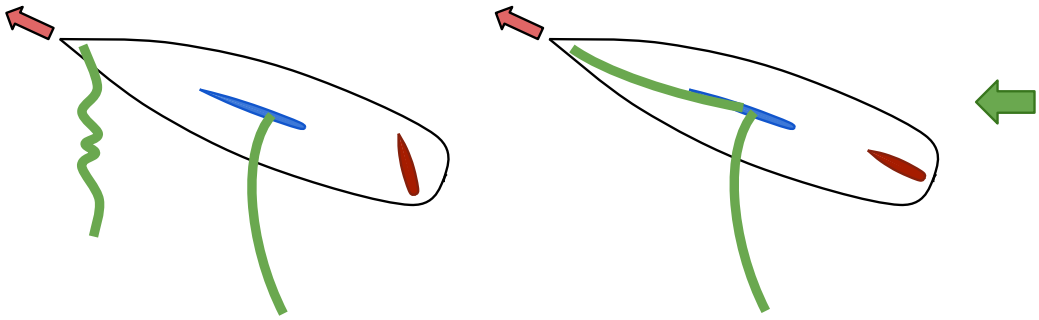


Figure 6.15: To the right; Increasing running stability by bringing in the Genoa sail. To the left; Genoa is left out, hence the rudder has a larger angle to maintain the same heading at this point of sail.

The feature is expressed mathematically in the following way when it is turned on.

$$\lambda_{tot} = \begin{cases} 0 & \text{if } \omega_{rel} \in [180° - \beta_{runningstability}, 180° + \beta_{runningstability}) \\ \lambda + \lambda_{moment} \in [0°, 90°] & \text{otherwise} \end{cases}, \tag{6.58}$$

where $\beta_{runningstability}$ is the threshold for activating the feature. It is implemented as a *synchronized variable*, see Section 5.3.1.1 and 5.2.5. Currently, this feature has not been verified as effective, see Section 7.6.

6.3.2 Modes of operation

Until now, four modes of control operation have been defined:

- *Semi-manual*
- *Heading hold*
- *Waypoint tracking*
- *Path following*

The following sections will describe these modes.

6.3.2.1 Semi-manual

In *Semi-manual* mode the user can control the desired moment, see Section 6.3.1, by the means of the same user interface slider used for controlling the rudder in full manual mode, see Section 5.2.2. In the same fashion, the sail angle of attack is given by the slider which controls the main sail in full manual mode.

Hence the main computer determines the sail angles by itself and the user does not need to consider the sails. As long as the heading is feasible, the user experiences the same ease as driving a remote car.

This mode of operation has proved itself very useful while testing not only the control allocation algorithm, but different aspects about the system. It has also been an effective tool for manually positioning the boat prior to testing of higher level algorithms.

6.3.2.2 Heading hold

The *heading hold* mode uses a basic proportional controller to maintain a certain heading, ψ_{ref}:

$$e = \psi_{ref} - \psi \tag{6.59}$$
$$N_{des} = K_p e, \tag{6.60}$$

where $\psi \in [0°, 360°)$ is the measured heading, see Section 6.3.1 and N_{des} the desired torque, which at a lower level is saturated to $[-1, 1]$. $K_p > 0$ is the proportional tuning parameter. K_p and $\psi_{ref} \in [0°, 360°)$ are passed down from the user interface. See Section 7.4 for further testing of this simple control law.

6.3.2.3 Waypoint tracking

The *waypoint tracking* mode is the simplest automated mode that closes the position loop. The overlying guidance system, see Section 6.4 passes waypoints (the desired position) down to this controller. The controller ensures progress towards the respective waypoint.

Progress towards the waypoint is simply ensured by passing the direction towards the waypoint as ψ_{ref} to the same heading controller as in *Heading hold* mode. This is a very simple approach that does not take sideslip into account, hence high performance is not expected from this mode.

The heading reference, ψ_{ref}, is calculated in the following way:

$$\mathbf{P}^n_\Delta = \mathbf{P}^n_{Ref.Waypoint} - \mathbf{P}^n_{boat} \tag{6.61}$$
$$\psi_{ref} = 90° - atan2(\mathbf{P}^n_\Delta), \tag{6.62}$$

where $\mathbf{P}^N_{Ref.Waypoint}$ and $\mathbf{P}^N_{boat}$ is the NED position of the waypoint and boat, respectively. $atan2(\mathbf{a}) \in [-\pi, \pi]$ returns the angle between the vector argument and the east axis, which is equivalent to $atan2(a_1, a_2)$.

In Section 7.6 *waypoint tracking* is tested with success.

6.3.2.4 Path following

The *path following* mode is another approach to close the position loop. The mode tries to not only ensure progress towards the next waypoint, but also remain on the line defined by the next waypoint and the previous waypoint. Hence, sideslip is taken into account.

The technique used to achieve this is called *Lookahead-based steering*[25]. This technique is based on calculating the boats distance perpendicular to the line, or *cross-track error*, e:

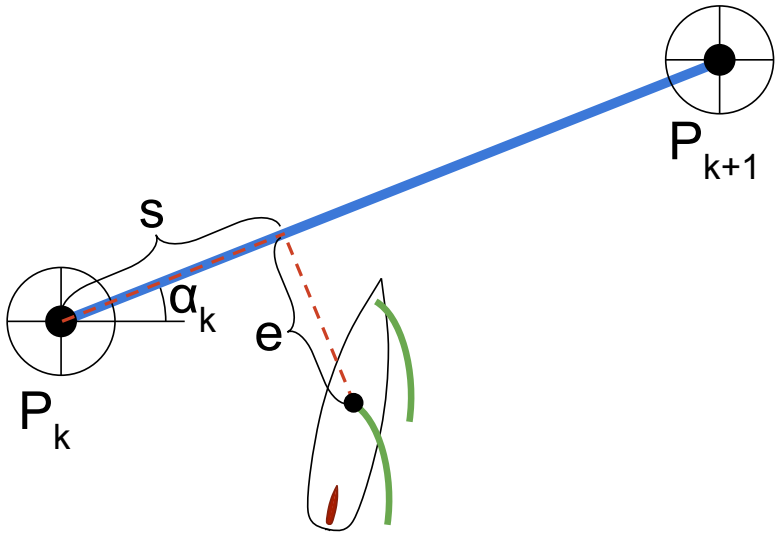


Figure 6.16: Lookahead-based steering[26].

$$\alpha_k = atan2(\mathbf{P}_{k+1} - \mathbf{P}_k) \tag{6.63}$$

$$\begin{bmatrix} s \\ e \end{bmatrix} = R(\alpha_k)^T(\mathbf{P}^n_{boat} - \mathbf{P}^n_k), \tag{6.64}$$

where α_k is the line heading, P_{k+1} and P_k is the next and previous waypoint respectively, s is the distance travelled along the line. R is the 2 dimensianl rotation matrix for rotations about the z/down-axis:

$$R(\psi) = \begin{bmatrix} c\phi & -s\phi \\ s\phi & c\phi \end{bmatrix}, \tag{6.65}$$

where $s\cdot = sin(\cdot)$ and $c\cdot = cos(\cdot)$.

e is used as negative feedback to the same heading controller as in *Heading hold* mode:

$$\chi_r = sat(K_p e + K_i \int e,\ [-90°, +90°]) \tag{6.66}$$

$$\psi_{ref} = \alpha_k + \chi_r, \tag{6.67}$$

where $\psi_{ref} \in [0°, 360°)$ is the desired heading, $K_p > 0$ the proportional tuning parameter and $K_i > 0$ the integral tuning parameter. $sat(a, b)$ returns a saturated to the range b. The tuning parameters are set by the user on the user interface.

As an alternative to saturating χ_r is to use atan [25]:

$$\psi_{ref} = \alpha_k + atan(K_p e + K_i \int e). \tag{6.68}$$

Unfortunately, there has not been time during this thesis to draw conclusions abut the performance of this mode, see Section 8.1.

6.4 Guidance

This chapter is focused on the guidance objective, which is to enable the boat to follow a predefined path/route. It has also been desirable to make a path planner that generates a feasible path towards a user defined location, taking into consideration the wind constraint and the geographical constraints. However, there has not been time to address this issue properly, but at the end of the chapter some experiments are presented.

From sections 6.3.2.3 and 6.3.2.4 it is established low level controllers that either ensurers progress towards a feasible waypoint or ensures progress along a line defined by two waypoints, respectively. Hence, the main task, of the guidance part of the system, is to pass feasible waypoints down to these controllers in order to conduct the mission. In this case, the mission consists of sailing through a set of waypoints or follow a certain path. The next two sections describes how the two mission types are addressed.

6.4.1 Waypoint tracking

When the mission is to sail through a set of waypoints, the *Waypoint tracking* mode is applied. The definition of sailing through a waypoint is to bring the boat within a certain radius,$r_{waypoint} > 0$, of the waypoint. It is assumed that the predefined set of waypoints are feasible and that the boat is able to sail through the set. The following algorithm is used to achieve waypoint tracking:

Algorithm 6.1 Waypoint tracking

$P \leftarrow P_1, P_2, ..., P_N$
$i \leftarrow 0$
repeat
  $P_{Ref.Waypoint} \leftarrow P_i$
  if $|P^n_{boat} - P^n_i| < r_{waypoint}$ **then**
    $i \leftarrow i + 1$
  end if
until $i == N$

, where P is the set of waypoints and $P_{Ref.Waypoint}$ is the reference waypoint from Section 6.3.2.3.

If a waypoint is infeasible, or becomes infeasible due to high ocean currents or high side-slip, the boat fails to complete the route. It is trivial to determine if that is the case and the guidance system should attempt to generate a new path, see Section 6.4.3.

6.4.2 Path following

When the mission is to follow a path, the *Path following* mode is used. A similar algorithm that of *Waypoint tracking* is used for this purpose:

Algorithm 6.2 Path following

$P \leftarrow P_1, P_2, ..., P_N$
$i \leftarrow 0$
repeat
  $P_k \leftarrow P_i$
  $P_{k+1} \leftarrow P_{i+1}$
  if $s > |P^n_{boat} - P^n_i|$ **then**
    $i \leftarrow i + 1$
  end if
until $i == N$

,where P consists of waypoints that makes up the path from the lines defined by coincident waypoints. P_{k+1} and P_k defines the current line as in Section 6.3.2.4. s is the distance travelled along the current path, see Section 6.3.2.4.

If a line is infeasible, or becomes infeasible due to high ocean currents or high side-slip, the boat fails to complete the route. It is trivial to determine if that is the case and the guidance system should attempt to generate a new path, see Section 6.4.3.

6.4.3 Path generation

From a practical perspective, any path or waypoint route may be divided into a finite set of lines. For curved paths, there is a trade off between the representation accuracy and number of line segments. Hence, using the previous control algorithms, the boat may follow any feasible path.

6.4.3.1 Triangle test path

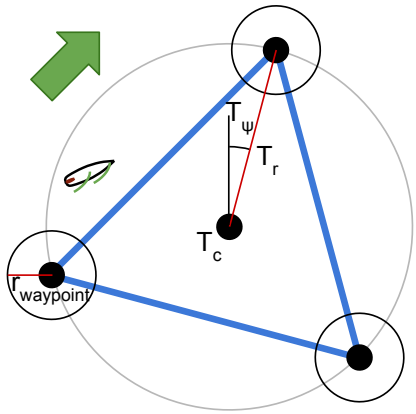


Figure 6.17: Equilateral triangle test path.

During field tests, a frequently used test path was the *triangle test path*. This is a generated path shaped as a equilateral triangle. When the triangle path is rotated such that one side is parallel to the wind, all sides of the triangle are feasible lines with some margin. Hence, the boat may sail along the triangle.

From the user interface, the user can adjust the center position, T_c, the triangle radius, T_r, the triangle heading, T_ψ, and if applicable; the waypoint radius, $r_{waypoint}$.

6.4.3.2 Map

A map is essential for the robotic sailing path planner. A path planner uses the map for validation of points and paths, thus avoiding running on shore. The user interface allows the user to view the boat location on a map.

Figure 6.18: Map Example

The simplest way of representing the map is by an image/bitmap. Each pixel corresponds to a location and the pixel's color indicates whether the location is legal. Thus a black and white image is sufficient to represent the map.

In the current implementation a color picture is used for illustrative purposes. Blue color indicates a legal position while other colors represent illegal positions. See Figure 6.18. Corresponding to the image/map, the south-west corner(the map origin) location is known. Furthermore the degree density per pixel is also known. Thus the pixel location in the map/image can be calculated from longitude latitude and visa versa:

$$P(\Theta) = (\Theta - C) * \rho \tag{6.69}$$

$$\Theta(P) = \frac{P}{\rho} + C, \tag{6.70}$$

where

$$P = \begin{bmatrix} \text{vertical pixel location} \\ \text{horiontal pixel location} \end{bmatrix} \tag{6.71}$$

$$\Theta = \begin{bmatrix} longitude \\ latitude \end{bmatrix} \tag{6.72}$$

$$C = \begin{bmatrix} \text{map origin longditude} \\ \text{map origin latitude} \end{bmatrix} \tag{6.73}$$

$$\rho = \text{deegrees per pixel.} \tag{6.74}$$

The map is Equirectangular projected[4], thus having the same angular pixel density in the latitudinal and longitudinal direction. Hence, objects on the map appear horizontally stretched when they are located close to the earth poles.

A small program has been written to automatically generate these maps from map data. The user specifies the coordinates of the desired resolution. The downside of using a high resolution is that the output image file becomes large. A large image file slows the path planner down as more lookups are needed.

The map data is downloaded from the Norwegian Mapping Authority [8] and hence the maps can be made to very high precision. The data is designed for 1:50 000 scale maps.

The map data is presented as polygons in a text file. Furthermore, the polygons may have one or several holes within. The polygon boundaries are illustrated in Figure 6.19. The path planner could use these polygons directly, but this would have been very computationally slow. In this case, the file would had to be searched to find the corresponding polygon of the lookup and further investigate if the lookup is inside the polygon.

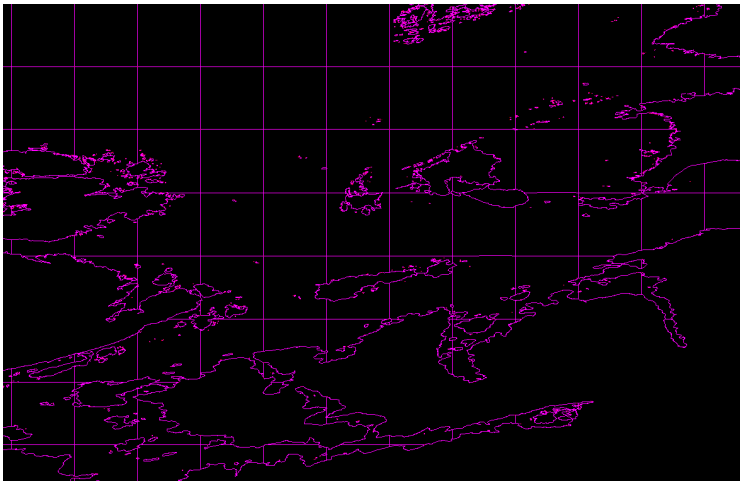


Figure 6.19: The map database consists of polygons

6.4.3.3 Path planner

In this section, a path planner for sailing is developed and suggested. The path planner produces optimal paths for sailing, taking the following into account:

- constraints of sailing
- geographical constraints and fixed obstacles
- ocean currents (not implemented or tested)
- moving obstacles where their future trajectories are known (not implemented or tested)
- can use weather forecasts for wind direction and ocean currents (not implemented or tested)

The path planner is *complete* and finds a solution if there is one. Unfortunately, there has not been time to to use this path planner in a field test.

A-star based path planner

The path planner is based on the *A-star search algorithm*, which is based on Dijkstra's algorithm. Dijkstra's algorithm and A-star both finds the shortest path through a graph, but A-star performs faster.A-stat also keeps track of the best known paths during the algorithm execution, which will be essential for the path planner. The A-star algorithm is stated below[20]:

Algorithm 6.3 A-star

```
1: function Astar(start, goal)
2:     closed_set ← the empty set                          ▷ The set of nodes already evaluated.
3:     open_set ← start                                    ▷ The set of tentative nodes to be evaluated
4:     came_from ← the empty map                           ▷ The map of navigated nodes.
5:
6:     g_score[start] ← 0                                  ▷ Cost from start along best known path.
7:     f_score[start] ← g_score[start] + heuristic_cost_estimate(start, goal)
8:
9:     while open_set is not empty do
10:        current ← the node in open_set having the lowest f_score[ ] value
11:        if current == goal then
12:            return reconstruct_path(came_from, goal)
13:        end if
14:
15:        remove current from open_set
16:        add current to closed_set
17:
18:        for each neighbor in neighbor_nodes(current) do
19:            if neighbor in closed_set then
20:                continue
21:            end if
22:
23:            tentative_g_score ← g_score[current] + dist_between(current, neighbor)
24:            if neighbor not in open_set or tentative_g_score < g_score[neighbor] then
25:                came_from[neighbor] ← current
26:                g_score[neighbor] ← tentative_g_score
27:                f_score[neighbor] ← g_score[neighbor]+heuristic_cost_estimate(neighbor, goal)
28:                if neighbor not in open_set then
29:                    add neighbor to open_set
30:                end if
31:            end if
32:        end for
33:    end while
34:    return failure
35: end function
```

The algorithm has a worst case performance of $\mathcal{O}(|E|)$ and a worst case memory complexity of $\mathcal{O}(|V|)$, where E is the number of edges in the graph, and V the number of vertices/points.

For this application the graph is an evenly distributed set of points in $\mathbb{R}^2$, representing valid legal positions for the boat. Here, a position is defined as legal by the corresponding point in the map image/bitmap, see Section 6.4.3.2. Hence, geographical constraints, shallow waters and other obstacles are taken into account by the map image/bitmap.

The algorithm uses a knowledge-plus-heuristic cost function of node x ($f\_score[x]$) to

determine the order in which the search visits nodes in the graph. The cost function is a sum of two functions[20]:

- The past path-cost function, which is the known cost of traversing the best known path to the current node x ($g\_score[x]$). In the current implementation, $g\_score[x]$ is iteratively obtained as the sum of $dist\_between(current, neighbor)$ along the best known path.
- A future path-cost function, which is an admissible "heuristic estimate" of the distance from x to the goal ($heuristic\_cost\_estimate(x,\ goal)$).

The $heuristic\_cost\_estimate(x,\ goal)$ function must be an admissible heuristic; that is, it must not overestimate the distance to the goal. Thus, for this application it must represent the euclidean/straight-line distance to the goal, since that is physically the smallest possible distance between any two points.

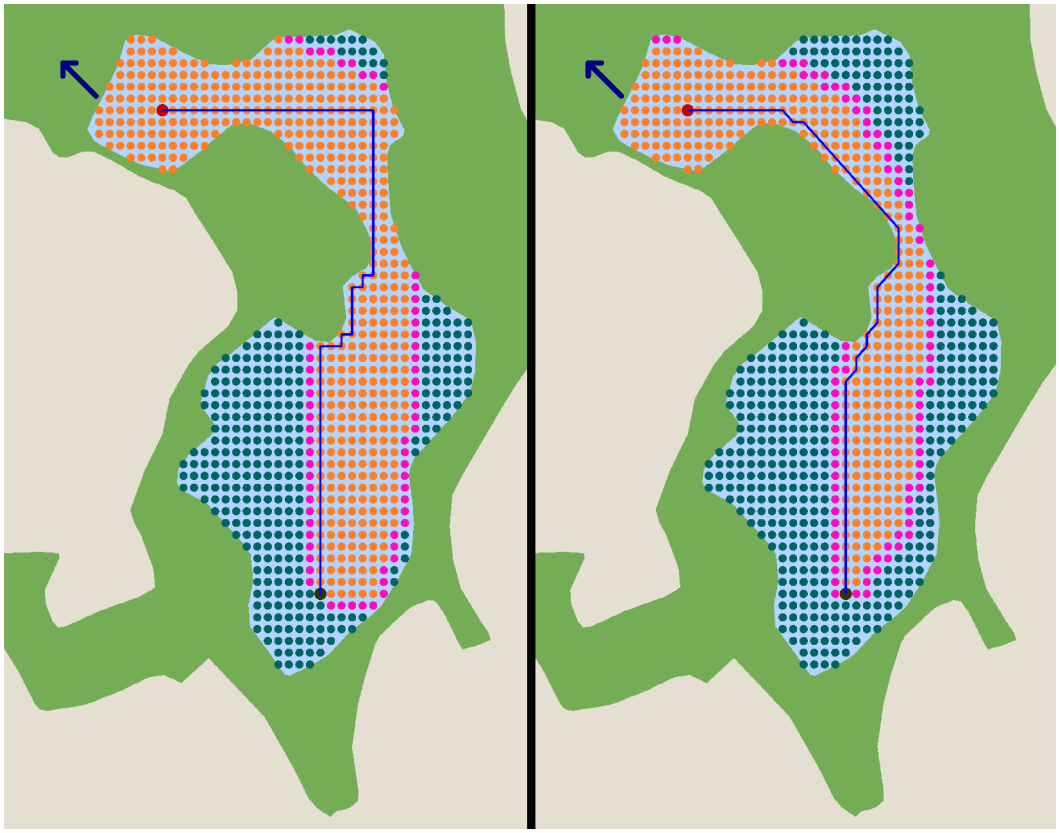


Figure 6.20: Results of the plain A-star algorithm. To the left, the graph is connected by the means of 4-neighbourhoods, while 8-neighbourhoods are used to the left. The coloured dots represents the nodes of the graph corresponding to positions on the map. Orange, pink and dark cyan nodes belong to the closed set, the open set and the unexplored nodes, respectively when the algorithm finishes. The large red dot is the start node, while the large dark brown node represent the goal node.

Figure 6.20 shows the result of the plain A-star algorithm used on an example scenario at Kyvannet in Trondheim, Norway known from Section 7. The generated map is shown

in the background, see Section 6.4.3.2. The map is stretched to mercator projection, hence lines of constant course are straight on the map[10]. For the implementation used in this scenario, note that:

- $dist\_between(current, neighbor)$ is simply the euclidean distance between *current* and *neighbor*. Hence, the objective is to obtain the shortest path through the graph. In this scenario the solution is not unique.
- The graph is defined by 4-neighbourhoods or 8-neighbourhoods (left result/right result in Figure 6.20, respectively). The neighbourhoods are defined in Figure 6.21. Hence, the $neighbor\_nodes(current)$ function is implemented accordingly. The path is optimal with respect to the graph, but clearly not to the problem of finding the shortest and most straight path. In fact, the path of the 4-neighbourhoods graph is similar to walking through a city centre where roads (the edges of the graph) are perpendicular to one another. In the next section this will be addressed.

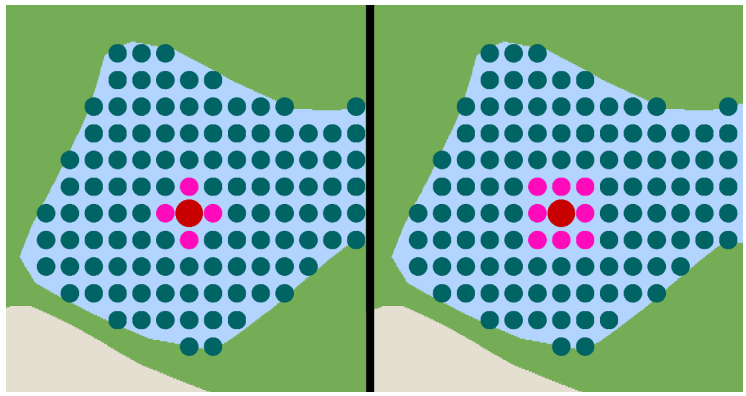


Figure 6.21: To the left, the 4-neighbourhood of the larger red node is shown in pink. To the right, the 8-neighbourhood.

Modifying the algorithm

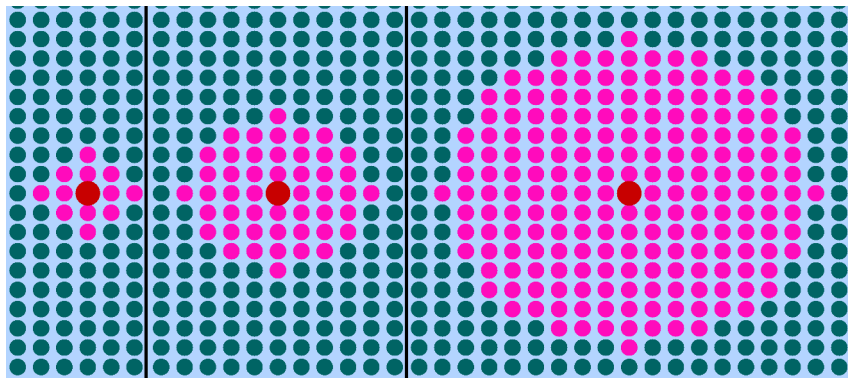


Figure 6.22: Circular neighbourhoods of the red larger node, with $r = 2$, $r = 4$ and $r = 8$ respectively.

Until now, the algorithm already takes into account the geographical constraints and fixed obstacles as the nodes of the graph are made up of legal nodes according to the map. The following problems remains:

1. From the previous, we learned that the graphs edges must be modified such that the shortest path through the graph coincide with the physical shortest path.
2. Take the constraints of sailing into account.
3. Take ocean current into account.
4. Avoidance of moving obstacles

To address problem 1, the graph has been modified, now each node does not have a 4- or 8-neighbourhood, but neighbouring nodes are defined as all nodes within a radius $r > 1$. See Figure 6.22. Hence, progression from one node to another can happen in many more angels, increasing quadratically with r. This has increased the number of edges in the graph. The results of this modification is shown in Figure 6.23 for several values of r

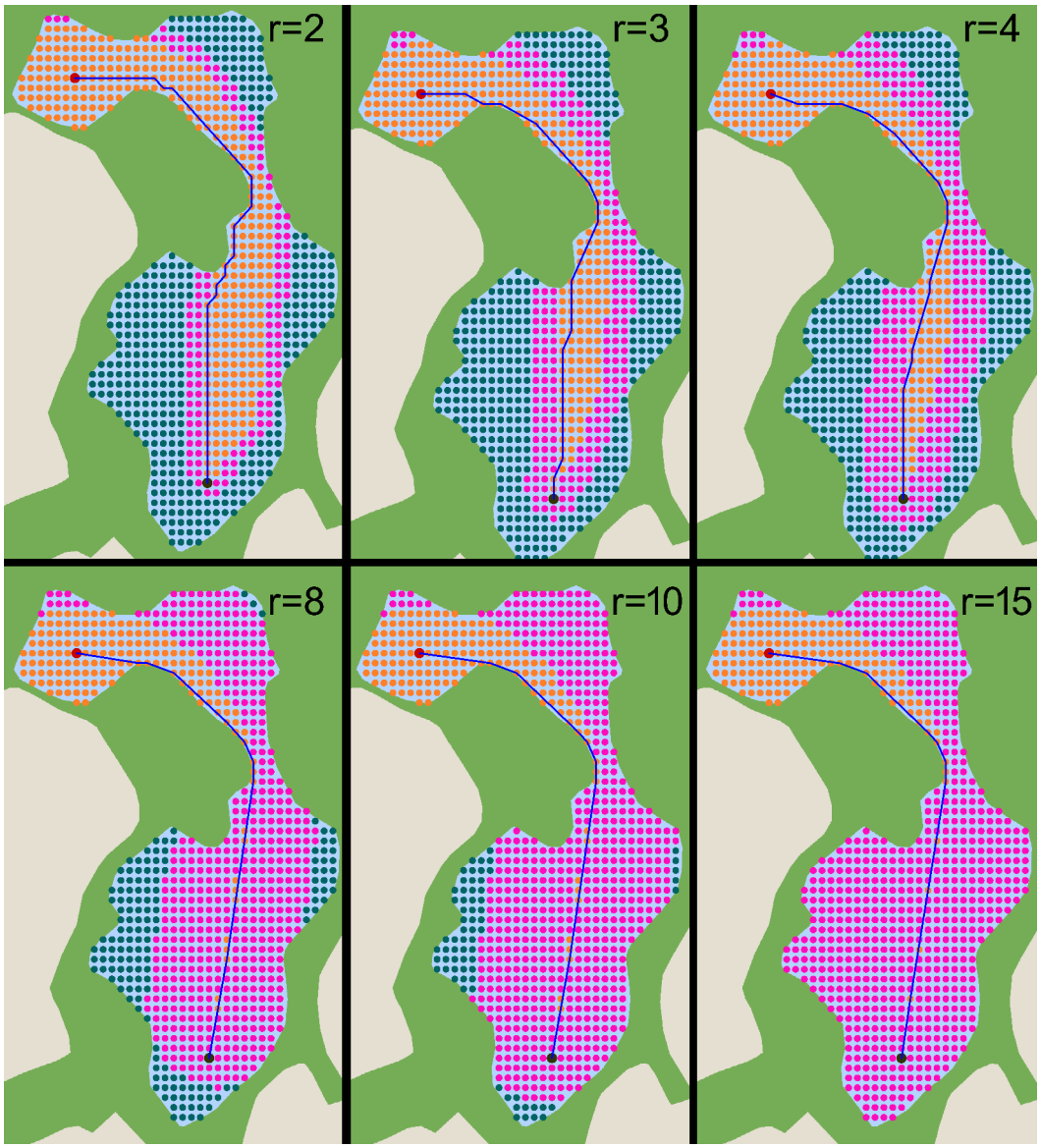


Figure 6.23: The results of A-star and the modified $neighbor\_nodes(current)$ versus different values of r.

Before proceeding with the discussion, the modified Astar algorythm is presented:

Algorithm 6.4 Sail boat path planner

```
1: function path_planner(start, goal, map, trajectories_of_moving_objects)
2:     closed_set ← the empty set                          ▷ The set of nodes already evaluated.
3:     open_set ← start                                    ▷ The set of tentative nodes to be evaluated
4:     came_from ← the empty map                           ▷ The map of navigated nodes.
5:
6:     time_spent[start] ← 0                               ▷ Time spent since start along best known path.
7:     g_score[start] ← 0                                  ▷ Cost from start along best known path.
8:     f_score[start] ← g_score[start] + heuristic_cost_estimate(start, goal)
9:
10:    while open_set is not empty do
11:        current ← the node in open_set having the lowest f_score[ ] value
12:        if current == goal then
13:            return reconstruct_path(came_from, goal)
14:        end if
15:
16:        remove current from open_set
17:        add current to closed_set
18:
19:        wind_direction ← wind_direction_at_current_node_and_time( current,
                                                                     time_spent[current])
20:        ocean_currents ← ocean_currents_at_current_node_and_time( current,
                                                                      time_spent[current])
21:        for each neighbor in neighbor_nodes( current,                    do
                                                map,
                                                wind_direction,
                                                ocean_currents,
                                                trajectories_of_moving_objects)
22:            if neighbor in closed_set then
23:                continue
24:            end if
25:
26:            travel_info ← relevant_information_about_the_traveled_path(came_from, time_spent)
27:            tentative_g_score ← g_score[current] + cost_to_neighbor( current,
                                                                        neighbor,
                                                                        travel_info,
                                                                        ocean_currents)
28:            if neighbor not in open_set or tentative_g_score < g_score[neighbor] then
29:                came_from[neighbor] ← current
30:                g_score[neighbor] ← tentative_g_score
31:                time_spent[neighbor] ← time_spent[current] + time_between( current,
                                                                              neighbor,
                                                                              travel_info,
                                                                              ocean_currents)
32:                f_score[neighbor] ← g_score[neighbor]+heuristic_cost_estimate(neighbor, goal)
33:                if neighbor not in open_set then
34:                    add neighbor to open_set
35:                end if
36:            end if
37:        end for
38:    end while
39:    return failure
40: end function
```

In the following, it is clear that the graph is defined by rules that apply equally to all nodes. Hence, the *neighbor_nodes*(...) will from this point on hold these rules and define the graph as the algorithm executes.

To address problem 2 and take into the constraints of sailing, the neighbourhoods (the graph or rules defined in *neighbor_nodes*(...)) have been further modified. A certain sector of the circular neighbourhood is removed corresponding to infeasible sailing directions, see Figure 6.24. The size of this infeasible sector is defined by $\psi_{infeasable}$ and is centred about the the opposite of the wind direction, $wind\_direction - 180°$. Without further modifications, this results in a feasible sailing path, see the left scenario in Figure 6.25. However, this will often result in a crocked path as the algorithm is not set to minimize the number of tacks along the path. A tack is an expensive manoeuvre when sailing, as speed is lost. The amount of tacks should be minimized. This leads us in to the next paragraph, where a cost can be set on such manoeuvres.

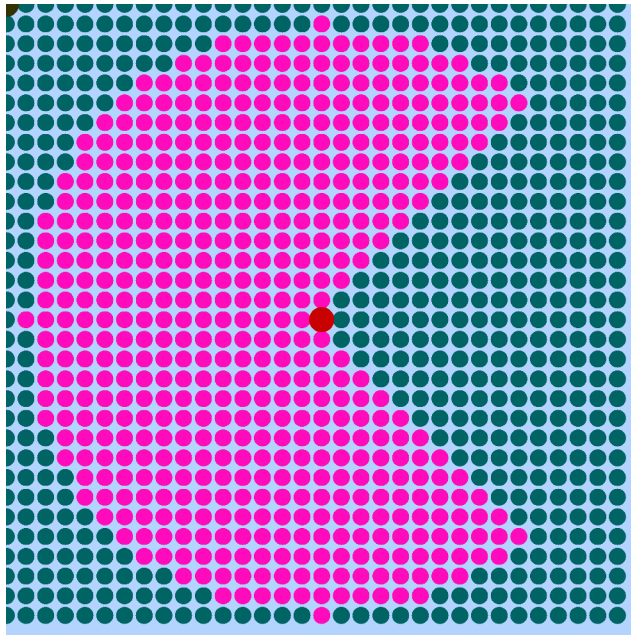


Figure 6.24: The neighbourhood of a node with the wind constraint taken into account. Here, the wind blows towards the left with respect to the image and $r = 15$ and $\psi_{infeasable} = 90°$

A nice feature of the A-star algorithm is that during its execution, it keeps track of the best known path onto the the node it is currently working on, *current*. To be more precise, all nodes in the best known path onto the *current* node is a part of the *closed_set*. Hence the best known path onto the *current* node can not change later in the execution. Thus, the implementation allows the cost of moving from *current* to *neighbor*, to depend on the path sailed so far. Furthermore, a time estimate,*time_spent*, is kept along the best known path. Hence, weather forecasts can be used to get the wind direction and ocean currents at the *current* node at the correct time. For now, in the current implementation, the wind remains constant and ocean current is absent. Hence, the *time_between*(...) function is not implemented. This would have required knowledge about the dynamics of the boat. One can argue that keeping the time estimate is of less use as it drifts of due to modelling error.

In the current implementation, the *cost_to_neighbor*(...) function adds a cost to changing

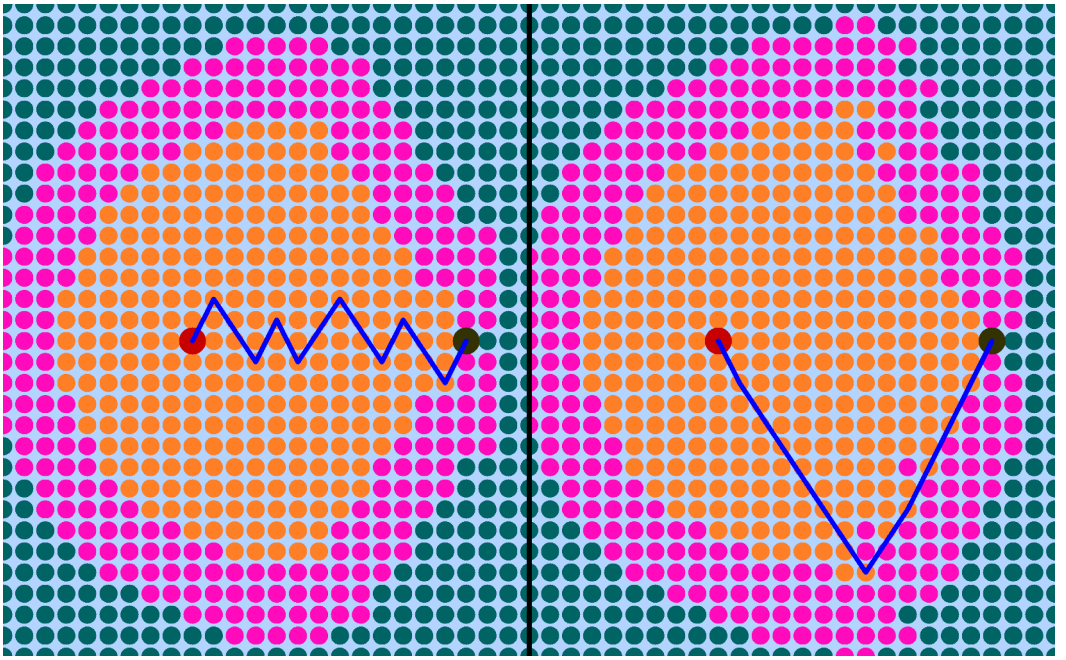


Figure 6.25: Results of the path planner in open water. To the left, the angular weighting scheme is not applied, resulting in a crocked path. The wind blows towards the left with respect to the image and $r = 4$ and $\psi_{infeasable} = 90°$

the course along the best known path, hence the algorithm favours paths with few tacks. The heading towards the current node, $\psi_{current}$, is passed to the function as the $travel\_info$ parameter. $\psi_{current}$ and the heading towards the next node, ψ_{next} are parameters of the angular cost. The total cost returned by $cost\_to\_neighbor(...)$ is:

$$cost = |neighbor - current| + p * \Delta(\psi_{current}, \psi_{next}), \tag{6.75}$$

where $\Delta(a°, b°)$ is the absolute value of the difference in angle $a°$ and $b°$. $p > 0$ is a cost parameter.

If ocean currents are to be taken into account, the respective extra travel cost of this must be added to the $cost\_to\_neighbor(...)$. Additional, the graph, defined by rules in the $neighbor\_nodes(...)$ function, must take the ocean current into account. It has not been time to experiment with this.

As the graph now is defined by rules in $neighbor\_nodes(...)$, it can be extended to take into account moving obstacles with known trajectories. As the $neighbor\_nodes(...)$ returns possible neighbours, or *"possible next steps"*, at a known time for the *current* node at hand, it can reject some or all of the possible next steps. It can typically reject neighbours/edges/*"possible next steps"* that will place the boat to close to a moving obstacle. This also remains as further work.

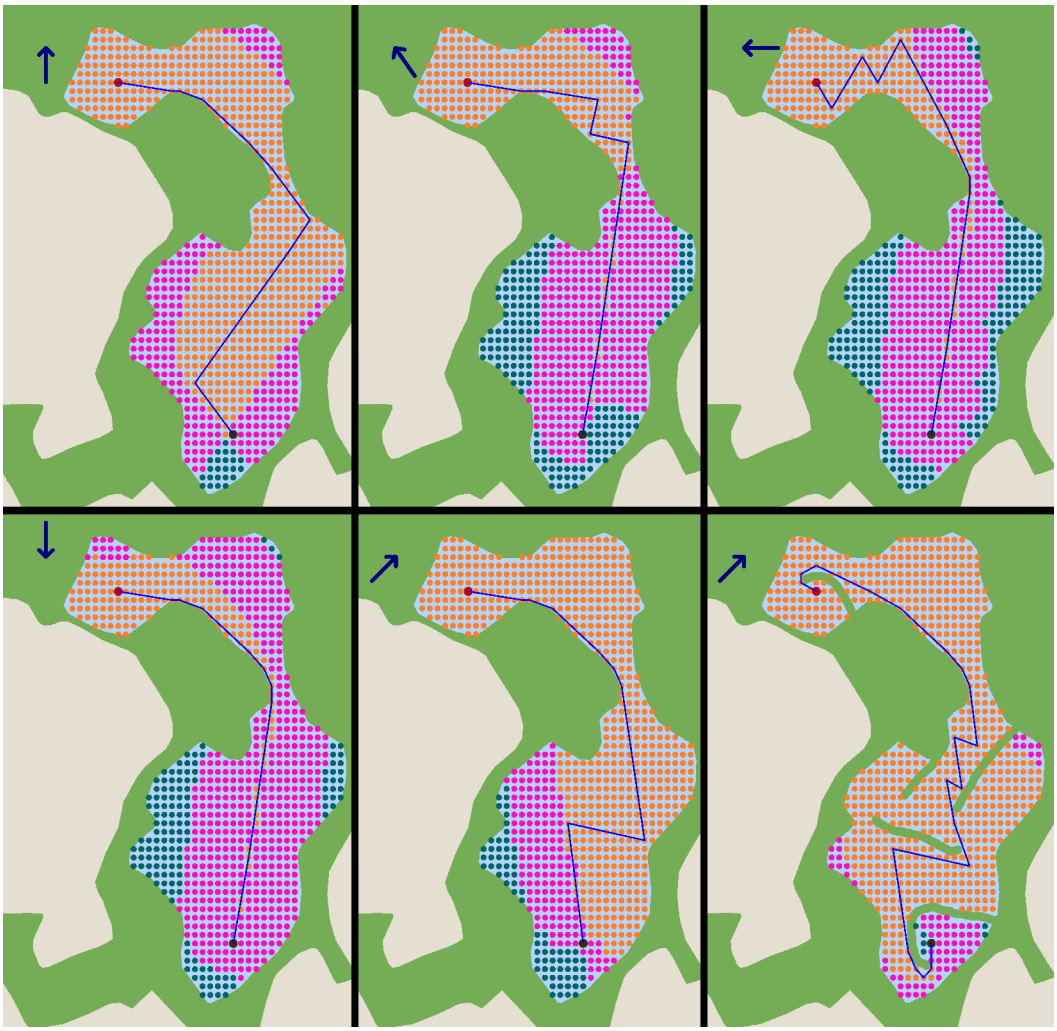


Figure 6.26: Results of the path planner at Kyvannet for different wind directions. To the bottom right, some extra obstacles are drawn into the map. Parameters used are $p = 0.5$, $r = 8$ and $\psi_{infeasable} = 90°$.

Chapter 7

Testing and Results

In this chapter all important tests are described. Thus, the process of improving the boat is addressed.

7.1 Logging and measurements

7.1.1 Purpose

The purpose of this early stage test was to see if the logging system was working and allowed for recording of the following measurements:

- Position, Speed and Course from the GPS module.
- Heading, which at the time was the compass angle calculated from the magnetometer. The magnetometer had been somewhat calibrated, but work on this was not finished.
- Roll and Pitch, which at the time was unfiltered.

Furthermore the purpose was to generate data which could be used to make automated illustration scripts.

7.1.2 Execution

The test began by powering up the boat. Afterwards the boat was carried outside. It was verified by the LED indicator on the GPS module that it had a GPS lock. The boat was than carried counter clockwise around a square at the NTNU campus, starting in the south west corner. The boat was approximately pointed into the direction it was carried, thus heading and course should correspond. In the third side of the rectangle, the boat was pitched front down. In the forth side, the boat was rolled to starboard side.

7.1.3 Results

- The log was working correctly, but there was an issue with the timestamps for all entry's.
- Scripts were made to automate illustration of logged data. These are shown in Figure 7.1, 7.2 and 7.3.
- The magnetometer data was promising although work on calibration was not finished. A software fault had led to that the data was offset by 90°.
- The GPS module data for speed and course are somewhat noisy.
- The unfiltered estimates for roll and pitch were noisy as expected, but clearly shows the trend for roll and pitch.
- From Figure 7.1 one can easily see that the GPS suffered from building reflections as the south-south-west pass deviates from the rectangular pattern. This should however not be a problem on open water.

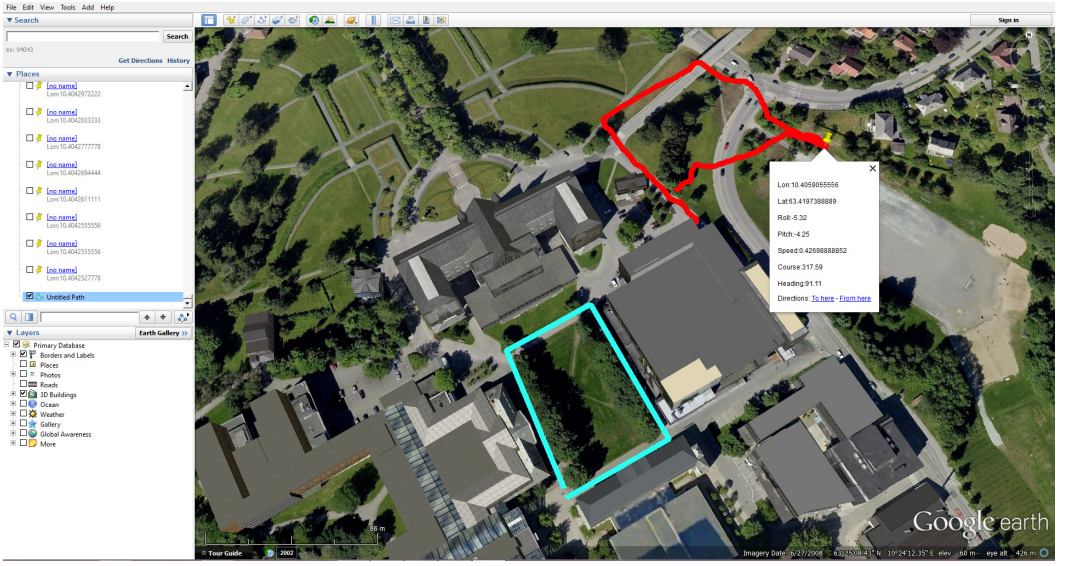


Figure 7.1: Data illustrated in Google Earth ™. The blue path is the actual path the boat was carried. The red path is the GPS measurement. By clicking on a point, more data are available.

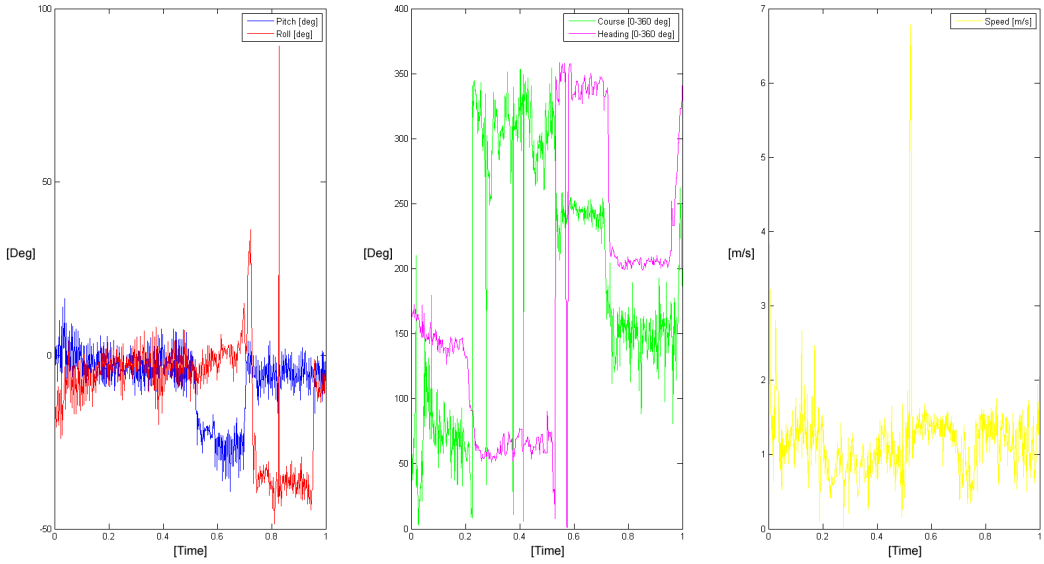


Figure 7.2: Logged data plotted. Red is roll, pitch is blue, green is course, pink is heading and yellow is speed.

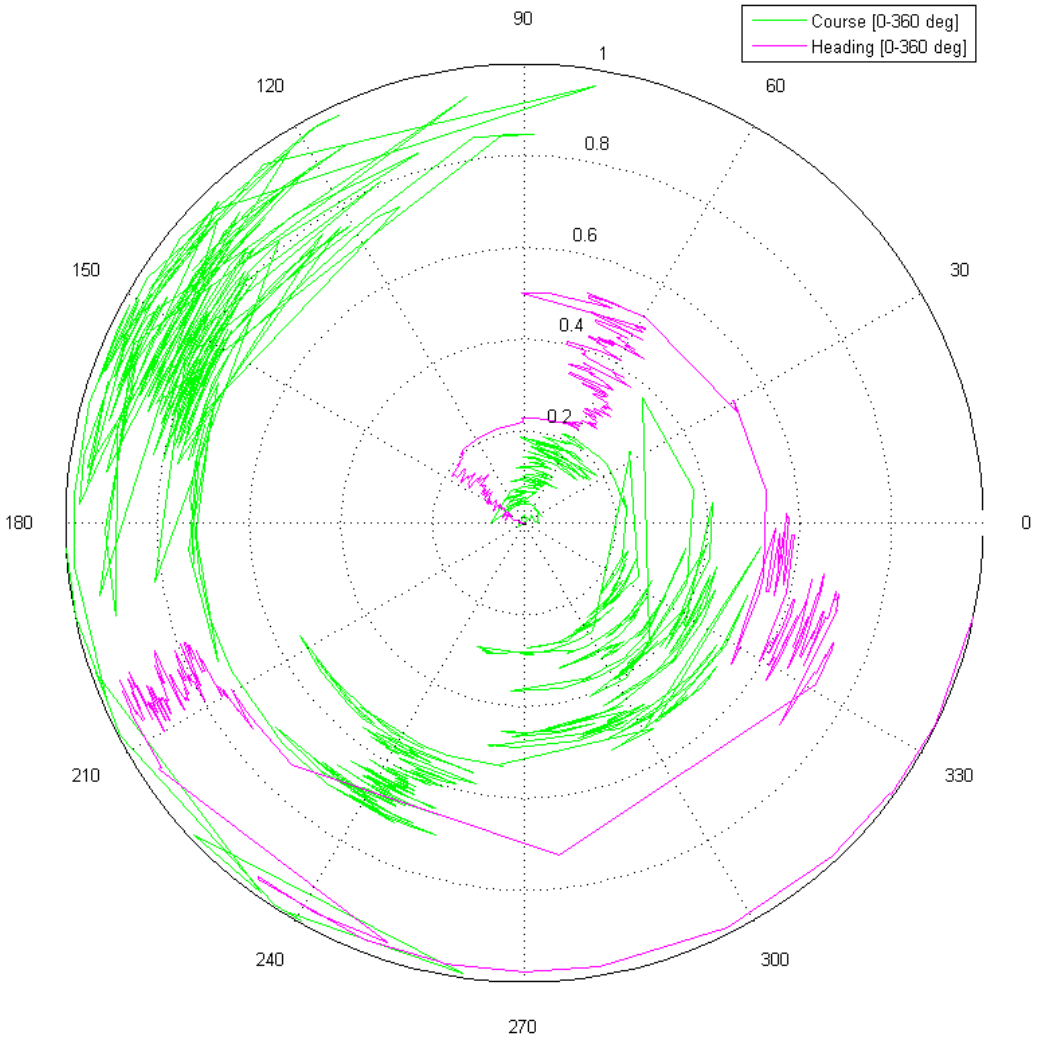


Figure 7.3: Course and heading plotted as a polar plot. The radius represents time.

7.2 Field test 1: First time in water and remote control

7.2.1 Purpose

The purpose of this *first time in water* test is to verify that the boat is seaworthy. This includes the following:

- The boat must float at the correct height and level.
- The boat must have a positive rightening momentum up to 90°.
- The boat must not take on water.

Secondly, the purpose is to verify the manual remote control capabilities of the boat by sailing it through the water.

7.2.2 Basic information

- Date: 05.04.15
- Location: Skansen, Trondheim
- Conditions:
  - Wind: 6 m/s rising to 12-14 m/s at the end
  - Weather: Rain

7.2.3 Execution and results

The test began by setting up the boat on shore. This work out as planned, but the weather conditions were starting to get worse. When the battery was plugged in, the system started properly and remote control was achieved on shore.

When the boat was fully rigged the system was shut down on purpose before testing the boat for water leakage. This was to minimize damage to the circuit boards in case there was a leak and to save the battery for later.

The boat was put on water and proved to stay afloat in the correct height. Furthermore, it stayed level and when tilting the boat 90 °there was still a significant rightening momentum as expected.

The boat was then brought ashore and hatches were opened to inspect for water leakages. There was spotted a small leak from one of the four bolts holding in place the keel in place. Further more it had started raining heavily and the wind speed was approximately above 10-12 m/s. Thus the rest of the test was cancelled.

After the the boat was brought back home and hatches were opened again, there was noticed a significant leak into one of the electronic compartments. The circuit boards were taken out, dipped in ethanol and left for drying. The circuits suffered no damage, but it

was clear that the compartments did not meet the requirements. This was a major setback as precious time, which in the following should have been used on control algorithms, had to be used to address this issue.

Figure 7.4 and Figure 7.5 shows some pictures from this test.

Figure 7.4: The boat on water.

Figure 7.5: The proud author and his sail boat.

7.3 Field test 2: Remote control

Figure 7.6: The boat sailing in close reach, seen from upwind.

7.3.1 Purpose

As the previous test partially failed, the current test share some of the purpose of the previous. In the current test, remote control is to be achieved. In specific, this includes the following:

- Determine the sailing capabilities of the boat.
- Determine the keel configuration that optimizes sailing capabilities.
- To reveal satisfactory function of the sail drives and sheets.

7.3.2 Basic information

- Date: 29.04.15
- Location: Kyvannet, Trondheim
- Conditions:
  - Wind: Less than 3 m/s
  - Wind direction: Towards north-east
  - Weather: Cloudy, chance of light rain

7.3.3 Execution and results

The test was conducted at *Kyvannet* in Trondheim. This is a freshwater lake. Here, the environment is more controlled in absence of other boats and ocean currents. This particular day, the wind speed was very low. However, it was sufficient to test the boat.

The boat was set up on shore. A small rescue dinghy was also inflated. The boat was towed out on the lake as it was launched in a wind shadow. The boat was remotely controlled from the rescue dinghy.

For this particular test, a common model plane radio control system was used. This was sufficient for the purpose of this test. The hardware framework would be verified in the next field test.

In the first run the boat sailed okay, but it was not properly balanced in respect to moments about the keel. There were too much sail area aft of the keel, causing the boat to turn into the wind. Hence the keel needed to be moved towards the aft. As the keel bracket was designed to allow many keel configurations, the keel can be reconfigured in a few minutes.

In the second run the keel proved to be properly adjusted and the boat sailed very well. It was easily controllable and stable in all points of sail in the given conditions.

Furthermore, the sail drives and sheets proved to work well. There are usually two main problems regarding sail drives on remotely controlled sail boats:

- The sheets sometimes get stuck and tangled into themselves or other objects on the deck.
- In very low winds, such as during this test, the friction between the sheet and pulleys can overcome the wind force acting on the sail. This causes the sail to not extract out to the correct position.

During one hour of testing, these problems did not occur.

All in all, the test went well and the boat was ready for the next step.

Figure 7.7: The boat sailing in close reach, seen from downwind.

Figure 7.8: The boat running downwind.

Figure 7.9: The boat sailing close hauled.

7.4 Field test 3: Heading hold

Figure 7.10: The boat remotely controlled using the hardware framework and user interface.

7.4.1 Purpose

The previous test revealed good sailing capabilities. The purpose of this test was hence to prove the capabilities of the hardware framework and achieve "heading hold" based on magnetometer readings. In more specific:

- Prove that the framework's data link operates with sufficient range and throughput in a stable fashion.
- Prove the framework's remote control capabilities.
- Prove that the framework can be used to control the boat in an automated fashion.
- Prove the frameworks capability to calibrate the magnetometer at test cite
- Prove that the attitude compensated compass reading can be used for "heading hold"
- Prove the functionality and usefulness of the *synchronized variables*. In other words use these to tune the "heading hold" control law in real time.
- Find tuning parameters for heading control by compass

7.4.2 Basic information

- Date: 14.05.15
- Location: Kyvannet, Trondheim

- Conditions:
  - Wind: 1-5 m/s (gusty)
  - Wind direction: Towards south-east
  - Weather: Cloudy, light rains

7.4.3 Execution and results

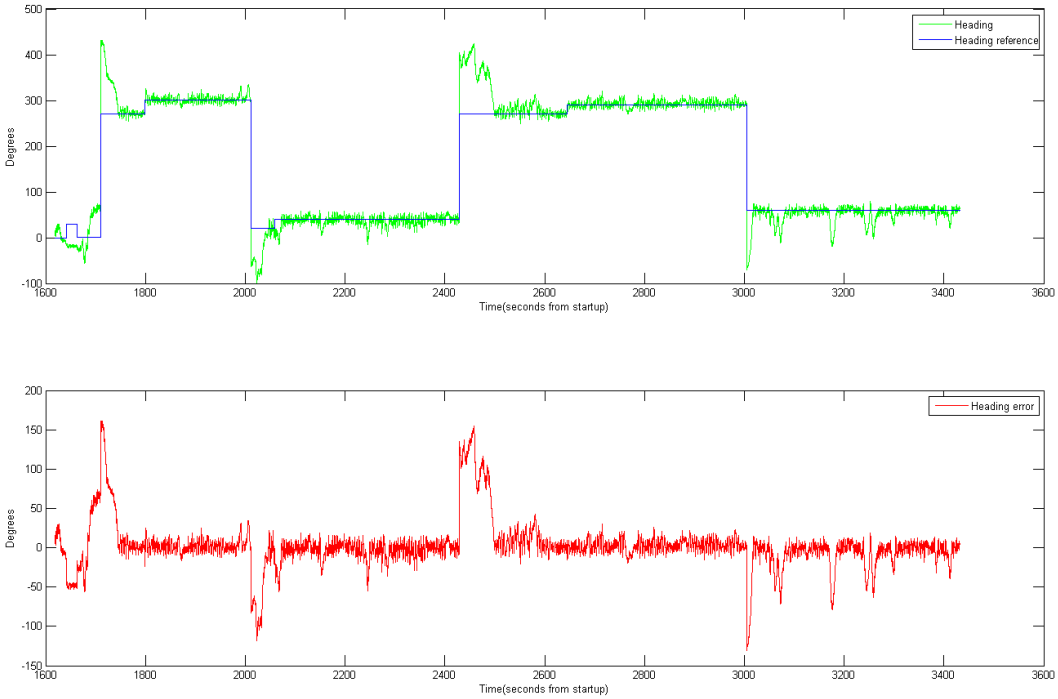


Figure 7.11: Heading reference, measured heading and heading error during "heading hold" based on magnetometer readings[1].

The test was conducted at *Kyvannet* as before. Also, the rescue dinghy was inflated and used as before. North-westerly winds, as it was this day, are not the best suited at Kyvannet with respect to the landscape. This will cause the wind to flow over some high grounds and down onto the lake, this causes gusty conditions and some variations in the wind direction. However, the test was still conducted with success.

Compass calibration After the boat was fully rigged and ready, the user interface was used to initiate the compass calibration algorithm. The boat was moved by hand in 3D rotations in order for the algorithm to sample a sufficient data set. Afterwards, the *finished* button was pushed and the boat calculated the calibration values based on the sampled data set. The results were displayed on the user interface and the values looked reasonable in comparison to desktop tests. The values would later prove to be correct as heading was correctly calculated during sailing.

[1]Notice that heading values are limited to the range [0 °,360 °). In this figure the heading values have been allowed to go outside this range for illustrative purposes. For values outside the range, true values are obtained by subtracting or adding a complete round(360 °).

Remote control After completing the calibration routine on shore, the boat was launched into water and towed to the opposite side of the lake where the wind was more stable. Here the user interface was used to control the boat. This worked well. However, sometimes capacitive touch screens perform poorly when they are covered with rain drops, and this has been an issue when using the same smart phone in other situations. Luckily, this did not become an issue this particular day as the rain was not heavy. It is also more difficult for the user to control the boat without physical knobs and sticks. The set-point indicators proved to be a nice indicator for the data link latency.

Data link performance During the remote control session, the data link proved to work sufficiently. The system felt responsive and the signal strength indicators indicated sufficient signal strength. However, the relay station was located at the floor in the dinghy and when the distance between the relay station and the boat became large, it was necessary to hold the relay station about 50 cm above the water level to get the same signal strength. The entire test was conducted with the maximum transmission power available in the radio modules. The transmission power can be set for both the relay station and the boat by the user interface ranging between 1 mW to 316 mW, see Section 5.2.3.

Tuning and variables After controlling the boat remotely, the user interface was used to set tuning parameters and a heading reference. The heading controller was turned on and new tuning parameters and references could be set while the controller was running. This worked well and made it easy to obtain tuning parameters.

Figure 7.12: The boat in "heading hold mode". The main sail angle is clearly released outwards by the controller in order to make the boat turn right away from the wind.

"Heading hold" After the initial guess the boat was immediately able to hold the heading based on the magnetometer readings. Figure 7.11 shows the heading reference, the measured heading and the error, while tacking against the wind. As the magnetometer

readings are very noisy, it was not expected that "heading hold" based on these measurements directly would not work that well. However, it was clearly visible that the boat had a *nervous* behaviour and could have sailed much more efficiently with a more precise and smooth measurement. From Figure 7.11 we can observe that the heading error oscillates with atleast $\pm$ 10 $^\circ$ after reaching the steady state behaviour.

In order for the controller to correctly alter the sails to set up a correcting moment, it needs to know whether the wind blows in from starboard or port side. As the wind direction sensor not yet had been fitted to the boat, this information was passed to the boat by the user via the user interface. This caused some of the tacks to take unnecessary long time as the wrong wind direction sometimes was passed by the user.

Unexpected fault After testing "heading hold" for about 30 minutes, the boat unexpectedly stopped sending updates about heading and the heading was not maintained. The boat was restarted via the user interface, the calibrated magnetometer values generated at the beginning of the test were loaded from file(these are automatically saved), the tuning variables were reset and the boat was quickly up and running again. After the boat was brought home, the logs were downloaded and quickly revealed which line of code that had caused the controller thread to fail. This proves the importance of having a log.

7.5 Field test 4: Waypoint tracking

7.5.1 Purpose

The purpose of this test was to achieve path following. This was the first test where the GPS antenna and wind direction sensor assembly was fitted to the boat. Hence, for the first time it was possible for the sails to operate in a fully automated mode. The purpose of the test is summarized as follows:

- Test and achieve fully automated sail trim.
- Test and achieve all operation modes:
  - Semi-manual
  - Heading hold
  - Waypoint tracking
  - Path following

7.5.2 Basic information

- Date: 31.05.15
- Location: Jonsvannet, Trondheim
- Conditions:
  - Wind: 0-6 m/s (alternating)
  - Wind direction: Towards east
  - Weather: Raining.

7.5.3 Execution and results

Unlike Kyvannet, there is no pier or table at Jonsvannet. Hence the boat was prepared while hanging from a three and launched from the rescue dinghy. Jonsvannet is a large lake($>14\text{km}^2$) and waves accumulates on the surface. While preparing the boat the wind blew steady, but just after launch it stopped. After half an hour the wind suddenly picked up a lot. Figure 7.13 shows the GPS data from the test.

The automated sail trim was promising when sailing close-hauled, at beam reach and at broad reach. However, in the somewhat rough conditions, the boat was not stable when running downwind. In fact, the boat would turn back into reach when trying to run downwind. This is visible in Figure 7.13 along the green path, where several attempts to run downwind(towards the right in the figure) were made.

This is not a problem as it is just a matter of reducing the main sail area to match the wind conditions. Bermuda rigged sail boats are in general somewhat unstable when running down wind. This is mainly due to the moment generated by the main sail as it is not centered, but protrude sideways. Hence the rudder must be large enough to overcome

this torque, when the wind is strong this is not the case. Reefing the main sail reduces the torque generated by the main sail and the rudder will suffice to maintain the running course. The boat has the possibility to reef the main sail by rolling it onto the mast, but must be done manually before launch. This is further tested in Section 7.6.

To make matters worse, the boat had a struggle to pass through the eye of the wind and change tack. The sail servos are very slow compared to the rudder servo. Hence, when the boat is commanded to do a sharp turn into the wind, the rudder is moved respectively almost at an instance. However, it takes time before the sail trim moves to a position where it allows the boat to turn sharply into the wind. During this time the rudder slows the boat down as it is set to a max position. Hence, at the time when the rudder and sail trim are aligned, the boat has lost much of its speed. As the boat speed is drastically reduced, the rudder does not contribute in turning the boat at the critical phase just before passing the eye of the wind.

This is issue if further addressed in Section 7.6.

It was clear that some adjustments had to be made before the test could succeed and the test was called off.

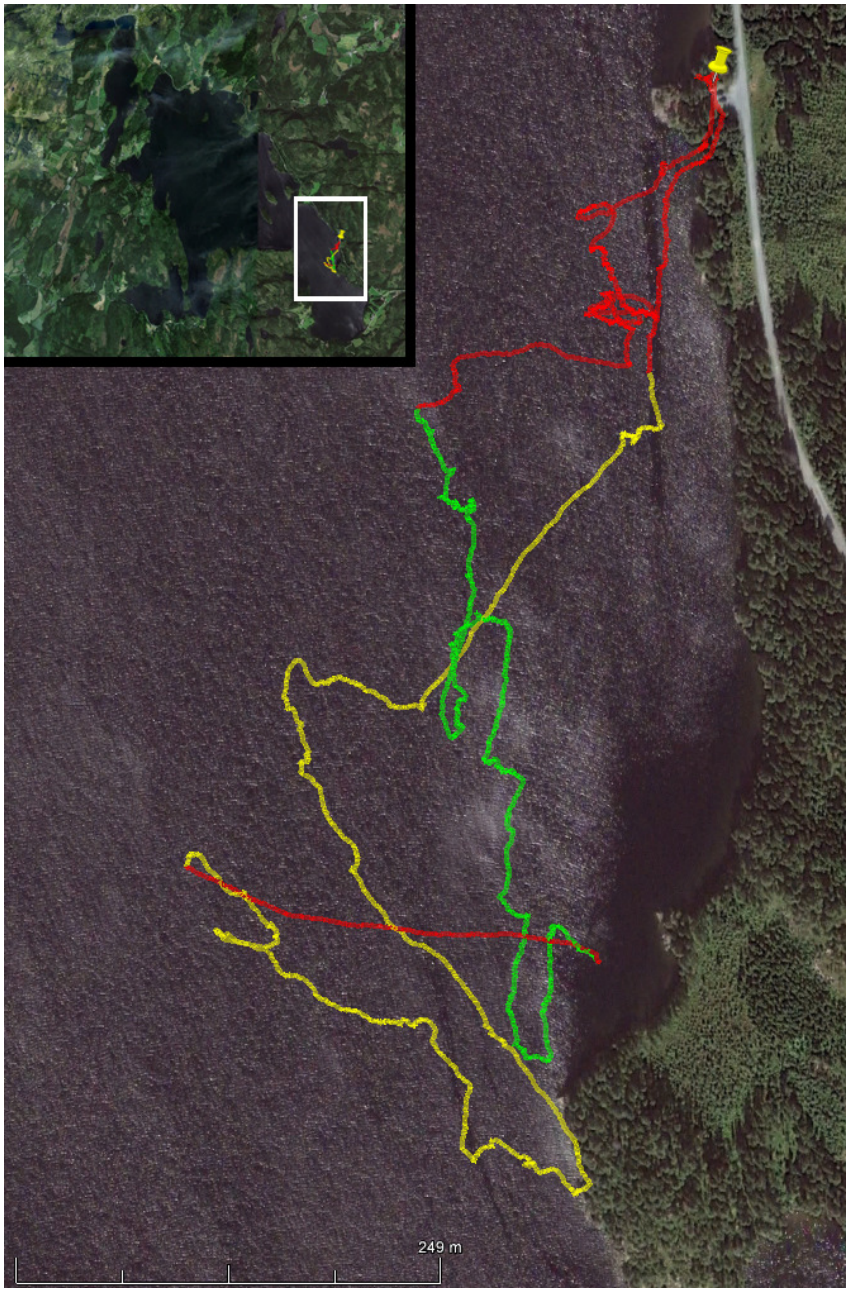


Figure 7.13: GPS data from field test 4 at Jonsvannet. Red indicates that the boat is being towed. Green indicates the beginning of the test where the boat is moving out onto the lake. Yellow indicates the end of the test where the boat is sailing back to launch cite. The launch cite is indicated by the placemark. The wind is approximately blowing towards the east (from the right). All data points have been offset equally such that the start of the route matches the launch cite. The GPS data is subject to slowly-varying error. Path following was not applied with success here. Imagery and mapping by Google-Earth^{TM}

7.6 Field test 5: Waypoint tracking revisited

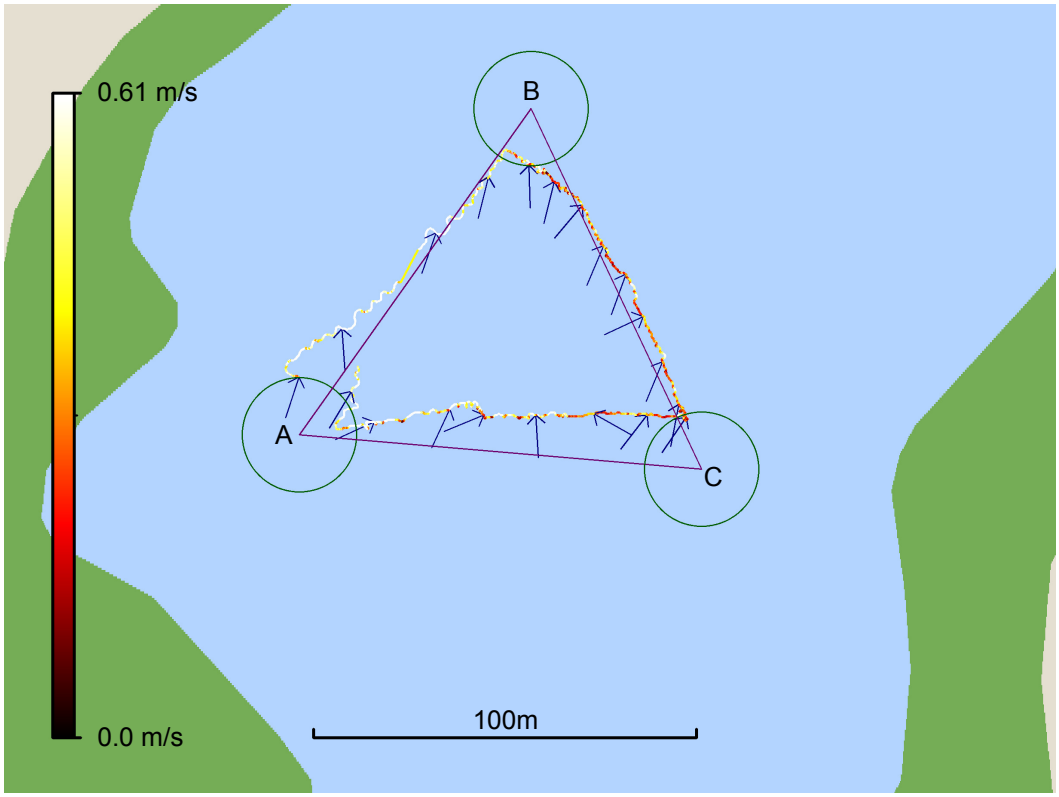


Figure 7.14: *Waypoint tracking*, round 1. Arrows indicate instantaneous wind direction measurement at the point the arrow is pointing at. The GPS data is prone to slow varying error and all data points are offset equally to illustrate the actual position on the map.

7.6.1 Purpose

The purpose of this test was the same as the previous test. Hence to achieve path following and address the issues revealed at the last test. In more specific:

- Test and achieve fully automated sail trim.
- Achieve stable running capabilities.
- Achieve tack changing capabilities.
- Test and achieve all operation modes:
  - Semi-manual
  - Heading hold
  - Waypoint tracking
  - Path following

7.6.2 Basic information

- Date: 2.06.15
- Location: Kyvannet, Trondheim
- Conditions:
  - Wind: 3 m/s
  - Wind direction: Towards north
  - Weather: Cloudy, dry.

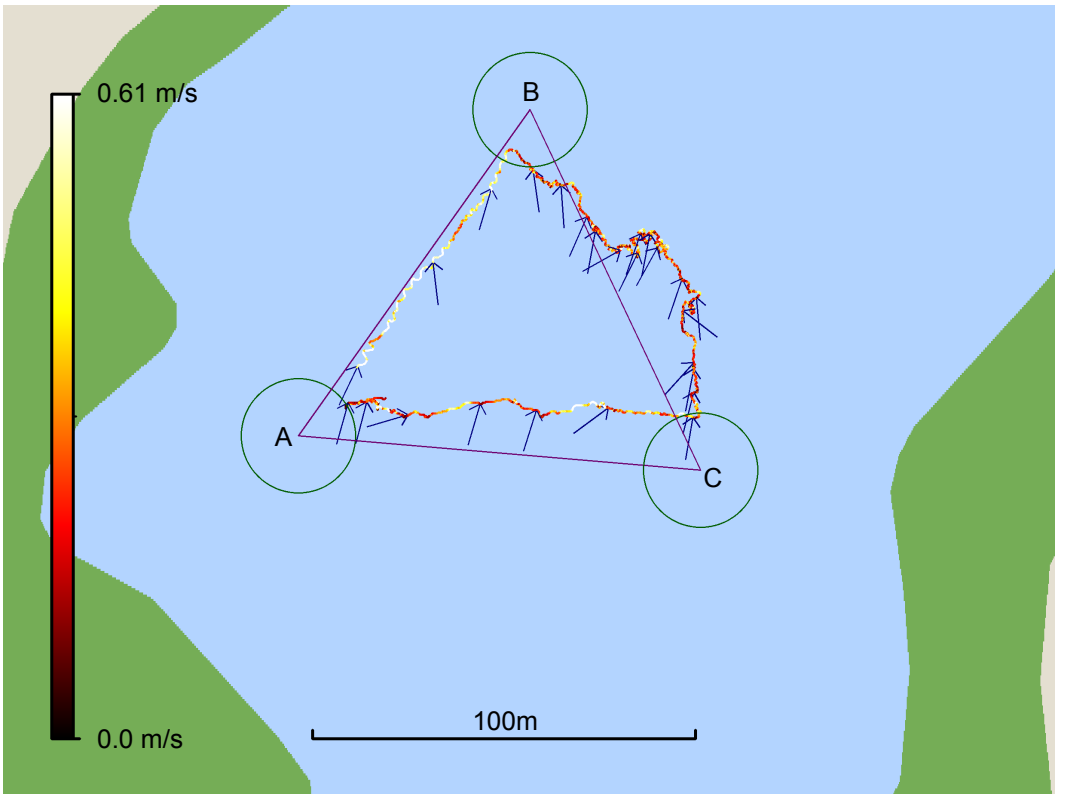


Figure 7.15: *Waypoint tracking*, round 2. Arrows indicate instantaneous wind direction measurement at the point the arrow is pointing at. The GPS data is prone to slow varying error and all data points are offset equally to illustrate the actual position on the map.

7.6.3 Execution and results

As the last test revealed stability issues when running downwind, the boat was set up with the main sail partially reefed. The main sail area was effectively reduced by about 20%. Furthermore the rudder area was increased by by attaching a small aluminium sheet to the existing rudder. After launching the boat in the usual way at Kyvannet, the boat was towed out by the rescue dinghy.

Running stability The boat was put into the *Semi-manual* mode and sailed around for a while. The boat was now able to run downwind in a controlled fashion. Furthermore it was able to change tack very efficiently.

Automated sail trim The automated sail trim was working as expected and performed well. In addition the *extra running stability* trim option was working. From the settings tab the threshold for this feature was set. See Section 5.2.5. However in automated modes, the *extra running stability* trim option did not function as intended. This as the feature was deactivated by large overshoots in *heading hold* when running downwind. Hence this feature should be activated not only with respect to the relative wind direction, but also to the desired heading.

Heading hold Even though the boat now was more than controllable when running downwind, the heading controller could have been improved for this point of sail. The controller was producing large overshoots in this point of sail as seen in Figrue 7.14 and Figure 7.15.

Position measurements Experiments with the GPS module have revealed the following:

- The GPS data was prone to a large error in position.
- The error is in magnitude of 100-1000 meters.
- This error varies with time very slowly, and hence it can be regarded as a constant offset.

There as not been time to investigate this issue further. In order to compensate for this error during testing, two synchronized variables have been made available in the settings tab to offset the measurement used by the system. This was done in the beginning of the test and the rest of the test was carried out without further concern.

Waypoint tracking Using the the settings tab in the user interface, a triangle path was laid out over the lake. The triangle was rotated, or attempted rotated, such that one of its sides were parallel to the wind. In this way, all sides of the triangle would remain feasible when sailing clockwise around it. In retrospect, the triangle should have been rotated a bit more in the clockwise direction. The first round of *waypoint tracking* is seen in Figure 7.14 and the second round in Figure 7.15. Both rounds are shown on an aerial photo of Kyvannet in Figure 7.16.

In the first round the wind was blowing steady and the boat was able to sail around the triangle at ease. In the second round the wind had started to fade as it was getting late. This explained the odd behaviour between corner B and C in round 2, as the wind was momentarily too faded in order for the boat to sail into it.

Path following After testing *waypoint tracking*, the lookahead-based steering algorithm was put into action. However, the wind was now faded too much to draw any conclusions on the matter.

Figure 7.16: *Waypoint tracking* at Kyvannet. The GPS data is prone to slow varying error and all data points are offset equally to illustrate the actual position on the map. Imagery and mapping by Google-Earth^{TM}

Chapter 8

Discussion

This project has been comprehensive, but determination and enthusiasm made it possible to achieve the goal. There are some remarks worth mentioning regarding the execution of this project.

8.1 Risk assessment and timing problems

To succeed with this project, the goal of autonomous sailing as defined in Chapter 1 should be reached. However, there was always a possibility that the goal would be partially fulfilled or not fulfilled at all. Hence, prior to starting with this master thesis, it was clear that tasks had to be conducted in a certain way in order to maximize the chances of achieving the goal.

The plan was to achieve the remotely controlled boat quickly. To be more specific, it would be remotely controlled using the implemented hardware framework and user interface described in sections 4 and 5. In this way, work could start making low level control laws that would provide a base for the next stage. The next stage would be the path planning stage, when most of the time should have been spent on path planning, algorithms and tuning

During execution of the project, achieving the remotely controlled boat proved to be too time consuming. During the last half of this master thesis, it was clear that completing an autonomous sailing boat on time had become a growing challenge. The problem was not that there was little time, but how that time was spent. In this project, several choices along the way led to inappropriate time consumption. It's hard to trace this back to a single fault, but answers to when and why these choices were made are to be discussed.

In general, the following statements have been made:

Keeping it simple The project has been an exciting one. It was easy to get carried away by all the possibilities it offered. At first it was desirable to build a more complex boat, with more features. Simplifications were made, but every omitted feature felt like a loss. Realizing that an absolute minimal sailing boat was the best approach, within the scope of

this thesis, happened at a late stage.

Optimistic time estimates and slack Another problem in the planning and execution of the project, was that time estimates for some tasks were too optimistic and more time slack should have been planned. Planning more slack could have contributed to keeping it simple, as in the above paragraph.

Optimistic ideas and "doing it yourself" With confidence gained from previous projects, the starting point for the project was based in a belief that almost anything could be done in-house within reasonable time. This state of mind led to unrealistic expectations about the complexity and duration of planned tasks. The first test conducted in water became a reality check with respect to the harsh environment at sea.

In the following, how these general problems interfered with specific parts of the project are discussed.

8.1.1 Mechanics

As this is a master thesis within cybernetics engineering, the design and production of mechanical components are not the most relevant aspects. Still, a good portion of time has been spent on making the boat from scratch.

It can be argued that basing the boat upon a pre made boat, as in a remotely controlled hobby/toy sail boat, would have been a time saver. However, a lot of adapting would still be required and it is not probable that this approach would have been better.

The main mechanical challenges have been:

1. Ensuring waterproof housing for electronics.
2. Reducing the risk of loosing the boat by making it waterproof and unsinkable.
3. Ensuring reliable operation of the sail rig.

Challenge 1 required a second attempt as field test 1 revealed severe flaws in the first design. Challenge 2 was solved the first time around, but to accommodate the new waterproof electronic compartments, changes had to be made. Hence, a lot of the mechanical work was done two times over and it was the mechanical work where most of the unscheduled time was lost.

Solving challenge 3 proved to be more predictable, but simplifications were made to reduce the build time.

8.1.2 Hardware framework

The hardware framework has proved fully functional and more than capable of conducting the tasks at hand. The I/O node has provided low level interfaces for the GPS, the wind

direction sensor, the servos, the inertial measurement unit and battery management. The use of a minimal scheduler made it easy, modular and maintainable to gather and pass data between this hardware and the main computer. The main computer has provided high level processing of the data and allowed for many features, and control algorithms, to reside. The structure of the framework made it easier to produce the necessary software required for the application.

However, the hardware framework could have been simplified. In the final product, it is clear that the distributed solution with CAN-bus functionality was unnecessary. A simpler protocol for exchange of data between the I/O node and the main computer could have been used. This could have simplified software and allowed for higher refresh rates for measurements and servo set-points.

To answer why the distributed solution was included in the first place, it can be pointed back to the system requirements in Section 4.1.1 where flexibility and expandability were important system requirements. As the system did not grow, hardware wise, as much as expected, these system requirements did not imply as expected. In fact, the final system only has two physical nodes on the network.

8.1.3 User interface

In Section 5, different user interfaces are discussed. The choice fell on a smart-phone user interface that required a relay station. The user interfacer has proved to be a powerful tool in field tests. Without the user interface in the selected form, testing would have been significantly less practical.

During the implementation of the relay station, some time was spent iterating unforeseen bugs. However, this is something you have to expect. A small amount of time could have been saved by making the user interface less graphically appealing.

8.2 Practical considerations

In the execution of this project, two particular practical considerations have noted themselves.

8.2.1 Individual project

In this type of work, there are a lot of things happening *"behind the scenes"*, as in work that is not worth mentioning in this document. Examples of such work are practical fiddling with mechanics, basic programming work, and preparations for field tests. For that reason, completing this project as an individual has been challenging. It is possible that a team of two persons could have completed this project with more than twice the accomplishments, as practicalities would be shared between the two.

8.2.2 Threshold for field tests

As this project work is about prototyping a physical system, testing has been essential. Through testing, experience is gained and the more the better.

After adjustments were made to ensure waterproof electronic compartments, the boat became significantly more difficult to handle. That is, pre and post testing, there are some work with installing and uninstalling the electronics etc. Additionally, the rig, keel and rudder must be installed/uninstalled at test cite and this requires more equipment at test cite. This creates a certain threshold for conducting a field test. It becomes more important that a field test provides positive results and hence the field test is often postponed to days with better weather conditions.

A different design could have eased the testing proses. Hence, more field tests and experience could have been gained.

Chapter 9

Conclusion

A small scale sail boat has been designed, built and automated. It is capable of sailing a predefined feasible track with no user interaction, using only the wind as propulsion.

After some iterations, the mechanical works of the boat fulfills the task. Time was lost since faults in the original design led to more mechanical work than first anticipated. A different design could also have simplified practical considerations regarding testing which could have increased the amount of field tests.

The boat has been equipped with flexible and powerful hardware for computerized control. This hardware gathers measurements, runs actuators and execute both high and low level control algorithms in a structured and expandable environment. Additionally, a smart phone user interface was made that proved to be a powerful tool in field tests.

A control allocation scheme for the sail boat has been proposed. A wind direction sensor was made and its measurements are used to control the lift in the sails. This scheme has proved efficient in tests. Some work remains to verify experimental work on increasing running stability.

The control allocation scheme has been used to achieve a semi manual mode witch lets the user sail the boat in all feasible directions without keeping sail trim in mind. A controller uses the control allocation scheme to close the heading loop. This controller are used to close the position loop by a waypoint tracker. Work on making a path planner for sail boats have been conducted. This has resulted in a complete and optimal path planner that takes the constraints of sailing and the geographical constraints into account. The path planner algorithm also accommodate ocean currents and moving obstacles.

Several field test have been conducted and experience has been gained in all. In the final field test the boat was able to sail the feasible track autonomously. More tests were desirable, but time was limited and the finish line was drawn.

9.1 Further work

This is a comprehensive project and there will always be room for improvements. In the following, some work notes itself as a natural start for further work.

Improving performance
It is believed that a little more tuning and small refinements quickly can further improve the boat's performance. This can be done by conducting more field tests and does not require any further implementation.

Path following
After succeeding with waypoint tracking, there was no time left to verify the path following controller. This should be a first choice for further work. It is expected that this should work out well and that only minor changes may be necessary.

Path planner in the loop
The next step is to integrate the proposed path planner. At this point, the user can click a selected location on the user interface map, and the boat will autonomously sail to that location. Path planning for sail boats may be further explored by further improvements of the path planner.

Bibliography

[1] An3192 application note; tilt compensated electronic compass. https://www.pololu.com/file/0J434/LSM303DLH-compass-app-note.pdf. Accessed: 6-05-2015.

[2] Angle of attack. http://en.wikipedia.org/wiki/Angle_of_attack. Accessed: 20-12-2014.

[3] Earth's magnetic field. http://en.wikipedia.org/wiki/Earth's_magnetic_field. Accessed: 6-05-2015.

[4] Equirectangular projection. https://en.wikipedia.org/wiki/Equirectangular_projection. Accessed: 16-06-2015.

[5] Hanse yachts. https://www.hanseyachts.com/. Accessed: 20-12-2014.

[6] Hard-iron and soft-iron calibration. http://dkc1.digikey.com/us/en/tod/Honeywell/Hard-Soft-Iron-Calibration_noaudio/Hard-Soft-Iron-Calibration_noaudio.html. Accessed: 6-05-2015.

[7] The international robotic sailing regatta. http://www.sname.org/SailBot/home/. Accessed: 20-12-2014.

[8] Kartverket. http://kartverket.no/Kart/Gratis-kartdata/. Accessed: 1-05-2015.

[9] Keel. http://en.wikipedia.org/wiki/Keel. Accessed: 20-12-2014.

[10] Mercator projection. https://en.wikipedia.org/wiki/Mercator_projection. Accessed: 16-06-2015.

[11] The microtransat challenge. http://www.microtransat.org/. Accessed: 20-12-2014.

[12] Nmea 0183. http://en.wikipedia.org/wiki/NMEA_0183. Accessed: 27-04-2015.

[13] *Proceedings of the 4th International Robotic Sailing Conference.*

[14] *Proceedings of the 5th International Robotic Sailing Conference.*

[15] *Proceedings of the 6th International Robotic Sailing Conference.*

[16] Rudder. http://en.wikipedia.org/wiki/Rudder. Accessed: 20-12-2014.

[17] Rules of world robotic sailing championship 2012. http://www.microtransat.org/wrsc2012/wrsc2012rules.pdf. Accessed: 20-12-2014.

[18] Sailing. http://en.wikipedia.org/wiki/Sailing. Accessed: 20-12-2014.

[19] Standard streams. http://en.wikipedia.org/wiki/Standard_streams#Standard_output_.28stdout.29. Accessed: 01-04-2015.

[20] A star search algorithm. https://en.wikipedia.org/wiki/A*_search_algorithm. Accessed: 16-06-2015.

[21] Vikingskiphuset, kulturhistorisk museum. http://www.khm.uio.no/besok-oss/vikingskipshuset/. Accessed: 1-12-2014.

[22] World robotic sailing championship 2012. http://www.microtransat.org/wrsc2012/. Accessed: 20-12-2014.

[23] World robotic sailing championship 2014. http://wrsc2014.com/. Accessed: 14-12-2014.

[24] R. W. Beard and T. w. McLain. *Small unmanned aircraft*. Princeton, 2012.

[25] M. Breivik and T. I. Fossen. Guidance laws for autonomous underwater vehicles. *InTech*, 2009.

[26] T. I. Fossen. *Handbook of marine craft hydrodynamics and motion control*. Wiley, 2011.

[27] M.-A. R.-R. B. G. Frédéric Plumet, Clément Pêtrès and S.-H. Ieng. Toward an autonomous sailing boat. *IEEE Journal of Oceanic Engineering*, 2015.

[28] I. Grigorik. *High Performance Browser Networking*. O'Reilly, 2013.

[29] L. Xiao and J. Jouffroy. Modeling and non-linear heading control for sailing yachts. *OCEANS*, 2011.