Development of an autonomous sailboat using Arduino

FISE 24

October 1, 2023

Virgile Pelle
virgile.pelle@ensta-bretagne.org

I would like to express my thanks to my internship supervisor Jian Wan, for his welcome and the opportunity for this internship. He was very committed and his advice were precious.

I am grateful to my school teacher, Luc Jaulin, for facilitating this internship opportunity.

I also want to thanks my colleagues and friends, Augustin Morge, Laurent Droudun and Mathieu Drillon for their teamwork. Together we gave ourselves a boost.

Contents

Acknowledgment . . . 1
Abstract . . . 3
Résumé . . . 3
Keywords . . . 3

1 Introduction **4**
1.1 The university . . . 4
1.2 Autonomous sailing boat . . . 4
1.3 Purpose of the internship . . . 4

2 Hardware **6**
2.1 Sensors . . . 6
2.1.1 Wind sensor . . . 6
2.1.2 Inertial measurement unit (IMU) . . . 6
2.1.3 GNSS receiver . . . 7
2.2 Actuator . . . 7
2.3 Communication components . . . 8
2.4 Wiring . . . 9

3 Software **11**
3.1 Arduino . . . 11
3.1.1 Sensor libraries . . . 11
3.1.2 Other libraries . . . 12
3.1.3 Main program . . . 13
3.2 Control algorithm . . . 13
3.2.1 True wind . . . 13
3.2.2 Mission Planner . . . 14
3.2.3 Follow line controller . . . 14

4 Monitoring **15**
4.1 Communication protocol . . . 15
4.2 User interface . . . 16

5 Results **17**
5.1 Canal test . . . 17
5.2 Lake testings . . . 18
5.3 Improvements . . . 19

6 Conclusion **21**

A Wiring Diagram **23**

B Class diagram **24**

Abstract

In this report I will review and analyse the work I did during a 4 months internship at Aston University in Birmingham. The subject of this internship was to control a sailing boat using a low consumption micro controller. The complexity of this project was to read all the different sensors information and to be able to control autonomously the boat using only an Arduino.

I had the opportunity to test the system several time on a small lakes. It was the occasion of testing all the sensors in real condition and see if the controller was appropriate.

I also developed a server to gather the data via an wireless connection to the boat. Then I was able to monitor the boat position and behaviour directly on a map.

Résumé

Dans ce rapport, je vais passer en revue et analyser le travail que j'ai effectué au cours d'un stage de 4 mois à l'université d'Aston à Birmingham. Le sujet de ce stage était de contrôler un bateau à voile en utilisant un microcontrôleur à faible consommation. La complexité de ce projet était de lire toutes les informations des différents capteurs et d'être capable de contrôler le bateau de manière autonome en utilisant seulement une Arduino.

J'ai eu l'occasion de tester le système à plusieurs reprises sur un petit lac. C'était l'occasion de tester tous les capteurs en conditions réelles et de voir si le contrôleur était approprié.

J'ai également développé un serveur pour collecter les données via une connexion sans fil au bateau. J'ai ainsi pu suivre la position et le comportement du bateau directement sur une carte.

Keywords

Sailing Boat, Autonomous Control Algorithm, Arduino, Sensor Integration, Real-Time Monitoring.

1 Introduction

1.1 The university

My internship took place in Aston University. It is situated in the centre of Birmingham, the second largest city in the UK located in the middle west of England. According to the edurank's website, Aston University is in the top 250 of the best university of the Europe. Indeed, with its 24 hectare campus, the university bring together a lot of faculties ranging from engineering to social sciences via economics, law, medicine and many others. In addition, the university emphasises social mobility, diversity and sustainability. As a result, it won the University of the Year award from the Guardian and also the price for the most inspiring building. [1]

Figure 1: Main building of Aston University

1.2 Autonomous sailing boat

Marine robotics is not a very developed field at the moment, so many companies are starting to do research in this area. Indeed autonomous boat would be a huge help for hydrographers and oceanographers for their research. Such boats could be also used for military marine defence, for rescuing shipwrecked and also to develop a more ecological system for freight transport.

1.3 Purpose of the internship

The purpose of the internship is developed into three axis:

1. To familiarise with various sensors such as IMU, weather vane, camera, GNSS receiver, Li-DAR and others for sensor data collection, filtering and fusion

2. To use Arduino/Raspberry Pi/NVIDIA boards to program control and learning algorithms through C++, Python and/or ROS1

3. To use some basic mechanical design and 3D print of sensor supporting mechanisms for the autonomous systems.

The main goal of this internship is to do some research on a sailing boat that has been used for remote control competitions in order to upgrade it and make it autonomous. The boat used for this project is shown on the Figure 2.There are some research that has already been done by others trainees in recent years. But this system was not robust enough and was too energy-intensive. That is why I have been asked to develop a new system using a much more energy-friendly embedded computer but thus much less performant. In the end the two

Figure 2: RC Laser boat

systems could be compared to know which one is the most efficient. The board chosen for this system is an Arduino board. This is a very famous and used micro controller in robotics field. It is low energy consumption and offers a lot of input and output. This is very useful because we will need a lot of sensors to acquire many data of the environment in order to control correctly the boat.

2 Hardware

The boat needs to be autonomous. That is why the boat needs to be equipped with many sensors to be able to understand its environment and to navigate. Some other components are also required for safety reason because the boat also have to be reached and controlled from the shore.

2.1 Sensors

2.1.1 Wind sensor

Figure 3: Weathervane Anemometer – 7911 – Davis Instruments sensor

The wind behaviour have to be precisely known to operate correctly a sailing boat. The wind sensor is composed of two sensors: a weather vane for its direction (in rad) and an anemometer for its speed (in $m.s^{-1}$). It was not possible to attach this sensor on top of the mast as on most of sailing boat to have an accurate measure. We decided to attached it on the front of the boat. But sometimes there was some strong and brief wind gusts bouncing on the sail and thus giving an incorrect perception of the wind direction. In order to avoid this, a first order low-pass filter was added on the wind direction measure (ϕ):

$$\phi = \phi_{measured} + (1 - \alpha)\phi_{previous}$$

With $\alpha = 0.2$.

2.1.2 Inertial measurement unit (IMU)

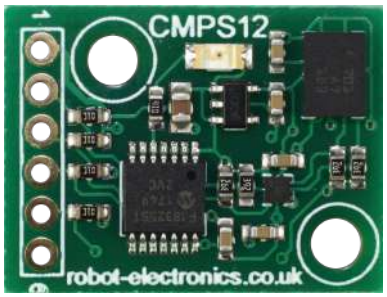

Figure 4: IMU - CMPS12

The heading of the boat is very important for the control algorithm. The boat evolve in a 2D plan but because the water is never completely flat and because the boat tend to roll on its longitudinal axis. The IMU needs to be tilt compensated. The selected IMU, the CMPS12, already have a tilt-compensation algorithm implemented in it. This IMU is automatically calibrated, the user is just required to perform simple movements to allow the CMPS12 to complete this. This will make the reading of the heading of the boat much more easier because there will not have any post processing of the data. Although the yaw reference for the IMU is set when powering it. So before launching the boat the IMU needs to be align manually using an external compass.

2.1.3 GNSS receiver

Figure 5: Grove - GPS Module *(without its antenna)*

The boat needs to be able to follow a list of given waypoints. This is why the boats need to have a GNSS receiver on board. The GNSS receiver chosen has a refresh rate of 1Hz. This is much enough because the boat will not sail too fast so it does not require more frequent updates.

2.2 Actuator

To control this sailing boat two servomotors are needed.

- One for the rudder: this one has to go to -45° to +45° whether the boat needs to turn right or left. The blue circle on Figure 6.
- One for the sail: this one can turn to 0° to 2160° (6 turns). Thanks to a system of pulley and a string attached to the end of the sail, the sail can be closed (aligned with the boat) and fully open (making a 90° angle with the boat). The red circle on Figure 6.

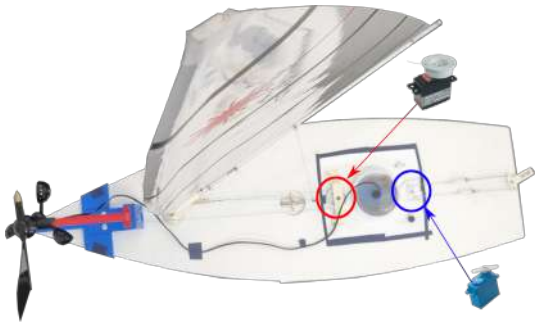

Figure 6: Top view of the boat with actuators position

2.3 Communication components

For safety reason and also in order not to lose the boat, it was important to be able to take the control of the boat from the shore at any moment. This is why a a radio receiver was added to the Arduino. As soon as the RC receiver gets a signal from the remote controller, it sends directly the command to the servomotors interrupting the controller program.

The problem of this system is that it is only a one way communication, meaning that the boat cannot send anything to the remote operator. Thus no real time feedback was possible.

Then I decided to replace it by an XBEE. Thus by plugging another XBEE on my computer I was able to control the boat but also to receive real time data about its mission. The XBEE is also using the 2.4GHz frequency so the range is not affected.

Figure 7: RC receiver

Figure 8: XBEE Module

2.4 Wiring

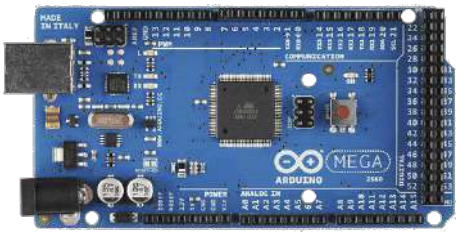

Figure 9: Arduino Mega 2560

The best Arduino for this project is the Arduino Mega because there is a lot of components to plug on it. This board has 54 digital pins, including 6 interrupts ones, 16 analog ones and 15 PWM ones. It also supports UART (4 ports), I2C (1 port) and SPI (1 port) communications protocols which will be needed for some components [2]

To simplify the wiring of every components, the Arduino board is equipped with a Grove-Mega Shield. Indeed this board supports standardized 4 pins connector (Signal 1,Signal 2,VCC and GND) which is very convenient for plugging and also avoid unwanted unplugging of some wires.

The servomotors require their own power supply. This is why an other shield is added to the previous one. It is the Adafruit servomotors interface. Servomotors wiring is then much easier and most importantly it has a dedicated 5V input for the servomotors.

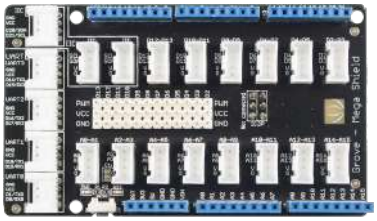

Figure 10: Grove-Mega shield

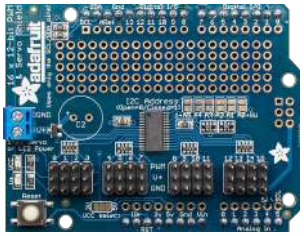

Figure 11: Adafruit servomotors interface

To power the whole system one battery with 2 output is used. Both output provide 12W (5V DC/2.4A). One output is connected directly to the servomotor interface and the other one to power the Arduino via its USB-B connector.

The first components to plug are the ones using interrupts pins because there is only 6 interrupting pins on this Arduino. Interrupt pins are very useful because they trigger a function as soon as a signal is received on this pin. The RC receiver needs 2 interrupt pins so it is plugged on D2 and D3 pins. The anemometer also need an interrupt pin that is why it is plugged on the D19 pin. But by doing this, a serial port become unusable (Serial1).

Most of the sensor needs a serial port to communicate with the Arduino. In this project the GNSs receiver, the IMU and the XBEE are using UART communication through a serial port. By default a serial port, using D0 and D1, is used to debug the program using a USB cable connected to the computer. Thus there is only 2 left serial port for 3 sensors. The XBEE is connected to D16 and D17 pins (Serial2) and the GNSS receiver to D14 and D15 pins (Serial3). The library SerialSoftware allows to emulate another serial port on 2 other digital pin. This library is used to connect the IMU to the Arduino board on pins D10 and D11.

For the wind vane, the sensor works with a variable resistance. To read the information it needs to be plugged on an analog pin (A14).

The SD card reader is using the SPI communication protocol and thus require 4 pins for data transmission:

- MISO (Master Input/Slave Output) on pin D50: the line for the slave to send data to the master.
- MOSI (Master Output/Slave Input) on pin D51: the line for the master to send data to the slave.
- SCLK (Clock) on pin D52: the line for the clock signal.
- SS/CS (Slave Select/Chip Select) on pin D53: the line for the master to select which slave to send data to.

The control of the servomotors are done via an I2C communication. It allow the master to send messages to several slaves. In this case the master is the Arduino board and the slaves are the 2 servomotors. Contrary to the SPI communication protocol, this one only requires 2 wire:

- SDA (Serial Data) on pin D20: the line for the master and slave to send and receive data.
- SCL (Serial Clock) on pin D21: the line that carries the clock signal.

3 Software

3.1 Arduino

There is many components to work with on this project. Moreover every components has its own way of sending the data. Trying to put everything in the same arduino code would have been quite impossible to do. So I decided to create libraries for every components. Then those libraries would be added to the main arduino code. On top of being more readable it is also very convenient for debugging because every sensor can be tested individually. All the codes are available on my GitHub [3].

3.1.1 Sensor libraries

Libraries are written in C++. Each library can contain one or several classes that will be accessible in the main arduino code once imported.

Every sensor has its own class:

- The GNSS class uses the TinyGPS+ library for obtaining the coordinates, the speed, the course over the ground and the current time via a serial connection. Many control algorithm does not use directly longitude and latitude, this is why there is also a function to get coordinates map on a 2D plane using a known latitude and longitude reference. This code is explained in the paragraph 3.2.2

- The IMU class is using the SoftwareSerial library to emulate a serial connection. The program has to request for every data wanted by sending a byte to the IMU and then the IMU will send the appropriate data. This program reads the yaw, the pitch and the roll of the IMU.

- The wind vane class is used to know the wind direction. To do so the program reads the analog value given by the potentiometer inside the wind wane. Then this value is mapped between 0° and 360°.

- The wind speed class is used to know the wind speed. The anemometer is wired to an interrupting pin and then for every revolution it makes, a function is triggered to increment a counter. And then by using the formula in the documentation for this sensor [4] we have the wind speed:

$$V = P(1.061/T)$$

  where V is the wind speed in $m.s^{-1}$ and P the number of revolution counted during the time T.

- The radio receiver class does not need to always be updated. Both channel (rudder and sail) are connected on interrupt pin. Those interruptions are triggered when a change in the PWN signal sent by the radio controller is detected. So the values are updated only when the radio controller is turned on. The program determines how long the signal for a channel is high and can then determine the desired position for the corresponding servomotor.

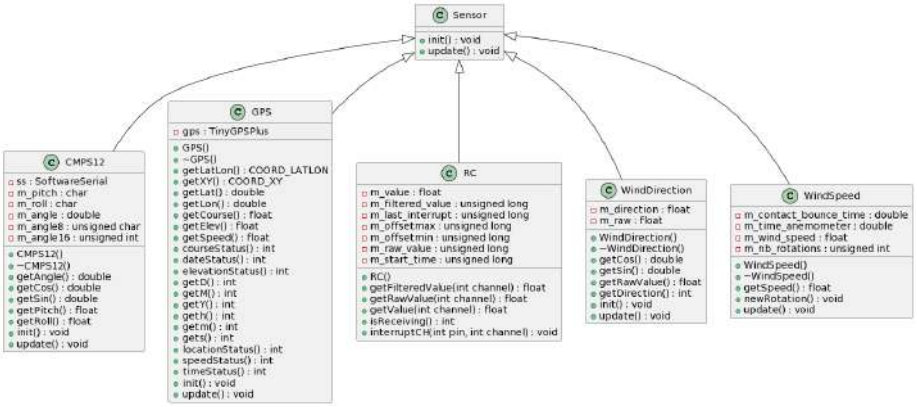

Figure 12: Sensor class diagram

To simplify the initialisation and the updating of every sensor they all inherit from a prototype object called "Sensor". The corresponding class diagram is shown on Figure 12. Thanks to this all sensors objects can be stored in a unique list and to update or initialise them the program just have to call the function *init()* or *update()* on every object in it.

There is also a class for the xbee. The XBEE communicates with the Arduino on a serial connection. To be able to discuss with an external xbee plugged on a computer, a communication procedure has been established. It will be explained in the communication section 4.1.

3.1.2 Other libraries

The main class is "Sailboat". This is the class representing the sailing boat. This class is making the code functionning easier because just by creating an object everything is already set up and it has access to every sensors and servomotors on the boat. Inside this class there is for example a unique function to update every sensors, and another one to update the position of the servos.

This class has a reference to a controller object. For the moment only 2 controllers have been developed:

- A "None Controller" which does nothing meaning that it does not send anything to the servomotors. It is the default controller and can be used before launching the boat or after a mission is finished, when the boat is being manipulated by someone.
- A "Follow Line" controller. This is the main controller used to follow a line defined with two GNSS coordinates. Its functioning will be explained in detail in the control algorithm section 3.2.3.

The controller is managed by a Mission Planner object. In this object a mission is saved as a list of waypoint. Knowing the current position of the boat it will set a new controller every time a waypoint is passed.

3.1.3 Main program

The main program is the only arduino code. This is the code to upload on the Arduino. When this code is uploaded on the Arduino it also include all the other libraries required seen before. During the setup function of the code all the sensors and the controller are initialised. And then the main loop will repeat until the boat is unplugged. Here is what is happening during the main loop of the program :

1. All the sensors are updated to have the current value and the controller compute new value for the actuators.
2. The servomotors position are updated accordingly.
3. All the value are logged for future debugging
4. The XBEE send the data to the remote operator.

3.2 Control algorithm

3.2.1 True wind

The wind sensor is attached directly on the boat, thus the collected data are in the boat's frame of reference. But in order to use the control algorithm those must be in the terrestrial reference frame. This is why it is important to calculate the true wind speed and direction (respectively TW_S and TW_D) [5]. We need the following information :

- the apparent wind speed (AW_S), measured by the anemometer
- the apparent wind angle (AW_A) in the boat reference frame, measured by the wind vane.
- the heading of the boat (H), it is the yaw measured by the IMU.
- the apparent wind direction (AW_D) will be calculated using H and AW_A.
- the speed and the course over the ground (respectively SOG and COG). Those information came from the GNSS receiver.

$$AW_D = H + AW_A$$

$$TW = \begin{pmatrix} SOG \times \sin(SOG) - AW_S \times \sin(AW_D) \\ SOG \times \sin(COG) - AW_S \times \cos(AW_S) \end{pmatrix}$$

$$TW_S = \|TW\|$$

$$TW_D = angle(TW)$$

3.2.2 Mission Planner

The objective of the sailing boat is to be able to follow a given mission. It can be to follow a simple line but also to follow a more complicated mission made of several line for example to make a path around a lake. The role of the mission planner is to update the line to follow regarding if the next waypoint has been reached. Whether it is for the controller or to check if a waypoint is reached we need to project the coordinates onto a 2D map. A reference latitude (ϕ_{ref}) and longitude (λ_{ref}) has to be chosen in the area of the mission.

$$\tilde{x} = R\cos(\phi\frac{\pi}{180})(\lambda - \lambda_{ref})\frac{\pi}{180}$$
$$\tilde{y} = R(\phi - \phi_{ref})\frac{\pi}{180}$$

Where ϕ, λ are respectively the latitude and longitude we want to project to obtain $\tilde{x}$, $\tilde{y}$ on a 2D plan. R is the earth radius.

Now that our coordinates are transformed into a a cartesian 2D plan all the calculation regarding the position of the boat are easier. To know if a waypoint is passed or not, let's consider a line from a to b and a boat representing by the point m (see Figure 13). The next waypoint (in our case b) will be considered as reached if $(b \vec{-} a)$ and $(m \vec{-} b)$ are in opposite direction. We can express that mathematically:

$$\text{if } (b-a)\cdot(m-b) < 0 \text{ then the waypoint b is reached}$$

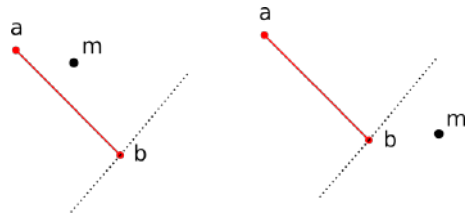

Figure 13: Determining whether a waypoint is passed or not

3.2.3 Follow line controller

For the line following algorithm, I chose Professor Jaulin's controller [6].

The controller takes in input the position of the boat m, its course θ, the direction of the true wind ψ_{tw}, the line ab and the hysteresis q used for close hauled sailing.

This algorithm outputs the angle of the sail $\delta_s max$ and the angle of the rudder δ_r. It also returns the hysteris q which has to be stored for the next iteration.

inputs : $m, \theta, \psi_{tw}, a, b, q$
e = det($\frac{b-a}{\|b-a\|}$,$m-a$) if $\|e\| > r$ then $q = sign(e)$ $\phi = \arctan(b-a)$ $\hat{\theta} = \phi - \arctan(\frac{e}{r})$ if($\cos(\psi_{tw} - \hat{\theta}) + \cos(\xi) < 0$) or ($(\|e\| - r) < 0$ and ($\cos(\psi_{tw} - \phi) + \cos(\xi) < 0$)) : $\hat{\theta} = \pi + \psi_{tw} - q\xi$ $\delta_r = \frac{\delta_{rmax}}{\pi} sawtooth(\theta - \hat{\theta})$ $\delta_{smax} = \frac{\pi}{2}(\frac{\cos(\psi_{tw} - \hat{\theta})+1}{2})^{\frac{\log(\frac{\pi}{2\xi})}{\log(2)}}$
outputs : $\delta_r, \delta_{smax}, q$

Table 1: Line following controller

4 Monitoring

4.1 Communication protocol

Thanks to the XBEE there is a bi lateral communication between the boat and a remote operator on the shore. I have established a communication protocol to interact remotely with the boat. This communication protocol is based on a slave/master communication in which the remote operator is the master and the boat is the slave. It means that the boat does not send anything on its own. It is always waiting for a message from the master (the operator) to respond something.

Message from the master are just a simple byte. Those commands are listed in the following table.

Byte to send	Function
0x69 (i)	Send boat informations (coords, wind etc.)
0x73 (s) → 0x6F (o)	Open a new log file and start logging
0x73 (s) → 0x63 (c)	Close the current log file and stop logging
0x6D (m) → ...	Receive a new mission
0x63 (c)	Enter control mode
0x71 (q)	Exit control mode
0x68 (h)	turn rudder left *(in control mode)*
0x6A (j)	close sail *(in control mode)*
0x6B (k)	open sail *(in control mode)*
0x6C (l)	turn rudder right *(in control mode)*

Thanks to this communication protocol once the boat is powered it is possible to check the sensors info, check the servomotors, start a mission and monitor it. But it was not very convenient because it was only in command line so very difficult to interpret informations

4.2 User interface

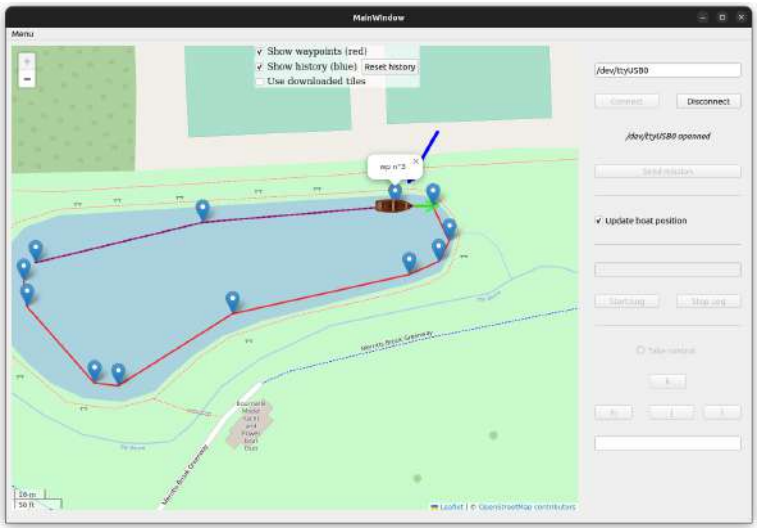

Figure 14: User interface for monitoring the boat

In order to make the process easier and to not make mistake, I implemented a user interface in python. You can find all the code and the installation procedure on my github [7]

The program of the user interface is directly connected to the boat via a xbee to be able to interact with the boat. In the background of the UI there is a script which gets all the data about the current mission and then upload them to a local database. This database is used by a website hosted in local to display the boat on a map. This is the map on the left side of the UI. This allows to monitor the boat position and its trajectory in real time. Moreover the wind is also shown on the map so it helps to see if the boat reacts correctly according to its target and to the wind.

Directly in the user interface it is also possible to take control of the boat, to enable the logging and to send a new mission.

5 Results

5.1 Canal test

Figure 15: First test on the canal

The first conducted testings were on a narrow canal. We decided to make the first tests here because it was very close to the university. For this test the boat just had to follow a simple line in the canal, from point 001 to 002 on the Figure 16. Here is what I learned during this test:

- I made a mistake by choosing the end point (002) on the river side to make the recovery easier.But because the canal was too narrow it ended up hitting the bank (blue circle on the Figure 16).
- When there is a big gust of wind on the side of the sailing boat it makes it roll a lot and thus the data from the IMU are irrelevant. This is what caused the right turn in the red circle on the Figure 16. In order to avoid that I added a simple low pass filter on the yaw value.
- It also appears that there is an offset on the boat's position. However the controller seemed to has correctly guide the boat on a line. One of the most likely causes would be that there was errors during the logging of the boat's position. It would also explained why the the boat's path is located on the bank of the canal.

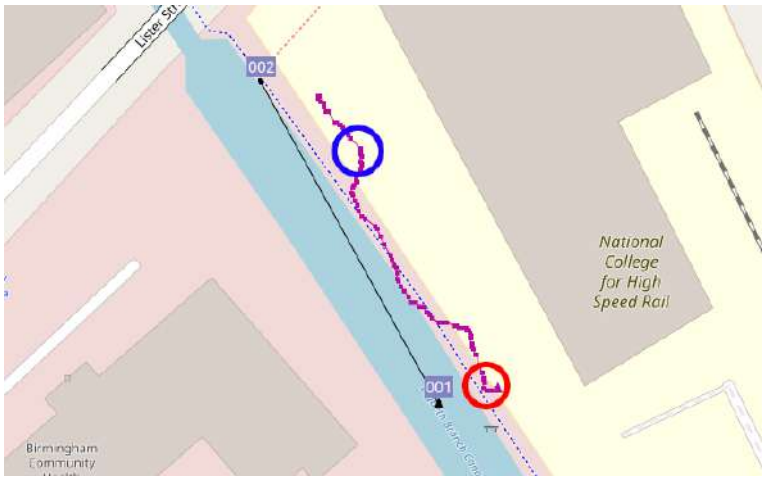

Figure 16: Log of the first test on the canal

5.2 Lake testings

We have find a better place to test the sailboat. This is a small lake used for remote controlled sailing boat race so it is perfect for our experimentations.

But unfortunately the firsts tests on this lake were did not work as planned. Indeed I had a lot of problem with the reception of the GNSS signal. On the Figure 17 we can see that the boat keeps loosing the GNSS signal and end up crashing on the bank of the lake. Every testings this day ended up the same way for the same reason.

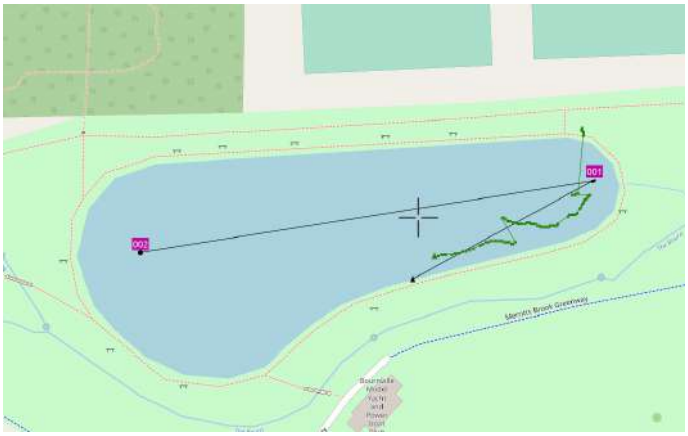

Figure 17: Log of the mission with failed GNSS reception

The next time we went on that lake, I decided to run the GNSS receiver antenna through the waterproof cap of the boat to enhance the reception of the signal. It was much better and the boat succeeded to cross the entire lake (Figure 18. Even if there has been one loss of the GNSS receiver. This test has been made with a cross wind. Another testing has been made to cross the lake

going against the wind on Figure 19. We can see on the log that the boat is trying to tack to get to the waypoint 002. Here is what I learned during those tests on the lake:

- As long as the boat is not trying to go upwind the line following algorithm is working pretty well. But the algorithm is not perfectly adjust to go upwind.
- Because the boat is very small, it is very sensitive to the wind. We can one more time observe on the Figure 19 that the boat has made some strange circle due to many wind gust that has disrupt the data acquisition (heading and wind direction).

Figure 18: Crossing of the lake (cross wind)

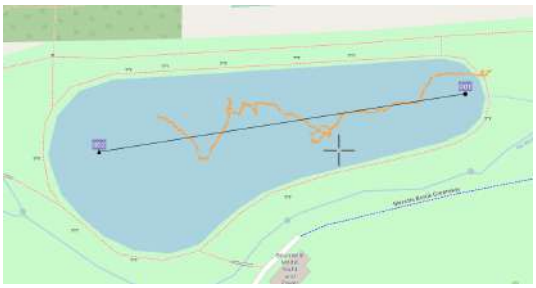

Figure 19: Crossing of the lake (upwind)

5.3 Improvements

Once on site it was not really possible to analyse the log and then understand what was working or not to test something else so I had to wait to be back at the university to analyse them. Moreover I haven't been able to run so many because the lake was quite far from the university where I was working. Anyway I had some good results but there is still room for improvement:

- The first thing that could be improved is the controller algorithm. It is difficult to make a model for every sailing boat. In the simulation the boat

was behaving perfectly but on the real boat some constant in the algorithm could have been changed appropriately. But it would have required a lot of test.

- It could have been interesting to implement another controller for doing another mission. For example a station keeping controller which goal would have been to keep the boat inside a circle around a target.
- The GNSS receiver caused a lot of mission to failed because sometimes the boat did not get any signal for several minutes. Maybe using a better antenna would have solved this problem.
- Every time I had to upload a new code, I had to take everything out of the boat and then put it back. There is a lot components and cables and not so many room inside the boat, so it was taking quite some time to do it correctly without unplugging some components accidentally. This is why it could have been great to tidy the inside using shorter cables and maybe soldering some components to avoid unexpected disconnection.

6 Conclusion

The main purpose of this internship was to fit a set of sensors on an Arduino to have an autonomous sailing boat. I was very excited about of this internship, mainly because the project was interesting, but also to have an experience abroad. It turned out to be a very rich experience, both intellectually and on a human level.

I has been able to finish the sensors integration on the boat. It includes a wind vane, an IMU and many other sensors to be have all the necessary information to control it. Then after finishing the controller algorithm, I have been able to run some experimentation on a lake.

Even though the experimentation were not all successful, I learned a lot from familiarising with the sensors, to analysing the test including programming a control algorithm. Moreover it gave me a better understanding of the world of research. Indeed, this internship gave me the opportunity to co-write a scientific document about this project [8].

References

[1] Wikipedia. Aston Univerity, 2023.

[2] Arduino Mega datasheet, 2023.

[3] V. Pelle. Github repository Sailboat code, 2023.

[4] Davis Instruments anemometer instructions, 2023.

[5] Calculating the True Wind and Why it Matters | Cruising Compass.

[6] Fabrice Le Bars Luc Jaulin. A simple controller for line following of sailboats. 2012.

[7] V. Pelle. Github repository for Sailboat-monitoring, 2023.

[8] Morge Augustin, Pelle Virgile, Wan Jian, and Jaulin Luc. Experimental Studies of Autonomous Sailing With a Radio Controlled Sailboat. *IEEE Access*, 10, 2022.

A Wiring Diagram

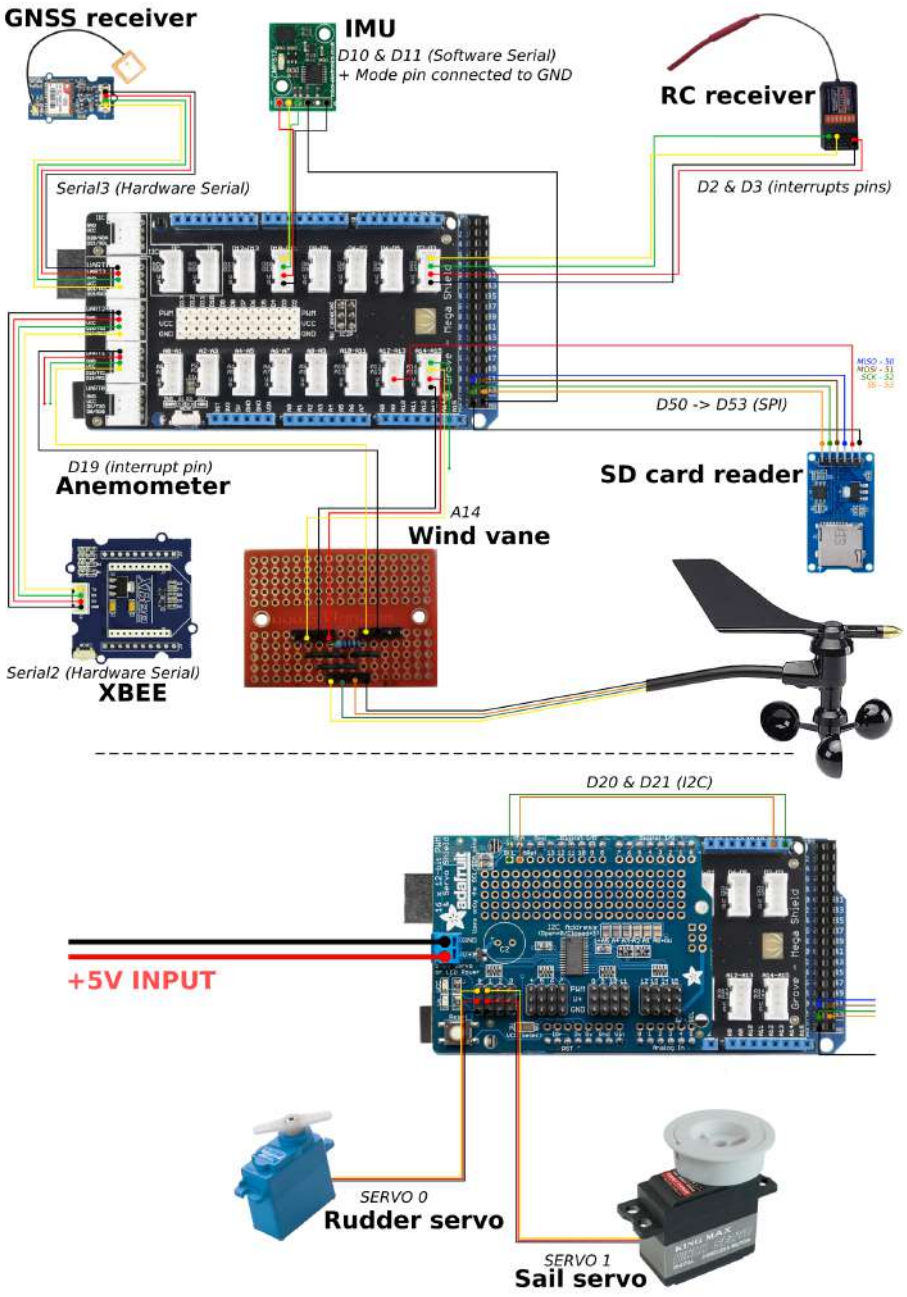

Figure 20: Wiring Diagram

B Class diagram

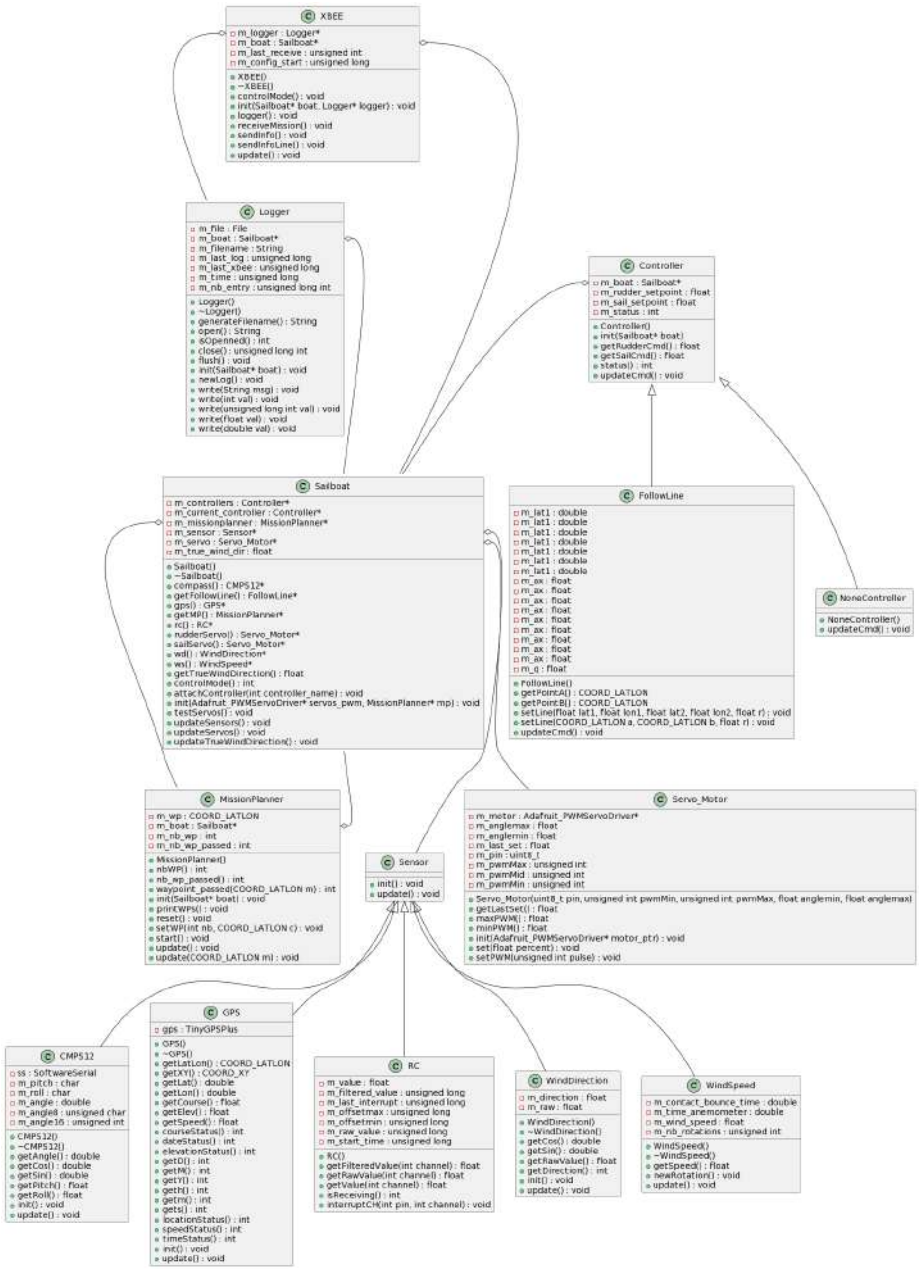

Figure 21: Class diagram

RAPPORT D'EVALUATION
ASSESSMENT REPORT

Merci de retourner ce rapport par courrier ou par voie électronique en fin du stage à :
At the end of the internship, please return this report via mail or email to:

ENSTA Bretagne – Bureau des stages - 2 rue François Verny - 29806 BREST cedex 9 – FRANCE
☎ 00.33 (0) 2.98.34.87.70 / stages@ensta-bretagne.fr

I - ORGANISME / *HOST ORGANISATION*

NOM / *Name* Aston University

Adresse / *Address* Aston Street, Birmingham B4 7ET

Tél / *Phone (including country and area code)* +44 07475 064087

Nom du superviseur / *Name of internship supervisor* Jian Wan

Fonction / *Function* Lecturer in Mechatronics and Robotics

Adresse e-mail / *E-mail address* j.wan3@aston.ac.uk

Nom du stagiaire accueilli / *Name of intern* Virgile Pelle

II - EVALUATION / *ASSESSMENT*

Veuillez attribuer une note, en encerclant la lettre appropriée, pour chacune des caractéristiques suivantes. Cette note devra se situer entre **A (très bien)** et **F (très faible)**
*Please attribute a mark from **A (excellent)** to **F (very weak).***

MISSION / *TASK*

- La mission de départ a-t-elle été remplie ? — ✓A B C D E F
  Was the initial contract carried out to your satisfaction?
- Manquait-il au stagiaire des connaissances ? — ☐ oui/*yes* ☑ non/*no*
  Was the intern lacking skills?

  Si oui, lesquelles ? / *If so, which skills?* _____

ESPRIT D'EQUIPE / *TEAM SPIRIT*

- Le stagiaire s'est-il bien intégré dans l'organisme d'accueil (disponible, sérieux, s'est adapté au travail en groupe) / *Did the intern easily integrate the host organisation? (flexible, conscientious, adapted to team work)*

  ✓A B C D E F

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here*_____

COMPORTEMENT AU TRAVAIL / *BEHAVIOUR TOWARDS WORK*

Le comportement du stagiaire était-il conforme à vos attentes (Ponctuel, ordonné, respectueux, soucieux de participer et d'acquérir de nouvelles connaissances) ?
Did the intern live up to expectations? (*Punctual, methodical, responsive to management instructions, attentive to quality, concerned with acquiring new skills)?*

A B C D E F

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* _____

_____

INITIATIVE – AUTONOMIE / *INITIATIVE – AUTONOMY*

Le stagiaire s'est –il rapidement adapté à de nouvelles situations ? A B C D E F
(Proposition de solutions aux problèmes rencontrés, autonomie dans le travail, etc.)

Did the intern adapt well to new situations? A B C D E F
(eg. suggested solutions to problems encountered, demonstrated autonomy in his/her job, etc.)

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* _____

_____

CULTUREL – COMMUNICATION / *CULTURAL – COMMUNICATION*

Le stagiaire était-il ouvert, d'une manière générale, à la communication ? A B C D E F
Was the intern open to listening and expressing himself /herself?

Souhaitez-vous nous faire part d'observations ou suggestions ? / *If you wish to comment or make a suggestion, please do so here* _____

_____

OPINION GLOBALE / *OVERALL ASSESSMENT*

❖ La valeur technique du stagiaire était : A B C D E F
Please evaluate the technical skills of the intern:

III - PARTENARIAT FUTUR / *FUTURE PARTNERSHIP*

❖ Etes-vous prêt à accueillir un autre stagiaire l'an prochain ?

Would you be willing to host another intern next year? ☑ oui/*yes* ☐ non/*no*

Fait à _____, le _____
In Birmingham, on 26/08/2022

Signature Entreprise _____ Signature stagiaire
Company stamp _____ *Intern's signature*

Merci pour votre coopération
We thank you very much for your cooperation